3-2016

Automation of Invoice Process

Ramya Gourisetty
St. Cloud State University

Automation of Invoice Process Fee Calculation

by

Ramya Gourisetty

A Starred Paper

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Masters of Engineering Management

March, 2016

Starred Paper Committee:
Ben Baliga, Chairperson
Hiral Shah
Balasubramanian Kasi

Abstract

This project has been carried out to automate the Billing Invoice Process fee calculation in an organization which provides software solutions to the Medical Device industry and integrated tele health services.

This Billing Invoice Application is built to create a Sale agreement for a new customer, Fee calculation and generate the Invoice for a customer. This project provides the functionality to calculate the fee before generating the Invoice of a customer. This system provides the functionality to manage monthly service fee and ASP fee, Supply fee, Shipping fee, Refurbishing fee, Miscellaneous Parts fee and also manages Bundle fee depends on the devices ordered by a customer. It also manages the fee calculation in an individual patient level.

The current process of dealing with the fee calculation should be replaced in a manner such that only a new customer’s sale agreement should be set up manually as per the customer’s contract and moving forward from that point, all the actions like bundling the devices based on customers order and transaction type, determining the price tier, calculating the all fees, generating the Invoice report etc., should be automated and the users should be able to monitor and have control over all the actions if necessary.

Acknowledgement

I would like to take this opportunity to thank Dr. Ben Baliga, Professor and Graduate Director for Engineering Management Program at St. Cloud State University for the valuable guidance, motivation and for his continuous support throughout the program.

I would like to thank Dr. Hiral Shah, Associate Professor for Engineering Management Program at St Cloud State University for all the support and guidance throughout the Program.

Also, I would like to thank Prof. Gary Nierengarten and also Dr. Kasi Balasubramaniam for taking time to guide and provide support throughout the entire project which has ultimately lead to the successful completion of the project.

This project wouldn’t have been a reality without each and every one of the above mentioned people.

Table of Contents

Page

Chapter Page

List of Tables

Table Page

List of Figures

Figure Page

Chapter I: Introduction

Introduction

The outline of the project is to automate the fee configuration module of the Billing Invoice application in an organization which provides software solutions to the Medical Device industry.

Billing Invoice System is built to generate the Invoice report for a customer. Automation of Fee configuration module is a part of Billing Invoice application which provides the functionality to calculate the fee of a customer before generating the invoice. Fee configuration is mainly depends on the Customer, it’s fee category like Bundle Fee, Supply Fee, Asp Fee, Shipping Fee, Refurbishing Fee, it’s Sale-Tax, CPI adjustment and it also depends on the order transaction like New order, Existing, Return, Exchange, Warranty and Replacement. This system manages customized fee if applied for the customer depends on the CPI adjustment and Sale Agreement contract. It also manages the fee calculation depends on Service Fee credit like if there is any discount applicable for a customer. This Implementation is followed by Software Development Life Cycle (SDLC) methodology.

The current process of dealing with fee calculation when replaced with a new system, only a new customer’s sale agreement should be set up manually as per the customer’s contract and moving forward from that instance, all the sequential actions performed should be automated and the users should be able to generate the Invoice report, monitor and have control over all the possible actions if necessary.

Problem Statement

Till now in the Finance department, the fee calculation is done manually by accounting team using the excel sheets before generating the invoice for a customer, which consumes a lot of man power and time. This project is to automate the fee calculation process write from the customer’s contract start date and until the invoice has been generated.

Nature and Significance of the Problem

For a growing Medical Device and Health Services company it is more important to do the things, which are more critical, with in no time and with accuracy. The current way of dealing with the fee calculations before generating the customers invoice should be replaced completely in such a way that only a new customers data or cost of the services should be entered manually and from there, bundling the devices depends on the order transaction (like new order, existing, warranty, replacement, exchange and returns) and calculating the fee for bundling devices, calculating shipping fee, refurbishing fee, supply fee, Asp fee, Sale-Tax etc., should be automated and the user should be able to monitor and review the invoices if necessary.

This project is initiated to resolve the problem by designing an application through which the users can manage all fee calculations depends on the transactions before generating the invoice of a customer. This application requires a username and a password to log in to their respective user profile. The various tools used are as follows.

Front end user interface: Telerik DevCraft's Desktop UI Controls.

Data binding: Microsoft Visual Studio using C#.

Database: Microsoft SQL Server.

Framework: C#.NET.

Objective of the Project

The primary objective of this project is to reduce the use of manual operations in the Billing Invoice Application before generating the Invoice of a customer. The high-level objectives are as follows:

- Fee Configuration process will be automated.
- Device bundling process will be automated.
- Replace all fee calculations with a stable, sophisticated and automatic procedure using a web application

Project Questions/Hypothesis

Once the project has been successfully implemented, following questions can be explained in a very detailed manner.

- What percentage of the usage of the manual work would be reduced after implementing the project?
- Which additional tasks can be achieved by using this project other than fee configuration, bundling devices?
- How stable is the automated project environment when compared to the manual work?
- How much accuracy of results can be expected from the automated process?

Limitations of the Project

Even though all of the fee configuration process is automated, still some human interaction is necessary while setting up the fee for all new customers depends on their contract. While monitoring for the unbundled devices depends on the transactions like returns and exchanges, the system acts according to the rules and objectives set for it to suggest the transaction is incorrect. This is totally depends on the customer. So depending up on the customer and the transactions, on which the product's decision making depends, are laid down manually.

Once it is updated manually, there need not be any manual updates till there is any change in the transaction or customer's requirement.

Development and testing team should support until the new module stabilizes. And in some cases, need to load the data manually before generating the invoice. This would be a very useful enhancement in the future which reduces the delay in the system that's caused because of non-availability of the user when there is a sudden change in the customers demand.

Definition of Terms

ASP Fee: ASP is known as Application Service Provider and it is applied to new, existing, exchange and return hub device.

Supply Fee: The supply fee will be calculated by supply category (like BP cuff) and quantity for each order.

Bundle Fee: This fee will defined depends on the device combination.

CPI filter: This will be used to determine which fee will be applied in CPI adjustment process and CPI date is specified in the Sale Agreement Tab.

Sales-Tax option: Will allow to specify if a customer has a fixed, Exempt or Sales-tax.

Refurbishing Fee: This fee is only applied to Return and Exchange Order type.

Minimum RPM Fee: Is used to determine minimum service fee charged if number of hub devices ordered lower than this value

SDLC: SDLC is a methodology which defines the software application development process.

SQL: SQL is known as Structured Query Language which is a relational database management system developed by Microsoft. Primary function of this product is to store or retrieves the data as request by a software application or by the user. This SQL server has different versions that can be used depending on the nature of the application and the size of the data.

Data Import: This is performed in the system manually depending upon the request raised by the user. The system will use appropriate web services to request import and load the data from Data Service.

Summary

This chapter is mainly focused on describing the introduction of the project, the problem statement, project objective, nature and significance of the project. Project questions that will be answered after the successful implementation of the project, limitations of the project and all the terms and definitions that are of primary importance for this project have also been described out in this chapter.

Chapter II: Background and Review of Literature

Introduction

This chapter mainly describes about the background related to the problem, reviewing the literature of the problem and also describes about the literature related to the methodology that has been implemented in the process of solving the problem. This literature review is the research for which methodology would be best suited for the current problem situation and it can be used for further analysis in future to gain more knowledge over the problem.

Background Related to the Problem

This Billing Invoice Application is mainly built to generate the invoice report for a customer. And which provides software solutions to the Health Device Industry and integrated tele health services. In this application fee calculation will play an important role and which consumes more time in manual work. In today's world, continuous change and improvements are required to meet with the ever changing demands of the society. Software applications are important which can play a very important role in any organization. A clearly defined software application lays the foundation for better control, efficiency and growth of any company.

This projects main purpose is to automate the fee configuration module in the Billing Invoice Application where the user can create Sale agreement page for a customer through this Billing Invoice Application and then depends on the customer's agreement, the fee calculation will be done through this automated module and an Invoice report can be generated to a customer. This application requires a username and a password to log in to their respective user profile. The various tools used are as follows.

For example, once a new customer’s sale agreement page is set up, depending upon the customer’s services and transactions like if the customer ordered a new device, peripheral and supply items, monthly service fee and device fee will be calculated according to the service automatically. After an extensive analysis of all the methodologies, traditional waterfall model is selected because of its rigidity and user friendly characteristics.

As part of this project, a plan was developed to gather the requirements, prepare specification, design, develop, test and implement the module using the traditional waterfall methodology. Waterfall model has given us the flexibility in managing all the phases of project life cycle with an aim to reduce the time and effort.

Literature Related to the Problem

Ae per Arthur Gehring, Datacap, Inc. and Optical Image Technology, Inc., every business is familiar with the arduous process of paying bills for goods and services, not to mention the sea of paper and the information management challenges it creates. The approval of invoices and resolving of billing discrepancies can require many man hours. Inefficiencies may prevent organizations from taking advantage of invoice discounts and may result in late payment fees or even damage to a company’s reputation. Invoice digital capture and automated workflow addresses these challenges. Together, they provide a seamless flow of accurate and timely information, expediting processing and saving precious human capital for more important projects. To automate the invoice processing the below technologies were used.

A Software Development Life Cycle (SDLC) is a series of phases which provides a model for software development and lifecycle management of an application. This mode is a conceptual model which is used in project management and defines the various stages of a software development project. Figure 1 shows different phases associated with Software Development Life Cycle.

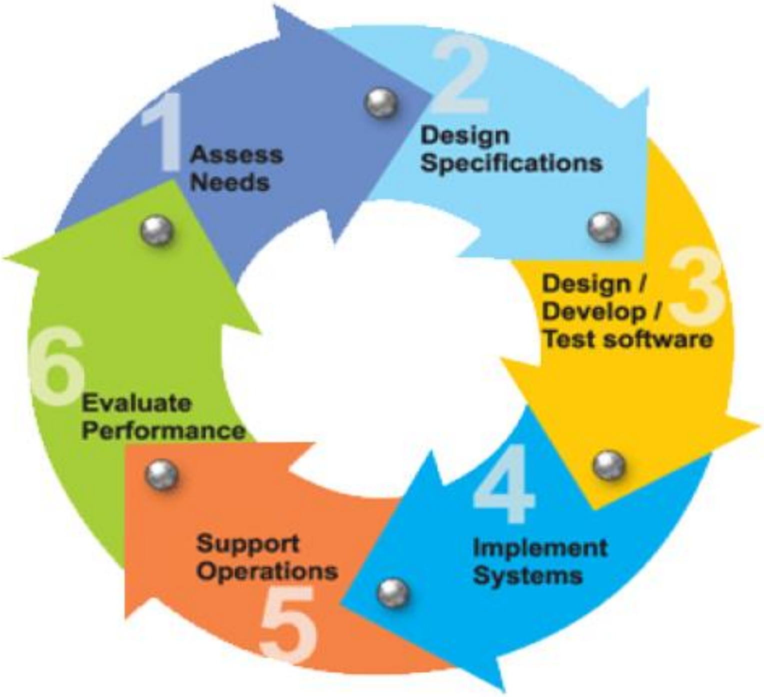


Figure 1: Different Phases of Software Development Life Cycle (Yaraamjad, 2006)

The development of an information system progresses through four major stages. Requirements engineering is one of the first stages: understanding client needs and defining them in the form of a structured set of requirements an information system must meet. This process, can be distinguished in two phases: (a) requirements capture, careful interaction with all those interested in the software application, usually called stakeholders, aimed at the fundamental purpose of obtaining information, the raw requirements; and (b) requirements analysis, i.e., the methodical study of the information acquired and translate it to an understanding of what the system must do to cater to client needs (Génova, Fuentes, Llorens, Hurtado, & Moreno, 2013).

Telerik DevCraft's Desktop UI Controls: Telerik DevCraf is a professional .NET development suite. Telerik empowers its customers to create compelling app experiences across any screen. Our end-to-end platform uniquely combines industry-leading UI tools with cloud services to simplify the entire app development lifecycle. Telerik UI for ASP.NET AJAX introduces a new Bootstrap-inspired skin, a navigation control designed for building responsive Web sites, as well as lightweight rendering mode for multiple control (Ramel, 2015).

With UI libraries for every .NET platform, tools that speed up development and intuitive API, DevCraft is the most complete toolbox for building modern and future-proof business applications. According to Onderj Balas (2015), at its core, communicating with an API simply involves creating and sending an HTTP request to the API, and then doing something with the response. Most modern APIs will return JSON, though you may run into APIs that return only XML. Many also allow the caller to specify which format they would like to receive data in by specifying the preferred format in the Accept header of the HTTP request (Balas, 2015; Ramel, 2015; Telerick DevCraft, n.d.).

C# language and the .NET Framework: According to Rabab J. Mohsin (2013), the .NET Framework is Microsoft’s object-oriented programming platform that simplifies application development in highly distributed environments, including the Internet C# programs run on the .NET Framework, an integral component of Windows that includes a virtual execution system called the common language runtime (CLR) and a unified set of class libraries. The CLR is the commercial implementation by Microsoft of the common language infrastructure (CLI), an international standard that is the basis for creating execution and development environments in which languages and libraries work together seamlessly.

Source code written in C# is compiled into an intermediate language (IL) that conforms to the CLI specification. The IL code and resources, such as bitmaps and strings, are stored on disk in an executable file called an assembly, typically with an extension of .exe or .dll. An assembly contains a manifest that provides information about the assembly's types, version, culture, and security requirements (Mohsin, 2013).

Microsoft SQL Server 2012: Ross and Stacia (2012) explores the exciting enhancements and new capabilities engineered into SQL Server, ranging from improvements in operation to those in reporting and management. This is a relational database management system developed by Microsoft. Like its predecessors, SQL Server 2012 comprises a set of programming extensions to enhance the Structured Query Language (SQL), a standard interactive and programming language for getting information from and updating a database.

New Capabilities of SQL 2012 compare to SQL 2008:

- Query Page splitting is implemented in sql 2012
- It can support up to 15,000 partitions.
- ORDER BY Clause with OOFSET/FETCH options.
- Analysis Services will include a new BI Semantic Model which is a 3-layer model that includes Data Model, Business Logic and Data Access (Ross & Stacia, 2012; http://whatis.techtarget.com/definition/SQL-Server-2012).

Literature Related to the Methodology

Software Development Life Cycles model is the methodology to be used widely in Software Engineering to ensure the success of the project and also considered as famous approach to the systems development life cycle. This describes a development method that is linear and sequential. This method is very simple to use and understand. In this method, the whole process of the software development is divided into multiple phases. In this methodology, once a phase of development is completed, then only the development needs to proceed for next phase. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases where the outcome of one phase acts the input for the next phase. The progress in software development is seen as flowing downwards through the phases of

- Requirements
- Analysis
- Design
- Implementation
- Testing
- Deployment

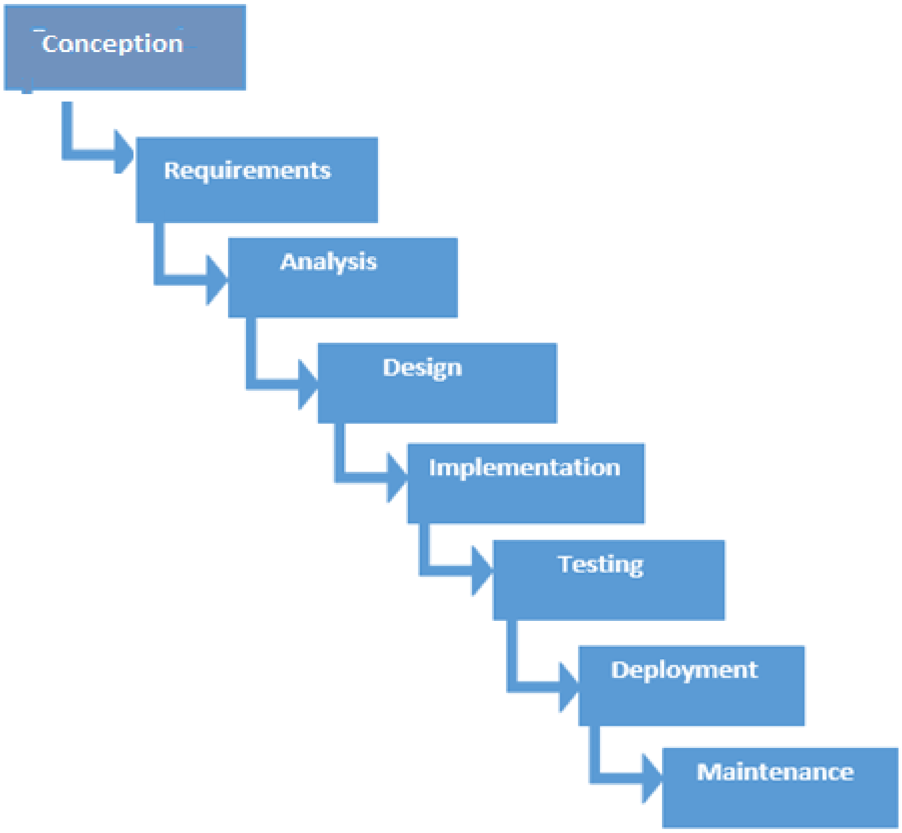


Figure 2: An Example of Waterfall Model (Waterfall Model, 2011)

Conception: This can be triggered when a problem is perceived and also it can be triggered by different factors like technological advances, legal requirements and competitive advantage.

Which involves identifying goals to be achieved after the problem is resolved, estimating benefits in the new system compare to the current system. In this case a business case provides the information regarding whether the manger needs to proceed with the proposed project or not before starting the development.

In this phase significantly enhance a system is identified, its cost is assessed, risks and other project planning approached are defined.

Requirements: This is the first phase of the project where all the requirements can be gathered and documented. In this phase, communication with the user is front loaded and involves a macro level study of requirements.

Analysis: In this phase gathered requirements will be analyzed in detail to determine whether the requirements are valid or invalid and arriving at the exact requirements of the proposed system.

This phase involves freezing the requirements before the design phase begins. Figure 3 shows effective requirements analysis includes four types of activity.

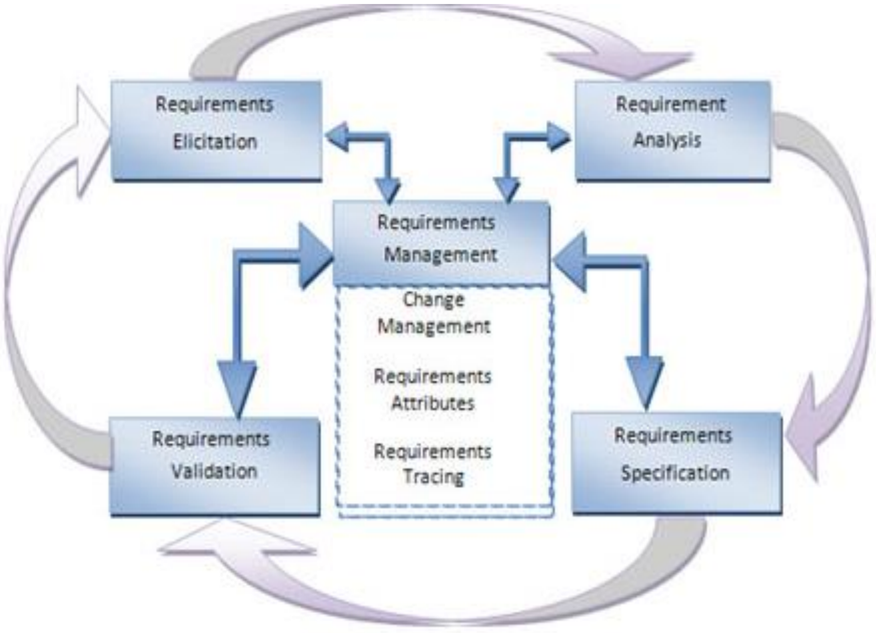


Figure 3: Requirements Analysis (P&R IM's)

Design: This phase involves translating the business requirements into a logical structure which is called design that can be implemented in a programming logic.

Implementation: In this phase actual code will written. This phase is belongs to developers in this waterfall method where they take the project requirements and specification and develop the application.

Testing: After the completion of development phase, this testing phase starts and where we test each unit and make sure that the developed code is working as expected. And all other testing activities are performed in this phase.

Deployment: After the completion of testing, make sure there is no issue or defect or any kind of problem and then the application or project is deployed to production. Then the user starts using the application.

Maintenance: In this phase after the project is deployed in production, always keep an eye on the application and provide all the necessary bug or issue fixes if occurs in production or reported by the user and also always keep updated the product with new updates developed or available.

Each software application developed is different and requires an appropriate SDLC process to be followed based on different factors. Some of the situations where the use of Waterfall model is most suitable are:

1. Requirements are very well documented, clear and concise.
2. Product definition is stable.
3. Technology is understood and is not dynamic.
4. No ambiguous requirements.
5. Ample resources with required expertise are available to support the product.
6. The project is short.

Advantages:

- Easy to use and understand.
- Good technical is part of the deliverables and it is easier for new developers to get up to speed during the maintenance phase.
- Total cost of the project plan can be estimated accurately after the requirements have been defined.
- In this phase errors are identified at the initial stage and hence time and money.
- Cost effective.
- Each phase developed completely.
- Very less chance to rework because of sequential process.

Disadvantages:

- Less effective if the requirements are not clearly defined at the earlier stage.
- It does not allow for much reflection or revision.
- Once the project is in the testing stage, it is very difficult to go back and update something that was not well-documented or thought upon in the concept stage.

Summary

This chapter is mainly focused on describing background of the problem, in depth details of the literature related to the problem. Also, all the background literature review towards the methodology of the project has been explained in a detailed manner. In the next chapters, the frame work of the project, detailed phases of implementations, budget, project timeline and their advantages will be listed.

Chapter III: Methodology

Introduction

This chapter focuses mainly on explaining the procedure that has been used for solving the problem and also understand about the methodology that has been used to achieve the solution for the problem. The main reason behind this module is to create an automated process for the user to calculate the fee before generating the invoice report. Also, data collection method and timeline of the project are explained in detailed manner.

Design of Study

Initial Requirement Meeting: In this phase, after the Business people/User approached the IT team about the product which can meet all the requirements they had in their mind, Senior Finance Manager, IT Project Manager and the Business Analyst will put their thoughts together in discussing the timelines of the project, budget of the project, resource allocation. Once they came to a conclusion regarding timelines, budget and all requirements that are expected by the users are collected by the Business Analyst. Then after requirements are analyzed and confirmed with user, Business Analyst will come up with high level and detailed requirement documents like BRD, Process documents. The main tasks in initiating the project are:

1. Determining the objectives for the project.
2. Defining the scope of the project.
3. Creating strategy for the project.
4. And creating work break down structure for the project.

Functional Design and Analysis: The projects Business Analyst have detailed out a functional specification document which met the initial requirements from the client. This can be updated based on the changes that have been proposed. And also if there are any additional requirements that were proposed by client or user can be added. It is always been good to share the design document with the user because if there is any change required, user can suggest before starting the next phase and more over always user has always been involved in the project and user will having direct transparency and they can suggest whenever it is required.

Technical Specification Design: This stage involves translating the identified requirements into a logical structure. After analyzing the requirement document from Business Analyst, developers came up with a technical specification document that met the initial requirements of the business. As these requirements are raw and converting view of an end user to reality is not easy, technical specifications are very important for the development to start. This specification has been modified many times once developers started working on the project. Some of the additional requirements that were proposed by users have been added.

Development and Testing: The functionality of this module has been divided into a number of smaller tasks which are assigned to each individual software developer. By end of every week all the work done by developers will be integrated into pre-production system. All the development done will be tested thoroughly before it is made available to the end users. After the tasks developed by the developers, those will be tested thoroughly in different stages of testing before it is made available to the end users.

- Unit Testing: In Unit Testing, developer should complete a series of tests that will execute each line of the code and achieve the expected result after completing the code. This should be completed prior to releasing the code for System or integration testing.
- System Testing: In the System Testing, a developer will be completing a series of tests which execute all of the code for each task and also interface with other code modules with the same system. This ensures the functionality within the system is operating correctly before the next stage of testing.
- Integration Testing: Which is called as End to End test, this testing involves all of the impact areas and applications. All modules should operate like production, should have the capability to send and receive the data same as manual process.
- Regression Testing: Testing the products or services that use the code modules that have been changed within this project to ensure they still function and are not negatively impacted by the changes for this project.
- User Acceptance Testing: User/Business will be testing the code with the test scenarios provided by the analyst and if the code passes the test, it will be moved to pre-production system.

If any issue found in testing, developers need to fix the bug and again need to go through all the testing process. This will be continued until the code worked perfectly.

Deployment: After the functionality developed as per the user's requirement, the next step is deploying the code. Before the deployment Business Analyst, Project Manager will have a meeting with users to discuss about the final product. After getting the confirmation from users, developers will deploy the code into production system.

Post Production Support: After moving all modules to the production system, need to give post production support.

Figure 4 describes the life cycle of the project. This project life cycle shows the different stages of a project. The different stages in projects life cycle include initiation, planning (which includes requirements gathering, architecture and design gathering), execution which means development cycle and finally project closure (which includes testing, monitoring and control and get the product feedback).

Once all the phases like initiating, planning, execution and closing, after the project deployed into production, need to give continuous support and maintenance if there is any issue raised by the user or even if any new requirement needs to be added to the application.

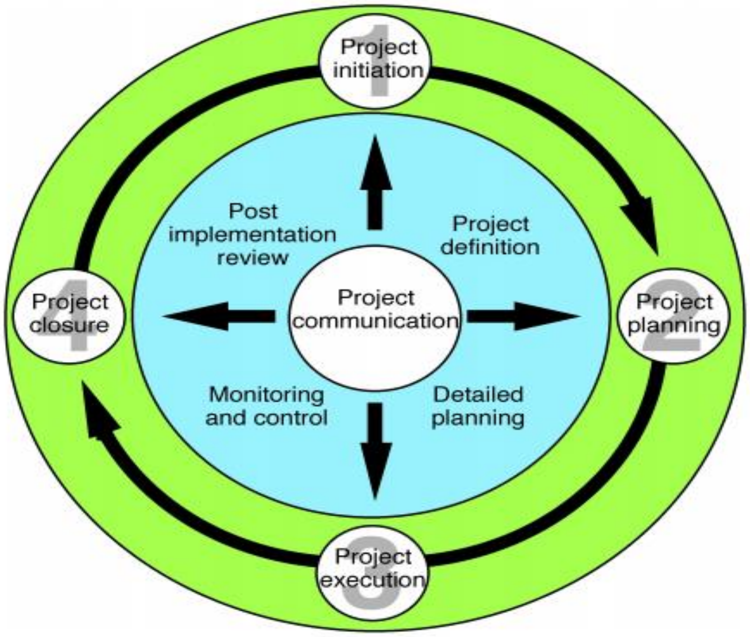


Figure 4: Project Life Cycle (Westland, 2006)

Framework and Integration: The framework will provide the basic structure and the objects that are necessary to implement as per the requirement of the project. In the integration phase, all the code will be integrated on to the demo and the developer systems which will be sent to the client for testing.

Data Collection

In general, the challenge of collecting software engineering data is to make sure that the collected data can provide useful information for project, process, and quality management and, at the same time, that the data collection process will not be a burden on development teams. Therefore, it is important to consider carefully what data to collect. The data must be based on well-defined metrics and models, which are used to drive improvements. Therefore, the goals of data collection should be established and the questions of interest should be defined before any data has been collected. Data classification schemes to be used and the level of precision must be carefully specified.

The collection form or template and data fields should be pretested. The amount of data to be collected and the number of metrics to be used need not be overwhelming. It is more important that the information extracted from the data to be focused, accurate, and useful rather being overloaded and finally turning out to be waste of resources towards the data collection efforts. Gathering large amounts of data is quite common when people starts to measure software without a prior specification of purpose, objective, profound versus trivial issues, and metrics and models (Kan, 2002).

Basili and Weiss (1984) propose a data collection methodology that could be applicable anywhere. The schema consists of six steps with considerable feedback and iteration occurring at several places:

1. Establish the goal of the data collection.
2. Develop a list of questions of interest.
3. Establish data categories.
4. Design and test data collection forms.
5. Collect and validate data.
6. Analyze data.

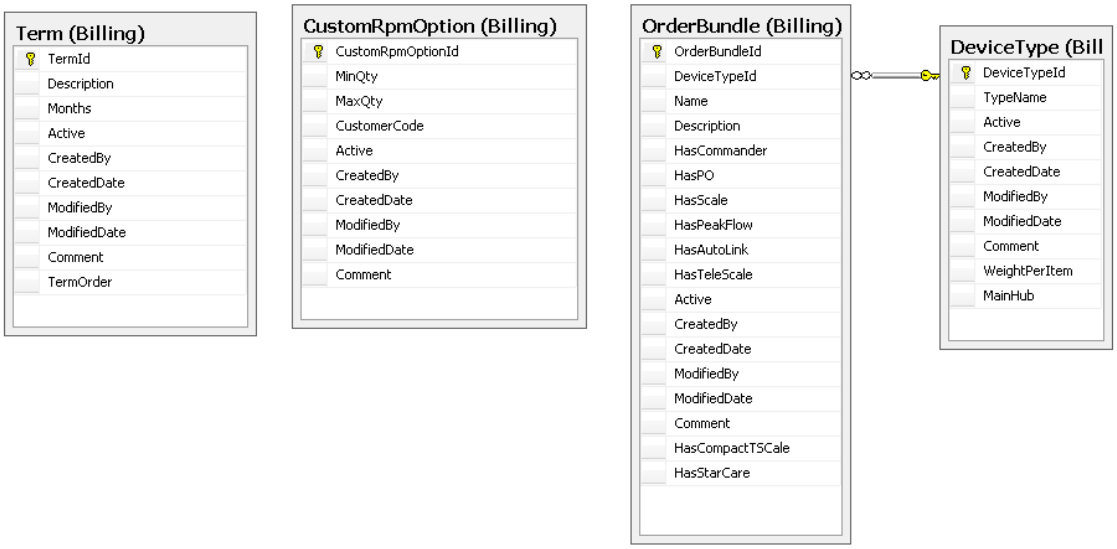
In this project whole data stored, generated and collected has been hosted in the database. In this application a user interface will call the data through the stored procedures which can access the data base and capture the required data only.

In the database the data will be stored in the form of tables and the tables will be in the form rows and columns and this can act as the address for a particular data object. In the table, each column represents some of the attributes and each row represents an individual occurrence of the object.

The data can be retrieved from database by using a stored procedure. A stored procedure is a group of SQL statements that has been created and stored in the database. Stored procedure will accept input parameters so that a single procedure can be used over the network by several clients using different input data. Stored procedure will reduce network traffic and increase the performance. If we modify stored procedure all the clients will get the updated stored procedure (Dasari, 2011).

Every data object will have a unique Id which acts as the primary key for that particular column. And here a duplicate record should not be entered into the table. For example if a record is already existed with an employee ID, a second record with same Id cannot be inserted into the table.

For example, as mentioned in the below Figure 5, the four tables below are mainly used to determine custom set of bundle fees for each customer.



Term (Billing)
TermId
Description
Months
Active
CreatedBy
CreatedDate
ModifiedBy
ModifiedDate
Comment
TermOrder

CustomRpmOption (Billing)
CustomRpmOptionId
MinQty
MaxQty
CustomerCode
Active
CreatedBy
CreatedDate
ModifiedBy
ModifiedDate
Comment

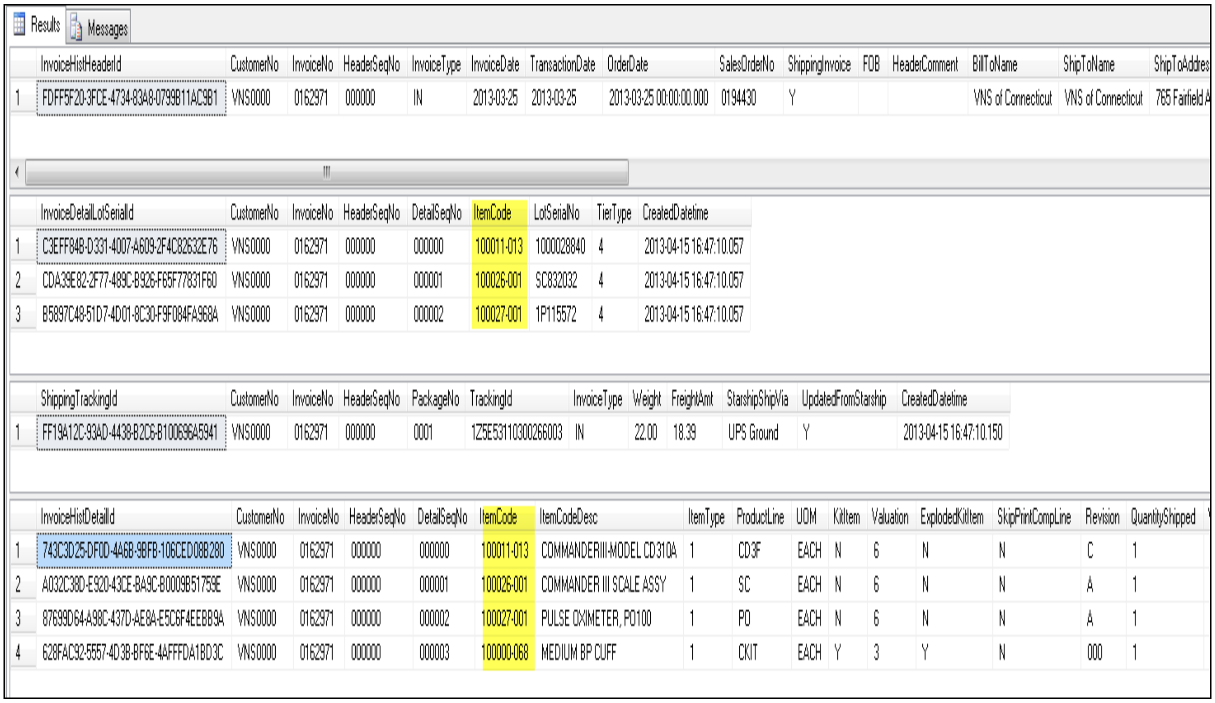
OrderBundle (Billing)
OrderBundleId
DeviceTypeId
Name
Description
HasCommander
HasPO
HasScale
HasPeakFlow
HasAutoLink
HasTeleScale
Active
CreatedBy
CreatedDate
ModifiedBy
ModifiedDate
Comment
HasCompactTSCale
HasStarCare

DeviceType (Bill
DeviceTypeId
TypeName
Active
CreatedBy
CreatedDate
ModifiedBy
ModifiedDate
Comment
WeightPerItem
MainHub

Figure 5: Database Table Structure

Below mentioned is an example of the data in this application which is collected by using a stored procedure.

Table 1: Sample Data Collection and Utilization



	InvoiceHistHeaderId	CustomerNo	InvoiceNo	HeaderSeqNo	InvoiceType	InvoiceDate	TransactionDate	OrderDate	SalesOrderNo	ShippingInvoice	FOB	HeaderComment	BillToName	ShipToName	ShipToAddres
1	FDFF5F20-3FCE-4734-83A8-0799B11AC9B1	VNS0000	0162971	000000	IN	2013-03-25	2013-03-25	2013-03-25 00:00:00.000	0194430	Y			VNS of Connecticut	VNS of Connecticut	765 Fairfield A

	InvoiceDetailLotSerialId	CustomerNo	InvoiceNo	HeaderSeqNo	DetailSeqNo	ItemCode	LotSerialNo	TierType	CreatedDatetime
1	C3EFF84B-D331-4007-A609-2F4C82632E76	VNS0000	0162971	000000	000000	100011-013	1000028840	4	2013-04-15 16:47:10.057
2	CDA39E82-2F77-489C-B926-F65F77831F60	VNS0000	0162971	000000	000001	100026-001	SC832032	4	2013-04-15 16:47:10.057
3	B5897C48-51D7-4D01-8C30-F9F084FA968A	VNS0000	0162971	000000	000002	100027-001	1P115572	4	2013-04-15 16:47:10.057

	ShippingTrackingId	CustomerNo	InvoiceNo	HeaderSeqNo	PackageNo	TrackingId	InvoiceType	Weight	FreightAmt	StarshipShipVia	UpdatedFromStarship	CreatedDatetime
1	FF19A12C-93AD-4438-B2C6-B100696A5941	VNS0000	0162971	000000	0001	1Z5E53110300266003	IN	22.00	18.39	UPS Ground	Y	2013-04-15 16:47:10.150

	InvoiceHistDetailId	CustomerNo	InvoiceNo	HeaderSeqNo	DetailSeqNo	ItemCode	ItemCodeDesc	ItemType	ProductLine	UOM	KitItem	Valuation	ExplodedKitItem	SkipPrintCompLine	Revision	QuantityShipped
1	743C3D25-DF0D-4A6B-9BFB-106CED08B280	VNS0000	0162971	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	1
2	A032C38D-E920-43CE-BA9C-B0009B51759E	VNS0000	0162971	000000	000001	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	1
3	87699D64-A98C-437D-AE8A-E5C6F4EEBB9A	VNS0000	0162971	000000	000002	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	1
4	628FAC92-5557-4D3B-BF6E-4AFFFDA1BD3C	VNS0000	0162971	000000	000003	100000-068	MEDIUM BP CUFF	1	CKIT	EACH	Y	3	Y	N	000	1

Data Analysis

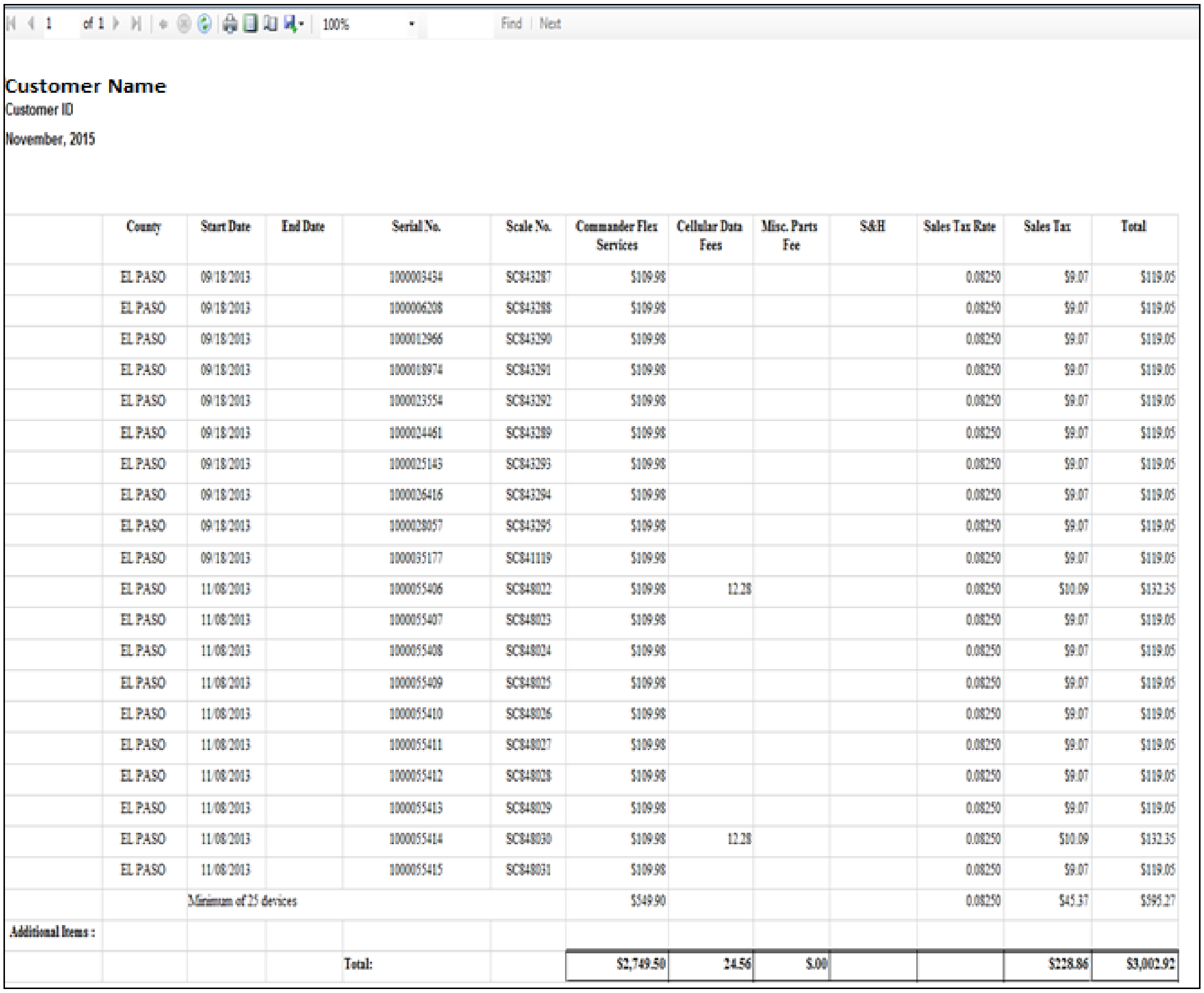
In general, the data analysis can be done in different methods. Analyzing the data involves examining the ways that derives the relationships, patterns and so on which can be found within it. That means subjecting it to statistical operations that can tell you not only what kinds of relationships seem to exist among variables, but also till what extent the output you are getting is correct.

There are two kinds of data which we need to necessarily analyze. Those are Quantitative Data which refer to the information that is collected as, or can be translated into, numbers, which can then be displayed and analyzed mathematically and another is Qualitative data which is are collected as descriptions, anecdotes, opinions, quotes, interpretations and so on and are generally either not able to be reduced to numbers, or are considered more valuable or informative if left as narratives.

In this project data analysis has done in the form of the data reports of different data formats like fee, contracts, customers, shipping charges etc. These are represented in the form of hipping Fee report, Invoice Report, MAS90 alerts report, Orders and Returns Report. These reports are created by the application where user will enter the input values to create the contract for a customer and enter price values depends on the contract in the application. By seeing the end generated report Invoice, users can view the data in the report and this will enable the users to easily understand the data entries are dealing with. To generate the reports RDLC

Below mentioned Table 2 is an example of report generated automatically by the application imported and user provided data.

Table 2: Data Report



Customer Name
Customer ID
November, 2015

	County	Start Date	End Date	Serial No.	Scale No.	Commander Flex Services	Cellular Data Fees	Misc. Parts Fee	S&H	Sales Tax Rate	Sales Tax	Total
	EL PASO	09/18/2013		1000003434	SC843287	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000006208	SC843288	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000012966	SC843290	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000018974	SC843291	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000023554	SC843292	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000024461	SC843289	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000025143	SC843293	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000026416	SC843294	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000028057	SC843295	$109.98				0.08250	$9.07	$119.05
	EL PASO	09/18/2013		1000035177	SC841119	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055406	SC848022	$109.98	12.28			0.08250	$10.09	$132.35
	EL PASO	11/08/2013		1000055407	SC848023	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055408	SC848024	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055409	SC848025	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055410	SC848026	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055411	SC848027	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055412	SC848028	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055413	SC848029	$109.98				0.08250	$9.07	$119.05
	EL PASO	11/08/2013		1000055414	SC848030	$109.98	12.28			0.08250	$10.09	$132.35
	EL PASO	11/08/2013		1000055415	SC848031	$109.98				0.08250	$9.07	$119.05
		Minimum of 25 devices				$549.90				0.08250	$45.37	$595.27
Additional Items :												
				Total:		$2,749.50	24.56	$.00			$228.86	$3,002.92

Project Timeline

The total timeline for the project was 9 months in which 6 months for development of the project and 2 to 3 months for support. The timelines for each individual phase is shown in the below table. The initial phase was the requirements phase where we get the inputs from the client. Then the functional/technical design, development, testing and deployment phase followed. Once the project is deployed, one month was allocated for maintenance and defect identification.

Table3: Project Timeline

Initial requirement meeting	July 2015
Functional/Technical Specification Design	July 2015
Development and Testing	July 2015-November 2015
Proposal	November 2015
Integration	November 2015
Pre-Production	December 2015
Go Production	January 2015
Post-Production Support	January 2015
Defense	February 2015

Summary

This chapter functions as the backbone for the complete project, explaining the users/readers in a very elaborated manner about the process flow, the framework for this project, the tools and techniques used for analysis of the project and timeline of the project. Next chapters will be focused towards showing the collected data and doing a wide analysis on it.

Chapter IV: Data Presentation and Analysis

Introduction

Chapter III mainly described regarding the concepts which serve as the background for the project. This chapter mainly describes presenting the data that was collected and also analyzing the collected data. And it also focuses on interpretation behind these analyses and results.

Data Presentation

This project application needs error-free data to perform accurate Invoice fee calculations to generate the Invoice Report to a customer. All input data will be provided in different formats like files, other sources and user will enter some data as an input through the application. The data files will be generated by using the internal applications and other business processes which will be referred to in this document as the data services.

The data services will include a set of data files which contains information originated in multiple systems, MAS90 application, OPRO and other applications. As mentioned in the below Figure 6, as per the data flow diagram this application is importing the data from different sources and by using stored procedure data will be loading into the Billing database. And then process the invoice and it will generate the invoice for a customer.

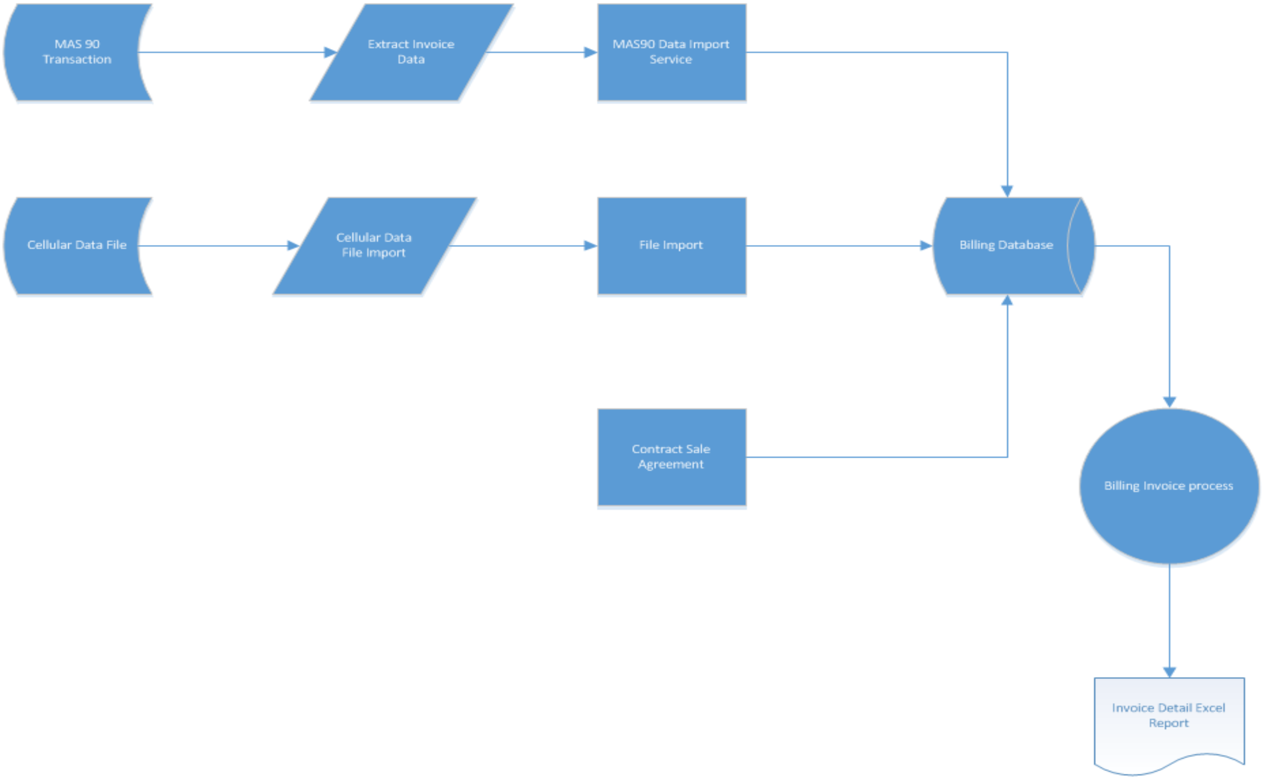


Figure 6: The Data Flow of the Application

As mentioned in Figure 6 above, customer's data will be pulled from MAS90 database and Cellular data will be received in the form of flat file from IT Support team which is imported from OPRO database.

Now the data will be stored in the form of tables and the table consists of rows and columns which can act as the address for a particular data object.

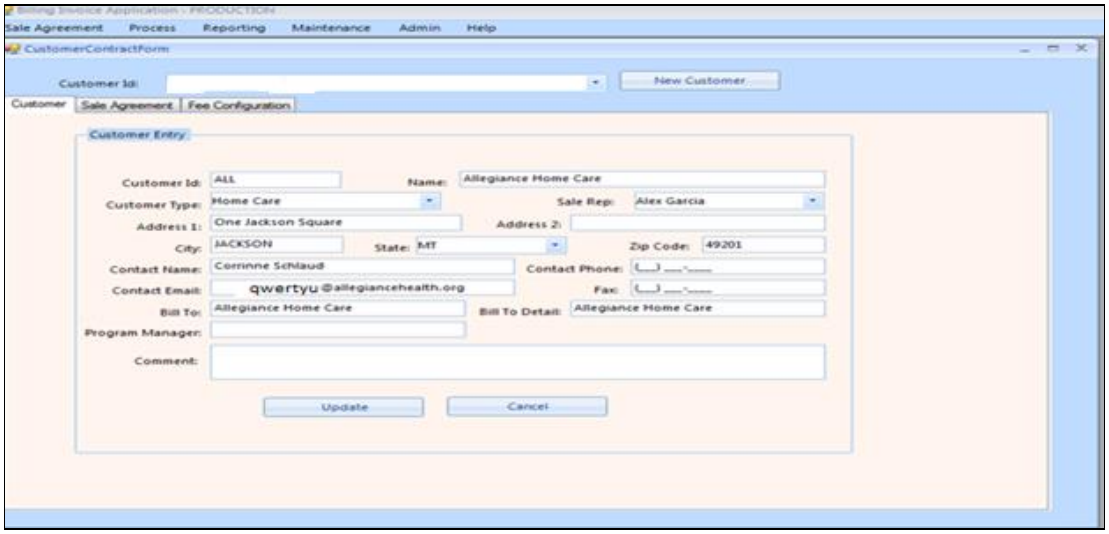
Apart from importing data from different servers and converting them into reports by stored procedures, application receives data from user by using UI pages. In this application, used different technologies and tools which are mentioned below used to create UI pages, passing the data like storing, updating in between front end screen and the database.

- * Telerik DevCraft's Desktop UI Controls
- * C# language and the .NET Framework

Data Analysis

As mentioned above, this fee configuration module is mainly developed to calculate the fee based different scenarios like transaction type, bundle devices, transaction date etc. before generating an invoice to the customer as shown below.

First of all the user will create the customer's contract page by entering the input values in the application as mentioned in Figure 7.



Customer Id:

Customer Id: ALL
Name: Allegiance Home Care
Customer Type: Home Care
Sale Rep: Alex Garcia
Address 1: One Jackson Square
Address 2:
City: JACKSON
State: MT
Zip Code: 49201
Contact Name: Corinne Schlaud
Contact Phone: (__) ___-____
Contact Email: qwertyu @allegiancehealth.org
Fax: (__) ___-____
Bill To: Allegiance Home Care
Bill To Detail: Allegiance Home Care
Program Manager:
Comment:

Figure 7: Customer's Entry Screen

And then user will create the Sale Agreement for a customer by entering the input values as mentioned in Figure 8 below.

Field Definition of Figure 8:

Contract Type: Standard and Custom.

Term: in months

Start Date: is a date when contract is started but not necessary as same as Billing Start Date.

Billing Start Date: is a date used to calculate prorate fees for first invoice

Ramp-up Days: is ramp-up period from contract start-date. This field is used to determine when RPM minimum Fee charge will be applied.

End Date: is a date when a customer ends contract earlier than the contract Term Date.

Term Date: is a date meets its term expiration date.

Minimum RPM: is used to determine minimum service fee charged if number of main devices ordered lower than this value.

Prepaid months: number of months will be applied to each main-hub device ordered.

Disenrollment %: percentage of RPM total that each customer is allowed.

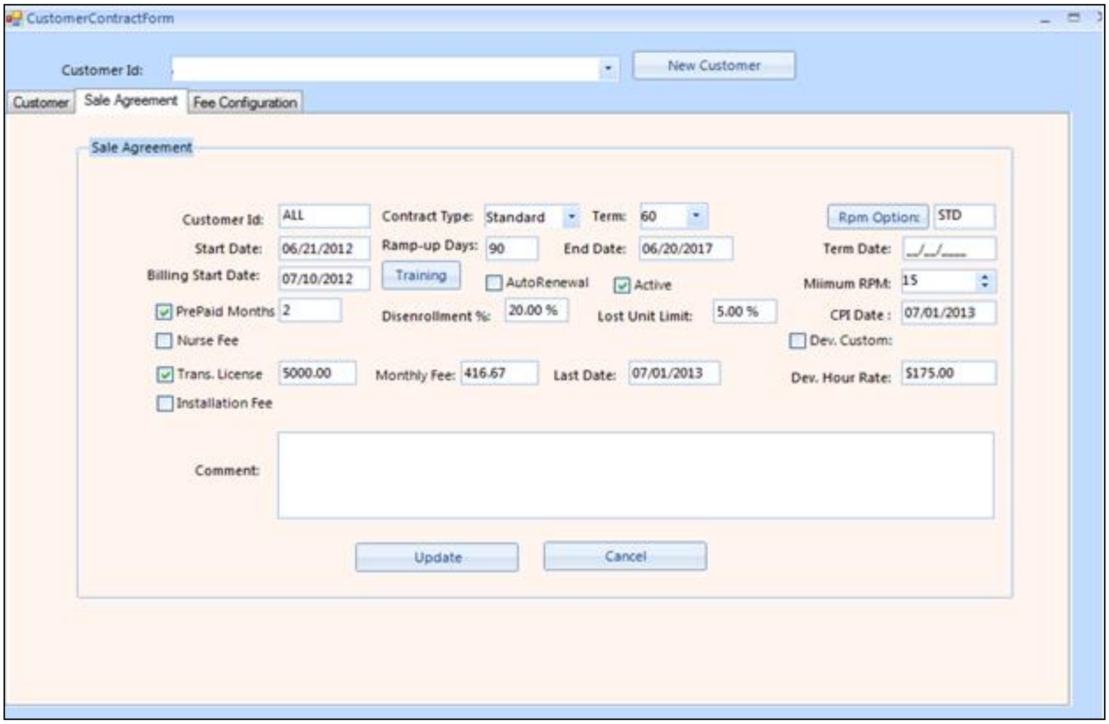
Lost Unit limit %: is used to calculate to determine number of RPM that each customer is allowed not to be charged.

CPI Date: is a date that CPI rate will be applied to specific types of fee listed in CPI filter.

Nurse fee: if applied.

Transmission license fee: one-time fee will be applied if Monthly fee and Last Date are not specified otherwise, monthly fee will be applied until its Last Date is met.

Development Hour Rate: is a rate to be charged if applied (not currently being used.) Installation Fee: one-time fee and works as same as Transmission fee if monthly fee is specified.



Customer Id: ALL
Contract Type: Standard
Term: 60
Rpm Option: STD
Start Date: 06/21/2012
Ramp-up Days: 90
End Date: 06/20/2017
Term Date: __/__/____
Billing Start Date: 07/10/2012
□ AutoRenewal
☑ Active
Miimum RPM: 15
☑ PrePaid Months 2
Disenrollment %: 20.00 %
Lost Unit Limit: 5.00 %
CPI Date : 07/01/2013
□ Nurse Fee
□ Dev. Custom:
☑ Trans. License 5000.00
Monthly Fee: 416.67
Last Date: 07/01/2013
Dev. Hour Rate: $175.00
□ Installation Fee
Comment:

Figure 8: Sale Agreement Screen

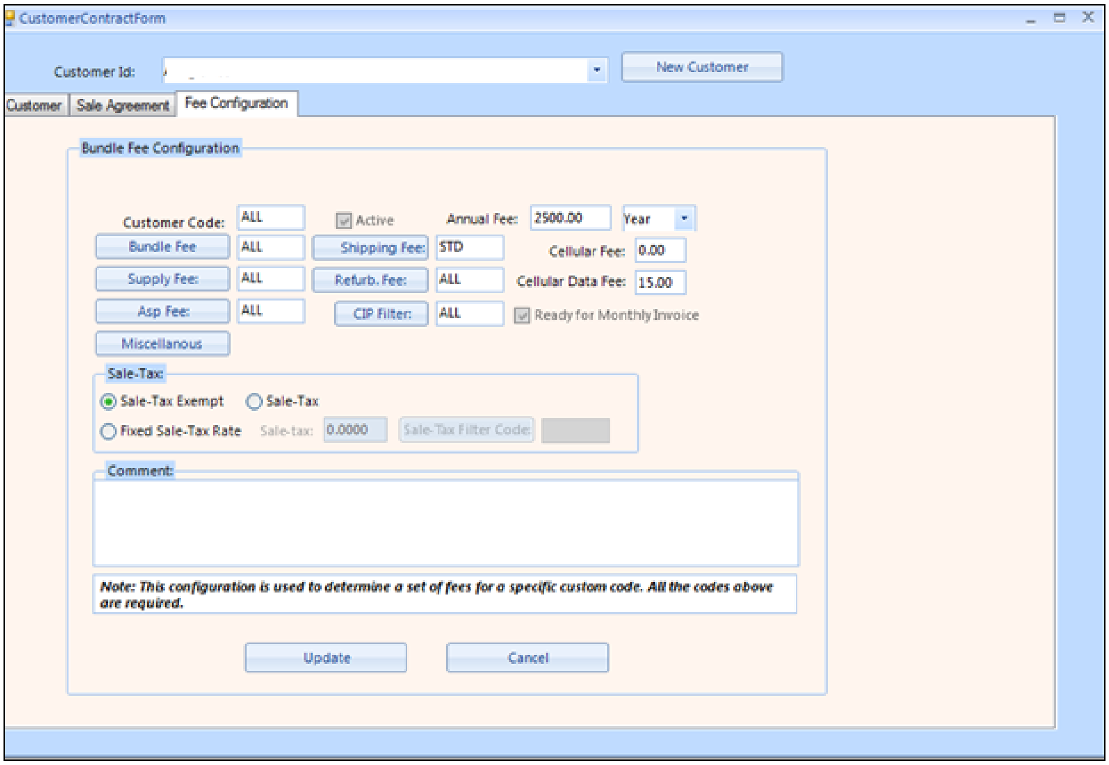
Fee Configuration Module: As mentioned above, by using the input which is received from different sources, developed a Fee Configuration module which is used to calculate the fee for a customer based on different scenarios. From RPM Option from the Sale Agreement screen gives the device count for a customer and depending upon the count, price tier will be determined. It is very important to set a custom code for each Fee Category. It recommends to use the customer code. Since no CPI adjustment is applied to Shipping Fee and most customers have the same shipping fee, its fine to use the ‘STD’ shipping fee bucket list. However, all the other fees such as Bundle Fee, Supply Fee, Asp Fee, Refurbishing fee should have its own set of fees per customer.

Why each customer should has its own custom set of fees for its own contract? Because fees will be increased or changed due to CPI adjustment yearly plus contract amendment if occurred. Customer can also have its own shipping fees other than Standard shipping fees if needed you can create and customize fees if applied (based on the Sale Agreement contract).

CPI filter: will be used to determine which fee will be applied in CPI adjustment process and CPI date is specified in the Sale Agreement Tab.

Sale-Tax option: will allow to specify if a customer has a fixed, Exempt or Sale-tax. If a customer is a Sale-tax customer, sale-tax rate will be determine d in the Invoice process based on the customer shipping address information. The system is currently using the Amazon Sale-tax service to get rate on the fly.

Below mentioned Figure 9 is the Fee Configuration UI page created to manage all different fees for each customer to generate the Invoice Report.



Customer Id:

Customer Code: ALL
☑ Active
Annual Fee: 2500.00 Year
Bundle Fee ALL
Shipping Fee: STD
Cellular Fee: 0.00
Supply Fee: ALL
Refurb. Fee: ALL
Cellular Data Fee: 15.00
Asp Fee: ALL
CIP Filter: ALL
☑ Ready for Monthly Invoice

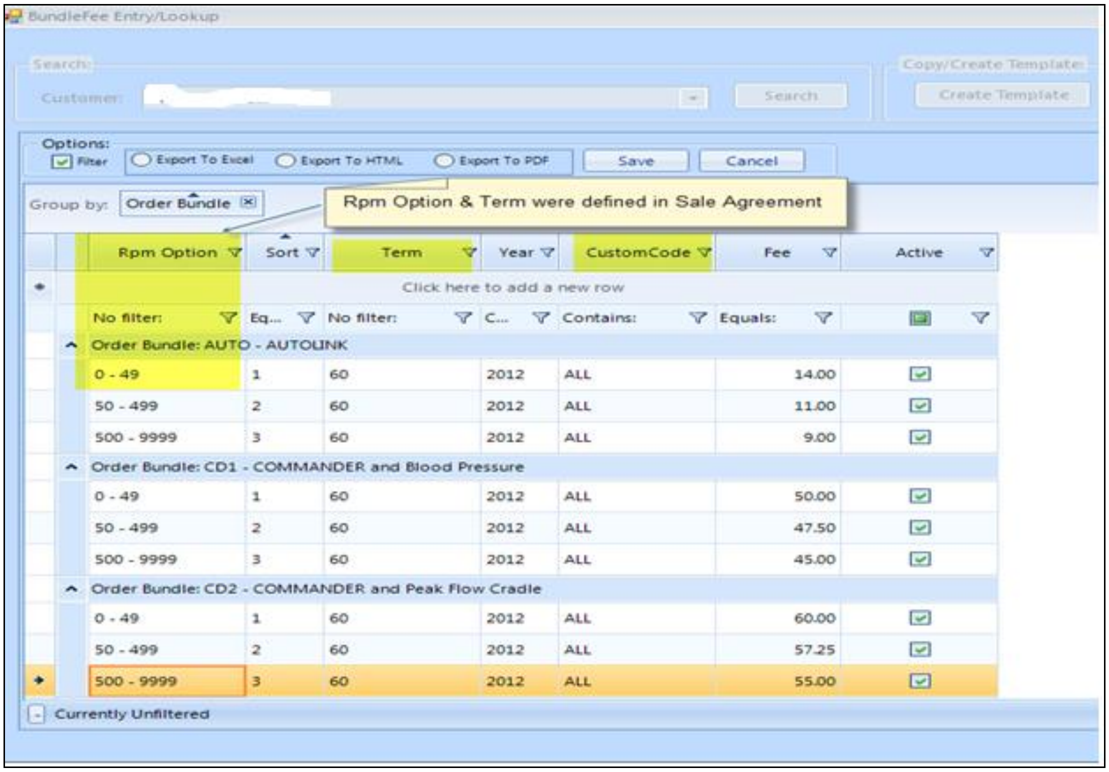
Sale-Tax:
● Sale-Tax Exempt ○ Sale-Tax
○ Fixed Sale-Tax Rate Sale-tax: 0.0000 Sale-Tax Filter Code:

Note: This configuration is used to determine a set of fees for a specific custom code. All the codes above are required.

Figure 9: Fee Configuration UI Screen

Bundle Fee/ Bundle Service Fee: Below mentioned Figure 10 is the Bundle Fee Entry Form, RPM Device Count which is derived from the Sale Agreement screen is checked against this screen. Depends on the device count and bundle type, the Bundle Service Fee will determined for a customer based on the transaction type. And RPM Option list varies from customer to customer as well as Term value. These values can be entered in Sale Agreement Screen and must be created and saved before going to Fee Configuration.

For example, as per Figure 10, the customer Rpm option is 0-49, 50-499, and 500-9999 which is a standard Rpm option bucket list; however, you can specify any options based on the Sale Agreement Contract. Term value can be 1, 3, 5 years and so on.



Customer: [redacted]

Group by: Order Bundle

Rpm Option	Sort	Term	Year	CustomCode	Fee	Active
No filter:	Eq...	No filter:	C...	Contains:	Equals:	
Order Bundle: AUTO - AUTOLINK						
0 - 49	1	60	2012	ALL	14.00	☑
50 - 499	2	60	2012	ALL	11.00	☑
500 - 9999	3	60	2012	ALL	9.00	☑
Order Bundle: CD1 - COMMANDER and Blood Pressure						
0 - 49	1	60	2012	ALL	50.00	☑
50 - 499	2	60	2012	ALL	47.50	☑
500 - 9999	3	60	2012	ALL	45.00	☑
Order Bundle: CD2 - COMMANDER and Peak Flow Cradle						
0 - 49	1	60	2012	ALL	60.00	☑
50 - 499	2	60	2012	ALL	57.25	☑
500 - 9999	3	60	2012	ALL	55.00	☑

Figure 10: Bundle Fee Entry Form

The bundle item will be created and its fee will be calculated on the fly in the Billing Invoice process. The fee is set by different tables like Orderbundle, CustomRpmOption, Term, and CustomCode values for appropriate active fee value and the fee will be determined as mentioned below.

New order: the fee will be prorated using invoice order transaction date to billing month-end date.

Existing order: The fee will be calculated as monthly fee.

Returned order: System will try to bundle all returned items into a bundle for each Customer No that has return items in the same invoice month. Bundle Service fee will be prorated using return shipping date (invoice date, invoice CM type). If the returned shipping date is prior to the previous month-end date, credit will be calculated and applied.

Exchange order: The service fee will be recalculated with appropriate type of hub device to get appropriate fee respectively.

ASP Fee: If the ASP fee is entered in ASP Entry screen as per the contract, need to populate it under ASP Fee column on the Invoice report. No proration is required. ASP fee is applied only for hub device (should not including test, demo, lost and the device have an End Date in previous month).

Supple Fee: Should not charge if the customer ordered supply item with the device.

Shipping Fee: Sum of all shipment fee for the patient/customer of that bill month and no shipping fee for lost, test, demo, transfer and warranty devices.

Refurbishing Fee: Need to charge refurbishing fee for each returned/exchange/replace device based on RefurbishingFeeEntry screen. If the customer/patient is upgrading the hub device (for example upgrading the Commander with Commander Flex), need to charge exchange fee instead of refurbishing fee. When we receive a termination note from the customer, we need to enter termination date in Sale Agreement and in that period should not charge Refurbishing fee

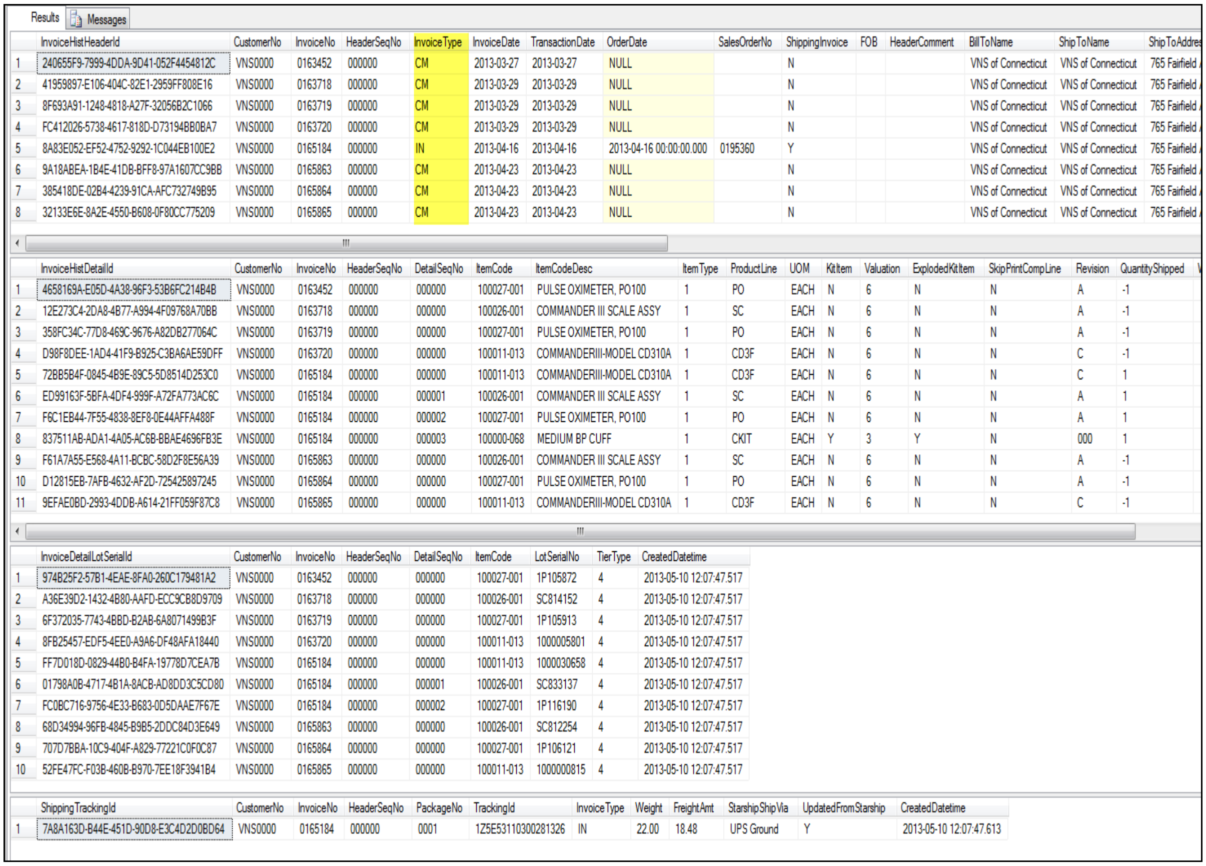
As mentioned above, all types of fee is calculated and then by using the below stored procedure, we can see the whole month transaction for specific customer which is mentioned in Figure 11 and 12.

```
use [Accounting]
go

declare @customerCode varchar(10),
       @StartDate date,
                     @EndDate date,
                     @BillMonth char(6)

set @customerCode = 'VNS'
set @BillMonth = '201304'
set @StartDate = '03/27/2013'
set @EndDate = '04/30/2013'

EXEC Billing.usp_GetMas90InvoiceHeaders @CustomerCode,@startdate,@EndDate
EXEC Billing.usp_GetMas90InvoiceDetails @CustomerCode,@startdate,@EndDate
EXEC Billing.usp_GetMas90InvoiceTrackings @CustomerCode,@startdate,@EndDate
        EXEC Billing.usp_GetMas90InvoiceLotSerials @CustomerCode,@startdate,@EndDate
```



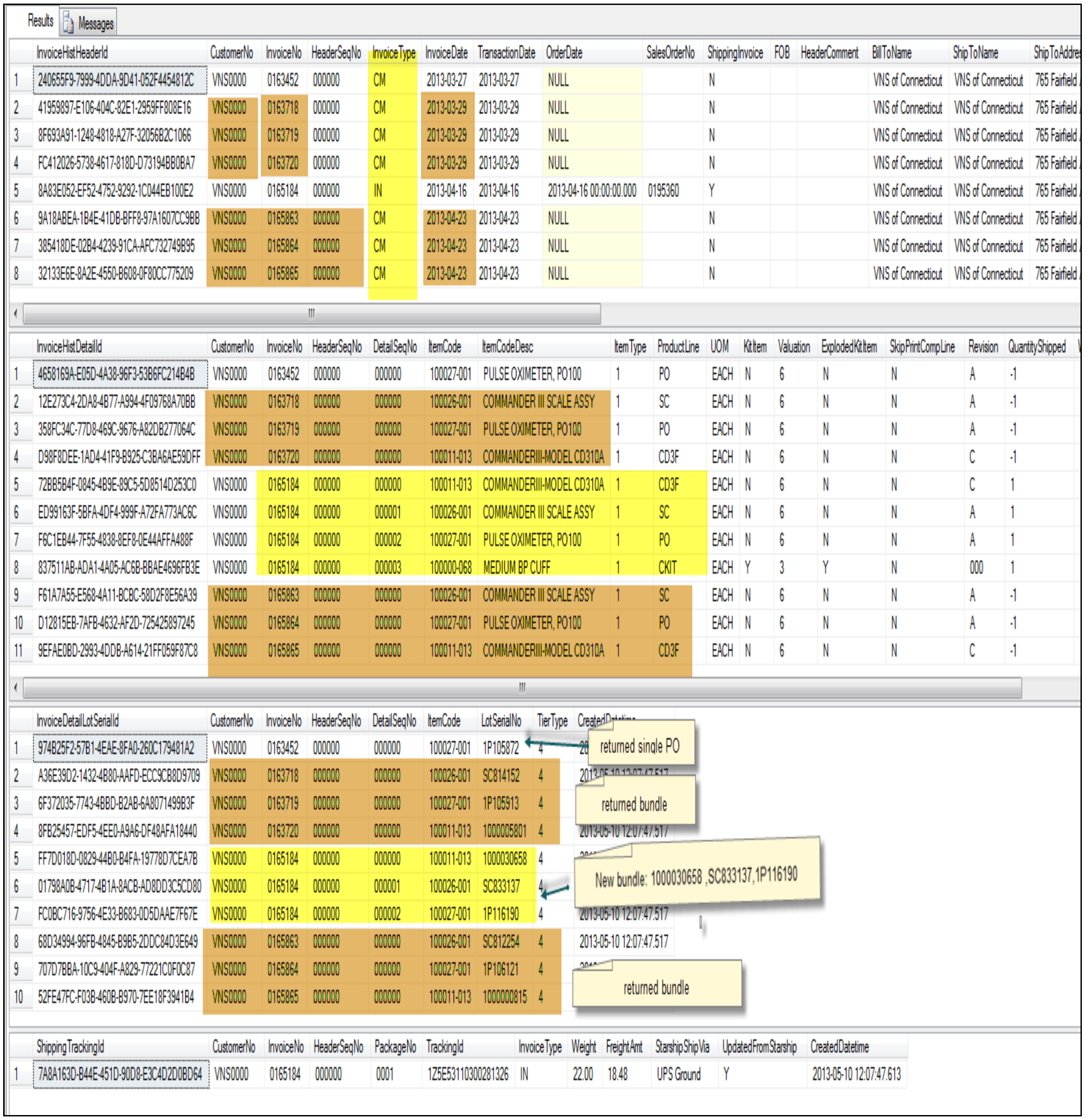
	InvoiceHistHeaderId	CustomerNo	InvoiceNo	HeaderSeqNo	InvoiceType	InvoiceDate	TransactionDate	OrderDate	SalesOrderNo	ShippingInvoice	FOB	HeaderComment	BillToName	ShipToName	ShipToAddres
1	240655F9-7999-4DDA-9D41-052F4454812C	VNS0000	0163452	000000	CM	2013-03-27	2013-03-27	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
2	41959897-E106-404C-82E1-2959FF808E16	VNS0000	0163718	000000	CM	2013-03-29	2013-03-29	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
3	8F693A91-1248-4818-A27F-32056B2C1066	VNS0000	0163719	000000	CM	2013-03-29	2013-03-29	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
4	FC412026-5738-4617-818D-D73194BB0BA7	VNS0000	0163720	000000	CM	2013-03-29	2013-03-29	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
5	8A83E052-EF52-4752-9292-1C044EB100E2	VNS0000	0165184	000000	IN	2013-04-16	2013-04-16	2013-04-16 00:00:00.000	0195360	Y			VNS of Connecticut	VNS of Connecticut	765 Fairfield
6	9A18ABEA-1B4E-41DB-BFF8-97A1607CC9BB	VNS0000	0165863	000000	CM	2013-04-23	2013-04-23	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
7	385418DE-02B4-4239-91CA-AFC732749B95	VNS0000	0165864	000000	CM	2013-04-23	2013-04-23	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
8	32133E6E-8A2E-4550-B608-0F80CC775209	VNS0000	0165865	000000	CM	2013-04-23	2013-04-23	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield

	InvoiceHistDetailId	CustomerNo	InvoiceNo	HeaderSeqNo	DetailSeqNo	ItemCode	ItemCodeDesc	ItemType	ProductLine	UOM	KitItem	Valuation	ExplodedKitItem	SkipPrintCompLine	Revision	QuantityShipped
1	4658169A-E05D-4A38-96F3-53B6FC214B4B	VNS0000	0163452	000000	000000	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	-1
2	12E273C4-2DA8-4B77-A994-4F09768A70BB	VNS0000	0163718	000000	000000	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	-1
3	358FC34C-77D8-469C-9676-A82DB277064C	VNS0000	0163719	000000	000000	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	-1
4	D98F8DEE-1AD4-41F9-B925-C3BA6AE59DFF	VNS0000	0163720	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	-1
5	72BB5B4F-0845-4B9E-89C5-5D8514D253C0	VNS0000	0165184	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	1
6	ED99163F-5BFA-4DF4-999F-A72FA773AC6C	VNS0000	0165184	000000	000001	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	1
7	F6C1EB44-7F55-4838-8EF8-0E44AFFA488F	VNS0000	0165184	000000	000002	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	1
8	837511AB-ADA1-4A05-AC6B-BBAE4696FB3E	VNS0000	0165184	000000	000003	100000-068	MEDIUM BP CUFF	1	CKIT	EACH	Y	3	Y	N	000	1
9	F61A7A55-E568-4A11-BCBC-58D2F8E56A39	VNS0000	0165863	000000	000000	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	-1
10	D12815EB-7AFB-4632-AF2D-725425897245	VNS0000	0165864	000000	000000	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	-1
11	9EFAE0BD-2993-4DDB-A614-21FF059F87C8	VNS0000	0165865	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	-1

	InvoiceDetailLotSerialId	CustomerNo	InvoiceNo	HeaderSeqNo	DetailSeqNo	ItemCode	LotSerialNo	TierType	CreatedDatetime
1	974B25F2-57B1-4EAE-8FA0-260C179481A2	VNS0000	0163452	000000	000000	100027-001	1P105872	4	2013-05-10 12:07:47.517
2	A36E39D2-1432-4B80-AAFD-ECC9CB8D9709	VNS0000	0163718	000000	000000	100026-001	SC814152	4	2013-05-10 12:07:47.517
3	6F372035-7743-4BBD-B2AB-6A8071499B3F	VNS0000	0163719	000000	000000	100027-001	1P105913	4	2013-05-10 12:07:47.517
4	8FB25457-EDF5-4EE0-A9A6-DF48AFA18440	VNS0000	0163720	000000	000000	100011-013	1000005801	4	2013-05-10 12:07:47.517
5	FF7D018D-0829-44B0-B4FA-19778D7CEA7B	VNS0000	0165184	000000	000000	100011-013	1000030658	4	2013-05-10 12:07:47.517
6	01798A0B-4717-4B1A-8ACB-AD8DD3C5CD80	VNS0000	0165184	000000	000001	100026-001	SC833137	4	2013-05-10 12:07:47.517
7	FC0BC716-9756-4E33-B683-0D5DAAE7F67E	VNS0000	0165184	000000	000002	100027-001	1P116190	4	2013-05-10 12:07:47.517
8	68D34994-96FB-4845-B9B5-2DDC84D3E649	VNS0000	0165863	000000	000000	100026-001	SC812254	4	2013-05-10 12:07:47.517
9	707D7BBA-10C9-404F-A829-77221C0F0C87	VNS0000	0165864	000000	000000	100027-001	1P106121	4	2013-05-10 12:07:47.517
10	52FE47FC-F03B-460B-B970-7EE18F3941B4	VNS0000	0165865	000000	000000	100011-013	1000000815	4	2013-05-10 12:07:47.517

	ShippingTrackingId	CustomerNo	InvoiceNo	HeaderSeqNo	PackageNo	TrackingId	InvoiceType	Weight	FreightAmt	StarshipShipVia	UpdatedFromStarship	CreatedDatetime
1	7A8A163D-B44E-451D-90D8-E3C4D2D0BD64	VNS0000	0165184	000000	0001	1Z5E53110300281326	IN	22.00	18.48	UPS Ground	Y	2013-05-10 12:07:47.613

Figure 11: Example of Customers Data



Results | Messages

	InvoiceHistHeaderId	CustomerNo	InvoiceNo	HeaderSeqNo	InvoiceType	InvoiceDate	TransactionDate	OrderDate	SalesOrderNo	ShippingInvoice	FOB	HeaderComment	BillToName	ShipToName	ShipToAddre
1	240655F9-7999-4DDA-9D41-052F4454812C	VNS0000	0163452	000000	CM	2013-03-27	2013-03-27	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
2	41959897-E106-404C-82E1-2959FF808E16	VNS0000	0163718	000000	CM	2013-03-29	2013-03-29	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
3	8F693A91-1248-4818-A27F-32056B2C1066	VNS0000	0163719	000000	CM	2013-03-29	2013-03-29	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
4	FC412026-5738-4617-818D-D73194BB0BA7	VNS0000	0163720	000000	CM	2013-03-29	2013-03-29	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
5	8A83E052-EF52-4752-9292-1C044EB100E2	VNS0000	0165184	000000	IN	2013-04-16	2013-04-16	2013-04-16 00:00:00.000	0195360	Y			VNS of Connecticut	VNS of Connecticut	765 Fairfield
6	9A18ABEA-1B4E-41DB-BFF8-97A1607CC9BB	VNS0000	0165863	000000	CM	2013-04-23	2013-04-23	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
7	385418DE-02B4-4239-91CA-AFC732749B95	VNS0000	0165864	000000	CM	2013-04-23	2013-04-23	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield
8	32133E6E-8A2E-4550-B608-0F80CC775209	VNS0000	0165865	000000	CM	2013-04-23	2013-04-23	NULL		N			VNS of Connecticut	VNS of Connecticut	765 Fairfield

	InvoiceHistDetailId	CustomerNo	InvoiceNo	HeaderSeqNo	DetailSeqNo	ItemCode	ItemCodeDesc	ItemType	ProductLine	UOM	KitItem	Valuation	ExplodedKitItem	SkipPrintCompLine	Revision	QuantityShipped
1	4658169A-E05D-4A38-96F3-53B6FC214B4B	VNS0000	0163452	000000	000000	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	-1
2	12E273C4-2DA8-4B77-A994-4F09768A70BB	VNS0000	0163718	000000	000000	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	-1
3	358FC34C-77D8-469C-9676-A82DB277064C	VNS0000	0163719	000000	000000	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	-1
4	D98F8DEE-1AD4-41F9-B925-C3BA6AE59DFF	VNS0000	0163720	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	-1
5	72BB5B4F-0845-4B9E-89C5-5D8514D253C0	VNS0000	0165184	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	1
6	ED99163F-5BFA-4DF4-999F-A72FA773AC6C	VNS0000	0165184	000000	000001	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	1
7	F6C1EB44-7F55-4838-8EF8-0E44AFFA488F	VNS0000	0165184	000000	000002	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	1
8	837511AB-ADA1-4A05-AC6B-BBAE4696FB3E	VNS0000	0165184	000000	000003	100000-068	MEDIUM BP CUFF	1	CKIT	EACH	Y	3	Y	N	000	1
9	F61A7A55-E568-4A11-BCBC-58D2F8E56A39	VNS0000	0165863	000000	000000	100026-001	COMMANDER III SCALE ASSY	1	SC	EACH	N	6	N	N	A	-1
10	D12815EB-7AFB-4632-AF2D-725425897245	VNS0000	0165864	000000	000000	100027-001	PULSE OXIMETER, PO100	1	PO	EACH	N	6	N	N	A	-1
11	9EFAE0BD-2993-4DDB-A614-21FF059F87C8	VNS0000	0165865	000000	000000	100011-013	COMMANDERIII-MODEL CD310A	1	CD3F	EACH	N	6	N	N	C	-1

	InvoiceDetailLotSerialId	CustomerNo	InvoiceNo	HeaderSeqNo	DetailSeqNo	ItemCode	LotSerialNo	TierType	CreatedD
1	974B25F2-57B1-4EAE-8FA0-260C179481A2	VNS0000	0163452	000000	000000	100027-001	1P105872	4	20
2	A36E39D2-1432-4B80-AAFD-ECC9CB8D9709	VNS0000	0163718	000000	000000	100026-001	SC814152	4	2013-05-10 12:07:47.517
3	6F372035-7743-4BBD-B2AB-6A8071499B3F	VNS0000	0163719	000000	000000	100027-001	1P105913	4	
4	8FB25457-EDF5-4EE0-A9A6-DF48AFA18440	VNS0000	0163720	000000	000000	100011-013	1000005801	4	2013-05-10 12:07:47.517
5	FF7D018D-0829-44B0-B4FA-19778D7CEA7B	VNS0000	0165184	000000	000000	100011-013	1000030658	4	201
6	01798A0B-4717-4B1A-8ACB-AD8DD3C5CD80	VNS0000	0165184	000000	000001	100026-001	SC833137	4	
7	FC0BC716-9756-4E33-B683-0D5DAAE7F67E	VNS0000	0165184	000000	000002	100027-001	1P116190	4	2013-05-10 12:07:47.517
8	68D34994-96FB-4845-B9B5-2DDC84D3E649	VNS0000	0165863	000000	000000	100026-001	SC812254	4	2013-05-10 12:07:47.517
9	707D7BBA-10C9-404F-A829-77221C0F0C87	VNS0000	0165864	000000	000000	100027-001	1P106121	4	201
10	52FE47FC-F03B-460B-B970-7EE18F3941B4	VNS0000	0165865	000000	000000	100011-013	1000000815	4	

	ShippingTrackingId	CustomerNo	InvoiceNo	HeaderSeqNo	PackageNo	TrackingId	InvoiceType	Weight	FreightAmt	StarshipShipVia	UpdatedFromStarship	CreatedDatetime
1	7A8A163D-B44E-451D-90D8-E3C4D2D0BD64	VNS0000	0165184	000000	0001	1Z5E53110300281326	IN	22.00	18.48	UPS Ground	Y	2013-05-10 12:07:47.613

Figure 12: Example of Customers Data

As per the above example, the customer had 2 returned bundles, 1 returned PO, and 1 New order of bundle.

Summary

Chapter IV is the successive important chapter which followed chapter III. Data presentation described how the data is imported from different sources and explains the actual data that is collected to evaluate the project. And it also described how UI pages are created and how the collected data is used to calculate the fee with an example. The next chapter will cover the result of the project, conclusions based on the results and possible recommendations for the betterment of the organization.

Chapter V: Results, Conclusion, and Recommendations

Introduction

Chapter IV is mainly focused on presenting the data which is collected and analyzing the collected data. This chapter will focuses on the final results of the project with conclusion from the result and also conceivable recommendations based on the results and conclusion.

Results

The automation of Fee Configuration is the important module of the Billing Invoice Application which has been successfully achieved. Based on the implementation of this module, manual interaction with the application has reduced tremendously which shows a great results that have been achieved by the final output of the project. Well documented the details of process which is very useful for everybody.

The main objective of the project is to provide the functionality to automate the fee calculation and also track the data by generating different reports. Along with this, it also provides the bundling process for the devices which were ordered by customer based on the transaction type. This project also has a functionality to record all the changes that a user makes along with the date and time of the change. This project not only provides additional security to the data that’s being handled but also provides an opportunity to revert back any changes that might have caused a mistake in the system.

Project Questions–Answered:

1. What percentage of the usage of manual work would be reduced after implementing the project?

As discussed above, this Fee Configuration module is mainly automated to calculate the fee depends on the different transactions type and bundle type before generating the invoice of a customer. From this automated application, it can be concluded that around 90% of the manual operations can reduced. And the remaining 10% of the manual efforts are required whenever the user wants to generate reports for particular month, need to import the data accordingly. And also the customer entry creation, Sale agreement creation and fee allocation should be entered manually for a customer.

2. Which additional tasks can be achieved using this project other than fee configuration devices?

Apart from fee configuration, bundle the devices based on the bundle matrix, device type as per the bundle matrix and transaction type. Price schedule is created depends on the device type. Created a report which shows list of issues occurred because of multiple reasons before loading the data into the billing system, so that this report can alert user to check whether the data is valid or not.

3. How stable is the automated project environment when compared to the manual work and how much accuracy of results can be expected from the automated process?

As discussed above, manual work is reduced 90% and after the implementation of the application into production. Generated the Invoice reports, shipping reports, etc., with the automated new system and reports generated manually. And then compared those every day for some days. Saw some manual calculation error compared to automated system.

Continuing this process for some days and decided that the accuracy and stability of the system is 100% except in some scenarios.

Conclusion

Automation of this module reduced the manual work by 90% and increased the accuracy of the system. And this project has been considered as one of the most important projects handled by the organization and appreciated by all higher lever management and customers. Along with the reduction in manual work, a few other functionalities which are very useful for the application are implemented. The reduction in the chances of manual errors were also one of the key aspects that was achieved in this project. Both these aspects directly shows the reduction in the utilization of the resources by the organization.

Based on the final results of the project, it has been a user friendly, stable and a flexible outcome.

Recommendations

1. Some of the data is imported in the form flat files which are produced by IT supporting team. Instead of this, direct access to the other database will reduce more manual work for other teams also. And the automated scheduled job will take care of the data extraction from other servers and storing into the Billing database.
2. Making changes to the project in the initial planning stage will be more beneficial in designing the product perfectly rather than making adjustments in the later stages of the project development.

References

Balas, O. (2015, October 8). *How to consume a WebAPI with RestSharp*. Retrieved from https://visualstudiomagazine.com/articles/2015/10/01/consume-a-webapi.aspx.

Basili, V., & Weiss, D. (1984, November). A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, *10*(3), 728-738.

Dasari, S. (2011, July 28). What is stored procedure in SQL server | what are the advantages of using stored procedures in SQL server. Retrieved from http://www.aspdotnet-suresh.com/2011/07/what-is-stored-procedure-in-sql-server.html.

Génova, G., Fuentes, J. M., Llorens, J., Hurtado, O., & Moreno, V. (2013, March). A framework to measure and improve the quality of textual requirements. *Requirements Engineering*, *18*(1), 25-41.

Kan, S. H. (2002). *Metrics & models in software quality engineering*. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Mohsin, R. J. (2013). *Design and implementation of web application using .Net framework.* Retrieved from https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx.

Ramel, D. (2015, February 25). *Telerik DevCraft update features new cross-platform controls*. Retrieved from https://visualstudiomagazine.com/articles/2015/02/25/telerik-devcraft-update.aspx.

Ross, M., & Stacia, M. (2012). *Introducing Microsoft SQL Server 2012*. Retrieved from http://www.functionx.com/sqlserver/.

Telerick DevCraft's Desktop UI Controls. (n.d.) Retrieved from http://www.telerik.com/devcraft.

Waterfall model. (2011). Retrieved from https://www.google.com/search?q=Waterfall+Model+2011&biw=1600&bih=799&tbm=isch&tbo=u&source=univ&sa=X&ved=0ahUKEwiPxOH77qXLAhXKKh4KHVzPD_cQsAQIGw.

Westland, J. (2006). *The project management lifecycle*. Kogan Page, Ltd.

What is SQL server? Retrieved from http://whatis.techtarget.com/definition/SQL-Server-2012.

Yaraamjad. (2006). *Different phases of software development lifecycle*. Retrieved from https://www.google.com/search?q=software+development+life+cycle+phases&sa=X&biw=1600&bih=799&tbm=isch&tbo=u&source=univ&ved=0ahUKEwjpgu3u8aXLAhVCGB4KHfSXCDQQsAQILw.