Design and Implementation of the IITG Enterprise Software System using a Service Oriented Architecture

Dissertation

Submitted in partial fulfilment of the requirements
of the degree of

Masters of Technology

by

Ferdous Ahmed Barbhuiya
(Roll No. 04410108)

Under the guidance of

Prof. Gautam Barua

Department of Computer Science and Engineering

Indian Institute of Technology Guwahati

2006-2007

Certificate

This is to certify that the work contained in the thesis entitled **“Design and Implementation of the IITG Enterprise Software System using a Service Oriented Architecture”** by Ferdous Ahmed Barbhuiya, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Prof. Gautam Barua
Professor
Department of Computer Science and Engineering,
Indian Institute of Technology Guwahati.

Acknowledgements

I would like to express my deep and sincere gratitude to my supervisors, Professor Gautam Barua, Professor, of Computer Science and Engineering, IIT Guwahati, for his valuable guidance. His wide knowledge and logical way of thinking have been of great value for me. His understanding, encouragement and personal guidance have provided a good basis for the present thesis.

This is a project based on collaborative effort. A number of students took part in the project implementing different components. The following students are as much parts of the project as I or my supervisor are. Without their contributions, very little would have been achieved. In Table 1 the details of different contributors of this project are given.

I am thankful to all the faculty members and staff of Computer Science and Engineering Department and Computer Centre who were very co-operative during my work. My sincere thanks goes to many staff and faculty members in different sections and departments who helped me in understanding the problem, shown interest in my work, took time to discuss and suggest me at different point of time out of their busy schedule.

I would like to thank Swapnil, Kamakhya, Pradnyesh and all my classmates for their help and support. Also I acknowledge the help and suggestions of everyone who has directly or indirectly helped me in carrying out the project

Finally, and above all, my heartfelt thanks to my beloved parents and brothers for their loving support.

Sl No.	Name	Affiliation	Component Contribution
1.	Abhishek Gupta	IITG BTech 04	Academic
2.	Sriharsha Gangam	IITG BTech 04	Academic
3.	Makbul Siram	IITG BTech 03	Administration
4.	Marella Aditya	IITG BTech 03	Administration
5.	Ashish Patro	IITG BTech 05	Administration
6.	Saswat Mohanty	NITS BTech 03	Research and Develop.
7.	Munish Nahin	NITS BTech 03	Research and Develop.
8.	Upanita Goswami	NITS BTech 03	Research and Develop.
9.	Gopajit Malakar	NITS BTech 03	Research and Develop.
10.	Sameer Aggarwal	IITG BTech 05	Student Affairs
11.	Kishlay	IITG BTech 05	Student Affairs
12.	Atif Jamil	IITG BTech 05	Placement
13.	Rohan Agarwal	IITG BTech 05	Placement
14.	Prodduturu Mounica	IITG BTech 05	Store and Purchase
15.	Monabili Basumatary	IITG BTech 05	Store and Purchase
16.	Naveen Cherukuri	IITG BTech 05	Finance
17.	Shikhar Sachan	IITG BTech 05	Hostel Affairs
18.	Alekhya Telekicherla	IITG BTech 05	Hostel Affairs
19.	Tati Raghuveer	IITG BTech 05	Engineering Cell
20.	Abhishek Anand	IITG BTech 06	Academic

Table 1: Details of diffrent contributors

Ferdous Ahmed Barbhuiya,
July 29, 2007
IIT Guwahati

Contents

List of Figures

List of Tables

Abstract

The project consists of the design and implementation of the IIT Guwahati Enterprise software system on a Service Oriented Architecture (SOA). Service is an entity that provides information or operational capability to its clients by exchanging messages. Service Oriented Architecture provides a system for design, development and management of a loosely coupled business application. The whole system is divided into smaller segments developed independent of each other. All dependencies, of one segment over the other, are catered to using web services. The reason why this particular architecture is chosen is that the complexity of application definition, development and maintenance is greatly reduced without losing the efficiency of the system. Also, with one application independent of other, it is possible to use a different technology and a different platform for different segments (with the constraint that they have to publish a set of web services, serving the needs of other segments). Web services are gaining more and more importance as a technology to develop distributed service-oriented applications. The different components of the system in an academic institute like IIT Guwahati are sufficiently loosely coupled for SOA to be used efficiently. With a large number of components, the man-hour required to maintain the systems will be low and the SOA will add flexibility to systems. allowing easy enhancement and upgradation. The current system can become the basis for systems of other educational institutions. The loosely coupled architecture, along with laid down programming practices provides a distributed framework for customising the system to suit the needs of an institution.

Chapter 1

Introduction

1.1 Introduction to office Automation

Office Automation is the attempt to use new technology to improve a working environment. It refers to the varied computer machinery and software used to digitally create, collect, store, manipulate, and relay office information needed for accomplishing basic tasks and goals. Raw data storage, electronic transfer, and the management of electronic business information comprise the basic activities of an office automation system. Office Automation helps to optimise or automate existing office procedures. The advent of the personal computer revolutionized office automation, and today, popular operating systems and user interfaces dominate office computer systems.

Generally, there are three basic activities of an office automation system: data storage of information, data exchange, and data management. Data storage usually includes office records and other primary office forms and documents. While data storage and manipulation is one component of an office automation system, the exchange of that information is another equally important component. Electronic transfer is a general application area that highlights the exchange of information between more than one user or participants. Office automation systems are also often used to track both short-term and long-term data in the realms of financial plans, workforce allocation plans, marketing expenditures, inventory purchases, and other aspects of business.

1.2 Introduction to IIT Guwahati

The Institute has eleven departments, and six centres covering all the major engineering and science disciplines, offering B. Tech., B. Des., M. Tech., Ph.D. and M.Sc. programmes. Presently there are about 2,200 students on rolls, more than 180 faculty members and about 260 support staff. Besides the departments and centres, the following are the major sections in the Institute administrative set-up: administration, academic affairs, students' affairs (includes the gymkhana, sports facilities, hostels), finance and accounts, stores and purchase, research and development, library, medical section, and the Institute Engineering Cell (handling construction and maintenance activities).

1.3 History of IIT Guwahati Office Automation

IIT Guwahati started planning for a software system in 2004. The first option was to use a system already in place in another IIT as the IITs have similar administrative set-ups. Systems being used at IIT Kanpur, IIT Delhi and at IIT Bombay were examined in brief. It was soon realised that none of the solutions were suitable. The primary reason was that technology changes so much in this field that any implementation more than three years old is dated and faces problems of support and inadequate use of current technologies. All the systems examined were implemented before 2000 and so were not found suitable.

The next option was to consider packaged, commercial off-the-shelf (COTS) software. With increasing interest in using software in the administrative process of educational institutions, a number of vendors have come up with such solutions. Solutions from global vendors were found to be inadequately configured to meet the needs of an Indian Institution. Solutions from CMC, TCS, and HCL were also examined. But none had the required level of flexibility built in that would allow the system to be changed in the future to meet new needs. Most of the solutions concentrated on grade management and financial management, Requirements of a residential campus were not properly addressed.

The third option examined was to get a bespoke implementation done by a reputed software vendor with the source being made available so that the Institute would own the software developed. This solution has the advantage of building a solution tailored to the need of the Institute, Further; it will be built by professionals from a leading IT vendor. Lastly, the software would be owned by the Institute so that it could future changes and enhancements. The Institute went ahead preparing a document that laid down the specifications. By the time the first version of the document was ready, work had gone into examining this strategy in more detail. It was realised that none of the leading vendors had requisite domain knowledge in the application area as they had not taken up any work in this area for any client in the recent past. So, unlike an application in say banking, insurance, or manufacturing, vendors could not offer the advantage of domain and professional experience to this application. In fact, many vendors wanted to enter the segment and wanted to use a contract with IIT Guwahati to gain experience. Further, the cost of such an implementation was found to be very high. With the first draft of the requirements ready, and with a pool of bright B.Tech students eager to take up summer assignments, it was decided to go in for an in-house implementation.

Chapter 2

Survey of Software Architectures in office Automation

Office automation is the planning, development and use of information technology tools to help people perform all tasks related to the functioning of an office. They are built to serve the needs of large organizations in terms of efficient data storage, retrieval, query, decision support and presentation. Typically they are large software systems and are part of the family of Information systems. A lot of research has gone into developing architectures and procedures for efficient information systems. Researchers have suggested different architectures like centralized, decentralized or hybrid architectures and their respective uses.[3][4][5][6]

Software architectures have attempted to deal with increasing levels of software complexity. But the level of complexity continues to increase, and traditional architectures seem to be reaching the limit of their ability to deal with the problem.[2]

At the same time, traditional needs of IT organizations persist; the need to respond quickly to new requirements of the business, the need to continually reduce the cost of IT to the business, and the ability to absorb and integrate new business partners and new customer sets, to name a few. Now Service Oriented Architecture (SOA) is being promoted in the industry as the next evolutionary step in software architecture to help IT organizations meet their ever more complex set of challenges.

Following are the popular architectures which are in use in implementing Information Systems.

2.1 Mainframe architecture

In a mainframe system all intelligence is within the central host computer. Users interact with the host through terminals that capture keystrokes and send that information to the host. User interaction can be done using PCs and workstations too. Mainframes and software built for them were popular in the 1960's and 70's, but there are still many organisations that use mainframe architecture for their information systems. As everything is centralised, tight integration of data is possible, and this leads to efficient implementations. A limitation though is that they do not easily support graphical user interfaces or access from geographically dispersed sites. While centralising the data is an idea that has again become popular, the architecture of hardware and software has changed.

2.2 File sharing architecture

The original PC networks were based on file sharing architectures, where the server downloads files from shared location to the desktop environment. The requested user job is then run (including logic and data) in the desktop environment. File sharing architectures work if shared usage is low, update contention is low, and the volume of data to be transferred is low[14][15]. Other limitation of this architecture is lack of security, lacks good graphical presentation etc. Clearly, this architecture is not suitable for a complex office system.

2.3 Client Server architecture

As a result of the limitations of file sharing architectures, the client/server architecture emerged. This approach introduced a database server to replace the file server. Using a relational database management system (DBMS), user queries could be answered directly. The client/server architecture reduced network traffic by providing a query response rather than total file transfer. It improves multi-user updating through a GUI front end to a shared database. In client/server architectures, Remote Procedure Calls (RPCs) or standard query language (SQL) statements are typically used to communicate between the client and server[14][15].

2.3.1 Two tier architectures

With two tier client/server architectures (also know as Two Tier Software Architectures or just client server architecture), the user system interface is usually located in the user's desktop environment and the database management services are usually in a server that is a more powerful machine that services many clients. Processing management is split between the user system interface environment and the database management server environment. The database management server provides stored procedures and triggers. There are a number of software vendors that provide tools to simplify development of applications for the two tier client/server architecture[14][15].

The two tier client/server architecture is a good solution for distributed computing when work groups are defined as a dozen to 100 people interacting on a LAN simultaneously. It does have a number of limitations. The main drawback is the scalability. When the number of users exceeds 100, performance begins to deteriorate.

This limitation is a result of the server maintaining a connection via "keep-alive" messages with each client, even when no work is being done. A second limitation of the two tier architecture is that implementation of processing management services using vendor proprietary database procedures restricts flexibility and choice of DBMS for applications. Current implementations of the two tier architecture provide limited flexibility in moving (repartitioning) program functionality from one server to another without manually regenerating procedural code[14][15]. Finally, it is difficult to control the software in the client systems. Presence of viruses leads to frequent crashes and the need to download client software and to propagate patches to the clients leads to lowering in reliability and accessibility of the system.

2.3.2 Three tier architectures

The three tier architecture (also referred to as the multi-tier architecture) emerged to overcome the limitations of the two tier architecture. In the three tier architecture, a middle tier was added between the user system interface client environment and the database management server environment. The client system needed no specialised software that needed maintenance. A web browser was the client interface.

There are a variety of ways of implementing this middle tier, such as transaction processing monitors, message servers, or application servers. The middle tier can perform queuing, application execution, and database staging. For example, if the middle tier provides queuing, the client can deliver its request to the middle layer and disengage because the middle tier will access the data and return the answer to the client. In addition the middle layer adds scheduling and prioritization for work in progress. The three tier client/server architecture has been shown to improve performance for groups with a large number of users (in the thousands) and improves flexibility when compared to the two tier approach. For three tier architectures many development environments are available for visually-oriented development of applications[14][15].

Three tier architecture with transaction processing monitor technology

The most basic type of three tier architecture has a middle layer consisting of Transaction Processing (TP) monitor technology. The TP monitor technology is a type of message queuing, transaction scheduling, and prioritization service where the client connects to the TP monitor (middle tier) instead of the database server. The transaction is accepted by the monitor, which queues it and then takes responsibility for managing it to completion, thus freeing up the client. When the capability is provided by third party middleware vendors it is referred to as ”TP Heavy” because it can service thousands of users. When it is embedded in the DBMS (and could be considered a two tier architecture), it is referred to as ”TP Lite” because experience has shown performance degradation when over 100 clients are connected. TP monitor technology also provides

- *the ability to update multiple DBMSes in a single transaction*

- *connectivity to a variety of data sources including flat files, non-relational DBMS, and the mainframe*
- *the ability to attach priorities to transactions*
- *robust security*

Using a three tier client/server architecture with TP monitor technology results in an environment that is considerably more scalable than a two tier architecture with direct client to server connection. For systems with thousands of users, TP monitor technology (not embedded in the DBMS) has been reported as one of the most effective solutions. A limitation to TP monitor technology is that the implementation code is usually written in a lower level language (such as COBOL), and not yet widely available in the popular visual toolsets[14].

Three tier with message server

Messaging is another way to implement three tier architectures. Messages are prioritized and processed asynchronously. Messages consist of headers that contain priority information, and the address and identification number. The message server connects to the relational DBMS and other data sources. The difference between TP monitor technology and message server is that the message server architecture focuses on intelligent messages, whereas the TP Monitor environment has the intelligence in the monitor, and treats transactions as dumb data packets. Messaging systems are good solutions for wireless infrastructures[14].

Three tier with an application server.

The three tier application server architecture allocates the main body of an application to run on a shared host rather than in the user system interface client environment. The application server does not drive the GUIs; rather it shares business logic, computations, and a data retrieval engine. Advantages are that with less software on the client there is less security to worry about, applications are more scalable, and support and installation costs are less on a single server than maintaining each on a desktop client[14]. The only limitation is that application server design should be used when security, scalability, and cost are major considerations[14].

Three tier with an ORB architecture.

It is based on improving interoperability and determine what the common Object Request Broker (ORB) can do. Developing client/server systems using technologies that support distributed objects holds great advantage, as these technologies support interoperability across languages and platforms, as well as enhancing maintainability and adaptability of the system. There are currently two prominent distributed object technologies:

- *Common Object Request Broker Architecture (CORBA)*
- *Component Object Model (COM, DCOM etc)*

Industry is working on standards to improve interoperability between CORBA and COM/DCOM. The Object Management Group (OMG) has developed a mapping between CORBA and COM/DCOM that is supported by several products [OMG 96].

The current distributed architectures such as CORBA, DCOM and RMI suffer from the following limitations:

- *They are location specific (i.e., a client must be pre-wired with the location of the naming/directory service)*
- *They are protocol dependent (i.e., a client and server can communicate only through previously agreed-upon protocols such as IIOP, ORPC, JRMP)*
- *They are tightly coupled (i.e., a client should be modified/upgraded whenever the service provider's interface changes)*
- *They do not support the notion of dynamic community formation and do not accept the notion of a device providing a service*

Distributed/collaborative enterprise architecture.

This software architecture is based on Object Request Broker (ORB) technology, but goes further than the Common Object Request Broker Architecture (CORBA) by using shared, reusable business models (not just objects) on an enterprise-wide scale. The benefit of this architectural approach is that standardized business object models and distributed object computing are combined to give an organization flexibility to improve effectiveness organizationally, operationally, and technologically. An enterprise is defined here as a system comprised of multiple business systems or subsystems. Distributed/collaborative enterprise architectures are limited by a lack of commercially-available object orientation analysis and design method tools that focus on applications [Shelton 93, Adler 95]. Distributed/collaborative enterprise architectures are limited by the lack of commercially-available, object-oriented analysis and design method tools that focus on applications (rather than large scale business modeling). Also the cost factor is huge for such kind of a system.

Service oriented architecture

Service Oriented Architecture is an architectural style whose goal is to achieve loose coupling among interacting software agents. A service is a unit of work done by a service provider to achieve desired end results for a service consumer. Both provider and consumer are roles played by software agents on behalf of their owners. What distinguishes an SOA from other architectures is loose coupling. Loose coupling means that the client of a service is essentially independent of the service. The way a client (which can be another service) communicates with the service doesn't depend on the implementation of the service. Significantly, this means that the client doesn't have to know very much about the service to use it. For instance, the client doesn't need to know what language the service is coded in or what platform the service runs on. The client communicates with the service according to a specified, well-defined interface, and then leaves it up to the service implementation to perform the necessary processing. If the implementation of the service changes, the client communicates with it in the same way as before, provided that the interface remains the same. Loose coupling enables services to be document-oriented (or document-centric). A document-oriented service accepts a document as input, as opposed to Java object. The client does not know or care what business function in the service will process the document. It's up to the service to determine what business function (or functions) to apply based on the content of the document.

Following are the Properties of Service Oriented Architecture

1. Reusability

   The biggest obstacle in reusing a software component is the incompatibility of the development platform or technology. In an SOA, the only characteristic of a service that a requesting application needs to know about is the public interface. The functions of an application are dramatically easier to access as a service in an SOA than in some other architecture. So integrating applications and systems can be much simpler.

2. Scalability

   The scalability of systems built on SOA is very high. Each time a new component needs to be added, all one has to do is to pick the segment to which it attaches. There is no need to touch the rest of the segments of the system. In case a new segment is to be added, it can immediately start functioning using the services published by the existing segments. Very less changes are required. And the best part is that most of the system is up and running while a change is being made in a segment.

3. Interoperability

   This is one of the most important features of SOA. The producer and consumer both are independent of each other and can function on different platforms and still communicate to each other using web services. There is no need for compatibility. The only constraint is that both the applications have to follow the same messaging format (SOAP, explained in the next Section).

4. Flexibility

   Loosely coupled services are typically more flexible than more tightly coupled applications. In a tightly coupled architecture, the different components of an application are tightly bound to each other, sharing semantics, libraries, and often sharing state. This makes it difficult to evolve the application to keep up with changing business requirements. The loosely coupled, document-based services in an SOA allow applications to be flexible, and easy to evolve with changing requirements.

5. Level of Integration

   By its very nature, SOA architecture is loosely-coupled. If the application requires tight integration, then such architecture may result in loss of functionality and may also result in an inefficient implementation. However, it is the job of the designer to partition the application appropriately, and almost all applications can be fruitfully partitioned to take advantage of the SOA architecture.

   An important component of SOA architecture is method used to connect the different components together, Web Services is the preferred choice and this is described next.

2.4 Web Services

According to the W3C a Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface that is described in a machine-processable format such as WSDL. Other systems interact with the Web service in a manner prescribed by its interface using messages, which may be enclosed in a SOAP envelope.[7] A web service serves a particular purpose in the system. It interacts with the system resources in its domain and returns particular pieces of information, on demand. Web services interact with clients to receive the clients' requests and return responses. In between the request and the response, a Web service applies appropriate business logic to process and fulfill a client's request. Designing an effective Web service starts with understanding the nature of the service to be provided-that is, how the service is going to interact with the client and how it is going to process and fulfill the client's request-coupled with how users of the service find and make their requests.

In a Web service scenario, a client makes a request to a particular Web service and the service, after processing the request, sends a response to the client to fulfill the request. When both the client and the Web service are implemented in a Java environment, the client makes the call to the service by invoking a Java method, along with setting up and passing the required parameters, and receives as the response the result of the method invocation. A Web service uses simple XML messages to communicate. The format of the message is specified by SOAP.

Web service needs to be accessible to its intended clients. Some Web services are intended for use by the general public. Other Web services are intended to be used only between trusted business partners (inter-enterprise), and still others are intended for use just within an enterprise (intra-enterprise). Regardless of whether a service is to be accessible to the public, other enterprises, or even within a single enterprise, the details about the Web service-its interface, parameters, where the service is located, and so forth are required to be made accessible to clients. WSDL provides a way for service providers to describe the basic format of web service requests. WSDL is used to describe what a web service can do, where it resides, and how to invoke it

Once the service is ready, there is an option to publish it in a registry. Publishing a service in a registry is a method of making the service available to clients. If one decides to publish ones service in a registry, the type of registry to be used is decided. Registries range from public registries to corporate registries available only within a single enterprise. UDDI is the standardized model used to publish a web service.

2.4.1 Reason for using Web Services

Today, the majority of software companies are implementing tools based around the new standards for Web services. Considering the fast development and the strong commitment of several important software companies like IBM and Microsoft, a wide range of ready-to-use services throughout the entire Web can be expected soon. Google for example, one of the most popular search engines on the web, already offers a Web service for web queries. An implementation based on the provided API is straight forward and requires but the basic programming skills. But Web services available on the Web will not remain the only application for these standards. Modularity and language-independency paved the way for combined applications, using basic Web services even in closed environments like corporate networks. Thus it is quite possible to implement organizations applications as Web services and make them accessible through published descriptions. The advantages of such architecture are obvious: services are completely independent of implementation or operating system and useable from everywhere within the entire network. Developers are encouraged to use the provided functionality without further knowledge of the involved code. Each Web service is registered at a centralized spot which makes service discovery much easier. Another trend in the development of computer systems is to adapt the human way of thinking, to address security aspects like access rights or judgment etc. An intuitive way of implementing web services to automate the offices of IIT Guwahati and provide access management is the main motivation for this project.

2.4.2 Publishing and Finding Web Services

In order to make itself available it uses three components for publication over the net:

1. SOAP or Simple Object Access Protocol is a standard for exchanging XML-based messages over a computer network-using HTTP. SOAP forms the foundation layer of the web services stacks, providing a basic messaging framework.

2. WSDL or Web Services Description Language is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. With SOAP, a communication between Web services is possible and structured and each participant knows how to send or receive the corresponding SOAP Message. The final step to complete the communication architecture of Web services is to define how to access a service once it is implemented. This is where the WSDL steps in. WSDL describes services as collections of network endpoints, or ports. Again it is an XML document with a defined

grammar where the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings.

WSDL documents use the following elements to describe a Web service:

- Types A container for data type definitions. In contrast to SOAP, WSDL can define types using some type system (such as XSD).
- Message A definition of the data being passed in a single RPC.
- Operation A description of an action (method) supported by the service.
- Port Type A set of operations supported by one or more endpoints.
- Binding A concrete data format specification for a particular port type.
- Port A single endpoint defined as a combination of a binding and the network address where it can be found.
- Service A collection of related endpoints.

Now that a Web service can be described completely, the only remaining problem is how a potential user can find the corresponding description (WSDL document). The following section deals with this last problem.

3. UDDI or Universal Description, Discovery and Integration is a directory model for web services. UDDI is a specification for maintaining standardized directories of information about web services, recording their capabilities, location and requirements in a universally recognized format. UDDI provides a method for publishing and finding service descriptions. It is a directory where Web services can be registered and assigned to a service provider, therefore forming a structure that resembles a yellow pages directory. Any user can browse the directory to search for a desired service and download the description in case of a match. UDDI is an industry initiative (ARIBA, IBM and Microsoft) to enable businesses to quickly and dynamically find and transact with each other. Searching a UDDI registry can be done by any UDDI browser or by the registry's Web interface if present. Microsoft for example, provides both a Web interface and the original UDDI registry to search for an entry. The corresponding WSDL File

can be found in the overview document if it was specified. The UDDI registry is implemented as multiple peer nodes where the registration across all nodes is replicated by node operators. Again, the data structure of the registry is XML yet a WSDL document cannot be published without additional precautions. The structure of a UDDI registry differs from a WSDL file because UDDI also supports business and service information. As a result, UDDI has no direct support for WSDL or any other service description mechanism. After adjusting the WSDL service description the UDDI registry contains the services general description, port bindings and a reference to the original WSDL File as Models. Every service interface of a WSDL file is published as a Model in the UDDI registry. A service interface includes types, message, portType and binding. Some elements of the business service are constructed using information from the WSDL service description. The service name for example, is used as the name of the UDDI's business service entry. Publishing the service description to a UDDI directory was the last step in the process to create a Web service.

2.4.3 Using Web Services

To finally use a Web service, several steps have to be performed. The order of the events, followed by a description of how to execute each step is given below.

1. Locating the Web service: This can either be done by browsing a public UDDI registry or by means of an existing WSDL document. It is possible to build a private UDDI registry as well. Private registries are easier to maintain due to their size but it can be hard to discover the UDDI registry's position. Sometimes, a company's main Web page is linked to WSDL documents, too.

2. Creating the SOAP Message: This is done by the development tool in most cases. Tools like Weblogic Workshop from BEA or Web service Development Kit from Microsoft will create valid SOAP messages for the methods described in the WSDL document or UDDI registry.

3. Transmission: Another advantage of message transport via HTTP is the

service provider's firewall setting. If the firewall permits Port 80 (HTTP POST/GET) connections, a SOAP message is able to pass through as well. If the firewall is unable to filter and process SOAP requests on the other hand, it leaves the system vulnerable to attackers who use the Web service's functionality for a potential attack.

4. Parsing the SOAP message is done by the provider's Application Server: The parser decides if the request is valid and decides which procedure to call.

5. Processing: The service provider calls all necessary procedures, or even other Web services, to complete the requested task.

6. Return the result: The result is wrapped in a SOAP reply and returned to the requestor where the client application can parse the message and evaluate the included data.

Chapter 3

The Proposed System

3.1 Required Attributes

The required attributes that need to be possessed by the proposed software at the minimum are as follows.

- **Fully Integrated Applications:** Integrated systems across processes constitute the most important requirement; data should be entered only once.
- **Single place integrated authentication system:** The proposed system must have a single authentication mechanism preferably from the existing LDAP database in use for computing services.
- **Role management:** Role management is of very crucial importance through out the system. A ’role’ can be defined as a set of access control grants that can be attributed to an account in the system (for example Director, Dean, HOD, Registrar, Payroll clerk etc.) Again a single user may have multiple roles. The roles are connected in the form of hierarchical tree based structure representing the ’Organizational structure’ of IITG. Appropriate procedures for ’shifting of roles’ must be there in adherence to the administrative protocol.
- **Authorization through User profile and roles:** Each of the functionality should be assignable to a role and in turn a role can be assigned to a user thereby ensuring the authorization.

- **Decision Support Tools:** Management requires on-line/real-time access to data and decision support tools.
- **Structured Design:** Common program architecture that minimizes maintenance efforts and facilitates sharing of resources across systems.
- **Flexible/ Upgradeable:** Systems must be able to be quickly adapted to changing organizational needs, and be upgradeable to emerging technologies.
- **Common User Interface:** Systems should have a similar ”look and feel” to minimize user learning curve across applications through graphical user interface.
- **Ad Hoc Analysis Tools (Report Generation):** Systems must have easy-to-use query tools to meet ad hoc analysis and report-generation requirements.
- **Industry Standards:** System should be built consistent with industry standards and run on multiple platforms; it also should support use of various input/output devices (e.g., scanning, Electronic Data Interchange), and support Internet access for transaction input and queries.
- **Client Integration:** Systems must be flexible to allow integrated use of Personal Computer applications for specialized analysis and presentation.
- **Web Access:** System must have the capability for web access to address requirements of IIT Guwahati Website Users.
- **Security and Control:** Systems must have appropriate features to prevent unauthorized access, to provide for review or approval of transactions by multiple departments, and to trace the processing of transactions by multiple users.
- **Storage and Backup:** Usually all data for present working year (which may be present financial year and/or present academic session) needs to be there online. Data for previous working years should be easily accessible restrictively. Appropriate modules for backing up data electronically and in hardcopies must be there.

3.2 Scope of Automation

The scope of the project is to implement a comprehensive, fully integrated Software System for IIT GUWAHATI. The project shall comprise the following components:

- Analysis of all functional components of the IIT Guwahati including activities of Departments, centres and sections. The following aspects of integrated software System should be looked at:
  - System functional requirements, appropriately prioritised, and interface requirements to related systems such as web enabled;
  - High-level input, processing and output specifications;
  - Identification of major existing processes/ procedures that could impede automation, and recommendations for remedial action;
  - User access requirements by location;
  - Platform software requirements i.e. system software, development platform(s), database(s), etc.
- Identification of a software package, or design and development of custom-developed software solution, for the IIT GUWAHATI integrated Software System.
  - The required customization of software packages required, and will have to identify the changes in the operating procedures of the current system so as to meet the features of the solution but should be minimised as much possible;
  - It will be required to develop the Detailed System Design (logical and physical), and the application software.
- User training and development of testing, implementation and operational procedures
- Installation of the software.

3.3 Existing state of Office Automation

IIT Guwahati currently operates in partly computerised and partly manual mode. In IIT Guwahati, some of the systems which are computerised to some extent are given below.

1. Accounting and Budgeting System (Based on Tally EES 8.2 unlimited user edition)

2. Stores and Purchase system (Based on Tally EES 8.2 unlimited user edition)

3. Payroll [includes module for GPF, CPF, Taxation system and other related activity to salary] (In house, developed in MS Access and Visual Basic)

4. Grade card and Academic records maintenance system [Used for generating grade card by entering data received from different department] (In house, developed in MS Access and Visual Basic)

5. Electricity Bill Generator [Used to generate electricity bill for the staff quarters and other residents] (In house Made with MS Access)

6. Departmental Inventory [used to keep track of equipments of departments] (In house Made with MS Access and ASP, php etc)

7. Engineering Cell Complaints (MySql)

8. Online notice board (In house Made with Perl/MySQL)

3.4 Infrastructure available at IIT Guwahati

The following are the glimpse of IT infrastructure already available in IIT Guwahati:

- The campus has PC's for almost every user and a large number of high-end servers.

- These PCs and servers are hooked together by a local area network.

- The campus is connected to the Internet through multiple leased lines.
- There are highly accessed intranet and internet websites

3.5 Approximate data volume

Table 3.1 gives the estimated number of existing records. This should give some idea of the volume of transactions. It is expected to comprehensively address issues relating to data volumes, hardware sizing and network configuration during the System Requirements Study component of the project.

Actor	**Estimated Numbers**
Employees	700
Student and alumni	10,000
Equipment	1,00,000
Others	3,00,000 records

Table 3.1: Approximate Data Volumes

Chapter 4

Automation of the offices of IIT Guwahati

4.1 Overview

The IIT administration has been divided into a number of departments, interacting with each other, creating a closely-knit system as show in figure 4.1.Finance and Accounts section lies at the centre of the system with Academics, Placements, Administration and Placements being some of the other important departments. The functions of each department have been listed below:

4.1.1 Finance and Accounts Section

IITG accounts are maintained under different customized Heads of Accounts. Expenditure sanctioning authority is vested in a number of officials like Deans, Head of Departments, General activities of accounts include budgeting, accounting, payroll, queries and reporting. Along with these correspondences with MHRD for funds related activities and receipt of fees are important activities of accounts section.

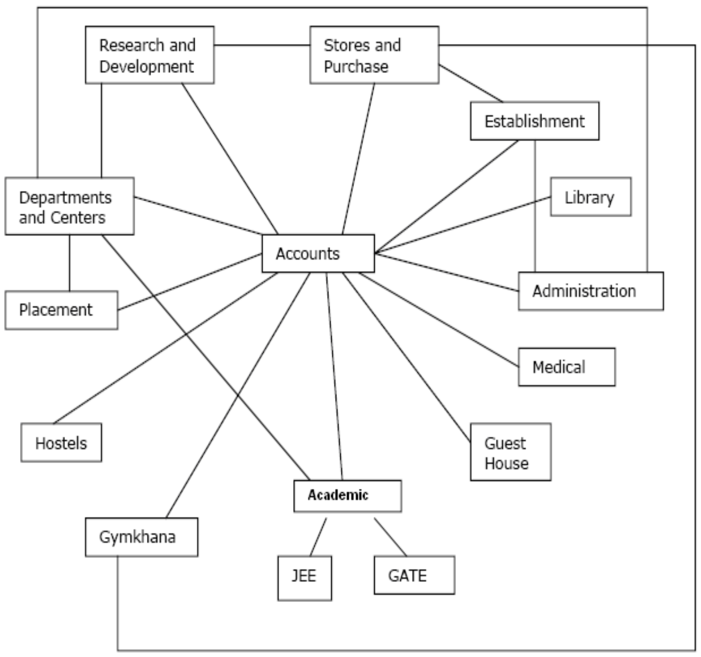

Figure 4.1: Offices of IIT Guwahati showing their Interdependency

4.1.2 Administrative Section

Activities of Administrative section include recruitment, employee information management, employee record management (leave, transfer, resignation, retirement etc.), generation of statistical report of employee database and announcement of holidays. The section also handles all matters related to statutory committees such as the Board of Governors, the Senate, etc. It takes care of legal affairs of the Institute.

4.1.3 Academic Section

Major activities of academic section are student's registration, scheduling of classes and classrooms, course registration, examination scheduling and maintaining academic records of students and publishing results.

4.1.4 Research and Development Section

Activities of Research and Development section are maintaining records of sponsored projects, including accounts, employee records, sponsoring agency records.

4.1.5 Medical Section

The main duties of medical section are patient registration, generation of electronic prescription, employee health record, laboratory records, and maintenance of inventory, reimbursement records and keeping medical leave records.

4.1.6 Departments and Centers

Academic activities of Departments include class/exam record maintenance, departmental library management, attendance of research scholar, and generation of no dues report for students leaving IIT Guwahati. The administrative activities of Departments include granting leave to employees, initiating purchase process for equipments and accessories and departmental inventory management system.

4.1.7 Stores and Purchase

Main activities of Stores and purchase section are tendering and receiving of bids, preparation of sanction and purchase orders, receiving of goods, central inventory management, synchronizing with accounts section for releasing bills.

4.1.8 Engineering Cell

The Engineering Cell handles all construction activities. Projects start with planning, then an architect is appointed. Then estimates are prepared, and then a tender call is issued. Received bids are evaluated, and a contract awarded. During construction, engineers inspect and monitor activities. Measurements of constructed portions are taken and bills prepared for payment.

4.1.9 Hostels

In hostels, one activity is to keep track of room occupancies. The other activity is to handle payment of mess bills by students. Catering is done by external contractors, and hostels monitor the bills and collect money from students and make payments to contractors. Late payments, dues, fines, payments for extra items, all these aspects have to be handled.

4.1.10 Student Affairs

The students' affairs system keeps the masters data of all students. Initial data is obtained from the JEE database prepared by all the IITs jointly. Data about a student's achievements and records are also maintained. Thus, awards, positions held, punishments received, are all maintained.

4.1.11 Placement

The Placement section handles all the on-campus placement activities. Students register for placement, companies express their desire to come for placement, and a calendar of interviews is created by the section. Data about companies have to be stored, details of the job positions of students also have to be kept.

4.2 Our Solution

The developed solution is discussed in the following sections.

4.2.1 Different subsystems

The best part about SOA is that we don't need to go into the workflow of other departments while modeling the flow of this section. Each time there is need of information from another department, it is assumed that a service exists to serve that need. And each time some information is to be made available, a service corresponding to that is published. So each of the subsystems (academic, admin etc) have been designed and developed independent of each other in a loosely coupled manner. Each of the modules has its independent database and set of applications. The obvious advantage is that the interdependency is reduced quite a lot and thereby is increased the maintainability of the systems. The teams for developing the individual subsystems were drawn from student for Summer Internship of various semesters from different institute like NIT Silchar, IIT Guwahati etc. The independent subsystems are then coupled with web services.

4.2.2 Authentication and Authorization

The Light Weight Directory Access Protocol (LDAP) directory service already available for the purpose of webmail authentication of IIT Guwahati users has been used to do the basic Authentication. As show in the Figure 4.2, the client will request the application server for any web application which will ask for the user credentials which will be verified in the LDAP server through a PAM-Java module namely JPAM[17]. On successful verification, the authorization module will contact the user role database and fetch the roles for that user. In case of return of multiple roles, user will be allowed to select the role. The access to the application will be on the basis of privilege of the role of that particular user.

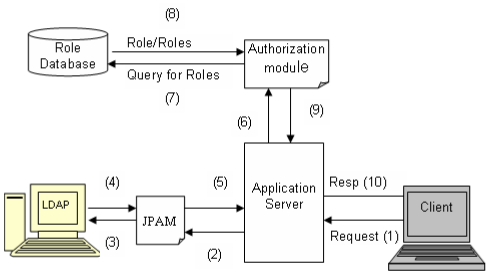

Figure 4.2: The Proposed Authentication System.

4.2.3 Security

The security issues are divided into two parts which are described as follows

System and Server Security

In this case the overall database server and application server is considered for possible attacks. The proposed scheme is given in figure 4.3

The database server and the authentication server are in a private network and separated from the user network by a firewall. These servers can be accessed only through application server, i.e. through the authentication and authorization module. Application server has an interface in the private network but can avail only the specific service which has been explicitly allowed in the firewall. Application server has another interface which is part of user network with a firewall to restrict the clients only to the desired service. The firewall has been implemented by iptables[16].

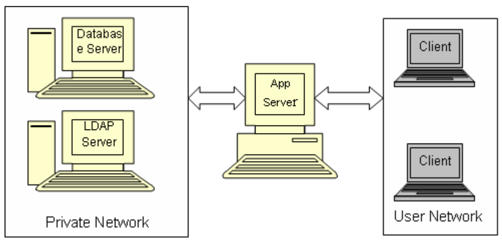

Figure 4.3: The System and Server Security.

4.2.4 Information flow security

The information flow security has been taken care by secure http. The Sun Application server has the support for https which was configured to make sure that data passing to and from Application server is encrypted.

From the Application Server, a digital certificate in JSSE (Java Secure Socket Extension) has been generated. This needs to be installed on the client machine for server identity verification. Similarly client certificate can also be generated from the JSSE which can be used in the client which will update sensitive data. Such operation will be denied without client certificate.

4.2.5 Role management

The role database is implemented as a mysql database. On successful authentication, the Authentication and authorization module which has been developed for this purpose is called and the role for the user is retrieved. Privileges are granted to roles which in turn roles are granted to users.

4.3 Functionalities of each subsystem

4.3.1 Academic subsystem

This subsystem covers the following broad functionalities:

- Course type record
- Define Courses
- Delete Courses
- Course Details
- Define Course Syllabus
- Add curriculum (core course)
- Add curriculum (elective course)
- Programme wise registered Credit limit
- Course offerings per semester
- Student Course Pre-registration
- Student Course Registration
- Student Grade Details
- Student Course Scheduler
- Automatic Grade and SPI, CPI Calculation

- Faculty Details
- Department Details
- Program Details
- Branch Master

This subsystem generates the following reports:

- Student Transcript
- Student Course Registration
- Grade List Course wise
- Curriculum details
- CPI, SPI details
- Faculty Details
- Course wise Grade List
- Grades
- Student List

4.3.2 Student Subsystem

This subsystem covers the following broad functionalities:

- Program Details (B. Tech, M. Tech, Phd)
- Student Category Details
- Specialization Details
- Student Details
- Scholarship Details

- Details of Students Selected For Scholarships
- Scholarship Payment Details
- Extra curricular activity details
- Student body (gymkhana) record details.
- Different student secretary details
- Achievement details
- Disciplinary action details

This subsystem generates the following reports:

- Generalised Student Query
- Achievement query
- The details of different types of Scholarship are maintained
- The list and details of all the students who are getting different scholarships can be maintained.
- Disciplinary action query.
- Query on student body office bearers

4.3.3 Administration Subsystem

Administration subsystem covers the aspects of Administration and Day-to-Day activities pertaining to an institute/Institute.

This subsystem maintains the following:

- Employee Category Details
- Pay Scale Details
- Promotions

- Joining Details
- Leave Types
- Leave Applications and sanctions
- LTC Details
- Dependent Details
- Gratuity
- Pension
- Beneficiaries
- Employee Insurance (GSLIS etc)
- Joining reports
- Yearly holidays master
- Conference grants details.

This subsystem generates the following reports:

- Admin Holiday Query
- Employee Query
- Employee Leave Record Query
- Employee LTC Record Query
- Employee Leave Balance Query
- Employee Conference Query
- Employee Position History Query
- Employee Insurance Beneficiary Query
- Employee Pension Beneficiary Query

- Employee Dependent Query
- Employee Gratuity Query

4.3.4 Research and Development Subsystem

Project Subsystem consists of all information pertaining to Research and Development projects being undertaken at Universities and Colleges. It maintains different versions of the projects and has the facility to revise an existing project version. It maintains the Employees/Agencies/Agency Contact Persons and Faculty Honorarium to Employees involved in the project.

This subsystem maintains the following:

- Details of Projects
- Employees involved in Projects
- Honorarium / Consultancy fees given to Employees
- Maintains different Sponsoring Agency Records
- Project Funds management
- Project Budget Management system

This subsystem generates the following reports:

- Details of funds received for a project - The report will give the inflow statement for a project for Project Development cycle.
- Statement of expenditure - The report will give the outflow statement for a project for Project Development cycle.
- Statement of Accounts - The report will give the inflow and outflow details for a project.

4.3.5 Store and Purchase Subsystem

Purchase Subsystem consists of all information pertaining to Indenting, Sanctioning, Purchase order, Suppliers and fund utilisation. This subsystem maintains the following:

- Supplier Master
- Supplier Contact details
- Item Master
- Budget master
- Purchase Indent raising
- Sanction sheet
- Creation of Purchase Order and Payments

This subsystem generates the following reports:

- Report for Supplier - The report gives the details of a supplier along with the contact details
- Report for Purchase Order - The report gives the details of a purchase order, along with the details of items ordered.
- Report for payment details for a Purchase Order - The report gives the details of payment made for a purchase order. Provides details of all the instalments of the payment.
- Report for Purchase Order for a duration - The report gives the details of all purchase orders for
  - Duration
  - A project or department,
  - All projects and/or departments.
- Report for fund utilization - The report gives the details of a fund already spent, and also committed amount.

4.3.6 Training and Placement Subsystem

The underlying objectives of Training and Placement system is to automate the working of Training and Placement Section which includes Maintaining Company Details, Invitation For Campus interview, Registration of Students, Preparation of Bio-Data, MIS reports required for Training Placement section and other value added services.

This subsystem maintains the following:

- Maintain Training and Placement Member Details
- Student and Company Registration
- Uploading of Resume
- Student and Company Notification After Registration
- Maintain and Search Company Details
- Requirement Matching of Companies to select student meeting there company criteria
- Maintains Details of Students Joining a Company
- Apply criteria for each student for ” No of Interviews”
- Maintain History of Company Selection Per Year
- Excel Sheet Interface to Upload Data in the system
- Summary of Final Placement

This subsystem generates the following reports:

- Details which are to be Sent to Companies
- Requirement Matching Report
- Schedule Details
- Student Company Status Report
- Placement record

4.3.7 Hostel Management Subsystem

This subsystem covers the following broad functionalities:

- Hostel Rooms Allocations Status
- Room allocations
- Mess bill generation
- Mess bill payment details updation
- Reports on room usage
- Reports on outstanding mess bill.

4.3.8 Finance Subsystem

This subsystem covers the following functionalities:

- Interfacing with tally 8.
- Tally generated XML data uploading.
- Tally generated XML parsing
- Storing tally parsed XML data into Mysql database

This subsystem generates the following reports:

- Department wise fund utilization
- Account head wise fund utilization
- Overall account statement
- Overall Fund status

4.3.9 Engineering Subsystem

This subsystem covers the following functionalities:

- Measurements Input
- Shape Information
- Area/volume calculation
- Tender Rates Input
- Tax master

This subsystem generates the following reports:

- Measurement Book Generation
- Bill Generation
- Work progress report

4.4 Proposed interconnection services in IITG office automation

- Student Academic Record - Service: This is a web service published by the Academic section, which reveals the information about student's details, past records and present status. It solely functions on the information available in the Academic Section's database.
- Obtain Student Database - restricted service: One of the important functions of Academic Section is to provide the students updated database to other departments like accounts, students affairs etc. This is restricted web service available to only selected departments.
- Student Personal Record -Service: This is a service published by the Student Affairs section, which reveals the information about students personal information like name, permanent address , contact no, hostel, local contact

details, guardian name and address etc. It solely functions on the information available in the Student Affairs Database.

- Employee Personal Record (without restriction) Service: This is a service published by the Administrative section, which reveals the information about employee's personal information like employee number, name, department, designation, address, contact no, etc. It solely functions on the information available in the Administrative Section Database.

- Employee Personal Record (with restriction) Service: This is a service published by the Administrative section, which reveals the information about employee's personal information like leave, LTC, dependent, and other facilities enjoyed by the employees etc. It is restricted and can only be viewed by the employee himself or the staffs involved in administrative section.

- Finance and Accounts (with restriction) Service: This is a service published by the Finance and Accounts section, which reveals the information about budget, different type of expenditure under different account head, fund status etc. It is restricted and can only be viewed by the concerned Head of different dept, centers, Sections and Director, Registrar etc. As finance and accounts section is already using Tally for their as their accounting software, it is decided not to alter it, but an interface is written to get the Tally data to MySql database in read only mode for viewing the accounting data for managerial purposes.

4.5 Flow of Information between various subsystems

While studying the existing system, it was initially found that the present keys are not sufficient to identify the records. For example, in the student subsystem, the roll no of a student cannot be used for the purpose because one student may have two roll numbers in the case of student changing a branch of study or taking admission into another program. The same thing is true for Administrative and management subsystem where the existing employee number is not sufficient to identify each person as an employee may resign from one post and join in another post. On further study, considering the reverse case, for any roll no or employee id, there is one student or employee and so it was found to be enough to identify any student or employee. Primary key for the local databases are id field which is actually serial numbers inside the local databases.

So for the flow of information, some global parameter needs to be passed which can do global identification. For the case of student, their roll no and for the case of employee their employee id will act as the parameter. It was also found that the email id of every member (student, staff, faculty, alumni) is unique and so this was chosen as login id and the existing LDAP directory service is used to authenticate the users with their password.

For the ease of the user this id will be transparent to the users and will be used only for the flow of information and record maintenance in the proposed system. User won't have to worry about the id. All the queries and processes will happen through the existing identifiers.

Chapter 5

Implementation Details and Technology Used

5.1 The Products Used

As this was an implementation where one of the major goals was ownership of the system by the Institute, a decision was taken to use only open source or free software. While open source software is the preferred choice, we had to make a compromise in one case, as open source alternatives were not found adequate. The software that were chosen are: Mysql as the database management system, Sun Application Server - platform edition as the Application Server, and LDAP server for storing user information. One of the major requirements was an adequate Web Services implementation as this is very important for SOA architecture.

5.2 Application Server

There are a number of commercial and open source / free application servers available in the market and a lot of them are quite comparable to each other. Sun systems application server - platform edition, Jboss, Apache Tomcat are some of the most popular open source / free application servers.

As mentioned above the needs of our architecture requires a good web services implementation. The Tomcat server is a Java based Web Application container that was created to run Servlets and JavaServer Pages (JSP) in Web applications. Though it provides good support for deployment of web applications, it does not provide any specific support for deployment of Web services. Both the JBoss application server and Sun systems application server provide support for web services. But there are a number of additional features available in Sun's application server, which are not present in JBoss. The first one is the availability of a ”deploy” tool in Sun application server, which can directly deploy WAR files as web applications creating web.xml descriptor files. In JBoss we need to generate the descriptors externally using a separate package. One more important difference is that Sun Systems App. Server has implemented web services security modules as described by Oasis Web Services Security (WSS) specifications. No such implementation has been done in JBoss.

As most of the other application server (e.g. Oracle 10g, weblogic etc) are proprietary and involved considerable amount of license cost, Sun Application server is the only server which is free and also can serve our purpose. Unfortunately it is not an open source product. However, as discussed above, none of the open source products had enough features to meet all the requirements. Hence we have chosen Sun Microsystems application server for our purpose. It is free for development, deployment and redistribution.

5.3 Database Server

We chose MySql 5.0 for the database management system. MySql is a multithreaded, multi-user, SQL Database Management System. MySql is available under the GNU General Public License (GPL). The latest version of MySql provides added facilities for partitioning of tables and their maintenance. For example, Optimize partition and Analyze partitions have been added. Although there are other open source database management systems, prominently, Postgres, the increasing popularity of MySql decided the issue in its favour.

5.4 Information Server

LDAP is the de facto standard in the open source world for implementing Directory Services. Since open source LDAP servers are available, there was really no other choice. Further, the existing infrastructure in the Computer Centre of the Institute uses LDAP for directory services.

5.5 Operating System Environment

Here too there was no choice as Linux is the de facto standard when it comes to open source. So all the above software are installed on Linux.

5.6 The Client Environment

In a 3 tier application environment, the client is just a browser. However, Ajax support is assumed in the browser. Ajax allows partial loads of web pages, improving response times by reducing network bandwidth requirements.

5.7 Web Application Development Styles

5.7.1 Web Application Deployment

To deploy any application, it must be packed into a WAR (Web ARchive) file. Alternatively, it can be placed in a directory which has a WAR Structure. A WAR structure is defined as follows

- The directory should contain a directory WEB-INF
- There should be an xml file web.xml inside the directory WEB-INF

Java classes/beans to be used should all be present in some package, and that package should be present in the classpath. It can also be present in the WEB-INF/classes directory.

Hence if there are index.jsp, page2.jsp and a bean SessionManager.class present in the package iitg, the following directory structure will prevail.

Module/

- index.jsp
- page2.jsp
- other image files
- other required directories
- WEB-INF/
  - web.xml
  - classes/
    - iitg/
      - SessionManager.class

Although normal HTML pages are perfectly valid in any application, we keep all pages as JSP files. For static pages (only HTML) it is an overhead (compilation of JSP, running of JSP and outputting of the same HTML which was present). However there is a chance that any static page may be made dynamic in future as requirements change. Hence at that time, there will be no broken links as no pages will be referring to any HTML page.

5.7.2 JSP Coding

JSP coding can generally be done without much problem. However, since we aim to use AJAX for this site, there are a few points had to be taken care of:

- As the main page loads, all links and buttons used on it and in subsequently loaded pages will be JavaScript links which will use the XMLHTTPRequest object to asynchronously ”GET” the next page from the server.
- Hence no form will have a SUBMIT button, but instead will have a normal button which will call a function to perform a ”POST” or GET operation, and will manually post all form values required for the next JSP page.

- Anchor tags in HTML pointing to another page will not be present. They will be replaced by anchor tags calling a JavaScript method to perform the ”GET” operation.

- The newly obtained pages will not contain any section links, logos and other links since they will already be present. These new pages will only have changed content, which be loaded in the relevant ”content” panel.

- To show that content is changed, an ”information panel” should be present, which will show that the new page is loaded, or some similar display, which will not intrude the user experience, while at the same time providing necessary information. For any information, the user just needs to check the ”info panel”.

- Other page components like buttons, links, etc should be modified with Javascript, and if necessary, using some server side help, but not by loading a completely new page.

- Use of JSP scriptlets in HTML pages is not promoted nowadays. It is hard for the HTML designer to interact with JSP scriptlets and check its outputs, leading to inferior design. However, in our case the designer and the programmer are the same, as also the bulk of design will be in the rest of the page, not in the ”content” panel. Hence it is safe to assume that JSP-HTML on same page will not cause any issues, and there is no need for further enhancements like Tag-libs. This is beneficial since a student handling this project in the future will not necessarily be proficient in J2EE and taglibs.

5.7.3 JavaScript coding

JavaScript is a very important factor of the development, since it can reduce more than half the potential load on the server and also increase the functionality of the application, giving a rich user-experience, with lesser coding time. However for this purpose, it is essential that the core functions generated by any team are communicated to all, reused, and designed extremely well, so that they can be modified to suit future requirements, without many problems.

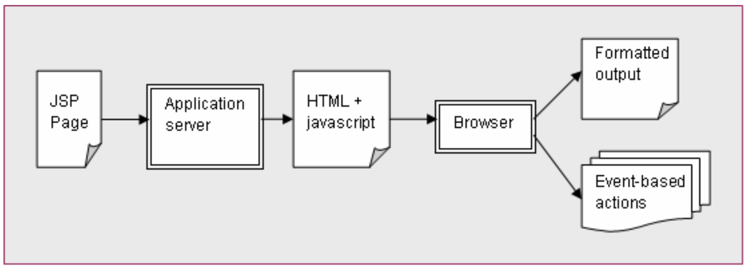

Figure 5.1: The style of Coding.

As shown in the figure 5.1, since a JSP page gives simple text as output, and output is eventually sent to the browser, even javascript may be generated using JSP.

This technique is especially necessary when javascript arrays or methods need parameters based on the server state. For example a query result contains as many columns as returned from the database. These columns can be sorted by javascript if the number of columns and their length, etc. is known. Thus, generating javascript using JSP makes the page independent of any module and thus the same page can be used in all modules without modification.

Chapter 6

Work Done

A web application for the Academic subsystem has already been deployed. This was developed with a team size 3. By using this system the course registration process has already became online. All the students of IIT Guwahati have registered their elective courses for the academic session 2006-2007. The system keeps track of backlogs, course drops, course adjustment etc. Also the system produces grades card for each of the students after online submission of the grades by respective faculty.

A web application for Research and Development Section has been developed with a team size of 5, which keeps track of the different research / consultancy projects their resources, finance, staff etc. The system is currently being tested, and data entry work is in progress.

A web application has been developed with a team size of 3 for automating the Administrative Section of IIT Guwahati. The system has already been deployed and partial data entry has been made. The system takes care of the service record of all employees.

Web applications for the following sections have been developed: Student Affairs, Hostel management, Placement, Stores and Purchase. The testing, data entry, and integration work of these systems are in progress.

The Finance and Accounts Section uses Tally. This has not been disturbed. A web application has been developed which extracts data from the Tally database once daily (or as programmed) and places the data in a Mysql database. Queries have been written to allow authorised users to view accounts data. Senior administrators can view the current state of expenditure, the status of particular account heads, and the budget position. Individual users can track the status of their bill, advance request, etc. More queries, including comparative anaylses with historical data needs to be developed and this is going to be taken up in the coming months.

A Web application for the Engineering Cell has been developed. This application automates some of the low level (and time consuming) functions of supervising construction work. The measurement book is at the core of supervision. Engineers have to note down dimensions of different shapes that have been constructed and then they have to do the calculations to get the area or the volume as per requirement. Then they have to add up all the calculations to get the ”measurements”. This is a time consuming and error prone task. The application takes in the raw data, information about the shape, and does all the calculations, including the additions. The rates of different items are entered once the tender is allotted. With this information (and other information such as taxes and other deductions to be made), bills are also prepared. This is another error prone operation due to the sheer volume of data. The application removes all these problems. The application is currently under testing and it is expected to be operational in the next three months.

Chapter 7

Conclusion and Future Work

The loosely coupled service oriented architecture was indeed of great help in designing this complex system which is fairly large in size and have many independent modules which don't have very high interactions among them. The automation system is already in use and progressing to achieve the goal of IIT Guwahati. The future work involves configuring a local Certifying Authority (CA) for identification and authentication of servers. Presently as there is only 1 server so the CA has not been implemented. Also writing api for user administration and management will add more flexibility in the system. Applications for sections not handled so far needs to be built. Some of the modules need further enhancements. A list of the work to be done is as follows:

1. Stores and Purchases (purchases: handling imports, mainly; Stores: assets register, consumables inventory)

2. Engineering Cell (planning, budgeting, keeping track of sanctions, deviations, time extensions, etc.)

3. Research and Development (more queries; changes based on experience)

4. Administration (recruitment sub-system (applicant information, experts' list information, selection scheduling), legal issues database, statutory meetings' record keeping (Board, BWC, Finance, Senate meetings records, annotations for easy searching, categorisation, etc.)

5. Academic (fine tuning based on experience; more queries; access of historical data)

6. Medical Section (only preliminary work has been done for this module) Systems for Departments and Centres (no work has been taken up so far)

7. Finance (pay roll package needs implementation; budget preparation package is required)

8. Library (library has a package, Libsys, to handle its functionalities. We need to add interfaces to / from this system to the rest of the system).

9. JEE, JAM, GATE (packages to handle internal operations are required; examples include keeping track of remunerations to be paid, of work allocated and done, of purchases made).

10. Work Flows to be installed so that many functions can be done electronically (leave application and grant; advance request and grant; etc.) Integration of the system to an IIT Guwahati Portal and to the all-IIT portal.

Bibliography

[1] Yih-Cheng Lee, Chi-Ming Ma, Shih-Chien Chou, “A Service-Oriented Architecture for Design and Development of Middleware”, Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSEC’05) 0-7695-2465-6/05

[2] Kishore Channabasavaiah, Kerrie Holley, Edward Tuggle, Jr., “”Migrating to a service-oriented architecture, Part 1”, 16 Dec 2003 `http://www-128.ibm.com/developerworks/library/ws-migratesoa/`

[3] Marijn Jansen, “Centralized or Decentralized Organization”, ACM International Conference Proceeding Series; Vol. 89, 2005

[4] Ramon Padilla Jr., “Is the best IT model the centralized or decentralized approach” , September 7, 2005 , `http://government.zdnet.com/p=1718`

[5] He Gou, Feng Chen, Yuxin Wang, “A Reusable Software Architecture Model for Manufactory Management Information System”, Proceedings of the 26th Annual International Computer Software and Applications Conference (COMPSAC’02) 0730-3157/02

[6] John C. Grundy, Mark D. Apperley, John G. Hosking, Warwick B. Mugridge, “A decentralized architecture for software process modeling and enactment”, Volume 2, Issue 5, Sept.-Oct. 1998 Page(s): 53 - 62,10.1109/4236.722231

[7] Booth, D., “W3C Working Group Note 11: Web Services Architecture”, World Wide Web Consortium (W3C), February 2004, `http://www.w3.org/TR/ws-arch/#stakeholder`

[8] Ed Ort, “Service-Oriented Architecture and Web Services: Concepts, Technologies, and Tools”, April 2005, `http://java.sun.com/developer/technicalArticles/WebServices/soa2/SOATerms.html#soawhy`

[9] Sean Brydon, Greg Murray, “Designing Web Services with the J2EE(TM) 1.4 Platform: JAX-RPC, SOAP, and XML Technologies”, Sun Microsystems, January, 2004 `http://java.sun.com/blueprints/guidelines/designing_webservices/html/`

[10] Wendy Boggs, Michael Boggs, “UML with Rational Rose 2002”, Sybex inc., 2002 ISBN: 0782140173

[11] Robin Schumacher, Improving Database Performance with Partitioning, January 2003 `http://mysql.osuosl.org/tech-resources/articles/performance-partitioning.html`

[12] Marijn Jansen, “Centralized or Decentralized Organization”, ACM International Conference Proceeding Series; Vol. 89, 2005

[13] Vaibhav Aggarwal, “Service Oriented Architecture Based MIS”, BTech Thesis, Dept of CSE, IIT Guwahati, April 2006

[14] George Schussel, “Client/Server Past, Present, and Future”, `http://www.dciexpo.com/geos/dbsejava.htm`

[15] Herb Edelstein, “Unraveling Client/Server Architecture.”,DBMS, (Vol 7, Num 5) Page 34-41, May 1994.

[16] Netfilter Core team, “Iptables, an userspace packet filtering program”, `http://www.netfilter.org/projects/iptables/index.html`.

[17] JPAM team, “JPAM, a Java-PAM bridge ”, `http://jpam.sourceforge.net/index.html`.