To Analyze the Data and Intent of the User

And

Purpose of Online Communication

Project Report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology.

in

Computer Science & Engineering

under the Supervision of

Prof. Ruchi Verma

By

Mr. Manu Raghuvanshi(111216)

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled "To Analyze the Data and Intent of the User And Purpose of Online Communication", submitted by Manu Raghuvanshi in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: **Supervisor's Name**

Prof. Ruchi Verma

Acknowledgement

It is my privilege to express my sincere regards to my project coordinator, Prof. Ruchi Verma, for her valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of my project.

I deeply express my sincere thanks to the Head of Department **Prof. Dr. S. P. Ghrera** for encouraging and allowing me to present the project on the topic "To Analyze the Data and Intent of the User And Purpose of Online Communication " at our department premises for the partial fulfillment of the requirements leading to the award of B.Tech degree.

I take this opportunity to thank all my lecturers who have directly or indirectly helped in my project. I pay my respect and love to my parents and all other family members for their love and encouragement through out my career. Last but not the least I express my thanks to my friends for their cooperation and support.

Date:

Name of the student

Manu Raghuvanshi
111216

Table of Content

ABSTRACT

From hundreds to thousands of computers, hubs to switched networks, and Ethernet to either ATM or 10Gbps Ethernet, administrators need more sophisticated network traffic monitoring and analysis tools in order to deal with the increase. These tools are needed, not only to fix network problems on time, but also to prevent network failure, to detect inside and outside threats, and make good decisions for network planning.

Computer software that can intercept and log traffic passing over a digital network or part of a network is better known as packet sniffer. The sniffer captures these packets by setting the NIC card in the promiscuous mode and eventually decodes them. The decoded information can be used in any way depending upon the intention of the person concerned who decodes the data (i.e. malicious or beneficial purpose). Depending on the network structure one can sniff all or just parts of the traffic from a single machine within the network.

The term "online communication" refers to reading, writing, and communication via networked computers. It encompasses synchronous computer-mediated communication (whereby people communicate in real time via chat or discussion software, with all participants at their computers at the same time);asynchronous computer-mediated communication (whereby people communicate in a delayed fashion by computer, using programs such as e-mail); and the reading and writing of online documents via the World Wide Web.

1. INTRODUCTION

INTRODUCTION

Communication is a purposeful activity of exchanging information and meaning across space and time using various technical or natural means, whichever is available or preferred.

Communication requires a sender, a message, a medium and a recipient, although the receiver does not have to be present or aware of the sender's intent to communicate at the time of communication; thus communication can occur across vast distances in time and space. Communication requires that the communicating parties share an area of communicative commonality. The communication process is complete once the receiver understands the sender's message.

1.1 What Is Online Communication?

The term "online communication" refers to reading, writing, and communication via networked computers. It encompasses ***synchronous*** computer-mediated communication (whereby people communicate in real time via chat or discussion software, with all participants at their computers at the same time);***asynchronous*** computer-mediated communication (whereby people communicate in a delayed fashion by computer, using programs such as e-mail); and the reading and writing of online documents via the World Wide Web. Second language researchers are interested in two overlapping issues related to online communication: (1) how do the processes which occur in online communication assist language learning in a general sense (i.e., online communication for language learning); and (2) what kinds of language learning need to occur so that people can communicate effectively in the online realm (i.e., language learning for online communication).

1.2 Background

Online communication dates back to late 1960s, when U.S. researchers first developed protocols that allowed the sending and receiving of messages via computer. The ARPANET, launched in 1969 by a handful of research scientists, eventually evolved into the Internet, bringing together some 200 million people around the world at the turn of the millennium.

Online communication first became possible in educational realms in the 1980s, following the development and spread of personal computers. The background on online communication in language teaching and research can be divided into two distinct periods, marked by the introduction of computer-mediated communication in education in the mid-1980s and the emergence of the World Wide Web in the mid-1990s.

1.3 Computer-Mediated Communication

In the first period, dating from the mid-1980s, language educators began to discover the potential of computer-mediated communication for language teaching. The integration of computer-mediated communication in the classroom itself divided into two paths: on the one hand, some educators began to use e-mail to set up *long-distance exchanges*, and, on the other hand, other educators began to use synchronous software programs to allow *computer-assisted conversation* in a single classroom.

Long-distance exchanges and computer-assisted conversation had overlapping, but distinctive, justifications. Both types of activities were seen to shift the focus from language form to language use in meaningful context and thereby increase student motivation . In addition, long-distance exchanges were viewed as brining about increased cultural knowledge from communication with native-speaking informants, and making reading and writing more authentic and collaborative. Those implementing computer-assisted conversation emphasized the linguistic benefits which could be achieved from rapid written interaction, such as better opportunities to process and try out new lexical or syntactic patterns as compared to oral interaction .

1.4 The World Wide Web

The World Wide Web is an international online database that allows the sharing of linked multimedia documents. These documents can be authored in a non-linear, layered and linked format, which is referred to as hypertext or hypermedia. The development and spread of the World Wide Web in the 1990s marked a second period in the use of online communication in language teaching. One the one hand, the Web allows additional modes of computer-mediated communication through Web-based chat rooms, bulletin boards, and discussion forums, thus making even more popular the kind of long-distance exchanges and computer-assisted conversation activities described above. In addition, the World Wide Web adds a new dimension to online communication

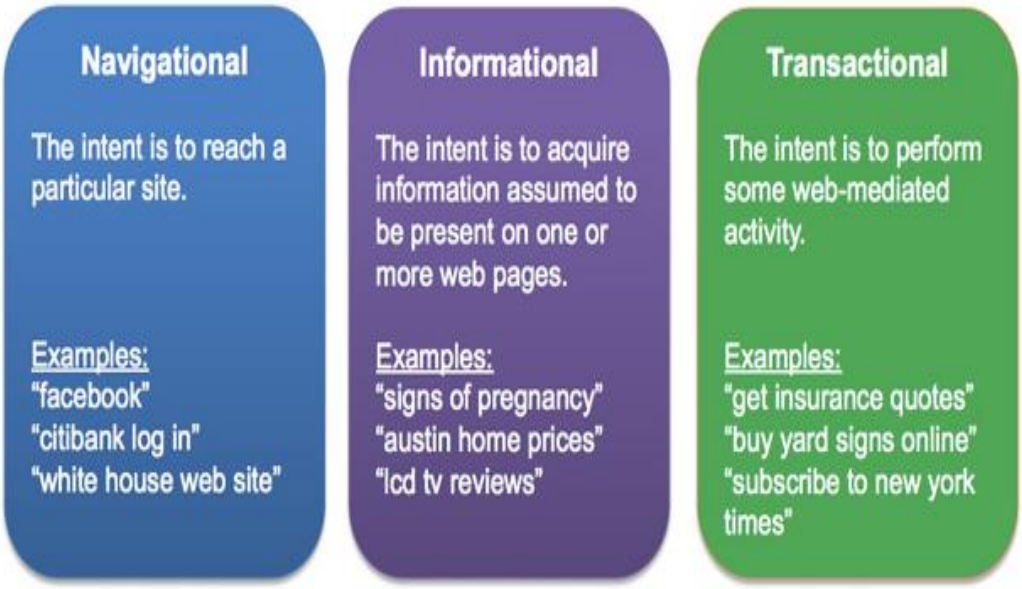
and learning by allowing students to find and read online documents on a variety of topics from throughout the world and to author and publish similar documents to share with others.

Some researchers have viewed the Web as an extension of an L2 culture or society; by engaging in Web-based activities, students can gradually become members of the community of English language speakers, in the same way that they might through other forms of immersion in a culture (Zhao, 1996). Others view the Web as an extension of a CD-ROM, in other words, a good environment to create multimedia language learning materials with the added advantage of allowing student interactivity. Others view the Web as an extension of (and alternative to) print, that is, a major new medium of literacy that needs to be mastered on its own terms for success in 21st century life. Since the Web is a vast and diverse environment, encompassing a huge variety of online documents and an array of evolving communications tools, it is perhaps overreaching to seek a single unitary framework to motivate its integration in the classroom.

1.5 Practice and Purpose

The Internet is by its nature a dynamic and interactive medium that requires a high degree of flexibility and interaction. At the same time, the highly decentralized and diverse nature of the Internet can make it a confusing and even chaotic medium for learners of, especially those at the beginning level. Simply leaving learners to their own resources on the Internet is unlikely to bring satisfying results, as beginning learners drop out in frustration and more advanced learners stagnate at the level of conversational chatting or superficial "net-surfing."

Searcher intent falls into three major categories: navigational, informational, and transactional.



1.6 About Packet Based Network Sniffer:

- A **"Packet Sniffer"** is a utility that **sniffs without modifying** the network's packets in any way.
- A firewall sees all of a computer's packet traffic as well, but it has the ability to block and drop any packets that its programming dictates. Packet sniffers merely watch, display, and log this traffic.
- One disturbingly powerful aspect of packet sniffers is their ability to place the hosting machine's network adapter into "promiscuous mode."
- Network adapters running in promiscuous mode receive not only the data directed to the machine hosting the sniffing software, but also all of the traffic on the physically connected local network. This capability allows packet sniffers to be used as potent spying tools.

1.7 What is a packet sniffer?

A Packet Sniffer is a tool that plugs into a computer network and monitors all network traffic. It Monitors traffic destined to itself as well as to all other hosts on the network. Packet sniffers can be run on both non-switched and switched networks.

1.8 Uses of a packet sniffer

Sniffing programs are found in two forms. **Commercial Packet Sniffers** are used to help maintain networks, while **Underground Packet Sniffers** are used by attackers to gain unauthorized access to remote hosts.

Listed below are some common uses of sniffing programs:

- Searching for clear-text usernames and passwords from the network.
- Conversion of network traffic into human readable form.
- Network analysis to find bottlenecks.
- Network intrusion detection to monitor for attackers.

1.9 How does a packet sniffer work?

A packet sniffer works by looking at every packet sent in the network, including packets not intended for itself. This is accomplished in a variety of ways. These sniffing methods will be described below. Sniffers also work differently depending on the type of network they are in.

- **Shared Ethernet**: In a shared Ethernet environment, all hosts are connected to the same bus and compete with one another for bandwidth. In such an environment packets meant for one machine are received by all the other machines. Thus, any machine in such an environment placed in promiscuous mode will be able to capture packets meant for other machines and can therefore listen to all the traffic on the network.
- **Switched Ethernet**: An Ethernet environment in which the hosts are connected to a switch instead of a hub is called a Switched Ethernet. The switch maintains a table keeping track of each computer's MAC address and delivers packets destined for a particular machine to the port on which that machine is connected. The switch is an intelligent device that sends 2 packets to the destined computer only and does not broadcast to all the machines on the network, as in the previous case. This switched Ethernet environment was intended for better network performance, but as an added benefit, a machine in promiscuous mode will not work here. As a result of this, most network administrators assume that sniffers don't

Work in a Switched Environment.

1.10 TECHNOLOGIES USED

1.10.1 About Java and Java Script

Although the names are almost the same Java is not the same as Java Script. These are two different techniques for Internet programming. Java is a programming language; JavaScript is a scripting language (as the name implies). The difference is that we can create real programs with Java. But often we just want to make a nice effect without having to bother about real programming. So JavaScript is meant to be easy to understand and easy to use. JavaScript authors should not have to care too much about programming.

We could say that JavaScript is rather an extension to HTML than a separate computer language. Of course this is not the 'official' definition but I think this makes it easier to understand the difference between Java and JavaScript, which share the same name and syntax.

Advantages of Java:

Creation of Java:

James Gosling conceived Java. Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan at Sun Micro Systems Incorporation in 1991. It took 18 months to develop the first working version. This language was initially called "OAK" in 1992 and public announcement of Java in 1995, many more contributed to the design and evolution of the language.

Java is Portable:

One of the biggest advantages Java offers is that it is portable. An application written in Java will run on all the major platforms. Any computer with a Java based browser can run the applications or applets written in the Java programming language. A programmer no longer has to write one program to run on a Macintosh, another program to run on a Windows machine, still another to run on a Unix machine and so on. In other words, with Java, developers write their programs only once.

The virtual machine is what gives Java a cross platform capabilities. Rather than being complied into machine language, which is different for each operating systems and computer architecture, Java code is compiled into byte codes. With other languages, the program code is complied into a language that the computer can understand. The problem is that other computers with

different machine instruction set cannot understand that language. Java code, on the other hand is complied into byte codes rather than a machine language. These byte codes go to the Java virtual machine, which executes them directly or translates them into the language that is understood by the machine running it.

In summary, these means that with the JDBC API extending Java, a programmer writing Java code can access all the major relational databases on any platform that supports the Java virtual machine.

Java is Object – Oriented:

Java is Object Oriented, which makes program design focus on what you are dealing with rather than on how you are going to do something. This makes it more useful for programming in sophisticated projects because one can break the things down into understandable components. A big benefit is that these components can then be reused.

Object oriented languages use the paradigm of classes. In simplest term, a class includes both the data and the functions to operate on the data. You can create an instance of a class, also called an object, which will have all the data members and functionality of its class. Because of this, you can think of a class as being like template, with each object being a specific instance of a particular type of class.

The class paradigm allows one to encapsulate data so that specific data values are those using the data cannot see function implementation. Encapsulation makes it possible to make the changes in code without breaking other programs that use that code. If for example the implementation of a function is changed, the change is invisible to another programmer who invokes that function, and it does not affect his/her program, except hopefully to improve it.

Java includes inheritance, or that ability to derive new classes from existing classes. The derived class, also called subclass, inherits all the data and the function of the existing class, referred to as the parent class. A subclass can add new data members to those inherited form the parent class. As far as methods are concerned, the subclass can reuse the inherited methods, as it is, or change them, or even add its own new methods.

Java Makes It Easy:

In addition to being portable and object oriented, Java facilitates writing correct code. Programmers spend less time writing Java code and a lot less time debugging it. In fact, developers have reported slashing development time by as much as two thirds.

Java automatically takes care of allocating and the reallocating memory, a huge potential source of errors. If an object is no longer being used (has no reference to it), then it is automatically removed from memory, or Garbage Collected by a low priority daemon thread called Garbage Collector.

Java's no pointer support eliminates big source errors. By using object references instead of memory pointers, problems with pointer arithmetic are eliminated, and problems with inadvertently accessing the wrong memory address are greatly reduced.

Java's strong typing cuts down on runtime errors, because Java enforces strong type checking, many errors are caught when code is complied. Dynamic binding is possible and often very useful, but static binding with strict type checking is used when possible.

Java keeps code simple by having just one way to do something instead of having several alternatives, as in some languages. Java also stays lean by not including multiple inheritances, which eliminates the errors and ambiguity that arise when you create a subclass that inherits from two or more classes. To replace capabilities, multiple inheritances provide, Java lets you add functionality to a class throw the use of interfaces.

Java is Extensible:

A big plus for Java is the fact it can be extended. It was purposely written to be lean with the emphasis on doing what it does very well, instead of tying to do everything from the beginning, it was return so that extending it is very easy. The java platform includes an extensive class library so that programmers can use already existing classes, as it is, create subclasses to modify existing classes, or implement to augment the capabilities of classes.

Java is Secure:

It is important that a programmer not be able to write subversive code for applications or applets. This is especially true with the Internet being used more and more extensively for services such as electronic commerce and electronic distribution of software and multimedia content.

The way memory is allocated and laid out. In java an object's location in memory is not determined until the runtime, as opposed to C and C++. As the result, a programmer cannot look at a class definition and figure out how it might be laid out in memory. Also since, Java has no pointers, a programmer cannot forge pointers to memory.

The Java Virtual Machine (JVM) doesn't trust any incoming code and subjects it to what is called Byte Code Verification. The byte code verifier, part if the virtual machine, checks that

- The format of incoming code is correct
- Incoming code doesn't forge pointers.
- It doesn't violate access restrictions.
- It access objects as what they are

The Java byte code loader, another part of the JVM, checks whether classes loaded during program execution are local of from across a network. Imported classes cannot be substituted for built in classes, and built in classes cannot accidentally reference classes bring in over a network.

The Java Security manager allows user to restrict entrusted Java applets so that they cannot access the local network, local files and other resources.

Java Performs Well:

Java performance is better than one might expect. Java's many advantages, such as having built in security and being interpreted as well as complied, do have a cost attached to them. However, various optimizations have been built, in, and the byte code interpreter can run very fast the cost it doesn't to do any checking. AS a result, Java has done quite respectably in performance tests. Its performance numbers for interpreted byte codes are usually more than adequate to run interactive graphical end user applications.

For situations that require unusually high performance, byte codes can be translated on the fly generating the final machine code for the particular CPU on which the application is running at run time. Java offers good performance with the advantages of high-level languages but without the disadvantages of C and C++. In the world of design trade-off, you can think of Java as providing a very attractive middle ground.

Java is Robust:

The multi plat formed environment of the WEB places extraordinary demands on a program, because it must execute reliably in a variety of systems. Thus the ability to create robust programs was given a high priority in the design of Java. To gain reliability, Java restricts you in a few key areas to force you to find your mistakes early in program developments. At the same time, Java frees you from having to worry about many of the most common causes of programming errors. Because Java is strictly typed language, it checks your code at compile time. However, it also checks your code at run time. In fact, many hard to track down bugs that often turn up in

hard to reproduce runtime situations are simply impossible to create in Java. Knowing that what you have written will behave in a predictable way under diverse conditions is a key feature of Java.

Java is Multithreaded:

Multithreading is simply the ability of a program to do more than one thing at a time. For example an application could be faxing a document at the same time it is printing another document. Or a program could process new inventory figures while it maintains a feed for current prices. Multithreading is particularly important in multimedia: a multimedia program might often be running a movie, running a audio track and display in text all at the same time.

Java Scales Well:

Java platform is designed to scale well, from portable consumer electronic devices to powerful desktop and server machines. The virtual machine takes a small footprint and Java byte code is optimized to be small and compact. As a result, Java accommodates the need for low storage and for low bandwidth transmission over the Internet. In the addition the Java operating system offers a standalone Java platform that eliminates host operating system overhead while still supporting the full Java Platform API. This makes Java ideal for low cost network computers whose sole purpose is to access the Internet.

Java and Internet:

The Internet helped catapult Java to the forefront of programming and Java in turn has had a profound effect on the Internet. The reason is simple: Java expands the universe of objects that can move about freely in cyberspace. In a network, there are two broad categories of objects transmitted between the Server and your Personal Computer: passive information and dynamic, active programs like an object that can be transmitted to your computer, which is a dynamic, self-executing program. Such a program would be an active agent ton the client computer, yet the server would initiate it. As desirable as dynamic, networked programs are, they also present serious problems in the areas of security and portability. Prior to Java cyberspace was effectively closed to half the entities that now live there. Java addresses these concerns and doing so, has opened the door to an exiting a new form of program.

The rise of server-side Java applications is one of the latest and most exciting trends in Java programming. It was first hyped as a language for developing elaborate client-side web content in the form of applets. Now, Java is coming into its own as a language ideally suited for server-side

development. Businesses in particular have been quick to recognize Java's potential on the server-Java is inherently suited for large client/server applications. The cross platform nature of Java is extremely useful for organizations that have a heterogeneous collection of servers running various flavors of the Unix of Windows operating systems. Java's modern, object-oriented, memory-protected design allows developers to cut development cycles and increase reliability. In addition, Java's built-in support for networking and enterprise API provides access to legacy data, easing the transition from older client/server systems.

Java Servlets are a key component of server-side java development. A Servlets is a small, plug gable extension to a server that enhances the server's functionality. Servlets allow developers to extend and customize and Java enabled server a web server, a mail server, an application server, or any custom server with a hitherto unknown degree of portability, flexibility and ease.

1.10.2 Swings:

Lightweight and Heavyweight components:

Peers are native user interface components delegated to AWT components. Peers take care of all the ground work in designing and painting the components including reacting to events. This keeps less burden on AWT components. This makes AWT components heavyweight.

On the other hand, swing components are called lightweight components as they do not have native peers. But swing's top-level containers like Frame, Applet, Window and Dialog are heavyweight. That is swing's lightweight components are rendered in heavyweight containers like Frame.

1.10.3 MS ACCESS:

Microsoft Office Access, previously known as Microsoft Access, is a system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software development tools. It is a member of the Microsoft Office suite of applications and is included in the Professional and higher versions for Windows and also sold separately.

Access stores data in its own format based on the Access Jet Database Engine. It can also import or link directly to data stored in other Access databases, Excel, SharePoint lists, text, XML, Outlook, HTML, dBase, Paradox, Lotus 1-2-3, or any ODBC-compliant data container including Microsoft SQL Server, Oracle, MySQL and PostgreSQL. Software developers and data architects can use it to develop application software and non-programmer "power users" can use it to build simple applications. Like other Office applications Access is supported by Visual Basic for Applications, an object-oriented programming

language that can reference a wide variety of objects, including DAO (Data Access Objects) and ActiveX Data Objects, and many other ActiveX components provided by Microsoft or by third parties. Visual objects used in forms and reports expose their methods and properties gracefully in the VBA programming environment, and a huge selection of Windows operating system functions can be declared and called from VBA code modules, making Access a rich programming environment.

1.10.3.1 Features:

Users can create tables, queries, forms and reports, and connect them together with macros. Advanced users can use VBA to write rich solutions with advanced data manipulation and user control.

The original concept of Access was for end users to be able to "access" data from any source. Other uses include: the import and export data to many formats including Excel, Outlook, ASCII, dBase, Paradox, FoxPro, SQL Server, Oracle, ODBC, etc. It also has the ability to link to data in its existing location and use it for viewing, querying, editing, and reporting. This allows the existing data to change and the Access platform to always use the latest data. It can perform heterogeneous joins between data sets stored across different platforms. Access is often used by people downloading data from enterprise level databases for manipulation, analysis, and reporting locally.

1.10.4 Microsoft Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to developcomputer programs for Microsoft Windows, as well as web sites, web applications and web services. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source-control systems (like Subversion) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++ and C++/CLI (via Visual C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), and F# (as of Visual Studio 2010). Support for other languages such as M, Python, and Ruby among others is available via language services installed separately. It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS. Java (and J#) were supported in the past.

Microsoft provides "Community" editions of its Visual Studio at no cost. Commercial versions of Visual Studio along with select past versions are available for free to students via Microsoft's DreamSpark program.

Architecture:

Visual Studio does not support any programming language, solution or tool intrinsically; instead, it allows the plugging of functionality coded as a VSPackage. When installed, the functionality is available as a *Service*. The IDE provides three services: SVsSolution, which provides the ability to enumerate projects and solutions; SVsUIShell, which provides windowing and UI functionality (including tabs, toolbars and tool windows); and SVsShell, which deals with registration of VSPackages. In addition, the IDE is also responsible for coordinating and enabling communication between services. All editors, designers, project types and other tools are implemented as VSPackages. Visual Studio uses COM to access the VSPackages. The Visual Studio SDK also includes the *Managed Package Framework* (*MPF*), which is a set of managed wrappers around the COM-interfaces that allow the Packages to be written in any CLI compliant language. However, MPF does not provide all the functionality exposed by the Visual Studio COM interfaces. The services can then be consumed for creation of other packages, which add functionality to the Visual Studio IDE.

Support for programming languages is added by using a specific VSPackage called a *Language Service*. A language service defines various interfaces which the VSPackage implementation can implement to add support for various functionalities. Functionalities that can be added this way include syntax coloring, statement completion, brace matching, parameter information tooltips, member lists and error markers for background compilation. If the interface is implemented, the functionality will be available for the language. Language services are to be implemented on a per-language basis. The implementations can reuse code from the parser or the compiler for the language. Language services can be implemented either in native code or managed code. For native code, either the native COM interfaces or the Babel Framework (part of Visual

Studio SDK) can be used. For managed code, the MPF includes wrappers for writing managed language services.

Visual Studio does not include any source control support built in but it defines two alternative ways for source control systems to integrate with the IDE. A Source Control VSPackage can provide its own customised user interface. In contrast, a source control plugin using the *MSSCCI* (Microsoft Source Code Control Interface) provides a set of functions that are used to implement various source control functionality, with a standard Visual Studio user interface. MSSCCI was first used to integrate Visual SourceSafe with Visual Studio 6.0 but was later opened up via the Visual Studio SDK. Visual Studio .NET 2002 used MSSCCI 1.1, and Visual Studio .NET 2003 used MSSCCI 1.2. Visual Studio 2005, 2008 and 2010 use MSSCCI Version 1.3, which adds support for rename and delete propagation as well as asynchronous opening.

Visual Studio supports running multiple instances of the environment (each with its own set of VSPackages). The instances use different registry hives (see MSDN'sdefinition of the term "registry **hive**" in the sense used here) to store their configuration state and are differentiated by their AppId (Application ID). The instances are launched by an AppId-specific .exe that selects the AppId, sets the root hive and launches the IDE. VSPackages registered for one AppId are integrated with other VSPackages for that AppId. The various product editions of Visual Studio are created using the different AppIds. The Visual Studio Express edition products are installed with their own AppIds, but the Standard, Professional and Team Suite products share the same AppId. Consequently, one can install the Express editions side-by-side with other editions, unlike the other editions which update the same installation. The professional edition includes a superset of the VSPackages in the standard edition and the team suite includes a superset of the VSPackages in both other editions. The AppId system is leveraged by the Visual Studio Shell in Visual Studio 2008.

Features

Code Editor:

Like any other IDE, it includes a code editor that supports syntax highlighting and code completion using IntelliSense for variables,functions, methods, loops and LINQ queries. IntelliSense is supported for the included languages, as well as for XML and forCascading Style Sheets and JavaScript when developing web sites and web applications. Autocomplete suggestions appear in amodeless list box over the code editor window, in proximity of the editing cursor. In Visual

Studio 2008 onwards, it can be made temporarily semi-transparent to see the code obstructed by it. The code editor is used for all supported languages.

The Visual Studio code editor also supports setting bookmarks in code for quick navigation. Other navigational aids include collapsing code blocks and incremental search, in addition to normal text search and regex search. The code editor also includes a multi-item clipboard and a task list. The code editor supports code snippets, which are saved templates for repetitive code and can be inserted into code and customized for the project being worked on. A management tool for code snippets is built in as well. These tools are surfaced as floating windows which can be set to automatically hide when unused or docked to the side of the screen. The Visual Studio code editor also supports code refactoring including parameter reordering, variable and method renaming, interface extraction and encapsulation of class members inside properties, among others.

Visual Studio features background compilation (also called incremental compilation). As code is being written, Visual Studio compiles it in the background in order to provide feedback about syntax and compilation errors, which are flagged with a red wavy underline. Warnings are marked with a green underline. Background compilation does not generate executable code, since it requires a different compiler than the one used to generate executable code.[25] Background compilation was initially introduced with Microsoft Visual Basic but has now been expanded for all included languages.

Debugger:

Visual Studio includes a debugger that works both as a source-level debugger and as a machine-level debugger. It works with bothmanaged code as well as native code and can be used for debugging applications written in any language supported by Visual Studio. In addition, it can also attach to running processes and monitor and debug those processes. If source code for the running process is available, it displays the code as it is being run. If source code is not available, it can show the disassembly. The Visual Studio debugger can also create memory dumps as well as load them later for debugging. Multi-threaded programs are also supported. The debugger can be configured to be launched when an application running outside the Visual Studio environment crashes.

The debugger allows setting breakpoints (which allow execution to be stopped temporarily at a certain position) and watches (which monitor the values of variables as the execution progresses). Breakpoints can be conditional, meaning they get triggered when the condition is met. Code can be stepped over, i.e., run one line (of source code) at a time. It can either *step into* functions to debug inside it, or *step over* it, i.e., the execution of the function body isn't available for manual inspection. The debugger supports *Edit and Continue*, i.e., it allows

code to be edited as it is being debugged. When debugging, if the mouse pointer hovers over any variable, its current value is displayed in a tooltip ("data tooltips"), where it can also be modified if desired. During coding, the Visual Studio debugger lets certain functions be invoked manually from the Immediate tool window. The parameters to the method are supplied at the Immediate window.

Designer:

Visual Studio includes a host of visual designers to aid in the development of applications. These tools include:

Windows Forms Designer

The Windows Forms designer is used to build GUI applications using Windows Forms. Layout can be controlled by housing the controls inside other containers or locking them to the side of the form. Controls that display data (like textbox, list box, grid view, etc.) can be bound to data sources like databases or queries. Data-bound controls can be created by dragging items from the Data Sources window onto a design surface. The UI is linked with code using an event-driven programming model. The designer generates either C# or VB.NET code for the application.

WPF Designer

The WPF designer, codenamed *Cider*, was introduced with Visual Studio 2008. Like the Windows Forms designer it supports the drag and drop metaphor. It is used to author user interfaces targeting Windows Presentation Foundation. It supports all WPF functionality including data binding and automatic layout management. It generates XAML code for the UI. The generated XAML file is compatible with Microsoft Expression Design, the designer-oriented product. The XAML code is linked with code using a code-behind model.

Web designer/development

Visual Studio also includes a web-site editor and designer that allows web pages to be authored by dragging and dropping widgets. It is used for developing ASP.NET applications and supports HTML, CSS and JavaScript. It uses a code-behindmodel to link with ASP.NET code. From Visual Studio 2008 onwards, the layout engine used by the web designer is shared with Microsoft Expression Web. There is also ASP.NET MVC support for MVC technology as a separate download andASP.NET Dynamic Data project available from Microsoft.

Class designer
The Class Designer is used to author and edit the classes (including its members and their access) using UML modeling. The Class Designer can generate C# andVB.NET code outlines for the classes and methods. It can also generate class diagrams from hand-written classes.

Data designer
The data designer can be used to graphically edit database schemas, including typed tables, primary and foreign keys and constraints. It can also be used to design queries from the graphical view.

Mapping designer
From Visual Studio 2008 onwards, the mapping designer is used by LINQ to SQL to design the mapping between database schemas and the classes that encapsulate the data. The new solution from ORM approach, ADO.NET Entity Framework, replaces and improves the old technology.

1.10.5 Microsoft SQL Server

Microsoft SQL (Structured Query Language) Server is a relational database management system developed byMicrosoft. As a database server, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet). There are at least a dozen different editions of Microsoft SQL Server aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with manyconcurrent users. Its primary query languages are T-SQL and ANSI SQL.

History

Prior to version 7.0 the code base for MS SQL Server was sold by Sybase SQL Server to Microsoft, and was Microsoft's entry to the enterprise-level database market, competing against Oracle, IBM, and, later, Sybase. Microsoft, Sybase and Ashton-Tate originally worked together to create and market the first version named SQL Server 1.0 for OS/2 (about 1989) which was essentially the same as Sybase SQL Server 3.0 on Unix, VMS, etc. Microsoft SQL Server 4.2 was shipped around 1992 (available bundled with IBM OS/2 version 1.3). Later Microsoft SQL Server 4.21 for Windows NT was released at the same time as Windows NT 3.1. Microsoft SQL Server v6.0 was the first version designed for NT, and did not include any direction from Sybase.

About the time Windows NT was released in July 1993, Sybase and Microsoft parted ways and each pursued its own design and marketing schemes. Microsoft negotiated exclusive rights to all versions of SQL Server written for Microsoft operating systems. (In 1996 Sybase changed the name of its product to Adaptive Server Enterprise to avoid confusion with Microsoft SQL Server.) Until 1994, Microsoft's SQL Server carried three Sybase copyright notices as an indication of its origin.

SQL Server 7.0 and SQL Server 2000 included modifications and extensions to the Sybase code base, adding support for the IA-64 architecture. By SQL Server 2005 the legacy Sybase code had been completely rewritten.

Since the release of SQL Server 2000, advances have been made in performance, the client IDE tools, and several complementary systems that are packaged with SQL Server 2005. These include:

- an extract-transform-load (ETL) tool (SQL Server Integration Services or SSIS)
- a Reporting Server
- an OLAP and data mining server (Analysis Services)
- several messaging technologies, specifically Service Broker and Notification Services

SQL Server 2012

At the 2011 Professional Association for SQL Server (PASS) summit on October 11, Microsoft announced that the next major version of SQL Server (codenamed "Denali"), would be SQL Server 2012. It was released to manufacturing on March 6, 2012. SQL Server 2012 Service Pack 1 was released to manufacturing on November 9, 2012, and Service Pack 2 was released to manufacturing on June 10, 2014.

It was announced to be the last version to natively support OLE DB and instead to prefer ODBC for native connectivity.

SQL Server 2012's new features and enhancements include AlwaysOn SQL Server Failover Cluster Instances and Availability Groups which provides a set of options to improve database availability, Contained Databases which simplify the moving of databases between instances, new and modified Dynamic Management Views and Functions, programmability enhancements including new spatial features, metadata discovery, sequence objects and the THROW statement, performance enhancements such as ColumnStore Indexes as well as improvements to OnLine and partition level operations and security

enhancements including provisioning during setup, new permissions, improved role management, and default schema assignment for groups

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

2.1 INTRODUCTION TO SYSTEM ANALYSIS:

System Analysis is first stage according to System Development Life Cycle model. This System Analysis is a process that starts with the analyst. Analysis is a detailed study of the various operations performed by a system and their relationships within and outside the system. One aspect of analysis is defining the boundaries of the system and determining whether or not a candidate should consider other related systems.

2.1.1 EXISTING SYSTEM

- The System administrators are intended to know about the exchange of data in their organization, but it is not possible to do so manually.
- In the late 90's a C based sniffer program was introduced which can capture only one packet at a time.
- The performance of this c based program is low as it takes more time to capture a packet decode and analyze it
- Where as a large exchange of packets flows in the network, it can capture only one packet at a time, this makes a lot of time to capture all the packets that flow through the network.

2.1.2 NEED FOR AUTOMATION

The reasons for automating the existing system is

- Existing sniffer used c based program which is a basic application model.
- This captures only single packet at a time, even when multiple packets are flowing in the network.
- In order to capture all the packets those flow through the network the existing sniffer performance level is low.
- Because of degradation in performance it takes much time to capture a packet.
- Existing sniffer was not build based on the GUI.

2.1.3 PROPOSED SYSTEM Packet sniffer, a special name of network monitors and analyzer, also usually stands for a means of collecting data and information. ISS defines sniffer as: Sniffer is a tool which utilizes network interfaces of computer to capture data packets which destination is other computers.

2.1.4 Benefits Of Proposed System:

- Analyze network problems.
- Detect network intrusion attempts.
- Gain information for effecting a network intrusion.
- Monitor network usage.
- Gather and report network statistics.
- Filter suspect content from network traffic
- Spy on other network users and collect sensitive information such as passwords.(depending on any content encryption methods which may be in use).
- Reverse engineer proprietary protocols used over the network.
- Debug client/server communications.
- Debug network protocol implementations.

2.2 FEASIBILITY STUDY:

Feasibility study is conducted once the problem is clearly understood. The feasibility study which is a high level capsule version of the entire system analysis and design process. The objective is to determine whether the proposed system is feasible or not and it helps us to the minimum expense of how to solve the problem and to determine, if the problem is worth solving. The following are the three important tests that have been carried out for feasibility study.

1. Technical Feasibility

2. Economical Feasibility

3. Operation Feasibility

Technical Feasibility:

In the technical feasibility study, one has to test whether the proposed system can be developed using existing technology or not. It is planned to implement the proposed system in JSP. The project entitled "Packet based network sniffer" is technically feasible because of the following reasons.

• All necessary technology exists to develop the system.

• The existing system is so flexible that it can be developed further.

Economical Feasibility:

As a part of this, the costs and benefits associated with the proposed systems are to be compared. The project is economically feasible only if tangible and intangible benefits outweigh the cost. We can say the proposed system is feasible based on the following grounds.

• The cost of developing the full system is reasonable.

• The cost of hardware and software for the application is less.

Operational Feasibility:

The project is operationally feasible because there is sufficient support from the project management and the users of the proposed system. Proposed system definitely does not harm and will not produce the bad results and no problem will arise after implementation of the system**.**

2.3 SYSTEM REQUIREMENTS SPECIFICATION:

2.3.1 FUNCTIONAL REQUIREMENTS:

Functional requirements capture the intended behavior of the system. This behavioral is expressed as services, tasks or functions the system is required to perform.

It uses very low resources while running .It uses java server like TOMCAT. That can be used freely and also requires the database. We could use M.S Access.

2.3.1.1 Modules:

Functional components of Packet Based Network Sniffer are

Only two types of users are available:

1. System Administrator
2. Clients or users.

The basic operations involved are:

1. Packet capturing
2. Packet Decoding

Administrator:

In an organization when two clients are exchanging their data administrator observes whether any unwanted data is interfering the organizations private data or not. By using the sniffer tool administrator can provide security to the entire network in an organization.

Users:

Users or clients are the people who exchange their data in an organization through network.

Packet Capturing:

In the network data is transferred in the form of chunks called packets. In order to capture a packet we have to first set the interface system to "promiscuous mode". After setting the system to promiscuous mode the sniffer activates and captures all the data that passes through the network.

Packet Decoding:

This module explains about the packet decoding process where the captured packets are decoded into hexadecimal format and are displayed on the list panel. Packet decoding is done using the OSI reference model.

2.3.2 PSEUDO REQUIREMENTS:

Pseudo requirements describe the design and implementation constraints imposed by the client. This includes the specification of the deployment platform, implementation language, or database management system. Our system includes the following pseudo requirements:

2.3.2.1 HARDWARE SPECIFICATION:

- Processor : Pentium 233MHz
- Hard Disk : 40GB
- Display device : 14" color monitor
- RAM : 256 MB(server), 128 MB(client)
- Internet Connection : 33.6kb/s modem

2.3.2.2 SOFTWARE SPECIFICATION:

- Operating System : Windows XP Professional
- Database : MS Access
- Web application server : Apache Tomcat 5.0
- Others : JSP, java script, swings
- Tools Used : Winpcap, Jpcap

2.4 OBJECT ORIENTED ANALYSIS

2.4.1 Diagrams in UML:

Diagrams are graphical presentation of set of elements. Diagrams project a system, or visualize a system from different angles and perspectives.

The UML has nine diagrams these diagrams can be classified into the following groups.

Static:

1. Class diagrams.
2. Object diagrams.
3. Component diagrams.
4. Deployment diagrams

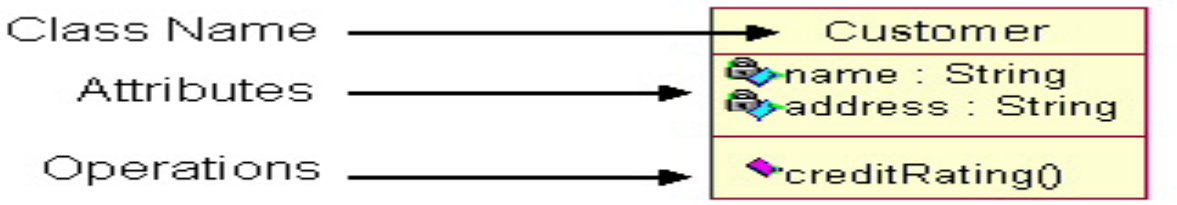
Dynamic:

1. Use case diagram.
2. Sequence diagram.
3. Collaboration diagram.
4. State chart diagram.
5. Activity diagram.

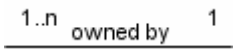
2.4.1.1Class diagram:

This shows a set of classes, interfaces, collaborations and their relationships. There are the most common diagrams in modeling the object oriented systems and are used to give the static view of a system.

Classes are composed of three things: a name, attributes, and operations. Below is an example of a class.



Association is a generic relationship between two classes and is modelled by a line connecting the two classes. This line also shows the feature multiplicity.

For example 

Aggregations indicate whole part-of relationship. It is represented by the symbol .

2.4.1.2Object diagram:

Shows a set of objects and their relationships and are used to show the data structures, the static snapshots of instances of the elements in a class diagram. Like class diagram, the object diagrams also address the static design view or process view of a system.

2.4.1.3Component diagram:

Shows a set of components and their relationships and are used to illustrate the static implementation view of a system. They are

related to class diagrams where in components map to one or more classes, interfaces of collaborations.

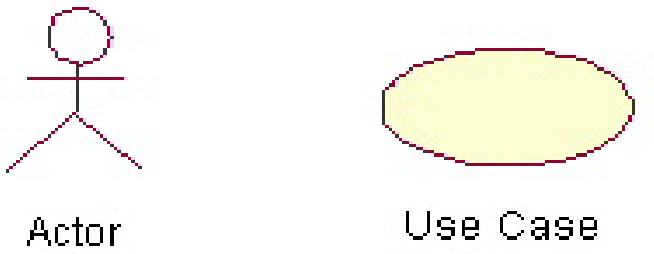
2.4.1.3Deployment diagram:

Shows a set of nodes and their relationships. They are used to show the static deployment view of the architecture of a system. They are related to the component diagrams where a node encloses one or more components.

2.4.1.3Use Case diagram:

Shows a set of use cases and actors and their relationships. These diagrams illustrate the static use case view of a system and are important in organizing and modeling the behaviors of a system.

. The two main components of a use case diagram are use cases and actors.



2.4.1.4Sequence diagram & collaboration diagram:

These two diagrams are semantically same i.e. the dynamics of a system can be modeled using one diagram and transform it to the other kind of diagram without loss of information. Both form the, Interaction diagram.

2.4.1.4.1Sequence diagram:

Sequence diagram is an interaction diagram which focuses on the time ordering of messages it shows a set of objects and messages exchange between these objects. This diagram illustrates the dynamic view of a system.

2.4.1.4.2Collaboration diagram:

This diagram is an interaction diagram that stresses or emphasizes the structural organization of the objects that send and receive messages. It shows a set of objects, links between objects and messages send and received by those objects. There are used to illustrate the dynamic vies of a system.

2.4.1.5State Chart Diagram:

State chart diagram shows a state machine consisting of states, transitions and activities these illustrates the dynamic view of a system. They focus on the event ordered Behavior of an object.

2.4.1.6Activity Diagram:

Activity diagram shows the flow from one activity to another with in a system. The activities may be sequential or branching objects that act and are acted upon. These also show the dynamic view of the system.

3. SYSTEM DESIGN

3.SYSTEM DESIGN

3.1 INTRODUCTION:

System designing is a solution ,"how to " approach to the creation of a new system. This important phase provides the understanding and procedural details necessary for implementing the system recommend in the feasible study. The design step produces a data design , an architectural design and a procedural design.

The data Design transform the information domain model created during analysis into data structures that will be required to implement the software.

The architectural design defines the relationship among the major structural components into a procedural description of the software. Source code is generated and testing is conducted to integrate and validate the software.

From a project Management of view , Software design is conducted in two steps.

Their preliminary design is concern with the transformation in to data and Software architecture.

Detailed design focuses on refinement to the architectural representation that lead to detailed data structure and algorithmic representations for software.

3.2 OBJECT ORIENTED DESIGN:

DATA DESIGN:

Data design is the first of the three design activities that are conducted during software engineering. The impact of the data structure and procedural complexity causes data design to have a profound influence on the software quality. The Concepts of information hiding and data abstraction provides the foundation for an approach to data design.

ARCHITECTURAL DESIGN:

The primary objective of the architectural design is to develop a modular program structure and represent the control relationship between modules. In addition, architectural design melds program structure and data defining interfaces that enable data to flow throughout the program.

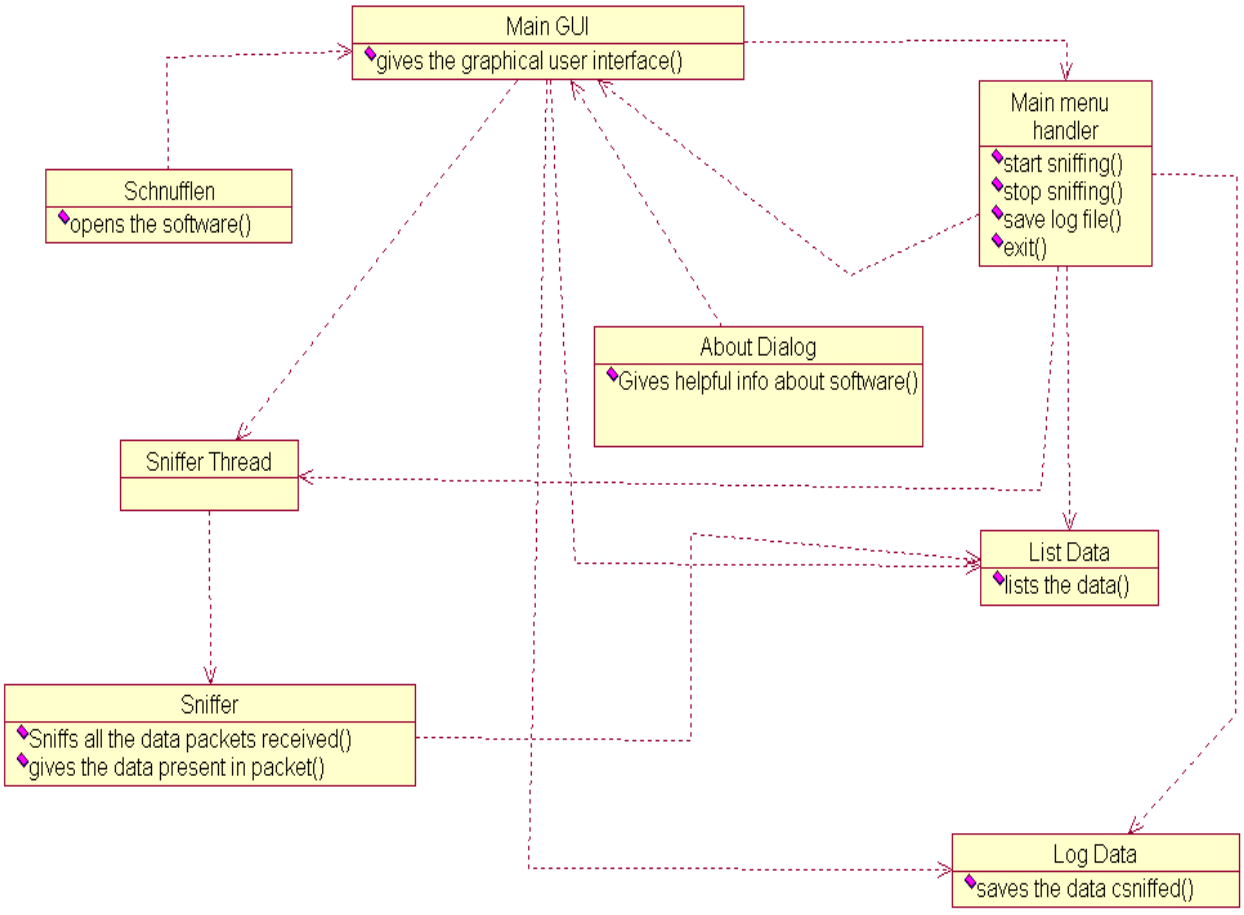
PROCEDURAL DESIGN:

The primary design occurs after data and program structure has been established. Procedural detail can be specific sing either of the following:

- Structured programming.
- Flowchart constructs.
- Decision tables.
- Structured English.

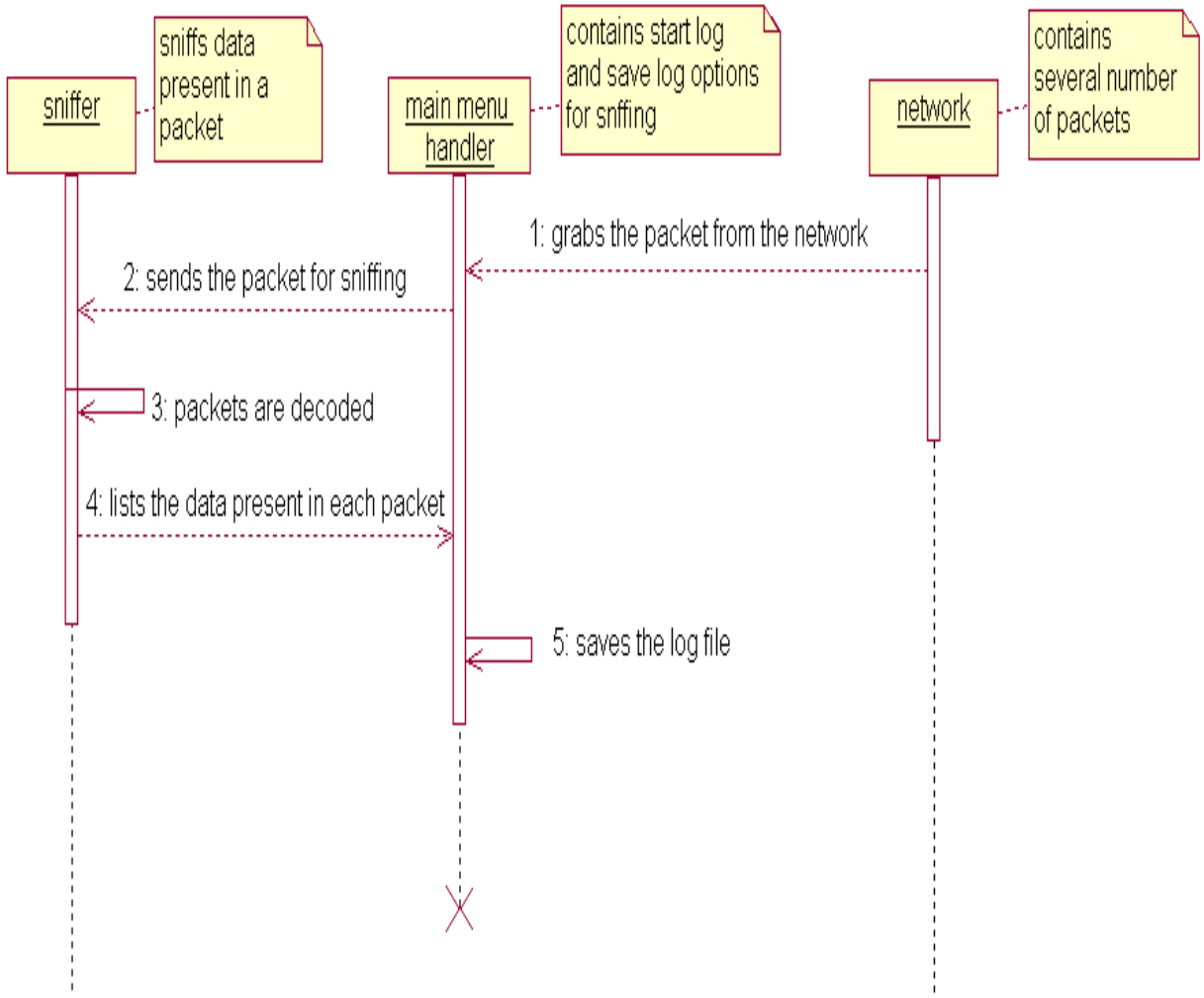
3.2.1 Class Diagrams:

This section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. And for each significant package, its decomposition into classes and class utilities.



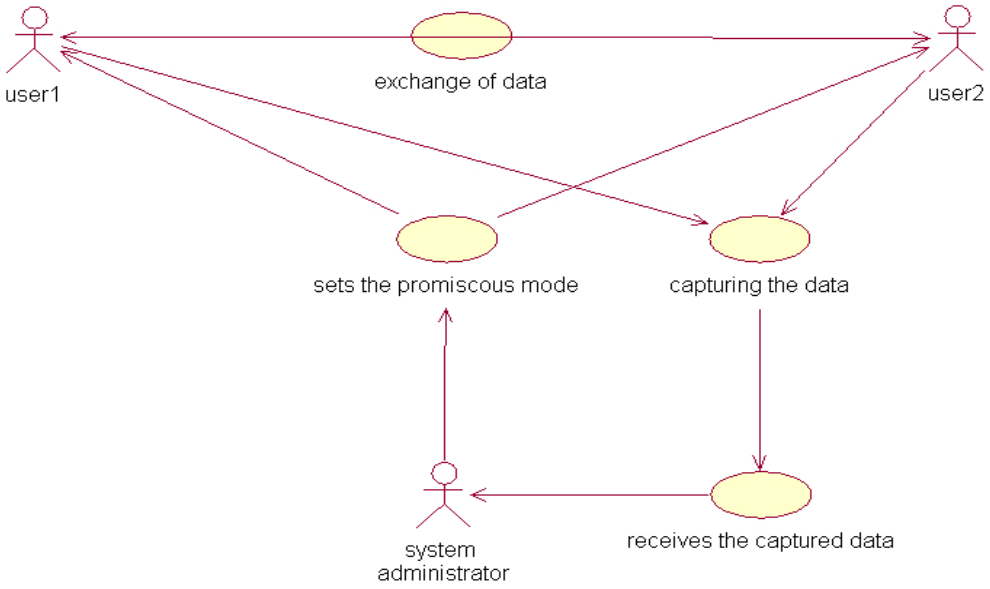
3.2.2 Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart.

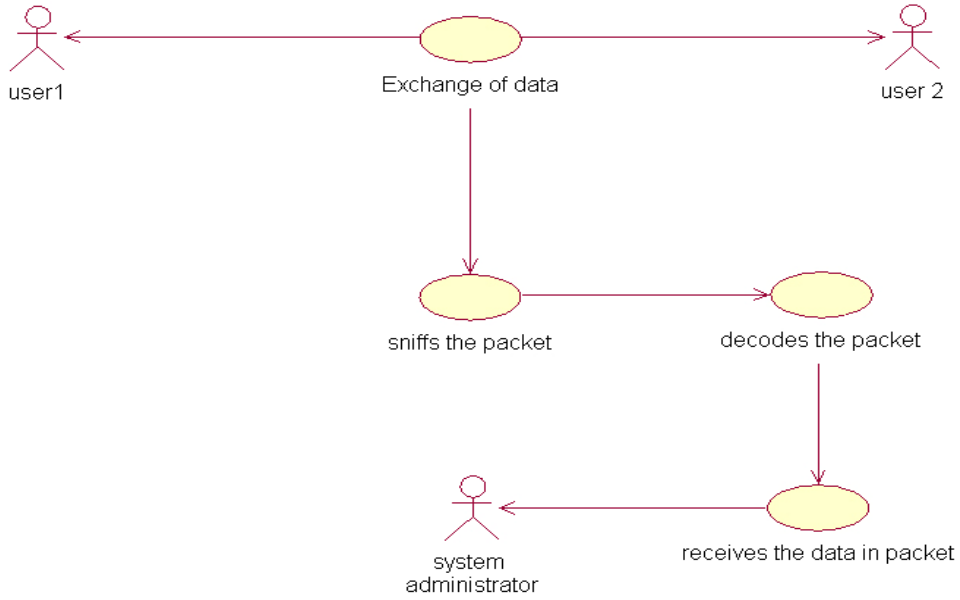


3.3 USE CASE DIAGRAMS

3.3.1 Packet Capturing:



3.3.2 Decoding the captured packet:



3.4 USER INTERFACE DESIGN:

User interface design or user interface engineering is the design of computers, appliances, machines, mobile communication devices, software applications, and websites with the focus on users experience and interaction. The goal of users interface design is to make the users interaction as simple and efficient as possible, in terms of accomplishing user goals-What is often called user-centered design. Good user interface design facilitates finishing the task at hand without drawing unnecessary attention to it. Graphic design may be utilized to apply theme or style to the interface without compromising its usability. The design process of an interface must balance the meaning of its visual elements that confirm the mental model of operation, and the functionality from a technical engineering perspective, in order to create a system that is both usable and easy to adapt to the changing user needs.

4. TESTING

4.1 SOFTWARE TESTING TECHNIQUES:

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, designing and coding.

4.1.1 TESTING OBJECTIVES:

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

These above objectives imply a dramatic change in view port.

Testing cannot show the absence of defects, it can only show that software errors are present.

4.1.2 TEST CASE DESIGN:

Any engineering product can be tested in one of two ways:

1. White Box Testing: This testing is also called as glass box testing. In this testing, by knowing the specified function that a product has been designed to perform test can be conducted that demonstrates each function is fully operation at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis Path Testing:

i. Flow graph notation
ii. Cyclomatic Complexity
iii. Deriving test cases
iv. Graph matrices

Control Structure Testing:

i. Condition testing
ii. Data flow testing
iii. Loop testing

2. Black Box Testing: In this testing by knowing the internal operation of a product, tests can be conducted to ensure that " all gears mesh", that is the

internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

i. Graph based testing methods
ii. Equivalence partitioning
iii. Boundary value analysis
iv. Comparison testing

4.2 SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set of activities that can be planned in advance and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be defined for software engineering process. Any software testing strategy should have the following characteristics:

1. Testing begins at the module level and works "outward" toward the integration of the entire computer based system.
2. Different testing techniques are appropriate at different points in time.
3. The developer of the software and an independent test group conducts testing.
4. Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

4.3 UNIT TESTING:

Unit testing focuses verification efforts in smallest unit of software design (module).

1. Unit test considerations
2. Unit test procedures

Integration Testing: Integration testing is a systematic technique for constructing the program structure while conducting tests to uncover errors associated with interfacing. There are two types of integration testing:

1. **Top-Down Integration**: Top down integration is an incremental approach to construction of program structures. Modules are integrated by moving down wards throw the control hierarchy beginning with the main control module.
2. **Bottom-Up Integration**: Bottom up integration as its name implies, begins construction and testing with automatic modules.

3. **Regression Testing**: In this contest of an integration test strategy, regression testing is the re execution of some subset of test that have already been conducted to ensure that changes have not propagate unintended side effects.

4.4 VALIDATION TESTING:

At the culmination of integration testing, software is completely assembled as a package; interfacing errors have been uncovered and corrected, and a final series of software tests – validation testing – may begin. Validation can be fined in many ways, but a simple definition is that validation succeeds when software functions in a manner that can be reasonably expected by the customer.

Reasonable expectation is defined in the software requirement specification – a document that describes all user-visible attributes of the software. The specification contains a section titled "Validation Criteria". Information contained in that section forms the basis for a validation testing approach.

4.4.1Validation Test Criteria:

Software validation is achieved through a series of black-box tests that demonstrate conformity with requirement. A test plan outlines the classes of tests to be conducted, and a test procedure defines specific test cases that will be used in an attempt to uncover errors in conformity with requirements. Both the plan and procedure are designed to ensure that all functional requirements are satisfied; all performance requirements are achieved; documentation is correct and human-engineered; and other requirements are met.

After each validation test case has been conducted, one of two possible conditions exist: (1) The function or performance characteristics conform to specification and are accepted, or (2) a deviation from specification is uncovered and a deficiency list is created. Deviation or error discovered at this stage in a project can rarely be corrected prior to scheduled completion. It is often necessary to negotiate with the customer to establish a method for resolving deficiencies.

4.4.2Configuration Review:

An important element of the validation process is a configuration review. The intent of the review is to ensure that all elements of the software configuration have been properly developed, are catalogued, and have the necessary detail to

support the maintenance phase of the software life cycle. The configuration review sometimes called an audit.

5. CONCLUSION

Online communication is a new phenomenon, having first come into existence toward the end of the 20th century. It is growing at one of the fastest rates of any new form of communication in human history, and its long-term impact is expected to be substantial. A not uncommon, and, in my eyes, justifiable, view is that online communication represents the most important development in human communication and cognition since the development of the printing press.

The increased prominence of online communication in society, with e-mail surpassing telephone conversation and even face-to-face conversation as a frequent tool of communication among some occupational groups (American Management Association International, 1998) and the World Wide Web rapidly expanding its presence and impact in fields ranging from academia to entertainment to marketing.

Having looked at what they are, why they work and how they are used, it is easy to view sniffers as both dangerous threats and powerful tools. Every user should understand they are vulnerable to these types of attacks and their best defense lies in encryption. Administrators and professionals need to know that these programs are superb diagnostic utilities that can, unfortunately, be used with malicious intent on any network. In common industry usage, a sniffer (with lower case "s") is a program that monitors and analyzes network traffic, detecting bottlenecks and problems. Using this information, a network manager can keep traffic flowing efficiently. Sniffer (with a capital "S") is a trademark owned by Network General. The generic term may have originated from Sniffer, which is said to be the first packet capture and decode software that was offered for the purpose of network analysis and troubleshooting.

It gives us with the in's and out’s of data transferred in organization. Packet sniffing is not only for crackers. It’s an extremely useful practice for administrators who are troubleshooting network problems.

6. BIBLIOGRAPHY

7.BIBLIOGRAPHY

1. OBJECT ORIENTED - BERND BRUEGGE & ALLEN H

2. SOFTWARE ENGINEERING - DUTOIT

3. UML - JOSEPH SCHNMULLER

4. JAVA COMPETE REFERENCE - HERBET SCHILD

5. INTRODUCING MICROSOFT SQL SERVER 2012

WEB SITES:

- **Christopher Klaus, The computer-security/ sniffers 1997,**

  http://www.faqs.org/faqs/ computer-security/sniffers/ .
- **Robert Graham, Network Sniffing,**

http://www.robertgraham.com/pubs/ sniffing-faq.html

- **Kevin J. Connolly, packet analyzer,**

  http://en.wikipedia.org/wiki/Packet_analyzer

- **Java script**

http://www.w3schools.com/JS/default.asp

- **JSP**

  http://www.jsptut.com/

- **Visual Basic**

  http://www.onlineprogrammingbooks.com/visualbasic-net/