Information Systems Department
Faculty of Computer and Information Sciences
Mansoura University

# Evaluation of Service Oriented Architecture in e-Learning

**THESIS**
Submitted in Partial Fulfillment of the Requirements for the Degree
of Master of Science in Information Sciences

**TO**
Department of Information Systems, Faculty of Computer
and Information Sciences, Mansoura University

**BY**
**Haitham Abdel Monem El-Ghareeb**
B.Sc., Information Systems Department
Faculty of Computer and Information Sciences,
Mansoura University

**SUPERVISED BY**

Prof. Dr. **Alaa El-Din Mohamed Riad**
Head of Information Systems Department
Faculty of Computer and Information Sciences
Mansoura University

Dr. **Ahmed El-Said Hassan**
Department of Electrical Engineering
Faculty of Engineering
Mansoura University

Mansoura – 2008

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَـٰنِ ٱلرَّحِيمِ

ٱقۡرَأۡ بِٱسۡمِ رَبِّكَ ٱلَّذِى خَلَقَ (١) خَلَقَ
ٱلۡإِنسَـٰنَ مِنۡ عَلَقٍ (٢) ٱقۡرَأۡ وَرَبُّكَ ٱلۡأَكۡرَمُ
(٣) ٱلَّذِى عَلَّمَ بِٱلۡقَلَمِ (٤) عَلَّمَ ٱلۡإِنسَـٰنَ مَا لَمۡ
يَعۡلَمۡ (٥)

Thesis Title

# Evaluation of Service Oriented Architecture in e-Learning

Name of the Researcher

Haitham Abdel Monem El-Ghareeb

Referees and Judging Committee

| # | Name | Occupation | Signature |
|---|---|---|---|
| 1 | Prof. Dr. Wael Fathi Abdel Wahed | Vice Dean for High Studies<br>Faculty of Computer and Information Sciences<br>Menofia University | |
| 2 | Ass. Prof. Dr. Taher Tawfek Hamza | Department of Computer Science<br>Faculty of Computer and Information Sciences<br>Mansoura University | |
| 3 | Prof. Dr. Alaa El-Din Mohamed Riad | Head of Information Systems Dept.<br>Faculty of Computer and Information Sciences<br>Mansoura University | |

Department Chair  Vice Dean for Higher Studies  Dean

Thesis Title

# Evaluation of Service Oriented Architecture in e-Learning

Name of the Researcher

Haitham Abdel Monem El-Ghareeb

Supervising Committee

| # | Name | Occupation | Signature |
|---|---|---|---|
| 1 | Prof. Dr. Alaa El-Din Mohamed Riad | Head of Information Systems Dept. Faculty of Computer and Information Sciences Mansoura University | |
| 2 | Dr. Ahmed El-Said Hassan | Department of Electrical Engineering Faculty of Engineering Mansoura University | |

Department Chair  Vice Dean for Higher Studies  Dean

جامعة المنصورة
كلية الحاسبات و المعلومات
قسم نظم المعلومات

# تقرير صلاحية

موضـوع الرســالة: **تقييم أسلوب الخدمة الموجه في التعليم الالكتروني**
**"Evaluation of Service Oriented Architecture in e-Learning"**

للبـــــاحـــــــــث: **هيثـم عبدالمنعـم الغـريب**
معيد بقسم نظم المعلومات
كلية الحاسبات و المعلومات
جامعة المنصورة

تعالج الرسالة هذا الموضوع في ست فصول أساسية تنقسم كالتالي:

الفصل الأول: يستعرض الباحث المشكلات الحالية التي تقف عائقا امام تحقيق التعليم الالكتروني الأهداف المرجوة من خلال استعراض خصائص نظم ادارة الجامعات و نظم ادارة التعليم و تحديد الحلقة المفقودة بينهما.

الفصل الثاني: يتعرض البحث للهيكلية خدمية التوجه بوجه عام موضحا أبرز خصائصها و مميزاتها و نقاط القوة التي يجب ابرازها عند استخدامها لتصميم النظم.

الفصل الثالث: يقدم هذا الفصل الهيكل خدمي التوجه المقترح من الباحث لتطبيقه على النظم الحالية لادارة الجامعات متضمنا تفاصيل نظام ادارة شؤون الطلاب و نظام ادارة المكتبات.

الفصل الرابع: يقدم هذا الفصل ثلاث هياكل خدمية التوجه لثلاث مكونات أساسية من نظام ادارة التعليم مقسمة على ثلاث أجزاء:

الجزء الأول: يوضح الباحث كيفية الاستفادة من الهيكلية خدمية التوجه في المكتبات الرقمية

الجزء الثاني: يقدم الباحث نظام ادارة المواد التعليمية باستخدام الهيكلية خدمية التوجه

الجزء الثالث: يقدم الباحث نظام ادارة التقييم باستخدام الهيكلية خدمية التوجه

الفصل الخامس: يقوم الباحث بتقييم الهياكل خدمية التوجه التي تم اقتراحها في الفصول السابقة.

الفصل السادس: يقدم خلاصة البحث موضحا أبرز المزايا التي تم تحقيقها من خلال الهيكلية المقترحة والنظرة المستقبلية للبحث.

و قد تم نشر بحثين:

1. **A. M. Riad, and H. A. El-Ghareeb**, "New Architecture for Mobile News Agent System", Egyptian Informatics Journal, Cairo University, 2007, Vol. 8, Issue 1.

2. **A. M. Riad, and H. A. El-Ghareeb,** "A Service Oriented Architecture to Integrate Web services and Software Agents in Course Management Systems", Egyptian Informatic Journal, Cairo Unviersity, 2007, Vol. 8, Issue 1.

## التوصيـــة:

الرسـالة صالحة للمنـاقـشة وتؤهل الباحـث للحصول على درجة الماجستير.

المشرف الرئيسي
الأستاذ الدكتور / علاء الدين محمد رياض

رئيس قسم نظم المعلومات
كلية الحاسبات و المعلومات
جامعة المنصورة

المشرف
الدكتور / أحمد السيد حسن

قسم الهندسة الكهربية
كلية الهندسة
جامعة المنصورة

*To my Mother, the one who taught me to be frank, bold, simple, honest, pure, and always believe in God*

*The one with me when I am happy or sad, support me in times good or bad*

*The fragrance of love, who sought me to make the effort, and do not wait for pay back*

*The one with the bright smile as sun, can't thank you enough for what you have done*

# Acknowledgments

First and foremost, I thank **ALLAH** for giving me incentive to go on and finish this work and giving me glimpses of heavenly hope in the darkest times.

I would like to thank my supervisor, **Professor Alaa El-Din Mohamed Riad**, head of the information systems department, for his unlimited support and encouragement, for providing me with valuable insights and encouraging me to always do better, for keeping my thesis on track and always giving me great and valuable ideas to enhance my work.

I would like to thank my supervisor, **Dr.Ahmed El-Said Hassan** for the insightful discussions, and ideas, for taking long times to discuss smallest details.

I would like to thank **Dr.Hamy El-Minir**, head of communications department of the Misr High Institute of Engineering for his support and motivation from the early stages of this work.

Last but not least, I would like to thank people who are not waiting for my thanks, who are keeping me up and pushing me forward, who are wishing me the best, praying for me, sacrificing everything they have, and giving everything they can just to see me where I am right now, **my family,** and **my friends**.

# Table of Contents

## Chapter One: Introduction



## Chapter Two: Service Oriented Architecture

# List of Figures

# List of Tables

# ABSTRACT

E-Learning is one of most interesting topics in recent years. E-Learning system is not an ordinary system because it requires the integration of many technologies to be adaptive and effective more than other systems do. While superficially e-Learning refers to the usage of Information and Communication Technologies (ICT) in the learning process, e-Learning still means much deeper than ICT in the learning process. E-Learning can not be achieved unless University Management Information Systems (UMIS) and Learning Management Systems (LMS) work together. UMIS includes: Faculty Information System, Student Affairs Management System or Student Information System (SIS), and Library Information System (LIS). LMS includes: Course Management System (CMS), Assessment Management System (AMS), and Digital Library.

Software integration has been widely known as a challenge for system architects. Software architecture is a key factor for system success. Choosing the software architecture to implement is affected mainly by the functional and nonfunctional system requirements. System integration is one of the main nonfunctional requirements that should be fulfilled in e-Learning. Though different software architectures can satisfy functional system requirements, nonfunctional system requirements remained a complex goal to achieve. Service Oriented Architecture (SOA) claimed to satisfy complex and sophisticated systems functional and nonfunctional requirements, especially via adopting Web services as the main SOA enabler. Though software agents can be used as SOA enablers, shortages have arose to prevent so. Mobile News Agent System (MNAS) is a mobile information agent based system proves that mobile information agent technology can not just be used in all information retrieval systems, because under some conditions, it might loose some of its advantages.

This thesis presents a Services based Architecture that applies to UMIS and LMS components and consists mainly of two layers: Interface layer, and Services layer. Interface layer interacts with system users via human

interface; that is portals, and with external organization services via machine interface; that is Web services. Services layer contains core system services and has three sub layers: Orchestration, Application Services, and Agents layer. Orchestration layer holds business logic presented by system processes as executable services. Business logic refers to different activities that can include Web services invocations, data manipulation, exception handling, and process termination. Application Services layer contains set of stateless Web services that are capable of performing certain tasks related to system entities.

This thesis highlights an evaluation framework that can be used to evaluate LMSs. Thesis evaluated the extent to which proposed SOA based architecture achieved in satisfying e-Learning functional and non-functional requirements. LMS Evaluation can include three different aspects: information system quality, pedagogical, and managerial characteristics.

Pedagogically, Thesis shows that the proposed SOA based architecture has helped e-Learning to achieve more than one goal. One of the critical limitations of a newly established educational institution is the lack of available well prepared courses. It is more applicable to use widely available courses that might be higher in quality than preparing new courses. Current LMS do not exploit courses shareability. This thesis proposed SOA based CMS as one of LMS components that addressed this shortage, automated discovering and importing of courses maintained and managed by external CMSs. Proposed CMS facilitates integration between different CMSs in order to share resources of educational institutions.

Mobile assessment is one of the M-Learning activities facilitated by proposed SOA based AMS. Mobile assessment refers to the capability of conducting assessments via mobile devices. Mobile assessment relies on external services that are not part of the AMS. Integrating different external systems and services to be virtually part of the educational institution AMS is one of integration challenges that was solved by the proposed SOA based AMS in this thesis.

The capability to integrate the different digital library contents and make it available to different LMS components is a clear example of the proposed

SOA based LMS capabilities to integrate different and standalone system components and make them available to each other. Proposed SOA based LMS in this thesis facilitated integration between software agents that play an important role in educational institutions and Web services. Also, integrating legacy systems and newly added systems is facilitated by the architecture presented in this thesis.

Future work includes addressing more and widely system processes that rely on the presented SOA based LMS and the implemented Web services to address agility in education institutions. Business Process Management Systems that rely on proposed SOA based LMS and UMIS is the next step to take.

# Chapter 1

# Chapter One

# INTRODUCTION

## 1. E-Learning Problem

E-Learning has been widely used to refer to computer based systems that not necessarily help main objectives of e-Learning. The main goal of introducing e-Learning is to revolutionize learning process; as technology has already revolutionized business. Utilizing modern technologies and new approaches to achieve brighter future learning that is not available today is the main goal of introducing e-Learning. Unfortunately, utilizing technology within learning process has not achieved objectives because requirements were not addressed correctly and clearly. Besides, the main learning process was neglected, only technological aspects were the main concern. In order to present effective e-Learning, requirements, current shortages, technological limitations should be addressed clearly and correctly to determine technological capabilities that satisfy requirements and overcome problems and shortages, and start walking in the right way towards effective future e-Learning. Learning should be leading technology towards effective future e-Learning, and technology should not be limiting it.

## 2. e-Learning

E-Learning is defined as the learning process created by interaction with digitally delivered content, services and support [1-3]. E-Learning involves intensive usage of Information and Communication Technology (ICT) to serve, facilitate, and revolutionize learning process. [4-8].

### 2.1 Learning Models

Figure 1.1 shows the three main learning models that are fully enabled by e-Learning [4-7]:

- **Traditional Learning** is the agreed on learning today learning, where students heads to a school/college to learn. Usage of ICT can enhance the learning process. Data show and power point slides usage; as an example; is a common implementation of e-Learning within traditional learning institutions [5].
- **Distance Learning** is the Educational situation in which the instructor and students are separated by time, location, or both. Education or training courses are delivered to remote locations via synchronous or asynchronous means of instruction [9]. Distance education does not preclude the use of the traditional classroom [10].

- **Blended Learning** is the combination of multiple models to learning [11]. It refers to learning models that combine traditional classroom practice with e-learning solutions. For example, students in a traditional class can be assigned both print-based and online materials [4].

Argues that claim each model's efficiency and effectiveness are widely available [9]. What matters the most is that 'e-Learning' was mentioned in the three models; unfortunately with a different perspective.

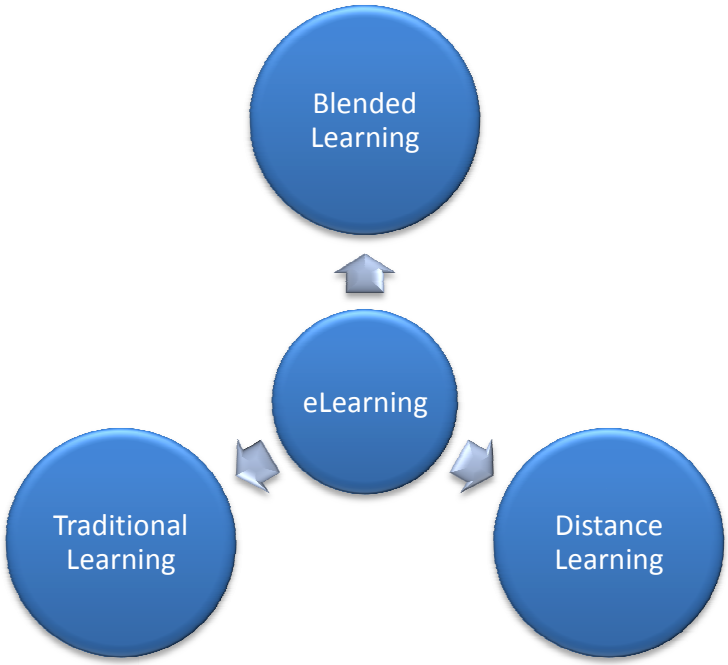


**Figure 1.1: Learning models**

## 3. Management Information Systems

E-Learning tends to revolutionize and manage the learning process [12], not only to manage universities. University Management Information Systems are not by itself the e-Learning.

### 3.1 University Management Information System

Managing universities activities requires University Management Information System (UMIS). UMIS refers broadly to a computer-based system 'collection of hardware, software, people, data, and information' that provides managers with the tools for organizing, evaluating and efficiently running their departments [13, 14].

Examples of UMIS components include Student Information System (SIS), Library Information System, Faculty Information System, and Finance System as illustrated in figure 1.2.

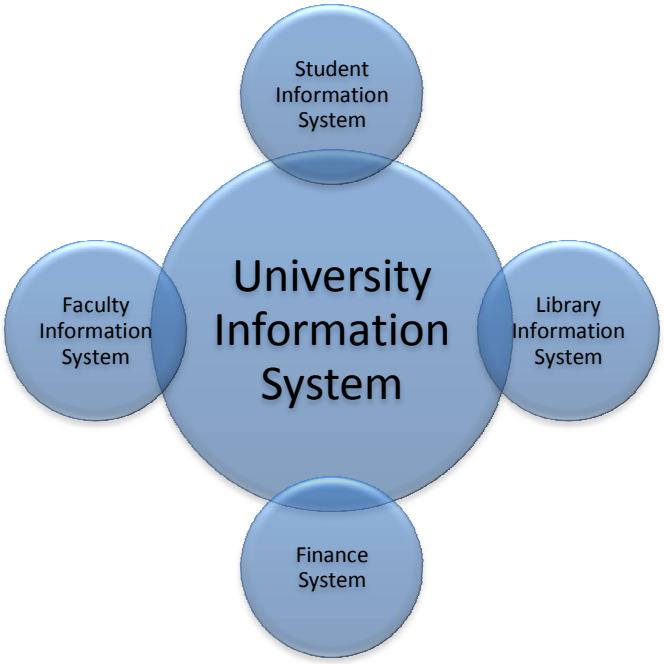


**Figure 1.2: A Prototypical University Management Information System**

### 3.1.1 Student Information System (SIS)

SIS is the information system responsible for managing students' data within the faculty and/or university. SIS typical student record includes ID, SSN (Social Security Number), Name, Age, Gender, Address (Street, City, Country), Email, Username, Password, DOB (Date Of Birth), Faculty, Year, Department [15].

SIS by itself is not an e-Learning system because personal data that SIS provides and manages differs in nature than data required for education [16]. Learner should be able to get a student profile that includes data like

- Detailed records of what learners have already learned (at the level of learning object, rather than a module or program).
- Profile of learning preferences.
- Development portfolio of transferable skills. A learning portfolio might also include a history of their interactions with their tutors, peers, and other significant learning conversations they may have had.

This kind of data is intended to be used to force learning process to be a learner oriented process[1] by adapting learning system to fit learner requirements, personal characteristics and capabilities. Unfortunately, SIS does not serve this purpose, and does not handle such data.

[1] Informative model is the model that forces one way of information transformation from instructor to learner, no matter what personal differences are, instead, learner oriented model modifies the system to fit each individual learner based on personal data, because people are different.

### 3.1.2 Library Information System

Library Information System is responsible for managing and automating libraries within faculties and/or universities. Automated Libraries are libraries that contain material in digitized form [17]. Automated Library Information System database record reflects the managerial tasks performed by librarians in order to effectively manage libraries. A typical Library Information System record will include Book ISBN, Name, Author(s), Keyword(s), and data like Section, List of all the books, List of books available, List of borrowed books, who is borrowing, when they should return, etc.

Automated Library Information System by itself is not e-Learning because library information systems do not serve the learning process. Learner should be able to access fully available digital libraries as part of the learning process.

### 3.1.3 Faculty Information System

Faculty Information System is responsible for managing and automating managerial activities related to Instructors, Employees, Courses, and intersection between them. A typical faculty information system database record includes Faculty data; ID, Name, Departments, Courses data; Course ID, Name, Description, Instructors data; ID, SSN (Social Security Number), Name, Age, Gender, Address (Street, City, Country), Email, Username, Password, DOB (Date Of Birth), Faculty, Year, Department; and Employees data; same as instructor's data with customized data about job [15].

Faculty Information System by itself is not e-Learning its main goal is to organize faculty and/or university managerial activities; the learning process is not the main orientation. Faculty information system capabilities are to generate courses report(s), for example, that includes course managerial issues. In order for faculty information system to be an e-Learning enabler, learning considerations should be considered, not just managerial.

### 3.1.4 Finance System

Finance system is responsible for managing financial issues related to any organization, even if this organization is a faculty and/or university. All learning involved partners agree that learning is not about financial issues, so finance system by itself is clearly not e-Learning.

## 3.2 UMIS Role

UMIS achieved success over the years and proved efficiency and effectiveness within educational institutions. UMIS is required for any successful e-Learning implementation in the three learning models, but with constraints about the role it should play. UMIS manages educational institutions, and more attention should be paid to the learning process with the presence of UMIS [18].

## 4. Prototypical e-Learning

Researchers attempt to define a prototypical e-Learning model over the years, resulting in many considerations, points of views, and acronyms that are hard to count. Each acronym reflects its presenter point of view, and features that must exist; from her/his point of view. Acronyms include [19]: Distance Education [20,21], Telecast [22], Adaptive Teaching System [23], Authoring System [24], CAI: Computer Assisted Instruction [25,26], Electronic Courses [27], Online Courses [27], CIT: Computer-Information-Television [28], Computer Managed Learning System [29], Computer Assisted Learning [29], Integrated Student Information System [30], CBT: Computer Based Training [31], LMS: Learning Management System [32], Interactive Learning Environment [33], Course Management System [34], Courseware Authoring Tool [35], Assessment Management System [36], Integrated Learning System [37], CAPA: Computer Assisted Personalized Approach [38], Collaborative Learning [39], Virtual College [27], VLE: Virtual Learning Environment [40], Virtual Conference [41], Virtual Classroom [42], WBT: Web Based Training [43], LCMS: Learning Content Management Systems [43], Web-based Interactive Course [44], PLE: Personal Learning Environment [45], Virtual University [46], Enterprise Course Management System [47]. By studying all mentioned acronyms, it is clear that there is confusion between tools, features, and concepts. In attempt to organize those acronyms, extract features, and tools that enable those features, clarify concepts, and keeping in mind that the learning process is our main concern, figure 1.3 attempts to clarify one of the ways to categorize those acronyms. Though digital library have not been mentioned explicitly as a main enabler of e-Learning, but it deserves to be added as part of the extended LMS due to its importance for the learning process.

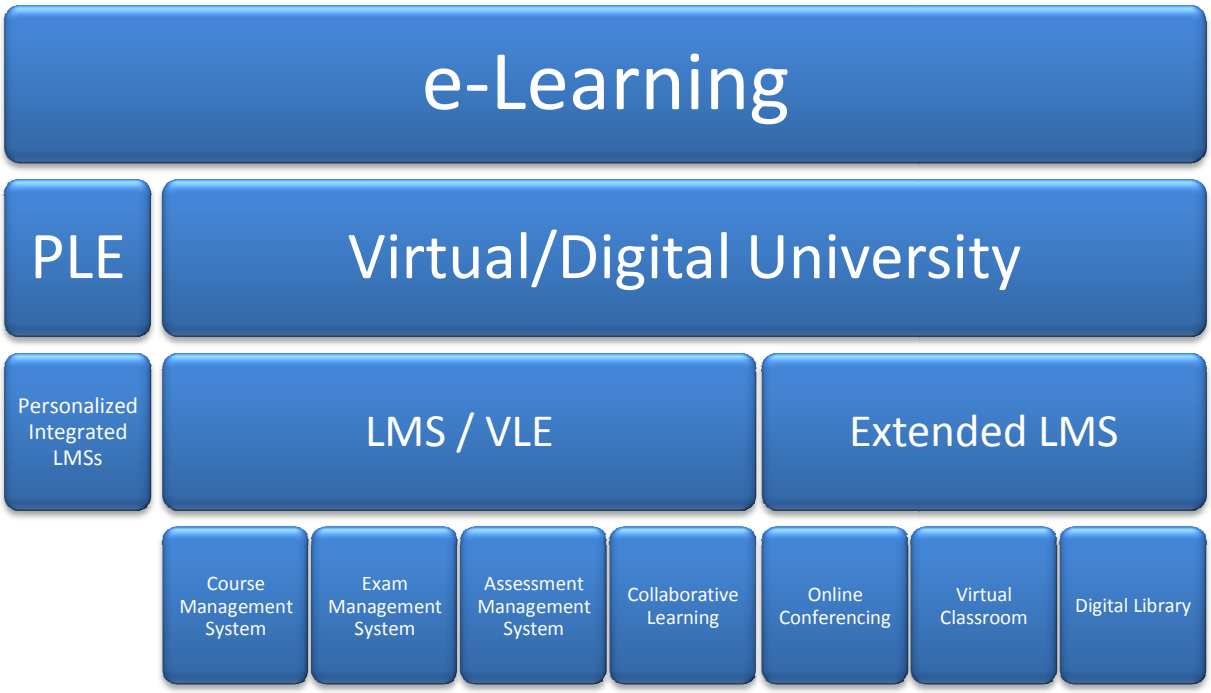


**Figure 1.3: e-Learning umbrella covers many acronyms**

e-Learning implements technology that enables Virtual/Digital University, and/or Personal Learning Environments. Virtual/Digital University is the University that implements Online Learning Management Systems / Virtual Learning[2] Environments and provides tools for Virtual College. E-Learning is the main concept that includes enabler technologies implementation for both Virtual/Digital University and Personal Learning Environments (PLE).

PLE represents a new trend in e-Learning that claims student's right to use only one gateway to be able to access different LMSs provided by different universities. Those different LMSs should be personalized and integrated within this gateway [45].

Universities and colleges are digitized by implementing ICT. The maximum extent of digital university is the Virtual University; where the whole learning process is managed and maintained digitally. LMS/VLE and Extended LMS are the main implementation of e-Learning today.

### 4.1 Learning Management System / Virtual Learning Environment

LMS and VLE are different acronyms for the same concept. For the rest of this thesis, LMS will be used to refer to both LMS and VLE. LMS is the Software that automates the administration of training. The LMS registers users, tracks courses in a catalog, records data from learners; and provides reports to management. An LMS is typically designed to handle courses by multiple publishers and providers. It usually doesn't include its own authoring capabilities; instead, it focuses on managing courses created by a variety of other sources [2,3]. A prototypical LMS is presented in [40]. LMS features can be categorized into four main separate systems as depicted in figure 1.4. Those four separate systems are concerned with Courses, Exams, Assessments, and Collaborative features. LMS can be thought of as the integration of four separate systems, each system presents specific functionalities via specific tools. Figure 1.5 depicts the most common features that should be available in each of those four separate systems.

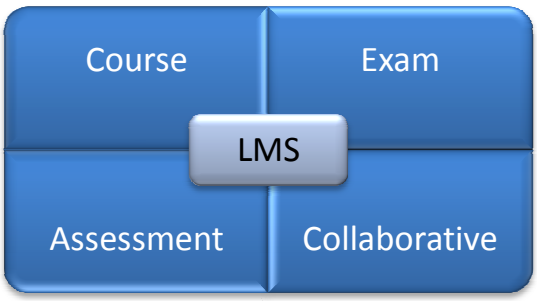


**Figure 1.4: Main functionalities served by LMS**

[2] Learning Management System (LMS) and Virtual Learning Environment (VLE) reflect the same implementation of the same concept. LMS as an acronym is widely used in the United States and was presented in 1980. VLE as an acronym is widely used in United Kingdom and it was presented in 1983 [20].


**Figure 1.5: LMS components' functionalities**

## 4.2 Extended LMS

Extended LMS includes functionalities that must not be mandatorily provided by LMS, but are preferred to exist. Intercommunication means and digital library are two examples of those functionalities. Virtual intercommunications means enhances communication between instructor, students, and instructors/students, and enables virtual communities to exist.

### 4.2.1 Online Conferencing

Conference is a prearranged meeting for consultation or exchange of information or discussion, especially one with a formal agenda [48]. Providing conference capabilities over internet is what is called Online Conferencing. Online Conferencing can be presented in one of three main forms:

- **Data Conferencing** [49]**:** Sharing data interactively among several users in different locations. Data conferencing is made up of whiteboards and application sharing and are often used in conjunction with an audio or videoconferencing connection.
  - **Whiteboards:** A whiteboard is the electronic equivalent of the chalkboard or flip chart. Participants at different locations simultaneously write and draw on an on-screen notepad viewed by everyone.

- **Application Sharing and Application Viewing:** Application sharing is the same as remote control software, in which multiple participants can interactively work in an application that is loaded on only one user's machine. Application "viewing" is similar to application "sharing;" however, although all users can see the document, only one person can actually edit it.

- **Audio Conferencing:** An audio communications session among three or more people who are geographically dispersed. It is provided by a conference function in a multiline telephone or by the telephone companies or using internet [50].
- **Video Conferencing:** A real time video session between two or more users or between two or more locations. Videoconferencing may comprise any number of end points [51].

### 4.2.2 Virtual Classroom

Virtual Classroom can be defined as the online learning space where students and instructors interact [10]. Virtual Classroom provides unique online features [52]

- Chat: two participants can exchange text to share thoughts between them.
- Discussion: Chat between more than two participants. Discussions can be public or private among part of the students.
- Question and Answer (Q&A): Individual participants may ask questions. Instructors may provide public or private answers.
- White Board: enable instructor to draw or display whatever compatible components.
- Group Browser: Participants can type the URLs in the address box, and sites are displayed to the entire group.
- Break Out Sessions: allows a subset of learners to enter a private chat area and use the virtual classroom tools.

### 4.2.3 Digital Library

Digital libraries are libraries that contain digital materials [17]. Digital Libraries implementations might include digital data from academic institutions, public libraries, government agencies, and museums [53]. Digital libraries play an important role in the learning process due to the tremendous amount of different digital data available to be accessed by any of the LMS components anytime, and anywhere. Digital data can be implicitly provided to instructors, and students without the awareness of the presence of the digital library.

## 5. UMIS or LMS

Universities require both UMIS and LMS for efficiency and effectiveness [19]. Neither UMIS nor LMS can replace the other. Figure 1.6 clarifies this distinction by presenting that University managerial requirements are addressed by UMIS, and Learning Process requirements are addressed by LMS, so University needs to implement both systems.

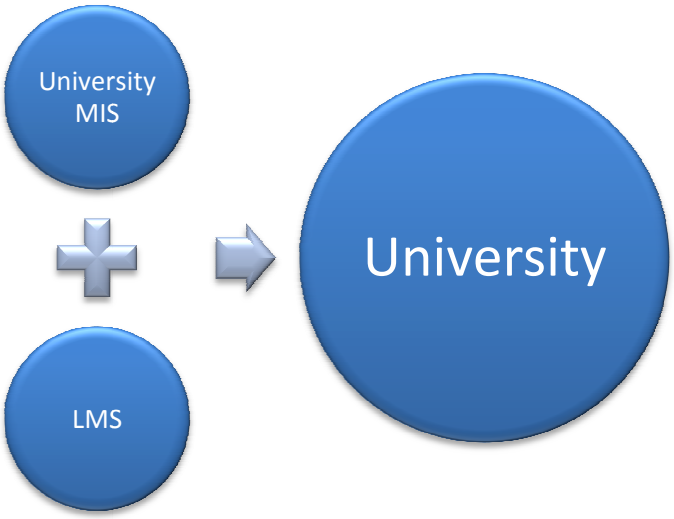


**Figure 1.6: University systems**

Sociotechnical systems recognized many years ago that organizations functioned most effectively when their social and technological networks were compatible [54]. Unfortunately, LMS providers attempted to overcome UMIS functionalities and include them into LMSs, leading to the malfunction of the learning process, and resulting in many LMS shortages as will be presented in section 7.

## 6. Current LMS

List of current LMSs include: .LRN [55], BlackBoard [56], Centra [57], COSE [58], LON-CAPA [59], Moodle [60], The Learning Manager [61], Angel [62], ATutor [63], Claroline [64], Desire2Learn [65], Eledge [66], IntaLearn [67], KEWL [68], WebMentor [69], Janison Toolbox [70], KnowEdge eLearning Suite [71], Unicon Academus [72], BSCW [73], Colloquia [74], eCollege AU+ [75], ILIAS [76], Internet Campus Solution [77], MimerDesk [78], SAKAI [79], and IBM Lotus [80]. Figure 1.7 categorizes presented LMSs as Open Source, Free, or Commercial. Open source LMSs are LMSs that binary download for source code is available. Free LMSs are LMSs that installer download and usage in free and unlimited, with no source code available. Commercial LMSs are neither open source nor free. Users pay for purchasing and running commercial LMSs. By surveying LMSs, it is clear to find that most LMSs implement same features as depicted in prototypical LMSs as the one depicted in [18]. One of the LMSs will be discussed in details in the next subsection.


**Figure 1.7: LMSs License Categories**

### 6.1 IBM Lotus LMS

IBM Lotus is one of the leading LMSs that implements too many complicated features at a high level [81,82]. IBM Lotus is one of LMS dominants [83]. Figure 1.8 presents a detailed Lotus architecture and list of functions that are performed by Lotus components.

IBM Lotus is AICC certified since 1997 [82]. By studying Lotus preview [84] and architecture [85], it becomes clear that it is not applicable to satisfy all the educational institutions requirements within one LMS no matter what efforts are attempted by companies. Concerning Lotus as one of LMS leaders, neither the presence of 11 servers nor letting down some of the functionalities is acceptable. Issues like scalability, interoperability, and integration have been mainly serious issues for current available commercial LMSs that forced many educational institutions towards in-house LMS development and implementation to satisfy educational institutions special requirements.

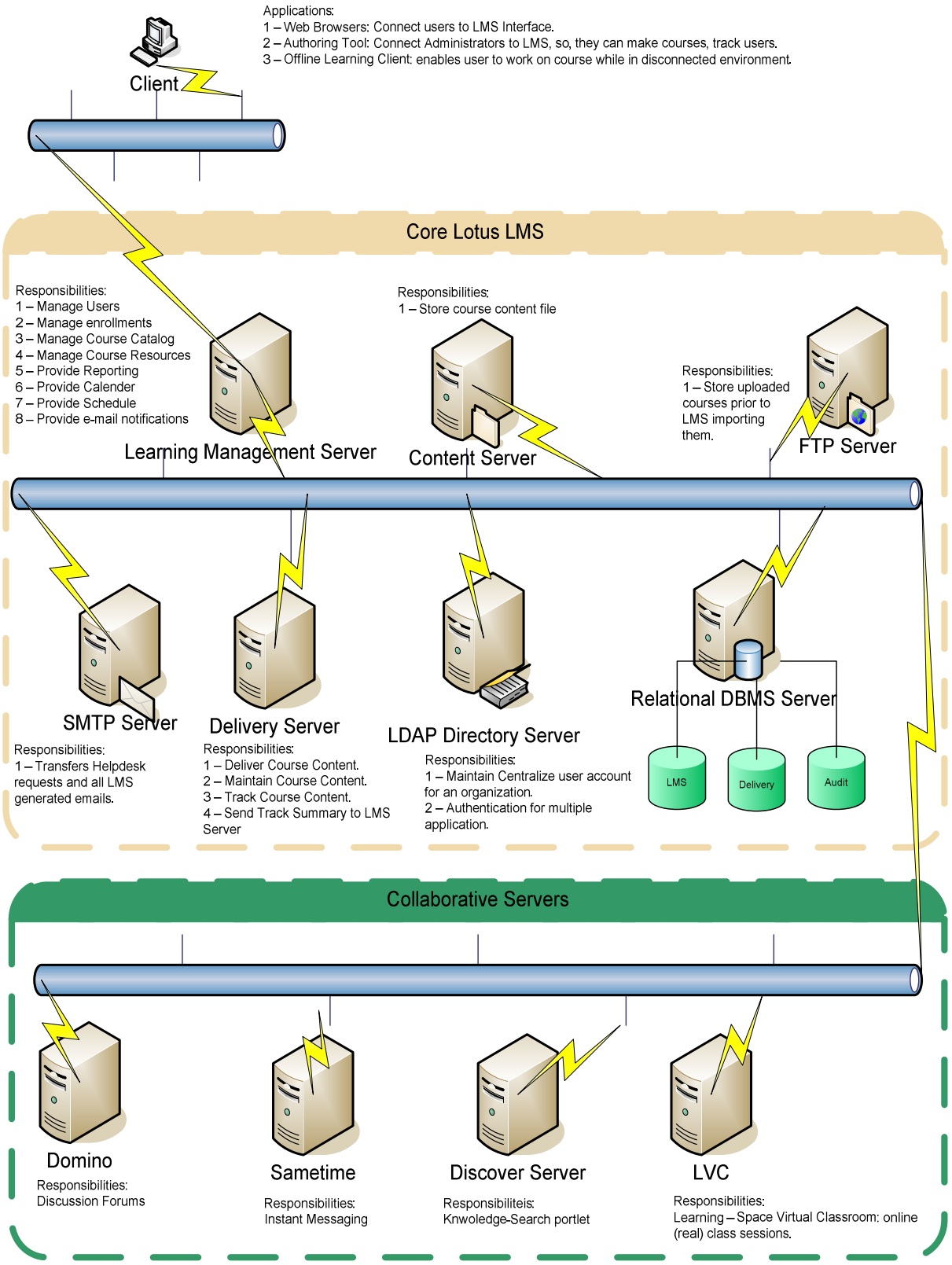


**Figure 1.8: Detailed Lotus Architecture**

## 7. Evaluation of Current LMSs

Research is an academic activity that comprises defining and refining the problems, formulating hypothesis or suggested solutions, collecting, organizing, and evaluating data, making deductions and reaching conclusions, and at last carefully testing the conclusions to determine whether they fit the formulating hypothesis [86]. Evaluation is a main step of scientific research that enables in concluding and reporting research results, efficiency, effectiveness, and goals achievement. Evaluation utilizes many of the same methodologies used in traditional scientific and social research, but because evaluation takes place within a political and organizational context, it requires group skills, management ability, political dexterity, sensitivity to multiple stakeholders and other skills that scientific and social research in general does not rely on as much [87]. Evaluation-is the collection and analysis of information by various methodological strategies to determine the relevance, progress, efficiency, effectiveness, and impact of programs activities [88 - 90]. Evaluating LMS is not like evaluating any software system, because LMS should address pedagogical aspects, beside architectural and managerial aspects as depicted in figure 1.9.

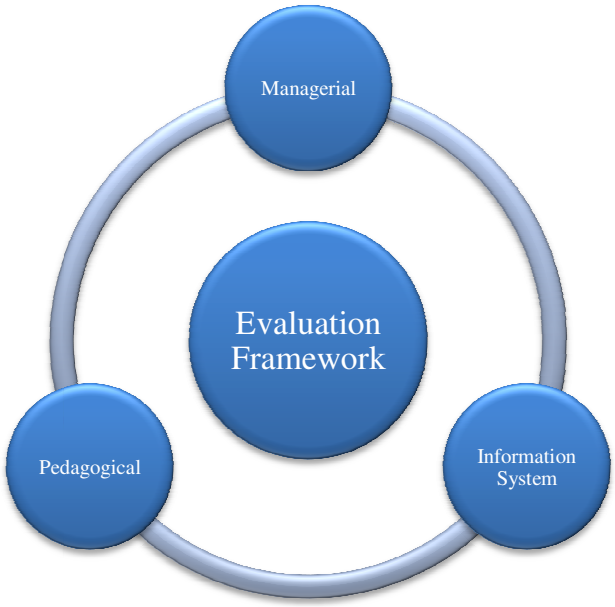


**Figure 1.9: LMS Evaluation Framework**

**Managerial LMS Evaluation:** Total Cost of Ownership (TCO), among other methodologies like Cost-Benefit Analysis (CBA), is widely used as an analytical and justification tool for software assessment, replacement, and acquisition projects. Unfortunately, TCO analysis is time-consuming to complete based on assumptions, and sometimes hard to quantify. TCO evaluation of a Financial Information Systems is available in [91]. Same TCO managerial evaluation model can be applied from the managerial perspective to LMS. Managerial LMS Evaluation is out of scope.

**Pedagogical Evaluation:** Pedagogically, LMS shall enable universities and educational institutions to provide educational services in an easy, effective, and efficient manner. LMS providers and evaluators must be aware of pedagogical effects that will affect instructors and students. Current LMSs do not provide the required pedagogical effects [7]. One of the reasons is technology limitations. If technologies applied in LMS will not enhance learning process then pedagogically it is not a necessity and unfortunately this is the case today.

### Information System Evaluation

One of the most important models to be used in evaluating information systems is the one presented by the standard ISO-IEC 9126. Figure 1.10 presents other architectural parameters that can be used in evaluating information systems. Presented parameters has not been addressed by ISO-IEC 9126 Standard, but still can be used by information systems evaluators. It is evaluators' responsibilities to determine the most valuable architectural aspects to be considered in the evaluation process. Another architectural evaluation model that deserves mention and well consideration is the one presented in [92].


**Figure 1.10: Non-ISO Addressed IS Quality Parameters**

The study of most well known LMSs; including Lotus, has lead to the realization of the reality that there are many shortages of current LMSs affecting e-Learning in educational institutions This part tends to present some of the shortages addressed through evaluations of current LMSs. Some of the addressed deficiencies are: Integration, Agility, Scalability, Extensibility, Flexibility, Interoperability, and Redundancy.

### 7.1 Integration deficiency

E-Learning solutions are clearly a combination of two main categories of applications: LMS and UMIS. As presented in figure 1.11, there are four integration challenges:

- Integration between each category composing applications.
- Integration between the two categories.
- Integration between educational institutions allover the world.
- Integration between educational institutions and external institutions.

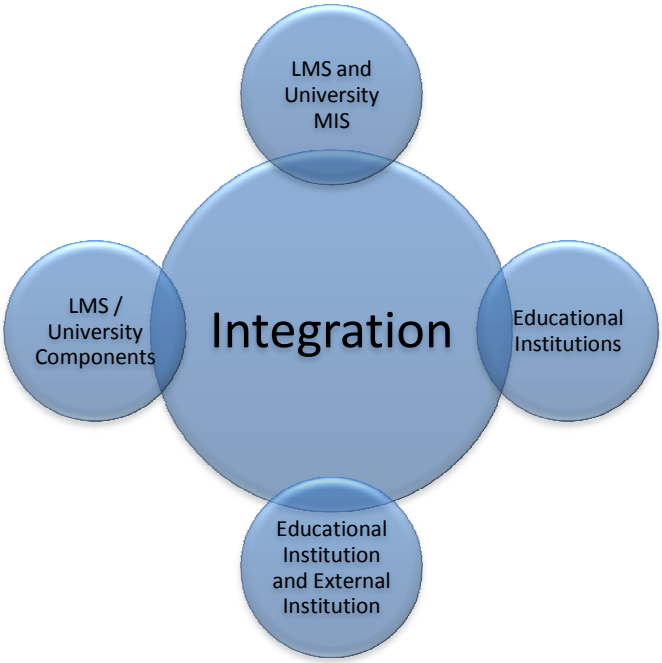


**Figure 1.11: University Integration Challenges**

#### 7.1.1 Category components integration

Composing applications might come from different software vendors, legacy applications that have been running for a long time needs to be integrated with new applications, and new software applications might be installed. Existing applications might stand in the way of installing new applications due to integration problems.

#### 7.1.2 UMIS and LMS integration

UMIS advances LMSs by decades [19]. UMIS achieved stability levels that most major educational institutions are running the same UMIS for almost a decade with no noticeable problems. LMS is the new concept that needs to be added to educational institutions, and it has to integrate with long time running UMIS. Due to differences in nature of design objectives between UMIS and LMS, integration issue should be addressed clearly. There will be new features that university will be able to provide incase full integration of UMIS and LMS took

place. The capability of presenting individualized assessments that can be generated by integrating SIS; which will include two profiles for student; a managerial profile, and an educational profile; and the assessment management system should be addressed.

The capability of presenting individualized learning plans by integrating CMS and SIS that holds the learning profile should be addressed. Personalized courses will be generated based on learner history profile. It is not acceptable by learning experts to fix time and change amount to be learned, instead, amount to be learned should be fixed and personalized. Such individualizations enable PLE.

### 7.1.3 Integrate different LMSs

It is hard to find an educational institution that implements only one LMS because educational institutions requirements are hardly met by one LMS [7]. Different LMSs, either within the same educational institutions or over different ones needs to be integrated. Integration over courses level as provided by utilizing SCORM and AICC is not the typical solution of integration problems, because educational institutions need to share more than courses.

### 7.1.4 Integrate educational institution and external institutions

Educational institutions are not isolated islands that work alone, but they are part of the society. Educational institutions need to be integrated with all other governmental institutions. Tasks like generating a report with all current certain grade students, or students that are qualified for certain task because they have attended certain courses should be easily executed. Querying the educational institutions databases should be available efficiently. Student's data shall be available to give the student ability to transfer his data between educational institutions as required.

## 7.2 Agility deficiency

From certain perspective, educational institutions are organizations, just like any other organizations. Organizations need agility, and so do educational intuitions. Agility is the ability to sense change and opportunity, respond quickly, and execute successfully. Educational institutions need to reflect governmental new or modified rules on current UMIS, overcome merge and acquisitions challenges, and provide new and customized services based on new added system features. An example of new customized services that should be presented when new system features are added is collaborative assignment. Collaborative assignment can be provided by the system after installing new collaborative tools. If the system were not built with agility in mind, new features adding or existing features editing becomes almost impossible, limiting the system from adapting and presenting new functionalities.

### 7.3 Scalability deficiency

System should expand and contract its resource pool easily to accommodate to heavier loads. Client – Server based LMSs can not scale to support large number of components, or interactions among components, within an active configuration [92]. Most today's LMSs are Client – Server based [41]. Scalability needs to be addressed to offer educational institutions with the LMS solutions that satisfy the increasing requirements.

### 7.4 Extensibility deficiency

Extensibility is the ability to add functionality to a system. Dynamic extensibility implies that functionality can be added to a deployed system without impacting the rest of the system [92]. That implies not only adding new features to existing LMS without the need to install the new version, but also adding customized functionalities required by the educational institution. Unfortunately, current LMSs are released periodically to provide new functionalities, with the need to install the new version every time. Besides, most commercial LMS are categorized according to functions available by each product level, without guarantee on what are exactly the actions required to be taken to make the move to the next level incase required.

### 7.5 Flexibility deficiency

Flexibility is the ability of a system to respond to potential internal or external changes affecting its value delivery, in a timely and cost effective manner. The ability to change or to be changed according to circumstances is not by default provided by Client – Server architectural based LMSs, and unfortunately most LMSs are Client – Server based [41]. Changes include addition/loss of servers, and types of services provided by the system.

### 7.6 Interoperability deficiency

Interoperability is the capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units [93]. Interoperability is concerned with processes require the system to communicate with external systems to accomplish tasks. Educational institutions need to interact with external systems to achieve tasks successfully. Interoperability between LMS and external systems needs to be addressed and enabled. Current commercial LMS architectures and used protocols are not provided with LMS. A standard interface to LMS functionalities is required.

### 7.7 Redundancy

Redundancy refers to the unmanaged, uncontrolled, and unwanted more than single time occurrences of either data or functionality.

### 7.7.1 Data Redundancy

Data redundancy refers to existence of the same piece of data in more than one place. Data is said to be in consistent state when all data redundancy are controlled. Data redundancy should be controlled for insert, update, and delete queries. Figure 1.12 depicts an example of uncontrolled data redundancy that causes inconsistent data and system state. SIS, CMS, assessment, exam, and finance information systems need student Data, unfortunately with different formats and different types. Questions like which system should manage this data? Shall every system manage its own data? Shall the system take the risk of having a redundant data? Need to be answered.

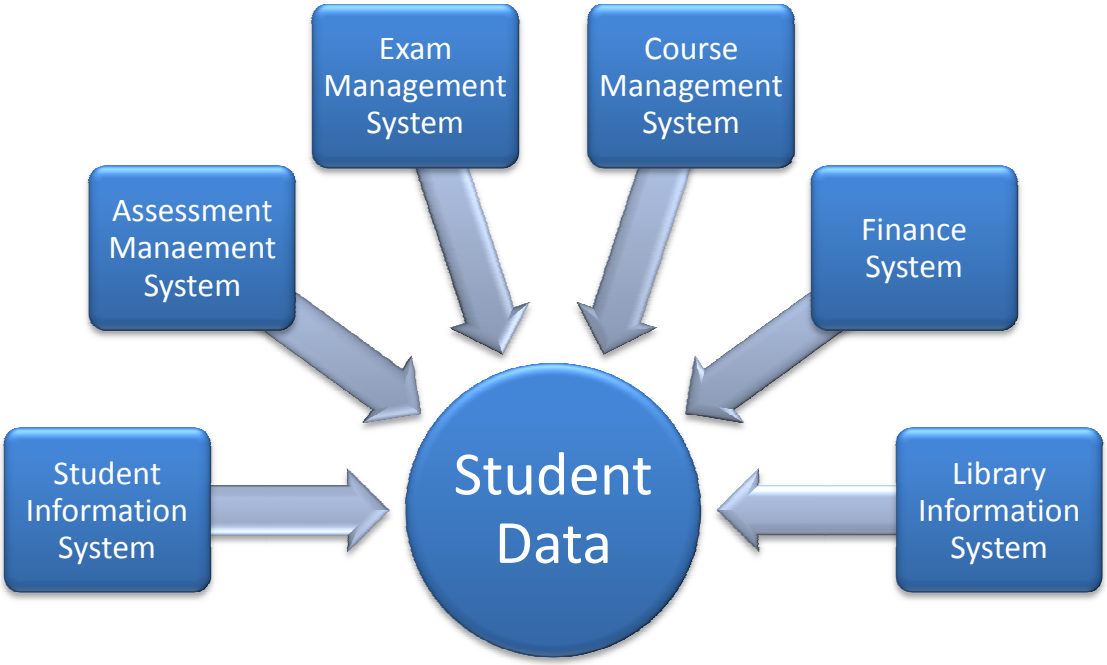


**Figure 1.12: Redundant Student Data**

### 7.7.2 Functionality Redundancy

Functionality overlapping and redundancy is clear from the literature review. Functionality redundancy comes from the attempt of LMS software vendors to increase the tasks performed by LMS to include functions that are not part of it. Redundancy is accepted only if it is managed.

## 8. Thesis Motivation

This chapter presented the information systems utilized by educational institutions. They are UMIS and LMS. UMIS serve educational institutions to automate managerial tasks, and LMS provides the required functionalities to automate the learning process. Educational institutions need both UMIS and LMS to satisfy e-Learning and enhance the learning process. Both UMIS and LMS are composed of collection of task specific applications. Educational institutions; and

as a result the learning process, can be enhanced by enhancing the information system that serves them. Shortages of current LMS are evaluated in order to find solutions that serve educational institutions to better address information, learning, and functionality requirements.

Service Oriented Architecture (SOA) is one of the software architectures that satisfy most of the non-functional requirements addressed by information systems. SOA principles need to be presented and studied in order to utilize SOA for UMIS and LMS to overcome current LMS shortages, limitations, and deficiencies. SOA as a design pattern is expected to solve many of the current information systems' problems by easing integration, interoperability, and agility.

The idea of this thesis is to adopt SOA in e-Learning via presenting UMIS and LMS based on SOA, and evaluate the proposed components to determine the efficiency of SOA. Evaluating LMS is not limited by evaluating the architectural style of the system, it shall be expanded to include pedagogical and managerial aspects too.

## 9. Thesis Goals

This thesis attempts to address the shortages of current University Management Systems and Learning Management Systems, specially the gap between the two mentioned information systems, highlighting some of the advantages that can be achieved by integrating both together, and evaluating SOA as the software architecture to integrate both.

Thesis activities and tasks are:

- Surveying e-Learning to determine what e-Learning means, and to formalize the huge differences between what e-Learning means for everyone.
- Studying Enterprise Architectures and Enterprise Non-Functional requirements, the relationship between both, and the capability of Enterprise Architectures to achieve non-functional requirements.
- Surveying Service Oriented Architecture as the software architecture that satisfies mostly all functional and non-functional requirements.
- Proposing the Services based Learning Management System and University Management System, with extended analysis, design, and implementation details.
- Evaluating the implemented Learning Management System and University Management System components.

## 10. Thesis Outline

Thesis includes 6 chapters that are organized as follows

**Chapter 1** Introduces e-Learning, models, acronyms, and defined problems. UMIS, LMS, evaluation of current LMS are addressed.

**Chapter 2** Presents an intensive SOA overview, characteristics and advantages, SOA enablers (like Software agents and Web services), Web services as main SOA enabler, advantages and key features of Web services.

**Chapter 3** Presents the architecture blueprint that will be used for all proposed architectures. Also, presents major UMIS services based components, besides analysis, design, and implementation details. Presented UMIS components are: Student Affairs Management System (SIS), and Library Management System.

**Chapter 4** Introduces LMS components and it is divided into three parts

**Part I** Presents SOA for Digital Library. Digital Library does not refer to automation of library activities and managing it electronically, but it refers to presenting University and Faculty books, research papers, and any other educational material in a digital format and making them available for students and for the rest of applications to integrate, and make use of.

**Part II** Presents SOA based Course Management System (CMS) that address new features and capabilities like searching, importing, and unlocking different course repositories, besides integrating software agents with Web services.

**Part III** Presents an extension to the SOA based LMS that addresses assessment, and the capability to introduce mobile assessment. It presents the experience of providing an online SOA based Assessment Management System (AMS). Students will get their usernames and passwords, log in, target their due assessment, take it, and check the result. Proposed SOA based AMS facilitates interoperability requirements of Mobile Assessment.

**Chapter 5** Presents an evaluation of the proposed UMIS and LMS components against some information system quality characteristics and some pedagogical features. From information system point of view, points like performance, integration and interoperability, compliance, security, maintainability, analyzability, decomposability and modularity, testability, portability via replaceability and scalability, simplicity, modifiability, and reusability are addressed. A Comparative performance analysis is presented to test SOA based systems user-perceived performance against non-SOA based systems. Pedagogically, SOA adoption enhanced the learning process activities and provided capabilities that was hard to present before, like unlocking and sharing course repositories of different LMSs, and M-Learning.

**Chapter 6** Concludes thesis summarizing evaluation results and highlighting information system and pedagogical advantages of adopting SOA in e-Learning.

# Chapter 2

# Chapter Two

# SERVICE ORIENTED ARCHITECTURE

## 1. Introduction

Service Oriented Architecture (SOA) is one of the software architectures that satisfy most of the non-functional requirements addressed by information systems. SOA principles need to be presented and studied in order to utilize SOA for UMIS and LMS to overcome current LMS shortages, limitations, and deficiencies. SOA as a design pattern is expected to solve many of the current information systems' problems by easing integration, interoperability, and agility.

## 2. Service and Service Orientation

Objects, components, and services are three main acronyms in software architecture that are interrelated. Objects have lead to components, and services can be thought of as specialized type of components.

### 2.1 Software Objects

Object is the encapsulation of data and behavior in one data type; class; as depicted in figure 2.1. Objects are instances of classes that exist for a certain period of time. Objects communicate with each other via exchanged messages. Messages are instructive, not descriptive. Object based development advances software design by providing more support for hiding behavior and data through objects, however, large number of interconnected objects create dependencies that can be difficult to manage. Object based software development is called Object-Oriented Analysis and Design (OOAD) [94]

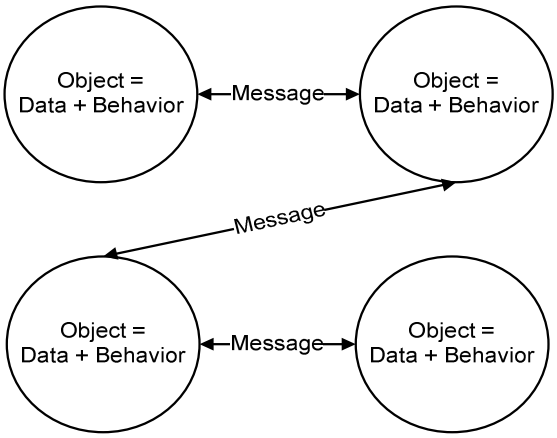


**Figure 2.1: Objects encapsulate data and behavior**

## 2.2 Software Components

Components are more sophisticated software modules than objects and require fundamental changes in systems thinking, software processes, and technology utilization. Software component is a unit of composition with contractually specified interfaces and explicit context dependencies [95]. It is a group of objects with has specified interface, working together to provide an application function, such as depicted in figure 2.2.

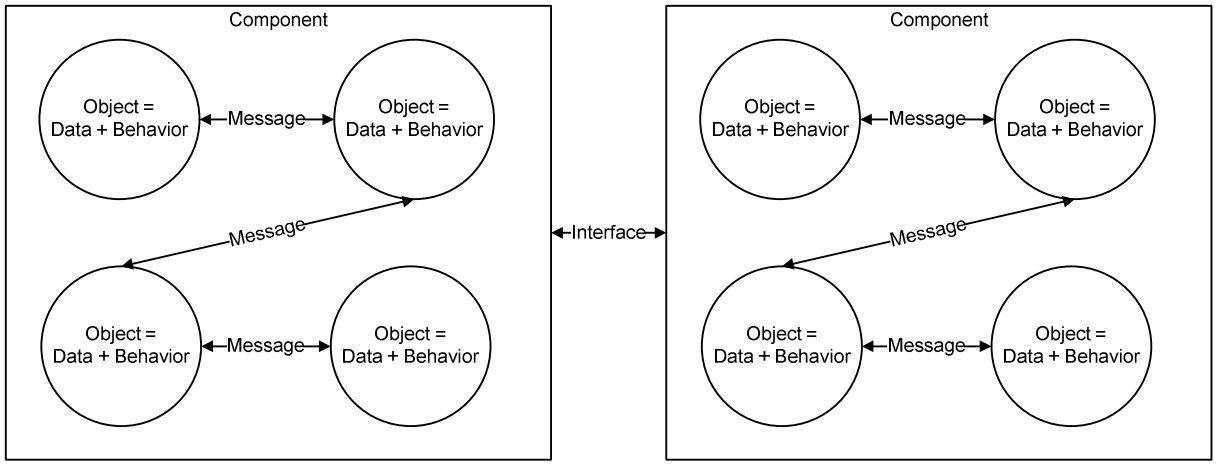


**Figure 2.2: Software Component**

Component may refer to many different software constructs, from single application logic to an entire functional system. In all cases, a component is a software package with one or more well defined interfaces. Components overlap the properties of object orientation, such as encapsulation and polymorphism, except it reduces the property of inheritance. In component thinking, inheritance is tightly coupled and unsuitable for most forms of packaging and reuse [96]. Instead, components reuse the functionality by invoking other objects and components rather than inheriting from them [97]. Reusable components are good reflection of effective software design. The development of software architecture based on component specifications supports parallel and independent building of the system parts. Many platform vendors have already produced software infrastructures which support component oriented technology.

## 2.3 Service and Service Models

Service is a component capable of performing a task [98]. Services can be categorized based on the nature of logic they encapsulate and the manner in which they are typically utilized within SOA as depicted in [99].

## 3. Service Oriented Architecture

SOA is defined as the policies, practices, frameworks that enable application functionality to be provided and consumed as sets of services published at a granularity relevant to the service

consumer. Services can be invoked, published and discovered, and are abstracted away from the implementation using a single, standards based form of interface [98 - 101].

### 3.1 Advantages of Service Oriented Architecture

SOA advantages are categorized into implementation, and organizational benefits.

#### 3.1.1 Implementation Benefits

Implementation benefits satisfy the loose coupling objective. Using a resource only via its published service and not by directly addressing the implementation gives system capabilities of [102]:

- Changes to the implementation by the service provider should not affect the service consumer. Services are exposed via standard interfaces and are thought about as black boxes that changes within it do not affect consumers.
- Service consumer could choose an alternative instance of the same service type without modifying requesting application. As long as new service implements the same interface, theoretically there are no problems.
- Service consumer and service provider do not have to implement same technologies for implementation, interface, or integration when Web services are used.

#### 3.1.2 Organizational / Business Benefits

Businesses are dealing with two fundamental concerns: the ability to change quickly (agility), and the need to reduce costs [103]. Business must adapt quickly to internal factors such as acquisitions and restructuring, or external factors like competitive forces and customer requirements to remain competitive. Cost-effective, flexible IT infrastructure is needed to support the business. SOA can realize several benefits to help organizations succeed. Organizational/Business benefits of adopting SOA are:

- **Leverage existing assets:** by wrapping existing software applications as services instead or rebuilding new solutions from scratch.
- **Easier to integrate:** integration on service level in order to satisfy business processes integration presents the highest integration technique that solved many problems as depicted in chapter three.
- **More responsive and faster time-to-adapt (Agility):** the ability to compose new services out of existing ones provides a distinct advantage to an organization that has to be agile to respond to demanding business needs. Leveraging existing components and services reduces the time needed to go through the software development life cycle leading to rapid development of new business services and allows an organization to respond quickly to changes. SOA provides the flexibility and responsiveness that is critical to businesses agility.

- **Reduce cost and increase reuse:** with core business service exposed in a loosely coupled manner, they can be more easily used and combined based on business needs. This means less duplication of resources, more potential for reuse, and lower costs.

### 3.2 SOA Technologies

SOA implementations include: Software Agents, and Web services.

#### 3.2.1 Software Agents

Software Agent is a computer system that is situated in some environment and is capable of autonomous actions in this environment in order to meet its design objectives [104,105].

Different SOA implementations using different software agents are presented in [106 – 110]. SOA implementation via mobile agents technology can be found in [108]. The main idea is about having one or more software agents perform certain task(s), those tasks can be exposed as services that compose SOA.

Software agents have many classification criteria. Agents can be classified as either Desktop, Internet, or Intranet agents according to their action environment [111]. Many internet related agents can be classified into subcategories as presented in [112]. Intelligent agents can be classified into Simple reflex agents, Model-based reflex agents, Goal-based agents, and Utility-based agents [113]. Agents can be also classified by application types, and by characteristics [114].

Software Agents have characteristics that make them suitable to perform complex functionalities. Characteristics include: Autonomy, Interactivity, Reactivity, Proactivity, Intelligence, and Mobility [114]. Agent is autonomous; it is capable of acting on its own. An agent is goal oriented, collaborative, and flexible, so, it must be autonomous. Agents are designed to interact with other agents, humans, or software programs (Interactivity). Instead of making a single agent conduct several tasks, additional agents can be created to handle un-delegated task. Agents perceive environment via preceptors [113] and respond to changes (Reactivity). Agents do not just act in response to their environment, but agents are able to exhibit goal-directed behavior by taking an initiative (proactive). Agent may need mobility to work on different machines. An agent with this capability is called mobile agent, it can transport itself across different system architectures and platforms, and is far more flexible than those that cannot. Many electronic commerce agents are mobile [114]. Mobile agent is an executing program that can migrate during execution from machine to machine in a heterogeneous network [110].

#### Multi-Agent System

Multi-Agent Systems (MASs) are becoming increasingly important: as a scientific discipline, as a software engineering paradigm, and as a commercially viable and innovative technology [115].

A Multi Agent is any system that contains [116]:

- Two or more agents;
- At least one autonomous agent; and
- At least one relationship between two agents where one satisfies goal of the other.

Many Multi-Agent frameworks have been presented. Some of Multi-Agent frameworks proposed from 2005 till now include works presented in [117 – 122].

MAS characteristics are [123]:

- Each agent has incomplete capabilities to solve problems.
- There is no global system control.
- Data is decentralized.
- Computation is asynchronous.

**Common MAS Architectures**

Agents can be organized in different MAS architectures. MAS architectures include: flat, fixed hierarchy, Subsumption, and Modular [124].

**Flat MAS Architecture:** Agent are directly contact all other agents. The system is either closed, so all agents know the location of all other agents, or open, which requires agent location mechanism. Agent location mechanism can be one of message based architectures. Figure 2.3 depicts Flat MAS Architecture.

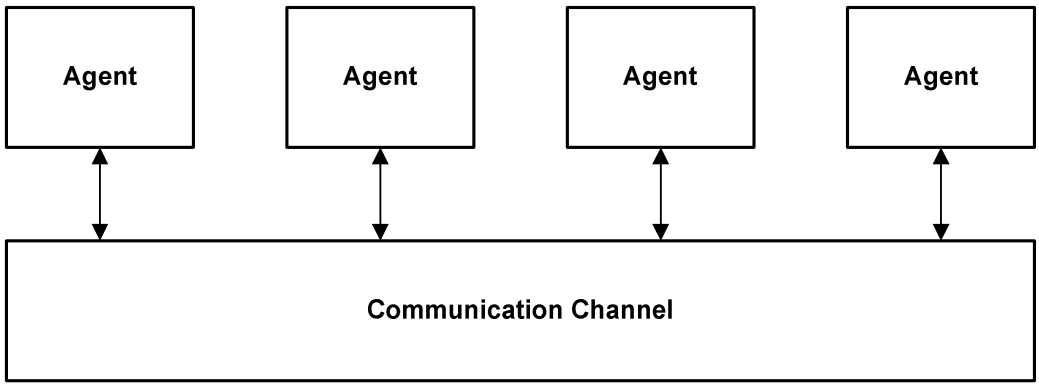


**Figure 2.3: Flat MAS Architecture**

**Fixed Hierarchy MAS Architecture:** Agents communicate only to agents directly above or below them in the hierarchy. Hierarchy is fixed leading to defects in systems scalability. Figure 2.4 depicts hierarchal MAS architecture.

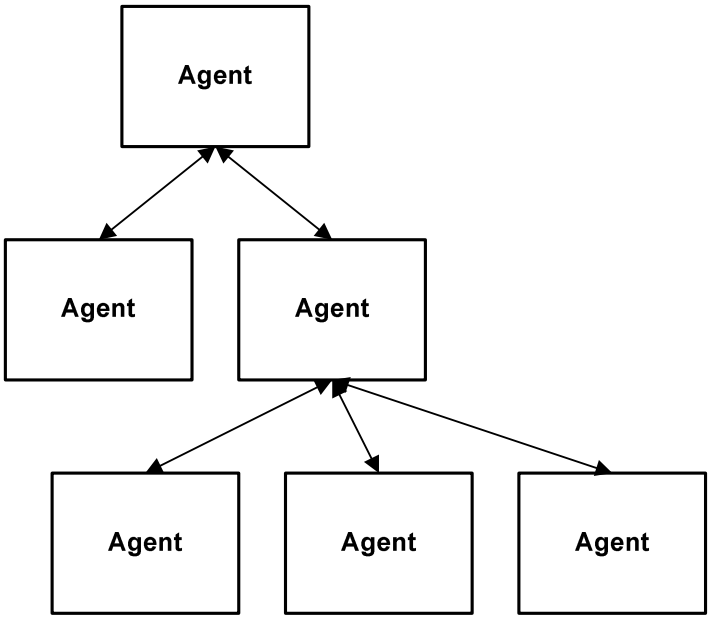


**Figure 2.4: Hierarchical MAS Architecture**

**Subsumption MAS Architecture:** Agent can contain other agents. Highly performance and wide scalability is the main advantage of subsumption architecture. Figure 2.5 shows subsumption MAS architecture.

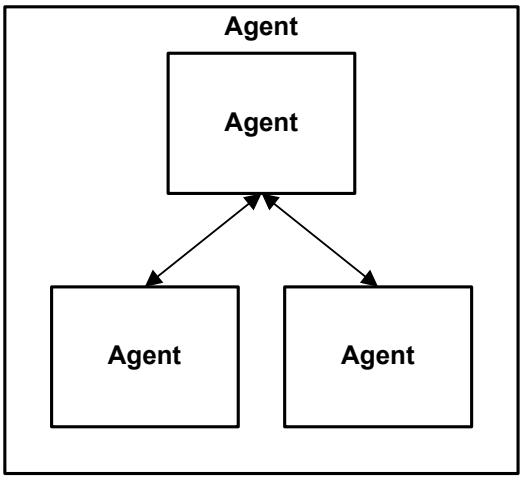


**Figure 2.5: Subsumption MAS Architecture**

**Modular MAS Architecture:** Agents are grouped in modules as presented in figure 2.6. Communication takes place between agents composing the module, and agents of different modules. Each module is specified with one or more task(s). Presenting systems as collection of modules facilitated problem solving.

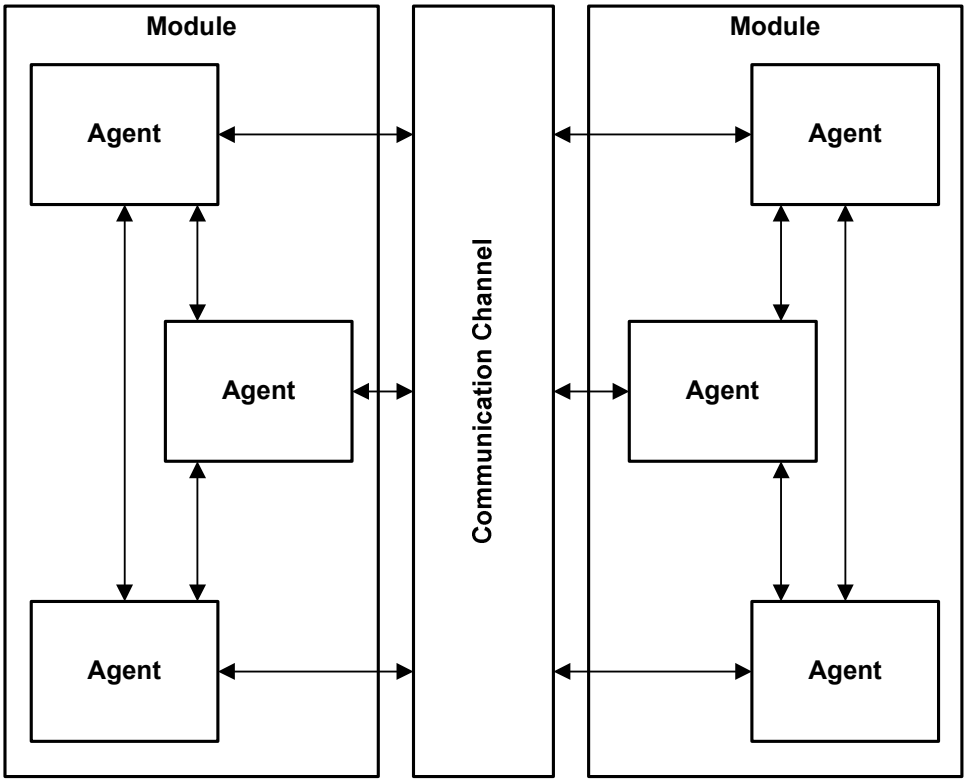


**Figure 2.6: Modular Organization**

Multi-Agent architecture standards were attempted in order to force MAS standardization and global integration. Knowledge Query and Manipulation Language (KQML) was presented in order to support knowledge sharing among agents [125]. Knowledge Interchange Format (KIF) is a computer-oriented language for the interchange of knowledge among disparate programs [126]. The OMG group proposed a reference model as an attempt to standardize the development of agent technologies [127]. KAoS is described as "an open distributed architecture for software agents." The KAoS architecture describes agent implementations, and elaborates on the interactive dynamics of agent-to-agent messaging communication by using conversation policies [128]. The Foundation for Intelligent Physical Agents (FIPA) is a multi-disciplinary IEEE standardizing group pursuing the standardization of agent technology. FIPA's approach to MAS development is based on a "minimal framework for the management of agents in an open environment" [129]. Unfortunately, as a result for all the standardization effort, there were no universally accepted commercially supported standard yet.

### 3.2.2 Web services

Web services are not just an integration solution. Web services technology is currently the major implementation of SOA [130]. All service models are specific to the WS-Coordination specification and related protocols [99]. Web services adoption by organizations solved many

problems. Web services is a general framework that expedites the sharing of heterogeneous data and software resources dispersed on the internet. The standard-based resource sharing and platform-neutral characteristics of web services have motivated many organizations to apply the technology in diverse areas, such as supply chain management, virtual enterprise, homeland defense, e-government, and e-business [131]. Software agents can consume Web services as presented in [132].

## 4. Web services Technology

Web services are applications that use standard transports, encodings, and protocols to exchange information [133]. A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. Web service has an interface described in a format that machines can process (specifically WSDL), Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with XML serialization in conjunction with other Web-related standards [134].

### 4.1 Web services Architecture

Web service technology is based on open technologies; this provides broad interoperability among different vendor solutions [103]. Figure 2.7 shows the basic Web services architecture, that consists of specifications (SOAP, WSDL, and UDDI) that support the interaction of a Web service requester with a Web service provider and the potential discovery of the Web service description [135]. The provider typically publishes a WSDL description of its Web service, and the requester accesses the description using a UDDI or other type of registry, and requests the execution of the provider's service by sending a SOAP message to it.

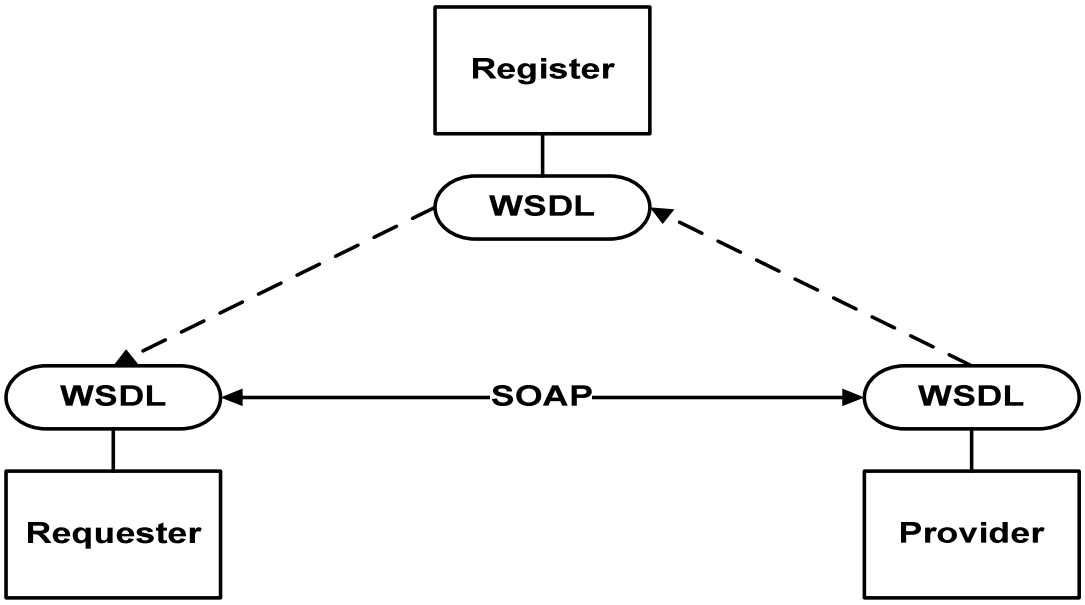


**Figure 2.7: Basic Web services Architecture**

### 4.2 Web services Key Features

Some of Web services key features are [103]:

- **Self-contained:** On the client side, a programming language with XML and HTTP client support is the only requirement. On the server side, a Web server and a servlet engine are required.
- **Self-describing:** format and content of request and response messages (loosely coupled application integration) is what only matters. Messages are descriptive, not instructive.
- **Modular:** Web services are a technology for deploying and providing access to business functions over the Web; J2EE, CORBA, and other standards are technologies for implementing Web services.
- **Web published, located, and invoked:** by applying the previously mentioned standards SOAP, XML, WSDL, and UDDI, Web services can be published, located, and invoked via internet.
- **Language independent:** interaction between a service provider and a service consumer is designed to be completely platform and language independent. Interaction requires a WSDL document to define the interface and describe the service, along with a network protocol (usually HTTP).
- **Interoperable:** Because service provider and service consumer do not share same platforms or languages; only communicate via standard protocols, interoperability is achieved.
- **Inherently open and standards based:** XML, SOAP, WSDL and HTTP are the technical foundation for Web services. A large part of the Web service technology has been built using open source projects.
- **Dynamic:** dynamic e-business can become a reality using Web services because, with UDDI and WSDL, the Web service description and discovery can be automated.
- **Compassable:** Web services can be aggregated to complex ones.

### 4.3 Advantages of Web services

Web services advantages arise from Web services key features. Web services were designed to satisfy all software requirements arose over the years, and avoid all drawbacks of previous technologies. Web services advantages include [133,136, 137, 138, 139]:

- **Interoperability:** Software interoperability is the capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units. Web services are interoperable.

- **Language agnostic:** Web services are neither based on a programming language nor on a programming data model. Any Programming language can be used to implement Web services.
- **Relatively simple:** Web services are simple to build, expose, and consume because they are:
  - Based on Web technologies: so, they are scalable, and easily to control security features.
  - Do not necessarily require a huge framework in memory: A small amount of code could expose a service as a Web Service, besides Web services can be used easily in today's Web Interface.
- **Loosely coupled applications:** applications based on Web services are loosely coupled applications by default. Loose coupling is a design goal that describes a resilient relationship between two or more applications, systems, or organizations with some kind of exchange relationship [140]. In Web services, exchange relationship is established via request/response messages.
- **Support of software industry leaders:** Hundreds of IT vendors have participated in the Web services standardization process under the sponsorship of the World Wide Web Consortium (W3C), Organization for the Advancement of Structured Information Standards (OASIS) and Web Services Interoperability Organization (WS-I). IT vendors include, not only: Microsoft, Oracle, bea, SAP, IBM, and Sun..
- **Integration with the World Wide Web:** Web services tended to integrate with World Wide Web from the very first beginning in order to use internet as the infrastructure. Internet integration provides the most advantage of adopting Web services within applications.

### 4.4 Web services as SOA enabler

Web services are relatively new technology that have received wide acceptance as an important implementation of SOA [103]. Web services is not SOA, and SOA is not Web services. It is important to differentiate between Web services; as a technology, and SOA as a design pattern. Web services enabled the maximum advantages of SOA, this is mainly why Web services is the main implementation and enabler of SOA. SOA presents systems as collection of services to be published, and consumed when required. When exposed services are Web services, system gains the advantage of Web services standardization, like interoperability. But still systems need to be studied carefully, and analyzed intensively to determine services will be exposed, suitable services interfaces, and design the required databases. SOA is not about exposing and consuming Web services to facilitate interoperability or integration or other requirements, but it is rather the science, art, and practice of building loosely coupled fine granular services to form applications.

## 5. Service Layers

Layers of abstraction are the main SOA advantages enabler [99]. By leveraging the concept of composition, specialized layers of services can be built. Each layer can abstract a specific aspect of the solution as presented in figure 2.8. Layers of abstraction differ from solution to another according to system requirements. Presented layers do not have to exist as an entity in all solutions. Architecting is the science and art of designing software systems, not the science of applying well defined steps. System architects are responsible for determine and defining the required layers of abstract to satisfy system requirements. Layering gives the enterprise capabilities to achieve enterprise-wide loose coupling by physically separating layers of services to collections of services that represent corporate business logic and other that represent technology-specific application logic. Each domain of the enterprise is freed of direct dependencies on the other [99]. Presented model depicts four abstraction layers, namely: presentation, orchestration, application services, and application layer.

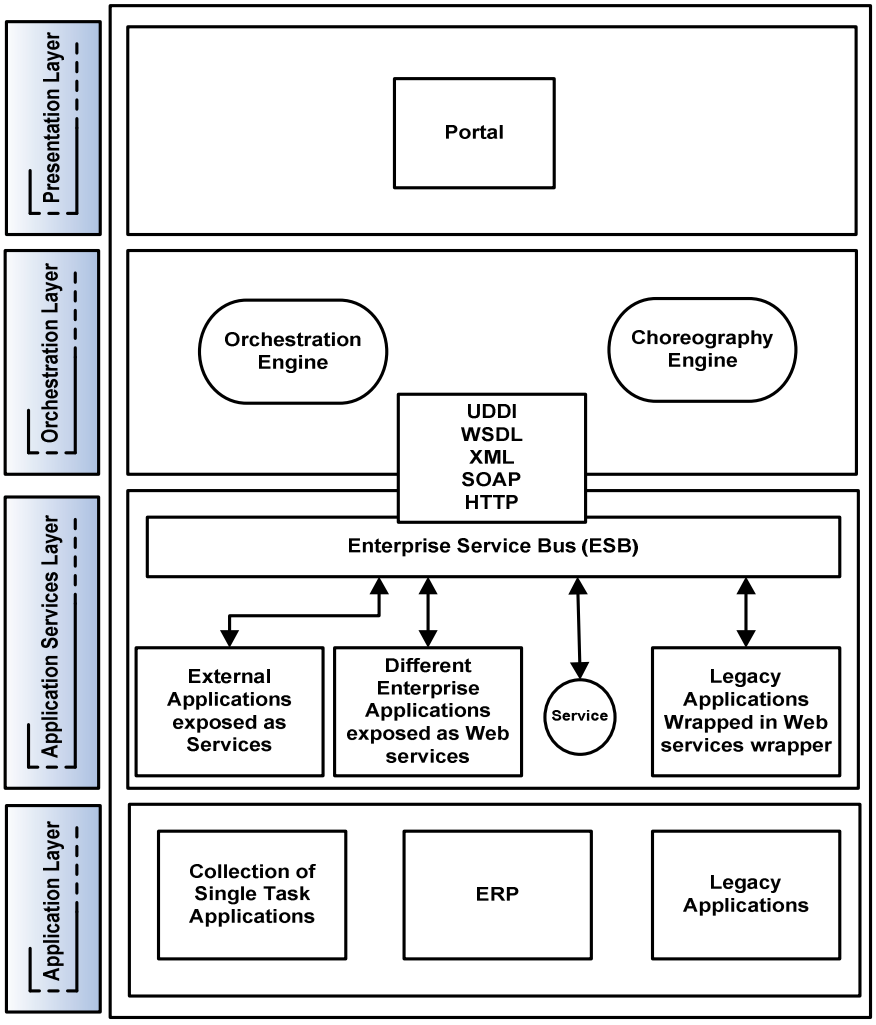


**Figure 2.8: Service Layers Model**

A Bottom-Up discussion of the presented model is presented to highlight the tremendous layering advantages.

### 5.1 Application Layer

Application layer holds current running applications within organizations. Application layer include legacy systems, software agents, single applications, and Enterprise Resource Planning applications (ERP). Legacy systems are computer systems that have been in operation for a long time, and whose functions are too essential to be disrupted by upgrading or integration with another system despite its poor competitiveness. Specific task software agents are required to serve and support system. Software agents can be thought of as the optimum solution for complex, complicated, long running, tracking, analyzing, and intelligent tasks. Software agents are black boxes that hide intelligence and functionality features as black boxes, unfortunately without a standard interface to be accessed. Single applications and ERP were not designed with integration in mind, so it is almost impossible for nowadays educational institutions running all those different kinds of applications to integrate them, so, the need to leave those applications just running is almost a necessity.

### 5.2 Application Services Layer

Application services layer holds applications exposed as services, newly added services, and legacy applications wrapped by standard Web services interface. Legacy systems compatibility with modern equivalents has been facilitated via wrapper services. Wrapper service is a type of integration service that encapsulates and exposes logic residing within a legacy system via standard Web services interface to be integrated in the new SOA based systems. Application Services are set of stateless Web services that perform certain task(s). Web services are stateless; they can not maintain business logic, operation flow, or user state. System process is the summation of tasks performed by one or more service of application services layer at the sequence maintained by orchestration layer services. Application Services are reusable among different processes, can be integrated in new applications, and can be extended address new system processes. Service granularity level reflects the extent to which entity related services are decomposed into.

### 5.3 Orchestration Layer

Web services are stateless services that can not maintain process logic, operation flow, or user state; so, the need of an orchestration layer to include process logic is addressed. Orchestration layer holds orchestration and choreography engines to maintain system process workflow logic, performance requirements, and system/user state. Orchestration expresses a system process that is typically owned by the organization, while choreography engine addresses the realm of collaboration between multiple services from different enterprises. Orchestration layer manages interaction details required to ensure that service operations are executed in a specific sequence.

Orchestration layer services maintain process logic that refers to mapping supported systems' processes into set of rules and sequences to be followed and maintained. Sequences are determined by supported system processes.

### 5.4 Presentation Layer

Presentation layer presents system's direct contact and interactions with users. Each user category shall have a separate portal that provides different functionalities. Process logic shall not be embedded within presentation layer. Interface design and implementation shall be separated from process logic. This separation has proven many advantages for business architectures and is recognized as a recommended design pattern. Portals provide flexibility to add new services and remove existing services from the interface upon need. Portal consumes Web services, and provides means of user interactions with the system either via displaying results or acquiring inputs.

## 6. Summary

Service Oriented Architecture is the software architecture that achieved all systems functional and non-functional requirements. This chapter presented:

- Intensive SOA overview
- SOA characteristics and advantages
- SOA enablers, like CORBA, Software Agents, and Web services
- Web services as main SOA enabler, advantages and key features of Web services

# Chapter 3

# Chapter Three

# PROPOSED UMIS USING SOA

## 1. Introduction

LMS can not achieve UMIS functionalities, and vice versa. University needs both LMS and UMIS, so, proposed service based architecture can not ignore neither of them. One of thesis's objectives is to present major components of a service based UMIS and LMS. This chapter will present proposed architecture of selected UMIS components based on both the surveys, and university requirements.

## 2. Proposed Architecture Characteristics

### 2.1 Layered Architecture

Figure 3.1 depicts guidelines of proposed service based architecture which is divided into four layers: data, application services, process logic, and interface layer. Interface layer utilizes process logic layer, which consumes application services layer that utilizes data layer.

#### 2.1.1 Data Layer

Implementation architecture consumes one Microsoft SQL Server 2005 database server that holds the data files and relevant stored procedures that reside beside data files on the same database server. Usage of stored procedures to act as a data layer provides many advantages on regarding performance and security. Though usage of stored procedures might be an overload on the database server, caching is the solution that enhances performance. From a security perspective, it is not a recommended practice to implicitly implement SQL statements in the Web services. Beside, stored procedures allow reusability on data level.

#### 2.1.2 Application Services Layer

A Microsoft Windows 2003 Server that has .net framework 2.0 installed satisfies the software requirements. Application services layer contains data services, which are Web services that directly utilize stored procedures for data access. This layer holds external consumed services. Internet is the connection medium between LMS and external services. Proposed LMS implement direct connection between services.

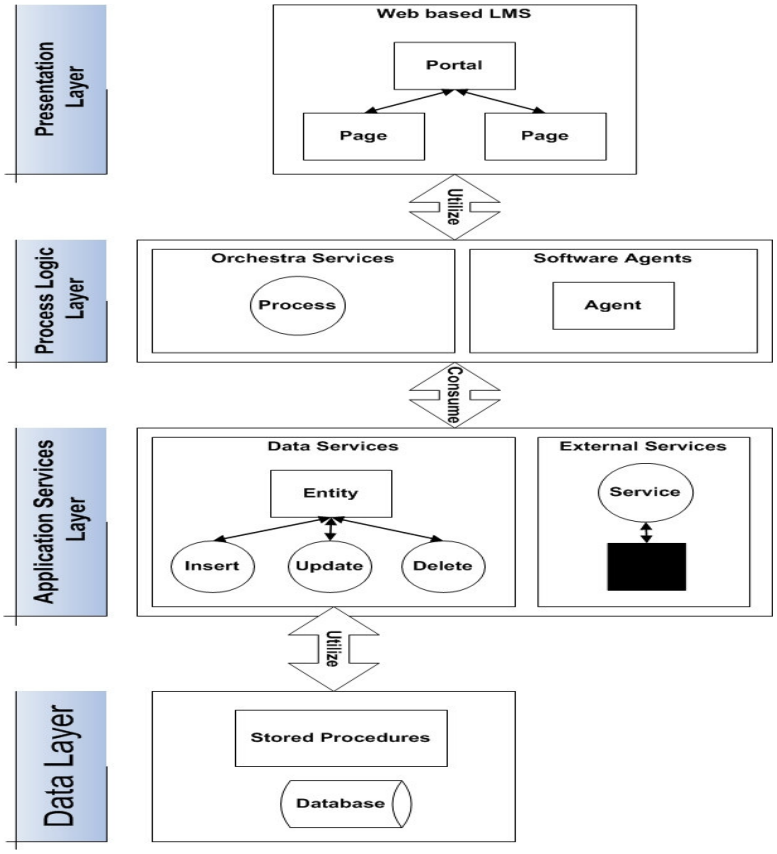


**Figure 3.1: Proposed Learning Management System Layered Architecture**

### 2.1.3 Process Logic Layer

Process logic layer holds orchestra Web services and software agents that reflect processes supported by the LMS. Orchestras and software agents consume application services layer and external Web services. Orchestras are implemented as Web services, and software agents are implemented as windows services that is always running and executing.

### 2.1.4 Presentation Layer

Portals are presented for human interaction with LMS. As depicted separately later in interface design section, portals present the capability to perform main functionalities in easy direct manner that reflects LMS services composability.

## 2.2 Development Framework

.Net framework is an emerging Microsoft technology that enables execution of any .net compiled code on any machine that has .net framework installed. Machines include: PDA; personal digital assistants, Tablet PCs, Smart Phones, and Computers. .Net framework is not a software developing tool, neither a programming language. .Net framework supports more than 30 programming languages, including Delphi, Visual

Java, and Visual C++. Choosing .net framework was affected by two other factors: efficiency of Microsoft SQL Server 2005, and the seamless integration between SQL Server and .net framework based programming languages. .Net framework deals with standards efficiently.

### 3. UMIS Proposed Components

Selected UMIS components to be analyzed, designed, implemented, and evaluated are: Student Information System (SIS), Library Information System (LIS), and Faculty Information System. SIS will hold students data, registration, and enrollment details. LIS will present the automation of the library traditional system without the availability of digital contents. Faculty Information System is the information system that holds faculties, and staff members' data.

### 4. Faculty Information System

Faculty Information System can be thought of as a collection of information systems that manage entities depicted in figure 3.2. Instructors and Employees can be further analyzed into categories as presented in table 3.1. Automating and managing the different activities of mentioned departments is a must for integrating different components of faculty within learning activities.

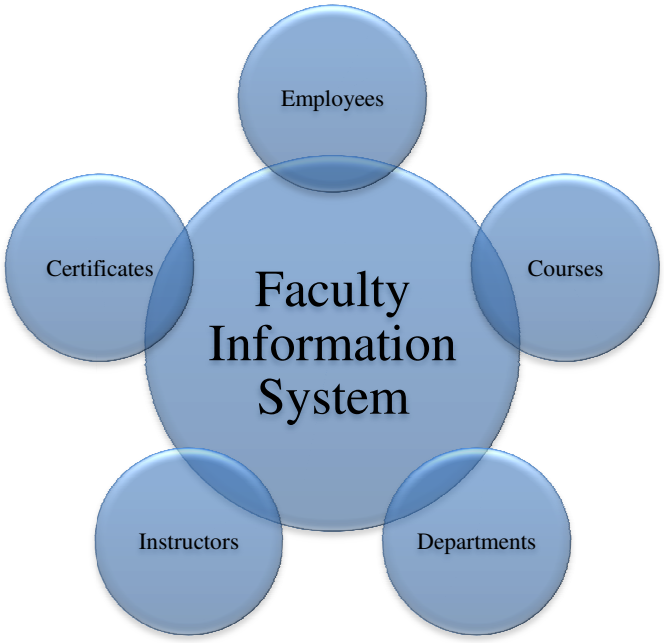


**Figure 3.2: Faculty Information System Entities**

**Table 3.1: Instructors and Employees Categories**

| Instructors | Employees |
|---|---|
| Dean | e-Learning Center |
| Vice Dean | Youth Activities |

| | |
|---|---|
| Heads of Departments | Dean Secretary |
| Doctors | Department Secretary |
| Assistant Teachers | Legal Issues Department |
| Teaching Assistants | Head of Employees Secretary |
| | Stuff Members Secretary |
| | Financial Department |
| | Head of Employees |
| | Import and Export |
| | Post-Graduate Studies |
| | Student Affairs |
| | Inventory |
| | Employee Affairs |
| | Library |

## 5. Student Affairs Management System (SIS)

SIS is the information system responsible for managing student data and all student related activities. Student Affairs is one of the departments that exist in all faculties in all universities. Student Information System reflects the processes initiated, managed, and maintained at that department.

### 5.1 Analysis of SIS

Figure 3.3 presents the main activities at Student Affairs Department.

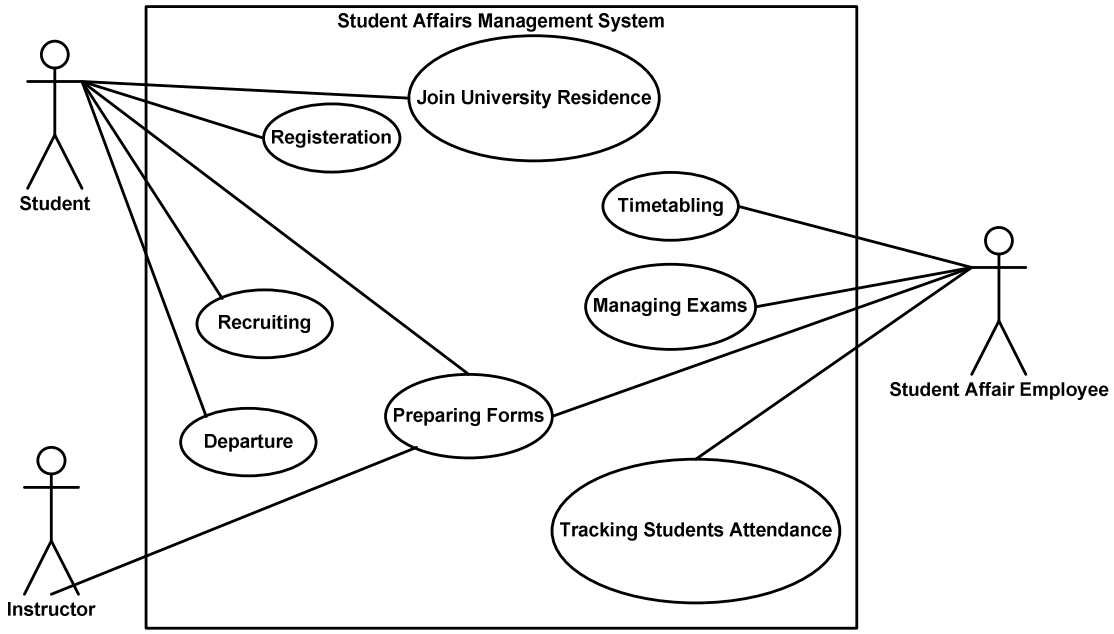


**Figure 3.3: SIS Use Case**

There are eight major processes at student affairs department:

- **Registration:** Students register in the faculty by fulfilling the required forms.
- **Join University Residence:** Students can join the university residence if they meet the predetermined requirements.
- **Recruiting:** Student Affairs is responsible for managing the army forces papers of student till they reach 28 years old or finish studying.
- **Time Tabling:** Student Affairs manages the term time table trying to satisfy almost all stuff members and students requirements.
- **Departure:** Students can submit a departure form incase they want to join another Faculty / University. This activity requires integration among different Student Affairs Management Systems to make this process globally available.
- **Tracking Students Attendance:** in practical faculties, students must attend lectures and labs to achieve educational goals. Student Affairs must keep track of student attendance and present attending report status available upon request by dean and / or instructor. Decisions regarding the way to punish the student that exceeds the maximum limit of absence are taken by dean and / or instructor.
- **Managing Exams:** involve the routine activities of preparing the disks, seat numbers, matching unique student IDs with seat numbers, preparing the answer sheets, announcing the exam schedule, delivering the seat numbers, and arranging the exam benches.
- **Preparing Forms:** Instructors and Students can request several different kinds of forms available at every Student Affairs department. Forms need to be automated, and composing data need to be stored in the database.

**Registration Analysis and Join University Residence**

Analysis of both processes shows that they share the same steps so; they are abstracted to registration operation as depicted in figure 3.4.

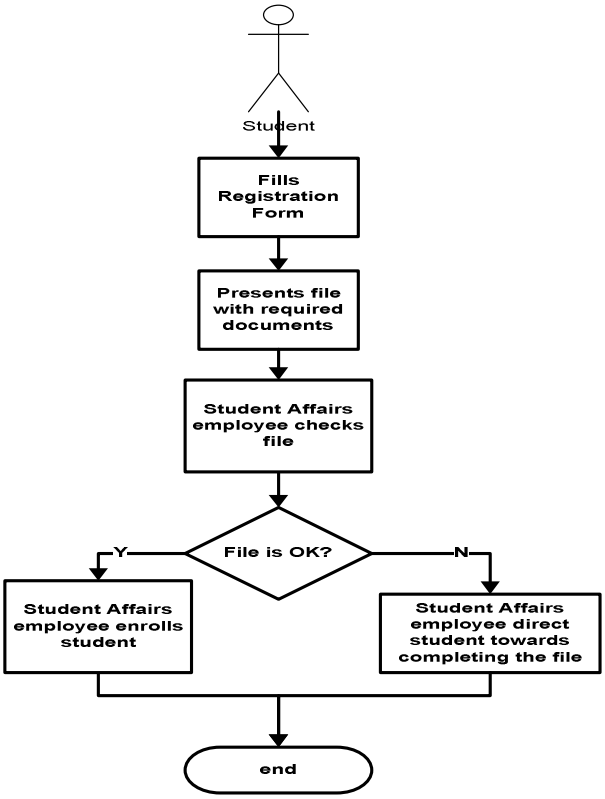


**Figure 3.4: Registration Analysis**

### Recruiting

Recruiting is one of the processes that can be split into two others: the first one is invoked by Student, and ends where the Student Affairs Employee starts the second one; the process that is achieved by him. Figure 3.5 presents the Student-Recruiting process, and figure 3.6 shows the Student Affairs Employee-Recruiting one.

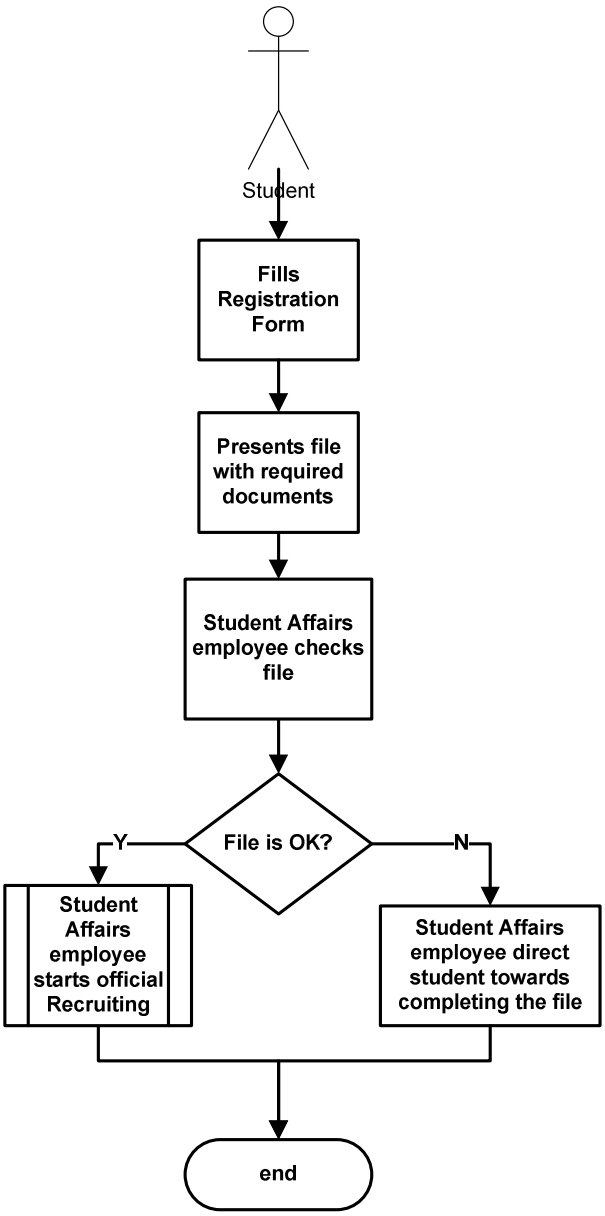


**Figure 3.5: Student – Recruiting Analysis**

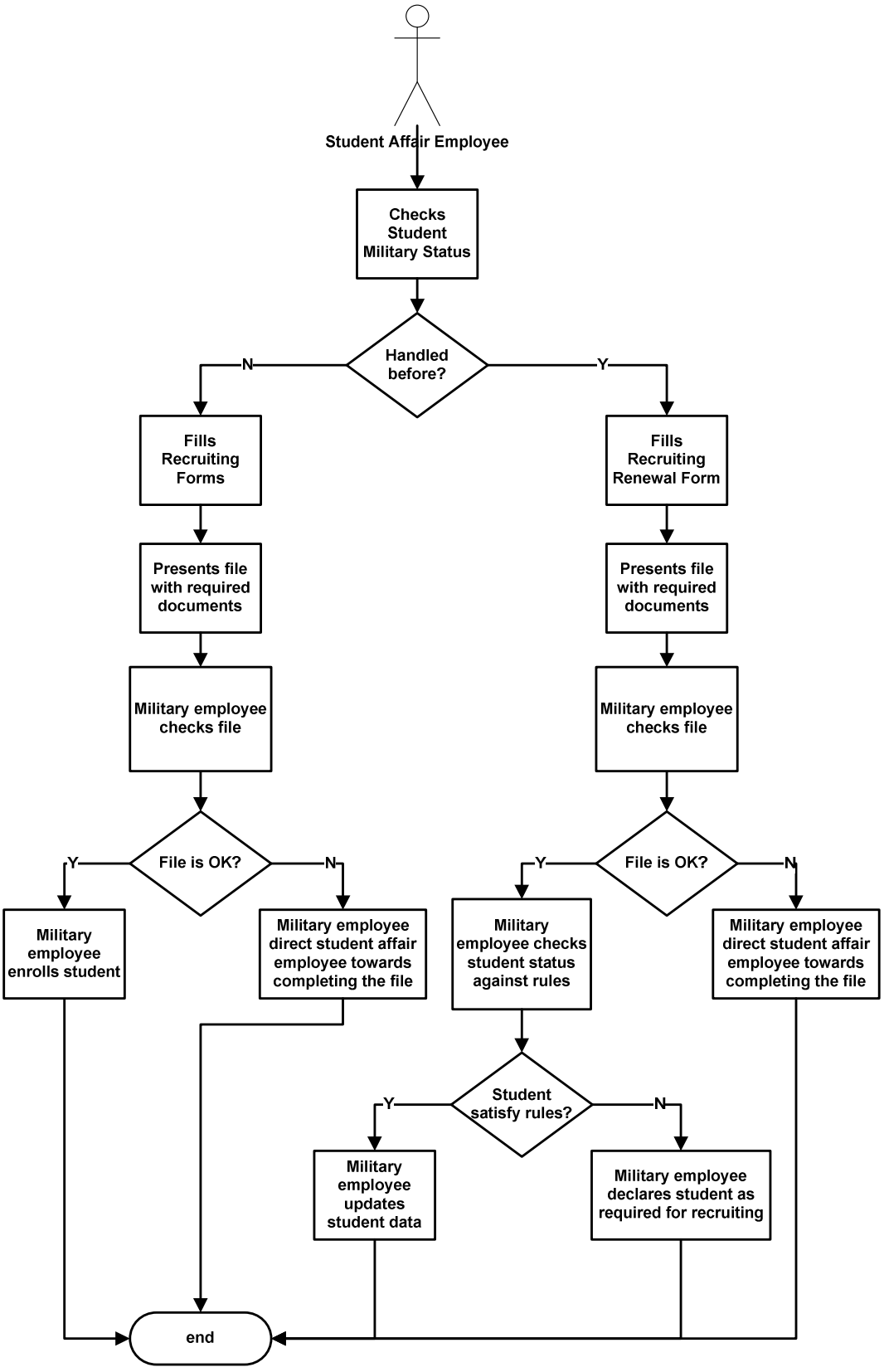


**Figure 3.6: Student Affairs Employee-Recruiting**

### Time Tabling

Time Tabling process is initiated by any Student Affairs Employee and it is the process that tends to make the faculty time table ready. It consumes the activities that are depicted in figure 3.7.

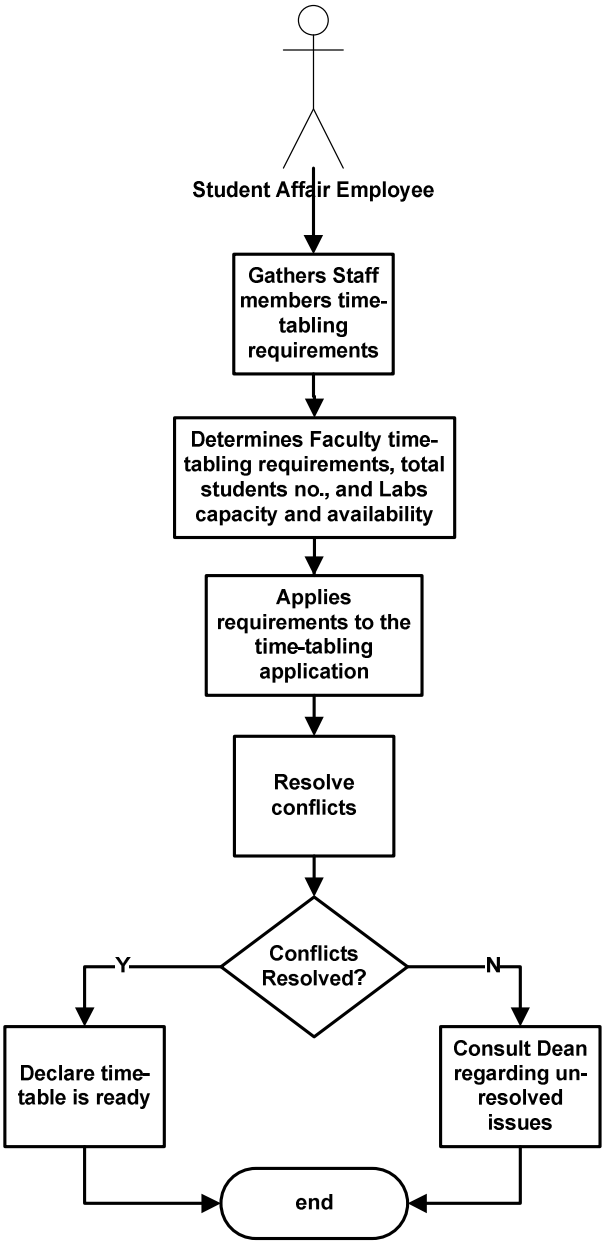


**Figure 3.7: Time-Tabling Analysis**

### Departure

Students can sign out from the faculty and head to another faculty. According to the status of the student, the place s/he is heading to, different situations can take place. Figure 3.8 shows the analysis of the student departure process.

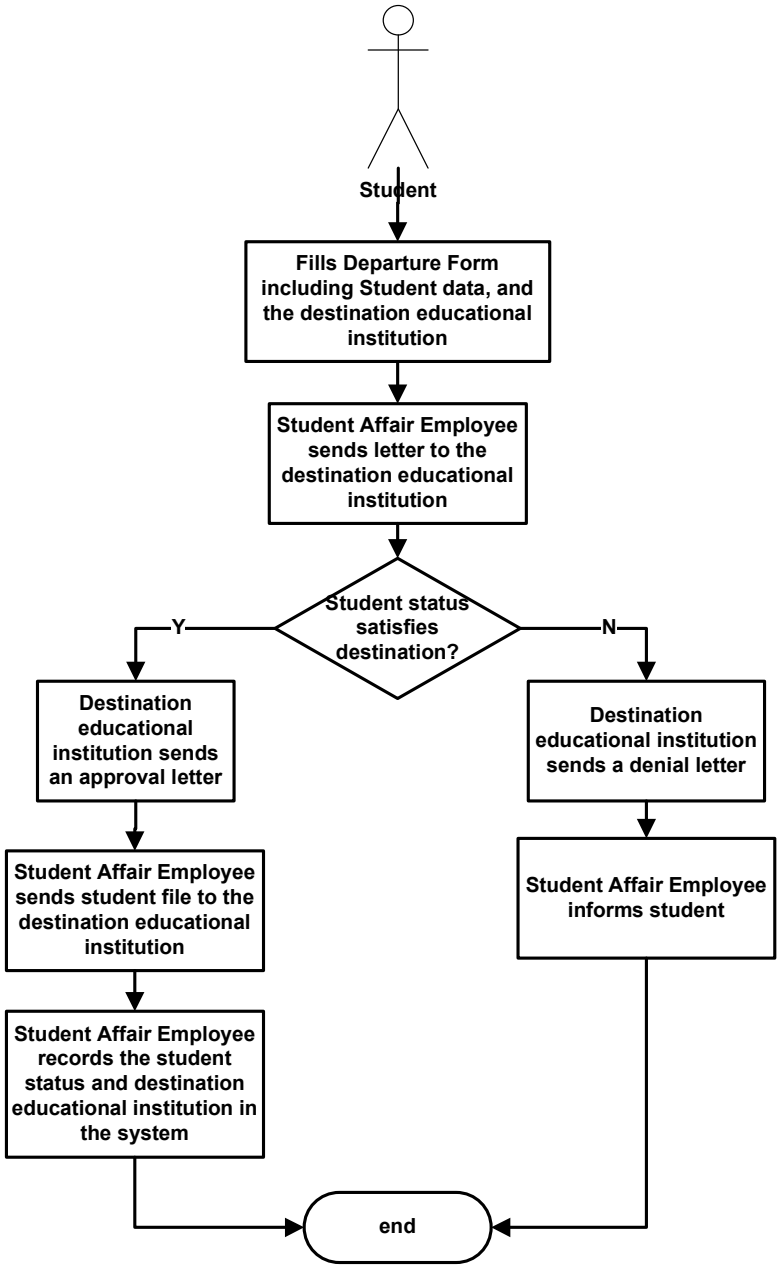


**Figure 3.8: Departure Analysis**

### Tracking Student Attendance

Faculties should keep track of student attendance in order to guarantee a healthy educational process. Tracking Student Attendance can be done by Student Affair Employee as shown in figure 3.9.

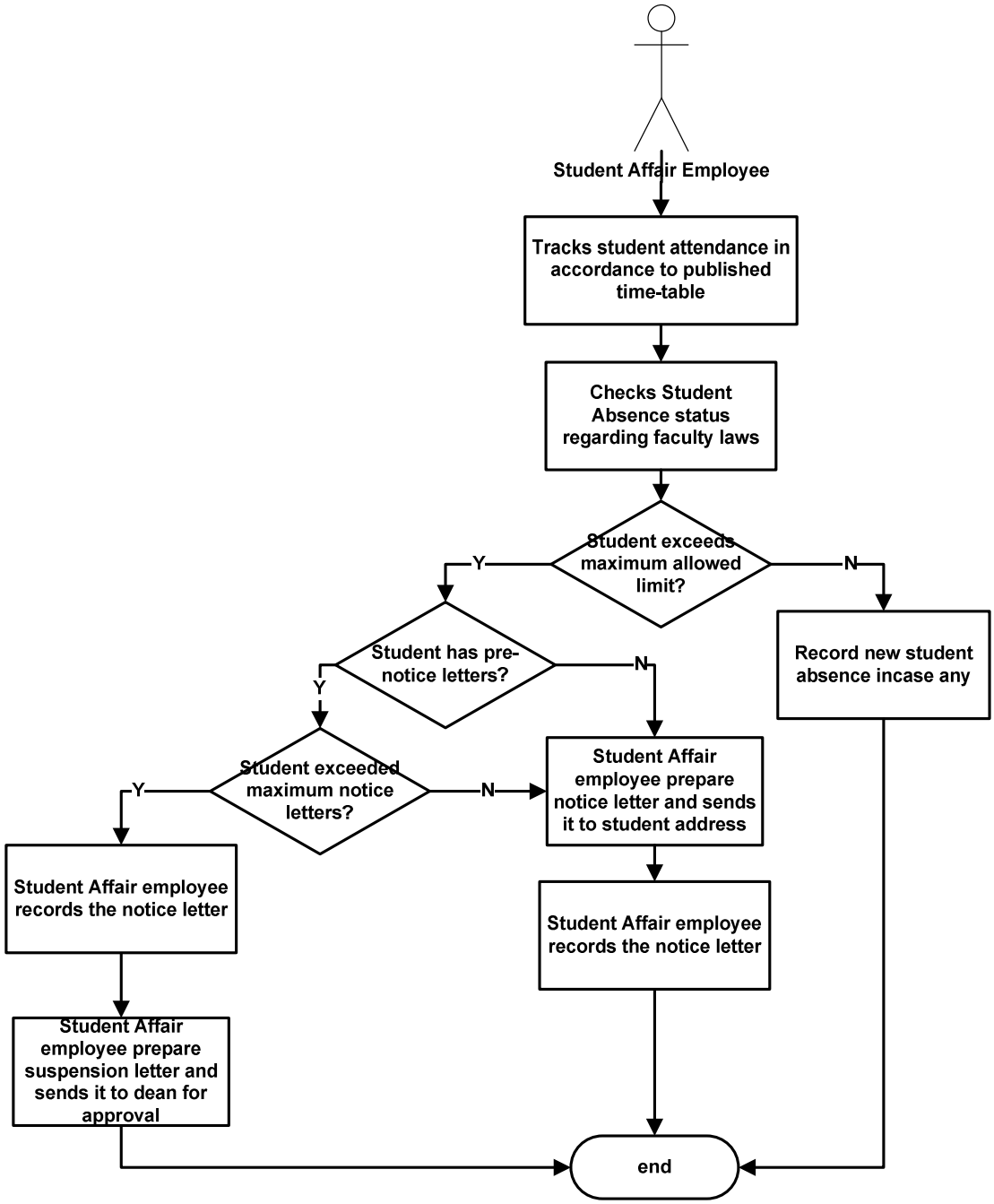


**Figure 3.9: Track Student Attendance Analysis**

### Managing Exams

From managerial point of view, exam involves activities that should be good taken care of as depicted in figure 3.10.

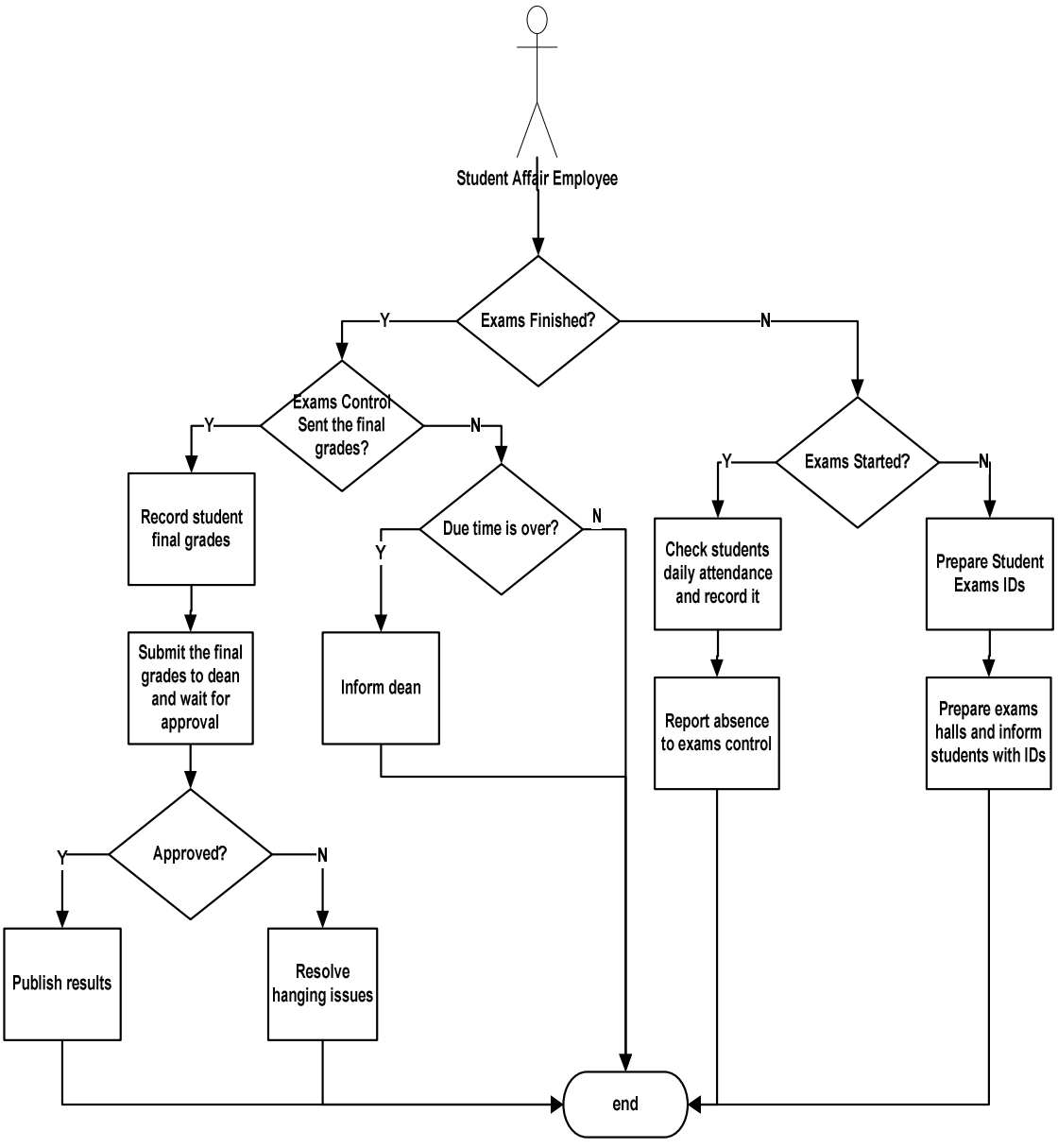


**Figure 3.10: Manage Exam Analysis**

### Preparing Different Forms

Students and / or Instructors can request certain forms to be able to present them to other organizations. Student Affairs Employee to prepare the form as shown in figure 3.11.

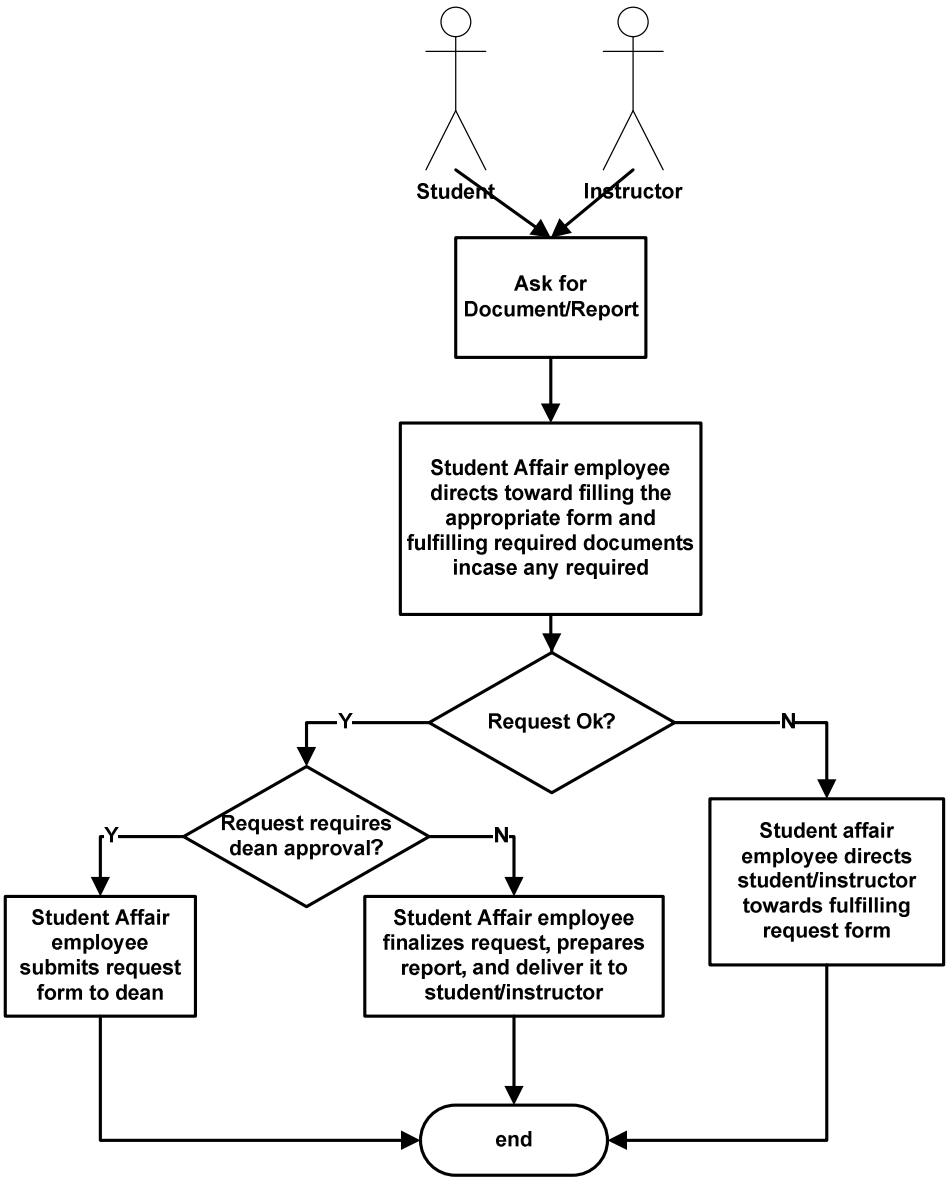


**Figure 3.11: Prepare Different Forms**

### 5.2 Design of SIS

Figure 3.12 presents the required Student Affairs entities based on analyzed presented system processes. Figure 3.13 depicts the Entity-Relationship (E-R) diagram of the system.

### 5.2.1 Proposed SIS Architecture

Figure 3.14 shows the proposed Services Based SIS on two parts, part (a) presents services related to employee, student, CV, instructor, and course managers. Part (b) addresses certification, e, reference, department, faculty, and questions managers. Entity Centric based Services are designed to reflect system entities.

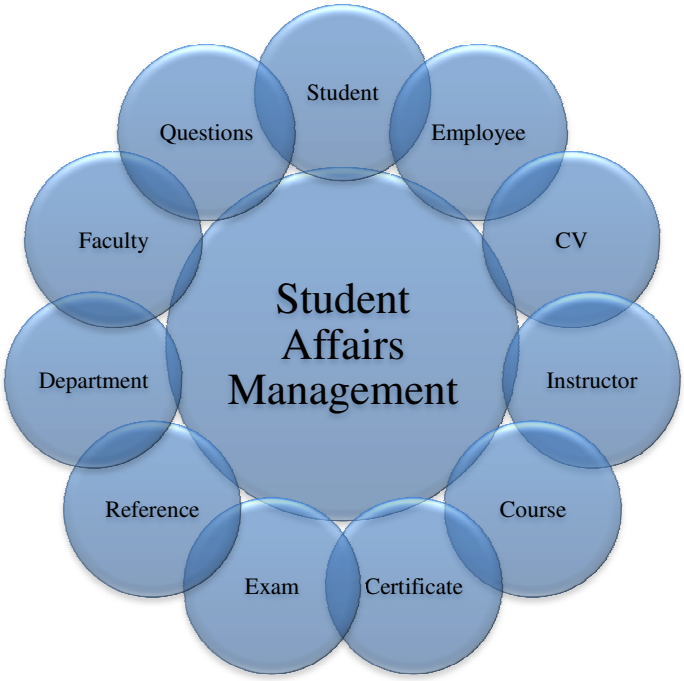


**Figure 3.12: SIS Entities**

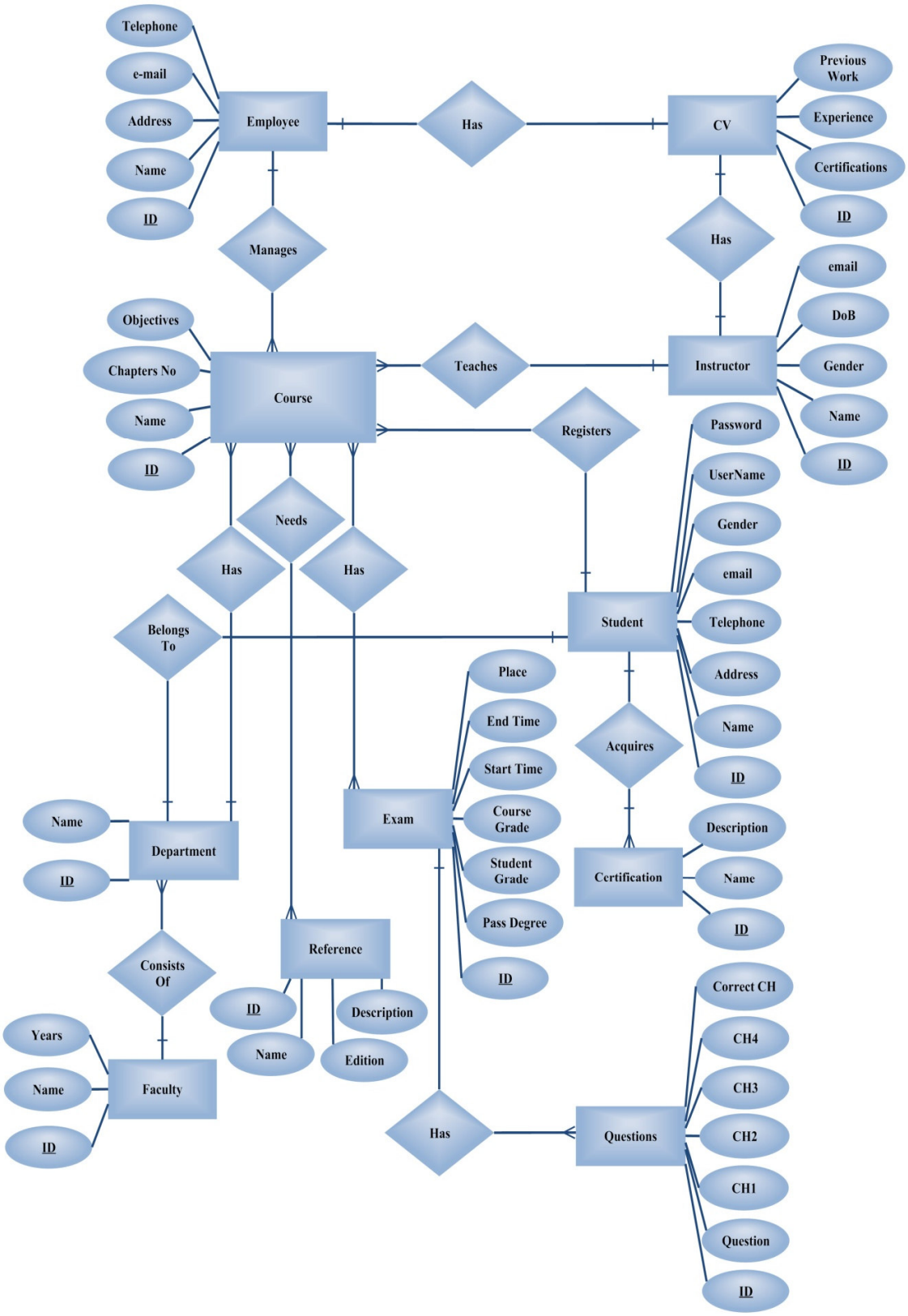


**Figure 3.13: SIS Entity-Relationship (E-R) Diagram**

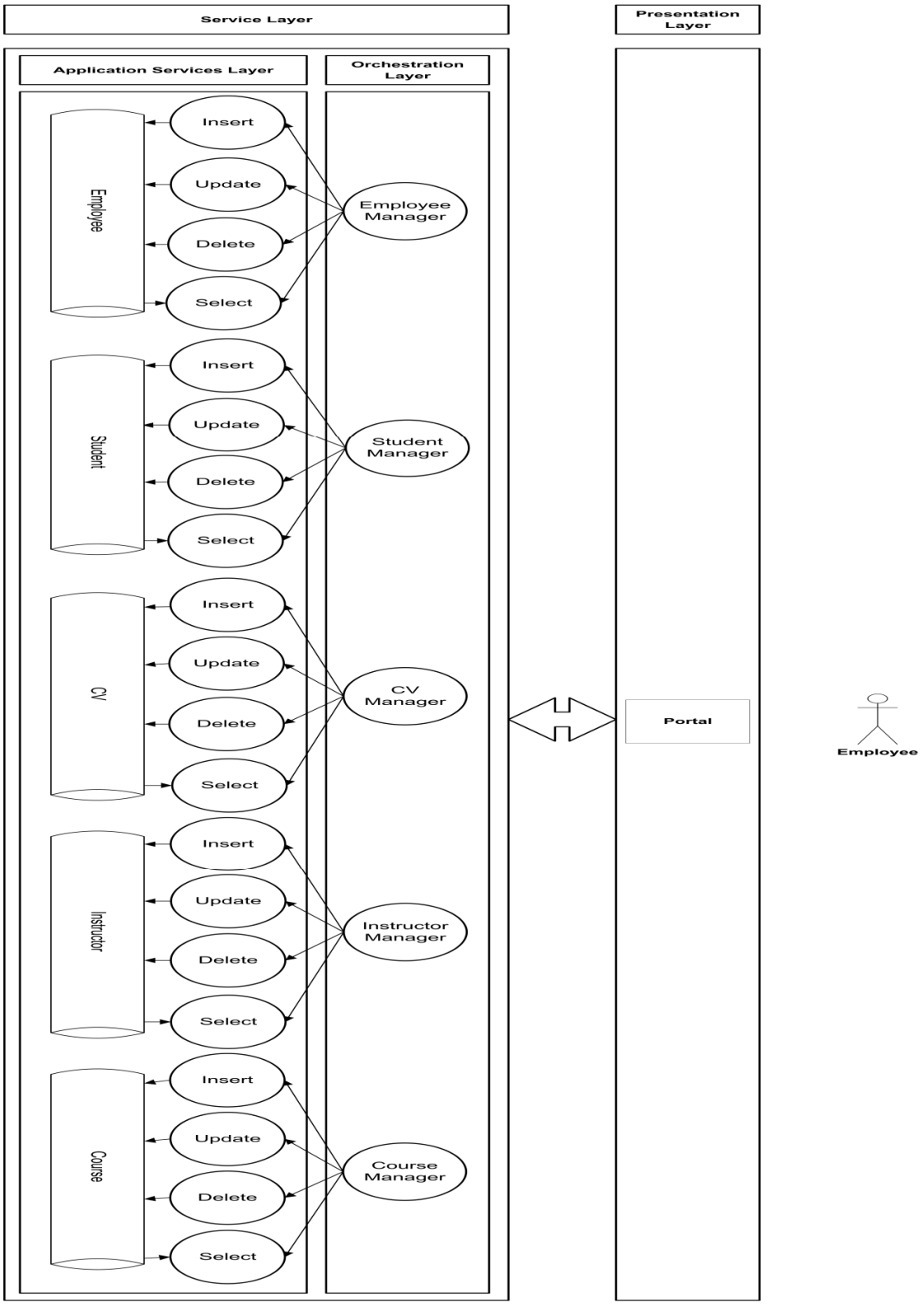


**Figure 3.14 - Part a: SIS Architecture**

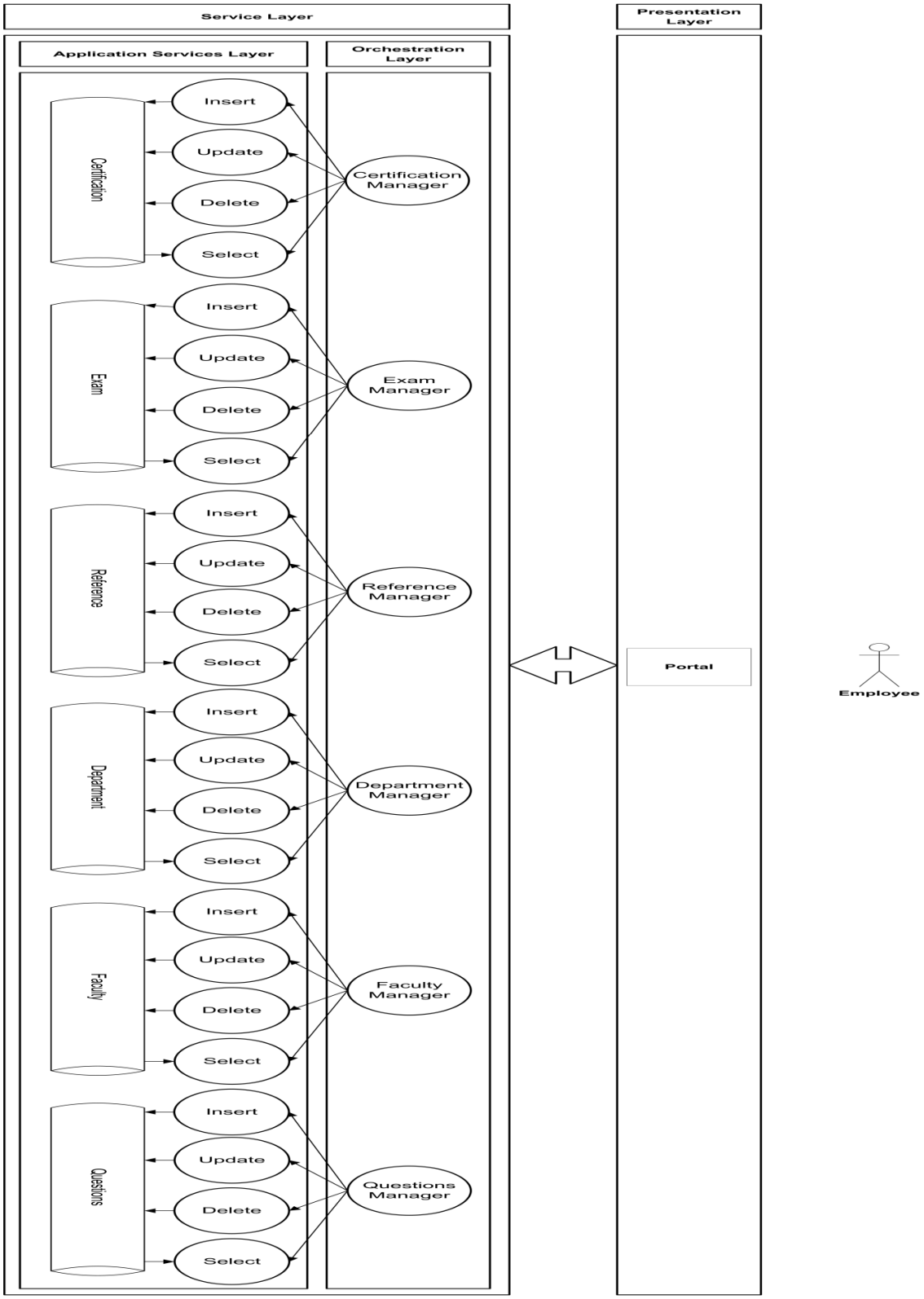


**Figure 3.14 - Part b: SIS Architecture**

### 5.2.2 Implement Database Tables

Figure 3.15 presents the implemented Student Affairs Management System database tables. Database tables include: instructor, course, administrator, administrator CV, exam, reference, questions, student, student courses, faculty, department, certificates, and student certificates.

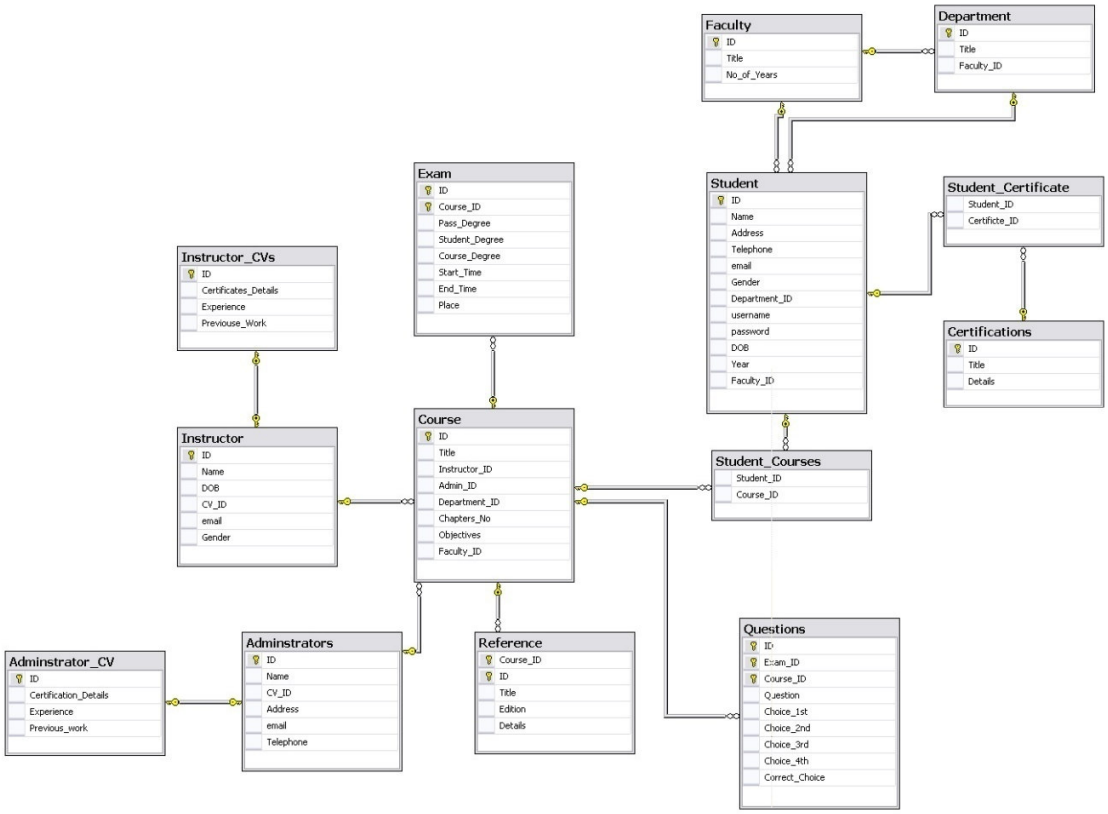


**Figure 3.15: Student Affairs Management System Implemented Database Tables**

### 5.2.3 Implement Stored Procedures

Table 3.2 presents a list of implemented stored procedures and parameters.

**Table 3.2: Student Information System Stored Procedures**

| Student Information System Stored Procedures | |
|---|---|
| **Stored Procedure Name** | **Input** |
| **Employee** | |
| Delete Employee | Employee ID [int] |
| Insert Employee | Employee ID [int]<br>Name [Varchar (30)]<br>CV ID [int]<br>Address [Varchar (30)]<br>email [Varchar (30)]<br>Telephone [Varchar (20)] |

| | |
|---|---|
| Exact Search By Name | Employee Name [Varchar (30)] |
| Partial Search By Name | Employee Name [Varchar (30)] |
| Select By ID | Employee ID [int] |
| Select All | |
| Update Employee | Employee ID [int]<br>Name [Varchar (30)]<br>CV ID [int]<br>Address [Varchar (30)]<br>email [Varchar (30)]<br>Telephone [Varchar (20)] |
| **Employee CV** | |
| Delete Employee CV | Employee CV ID [int] |
| Insert Employee CV | CV ID [int]<br>Certification Details [Varchar (50)]<br>Experiences [Varchar (50)]<br>Previous Work [Varchar (30)] |
| Select By ID | Employee CV ID [int] |
| Select All | |
| Update Employee CV | CV ID [int]<br>Certification Details [Varchar (50)]<br>Experiences [Varchar (50)]<br>Previous Work [Varchar (30)] |
| **Certification** | |
| Delete Certification | Certification ID [int] |
| Insert Certification | Certificate ID [int]<br>Name [Varchar (50)]<br>Details [Varchar (50)] |
| Select By ID | Certification ID [int] |
| Select All | |
| Update Certification | Certificate ID [int]<br>Name [Varchar (50)]<br>Details [Varchar (50)] |
| **Course** | |
| Delete Course | Course ID [int] |
| Insert Course | Course ID [int]<br>Name [Varchar (30)]<br>Instructor ID [int]<br>Employee ID [int]<br>Department ID [int]<br>Chapters NO [int]<br>Objectives [Varchar (50)]<br>Faculty ID [int] |
| Exact Search By Name | Course Name [Varchar (30)] |
| Partial Search By Name | Course Name [Varchar (30)] |
| Select By ID | Course ID [int] |
| Select All | |
| Update Course | Course ID [int]<br>Name [Varchar (30)]<br>Instructor ID [int] |

| | |
|---|---|
| | Employee ID [int]<br>Department ID [int]<br>Chapters NO [int]<br>Objectives [Varchar (50)]<br>Faculty ID [int] |
| **Department** | |
| Delete Department | Faculty ID [int]<br>Department ID [int] |
| Insert Department | Department ID [int]<br>Name [Varchar (30)]<br>Faculty ID [int] |
| Select By ID | Faculty ID [int]<br>Department ID [int] |
| Select All | |
| Update Department | Department ID [int]<br>Name [Varchar (30)]<br>Faculty ID [int] |
| **Exam** | |
| Delete Exam | Exam ID [int]<br>Course ID [int] |
| Insert Exam | Exam ID [int]<br>Course ID [int]<br>Pass Degree [int]<br>Student Degree [int]<br>Course Degree [int]<br>Start Time [Varchar (20)]<br>End Time [Varchar (20)]<br>Place [Varchar (30)] |
| Select By ID | Exam ID [int]<br>Course ID [int] |
| Select All | |
| Update Exam | Exam ID [int]<br>Course ID [int]<br>Pass Degree [int]<br>Student Degree [int]<br>Course Degree [int]<br>Start Time [Varchar (20)]<br>End Time [Varchar (20)]<br>Place [Varchar (30)] |
| **Faculty** | |
| Delete Faculty | Faculty ID [int] |
| Insert Faculty | Faculty ID [int]<br>Name [Varchar (50)]<br>Years [int] |
| Select By ID | Faculty ID [int] |
| Select All | |
| Update Faculty | Faculty ID [int]<br>Name [Varchar (50)]<br>Years [int] |

| **Instructor** | |
|---|---|
| Delete Instructor | Instructor ID [int] |
| Insert Instructor | Instructor ID [int]<br>Name [Varchar(50)]<br>DOB [varchar (20)]<br>Instructor CV ID [int]<br>email [Varchar (50)]<br>Gender [Char (10)] |
| Exact Search By Name | Instructor Name [Varchar (30)] |
| Partial Search By Name | Instructor Name [Varchar (30)] |
| Select By ID | Instructor ID [int] |
| Select All | |
| Update Instructor | Instructor ID [int]<br>Name [Varchar(50)]<br>DOB [varchar (20)]<br>Instructor CV ID [int]<br>email [Varchar (50)]<br>Gender [Char (10)] |
| **Instructor CV** | |
| Delete Instructor CV | Instructor CV ID [int] |
| Insert Instructor CV | Instructor CV ID [int]<br>Certification Details [Varchar (50)]<br>Experience [Varchar (50)]<br>Previouse Work [Varchar (50)] |
| Select By ID | Instructor CV ID [int] |
| Select All | |
| Update Instructor CV | Instructor CV ID [int]<br>Certification Details [Varchar (50)]<br>Experience [Varchar (50)]<br>Previouse Work [Varchar (50)] |
| **Questions** | |
| Delete Question | Course ID [int]<br>Exam ID [int]<br>Question ID [int] |
| Insert Question | Question ID [int]<br>Exam ID [int]<br>Course ID [int]<br>Question [Varchar (50)]<br>Choice1 [Varchar (30)]<br>Choice2 [Varchar (30)]<br>Choice3 [Varchar (30)]<br>Choice4 [Varchar (30)]<br>Correct Choice [Varchar (30)] |
| Select By ID | Question ID [int]<br>Course ID [int]<br>Exam ID [int] |
| Select All | |
| Update Question | Question ID [int]<br>Exam ID [int] |

| | |
|---|---|
| | Course ID [int]<br>Question [Varchar (50)]<br>Choice1 [Varchar (30)]<br>Choice2 [Varchar (30)]<br>Choice3 [Varchar (30)]<br>Choice4 [Varchar (30)]<br>Correct Choice [Varchar (30)] |
| **Reference** | |
| Delete Reference | Reference ID [int]<br>Course ID [int] |
| Insert Reference | Course ID [int]<br>Reference ID [int]<br>Name [Varchar (30)]<br>Edition [Varchar (30)]<br>Description [Varchar (50)] |
| Select All | Course ID [int] |
| Select By ID | Reference ID [int]<br>Course ID [int] |
| Update Reference | Course ID [int]<br>Reference ID [int]<br>Name [Varchar (30)]<br>Edition [Varchar (30)]<br>Description [Varchar (50)] |
| **Student** | |
| Delete Student | Student ID [int] |
| Insert Student | Student ID [int]<br>Name [Varchar (30)]<br>Address [Varchar (30)]<br>Telephone [Varchar (30)]<br>email [Varchar (30)]<br>Gender [Char (10)]<br>Department ID [int]<br>Username [Varchar (30)]<br>Password [Varchar (30)]<br>DOB [Varchar (30)]<br>Year [Char (10)]<br>Faculty ID [int] |
| Exact Search By Name | Student Name [Varchar (30)] |
| Partial Search By Name | Student Name [Varchar (30)] |
| Select All | |
| Select By ID | Faculty ID [int]<br>Department ID [int]<br>Student ID [int] |
| Update Student | Student ID [int]<br>Name [Varchar (30)]<br>Address [Varchar (30)]<br>Telephone [Varchar (30)]<br>email [Varchar (30)]<br>Gender [Char (10)] |

| | |
|---|---|
| | Department ID [int]<br>Username [Varchar (30)]<br>Password [Varchar (30)]<br>DOB [Varchar (30)]<br>Year [Char (10)]<br>Faculty ID [int] |
| **Student Certificates** | |
| Insert Student Certificate | Student ID [int]<br>Certificate ID [int] |
| Select All | |
| **Student Courses** | |
| Insert Student Course | Student ID [int]<br>Course ID [int] |
| Select All | |

### 5.2.4 Implement Classes

SIS class diagram include definition for the main classes and methods that form the system. Class diagram includes instructor, course, administrator, administrator CV, exam, reference, questions, student, student courses, faculty, department, certificates, and student certificates with the attributes highlighted in the E-R diagram and at least the four methods (insert, update, delete, select) presented in the proposed architecture.

## 6. Library Information System

LIS is responsible for automating library activities and has been widely known and accepted for more than forty years. LISs represents the main library automation functionality and almost share the same features and functionalities among them as presented in proposed LIS.

### 6.1 Analysis of LIS

Figure 3.16 presents LIS Use Case.

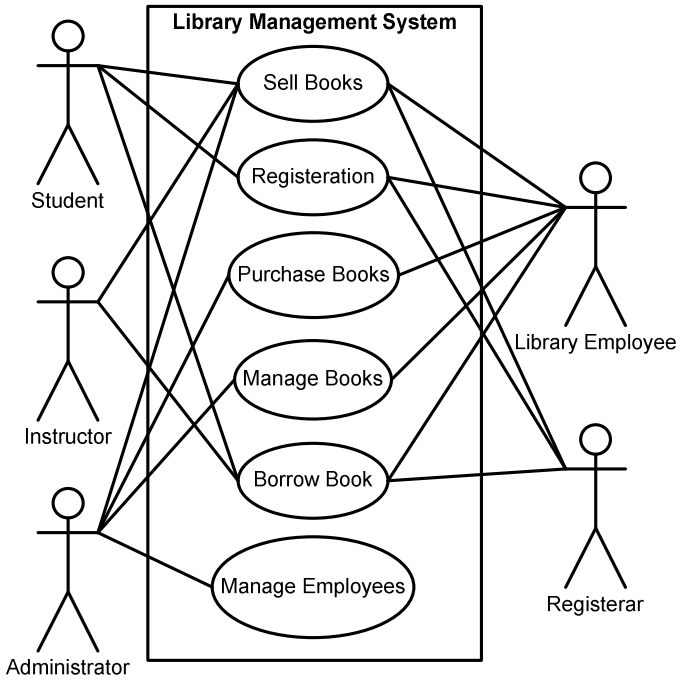


**Figure 3.16: LIS Use Case**

LIS Processes include:

- **Registration:** students, instructors, and outsider faculty members (registrars) can register for the library to have the ability to borrow and / or buy books.
- **Purchase Books:** invoked either to satisfy internal library requirements of newly released books or to satisfy clients' requests. Purchase Books stores data about Purchase Orders from the library to Suppliers, and Suppliers details.
- **Lend Books:** Instructors, Students, Employees, and Registrants can borrow one or more of the books available in the library. A record of Library Clients and Borrow activity details needs to be recorded.
- **Sell Books:** Library Clients can buy borrowed books, or request non-available books via special order.
- **Manage Library Employees Data:** involves the main operations of: Insert, Update, and Delete employees.
- **Manage Books Data:** considers managing and recording data about existing books in the library.

LIS automates the common activities of educational institutions libraries.

**Registration**

Registration to the library requires the steps shown in figure 3.17.

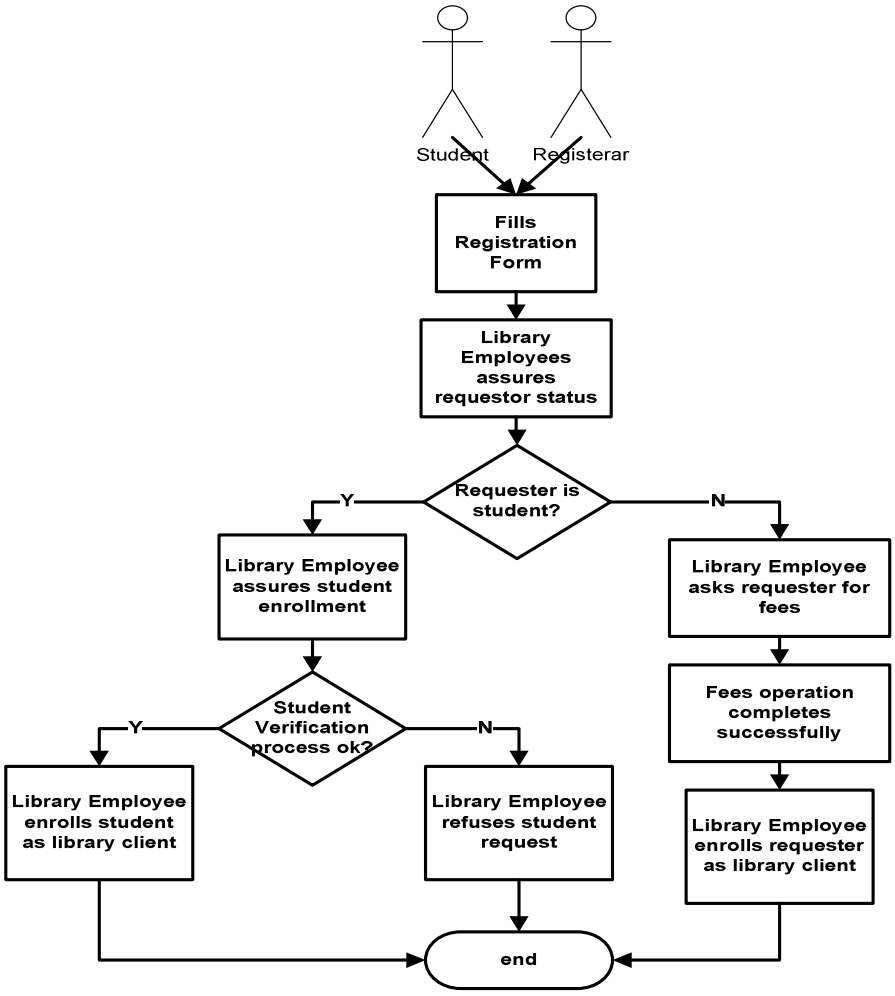


**Figure 3.17: Registration Analysis**

### Sell Books

Figure 3.18 shows the steps that instructor, student, or ordinary registrar has to go through to buy a book from the library.

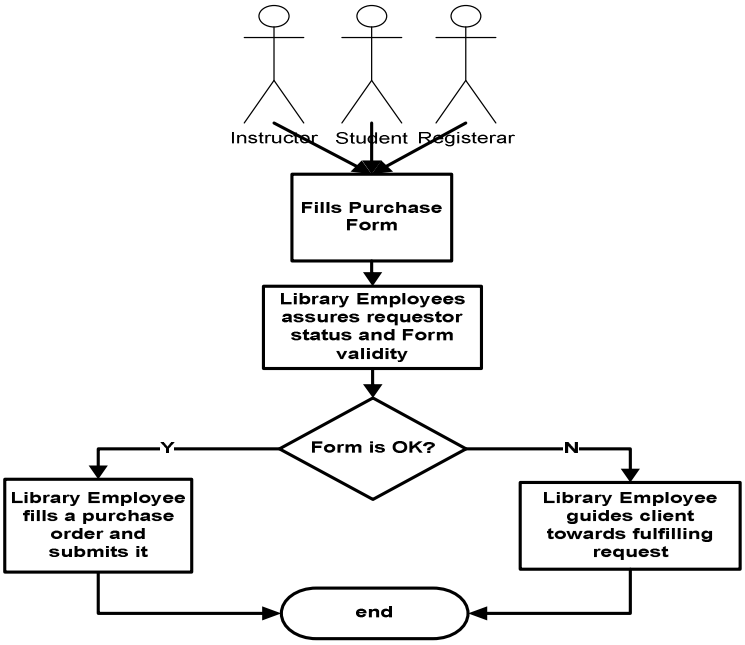


**Figure 3.18: Sell Books Analysis**

### Purchase Books

Purchase book activities is depicted in figure 3.19 where Library Employee purchases books from suppliers either to be available for loan by library or to be sold.

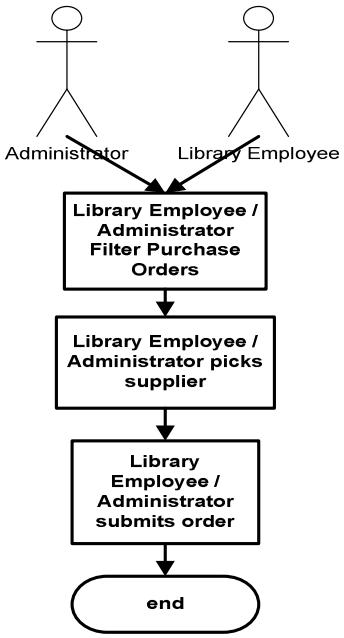


**Figure 3.19: Purchase Book Analysis**

### Borrow Books

Figure 3.20 shows the steps the borrower has to go through to borrow a book from the library.

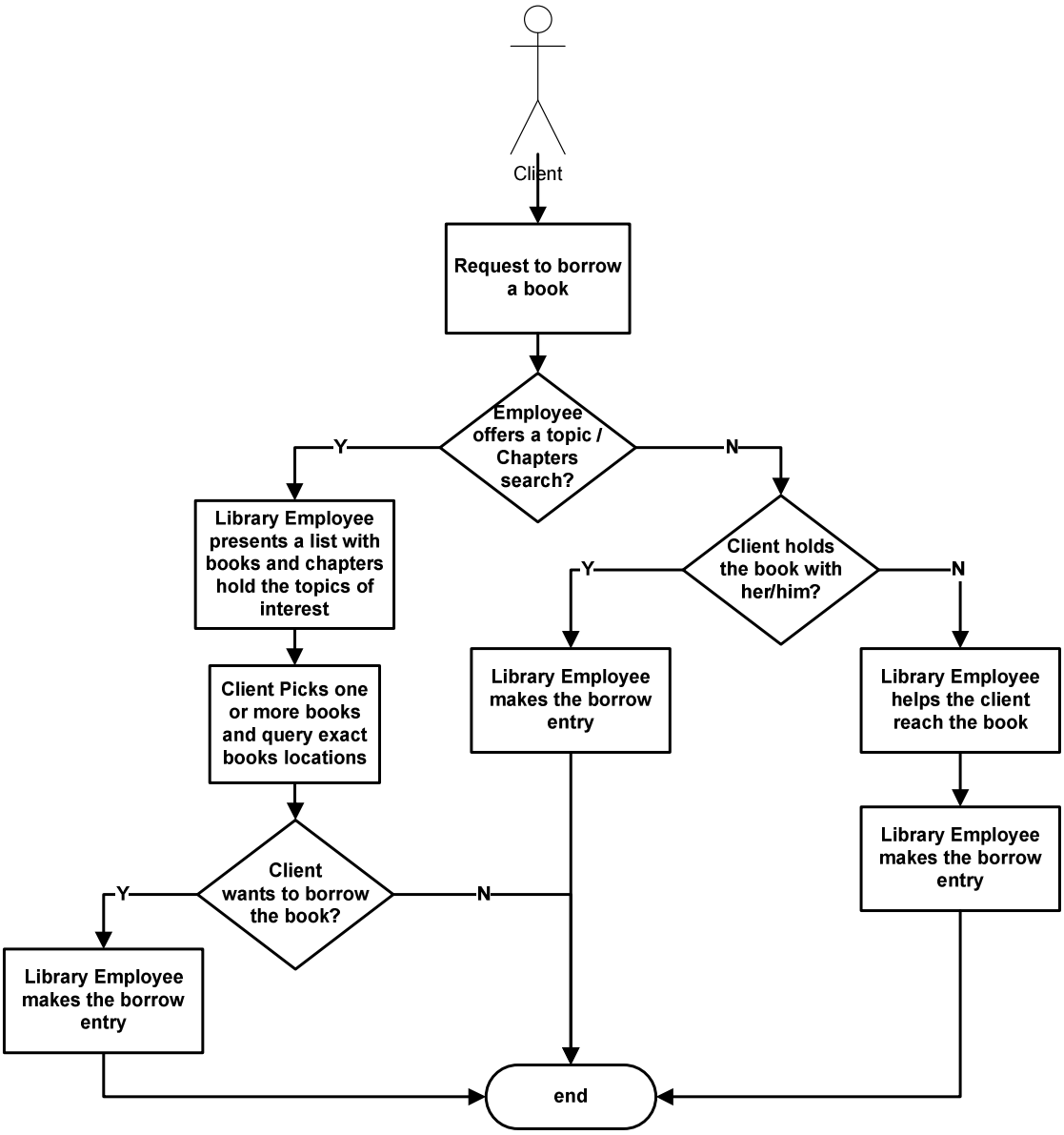


**Figure 3.20: Borrow Book**

## 6.2 Design of LIS

Further analysis of proposed processes implies system entities presented in figure 3.21. LIS Entity-Relationship diagram is depicted in figure 3.22.


**Figure 3.21: LIS Entities**

### 6.2.1 Proposed LIS Architecture

Figure 3.23 presents the proposed services based LIS architecture. The architecture consists of two layers: presentation, and services layer. Services layer holds two sub layers: orchestration and application services layer. Orchestration layer addresses entity centric services like employee, client, author, publisher, book, order, purchase order, and supplier managers that are responsible for consuming services of Insert, Update, Delete, and Retrieve the system entities presented in the application services layer. Presentation layer portal consumes the exposed Web services presented at services layer.

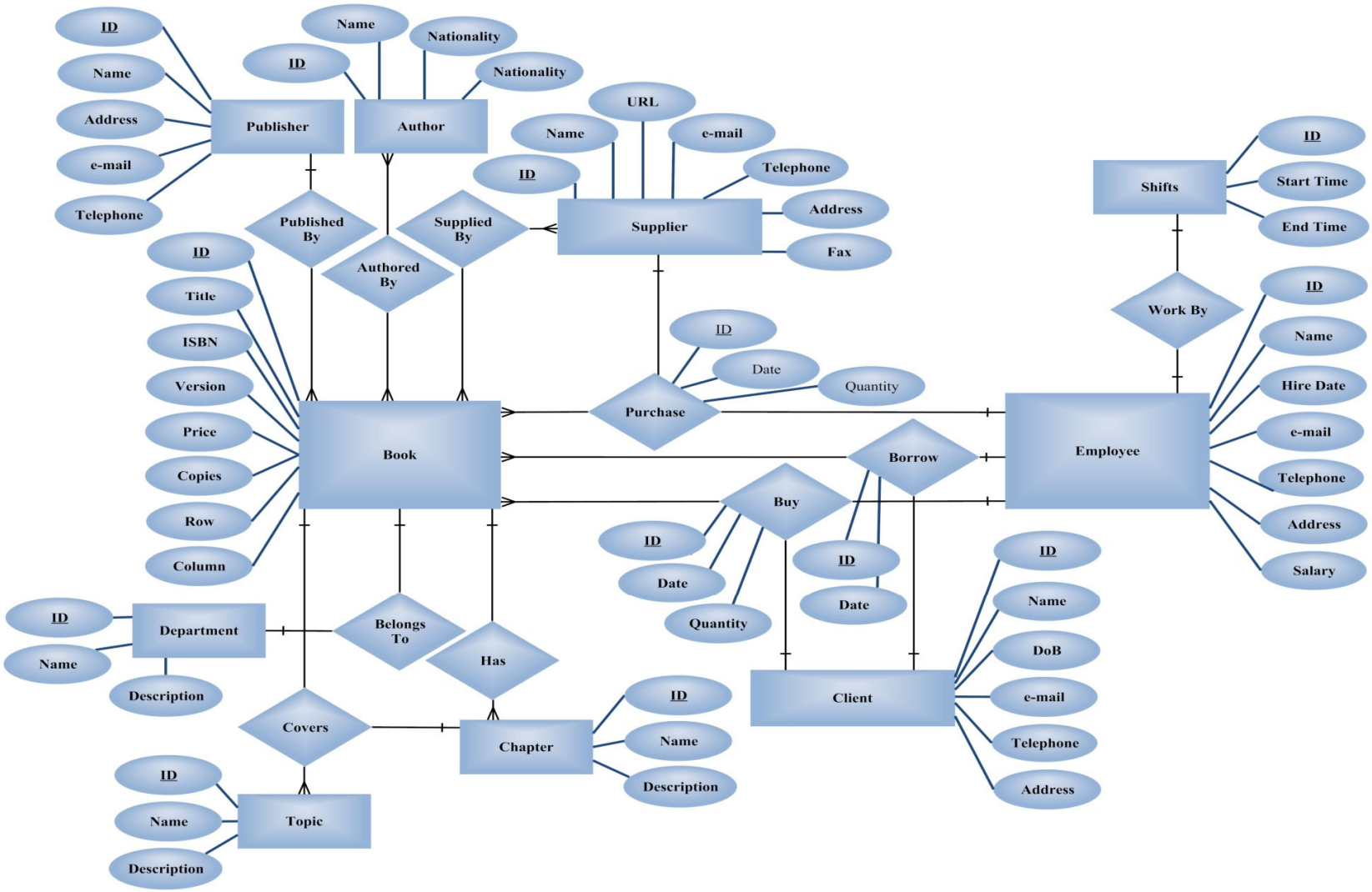


**Figure 3.22: LIS E-R Diagram**

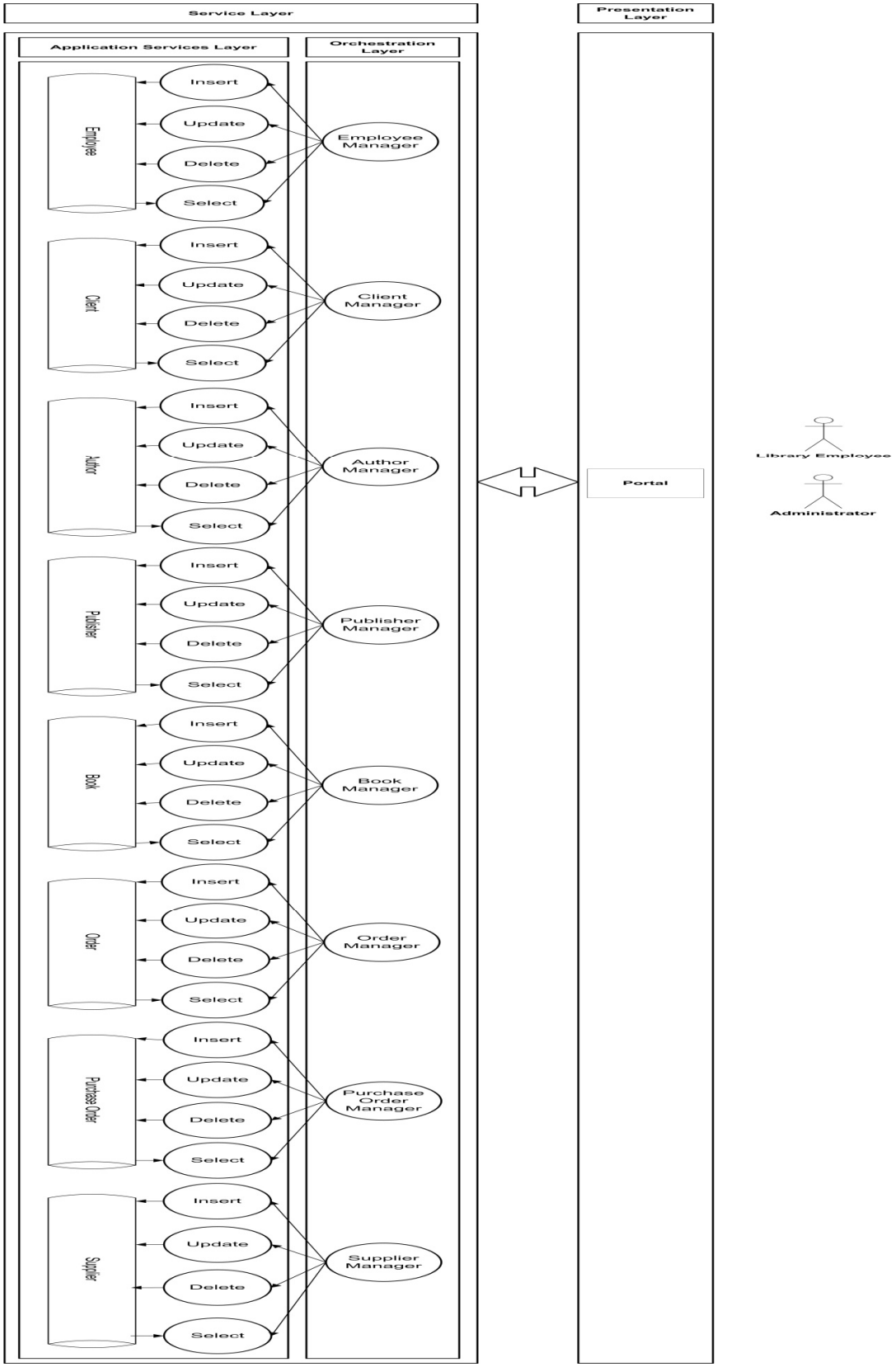


**Figure 3.23: LIS Proposed Architecture**

### 6.2.2 LIS Database Tables

Figure 3.24 presents the implemented LIS database tables. Database tables include: employee, shift, author, borrow, chapter, book, client, order, order details, publisher, topics, department, purchase order, purchase order details, and supplier.

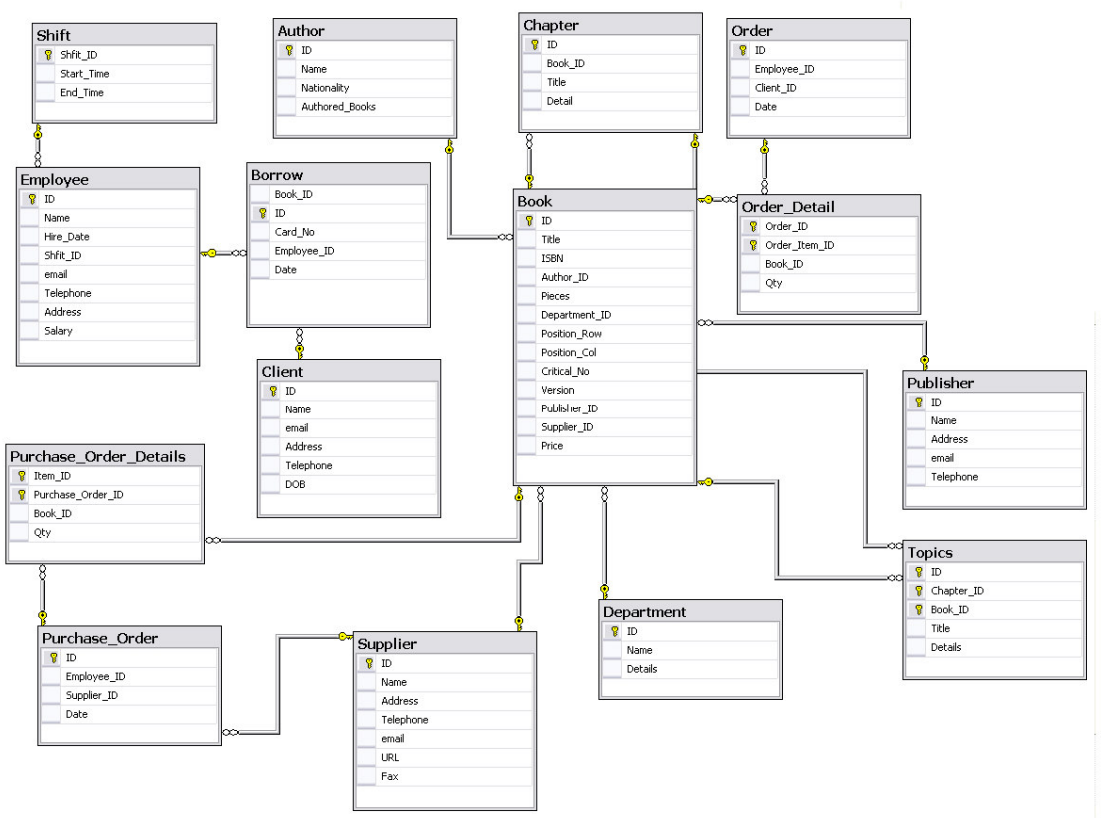


**Figure 3.24: LIS Database Tables**

### 6.2.3 LIS Stored Procedures

Table 3.3 lists the implemented LIS stored procedures and parameters of each one.

**Table 3.3: LIS Stored Procedures**

| LIS Stored Procedures | |
|---|---|
| **Stored Procedure Name** | **Input** |
| **Author** | |
| Delete Author | Author ID [int] |
| Insert Author | Author ID [int]<br>Author Name [Varchar (30)]<br>Nationality [Varchar (30)]<br>No. of Authored Books [int] |
| Exact Search Author Name | Author Name [Varchar (30)] |
| Partial Search Author Name | Author Name [Varchar (20)] |
| Display All Authors | |

| | |
|---|---|
| Display Author | Author ID [int] |
| Update Author | Author ID [int]<br>Author Name [Varchar (30)]<br>Nationality [Varchar (30)]<br>No. of Authored Books [int] |
| **Book** | |
| Delete Book | Book ID [int] |
| Insert Book | Book ID [int]<br>Book Name [Varchar (40)]<br>ISBN [Varchar(20)]<br>Author ID [int]<br>Price [int]<br>Department ID [int]<br>Row [int]<br>Column[int]<br>Critical No.[int]<br>Version [Varchar (8)]<br>Publisher ID [int]<br>Supplier ID [int] |
| Exact Search Book Name | Book Name [Varchar (40)] |
| Partial Search Book Name | Book Name [Varchar (40)] |
| Display All Books | |
| Display Book | Book ID [int] |
| Update Book | Book ID [int]<br>Book Name [Varchar (40)]<br>ISBN [Varchar(20)]<br>Author ID [int]<br>Price [int]<br>Department ID [int]<br>Row [int]<br>Column[int]<br>Critical No.[int]<br>Version [Varchar (8)]<br>Publisher ID [int]<br>Supplier ID [int] |
| **Borrow** | |
| Delete Borrow | Borrow ID [int] |
| Insert Borrow | Book ID [int]<br>Borrow ID [int]<br>Client ID [int]<br>Employee ID [int]<br>Date [Varchar (30)] |
| Display All Borrows | |
| Display Borrow | Borrow ID [int] |
| Update Borrow | Book ID [int]<br>Borrow ID [int]<br>Client ID [int]<br>Employee ID [int]<br>Date [Varchar (30)] |

| **Chapter** | |
|---|---|
| Delete Chapter | Book ID [int]<br>Chapter ID [int] |
| Insert Chapter | Chapter ID [int]<br>Book ID [int]<br>Name [Varchar (30)]<br>Description [varchar (40)] |
| Exact Search Chapter Name | Chapter Name [Varchar (30)] |
| Partial Search Chapter Name | Chapter Name [Varchar (30)] |
| Display Book Chapters | Book ID [int] |
| Display Chapter | Book ID [int]<br>Chapter ID [int] |
| Update Chapter | Chapter ID [int]<br>Book ID [int]<br>Name [Varchar (30)]<br>Description [varchar (40)] |
| **Client** | |
| Delete Client | Client ID [int] |
| Insert Client | Client ID [int]<br>Name [Varchar (50)]<br>e-mail [Varchar (30)]<br>Address [Varchar (30)]<br>Telephone [Varchar (50)]<br>DOB [DateTime] |
| Display All Clients | |
| Display Client | Client ID [int] |
| Update Customer | Client ID [int]<br>Name [Varchar (50)]<br>e-mail [Varchar (30)]<br>Address [Varchar (30)]<br>Telephone [Varchar (50)]<br>DOB [DateTime] |
| **Department** | |
| Delete Department | Department ID [int] |
| Insert Department | Department ID [int]<br>Name [char (10)]<br>Description [Varchar (50)] |
| Display All Departments | |
| Display Department | Department ID [int] |
| Update Department | Department ID [int]<br>Name [char (10)]<br>Description [Varchar (50)] |
| **Employee** | |
| Delete Employee | Employee ID [int] |
| Insert Employee | Employee ID [int]<br>Name [Varchar (20)]<br>Hire Date [DateTime]<br>Shift ID [int]<br>e-mail [Varchar (20)] |

| | |
|---|---|
| | Telephone [nvarchar (20)]<br>Address [Varchar (30)]<br>Salary [int] |
| Display All Employees | |
| Display Employee | Employee ID [int] |
| Update Employee | Employee ID [int]<br>Name [Varchar (20)]<br>Hire Date [DateTime]<br>Shift ID [int]<br>e-mail [Varchar (20)]<br>Telephone [nvarchar (20)]<br>Address [Varchar (30)]<br>Salary [int] |
| **Order** | |
| Delete Order | Order ID [int] |
| Insert Order | Order ID [int]<br>Employee ID [int]<br>Client ID [int]<br>Date [Varchar (16)] |
| Display All Orders | |
| Display Order | Order ID [int] |
| Update Order | Order ID [int]<br>Employee ID [int]<br>Client ID [int]<br>Date [Varchar (16)] |
| **Order Details** | |
| Delete Order Detail | Order ID [int]<br>Order Detail ID [int] |
| Insert Order Detail | Order ID [int]<br>Order Item ID [int]<br>Book ID [int]<br>Quantity [int] |
| Display All Order Details | Order ID [int] |
| Display Order Detail | Order ID [int]<br>Order Item ID [int] |
| Update Order Detail | Order ID [int]<br>Order Item ID [int]<br>Book ID [int]<br>Quantity [int] |
| **Publisher** | |
| Delete Publisher | Publisher ID [int] |
| Insert Publisher | Publisher ID [int]<br>Name [Varchar (50)]<br>Address [Varchar (50)]<br>e-mail [Varchar (50)]<br>Telephone [Varchar (15)] |
| Exact Search Publisher Name | Publisher Name [Varchar (50)] |
| Partial Search Publisher Name | Publisher Name [Varchar (50)] |
| Display All Publishers | |

| | |
|---|---|
| Display Publisher | Publisher ID [int] |
| Update Publisher | Publisher ID [int]<br>Name [Varchar (50)]<br>Address [Varchar (50)]<br>e-mail [Varchar (50)]<br>Telephone [Varchar (15)] |
| **Purchase Order** | |
| Delete Purchase Order | Purchase Order ID [int] |
| Insert Purchase Order | Purchase Order ID [int]<br>Employee ID [int]<br>Supplier ID [int]<br>Date [Varchar (16)] |
| Display All Purchase Orders | |
| Display Purchase Order | Purchase Order ID [int] |
| Update Purchase Order | Purchase Order ID [int]<br>Employee ID [int]<br>Supplier ID [int]<br>Date [Varchar (16)] |
| **Purchase Order Details** | |
| Delete Purchase Order Detail | Purchase Order ID [int]<br>Purchase Order Detail ID [int] |
| Insert Purchase Order Detail | Purchase Order Item ID [int]<br>Purchase Order ID [int]<br>Book ID [int]<br>Quantity [int] |
| Display All Purchase Order Details | Purchase Order ID [int] |
| Display Purchase Order Detail | Purchase Order ID [int]<br>Purchase Order Item ID [int] |
| Update Purchase Order Detail | Purchase Order Item ID [int]<br>Purchase Order ID [int]<br>Book ID [int]<br>Quantity [int] |
| **Shift** | |
| Delete Shift | Shift ID (int) |
| Insert Shift | Shift ID [int]<br>Start Time [Varchar (20)]<br>End Time [Varchar (20)] |
| Display All Shifts | |
| Display Shift | Shift ID [int] |
| Update Shift | Shift ID [int]<br>Start Time [Varchar (20)]<br>End Time [Varchar (20)] |
| **Supplier** | |
| Delete Supplier | Supplier ID (int) |
| Insert Supplier | Supplier ID [int]<br>Name [Varchar (50)]<br>Address [Varchar (50)]<br>Telephone [Varchar(50)]<br>e-mail [Varchar(50)] |

| | |
|---|---|
| | URL [Varchar(50)]<br>Fax [Varchar(50)] |
| Exact Search Supplier Name | Supplier Name [Varchar (50)] |
| Partial Search Supplier Name | Supplier Name [Varchar (50)] |
| Display All Suppliers | |
| Display Supplier | Supplier ID [int] |
| Update Supplier | Supplier ID [int]<br>Name [Varchar (50)]<br>Address [Varchar (50)]<br>Telephone [Varchar(50)]<br>e-mail [Varchar(50)]<br>URL [Varchar(50)]<br>Fax [Varchar(50)] |
| **Topic** | |
| Delete Topic | Book ID (int)<br>Chapter ID (int)<br>Topic ID (int) |
| Insert Topic | Topic ID [int]<br>Chapter ID [int]<br>Book ID [int]<br>Name [Varchar (30)]<br>Details [Varchar (40)] |
| Exact Search Topic Name | Topic Name [Varchar (30)] |
| Partial Search Topic Name | Topic Name [Varchar (30)] |
| Display Chapter Topics | Book ID [int]<br>Chapter ID [int] |
| Display Topic | Book ID [int]<br>Chapter ID [int]<br>Topic ID [int] |
| Update Topic | Topic ID [int]<br>Chapter ID [int]<br>Book ID [int]<br>Name [Varchar (30)]<br>Details [Varchar (40)] |

### 6.2.4 LIS Classes

LIS class diagram include classes that address employee, client, author, publisher, book, order, purchase order, and supplier. Classes address at least the four main database operations: insert, update, delete, and select.

## 7. Summary

This chapter highlighted the proposed architecture characteristics and layered nature. SIS, and LIS are presented. Use cases, system processes, analysis and design details like entities, E-R diagrams, class diagrams, database tables, and proposed service based architectures. Faculty Information System provides the required functionalities to store data about faculties, but SIS addresses the internal faculty requirements. LIS automates all library activities and exposes library components via service layer that can be utilized within different external systems.

# Chapter 4

## Chapter Four

# PROPOSED LMS USING SOA

This chapter presents the service based architecture of LMS components depicted in figure 4.1. Chapter four is divided into three parts: Part I presents Digital Library, Part II presents CMS, and Part III presents AMS.

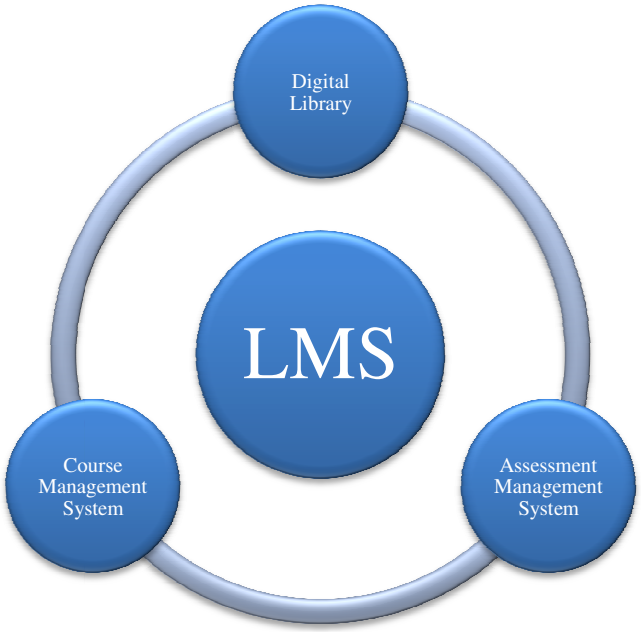


**Figure 4.1: Proposed LMS Components**

# Part I

# Chapter Four – Part I

# DIGITAL LIBRARY

## 1. Introduction

This chapter presents the service based architecture of the first proposed component of the LMS: Digital Library. Digital Library aims to present a digital store for faculty contents that needs to be available for future usage in electronic format. Digital Library can be used to present external available digital books, and digital materials available by the system, like: books authored by internal authors / instructors, under graduate, graduation, and post graduate project's materials like source code and documentation.

## 2. Problem Definition and Proposed Solution

Most of Students face repeatable situations where they need to make decisions and can not find help. Examples of those situations are:

- Students need to find a database that holds a list and available content of Graduation Projects performed by students of the same faculty from previous years. This Graduation Projects database can be used as guidance to new students to help choosing the right graduation project, besides the capability to search and retrieve the digital contents of those graduation projects.
- Students can find it conflicting to choose between the different departments of the faculty. In order to help students choose the right department, we can provide a structure of courses taught in each department, contents of those courses, and relationships between those materials.

**Proposed Solution:**

- Data base system that holds:
  - Under-Graduate Projects
  - Graduation Projects
  - Post-Graduate Projects

In the electronic format that gives the capability to search within as a Digital Library.

- Present Books authored by Faculty professors in the electronic format; available also as a digital library.

- Present On-Line magazine of the faculty. This magazine is edited, and authored by students' faculty.
- Present On-Line guide for students to help them choose among different departments in the faculty.
- Present On-Line guide for fresh students with external links to resources that they can consume through the four years of study.
- Present students guide for specialized laboratories in the faculty. There are laboratories made available for Computer Networks, others are made for programming requirements, and so on. Students can use those laboratories.
- Make all available digital contents applicable for online comments by readers.

### 3. Digital Library Analysis

Figure 4.2 presents the Digital Library Use Case. Processes include:

- Review Book
- Review Under Graduate Project
- Review Graduation Project
- Review Post Graduate Project
- Make Under Graduate Project
- Make Graduation Project
- Make Post Graduate Project

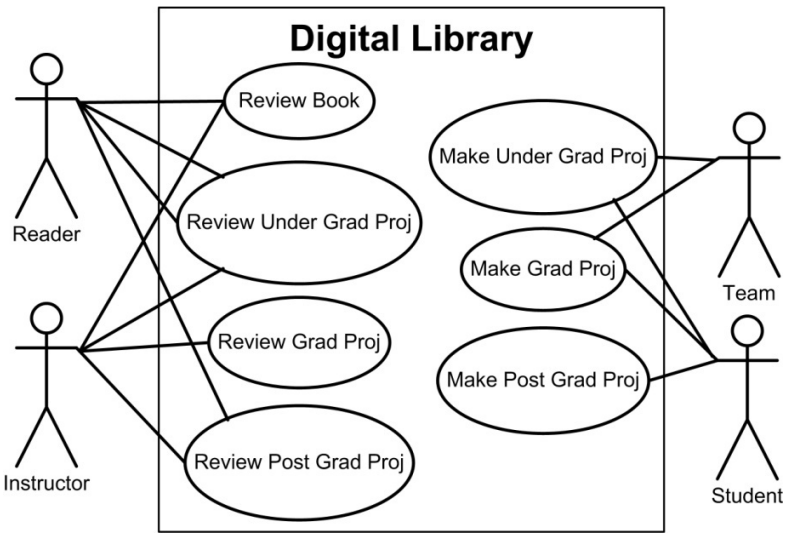


**Figure 4.2: Digital Library Use Case**

### 4. Digital Library Design

Figure 4.3 presents Digital Library entities. Figure 4.4 shows the Entity Relationship (E-R) diagram of Digital Library.

## 4.1 Proposed Digital Library Architecture

Figure 4.5 (a,b) depicts the proposed entity centric – services based Digital Library architecture. Part (a) covers Student, Reader Review, Author/Instructor, Publisher, Book, and Team managers. Every manager lies in the Orchestration layer; that is a sub layer of the Services layer, and consumes application services layer insert, update, delete, and select services. Part (b) presents Topic, Chapter, Project, and Course managers.

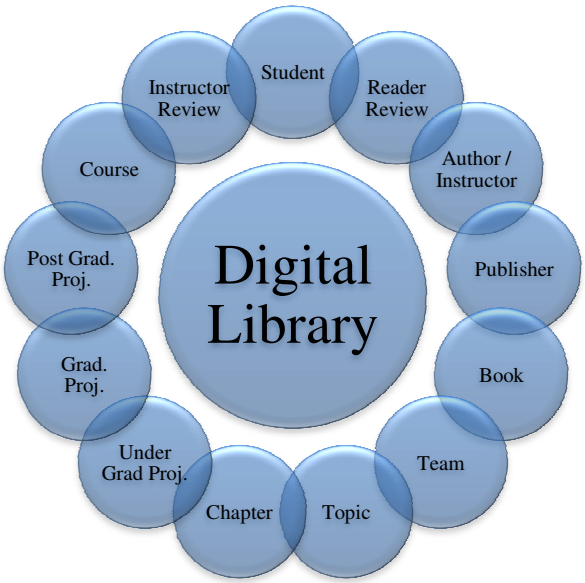


**Figure 4.3: Digital Library Entities**

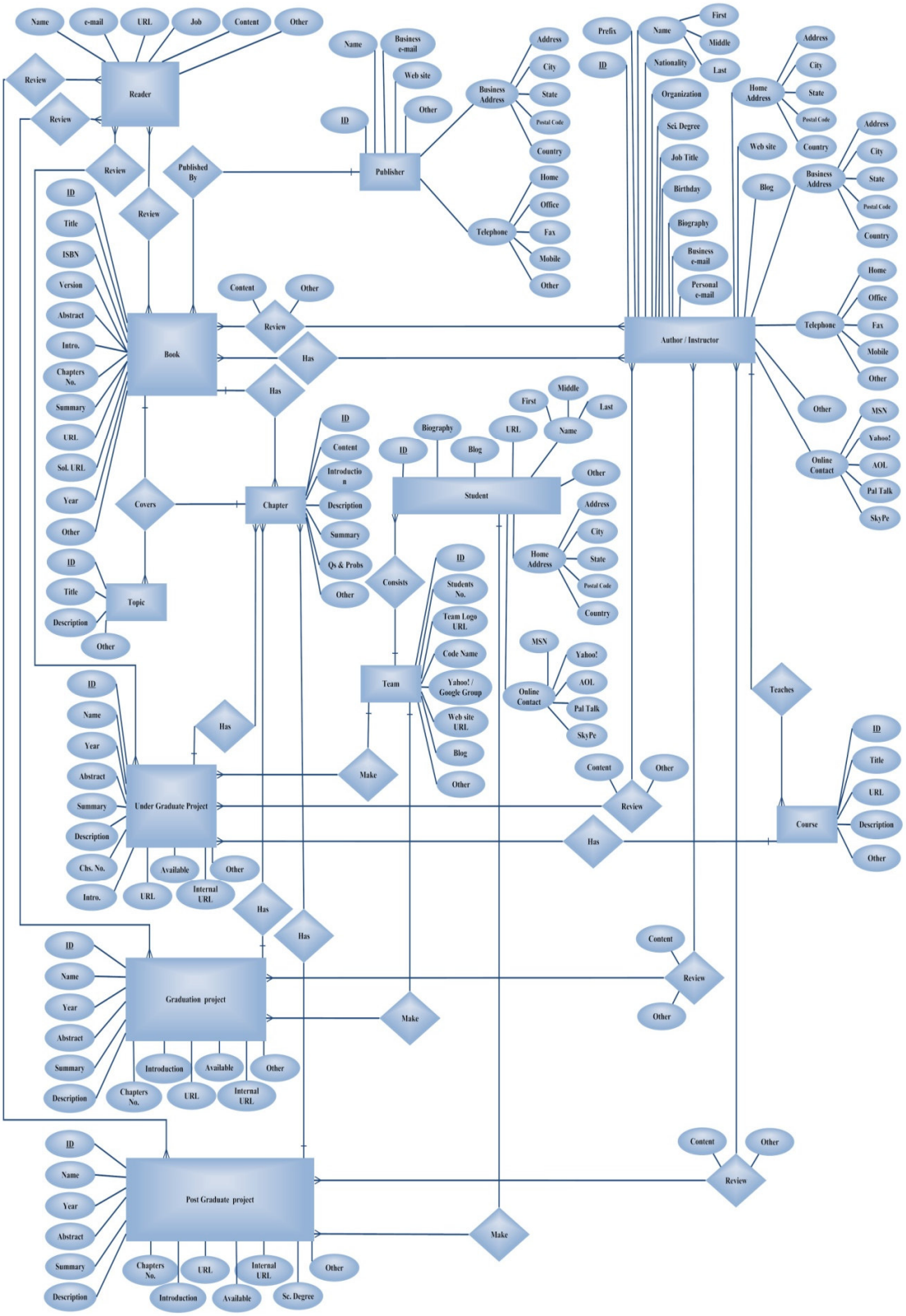


**Figure 4.4: Digital Library Entity-Relationship (E-R) Diagram**

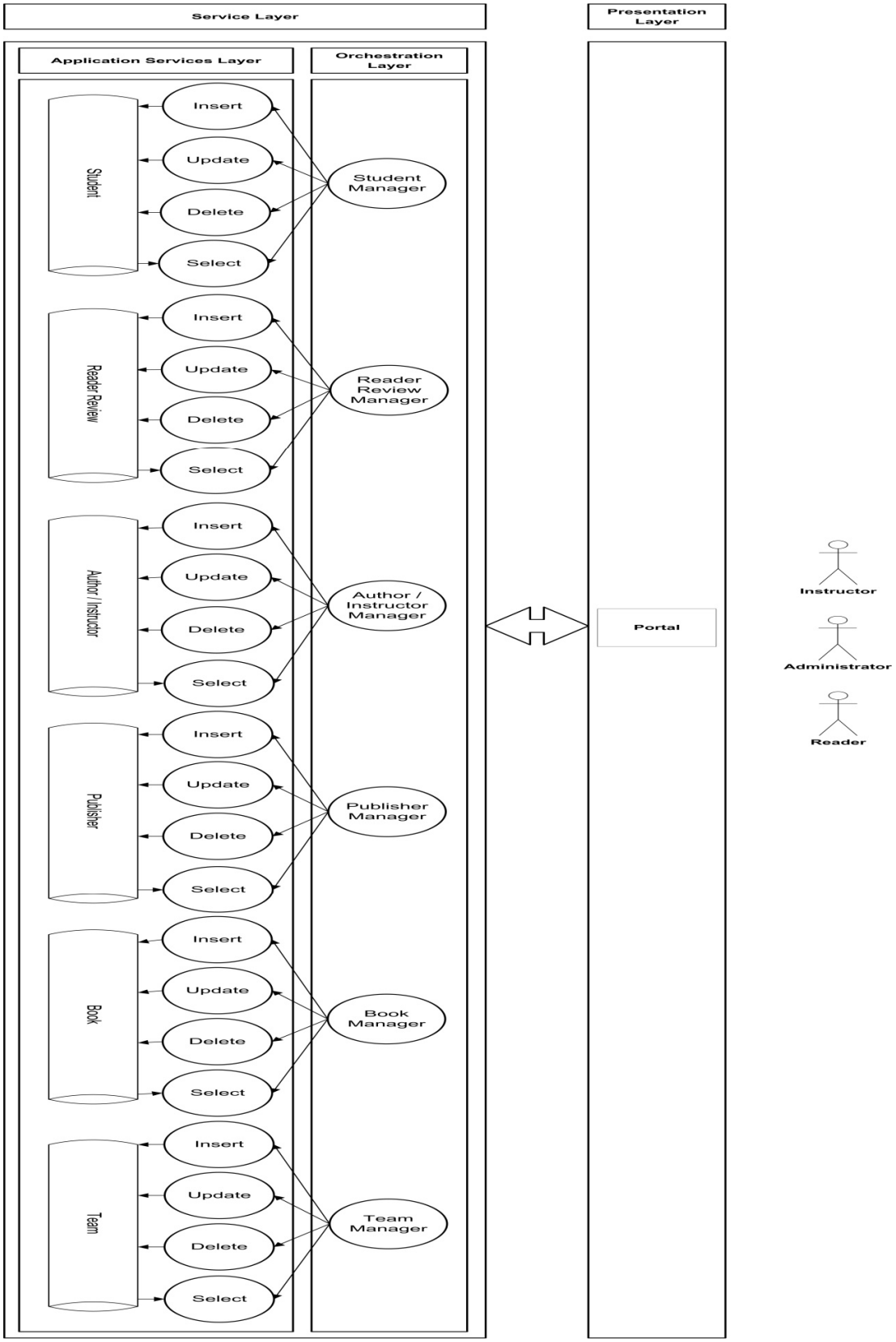


**Figure 4.5 (a): Digital Library Proposed Architecture**

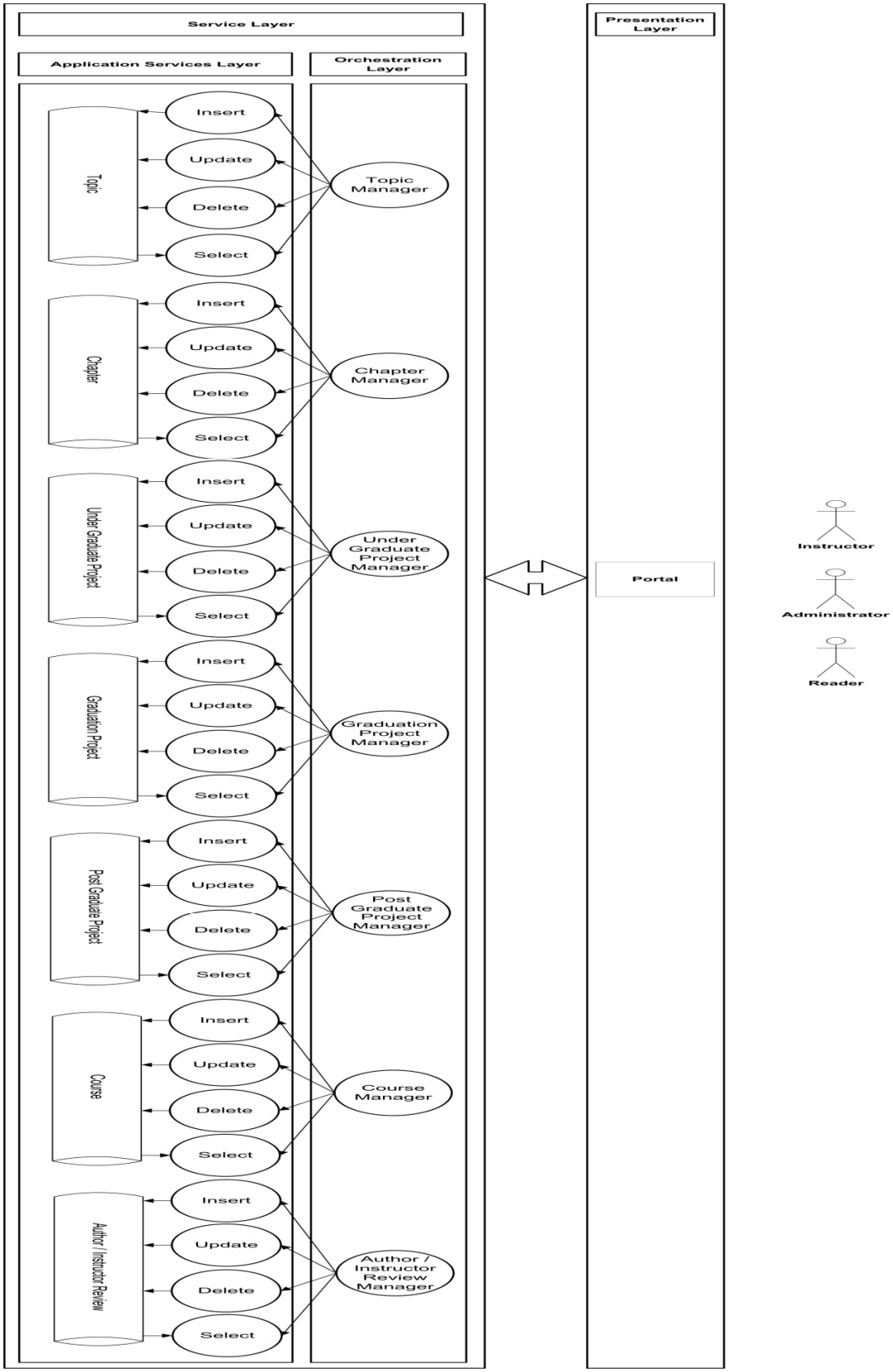


**Figure 4.5 (b): Digital Library Proposed Architecture**

## 4.2 Digital Library Database Tables

Figure 4.6 presents the implemented database tables of Digital Library.

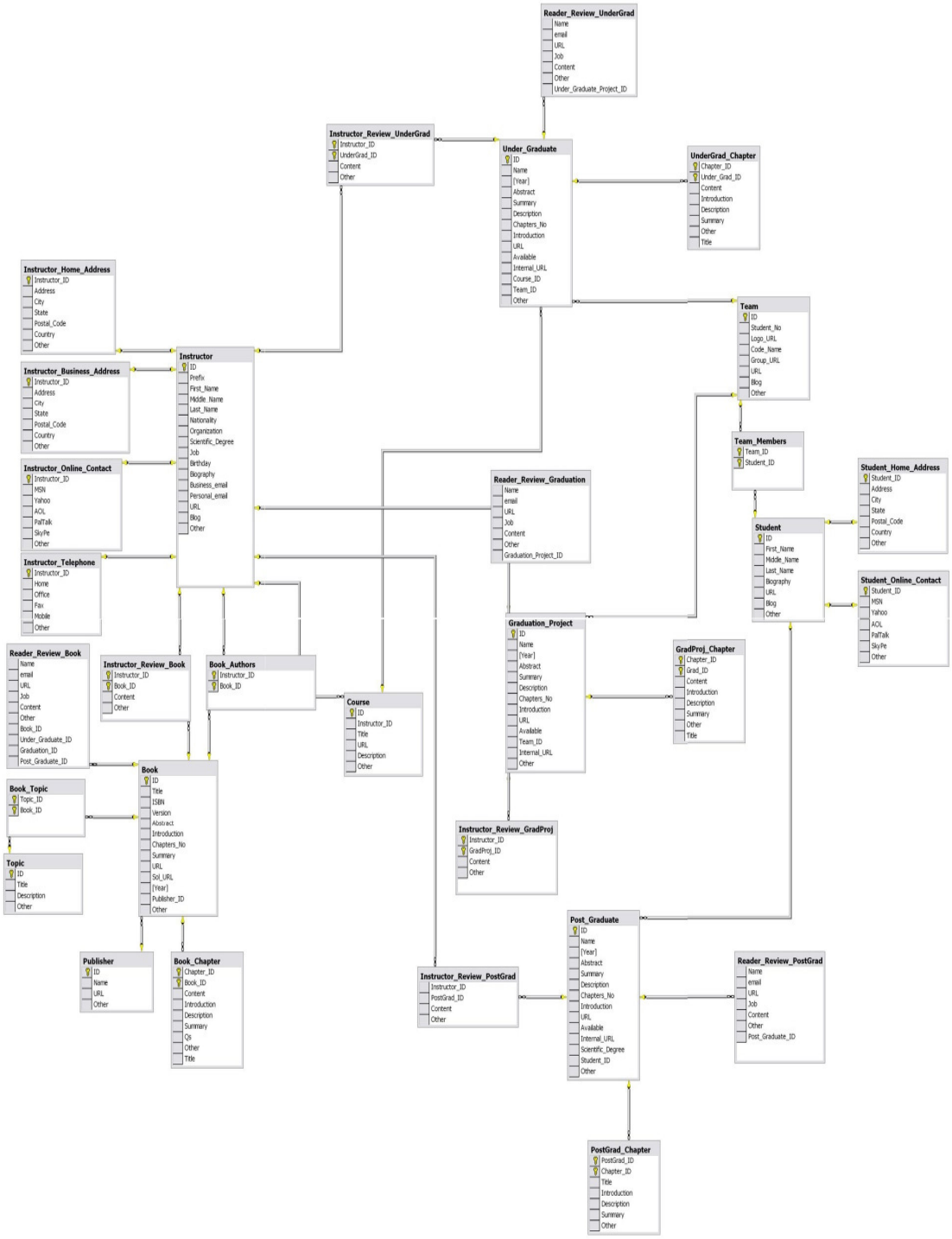


**Figure 4.6: Digital Library Database Tables**

## 4.3 Digital Library Classes

Figure 4.7 shows the digital library class diagram


**Figure 4.7: Digital Library Class Diagram**

## 5. Summary

Digital Library aims to present a digital store for faculty contents that needs to be available for future usage in electronic format. Digital Library can be used to present external available digital books, and digital materials available by the system, like: books authored by internal authors / instructors, under graduate, graduation, and post graduate project's materials like source code and documentation.

# Part II

# Chapter Four – Part II

# COURSE MANAGEMENT SYSTEM

## 1. Introduction

This chapter presents the proposed CMS. Proposed CMS facilitates integration among different CMSs. An automated course search, import, and deposit process is presented. This automated process requires governance of business rules. Business rules might be limiting system efficiency, so they must be monitored and modified when needed. Manage business rules process is presented to achieve this goal. Utilizing SOA to integrate Web services and software agents in CMSs highlighted the unlimited advantages of Web services and its capabilities to facilitate software agents' integration within systems.

## 2. CMS Analysis

Proposed CMS addresses two processes namely Search and Manage Rules as depicted in the use case diagram at figure 4.8. Search process enables instructor and CMS to search for courses within course repositories managed and maintained by internal and external CMSs, import, and display those courses. There are managerial issues that should be addressed to enable this process. Managerial issues are managed at Manage Rules process.

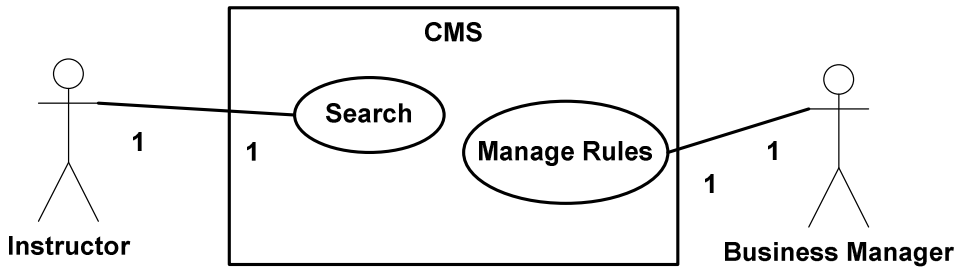


**Figure 4.8: CMS Use-Case**

### 2.1 Proposed CMS Architecture

Figure 4.9 depicts the proposed CMS architecture that enables automation of courses integration. From architectural perspective, proposed system consists of two main layers that present its frame: Presentation layer, and Service layer.

**Presentation Layer**

It is the layer that presents user interface. System's scope is limited to instructors and business managers. Those are the end users of the system. Display service is the service that is responsible for preparing dynamic output prepared to users, calling the suitable web page to display data. More than web page is prepared to address different functionalities. Display service passes suitable set of data to appropriate pages, and acquires input from users incase there are. Separation of interface design and implementation from business logic has proven many advantages.

**Service Layer**

Service layer consists of three sub layers: Orchestration Layer, Application Services Layer, and Agents Layer.

**Orchestration Layer**

Orchestration layer manages interaction details required to ensure that service operations are executed in a specific sequence. Sequences are determined based on processes supported by system. Orchestras within this layer are: Course Manager, Rules Manager, Exception Manager, Financial Manager, Mail Manager, Faculties Manager, Instructor's Manager, and Feedback Manager.

**Application Services Layer**

Application Services are set of stateless services perform certain. Process is the summation of tasks performed by one or more application services layer in the sequence that is maintained at orchestration layer services. Services of Application Services Layer are: Discover/Recommend Display, Check Capability, Import, Insert Rule, Pay, Raise Exception, Manage Rules, Send Mail, Manage Courses, Search Process Exception Manager, Manage Rules Exception Manager, Manage Faculties, Manage Instructors, and Manage Feedback.

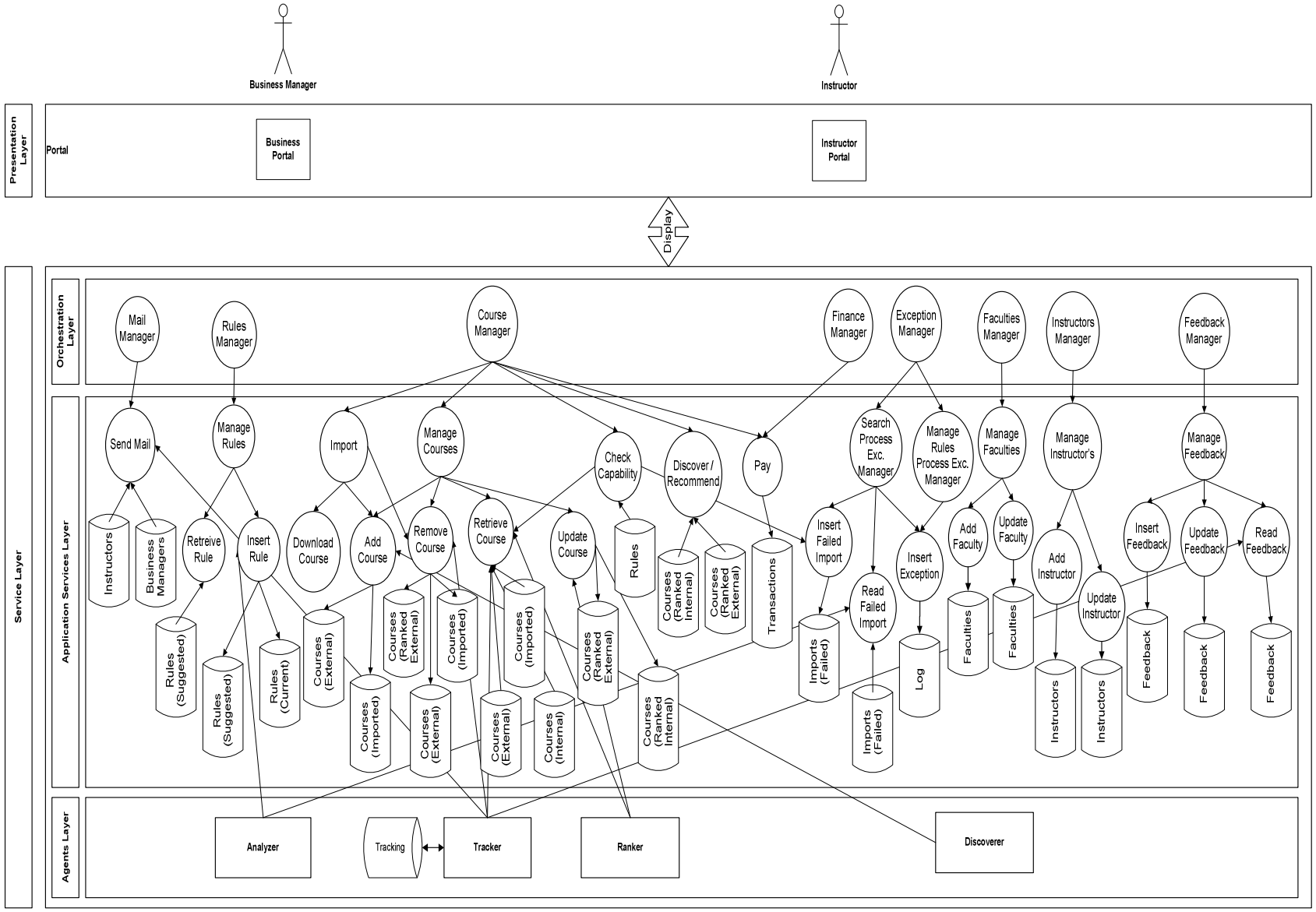


**Figure 4.9: Proposed CMS Architecture**

**Agents' Layer**

Specific task agents are required to serve system. Software agents are always the optimum solution for track and analysis tasks. Software agents are: Discoverer Agent, Ranker Agent, Analyzer Agent, and Tracker.

- **Discoverer:** It is the software agent that is responsible for preparing list of external course providers and indexing metadata about courses.
- **Ranker:** It is the software agent that is responsible for ranking different system's components based on ranking rules. System components that should be ranked are: Universities, Faculties, Instructors, and Courses.
- **Tracker:** Proposed CMS depends heavily on instructor's feedbacks to enhance system's performance.
- **Analyzer:** Analyzer is the software agent that is responsible for analyzing previously stored failed imports by detecting the most cause of failed imports and suggesting new conditions to decrease such problem. Suggested rules are inserted temporarily in the suggested rules database, waiting for the approval/denial of business manager.

Course Management System is one of the Learning Management System components that is interested in automating the activities related to courses discovery, downloading, and tracking.

**2.2 Search Process Analysis**

Search process is initiated by Instructor, managed and maintained by Course Manager. Figure 4.10 depicts analysis of Search process. Highlighted shapes present activities performed by course manager. Other shapes present non-course manager activities. Non-course manager activities are presented within the architecture as stand alone services that are called explicitly by course manager.

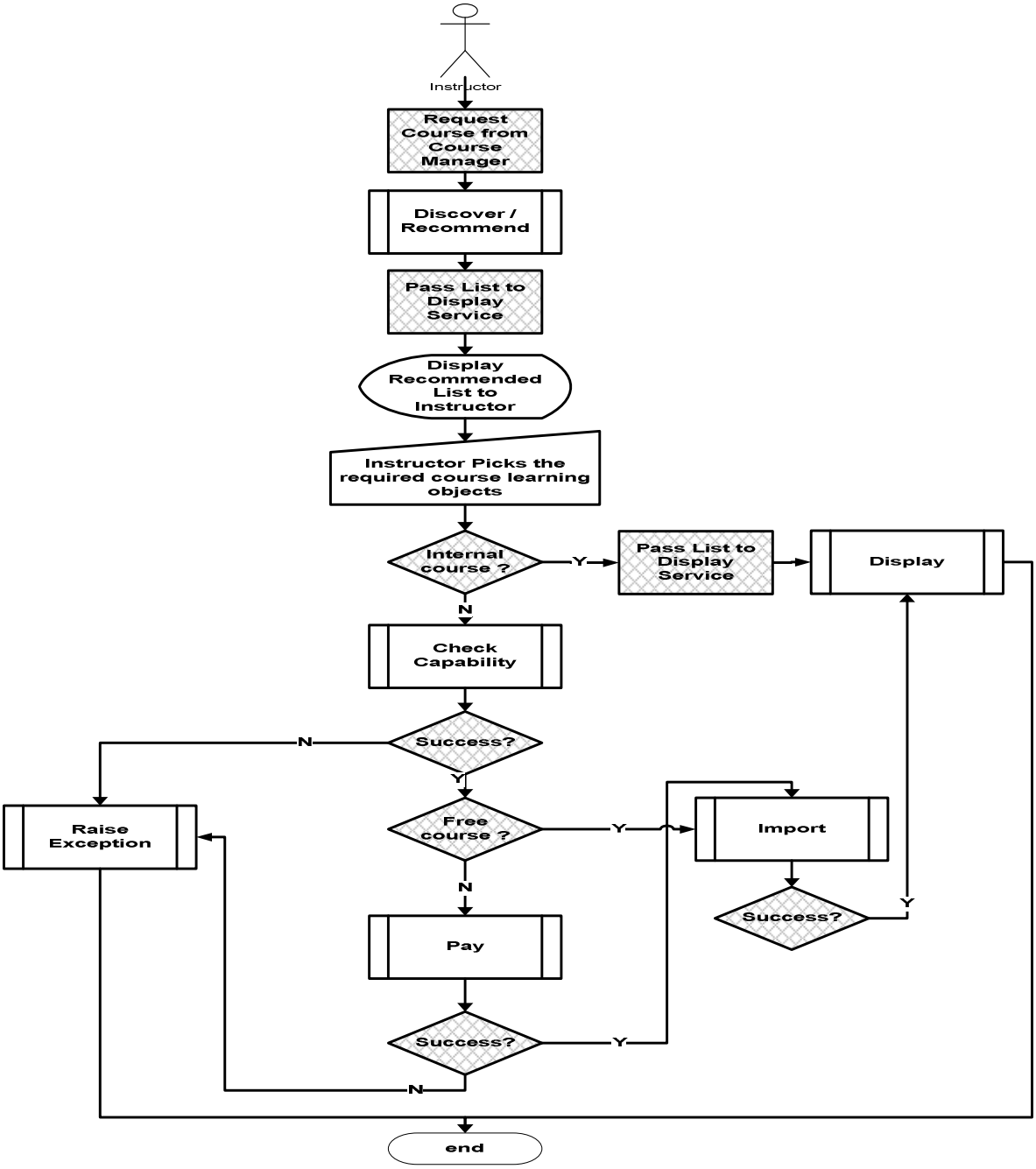


**Figure 4.10: Search Process Analysis**

## 2.3 Manage Rules Process Analysis

Manage Rules process is initiated by Business Manager, managed and maintained by Rules Manager as shown in figure 4.11. Highlighted shapes present activities performed by rules manager. Other shapes present non-rules manager activities. Non-rules manager activities are presented within the architecture as stand alone services that are called explicitly by rules manager.

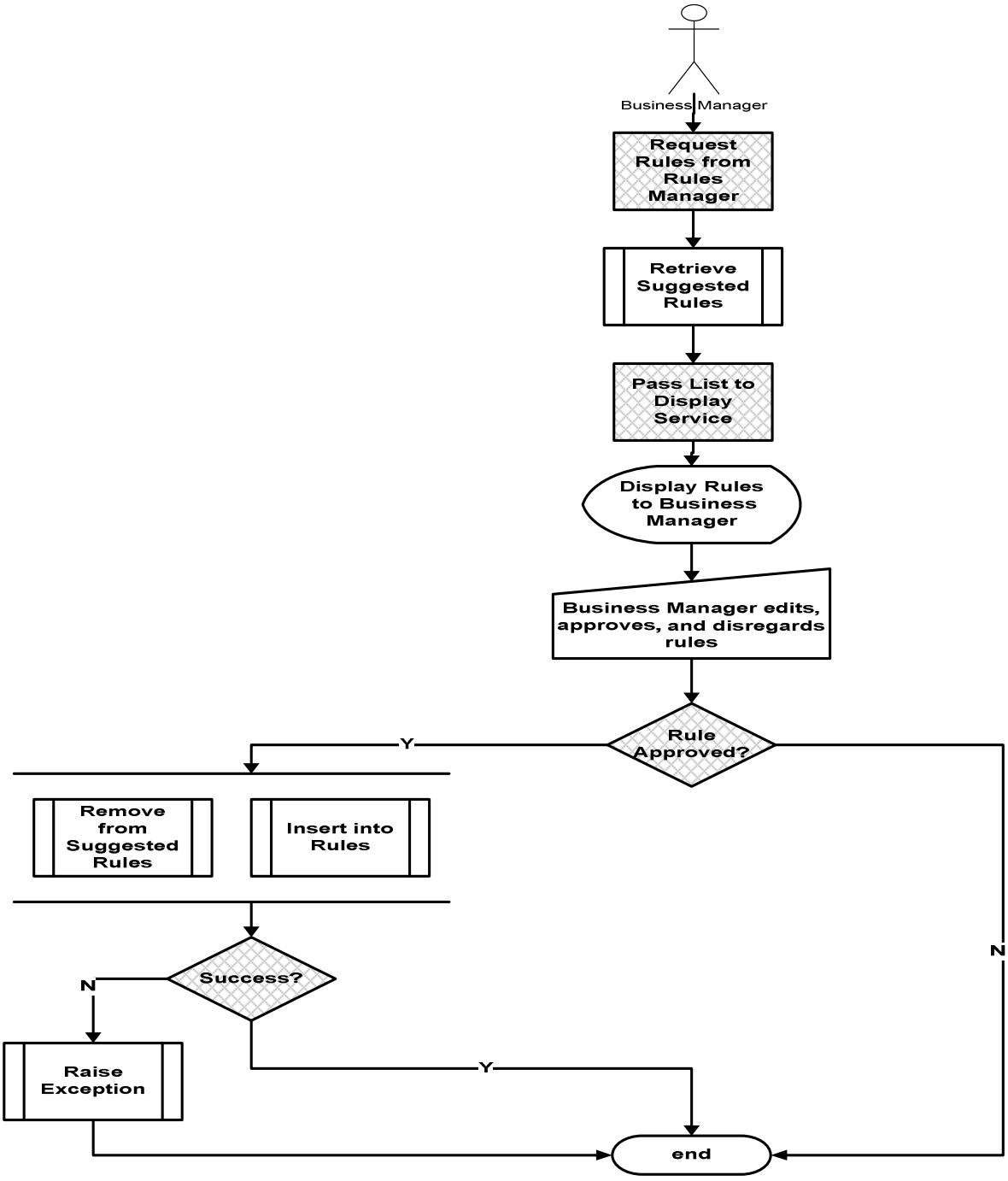


**Figure 4.11: Manage Rules Process Analysis**

## 2.4 Software Agents Analysis

This part presents the design process of the four software agents highlighted in the CMS. The four software agents are: Discoverer, Ranker, Tracker, and Analyzer.

### 2.4.1 Discoverer Agent Analysis

It is the software agent that is responsible for preparing list of external course providers and indexing metadata about courses. Discoverer agent makes use of standard SOAP/XML based Web services that integrates internal and external applications [1]. Discoverer software agent integrates internally with exposed Web services like Course manager in order to insert data and metadata about courses. Figure 4.12 depicts Discoverer software agent architecture. There are three main interfaces for Discoverer agent: Internal interface, external interface, and human interface. Discoverer software agent integrates externally with educational institutions exposed Web services in order to initiate and maintain an internal catalog of external courses providers and courses. Web services technologies enabled integration flexibility and maintainability [1]. Internal interface presents integrating Discoverer software agent with CMS's Web services on service level. Discoverer needs to consume Manage Courses service in order to update courses database. Discoverer is not directly inserting data into courses database. All operations have to be done through Manage Courses Web service in order to obtain service level integration benefits. Human interface is required to enable instructor and business manager insert data about courses and courses' providers. Programmer interface is required to inform Discoverer about available external Web services. Universal Description Discovery and Integration (UDDI) provides a platform-independent way of describing services, discovering businesses, and integrating business services using the Internet [30].

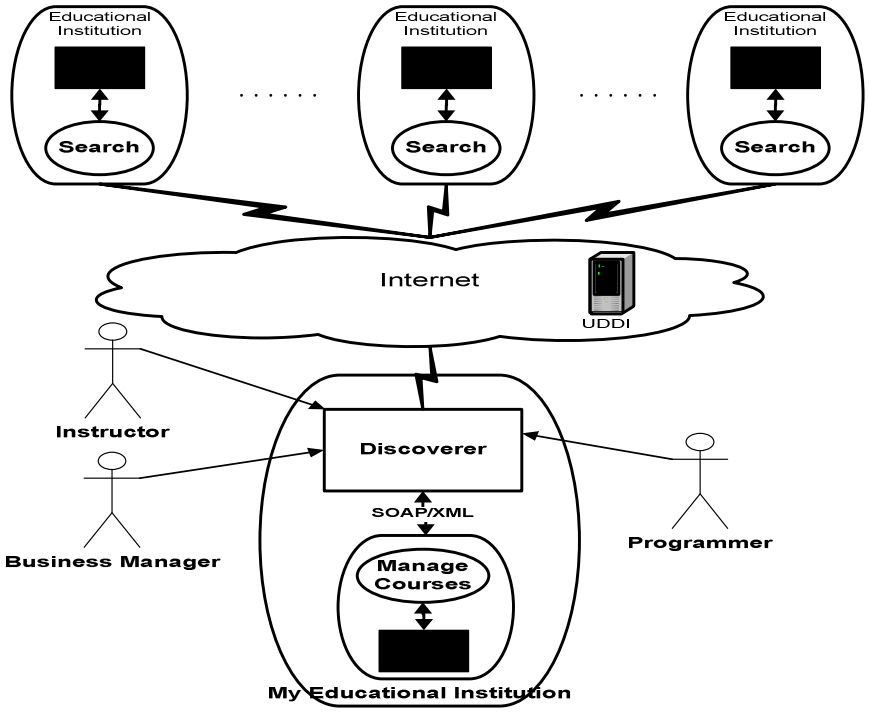


**Figure 4.12: Discoverer Agent Architecture**

### 2.4.2 Ranker Agent Analysis

It is the software agent that is responsible for ranking different system's components based on ranking rules. System components that should be ranked are: Universities, Faculties, Instructors, and Courses. Different formulas used to rank each of them. Ranker agent integrates internally with the proposed system on service level via system's exposed Web services. Ranker agent consumes suitable Web services to insert, update, retrieve, or delete items that to be rank. Figure 4.13 depicts a Course Ranker process design. Course Ranker consumes two services in the early phase of running, and another two services as depicted.

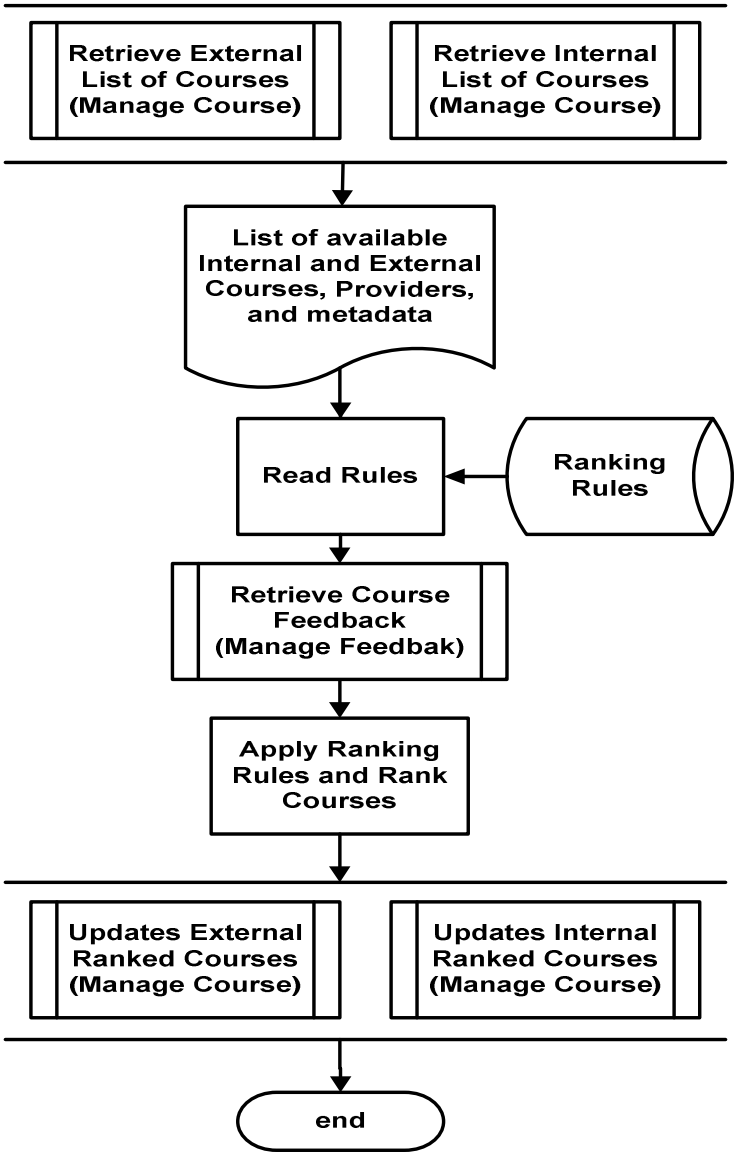


**Figure 4.13: Courses Ranking Process Analysis**

### 2.4.3 Tracker Agent Analysis

Proposed CMS depends heavily on instructor's feedbacks to enhance system's performance. Figure 4.14 depicts of tracking instructor's feedback of imported courses process design.

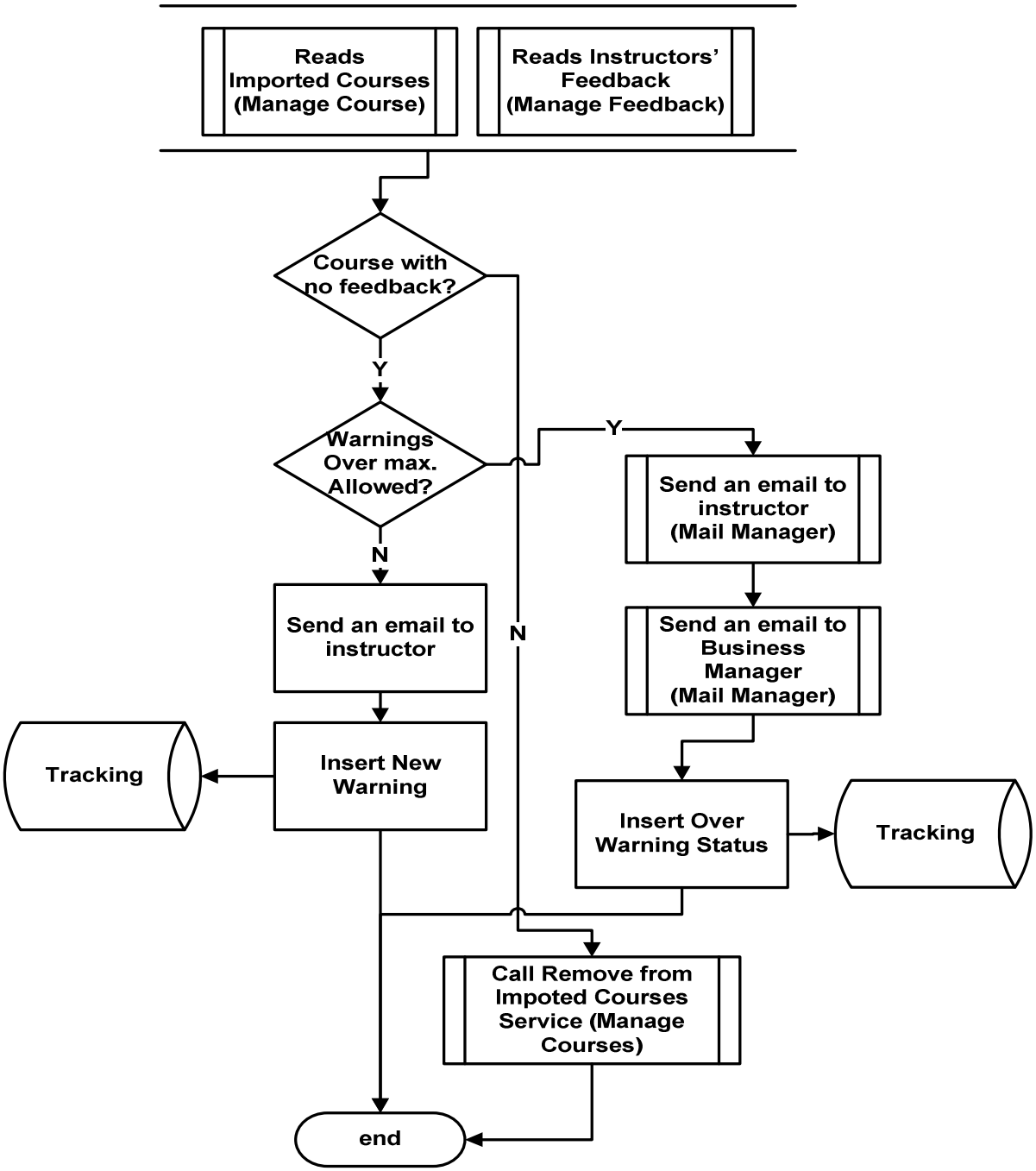


**Figure 4.14: Imported Courses Tracker Process Analysis**

### 2.4.4 Analyzer Agent Analysis

Analyzer is the software agent that is responsible for analyzing previously stored failed imports by detecting the most cause of failed imports and suggesting new conditions to decrease such problem. Suggested rules are inserted temporarily in the suggested rules database, waiting for the approval/denial of business manager. Analysis of failed imports tend mainly to increase system overall performance by adapting system to meet/exceed instructor's expectations. Figure 4.15 depicts Analyzer agent process analysis.

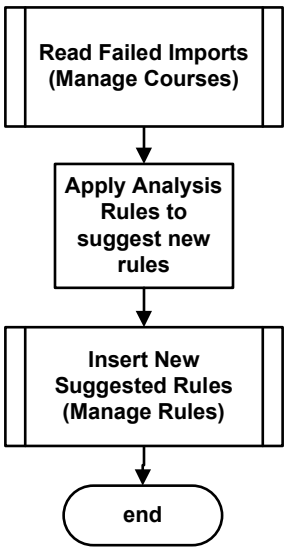


**Figure 4.15: Analyzer Agent Process Analysis**

## 3. Design of CMS

CMS design include design details of the Search, Manage Rules Process Design, and designing the software agents. Each process design includes defining composing services and designing each of them.

### 3.1 Search Process Design

Figure 4.16 depicts Search process design. Search process utilizes more than one service. Services that are utilized by Search process are: Discover/Recommend Display, Check Capability, Manage Courses, and Import.

### 3.1.1 Discover/Recommend Service

It is the service that is responsible for searching internal and external list of ranked courses to prepare a list of ranked courses that satisfies user query. Recommend is the service responsible for ranking results prepared by Discover service by relevance of its keywords and user query keywords. Detailed design for Discover/Recommend activities is depicted in figure 4.16 under Discover/Recommend side title.

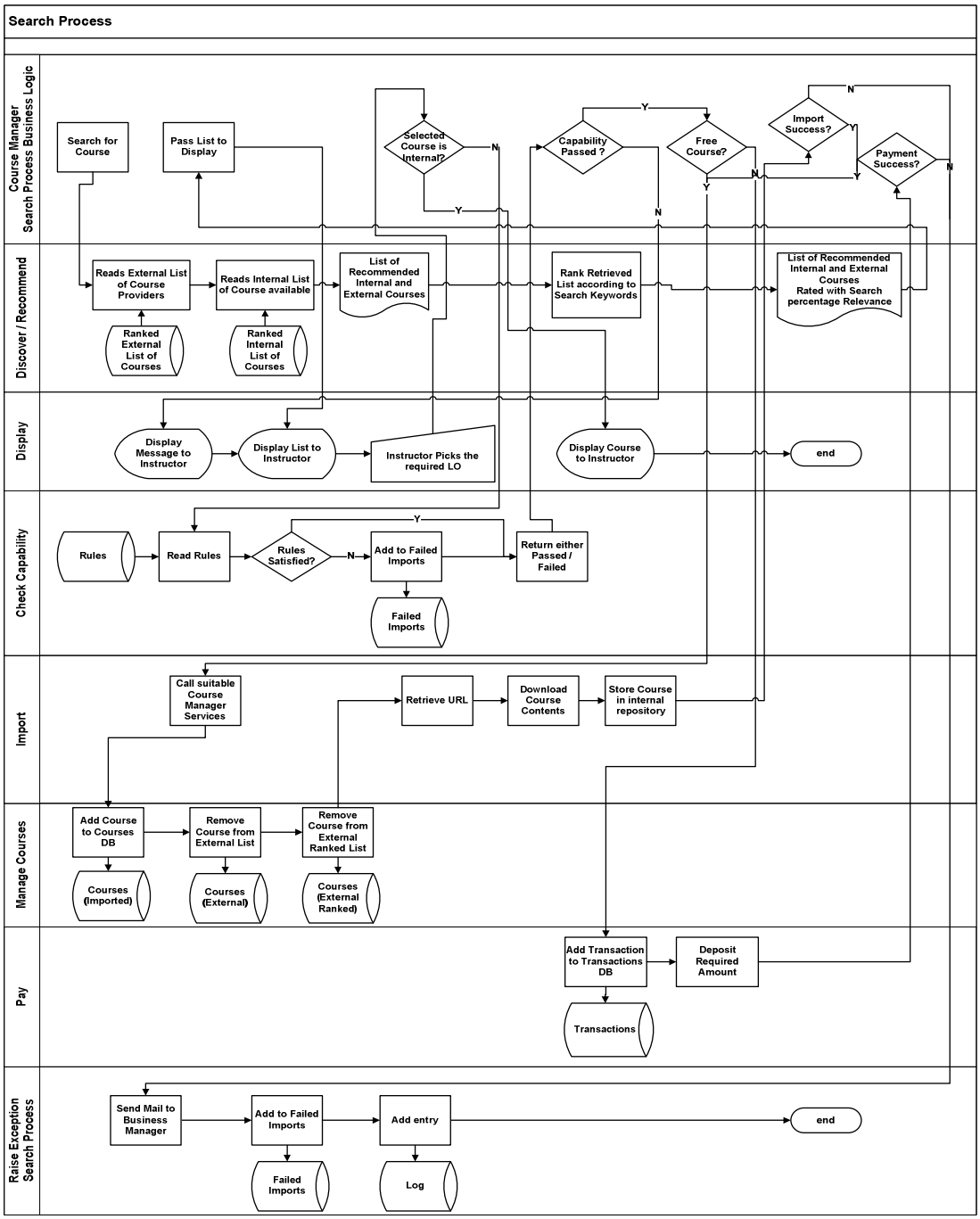


**Figure 4.16: Search Process Design**

Figure 4.17 depicts tables required by Discover/Recommend to achieve tasks. Courses_External_Ranked, and Courses_Internal_Ranked are the tables required by Discover to prepare list of courses. Keywords table is the table required by Recommend to rank list by relevance to user's keywords.

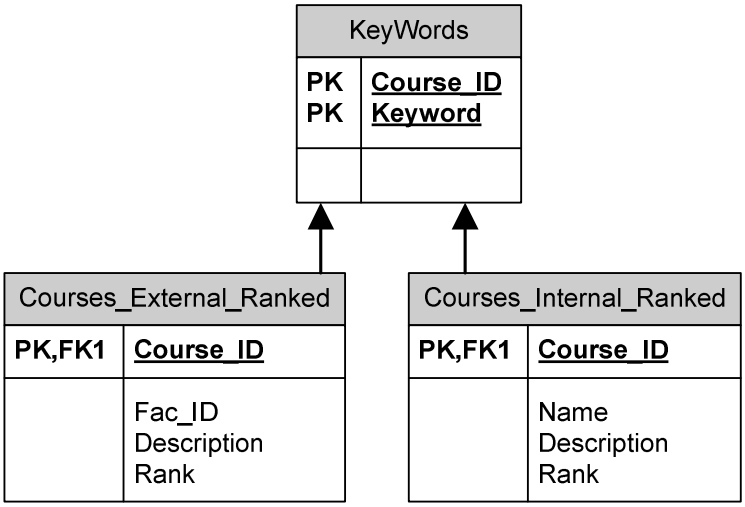


**Figure 4.17: Tables of Discover/Recommend Service**

### 3.1.2 Display Service

It is the service that is responsible for preparing data that will be displayed, passing this data to the proper page at presentation layer, and acquiring input from end user. Display service manages different kinds of data. Those data include:

- Display list of courses: at presentation layer, there should be a display course page that is waiting for the dataset that will be passed by course manager.
- Display Notification message: about rules, there should be a display rules page that is waiting for the dataset that will be passed by course manager.

Display service takes input from instructor and business manager. Example of instructor input is Course_ID. Example of Business Manager Input is Rules_Approved. Figure 4.16 depicts detailed activities performed by Display service to support Search process.

### 3.1.3 Check Capability Service

It determines either the course satisfies organizational rules or not, by doing so, it actually determines if course can be imported or not. Rules not only include financial issues; like limits provided for each instructor, but they also reflect pedagogical aspects of the educational institution. Educational institutions must guarantee certain pedagogical level of courses. Check Capability Service requires database tables depicted in figure 4.18.

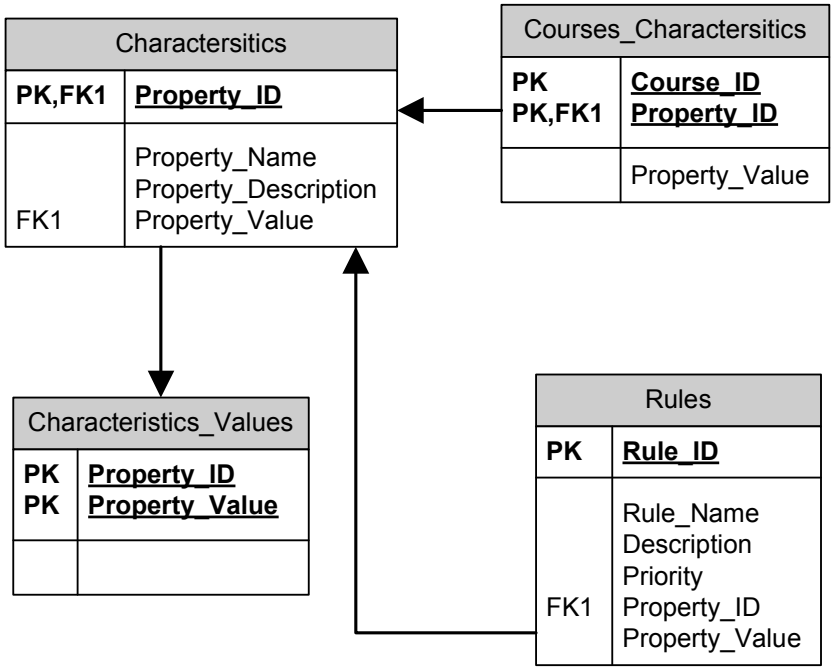


**Figure 4.18: Tables of Check Capability Service**

### 3.1.4 Import Service

It consumes other services to achieve the required task, which is download course file and update database. In order to update database, import service calls Manage Course appropriate services. Download Course It is the functionality that holds programming logic that is responsible for downloading the course file into FTP server.

### 3.1.5 Manage Courses Service

Manage Courses refer to the capability to implement basic databases operations upon certain entity. Basic database operations are: insert, update, and delete [29]. As courses are the entity of interest to search process, add, remove, and update operations should be supported. Courses are classified into three categories: Internal, External, and Imported. Internal courses refer to courses developed by institution's instructors. External courses refer to courses that are available in other educational institutions and have not been imported yet. Imported courses refer to downloaded courses form external sources. Internal and External courses are ranked in different tables. Manage Course Service is capable of performing basic database operations on those different tables. Figure 4.19 depicts required database tables to support basic database activities.

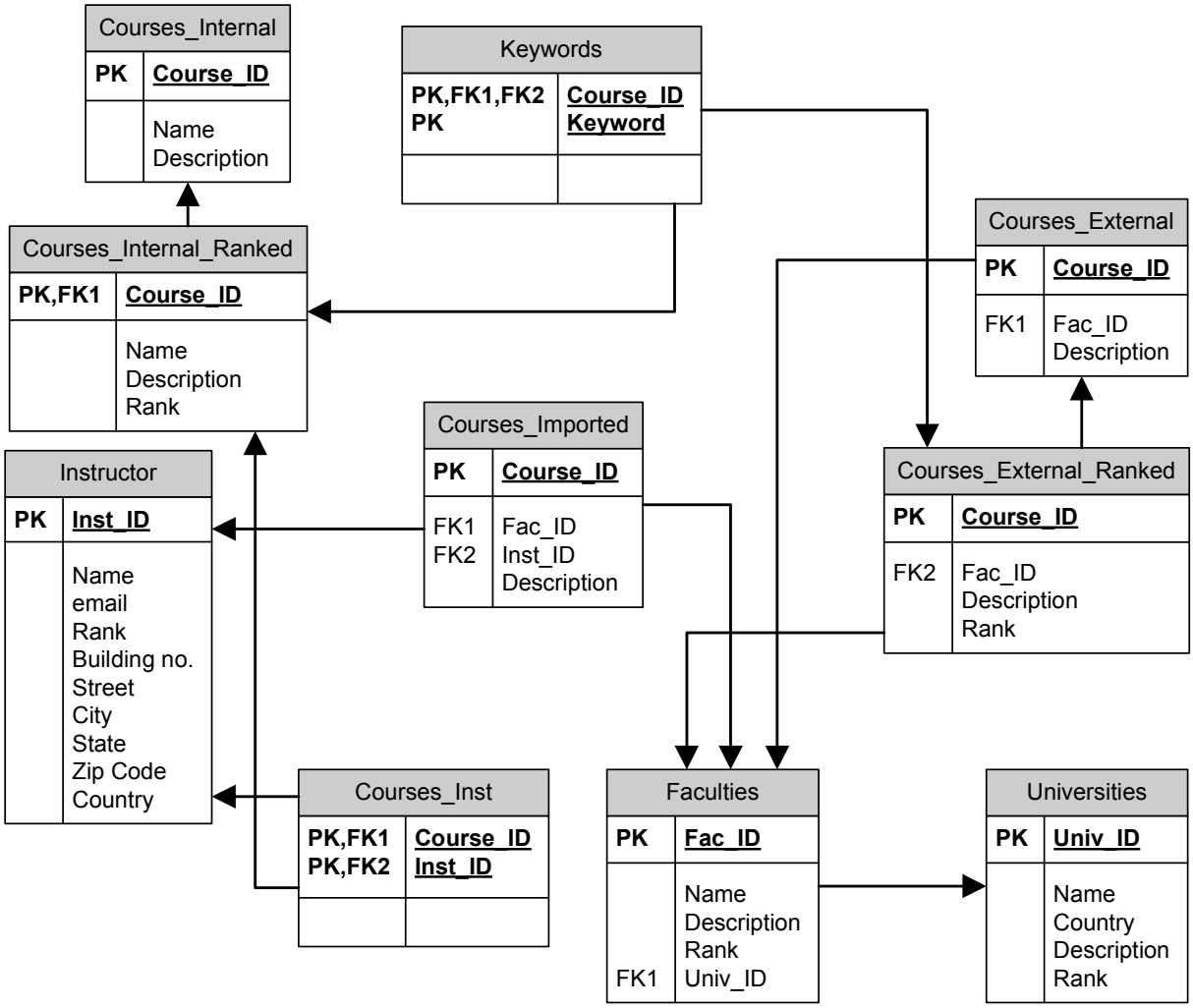


**Figure 4.19: Tables of Manage Courses Service**

### 3.1.6 Pay Service

Responsible for depositing money into accounts determined by the transaction. Figure 4.16 depicts detailed activities by Pay service. Pay service should manage transactions carefully by adding transaction entry to transactions table before attempting the process for it to be able to rollback non successful transactions. Figure 4.20 depicts required table to support financial transactions by the Pay service.

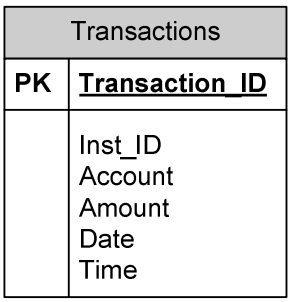


**Figure 4.20: Tables of Pay Service**

### 3.1.7 Raise Exception Search Process Service

Incase any kind of errors happened during the Search process, a Search Process Exception is raised. Raise Exception Service performs two main functionalities: Adds an entry to Failed Imports table, and adds an entry to Log table. Figure 4.16 depicts activities performed by Raise Exception Search Process Service. Figure 4.21 depicts required tables by Raise Exception Search Process Service.

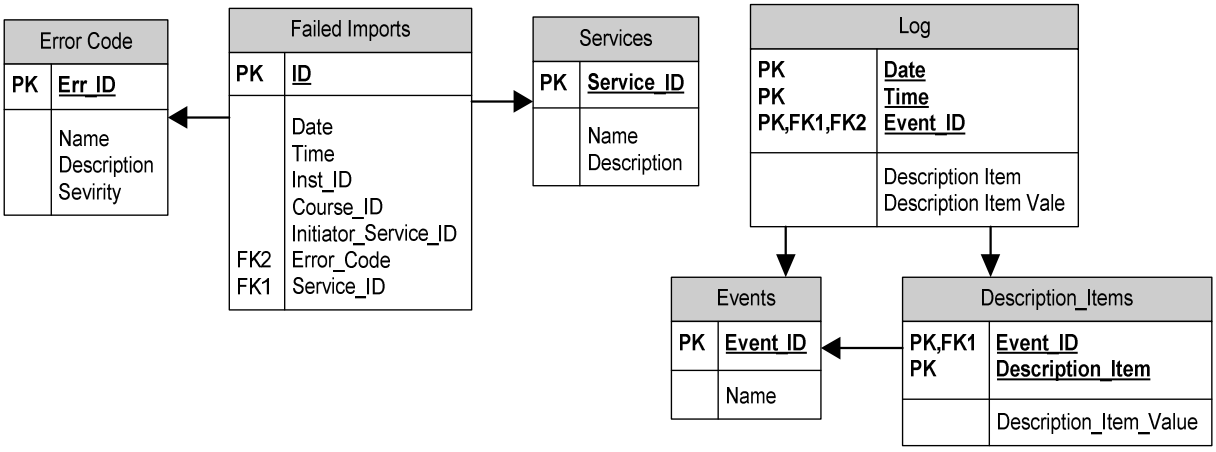


**Figure 4.21: Tables of Raise Exception Search Process Service**

## 3.2 Manage Rules Process Design

Figure 4.22 depicts Manage Rules Process Design. Manage rules process is initiated by Business Manager and managed by Rules Manager. Manage Rules Process utilizes the following services: Retrieve Suggested Rules, Display, Insert Rule, and Raise Exception.

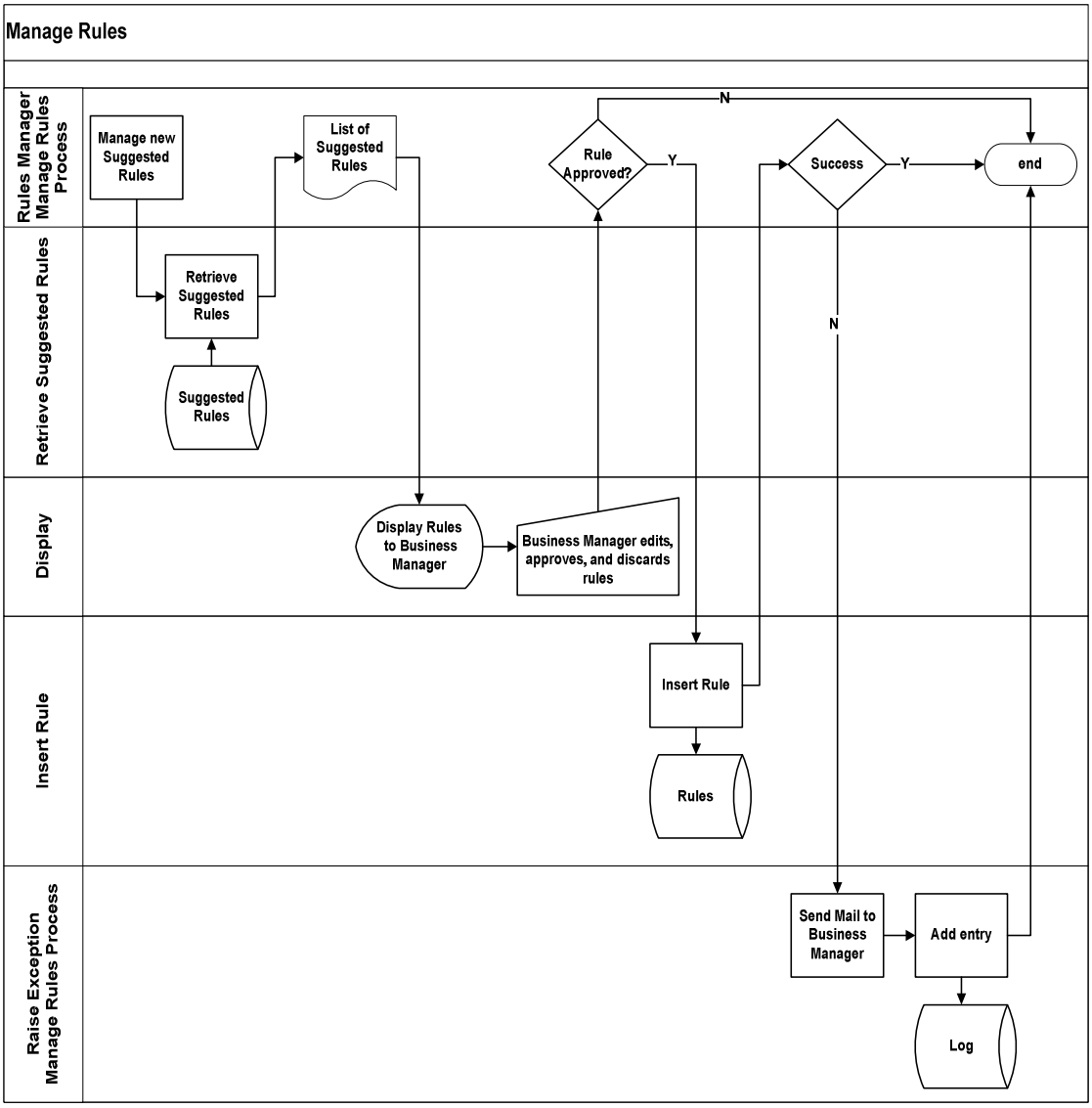


**Figure 4.22: Manage Rules Process**

### 3.2.1 Manage Rules Service

Manage Rules Service supports basic database operations on rules. There are three rule's categories: Rules, Analysis Rules, and Suggested Rules. Rules table presents rules of educational institution to rank courses and determining applicability of course import. Analysis rules table is the table that holds analysis rules will be used by Analyzer agent to apply them on failed imports, in order to generate suggested rules. Suggested rules table is the table that holds either new rules or edited current rules suggested by Analyzer agent and waiting for approval by Business Manager. Figure 4.23 depicts tables required by Manage Rules Service

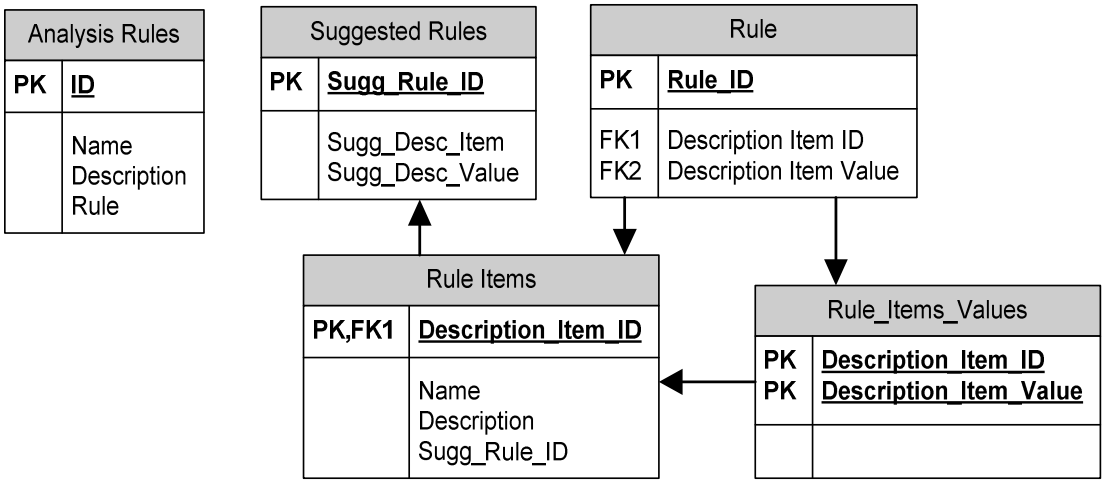


**Figure 4.23: Tables of Manage Rules Service**

### 3.2.2 Raise Exception Manage Rules Service

Activities performed by Raise Exception Manage Rules Service are depicted within Figure 4.16. Raise Exception Manage Rules Service requires only log table in order to add entry. Figure 4.21 includes Log, Events, and Description_Items tables design.

## 3.3 Software Agents Design

This part will cover the design of two of the systems software agents. Those are: Tracker, and Ranker.

### 3.3.1 Tracker Agent

Proposed CMS depends heavily on instructor's feedbacks to enhance system's performance. In proposed system, tracking focuses mainly on instructors filling their feedback forms. Every instructor imported a course should give her/his feedback for this course. Imported courses feedback tracking requires consumption of three Web services. Figure 4.24 depicts tracking required tables.

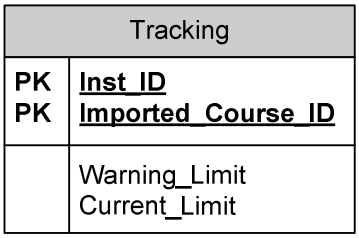


**Figure 4.24: Imported Courses Tracking Required Table**

### 3.3.2 Ranker Agent

Course Ranker consumes two services: Retreive Course, and Update Course. Pedagogical and managerial rules are presented with different priorities to reflect system's interest. Figure 4.25 depicts ranking rules required tables. Ranking is either based on instructor's feedback, or mathematical formula. Courses, Instructors, and Faculties ranking is based on instructors' feedback for those different items. Several feedback forms with one to five grading are presented to instructors to evaluate different items. Rank items and inserted values are saved, and Ranker gets average of all those items to present final rank of any of the saved items. Thresholds are presented in different tables to enable Ranker to classify ranked item either accepted 'ranking value over threshold' or not accepted 'ranking value is below threshold'. Universities ranking is presented as a formula that is the average of all faculties composing this university ranks.

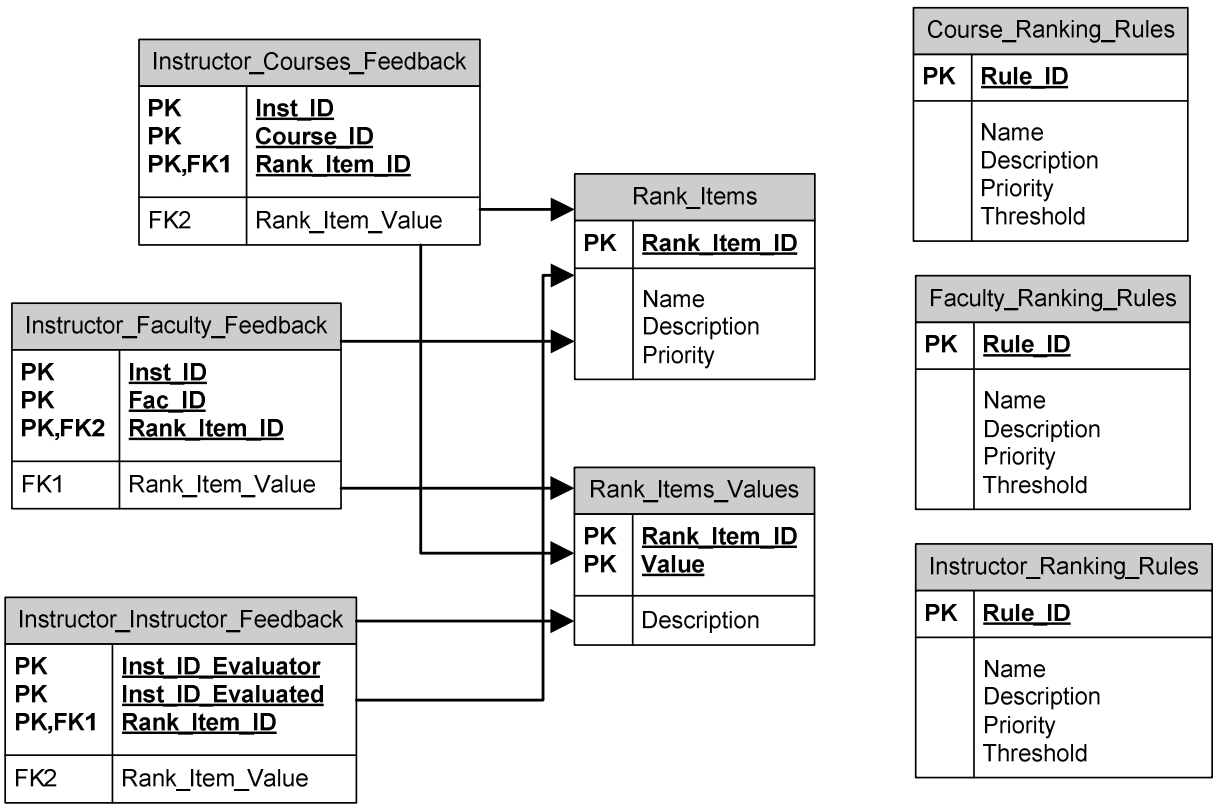


**Figure 4.25: Tables of Ranker Software Agent**

## 4. Summary

This chapter presented a proposed CMS that facilitates integration among different CMSs. An automated course search, import, and deposit process is presented. This

automated process requires governance of business rules. Business rules might be limiting system efficiency, so they must be monitored and modified when needed. Manage business rules process was presented to achieve this goal. Utilizing SOA to integrate Web services and software agents in CMSs highlighted the unlimited advantages of Web services and its capabilities to facilitate software agents integration within systems. CMS should be thought of as a collection of stateless Web services. SOA provides a fine granularity and modularity that solves many integration problems, but adds complexity to systems design. SOA is a design pattern that helped enterprises overcome integration obstacles, and gain agile and interoperable advantages within architectures. Pedagogical, social, and managerial advantages of added processes include:

- Overcome lack of internal courses.
- Get use of external, higher pedagogical features courses.
- Shareability among different educational institutions.
- Competition increment adds to quality (indirect effect).
- Increase Return-On-Investment by selling courses.

# Part III

# Chapter Four – Part III

# ASSESSMENT MANAGEMENT SYSTEM

## 1. Introduction

Assessment is one of the learning activities that can be achieved electronically and via mobile devices. Assessment Management System (AMS) is part of LMS that is responsible for providing, managing, automating, and facilitating the assessment process. Mobile assessment adoption in current LMS requires architectural modifications to reflect interoperability characteristics required by mobile supported processes. Mobile Learning (M-Learning) is an approach to E-Learning that utilizes mobile devices and should be enabled by educational institution Learning Management System (LMS). Unfortunately, M-Learning is not the same at all educational institutions. Mobile assessment refers to the capability of conducting assessments via mobile devices. Mobile assessment relies on external services that are not part of the LMS.

## 2. AMS Analysis

Mobile assessment utilizes mobile services architecture to deliver interactive messaging automatically. Interactive messaging include sending assessment questions and receiving multiple responses SMSs. Figure 4.26 presents Mobile Services Architecture required to enable learners to interact via mobile SMS with LMS that resides in university server. Mobile services architecture can have different implementations. Understanding the mobile service architecture is critical for the design and implementation of LMS.

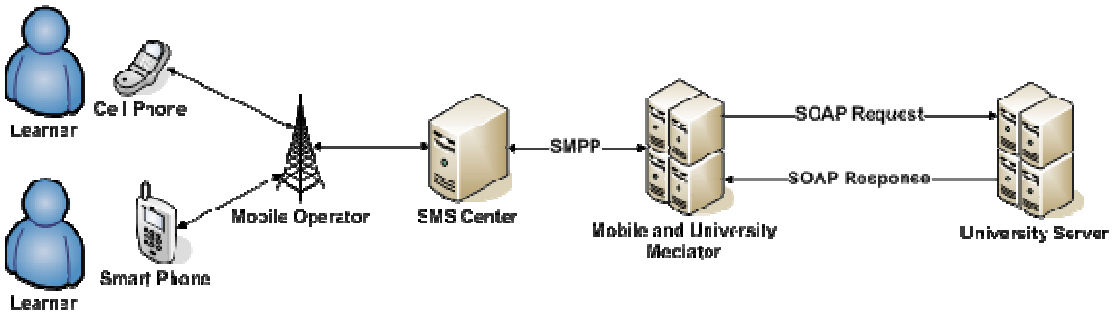


**Figure 4.26: Mobile Services Architecture**

Learner is connected to her/his mobile service provider via cell or smart phones. Mobile operators implement one or more SMS centers. Those are centers that manage sending and receiving of SMSs. University needs a mediator in the way to SMS center. Mediator

lies in the middle between mobile operators and LMS. Mediator connects directly to different SMS centers using Short-Message Peer-to-Peer (SMPP) protocol. SMPP is a telecommunication industry protocol for exchanging SMS messages between SMS peer entities such as SMS center. It is often used to allow third parties to submit messages in bulk. SMPP has been designed to offer services for various cellular networks such as GSM, CDMA, and TDMA. Mediator is connected directly the University server over the Internet, via standard Web services. Mediator receives and sends SMSs and exposes two Web service that give LMS the capability to access SMSs received, and to send SMSs. Via SOAP request, university can receive SMSs aimed to it, and via SOAP response, university can send new SMSs. University LMS should manage sessions with different users, utilizes data within SMSs in managing learners profiles. LMS shall maintain a record of all sent and received SMSs for managerial, financial, and educational issues. Proposed extended LMS addresses a new added process namely Take Mobile Assessment as depicted in the use case diagram at figure 4.27. Figure 4.28 presents AMS entities.

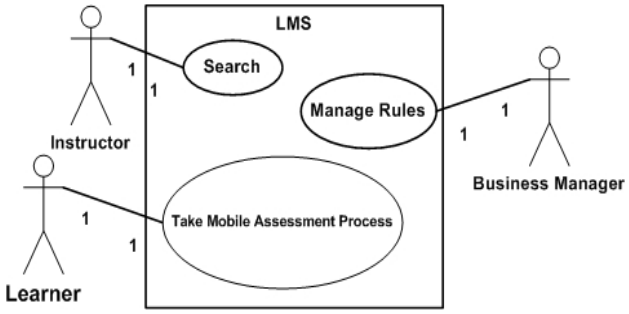


**Figure 4.27: AMS Use Case**

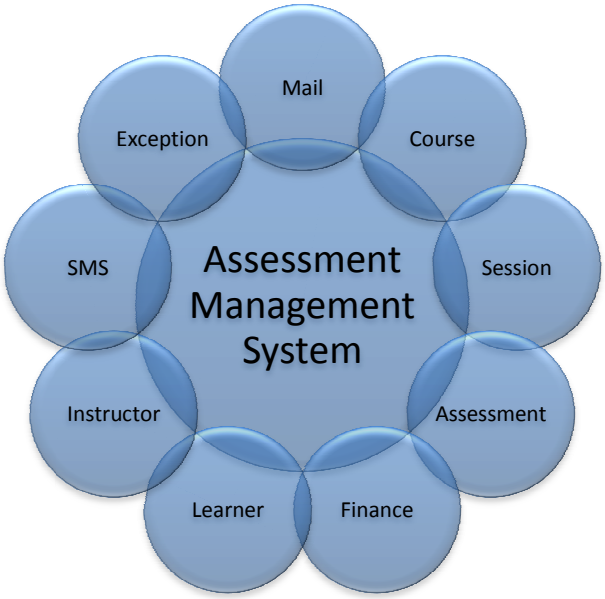


**Figure 4.28: AMS Entities**

### 2.1 Proposed AMS Architecture

Proposed architecture as presented in figure 4.29 consists of two layers: Interface layer, and Service layer. Interface layer interacts with instructors, learners, and business managers via human interface, that is portals, and with external organization services via machine interface, that is Web services. Service layer contains core LMS services and has three sub layers: Orchestration, Application Services, and Agents layer. Orchestration layer holds business logic presented by system processes as executable services. Business logic refers to different activities that can include Web services invocations, data manipulation, exception handling, and process termination. Application Services layer contains set of stateless Web services that are capable of performing certain tasks related to system entities. System entities reflect distinguishable objects within system that should be reflected in the LMS, like instructors, courses, and assessments. Web services are stateless because they can not maintain business logic, operation flow, or user state. Agents layer presents the suggested required software agents to serve the overall system. Agents layer presents Tracker software agent that keep track of learner assessments due dates.

### 2.2 Take Assessment Process Analysis

Figure 4.30 presents Take Mobile Assessment process analysis. Take Mobile Assessment process is initiated by learner, managed and maintained by Assessment Manager, and Session Manager. Assessment Manager and Session Manager are both Orchestration layer services. Managing and maintaining take mobile assessment process refers to keeping track of assessment items retrieved, validating learner's answers, handling exceptions, and maintaining actions sequence and Web services invocations. Figure 4.30 implicitly defines LMS boundary for take mobile assessment process. Out of LMS scope refers to activities fall outside LMS boundary and can not be maintained, neither controlled by LMS. Activities related to mobile operators and mediators are out of LMS boundary. LMS scope activities are handled, maintained, and controlled by LMS. Take Mobile Assessment process begins for LMS once it retrieves SMS that is aimed for assessment. Take Mobile Assessment process is an example of processes that contain activities outside organizational boundaries.

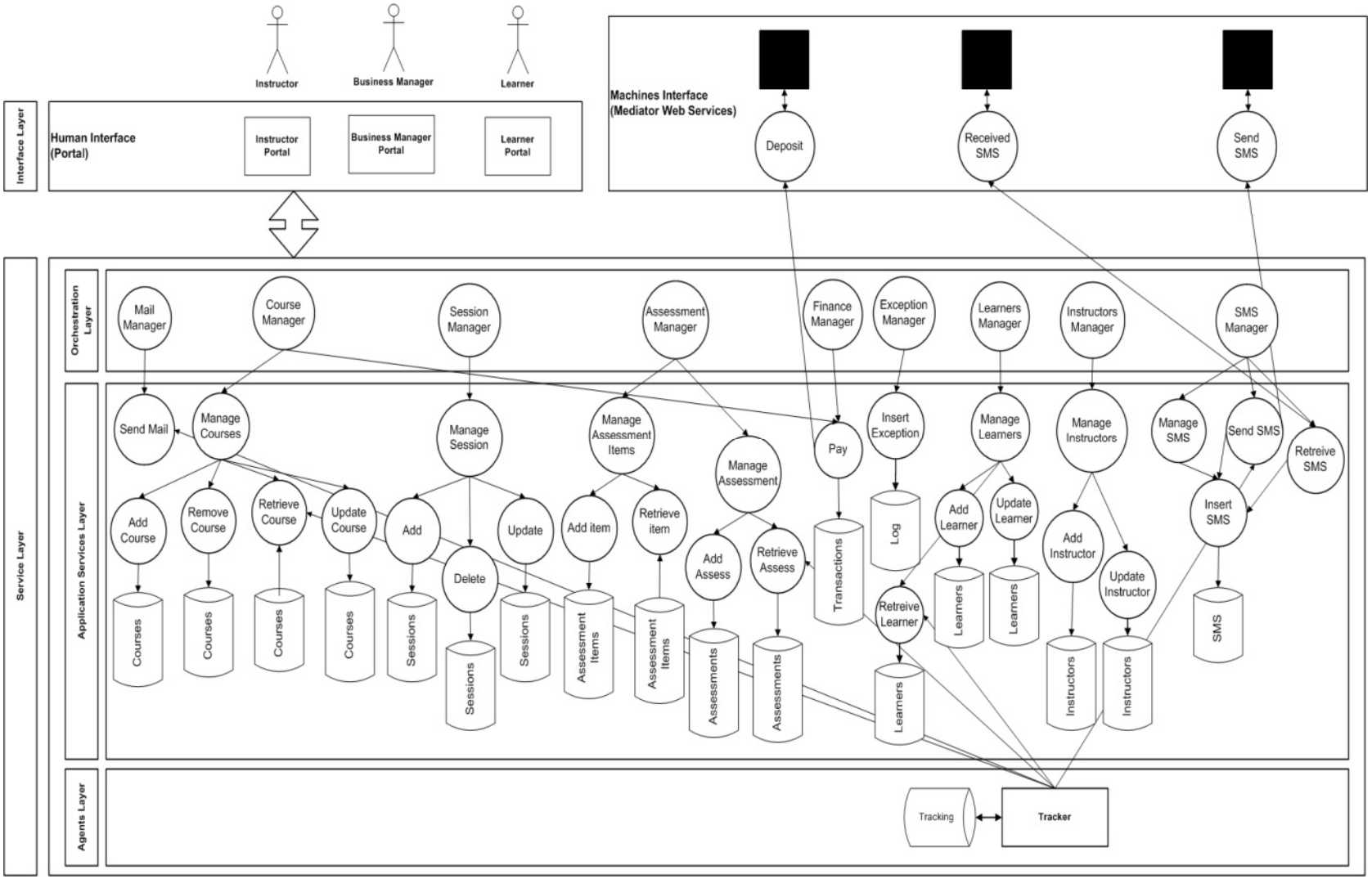


**Figure 4.29: Proposed AMS Architecture**

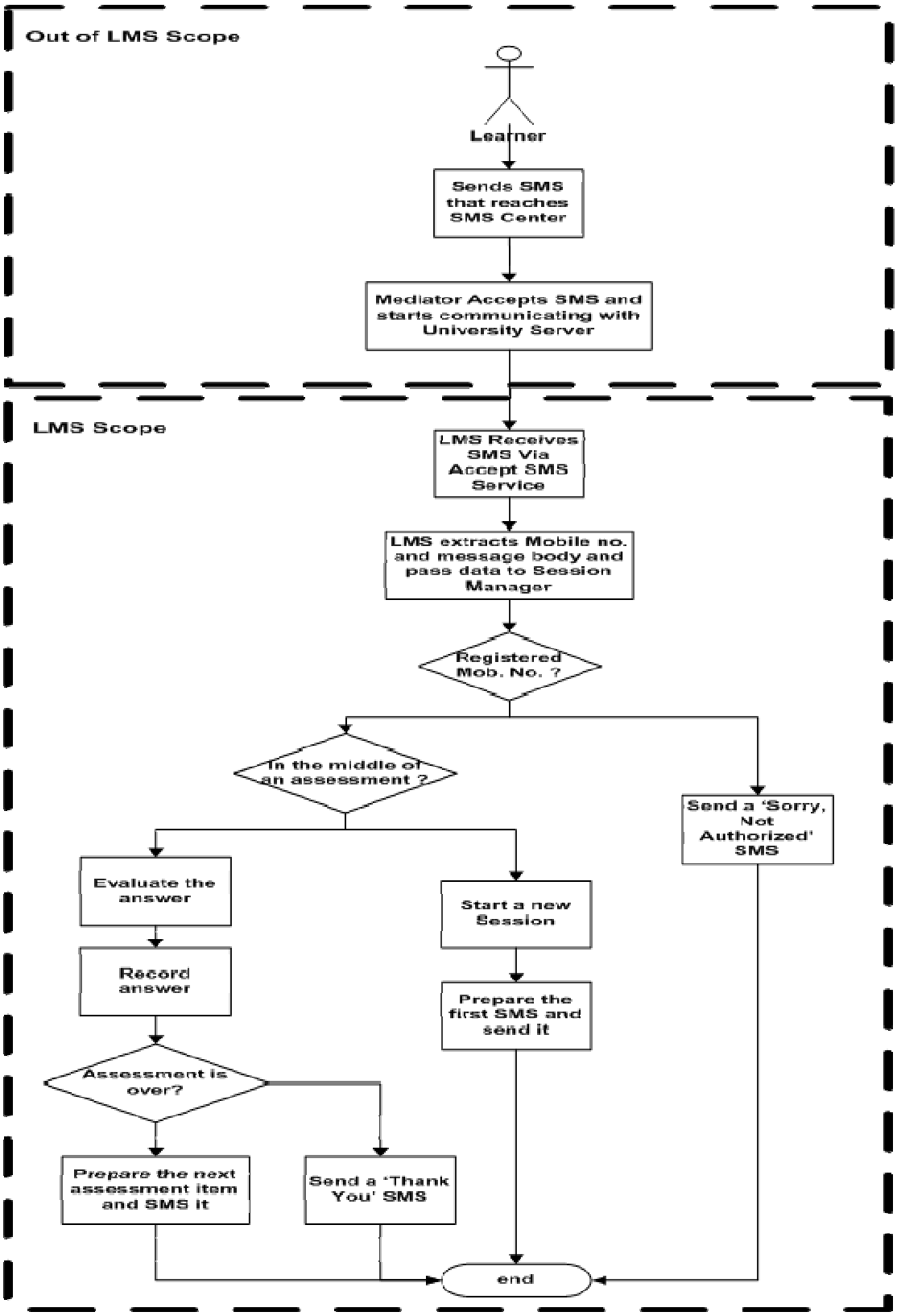


**Figure 4.30: Take Mobile Assessment Process Analysis**

## 2.3 Tracker Agent

The role of software agents in tracking users' activities has been widely known and accepted. Figure 4.31 illustrates analysis of tracking process. Tracking process is initiated and performed by tracker agent and aims to track learners missing assessments and take some action to remind students with due dates of assessments. Tracking process consumes five Web services: Read Learner Data, Read Course Data, Read Assessment Data, Send Mail, and Send SMS.

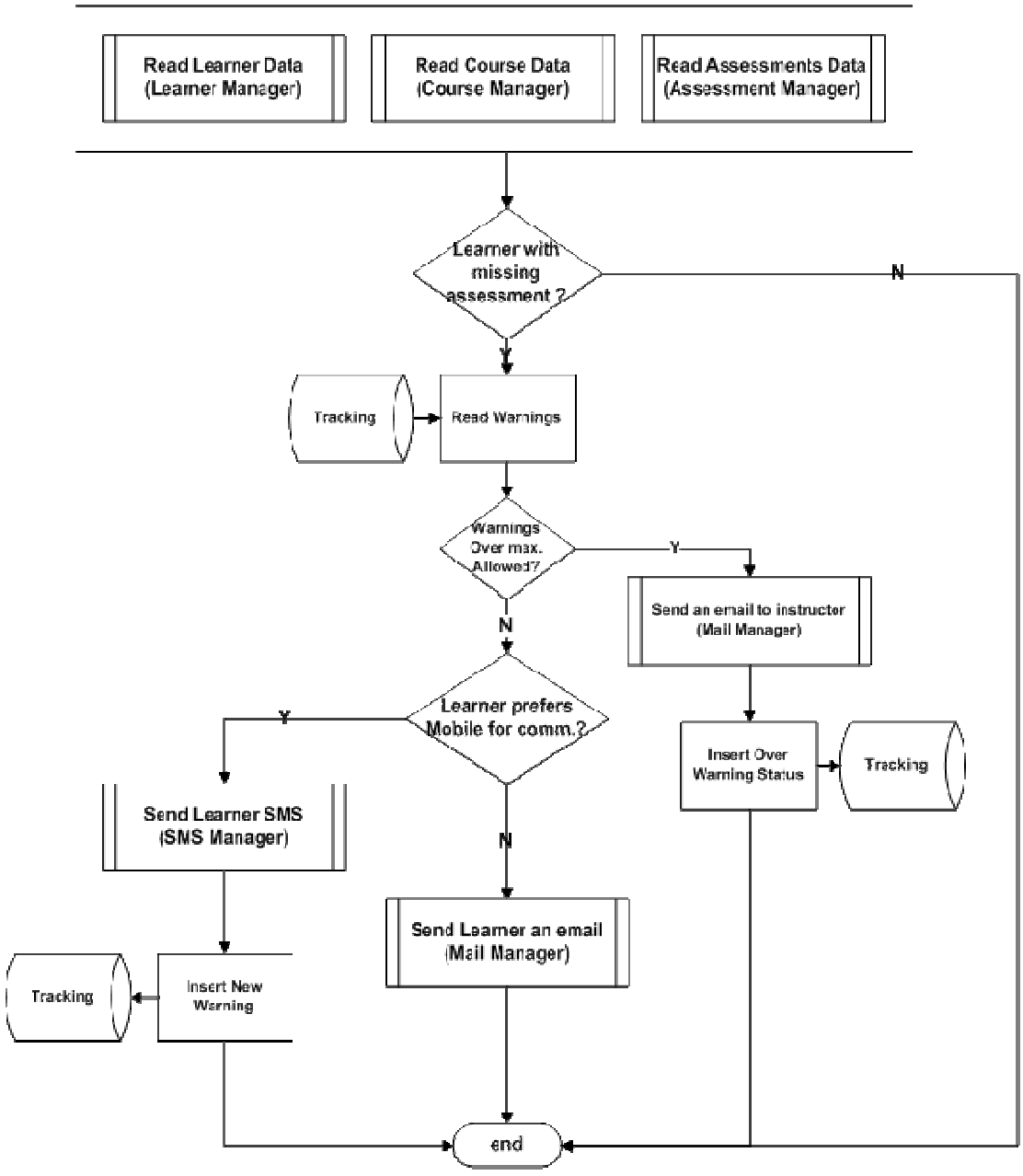


**Figure 4.31: Tracking Process Analysis**

## 3. AMS Design

AMS design task includes automating the new added process, and designing the required software agent. Automating take mobile assessment process includes partitioning the process into reusable services, designing services, and designing database tables required to support each service.

### 3.1 Take Mobile Assessment Process Design

Figure 4.32 presents Take Mobile Assessment Process design. Take Mobile Assessment Process is managed and maintained by assessment manager, and session manager. Take mobile assessment process utilizes Web services namely: Manage SMS, Manage Learner, Session Manager, Manage Assessment Items, Manage Session, and Insert Exception. LMS services can be reused to serve different processes. Design of Manage SMS, Manage Learner, Manage Assessments, Manage Assessment Items, Manage Sessions, and Session Manager Services are presented in details.

#### 3.1.1 Manage SMS Service Design

Manage SMS presents a collection of services that include Retrieve SMS, Send SMS, Insert SMS, and Delete SMS. Retrieve and Send SMS are two services that map mediator exposed Web services. LMS maps exposed mediator's Send and Receive SMSs Web services into internal Retrieve and Send SMS. When LMS invokes both services, it is actually invoking Mediator's Send and Receive SMS services. This mapping is tended to act as an isolation layer to enable changing mediator upon need. Insert SMS keeps a log of all received and sent SMSs. LMS maintains a record of all sent and received SMSs for managerial, financial, and educational issues. Records can be used to calculate fees to be paid for mediator monthly according to usage statistics. Figure 4.33 shows the required database table for Insert SMS service that is invoked every time a SMS is sent or received. Delete SMS utilizes the same database table and is invoked periodically to clear SMS record from too old SMSs based on date and time basis.

#### 3.1.2 Manage Learner Service Design

Manage Learner encapsulates the three primary database operations insert, update, and delete. Learners' data and profiles are maintained and updated based on attended assessments. Figure 4.34 presents required database tables to maintain learners' data and profile. Course_Assessments table holds assessments of each course. Learner_Courses table holds courses attended by each learner. Learner_Assessments table holds assessments attended by learner, and related data like student score for this assessment, date, and time student attended the assessment.

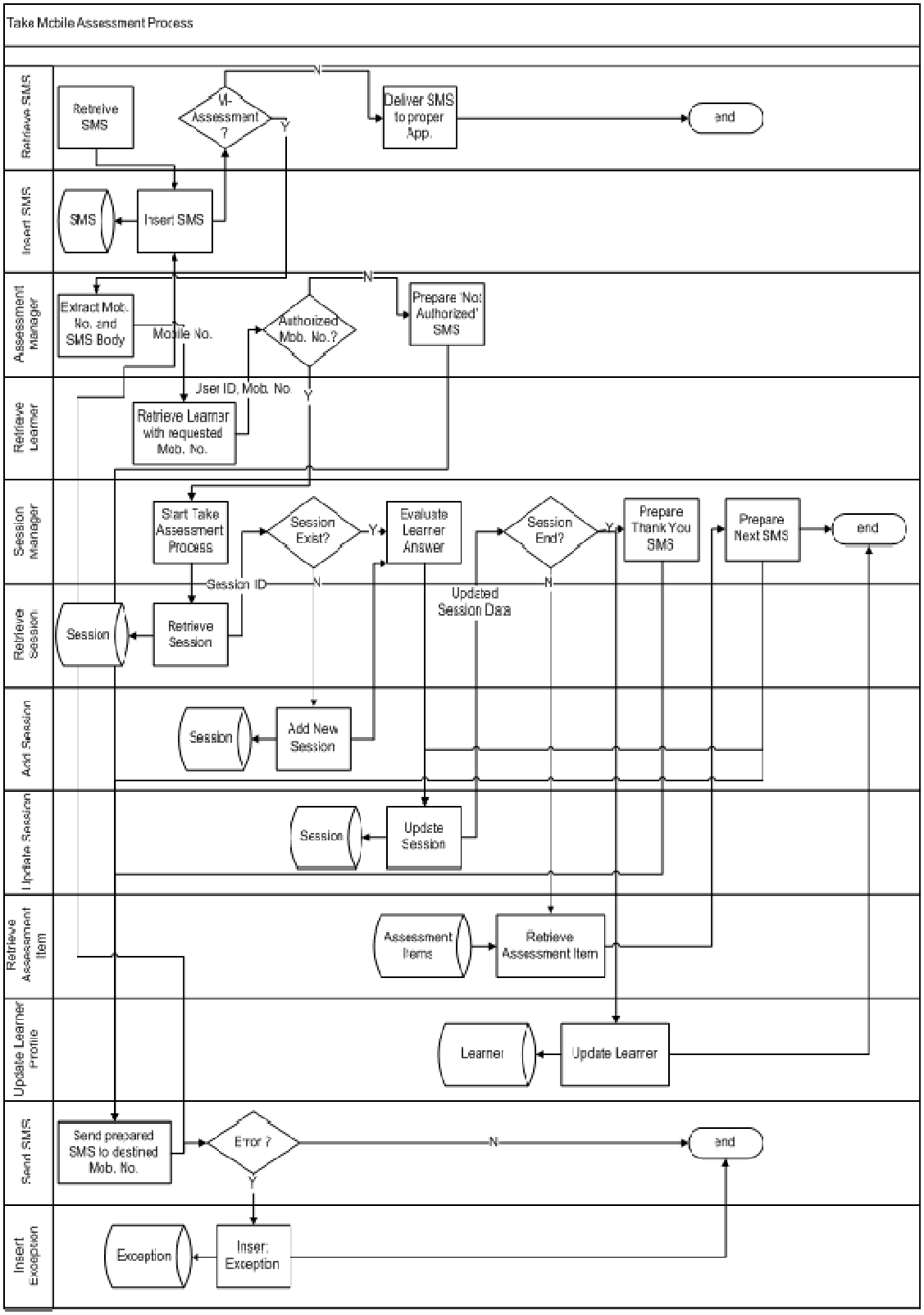


**Figure 4.32: Take Mobile Assessment Process Design**

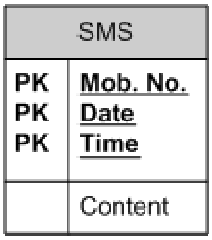


**Figure 4.33: Table of Insert SMS Service**

Assessment dates are stored, so LMS can determine learners missing assessments to be inserted in missing_assessments table. Learner_Missing_Asse table holds assessments that learners should have attended but they did not. Learner_Missing_Asse is used by tracker agent to remind students of missing assessments.

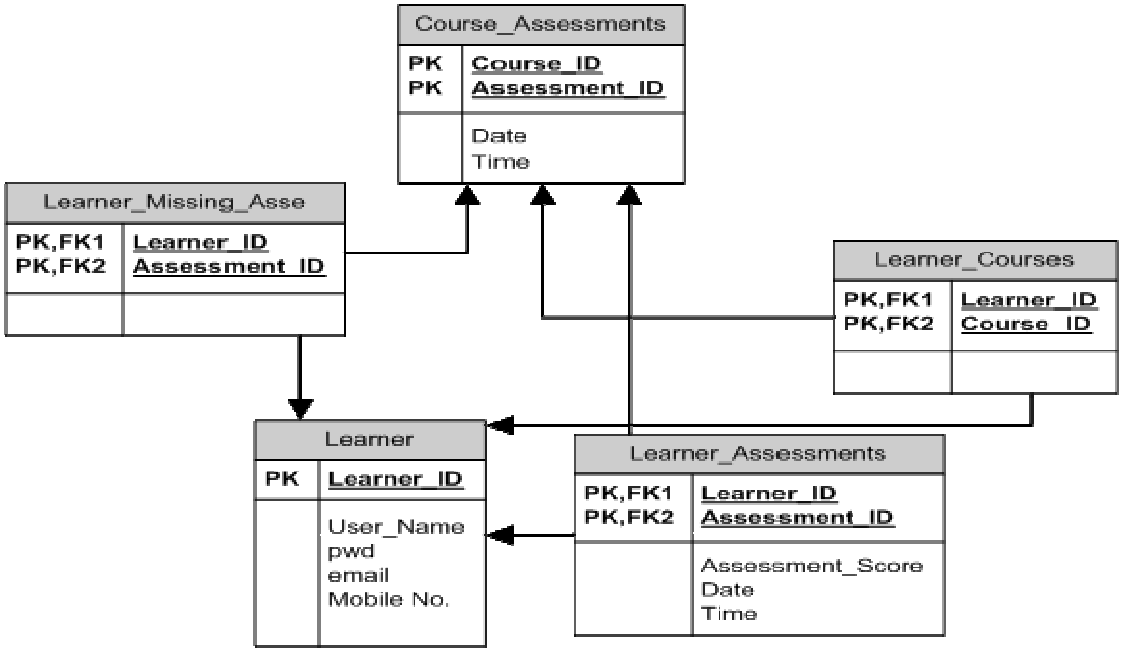


**Figure 4.34: Tables of Manage Learner Services**

### 3.1.3 Manage Assessments Service Design

Figure 4.35 shows required database tables to store and manage assessments. Manage assessments service is concerned with the three main database operations insert, update, and delete assessments themselves, not assessment items. Assessments should contain variant difficulty levels of assessment items. Assessments are categorized according to difficulty level into easy, medium, and difficult. Instructor is responsible for categorizing difficulty level of each assessment from educational perspective. An automated assessment categorization can be achieved via calculating percentage of difficult, medium, and easy assessment items composing the assessment.

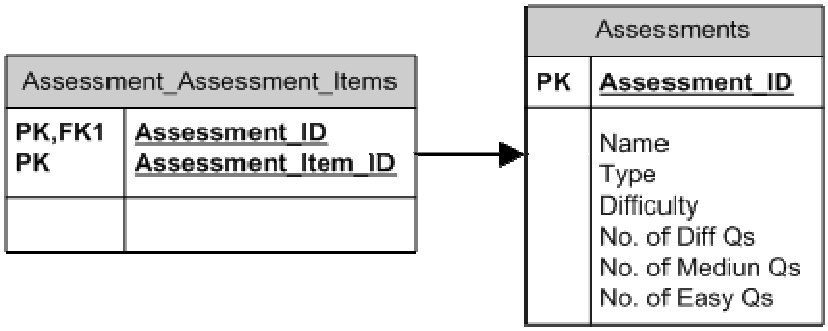


**Figure 4.35: Tables of Manage Assessments Services**

### 3.1.4 Manage Assessment Items Service Design

Assessment items are the main factor of assessments to either achieve required goals or not. Assessment items need to be well prepared to enable assessment efficiency. Assessment items are stored and managed separately to form an assessment items repository. Assessment items repository enable sharing assessment items among different assessments to overcome interference of topics among courses and to enrich assessment items bank. Sharing assessment items among assessments is done under instructors' supervision. MCQs, True/False, and Complete the missing word are examples of applicable mobile assessment items. From assessment items difficulty point of view, LMS has three classes: easy, medium, and hard. Assessment item difficulty level is determined by instructor either while inserting or updating assessment item. Figure 4.36 depicts the required database tables for manage assessment items service.

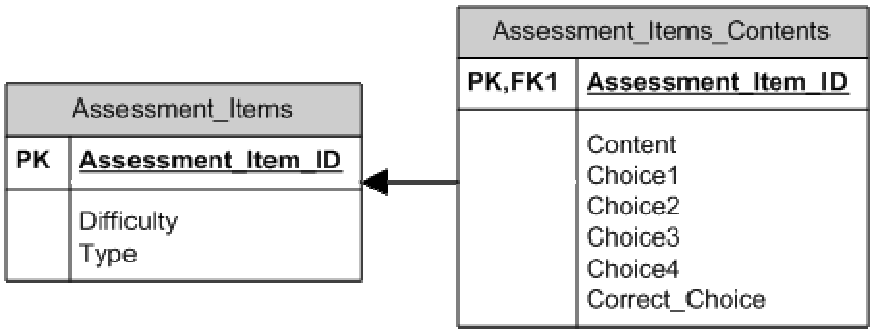


**Figure 4.36: Manage Assessment Items Services**

### 3.1.5 Session Manager Service Design

Session refers to the period of time in which the same user interacts with the system. Mobile user interacts with the system via multiple discrete responses. Web services are stateless, and sessions are needed to enable stateless protocols to track user interaction with the system. In LMS, Session refers to the period of time the learner is identified by the system as in the middle of a process. Session managers maintain users' interactivity data temporarily in the database. Session Manager Service holds the business logic

required to enable the supported processes. Temporary database accesses are left to manage session service. Session manager activities include recognizing either the learner is in the middle of an assessment or attempting to a new one, and then invoking the suitable Web services based on user state. Some of the Web services session manager invokes are add session, update session, update learner profile, and send SMS.

### 3.1.6 Manage Session Service Design

Manage session service is responsible for performing the three main database operations insert, update, and delete sessions. Figure 4.37 shows the required tables for manage session services. Session data include a unique session ID that becomes available for reuse when session ends, Learner ID that reflects the learner unique ID stored for each learner, learner mobile number, and score of the learner in the assessment. Score increases when learner correctly answer assessment item. Another database table is required to keep track of assessment items sent to the learner during the session, so sending of repeated assessment items is avoided.

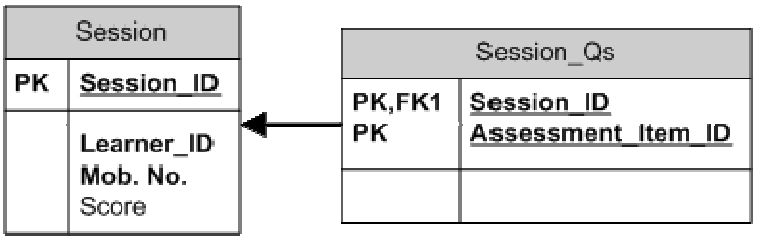


**Figure 4.37: Tables of Manage Session Services**

### 3.1.7 Manage Exception

Exceptions refer to non-ordinary flow of events or actions. Well defined information systems should address and handle exceptions carefully. Feed forward information systems needs to keep record of exceptions took place to be analyzed. Software agents can perform analysis functions on recorded exceptions. Figure 4.38 depicts the database table required to keep exception log. Exception log is available for business manager and system administrator to maintain system state. Figure 4.39 shows the overall implemented Assessment Management System database tables.

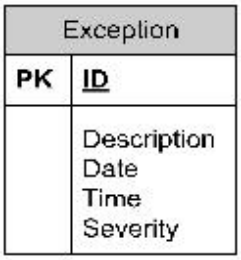


**Figure 4.38: Table of Insert Exception Service**

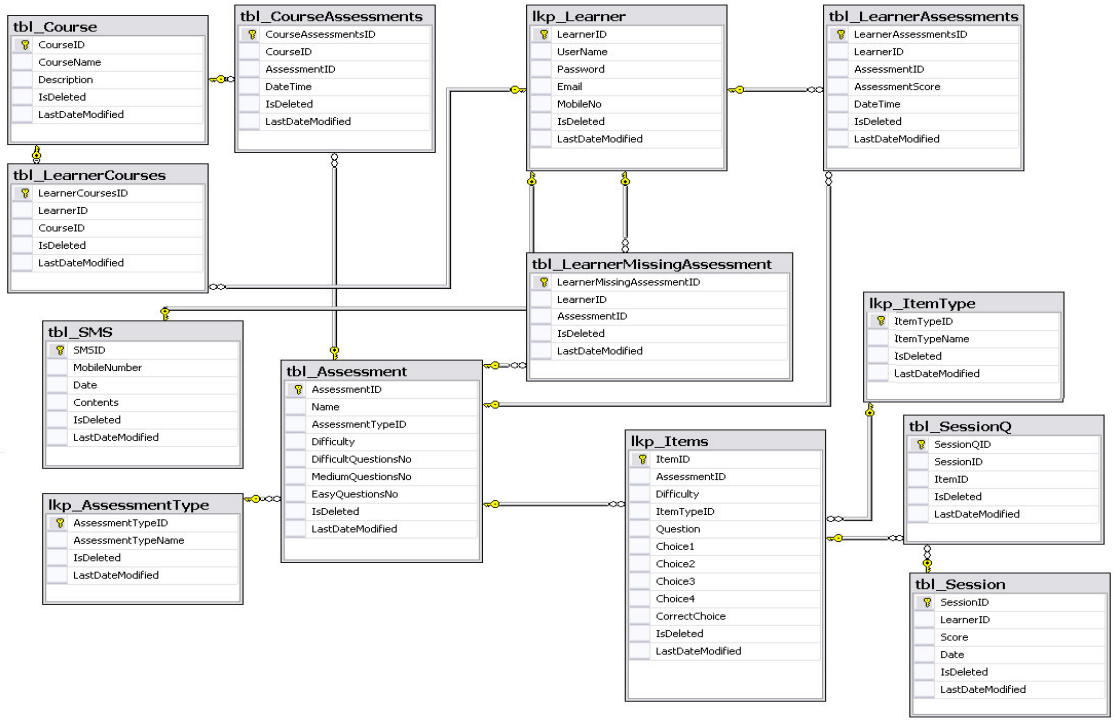


**Figure 4.39: Overview of AMS Database Tables**

## 3.2 Tracker Agent

Tracking process consumes five Web services: Read Learner Data, Read Course Data, Read Assessment Data, Send Mail, and Send SMS. Figure 4.40 presents the database table required for tracker to keep track of current submitted learners warnings, and the maximum number of allowed warnings. Tracker agent gets information about the learner missing assessments by utilizing the manage learner.

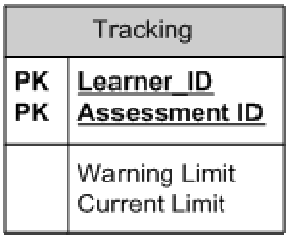


**Figure 4.40: Table of Tracker Agent**

## 3.3 AMS Classes

Figure 4.41 presents Services layer SMS Manager class diagram. Figure 4.42 depicts Assessment Management System class diagram,

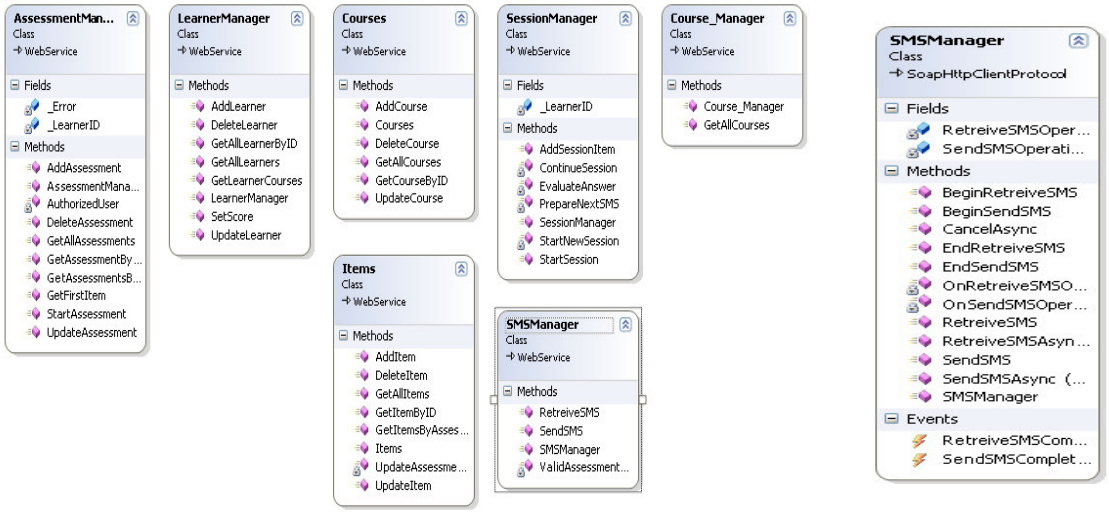


**Figure 4.41: Service Layer and SMS Manager Class Diagram**

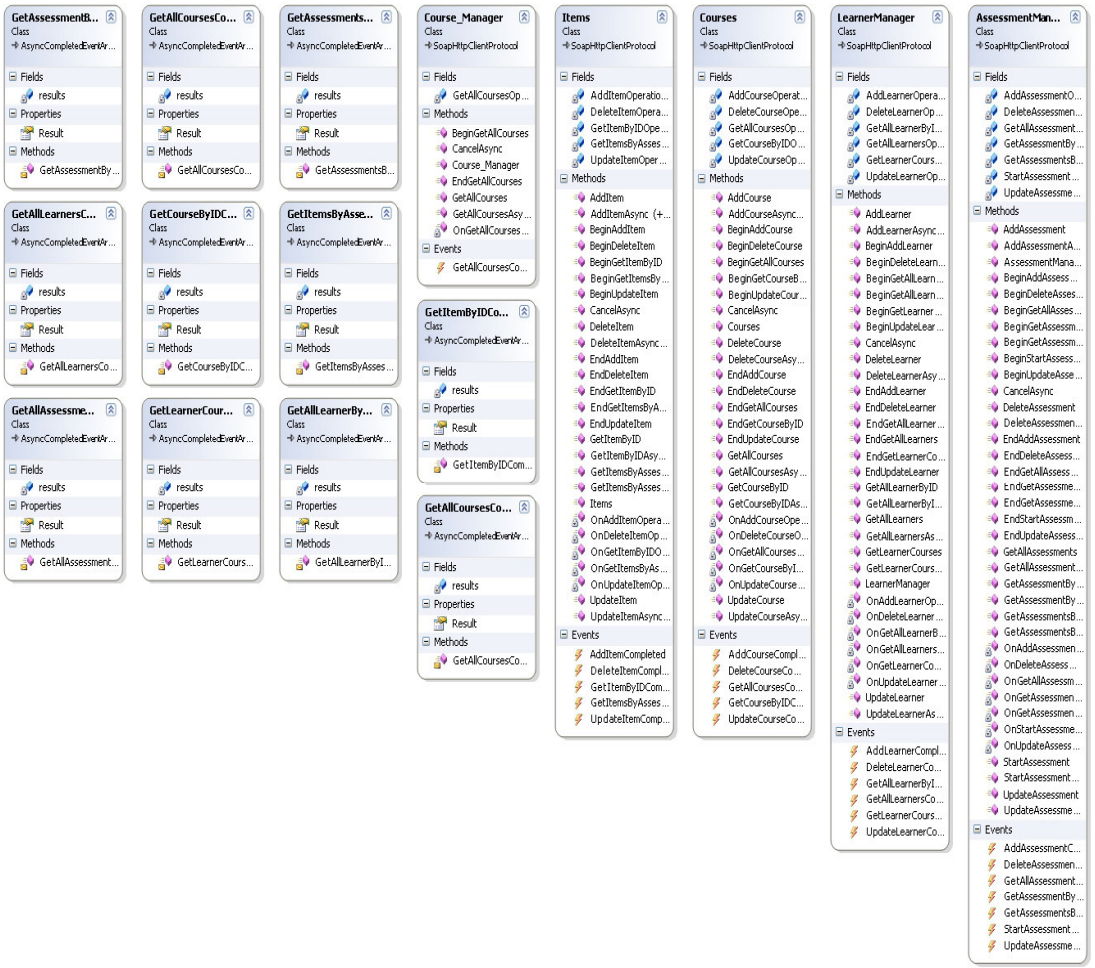


**Figure 4.42: AMS Class Diagram**

## 4. Mobile Simulator Interface Design

Figure 4.43 shows the mobile simulator interface available via the portal to test the AMS mobile functionality.


**Figure 4.43: Mobile Simulator**

## 5. Learner interaction with LMS

Take Mobile Assessment process relies on the sequence of SMSs learner sends and receives with the LMS. Figure 4.44 shows a classical interaction between learner and Assessment Management System during the take mobile assessment process. Highlighted shapes present SMSs sent by learner, and other shapes present SMSs received. Learner sends SMS with assessment ID, receives the first assessment item, responds with the appropriate choice, and finally receives a summary with the result. The receive assessment item and response sent by learner operation is repeated as many times as assessment items for each assessment.

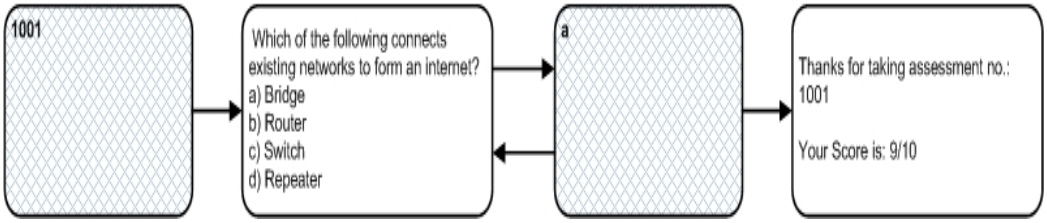


**Figure 4.44: Classical Learner – AsMS Mobile Interaction**

## 6. Summary

Assessment is an integral part of the learning process, and a learning activity that can be achieved efficiently via mobile devices. Due to differences in mobile architectures, and as a result of lack of interoperability in current commercial LMS with external systems, mobile assessment may find difficulties in implementation. SOA as a design pattern addresses non functional requirements like interoperability and integration to achieve

systems agility. SOA utilizes Web services as main enabler to achieve addressed design goals. Interoperability is a main challenge for systems adopting and interacting with external components that. Interoperability between LMS and mobile mediator is achieved via adoption of Web services based SOA in LMS. SOA provides a fine granularity and modularity via standard interface that solves integration and interoperability challenges, but adds complexity to systems design. Pedagogically, mobile assessment is important due to the assessment importance process to the learning process. Mobile assessment facilitated and encouraged learners to attend assessments and enabled distance education by expanding interactivity tools available to learners to include mobile devices.

## 7. Interface Design

Figure 3.22 shows the mobile simulator interface available via the portal to test the assessment management system mobile functionality. Figure 3.23 presents Assessment Management System entities. Extended Assessment Management System Design is available in Chapter 4 – Par I.


**Figure 3.22: Mobile Simulator**

Summary

Assessment is an integral part of the learning process, and a learning activity that can be achieved efficiently via mobile devices. Due to differences in mobile architectures, and as a result of lack of interoperability in current commercial LMS with external systems, mobile assessment may find difficulties in implementation. SOA as a design pattern addresses non functional requirements like interoperability and integration to achieve systems agility. SOA utilizes Web services as main enabler to achieve addressed design goals. Interoperability is a main challenge for systems adopting and interacting with external components that. Interoperability between LMS and mobile mediator is achieved via adoption of Web services based SOA in LMS. SOA provides a fine granularity and modularity via standard interface that solves integration and interoperability challenges, but adds complexity to systems design. Pedagogically, mobile assessment is important due to the assessment importance process to the learning process. Mobile assessment facilitated and encouraged learners to attend assessments and enabled distance education by expanding interactivity tools available to learners to include mobile devices.

# Chapter 5

## Chapter Five

# EVALUATION OF PROPOSED ARCHITECTURE

### 1. Introduction

This chapter answers the many questions arose about SOA efficiency and capabilities regarding overcoming UMIS and LMS limitations and presents the results of evaluating the proposed UMIS and LMS SOA based components. From information system point of view, points like performance, integration and interoperability, compliance, security, maintainability, analyzability, decomposability and modularity, testability, portability via replaceability and scalability, simplicity, modifiability, and reusability are addressed. A Comparative performance analysis study is available to test SOA based systems user-perceived performance against non-SOA based systems. Pedagogically, SOA adoption needs to enhance the learning process activities and provide capabilities that was hard to present before.

### 2. Performance

One of the main reasons to focus on styles for network-based applications is because component interactions can be the dominant factor in determining user-perceived performance and network efficiency. Since the architectural style influences the nature of those interactions, selection of an appropriate architectural style can make the difference between success and failure in the deployment of a service-based application. Performance refers to both Network and User-perceived performance.

### 2.1 Network Performance

Service Oriented Architecture based application relies heavily on messaging. Debates that relying on messaging causes delays more than non-messaging applications. Besides, it is clear now that SOA based applications need to add extra headers to manage requests and responses in standard format. Network performance of proposed SOA based architecture was evaluated against non-SOA based architecture.

**Types of Delay in Packet-Switched Networks are:**

- **Processing Delay**: The time required to examine the packet's header and determine where to direct the packet is part of the processing delay. The processing delay can also include other factors, such as the time needed to check for bit-level errors in that packet that occurred in transmitting the packet's bits from the upstream node to router A. Processing delays in high-speed routers are typically on the order of microseconds or less.
- **Queuing Delay:** At the queue, the packet experiences a queuing delay as it waits to be transmitted on the link. The length of the queuing delay of a specific packet will depend on the number of earlier-arriving packets that are queued and waiting for transmission across the link. If the queue is empty and no other packet is currently being transmitted, then packet's queuing will be zero. On the other hand, if the traffic is heavy and many other packets are also waiting to be transmitted, the queuing delay will be long. The number of packets that an arriving packet might expect to find is a function of the intensity and nature of the traffic arriving at the queue. Queuing delays can be on the order of microseconds to milliseconds in practice.
- **Transmission Delay:** Assuming that packets are transmitted in a first-come-first-served manner, as is common in packet-switched networks; packet can be transmitted only after all the packets that have arrived before it have been transmitted. Denote the length of the packet by $L$ bits, and denote the transmission rate of the link from router A to router B by $R$ bits/sec. The rate $R$ is determined by the transmission rate of the link from router A to router B. The transmission delay (also called the store-and-forward delay) is $L/R$. This is the amount of time that is required to push (that is, transmit) all of the packets bits into the link. Transmission delays are typically on the order of microseconds to milliseconds in practice.
- **Propagation Delay:** Once a bit is pushed onto the link, it needs to propagate to router B. the time required to propagate from the beginning of the link to router B is the propagation delay. The bit propagates at the propagation speed of the link. The propagation speed depends on the physical medium of the link. The propagation delay is the distance between two routers divided by the propagation speed.

Figure 5.1 shows one of the services extra header added by the service as the Request, while figure 5.2 shows the response format. The request and response are related to Insert Author service; that is one of the service designed and implemented in LIS.

Figure 5.1 includes text within dashed boxes that represent static header added every time when the service is invoked. Text outside dashed boxes presents header added once for every record. That means, based on what is depicted in figure 5.1, when there will be one author to add, static header will be repeated once for the request (that is the header inside dashed box) while header outside the dashed box will be repeated as many times as the number of authors will be inserted. Values (int, string, string, int) are replaced with real life author values.

```
POST /desktop_library/auther_desktop_library/Service.asmx
HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/auther_Insert"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <author_Insert xmlns="http://tempuri.org/">
      <author_id>int</author_id>
      <author_name>string</author_name>
      <nationality>string</nationality>
      <no_authrd_book>int</no_authrd_book>
    </author_Insert>
  </soap:Body>
</soap:Envelope>
```

**Figure 5.1: Insert Author SOAP Request**

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <author_InsertResponse xmlns="http://tempuri.org/">
      <author_InsertResult>bool</author_InsertResult>
    </author_InsertResponse>
  </soap:Body>
</soap:Envelope>
```

**Figure 5.2: Insert Author SOAP Response**

This comparative network analysis note will rely on the delay calculation formula, where:

**Total Delay = Transmission delay + Propagation delay + Processing delay + Queuing delay**

But it is well known that processing and queuing delay are less than micro seconds, so they are ignored, so the delay formula will be:

**Total Delay = Transmission Delay + Propagation Delay**

Transmission delay, where:

$$Dtrans = (M + N - 1)\, L / R$$

Where:

- M = no. of communication links
- N = no of packets
- L = packet size
- R = Transmission Rate

Transmission delay is affected by file size (F). In the previous formula F = N*L.

By analyzing data in figure 5.1, it is noticeable that there are three data categories:

- **Static Header:** This header occurs once for each service invocation no matter how many records in the request. Characters for this header = 463 characters
- **XML Tags:** Those tags are the overload of requests and responses. Those tags are named by developer, so they are not static every time, but in the presented example there are 179 characters
- **Actual Data:** Those are the record details to be inserted after invoking the Insert Web service.

So, Total extra characters to insert an author = Static Header + XML Tags (642 characters; more than half a packet size). So, file size in SOA will be:

$$F_{SOA} = F + SH + RH * R$$

Where:

- F = total data size without headers
- SH = Static Header
- RH = XML tags required to represent a single record
- R = no of records

This formula depicts that added extra headers differs according to the no. of records to be handled, and differs from an application to another (because the header used to represent an author might be different from the one used to represent a book) so network performance differs from an application to another. It is the system architect responsibility to decrease the transferred data over the network to the maximum extent (so decrease network delay) because it is clear now that the added headers by SOA is not the headers that can be neglected easily.

### 2.2 User Perceived Performance

User-perceived performance differs from network performance in that the performance of an action is measured in terms of its impact on the user in front of an application rather than the rate at which the network moves information. The primary measures for user-perceived performance is completion time. Completion is the amount of time taken to

complete an application action. SOA based Systems is evaluated regarding User-perceived performance against other two well known architectures. Three different Library Management System architectures were implemented and user-perceived performance was measured against same sample data and against same queries.

**Evaluated Architectures**

Three Library Management System (LIS) architectures were evaluated. The first one is the Services based LIS architecture presented in Chapter 3. The other two Library Management Systems are: Parametrized Query, and Stored Procedure based architectures. Figure 5.3 presents the Parametrized Query based architecture where SQL statements exist within the web pages and accesses database directly. Figure 5.4 highlights the separation between data layer and application layer by the presence of Stored Procedures as a middle layer in-between portal and database. Portal consumes stored procedures to access databases. Figure 5.5 shows the services based architecture with the services layer available in between the portal and database layer to present a standard based interface layer that consumes stored procedures and available for portals. Table 5.1 presents the Advantages and disadvantages of Parameterized Query adaption in software systems. Table 5.2 shows the advantages and disadvantages of Stored Procedure based software architectures. Services based architectures advantages were intensively discussed in chapter 2.

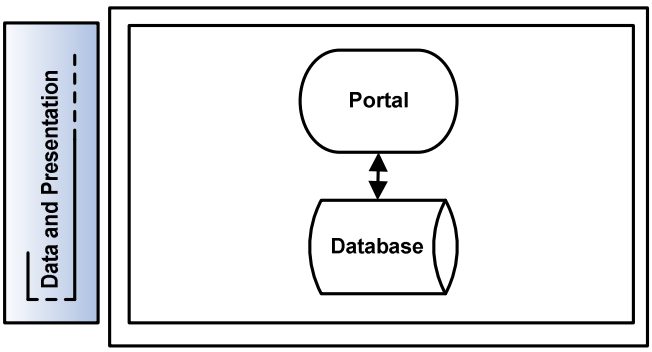


**Figure 5.3: Parameterized Query Based LIS Architecture**

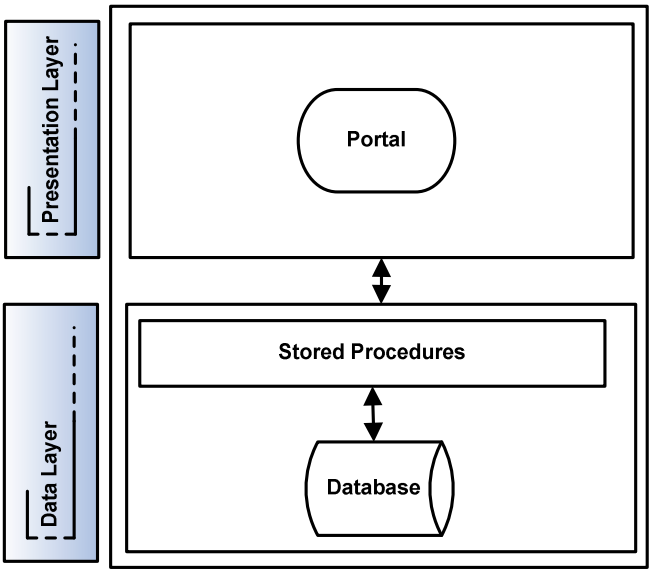


Figure 5.4: Stored Procedure Based LIS Architecture

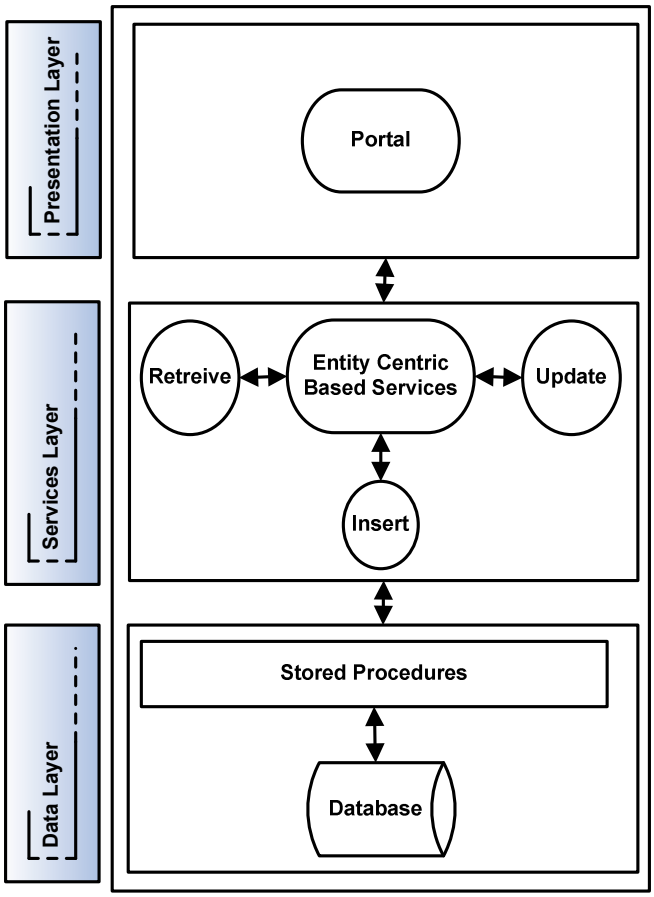


Figure 5.5: Services Based LIS Architecture

**Table 5.1: Advantages and Disadvantages of Parameterized Quereis**

| Parameterized Queries | |
|---|---|
| **Advantages** | **Dis-Advantages** |
| Code reusability for different tables, objects, and databases | Hard to debug, and error management becomes more unreliable |
| Use variable names in statements that require constants | Temporary tables from the main statement cannot be used, unless they are global |
| Return ROWSETs with a variable number of columns and / or variable column names | Algorithm time adds up to the time of dynamic SQL execution |
| Sorting by any column from a table | Difficulty of maintenance is difficult schema is hard coding in the dynamic code |
| | Security can be comprised with SQL injection |

**Table 5.2: Advantages and Disadvantages of Stored Procedures**

| Stored Procedures | |
|---|---|
| **Advantages** | **Dis-Advantages** |
| Improve security of database server | Slower than equivalent application code |
| Offer a mechanism to abstract data access routines which can improve code maintainability | Lead to application logic fragmentation between database and application tier |
| Reduce network traffic | Difficult to debug |
| Implement common routines and make them accessible from multiple applications | Most object-relational mapping systems cannot seamlessly exploit stored procedures |
| Database centric logic can be isolated | |
| Improve application portability by moving data logic into stored procedures | |

### Analysis Tools

More than one analysis tool has been used while evaluating and analyzing proposed architectures performance. Analysis tools include, not only, Firebug; an add-on Firefox. Firebug integrates with Firefox to put a wealth of web development tools while the system developer browses web based systems. Editing, debugging, and monitoring browsed web pages are applicable. Main features of Firebug include the capability to: Inspect and edit HTML, Visualize CSS metrics, Monitor Network activity, Debug and

profile JavaScript, measure performance and find bottlenecks fast, Quickly find errors, Explore the DOM, Execute JavaScript on the fly, and Log for JavaScript [20,21].

The most interesting feature of Firebug is the capability to watch the timeline of the web page unfolds. Firebug shows a bar for each file that displays when the file started and stopped loading relative to all the other files. Another interesting feature is Firebug's capability to Examine HTTP Headers that contain important information like the file mime type, web server type, caching directives, and cookie. HTTP headers have been used heavily in Services-based Architecture.

Firebug was used heavily in the developing process based on presented functionalities, but in the analysis process it was used mainly to measure the timeline of page unfolds so measure the user perceived performance of the three LIS actitectures.

**Performance Measures**

Arithmetic Mean (Average) is the value obtained by dividing the set of quantities by the number of quantities in the set, so the Arithmetic Mean for each presented architecture regarding each presented process = Total time consumed to perform operation / Total no. of records. Another statistical measure that is used to evaluate the performance of presented architecture is Mode, which is the Most Repeated Value for each operation. Figure 5.6 illustrates statistics of the Insert process for the three implemented Library Management System architectures. Table 5.3 is a summary of the arithmetic mean and mode for each architecture. Services based architecture presented the highest arithmetic mean and mode values with no much performance enhancement to be mentioned. Stored Procedure based architecture was the best for the insert operation.

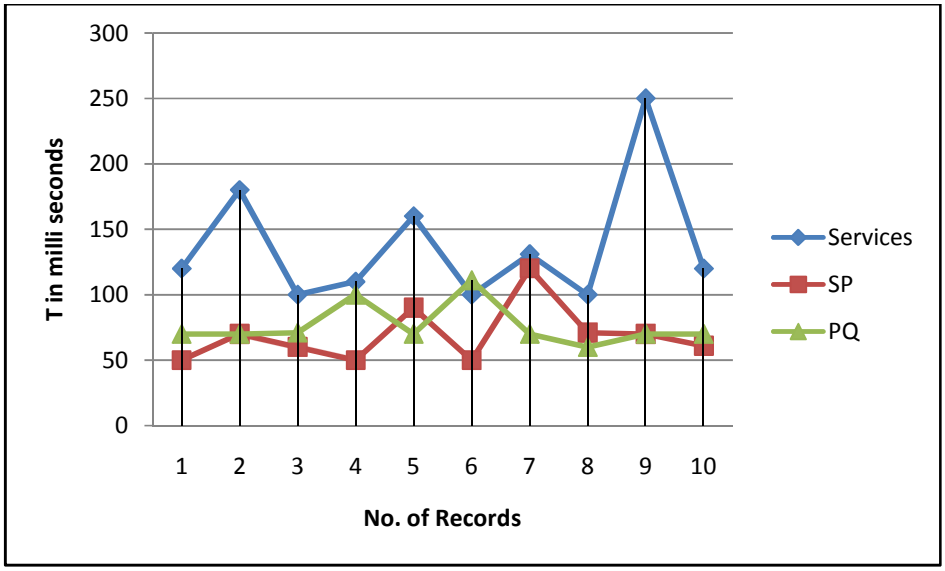


**Figure 5.6: Insert Performance Measures of the three LIS Architectures**

**Table 5.3: Insert Operation Measurements Summary – Measures are in Milli-Seconds per Record**

| Architecture | Arithmetic Mean | Mode |
|---|---|---|
| **Service Based** | 137.1 | 100 |
| **Stored Procedure Based** | 69.2 | 50 |
| **Parameterized Query Based** | 76.2 | 70 |

Figure 5.7 presents the statistics of the Update operation performed by the three implemented Library Management System architectures followed by table 5.4 that displays summary of arithmetic mean and average of the same operation. Services based architecture is the lowest in performance compared to stored procedure and parameterized query based architectures. Stored Procedures based architecture, is the best performance of the Update operation.

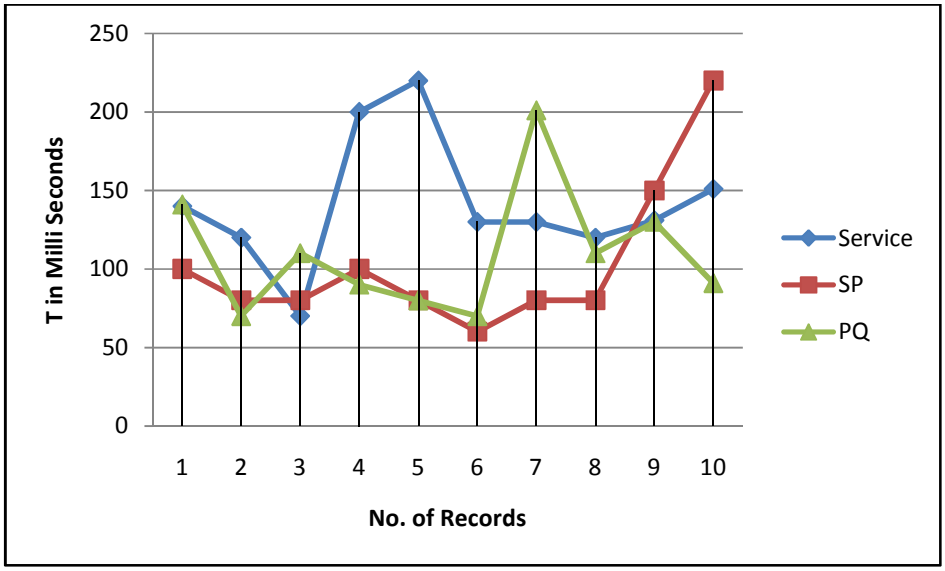


**Figure 5.7: Update Performance Measures of the three LIS Architectures**

**Table 5.4: Update Operation Measurements Summary – Measures are in Milli-Seconds per Record**

| Architecture | Arithmetic Mean | Mode |
|---|---|---|
| **Service Based** | 141.2 | 120 |
| **Stored Procedure Based** | 103 | 80 |
| **Parameterized Query Based** | 109.3 | 70 |

Figure 5.8 presents statistics of the Select By ID operation of the three implemented Library Management System architectures, followed by table 5.5 that summarizes the arithmetic mean and mode of the three presented performance measures. Services based architecture is the highest in ranges. While arithmetic mean and mode depicts that parameterized query based architecture performance is better than the stored procedure based one, it is noticed that parameterized query based architecture was highly affected by the amount of data retrieved, and its performance was not within small ranges; not like stored procedure one.

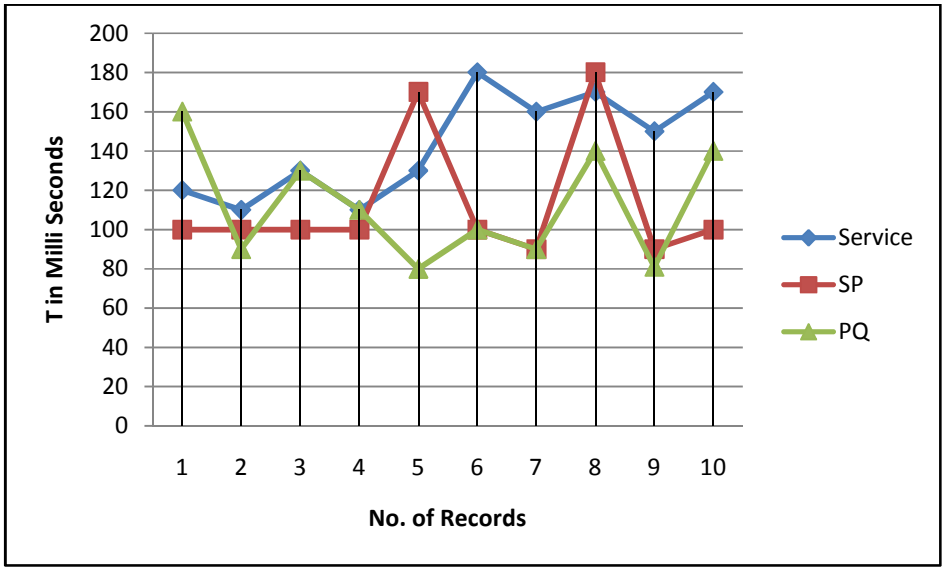


**Figure 5.8: Select By ID Performance Measures of the three LIS Architectures**

**Table 5.5: Select By ID Operation Measurements Summary – Measures are in Milli-Seconds per Record**

| Architecture | Arithmetic Mean | Mode |
|---|---|---|
| **Service Based** | 143 | 110 |
| **Stored Procedure Based** | 113 | 100 |
| **Parameterized Query Based** | 112.1 | 90 |

Figure 5.9 presents the total amount of time required by each of the three implemented Library Management System architecture to retrieve all data stored in the database, with no filter applied. Stored Procedure based architecture achieved the best time, services based architecture required time to retrieve the stored records exceeded the double time consumed by stored procedure based system, and parameterized query based system performance lies in between.

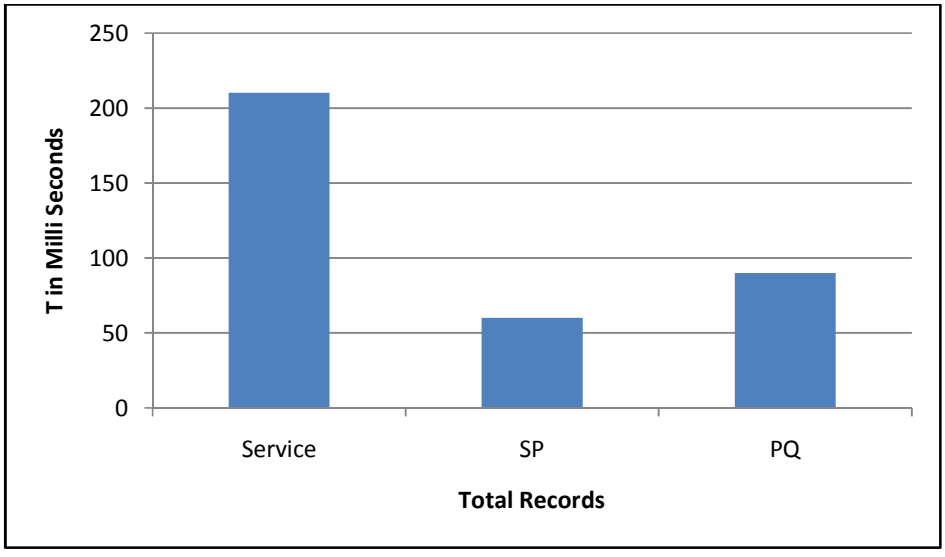


**Figure 5.9: Display All Performance Analysis of the three LIS Architectures**

**Comments on Results**

The total time required by LIS to satisfy user request for any operation since user initiates request using the sample data is:

Total Elapsed Time = Network Time + Processing Time + Display Time

1. Network Time; include:
   - Time to Send Request from User to Web server
   - Time to send DB request from Web server to DB server
   - Time to send reply from DB Server to Web server
   - Time to send Response from Web server to User

Because requests were initiated from the same PC each time, and the database exists on the same application server, network delay is almost equal in all requests; ignoring states of temporarily congestion, so network consumed time is not the main reason for performance differences between the three architectures.

2. Processing Time; include:
   - Web server processing of the sent request pages (include: Parameters processing)
   - Web server preparing the request (incase Services are used)
   - SQL Server processing of the request (insert, update, and retrieve operations)
   - Web server processing of the response

Though generally, processing time is neglected in almost all application evaluation strategies due to the huge revolution in hardware computing and the presence of new highly performance technologies, but performance in the services based architecture was affected. The difference in delays between the three architectures is the result of performance delays. Web services consumption, extra HTTP headers processing, preparing requests and waiting for responses result in the highlighted performance measures differences.

3. Display Time
   - Time to display web page that holds response data on the user screen.

The returned data size is the same for each request applied to the three different architectures. The Utilized analysis tool allowed the unfolding of the response page that holds the processed data from the rest of the portal interface regarding display time, so the Display time is not affected by the architecture implementation at all.

From the performance analysis presented and after evaluation of the three architectures, it is clear that the time consumed to perform the same operations using the services based architecture exceeds the time consumed to perform the same operation using either the

stored procedure architecture, or the parameterized query one. But it should be clear that while other architectures might perform faster than SOA based applications, those architectures can not do what SOA does.

### 3. Functionality

Functionality evaluation focuses on the existence of a set of functions and their specified properties. Functions are those that satisfy stated or implied needs.

### 3.1 Integration and Interoperability

A slight difference between integration and interoperability exists, where integration means the capability of an application to perform its tasks without the need from other application, but still the fact that results of performing those tasks needs to be shared with other applications. On the other hand, interoperability refers to the application need to communicate with other applications to achieve the task.

It is really worth the effort to see that systems can share their effects within a single operation by integrating them on service level, not just on data or application level. Assessment Management System did not have to access Student Affairs Information System database tables to retrieve and update student table data; instead, it just invoked the Update_Student service exposed by it.

Assessment Management System is a real example on the interoperability extent that SOA based applications can achieve. The Take Assessment Process needs interoperability between Assessment Management System and external systems. Without this interoperability, Mobile assessment would not have taken place at all. SOA utilization in the system gave the system capability to expose standard interfaces that act like sockets to be plugged in to connect systems. We did not even needed system architects from the mobile company to manage the application, we consumed the standard interface they exposed and Assessment Management System were able to provide the new Mobile assessment process.

### 3.2 Compliance

Compliance refers to the software adherence to application related standards of conventions or regulations in laws and similar prescriptions. Proposed SOA based system implemented the abstraction and separation of layer concerns. Orchestration and application services layer were separated to achieve the compliance goal. It was clear in the Assessment Management System when the learner had to take an assessment within 24 hours; this simple data was stored explicitly in the database and managed by the Manage Assessment service. When the system needs to change the period of which the

student can take the assessment, changes are not made to the application, neither at application services layer, nor at orchestras.

### 3.3 Security

Security refers to software ability to prevent unauthorized access, whether accidental or deliberate, to programs or data. Classical SOA; by default; utilizes certain security hints that enhance the overall system security, they include:

- The first step for security is to hide the existence of the service from the attackers, and this can happen easily via SOA. Middleware can be placed in the middle between consumer and the service, this middleware guides the request to the service while protecting it from attacks.
- Web services make use of the enhancements performed by Web services manufacturers (Microsoft and IBM for example). Web services 2.0 and Web Services Enhancements (WSE) are examples of by default acquired advantages of implementing Web services overtime.
- Web services act as an isolating layer of database; there is no direct access to the database available, and operations are stated within the service. The service requestor does not even need to have an account on the database.

## 4. Maintainability

Maintainability refers to the ease and speed with which systems can be understood and modified.

### 4.1 Analyzability, Decomposability, and Modularity

Analyzability is the effort needed for diagnosis of deficiencies or causes of failures or for identification of parts to be modified. Decomposability is the process of breaking down a system into its smaller components. These components may themselves be systems (subsystems) and can be broken down into their components as well. This decomposition results in smaller and less complex pieces that are easier to understand than larger, complicated pieces. Decomposing a system allows analyst to focus on one part of the system, making it easier to think of how to modify that one part independent of the entire system Modularity is a direct result of decomposability. It refers to dividing the system into chunks or modules of a relatively uniform size. Modules can represent a system simply, making it easier to understand and easier to redesign and rebuild. Composing systems into stand alone services makes it more analyzable, because it gets decomposable and modular. SOA based applications are easier to determine areas of failure.

### 4.2 Testability

Software testing is the process used to measure the quality of developed software. Usually, quality is constrained to such topics as correctness, completeness, security, but can also include more technical requirements as described under the ISO standard ISO

9126, such as capability, reliability, efficiency, portability, maintainability, compatibility, and usability. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors.

**Levels of Testing**

- **Unit Testing:** tests the minimal software component, or module. Each unit (basic component) of the software is tested to verify that the detailed design for the unit has been correctly implemented. In an Object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

The minimal software component or module is: Service. Each service can be easily tested by itself to assure it performs the required functionality. WSDL provides a standard interface that enables service test even without the need to write a testing application. Applying SOA saves time of testing so, there can be completely new systems build without the need to apply unit testing, because they utilize previously tested services.

- **Functional Testing:** tests at any level (class, module, interface, or system) for proper functionality as defined in the specification.

Functional test needs to be applied to units and system level. By applying functionality test to services via easy by default designed services based unit test, it is clear that the available unit test facilitated the functional testing.

- **Integration Testing:** exposes defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

Because proposed SOA based system utilizes Web services as the main SOA enabler, and because Web services utilizes standard protocols set, integration shortages and limitations were overcome. Integration requirements are not hard to design, implement, and tested incase it is based on Web services based SOA. Integration has turned to be the art of connecting the proper communication points to each other within systems.

**5. Portability**

Portability is the ability of software to be transferred from one environment to another.

### 5.1 Replaceability

It is the opportunity and effort using the software in the place of specified other software in the environment of that software. One of the main advantages of SOA is that it enables replaceability of services utilized from outside incase that the new utilized services maintain the same service interface. It will take just changing the address URI to the new address to utilize the new service.

### 6. Scalability

Scalability is the ease with which a system or component can be modified to fit the problem area. Anarchic Scalability refers to the need for architectural elements to continue operating when they are subjected to an anticipated load, or when given malformed or maliciously constructed data, since they may be communicating with elements outside their organizational control. The architecture must be amenable to mechanisms that enhance scalability. Scalability can be achieved via hardware, or software, or both.

### 6.1 Hardware Scalability

#### Modify Hardware Requirements

Highly rated requested services can be separated on stand alone servers specialized for those services in order to scale the overall system. If this option is not applicable, the servers that hold the highly rated requested services can get hardware upgrades faster than the rest of the system, so the overall system performance can be enhanced.

#### Load Balancing

Scalability can be achieved by Load Balancing Servers. Load Balancing Servers are the servers responsible for balancing loads between multiple servers hold the same required services. Figure 5.10 depicts the classical Web service consumption scenario that is the one without a load balancing server as depicted in figure 5.11. In figure 5.11, the load balancing server is responsible for balancing the requests between the two application servers that hold the same service.

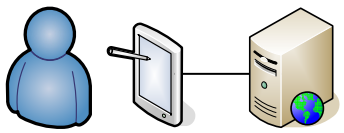

**Figure 5.10: Classical Web service Consumption Scenario**

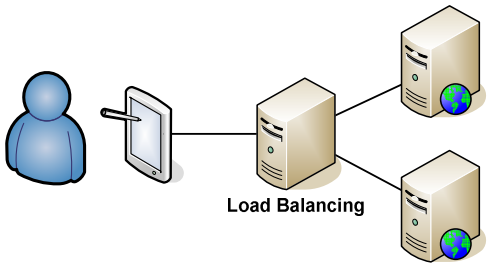


**Figure 5.11: Load Balanced Web service Consumption Scenario**

### 6.2 Software Scalability

- System modularity enables it to be enhanced service by service so the system lifecycle is enhanced on the software scalability level. Services that highly rated requested can be enhanced, and optimized, so the overall system will be enhanced.
- Web services based SOA is immune to malicious software attacks because exchanged messages between requestors and services providers do not hold code to be executed, it only holds XML based data. Messages are not instructive, they are directive.

### 7. Simplicity

The primary means by which architectural styles induce simplicity is by applying the principle of separation of concerns to the allocation of functionality within components.

Separation of concerns into layers is what enables SOA to be simple. Layers enable the system to be understandable and less complex because by checking in which layer the service exists gives a general overview of the type of this service functionality.

### 8. Modifiability

Modifiability is about the ease with which a change can be made to application architecture. A particular concern of network-based systems is dynamic modifiability, where the modification is made to a deployed application without stopping and restarting the entire system. Even if it were possible to build software system that perfectly matches the requirements of its users, those requirements will change over time just as society changes over time.

### 8.1 Extensibility

Extensibility was tested in the proposed architecture by adding functionalities to the system that required adding new services that did not exist in the first place. Because services are loosely coupled, they can be added and consumed without affecting any other services. New added services can utilize existing services, consume databases and

stored procedures, wrap legacy systems, and any functionality required without affecting the existing system.

### 8.2 Reusability

Reusability is a property of application architecture if its components, connectors, or data elements can be reused, without modification, in other applications. It the practice that a segment of source code can be used again to add new functionalities with slight or no modification. IEEE 90 defines it as the degree to which a software module or other work product can be used in more than one computing program or software system. Reusable module and classes reduce implementation time, increase the likelihood that prior testing and use has eliminated bugs and localizes code modification when a change in implementation is required.

Reusability is achieved in the proposed architecture on two levels: Internal and External. Internal reusability refers to the application capability to use the same implemented functionality more than one time without modification. This happened for example with the Update functionality, where it consumed Delete and Insert functions. Besides, functions were not written every time from the start, on the other hand, they were recalled from the implemented service. External Reusability refers to the external systems that consumed the exposed internal services to achieve functionalities.

## 9. Pedagogical Evaluation

Incase architectural and information systems aspects of University Management Systems and Learning Management Systems needed the SOA based system, pedagogical aspects are affected by the proposed SOA based architecture, at least indirectly. Learning Management Systems might think that there is no more functionality to be provided, but SOA proved there is still.

- E-Learning solutions was not affected; from the pedagogical point of view, by adapting SOA. There was no pedagogical requirement that was satisfied by non-SOA Learning Management Systems that was not available when SOA was adopted.
- SOA adoption facilitated integration of software agents within new proposed systems. Software agents have played; and still, major roles in e-Learning. Integrating software agents with Web services was presented successfully in proposed Course Management System and Assessment Management System as an example of the success of integration.
- Mobile assessment refers to the capability of conducting assessments via mobile devices. Mobile assessment relies on external services that are not part of the LMS. Integrating different external systems and services to be virtually part of the educational institution LMS is one of integration challenges. Mobile Learning (M-Learning) is an approach to electronic learning (E-learning) that simply utilizes mobile devices, yet it can also be viewed as a quiet different learning experience (Hulme and John 2005). It is possible to force series of interactive

SMS exchanges between learner and LMS to achieve completion of a task or goal. Learner will take part, and complete the task. M-learning has been used as a pre and/or post activity to other types of learning. M-learning has been widely considered and implemented. Assessment for learning can be thought as one of the post learning activity that can be achieved via mobile phones. Mobile Learning was successfully implemented and the main enabler was SOA adoption.

- Unlocking Course Repositories "Automating the Discovery, Downloading, and Paying of Shared Courses": One of the critical limitations of a newly established educational institution is the lack of available well prepared courses. It is more applicable to use widely available courses that might be higher in quality than preparing new courses. Current course management systems do not exploit courses shareability. To address this shortage, a Course Management System (CMS) is proposed to highlight automated discovering and importing of courses maintained and managed by external CMSs. It is generally a hard job for instructor to prepare electronic course contents. Today's commercial CMSs do not address the capability of automated search and import of external courses, especially if courses are not free. Courses are about quality, not quantity. Departments want fewer modules that incorporate simulations and interactivity. Educational institutions can increase Return-On-Investment (ROI) by selling courses.
- Digital Library contents are available to all LMS components to utilize, search within, and enrich the learning activity with valuable contents without the need to adopt new systems. SOA facilitated the integration between LMS components and Digital Library solution.

## 10. Summary

This chapter presented an evaluation framework that can be used to evaluate LMS. Proposed evaluation framework attempted to evaluate LMS from more than one perspective, so a general overview of the proposed LMS can be achieved. Proposed evaluation framework can be used as a whole, or by topic.

SOA adoption within e-Learning in the form of University Management System and Learning Management System presented information systems' advantages as well as pedagogical ones. It is clear that there is still more to be discovered and more advantages will become available upon adopting SOA in e-Learning.

Pedagogically, SOA has helped e-Learning achieve more than one goal. One of the critical limitations of a newly established educational institution is the lack of available well prepared courses. It is more applicable to use widely available courses that might be higher in quality than preparing new courses. Current LMS do not exploit courses shareability. Proposed SOA based LMS addressed this shortage, automated discovering and importing of courses maintained and managed by external LMSs. Proposed LMS facilitates integration between different LMSs in order to share resources of educational institutions.

SOA facilitated integration between software agents that play an important role in educational institutions and Web services; that is the core of proposed SOA LMS. Also, integrating legacy systems and newly added systems is facilitated by SOA.

Also, Mobile Learning (M-Learning); which is an approach to E-Learning that utilizes mobile devices is enabled by proposed LMS. M-Learning can be adapted via SOA based LMS. Mobile assessment is one of the M-Learning activities facilitated by proposed SOA based architecture. Mobile assessment relies on external services that are not part of the LMS. Integrating different external systems and services to be virtually part of the educational institution LMS is one of integration challenges. The capability to integrate the different digital library contents and make it available to different LMS components is a clear example of the SOA capabilities to integrate different and standalone system components and make them available to each other.

# Chapter 6

## Chapter Six

# CONCLUSION

This thesis presented a proposed University Management Information System (UMIS) and Learning Management System (LMS) components based on Service Oriented Architecture (SOA) that has achieved information system and pedagogical success based on the presented evaluation framework that utilizes information system, pedagogical, and managerial aspects of systems.

SOA as a design pattern addresses non functional requirements like interoperability and integration to achieve systems agility. SOA provides a fine granularity and modularity that solves many integration problems, but adds complexity to systems design. SOA is a design pattern that helped enterprises overcome integration obstacles, and gain agile and interoperable advantages within architectures. SOA utilizes Web services as main enabler to achieve addressed design goals.

Proposed LMS facilitates integration among different LMSs. An automated course search, import, and deposit process was presented. This automated process requires governance of business rules. Business rules might be limiting system efficiency, so they must be monitored and modified when needed. Manage business rules process was presented to achieve this goal. Utilizing SOA to integrate Web services and software agents in LMSs highlighted the unlimited advantages of Web services and its capabilities to facilitate software agents' integration within systems. LMS should be thought of as a collection of stateless Web services. Pedagogical, social, and managerial advantages of added processes include:

- Overcome lack of internal courses
- Get use of external, higher pedagogical features courses
- Shareability among different educational institutions
- Competition increment adds to quality (indirect effect)
- Increase Return-On-Investment (ROI) by selling courses

Assessment is an integral part of the learning process, and a learning activity that can be achieved efficiently via mobile devices. Due to differences in mobile architectures,

and as a result of lack of interoperability in current commercial LMS with external systems, mobile assessment may find difficulties in implementation. Interoperability between LMS and mobile mediator is achieved via adoption of Web services based SOA in LMS. Pedagogically, mobile assessment is important due to the assessment importance process to the learning process. Mobile assessment can facilitate and encourage students to attend assessments and enable distance education by expanding interactivity tools available to students to include mobile devices.

When more than software architecture satisfies information systems functional and nonfunctional requirements, performance is a key element of choosing the software architecture to be implemented. From the comparative analysis study of a three different Library Management System architectures: Parameterized Query, Stored Procedures, and Services based architectures, it was clear that neither network nor user-perceived performance were affected to the extent that limits the utilization and implementation of proposed SOA system, especially when compared to the non-functional requirements gained by adopting SOA, like integration and interoperability, compliance, security, maintainability, analyzability, decomposability and modularity, testability, portability via replaceability and scalability, simplicity, modifiability, and reusability.

Future work includes addressing more and widely system processes that rely on the presented SOA based LMS and the implemented Web services to address agility in education institutions. Business Process Management Systems that rely on proposed SOA based LMS and UMIS is the next step to take.

# Bibliographies

1. **A. M. Riad** “eUniversity: An Intelligent System for E-Learning”, Egyptian Informatics Journal, Faculty of Computers and Information, Vol. 6, No 2, Cairo Univ., December 2005.
2. **H. El-Hadidi, A. M. Riad, and M. Matsumoto,**" A Multi-agent System to Enhance Information Retreival Results", IEIC 2007, Japan, 10-14 September 2007, BS-10-16.
3. **V. W. Chairman**, “A Glossary of e-Learning terms and acronyms”, eLearning Network, http://www.elearningnetwork.org/articles/article9.doc
4. **L. T. Victoria**, “ICT in Education”, Asia-Pacific Development Information Programme, http://www.apdip.net
5. **W. Horton, and C. Horton**, “E-Learning Tools and Technologies”, Wiley Publishing Inc., 2003
6. **M. G. Moore, and W. G. Anderson**, “Handbook of Distance Education”, Lawrence Erlbaum Associates, 2003.
7. **T. M. Duffy, and R. J. Kirkley,** ”Learner-Centered Theory and Practice in Distance Education”, Lawrence Erlbaum Associates, 2004.
8. **Z. Ma**, “Web-Based Intelligent E-Learning: Technologies and Applications”, Information Science Publishing, 2006.
9. The Commonwealth of Learning, “An Introduction to Open and Distance Learning”, http://www.col.org/ODLIntro/introODL.htm
10. **E. Kaplan-Leiserson**, “ASTD's Source for E-Learning. Learning Circuits”, http://www.learningcircuits.org/glossary
11. Blended learning, Wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/Blended_learning
12. **A. Bork**, “The Future of Learning: An Interview with Alfred Bork”, Review, Educom, 1999, Vol. 34.
13. MIS, Webopedia, http://www.webopedia.com/TERM/M/MIS.html
14. **L. E. Long, and N. Long**, ”Introduction to Computers and Information Systems: The Internet Edition”, Prentice Hall, 1996.
15. **A. M. Riad, and H. A. El-Ghareeb,** “A Service Oriented Architecture to Integrate Web services and Software Agents in Course Management Systems”, Egyptian Informatic Journal, Cairo Unviersity, 2007, Vol. 8, Issue 1.
16. **S. Britain, and O. Liber**, ”A Framework for the Pedagogical Evaluation of Virtual Learning Environments”, JISC Technology Applications Programme, 2008.
17. **P. Noerr**, “99 Questions about Digital Libraries”, VALA - Libraries, Technology and the Future Inc., http://www.vala.org.au/vala2000/2000pdf/Noerr.pdf
18. Supporting Education and Research - A higher education committee, Joint Information Systems Committee (JISC), Great Britain Higher Education, http://www.jisc.ac.uk
19. History of virtual learning environments, Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/History_of_virtual_learning_environments

20. What is Distance Learning? For Adult Educators - California Distance Learning Project (CDLP), http://www.cdlponline.org/index.cfm?fuseaction=whatis&pg=3
21. **B. L. Hardy, and K. Bower**, "From correspondence to cyberspace: Changes and challenges in distance education", New Directions for Community Colleges, New Directions for Community Colleges, 2004, pp. 5 - 12.
22. 50 Years of HoustonPBS History, Channel 8, HoustonPBS, http://www.houstonpbs.org/site/PageServer?pagename=abt_history
23. **A. S. Patel, K. Bernard**, "Intelligent Tutoring: from SAKI to Byzantium", MCB UP Ltd, 2001.
24. **D. Bitzer, E. Lyman, and J. Easley,** "The Uses of PLATO: A Computer-Controlled Teaching System", Coordinated Science Laboratory, University of Illinois, 1965.
25. **K. A. Hall**, "Computer-Assisted Instruction: Status in Pennsylvania", Pennsylvania State University, 1970.
26. **J. V. Hebenstreit**, "Computer-Assisted Instruction in France: present situation and prospects for the future", Ottawa: National Research Council Canada, Proceedings of the Third Canadian Symposium on Instructional Technology, 1980, pp. 77-91.
27. **S. Hunka, and G. Buck**, "The Rise and Fall of CAI at the University of Alberta's Faculty of Education", Canadian Journal of Educational Communication, 1996, Vol. 21, pp. 153-170.
28. **W. Rappaport, and E. Olenbush**, "Tailor-Made Teaching through TICCIT", Mitre Matrix, 1975.
29. **W. Broderick, J. Brahan, and R. Shevel**, "An instructional management system for NATAL-74", National Research Council Canada, Third Canadian Symposium on Instructional Technology, 1980.
30. **M. Yost**, "Extracting Data From Integrated Student Information Systems", Eric, 1983.
31. Computer Based Training, Wkikipedia, http://en.wikipedia.org/wiki/Computer-based_training
32. Learning Management System, Wikipedia, http://en.wikipedia.org/wiki/Learning_management_system
33. **A. Feenberg**, "Distance Learning: Promise or Threat?", 1999, http://www-rohan.sdsu.edu/faculty/feenberg/TELE3.HTM
34. **H. Lowood**, "Supplement: Current Bibliography in the History of Technology", JSTOR, Technology and Culture, 1990, Vol. 33, pp. 1-138.
35. **M. Benjemann, and E. Cresson**, "Educational Software and Multimedia", Task Force, European Commission, 1996.
36. Technologies for Online Interoperable Assessment (TOIA), http://www.toia.ac.uk/toia-ams.html
37. **L. Butler**, "Curriculum Development System for Navy Technical Training", Eric, 1990.

38. Highlight History of Extension in Wisconsin 1862 to 1999, The University of Wisconsin-Extension, http://www.uwex.edu/about/history/
39. **J. M. Gundry, "**Web 2.0 Social Media", Knowledge Ability Ltd, 2006, http://www.knowab.co.uk/socialmedia
40. **S. Britain, and O. Liber**, "A Framework for Pedagogical Evaluation of Virtual Learning Environments", JTAP - JISC Technology Application Programme, University of Wales - Bangor, 1999, http://www.leeds.ac.uk/educol/documents/00001237.htm
41. **T. Anderson, and R. Mason**, "The Bangkok Project: New tool for Professional Development", American Journal of Distance Education, 1993, Vol. 7, pp. 5 - 18.
42. Global Virtual Classroom, http://www.virtualclassroom.org
43. e-Learning from WBT Systems, http://www.wbtsystems.com
44. **D. Holden**, "Restructuring Schools on a Service-Industry Model", T.H.E. Journal, March 1994, Vol. 21, pp. 70 - 71.
45. History of personal learning environments, Wikipedia, http://en.wikipedia.org/wiki/Personal_Learning_Environment
46. Michigan State University's Virtual University, http://vu.msu.edu/site/
47. **A. Jafari**, "Putting Everyone and Every Course Online: The Oncourse Environment", WebNet Journal, 1999, Vol. 37.
48. The Free Dictionary, Farlex, http://www.thefreedictionary.com/conference
49. Data Conferencing, PCmag.com Encyclopedia, http://www.pcmag.com/encyclopedia_term/0,2542,t=data+conferencing&i=40760,00.asp
50. Audio Conferencing, PCMag Encyclopedia, http://www.pcmag.com/encyclopedia_term/0,2542,t=audio+conferencing&i=38177,00.asp
51. Video Conferencing. PCMag Encyclopedia, http://www.pcmag.com/encyclopedia_term/0,2542,t=videoconferencing&i=53874,00.asp
52. Blackboard Virtual Classroom Features, Fayeiteville State University, http://www.uncfsu.edu/conted/facultysupport/PDF/Bb6-Chat-Features-Comparison.pdf
53. P. Noerr, Digital Library Toolkit, Sun Education and Research, 2003, http://www.sun.com/products-n-solutions/edu/whitepapers/digitaltoolkit.html
54. **D. Whitelock, and S. Wheeler**, "ALT-C 2006: The next generation", Edinburgh : Heriot-Watt University, The next generation: 13th International Conference, 2006.
55. .LRN Home: Learn, Search, Network, http://www.dotlrn.org
56. Blackboard: Educate, Innovate Everywhere, http://www.blackboard.com/us/index.aspx
57. Centra-Saba, http://www.saba.com/centra-saba
58. COSE: Creation Of Study Environments, http://www.staffs.ac.uk/COSE/
59. LON-CAPA, http://www.lon-capa.org/
60. moodle. A Free, Open Source Course Management System for Online Learning, http://www.moodle.org/
61. The Learning Manager, http://www.thelearningmanager.com/

62. Angel Learning, http://www.angellearning.com/
63. ATutor, http://www.atutor.ca/
64. Claroline, http://www.claroline.net/index.php
65. Desire2Learn: Innovative Learning Technology, http://www.desire2learn.com/
66. Eledge: Open Learning Management System, The University of Utah http://eledge.sourceforge.net/
67. IntraLearn: Learning Server, http://www.intralearn.com/
68. e-Learning at the University of Western Cape (UWC), http://kewl.uwc.ac.za
69. Avilar LMS, http://www.avilar.com/products/lms.html
70. Janison, http://www.janison.com.au/janison/default.asp
71. KnowEdge: eLearning Suite, http://www.knowedge.net/
72. Unicon: Open Source Commercial Support, http://www.unicon.net/
73. BSCW: Basic Support for Cooperatvie Work, http://bscw.fit.fraunhofer.de/
74. Colloquia, http://www.colloquia.net/
75. eCollege, http://www.ecollege.com
76. ILIAS, http://www.ilias.de/ios/index-e.html
77. Internet Campus Solutions, Jenzabar, http://www.jenzabar.com/
78. Collaborative Learning, Dicole, http://freshmeat.net/projects/mimerdesk/
79. SAKAI: Collaboration and Learning Environment For Education, http://sakaiproject.org/
80. IBM Lotus Software, http://www-306.ibm.com/software/lotus/
81. **N. Ebel**, "Lotus Notes / Domino Administration: Lotus Groupware", Pearson Education, 2004.
82. **C. Fallon, and S. Brown**, "E-Learning Standards: A Guide to Purchasing, Developing, and Deploying Standards-conformant e-Learning", CRC Press, 2003.
83. **M. Sicilia**, "Competencies in Organizational E-learning: Concepts and Tools", Idea Group Inc (IGI), 2007.
84. **T. Hall**, "A preview of Lotus Learning Management System", IBM, 2003.
85. **E. Bowling**, "Understanding the architecture of the Lotus Learning Management System", IBM, 2003, http://www-128.ibm.com/developerworks/lotus/library/ls-LMS_architecture/?ca=drs-11/10/2006
86. **Y. K. Singh**, "Fundamentals of Research Methodology and Statistics", New Age Publishers, 2006.
87. **W. Trochim**, "Introduction to Evaluation", 2006, http://www.evaluation.lars-balzer.name/links/online-texts-what-is-evaluation/
88. **W.J. Schurink**, "Victimization: Nature & Trends", Human Sciences Research Council, 1992.
89. **L. Rojas**, and **F. Serpa**, "Introduction to Program Evaluation", Professional Research and Evaluation Group.

90. Evaluation, The Joint Committee on Standards for Educational, Thousand Oaks, CA: Sage Publications, Inc., 1994, The Program Evaluation Standards.
91. Evaluating Financial Management Software, Microsoft, 2007 http://www.findwhitepapers.com/index.php?option=com_categoryreport&task=viewabstract&title=956&id=12&vid=221&cat=111&pathway=no.
92. **R. T. Fielding**, "Architectural Styles and the Design of Network-based Software Architectures", Irvine: University of California, 2001.
93. ISO/IEC JTC1 SC36 N0646, ISO/IEC SC36: Information Technology for Learning, Education, and Training, 2003, http://jtc1sc36.org/doc/36N0646.pdf. ISO/IEC JTC1 SC36 WG4
94. **J. Liberty**, "Programming C#", O'Reilly, 2005.
95. **I. Sommerville**,"Software Engineering", Addison-Wesley, 2004.
96. **S. Güner**, "Architectural Approaches, Concepts and Methodologies of Service Oriented Architecture", Software System Institute, TUHH: Technical University Hamburg Harburg, 2005.
97. **R. Malveau, and T. Mowbray**, "Software Architecture: Basic Training", Prentice Hall, 2004.
98. CBDI Service Oriented Architecture Practice Portal, http://www.cbdiforum.com/
99. **T. Erl**, "Service Oriented Architecture: A Field Guide to Integrating XML and Web services", Prentice Hall, 2004.
100. **A. Macaulay**, "Enterprise Architecture Design and the Integrated Architecture Framework", Microsoft Architects Journal, 2004, Vol. 1, pp. 4-9.
101. **D. Sprott, and L. Wilkes**,"Understanding Service Oriented Architecture", Microsoft Architect Journal, 2004, Vol. 1, pp. 10-17.
102. **L. Wilkes, and R. Veryard**, "Service Oriented Architecture - Considerations for Agile Systems", Microsoft Architect Journal, 2004, Vol. 2, pp. 11-23.
103. **M. Endrei**, "Patterns: Service Oriented Architecture and Web Services", IBM RedBooks, 2004.
104. **M. Wooldridge**, "Introduction to MultiAgent Systems", Wiley, 2002.
105. **A. M. Riad, A. and H. A. El-Ghareeb**, "New Architecture for Mobile News Agent System", Egyptian Informatics Journal, Cairo University, 2007, Vol. 8, Issue 1.
106. **T. I. Zhang, and H. B. Jiang**, "A Framework of Incorporating Software Agents into SOA", Spain : IASTED, Artificial Intelligence and Soft Computing, 2005.
107. **R. G. Qui**, "Enterprise Service Computing: From Concept to Deployment", IGI Global, 2007.
108. Software Building Blocks - Mobile Agent Based Service Oriented Architecture (ASOA), Mobile Agent Technologies, 2005. http://www.agentos.net/Software%20Building%20Blocks.pdf

109. **I. Zinnikus**, "A Model Driven Approach to Agent-Based Service-Oriented-Architectures", ATHENA - European Integrated Project, http://www.athena-ip.org/dmdocuments/pu/ZinnikusEtAl.pdf
110. **J. Poozhikunnel**, "Developing a Simple Service Oriented Architecture", internet.com, http://15seconds.com/issue/050119.htm
111. **A. Caglayan**, and **C. Harrison**, "Agent Sourcebook: A Complete Guide to Desktop, Internet, and Intranet Agents", Wiley, 1997.
112. **F. Cheong**, "Internet Agents: Spiders, Wanderers, Brokers, and Bots", New Riders Publishing, 1996.
113. **S. J. Russell**, and **P. Norvig**, "Artificial Intelligence: Modern Approach", Prentice Hall, 2003.
114. **E. Turban**, **J. E. Aronson,** and **T. Liang**, "Decision Support Systems and Intelligent Systems", Prentice Hall, 2004.
115. **K. Syeara**, "Multi-Agent Systems and Applications", Springer, 2001.
116. **M. d'Inverno, and M. Luck**, "Understanding Agent Systems", Springer, 2003.
117. **D. Moldt, and J. Ortmann,** "A Conceptual and Practical Framework for Web-Based Processes in Multi-Agent Systems", Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004, Volume 3 (AAMAS'04).
118. **H. C. LAU, L. ZHANG, and C. LIU**, "Solving Generalized Open Constraint Optimization Problem Using Two-level Multi-agent Framework", IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2005.
119. **J. Mostafa, W. Ke, and Y. Fu**, "Automated text classification using a multi-agent framework", Fifth ACM/IEEE-CS Joint conference on Digital libraries (JCDL '05), 2005.
120. **S. Roy, S. Dasgupta,** and **N. Mukherjee,** "A Multi-agent Framework for Resource Brokering of Multiple Concurrent Jobs in Grid Environment", Proceedings of The Fifth International Symposium on Parallel and Distributed Computing (ISPDC'06), 2006.
121. **H. EL Yamany**, **M. Capretz**, **and L. Capretz**, "A Multi-Agent Framework for Testing Distributed Systems", 30th Annual International Computer Software and Applications Conference (COMPSAC'06), 2006.
122. **W. Chen, and S. Tseng**, "A Novel Multi-agent Framework for the Design of Home Automation", International Conference on Information Technology (ITNG'07), 2007.
123. **R. A. Flores-Mendez**, "Towards a standardization of multi-agent system framework", ACM Press, Summer 1999, Crossroads, Vol. 5, pp. 18 - 24.
124. **P. Bergen**, Garfixia Software Architectures, http://www.dossier-andreas.net/software_architecture/
125. **T. Finin**, "KQML as an Agent Communication Language", 3rd International Conference on Information and Knowledge Management (CIKM'94), 1994.
126. **M. R. Genesereth**, "draft proposed American National Standard (dpANS)", Knowledge Interchange Format, NCITS.T2/98-004, http://logic.stanford.edu/kif/dpans.html

127. **S. Virdhagriswaran, D. Osisek, and O'Connor**, “Standardizing agent technology”, ACM Press, 1995, Vol. 3.
128. **J. M. Bradshaw**, “KAoS: An Open Agent Architecture Supporting Reuse, Interoperability, and Extensibility”, http://ksi.cpsc.ucalgary.ca/KAW/KAW96/bradshaw/KAW.html
129. FIPA: Foundation for Intelligent Physical Agents, http://www.FIPA.org
130. **X. Bai**, “A Multi-Agent Based Framework for Collaborative Testing on Web Services”, Fourth IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems and Second International Workshop on Collaborative Computing, Integration, and Assurance (SEUS-WCCIA’06), 2006.
131. **X. Zhou**, “Frontiers of WWW Research and Development -- APWeb 2006”, Springer, 2006.
132. **M. N. Huhns,** “Agents as Web services”, IEEE Internet Computing, 2002.
133. **P. Shodjai**, ”Web services and the Microsoft Platform”, Microsoft Software Developers Network (MSDN), 2006.
134. **D. Sprott**, “A Web service Maturity Model”, CBDI, http://roadmap.cbdiforum.com/reports/maturity/maturity2.php
135. **D. Booth, “**Web Services Architecture”, W3C, 2004.
136. **E. Newcomer, and G. Lomow,** “Understanding SOA with Web Services”, Addison Wesley Professional, 2004.
137. **P. Hégaret,** “Introduction to Web Services”, W3C, 2003.
138. **T. Erl**, ”Service Oriented Architecture: Concepts, Technology, and Design”, Prentice Hall, 2005.
139. **D. K. Barry**, “Web Services and Service-Oriented Architecture: The Savvy Manager's Guide”, Morgan Kaufmann Publishers, 2003.
140. **Q. Wall**, “Understanding the Service Lifecycle within a SOA: Design Time”, BEA WebLogic, 2006.

# ملخص

التعليم الالكتروني هو احد اهم المواضيع التي برزت في السنوات الاخيرة و حققت الكثير من النجاح واكتسبت شعبية في المؤسسات التعليمية. نظم ادارة التعليم الالكتروني ليست كسائر النظم العادية لانها تتطلب التكامل بين العديد من التكنولوجيات التي تعين نظم ادارة التعليم الالكتروني على التكيف و تمنحها فعالية اكثر من غيرها من النظم. ظاهريا قد يشيرالتعليم الالكتروني الى استخدام تكنولوجيا المعلومات و الاتصالات في عملية التعليم ، لكن بالطبع فان التعليم الالكتروني لهو اعمق من ذلك بكثير .التعليم الالكتروني يشمل المنظومة المتكاملة التي تتيح للفرد حرية أكثر في التعلم و تتيح للمجتمع امكانية احداث ثورة في مجال التعليم و لتحقيق هذه الثورة كان لزاما على المؤسسات التعليمية ان تدرك أهمية التكامل بين نظم المعلومات الاداريه للجامعات و نظم إدارة التعليم. نظم المعلومات الادارية للجامعات هي مجموعة نظم المعلومات التى تم تنفيذها على نطاق واسع ، واستخدمت لفترة طويلة لاشباع الجوانب الاداريه للمؤسسات التعليمية وتشتمل كمثال لمكوناتها على نظام إدارة شؤون الطلاب ، ونظام ادارة المكتبات. على صعيد آخر، فان نظم إدارة التعليم هي مجموعة من التطبيقات البرمجيه المسؤوله عن ادارة وميكنة الأنشطة المتعلقة بعملية التعليم مثل: نظام ادارة المواد التعليمية ، و نظام ادارة التقييم ، والمكتبةالرقمية. يحتاج التعليم الالكتروني الى تكامل نظم المعلومات الاداريه للجامعات و نظم إدارة التعليم.

ان تكامل الانظمة المختلفة مع بعضها هو احد اكبر التحديات التي تواجه مهندسي الأنظمة حاليا ولنجاح هذا التكامل لا بد من الاهتمام بالهيكل البنائي للنظام و الذي هو احد اهم عوامل نجاحه ولذلك فان اختيار الهيكل البنائي للنظام يعد احد اهم الخطوات لبنائه. قدرة النظام على التكامل مع الانظمة الاخرى هي أحد أهم المتطلبات غير الوظيفية للنظام و التي ينبغي الوفاء بها لتحقيق مستوى أعلى في التعليم الالكتروني.

بالرغم من وجود العديد من الهياكل البنائية التي لديها القدرة على تلبية متطلبات النظام الوظيفية الا انه ما زالت تلبية المتطلبات غير الوظيفية للنظام مهمة معقدة. الهيكل خدمي التوجه هو احد الانظمة التي برزت حديثا محاولة اثبات نجاحها في تلبية متطلبات النظام الوظيفية و غير الوظيفية خاصة عند استخدام خدمات الويب كمعول اساسي لها. بالرغم من محاولات استخدام الوكلاء كمعول لهيكل البناء خدمي التوجه الا ان انظمة مثل نظام وكيل الاخبار المتنقل والذي هو عبارة عن وكيل المعلومات المتنقل أثبت أن الوكيل المتنقل تكنولوجيا لا يمكن ان يستخدم فى جميع نظم استرجاع المعلومات ، لانه في ظل بعض الظروف ، قد يفقد بعض مزاياه.

قدمت هذه الرسالة هيكل النظام المقترح والذي يتكون من مستويين: مستوى العرض و مستوى الخدمة. مستوى العرض هو المسؤول عن التفاعل مع المستخدم اما عن طريق عرض

المعلومات أو اتاحة الامكانية لادخال المعطيات ، وهو المسؤول في ذات الوقت عن اتاحة طريقة لتفاعل النظام مع النظم الخارجية الأخرى. مستوى الخدمة هو المستوى الذي يحتوى جوهر الخدمات المقدمة من النظام و ينقسم الى ثلاث مستويات فرعية: مستوى التنظيم، مستوى خدمات التطبيقات، و مستوى الوكلاء. مستوى التنظيم هو المسؤول عن تقديم القواعد المنطقية لتنظيم سير عمليات النظام. مستوى خدمات التطبيقات يحتوى مجموعة من الخدمات التي لها القدرة على القيام بمهام محددة. مستوى الوكلاء يقدم مجموعة مقترحة من الوكلاء لدعم النظام.

يجب تقييم الاستفادة من الهيكل المقترح في التعليم الالكتروني لتحديد مدى فعاليته. تم اقتراح هيكلية لتقييم قدرة الهيكل المقترح على اشباع المتطلبات الوظيفية وغير الوظيفيه للتعليم الالكتروني. الهيكل المقترح للتقييم في هذا البحث يشمل ثلاث مستويات: تقييم نظام المعلومات ، الجوانب التعليمية ، والجوانب الاداريه.

من وجهة نظر نظم المعلومات ، فان الهيكل خدمي التوجه الذي تم تقديمه في البحث استطاع تحقيق العديد، ان لم يكن كل، المتطلبات غير الوظيفية للنظام و التي لا توفرها أي هياكل بنائية أخرى. من أهم المتطلبات غير الوظيفية التي أشبعها الهيكل المقترح: التبادلية والتكامل ، الامتثال ، الأمن ، الصيانة ، القابلية للتحليل والنمطية، قابليه الاختبار، سهولة الاستبدال، البساطة، القدرة على التعديل.

على المستوى التعليمي ، ساعد الهيكل المقترح في هذا البحث التعليم الالكتروني لتحقيق العديد من الأهداف. نقص المواد التعليمية جيدة التحضير هي واحدة من التحديات الحرجة للمؤسسات التعليميه حديثة العهد. انه لمن الملائم استخدام المواد التعليميه المنتشرة بوفرة و التي قد تكون على مستوى من الجوده أعلى من المواد التعليميه التي سيتم تحضيرها. نظم ادارة المواد التعليمية الحالية لا تقدم أنسب الحلول في مشاركة المواد التعليمية. نظرا لهذا القصور، قدم البحث نظاما لادارة المواد التعليميه لابراز ميكنة عملية استكشاف المواد التعليمية داخليا و خارجيا و عملية ادراجها الى داخل النظام لاتاحة الامكانية لتكامل نظم مختلفه لادارة المواد التعليمية.

اضافة الى ذلك، فان الهيكل المقدم أدي الى تسهيل التكامل بين الوكلاء التى تلعب دورا هاما فى المؤسسات التعليميه وخدمات الويب والتي هي جوهر الهيكل المقدم، اضافة الى القدرة على دمج النظم القديمة و مكونات النظم الجديدة المقترحة. ليس ذلك فحسب، بل ان الهيكل المقترح منح القدرة لاستخدام اجهزة الجوال و دمجها في عملية التعليم وقد برز ذلك في البحث من خلال اتاحة القدرة لإجراء عملية التقييم و التى هى واحدة من العمليات الاساسية في التعليم عبر أجهزة الجوال. القدرة على اتاحة المحتوى الرقمى للمكتبات الرقمية عن طريق توفير خدمات البحث و امكانية عرض النتائج لمختلف تطبيقات التعليم الالكتروني كانت أحد أبرز أمثلة التكامل بين النظم والتي أتاحها الهيكل المقترح في البحث.

**جامعة المنصورة**
**كلية الحاسبات و المعلومات**
**قسم نظم المعلومات**

# تقييم أسلوب الخدمة الموجة في التعليم الالكتروني

**رسالة**
مقدمة إلي قسم نظم المعلومات – كلية الحاسبات والمعلومات – جامعة المنصورة
للحصول علي درجة الماجيستير فى نظم المعلومات

اسم الباحث
## هيثم عبدالمنعم الغريب
المعيد بقسم نظم المعلومات – كلية الحاسبات و المعلومات
جامعة المنصورة

تحت إشراف

## أ.د. علاء الدين محمد رياض
رئيس قسم نظم المعلومات
كلية الحاسبات والمعلومات – جامعة المنصورة

## د. أحمد السيد حسن
مدرس بقسم الهندسة الكهربية
كلية الهندسة – جامعة المنصورة

المنصورة - ٢٠٠٨