



SYSTEM ANALYSIS AND DESIGN :
A MUSIC EXAMINATION PROTOTYPE SYSTEM FOR TRINITY COLLEGE
LONDON

by
Mr. Nithi Juabsamai

A Final Report of the Three-Credit Course
CE 6998 Project

Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Science
in Computer and Engineering Management
Assumption University

July 2002

**SYSTEM ANALYSIS AND DESIGN:
A MUSIC EXAMINATION PROTOTYPE SYSTEM FOR TRINITY COLLEGE
LONDON**

by
Mr. Nithi Juabsamai



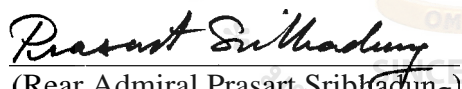
Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Science
in Computer and Engineering Management
Assumption University

July 2002

Project Title	System Analysis and Design: A Music Examination Prototype System for Trinity College London
Name	Mr. Nithi Juabsamai
Project Advisor	Rear Admiral Prasart Sribhadung
Academic Year	July 2002


The Graduate School of Assumption University has approved this final report of the three-credit course, CE 6998 PROJECT, submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer and Engineering Management.

Approval Committee:


(Rear Admiral Prasart Sribhadung)
Advisor


(Prof. Dr. Srisakdi Chanuonman)
Chairman

(Dr. Chamnong n hirapanich)
Dean and Co-advisor


(Assoc. Prof. Somchai Thayarnyong)
MUA Representative

July 2002

ABSTRACT

The objectives of this project are to design, develop, implement, and test the prototype system to support the "Candidate Data Entry" system (CDE) used by Progress Center, Thailand representative examination center of Trinity College London (TCL), in order to enhance the capability of CDE system and fulfill the management and customer requirements.

CDE system developed by TCL cannot serve all Progress Center requirements. Many tasks have to be done manually before candidates' information are able to be input to CDE system. The examination schedule function of CDE system cannot support the change, cancel, and swap requirements so this job is also done manually. More functions and reports are needed by management and customers so the supporting system is required. The supporting system, named CDE Kid, is designed and developed based on "Software Prototyping" concept. Prototype system is a parallel testing with the manual system and the test results are recorded for system evaluation.

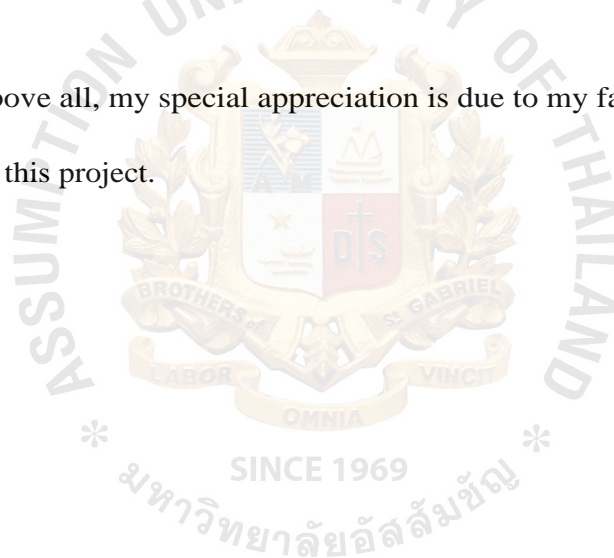
The supporting system, CDE Kid, can be further modified based on Web Application and Customer Relation Management (CRM) concept. So, candidates can register for the examination, update their information, and track the registration and examination results via the Internet.

ACKNOWLEDGEMENT

This project is successfully done with the great help and cooperation through many people. I would like to express my deepest appreciation to Rear Admiral Prasart Sribhadung whose advice is very practical for studying the data and language use in developing this project.

I am forever grateful to Mrs. Jintana Sivayathon, Managing Director and staffs at Progress Center for their time helpful and information services in supporting, developing and testing the Music Examination Prototype System for Trinity College London.

Above all, my special appreciation is due to my family for their encouragement to finalize this project.



Nithi Juabsamai

TABLE OF CONTENTS

<u>Chapter</u>	<u>Page</u>
ABSTRACT	
ACKNOWLEDGEMENTS	ii
LIST OF FIGURES	vi
LIST OF TABLES	viii
I. INTRODUCTION	1
1.1. Organization Profiles	1
1.2. Objectives	2
1.3. Scope	2
1.4. Deliverables	3
II. LITERATURE REVIEW	4
2.1. Software Development Process	4
2.2. Analysis Concept and Principle	10
2.3. Analysis Modeling	16
2.4. Software Design	23
2.5. Interface Design	25
2.6. Software Testing	27
III. CURRENT SYSTEM ANALYSIS	29
3.1. Level 0 Data Flow Diagram, Context Model	29
3.2. Level 1 Data Flow Diagram	30
3.3. Problem And Requirement	32
3.4. Database Analysis	33
3.5. Summary	38

<u>Chapter</u>	<u>Page</u>
IV. NEW SYSTEM DESIGN	39
4.1. Level 0 Data Flow Diagram, Context Model	39
4.2. Level 1 Data Flow Diagram	39
4.3. Level 2 Data Flow Diagram	42
4.4. Entity Relationship Diagram	46
4.5. Process Specification	47
4.6. Summary	49
V. PROTOTYPING CANDIDATE DATA ENTRY SUPPORT SYSTEM (CDE Kid)	50
5.1. Database	51
5.2. Input/Output Interface	55
5.3. Coding	57
5.4. Summary	57
VI. TESTING AND EVALUATION	58
6.1. Prototype Testing	58
6.2. Parallel Testing	59
6.3. Evaluation	61
VII. CONCLUSIONS AND RECOMMENDATIONS	65
7.1. Conclusions	65
7.2. Recommendations	66
APPENDIX A INDEX	68
APPENDIX B 2 ND DFD OF THE EXISTING SYSTEM	70
APPENDIX C DATA DICTIONARY FOR LINKED TABLE	75
APPENDIX D INPUT/OUTPUT INTERFACE	88

<u>Chapter</u>		<u>Page</u>
APPENDIX E	TESTING INPUT AND OUTPUT	97
BIBLIOGRAPHY		102



LIST OF FIGURES

Figure	Page
2.1 The Linear Sequential Model	4
2.2 The Prototyping Paradigm	8
2.3 The Structure of the Analysis Model	17
2.4 Information Flow Model	20
2.5 Basic DFD Notation	21
2.6 Information Flow Refinement	22
2.7 Translating the Analysis Model into a Software Design	24
3.1 Context Model of Progress Center Examination System	29
3.2 the level 1 DFDs of Progress Center Examination System	31
3.3 Flow of Data between Two Databases	34
3.4 List of Table in Data2000.mdb	35
3.5 CandidateRegistration Table	36
3.6 GradeExam Table	37
4.1 Context Model of CDE Kid System	40
4.2 Level1 DFDs of CDE Kid System	41
4.3 Level2 DFDs of Process 2	43
4.4 Level 2 DFDs of Process 7	44
4.5 Level 2 DFDs of Process 8	45
4.6 Entity Relationship Diagram	46
5.1 Application Form Input Screen	56
5.2 Print Application by Teacher Screen	57
B.1 2 nd DFD of Process 2	71

<u>Figure</u>	<u>Page</u>
B.2 ^{2nd} DFD of Process 7	72
B.3 2" DFD of process 10	73
C.1 Database Structure of Linked Table	87
D.1 Application Screen	89
D.2 Candidate Form	89
D.3 Application Form	90
D.4 Reschedule Form (Timetable Management)	90
D.5 Exam Score Form (Exam Result Entry Form)	91
D.6 Summary on Teacher Inquiry Form	92
D.7 Summary on Requested Date Inquiry Form	93
D.8 Examination Schedule	94
D.9 Examination Summary for Teacher	94
D.10 Examination Ranking Report	95
D.11 Examination Result Appointment Slip	95
D.12 Certificate Appointment Slip	96

LIST OF TABLES

Table	Page
5.1 Candidate Table	51
5.2 Application Table	52
5.3 TimeTable Table	53
5.4 ExamResult Table	54
5.6 Option Table	55
6.1 Time Efforts for Manual System	59
6.2 Time Efforts for CDE Kid	60
6.3 Cash Flow for Manual System	62
6.4 Cash Flow for CDE Kid	63
6.5 Cash Flow of the Replacement Analysis	64
C.1 CandidateRegistration Table	76
C.2 GradeExam Table	77
C.3 Grade Subject Table	80
C.4 School Table	83
C.5 Teacher Table	85
E.1 Testing Candidate Information Table	98
E.2 Changing Requested Date Request Table	99
E.3 Examination Result Table	100

1. INTRODUCTION

1. Organization Profile.

In 1877 Trinity College of Music was found to offer external practical examinations in Music. Since then Trinity has expanded and now assesses Drama & Speech, English for Speakers of Other Languages (ESOL), and degree level programs in professional dance, acting and stage management. By 1992, the External Examinations Department had grown so large that it became a separate organization in its own right known as Trinity College London.

As a leading international examinations board in the performing and communicative arts, Trinity College London (TCL) provides clients around the world, at times and places convenient to them, with appropriate, reliable and recognized personal assessments conducted by qualified examiners. Trinity is a non-profit-making organization which uses surplus income to support the development and appreciation of skills in music, in drama and speech, and in English language learning and teaching.

The first syllabus for both written and practical examinations allowed students to enter for either Junior or Senior grades. In 1877, over 1,100 students took part in the first theory examinations with the first practical examinations held in the following year at Stroud in Gloucestershire. By 1881, Trinity had extended its examining activities to cover India, South Africa, Ceylon, New Zealand and Australia. By the time of the fifty-year Jubilee, over 50,000 students were sitting for Trinity music examinations annually.

In Thailand Progress Center has been the representative examination center of Trinity College London since 1999. Progress Center provides the both practical and written music examinations in many instruments such as piano, violin, flute, singing,

etc. There are two examination sessions a year for around 2,500 candidates who come from Bangkok and upcountry to take an examination.

Trinity developed the examination system and distributed the client application named CDE2000 (Candidate Data Entry 2000) to all examination centers around the world. This application can register the candidate, export data to files for sending to head office in London, arrange the timetable, and other useful functions. Most of these functions can use very well but the registration and timetable arrangement functions are the problems for examination center in Thailand because of the culture of teachers and students in Thailand. Progress Center must do the above functions by manual to solve the problems and it takes a lot of time and efforts. So, they need another system to fulfill the requirement and handle these problems.

1.2. Objectives

The objectives of this project are to develop a useable and appropriate prototype system to replace the manual jobs and to ensure that the new system is able to work together with CDE2000 system.

1.3. Scope

- Identify the problem.
- Establish the Progress Center requirement.
- Analyze the existing CDE2000 system with Data Flow Diagram.
- Design the new system
- Develop the prototype program
- Test and evaluate the prototype
- Recommend for further development

1.4. Deliverables

Final Project Report

Prototype application



II. LITERATURE REVIEW

2.1. Software Development Process

2.1.1 The Linear Sequencing Model

Figure 2.1 illustrates the linear sequential model for software engineering. Sometimes called the "classic life cycle" or the "waterfall model," the linear sequential model suggests a systematic, sequential approach to software development that begins at the system level and progresses through analysis, design, coding, testing, and maintenance.

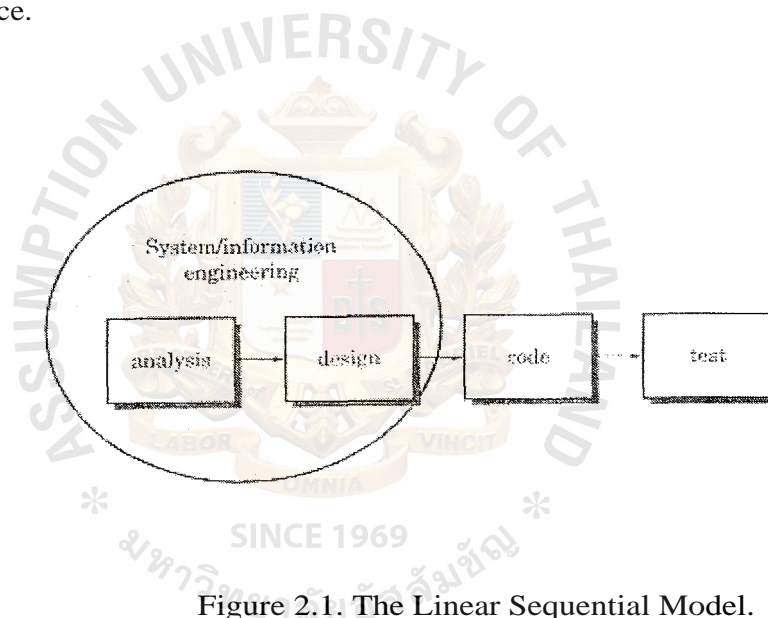


Figure 2.1. The Linear Sequential Model.

Modeled after the conventional engineering cycle, the linear sequential model encompasses the following activities:

- (1) System/information engineering and modeling.

Because software is always part of a larger system (or business), work begins by establishing requirements for all system elements and then allocating some subset of these requirements to software. This system view is essential when software must interface with other elements such as

hardware, people, and database. System engineering and analysis encompasses requirements gathering at the system level with a small amount of top-level analysis and design. Information engineering encompasses requirements gathering at the strategic business level and at the business area level

(2) Software requirements analysis.

The requirements gathering process is intensified and focused specifically on software. To understand the nature of the program(s) to be built, the software engineer ("analyst") must understand the information domain for the software, as well as required function, behavior, performance, and interfacing. Requirements for both the system and the software are documented and reviewed with the customer.

(3) Design.

Software design is actually a multi-step process that focuses on four distinct attributes of a program: data structure, software architecture, interface representations, and procedural (algorithmic) detail. The design process translates requirements into a representation of the software that can be assessed for quality before code generation begins. Like requirements, the design is documented and becomes part of the software configuration.

(4) Code generation.

The design must be translated into a machine readable form. The code generation step performs this task. If design is performed in a detailed manner, code generation can be accomplished mechanistically.

(5) Testing.

Once code has been generated, program testing begins. The testing process focuses on the logical internals of the software, assuring that all statements have been tested, and on the functional externals-that is, conducting tests to uncover errors and ensure that defined input will produce actual results that agree with required results.

(6) Maintenance.

Software will undoubtedly undergo change after it is delivered to the customer (a possible exception is embedded software). Change will occur because errors have been encountered, because the software must be adapted to accommodate changes in its external environment or because the customer requires functional or performance enhancements. Software maintenance reapplies each of the preceding phases to an existing program rather than a new one.

The linear sequential model is the oldest and the most widely used paradigm for software engineering. However, criticism of the paradigm has caused even active supporters to question its efficacy. Among the problems that are sometimes encountered when the linear sequential model is applied are:

- (1) Real projects rarely follow the sequential flow that the model proposes. Although the linear model can accommodate iteration, it does so indirectly. As a result, changes can cause confusion as the project team proceeds.
- (2) It is often difficult for the customer to state all requirements explicitly. The linear sequential model requires this and has difficulty accommodating the natural uncertainty that exists at the beginning of many projects.

- (3) The customer must have patience. A working version of the program(s) will not be available until late in the project time-span. A major blunder, if undetected until the working program is reviewed, can be disastrous.
- (4) Developers are often delayed unnecessarily. In an interesting analysis of actual projects, Bradac year found that the linear nature of the classic life cycle leads to "blocking states" in which some project team members must wait for other members of the team to complete dependent tasks. In fact, the time spent waiting can exceed the time spent on productive work! The blocking states tend to be more prevalent at the beginning and end of a linear sequential process.

Each of these problems is real. However, the classic life cycle paradigm has a definite and important place in software engineering work. It provides a template into which methods for analysis, design, coding, testing, and maintenance can be placed. The classic life cycle remains the most widely used process model for software engineering. While it does have weaknesses, it is significantly better than a haphazard approach to software development.

2.1.2. The Prototyping Model

Often, a customer defines a set of general objectives for software but does not identify detailed input, processing, or output requirements. In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human-machine interact should take. In these and many other situations, a prototyping paradigm may offer the best approach.

The prototyping paradigm (Figure 2.2) begins with requirements gathering. Developer and customer meet and define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is

mandatory. A "quick design" then occurs. The quick design focuses on a representation of those aspects of the software that will be visible to the customer/user (e.g., input approaches and output formats). The quick design leads to the construction of a prototype. The prototype is evaluated by the customer/user and is used to refine requirements for the software to be developed. Iteration occurs as the prototype is tuned to satisfy the needs of the customer, at the same time enabling the developer to better understand what needs to be done.

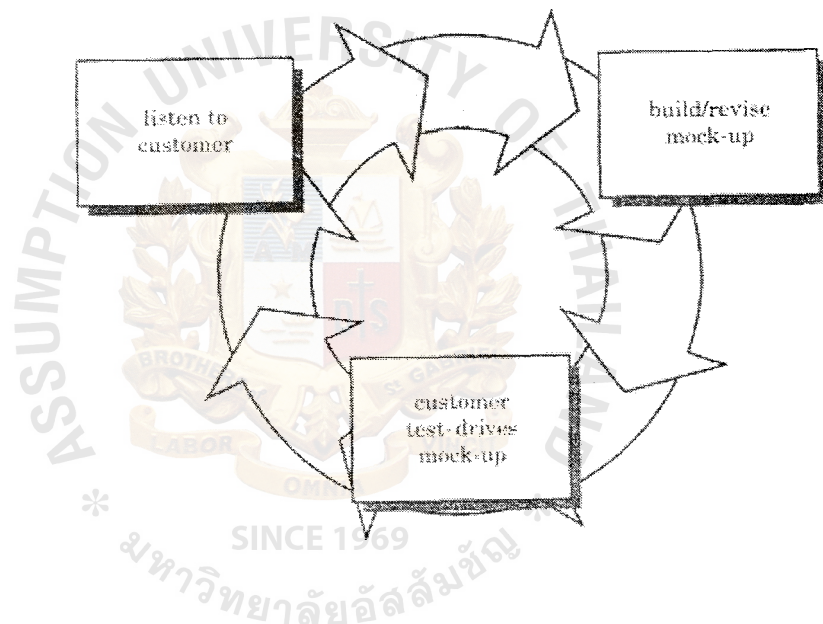


Figure 2.2. The Prototyping Paradigm.

Ideally, the prototype serves as a mechanism for identifying software requirements. If a working prototype is built, the developer attempts to make use of existing program fragments or applies tools (e.g., report generators, window managers, etc.) that enable working programs to be generated quickly.

In most projects, the first system built is barely usable. It may be too slow, too big, awkward in use or all there. There is no alternative but to start again, smarter but smarter, and build a redesigned version in which these problems are solved... When a

new system concept or new technology is used, one has to build a system to throw away, for even the best planning is not so omniscient as to get it right the first time. The management question, therefore, is not whether to build a pilot system and throw it away. You will do that. The only question is whether to plan in advance to build a throwaway, or to promise to deliver the throwaway to customers....

The prototype can serve as "the first system." The one that Brooks year recommends we throw away. But this may be an idealized view. It is true that both customers and developers like the prototyping paradigm. Users get a feel for the actual system and developers get to build something immediately. Yet, prototyping can also be problematic for the following reasons:

- (1) The customer sees what appears to be a working version of the software, unaware that prototype is held together "with chewing gum and baling wire," unaware that in the rush to get it working we haven't considered overall software quality or long-term maintainability. When informed that the product must be rebuilt so that high levels of quality can be maintained, the customer cries foul and demands that "a few fixes" be applied to make the prototype a working product. Too often, software development management relents.
- (2) The developer often makes implementation compromises in order to get a prototype working quickly. An inappropriate operating system or programming language may be used simply because it is available and known; an inefficient algorithm may be implemented simple to demonstrate capability. After a time, the developer may become familiar with these choices and forget all the reasons why they were inappropriate. The less-than-ideal choice has now become an integral part of the system.

Although problems can occur, prototyping can be an effective paradigm for software engineering. The key is to define the rules of the game at the beginning; that is, the customer and developer must both agree that the prototype is built to serve as a mechanism for defining requirements. It is then discarded (at least in part), and the actual software is engineered with an eye toward quality and maintainability.

2.2. Analysis Concepts and Principle.

2.2.1. Requirement Analysis

Requirement analysis enable the system engineer to specify software function and performance, indicates software's interface with other system elements, and establishes constraints that software must meet. Requirements analysis allows the software engineer (often call *analyst* in this role) to refine the software allocation and build models of the data, functional, and behavioral domains that will be treated by software. Requirements analysis provides the software designer with models that can be translated into data, architectural interface, and procedural design. Finally, the requirement specification provides the developer and the customer with the means to assess quality once software is built.

Software requirements analysis may be divided into five areas of effort: (1) problem recognition; (2) evaluation and synthesis; (3) modeling; (4) specification; and (5) review.

Initially, the analysis studies the system specification (if one exists) and the software project plan. It is important to understand software in a system context and to review the software scope that was used to generate planning estimates. Next communication for analysis must be established so that problem recognition is ensured. The goal of the analyst is recognition of the basic problem elements as perceived by the user/customer.

Problem evaluation and solution synthesis is the next major area of effort for analysis. The analyst must define all externally observable data objects, evaluate the flow and content of information; define and elaborate all software functions; understand software behavior in the context of events that affect the system; establish system interface characteristics; and uncover additional design constraints. Each of these tasks serves to describe the problem so that an overall approach or solution may be synthesized.

For example, an inventory control system is required for a major supplier of auto parts. The analyst finds that problems with the current manual system include (1) inability to obtain the status of a component rapidly; (2) two or three day turnaround to update a card file; (3) multiple reorders to the same vendor because there is no way to associate vendors with components, and so on. Once problems have been identified, the analyst determines what information is to be produced by the new system and what data will be provided to the system. For instance, the customer desires a daily report that indicates what parts have been taken from inventory clerks will log the identification number of each part as it leaves the inventory area.

Upon evaluating current problems and desired information (input and output), the analyst begins to synthesize one or more solutions. To begin, the data, processing functions, and behavior of the system are defined in detail. Once this information has been established, basic architectures for implementation are considered. A client/server approach would seem to be appropriate, but does it fall within the scope outlined in the software plan? A database management system would seem to be required, but is the user/customer's need for associativity justified? The process of evaluation and synthesis continues until both analyst and customer feel confident that software can be adequately specified for subsequent development steps.

Throughout evaluation and solution synthesis, the analyst's primary focus is on "what," not "how." What data does the system produce and consume, what functions must the system perform, what interfaces are defined, and what constraints apply?

During the evaluation and solution synthesis activity, the analyst creates model of the system in an effort to better understand data and control flow, functional processing and behavioral operation, and information content. The model serves as a foundation for software design and as the basis for the creation of a specification for the software.

It is important to note that detailed specification may not be possible at this stage. The customer maybe unsure of precisely what is required. The developer maybe be unsure that a specific approach will properly accomplish function and performance. For these and many other reasons, an alternative approach to requirements analysis, called prototyping, may be conducted. We discuss prototyping later in this chapter.

2.2.2. Software Prototyping

Analysis should be conducted regardless of the software engineering paradigm that is applied. However, the form that analysis takes will vary. In some cases it is possible to apply operational analysis principles and derive a model of software from which a design can be developed. In other situations, requirements gathering is conducted, the analysis principles are applied, and a model of the software to be built, called a prototype, is constructed for customer and developer assessment. Finally, there are circumstances that require the construction of a prototype at the beginning of analysis, since the model is the only means through which requirements can be effectively derived. The model then evolves into production software.

Boar year justifies the prototyping technique in this way:

Most currently recommended methods for defining business system requirements are designed to establish a final, complete, consistent, and correct set of requirements

before the system is designed, constructed, seen or experienced by the user. Common and recurring industry experience indicates that despite the use of rigorous techniques, in many cases users still reject applications as neither correct nor complete upon completion. Consequently, expensive, time consuming, and divisive rework is required to harmonize the original specification with the definitive test of actual operational needs. In the worst case, rather than retrofit the delivered system, it is abandoned. Developers may build and test against specifications but users accept or reject against current and actual operational realities.

Although the above quote represents an extreme view, its fundamental argument is sound. In many (but not all) cases, the construction of a prototype, possibly coupled with systematic analysis methods, is an effective approach to software engineering.

(1) Selecting the Prototyping Approach

The prototyping paradigm can be either closed-ended or open-ended. The closed-ended approach is often called throwaway prototyping. Using this approach, a prototype serves solely as a rough demonstration of requirements. It is then discarded, and the software is engineered using a different paradigm. An open-ended approach, called evolutionary prototyping, uses the prototype as the first part of an analysis activity that will be continued into design and construction. The prototype of the software is the first evolution of the finished system.

Before a closed-ended or open-ended approach can be chosen, it is necessary to determine whether the system to be built is amenable to prototyping. A number of prototyping candidacy factors[BOA84] can be defined: application area, application complexity, customer characteristics, and project characteristics.

In general, any application that creates dynamic visual displays, interacts heavily with a human user, or demands algorithms or combinatorial processing that must be developed in an evolutionary fashion as a candidate for prototyping. However, these application areas must be weighed against application complexity. If a candidate application (one that has the characteristics noted above) will require the development of tens of thousands of lines of code before any demonstrable function can be performed, it is likely to be too complex for prototyping. If the complexity can be partitioned, however, it may still be possible to prototype portions of the software.

Because the customer must interact with the prototype in later steps, it is essential that (1) customer resources be committed to the evaluation and refinement of the prototype, and (2) the customer be capable of making requirements decisions in a timely fashion. Finally, the nature of the development project will have a strong bearing on the efficacy of prototyping. Is project management willing and able to work with the prototyping method? Are prototyping tools available? Do developers have experience with prototyping methods?

(2) Prototyping Methods and Tools

For software prototyping to be effective, a prototype must be developed rapidly so that the customer may assess results and recommend changes. To conduct rapid prototyping, three generic classes of methods and tools are available:

- (a) Fourth Generation Techniques encompass a broad array of database query and reporting languages, program and

application generators and other very high level nonprocedural languages. Because 4GT enables the software engineer to generate executable code quickly, they are ideal for rapid prototyping.

- (b) **Reusable Software Components.** Another approach to rapid prototyping is to assemble, rather than build, the prototype by using a set of existing software components. A software component may be a data structure (or database) or a software architectural component (i.e., a program) or a procedural component (i.e., a module). In each case the software component must be designed in a manner that enables it to reuse without detailed knowledge of its internal working. Melding prototyping and program component reuse will work only if a library system is developed so that components that do exist can be catalogued and then retrieved. Although a number of tools have been developed to meet this need, much work remains to be done in this area. It should be noted that an existing software product can be used as a prototype for a "new, improved" competitive product. In a way, this is a form of reusability for software prototyping.

- (c) **Formal Specification and Prototyping Environments.** Over the past two decades, a number of formal specification languages and tools have been developed as a replacement for natural language specification techniques. Today, developers of these formal languages are in the process of developing interactive

environments that (1) enable an analyst to interactively create a language based specification of a system or software (2) invoke automated tools that translate the language-based specification into executable code, and (3) enable the customer to use the prototype executable code to refine formal requirements.

2.3. Analysis Modeling

².3.1.A Brief History

Like many important contributions to software engineering, structured analysis was not introduced with a single landmark paper or book that was a definitive treatment of the subject. Early work in analysis modeling begun in the late 1960s and early 1970s, but the first appearance of the structured analysis approach was as an adjunct to another important topic-structured design. Researchers needed a graphical notation for representing data and the processes that transformed it. These processes would ultimately be mapped into a design architecture.

The term "structured analysis," originally coined by Douglas Ross, was popularized by DeMarco. In his book on the subject, DeMarco introduced and named the key graphical symbols that enabled an analyst to create information flow models, suggested heuristics for the use of these symbols, suggested that a data dictionary and processing narratives could be used as a supplement to the information flow models, and presented numerous examples that illustrated the use of this new method. In the years that followed, variations of the structured analysis approach were suggested by Page-Jones, Gane and Sarson, and many others. In every instance, the method focused on information systems applications and did not provide an adequate notation to address the control and behavioral aspects of real-time engineering problems.

By the mid-1980s, the deficiencies of structured analysis (when attempts were made to apply the method to control-oriented applications) became painfully apparent. Real-time "extensions" were introduced by Ward and Mellor and later by Hatley and Pirbhai. These extensions resulted in a more robust analysis method that could be applied effectively to engineering problems. Attempts to develop one consistent notation have been suggested, and modernized treatments have been published to accommodate the use of CASE tools.

2.3.2. The Elements Of The Analysis Model

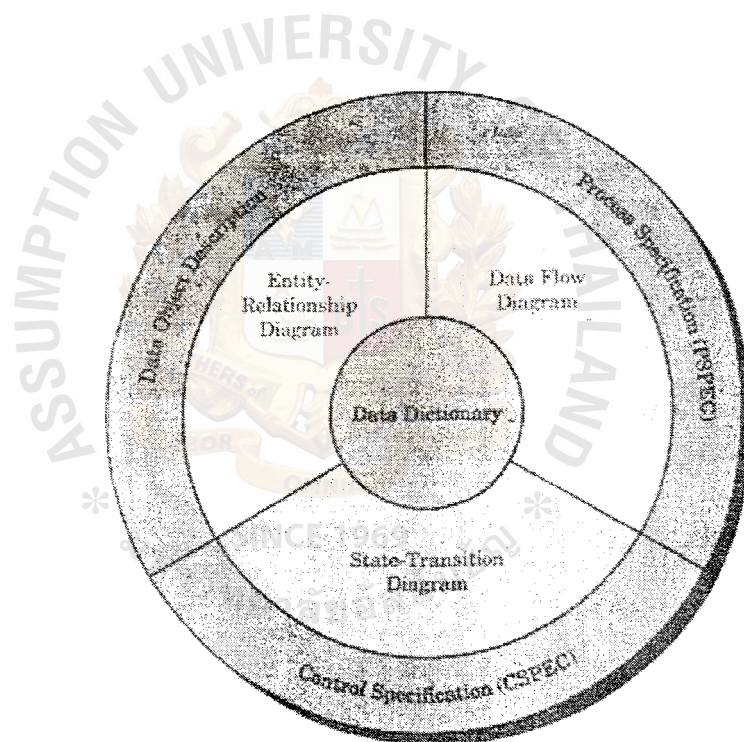


Figure 2.3. The Structure of the Analysis Model.

The analysis model must achieve three primary objectives: (1) to describe what the customer requires, (2) to establish a basis for the creation of a software design, and (3) to define a set of requirements that can be validated once the software is built. To

accomplish these objectives, the analysis model derived during structured analysis takes the form illustrated in Figure 2.3.

At the core of the model lies the data dictionary-a repository that contains descriptions of all data objects consumed or produced by the software. Three different diagrams surround the core. The entity-relationship diagram (ERD) depicts relationships between data objects. The ERD is the notation that is used to conduct the data modeling activity. The attributes of each data object noted in the ERD can be described using a data object description.

The data flow diagram (DFD) serves two purposes: (1) to provide an indication of how data are transformed as they move through the system, and (2) to depict the functions (and subfunctions) that transform the data flow. The DFD provides additional information that is used during the analysis of the information domain and serves as a basis for the modeling of function. A description of each function presented in the DFD is contained in a process specification (PSPEC).

The state-transition diagram (STD) indicates how the system behaves as a consequence of external events. To accomplish this, the STD represents the various modes of behavior (called states) of the system and the manner in which transitions are made from state to state. The STD serves as the basis for behavioral modeling. Additional information about control aspects of the software is contained in the control specification (CSPEC).

The analysis model encompasses each of the diagrams, specifications, and descriptions, and the dictionary noted in Figure 2.3. A more detailed discussion of these elements of the analysis model is presented in the sections that follow.

2.3.3. Data Modeling

Data Modeling answers a set of specific questions that are relevant to and data processing application. What are the primary data objects to be processed by the system? What is the composition of each data object and what attributes describe the object? Where do the objects currently reside? What are the relationships between each object and other objects? What is the relationship between the objects and the processes that transform them?

To answer these questions, data modeling methods make use of the entity-relationship diagram (ERD). The ERD enables a software engineer to identify data objects and their relationships using a graphical notation. In the context of structured analysis, the ERD defines all data that are input, stored, transformed, and produced with in an application.

The entity-relationship diagram focuses solely on data (and therefore satisfies the first operational analysis principle), representing a "data network" that exists for a given system. The ERD is especially useful for applications in which data and the relationships that govern data are complex. Unlike the data flow diagram, data modeling considers data independently of the processing that transforms the data.

2.3.4 Data Flow Diagrams

As information moves through software, it is modified by a series of transformations. A data flow diagram (DFD) is a graphical technique that depicts information flow and the transforms that are applied as data move from input to output. The basic form of a data flow diagram is illustrated in Figure 2.4. The DFD is also known as a data flow graph or a bubble chart.

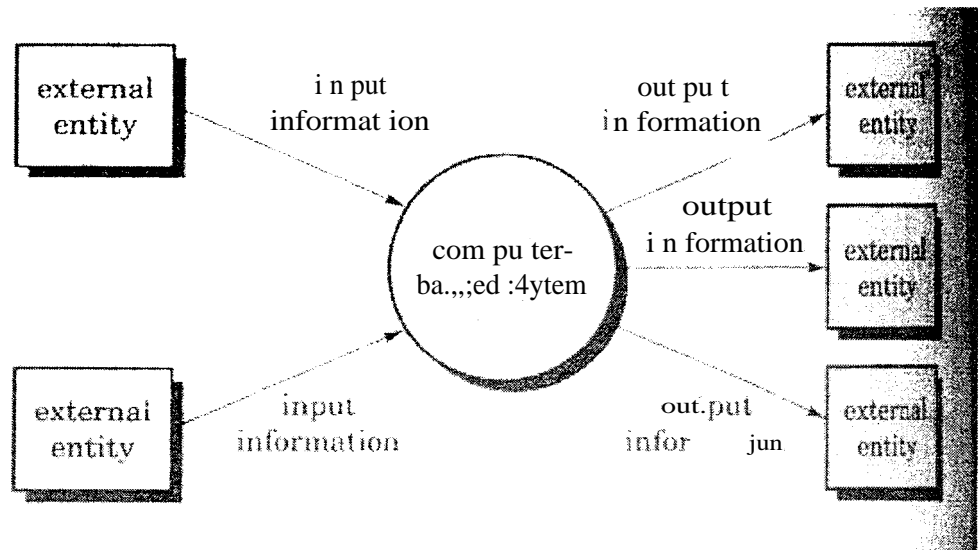
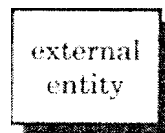


Figure 2.4. Information Flow Model.

The data flow diagram may be used to represent a system or software at any level of abstraction. In fact, DFDs may be partitioned into levels that represent increasing information flow and functional detail. Therefore, the DFD provides a mechanism for functional modeling as well as information flow modeling. In so doing, it satisfies the second operational analysis principle (i.e., creating a functional model).

A level 0 DFD, also called a fundamental system model or a context model, represents the entire software element as a single bubble with input and output data indicated by incoming and outgoing arrows, respectively. Additional processes (bubbles) and information flow paths are represented as the level 0 DFD is partitioned to reveal more detail. For example, a level 1 DFD might contain five or six bubbles with interconnecting arrows. Each of the processes represented at level 1 are subfunctions of the overall system depicted in the context model.



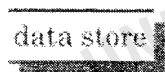
A producer or consumer of information that resides outside the bounds of the system to be modeled



A transformation of information a function that resides within the bounds of the system to be modeled



Indicates the direction of data flow



Processes may be as buffer or queue or as a relational database

Figure 2.5. Basic DFD notation.

The basic notation used to create a DFD is illustrated in Figure 2.5. A rectangle is used to represent an external entity, that is, a system element (e.g., hardware, a person, another program) or another system that produces information for transformation by the software or receives information produced by the software. A circle represents a process or transform that is applied to data and changes it in some way. An arrow represents one or more data items or data objects. All arrows on a DFD should be labeled. The double line represents a data store-stored information that is used by the software. The simplicity of DFD notation is one reason why structured analysis techniques are the most widely used.

It is important to note that no explicit indication of the sequence of processing is supplied by the diagram. Procedure or sequence may be implicit in the diagram, but explicit procedural representation is generally delayed until software design.

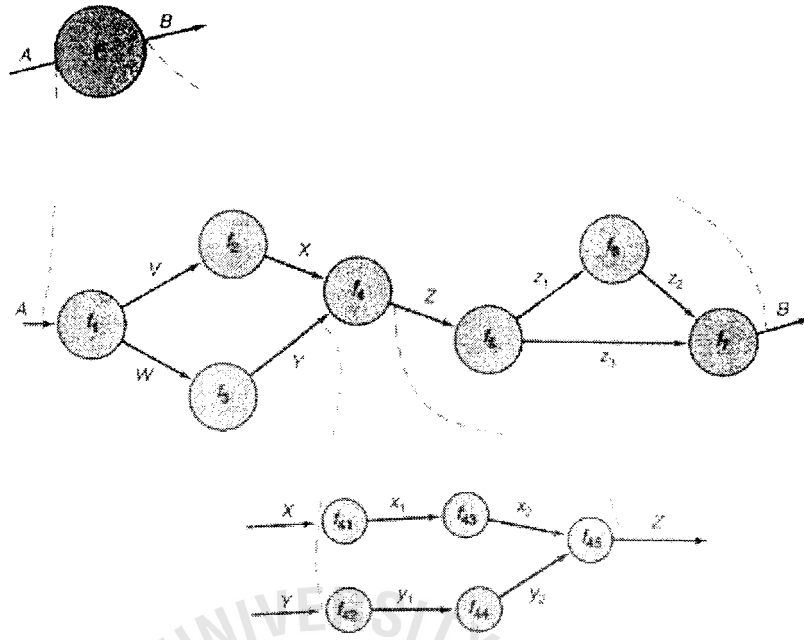


Figure 2.6. Information Flow Refinement.

As noted earlier, each of the bubbles may be refined or layered to depict more detail. Figure 2.6 illustrates this concept. A fundamental model for system F indicates the primary input is A and ultimate output is B . We refine the F model into transform f_1 to f_7 . Note that information flow continuity must be maintained, that is, input and output to each refinement must remain the same. This concept, sometimes called balancing, is essential for the development of consistent models. Further refinement of f_4 depicts detail in the form of transform f_{41} to f_{45} . Again, the input (X , Y) and output (Z) remain unchanged.

The data flow diagram is a graphical tool that can be very valuable during software requirements analysis. However, the diagram can be misinterpreted if its function is confused with the flowchart. A data flow diagram depicts information flow

without explicit representation of procedural logic (e.g., conditions or loops). It is not a flowchart with rounded edges!

The basic notation used to develop a DFD is not in itself sufficient to describe requirements for software. For example, an arrow shown in a DFD represents a data object that is input to or output from a process. A data store represents a data object that is input to or output from a process. A data store represents some organized collection of data. But what is the content of the data implied by the arrow or depicted by the store? If the arrow (or the store) represents a collection of objects, what are they? These questions are answered by applying another component of the basic notation for structured analysis-the data dictionary.

Finally, the graphical notation represented in Figure 2.5 must be augmented with descriptive text. A processing specification (PSPEC) can be used to specify the processing details implied by a bubble with in a DFD. The processing specification describes the input to a function, the algorithm that is applied and the input and the output that is produced. In addition, the PSPEC indicates restrictions and limitations imposed on the process (function), performance characteristics that are relevant to the process, and design constraints that may influence the way in which the process will be implemented.

2.4. Software Design

Software design sits at the technical kernel of the software engineering process and is applied regardless of the software process model that is used. Once software requirements have been analyzed and specified, software design is the first of three technical activities-design, code generating, and testing-that are required to build and verify the software. Each activity transform information in a manner that ultimately results in validated computer software.

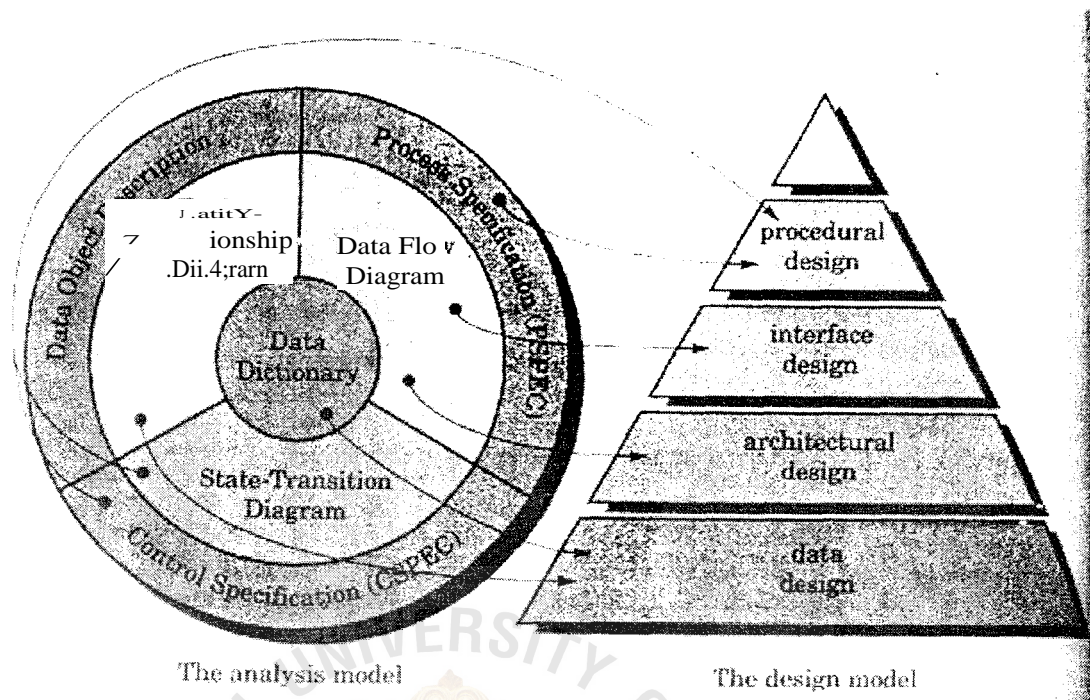


Figure 2.7. Translating the Analysis Model into a Software Design.

Each of the elements of the analysis model provides information that is required to create a design model. The flow of information during software design is illustrated in Figure 2.7. Software requirements, manifested by the data, functional, and behavioral models, feed the design step. Using one of a number of design methods, the design step produces a data design, and architectural design, an interface design, and a procedural design.

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity-relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity.

The architectural design defines the relationship among major structural elements of the programs. This design representation-the modular framework of a computer

program-can be derived from the analysis model(s) and the interaction of subsystems defined within the analysis model.

The interface design describes how the software communicates within itself, to systems that inter-operate with it, and with humans who use it. An interface implies a flow of information. Therefore, the data flow diagram provides the information required for interface design.

The procedural design transform structural elements of the program architecture into a procedural description of software components. Information obtained from the PSPEC, CSPEC, and STD serve as the basis for procedural design.

During Design we make decisions that will ultimately affect the success of software construction, and as important, the ease with which software can be maintained. But why is design so important?

The importance of software design can be stated with a single word-quality. Design is the place where quality is fostered in software development. Design provides us with representations of software that can be assessed for quality. Design is the only way that we can accurately translate a customer's requirements into a finished software product or system. Software design serves as the foundation for all software engineering and software maintenance steps that follow. Without design, we risk building an unstable system-one that will fail when small changes are made; one that may be difficult to test; one whose quality cannot be assessed until late in the software engineering process, when time is short and much money has already been spent.

2.5. Interface Design

The architectural design provides a software engineer with a picture of the program structure. Like the blueprint for a house, the overall design is not complete without a representation of doors, windows, and utility connections for water,

electricity, and telephone (not to mention cable TV). The "doors, windows, and utility connections" for computer software comprise the interface design of a system.

Interface design focuses on three areas of concern; (1) the design of interfaces between software modules; (2) the design of interfaces between the software and other nonhuman producers and consumers of information (i.e., other external entities); and (3) the design of the interface between a human (i.e., the user) and the computer.

2.5.1.1 Internal and External Interface Design

The design of internal program interfaces, sometimes called intermodular interface design, is driven by the data that must flow between modules and the characteristics of the programming language in which the software is to be implemented. In general, the analysis model contains much of the information required for intermodular interface design. The data flow diagram describes how data objects are transformed as they move through a system. The transforms of the DFD are mapped into modules within the program structure. Therefore, the arrows (data objects) flowing into and out of each DFD transform must be mapped into a design for the interface of the module that corresponds to that DFD transform.

External interface design begins with an evaluation of each external entity represented in the DFDs of the analysis model. The data and control requirements of the external entity are determined and appropriate external interfaces are designed.

Both internal and external interface design must be coupled with data validation and error handling algorithms within a module. Because side effects propagate across program interfaces, it is essential to check all data flowing from module to module (or to the outside world) to ensure that the data conform to bounds established during requirements analysis.

2.5.2. User Interface Design

User interface design has as much to do with the study of people as it does with technology issues. Who is the user? How does the user learn to interact with a new computer-based system? How does the user interpret information produced by the system? What will the user expect of the system? These are only a few of the many questions that must be asked and answered as part of user interface design.

2.6. Software Testing

Testing presents an interesting anomaly for the software engineer. Earlier in the software process, the engineer attempts to build software from an abstract concept to a tangible implementation. Now comes testing. The engineer creates a series of test cases that are intended to "demolish" the software that has been built. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

Software developers are by their nature constructive people. Testing requires that the developer discard preconceived notions of the "correctness" of software just developed and overcome a conflict of interest that occurs when errors are uncovered.

2.6.1. Testing Objectives

A number of rules that can serve well as testing objectives:

- (1) Testing is a process of executing a program with the intent of finding an error.
- (2) A good test case is one that has a highly probability of finding an as-yet undiscovered error.
- (3) A successful test is one that uncovers an as-yet undiscovered error.

If testing is conducted successfully (according to the objective stated above), it will uncover errors in the software. As a secondary benefit, testing demonstrates that

software functions appear to be working according to specification and that performance requirements appear to have been met. In addition, data collected as testing provides a good indication of software reliability and some indication of software quality as a whole.

2.6.2. Testing Principles

Before applying methods to design effective test cases, a software engineer must understand the basic principles that guide software testing.

- (1) All tests should be traceable to customer requirements.
- (2) Tests should be planned long before testing begins.
- (3) The Pareto principle applies to software testing.
- (4) Testing should begin "in the small" and progress toward testing "in the large".
- (5) Exhaustive testing is not possible.
- (6) To be most effective, testing should be conducted by an independent third party.

2.6.3. Testability

Software testability is simply how easily a computer program can be tested. Since testing is so profoundly difficult, it is necessary to know what can be done to streamline it. Sometimes programmers are willing to do things that will help the testing process, and a checklist of possible design points, features, and so on can be useful in negotiating with them.

III. CURRENT SYSTEM ANALYSIS

3.1. Level 0 Data Flow Diagram, Context Model

We can analyze the current examination system by using "The Functional Modeling And Information Flow" concept. The Figure 3.1 shows a fundamental system model or a context model.

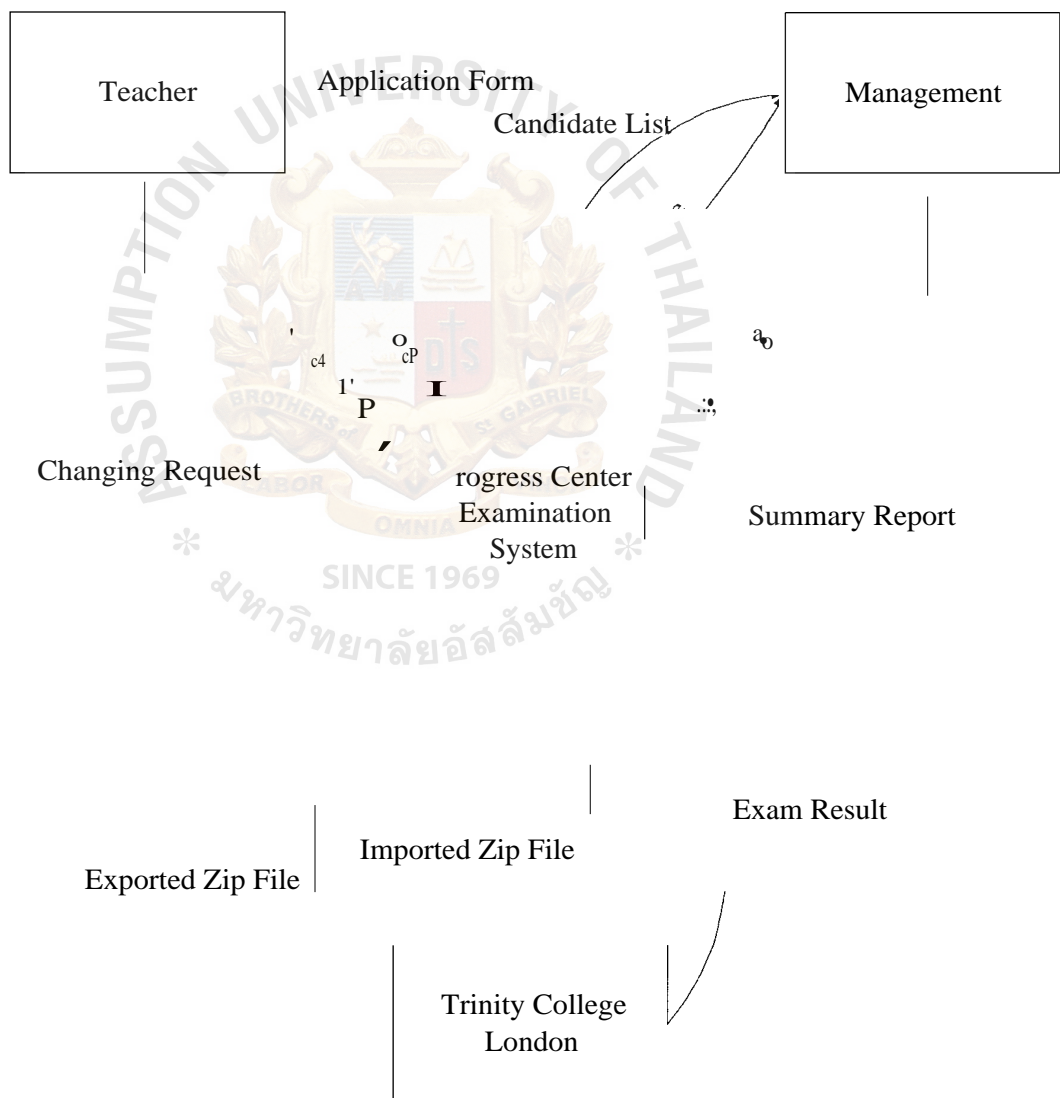


Figure 3.1. Context Model of Progress Center Examination System.

From the above figure we know that the system has 4 input, 7 output and three external entity. The next level of data flow diagram will reveal more details about the system, how the information flows and what the major processes are in this system.

3.2. Level 1 Data Flow Diagram

Figure 3.2 illustrates all major processes that transform the incoming data to the outgoing information. The First process is to receive the application form from the teacher. All candidates must have one reference teacher as they cannot apply themselves. The application forms will be verified and sent to the second process and then they will be sorted by requested examination date and grouped by the teacher. At the end of application period, the candidate list will be generated and submitted to management level. Next, data will be keyed in to CDE2000 (Candidate Data Entry 2000 System). The summary report is generated by CDE2000 and submitted to management level. Next, CDE2000 will export the data into 6 files that must be zipped into one file and then sent to Trinity via E-mail. When Trinity receives the zip file, it will update the data into Trinity System, assign automatically the "TCL Number" that identifies the candidate while the examination session, zips the update data and sends it to Progress Center.

When Progress Center receives the zipped file, it will be extracted into 6 text files and imported into database of CDE2000. The timetable will be generated by program but the timetable must be modified because it does not fit to use. Therefore timetable must be adapted manually in the seventh process and then printed to management level. The appointment slip will be generated and submitted to the teacher to inform when the examination dates are and what their student's TCL No are.

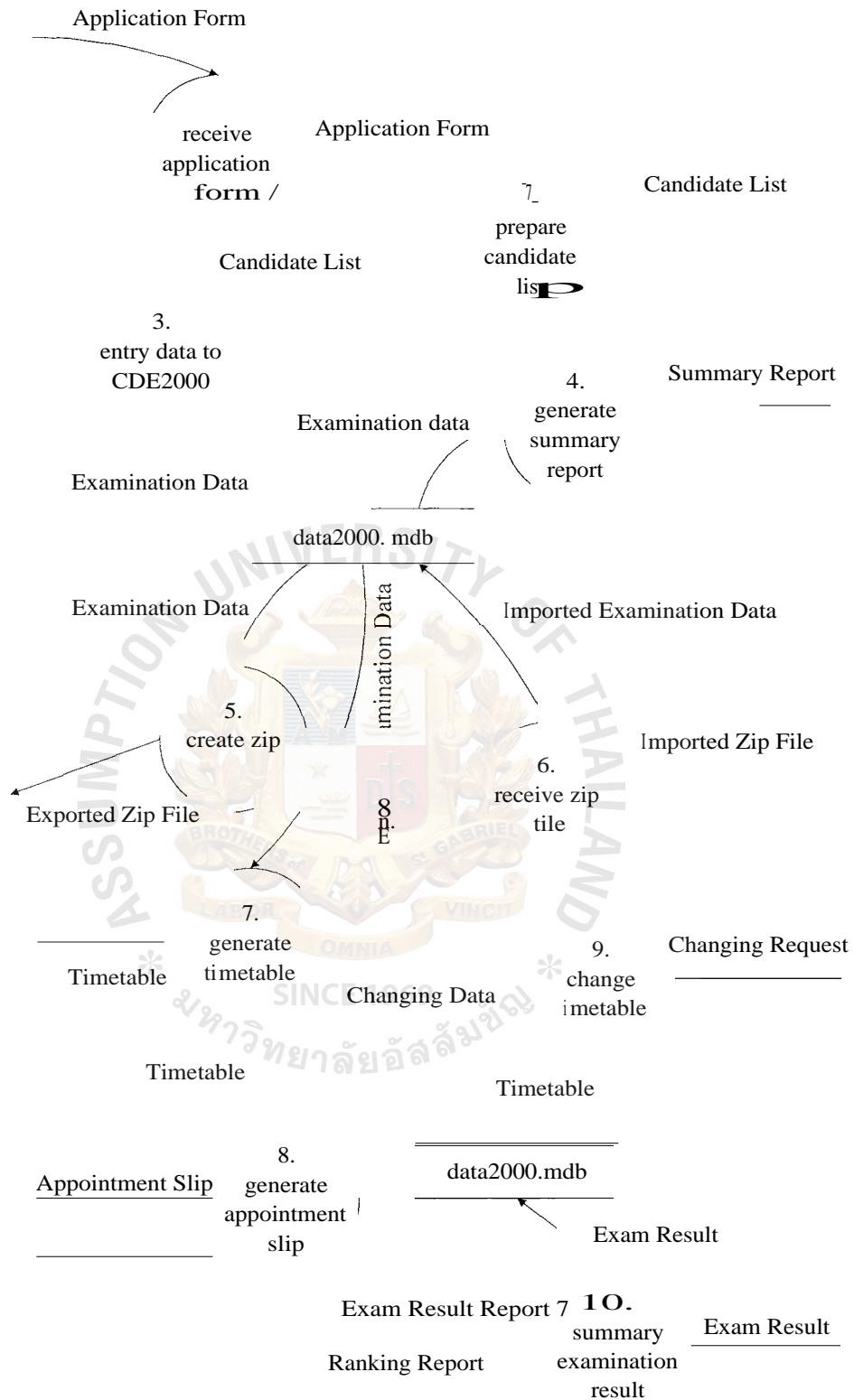


Figure 3.2. the Level 1 DFDs.

Some candidates cannot take an examination on the assigned date and some teachers want to swap the examination date among their students or want to postpone the examination so they must submit the changing request to Progress Center. In the ninth process, Progress Center will verify the request whether the timetable can be modified or not. If their request is accepted, the timetable will be modified and a new one will be printed and sent to management. When the examination finishes the examiner will go back to London and the result will be entered to the system of Trinity. The examination result will be sent to Progress Center and then they will be sent to the teacher. Moreover, Progress Center will sort and rank the candidate result for each grade of each instrument.

After we analyze the process and information flow, there are 50 manual processes that we should develop another system to replace the manual job to reduce time and effort. Process number 3,4,5,and 6 are the process of CDE2000 while process number 1,2,7,8,9,and 10 are manual processes. Therefore we must further analyze the manual process to get more information that we can design and develop the prototype software. The level2 DFDs of the manual processes are included in Appendix B.

From the level 1 DFDs we can conclude that we can divide the manual processes into 2 phases, before exporting from CDE2000 and after importing to CDE2000. Next step is to identify the problem of the existing system and the requirement of the new system.

3.3. Problem and Requirement

- (¹) Data input to CDE2000 should be sorted by examination date and on each examination date data should be grouped together by a teacher because teachers who send candidates usually come to see their students and do not

want some candidates to be inserted among their candidates. The grouped candidate list is easier for Progress Center to find or search the information.

- (2) Before the end of the application period, some teachers or candidates may want to postpone the examination date so the new system must rearrange the sequence of the candidate.
- (3) More reports are needed for management decision. Enquiry from screen is preferred.
- (4) The new system must be able to generate the printable timetable that includes these functions: break insertion, moving candidate up and down in each day, canceling the candidate, postponing the candidate to the other days, and swapping between two candidates.
- (5) Examination Result Appointment Slip and Certificate Appointment Slip must be generated and printed from the new system.
- (6) The examination ranking result divided by each grade of each instrument is automatically calculated after the examination results are input.

3.4. Database Analysis

Using Microsoft Access 95 as a database engine CDE2000 is developed with Visual Basic 6.0 and connected to database with DAO. Before we design another system we must know the structure of the existing database in order to design the new database appropriately.

Because of the database security we will not manipulate directly the CDE2000 database but we will create a new database instead. Input Data will be stored to new database and then they will be updated to CDE2000 Database with SQL command at the end of application period and after data from TCL are imported to CDE2000 the new system will update its database either. However the master file maintained in

CDE2000 such as "Exam Fee Table", "Teacher Table", and etc. must be shared. Figure 3.3 illustrates the flow of data between two databases.

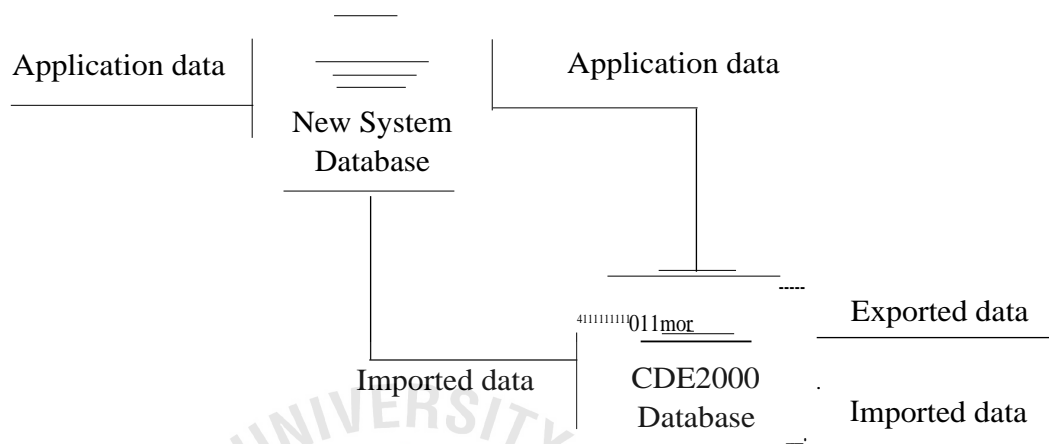


Figure 3.3. Flow of Data between Two Databases.

Hence, we must know which tables in CDE2000 database or data2000.mdb file are the master files that we can query data from and which tables we have to update data from the new system database and what is the appropriate key or index to identify the record in those tables.

From the figure 3.4 there are 16 tables in data2000.mdb file. These tables can be separated in three categories. First category is the master file and we will not update data in these tables that just only read data so we can share these tables. Tables in this category are "ExamFee" which maintains the fee of examination for each grade in each instrument, "GradeSubject" which keeps the information about each grade in each subject such as subject code, maximum score in each section, length of examination, and so forth. "School" and "Teacher" are in the first category.

Rig Data2000 : Database				
Tables	Queries	Forjris		
CandidateRegistration	17	Teacher	21 ³ en	I
Centre		TimeTable	Design	
Country		TimeT ableB in		
ExamFee		TimeTableDays	NP"	
FeeAdjustments		r=i TimeTableDrag		
GradeExam		EI TimeT ableFixed		
GradeSubject		EI Users		
School		EI Venue		

Figure 3.4. List of Table in Data2000.mdb.

Tables in the second category are updated by the SQL command from the new system in order to be manipulated by CDE2000 and exported to TCL. These tables are "CandidateRegistration" and "GradeExam".

The last category contains the tables that don't relate to the new system. They are used by the CDE2000 only not for the new system such as TimeTable but we will create the new one in the new database instead.

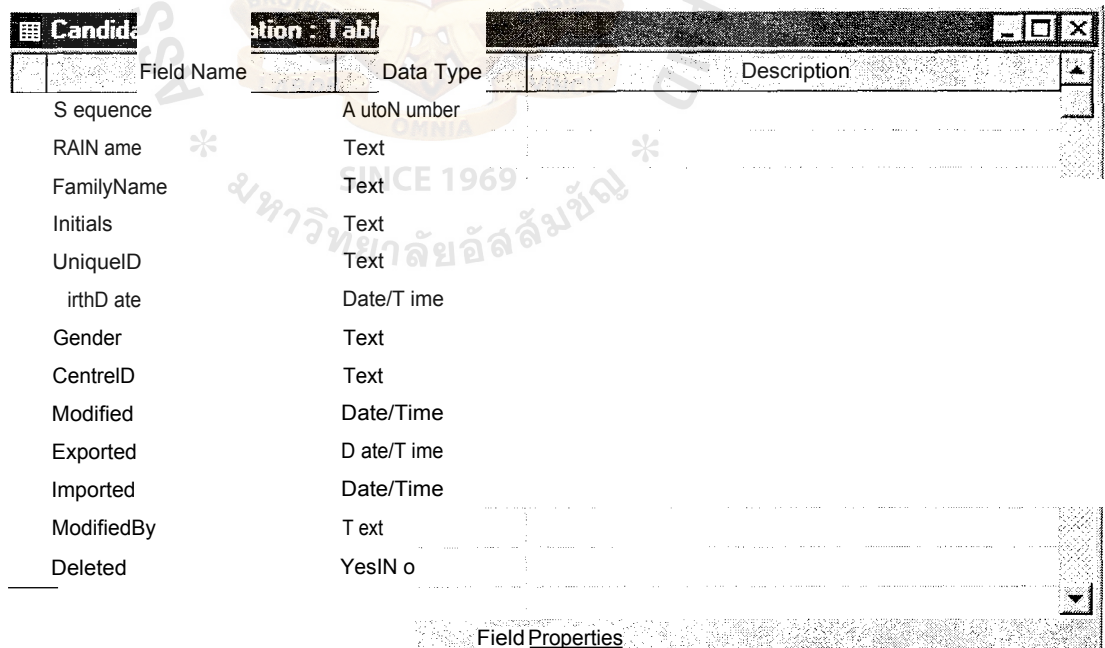
Therefore we analyze only the following tables.

- (1) ExamFee keeps the information about the fee of the examination for each grade in each subject. This table maintains not only the fee of the current year but also the history.
- (2) GradeSubject contains the data of the examination subject in each grade.

There are a lot of fields in this table that should be split. Primary Key is the 5 meaning digits "SubjectID". The first two digits come from grade field while

last two digits come from the subject code field. The mid-digit is used to identify the year of examination.

- (3) School table keeps the candidate's school data. Primary key is SchoolID and CenterID.
- (4) Teacher table keeps the record of teachers who send their student to take the examination. Primary key is TeacherID and CenterID.
- (5) CandidateRegistration maintains data about the candidates. Primary key is Sequence field and the UniqueID field is the index that will be assigned by TCL. All of the attributes in this table are shown in the Figure 3.6. FullName field must be required, Sequence is automatically recorded by input order, and UniqueID is assigned by TCL.



Field Name	Data Type	Description
Sequence	AutoNumber	
FullName	Text	
FamilyName	Text	
Initials	Text	
UniqueID	Text	
BirthDate	Date/Time	
Gender	Text	
CentreID	Text	
Modified	Date/Time	
Exported	Date/Time	
Imported	Date/Time	
ModifiedBy	Text	
Deleted	Yes/No	

Field Properties

Figure 3.5. CandidateRegistration Table.

(6) GradeExam is the table that contains the examination data. This table is very big, so it can be separated into two sections. The first section maintains the application data and the second section keeps the examination result for each candidate. Because we will not use this program for generate timetable, we will focus on the first section only.

GradeExam : T1			AO
Field Name	Data Type	Description	
CandidateSequence	Number		
Sequence	AutoNumber		
UniqueID	Text		
SessionID	Text		
UniqueGradeExamID	Text		
MonthYear	Text		
ExamDate	Date/Time		
ExamTime	Text		
SubjectID	Text		
TeacherID	Text		
FullTeacherID	Text		
TrinityTeacherID	Text		
SchoolID	Text		
SchoolID	Text		
TrinitySchoolID	Text		
VenueID	Text		
FLIP/VenueID	Text		
TrinityVenueID	Text		
CentreID	Text		

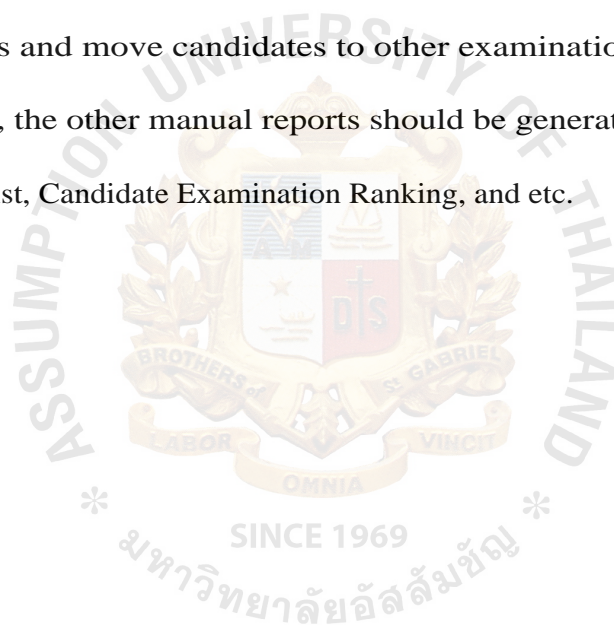
Figure 3.6 GradeExam Table

Figure 3.6 illustrates only the first section of GradeExam Table that includes only the application data. Primary key is CandidateSequence with Sequence. SessionID maintains TCL No that is after importing data from TCL, this field must be updated.

3.5. Summary

We can conclude that the new system must append the application data into two tables that are CandidateRegistration and GradeExam. After that we can use the CDE2000 to generate the summary report for management verification. Then data will be exported by CDE2000 in order to submit to TCL. TCL No. will be assigned and sent back to Progress Center. When we import the TCL No. to CDE2000, the new system will retrieve TCL No. from CDE2000 and generate Timetable.

Timetable module of the new system must be able to change the data, swap candidates and move candidates to other examination date when they request. Moreover, the other manual reports should be generated by the new system e.g., Absentee list, Candidate Examination Ranking, and etc.



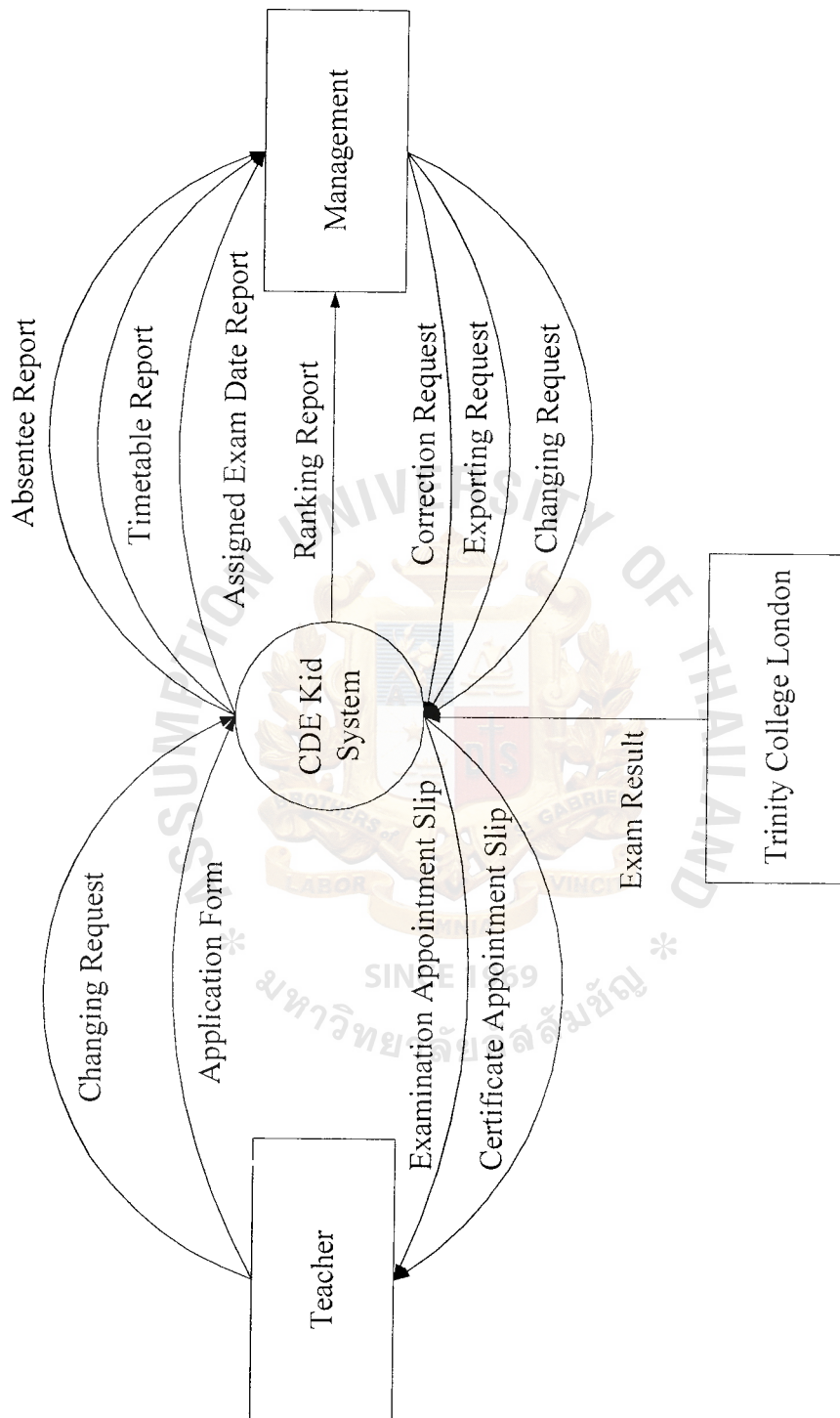
IV. NEW SYSTEM DESIGN

4.1. Level 0 Data Flow Diagram, Context Model

After we finish the analysis phase, designing a new system is necessary. Figure 4.1 shows the context level of the new system named "CDE Kid". There are 3 external entities same as the existing system but there are both 6 input and output data. Next, level 1 DFDs is designed to illustrate how the input data are transformed to output and what those processes are.

4.2. Levell Data Flow Diagram

Figure 4.2 illustrates the Levell DFDs of CDE Kid that contains 11 main processes. The first process is to file the candidate application foini into database via screen interface. When all application forms' data are entered, the second process, candidate sorting, is done to group the data together by requested examination date and teacher. Next, these data will be sorted into the proper sequence in order to be searched easily. The "Assigned Exam Date" report will be generated and then submitted to the management level. Management may have some requests to change or correct some data such as moving or swapping some candidates, correcting the false data, rearranging the candidate sequence, and etc. These correction requests will be implied in the forth process and the new Assigned Exam Date report must be printed and sent to the management again. If there are no more correction request, CDE Kid will export data into the database of CDE2000, data2000.mdb, and save the exporting status to the database to ensure that it should not export data again. Otherwise, the unexpected error may occur. When we finish the exporting data process, we will move to do the CDE2000 proc.:sses that are printing summary report and exporting data into text files, zip them, and send to TCL.



Context Model of CDE Kid System.

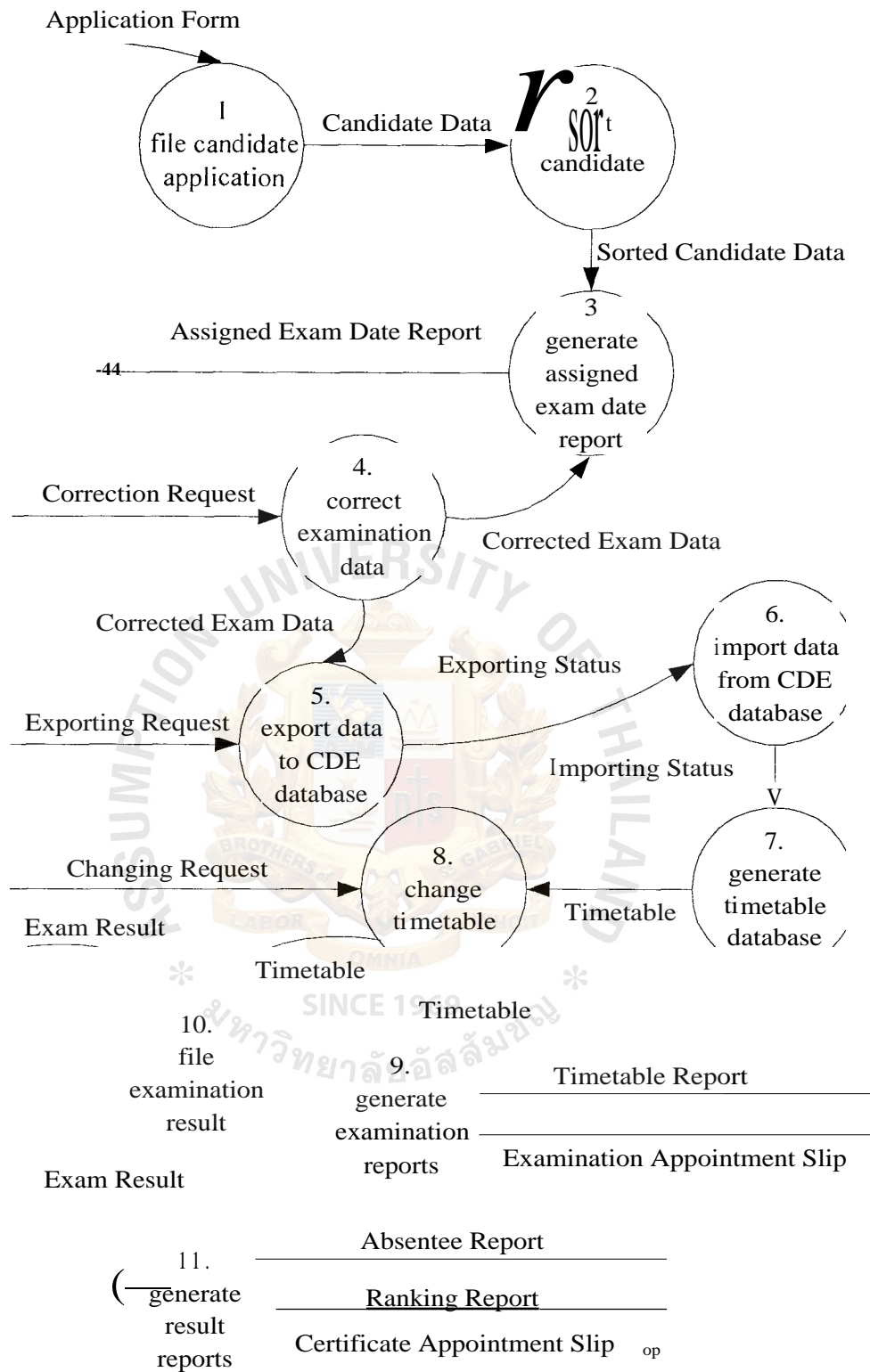


Figure 4.2. Level1 DFDs of CDE Kid System.

When the data from TCL is imported to CDE2000, the sixth process will update the data from data2000.mdb to CDE Kid. Next, the timetable will be generated and then this timetable must be modified to be the most suitable timetable in the eighth process. In the following process the timetable will be printed and sent to management to review and the examination appointment slip will be printed and sent to teachers to confirm the examination date and time for their student. However, the changing requests from both management and teachers must be fulfilled. These changing requests may be the break insertion, changing the examination sequence, postponing or changing the examination date and time, canceling the examination, swapping examination date among candidates. Therefore, the eighth and ninth process may be done again and again until the examination finishes.

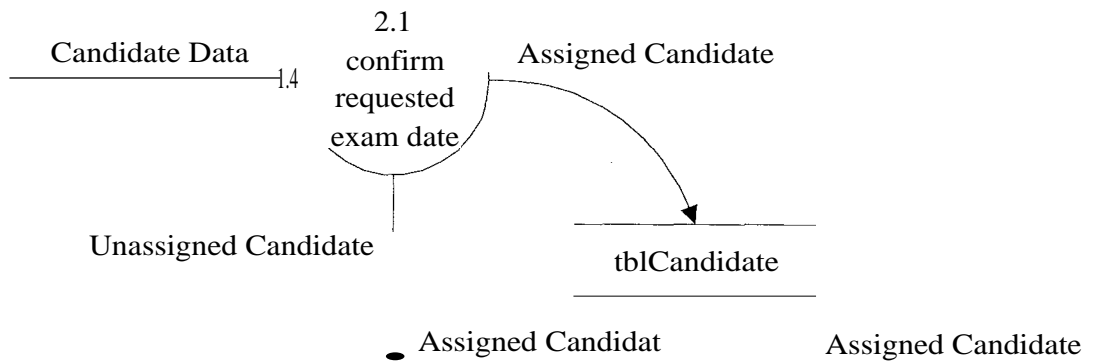
Around two weeks after the last examination date, the examination results and certifications will be sent by TCL to Progress Center. The examination results must be keyed in and then the result reports will be printed. These reports are the absentee report, ranking report, and certificate appointment slip.

Process number 2,7,8 have the Level2 DFDs to show more details of these processes while the others cannot be divided into sub-process.

4.3. Level2 Data Flow Diagram

4.3.1. Level2 DFDs of Process 2

Process 2, which is candidate sorting, transport is the candidate data from process 1 to sorted candidate data and then sends to process 3. The first sub-process is to confirm the requested examination date. If it is able to assign the examination date by the requested date, it will be done. Otherwise, the process 2.2 will be implied to assign the suitable examination date instead.



12.'7

assign the
exam date

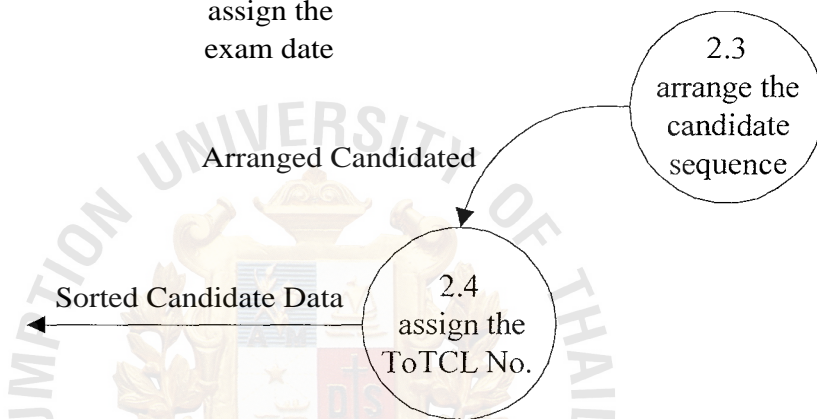


Figure 4.3. Level2 DFDs of Process 2.

When all candidates are assigned their proper examination date, the process 2.3 will begins to arrange the sequence, the candidate data will be grouped and sorted ascending by assigned examination date and teacher, order number in "Order" field will be assigned respectively. Then, the candidate sequence will be arranged in the proper order. Next, the ToTCL number will be assigned in the process 2.5. The candidate examination data ordered by ToTCL number are now "Sorted Candidate Data".

4.3.2. Level 2 DFDs of Process 7

The forth process, examination data correction, is illustrated in figure 4.4

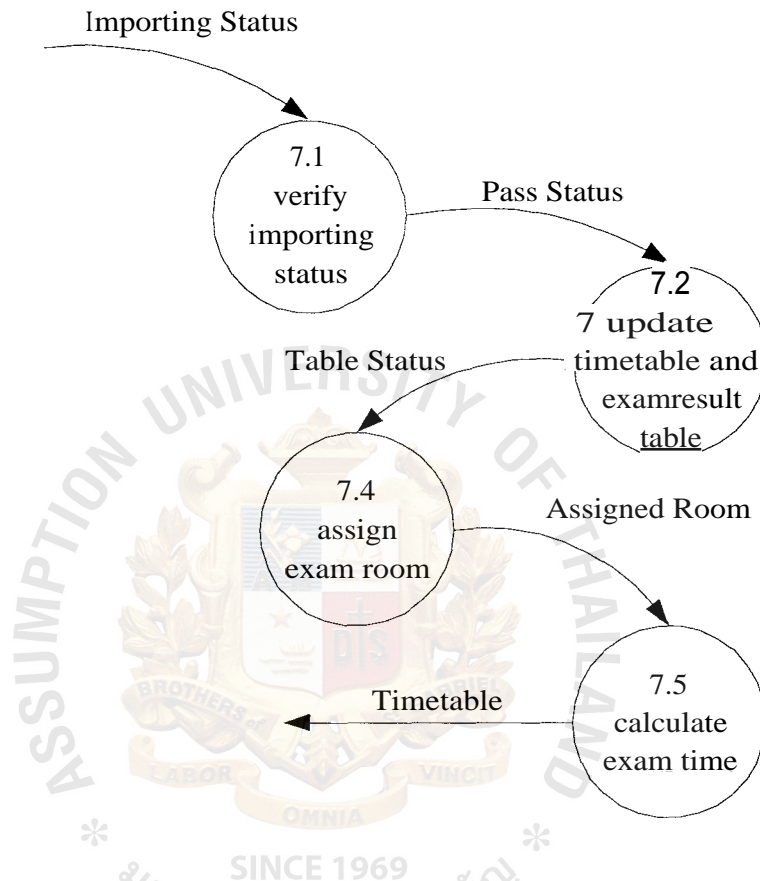


Figure 4.4. Level 2 DFDs of Process 7.

If the imported data are updated to CDE Kid database, the process 7.2 will update data into TimeTable and ExamResult table. The candidates will be assigned the examination room and then the start and finish examination times are calculated to accomplish the rough timetable.

4.3.3. Level 2 DFDs of Process 8

After the timetable is generated in the seventh process, it is not complete yet because the break must be inserted three times a day, two times for coffee breaks and lunch break for examiner. Moreover, the changing examination date, sequence, and room functions are required. Figure 4.5 shows these sub-processes.

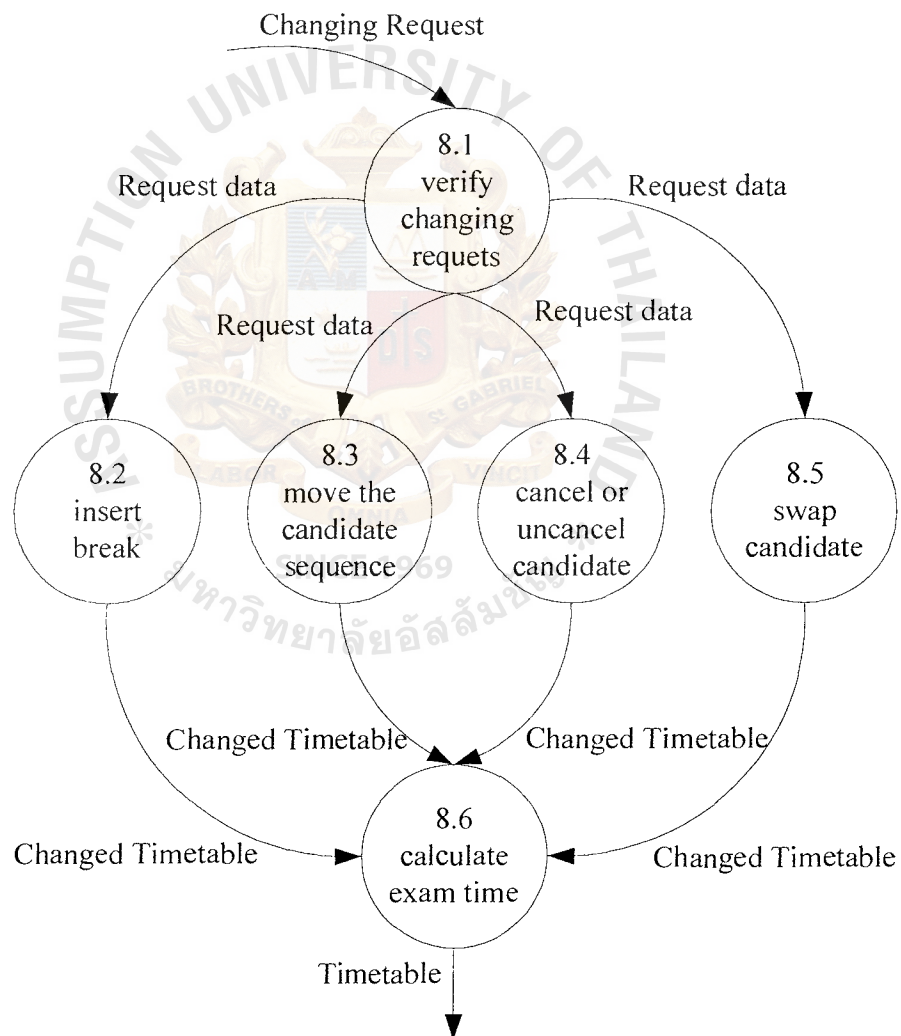


Figure 4.5. Level 2 DFDs of Process 8.

4.4. Entity Relationship Diagram

From the context model and Data Flow Diagram we can draw the relationships among the entity as shown in Figure 4.6.

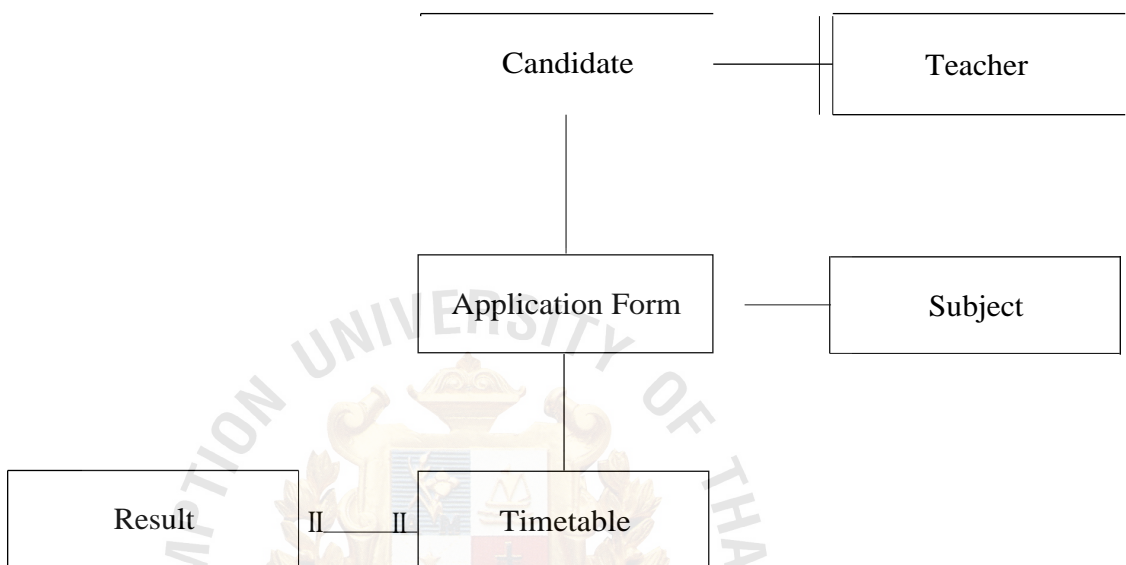


Figure 4.6. Entity Relationship Diagram.

From the ERD, there are six entities and 5 relationships. The relationship between candidate and teacher is many to one, which the meaning is each candidate must be instructed by one teacher while each teacher may teach many students. Candidate can apply more than one examination, usually one theory and one practical examination, not two practical instruments. On the other hand, one application form belongs to one candidate with only one subject. So the relationship between candidate and application form is one to one. One examination subject may not be applied or may be applied many times by many candidates. Each timetable has just only one examination result and one result belongs to only one examination time.

4.5. Process Specification

From the level 1 DFDs the most complicated and important processes are process no. 5 and 6 that are the exporting and importing process respectively because this process will interact or exchange data between CDE2000 and CDE Kid. So we will design the process specification (PSPEC) for these processes.

4.5.1. PSPEC of Process 5

Name: Export Data to CDE Database

Receive: Exporting Request, Corrected Exam Data

Process:

IF ExportingStatus = False Then

DELETE * FROM CandidateRegistration IN Data2000.mdb;

INSERT INTO CandidateRegistration IN Data2000.mdb

SELECT * FROM Candidate;

DELETE * FROM GradeExam IN Data2000.mdb;

* INSERT INTO GradeExam IN Data2000.mdb SELECT *
FROM Application Form;

SET ExportingStatus = TRUE;

ELSE

Print "This process is done already";

END IF

Output: ExportingStatus

Remark: Corrected Exam Data = Candidate + Application Form

(More details see Appendix C, Data Dictionary)

4.5.2. PSPEC of Process 6

Name: Import Data from CDE Database

Receive: ExportingStatus

Process:

If ExportingStatus and not ImportingStatus then

UPDATE BackTCL From Candidate BY SessionID From
CandidateRegistration IN Data2000.mdb WHERE

ToTcl = Sequence;

DELETE * FROM ExamResult;

INSERT INTO ExamResult (BackTCL) SELECT BackTCL
FROM Candidate;

DELETE * FROM TimeTable;

INSERT INTO TimeTable (ExamDate,Room, DateSequence,
BackTCL, ExamMin) SELECT ExamDate, 1 As DateSequence,

* BackTCL, BackTCL, LengthOfExam FROM Candidate INNER
JOIN GradeSubject ON SubjectNo = SubjectID WHERE
BackTCL Is Not Null And Candidate.ExamDate Is Not Null

ORDER BY BackTCL;

ImportingStatus = False;

ELSE

Print "This process is done already";

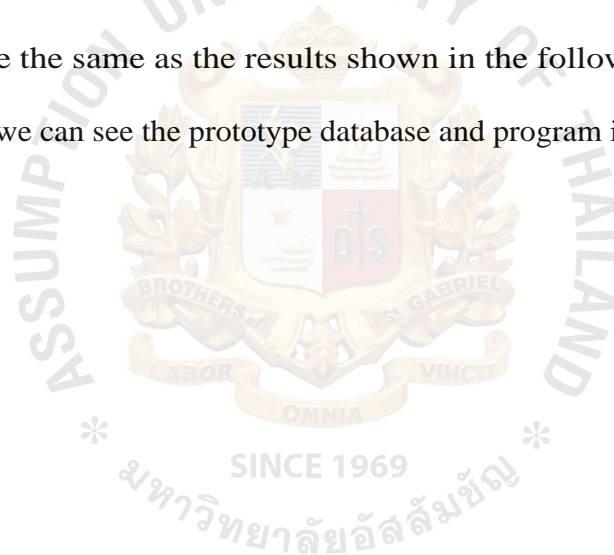
END IF

OUTPUT: ImportingStatus

4.6. Summary

In this chapter, we apply the system design concept by designing the Data Flow Diagram in Context level, level 1, and level 2. From the DFDs we define what is input, output, and process and how the data flow in this system. The Entity Relationship Diagram is designed either to identify which entities we must maintain for their information. Data Dictionary design results are in the appendix C to define the meaning and the structure of Data Flow and Data Store.

There are two complicated processes when we design the Process Specification. Database Design and Interface Design are excluded from this chapter because both designs are the same as the results shown in the following Chapter, Prototyping. Therefore, we can see the prototype database and program interface in Chapter 5.



V. PROTOTYPING CANDIDATE DATA ENTRY SUPPORT SYSTEM (CDE Kid).

To develop the prototype system, we select Microsoft Access 97 as a development tool because of the following reasons.

- (1) It is easy to develop and understand. MS-Access is able to use VBA (Visual Basic for Application), which is the subset of Visual Basic that will be a development tool for the real system to develop a source program or a module. If we need to use Visual Basic 6.0 to develop client — server application or ASP to develop web based application, it's very easy to convert VBA in MS-Access to new application.
- (2) It is able to link database folin Data2000.mdb, which is the MS-Access 95 database, to CDE Kid. The Linked Table method provides the faster result in query data than other methods such as DAO. It is able to see the table linked from data2000.mdb and join table together in query as the original table created in CDE Kid. Moreover, MS-Access provides many powerful tools that help us developing application easier.
- (3) There is only one computer to run this application. If we need more computers to run this system, we can further modify this application based on Access Client-Server application concept.
- (4) There are 2 examination sections each year and there are around 2,000 candidates. So, it is estimated that there are around 5000 records for CDE kids with in one year. Number of candidates will be increased by 10% each year. Regarding to MS-Access capability, it can handle this situation with out any problems.

5.1. Database

In this step we will map the ERD from chapter 4 into the database structure and create tables that store the data.

5.1.1. Candidate Table

Table 5.1. Candidate Table.

FIELD	TYPE	DESCRIPTION
CandidateID	Text, 4 digits	Candidate Identify Number. Primary key.
Name	Text, 50 digits	Candidate Name
Surname	Text, 100 digits	Candidate Surname
Gender	Discrete, 1 digit	1=Male, 2=Female
BirthDay	Date	Short Date Format

Table 5.1 Illustrates the candidate table maintaining the candidate information. We store only the necessary data such as name, surname, gender, and date of birth. Progress Center will give a diploma to those who have got the highest score in each subject. If there are more than one candidate, the youngest candidate will get the rewarded so the date of birth must be maintained. Other information (such as telephone, address, etc.) are not required because Progress Center will contact the candidates through their teacher.

5.1.2. Application

Table 5.2. Application Table.

FIELD	TYPE	DESCRIPTION
AppNo	Text, 4 digits	No. of application Form (Primary key)
AppDate	Date	Submitted Date
Sequence	Integer	Input Sequence
CandidateID	Text, 4 digits	Foreign key
SubjectNo	Text, 5 digits	Foreign key
CertificateSchool	Boolean	Verify if school's name in certificate is included.
CertificateTeacher	Boolean	Verify if teacher's name in certificate is included.
CertificateTQual	Boolean	Verify if teacher's qualification in certificate is included.
SchoolID	Text	Foreign key
TeacherID	Text, 8 digits	Foreign key
Order	Integer	Sorted Order No
ToTCL	Integer	Export to CDE sequence
BackTCL	Text, 8 digits	TCL No. (Exam No.)
AssExamDate	Date	ExamDate Assigned by Progress

From The table 5.2, primary key of this table is AppNo that is the running number of application form. CandidateID is the foreign key used to be linked to Candidate table, These foreign keys SubjectNo, TeacherID, and SchoolID are used to retrieve data from GradeSubject, Teacher, SchoolID table in Data2000.mdb respectively.

5.1.3. TimeTable

Table 5.3 shows the structure of TimeTable which stores the timetable information including breaks for each day.

Table 5.3. TimeTable Table.

FIELD	TYPE	DESCRIPTION
ExamDate	Date	Exam Date assigned by TCL
Room	Integer	Examination Room 1 or 2
Sequence	Integer	Exam Sequence for each room on each day
BackTCL	Text, 8 digits	TCL No.
ExamMin	Integer	Length of exam in minute
AccExamMin	Integer	Start time in minute
FinishMin	Integer	Finish exam time in minute
Cancel	Boolean	

Primary key is the ExamDate, Room, Sequence fields combined together. BackTCL is the foreign key to retrieve the candidate examination data. Although we can retrieve the ExamMin data from other table by using foreign key but it's often used to refresh and recalculate other fields' data. So, we denormalize this table in order to get more efficiency and accept somewhat redundancy.

5.1.4. ExamResult

The data from examination report submitted from TCL are stored in the ExamResult table. The structure of this table is shown in Table 5.4.

Table 5.4. ExamResult Table.

FIELD	TYPE	DESCRIPTION
BackTCL	Text, 8 digits	Primary Key
ExamTime	Text, 5 digits	
Result1	Integer	
Result2	Integer	
Result3	Integer	
Result4	Integer	
Result5	Integer	
Result6	Integer	
Result7	Integer	
Result8	Integer	
Result9	Integer	
Result 10	Integer	
Total	Integer	
Rank	Integer	

5.1.5. Linked Table

From the ERD we should design 3 more tables that are the GradeSubject, School, and Teacher. But these tables are already in the Data2000.mdb so we will link these tables to get data as the master files that will not be inserted, deleted, or updated any data in these tables. The database structure for these tables are included in Appendix C.

5.1.6.CDE Kid System Table

In order to keep the system status we design two more tables. The first table is the Option table illustrated in Table 5.6 and the second one is DeletedTimeTable table that maintains the timetable data deleted by user. If candidate who requests to cancel the examination changes his/her mind, we can restore his/her data into timetable. The table structure of DeletedTimeTable is the same as TimeTable.

The meaning of each field in Option table is defined in description of Table 5.6.

Table 5.6. Option Table.

FIELD	TYPE	DESCRIPTION
Initial	Text, 4 digits	CenterID of Progress Center
SubjectYear	Integer	1 digit year of examination
CDEAddress	Text, 50 digits	Application path of CDE2000
blnImport	Boolean	Verify if data are exported to Data2000.mdb
blnExport	Boolean	Verify if data are imported from Data2000.mdb
binTable	Boolean	Verify if TimeTable data are generated

5.2. Input/Output Interface

We use the screens or forms to receive the input from data entry and the reports to be output interface. Reports will be displayed in the print preview mode first then user will make a decision whether to print it or not. Screens are in the GUI (Graphic User Interface) mode that user can interface in the graphic mode. Not only the system procedures but also the events of interface must be concerned in the prototype

development phase. Figure 5.1 illustrates the example of screen input, Application Form.

Application Form is mapped from the first process of new system Level 1 DFD in Chapter 4 to file the application form data to database. There are a lot of events we must design more to fulfil the requirement. For example, after we select the examination subject from drop down menu, grade and subject code must be displayed in the label.

Therefore, we must test the prototype to ensure that it processes correctly as we design and works properly. More functions may be added into the form such as the record find function (binocular icon or the right button) that helps users to search their desired record easily.

Application Form

NO: 1335 AppNo: 1001 Appl. Jton Date: 16 Apr 2002

Candidate

Name: IChavajed Maskulrat
 Date of Birth: 18/03/1883
 Gender: Female

Examination Type

Subject: [Dropdown]
 Grade: 01 Code: 05

Certificate Information

School: iBangkok Christian College
 Teacher Name: Bancha anavijit
☒ Teacher's Qualification

Feat [Icons]

Figure 5.1. Application Form Input Screen.

St. Gabriel's Library, Ail

For the output interface or report, we may create form to input the criteria used to generate the desired report. Figure 5.2 shows the "Print Application By Teacher" screen. This screen will receive the criteria for retrieving data to be shown in the report. To design the appropriate screen, we should ask the user requirement again to know the exact criteria they need.

in Print Application By Teacher

.....
Select All

Select by TeacherID

Select by Teacher Name

PreView

-ancel

Figure 5.2. Print Application by Teacher Screen.

The whole input/output interface screens are in Appendix D.

5.3. Coding

We both use Access Macro and VBA for coding the prototype. Access Macro is a faster method to generate output but it cannot do the complicated jobs. Therefore, we need VBA to develop the complicate processes.

To connect to the database, we use two concepts that are SQL (Structure Query Language) and DAO (Data Access Object).

5.4. Summary

The prototype of CDE Kid is a developed base on the system analysis and design concept documented in the Chapter 3 and 4. Before lunching this prototype, we must test and evaluate it to ensure that all functions can be used well.

VI. TESTING AND EVALUATION.

6.1. Prototype Testing

To ensure that the prototype application can accomplish all requirements with the most efficiency and effectiveness, we have to test them with the real data and situations.

The test case must be designed with a lot of different cases in order to test all functions.

- (1) 20 Candidates, sent by 3 teachers, apply for both practical and written examinations in the several of grades and subjects.
- (2) 5 Candidates want to change their request date before the data are exported to TCL.
- (3) Export data to CDE.
- (4) Assign TCL No to CDE and the import data back to CDE Kid.
- (5) In one day candidates are allocated into two exam rooms.
- (6) One candidate wants to change the sequence; another candidate wants to change the exam date. Two candidates want to swap their exam date.
- (7) Input the examination result for 17 candidates Three more candidates are absentees.
- (8) Generate the examination ranking.

The test case data input and the result are shown in Appendix E, Testing Input and Output.

If there are any errors or problems, we must go back to the prototyping stage to fix the errors. The testing and coding are implied further and further until the prototype is able to work properly without any errors. Then the prototype is a lunched to use as the parallel system with the existing sys'.m.

6.2. Parallel System

To ensure the CDE Kid prototype, we ran the system together with the manual system for the second examination session in 2001. This session began the application period from August 2001 till September 2001. The data back from TCL came in the mid of October 2001. The examination period started on November 2, 2001 and finished before Christmas (December 22, 2001).

When the prototype was completed and ready to run, it was in the examination period. Therefore, Progress Center management assigned one officer to enter data to CDE Kid and process the system the same as the current system.

This examination was finished on December 22, 2001 and the last examination report was submitted by TCL on December 26, 2001. Progress Center finished all tasks on January 9, 2002.

The following tables show the time efforts to accomplish the second examination section in 2001 with 746 candidates by the manual system and CDE Kid respectively.

Table 6.1. Time Efforts for Manual System.

Tasks/Processes	Time Efforts	Man-Hours
1. Sorting Candidate List	3 officers within 5 days	112.50
2. Enter data to CDE2000	1 officers within 1 days	7.50
3. Generate TimeTable with MS-Excel.	3 officers within 5 days	112.50
4. Generate Examination Appointment Slip	3 officers within 2 days	45.00
5. Change TimeTable information during examination period.	Average 0.5 man-hour a day (35 days)	17.50
6. Result and Ranking Summary Candidate	3 officers within 3 days	67.50

Table 6.1. Time Efforts for Manual System. (Continued)

Tasks/Processes	Time Efforts	Man-Hours
7. Generate the candidate Exam Result, Certificate List, and Label submitted to teacher.	5 officers within 3 days	112.50
Total		475.00

From Table 6.1 we notify that 1 working day has 7.5 hours. For Task 5, changing timetable information, we estimate the average time efforts because we do not record the changing request. This information is estimated by the management level.

Task 6 is to verify who passes the examination and find out the top 3 highest result for each grade and subject. Task 7 is to generate the certificate list and label. Then Progress center submits the exam results to the teacher by mail enclosed with Certificate appointment slip.*

Table 6.2. Time Efforts for CDE Kid.

Tasks/Processes	Time Efforts	Man-Hours
1. Enter Data to CDE Kid	1 officers within 4 days	30.00
2. Sorting Data with CDE Kid	2 officers within 2 days	30.00
3. Export data to CDE2000	1 officers within 0.5 hour	0.50
4. Import data and Generate TimeTable.	1 officers within 2 days	15.00
5. Generate Examination Appointment Slip	1 officers within 1 hour	1.00

Table 6.2. Time Efforts for CDE Kid. (Continued)

Tasks/Processes	Time Efforts	Man-Hours
6. Change TimeTable information during examination period.	Estimated 25% of manual	4.38
7. Result and Ranking Summary Candidate	1 officers within 1 days	7.50
8. Generate the candidate Exam Result, Certificate List, and Label submitted to teacher.	1 officers within 1 days	7.50
Total		95.88

6.3. Evaluation

This section will evaluate the CDE Kid in economic terms by applying the engineering economic analysis concepts. The Replacement Analysis with NPW (Net Present Worth) is selected. Management assumes the interest rate is equal to 0%.

Management forecasts that the candidate rate will grow by 10% each year and for this year the summary candidates for both sessions are around 2,500 applications. CDE Kid has 5 life years. The average salary for one officer is around 10,000 baht a month (26 working days), so it is 51.28 baht for one man-hour. Salary is assumed to increase 5% every year. The information for the manual system is as follows.

For 746 applications use 475 man-hours, so we estimate that it is equal to 0.6367 man-hour per one application. Table 6.3 shows the cash flow for the manual system.

So, we can calculate the cost of man-hours used to accomplish the project for each year. The cash flow has the minus sign to indicate that it is the out flow or we pay cash

to operate the system. With the same situation, we can calculate the cash flow for CDE Kid.

Table 6.3. Cash Flow for Manual system.

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5
No. of Applications		2000	2200	2420	2662	2928
Man-hours		1273.40	1400.74	1540.81	1694.90	1864.26
Est. baht/man-hours		51.28	53.84	56.53	59.36	62.33
Cash Flow (Baht)	- 65,300	- 75,416	- 87,102	- 100,609	- 116,199	

First we will calculate the setup cost, There are two costs - prototype development cost and hardware cost. It takes three months to analyze, design, develop, and test the prototype, CDE Kid. The programmer salary is 15,000 baht so the prototype development cost is 45,000 .

CDE Kid needs one stand alone PC which costs 25,000 baht. Before using CDE Kid for each session, CDE Kid needs to be setup by a programmer in about 1 day. So, the annually setup cost is equal to 1,154 baht for the first year.

Table 6.4 displays the Cash Flow for CDE Kid. For 746 applications use 95.88 man-hours, so it is equal to 0.1285 man-hour/application.

Table 6.4. Cash Flow for CDE Kid.

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5
Initial Cost						
Hardware	25,000					
Development Cost	45,000					
Total Initial Cost	70,000					
Annual Cost						
No. of Applications		2000	2200	2420	2662	2928
Man-hours		257	282.7	310.97	342.067	376.248
Est. baht/man-hour		51.28	53.84	56.53	59.36	62.33
Operate Cost (Baht)		13,179	15,221	17,579	20,305	23,452
Setup Cost (Baht)		1,154	1,212	1,272	1,336	1,403
Total Annual Cost		14,333	16,433	18,851	21,641	24,855
Cash Flow (Baht)	- 70,000	- 14,333	- 16,433	- 18,851	- 21,641	- 24,855

To replace the manual system with CDE Kid, we will calculate the cash flow of the replacement by the following equation.

$$\text{CASH FLOW}_{\text{Replacement}} - \text{CASH FLOW}_{\text{new system}} = \text{CASH FLOW}_{\text{existing system}}$$

Hence, we can calculate the cash flow of the replacement shown in Table 6.5.

Table 6.5. Cash Flow of the Replacement Analysis.

	Year 0	Year 1	Year 2	Year 3	Year 4	Year 5
CF of new system	-70,000	-14,333	-16,433	-18,851	-21,641	-24,855
CF of manual system		-65,300	-75,416	-87,102	-100,609	-116,199
Cash Flow (Baht)	-70,000	50,967	58,983	68,251	78,968	91,344
Acc. Cash Flow	-70,000	-19,033	39,950	108,201	187,169	278,513

From Table 6.5 the NPW of the replacement analysis is 278,513 baht. It shows that the CDE Kid development helps Progress Center save 278,513 baht within 5 years of the system life. So this is a good investment.

Moreover, the profit is 278,513 and the investment is 70,000. Therefore we are able to calculate Profitability Index (PI) by the following formula.

$$PI = \frac{\text{Profit}}{\text{Investment}}$$

$$= \frac{278,513}{70,000}$$

3.98

The PI shows that this project is able to generate 3.98 times profit of the investment.

VII. CONCLUSIONS AND RECOMMENDATIONS

7.1. Conclusions

After we have run a parallel testing with stand-alone computer, the result from the system is agreed by the management level. Information is managed easier with CDE Kid than the manual system so it saves processing time and efforts. Reliability and security of information is better than the existing procedure.

When data are exported from CDE Kid to CDE2000, all necessary data are manipulated and sent to CDE2000 database (data2000.mdb). CDE2000 can be processed without any problem especially exporting data from CDE2000 that is sent to Trinity College London by e-mail.

Incoming files from Trinity College London are imported to CDE2000 system and then we run CDE Kid to import data from CDE2000 respectively. After receiving data from CDE2000, we can further process the following steps until the last process without any error.

Examination Schedule function in CDE Kid is more flexible than CDE2000. More necessity reports, such as absentee report, ranking report, and etc., are able to developed to satisfy management requirement.

After the parallel testing is finished, we analyze the feasibility of system implementation and find that.

- (1) The man-hour (human effort) is reduced by 379.12 hours or around 80%.
- (2) Within 5 years usage, we can save processing cost around 278,513 bahts.
- (3) Profitability index is 3.98

We can conclude that CDE Kid can be used as a stand alone system with efficiency and effectiveness.

7.2. Recommendations

CDE Kid is developed by using MS-Access97 that has restrict capabilities for a huge of data. Reliability, stability, and capability of the system developed by MS-Access97 decreases if there are more than 5 clients connected at the same time.

Moreover, data entry must input the candidate information from the application form. Within this process, the error possibility is high because there are a lot of data. It is better if teachers or candidates input examination information to the system by themselves.

Teachers and candidates usually inquire information such as TCL number, examination schedule, examination result and etc. by telephone. We may add more inquiry's channel especially Intranet/Internet for them so that they can search information by themselves.

After we analyze the CDE Kid system, we find that CDE Kid may be modified and added more function in order to increase capability and decrease human errors by:

- (1) Migrating front-end application from MS-Access 97 to MS-Access XP or Visual Basic 6.0. Because CDE Kid is developed based on VBA (Visual Basic for Application), that is subset of Visual Basic, so it's easier to migrate to MS-Access XP and Visual Basic.
- (2) Developing registry and inquiry system for teachers and candidates, Using the Internet is recommended because it is easier for them than desktop applications with the remote access. We can develop a web site by using ASP or ASP.net technology that is based on Visual Basic language.
- (3) Changing database engine from Microsoft Jet (MS-Access 97) to MSDE (Microsoft Desktop Engine), based on MS-SQL Server 2000, for CDE Kid because SQL Server is more powerful and reliability than Microsoft Jet 4.0

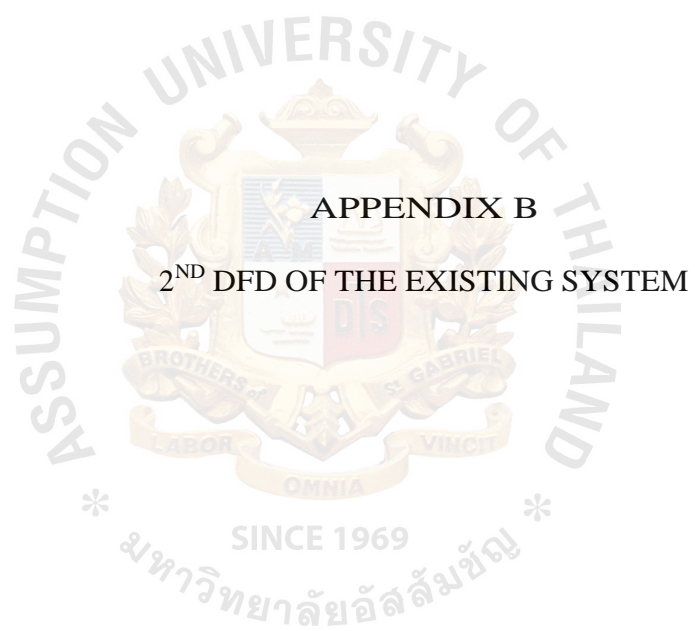
and MSDE is compatibility to MS-Access XP. For the Internet web site, we can use both MS-Access as database or MS-SQL Server depending on our ISP.





INDEX

	<u>Page</u>
Assigned Exam Date	39
Candidate Data Entry 2000 System (CDE2000)	2, 30
CDE Kid	39
Context Model	20, 29
DATA FLOW DIAGRAM, DFD	18
DAO	57
DFDs	20
Entity Relationship Diagram (ERD)	19
Fundamental System Model	20, 29
Graphics User Interface(GUI)	53
Parallel System	58
Process Center Examination System	29
Process Specification (PSPEC)	18
Prototype Program	7-10, 12-16
PSPEC	18
SQL command	35
TCL	1
TCL Number	30
The Functional Modeling And Information Flow	29
The Linked Table method	50
Trinity College London	1



2ND DFD OF THE EXISTING SYSTEM

B.1 The 2nd DFD of Process 2

Application Form

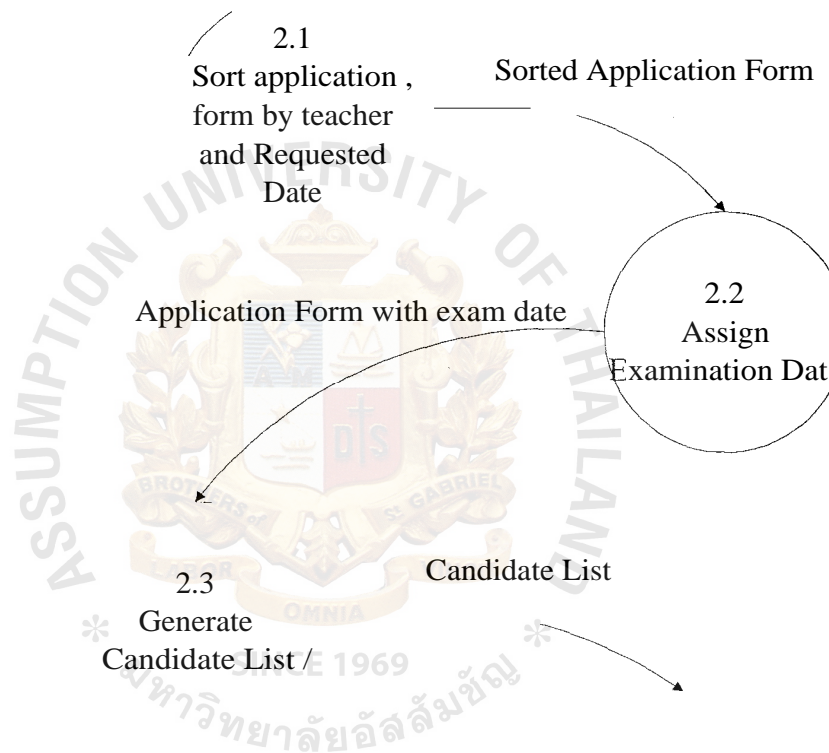


Figure B.1. 2nd DFD of Process 2.

Application form sent by process 1 is sorted by Teacher ID and Requested Date in process 2.1. Then, the examination date is assigned for each candidate in process 2.2. The candidate who has the same teacher should take the examination on the same day. Next, within process 2.3 candidate list is generated.

B.2 The 2nd DFD of process 7

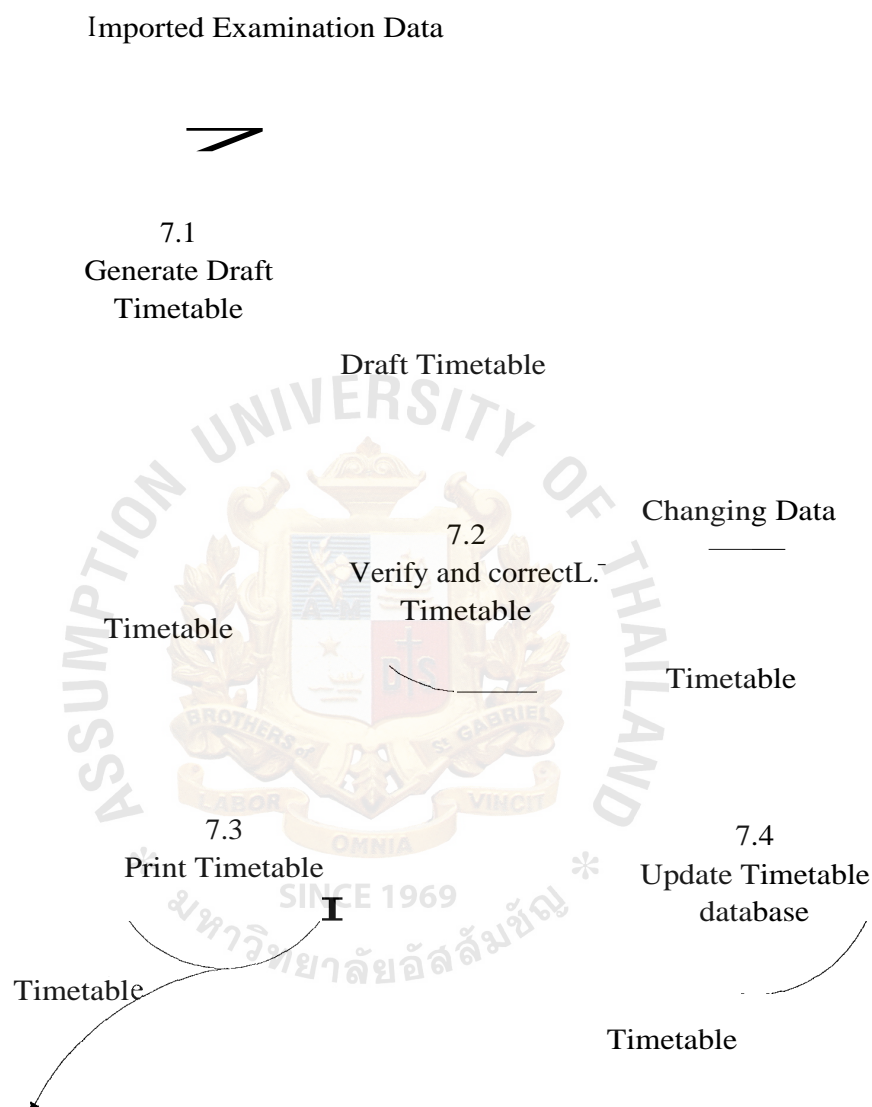


Figure B.2. 2nd DFD of Process 7.

When the examination data are imported to Data2000.mdb, Progress Center will generate a draft timetable. Next, a draft timetable is verified and corrected (inserted,

deleted, updated, and changed) depending on the changing data from management.

Corrected timetable is updated to database and printed in the following process.

B.3 The 2nd DFD of Process 10

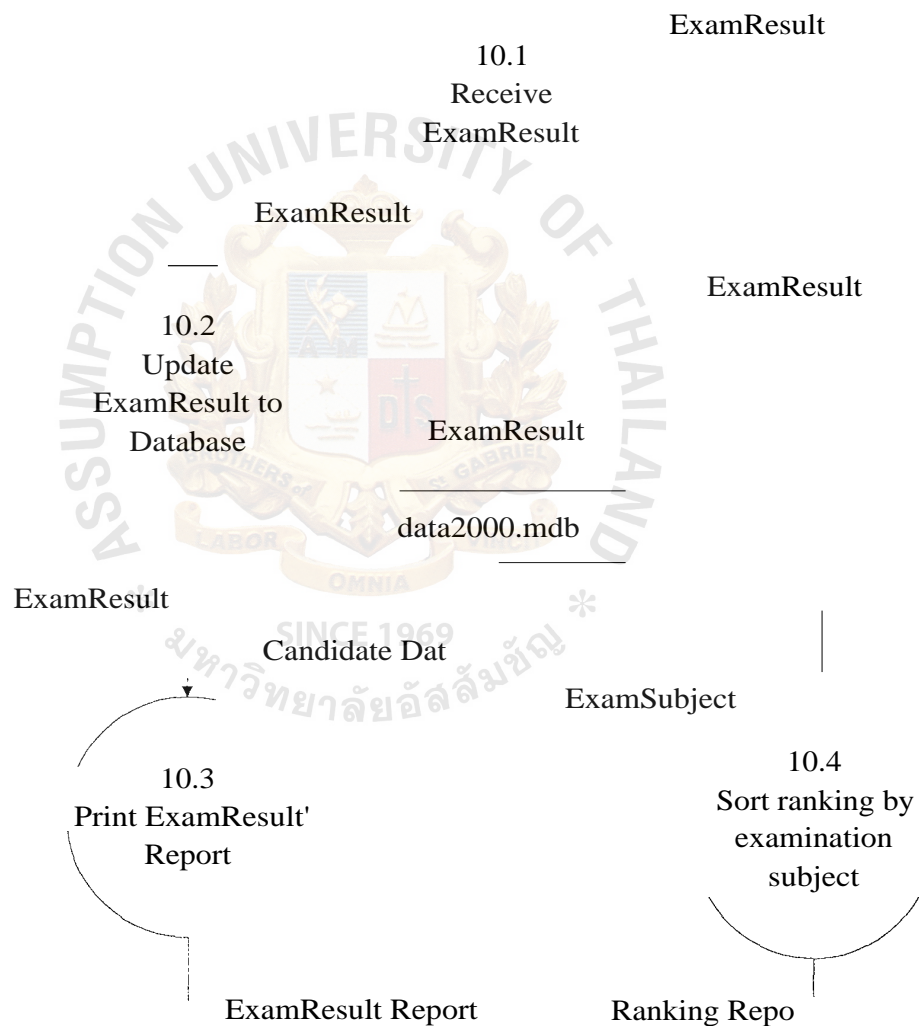


Figure B.3. 2nd DFD of Process 10.

After receiving the examination result from TCL, it will be updated to database. In process 10.3, candidate information is retrieved from in order to generate and print ExamResult report. Moreover, ExamResult data are sorted by examination subject in process 10.4. Then, ranking report is generated and printed out.





APPENDIX C

DATA DICTIONARY FOR LINKED TABLE

DATA DICTIONARY FOR LINKED TABLE

Table C.1. CandidateRegistration Table.

No.	Name
Sequence (P.K.)	
Full candidate name (name + " " + surname)	
Candidate surname	
Candidate's initial (M)	
Unique ID assigned to candidate (exported to CDE Kid as BackTCL)	
Date of birth	
Candidate's Gender	
Examination Center ID (Thailand center ID equals to 0845)	
Modification date	
Date of file exported to Trinity College London	
Date of file from Trinity College London imported to CDE2000	

Table 2. GradeExam Table.

Index	Field Name	Description	Data Type
1.	Sequence (P.K.)	Automatically sequence number	Integer
2.	CandidateSequence (F.K.)	Sequence number of candidate	Integer
3.	CandidateID	Unique ID assigned by CDE2000 system	Integer
4.	ExamID	Examination ID for candidate (mapped to BackTCL in CDE Kid)	Integer
5.	UniqueGradeExamID	Unique ID assigned by TCL	Integer
6.	Month Year	Examination session	String
7.	ExamTime	Examination time	String
8.	SubjectID (F.K.)	Examination subject id	Integer
9.	TeacherID	Candidate's teacher id assigned by examination center	Integer
10.	FullTeacherID (F.K.)	Full candidate's teacher id (CenterID + TeacherID)	String
11.	TrinityTeacherID	Candidate's teacher id assigned by TCL	Integer
12.	SchoolID	Candidate's school id assigned by examination center	Integer
13.	FullSchoolID (F.K.)	Full candidate's school id (CenterID + SchoolID)	String

26. 25.

Head of the Examination Center

N Z a)

TrinitySchoolID	Candidate's School id assigned by TCL	
15. VenueID	Examination venue id assigned by examination center	
16. FullVenueID	Full examination venue id (CenterID + VenueID)	
17. TrinityVenueID	Examination venue id assigned by TCL	
18. CenterID	Examination center id (Thailand, Progress Center, is 0845)	
19. Result1	1 st Examination Result (Between 0 and 100)	00
20. Result2	2 nd Examination Result (Between 0 and 100)	a)
21. Result3	3 rd Examination Result (Between 0 and 100)	a)
22. Result4	4 th Examination Result (Between 0 and 100)	24
23. Result5	5 th Examination Result (Between 0 and 100)	24
24. Result6	6 th Examination Result (Between 0 and 100)	24
25. Result7	7 th Examination Result (Between 0 and 100)	24
26. Result8	8 th Examination Result (Between 0 and 100)	24

GradeExam Table. (Continued)

Z		a)	
Z		a)	
Result9	9	Examination Result (Between 0 and 100)	
Result10	10	Examination Result (Between 0 and 100)	
OverallResult		Overall examination result	
Id	Id		
Mo	Mo		
Exp	Exp	Exported date to TCL	
Imp	Imp	Imported date from TCL	
34. ModifiedBy		User ID who modify record	
		Verify if this record is canceled by TCL or not	

No		a)	
SubjectID (P	Examination subject id	kr)	
Subj ectArea	Examination subject area (ESOL, Music, Speech)	ayk a)	
	Examination subject title		
Grade	Examination subject grade	a)	
SubjectCode	Identify year of subject issued	a)	
Description	Description of examination subject	aJ	
GradeDescription	Description of examination subject in each grade	(L)	
PrintDescription	Description of examination subject printed on certificate	.	
DuetEnsemble	Identify that this subject is duet ensemble	aC)	
PracticalOrWritten	Identify that this subject is practical or written	H	
LengthOfExam	Examination time	! :	a)
12. PassMark	Pass mark	CU	a)
13. MeritMark	Merit mark	I)	

GradeSubject Table. (Continued)

DistinctionMark	Distinction mark
MaximumSectionalMark 1	Maximum mark for Result1 in GradeExam
MaximumSectionalMark2	Maximum mark for Result2 in GradeExam
MaximumSectionalMark3	Maximum mark for Result3 in GradeExam
MaximumSectionalMark4	Maximum mark for Result4 in GradeExam
MaximumSectionalMark5	Maximum mark for Result5 in GradeExam
MaximumSectionalMark6	Maximum mark for Result6 in GradeExam
MaximumSectionalMark7	Maximum mark for Result7 in GradeExam
MaximumSectionalMark8	Maximum mark for Result8 in GradeExam
MaximumSectionalMark9	
MaximumSectionalMark 10	
26. AMPM	Examination period (A.M. or P.M.)

GradeSubject Table. (Continued)

Z1		
	Identify that this record is canceled by TCL or not	



Table 2 C.4. School Table.

No.	Name	Description	Data Type
1.	FullSchoolID (P.K.)	Full candidate's school id	Integer (8)
2.	SchoolID	Candidate's school id assigned by examination center	Integer(4)
3.	hoolID	Candidate's school id assigned by TCL	Integer(4)
4.	ExamID	Examination center id	Integer(4)
5.	School Name	School name	Text(60)
6.	ContSalutation	Salutation of contact person	Text(10)
7.	ContForeNames	Contact person's name	Text(10)
8.	ContSurname	Contact person's surname	Text(10)
9.	ContFullName	Contact person's fullname	Text(20)
10.	Address1	School's address	Text(20)
11.	Address2	School's address continue	Text(20)
12.	Address3	School's address continue	Text(20)
13.	Town	School town	Text(20)

School Table. (Continued)

No. Name	
1	Country
2	Country
3	Country ID
4	School's telephone number
5	School's fax number
6	School's email
7	Modification Date
8	Exported date to TCL
9	Imported date from TCL
10	User ID who modify record
11	Verify if this record is canceled by TCL or not

Table 1 Teacher Table.

Teacher ID	Teacher Name	Teacher Address
1	Mr. A. B.	123/456 Moo 1, Bangkok 10110
2	Mr. C. D.	789/1011 Moo 2, Chiang Mai 50110
3	Mr. E. F.	567/890 Moo 3, Phuket 83110
4	Mr. G. H.	345/678 Moo 4, Nakhon Si Thammarat 91110
5	Mr. I. J.	234/567 Moo 5, Surat Thani 84110
6	Mr. K. L.	123/456 Moo 6, Krabi 81110
7	Mr. M. N.	987/654 Moo 7, Ranong 85110
8	Mr. O. P.	876/543 Moo 8, Trang 92110
9	Mr. Q. R.	765/432 Moo 9, Yala 95110
10	Mr. S. T.	654/321 Moo 10, Pattani 96110
11	Mr. U. V.	543/210 Moo 11, Songkhro 97110
12	Mr. W. X.	432/109 Moo 12, Narathiwat 99110
13	Mr. Y. Z.	321/098 Moo 13, Aceh 85110
14	Mr. AA. BB.	210/987 Moo 14, Sumatra 85110
15	Mr. CC. DD.	109/876 Moo 15, Java 85110
16	Mr. EE. FF.	098/765 Moo 16, Kalimantan 85110
17	Mr. GG. HH.	987/654 Moo 17, Sulawesi 85110
18	Mr. II. JJ.	876/543 Moo 18, Maluku 85110
19	Mr. KK. LL.	765/432 Moo 19, Papua 85110
20	Mr. MM. NN.	654/321 Moo 20, Irian Jaya 85110

Teacher Table. (Continued)

No. Name		OO	
CO	CO	CO	CO
CO	CO	CO	CO
CO	CO	CO	CO
HomeTel	Teacher's home telephone number		
WorkTel	Teacher's workplace telephone number		
19. Fax	School's fax number		
20. Email	School's email		
21. Qualifications	Teacher's qualification		
22. Modified	Modification Date		
23. Exported	Exported date to TCL		
24. Imported	Imported date from TCL		
25. ModifiedBy	User ID who modify record		
26. Deleted	Verify if this record is can be deleted		

CandidateRegistration	
-----------------------	--

GradeSubject	
SubjectID (P.K.)	

Teacher	
TeacherID (P	

o ∞	
FullSchoolIID (P.	



Database Structure of Linked Table.



APPENDIX D

INPUT AND OUTPUT INTERFACE

INPUT AND OUTPUT INTERFACE

D.1 MAIN INPUT INTERFACE

D.1.1 Application Screen (include menu bar)

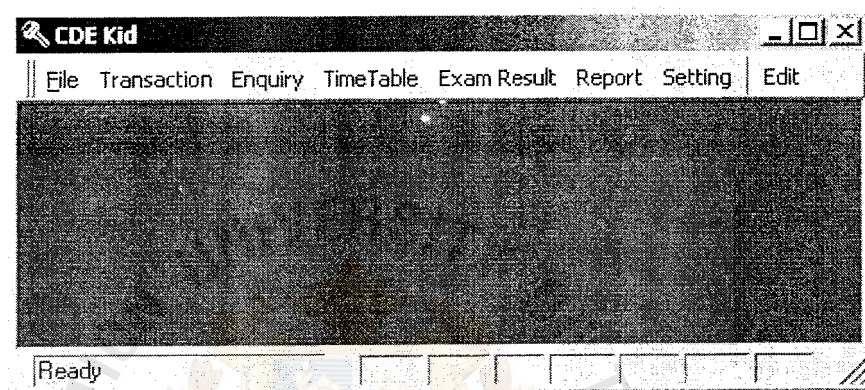


Figure D.1. Application Screen.

D.1.2. Main Input Screen

A screenshot of a "Candidate" form. The form contains the following fields and values:

- Name: Srindhorn
- Surname: Imudom
- Gender: ☒ Male ☐ Female
- Birthday: 25/01/1963

At the bottom of the form, there is a record navigation bar that says "Record: 14" followed by navigation icons and "1" and "of 12".

Figure D.2. Candidate Form.

Application form

NO 133 AppNo: 1001 Application Date 16 Apr 2002

Teacher: IBancha Chanaapt
108450004

Requested Date: 03 Jun 2002 ☒ morning ☐ Afternoon

Candidate Name: Chavaied Maskuhat
Date of Pith: 1 709/19/33
Gender: ☒ Male ☐ Female
New Candidate

amination Type
Practice Theory Subject: Grade: 01 ☒ 05

Gerrit ate Information
School: iBangkok Christian College
Teacher Name: Banda Chanavut
Teacher's Qualification

Edit I 11

Figure D.3. Application Form.

Reschedule

Exam Date: 03/06/2002 Room: 1

Start	Finish	TCL No	Candidate Name	Subject Tree	Grade	Teacherttaree	RequestedD
9.12	9.24	8450004	Pajaval Suwanaphan	Solo Piano	03	Eakkaraj Kass nthorn	04105/2002
9.24	9.44		Break				
9.44	10.00	8450003	Santa Riensavapak	Solo Piano	05	Chan intorn Samakabutra	02105/2002

Record: 14 of 3

EDIT INSERT BREAK UP DOWN CANCEL DELETE MOVE TO SNAP

Figure D.4. Reschedule Form (Timetable Management).

X

Find by Name : Da Suwansang Janjar(21

I BackTCL:	8450001	Result1:	20
		Result2:-	20
CandidateName:	Dao Swivansang Janjaroen	P...sult3:	20
Subjecifitle:	Solo Piano	ResuR4:	15
			10
Grade:	03	Result6:	2 0
		Resu ..	0
TeacherName:	Bancha Chanaviiiit	F...esultor	0
		ResUltg:	<< .0
		R...esultl0;	.0
			55

EDIT

Figure D.5. Exam Score form(xam Result Entry Form).

D.2 MAIN OUTPUT INTERFACE

D.2.1. Output Screen (Inquiry Form)

Summary by Teacher

Teacher ID: 11182150601	Name: 11Fal,...araj Kasinthoin
Requested Code: 1 04/08/20(c	idtl: ss 111 asdvok.d
No of r2-andidare	jjiktisdf
Tot."1Exal	Bangkok 10111
Work Phone	Home Phone

Entry No	Assigned No	TCL No	Name	LxamDate	Subject	Grade
1	4	18.150004	Hap	105/Hh:2002		

Navigation: [Previous] [Next] [Home] [Print]

Figure D.6. Summary of Teacher Inquiry Form.

Summary by Requested Date				
Req Date	ID	Name	No of Can	Exam Time
13/06/20021	0004	B an cha Chanavijit	2	23
03/0612002	0002	Chan i Worn Sam akab utra	1	16
0410bIN02	0001	Eakkaraj Kas i nth o rn	1	12
Total			4	51

Figure D.7. Summary of Requested Date Inquiry Form.

D.2.2 Output Report

<i>TCL Examination Schedule</i>						
Date	03/06/2002		Room	1		
<i>Shut</i>	<i>Finish</i>	<i>TCL So</i>	<i>Candi,IntelVoine</i>	<i>reacherNome</i>	<i>Sn hject Title</i>	<i>Grade</i>
9.12	- 9.24	3450004	Pajavat Suwanaphan	Eakkaraj K.asinthom	Solo Piano	03
9.44	- 10.00	0450003	Kanta Riensavapak	Chanintorn Samakabutra	Solo Piano	05
Date	05/06/2002		Room	1		
<i>Simi</i>	<i>Finish</i>	<i>TCL Vi</i>	<i>CandidateNinne</i>	<i>IcacherNain.-</i>	<i>Sub:in-trifle</i>	<i>Grade</i>
9.00	- 9.12	8450001	Dao Suwansang Janjaroen	Bancha Chanavijit	Solo Piano	03
10.00	- 10.11	8450002	Chavajed Maskulrat	Bancha Chanavijit	Solo Piano	01

Figure D.B. Examination Schedule.

Exam Schedule By Teacher

11-Tri'01914: 03450001 ^d 91B : Eakkaraj Ka sinthorn ^d ที่อยู๋ : 111 as dvokd ijjkij s df Bangkok 10111						
171.11.Wi (14111): 12 ^a จ้11.vnirsemiTivam: 1						
Examination Date	Room	Time	TCL No	Candidate Name	Subject	Grade
03106/2002	1	9.12 - 9.24	0450004	Pajavat Stavanaphan	Solo Piano	03

Figure D.9. Examination Summary for Teacher.

Solo Piano		03		
1	8450004 Pajavat Suwanaphan	Eakkaraj Kasinthorn	Solo Piano 03	90
2	8450001 Dao Suwansang Janjaroen	Bancha Chanavijit	Solo Piano 03	85

[illegible]

19.14 41 CERTIFICATE

cdha

Eakkaraj Kasinthorn (08450001)

111 asaVokd

ijjkfjsdf

Bangkok 10111

1a`' lqini Eakkaraj Ka s inth orn I4

fln'l 411 Jil viSii",,];••ii101,f,clDnna•14.1 71108`1Vinqt;11.01...ralii` lirilliJitDNRYi89N1In141111101).6Trinity

iilecii,60; 118.001:1² LINFOriirigLi i 'il 64-111,E7aUvi i Len•M N TY 1m,t^{n±-}

Tun-iffyvonmmnulLuirilemh*I6q11,-kriN.4l-14vfo \16Irxan -ralun-wira;ii

41 uu:4.1-11111rill

Figure D.12. Certificate Appointment Slip.





APPENDIX E
TESTING INPUT AND OUTPUT

APPENDIX E. TESTING INPUT AND OUTPUT

E.1 Testing input

Table E.1. Testing Candidate Information Table.

Candidate	Teacher	Subject	Level	P/W	Date
Nantarat Juabsamai	Bancha Chanavijit	Piano	7	P	15/08/2002
Tai Boontae	Bancha Chanavijit	Piano	6	P	15/08/2002
Chai Boontae	Bancha Chanavijit	Singing	7	P	15/08/2002
Anake U-kate	Bancha Chanavijit	Piano	4	P	15/08/2002
Panwilai Meesub	Bancha Chanavijit	Piano	5	P	16/08/2002
Boonchai Dangmak	Bancha Chanavijit	Theory	5	W	16/08/2002
Areeya Urode	Eakkaraj Kasinthorn	Piano	7	P	17/08/2002
Ancharee Rakthai	Eakkaraj Kasinthorn	Piano	7	P	17/08/2002
Oraphan Rakthai	Eakkaraj Kasinthorn	Singing	7	P	17/08/2002
Nithad Rakthai	Eakkaraj Kasinthorn	Theory	5	W	16/08/2002
Chakara Urode	Eakkaraj Kasinthorn	Theory	5	W	16/08/2002
Jennifer Tia	Eakkaraj Kasinthorn	Piano	4	P	18/08/2002
Marisa Tia	Eakkaraj Kasinthorn	Piano	5	P	18/08/2002
Surang Bawornpong	Eakkaraj Kasinthorn	Theory	6	W	16/08/2002
Mali Homthong	Jatupoom Komeluecha	Singing	5	P	17/08/2002
Eddy Lee	Jatupoom Komeluecha	Theory	5	W	16/08/2002
Jason Lee	Jatupoom Komeluecha	Theory	5	W	17/08/2002
Wilai Hongthong	Jatupoom Komeluecha	Piano	6	P	17/08/2002

Table E.I. Testing Candidate Information Table. (Continued)

Candidate	Teacher	Subject	Level	P/W	Date
Sunisa Jet	Jatupoom Komeluecha	Singing	6	P	18/08/2002
Manop Daengdee	Jatupoom Komeluecha	Piano	7	P	17/08/2002

E.2 Changing Requested Date

After Examination dates are assigned by Progress Center, 5 candidates would like to change examination date.

Table E.2. Changing Requested Date Request Table.

Candidate	Teacher	Assigned Date	Change Request
Panwilai Meesub	Bancha Chanavijit	16/08/2002	15/08/2002
Jennifer Tia	Eakkaraj Kasinthorn	18/08/2002	17/08/2002
Marisa Tia	Eakkaraj Kasinthorn	18/08/2002	17/08/2002
Jason Lee	Jatupoom Komeluecha	17/08/2002	16/08/2002
Sunisa Jet	Jatupoom Komeluecha	18/08/2002	17/08/2002

E.3 Timetable Changing Request

After Timetable is generated, there are 3 changing requests

1. Nantarat Juabsamai, Piano level 7, requests to change examination sequence to the last sequence on 15/08/2002.
2. Mali Homthong, Singing level 5, requests to change examination date from 17/08/2002 to 15/08/2002.

3. Tai Boontae, piano level 6, requests to swap his examination date to Ancharee Rakthai, piano level 7.

E.4 Examination Result

Table E.3. Examination Result Table.

Candidate	Subject	Level	1st	2nd	3rd	4 th	5 th	Total
Nantarat Juabsamai	Piano	7	15	16	18	20	18	87
Tai Boontae	Piano	6	17	16	17	18	17	85
Chai Boontae	Singing	7	24	21	19	19	-	83
Anake U-kate	Piano	4	18	18	19	17	20	92
Panwilai Meesub	Piano	5	17	19	20	17	16	89
Boonchai Dangmak	Theory	5	24	23	25	21	-	93
Areeya Urode	Piano	7	20	18	15	15	19	87
Ancharee Rakthai	Piano	7	18	16	18	17	15	84
Oraphan Rakthai	Singing	7	0	0	0	0	0	0
Nithad Rakthai	Theory	5	19	19	13	15	-	66
Chakara Urode	Theory	5	24	19	21	18	-	82
Jennifer Tia	Piano	4	0	0	0	0	0	0
Marisa Tia	Piano	5	19	17	16	18	17	87
Surang Bawornpong	Theory	6	21	21	24	19	-	85
Mali Homthong	Singing	5	21	23	18	18	-	80
Eddy Lee	Theory	5	19	18	22	21	-	80
Jason Lee	Theory	5	18	16	20	20	-	74
Wilai Hongthong	Piano	6	15	14	12	15	16	72

Table E.3. Examination Result Table. (Continued)

Candidate	Subject	Level	1st	Ind	3rd	4th	5th	Total
Sunisa Jet	Singing	6	16	14	17	16	-	63
Manop Daengdee	Piano	7	0	0	0	0	0	0



BIBLIOGRAPHY

- Brigham, Eugene F., Gapenski, Louis C., and Ehrhardt, Michael C. Financial Management. Florida: The Dryden Press, 1999.
2. Kendall, Kenneth E., and Kendall, Julie E. System Analysis and Design. New Jersey: Prentice-Hall, 1998.
 3. Pressman, Roger S. Software Engineering. Singapore: McGraw-Hill, 1997.



St. Gabriel's Library, Al

