

LECTURE NOTES ON

DATA MINING& DATA WAREHOUSING



GraphicEra
(Deemed to be University)
Accredited by NAAC with Grade A

Chapter-1

1.1 What Is Data Mining?

Data mining refers to extracting or mining knowledge from large amounts of data. The term is actually a misnomer. Thus, data mining should have been more appropriately named as knowledge mining which emphasizes on mining from large amounts of data.

It is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems.

The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use.

The key properties of data mining are

- Automatic discovery of patterns
- Prediction of likely outcomes
- Creation of actionable information
- Focus on large datasets and databases

1.2 The Scope of Data Mining

Data mining derives its name from the similarities between searching for valuable business information in a large database — for example, finding linked products in gigabytes of store scanner data — and mining a mountain for a vein of valuable ore. Both processes require either sifting through an immense amount of material, or intelligently probing it to find exactly where the value resides. Given databases of sufficient size and quality, data mining technology can generate new business opportunities by providing these capabilities:

Automated prediction of trends and behaviors. Data mining automates the process of finding predictive information in large databases. Questions that traditionally required extensive handson analysis can now be answered directly from the data — quickly. A typical example of a predictive problem is targeted marketing. Data mining uses data on past promotional mailings to identify the targets most likely to maximize return on investment in future mailings. Other predictive problems include forecasting bankruptcy and other forms of default, and identifying segments of a population likely to respond similarly to given events.

Automated discovery of previously unknown patterns. Data mining tools sweep through databases and identify previously hidden patterns in one step. An example of pattern discovery is the analysis of retail sales data to identify seemingly unrelated products that are often purchased together. Other pattern discovery problems include detecting fraudulent credit card transactions and identifying anomalous data that could represent data entry keying errors.

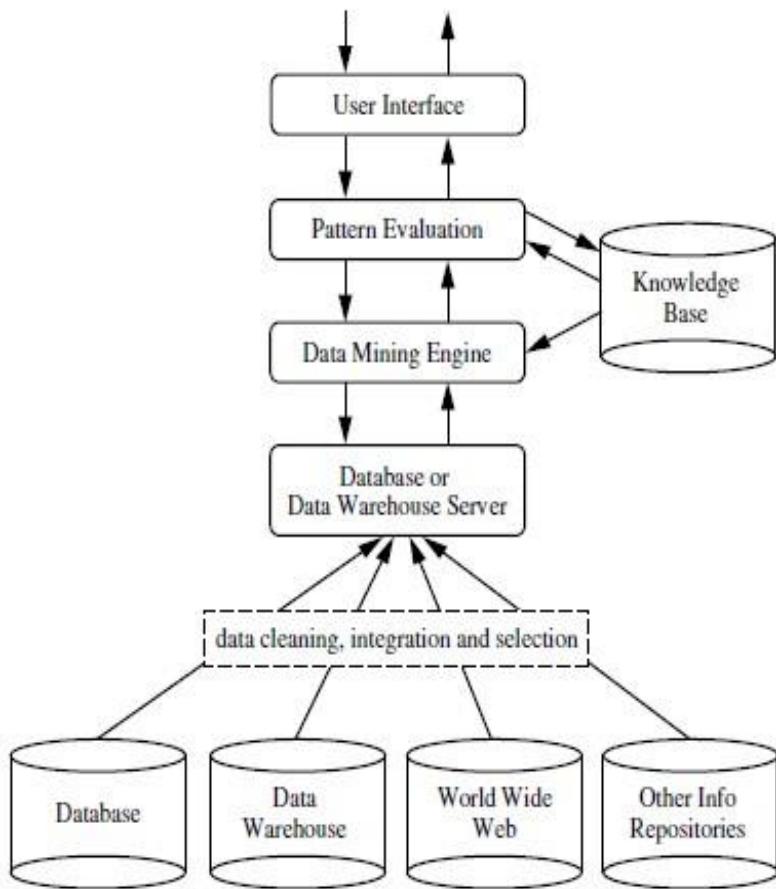
1.3 Tasks of Data Mining

Data mining involves six common classes of tasks:

- **Anomaly detection (Outlier/change/deviation detection)** – The identification of unusual data records, that might be interesting or data errors that require further investigation.
- **Association rule learning (Dependency modelling)** – Searches for relationships between variables. For example a supermarket might gather data on customer purchasing habits. Using association rule learning, the supermarket can determine which products are frequently bought together and use this information for marketing purposes. This is sometimes referred to as market basket analysis.
- **Clustering** – is the task of discovering groups and structures in the data that are in some way or another "similar", without using known structures in the data.
- **Classification** – is the task of generalizing known structure to apply to new data. For example, an e-mail program might attempt to classify an e-mail as "legitimate" or as "spam".
- **Regression** – attempts to find a function which models the data with the least error.
- **Summarization** – providing a more compact representation of the data set, including visualization and report generation.

1.4 Architecture of Data Mining

A typical data mining system may have the following major components.



1. Knowledge Base:

This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies, used to organize attributes or attribute values into different levels of abstraction.

Knowledge such as user beliefs, which can be used to assess a pattern's interestingness based on its unexpectedness, may also be included. Other examples of domain knowledge are additional interestingness constraints or thresholds, and metadata (e.g., describing data from multiple heterogeneous sources).

2. Data Mining Engine:

This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.

3. Pattern Evaluation Module:

This component typically employs interestingness measures interacts with the data mining modules so as to focus the search toward interesting patterns. It may use interestingness thresholds to filter out discovered patterns. Alternatively, the pattern evaluation module may be integrated with the mining module, depending on the implementation of the data mining method used. For efficient data mining, it is highly recommended to push the evaluation of pattern interestingness as deep as possible into the mining process so as to confine the search to only the interesting patterns.

4. User interface:

This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task, providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results. In addition, this component allows the user to browse database and data warehouse schemas or data structures, evaluate mined patterns, and visualize the patterns in different forms.

1.5 Data Mining Process:

Data Mining is a process of discovering various models, summaries, and derived values from a given collection of data.

The general experimental procedure adapted to data-mining problems involves the following steps:

1. State the problem and formulate the hypothesis

Most data-based modeling studies are performed in a particular application domain. Hence, domain-specific knowledge and experience are usually necessary in order to come up with a meaningful problem statement. Unfortunately, many application studies tend to focus on the data-mining technique at the expense of a clear problem statement. In this step, a modeler usually specifies a set of variables for the unknown dependency and, if possible, a general form of this dependency as an initial hypothesis. There may be several hypotheses formulated for a single problem at this stage. The first step requires the combined expertise of an application domain and a data-mining model. In practice, it usually means a close interaction between the data-mining expert and the application expert. In successful data-mining applications, this cooperation does not stop in the initial phase; it continues during the entire data-mining process.

2. Collect the data

This step is concerned with how the data are generated and collected. In general, there are two distinct possibilities. The first is when the data-generation process is under the control of an expert (modeler): this approach is known as a designed experiment. The second possibility is when the expert cannot influence the data-generation process: this is known as the observational approach. An observational setting, namely, random data generation, is assumed in most data-mining applications. Typically, the sampling distribution is completely unknown after data are collected, or it is partially and implicitly given in the data-collection procedure. It is very important, however, to understand how data collection affects its theoretical distribution, since such a priori knowledge can be very useful for modeling and, later, for the final interpretation of results. Also, it is important to make sure that the data used for estimating a model and the data used later for testing and applying a model come from the same, unknown, sampling distribution. If this is not the case, the estimated model cannot be successfully used in a final application of the results.

3. Preprocessing the data

In the observational setting, data are usually "collected" from the existing databases, data warehouses, and data marts. Data preprocessing usually includes at least two common tasks:

1. Outlier detection (and removal) – Outliers are unusual data values that are not consistent with most observations. Commonly, outliers result from measurement errors, coding and recording errors, and, sometimes, are natural, abnormal values. Such nonrepresentative samples can seriously affect the model produced later. There are two strategies for dealing with outliers:

- a. Detect and eventually remove outliers as a part of the preprocessing phase, or
- b. Develop robust modeling methods that are insensitive to outliers.

2. Scaling, encoding, and selecting features – Data preprocessing includes several steps such as variable scaling and different types of encoding. For example, one feature with the range [0, 1] and the other with the range [-100, 1000] will not have the same weights in the applied technique; they will also influence the final data-mining results differently. Therefore, it is recommended to scale them and bring both features to the same weight for further analysis. Also, application-specific encoding methods usually achieve dimensionality reduction by providing a smaller number of informative features for subsequent data modeling.

These two classes of preprocessing tasks are only illustrative examples of a large spectrum of preprocessing activities in a data-mining process.

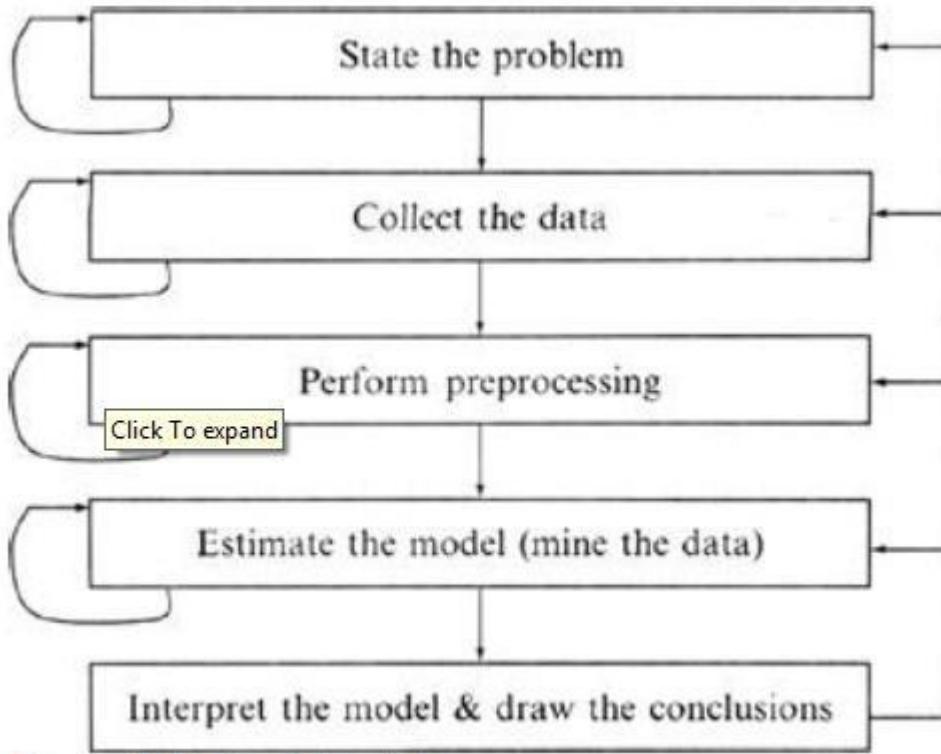
Data-preprocessing steps should not be considered completely independent from other data-mining phases. In every iteration of the data-mining process, all activities, together, could define new and improved data sets for subsequent iterations. Generally, a good preprocessing method provides an optimal representation for a data-mining technique by incorporating a priori knowledge in the form of application-specific scaling and encoding.

4. Estimate the model

The selection and implementation of the appropriate data-mining technique is the main task in this phase. This process is not straightforward; usually, in practice, the implementation is based on several models, and selecting the best one is an additional task. The basic principles of learning and discovery from data are given in Chapter 4 of this book. Later, Chapter 5 through 13 explain and analyze specific techniques that are applied to perform a successful learning process from data and to develop an appropriate model.

5. Interpret the model and draw conclusions

In most cases, data-mining models should help in decision making. Hence, such models need to be interpretable in order to be useful because humans are not likely to base their decisions on complex "black-box" models. Note that the goals of accuracy of the model and accuracy of its interpretation are somewhat contradictory. Usually, simple models are more interpretable, but they are also less accurate. Modern data-mining methods are expected to yield highly accurate results using highdimensional models. The problem of interpreting these models, also very important, is considered a separate task, with specific techniques to validate the results. A user does not want hundreds of pages of numeric results. He does not understand them; he cannot summarize, interpret, and use them for successful decision making.



The Data mining Process

1.6 Classification of Data mining Systems:

The data mining system can be classified according to the following criteria:

- Database Technology
- Statistics
- Machine Learning
- Information Science
- Visualization
- Other Disciplines

Some Other Classification Criteria:

- Classification according to kind of databases mined
- Classification according to kind of knowledge mined
- Classification according to kinds of techniques utilized
- Classification according to applications adapted

Classification according to kind of databases mined

We can classify the data mining system according to kind of databases mined. Database system can be classified according to different criteria such as data models, types of data etc. And the data mining system can be classified accordingly. For example if we classify the database according to data model then we may have a relational, transactional, object- relational, or data warehouse mining system.

Classification according to kind of knowledge mined

We can classify the data mining system according to kind of knowledge mined. It means data mining system are classified on the basis of functionalities such as:

- Characterization
- Discrimination
- Association and Correlation Analysis
- Classification
- Prediction
- Clustering
- Outlier Analysis
- Evolution Analysis

Classification according to kinds of techniques utilized

We can classify the data mining system according to kind of techniques used. We can describes these techniques according to degree of user interaction involved or the methods of analysis employed.

Classification according to applications adapted

We can classify the data mining system according to application adapted. These applications are as follows:

- Finance
 - Telecommunications
 - DNA
 - Stock Markets •
- E-mail

1.7 Major Issues In Data Mining:

•Mining different kinds of knowledge in databases. - The need of different users is not the same. And Different user may be interested in different kind of knowledge. Therefore it is necessary for data mining to cover broad range of knowledge discovery task.

•Interactive mining of knowledge at multiple levels of abstraction. - The data mining process needs to be interactive because it allows users to focus the search for patterns, providing and refining data mining requests based on returned results.

•Incorporation of background knowledge. - To guide discovery process and to express the discovered patterns, the background knowledge can be used. Background knowledge may be used to express the discovered patterns not only in concise terms but at multiple level of abstraction.

•Data mining query languages and ad hoc data mining. - Data Mining Query language that allows the user to describe ad hoc mining tasks, should be integrated with a data warehouse query language and optimized for efficient and flexible data mining.

- **Presentation and visualization of data mining results.** - Once the patterns are discovered it needs to be expressed in high level languages, visual representations. These representations should be easily understandable by the users.
- **Handling noisy or incomplete data.** - The data cleaning methods are required that can handle the noise, incomplete objects while mining the data regularities. If data cleaning methods are not there then the accuracy of the discovered patterns will be poor.
- **Pattern evaluation.** - It refers to interestingness of the problem. The patterns discovered should be interesting because either they represent common knowledge or lack novelty.
- **Efficiency and scalability of data mining algorithms.** - In order to effectively extract the information from huge amount of data in databases, data mining algorithm must be efficient and scalable.
- **Parallel, distributed, and incremental mining algorithms.** - The factors such as huge size of databases, wide distribution of data, and complexity of data mining methods motivate the development of parallel and distributed data mining algorithms. These algorithms divide the data into partitions which are further processed parallel. Then the results from the partitions are merged. The incremental algorithms update databases without having to mine the data again from scratch.

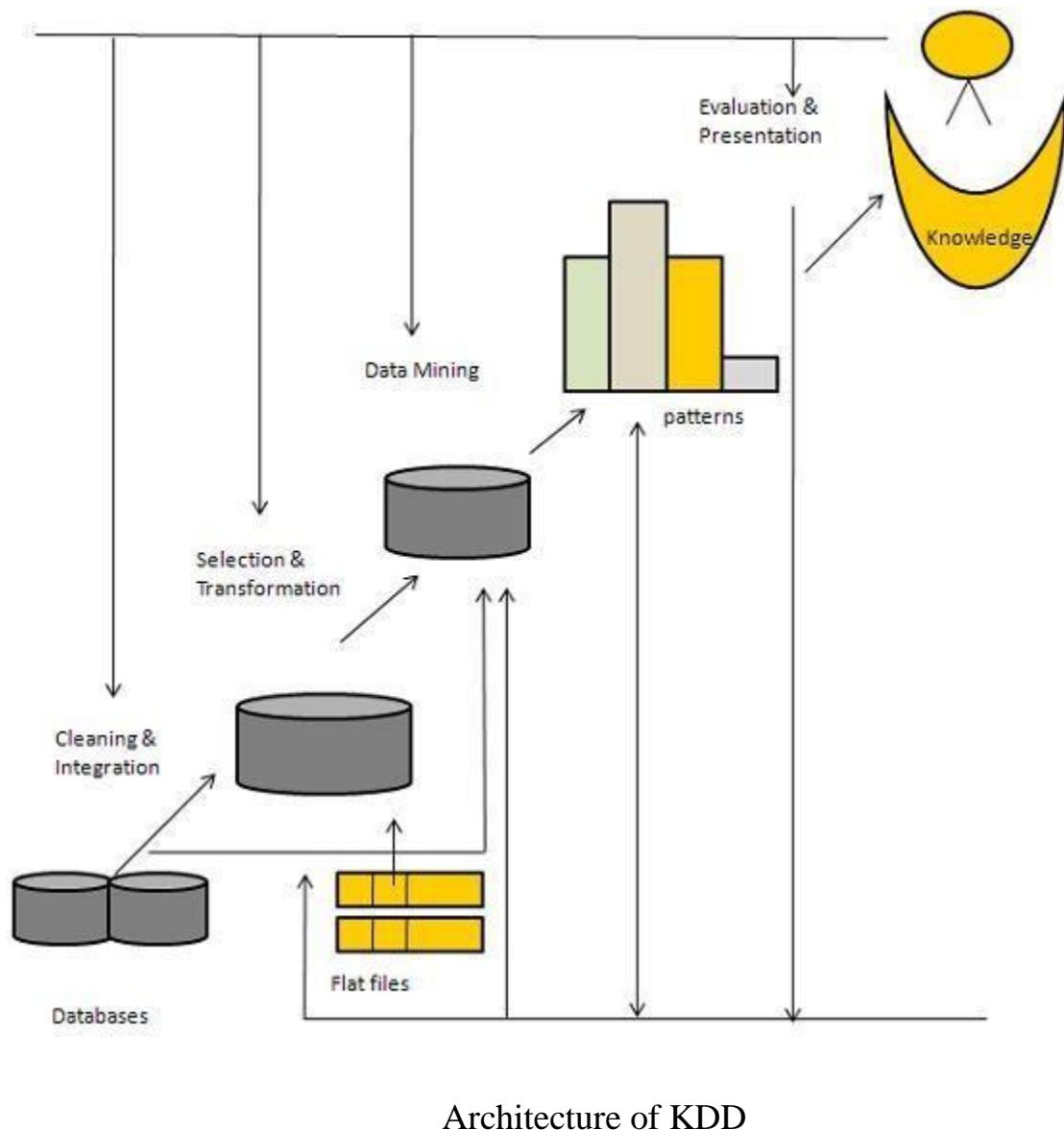
1.8 Knowledge Discovery in Databases(KDD)

Some people treat data mining same as Knowledge discovery while some people view data mining as an essential step in process of knowledge discovery. Here is the list of steps involved in knowledge discovery process:

- **Data Cleaning** - In this step the noise and inconsistent data is removed.

- **Data Integration** - In this step multiple data sources are combined.
- **Data Selection** - In this step relevant to the analysis task are retrieved from the database.
- **Data Transformation** - In this step data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations.
- **Data Mining** - In this step intelligent methods are applied in order to extract data patterns.
- **Pattern Evaluation** - In this step, data patterns are evaluated.
- **Knowledge Presentation** - In this step,knowledge is represented.

The following diagram shows the process of knowledge discovery process:



1.9 Data Warehouse:

A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process.

Subject-Oriented: A data warehouse can be used to analyze a particular subject area. For example, "sales" can be a particular subject.

Integrated: A data warehouse integrates data from multiple data sources. For example, source A and source B may have different ways of identifying a product, but in a data warehouse, there will be only a single way of identifying a product.

Time-Variant: Historical data is kept in a data warehouse. For example, one can retrieve data from 3 months, 6 months, 12 months, or even older data from a data warehouse. This contrasts with a transactions system, where often only the most recent data is kept. For example, a transaction system may hold the most recent address of a customer, where a data warehouse can hold all addresses associated with a customer.

Non-volatile: Once data is in the data warehouse, it will not change. So, historical data in a data warehouse should never be altered.

1.9.1 Data Warehouse Design Process:

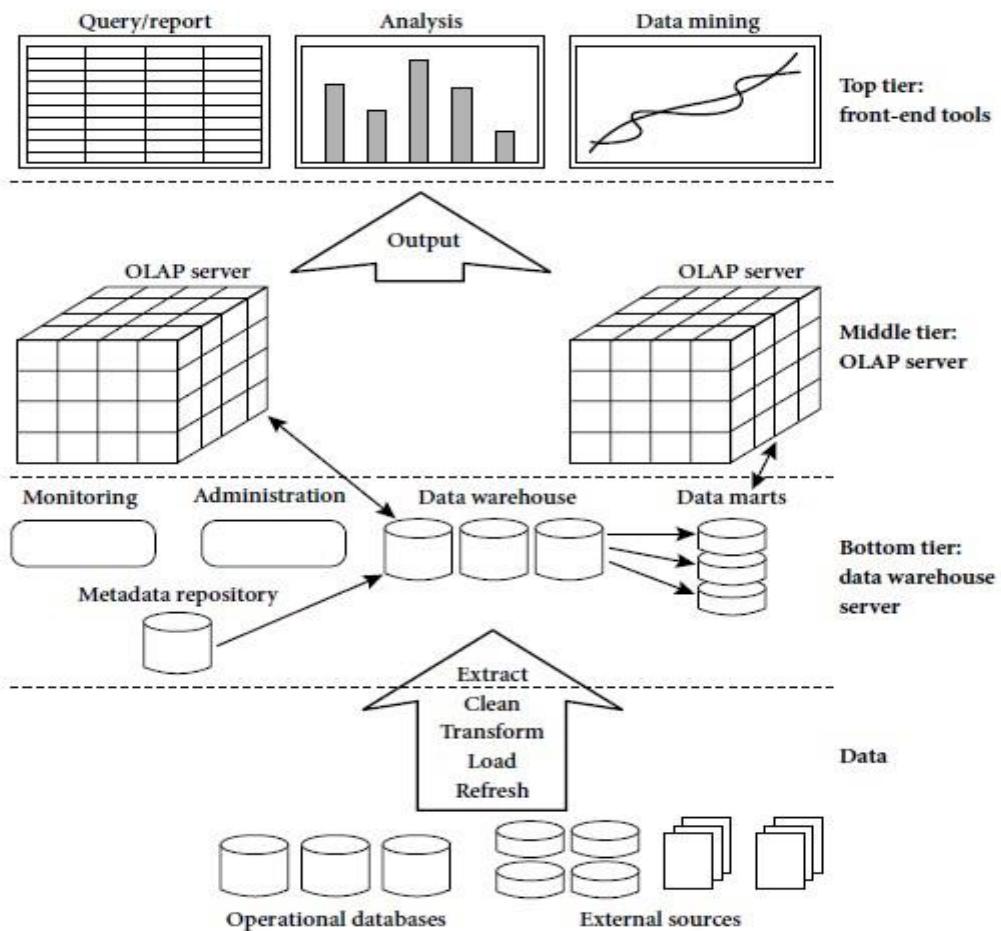
A data warehouse can be built using a *top-down approach*, a *bottom-up approach*, or a *combination of both*.

- The top-down approach starts with the overall design and planning. It is useful in cases where the technology is mature and well known, and where the business problems that must be solved are clear and well understood.
- The bottom-up approach starts with experiments and prototypes. This is useful in the early stage of business modeling and technology development. It allows an organization to move forward at considerably less expense and to evaluate the benefits of the technology before making significant commitments.
- In the combined approach, an organization can exploit the planned and strategic nature of the top-down approach while retaining the rapid implementation and opportunistic application of the bottom-up approach.

The warehouse design process consists of the following steps:

- Choose a business process to model, for example, orders, invoices, shipments, inventory, account administration, sales, or the general ledger. If the business process is organizational and involves multiple complex object collections, a data warehouse model should be followed. However, if the process is departmental and focuses on the analysis of one kind of business process, a data mart model should be chosen.
- Choose the grain of the business process. The grain is the fundamental, atomic level of data to be represented in the fact table for this process, for example, individual transactions, individual daily snapshots, and so on.
- Choose the dimensions that will apply to each fact table record. Typical dimensions are time, item, customer, supplier, warehouse, transaction type, and status.
- Choose the measures that will populate each fact table record. Typical measures are numeric additive quantities like dollars sold and units sold.

1.9.2 A Three Tier Data Warehouse Architecture:



Tier-1:

The bottom tier is a warehouse database server that is almost always a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources (such as customer profile information provided by external consultants). These tools and utilities perform data extraction, cleaning, and transformation (e.g., to merge similar data from different sources into a unified format), as well as load and refresh functions to update the data warehouse. The data are extracted using application program interfaces known as gateways. A gateway is supported by the underlying DBMS and allows client programs to generate SQL code to be executed at a server.

Examples of gateways include ODBC (Open Database Connection) and OLEDB (Open Linking and Embedding for Databases) by Microsoft and JDBC (Java Database Connection).

This tier also contains a metadata repository, which stores information about the data warehouse and its contents.

Tier-2:

The middle tier is an OLAP server that is typically implemented using either a relational OLAP (ROLAP) model or a multidimensional OLAP.

- OLAP model is an extended relational DBMS that maps operations on multidimensional data to standard relational operations.
- A multidimensional OLAP (MOLAP) model, that is, a special-purpose server that directly implements multidimensional data and operations.

Tier-3:

The top tier is a front-end client layer, which contains query and reporting tools, analysis tools, and/or data mining tools (e.g., trend analysis, prediction, and so on).

1.9.3 Data Warehouse Models:

There are three data warehouse models.

1. Enterprise warehouse:

- An enterprise warehouse collects all of the information about subjects spanning the entire organization.
- It provides corporate-wide data integration, usually from one or more operational systems or external information providers, and is cross-functional in scope.
- It typically contains detailed data as well as summarized data, and can range in size from a few gigabytes to hundreds of gigabytes, terabytes, or beyond.
- An enterprise data warehouse may be implemented on traditional mainframes, computer superservers, or parallel architecture platforms. It requires extensive business modeling and may take years to design and build.

2. Data mart:

- A data mart contains a subset of corporate-wide data that is of value to a specific group of users. The scope is confined to specific selected subjects. For example, a marketing data mart may confine its subjects to customer, item, and sales. The data contained in data marts tend to be summarized.
- Data marts are usually implemented on low-cost departmental servers that are UNIX/LINUX- or Windows-based. The implementation cycle of a data mart is more likely to be measured in weeks rather than months or years. However, it may involve complex integration in the long run if its design and planning were not enterprise-wide.
- Depending on the source of data, data marts can be categorized as independent or dependent. Independent data marts are sourced from data captured from one or more operational systems or external information providers, or from data generated locally within a particular

department or geographic area. Dependent data marts are sourced directly from enterprise data warehouses.

3. Virtual warehouse:

- A virtual warehouse is a set of views over operational databases. For efficient query processing, only some of the possible summary views may be materialized.
- A virtual warehouse is easy to build but requires excess capacity on operational database servers.

1.9.4 Meta Data Repository:

Metadata are data about data. When used in a data warehouse, metadata are the data that define warehouse objects. Metadata are created for the data names and definitions of the given warehouse. Additional metadata are created and captured for timestamping any extracted data, the source of the extracted data, and missing fields that have been added by data cleaning or integration processes.

A metadata repository should contain the following:

- A description of the structure of the data warehouse, which includes the warehouse schema, view, dimensions, hierarchies, and derived data definitions, as well as data mart locations and contents.
- Operational metadata, which include data lineage (history of migrated data and the sequence of transformations applied to it), currency of data (active, archived, or purged), and monitoring information (warehouse usage statistics, error reports, and audit trails).

- The algorithms used for summarization, which include measure and dimension definitionalgorithms, data on granularity, partitions, subject areas, aggregation, summarization, and predefined queries and reports.
- The mapping from the operational environment to the data warehouse, which includessource databases and their contents, gateway descriptions, data partitions, data extraction, cleaning, transformation rules and defaults, data refresh and purging rules, andsecurity (user authorization and access control).
- Data related to system performance, which include indices and profiles that improovedata access and retrieval performance, in addition to rules for the timing and scheduling of refresh, update, and replication cycles.
- Business metadata, which include business terms and definitions, data ownershipinformation, and charging policies.

1.10 OLAP(Online analytical Processing):

- OLAP is an approach to answering multi-dimensional analytical (MDA) queries swiftly.
- OLAP is part of the broader category of business intelligence, which also encompasses relational database, report writing and data mining.
- OLAP tools enable users to analyze multidimensional data interactively from multiple perspectives.

OLAP consists of three basic analytical operations:

- Consolidation (Roll-Up)
- Drill-Down

➤ Slicing And Dicing

- Consolidation involves the aggregation of data that can be accumulated and computed in one or more dimensions. For example, all sales offices are rolled up to the sales department or sales division to anticipate sales trends.
- The drill-down is a technique that allows users to navigate through the details. For instance, users can view the sales by individual products that make up a region's sales.
- Slicing and dicing is a feature whereby users can take out (slicing) a specific set of data of the OLAP cube and view (dicing) the slices from different viewpoints.

1.10.1 Types of OLAP:

1. Relational OLAP (ROLAP):

- ROLAP works directly with relational databases. The base data and the dimension tables are stored as relational tables and new tables are created to hold the aggregated information. It depends on a specialized schema design.
- This methodology relies on manipulating the data stored in the relational database to give the appearance of traditional OLAP's slicing and dicing functionality. In essence, each action of slicing and dicing is equivalent to adding a "WHERE" clause in the SQL statement.
- ROLAP tools do not use pre-calculated data cubes but instead pose the query to the standard relational database and its tables in order to bring back the data required to answer the question.

- ROLAP tools feature the ability to ask any question because the methodology does not limit to the contents of a cube. ROLAP also has the ability to drill down to the lowest level of detail in the database.

2. Multidimensional OLAP (MOLAP):

- MOLAP is the 'classic' form of OLAP and is sometimes referred to as just OLAP.
- MOLAP stores this data in an optimized multi-dimensional array storage, rather than in a relational database. Therefore it requires the pre-computation and storage of information in the cube - the operation known as processing.
- MOLAP tools generally utilize a pre-calculated data set referred to as a data cube.

The data cube contains all the possible answers to a given range of questions.

- MOLAP tools have a very fast response time and the ability to quickly write back data into the data set.

3. Hybrid OLAP (HOLAP):

- There is no clear agreement across the industry as to what constitutes Hybrid OLAP, except that a database will divide data between relational and specialized storage.
- For example, for some vendors, a HOLAP database will use relational tables to hold the larger quantities of detailed data, and use specialized storage for at least some aspects of the smaller quantities of more-aggregate or less-detailed data.
- HOLAP addresses the shortcomings of MOLAP and ROLAP by combining the capabilities of both approaches.

- HOLAP tools can utilize both pre-calculated cubes and relational data sources.

1.11 Data Preprocessing:

1.11.1 Data Integration:

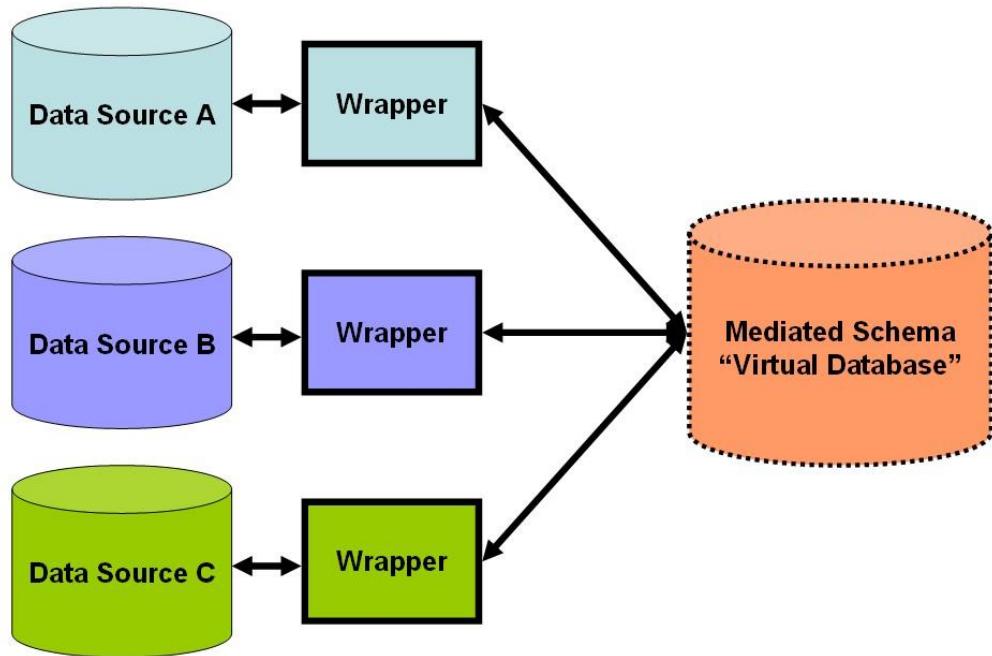
It combines data from multiple sources into a coherent data store, as in data warehousing. These sources may include multiple databases, data cubes, or flat files.

The data integration systems are formally defined as triple $\langle G, S, M \rangle$

Where G: The global schema

S: Heterogeneous source of schemas

M: Mapping between the queries of source and global schema



1.11.2 Issues in Data integration:

1. Schema integration and object matching:

How can the data analyst or the computer be sure that customer id in one database and customer number in another reference to the same attribute.

2. Redundancy:

An attribute (such as annual revenue, for instance) may be redundant if it can be derived from another attribute or set of attributes. Inconsistencies in attribute or dimension naming can also cause redundancies in the resulting data set.

3. detection and resolution of data value conflicts:

For the same real-world entity, attribute values from different sources may differ.

1.11.3 Data Transformation:

In data transformation, the data are transformed or consolidated into forms appropriate for mining.

Data transformation can involve the following:

- **Smoothing**, which works to remove noise from the data. Such techniques include binning, regression, and clustering.
- **Aggregation**, where summary or aggregation operations are applied to the data. For example, the daily sales data may be aggregated so as to compute monthly and annual total amounts. This step is typically used in constructing a data cube for analysis of the data at multiple granularities.
- **Generalization of the data**, where low-level or —primitive— (raw) data are replaced by higher-level concepts through the use of concept hierarchies. For example, categorical attributes, like street, can be generalized to higher-level concepts, like city or country.
- **Normalization**, where the attribute data are scaled so as to fall within a small specified range, such as 1:0 to 1:0, or 0:0 to 1:0.
- **Attribute construction** (or feature construction), where new attributes are constructed and added from the given set of attributes to help the mining process.

1.11.4 Data Reduction:

Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results.

Strategies for data reduction include the following:

- **Data cube aggregation**, where aggregation operations are applied to the data in the construction of a data cube.
- **Attribute subset selection**, where irrelevant, weakly relevant, or redundant attributes or dimensions may be detected and removed.
- **Dimensionality reduction**, where encoding mechanisms are used to reduce the dataset size.
- **Numerosity reduction**, where the data are replaced or estimated by alternative, smaller data representations such as parametric models (which need to store only the model parameters instead of the actual data) or nonparametric methods such as clustering, sampling, and the use of histograms.
- **Discretization and concept hierarchy generation**, where raw data values for attributes are replaced by ranges or higher conceptual levels. Data discretization is a form of numerosity reduction that is very useful for the automatic generation of concept hierarchies. Discretization and concept hierarchy generation are powerful tools for datamining, in that they allow the mining of data at multiple levels of abstraction.

Chapter-2

2.1 Association Rule Mining:

- Association rule mining is a popular and well researched method for discovering interesting relations between variables in large databases.
- It is intended to identify strong rules discovered in databases using different measures of interestingness.
- Based on the concept of strong rules, Rakesh Agrawal et al. introduced association rules.

Problem Definition:

The problem of association rule mining is defined as:

Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of n binary attributes called *items*.

Let $D = \{t_1, t_2, \dots, t_m\}$ be a set of transactions called the *database*.

Each transaction in D has a unique transaction ID and contains a subset of the items in I .

A *rule* is defined as an implication of the form $X \Rightarrow Y$ where

$X, Y \subseteq I$ and $X \cap Y = \emptyset$

The sets of items (for short *itemsets*) X and Y are called *antecedent* (left-hand-side or LHS) and *consequent* (right-hand-side or RHS) of the rule respectively.

Example:

To illustrate the concepts, we use a small example from the supermarket domain. The set of items is $I = \{\text{milk}, \text{bread}, \text{butter}, \text{beer}\}$ and a small database containing the items (1 codes presence and 0 absence of an item in a transaction) is shown in the table.

An example rule for the supermarket could be $\{\text{butter}, \text{bread}\} \Rightarrow \{\text{milk}\}$ meaning that if butter and bread are bought, customers also buy milk.

Example database with 4 items and 5 transactions

Transaction ID	milk	bread	butter	beer
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0

2.1.1 Important concepts of Association Rule Mining:

- The **support** $\text{supp}(X)$ of an itemset X is defined as the proportion of transactions in the data set which contain the itemset. In the example database, the itemset $\{\text{milk}, \text{bread}, \text{butter}\}$ has a support of $1/5 = 0.2$ since it occurs in 20% of all transactions (1 out of 5 transactions).

- The **confidence** of a rule is defined

$$\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y) / \text{supp}(X).$$

For example, the rule $\{\text{butter}, \text{bread}\} \Rightarrow \{\text{milk}\}$ has a confidence of $0.2/0.2 = 1.0$ in the database, which means that for 100% of the transactions containing butter and bread the rule is correct (100% of the times a customer buys butter and bread, milk is bought as well). Confidence can be interpreted as an estimate of the probability $P(Y|X)$, the probability of finding the RHS of the rule in transactions under the condition that these transactions also contain the LHS.

- The **lift** of a rule is defined as

$$\text{lift}(X \Rightarrow Y) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X) \times \text{supp}(Y)}$$

or the ratio of the observed support to that expected if X and Y were independent. The

rule $\{\text{milk}, \text{bread}\} \Rightarrow \{\text{butter}\}$ has a lift of $\frac{0.2}{0.4 \times 0.4} = 1.25$.

- The **conviction** of a rule is defined as

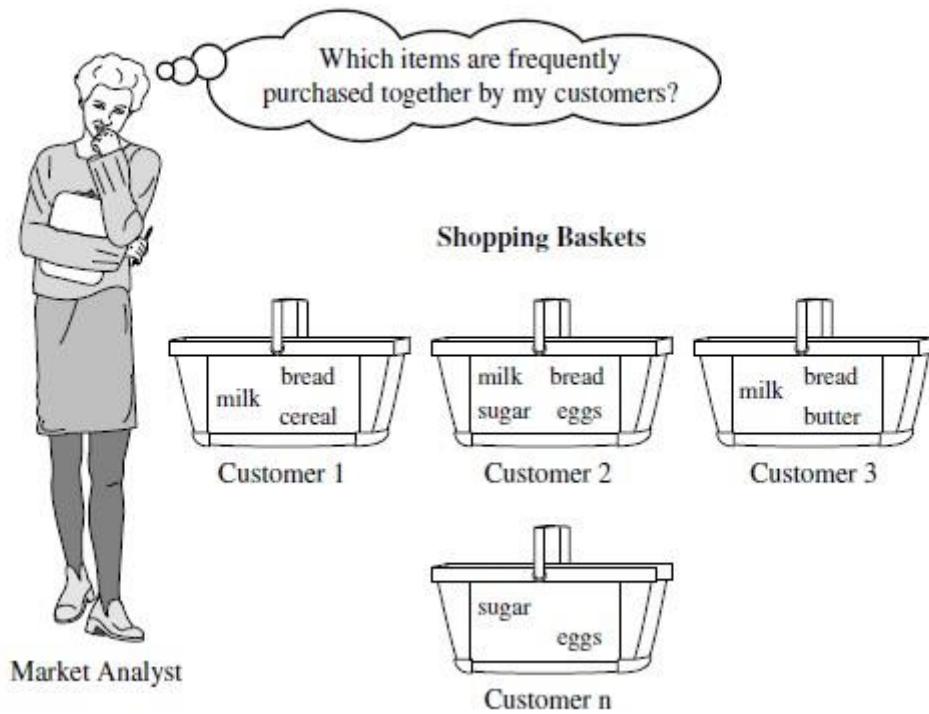
$$\text{conv}(X \Rightarrow Y) = \frac{1 - \text{supp}(Y)}{1 - \text{conf}(X \Rightarrow Y)}$$

The rule $\{\text{milk}, \text{bread}\} \Rightarrow \{\text{butter}\}$ has a conviction of $\frac{1 - 0.4}{1 - .5} = 1.2$,

and can be interpreted as the ratio of the expected frequency that X occurs without Y (that is to say, the frequency that the rule makes an incorrect prediction) if X and Y were independent divided by the observed frequency of incorrect predictions.

2.2 Market basket analysis:

This process analyzes customer buying habits by finding associations between the different items that customers place in their shopping baskets. The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers. For instance, if customers are buying milk, how likely are they to also buy bread (and what kind of bread) on the same trip to the supermarket. Such information can lead to increased sales by helping retailers do selective marketing and plan their shelf space.



Example:

If customers who purchase computers also tend to buy antivirus software at the same time, then placing the hardware display close to the software display may help increase the sales of both items.

In an alternative strategy, placing hardware and software at opposite ends of the store may entice customers who purchase such items to pick up other items along the way. For instance, after deciding on an expensive computer, a customer may observe security systems for sale while heading toward the software display to purchase antivirus software and may decide to purchase a home security system as well. Market basket analysis can also help retailers plan which items to put on sale at reduced prices. If customers tend to purchase computers and printers together, then having a sale on printers may encourage the sale of printers *as well as* computers.

2.3 Frequent Pattern Mining:

Frequent pattern mining can be classified in various ways, based on the following criteria:

1. Based on the completeness of patterns to be mined:

- We can mine the complete set of frequent itemsets, the closed frequent itemsets, and the maximal frequent itemsets, given a minimum support threshold.
- We can also mine constrained frequent itemsets, approximate frequent itemsets, nearmatch frequent itemsets, top-k frequent itemsets and so on.

2. Based on the levels of abstraction involved in the rule set:

Some methods for association rule mining can find rules at differing levels of abstraction.

For example, suppose that a set of association rules mined includes the following rules where X is a variable representing a customer:

$$\text{buys}(X, \text{---computer} \parallel) \Rightarrow \text{buys}(X, \text{---HP printer} \parallel) \quad (1)$$

$$\text{buys}(X, \text{---laptop computer} \parallel) \Rightarrow \text{buys}(X, \text{---HP printer} \parallel) \quad (2)$$

In rule (1) and (2), the items bought are referenced at different levels of abstraction (e.g., $\text{---computer} \parallel$ is a higher-level abstraction of $\text{---laptop computer} \parallel$).

3. Based on the number of data dimensions involved in the rule:

- If the items or attributes in an association rule reference only one dimension, then it is a single-dimensional association rule.

$\text{buys}(X, \neg\text{computer} \parallel) \Rightarrow \text{buys}(X, \neg\text{antivirus software} \parallel)$

- If a rule references two or more dimensions, such as the dimensions age, income, and buys, then it is a multidimensional association rule. The following rule is an example of a multidimensional rule: $\text{age}(X, \neg 30, 31 \dots 39 \parallel) \wedge \text{income}(X, \neg 42K, \dots 48K \parallel) \Rightarrow \text{buys}(X, \neg\text{high resolution TV} \parallel)$

4. Based on the types of values handled in the rule:

- If a rule involves associations between the presence or absence of items, it is a Boolean association rule.
- If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule.

5. Based on the kinds of rules to be mined:

- Frequent pattern analysis can generate various kinds of rules and other interesting relationships.
- Association rule mining can generate a large number of rules, many of which are redundant or do not indicate a correlation relationship among itemsets.
- The discovered associations can be further analyzed to uncover statistical correlations, leading to correlation rules.

6. Based on the kinds of patterns to be mined:

- Many kinds of frequent patterns can be mined from different kinds of data sets.
- Sequential pattern mining searches for frequent subsequences in a sequence data set, where a sequence records an ordering of events.

- For example, with sequential pattern mining, we can study the order in which items are frequently purchased. For instance, customers may tend to first buy a PC, followed by a digital camera, and then a memory card.
- Structured pattern mining searches for frequent substructures in a structured data set.
- Single items are the simplest form of structure.
- Each element of an itemset may contain a subsequence, a subtree, and so on.
- Therefore, structured pattern mining can be considered as the most general form of frequent pattern mining.

2.4 Efficient Frequent Itemset Mining Methods:

2.4.1 Finding Frequent Itemsets Using Candidate Generation: The Apriori Algorithm

- Apriori is a seminal algorithm proposed by R. Agrawal and R. Srikant in 1994 for mining frequent itemsets for Boolean association rules.
- The name of the algorithm is based on the fact that the algorithm uses *prior knowledge* of frequent itemset properties.
- Apriori employs an iterative approach known as a *level-wise* search, where k -itemsets are used to explore $(k+1)$ -itemsets.
- First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L_1 . Next, L_1 is used to find L_2 , the set of frequent 2-

itemsets, which is used to find L_3 , and so on, until no more frequent k -itemsets can be found.

- The finding of each L_k requires one full scan of the database.
- A two-step process is followed in Apriori consisting of join and prune action.

Algorithm: Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.

Input:

- D , a database of transactions;
- min_sup , the minimum support count threshold.

Output: L , frequent itemsets in D .

Method:

```

(1)   L1 = find_frequent_1-itemsets(D);
(2)   for (k = 2; Lk-1 ≠ φ; k++) {
(3)       Ck = apriori_gen(Lk-1);
(4)       for each transaction t ∈ D { // scan D for counts
(5)           Ct = subset(Ck, t); // get the subsets of t that are candidates
(6)           for each candidate c ∈ Ct
(7)               c.count++;
(8)           }
(9)       Lk = {c ∈ Ck | c.count ≥ min_sup}
(10)    }
(11)   return L = ∪k Lk;

procedure apriori_gen(Lk-1: frequent (k - 1)-itemsets)
(1)   for each itemset I1 ∈ Lk-1
(2)       for each itemset I2 ∈ Lk-1
(3)           if (I1[1] = I2[1]) ∧ (I1[2] = I2[2]) ∧ ... ∧ (I1[k - 2] = I2[k - 2]) ∧ (I1[k - 1] < I2[k - 1]) then {
(4)               c = I1 × I2; // join step: generate candidates
(5)               if has_infrequent_subset(c, Lk-1) then
(6)                   delete c; // prune step: remove unfruitful candidate
(7)               else add c to Ck;
(8)           }
(9)       return Ck;

procedure has_infrequent_subset(c: candidate k-itemset;
                               Lk-1: frequent (k - 1)-itemsets); // use prior knowledge
(1)   for each (k - 1)-subset s of c
(2)       if s ∉ Lk-1 then
(3)           return TRUE;
(4)       return FALSE;

```

Example:

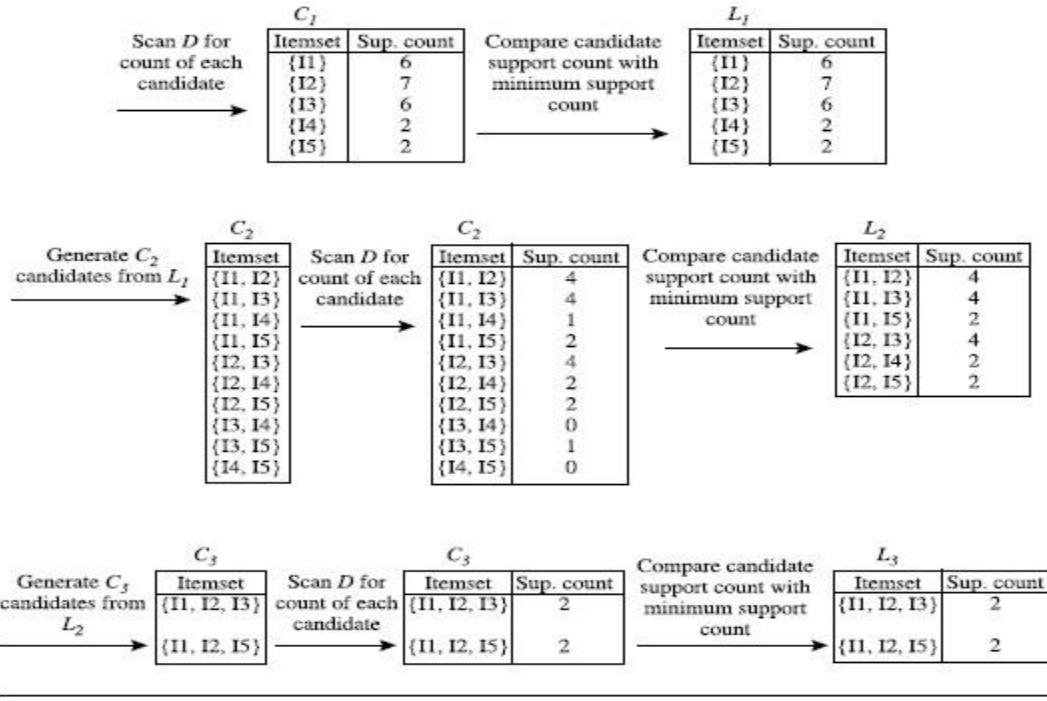
TID	List of item IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3

T800	I1, I2, I3, I5
T900	I1, I2, I3

There are nine transactions in this database, that is, $|D| = 9$.

Steps:

1. In the first iteration of the algorithm, each item is a member of the set of candidate1itemsets, C1. The algorithm simply scans all of the transactions in order to countthe number of occurrences of each item.
2. Suppose that the minimum support count required is 2, that is, $\text{min sup} = 2$. The set of frequent 1-itemsets, L1, can thenbe determined. It consists of the candidate 1-itemsets satisfying minimum support.In our example, all of the candidates in C1 satisfy minimum support.
3. To discover the set of frequent 2-itemsets, L2, the algorithm uses the join L1 on L1 togenerate a candidate set of 2-itemsets, C2.No candidates are removed fromC2 during the prune step because each subset of thecandidates is also frequent.
- 4.Next, the transactions inDare scanned and the support count of each candidate itemsetInC2 is accumulated.
5. The set of frequent 2-itemsets, L2, is then determined, consisting of those candidate2itemsets in C2 having minimum support.
6. The generation of the set of candidate 3-itemsets,C3, Fromthejoin step, we first get $C3 = L2 \times L2 = (\{I1, I2, I3\}, \{I1, I2, I5\}, \{I1, I3, I5\}, \{I2, I3, I4\}, \{I2, I3, I5\}, \{I2, I4, I5\})$. Based on the Apriori property that all subsets of a frequentitemsetmust also be frequent, we can determine that the four latter candidates cannotpossibly be frequent.
- 7.The transactions in D are scanned in order to determine L3, consisting of those candidate 3-itemsets in C3 having minimum support.
- 8.The algorithm uses $L3 \times L3$ to generate a candidate set of 4-itemsets, C4.



Generation of candidate itemsets and frequent itemsets, where the minimum support count is 2.

- (a) Join: $C_3 = L_2 \bowtie L_2 = \{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\} \bowtie \{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\} = \{\{I1, I2, I3\}, \{I1, I2, I5\}, \{I1, I3, I5\}, \{I2, I3, I4\}, \{I2, I3, I5\}, \{I2, I4, I5\}\}.$
- (b) Prune using the Apriori property: All nonempty subsets of a frequent itemset must also be frequent. Do any of the candidates have a subset that is not frequent?
 - The 2-item subsets of $\{I1, I2, I3\}$ are $\{I1, I2\}$, $\{I1, I3\}$, and $\{I2, I3\}$. All 2-item subsets of $\{I1, I2, I3\}$ are members of L_2 . Therefore, keep $\{I1, I2, I3\}$ in C_3 .
 - The 2-item subsets of $\{I1, I2, I5\}$ are $\{I1, I2\}$, $\{I1, I5\}$, and $\{I2, I5\}$. All 2-item subsets of $\{I1, I2, I5\}$ are members of L_2 . Therefore, keep $\{I1, I2, I5\}$ in C_3 .
 - The 2-item subsets of $\{I1, I3, I5\}$ are $\{I1, I3\}$, $\{I1, I5\}$, and $\{I3, I5\}$. $\{I3, I5\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I1, I3, I5\}$ from C_3 .
 - The 2-item subsets of $\{I2, I3, I4\}$ are $\{I2, I3\}$, $\{I2, I4\}$, and $\{I3, I4\}$. $\{I3, I4\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I2, I3, I4\}$ from C_3 .
 - The 2-item subsets of $\{I2, I3, I5\}$ are $\{I2, I3\}$, $\{I2, I5\}$, and $\{I3, I5\}$. $\{I3, I5\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I2, I3, I5\}$ from C_3 .
 - The 2-item subsets of $\{I2, I4, I5\}$ are $\{I2, I4\}$, $\{I2, I5\}$, and $\{I4, I5\}$. $\{I4, I5\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I2, I4, I5\}$ from C_3 .
- (c) Therefore, $C_3 = \{\{I1, I2, I3\}, \{I1, I2, I5\}\}$ after pruning.

Generation and pruning of candidate 3-itemsets, C_3 , from L_2 using the Apriori property.

2.4.2 Generating Association Rules from Frequent Itemsets:

Once the frequent itemsets from transactions in a database D have been found, it is straightforward to generate strong association rules from them.

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{\text{support_count}(A \cup B)}{\text{support_count}(A)}.$$

The conditional probability is expressed in terms of itemset support count, where $\text{support_count}(A \cup B)$ is the number of transactions containing the itemsets $A \cup B$, and $\text{support_count}(A)$ is the number of transactions containing the itemset A . Based on this equation, association rules can be generated as follows:

- For each frequent itemset I , generate all nonempty subsets of I .
- For every nonempty subset s of I , output the rule “ $s \Rightarrow (I - s)$ ” if $\frac{\text{support_count}(I)}{\text{support_count}(s)} \geq \text{min_conf}$, where min_conf is the minimum confidence threshold.

Example:

Generating association rules. Let's try an example based on the transactional data for *AllElectronics* shown in Table 5.1. Suppose the data contain the frequent itemset $I = \{I1, I2, I5\}$. What are the association rules that can be generated from I ? The nonempty subsets of I are $\{I1, I2\}$, $\{I1, I5\}$, $\{I2, I5\}$, $\{I1\}$, $\{I2\}$, and $\{I5\}$. The resulting association rules are as shown below, each listed with its confidence:

$I1 \wedge I2 \Rightarrow I5,$	$\text{confidence} = 2/4 = 50\%$
$I1 \wedge I5 \Rightarrow I2,$	$\text{confidence} = 2/2 = 100\%$
$I2 \wedge I5 \Rightarrow I1,$	$\text{confidence} = 2/2 = 100\%$
$I1 \Rightarrow I2 \wedge I5,$	$\text{confidence} = 2/6 = 33\%$
$I2 \Rightarrow I1 \wedge I5,$	$\text{confidence} = 2/7 = 29\%$
$I5 \Rightarrow I1 \wedge I2,$	$\text{confidence} = 2/2 = 100\%$

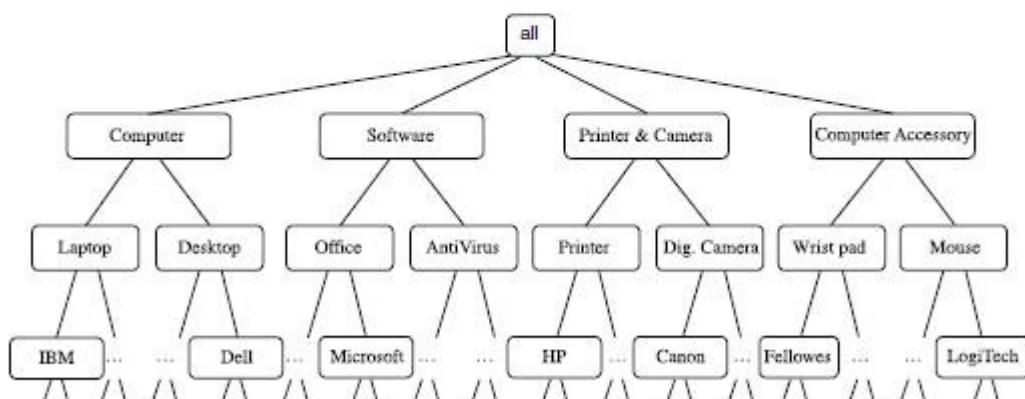
2.5 Mining Multilevel Association Rules:

- For many applications, it is difficult to find strong associations among data items at low or primitive levels of abstraction due to the sparsity of data at those levels.
- Strong associations discovered at high levels of abstraction may represent commonsense knowledge.
- Therefore, data mining systems should provide capabilities for mining association rules at multiple levels of abstraction, with sufficient flexibility for easy traversal among different abstraction spaces.

- Association rules generated from mining data at multiple levels of abstraction are called multiple-level or multilevel association rules.
- Multilevel association rules can be mined efficiently using concept hierarchies under a support-confidence framework.
- In general, a top-down strategy is employed, where counts are accumulated for the calculation of frequent itemsets at each concept level, starting at the concept level 1 and working downward in the hierarchy toward the more specific concept levels, until no more frequent itemsets can be found.

A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts. Data can be generalized by replacing low-level concepts within the data by their higher-level concepts, or ancestors, from a concept hierarchy.

<i>TID</i>	<i>Items Purchased</i>
T100	IBM-ThinkPad-T40/2373, HP-Photosmart-7660
T200	Microsoft-Office-Professional-2003, Microsoft-Plus!-Digital-Media
T300	Logitech-MX700-Cordless-Mouse, Fellowes-Wrist-Rest
T400	Dell-Dimension-XPS, Canon-PowerShot-S400
T500	IBM-ThinkPad-R40/P4M, Symantec-Norton-Antivirus-2003
...	...



A concept hierarchy for *AllElectronics* computer items.

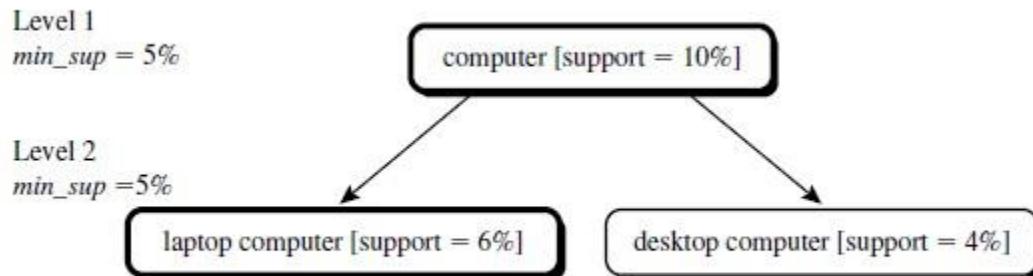
The concept hierarchy has five levels, respectively referred to as levels 0 to 4, starting with level 0 at the root node for all.

- Here, Level 1 includes computer, software, printer&camera, and computer accessory.
- Level 2 includes laptop computer, desktop computer, office software, antivirus software
- Level 3 includes IBM desktop computer, . . . , Microsoft office software, and so on.
- Level 4 is the most specific abstraction level of this hierarchy.

2.5.1 Approaches ForMining Multilevel Association Rules:

1.UniformMinimum Support:

- The same minimum support threshold is used when mining at each level of abstraction.
- When a uniform minimum support threshold is used, the search procedure is simplified.
- The method is also simple in that users are required to specify only one minimum support threshold.
- The uniform support approach, however, has some difficulties. It is unlikely that items at lower levels of abstraction will occur as frequently as those at higher levels of abstraction.
- If the minimum support threshold is set too high, it could miss some meaningful associations occurring at low abstraction levels. If the threshold is set too low, it may generate many uninteresting associations occurring at high abstraction levels.

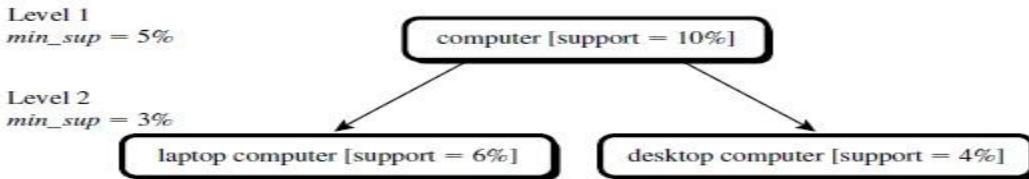


Multilevel mining with uniform support.

2.Reduced Minimum Support:

- Each level of abstraction has its own minimum support threshold.
- The deeper the level of abstraction, the smaller the corresponding threshold is.

- For example, the minimum support thresholds for levels 1 and 2 are 5% and 3%, respectively. In this way, —computer,|| —laptop computer,|| and —desktop computer|| are all considered frequent.



3. Group-Based Minimum Support:

- Because users or experts often have insight as to which groups are more important than others, it is sometimes more desirable to set up user-specific, item, or group based minimal support thresholds when mining multilevel rules.
- For example, a user could set up the minimum support thresholds based on product price, or on items of interest, such as by setting particularly low support thresholds for laptop computers and flash drives in order to pay particular attention to the association patterns containing items in these categories.

2.6 Mining Multidimensional Association Rules from Relational Databases and Data Warehouses:

- Single dimensional or intradimensional association rule contains a single distinct predicate (e.g., buys) with multiple occurrences i.e., the predicate occurs more than once within the rule.

buys(X, —digital camera)||=>buys(X, —HP printer)||

- Association rules that involve two or more dimensions or predicates can be referred to as multidimensional association rules.

age(X, “20...29”) ^ occupation(X, “student”) => buys(X, “laptop”)

- Above Rule contains three predicates (age, occupation, and buys), each of which occurs only once in the rule. Hence, we say that it has no repeated predicates.
- Multidimensional association rules with no repeated predicates are called interdimensional association rules.
- We can also mine multidimensional association rules with repeated predicates, which contain multiple occurrences of some predicates. These rules are called hybrid multidimensional association rules. An example of such a rule is the following, where the predicate buys is repeated: $\text{age}(X, -20\dots29) \wedge \text{buys}(X, -\text{laptop}) \Rightarrow \text{buys}(X, -\text{HP printer})$

2.7 Mining Quantitative Association Rules:

- Quantitative association rules are multidimensional association rules in which the numeric attributes are *dynamically* discretized during the mining process so as to satisfy some mining criteria, such as maximizing the confidence or compactness of the rules mined.
- In this section, we focus specifically on how to mine quantitative association rules having two quantitative attributes on the left-hand side of the rule and one categorical attribute on the right-hand side of the rule. That is

$Aquan1 \wedge Aquan2 \Rightarrow Acat$ where $Aquan1$ and $Aquan2$ are tests on quantitative attribute interval $Acat$ tests a categorical attribute from the task-relevant data.

- Such rules have been referred to as two-dimensional quantitative association rules, because they contain two quantitative dimensions.
- For instance, suppose you are curious about the association relationship between pairs of quantitative attributes, like customer age and income, and the type of television (such as *high-definition TV*, i.e., *HDTV*) that customers like to buy. An example of such a 2-D quantitative association rule is $\text{age}(X, -30\dots39) \wedge \text{income}(X, -42K\dots48K) \Rightarrow \text{buys}(X, -HDTV)$

2.8 From Association Mining to Correlation Analysis:

- A correlation measure can be used to augment the support-confidence framework for association rules. This leads to *correlation rules* of the form
 $A \Rightarrow B$ [support, confidence, correlation]
- That is, a correlation rule is measured not only by its support and confidence but also by the correlation between itemsets A and B . There are many different correlation measures from which to choose. In this section, we study various correlation measures to determine which would be good for mining large data sets.
- Lift is a simple correlation measure that is given as follows. The occurrence of itemset A is independent of the occurrence of itemset B if $P(A \cup B) = P(A)P(B)$; otherwise, itemsets A and B are dependent and correlated as events. This definition can easily be extended to more than two itemsets.

The lift between the occurrence of A and B can be measured by computing

$$\text{lift}(A, B) = \frac{P(A \cup B)}{P(A)P(B)}.$$

- If the $\text{lift}(A, B)$ is less than 1, then the occurrence of A is negatively correlated with the occurrence of B .
- If the resulting value is greater than 1, then A and B are positively correlated, meaning that the occurrence of one implies the occurrence of the other.
- If the resulting value is equal to 1, then A and B are independent and there is no correlation between them.

Chapter-3

3.1 Classification and Prediction:

- Classification and prediction are two forms of data analysis that can be used to extract models describing important data classes or to predict future data trends.
- Classification predicts categorical (discrete, unordered) labels, *prediction* models continuous valued functions.
- For example, we can build a classification model to categorize bank loan applications as either safe or risky, or a prediction model to predict the expenditures of potential customers on computer equipment given their income and occupation.
- A predictor is constructed that predicts a continuous-valued function, or ordered value, as opposed to a categorical label.
- Regression analysis is a statistical methodology that is most often used for numeric prediction.
- Many classification and prediction methods have been proposed by researchers in machine learning, pattern recognition, and statistics.
- Most algorithms are memory resident, typically assuming a small data size. Recent data mining research has built on such work, developing scalable classification and prediction techniques capable of handling large disk-resident data.

3.1.1 Issues Regarding Classification and Prediction:

1. Preparing the Data for Classification and Prediction:

The following preprocessing steps may be applied to the data to help improve the accuracy, efficiency, and scalability of the classification or prediction process.

(i) Data cleaning:

- This refers to the preprocessing of data in order to remove or reduce *noise* (by applying smoothing techniques) and the treatment of *missing values* (e.g., by replacing a missing value

with the most commonly occurring value for that attribute, or with the most probable value based on statistics).

- Although most classification algorithms have some mechanisms for handling noisy or missing data, this step can help reduce confusion during learning.

(ii)Relevance analysis:

- Many of the attributes in the data may be *redundant*.
- Correlation analysis can be used to identify whether any two given attributes are statistically related.
- For example, a strong correlation between attributes A_1 and A_2 would suggest that one of the two could be removed from further analysis.
- A database may also contain *irrelevant* attributes. Attribute subset selection can be used in these cases to find a reduced set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes.
- Hence, relevance analysis, in the form of correlation analysis and attribute subset selection, can be used to detect attributes that do not contribute to the classification or prediction task.
- Such analysis can help improve classification efficiency and scalability.

(iii)Data Transformation And Reduction

- The data may be transformed by normalization, particularly when neural networks or methods involving distance measurements are used in the learning step.
- Normalization involves scaling all values for a given attribute so that they fall within a small specified range, such as -1 to +1 or 0 to 1.
- The data can also be transformed by *generalizing* it to higher-level concepts. Concept hierarchies may be used for this purpose. This is particularly useful for continuous valued attributes.
- For example, numeric values for the attribute *income* can be generalized to discrete ranges, such as *low*, *medium*, and *high*. Similarly, categorical attributes, like *street*, can be generalized to higher-level concepts, like *city*.
- Data can also be reduced by applying many other methods, ranging from wavelet transformation and principle components analysis to discretization techniques, such as binning, histogram analysis, and clustering.

3.1.2 Comparing Classification and Prediction Methods:

➤ **Accuracy:**

- The accuracy of a classifier refers to the ability of a given classifier to correctly predict the class label of new or previously unseen data (i.e., tuples without class label information).
- The accuracy of a predictor refers to how well a given predictor can guess the value of the predicted attribute for new or previously unseen data.

➤ **Speed:**

This refers to the computational costs involved in generating and using the given classifier or predictor.

➤ **Robustness:**

This is the ability of the classifier or predictor to make correct predictions given noisy data or data with missing values.

➤ **Scalability:**

This refers to the ability to construct the classifier or predictor efficiently given large amounts of data.

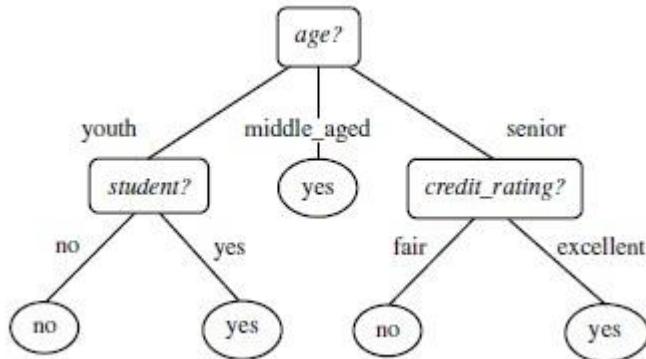
➤ **Interpretability:**

- This refers to the level of understanding and insight that is provided by the classifier or predictor.
- Interpretability is subjective and therefore more difficult to assess.

3.2 Classification by Decision Tree Induction:

- Decision tree induction is the learning of decision trees from class-labeled training tuples.
- A decision tree is a flowchart-like tree structure, where ➤ Each internal node denotes a test on an attribute.

- Each branch represents an outcome of the test.
- Each leaf node holds a class label.
- The topmost node in a tree is the root node.



- The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery.
- Decision trees can handle high dimensional data.
- Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans.
 - The learning and classification steps of decision tree induction are simple and fast.
 - In general, decision tree classifiers have good accuracy.
 - Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology.

3.2.1 Algorithm For Decision Tree Induction:

Algorithm: `Generate_decision_tree`. Generate a decision tree from the training tuples of data partition D .

Input:

- Data partition, D , which is a set of training tuples and their associated class labels;
- $attribute_list$, the set of candidate attributes;
- $Attribute_selection_method$, a procedure to determine the splitting criterion that “best” partitions the data tuples into individual classes. This criterion consists of a $splitting_attribute$ and, possibly, either a $split point$ or $splitting_subset$.

Output: A decision tree.

Method:

- (1) create a node N ;
- (2) If tuples in D are all of the same class, C then
 - (3) return N as a leaf node labeled with the class C ;
 - (4) If $attribute_list$ is empty then
 - (5) return N as a leaf node labeled with the majority class in D ; // majority voting
 - (6) apply $Attribute_selection_method(D, attribute_list)$ to find the “best” $splitting_criterion$;
 - (7) label node N with $splitting_criterion$;
 - (8) If $splitting_attribute$ is discrete-valued and
 - multiway splits allowed then // not restricted to binary trees
 - (9) $attribute_list \leftarrow attribute_list - splitting_attribute$; // remove $splitting_attribute$
- (10) for each outcome j of $splitting_criterion$
 - // partition the tuples and grow subtrees for each partition
 - (11) let D_j be the set of data tuples in D satisfying outcome j ; // a partition
 - (12) If D_j is empty then
 - (13) attach a leaf labeled with the majority class in D to node N ;
 - (14) else attach the node returned by $Generate_decision_tree(D_j, attribute_list)$ to node N ;
- (15) return N ;

The algorithm is called with three parameters:

- Data partition
- Attribute list
- Attribute selection method
 - The parameter attribute list is a list of attributes describing the tuples.
 - Attribute selection method specifies a heuristic procedure for selecting the attribute that —best|| discriminates the given tuples according to class.
 - The tree starts as a single node, N , representing the training tuples in D .

- If the tuples in D are all of the same class, then node N becomes a leaf and is labeled with that class.
- All of the terminating conditions are explained at the end of the algorithm.
- Otherwise, the algorithm calls Attribute selection method to determine the splitting criterion.
- The splitting criterion tells us which attribute to test at node N by determining the best way to separate or partition the tuples in D into individual classes.

There are three possible scenarios. Let A be the splitting attribute. A has v distinct values, $\{a_1, a_2, \dots, a_v\}$, based on the training data.

1 A is discrete-valued:

- In this case, the outcomes of the test at node N correspond directly to the known values of A .
- A branch is created for each known value, a_j , of A and labeled with that value.
- A need not be considered in any future partitioning of the tuples.

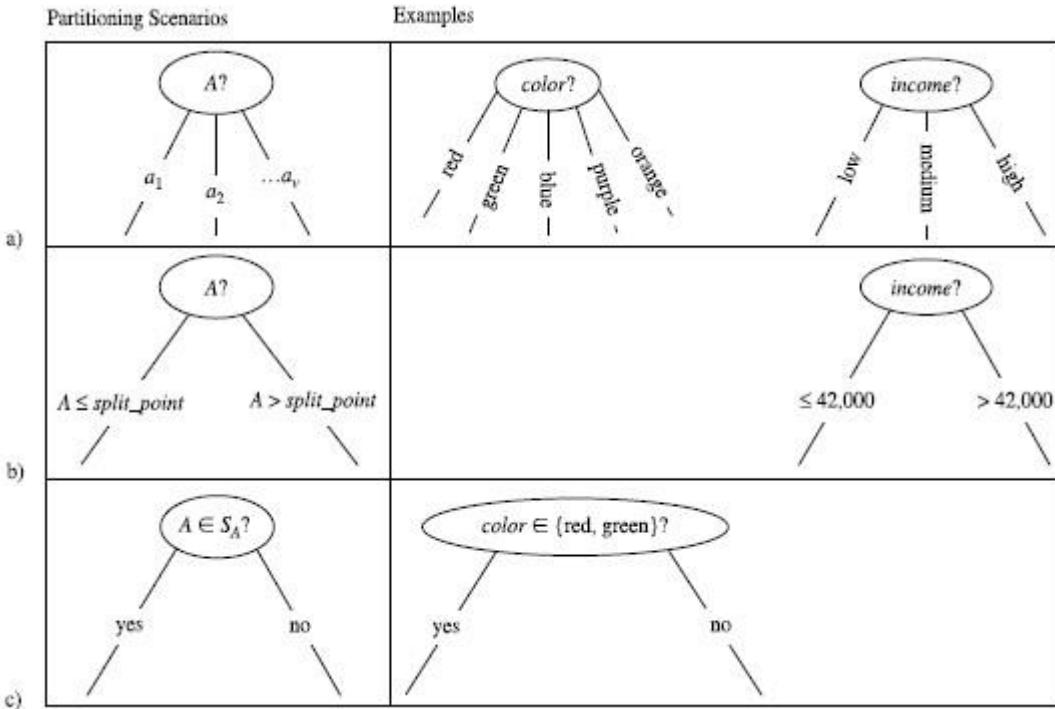
2 A is continuous-valued:

In this case, the test at node N has two possible outcomes, corresponding to the conditions $A \leq \text{split point}$ and $A > \text{split point}$, respectively where split point is the split-point returned by Attribute selection method as part of the splitting criterion.

3 A is discrete-valued and a binary tree must be produced:

The test at node N is of the form— $A \in SA?$.

SA is the splitting subset for A , returned by Attribute selection method as part of the splitting criterion. It is a subset of the known values of A .



(a) If A is Discrete valued (b) If A is continuous valued (c) If A is discrete-valued and a binary tree must be produced:

3.3 Bayesian Classification:

- Bayesian classifiers are statistical classifiers.
- They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.
- Bayesian classification is based on Bayes' theorem.

3.3.1 Bayes' Theorem:

- Let X be a data tuple. In Bayesian terms, X is considered —evidence.||and it is described by measurements made on a set of n attributes.

- Let H be some hypothesis, such as that the data tuple X belongs to a specified class C . For classification problems, we want to determine $P(H|X)$, the probability that the hypothesis H holds given the —evidence or observed data tuple X .
- $P(H|X)$ is the posterior probability, or a posteriori probability, of H conditioned on X .
- Bayes' theorem is useful in that it provides a way of calculating the posterior probability, $P(H|X)$, from $P(H)$, $P(X|H)$, and $P(X)$.

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}.$$

3.3.2 Naïve Bayesian Classification:

The naïve Bayesian classifier, or simple Bayesian classifier, works as follows:

1. Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n -dimensional attribute vector, $X = (x_1, x_2, \dots, x_n)$, depicting n measurements made on the tuple from n attributes, respectively, A_1, A_2, \dots, A_n .
2. Suppose that there are m classes, C_1, C_2, \dots, C_m . Given a tuple, X , the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X .

That is, the naïve Bayesian classifier predicts that tuple X belongs to the class C_i if and only if

$$P(C_i|X) > P(C_j|X) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

Thus we maximize $P(C_i|X)$. The class C_i for which $P(C_i|X)$ is maximized is called the maximum posterior hypothesis. By Bayes' theorem

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}.$$

3. As $P(X)$ is constant for all classes, only $P(X|C_i)P(C_i)$ need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, $P(C_1) = P(C_2) = \dots = P(C_m)$, and we would therefore maximize $P(X|C_i)$.

Otherwise, we maximize $P(X|C_i)P(C_i)$.

4. Given data sets with many attributes, it would be extremely computationally expensive to compute $P(X|C_i)$. In order to reduce computation in evaluating $P(X|C_i)$, the naive assumption

of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the tuple. Thus,

$$\begin{aligned} P(X|C_i) &= \prod_{k=1}^n P(x_k|C_i) \\ &= P(x_1|C_i) \times P(x_2|C_i) \times \cdots \times P(x_n|C_i). \end{aligned}$$

We can easily estimate the probabilities $P(x_1|C_i)$, $P(x_2|C_i)$, : : : , $P(x_n|C_i)$ from the training tuples. For each attribute, we look at whether the attribute is categorical or continuous-valued. For instance, to compute $P(X|C_i)$, we consider the following:

- If A_k is categorical, then $P(x_k|C_i)$ is the number of tuples of class C_i in D having the value x_k for A_k , divided by $|C_{i,D}|$ the number of tuples of class C_i in D .
- If A_k is continuous-valued, then we need to do a bit more work, but the calculation is pretty straightforward.

A continuous-valued attribute is typically assumed to have a Gaussian distribution with a mean μ and standard deviation σ , defined by

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}).$$

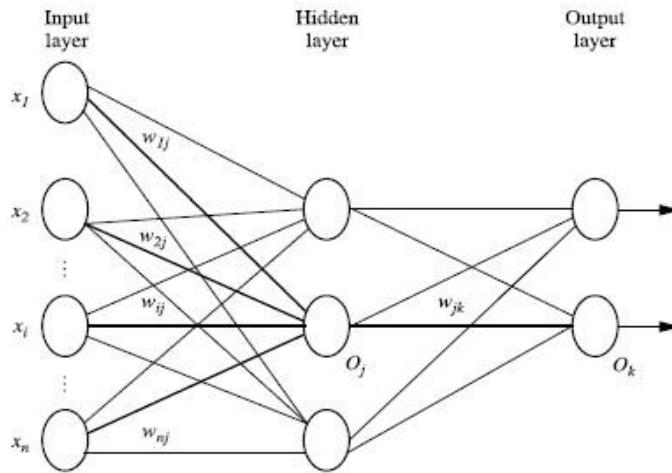
5. In order to predict the class label of X , $P(X|C_i)P(C_i)$ is evaluated for each class C_i . The classifier predicts that the class label of tuple X is the class C_i if and only if

$$P(X|C_i)P(C_i) > P(X|C_j)P(C_j) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

3.4 A Multilayer Feed-Forward Neural Network:

- The backpropagation algorithm performs learning on a multilayer feedforward neural network.
- It iteratively learns a set of weights for prediction of the class label of tuples.
- A multilayer feed-forward neural network consists of an input layer, one or more hidden layers, and an output layer.

Example:



- The inputs to the network correspond to the attributes measured for each training tuple. The inputs are fed simultaneously into the units making up the input layer. These inputs pass through the input layer and are then weighted and fed simultaneously to a second layer known as a hidden layer.
- The outputs of the hidden layer units can be input to another hidden layer, and so on. The number of hidden layers is arbitrary.
- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction for given tuples

3.4.1 Classification by Backpropagation:

- Backpropagation is a neural network learning algorithm.
- A neural network is a set of connected input/output units inwhich each connection has a weight associated with it.
- During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples.
- Neural network learning is also referred to as connectionist learning due to the connections between units.
- Neural networks involve long training times and are therefore more suitable for applicationswhere this is feasible.

- Backpropagation learns by iteratively processing a data set of training tuples, comparing the network's prediction for each tuple with the actual known target value.
- The target value may be the known class label of the training tuple (for classification problems) or a continuous value (for prediction).
- For each training tuple, the weights are modified so as to minimize the mean squared error between the network's prediction and the actual target value. These modifications are made in the —backwards direction, that is, from the output layer, through each hidden layer down to the first hidden layer hence the name is backpropagation.
- Although it is not guaranteed, in general the weights will eventually converge, and the learning process stops.

Advantages:

- It include their high tolerance of noisy data as well as their ability to classify patterns on which they have not been trained.
- They can be used when you may have little knowledge of the relationships between attributesand classes.
- They are well-suited for continuous-valued inputs and outputs, unlike most decision tree algorithms.
- They have been successful on a wide array of real-world data, including handwritten character recognition, pathology and laboratory medicine, and training a computer to pronounce English text.
- Neural network algorithms are inherently parallel; parallelization techniques can be used to speed up the computation process.

Process:

Initialize the weights:

The weights in the network are initialized to small random numbers ranging from -1.0 to 1.0, or -0.5 to 0.5. Each unit has a *bias* associated with it. The biases are similarly initialized to small random numbers.

Each training tuple, X , is processed by the following steps.

Propagate the inputs forward:

First, the training tuple is fed to the input layer of the network. The inputs pass through the input units, unchanged. That is, for an input unit j , its output, O_j , is equal to its input value, I_j . Next, the net input and output of each unit in the hidden and output layers are computed. The net input to a unit in the hidden or output layers is computed as a linear combination of its inputs.

Each such unit has a number of inputs to it that are, in fact, the outputs of the units connected to it in the previous layer. Each connection has a weight. To compute the net input to the unit, each input connected to the unit is multiplied by its corresponding weight, and this is summed.

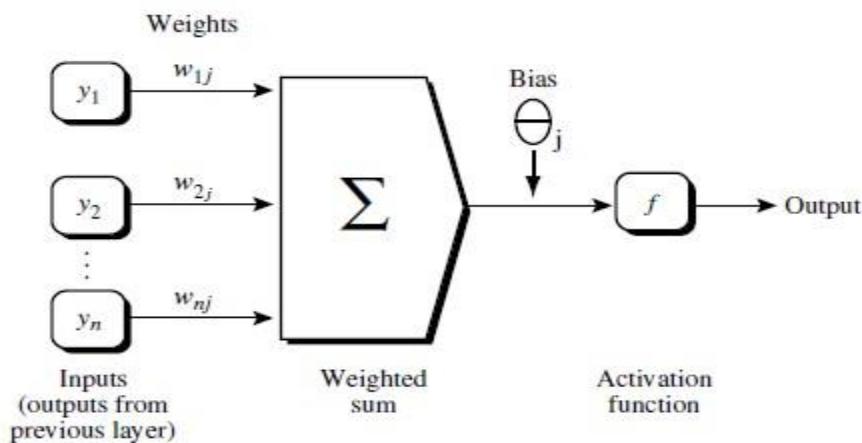
$$I_j = \sum_i w_{ij} O_i + \theta_j,$$

where w_{ij} is the weight of the connection from unit i in the previous layer to unit j ;

O_i is the output of unit i from the previous layer

θ_j is the bias of the unit & it acts as a threshold in that it serves to vary the activity of the unit.

Each unit in the hidden and output layers takes its net input and then applies an activation function to it.



Backpropagate the error:

The error is propagated backward by updating the weights and biases to reflect the error of the network's prediction. For a unit j in the output layer, the error Err_j is computed by

$$Err_j = O_j(1 - O_j)(T_j - O_j)$$

where O_j is the actual output of unit j , and T_j is the known target value of the given training tuple.

The error of a hidden layer unit j is

$$Err_j = O_j(1 - O_j) \sum_k Err_k w_{jk}$$

where w_{jk} is the weight of the connection from unit j to a unit k in the next higher layer, and Err_k is the error of unit k .

Weights are updated by the following equations, where Dw_{ij} is the change in weight w_{ij} :

$$\Delta w_{ij} = (l) Err_j O_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

Biases are updated by the following equations below

$$\Delta \theta_j = (l) Err_j$$

$$\theta_j = \theta_j + \Delta \theta_j$$

Algorithm:

Input:

- D , a data set consisting of the training tuples and their associated target values;
- l , the learning rate;
- $network$, a multilayer feed-forward network.

Output: A trained neural network.

Method:

- (1) Initialize all weights and biases in $network$;
- (2) while terminating condition is not satisfied {
- (3) for each training tuple X in D {
- (4) // Propagate the inputs forward:
- (5) for each input layer unit j {
- (6) $O_j = I_j$; // output of an input unit is its actual input value
- (7) for each hidden or output layer unit j {
- (8) $I_j = \sum_i w_{ij} O_i + \theta_j$; //compute the net input of unit j with respect to the previous layer, i
- (9) $O_j = \frac{1}{1+e^{-I_j}}$; } // compute the output of each unit j
- (10) // Backpropagate the errors:
- (11) for each unit j in the output layer
- (12) $Err_j = O_j(1 - O_j)(T_j - O_j)$; // compute the error
- (13) for each unit j in the hidden layers, from the last to the first hidden layer
- (14) $Err_j = O_j(1 - O_j) \sum_k Err_k w_{jk}$; // compute the error with respect to the next higher layer, k
- (15) for each weight w_{ij} in $network$ {
- (16) $\Delta w_{ij} = (l)Err_j O_i$; // weight increment
- (17) $w_{ij} = w_{ij} + \Delta w_{ij}$; } // weight update
- (18) for each bias θ_j in $network$ {
- (19) $\Delta \theta_j = (l)Err_j$; // bias increment
- (20) $\theta_j = \theta_j + \Delta \theta_j$; } // bias update
- (21) }

3.5 k-Nearest-Neighbor Classifier:

- Nearest-neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it.
- The training tuples are described by n attributes. Each tuple represents a point in an n -dimensional space. In this way, all of the training tuples are stored in an n -dimensional pattern space. When given an unknown tuple, a k -nearest-neighbor classifier searches the pattern space for the k training tuples that are closest to the unknown tuple. These k training tuples are the k nearest neighbors of the unknown tuple.

- Closeness is defined in terms of a distance metric, such as Euclidean distance.
- The Euclidean distance between two points or tuples, say, $X_1 = (x_{11}, x_{12}, \dots, x_{1n})$ and $X_2 = (x_{21}, x_{22}, \dots, x_{2n})$, is

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^n (x_{1i} - x_{2i})^2}.$$

In other words, for each numeric attribute, we take the difference between the corresponding values of that attribute in tuple X_1 and in tuple X_2 , square this difference, and accumulate it.

The square root is taken of the total accumulated distance count.

Min-Max normalization can be used to transform a value v of a numeric attribute A to v_0 in the range $[0, 1]$ by computing

$$v' = \frac{v - min_A}{max_A - min_A},$$

where min_A and max_A are the minimum and maximum values of attribute A

- For k -nearest-neighbor classification, the unknown tuple is assigned the most common class among its k nearest neighbors.
- When $k = 1$, the unknown tuple is assigned the class of the training tuple that is closest to it in pattern space.
- Nearest neighbor classifiers can also be used for prediction, that is, to return a real-valued prediction for a given unknown tuple.
- In this case, the classifier returns the average value of the real-valued labels associated with the k nearest neighbors of the unknown tuple.

3.6 Other Classification Methods:

3.6.1 Genetic Algorithms:

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, genetic learning starts as follows.

- An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits. As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A₁ and A₂, and that there are two classes, C₁ and C₂.
- The rule —IF A₁ ANDNOT A₂ THEN C₂— can be encoded as the bit string —100, where the two leftmost bits represent attributes A₁ and A₂, respectively, and the rightmost bit represents the class.
- Similarly, the rule —IF NOT A₁ AND NOT A₂ THEN C₁— can be encoded as —001.
- If an attribute has k values, where k > 2, then k bits may be used to encode the attribute's values.

Classes can be encoded in a similar fashion.

- Based on the notion of survival of the fittest, a new population is formed to consist of the fittest rules in the current population, as well as offspring of these rules.
- Typically, the fitness of a rule is assessed by its classification accuracy on a set of training samples.
- Offspring are created by applying genetic operators such as crossover and mutation.
- In crossover, substrings from pairs of rules are swapped to form new pairs of rules.
- In mutation, randomly selected bits in a rule's string are inverted.
- The process of generating new populations based on prior populations of rules continues until a population, P, evolves where each rule in P satisfies a pre specified fitness threshold.
- Genetic algorithms are easily parallelizable and have been used for classification as well as other optimization problems. In data mining, they may be used to evaluate the fitness of other algorithms.

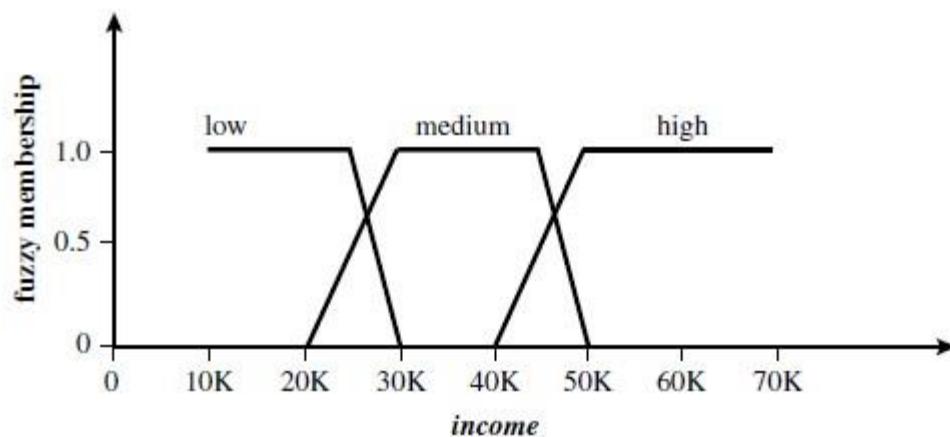
3.6.2 Fuzzy Set Approaches:

- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree of membership that a certain value has in a given category. Each category then represents a fuzzy set.

Fuzzy logic systems typically provide graphical tools to assist users in converting attribute values to fuzzy truthvalues.

- Fuzzy set theory is also known as possibility theory.
It was proposed by Lotfi Zadeh in 1965 as an alternative to traditional two-value logic and probability theory.
- It lets us work at a high level of abstraction and offers a means for dealing with imprecise measurements of data.
- Most important, fuzzy set theory allows us to deal with vague or inexact facts.
- Unlike the notion of traditional —crisp— sets where an element either belongs to a set S or its complement, in fuzzy set theory, elements can belong to more than one fuzzy set.
- Fuzzy set theory is useful for data mining systems performing rule-based classification.
- It provides operations for combining fuzzy measurements.
- Several procedures exist for translating the resulting fuzzy output into a *defuzzified* or crisp value that is returned by the system.
- Fuzzy logic systems have been used in numerous areas for classification, including market research, finance, health care, and environmental engineering.

Example:



3.7 Regression Analysis:

- Regression analysis can be used to model the relationship between one or more independent or predictor variables and a dependent or response variable which is continuous-valued.
- In the context of data mining, the predictor variables are the attributes of interest describing the tuple (i.e., making up the attribute vector).
- In general, the values of the predictor variables are known.

-

The response variable is what we want to predict.

3.7.1 Linear Regression:

- Straight-line regression analysis involves a response variable, y , and a single predictor variable x .
- It is the simplest form of regression, and models y as a linear function of x . That is, $y = b + wx$ where the variance of y is assumed to be constant b and w are regression coefficients specifying the Y-intercept and slope of the line.
- The regression coefficients, w and b , can also be thought of as weights, so that we can equivalently write, $y = w_0 + w_1x$
- These coefficients can be solved for by the method of least squares, which estimates the best-fitting straight line as the one that minimizes the error between the actual data and the estimate of the line.
- Let D be a training set consisting of values of predictor variable, x , for some population and their associated values for response variable, y . The training set contains $|D|$ data points of the form $(x_1, y_1), (x_2, y_2), \dots, (x_{|D|}, y_{|D|})$.

The regression coefficients can be estimated using this method with the following equations:

$$w_1 = \frac{\sum_{i=1}^{|D|} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|} (x_i - \bar{x})^2}$$

$$w_0 = \bar{y} - w_1 \bar{x}$$

where \bar{x} is the mean value of $x_1, x_2, \dots, x_{|D|}$, and \bar{y} is the mean value of $y_1, y_2, \dots, y_{|D|}$. The coefficients w_0 and w_1 often provide good approximations to otherwise complicated regression equations.

3.7.2 Multiple Linear Regression:

- It is an extension of straight-line regression so as to involve more than one predictor variable.

-

It allows response variable y to be modeled as a linear function of, say, n predictor variables or attributes, A_1, A_2, \dots, A_n , describing a tuple, X .

- An example of a multiple linear regression model based on two predictor attributes or variables, A_1 and A_2 , is $y = w_0 + w_1x_1 + w_2x_2$

where x_1 and x_2 are the values of attributes A_1 and A_2 , respectively, in X .

- Multiple regression problems are instead commonly solved with the use of statistical software packages, such as SAS, SPSS, and S-Plus.

3.7.3 Nonlinear Regression:

- It can be modeled by adding polynomial terms to the basic linear model.
- By applying transformations to the variables, we can convert the nonlinear model into a linear one that can then be solved by the method of least squares.
- Polynomial Regression is a special case of multiple regression. That is, the addition of high-order terms like x^2, x^3 , and so on, which are simple functions of the single variable, x , can be considered equivalent to adding new independent variables.

Transformation of a polynomial regression model to a linear regression model:

Consider a cubic polynomial relationship given by

$$y =$$

$$w_0 + w_1x + w_2x^2 + w_3x^3$$

To convert this equation to linear form, we define new variables: x_1

$$= x, x_2 = x^2, x_3 = x^3$$

It can then be converted to linear form by applying the above assignments, resulting in the equation $y = w_0 + w_1x + w_2x^2 + w_3x^3$ which is easily solved by the method of least squares using software for regression analysis.

3.8 Classifier Accuracy:

- The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.

- In the pattern recognition literature, this is also referred to as the overall recognition rate of the classifier, that is, it reflects how well the classifier recognizes tuples of the various classes. The error rate or misclassification rate of a classifier, M, which is simply $1 - \text{Acc}(M)$, where $\text{Acc}(M)$ is the accuracy of M.
- The confusion matrix is a useful tool for analyzing how well your classifier can recognize tuples of different classes.
- True positives refer to the positive tuples that were correctly labeled by the classifier.
- True negatives are the negative tuples that were correctly labeled by the classifier.
- False positives are the negative tuples that were incorrectly labeled.
- How well the classifier can recognize, for this sensitivity and specificity measures can be used.

Accuracy is a function of sensitivity and specificity.

$$\text{accuracy} = \text{sensitivity} \frac{\text{pos}}{(\text{pos} + \text{neg})} + \text{specificity} \frac{\text{neg}}{(\text{pos} + \text{neg})}.$$

$$\text{sensitivity} = \frac{t_pos}{pos}$$

$$\text{specificity} = \frac{t_neg}{neg}$$

$$\text{precision} = \frac{t_pos}{(t_pos + f_pos)}$$

where
 t_pos is the number of true
 positives
 pos is the number of positive
 tuples
 t_neg is the number of true
 negatives
 neg is the number of negative
 tuples,
 f_pos is the number of false
 positives

Chapter-4

4.1 Cluster Analysis:

- The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering.
- A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters.
- A cluster of data objects can be treated collectively as one group and so may be considered as a form of data compression.
- Cluster analysis tools based on k-means, k-medoids, and several methods have also been built into many statisticalanalysis software packages or systems, such as S-Plus, SPSS, and SAS.

4.1.1 Applications:

- Cluster analysis has been widely used in numerous applications, including market research, pattern recognition, data analysis, and image processing.
- In business, clustering can help marketers discover distinct groups in their customer bases and characterize customer groups based on purchasing patterns.
- In biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionality, and gain insight into structures inherent in populations.
- Clustering may also help in the identification of areas of similar land use in an earth observation database and in the identification of groups of houses in a city according to house type, value, and geographic location, as well as the identification of groups of automobile insurance policy holders with a high average claim cost.
- Clustering is also called data segmentation in some applications because clustering partitions large data sets into groups according to their *similarity*.

- Clustering can also be used for outlier detection, Applications of outlier detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce.

4.1.2 Typical Requirements Of Clustering InData Mining:

➤ **Scalability:**

Many clustering algorithms work well on small data sets containing fewer than several hundred data objects; however, a large database may contain millions of objects. Clustering on a sample of a given large data set may lead to biased results.

Highly scalable clustering algorithms are needed.

➤ **Ability to deal with different types of attributes:**

Many algorithms are designed to cluster interval-based (numerical) data. However, applications may require clustering other types of data, such as binary, categorical (nominal), and ordinal data, or mixtures of these data types.

➤ **Discovery of clusters with arbitrary shape:**

Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures. Algorithms based on such distance measures tend to find spherical clusters with similar size and density.

However, a cluster could be of any shape. It is important to develop algorithms that can detect clusters of arbitrary shape.

➤ **Minimal requirements for domain knowledge to determine input parameters:**

Many clustering algorithms require users to input certain parameters in cluster analysis (such as the number of desired clusters). The clustering results can be quite sensitive to input parameters. Parameters are often difficult to determine, especially for data sets containing high-dimensional objects. This not only burdens users, but it also makes the quality of clustering difficult to control.

➤ **Ability to deal with noisy data:**

Most real-world databases contain outliers or missing, unknown, or erroneous data.

Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality.

➤ **Incremental clustering and insensitivity to the order of input records:**

Some clustering algorithms cannot incorporate newly inserted data (i.e., database updates) into existing clustering structures and, instead, must determine a new clustering from scratch. Some clustering algorithms are sensitive to the order of input data.

That is, given a set of data objects, such an algorithm may return dramatically different clusterings depending on the order of presentation of the input objects.

It is important to develop incremental clustering algorithms and algorithms that are insensitive to the order of input.

➤ **High dimensionality:**

A database or a data warehouse can contain several dimensions or attributes. Many clustering algorithms are good at handling low-dimensional data, involving only two to three dimensions. Human eyes are good at judging the quality of clustering for up to three dimensions. Finding clusters of data objects in high-dimensional space is challenging, especially considering that such data can be sparse and highly skewed.

➤ **Constraint-based clustering:**

Real-world applications may need to perform clustering under various kinds of constraints. Suppose that your job is to choose the locations for a given number of new automatic banking machines (ATMs) in a city. To decide upon this, you may cluster households while considering constraints such as the city's rivers and highway networks, and the type and number of customers per cluster. A challenging task is to find groups of data with good clustering behavior that satisfy specified constraints.

➤ **Interpretability and usability:**

Users expect clustering results to be interpretable, comprehensible, and usable. That is, clustering may need to be tied to specific semantic interpretations and applications. It is important to study how an application goal may influence the selection of clustering features and methods.

4.2 Major Clustering Methods:

- Partitioning Methods
- Hierarchical Methods

- Density-Based Methods
- Grid-Based Methods
- Model-Based Methods

4.2.1 Partitioning Methods:

A partitioning method constructs k partitions of the data, where each partition represents a cluster and $k \leq n$. That is, it classifies the data into k groups, which together satisfy the following requirements:

- Each group must contain at least one object, and Each
- object must belong to exactly one group.

A partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another.

The general criterion of a good partitioning is that objects in the same cluster are close or related to each other, whereas objects of different clusters are far apart or very different.

4.2.2 Hierarchical Methods:

A hierarchical method creates a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed.

- ❖ The agglomerative approach, also called the bottom-up approach, starts with each object forming a separate group. It successively merges the objects or groups that are closest to one another, until all of the groups are merged into one or until a termination condition holds.
- ❖ The divisive approach, also called the top-down approach, starts with all of the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters, until eventually each object is in one cluster, or until a termination condition holds.

Hierarchical methods suffer from the fact that once a step (merge or split) is done, it can never be undone. This rigidity is useful in that it leads to smaller computation costs by not having to worry about a combinatorial number of different choices.

There are two approaches to improving the quality of hierarchical clustering:

- ❖ Perform careful analysis of object —linkages at each hierarchical partitioning, such as in Chameleon, or
- ❖ Integrate hierarchical agglomeration and other approaches by first using a hierarchical agglomerative algorithm to group objects into microclusters, and then performing macroclustering on the microclusters using another clustering method such as iterative relocation.

4.2.3 Density-based methods:

- ❖ Most partitioning methods cluster objects based on the distance between objects. Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes.
- ❖ Other clustering methods have been developed based on the notion of density. Their general idea is to continue growing the given cluster as long as the density in the neighborhood exceeds some threshold; that is, for each data point within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of points. Such a method can be used to filter out noise (outliers) and discover clusters of arbitrary shape.
- ❖ DBSCAN and its extension, OPTICS, are typical density-based methods that grow clusters according to a density-based connectivity analysis. DENCLUE is a method that clusters objects based on the analysis of the value distributions of density functions.

4.2.4 Grid-Based Methods:

- ❖ Grid-based methods quantize the object space into a finite number of cells that form a grid structure.
- ❖ All of the clustering operations are performed on the grid structure i.e., on the quantized space. The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantized space.
- ❖ STING is a typical example of a grid-based method. Wave Cluster applies wavelet transformation for clustering analysis and is both grid-based and density-based.

4.2.5 Model-Based Methods:

- ❖ Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model.
- ❖ A model-based algorithm may locate clusters by constructing a density function that reflects the spatial distribution of the data points.
- ❖ It also leads to a way of automatically determining the number of clusters based on standard statistics, taking —noise|| or outliers into account and thus yielding robust clustering methods.

4.3 Tasks in Data Mining:

- Clustering High-Dimensional Data
- Constraint-Based Clustering

4.3.1 Clustering High-Dimensional Data:

- It is a particularly important task in cluster analysis because many applications require the analysis of objects containing a large number of features or dimensions. • For example, text documents may contain thousands of terms or keywords as features, and DNA micro array data may provide information on the expression levels of thousands of genes under hundreds of conditions.
- Clustering high-dimensional data is challenging due to the curse of dimensionality.
- Many dimensions may not be relevant. As the number of dimensions increases, the data

become increasingly sparse so that the distance measurement between pairs of points become meaningless and the average density of points anywhere in the data is likely to be low. Therefore, a different clustering methodology needs to be developed for high-dimensional data.

- CLIQUE and PROCLUS are two influential subspace clustering methods, which search for clusters in subspaces of the data, rather than over the entire data space.
- Frequent pattern-based clustering, another clustering methodology, extracts distinct frequent patterns among subsets of dimensions that occur frequently. It uses such patterns to group objects and generate meaningful clusters.

4.3.2 Constraint-Based Clustering:

- It is a clustering approach that performs clustering by incorporation of user-specified or application-oriented constraints.
- A constraint expresses a user's expectation or describes properties of the desired clustering results, and provides an effective means for communicating with the clustering process.
- Various kinds of constraints can be specified, either by a user or as per application requirements.
- Spatial clustering employs with the existence of obstacles and clustering under user-specified constraints. In addition, semi-supervised clustering employs for pairwise constraints in order to improve the quality of the resulting clustering.

4.4 Classical Partitioning Methods:

The most well-known and commonly used partitioning methods are

- ❖ The k -Means Method
- ❖ k-Medoids Method

4.4.1 Centroid-Based Technique: The K-Means Method:

The k-means algorithm takes the input parameter, k , and partitions a set of n objects into k clusters so that the resulting intracluster similarity is high but the intercluster similarity is low. Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity.

The k -means algorithm proceeds as follows.

- First, it randomly selects k of the objects, each of which initially represents a cluster mean or center.
- For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean.
- It then computes the new mean for each cluster.
- This process iterates until the criterion function converges.

Typically, the square-error criterion is used, defined as

$$E = \sum_{i=1}^k \sum_{p \in C_i} |p - m_i|^2,$$

where E is the sum of the square error for all objects in the data set
 p is the point in space representing a given object
 m_i is the mean of cluster C_i

4.4.1 The k-means partitioning algorithm:

The k -means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

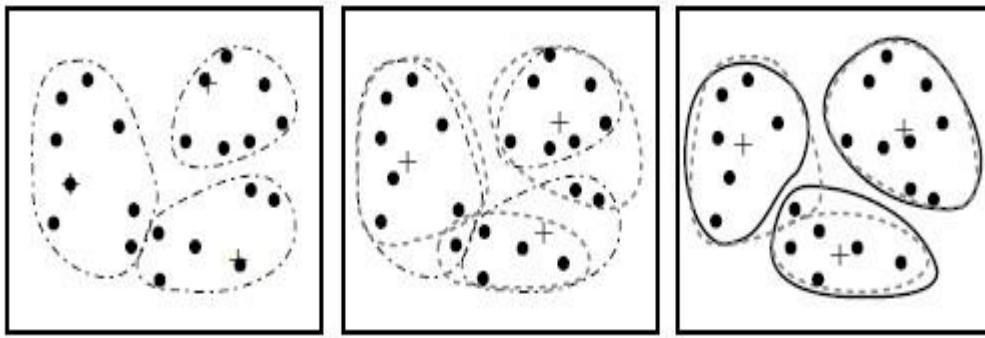
Input:

- k : the number of clusters,
- D : a data set containing n objects.

Output: A set of k clusters.

Method:

- (1) arbitrarily choose k objects from D as the initial cluster centers;
- (2) **repeat**
- (3) (re)assign each object to the cluster to which the object is the most similar,
 based on the mean value of the objects in the cluster;
- (4) update the cluster means, i.e., calculate the mean value of the objects for
 each cluster;
- (5) **until** no change;



Clustering of a set of objects based on the k -means method

4.4.2 The k -Medoids Method:

- The k -means algorithm is sensitive to outliers because an object with an extremely large value may substantially distort the distribution of data. This effect is particularly exacerbated due to the use of the square-error function.
- Instead of taking the mean value of the objects in a cluster as a reference point, we can pick actual objects to represent the clusters, using one representative object per cluster. Each remaining object is clustered with the representative object to which it is the most similar.
 - The partitioning method is then performed based on the principle of minimizing the sum

of the dissimilarities between each object and its corresponding reference point. That is, an absolute-error criterion is used, defined as

$$E = \sum_{j=1}^k \sum_{p \in C_j} |p - o_j|,$$

where E is the sum of the absolute error for all objects in the data set

p is the point in space representing a given object in cluster C_j o_j is the representative object of C_j

- The initial representative objects are chosen arbitrarily. The iterative process of replacing representative objects by non representative objects continues as long as the quality of the resulting clustering is improved.
- This quality is estimated using a cost function that measures the average dissimilarity between an object and the representative object of its cluster.
- To determine whether a non representative object, o_j random, is a good replacement for a current representative object, o_j , the following four cases are examined for each of the nonrepresentative objects.

Case 1:

p currently belongs to representative object, o_j . If o_j is replaced by o_{random} as a representative object and p is closest to one of the other representative objects, $o_i, i \neq j$, then p is reassigned to o_i .

Case 2:

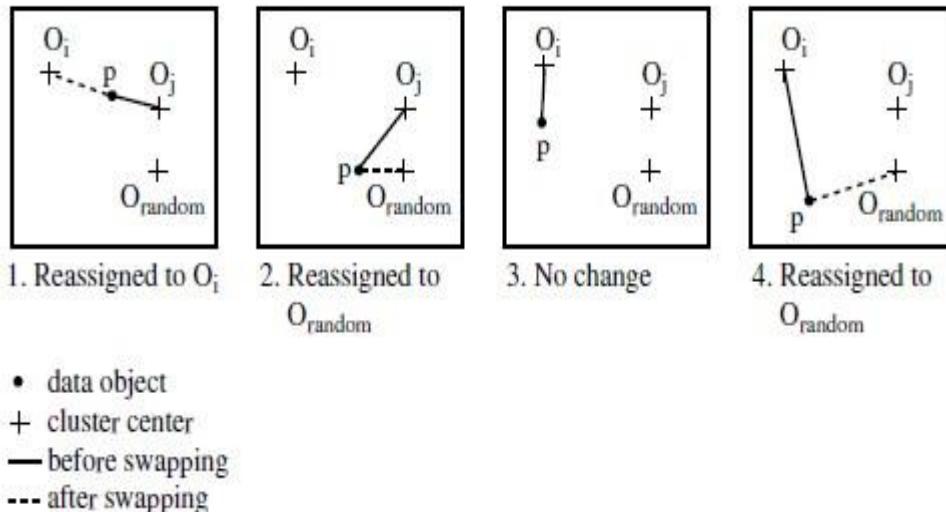
p currently belongs to representative object, o_j . If o_j is replaced by o_{random} as a representative object and p is closest to o_{random} , then p is reassigned to o_{random} .

Case 3:

p currently belongs to representative object, o_i , $i \neq j$. If o_j is replaced by o_{random} as a representative object and p is still closest to o_i , then the assignment does not change.

Case 4:

p currently belongs to representative object, o_i , $i \neq j$. If o_j is replaced by o_{random} as a representative object and p is closest to o_{random} , then p is reassigned to o_{random} .



Four cases of the cost function for k -medoids clustering

4.4.2 The k-Medoids Algorithm:

The k-medoids algorithm for partitioning based on medoid or central objects.

Input:

- k : the number of clusters,
- D : a data set containing n objects.

Output: A set of k clusters.

Method:

- (1) arbitrarily choose k objects in D as the initial representative objects or seeds;
- (2) **repeat**
- (3) assign each remaining object to the cluster with the nearest representative object;
- (4) randomly select a nonrepresentative object, o_{random} ;
- (5) compute the total cost, S , of swapping representative object, o_j , with o_{random} ;
- (6) if $S < 0$ then swap o_j with o_{random} to form the new set of k representative objects;
- (7) **until** no change;

The k -medoids method is more robust than k -means in the presence of noise and outliers, because a medoid is less influenced by outliers or other extreme values than a mean. However, its processing is more costly than the k -means method.

4.5 Hierarchical Clustering Methods:

- A hierarchical clustering method works by grouping data objects into a tree of clusters.
- The quality of a pure hierarchical clustering method suffers from its inability to perform adjustment once a merge or split decision has been executed. That is, if a particular merge or split decision later turns out to have been a poor choice, the method cannot backtrack and correct it.

Hierarchical clustering methods can be further classified as either agglomerative or divisive, depending on whether the hierarchical decomposition is formed in a bottom-up or top-down fashion.

4.5.1 Agglomerative hierarchical clustering:

- This bottom-up strategy starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster or until certain termination conditions are satisfied.

- Most hierarchical clustering methods belong to this category. They differ only in their definition of intercluster similarity.

4.5.2 Divisive hierarchical clustering:

- This top-down strategy does the reverse of agglomerative hierarchical clustering by starting with all objects in one cluster.
- It subdivides the cluster into smaller and smaller pieces, until each object forms a cluster on its own or until it satisfies certain termination conditions, such as a desired number of clusters is obtained or the diameter of each cluster is within a certain threshold.

4.6 Constraint-Based Cluster Analysis:

Constraint-based clustering finds clusters that satisfy user-specified preferences or constraints. Depending on the nature of the constraints, constraint-based clustering may adopt rather different approaches.

There are a few categories of constraints.

➤ **Constraints on individual objects:**

We can specify constraints on the objects to be clustered. In a real estate application, for example, one may like to spatially cluster only those luxury mansions worth over a million dollars. This constraint confines the set of objects to be clustered. It can easily be handled by preprocessing after which the problem reduces to an instance of unconstrained clustering.

➤ **Constraints on the selection of clustering parameters:**

A user may like to set a desired range for each clustering parameter. Clustering parameters are usually quite specific to the given clustering algorithm. Examples of parameters include k , the desired number of clusters in a k-means algorithm; or ϵ the radius and the minimum number of points in the DBSCAN algorithm. Although such user-specified parameters may

strongly influence the clustering results, they are usually confined to the algorithm itself. Thus, their fine tuning and processing are usually not considered a form of constraint-based clustering.

➤ **Constraints on distance or similarity functions:**

We can specify different distance or similarity functions for specific attributes of the objects to be clustered, or different distance measures for specific pairs of objects. When clustering sportsmen, for example, we may use different weighting schemes for height, body weight, age, and skill level. Although this will likely change the mining results, it may not alter the clustering process per se. However, in some cases, such changes may make the evaluation of the distance function nontrivial, especially when it is tightly intertwined with the clustering process.

➤ **User-specified constraints on the properties of individual clusters:**

A user may like to specify desired characteristics of the resulting clusters, which may strongly influence the clustering process.

➤ **Semi-supervised clustering based on partial supervision:**

The quality of unsupervised clustering can be significantly improved using some weak form of supervision. This may be in the form of pairwise constraints (i.e., pairs of objects labeled as belonging to the same or different cluster). Such a constrained clustering process is called semi-supervised clustering.

4.7 Outlier Analysis:

- There exist data objects that do not comply with the general behavior or model of the data. Such data objects, which are grossly different from or inconsistent with the remaining set of data, are called outliers.
- Many data mining algorithms try to minimize the influence of outliers or eliminate them all together. This, however, could result in the loss of important hidden information because one person's noise could be another person's signal. In other words, the outliers may be of particular interest, such as in the case of fraud

detection, where outliers may indicate fraudulent activity. Thus, outlier detection and analysis is an interesting data mining task, referred to as outlier mining.

- It can be used in fraud detection, for example, by detecting unusual usage of credit cards or telecommunication services. In addition, it is useful in customized marketing for identifying the spending behavior of customers with extremely low or extremely high incomes, or in medical analysis for finding unusual responses to various medical treatments.

Outlier mining can be described as follows: Given a set of n data points or objects and k , the expected number of outliers, find the top k objects that are considerably dissimilar, exceptional, or inconsistent with respect to the remaining data. The outlier mining problem can be viewed as two subproblems:

- Define what data can be considered as inconsistent in a given data set, and
- Find an efficient method to mine the outliers so defined.

Types of outlier detection:

- Statistical Distribution-Based Outlier Detection
- Distance-Based Outlier Detection
- Density-Based Local Outlier Detection
- Deviation-Based Outlier Detection

4.7.1 Statistical Distribution-Based Outlier Detection:

The statistical distribution-based approach to outlier detection assumes a distribution or probability model for the given data set (e.g., a normal or Poisson distribution) and then identifies outliers with respect to the model using a discordancy test. Application of the test requires knowledge of the data set parameters. Knowledge of distribution parameters such as the mean and variance and the expected number of outliers.

A statistical discordancy test examines two hypotheses:

- A working hypothesis
- An alternative hypothesis

A working hypothesis, H, is a statement that the entire data set of n objects comes from an initial distribution model, F, that is,

$$H : o_i \in F, \text{ where } i = 1, 2, \dots, n.$$

The hypothesis is retained if there is no statistically significant evidence supporting its rejection. A discordancy test verifies whether an object, o_i , is significantly large (or small) in relation to the distribution F. Different test statistics have been proposed for use as a discordancy test, depending on the available knowledge of the data. Assuming that some statistic, T, has been chosen for discordancy testing, and the value of the statistic for object o_i is v_i , then the distribution of T is constructed. Significance probability, $SP(v_i) = \text{Prob}(T > v_i)$, is evaluated. If $SP(v_i)$ is sufficiently small, then o_i is discordant and the working hypothesis is rejected.

An alternative hypothesis, H, which states that o_i comes from another distribution model, G, is adopted. The result is very much dependent on which model F is chosen because o_i may be an outlier under one model and a perfectly valid value under another. The alternative distribution is very important in determining the power of the test, that is, the probability that the working hypothesis is rejected when o_i is really an outlier.

There are different kinds of alternative distributions.

- **Inherent alternative distribution:**

In this case, the working hypothesis that all of the objects come from distribution F is rejected in favor of the alternative hypothesis that all of the objects arise from another distribution, G:

$$H : o_i \notin G, \text{ where } i = 1, 2, \dots, n$$

F and G may be different distributions or differ only in parameters of the same distribution.

There are constraints on the form of the G distribution in that it must have potential to produce outliers. For example, it may have a different mean or dispersion, or a longer tail.

- **Mixture alternative distribution:**

The mixture alternative states that discordant values are not outliers in the F population, but contaminants from some other population, G. In this case, the alternative hypothesis is

$$\overline{H} : o_i \in (1 - \lambda)F + \lambda G, \quad \text{where } i = 1, 2, \dots, n.$$

- **Slippage alternative distribution:**

This alternative states that all of the objects (apart from some prescribed small number) arise independently from the initial model, F, with its given parameters, whereas the remaining objects are independent observations from a modified version of F in which the parameters have been shifted.

There are two basic types of procedures for detecting outliers:

Block procedures:

In this case, either all of the suspect objects are treated as outliers or all of them are accepted as consistent.

Consecutive procedures:

An example of such a procedure is the *insideout* procedure. Its main idea is that the object that is least likely to be an outlier is tested first. If it is found to be an outlier, then all of the more extreme values are also considered outliers; otherwise, the next most extreme object is tested, and so on. This procedure tends to be more effective than block procedures.

4.7.2 Distance-Based Outlier Detection:

The notion of distance-based outliers was introduced to counter the main limitations imposed by statistical methods. An object, o, in a data set, D, is a distance-based (DB) outlier with parameters pct and dmin, that is, a DB(pct;dmin)-outlier, if at least a fraction, pct, of the objects in D lie at a distance greater than dmin from o. In other words, rather than relying on statistical tests, we can think of distance-based outliers as those objects that do not have enough neighbors, where neighbors are defined based on distance from the given object. In comparison with statistical-based methods, distance-based outlier detection generalizes the ideas behind discordancy testing for various standard distributions. Distance-based outlier detection avoids the excessive computation that can be associated with fitting the observed distribution into some standard distribution and in selecting discordancy tests.

For many discordancy tests, it can be shown that if an object, o , is an outlier according to the given test, then o is also a DB(pct , d_{min})-outlier for some suitably defined pct and d_{min} . For example, if objects that lie three or more standard deviations from the mean are considered to be outliers, assuming a normal distribution, then this definition can be generalized by a DB(0.9988, 0.13s) outlier.

Several efficient algorithms for mining distance-based outliers have been developed.

Index-based algorithm:

Given a data set, the index-based algorithm uses multidimensional indexing structures, such as R-trees or k-d trees, to search for neighbors of each object o within radius d_{min} around that object. Let M be the maximum number of objects within the d_{min} -neighborhood of an outlier. Therefore, once $M+1$ neighbors of object o are found, it is clear that o is not an outlier. This algorithm has a worst-case complexity of $O(n2^k)$, where n is the number of objects in the data set and k is the dimensionality. The index-based algorithm scales well as k increases. However, this complexity evaluation takes only the search time into account, even though the task of building an index in itself can be computationally intensive.

Nested-loop algorithm:

The nested-loop algorithm has the same computational complexity as the index-based algorithm but avoids index structure construction and tries to minimize the number of I/Os. It divides the memory buffer space into two halves and the data set into several logical blocks. By carefully choosing the order in which blocks are loaded into each half, I/O efficiency can be achieved. **Cell-based algorithm:**

To avoid $O(n^2)$ computational complexity, a cell-based algorithm was developed for memoryresident data sets. Its complexity is $O(c^k + n)$, where c is a constant depending on the number of cells and k is the dimensionality.

In this method, the data space is partitioned into cells with a side length equal to $\frac{d_{min}}{2\sqrt{k}}$.

Each cell has two layers surrounding it. The first layer is one cell thick, while the second is

$\lceil 2\sqrt{k} - 1 \rceil$ cells thick, rounded up to the closest integer. The algorithm counts outliers on a cell-by-cell rather than an object-by-object basis. For a given cell, it accumulates three counts—the number of objects in the cell, in the cell and the first layer together, and in the cell and both

layers together. Let's refer to these counts as cell count, cell + 1 layer count, and cell + 2 layers count, respectively.

Let M be the maximum number of outliers that can exist in the d_{min} -neighborhood of an outlier.

- An object, \mathbf{o} , in the current cell is considered an outlier only if cell + 1 layer count is less than or equal to M. If this condition does not hold, then all of the objects in the cell can be removed from further investigation as they cannot be outliers.
- If cell + 2_layers_count is less than or equal to M, then all of the objects in the cell are considered outliers. Otherwise, if this number is more than M, then it is possible that some of the objects in the cell may be outliers. To detect these outliers, object-by-object processing is used where, for each object, \mathbf{o} , in the cell, objects in the second layer of \mathbf{o} are examined. For objects in the cell, only those objects having no more than M points in their d_{min} -neighborhoods are outliers. The d_{min} -neighborhood of an object consists of the object's cell, all of its first layer, and some of its second layer.

A variation to the algorithm is linear with respect to n and guarantees that no more than three passes over the data set are required. It can be used for large disk-resident data sets, yet does not scale well for high dimensions.

4.7.3 Density-Based Local Outlier Detection:

Statistical and distance-based outlier detection both depend on the overall or global distribution of the given set of data points, D. However, data are usually not uniformly distributed. These methods encounter difficulties when analyzing data with rather different density distributions.

To define the local outlier factor of an object, we need to introduce the concepts of k-distance, k-distance neighborhood, reachability distance,¹³ and local reachability density.

These are defined as follows:

The k-distance of an object p is the maximal distance that p gets from its k-nearest neighbors. This distance is denoted as k-distance(p). It is defined as the distance, $d(p, o)$, between p and an object $\mathbf{o} \in D$, such that for at least k objects, $\mathbf{o}_0 \in D$, it holds that $d(p, \mathbf{o}_0) \leq d(p, \mathbf{o})$. That is,

there are at least k objects in D that are as close as or closer to p than o , and for at most $k-1$ objects, $o \in D$, it holds that $d(p; o) < d(p, o)$.

That is, there are at most $k-1$ objects that are closer to p than o . You may be wondering at this point how k is determined. The LOF method links to density-based clustering in that it sets k to the parameter $rMinPts$, which specifies the minimum number of points for use in identifying clusters based on density.

Here, $MinPts$ (as k) is used to define the local neighborhood of an object, p .

The k -distance neighborhood of an object p is denoted $N_{k\text{distance}(p)}(p)$, or $N_k(p)$ for short. By setting k to $MinPts$, we get $N_{MinPts}(p)$. It contains the $MinPts$ -nearest neighbors of p . That is, it contains every object whose distance is not greater than the $MinPts$ -distance of p . The reachability distance of an object p with respect to object o (where o is within the $MinPts$ -nearest neighbors of p), is defined as $\text{reach_dist}_{MinPts}(p, o) = \max\{\text{MinPts_distance}(o), d(p, o)\}$.

Intuitively, if an object p is far away, then the reachability distance between the two is simply their actual distance. However, if they are sufficiently close (i.e., where p is within the $MinPts$ -distance neighborhood of o), then the actual distance is replaced by the $MinPts$ distance of o . This helps to significantly reduce the statistical fluctuations of $d(p, o)$ for all of the p close to o .

The higher the value of $MinPts$ is, the more similar is the reachability distance for objects within the same neighborhood.

Intuitively, the local reachability density of p is the inverse of the average reachability density based on the $MinPts$ -nearest neighbors of p . It is defined as

$$\text{lrd}_{MinPts}(p) = \frac{|N_{MinPts}(p)|}{\sum_{o \in N_{MinPts}(p)} \text{reach_dist}_{MinPts}(p, o)}.$$

The local outlier factor (LOF) of p captures the degree to which we call p an outlier. It is defined as

$$\text{LOF}_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{\text{lrd}_{MinPts}(o)}{\text{lrd}_{MinPts}(p)}}{|N_{MinPts}(p)|}.$$

It is the average of the ratio of the local reachability density of p and those of p 's

$MinPts$ -nearest neighbors. It is easy to see that the lower p 's local reachability density is, and the higher the local reachability density of p 's $MinPts$ -nearest neighbors are, the higher $LOF(p)$ is.

4.7.4 Deviation-Based Outlier Detection:

Deviation-based outlier detection does not use statistical tests or distance-based measures to identify exceptional objects. Instead, it identifies outliers by examining the main characteristics of objects in a group. Objects that —deviate from this description are considered outliers. Hence, in this approach the term deviations is typically used to refer to outliers. In this section, we study two techniques for deviation-based outlier detection. The first sequentially compares objects in a set, while the second employs an OLAP data cube approach.

Sequential Exception Technique:

The sequential exception technique simulates the way in which humans can distinguish unusual objects from among a series of supposedly like objects. It uses implicit redundancy of the data. Given a data set, D , of n objects, it builds a sequence of subsets, $\{D_1, D_2, \dots, D_m\}$, of these objects with $2 \leq m \leq n$ such that

$$D_{j-1} \subset D_j, \quad \text{where } D_j \subseteq D.$$

Dissimilarities are assessed between subsets in the sequence. The technique introduces the following key terms.

Exception set:

This is the set of deviations or outliers. It is defined as the smallest subset of objects whose removal results in the greatest reduction of dissimilarity in the residual set.

Dissimilarity function:

This function does not require a metric distance between the objects. It is any function that, if given a set of objects, returns a low value if the objects are similar to one another. The greater the dissimilarity among the objects, the higher the value returned by the function. The dissimilarity of a subset is incrementally computed based on the subset prior to it in the sequence. Given a subset of n numbers, $\{x_1, \dots, x_n\}$, a possible dissimilarity function is the variance of the numbers in the set, that is,

$$\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2,$$

where x is the mean of the n numbers in the set. For character strings, the dissimilarity function may be in the form of a pattern string (e.g., containing wildcard characters) that is used to cover all of the patterns seen so far. The dissimilarity increases when the pattern covering all of the strings in D_{j-1} does not cover any string in D_j that is not in D_{j-1} .

Cardinality function:

This is typically the count of the number of objects in a given set.

Smoothing factor:

This function is computed for each subset in the sequence. It assesses how much the dissimilarity can be reduced by removing the subset from the original set of objects.

Distributed Systems – Lecture Notes



GraphicEra

Deemed to be University

Accredited by NAAC with Grade A

NBA Accredited Programs in ECE, CSE & ME
Approved by AICTE, Ministry of HRD, Govt. of India

DEHRADUN, UTTARAKHAND, INDIA

Contents

1 Introduction to Distributed Systems	6
2 Introduction to Computer Networking	11
Introduction	11
OSI Model and Protocols	11
What is a socket?	13
Address family control	13
Socket types	14
UDP Protocol	14
UDP Socket	15
Creating socket	15
Binding socket	16
Questions:	18
UDP packet (datagram)	18
Sending a message inside UDP packet	19
Receiving message	20
Dealing with Big Amount of Data	21
Sending big message in multiple UDP packets with no additional header but using defined message terminator	21
Receiving big message in multiple UDP packets with no additional header but using	

defined message terminator	22
UDP Multicasting	
23	
Sending Multicast	
23	
Receiving Multicast	
25	
Programming Application layer protocol on top of UDP	
27	
Part 1: Message Board Protocol	
27	
Designing the protocol	
27	
1	
Implementation of the protocol	
31	
Client-side	
32	
Limitations (considered in protocol)	
32	
Server-side	
32	
Part 2: Stateful Protocol (Introducing sessions on UDP)	
33	
Design	
33	
TCP Protocol	
35	
TCP Socket	
36	
Creating socket	
38	
Binding socket	
40	
Listening	
41	
Accepting Connections (server-side)	
42	

Connecting (client-side)	43
Questions:	43
TCP packet/header	43
Receiving using TCP connection	44
Sending a message using TCP connection	46
Dealing with Big Amount of Data	47
Sending big message using TCP	47
Receiving big message using TCP	47
TCP socket shutdown routines	49
Programming Application layer protocol on top of TCP	49
Server-side	50
Client-side	50
Changes in MBoard protocol	51
Handling broken pipe exceptions	51
3 Threads	52
Introduction	52
What is multitasking?	52
Processes and Threads	53
Processes	53
Threads and	54

Advantages of threading	55
Thread Managers	55

Kernel-level thread managers	55
User-level thread managers	56
Python thread manager	56
Threads modules in python	56
Thread module	56
Threading module	57
Declaring a thread:	58
Using Argument with thread:	58
Identifying the current thread:	59
Logging debug:	59
Daemon threads:	60
Locks in threads:	62
Conditional Variables in threads:	64
Events in threads:	66
Multiprocessing module	66
4 Shared state	67
Introduction	67
Threads in network applications	67
Simple example	67
Client-server architectures	70
Request-Response protocols	70
Adding thread models to the server side	72
Thread-per-connection using different thread models	72
Thread-per-request using different thread models	76
Combining thread-per-socket and thread-per-request	81
Combining threading and multiprocessing	82
.....	83
Shared state	86
Short-lived vs. Long-lived TCP connections	86
Make clients get the updates immediately	88
Concurrent modification	88

5 Remote Procedure Calls (RPC) and Distributed Objects (DO)	90
Remote Procedure Calls	90
How Does it Work?	90
Programming RPC	92
RPC Advantages	92
RPC Disadvantages	93
RPC Implementations	93
RPC Application	94
Distributed Objects	94
Distributed Objects Frameworks	97
Pyro	97
How does it work?	98
Pyro Application	98
From objects to components	98
6 Indirect communication	104
Key properties	106
Space uncoupling	106
Time uncoupling	106
Space and time coupling	106
Asynchronous communication	106
Group communication	107
Publish-subscribe systems	108
Distributed events frameworks	109
Message queues	109

Introduction

These are the Lecture Notes for the course LTAT.06.007 Distributed Systems. The chapters appear in order to support learning the basic concepts of network programming and distributed systems. The aim is to give practical guidance and working examples for participants of the course to gain practical knowledge on how to build distributed applications.

Chapter 1

Introduction to Distributed Systems

Examples of Distributed Systems can be found everywhere: for example in the areas of finance and commerce, in applications to support the information society, creative industry and entertainment, healthcare, education, transport and logistics, science, environmental management *etc.* Here we will start with discussing the main properties of distributed systems as well as main challenges that one faces when starting dealing with them.

What is a distributed system?

Definition 1. A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages.

Definition 2. A distributed system consists of a collection of autonomous computers linked by a computer network and equipped with distributed system software. This software enables computers to coordinate their activities and to share the resources of the system hardware, software, and data.

When talking about a particular distributed systems architecture one might want to ask three key questions:

1. **What are the components?** Components in distributed systems are usually called nodes. First thing to ask is how many of what type and what purpose nodes are there? (For example computing nodes, special devices, sensors, network devices etc.)
The nodes are connected with links - the components for information transportation.
2. **How do the nodes communicate?** Here come the communication protocols, communication technology. Also, lot of the functional properties of a distributed system depend on link speeds (latency and throughput). To make the communication to happen there is a need for communication software.
3. **How the coordination between the components** is achieved? Here the usual aim is to achieve as reliable system as possible. For this quite a few issues need to be solved, like **fault-tolerance, security**, well-defined **resource sharing policies** to avoid **dead-locks, guaranteed delivery** of services etc.

Sharing resources

All distributed systems involve certain resources to be shared between the components with the end users.
What are these resources?

The resources can be **Hardware** (computers, servers, computer clusters, laptops, tablets, smartphones and smart-watches, Internet of Things (IoT) devices, sensors, data bandwidth on routes between nodes, CPU cycles etc.) Another group of resources can be described with the common term – **Data**. This involves data stored in databases, storage devices etc. But this includes also all kind of software together with its development (both proprietary as well as community-developed open source software.)

Note though that not every single resource is meant for sharing. But for those resources that are shareable we have to say that different resources are handled in different ways. However, some generic requirements can be outlined.

To be able to access some resource one needs to be able to name it. This means, we need a **(i) namespace** for identification of a group of resources. In order to get it working in a distributed environment there should be a mechanism for **(ii) name translation to network address**. And of course, in case of multiple access, possibly in a competing setup, well-defined **(iii) synchronization** is needed to avoid inconsistencies, dead-lock and starvation situations.

How to characterize a distributed system?

From above we can conclude, that distributed systems are characterized by **(a) concurrency of components**, meaning that all parts of a distributed system are independent and at the same time are competing for shared resources. The second feature that can be outlined is the **(b) lack of a global clock**. This means, that each processor on each node is running on its own clockrate and there is no synchronization of CPU clocks. Independence of the components yields also **(c) independent failures of components**, meaning that the design of a distributed system needs to take into account the fact that whichever node can crash at whichever time or become unavailable because of broken communication. This leads to a famous quote by Leslie Lamport: *"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable."*

What makes distributed systems challenging?

One can easily name hundreds of challenges related to distributed systems' design and operation. We cover here only the following challenges: Scalability, Openness, Security, Heterogeneity of Components, Failure handling, Concurrency of components, Transparency and Providing quality of service.

Scalability

Consider an emerging new web-service, which is being developed while it still wasn't known and popular with users. Now say, suddenly word spreads around about this fantastic new possibility and the service will start to have 1000% more hits each month from all around the Globe. What can be a remedy? (You may know the answer that replication and caching might help here.) But the property of a distributed system to respond to growing number of requests is called Scalability.

Scalability is the ability to work when the system load or the number of users increases.

With designing and maintaining scalable distributed systems there exist a number of challenges:

- (1) How to control the cost of physical resources? – one could build a perfect system with an unlimited amount of resources while, on the other hand, one could always try to minimise the cost as much as possible with taking certain risks. As one cannot exactly predict the future, the answer here would be flexibility.

Like in elastic cloud systems nowadays.

- (2) How to control performance loss? This is one of the main aims in designing reliable and lively distributed systems. General idea is to spread the load around in the system dynamically.
- (3) Preventing software resources running out. For example, IPV4 32-bit Internet addresses were predicted to run out already around the Millennium time.
- (4) Avoiding performance bottlenecks. This is actually one of the main concerns with the challenge (2). Careful planning and designing of the system is the way to go.

Openness

When talking about an open computer system the key question to ask is: *(A) Can the system be extended and reimplemented in various ways?* The main criteria to be able to answer "yes" to this question is that the key interfaces need to be published.

One can view a distributed system as a collection of independent computer systems that work together to fulfill a common goal. Therefore, with an open distributed system one should ask in addition to question (A) the following: *(B) Can new resource-sharing services be added and made available for use by variety of client programs?* To achieve this one would need (B1) Uniform communication mechanism and (B2) Published interfaces to shared resources.

Security

Security issues are gradually becoming more and more prevailing with computer systems in distributed environments making their way often even into the daily news headlines. This is due to real values are involved in the system, like bank transfers in digitalized transactions being moved over the networks connecting certain dedicated servers or credit card payments for goods or services by real users. This makes it an attractive target for hackers trying to interfere with the processes with the aim of earning or stealing some money, or some political reasons, or just out of the fun of hacking by itself. Common term for all security-related questions in distributed systems is **Cybersecurity**.

The main three terms that are the building blocks in distributed systems' security are:

Confidentiality - Protection against disclosure to unauthorized individuals – the techniques on how to conceal third parties from being able to access the actual content of the information even when being able to possess a copy of the data in an encrypted form. Key techniques here are (symmetric and unsymmetric) cryptographic algorithms providing the access to the original content of a message only by having corresponding credentials.

Integrity - Protection against alteration or corruption – ensuring that messages have not been changed or corrupted while travelling in an open network. The common technique is to use digital signatures based on public key encryption algorithms for proof of integrity.

Availability - Protection against interference with the means to access the resources – an effort to avoid denial of access and denial of service to shared resources in a distributed system.

Cybersecurity is a discipline by its own dealing with a big variety of attacks and protection techniques to avoid them. Some of the concepts are well-provable with relevant mathematical approaches while some are in a state of a conjecture with no known proof of existence of the opposite claim (like the conjecture, that the only way to find prime number

multipliers of a big number is to use the method of trial-and-error – the basis of RSA algorithm for public-private key encryption systems.)

Heterogeneity of Components

With distributed systems we can tell two things for sure: 1. each of the nodes has some connection to at least one other node in the system and 2. each node has at least one CPU connected to the network interface running certain control program to be able to provide the ability of communication with the rest of the distributed system. All the rest might be (and usually is) difficult to define with common claims due to huge variety in hardware and software resources, operating systems, network connections, programming languages used for creating the needed functionality *etc.* All this contributes to the challenge of creating an operative environment with huge heterogeneity within the system components, connections and their control. An adopted way to handle heterogeneity is to use **middleware** - software layer providing programming abstraction while at the same time masking heterogeneity of underlying network technologies, pieces of hardware in use, operating systems, and bundles of software.

Failure handling

In an ideal World there would not be any failures at all. But with distributed systems we are talking about real thing developed by people, usually many people in collaboration and often even spread around different locations around the Globe. Therefore one has to face the situations created by failures that have already occurred, but much better: to try to forecast all possible failure conditions to be able to prevent them before they have had their chance to show up. Developing a distributed systems is not just programming. It is about designing and playing through different scenarios with all kind of exceptions taken into account with the thought in mind that when nothing works – how to still get the work done in the end?

There are certain techniques for dealing with failures. As the first measure one has to be able to **detect the failures**. Detected failures can then be **masked** to allow taking relevant action to handle them. With distributed systems one always has to take into account possibility of messages not going through due to network overloads and downtime. Therefore one has to design scenarios for message retransmission on the need. More serious cases are the ones with a crucial node itself crashing or becoming unavailable. One can consider **replication** of key services, databases, resources for the replica being able to take over the responsibilities as soon as such need occurs.

Therefore, one of the key distributed systems' properties as well as challenges is their ability of **tolerating failures**. When designing a system one might want to ask – what is the level of failure situations that the system is still able to **recover** from? The main goal is to achieve **high availability** – measure of the proportion of time that the resource is available for use.

Concurrency of components

As soon there are some resources to share between multiple parties there is built in a question of how to handle situations when there are more than one counterpart competing for the same resource simultaneously. It adds a new order magnitude to the complexity. In such cases there exist possibilities for dead-lock situations. But if one is not careful enough, there might also occur possibilities for unfair treatment towards certain counterparts and possibilities for their starvation. A classical illustration to this is the example of Dining Philosophers.

In the case of distributed systems the challenge from the possibility of messages getting delayed or dropped adds an additional level of complexity. For a well-designed systems the proof of no deadlock situations as well

as fair handling of shared resources' requests becomes therefore much more complex.

Transparency

While a user connects to the Internet with say, a request for certain information about different possibilities to travel from Tartu to say, Venice, [s]he does not want to know all the details about the underlying service details and sub-requests from different airlines, bus or train services etc. that happen seemlessly in the background. The user wants the answer to appear on her/his device screen and as quickly as possible. The service is **transparent** to the user in a way that there is no need to realize that there is a distributed system underneath serving the request and composing the answer.

Transparency - Concealment from the user and the application programmer of the separation of components in a Distributed Systems for the system to be perceived as a whole rather than a collection of independent components.

One can say that there exist **access transparency** - the user percieves (accesses) the distributed system as it were a single node instead of (possibly complex) set of queries between different nodes. Also, the user does not care where exactly the piece of information came from - **location transparency** - nor does the user care that accessed resource physical location got recently changed from a server in US to Ireland - **mobility transparency**. Also, usually the user does not want to get a message telling that the query was already n-th such a request at the very moment and it got served as the second in the queue - no, the user just needs the information - with the **concurrency transparency**. Likewise, the user need not to know that the piace of information came from a certain cache of got served by the server replica number X - **replication transparency**. Either, in case of any errors that might have occurred the user would like the system to get recovered instantly without any substantial delays or error messages - **failure transparency**.

All the above mentioned challenges are together part of the final challenge we cover here:

Providing quality of service

Main nonfunctional properties of distributed systems that affect **Quality of Service (QoS)** are **Reliability**, **Security**, and **Performance**. With these properties having been met by the design and implementation, there is still possibility for the system loosing its original functionality due to changing system loads from increasing volume of requests as well as needs for changes in the system architecture itself to adapt to changing requirements. This is described as the system **adaptability**.

Chapter 2

Introduction to Computer Networking

Introduction

In this chapter we will introduce IP network and learn basics of IP communications. We start refreshing our knowledge of OSI model, make sure we remember what Physical Layer, Link Layer, Network and Transport layer standing for. Next we will discuss the term “protocol” and try to relate it to OSI model, making sure we understand the protocols can be related to different Layers of OSI model.

OSI Model and Protocols

1. *OSI model is a conceptual model, which characterizes and standardizes the communication functions of any computing system or telecommunication without regard to the technology involved or their internal structure.*

The model is divided into abstraction layers and they are as follows:

7. Application layer: Network Process to Application [Serves as the window for users and application processes to access the network services]
6. Presentation layer: Data Representation and Encryption [Formats the data to be presented to the Application layer. It can be viewed as the Translator for the network]
5. Session layer: Inter-host Communication [Allows session establishment between processes running on different stations]
4. Transport layer: End-to-End Connections and Reliability [Ensures that messages are delivered error-free, in-sequence, and with no losses or duplications]
3. Network layer: Path Determination and IP (Logical Addressing) [Controls the operations of the subnet, deciding which physical path the data takes]
2. Data link layer: MAC and LLC (Physical Addressing) [Provides error-free transfer of data frames from one node to another over the Physical layer]
 1. Physical layer: Media, Signal, and Binary Transmission [Concerned with the transmission and reception of the unstructured raw bit stream over the physical medium]

OSI model was created by ISO and is considered a “reference” model to explain aspects of network communications where multiple protocols and technologies are evolved. OSI model is however not a standard that all the protocols are following.

-
2. *Communication protocol is a set of rules allowing two or more endpoints to exchange information. Rules defining the way the user data is broken down into data units*
 3. *Protocol data unit unique piece of information delivered among the endpoints. Data unit is considered atomic (unbreakable) in the scope of protocol.*

Data unit may contain control information (header) or the user data (payload). In a layered network, the data unit of a higher layer becomes a payload of lower layer's data unit. Moreover, big data units of higher layer may be broken down and assigned as a payload to multiple data units of lower layer. Example of TCP/IP running on Ethernet: IP packet (Layer 3) on top of multiple Ethernet frames (Layer 2).

Data units:

- Data: higher abstraction layers (application, presentation, session)
- Segment: transport layer
- Packet: networking layer
- Frame: data link layer
- Bits: physical layer

Data movement between layers of the OSI model:

- Encapsulation:
- De-encapsulation

4. *TCP/IP model (simplified) derived from OSI model, however is only focusing on aspects of TCP/IP. For example session layer here is absent as the sessions in TCP/IP might be organized by transport layer (TCP) or application layer (UDP).*

1. Application layer
2. Transport layer
3. Internet layer
4. Network interface (Link) layer

Question: Let's remember a bit of Network Technology and try to organize the following protocols according to TCP/IP model: ARP, USB, RS-232, IEEE 802.11 , PPP , IPX , AppleTalk , IGMP , HTTP , FTP , IMAP, SMTP

What is a socket?

5. *A socket is one end-point of a two-way communication link between two programs or applications running on the network.*

In the scope of operating system the socket is a descriptor (identified by unique number).

6. *A descriptor is an abstract indicator (a unique number) assigned by the OS to any resource currently opened for I/O operations (read, write or both).*

The operations allowed on socket are therefore similar to file:

- `socket()` - Creating a bare socket object
- `close()` - Closing the socket

however additional methods are there on socket:

- `bind()` - Binding socket to local IP address and port
- `connect()` - Connecting a local socket to a remote address and port. Is required in case of TCP and is not there for UDP. Can you explain why ? (If not we will anyway discuss it later).
- ... more methods of socket you may find from socket API of Python

and in place of read/write/seek there are:

- `send():` Send data through connected socket (TCP only)
- `sendto():` Send data to particular recipient in UDP protocol
- `recvfrom():` Receive data from particular recipient in UDP protocol

The role of the socket is to allow opening a communication channel that is used by programs or applications to transmit data back and forth locally or across Internet.

Sockets have two primary properties controlling how the transmission of data is done. These properties can be resumed as follows:

1. The address family controls the Open Systems Interconnection (OSI) network layer protocol used
2. The socket type controls the transport layer protocol.

Address family control

In python we have three supported address families:

- 1. **AFINET**, which is used for IPv4 Internet addressing.
- 2. **AFINET6**, is used for IPv6 Internet addressing.
- 3. **AFUNIX** is the address family for Unix Domain Sockets (UDS).

GeneralKnowledge:

- **IPv4** addresses are composed of four octal values separated by dots (e.g. 10.10.10.6). This latter are commonly referred to as IP addresses.
- **IPv6** is the next generation of Internet protocol, which supports 128-bit addresses.
- **UDS** is an inter-process communication protocol available on POSIX-compliant systems

Socket types

When it comes to socket types usually we have either **SOCK_DGRAM** for user datagram protocol (UDP) or **SOCK_STREAM** for Transmission control protocol (TCP).

Note: Python contains other socket types but they are less commonly used, therefore they are not covered here.

UDP Protocol

7. User Datagram Protocol (UDP) is a Transport Layer communication protocol relying on IP protocol for addressing the endpoints. It is used for establishing low-latency and loss tolerating (data delivery not guaranteed by UDP) connections between applications on the net.

The UDP protocol is a message-oriented protocol and it does not need a long-lived connection (there is no dedicated pipes between the endpoints, and each message has to be addressed explicitly). The messages sent via UDP must fit within a single packet and the delivery is not guaranteed. This makes UDP communication very analogical to sending a mail by post without tracking it or delivery notification. In addition the size of the deliverable is limited by maximal datagram size, and it is impossible to send big amounts without fragmentation.

Note: Summarizing features of UDP

- Data sent in datagrams, payload of the datagram can not be bigger than the maximal size of the datagram. Bigger blocks of data have to be split manually before sending and assembled back on delivery. [65,535 bytes (8 byte header + 65,527 bytes of data)]
- There is no delivery control, datagrams may be lost. If the delivery needs to be confirmed it has to be programmed manually.
- The order of the packets is not preserved during delivery and they may be received in different order. If the order is important (for example for sending fragmented blocks) the sequencing has to be programmed manually.
- It is possible to send one packet to many recipients (broadcast sending)
- It is possible to receive from any source by the same socket.
- **It is possible to send and receive from the same socket interleaved.**

In the next sections, we will step by step implement the simple messaging protocol between 2 endpoints relying on UDP.

UDP Socket

First of all lets make sure we understand the “creating” and “binding” operations of the sockets.

Creating socket

8. Creating a socket is operation of registering new socket descriptor in OS (like opening a file for reading or writing). As a result of this operation, the OS has a new descriptor in the list of open descriptors.

Please note that “creating” the socket is like building the mailbox of your home.

Here is an example code showing how to create the UDP socket and how to check what descriptor assigned to

```
# From socket module we import the required structures and constants. from socket import AF_INET,  
SOCK_DGRAM, socket  
  
# Sleep function will be used to keep application running for a while from time import sleep  
# And we also will need the process id given by OS when we execute this code from os import getpid  
# Main method if __name__ ==  
'__main__':  
    print 'Application started'  
    print 'OS assigned process id: %d' % getpid()  
  
    # Creating a UDP/IP socket s =  
    socket(AF_INET, SOCK_DGRAM)  
    print 'UDP Socket created'  
    print 'File descriptor assigned by OS:', s.fileno()  
    wait_secs = 60*5 print 'Waiting %d seconds before termination ...' % wait_secs  
    sleep(wait_secs)  
    print 'Terminating ...'
```

it by the OS:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

Executing the code should print the following result, please note that letters X and Y will be denoting the process ID and the descriptor respectively and are specific to your environment (each time you run the code it will give different numbers):

```
Application started OS assigned process id: X  
UDP Socket created  
File descriptor assigned by OS: Y  
Waiting 300 seconds before termination ...
```

Now while the code is running for 5 minutes we have time to refer to the list of registered descriptors of the OS and check if we see our running application process with a corresponding descriptor:

1. in Linux systems the Kernel hold raw process information in folder-like structures (populated by Kernel). So the very primitive folder listing call is enough (the ls command). Issue the following command in shell, make sure in place of X you use the process ID reported by the executed Python code and in place of Y - the descriptor):

```
1 ~$ ls -l /proc/X/fd/Y
```

... in the output you should now see the socket's descriptor that were assigned by the OS to a process number X:

```
lrwx----- 1 user user 64 Aug 18 13:13 Y -> socket:[129069931]
```

2. in MAC systems the unbound sockets are visible by lsof command, so it is enough to print all the UDP descriptors (lsof -i 4udp) and filter the ones assigned to specific process ID (grep X). Issue the following

```
$ lsof -i 4udp |grep X
```

command in shell, make sure in place of X you use the process ID reported by the executed Python code:

1

... in the output you should now see the socket's descriptor that were assigned by the OS to a process number X:

```
Python      2062      user      3u IPv4 0x90964583fac7dd81          0t0 UDP *:*
```

If you can see the descriptor of a socket you have created in your code, you may continue to the next step of binding a socket. The currently running application (if not terminated after 5 minutes) may be terminated manually (Ctrl+C if running in shell explicitly). You may notice that after the application was terminated all its descriptors (including our UDP socket) are gone (closed automatically by the OS). Alternatively in your code you may issue close() method on socket before terminating the application (2 last lines of the reference code

```
sleep(wait_secs)  
print 'Closing the UDP socket ...' s.close()  
print 'Terminating ...'
```

are modified):

1
2
3
4

Binding socket

We know already:

1. Sockets represent transport endpoints exposed to an application willing to transfer data over network 2.

UDP protocol is a transport protocol on top of IP addressing protocol We will learn:

1. Transport address or socket address
 2. UDP port
 3. Binding to random free port
 4. Binding to loop-back address only
 5. Binding to all addresses
- 9.** *Binding a socket is an operation of assigning a transport address to the socket (like declaring an address of the new mailbox to the post service - so the postman can deliver/send-out mails to/from the mailbox). As a result of this operation, the OS has the UDP socket in the list of bound sockets. Moreover at this point the socket can actually send/receive data and it is visible to remote systems (if there is network connectivity, if socket is not locked by firewall and if remote endpoint is not hidden behind NAT).*
- 10.** *Transport address or socket address is the combination of an IP address and a port number.*
- 11.** *UDP port is used to help distinguish different user requests and optionally a checksum to verify that the data arrived intact.*

```
from socket import AF_INET, SOCK_DGRAM, socket
from os import getpid

if __name__ == '__main__':
    print 'Application started'
    print 'OS assigned process id: %d' % getpid()

    s = socket(AF_INET, SOCK_DGRAM)
    print 'UDP Socket created'
    print 'File descriptor assigned by OS:', s.fileno()
    # Binding the UDP/IP socket to loopback address and port 7777
    s.bind(('127.0.0.1', 7777))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Wait for user input before terminating application
    raw_input('Press Enter to terminate ...')

    s.close()
    print 'Terminating ...'
```

In the following code we will create socket and bind it to loop-back address and UDP port 7777:

```
6
7
8
9
10
11
12
13
14
15
16
17
18
```

After starting the application it will stay running till the user hits “Enter” button. Keeping the application running, refer to active network connections table of the OS by issuing netstat command: 1. In Linux systems, from the shell

```
1 ~$ netstat -anpu | grep X
```

where X is the process ID printed by the executed Python code. As a result the following information will be shown (where X is PID reported by the executed Python code):

```
udp 0 0 127.0.0.1:7777 0.0.0.0:* X/python
```

2. In Mac from the terminal

```
1 ~$ netstat -n -p udp| grep X
```

where X is the port number used in the Python code (e.g. 127.0.0.1.7777). The results will be as follows:

```
udp4      0      0 127.0.0.1.7777          *.*
```

As you can see in the output, the corresponding UDP socket issued by Python process (PID X) is registered with transport address 127.0.0.1:7777 which means the other applications/systems may refer to our running Python code by sending the UDP packets to UDP port 7777 on IP 127.0.0.1 (which limits the scope to the local OS only).

Questions:

1. What will happen to our socket if we modify the port number 0 in place of port 7777 in our code ?
Please try it and take a look into netstat table.
2. What if do not want to limit the app. to loop-back address 127.0.0.1 ? What should be modified in the code ?
3. How do we bind to all interfaces without specifying all the IP-s in particular ?
4. Can we bind to port number lower than 1024 ? Try binding to port 1023 for example ...

Now we know how to create and bind UDP sockets for the application to be able to communicate to the other instances in the IP network using UDP protocol. Next we will discuss how to send / receive data over UDP sockets.

UDP packet (datagram)

12. *UDP packet or UDP datagram is transport unit of the UDP protocol. It contains 8 bytes of UDP header and the payload. The UDP header contains the following:*

- *Source port number (2 bytes)*
- *Destination port number (2 bytes)*
- *Datagram size (2 bytes)*
- *Checksum (2 bytes), computed over entire payload, other UDP header fields and IP header fields.*

Considering the fact that there is only 2 bytes reserved for storing the size of attached payload is limited to 64K.

13. *Transport unit (also referred as message unit or protocol data unit) is information delivered as an atomic unit among peer entities. Atomic here means no further fragmentation in the scope of the protocol. Data unit contains control information, address information and user data (payload).*

We may think of UDP datagram as an envelope for a written brief we would like to send using post service. **14.**

The process of enveloping the user data into one or many datagrams is called encapsulation

15. *De-encapsulation is then reverse process of assembling the user data from one or many received datagrams*

Sending a message inside UDP packet

In Python language the socket API takes care of creating UDP datagrams from user data. Here is an example of

```
from socket import AF_INET, SOCK_DGRAM, socket
if __name__ == '__main__':
    print 'Application started' # Creating a UDP/IP socket s =
    socket(AF_INET, SOCK_DGRAM)
    # Binding the UDP/IP socket to address and port s.bind(('127.0.0.1',7777))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Sending the message message = 'Hello
    world' destination = ('127.0.0.1',7778)
    s.sendto(message,destination)
    print 'Sent message to %s:%d' % destination
    print 'Payload length %d bytes: [%s]' % (len(message),message)
    raw_input('Press Enter to terminate ...')
    print 'Closing the UDP socket ...' s.close()
    print 'Terminating ...'
```

sending the string "Hello world!" to the peer by address 127.0.0.1:7778:

```
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

```
Application started
Socket is bound to 127.0.0.1:7777
Sent message to 127.0.0.1:7778 Payload length 11
bytes: [Hello world!] Press Enter to terminate ...
```

The output is then looking like:

5

When we issue `sendto(...)` method on the bound socket object the new datagram is created using the data we provide over first argument (the message variable) then the destination is set to UDP header considering the second argument (the destination address IP and port). Afterwards, the source address is set using the address our socket is bound to (`s.getsockname()`). At the end, the length of the payload is set considering the size of the data (length of the message string). Finally the checksum is calculated using the payload and the filled header fields and the ready datagram is delegated to OS to be sent.

As you can see here we do not have a structure for an UDP datagram to manipulate, all manipulation is done by socket API. We have therefore to take into account the facts: that data blocks bigger than 2^{16} have to be split manually into smaller blocks before issuing `send()` method of the socket API. The second fact is that if skip the binding socket and we send the data explicitly after creating the socket, the socket API will do binding automatically and socket will be bound to ('0.0.0.0',0). Which means the first unused port and all addresses on all interfaces. You may test it using alternative code below:

```

from socket import AF_INET, SOCK_DGRAM, socket
if __name__ == '__main__':
    print 'Application
        started' # Creating a UDP/IP socket s =
    socket(AF_INET, SOCK_DGRAM)
    # Sending the message (socket is still unbound) message = 'Hello
    world!' destination = ('127.0.0.1',7778)

1
2
3
4
5
6
7
8
    s.sendto(message,destination)
    print 'Sent message to %s:%d' % destination
    print 'Payload length %d bytes: [%s]' % (len(message),message) # Let's check how the socket
    API did bind the socket print 'Socket was automatically bound to %s:%d' % s.getsockname()
    raw_input('Press Enter to terminate ...')
    print 'Closing the UDP socket ...' s.close()
    print 'Terminating ...'

9
10
11
12
13
14
15
16
17

```

```

Application started
Sent message to 127.0.0.1:7778
Payload length 11 bytes: [Hello world!]
Socket was automatically bound to 0.0.0.0:40154 Press Enter to terminate ...

```

The output is then looking like:

5

As you can see the socket API did automatically assign the pattern 0.0.0.0 as destination IP and randomly pick up port 40154 for our socket.

Receiving message

Communication is typically happening between at least 2 entities, therefore in our example we need the second entity. The first one was the sender described in previous section, and it's transport endpoint (socket address) was 127.0.0.1:7777. The listener is the second and its address may be easily deducted from the sender's code above (it is 127.0.0.1 7778). So both endpoints happen to run on the same host OS. Let's introduce the code of

```
from socket import AF_INET, SOCK_DGRAM, socket
if __name__ == '__main__':
    print 'Application
        started' # Creating a UDP/IP socket s =
    socket(AF_INET, SOCK_DGRAM)
    # Binding the UDP/IP socket to address and port s.bind(('127.0.0.1',7778))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Receiving the message, maximal payload to receive in one peace recv_buffer_length = 1024
    print 'Waiting for message ...'
    message,source = s.recvfrom(recv_buffer_length)
    print 'Received message from %s:%d' % source
    print 'Payload length %d bytes: [%s]' % (len(message),message)
    raw_input('Press Enter to terminate ...')
    print 'Closing the UDP socket ...' s.close()
    print 'Terminating ...'
```

the listener/receiver:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

As you can see the code is not that much different compared to the sender's code, however we can notice one new function `recvfrom` called on the socket with 1024 as an argument and returning the payload (message) and the sender's address (source) at once. The value 1024 is critical as we do not know what will be the size of the message, we just agree we receive in 1024 bytes in one attempt (and if there is more we have the call `recvfrom` again).

The output is then looking like:

```

Application started
Socket is bound to 127.0.0.1:7778 Waiting for message
...
Received message from 127.0.0.1:50516 Payload length
11 bytes: [Hello world!] Press Enter to teminate ...

```

5

Our message is successfully received.

Please note, here recvfrom() method will block the code execution till something is received on corresponding socket by the OS. Calling blocking receive with no timeout we assume the remote endpoint will eventually send the data. Otherwise the receive method will block forever as the default timeout is set to None.

Dealing with Big Amount of Data

Sending big message in multiple UDP packets with no additional header but using defined message terminator

```

from socket import AF_INET, SOCK_DGRAM, socket

if __name__ == '__main__':
    print 'Application started' # Creating a UDP/IP socket s =
    socket(AF_INET, SOCK_DGRAM)
    # Sending the message (socket is still unbound)
    # We will use the break line to indicate the end of message
    # then the receiver may receive without knowing the full length
    # of the message he is receiving term = '\n'
    message = 'Hello world!'*7000+term
    max_len = 1024 # Maximal pay-load size to send in one UDP packet destination = ('127.0.0.1',7778)

    # In case there is more data, send it in peaces m = message # this will be reduced in
    progress of sending sent = 0      # how much was sent parts = 0          # count
    total sent packets
    # Cut the message into blocks and send out till the message is over while sent < len(message):
        m_send = m[:max_len] if len(m) > max_len else m m = m[max_len:]
        sent += s.sendto(m_send,destination) parts += 1
        print 'Sent %d bytes of %d ...' % (sent,len(message))

    print 'Sent message to %s:%d in %d parts' % (destination + (parts,)) print 'Pay-load length %d
bytes: [%s]' % (len(message),message)

```

1
2
3
4
5
6
7

```
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
```

```
raw_input('Press Enter to terminate ...')
print 'Closing the UDP socket ...' s.close()
print 'Terminating ...'
```

```
31
32
33
34
```

Receiving big message in multiple UDP packets with no additional header but using defined

message terminator

```
from socket import AF_INET, SOCK_DGRAM, socket

if __name__ == '__main__':
    print 'Application started' # Creating a UDP/IP socket s =
    socket(AF_INET, SOCK_DGRAM)
    # Binding the UDP/IP socket to address and port s.bind(('127.0.0.1',7778))
    print 'Socket is bound to %s:%d' % s.getsockname()

    # Receiving the message, maximal pay-load to receive in one peace recv_buffer_length = 1024
    term = '\n' # Terminator indicating the message is over message = "" # This will grow in
    progress of receiving

    print 'Waiting for message ...'
    # Append message block by block till terminator is found while not
    message.endswith('\n'):
        m,source = s.recvfrom(recv_buffer_length)
        print 'Received block of %d from %s:%d' % ( (len(m)),) +source)
        message +=m

    print 'Total length %d bytes: [%s]' % (len(message),message)
    raw_input('Press Enter to terminate ...')

    print 'Closing the UDP socket ...' s.close()
    print 'Terminating ...'
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26

Reflection: as you can see the receiver does not distinguish the senders (assuming there is only one sender sending at once). What will happen if multiple senders will send simultaneously messages to the receiver ?

Ensuring Packet Arrival Order

UDP protocol does not guarantee that the packets arrive in order; however, if the design requires it then the developer must ensure the order manually, typically using the sequence markers inside blocks. Regarding the previous send/receive example developer would want to add a sequence number at the beginning of each block created. Receiver must then be aware of the existence of a sequence marker in the message, and handle the order of received blocks before appending to the message being received.

Note: In our example we chose to use the terminator to mark the end of message, however we could avoid using special terminator character and just add the total message length into first block of the message.

UDP Multicasting

Why multicasting is important? Imagine you are in a situation where you have to send the **same** message or data to many hosts (without the need of iteratively sending the message to each peer individually). Multicasting provides a solution to our case. Since in our case multicasting is based UDP, the transmission is by default not reliable. As we explain before using the UDP does not give us any feedback about the status of the process of delivering the message or the data. Therefore, the user who sent a multicast has no idea how many peers really received the packets (except if the peers were designed to send a feedback manually).

Note: Multicast allows you to send message only to the interested parties and the message is transmitted only once (saves a lot of bandwidth). In order to receive multicast the receiver must be subscribed to particular multicast group in the network. Multicast group is just a specific IP address ranges (according to IANA):

- 224.0.0.0 - 224.0.0.255 [Reserved for special “well-known” multicast addresses].
- 224.0.1.0 - 238.255.255.255 [Global scope multi-cast, for cross-network multi-casting].
 - This is however typically blocked by ISPs (preventing flood from private to public subnets).
- 239.0.0.0 - 239.255.255.255 [Administratively-scope (local) multicast addresses].
 - This section of addresses have actually more segregation into sub-categories, but we are not going into details here as we will be using only the two first address ranges.

Sending Multicast

Now we will create a multicast sender that will send a message to a multicast group. The following example illustrate how to do it:

```
#!/usr/bin/python
#
# Implements Python UDP multicast sender
#
# Based on the code from:
# http://stackoverflow.com/questions/603852/multicast-in-python

"""Simple UDP Multicast sender
@author: devel
"""

# Tired of print ----- # ... setup Python logging as defined in:
# https://docs.python.org/2/library/logging.html import logging

1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

```

15FORMAT = '%(asctime)-15s %(levelname)s %(message)s'
16logging.basicConfig(level=logging.DEBUG,format=FORMAT)
17LOG = logging.getLogger()
18# Needed imports -----
19# Socket from socket import socket
20# Socket attributes
21from socket import AF_INET, SOCK_DGRAM, SOL_SOCKET, SO_REUSEADDR
22from socket import IPPROTO_IP, IP_MULTICAST_LOOP, IP_DEFAULT_MULTICAST_TTL
23# from time import sleep
24# Constants -----
25# The multi-cast group address we are going to use
26__mcast_addr = '239.1.1.1'
27# The multi-cast group port
28__mcast_port = 53124
29# The multi-cast time-to-live (router hops)
30__mcast_ttl = 4
31# Broadcast interval seconds
32__mcast_delay = 5
33# Particular message the server sends to identify himself
34__mcast_message = u'Hello world!'
35# Initialization -----
36# Declare UDP socket
37__s = socket(AF_INET, SOCK_DGRAM)
38LOG.debug('UDP socket declared ...')
39# Reusable UDP socket? I am not sure we really need it ...
40__s.setsockopt(SOL_SOCKET,SO_REUSEADDR,1)
41LOG.debug('UDP socket reuse set ...')
42# Enable loop-back multi-cast - the local machine will also receive multicasts
43__s.setsockopt(IPPROTO_IP,IP_MULTICAST_LOOP,1)
44LOG.debug('Enabled loop-back multi-casts ...')
45# Set multi-cast time-to-live, let say we go up 4 sub-nets
46__s.setsockopt(IPPROTO_IP, IP_DEFAULT_MULTICAST_TTL, __mcast_ttl)
47LOG.debug('Set multicast time-to-live [%d] hops ...', __mcast_ttl)
48# For multi-casting we do not need any specific bindings
49# ... socket will be bound automatically to interface with sub-net providing
50# default gateway, the port will be also defined randomly
51# Serving -----
52# Server forever (Ctrl+C to kill)
53__n = 0
54while 1:
55try:
56# Send out multi-cast message
57__s.sendto(__mcast_message, (__mcast_addr, __mcast_port))
58__n += 1
59LOG.debug('[%d] Multi-cast sent ...', __n)
60sleep(__mcast_delay)
61except (KeyboardInterrupt, SystemExit) as e:

```

```

62LOG.info('Terminating server ...\\n')
63break
64# Termination -----
65# Clean-up the socket
66__s.close()

    LOG.info("Server terminated ...")

67LOG.debug('Socket closed ...')
68

```

```

2016-08-19 16:28:05,828 DEBUG UDP socket declared ...
2016-08-19 16:28:05,828 DEBUG UDP socket reuse set ...
2016-08-19 16:28:05,828 DEBUG Enabled loop-back multi-casts ...
2016-08-19 16:28:05,828 DEBUG Set multicast time-to-live [4] hops ...
2016-08-19 16:28:05,828 DEBUG [1] Multi-cast sent ...
2016-08-19 16:28:10,833 DEBUG [2] Multi-cast sent ...
2016-08-19 16:28:15,838 DEBUG [3] Multi-cast sent ...
2016-08-19 16:28:20,841 DEBUG [4] Multi-cast sent ...
^C
2016-08-19 16:28:24,321 INFO Terminating server ...
2016-08-19 16:28:24,322 DEBUG Socket closed ...
2016-08-19 16:28:24,322 INFO Server terminated

```

Output should look like:

5

10

Receiving Multicast

At this level, we will create the receiver server.

The example below illustrate source code of the multicast receiver:

```

#!/usr/bin/python
#
# Implements Python UDP multicast receiver #
# Based on the code from:
# http://stackoverflow.com/questions/603852/multicast-in-python

"""Simple UDP Multicast receiver
   @author: devel
"""

# Tired of print ----- # ... setup Python logging as defined in:
# https://docs.python.org/2/library/logging.html import logging
FORMAT = '%(asctime)-15s %(levelname)s %(message)s' logging.basicConfig(level=logging.DEBUG,format=FORMAT)
LOG = logging.getLogger()
# Needed imports -----
# Socket from socket import socket
# Socket attributes
from socket import AF_INET, SOCK_DGRAM, SOL_SOCKET, SO_REUSEADDR, SOL_IP from socket import
IPPROTO_IP, IP_MULTICAST_LOOP, IP_DEFAULT_MULTICAST_TTL from socket import inet_aton,
IP_ADD_MEMBERSHIP
# Constants -----
# The multi-cast group address we are going to use
__mcast_addr = '239.1.1.1'
# The multi-cast group port
__mcast_port = 53124
# The multi-cast time-to-live (router hops)

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26

```

27
28
29
30
|__mcast_ttl = 4
# Receiving buffer
__mcast_buffer = 1024
# Initialization -----
# Declare UDP socket
__s = socket(AF_INET, SOCK_DGRAM)
LOG.debug('UDP socket declared ...')
# Reusable UDP socket? I am not sure we really need it ...
__s.setsockopt(SOL_SOCKET,SO_REUSEADDR,1)
LOG.debug('UDP socket reuse set ...')
# Enable loop-back multi-cast - the local machine will also receive multicasts
__s.setsockopt(IPPROTO_IP,IP_MULTICAST_LOOP,1)
LOG.debug('Enabled loop-back multi-casts ...')
# Set multi-cast time-to-live, let say we go up 4 sub-nets
__s.setsockopt(IPPROTO_IP, IP_DEFAULT_MULTICAST_TTL, __mcast_ttl)
LOG.debug('Set multi-cast time-to-live [%d] hops ...', __mcast_ttl)
# Bind UDP socket to listen to multi-casts
__s.bind((__mcast_addr,__mcast_port))
LOG.debug('Socket bound to %s:%s' % __s.getsockname())
__s.setsockopt(SOL_IP, IP_ADD_MEMBERSHIP,
               inet_aton(__mcast_addr) + inet_aton('0.0.0.0'))
# Receiving -----
# Listen forever (Ctrl+C) to kill while 1:
try:
    # Receive multi-cast message
    message,addr = __s.recvfrom(__mcast_buffer)
    LOG.debug('Received From: %s:%s [%s]' % (addr+(message,))) except (KeyboardInterrupt,
    SystemExit) as e:
        LOG.info('Terminating client ...\\n') break
# Termination -----
# Clean-up the socket
__s.close()
LOG.debug('Socket closed ...')
LOG.info("Server terminated ...")

```

31
32
33
34
35
36
37
38
39
40
41
42

```
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
```

```
2016-08-19 16:27:59,413 DEBUG UDP socket declared ...
2016-08-19 16:27:59,413 DEBUG UDP socket reuse set ...
2016-08-19 16:27:59,413 DEBUG Enabled loop-back multi-casts ...
2016-08-19 16:27:59,413 DEBUG Set multi-cast time-to-live [4] hops ...
2016-08-19 16:27:59,413 DEBUG Socket bound to 239.1.1.1:53124
2016-08-19 16:28:05,828 DEBUG Received From: 172.17.161.232:54469 [Hello world!]
2016-08-19 16:28:10,833 DEBUG Received From: 172.17.161.232:54469 [Hello world!]
2016-08-19 16:28:15,838 DEBUG Received From: 172.17.161.232:54469 [Hello world!] 2016-08-19 16:28:20,841 DEBUG
Received From: 172.17.161.232:54469 [Hello world!] ^C
2016-08-19 16:28:23,850 INFO Terminating client ...
2016-08-19 16:28:23,850 DEBUG Socket closed ...
2016-08-19 16:28:23,850 INFO Server terminated ...
```

The output should look like this:

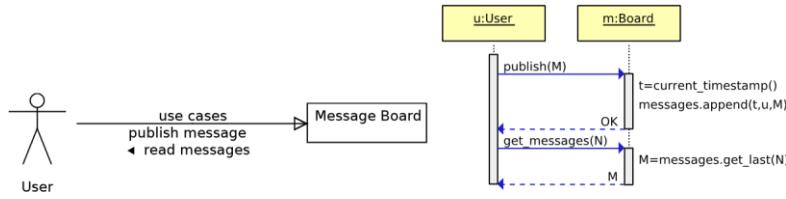


Figure 2.1: Message board scenario #1 use cases and interaction diagram

Programming Application layer protocol on top of UDP

Here we are ready to implement a protocol for an application, but first of all let's agree what sort of application we will program.

Let's say we want the public message board service to be shared with any number of users who want to publish or read messages. So we have two roles for network entities - the "Message Board" and the "User".

The required scenarios are:

- Scenario:
 - Part 1: The message board service is responsible for serving user requests.
 - * User may request to submit his message to public board
 - * User may request the list of messages currently published on the board –
 - Part 2: Complimentary requirement.
 - * User may request to submit big message (for example, text and pictures, videos or archive attachments)

Part 1: Message Board Protocol

Designing the protocol

Scenario #1 is illustrated in Figure 2.1. As we can notice the "User" refers to "Message Board" and can request to publish his message or he may request all messages to view. As result, the "User" is always the first who initiates the communication, as there is no use case when "Message Board" initiates communication. Therefore, the client-server architecture here is the most suitable: one board can serve many users and, one user may refer to only one board at once (user cannot change the board during runtime). Basically, once the application gets started, it is connected to the specified board and is staying connected till application dies (next time it is started user may specify different board to refer to).

We have declared 2 operations between the previously shown entities. Both operations evolve transmitting some information between the entities. And we already know our communication medium - it's the UDP transport protocol (we have to refer to Socket API of Python). Now let's agree how this information is transmitted, in other words: let's declare the protocol.

By definition, we need a special sort of message, where we specify the type of action (publish or retrieve) that is associated to this message. In addition, dependent on the type there can be the corresponding "text message" that has to be published or the number of the messages to retrieve in case of retrieval action.

In our case, the special message is exactly a definition of a protocol data unit or PDU (maximal transportable piece of data in one transaction). Plus, we have to carry our data protocol data units over UDP; therefore,

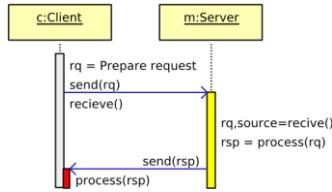


Figure 2.2: Sequence diagram of a simple Request-Response Protocol

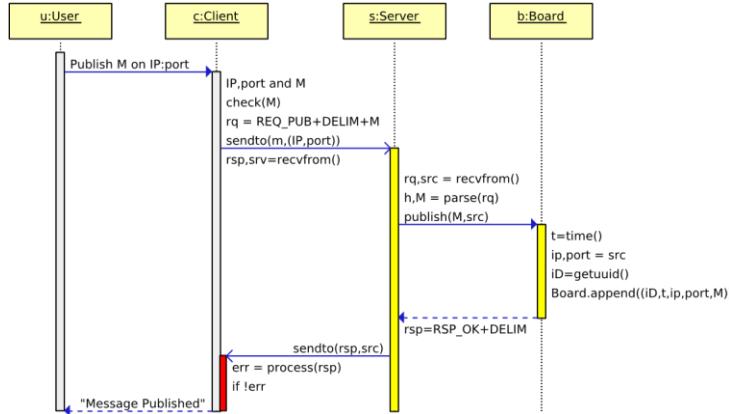


Figure 2.3: Sequence diagram of “Publish” routine

we should take into account the features of the UDP. The one that affects us most is the inability to send the big message reliably. So we either implement our own routines: for 1.) guaranteeing delivery 2.) preserving data. Alternatively we may just adapt our protocol data unit towards UDP. And in the first scenario we rely on the easier way and declare the maximal size of our PDU as maximal length of the UDP packet's payload (or the maximal size of the UDP packet minus its header length: 2^{16} -32 bytes). In order to guarantee delivery we will introduce the server response messages: each request sent by a client has to be confirmed by the server. Client considers the request delivered only if corresponding confirmation is received from the server. The described pattern is usually referred as request-response pattern, refer to Figure 2.2.

Now knowing the characteristics of our PDU let's declare the control options or the way we inform the endpoint what do we want. In other words, let's introduce the header of our PDU. Now we can say the header is of fixed length (first N bytes) and in these N bytes, we store all the control options we need. For example, in the first byte we specify which sort of action is requested (0x0 for publishing and 0x1 for retrieval). However, if we do not reserve enough bytes in the header, it will be hard to extend the protocol with more options. Another disadvantage is that not used space is a wasted space in PDU's header and it could be used to carry the payload. The solution here would be the floating header of various size with special delimiter byte denoting the end of the header and the start of the payload.

Remark: In fact what bytes to we use for denoting a delimiter (if use floating header) or denoting actions is up to the developer. In our code examples we intentionally use characters only (for our own convenience in debugging - strings are readable if you refer to raw UDP packet's payload when monitoring network).

After we declared the PDU, its header and payload options, let's discuss the mentioned use cases, and how these might be reflected in our protocol:

- User wants to publish the message (control code "1") The routines are quiet straight forward, refer to Figure 2.3:

1. Client side:

- the client reads the user's message and server's address (IP, port)
- client cuts off the tail in case the message is bigger then our maximal PDU size (excluding PDU header length).
- client wraps the message into request (adding the header containing the control code "1" publish and ":" - terminating header)
- client sends UDP packet with the request to the server on the address specified by the user

2. Server side:

- server receives new UDP packet and extracts the payload (the client's request)
- server checks the header and determines the operation that the client has requested
- server proceeds with "publish" routine and gets the client's message from the request
- server issues publish routine of the message board, giving message and the client's socket address as an argument
- message board gets the current time stamp and stores the values: times tamp, client's IP, client's port, message
- server prepares the response (adding header containing control code "0" - No Error and ":" terminating header)
- server sends the UDP packet with response to client on the address the request was received from

3. Client side:

- client receives the UDP packet and extracts the server's response
- client checks the header and determines if control code reports "No Error"
- client reports "Message Published" to user and terminates

- User wants to see N last published messages:

This scenario is a bit more tricky, as we requesting multiple messages and the size of a result payload might easily exceed the maximal allowed payload of our PDU. Therefore we introduce two additional sub-routines, and two additional control codes for PDU header:

- Get iD-s of last N messages published (control code "2")
 - * Client sends the request with desired number of last messages to read (N)
 - * Server responses with the list of iDs of the corresponding messages
- Get message by iD (control code "3")
 - * Client sends the request with the iD of the desired message
 - * Server responses with the requested message

The whole routine "User wants to see N last published messages" is then happening in multiple steps, where each step is request/response iteration:

1. Client retrieves the list of iDs (of the last N messages published)
2. For each of the received iD the client then fetches the message text in separate request

When the list of last N messages is collected, these are shown to user, leaving an impression of single routine. The whole routine is illustrated in Figure 2.4.

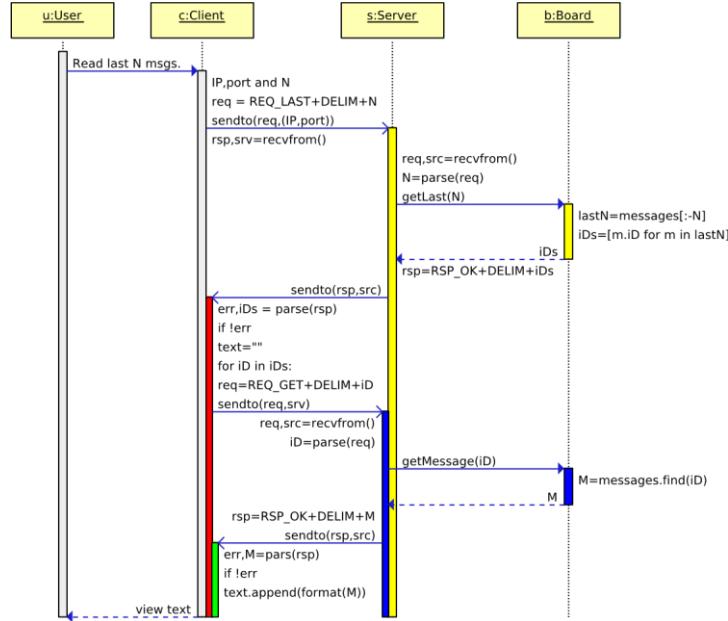


Figure 2.4: Sequence diagram of “Read Last N messages” routine

The resulting protocol is therefore as follows:

- request-response stateless protocol, refer to Figure 2.2
 - client sends a request to server and waits for a confirmation (a response)
 - request/response are not longer than one data unit (PDU)
- protocol data unit (PDU) size = maximal UDP payload size
- request/response PDU has floating header with ':' as delimiter, all the numerical values are stringencoded (the whole PDU data is then easily readable as string, for the sake of tracing the protocol).
- the request codes:
 - REQ_PUBLISH = '1'
Client wants to publish new message
Payload is the text of the user's message (till the end of UDP packet) Example of the total message:
* "1>Hello world!"
 - REQ_LAST = '2'
Client wants the iDs of the last N messages
Payload is one integer N (string coded) Example

of the total message:

- * "2:10"
- REQ_GET = '3'
 - Client wants the message by iD
 - Payload is one integer iD (string code) Example of the total message:
 - * "3:4325204352"
- response codes:
 - RSP_BADFORMAT = '1'
 - Server could not parse the header of the request No payload
 - RSP_MSGNOTFOUND = '2'
 - Server could not find the message by the iD specified No payload
 - RSP_UNKNCONTROL = '3'
 - Server parsed the header, but there is no action for this control code No payload
 - RSP_OK = '0'
 - No error
- Payload is different dependent on the request:
 - * REQ_PUBLISH
 - No payload
 - * REQ_LAST
 - Payload is the list of integers (string coded, delimited by ':') Example of the total message:
 - "0:235432:242435243:3452425423"
 - * REQ_GET
 - Payload is a requested message with all its attributes: time stamp when published (float: unix epoch time), sender-socket's IP address (string) and port (int), actual message text (string). All attributes are string encoded and appended using ':' as delimiter.
 - Example of the total message:
 - "0:12343241.3241:127.0.0.1:34234>Hello world!"

At this point the design of the protocol should be clear, let's proceed with the implementation.

Implementation of the protocol

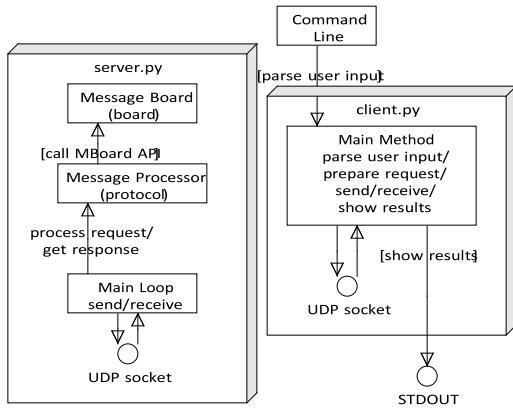


Figure 2.5: Message Board components diagram: Server-side (on the left), Client-side (on the right)

Implementing part 1 of our scenario is straightforward, we organized the code into 4 Python modules (refer to: 2.5):

- **board.py**
Implements the MBoard API (the data structures for holding messages, the methods for publishing and retrieving the messages).
- **protocol.py:**
Implements the MBoard protocol's server side: message processor (parsing request header, extracting payload, calling the MBoard API dependent on request, composing response). Also holds the protocol constants: control codes for requests, header delimiter, response codes.
- **server.py:**
Implements the MBoard server: arguments processor, socket binding, main serving loop (receive, process, response).
- **client.py:**
Implements the MBoard client: arguments processor, and methods for requesting the data according to routines: “publish message” or “view last N messages”

Client-side

Let's start first by the client implementation. The client Implements message board UDP client and it is a simple command-line application. Currently no interactive mode implemented. The application does the request and dies. In addition, the user has to provide the IP address of the server at least.

Default behavior is to fetch all the messages from the Board and show them to the user. Refer to --help to get the details about the options.

```
$python client.py -h
```

The current implementation considers all user input is by default in UTF-8, no additional encoding (providing non utf-8 characters in user's input may produce error)

Limitations (considered in protocol)

Protocol implements state-less message board protocol (MBoard protocol). In this protocol, we assume that the client never sends the big message (that would need multiple UDP packets). In case the client sent a long text, the only text contained in the packet will be the one respecting the size and the remaining will be lost. Therefore, if you want to preserve the long text, you have to take into account the UDP packet behavior and contained. This latter means that you have to split your message according to the authorized size that can be contained by UDP packets and send you message in multiple packets. However, the ID of the corresponding message are sent first, then the client asks messages one by one using IDs, refer to Figures 2.3 and 2.4.

Server-side

The server implements message board UDP server (server.py) and it is using static message board implementation (board.py) and implementation of message board protocol (protocol.py). The server is using simple state-less protocol, assuming client can only send one UDP packet per interaction (2 packets received from the same origin are processed in separate interactions).

The server does not assure if the packets arriving is in sequence from the same origin (each new arrived packet is considered as a new client). With a protocol like this it is impossible to sent messages in multiple UDP packets from the same client; as the server may receive packets from different clients interleaved and does not organize packets into client sessions.

Having client sessions requires to store additional information about a client on server side (client state). For example, in order to allow sending messages in multiple UDP packets, the partially received messages must be temporally stored by server (till server receives all the UDP packets belonging to this message). Keeping the information about client state on server side is the key idea of stateful protocols (see the next section for more details).

The server may be started from command line using:

```
$python server.py
```

The default behavior is to start listening on localhost address (127.0.0.1) and port 7777. Once application is started it runs forever, if not interrupted with Ctrl+C.. In order to see other options of the applications use refer to application's help:

```
$python server.py -h
```

Part 2: Stateful Protocol (Introducing sessions on UDP)

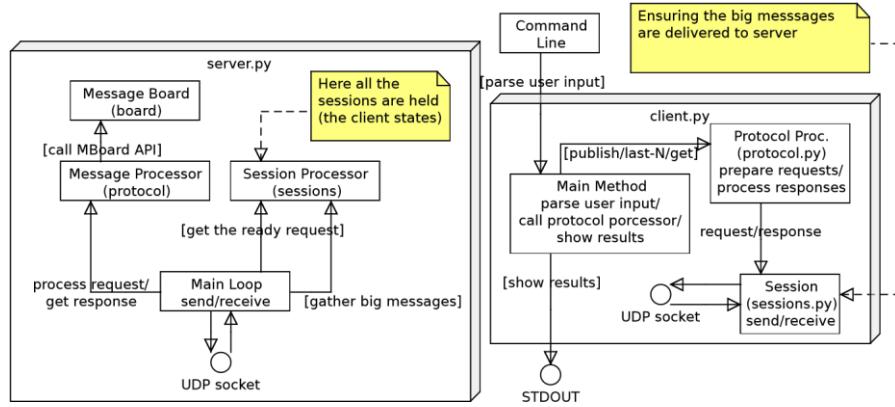


Figure 2.6: Message Board components diagram: Server-side (on the left), Client-side (on the right)

Concerning the part 2 of our example implementation, it will be divided into five main modules (client, protocol, session, server and board), refer to Figure 2.6.

Remark: At this level we are expecting you can read diagrams we provided and make the analogy with source code of the example application. Further we will not elaborate our code and diagrams like in previous section, rather we expect you to capable of doing it by yourself, in case of questions we are there in the Seminars and Office hours to help you.

Design

In general the design did not change that much compared to part one, except that we introduced the module Sessions which take care of communication and brings the notion of stateful aspect to our system. In details (refer to Figure 2.7)¹:

1. In part 1 the MBoard protocol data units (containing requests/responses) were delegated directly to UDP and therefore we could not have big messages sent at once.
2. In part 2 the MBoard protocol data units (containing requests/responses) are delegated to session protocol to ensure delivery. And the sessions takes care of splitting the big message and sent it over UDP ensuring delivery.

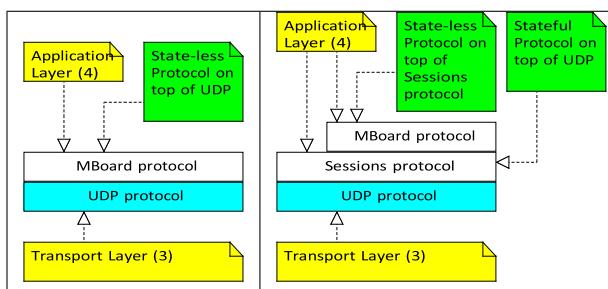


Figure 2.7: Protocol layers of Part 1 (on the left) and Part 2 (on the right)

¹ In this diagram we refer to TCP model layers

The most intuitive way of ensuring the big message delivery is to split it into blocks and send them one by one, confirming delivery of each one. Once all blocks are delivered the final assembly happens, refer to Figure 2.8).

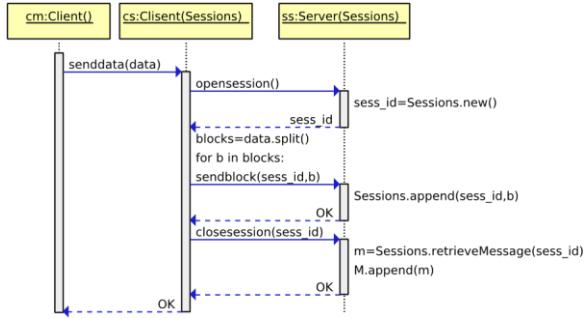


Figure 2.8: Simple illustration of the block delivery protocol

However on the server side the blocks of different sessions might appear interleaved as two clients may send two different big messages to the server in blocks. In this case server must understand what block belongs to to what client and store those into separate buffers (assembling big message). Therefore before starting sending we ask server to give us the session id for particular big message delivery, server keeps information of session id and the socket address of client requested the session. Client then uses session id in with each block it sends. Once full message is assembled the server threats the whole message as received from the client who has issued session.

The sequences diagram of the sessions protocol is illustrated in Figure 2.9 , the implementation is in the material folder provided [UDPApplication/sessions].

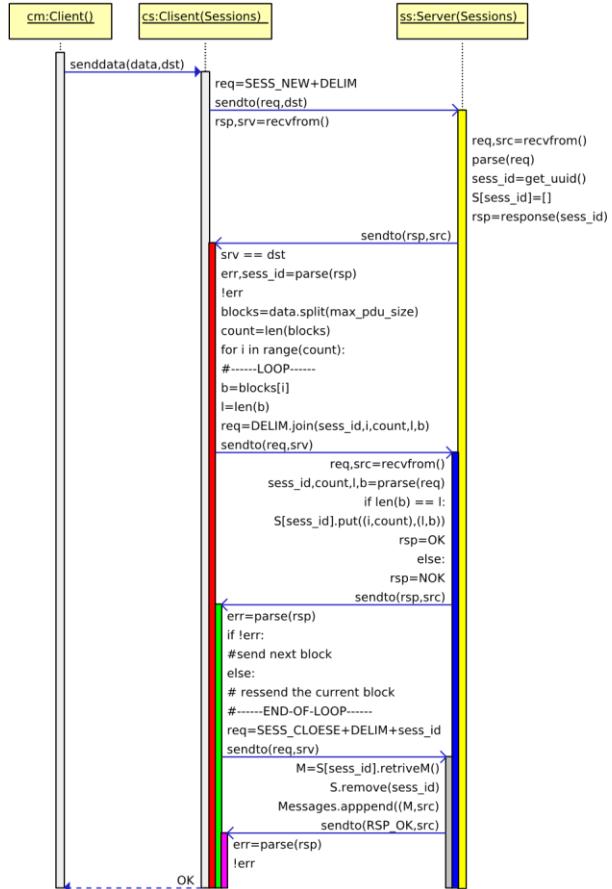


Figure 2.9: Sessions Protocol on UDP

TCP Protocol

We are expecting you that you assimilated most of the information introduced in the previous section on UDP. However, we will make some recall for important definitions as a reminder if necessary.

16. Transmission Control Protocol (TCP) is a standard reliable data delivery that defines how to establish and maintain a connection or network conversation via which data can be exchanged. Moreover, TCP is one of the transport layers suggested in the OSI layer and it is used to create connection between remote machine by transporting and ensuring delivery of messages over the network and the Internet.

Note: Summarizing features of TCP

- Connection oriented: An application requests a "connection" to destination and uses connection to transfer data - UDP does not use "connections" - each datagram is sent independently!
- Point-to-point: A TCP connection has two endpoints (no broadcast/multicast)
- Reliability: TCP guarantees that data will be delivered without loss, duplication or transmission errors
- Full duplex: Endpoints can exchange data in both directions simultaneously
- Reliable connection startup: TCP guarantees reliable, synchronized startup between endpoints (using "three-way handshake")
- Graceful connection shutdown: TCP guarantees delivery of all data after endpoint shutdown

Having in mind those features you may think of TCP as of a pipeline between 2 endpoints. The pipeline consists of 2 pipes with opposite flow. Both endpoints have 2 pipe valves Rx and Tx valve (refer to : Figure 2.10 on page 36). Having a pipeline like that we guarantee nothing gets lost during the delivery (otherwise the pipe is broken); additionally we guarantee the order of delivery (FIFO - first in first out). However we lose some features, as you can see the pipes associating two sockets only, if we had a third party "Socket C", then we would have needed an additional pipelines "A<->C" and "B<->C".

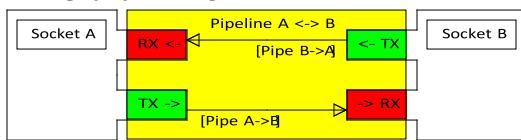


Figure 2.10: Illustrating TCP on pipeline example

Note: Broken pipe exception

Good feature of TCP pipes is the immediate report once the active pipe is broken. It happens in case there is a connectivity issues between the end-points (lost signal, broken cable, no gateway etc.). By analogy with cable powering the desktop lamp, there will be immediate feedback once the cable gets damaged - the lamp will stop shining.

In the next sections, we will step by step implement the simple messaging protocol between 2 endpoints relying on TCP.

TCP Socket

Before we start showing how to manipulate TCP socket, we have to introduce the states of TCP socket. TCP is a stateful Transport Layer (3) protocol. We have in fact introduced stateful protocols already and even implemented one on Application Layer (4)², having additional PDU headers, options and additional structures on both endpoints. In TCP it is all implemented in the OS (Berkly or POSIX socket interface are implemented in Linux and Mac, Microsoft has its own interface - Winsock which closely follows the POSIX standard). In general, in TCP a connection progresses through a series of states during its lifetime (refer to:

Figure 2.11 on page 38)³. The states are as follows:

- LISTEN: represents waiting for a connection request from any remote TCP socket address (IP and port).
- SYN-SENT: represents waiting for a matching connection request after having sent a connection request.
- SYN-RECEIVED: represents waiting for a confirming connection request acknowledgment after having both received and sent a connection request.
- ESTABLISHED: represents an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.
- FIN-WAIT-1: represents waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.
- FIN-WAIT-2: represents waiting for a connection termination request from the remote TCP.
- CLOSE-WAIT: represents waiting for a connection termination request from the local user.
- CLOSING: represents waiting for a connection termination request acknowledgement from the remote TCP.
- LAST-ACK: represents waiting for an acknowledgment of the connection termination request previously sent to the remote TCP. (it also includes an acknowledgement of its connection termination request)
- TIME-WAIT: represents waiting for enough time to pass to be sure the remote TCP received the acknowledgement of its connection termination request.
- CLOSED: represents no connection state at all.

Note: CLOSED is a fictional state because it represents the state when there is no TCP pipe, thus no connection.

In general, TCP connection progresses from one state to another in response to events. The events are:

- The user calls/Server receives incoming connection request (SYN sent/SYN received)
- Open, set socket into listening mode
- Send, sending the data, ensuring its delivery
- Receive, receiving the data, ensuring its delivery

² In this example we refer to TCP model layers

³ TCP/IP Illustrated, Volume 2: The Implementation by Gary R. Wright and W. Richard Stevens

- Close, putting from states ESTABLISHED or LISTENING into state CLOSED. In case of ESTABLISHED state we also wait for remote endpoint to confirm the closing, therefore the FIN_WAIT states.
And in case no confirmation received we close the socket after certain time out.
- Abort, when socket in state LISTENING denies to accept new connection. In case of exceeding the maximal number of simultaneous sockets in state ESTABLISHED.
- Status, when socket in state ESTABLISHED sends keep-alive to inform remote socket of its presence (in case there pipe is idling with no active data transmission).
- Incoming segments / flags (SYN, ACK, RST, FIN, and timeout)

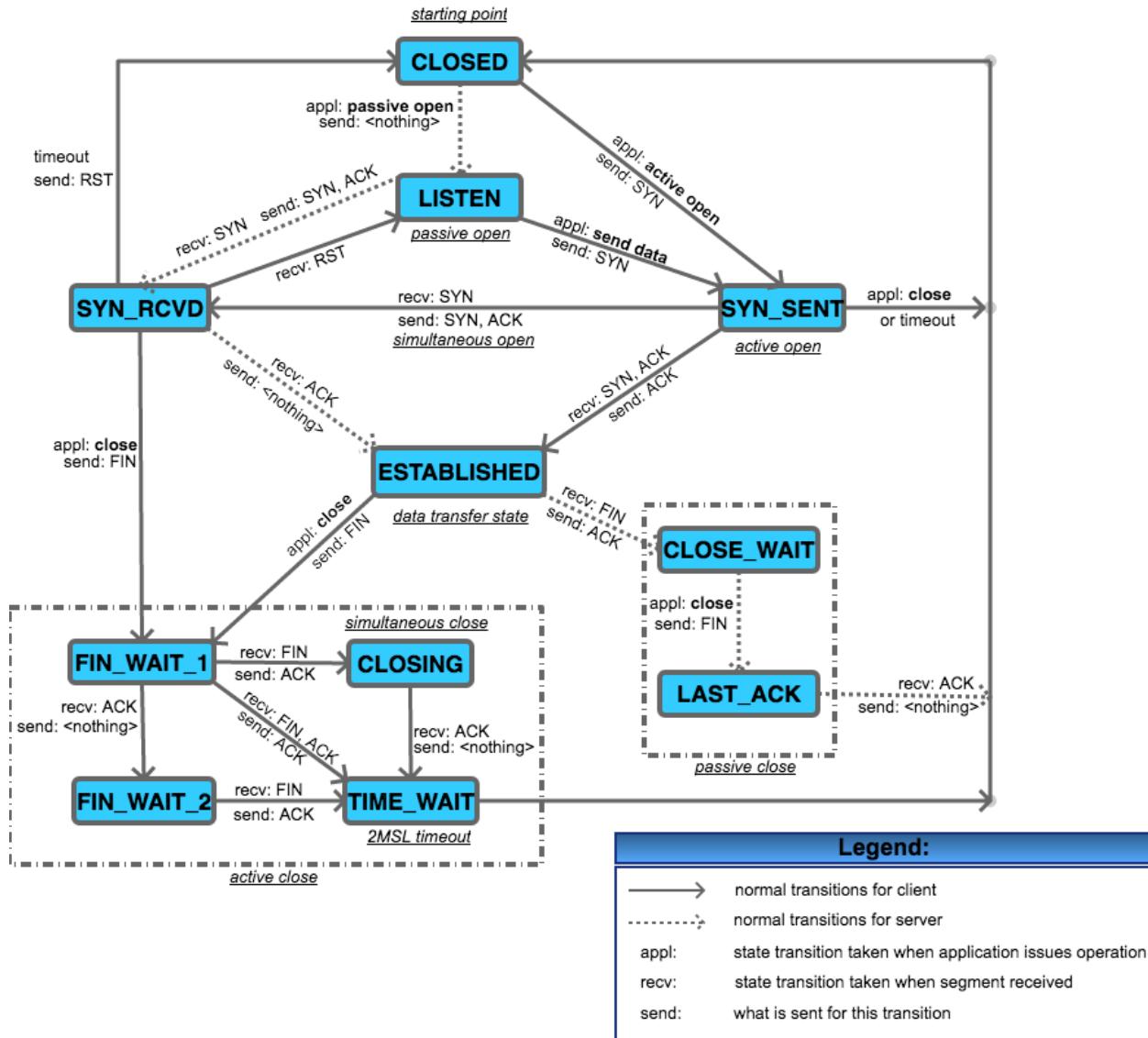


Figure 2.11: TCP State Transition Diagram

- SYN: Initiate connection
- ACK: Acknowledge received data

- FIN: Close a connection
- RST: Abort the connection in response to an error

The “creating” and “binding” operations of a TCP socket are similar to the ones of a UDP socket.

Creating socket

Here is an example code showing how to create the TCP socket and how to check what descriptor assigned to it by the OS:

```
# From socket module we import the required structures and constants. from socket import AF_INET,
SOCK_STREAM, socket

# Sleep function will be used to keep application running for a while from time import sleep
# And we also will need the process id given by OS when we execute this code from os import getpid
# Main method if __name__ ==
'__main__':
    print 'Application started'
    print 'OS assigned process id: %d' % getpid()
    # Creating a TCP/IP socket s = socket(AF_INET,
    SOCK_STREAM)
    print 'TCP Socket created'
    print 'File descriptor assigned by OS: ', s.fileno()
    wait_secs = 60*5 print 'Waiting %d seconds before termination ...' % wait_secs
    sleep(wait_secs)
    print 'Terminating ...'
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

Executing the code should print the following result, please note that letters X and Y will be denoting the process ID and the descriptor respectively and are specific to your environment (each time you run the code it will give different numbers):

Application started OS assigned process id: X

TCP Socket created

File descriptor assigned by OS: Y
Waiting 300 seconds before termination ...

Now while the code is running for 5 minutes we have time to refer to the list of registered descriptors of the OS and check if we see our running application process with a corresponding descriptor:

1. in Linux systems the Kernel hold raw process information in folder-like structures (populated by Kernel). So the very primitive folder listing call is enough (the ls command). Issue the following command in shell, make sure in place of X you use the process ID reported by the executed Python code and in place of Y -

```
~$ ls -l /proc/X/fd/Y
```

the descriptor):

1

... in the output you should now see the socket's descriptor that were assigned by the OS to a process number X:

```
lrwx----- 1 user user 64 Aug 18 13:13 Y -> socket:[129069931]
```

2. in MAC systems the unbound sockets are visible by lsof command, so it is enough to print all the TCP descriptors (lsof -i 4tcp) and filter the ones assigned to specific process ID (grep X). Issue the following

```
$ lsof -i 4tcp |grep X
```

command in shell, make sure in place of X you use the process ID reported by the executed Python code:

1

... in the output you should now see the socket's descriptor that were assigned by the OS to a process number X:

Python	2062	user	3u IPv4 0x90964583fac7dd81	0t0 TCP *:*
--------	------	------	----------------------------	-------------

If you can see the descriptor of a socket you have created in your code, you may continue to the next step of binding a socket. The currently running application (if not terminated after 5 minutes) may be terminated manually (Ctrl+C if running in shell explicitly). You may notice that after the application was terminated all its descriptors (including our TCP socket) are gone (closed automatically by the OS). Alternatively in your code you may issue close() method on socket before terminating the application (2 last lines of the reference code

```
sleep(wait_secs)
print 'Closing the TCP socket ...' s.close()
print 'Terminating ...'
```

are modified):

1
2
3
4

Please note that "creating" socket is yet not enough to start communicating, remember for UDP "creating" socket was like

“constructing a new mailbox”; now for TCP an analogy to creating socket would be “mounting the valves for an upcoming pipeline” or “mounting a socket/plug of a 220V power cable”.

Binding socket

As we know, after creating the socket, we have to provide a transmission endpoint; tell to the OS/kernel what IP address and TCP port we would like to use. As we saw in a previous chapter, binding a UDP socket to an address, made a UDP socket reachable from the network; However, binding TCP socket gives us the choice to choose either “connecting” or “listening from that socket” (by analogy “calling-out” or “waiting for a call” when using a phone).

The socket “listening” and “connecting” states will be explained later. The “binding” however is similar to a UDP socket.

```
from socket import AF_INET, SOCK_STREAM, socket
from os import
getpid

if __name__ == '__main__':
    print 'Application started'
    print 'OS assigned process id: %d' % getpid()

    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    print 'File descriptor assigned by OS:', s.fileno()
    # Binding the TCP/IP socket to loopback address and port 7777
    s.bind(('127.0.0.1',7777))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Wait for user input before terminating application
    raw_input('Press Enter to terminate ...')

    s.close()
    print 'Terminating ...'
```

In the following code we will create TCP socket and bind it to loop-back address and TCP port 7777:

```
1
2
3
4
5
6
7
8
9 10
11
12
13
14
15
16
17
18
```

After starting the application it will stay running till the user hits “Enter” button.

As you can see in the output, the corresponding TCP socket issued by Python process (PID X) is registered with transport address 127.0.0.1:7777 which means the running python code can now “listen” to new connections on that socket address (IP:port) or initiating new connection or “connecting” to a remote socket using local bound socket.

Now we know how to create and bind TCP sockets for the application to be able to communicate to the other instances in the IP network using TCP protocol. Before we proceed to “sending” / “receiving” the data over TCP, let’s make sure we understand “listen” and “connect” operations as these are the key features of TCP sockets.

Listening

TCP Socket in the “listening” state is like an idling phone, expecting an incoming call. Once the call is received we literally should “pick-up a the phone” and start conversation. Let’s first of all illustrate how TCP socket looks

```
from socket import AF_INET, SOCK_STREAM, socket from os import
getpid
if __name__ == '__main__':
    print 'Application started'
    print 'OS assigned process id: %d' % getpid()
    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    print 'File descriptor assigned by OS:', s.fileno()
    # Binding the TCP/IP socket to loopback address and port 7777 s.bind(('127.0.0.1',7777))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Put socket into listening state
    backlog = 0 # Waiting queue size, 0 means no queue s.listen(backlog)
    print 'Socket %s:%d is in listening state' % s.getsockname()
    # Wait for user input before terminating application raw_input('Press Enter to terminate ...')
    s.close()
    print 'Terminating ...'
```

like in “listening” state, please refer to the code:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
```

```
Application started
OS assigned process id: 24335
TCP Socket created
File descriptor assigned by OS: 3
Socket is bound to 127.0.0.1:7778
Socket 127.0.0.1:7778 is in listening state Press Enter to terminate ...
```

The output:

Check the socket state using netstat command:

```
~$ netstat -lnt | grep X
```

1. In Linux systems, from the shell

1

where X is the process ID printed by the executed Python code. As a result the following information will be shown (where X is PID reported by the executed Python code):

```
tcp 0 0 127.0.0.1:7777 0.0.0.0:* X/python
```

2. In Mac from the terminal

1

where X is the port number used in the Python code (e.g. 127.0.0.1.7777). The results will be as follows:

tcp4	0	0 127.0.0.1.7777	*.*
------	---	------------------	-----

```
~$ netstat -p tcp -a
```

3. In Windows in command line (cmd):

1

will give a list of all states of all TCP sockets including our one which should be listed as:

```
TCP 127.0.0.1:7777 HOSTNAME LISTENING
```

At this point we may proceed to an “accept” method that does-the pickup on the TCP socket in the “listening” state.

Accepting Connections (server-side)

Accepting the TCP connection by the TCP socket in listening state is like to pick-up a phone. Once the phone is picked-up the conversation may start. According to TCP state diagram the accept operation can only occur on TCP socket in listening state. TCP accept operation creates another socket on server just to communicate with the client that was requesting connection. Calling accept method will also block until the incoming TCP connection on corresponding socket has occurred. The code illustrating “listener-socket” accepting incoming

```
from socket import AF_INET, SOCK_STREAM, socket
if __name__ == '__main__':
    print 'Application started'
    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    # Binding the TCP/IP socket to loop-back address and port 7777 s.bind(('127.0.0.1',7777))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Put socket into listening state
    backlog = 0 # Waiting queue size, 0 means no queue s.listen(backlog)
    print 'Socket %s:%d is in listening state' % s.getsockname()
    client_socket,client_addr = s.accept()
    print 'New client connected from %s:%d' % client_addr print 'Local end-point
    socket bound on: %s:%d'\n
    '' % client_socket.getsockname()
    # Wait for user input before terminating application raw_input('Press Enter
    to terminate ...') s.close()
    print 'Terminating ...'
```

connections and creating socket in established state:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

... keeping the server-side code running we proceed to the client-side code.

Connecting (client-side)

For client side socket it is simply the connect operation we need to issue after creating the socket. The code is

```
from socket import AF_INET, SOCK_STREAM, socket
if __name__ == '__main__':
    print 'Application started'
    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    # No binding needed for client, OS will bind the socket automatically
    # when connect is issued server_address =
    ('127.0.0.1',7777) # Connecting ...
    s.connect(server_address)
    print 'Socket connected to %s:%d' % s.getpeername() print 'Local end-point is bound
    to %s:%d' % s.getsockname() # Wait for user input before terminating application
    raw_input('Press Enter to terminate ...') s.close()
    print 'Terminating ...'
```

as follows:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

Questions:

1. What will happen to the running applications in case we hard-interrupt the connectivity (unplugging the Ethernet cable or hardware switching the wireless adapter off)?

TCP packet/header

17. *TCP packet is transport unit of the TCP protocol. It contains between 20 bytes till 60 bytes of TCP header and the payload.*

The TCP header contains the following:

-
- Source address (2 bytes) [identifies the sending port]
 - Destination address (2 bytes) [identifies the receiving port]
 - Sequence number (4 bytes) has a dual role:
 - if the SYN flag is set to (1), then this is an initial sequence number.
 - if the SYN flag is set to (0), then this is the accumulated sequence number of the first data byte of this segment for the current session.
 - Acknowledgement number (if ACK set)(4 bytes)
 - Data offset (1-st bit of 1 byte) [Specify the size of TCP header in 32-bit words.]
 - Reserved (2-nd of 1 byte) [for future use and should be set to zero]
 - Flags (3 bits: 3-d, 4-th and 5-th of 1 byte) [combination of those gives us the following flags:
 - NS (1 bit) – ECN-nonce concealment protection (experimental: see RFC 3540).
 - CWR (1 bit) – Congestion Window Reduced (CWR) flag is set by the sending host to indicate that it received a TCP segment with the ECE flag set and had responded in congestion control mechanism (added to header by RFC 3168).
 - ECE (1 bit) – ECN-Echo has a dual role, depending on the value of the SYN flag. It indicates:
 - * If the SYN flag is set (1), that the TCP peer is ECN capable
 - * If the SYN flag is clear (0), that a packet with Congestion Experienced flag set (ECN=11) in IP header received during normal transmission (added to header by RFC 3168). This serves as an indication of network congestion (or impending congestion) to the TCP sender.
 - URG (1 bit) – indicates that the Urgent pointer field is significant.
 - ACK (1 bit) – indicates that the Acknowledgment field is significant. All packets after the initial SYN packet sent by the client should have this flag set.
 - PSH (1 bit) – Push function. Asks to push the buffered data to the receiving application.
 - RST (1 bit) – Reset the connection
 - SYN (1 bit) – Synchronize sequence numbers. Only the first packet sent from each end should have this flag set. Some other flags and fields change meaning based on this flag, and some are only valid for when it is set, and others when it is clear.
 - FIN (1 bit) – No more data from sender]
 - Windows size (4 bytes) [the size of the receiving window]
 - Urgent pointer (if URG set) (2 bytes) [Indicate the last urgent data bytes if the URG is set.]
 - Checksum (2 bytes) [used for error-checking of the header and data].
 - Options (4 bytes). The length of this field is determined by the data offset field. Options have up to three fields:
 - Option-Kind (1 byte),
 - Option-Length (1 byte), - Option-Data (variable).

Receiving using TCP connection

Here we use again the same example with client and server, and we demonstrate how server starts receiving a client's message when there is a new client connected (does not have to be this way, and the server might start sending the data to client first, and then proceeds to receiving). For receiving we issue `recv(...)` with buffer length to collect the received data (this is similar to UDP `recvfrom` method), according to socket API:

Note:

- `socket.recv(bufsize[, flags])`
 - Received data from the socket. The return value is a string representing the data received. The maximum amount of data to be received at once is specified by `bufsize`. See the Unix manual page `recv(2)` for the meaning of the optional argument `flags`; it defaults to zero.

Keep in mind that send/receive we cannot do on the server's listener-socket, till it accepts new client by using `accept()` method, which will lead to the creation of a new socket for this particular connected client (we call it client-socket). The client socket can then be used to send/receive to this particular client. And here we demonstrate sequence of 1.) creating socket 2.) binding it 3.) making it a listener-socket 4.) accepting new client

```
from socket import AF_INET, SOCK_STREAM, socket
if __name__ == '__main__':
    print 'Application started'
    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    # Binding the TCP/IP socket to loop-back address and port 7777
    s.bind(('127.0.0.1',7777))
    print 'Socket is bound to %s:%d' % s.getsockname()
    # Put socket into listening state
    backlog = 0 # Waiting queue size, 0 means no queue
    s.listen(backlog)
    print 'Socket %s:%d is in listening state' % s.getsockname()
    client_socket,client_addr = s.accept()
    print 'New client connected from %s:%d' % client_addr
    print 'Local end-point'
    print socket bound on: %s:%d
    print '' % client_socket.getsockname()
    # Once the client is connected start receiving the data using its socket:
    recv_buffer_length = 1024
    message = client_socket.recv(recv_buffer_length)
    print 'Received %d bytes from %s:%d' % (len(message),)+client_addr
    print 'Received message: \n%s' % message
    # Wait for user input before terminating application
    raw_input('Press Enter to terminate ...')
    client_socket.close()
    print 'Closed the client socket'
    s.close()
    print 'Closed the server socket'
    print 'Terminating ...'
```

and creating client-socket 5.) receiving a message using client-socket:

```
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
```

```
Application started
TCP Socket created
Socket is bound to 127.0.0.1:7777
Socket 127.0.0.1:7777 is in listening state ...
```

The output is then looking like:

5

Keeping the server's code running you may proceed with the client's code (next section). The client will then

```
...
New client connected from 127.0.0.1:59484 Local end-point
socket bound on:
127.0.0.1:7777
Received 11 bytes from 127.0.0.1:59484 Received message:
Hello World!
Press Enter to terminate ...
```

send the data and then the server continues to execute with the output:

5

Our message is successfully received. Now let's make sure we close both sockets (client-socket and listenersocket). For the TCP socket in listening or established state it is critical to enforce close() method before terminating an application. If we not do so, the socket will stay there waiting for socket timeout till it gets cycled by the OS, due to statefulness of the TCP protocol (refer to Figure 2.11).

Sending a message using TCP connection

Sending the data using TCP is simplified a lot compared to UDP. Since there is a defined pipe between source and destination, and the pipe guarantees the delivery regardless of how much data we want to send (just we have to be sure, the remote endpoint is ready to receive that much). Therefore no additional addressing on

```
from socket import AF_INET, SOCK_STREAM, socket
if __name__ == '__main__':
    print 'Application started'
    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    # No binding needed for client, OS will bind the socket automatically
    # when connect is issued server_address =
    ('127.0.0.1',7777) # Connecting ...
    s.connect(server_address)
    print 'Socket connected to %s:%d' % s.getpeername() print 'Local end-point is bound
    to %s:%d' % s.getsockname() message = 'Hello World!'
    if s.sendall(message) == None:
        print 'Send %d bytes to %s:%d' % ( (len(message),)+s.getpeername())
    # Wait for user input before terminating application raw_input('Press Enter
    to terminate ...') s.close()
    print 'Terminating ...'
```

send method is needed (socket is already connected to the endpoint). The code for sending the message:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

The output is then looking like:

```
Application started TCP Socket created
Socket connected to 127.0.0.1:7777
Local end-point is bound to 127.0.0.1:58244
Send 11 bytes to 127.0.0.1:7777 Press Enter to
terminate ...
Terminating ...
```

5

When we issue `sendall(...)` method on the socket object of the connected state the socket API then takes care of sending the data using TCP protocol. The data gets split and sent in blocks if necessary, and for each block sent the remote endpoint sends acknowledgement, ensuring the delivery. This way the TCP guarantees the delivery of the whole data. The `sendall(...)` method always returns `None` in case of successfully delivered data. In case of data-loss, it rises an Exception. There is also another method `send(...)` which gives more details regarding delivery, it just reports the number of bytes delivered. The application has to check if it was the right amount of bytes that was intended to be delivered (in this case we avoid throwing exceptions). According to socket API:

Note:

- `socket.send(string[, flags])`
 - Send data to the socket. The socket must be connected to a remote socket. The optional flags argument has the same meaning as for `recv()` above. Returns the number of bytes sent. Applications are responsible for checking that all data has been sent; if only some of the data was transmitted, the application needs to attempt delivery of the remaining data. For further information on this concept, consult the Socket Programming HOWTO.
- `socket.sendall(string[, flags])`
 - Send data to the socket. The socket must be connected to a remote socket. The optional flags argument has the same meaning as for `recv()` above. Unlike `send()`, this method continues to send data from string until either all data has been sent or an error occurs. `None` is returned on success. On error, an exception is raised, and there is no way to determine how much data, if any, was successfully sent.

Dealing with Big Amount of Data

Sending big message using TCP

```
from socket import AF_INET, SOCK_STREAM, socket
if __name__ == '__main__':
    print 'Application
        started' # Creating a UDP/IP socket s =
    socket(AF_INET, SOCK_STREAM)
    # Sending the message (socket is still unbound)
    # We will use the break line to indicate the end of message
    # then the receiver may receive without knowing the full length
    # of the message he is receiving term = '\n'
    message = 'Hello world'*7000+term destination
    = ('127.0.0.1',7777) # Connect to the server
    s.connect(destination)
    print 'Connected to the server %s:%d' % s.getpeername() print 'Local end-point
        bound on %s:%d' % s.getsockname()
    # Send the data
    s.sendall(message)
    print 'Sent message to %s:%d' % destination
    print 'Pay-load length %d bytes: [%s]' % (len(message),message)
    raw_input('Press Enter to terminate ...')
    print 'Closing the TCP socket ...' s.close()
    print 'Terminating ...'
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

Receiving big message using TCP

```
from socket import AF_INET, SOCK_STREAM, socket
if __name__ == '__main__':
    print 'Application
        started' # Creating a TCP/IP socket s =
    socket(AF_INET, SOCK_STREAM)
    # Binding the UDP/IP socket to address and port s.bind(('127.0.0.1',7777))
    print 'Socket is bound to %s:%d' % s.getsockname() # Turn the socket
    into listener-socket s.listen(0)
    print 'Socket is listening on %s:%d' % s.getsockname()
    # Wait for a client to connect client_socket,client_addr = s.accept()
    print 'Client connected from %s:%d' % client_addr
    # Receiving the message,
    # here we need to the socket API what is the size of the block
    # that we are ready to receive at once
    recv_buffer_length = 1024 term = '\n'
    # Terminator indicating the message is over message = ''
    # This will grow in progress of receiving print 'Waiting for message
    ...
    # Append message block by block till terminator is found while not
    message.endswith('\n'):
        m = client_socket.recv(recv_buffer_length)
        print 'Received block of %d from %s:%d'\
            '' % ( (len(m)),) + client_socket.getpeername())
        message +=m
    print 'Total length %d bytes: [%s]' % (len(message),message)
    raw_input('Press Enter to terminate ...') client_socket.close()
    print 'Closed client socket...' s.close()
    print 'Closed the listener socket ...' print 'Terminating ...'
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36

Reflection: as you can see the receiver knows exactly from what origin is the data received, as the TCP socket in connected state has only two endpoints. Therefore the `recv(...)` method does not return any additional information from where the data was received.

Ensuring Packet Arrival Order

TCP guarantees the data is delivered in the same order it was sent, therefore we do not need to program any additional routines or protocols to ensure the arrival order.

Note: The receiver has to know how much data there will be sent by sender also in case of TCP. In our example we chose to use the terminator to mark the end of message, however we could avoid using special terminator character and just add the total message length into first block of the message.

TCP socket shutdown routines

Shutdown is used in case we need to close the communication in one or both direction. As we mentioned before TCP socket has two transmitting pipes, the shutdown routine can close one of them or both. Usually shutdown is used to denote the end of transmission in one direction and start transmission in the other one. For example, when client did finish sending the data, the corresponding TX pipe is closed on client side and the client proceeds to receiving. Once client has closed the TX pipe, the server will receive empty buffer calling the `recv(...)`, and understand that there is the end of received data. According to socket API:

- `socket.shutdown(how)`
 - Shut down one or both halves of the connection. If `how` is `SHUT RD`, further receives are disallowed. If `how` is `SHUT WR`, further sends are disallowed. If `how` is `SHUT RDWR`, further sends and receives are disallowed. Depending on the platform, shutting down one half of the connection can also close the opposite half (e.g. on Mac OS X, `shutdown(SHUT WR)` does not allow further reads on the other end of the connection).

The socket shutdown also does not close the I/O descriptors of the socket (the `close(...)` method does it).

Programming Application layer protocol on top of TCP

We will eventually implement the same example of MBoard protocol as in the previous chapter, but this time on top of TCP.

We may keep the same scenario with user willing to publish the message or to read all messages. Thanks to TCP it is not that important how big is the message the user wants to publish or how many messages are there already on server-side.

The following we leave one-to-one like it was declared in the MBoard of UDP version:

Note:

Let's say we want the public message board service to be shared with any number of users who want to publish or read messages. So we have two roles for network entities - the "messageboardservice" and the "user".

Regarding scenarios we have changes though, definitely we do not need separate scenario for handling the big messages (like in UDP). Regarding the first scenario, we just say big messages are by default allowed (the only limits regarding the message length are now OS specific - maximal length of command line arguments, and on the server side - the amount of RAM to process the messages).

The resulting scenario is, refer to Figure :

- The message board service is responsible for serving user requests:
 - User may request to submit his message to public board
 - User may request the list of messages currently published on the board

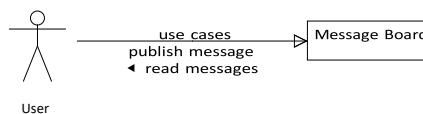


Figure 2.12: Use case of MBoard

Server-side

Regarding the protocol design, we will still use the request-response protocol (MBoard protocol), we just slightly adjust it taking into account the TCP features. The server side (server.py) we change it into TCP connection handler, once the server bound the socket, it is put into listening state and falls into loop. Within the loop we have the following processes:

1. Accept the client's connection (blocks until new client is connected)
2. Receive the data using client's socket till client issues shutdown on writing end (TX) of his socket.
3. Process the client's sent data (the request), prepare the response
4. Send the response data using client's socket
5. Close the client socket

You may have noticed we do not start accepting new client till we finished serving the previous one. Eventually it may happen that there will be new client asking for connection while the server is busy (serving the previous client as we still are using only one main loop with no additional threads). In order no to lose client we allow the listener socket to “remember” the clients who asked for connection, but whose connections is not yet accepted by the server. This is done using backlogging of the listener, according to socket API:

- `socket.listen(backlog)`
 - Listen for connections made to the socket. The backlog argument specifies the maximum number of queued connections and should be at least 0; the maximum value is system-dependent (usually 5), the minimum value is forced to 0.

Client-side

On client-side the sequence are mostly the same:

1. Process user input (get server's IP and port; options to publish message or read the last N messages)
 - (a) In our implementation all the required values are asked from the user using command line arguments
2. Connect to server (create TCP socket, issue connect method)
3. Once connected to server:
 - (a) Prepare the request data
 - (b) Send data using connected socket
 - (c) Shutdown writing pipe (TX) of the socket, when sending is over
 - (d) Receive the response
 - i. Show the results to the user
4. Disconnect the socket (issue close method)
5. Terminate the client's application

Changes in MBoard protocol

The routine “get last N messages” does not have to be split anymore into two subroutines. Therefore, we may use just one request/response iteration to get all the messages. New control code REQ GET N LAST (4) which instructs the server to give last N messages in one response.

Handling broken pipe exceptions

TCP is reliable enough and the data usually gets delivered, otherwise TCP reports socket error. This usually happens if send or receive is issued but at the same time the host gets disconnected from the network. These error reports may be taken into account when programming the application. In our case, for example on the server side when the send or receive is interrupted by socket error, we stop handling the client, close its socket, then socket is proceeded to

the next client. On the other hand, if we have the same error on the client side, we just report communication error to the user and terminate.

Chapter 3

Threads

Introduction

During this lecture we will introduce threads, processes and synchronization among them. We start by introducing threads and its operations. Then, we will try to understand how to create and manipulate threads in python using examples and illustrations. In the second half we will introduce the network programming patterns using threads and processes. We conclude as usual showing the example of application illustrating how the obtained knowledge can be applied.

What is multitasking?

Before defining Multitasking let's define first the meaning of task in computer programming.

18. *Task is a generic term referring to an activity carried out by software that occurs independently from other activities in the system.*

Therefore, the multitasking can be defined as follows:

19. *Multitasking is the capability to perform more than one task at a time.*

A concept where multiple tasks are executed interleaved over certain period of time. The concept is an opposed to sequential execution - where the new task can not be started till the old is finished. Please note, that the term multitasking does not automatically mean executing tasks in parallel (concurrently). It means there is just more than one task progressing in time. The illusion of simultaneous execution is usually guaranteed letting the executor rapidly switch between the tasks (refer to Figure 3.1). In this case executor does a small portion of the first task, leaves it and proceeds with the second task etc. In fact, the described approach is widely applied in single executor environments:

- Instruction pipelining in obsolete single-core processors
- Cooperative multitasking in obsolete OS versions (Windows 3.x, 9x, etc.)

However, the amount of overall time required for completing for example three tasks is still the same for both sequential and multitasking concepts. Therefore, there is no actual gain in performance in multitasking, however it leaves an "impression" like the tasks are executed simultaneously. The only way to gain in

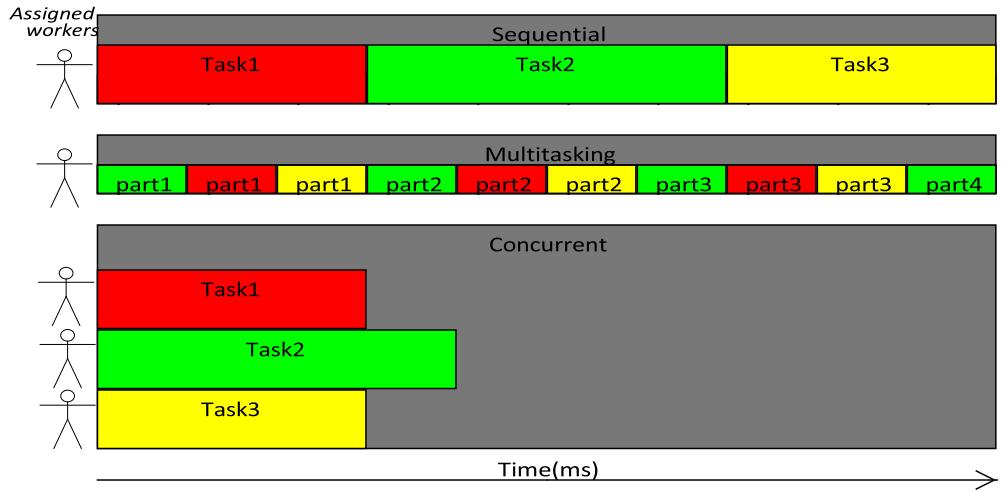


Figure 3.1: Task Execution Concepts

performance (have the tasks done faster) is to distribute the tasks among the multiple workers and let them execute. In this case there is “real” simultaneous execution of the tasks, and overall execution time is therefore reduced. The described concept is called “parallel” or “concurrent” and at this point we may introduce Threads and Process (correspond to workers in concurrent concept).

Processes and Threads

The terms Process and Thread are both related to the independent execution sequence (the worker assigned for a specific task) and are often confused. Processes are referred as threads and vice versa. Therefore we have to introduce and elaborate both terms before we actually start programming. In simple words: each time we start an application, the OS starts a separate process and allocates required resources (virtual address space, executable code, I/O descriptor, sockets, etc. what may be required by the application). Additionally inside the process one or many Threads are launched in order to execute the application’s code. In case we have simple application with “main” method only, we will have only one “main” thread running inside our application process.

Starting a “main” thread is done automatically (without any calls to threading API) and therefore you might mistakenly think that there is no threads and the “main” method is run by the process directly. In case of multi-threaded applications the process will have several threads running and accessing the resources allocated for a process. The application has to be programmed taking into account the fact of possible concurrent use of resource, otherwise there is high risk for facing hardly traceable and irregular errors during application’s runtime.

Even more “human friendly” explanation of processes and threads is given in George Coulouris book (see “Flies in a Jar” example):

“... An execution environment consists of a stoppered jar and the air and food within it. Initially, there is one fly – a thread – in the jar. This fly can produce other flies and kill them, as can its progeny. Any fly can consume any resource (air or food) in the jar. Flies can be programmed to queue up in an orderly manner to consume resources. If they lack this discipline, they might bump into one another within the jar – that is, collide and produce unpredictable results when attempting to consume the same resources in an unconstrained manner. Flies can communicate with (send messages to) flies in other jars, but none may escape from the jar, and no fly from outside may enter it. In this view, originally a UNIX process was a single jar with a single sterile fly within it.”

Processes

A process is a passive entity such as a file on disk storage and it is also an active one. Besides, more processes can refer to the same program. This means, two instances of the same program have the same code section but with different current activities. From this perspective we can define Process as follows:

20. *Process is just a program in execution. It is modeled as a finite state machine which transits from state to state by executing a sequence of atomic actions.*

When it comes to the modern operating systems, it seems for the user like the OS is capable of running simultaneously the programs. This is not true, but they are using something called **timesharing** to manage multiple programs, which give this illusion of simultaneous execution.

21. *A timesharing operating system is that in which each task is given some time to execute and all tasks are given time so that all processes run seamlessly without any problem.*

As we defined before each running program counts as a **process** in UNIX terminology and also in Windows. Therefore, multiple copies of a program running counts as multiple processes. The processes will run by taking turns. This means, one process might run for a few milliseconds causing the OS to run till it is interrupted by the timer hardware. At this level, the OS saves the current state of the interrupted process in order to resume it later, then selects the next candidate process to give it a chance to run. This latter phenomena is called **context switch** (because the running CPU has switched from one process to another). The cycle is repeated and any given process will get the Chance to run till eventually all processes are finished. The period of time when the process is allowed to run is called a **time slice** or **quantum**.

Moreover, the OS maintains a **process table**, where all current processes are listed. Each process will have state either **Run** or **Sleep**.

The Run state means that the current process is ready to run and the OS will choose on the processes in Run each time a turn ends. The Sleep state means that the process is waiting for an event to occur and we say that the process is **blocked**.

Threads and

A thread is the entity within a process that can be scheduled for execution. All threads of a process share its virtual address space and system resources. Therefore threads occupy much less memory, plus it take less time to create than do processes.

The main goal of threads is to make it possible to write programs which run multiple tasks, and execute them in an efficient manner. Nowadays, the role of threads has become very important in applications programming such as in web servers development or GUI programs.

22. *Thread can be defined as a single sequential flow of control within a program. Sometimes called "lightweight" process because it runs within the context of a full-blown program and uses the resources allocated for that program and its environment.*

To resume our definition of threads, threads are the smallest units that can be scheduled in an operating system. Usually, they are contained in processes and there can be many of them within the same process.

Moreover, as we stated before they share also all resources provided for the process in question.

There are two different kind of threads:

- Kernel threads
-

-
- User threads (or user-space threads)

For your information kernel threads are part of the operating system, meanwhile user threads are an extension of the function concept of a programming language (does not exit in the kernel).

Advantages of threading

We can resume the advantage of threading into four main ones:

- First, it allows programs to run faster on computer systems with the use of multiple CPUs, thanks to its concurrent execution aspect.
- Second, it help to maintain the program to be responsive in both cases: single or multiple CPU.
- Third, all the threads of a process can share the memory of the global variables allocated for that process and any change affect to them is noticeable by all the threads. (But thread can have also local variables)
- With threading it is possible to hide the time that it takes to send/receive messages (or synchronization) in parallel applications behind doing some useful activities simultaneously. Doing the communication in a separate thread does not block the CPU from performing some calculations, as an example.

Thread Managers

Thread manager is like a mini-operating system similar to a real operating system that maintains a table of processes, where thread system's thread manager maintains a table of threads. The moment a thread abandon the CPU or has its turn **Pre-empted**, then the thread manager looks in the table for another thread to activate.

23. Pre-emption refers to the temporary interruption and suspension of a task, without asking for its cooperation, with the intention to resume that task later. This act is called a context switch and is typically performed by the pre-emptive scheduler, a component in the operating system authorized to pre-empt, or interrupt, and later resume tasks running in the system.

A thread being **pre-empted** is the moment when an active thread within a given time slice is being interrupted by hardware the thread from a time cause control of the CPU and transfer it to the thread manager.

Another analogy with processes, a process is either in sleep mode or run mode, the same apply to threads. A thread is either ready to be given a turn to run, or is waiting for some event. As already mentioned, thread systems are either **kernel-level** or **user-level**.

Kernel-level thread managers

In kernel-level each thread is really like a process and the thread manager is like an OS. When different threads are set up by a given application they will take turns running like processes do. The moment a thread is activated, it has a specific time slice for running. When the time slice is over it will get pre-empted. Another case is when the thread reaches a point at which it has to wait for some other event to occurs before continuing, then it will voluntarily relinquish its time slice (this type of threads are used in Unix system and windows as well).

Remark: In kernel-level thread manager the fact that threads act exactly as a process, thus they will appear as a single running process when executing **ps** process command line in Unix.

User-level thread managers

User-level thread systems, on the other hand, are “private” to the application. Here the threads are not pre-empted; on the contrary, a given thread will continue to run until it voluntarily gives up control of the CPU, either by calling some “yield” function or by calling a function by which it requests a wait for some event to occur.

Remark: In User-level thread manager the fact that threads are private to the application, thus they will not appear as a single running process when executing **ps** process-command line in Unix but you will see the application to which they are associated with.

Note: Comparison

Kernel-level threads have the advantage that they can be used on multiprocessor systems, thus achieving true parallelism between threads.

On the other hand, User-level threads can allow one to produce code which is much easier to write, to debug, cleaner and clearer.

Python thread manager

Even though Python’s threads mechanisms are built on top of the underlying platform’s threads system, this is basically transparent to the application programmer. Which means that python threads are a blend of the kernel- and user-level approaches, but still more towards the user-level.

The interpreter keeps track on how long the current thread has been executing, in terms of the number of Python byte code instructions have executed. When it reaches a certain number, by default 10, another thread will be given a turn. Such a switch will also occur if a thread reaches an I/O statement.

Thus Python threads are pre-emptive. Internally, Python maintains a Global Interpreter Lock to ensure that only one thread has access to the interpreter at a time.

Threads modules in python

Python threads are accessible via two modules, **thread.py**, **threading.py**, and **multiprocessing.py**.

Note for your information: The **thread** module has been considered as deprecated for quite a long time. Users have been encouraged to use the **threading** module instead.

Thread module

The simplest and easy way to use a Thread is to instantiate it with a target function and call start new thread() to let it begin working.

```
import thread

def User():
    """thread User function"""
    print 'Hello I am
    User'
    return

for i in range (4):
    thread.start_new_thread(User,())

raw_input("Click Enter to Terminate ...")
```

1
2
3
4
5
6
7
8
9
10
11

Click Enter to Terminate ...

Hello I am User
Hello I am User
Hello I am User
Hello I am User

The output should look like this:

5

Although it is very effective for low-level threading, but the thread module is very limited compared to the newer threading module.

Threading module

The latest version of threading module in python is very powerful. It has high-level support for threads in comparison with the thread module presented in the previous section.

The Threading module contains all the methods of the thread module and more. Here is a collection of examples of additional methods that are useful for our course:

- **threading.activeCount()**: Returns the number of thread objects that are active.
- **threading.currentThread()**: Returns the number of thread objects in the caller's thread control.
- **threading.enumerate()**: Returns a list of all thread objects that are currently active.

Moreover, the thread class implemented threading are as follows:

- **run()**: The run() method is the entry point for a thread.
- **start()**: The start() method starts a thread by calling the run method.
- **join([time])**: The join() waits for threads to terminate.
- **isAlive()**: The isAlive() method checks whether a thread is still executing.
- **getName()**: The getName() method returns the name of a thread.
- **setName()**: The setName() method sets the name of a thread.

Now let's start exploring how to use some of the features or functions of threading module. Here we will show most important ones but not everything; therefore, be curious and try to explore more features about the module by your own.

Declaring a thread:

Concerning using the module threading is very simple you can trigger the threads by call the function start(), the

```
import threading

def User():
    """thread User function"""
    print('Hello I am'
          User'
    return
threads = []
for i in range(4):
    t=threading.Thread(target=User) threads.append(t)
    t.start()

raw_input("Click Enter to Terminate ...")
```

following example will demonstrate how to use it:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
```

The output should look like this:

```
Hello I am User  
Hello I am User  
Hello I am User  
Hello I am User  
Click Enter to Terminate ...
```

5

Using Argument with thread:

After demonstrating how to start threads in threading module, let's try now to spawn a thread by passing argument

```
import threading  
  
def User(counter):  
    """thread User function""" print 'Hello I am User number: %s' %  
    counter  
    return  
threads = []  
for i in range (4):  
    t=threading.Thread(target=User, args=(i,)) threads.append(t)  
    t.start()  
  
raw_input("Click Enter to Terminate ...")
```

that it will include in its printing action:

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13
```

```
Hello I am User number: 0  
Hello I am User number: 1  
Hello I am User number: 2  
Hello I am User number: 3  
Click Enter to Terminate ...
```

The output should look like this:

5

Identifying the current thread:

At this stage, we would like not only to use threads but also identifying them. Thus, we will use arguments to identify or name the thread by choosing a label. Normally, each thread instance has a name with a default value associate to it that we can change as the thread is created. The action of naming the threads is very important and useful in server processes especially when you are handling multiple service threads in different

```
import threading import time

def User():
    print threading.currentThread().getName(), 'Arrived'
    time.sleep(2) print threading.currentThread().getName(), 'Leaving'

def session(): print threading.currentThread().getName(), 'Starting'
    time.sleep(3)
    print threading.currentThread().getName(), 'Exiting'

t = threading.Thread(name='session', target=session) u =
threading.Thread(name='User', target=User) u2 =
threading.Thread(target=User)# use default name u.start() u2.start()
t.start()
```

operations.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

The output should look like this: (in this output the name of the current thread is on each line except the line with "Thread-1" is the unnamed thread u2)

```
User Thread-1 Arrived  
Arrived session Starting  
Thread-1 Leaving User  
Leaving session Exiting
```

5

Logging debug:

Most of clean code or program we don't use print statement as a mean to debug. Thanks to the logging module, it supports embedding the thread name in every log message. The following code demonstrates how to make

```
import threading  
import time import  
logging  
  
logging.basicConfig(level=logging.DEBUG,  
                    format='[%(levelname)s]%(threadName)-10s%(message)s',  
)  
use of it:  
1  
2  
3  
4  
5  
6
```

```
def User():  
    logging.debug('Arrived') time.sleep(2)  
    logging.debug(Leaving)  
  
def session(): logging.debug('Starting') time.sleep(3)  
    logging.debug('Exiting')  
  
t = threading.Thread(name='session', target=session) u =  
threading.Thread(name='User', target=User) u2 =  
threading.Thread(target=User)# use default name u.start() u2.start()  
t.start()
```

7
8
9
10
11
12

13

```
14
15
16
17
18
19
20
21
22
23
24
```

```
[DEBUG] (User      ) Arrived
[DEBUG] (Thread-1 ) Arrived
[DEBUG] (session   ) Starting
[DEBUG] (User      ) Leaving
[DEBUG] (Thread-1 ) Leaving
[DEBUG] (session   ) Exiting
```

The output should look like this:

5

Daemon threads:

Till now all the examples we showed the programs waited till all the threads exit and complete their work. However, when a program spawns a thread as daemon that runs without blocking the main program from exiting. The use of daemon thread is important when you want your thread to complete its task no matter what

```
import threading
import time
import logging

logging.basicConfig(level=logging.DEBUG,
                    format='[%(levelname)s] (%(threadName)-10s) %(message)s',
                    )

def User1(): logging.debug('Arrived') time.sleep(2)
    logging.debug('Leaving')

d= threading.Thread(name='User1', target=User1)
```

happened to the hosting program. let's demonstrate how it works using the **setDaemon()** method:

1

2

```
3
4
5
6
7
8
9
10
11
12
13
d.setDaemon(True) #we set User1 to daemon thread

def User2():
    logging.debug('Arrived') logging.debug('Leaving')

t = threading.Thread(name='User2', target=User2)

d.start()
t.start()

14
15
16
17
18
19
20
21
22
23
```

The output should look like this:

```
[DEBUG] (User1      ) Arrived
[DEBUG] (User2      ) Arrived
[DEBUG] (User2      ) Leaving
```

you can notice from the output that User1 thread did not leave because it is a daemon thread. Since all of the non-daemon threads (including the main thread) exit before the daemon thread wakes up from its two second sleep.

In order to wait until a daemon thread has completed its work, we can use the **join()** method.

```
import threading
import time import
logging

logging.basicConfig(level=logging.DEBUG,
                    format='[%(levelname)s]%(threadName)-10s%(message)s',
)

def User1(): logging.debug('Arrived') time.sleep(2)
    logging.debug(Leaving)

d= threading.Thread(name='User1', target=User1)
d.setDaemon(True) #we set User1 to daemon thread

def User2():
    logging.debug('Arrived') logging.debug('Leaving')

t = threading.Thread(name='User2', target=User2)

d.start()
t.start()

d.join()
t.join()
```

1
2
3
4
5
6

7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26

The output should look like this:

```
[DEBUG] (User1      ) Arrived  
[DEBUG] (User2      ) Arrived  
[DEBUG] (User2      ) Leaving  
[DEBUG] (User1      ) Leaving
```

Locks in threads:

It is very important to prevent and control corruption of data when handling any communication. Thus, it is necessary to watch out the accesses to an object. In order to do so, we have to use **lock** object in our code.

24. Lock is a synchronization primitive which is not owned by a particular thread when locked.

A primitive lock is one of two states: “locked” or “unlocked” and it is created in the unlocked state. Moreover, it has two basic methods: **acquire()** and **release()**.

Let's now discuss some specific cases:

- In case the state is unlocked, then **acquire()** will changes the state to locked and returns immediately.
- in case the state is locked, then **acquire()** will block until a call to **release()** in another thread changes it to unlocked, then **acquire()** call resets it to locked and return.

Remark: The **release()** method should only be called in the locked state; it changes the state to unlocked and returns immediately. If an attempt is made to release an unlocked lock, a **RuntimeError** will be raised.

Here is an illustration of lock object:

```
import threading
import time import
logging

logging.basicConfig(\ level=logging.DEBUG,\ 
    format='%(threadName)-9s %(message)s',)

class Counter(object):
    def __init__(self, start = 0):
        self.lock = threading.Lock() self.value = start

    def increment(self): logging.debug('Waiting for a lock')
        self.lock.acquire()
        try:
            logging.debug('Acquired a lock') self.value = self.value +
            1 logging.debug('Value %d' % self.value)
        finally:
            logging.debug('Released a lock')
            self.lock.release()

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

```

def decrement(self): logging.debug('Waiting for a lock')
    self.lock.acquire()
    try:
        logging.debug('Acquired a lock') self.value = self.value - 1
        logging.debug('Value %d' % self.value)
    finally:
        logging.debug('Released a lock') self.lock.release()

def worker_a(counter):
    for i in range(10): logging.debug('Sleeping %0.02f', 1)
        time.sleep(1) counter.increment()
    logging.debug('Done')

def worker_b(counter):
    for i in range(10): logging.debug('Sleeping %0.02f', 2)
        time.sleep(2) counter.decrement()
    logging.debug('Done')

if __name__ == '__main__':
    counter = Counter()
    t_inc = threading.Thread(target=worker_a, args=(counter,)) t_dec =
    threading.Thread(target=worker_b, args=(counter,)) t_inc.start() t_dec.start()
    logging.debug('Waiting for worker threads') t_inc.join()
    t_dec.join()
    logging.debug('Counter final value: %d', counter.value)

```

24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43

```
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
```

```
(Thread-1 ) Sleeping 1.00
(Thread-2 ) Sleeping 2.00
(MainThread) Waiting for worker threads
(Thread-1 ) Waiting for a lock
(Thread-1 ) Acquired a lock
(Thread-1 ) Value 1
(Thread-1 ) Released a lock
(Thread-1 ) Sleeping 1.00
(Thread-1 ) Waiting for a lock
(Thread-1 ) Acquired a lock
(Thread-1 ) Value 2
(Thread-1 ) Released a lock
(Thread-1 ) Sleeping 1.00
(Thread-2 ) Waiting for a lock
(Thread-2 ) Acquired a lock
```

The output should look like this:

5

10

15

```
(Thread-2 ) Value 1
(Thread-2 ) Released a lock (Thread-2 )
```

```
Sleeping 2.00 ....
```

Conditional Variables in threads:

Typical situation in thread programming is when we need to acquire a lock for a queue and pop one object from the queue. The queue is empty from objects but at the same time there is an initiated process of populating the queue with objects by another thread. In this situation we need to wait until the queue has at least one object. Then, we can of course put the thread into the sleeping mode for a while and retry checking the queue later (if there are new objects in the queue). However, this action does not guarantee us the ideal timing as the sleeping timeout does not necessarily happen at the same time when the queue gets populated with at least one object. In fact, what we want is that our sleeping thread gets woken up exactly at the time the queue gets first object inserted into it.

This is typical scenario for Conditional Variables and our first thread is waiting on condition "until queue is empty". The waiting thread then expects, it will get notification from someone (by itself the waiting thread will never wake up until interrupted). The second thread (the one populating the queue) knows that there might be other threads waiting for queue to get at least one element. Therefore once the first element is inserted, the populating thread notifies the waiting thread. The consumer thread in this case gets woken up exactly at the time the queue has new element inserted, and this is how exact timing is guaranteed between populating and consuming threads. Please note in this example we do not have any threads "idling free", which means iterating in loop: if the queue is empty - sleep, else - consume an element.

25. Conditions are a wrapper around an underlying Lock that provide wait/notify functionality. Condition is used when threads are interested in waiting for something to become true (by issuing **wait() method on conditional variable), and once its true, the thread has exclusive access to some shared resource. On the other hand, the **notify()** and **notifyAll()** are there to inform waiting threads about the fact that something it was waiting for became true.**

```
import logging import
threading import time

logging.basicConfig(\ level=logging.DEBUG,\ 
    format='%(asctime)s (%(threadName)-2s) %(message)s',)

class Queue:
    def __init__(self):
        self.__q = []
        self.__cv = threading.Condition()

    def put(self,e):
        with self.__cv:
            logging.debug('Added element')
            self.__q.append(e) if len(self.__q) == 1:
```

Here is the example of Conditional Variables in threads:

1
2
3
4
5
6
7

8

```
9
10
11
12
13
14
15
16
17
18
# First element added - notify all waiting threads logging.debug('Queue is no
more empty, notifying all'\
                     'waiting threads')
self.__cv.notifyAll()

def get(self): e = None with
    self.__cv:
        if len(self.__q) <= 0: logging.debug('Queue is empty, waiting ...')
            self.__cv.wait()
        e = self.__q.pop()
return e

def consumer(q):
    """wait for the condition and use the resource"""
    logging.debug('Starting consumer thread')
    for i in range(5):
        e = q.get()
        logging.debug('Retrieved: %d' % e) time.sleep(1)

def producer(q):
    """set up the resource to be used by the consumer"""
    logging.debug('Starting producer thread')
    for i in range(5):
        q.put(i)
        logging.debug('Added: %d' % i) time.sleep(3)

q = Queue()
cons = threading.Thread(name='Consumer', target=consumer, args=(q,)) prod =
threading.Thread(name='Producer', target=producer, args=(q,)) cons.start() prod.start() cons.join()
prod.join()
logging.debug('Terminating ...')

19
20
21
22
23
24
25
26
```

```
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
```

```
2016-10-03 09:27:35,002 (Consumer) Starting consumer thread
2016-10-03 09:27:35,005 (Consumer) Queue is empty, waiting ...
2016-10-03 09:27:35,010 (Producer) Starting producer thread
2016-10-03 09:27:35,010 (Producer) Added element
2016-10-03 09:27:35,010 (Producer) Queue is no more empty, notifying allwaiting threads
2016-10-03 09:27:35,011 (Producer) Added: 0
2016-10-03 09:27:35,011 (Consumer) Retrieved: 0
2016-10-03 09:27:36,012 (Consumer) Queue is empty, waiting ...
2016-10-03 09:27:38,014 (Producer) Added element
2016-10-03 09:27:38,014 (Producer) Queue is no more empty, notifying allwaiting threads
2016-10-03 09:27:38,014 (Producer) Added: 1
2016-10-03 09:27:38,015 (Consumer) Retrieved: 1
2016-10-03 09:27:39,016 (Consumer) Queue is empty, waiting ...
```

The output should look like this:

```
2016-10-03 09:27:41,016 (Producer) Added element
2016-10-03 09:27:41,016 (Producer) Queue is no more empty, notifying allwaiting threads
2016-10-03 09:27:41,017 (Producer) Added: 2
2016-10-03 09:27:41,017 (Consumer) Retrieved: 2
2016-10-03 09:27:42,018 (Consumer) Queue is empty, waiting ...
2016-10-03 09:27:44,017 (Producer) Added element
2016-10-03 09:27:44,018 (Producer) Queue is no more empty, notifying allwaiting threads
2016-10-03 09:27:44,018 (Producer) Added: 3
2016-10-03 09:27:44,018 (Consumer) Retrieved: 3
2016-10-03 09:27:45,019 (Consumer) Queue is empty, waiting ...
2016-10-03 09:27:47,019 (Producer) Added element
2016-10-03 09:27:47,020 (Producer) Queue is no more empty, notifying allwaiting threads
2016-10-03 09:27:47,020 (Producer) Added: 4
2016-10-03 09:27:47,020 (Consumer) Retrieved: 4
2016-10-03 09:27:50,021 (MainThread) Terminating ...
```

10

15

20

25

Events in threads:

Another possibility to synchronize the Threads in python is to use Event objects. For cultivating yourself about Events in threading check the link below: http://www.bogotobogo.com/python/Multithread/python_multithreading_Event_Objects_between_Threads.php

Multiprocessing module

Python API for managing Processes is reduced in “multiprocessing” module and is very similar to the one of the “threading” module. This way it is much more easier for programmer to master the Multiprocessing API once he/she knows the Threading API. Surely these two APIs are not one-to-one similar, all the differences are quiet understandable once we understand the concepts of threads and processes.

The differences in Threading and Multiprocessing API are explained well in the tutorial below:

<https://pymotw.com/2/multiprocessing/basics.html>

Chapter 4

Shared state

Introduction

In the previous chapter, we introduced threads and thread programming models [7]. Apparently, Python threads are not good for parallel computing, as the Python's global interpreter lock (GIL) allows only one thread to execute the code at the time. However the waiting threads are allowed to perform system calls. Handling I/O descriptors are one of those calls (read/write on file or send/receive on socket) [4]. In other words Python threads may outperform on parallel computing but they are still good for parallel I/O handling.

As for true parallel execution of the code, different module - *multiprocessing* is advised [5].

Therefore, this time we will employ the thread programming models to handle network related I/O routines in our applications. We will start by improving request-response protocol servers focusing on servers-side mostly. After defining the server-side multi-threading patterns for request-response protocols, we will switch to more advanced long-lived TCP connections and cover the application protocols that allow server to send asynchronous notifications to clients. At this level, we will apply multi-threading for both client and server. Then, we will cover these essential aspects of threads and network – we will introduce the shared state topic and illustrate on simple network application (real-time concurrent modification of a shared data structure).

Threads in network applications

We should remember that the essential idea behind threads or multiple processes is the ability to execute in separate flow. However, we figured out that when it comes to computing, the Python **threading** module is not very suitable compared to **multiprocessing**. In addition, if we need more efficient computations, we have to deploy all the cores of our CPU equally and this is exactly the task of multiprocessing module in Python. In case we just need to do many I/O routines simultaneously (like reading many files in parallel), the threading module of Python is quite suitable and can be recommended for handling these routines.

Simple example

Sending and receiving the data using the network sockets are also I/O tasks. Therefore, it is not a bad idea to handle each particular socket in a separate thread. For example, having an application to download four files, we will eventually win in performance if we do download all four simultaneously as opposed to downloading all four sequentially. In the following code, we use Python's **urllib** module to handle HTTP

```
from tempfile import mktemp
import logging import urllib from
time import time
logging.basicConfig(level=logging.DEBUG,\n
                    format='%(asctime)s (%(threadName)-2s) %(message)s',)

def download(url):
    logging.info('Downloading %s' % url) tmpfilepath= mktemp()
    urllib.urlretrieve(url,tmpfilepath) logging.info('Finished downloading %s' % url)

if __name__ == '__main__':
    urls = [
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/testing/linux-4.9-rc1.tar.xz',
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.8.2.tar.xz',
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.7.8.tar.xz',
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.4.25.tar.xz' ]

    logging.debug('Application start ...') start = time() for u
    in urls:
        download(u)
    t = time() - start
    logging.debug('Terminating, total time spent %d sec ...' % t)
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

Figure 4.1: HTTP Download manager, single threaded

protocol (to download the files using HTTP), the first example is the sequential version (refer to Figure 4.1); and the next one (refer to Figure 4.2) illustrates the same application using threads. Here, we have used a *Delegation or Manager-Workers* thread programming model where the main method assigns a download task (the *download* method with a URL argument) to a worker thread.

```
from tempfile import mktemp import
logging import urllib from time import
time from threading import Thread

logging.basicConfig(level=logging.DEBUG,
                    format='%(asctime)s (%(threadName)-2s) %(message)s', )

def download(url):
    logging.info('Downloading %s' % url) tmpfilepath= mktemp()
    urllib.urlretrieve(url,tmpfilepath) logging.info('Finished downloading %s' % url)

if __name__ == '__main__':
    urls = [
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/testing/linux-4.9-rc1.tar.xz',
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.8.2.tar.xz',
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.7.8.tar.xz',
        'https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.4.25.tar.xz' ]

    logging.debug('Application start ...') threads = [] start =
    time() for u in urls:
        t = Thread(target=download,args=(u,)) threads.append(t)
        t.start()
    for t in threads:
        t.join()
    t = time() - start
    logging.debug('Terminating, total time spent %d sec ...' % t)
```

1
2
3
4
5
6
7
8
9
10
11

12

```
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
```

```
from socket import AF_INET, SOCK_STREAM, socket, SHUT_WR
if __name__ == '__main__':
    print 'Application started'
    s = socket(AF_INET, SOCK_STREAM)
    print 'TCP Socket created'
    s.bind(('127.0.0.1',7777))
    s.listen(1)
    client_socket,client_addr = s.accept()
    print 'New client connected from %s:%d' % client_addr
    print 'Local end-point socket bound on: %s:%d' \
          '' % client_socket.getsockname()
    # Simulate service offer by sending one byte to client
    client_socket.send('1')
    client_socket.shutdown(SHUT_WR)
    print 'Service offered, client is using a service ...'
    # Wait for the client to close connection
    # This one will block till client sends something or disconnects
    client_socket.recv(1)
    print 'Client finished using service' # Here we assume
    # Client is gone
    client_socket.close()
    print 'Client disconnected'
    # Wait for user input before terminating server application
    raw_input('Press Enter to terminate ...') s.close()
    print 'Terminating ...'
```

Figure 4.2: HTTP Download manager multi- threaded

```
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
```

Figure 4.3: Single threaded server, one client at most (singlethreaded server one client.py)

Client-server architectures

Now let's take a look how the thread programming models are applied for server side applications. We will start with more simple example of request-response protocol. As we saw in previous chapter, the pattern has one server and some clients. The clients connect and make requests to the server. The server accepts the connections and replies to the received messages. Let's first of all take a look at reference implementation:

Request-Response protocols

Let's take the server side created during the TCP lecture, this example has a limitation when two clients try to get connected at the same time – the server will interact only with one of them. The second one will be rejected (with TCP connection reset). Even if we increase the backlog size – the second client still has to wait in a queue till the server has finished processing the request of the first client. In the next code (refer to Figure 4.3) there is a server capable of serving only one client. And the next one (refer to Figure 4.4) the same server is capable of serving many clients, however the client requests are still processed sequentially.

Testing the servers with client's simulator

For better understanding of what happens to our clients let's perform a simple simulation. Suppose multiple clients are connecting to the server simultaneously. Once a client gets connected it expects the server to start dealing with it immediately, otherwise the client gets bored and leaves. The code in Figure 4.5 illustrates the corresponding simulator. Let's see how the server will perform with 5 clients simultaneously. We will

```

from socket import AF_INET, SOCK_STREAM, socket, SHUT_WR from socket import
error as soc_err

def handle_client(client_socket):
    try:
        print 'New client connected from %s:%d' % client_addr print 'Local end-point
socket bound on: %s:%d'\n
        '' % client_socket.getsockname()
        # Simulate service offer by sending one byte to client client_socket.send('1')
        client_socket.shutdown(SHUT_WR)
        print 'Service offered, client %s:%d is using a service ...'\n % client_addr
        # Wait for the client to close connection
        # This one will block till client sends something or disconnects client_socket.recv(1)
        print 'Client %s:%d finished using service' % client_addr
        # Here we assume client is gone except soc_err as e:
        if e.errno == 107: print 'Client %s:%d left before server could handle it'\n
            '' % client_addr else:
        print 'Error: %s' % str(e)
    finally:
        client_socket.close()
    print 'Client %s:%d disconnected' % client_addr

if __name__ == '__main__':
    print 'Application started'
    s = socket(AF_INET, SOCK_STREAM) s.bind(('127.0.0.1',7777))
    s.listen(1)
    print 'Socket %s:%d is in listening state' % s.getsockname() print 'Falling to serving loop,
press Ctrl+C to terminate ... try:
    while 1:
        client_socket = None
        print 'Awaiting new clients ...'
        client_socket,client_addr = s.accept() handle_client(client_socket)
    except KeyboardInterrupt:
        print 'Ctrl+C issued closing server ...'
    finally:
        if client_socket != None:
            client_socket.close()
        s.close()
    print 'Terminating ...'

```

1
2
3
4
5
6
7

```
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
```

Figure 4.4: Single threaded server, many clients (singlethreaded server many clients.py)

run the simulator's code as it is (Figure 4.5), having each client keeping the server busy for 5 seconds, and in case client cannot get immediate service, it waits for 15 seconds and leaves. The simulation results are shown in Figure 4.6 corresponding server's log is shown in Figure 4.7. As you can see, the server could handle 4 clients, and did not manage to handle last one before it disconnected. The reason is of course the sequential handling of the clients. The first client got service immediately and kept server busy for 5 seconds. The remaining 4 clients were still waiting loosing the seconds till timeout. Server managed to handle another 3 clients losing 5 seconds for each, the last client however was gone already (15 seconds timeout).

Question: What will be the results if the client can wait only for 5 seconds to get service, and each client keeps server busy for 30 seconds ?

Global Variables:

-
- DEFAULT TIMEOUT SEC = 5
 - DEFAULT STAY_CONNECTED SEC = 30

Adding thread models to the server side

In order to make our server code more flexible in handling many clients simultaneously, we have to introduce threads. Next, the previous server code will be rewritten to handle multiple clients connections using threads. To do so, we have two options:

- Dealing with each connection in a separate thread (thread-per-connection)
- Dealing with each request in a separate thread (thread-per-request)
 - * in case there are many requests sent using one connection

Thread-per-connection using different thread models

- Delegation (Manager-Workers):
 - The connection handling is explicitly delegated to a new thread. Each time we call the *handle_client* method, we call it in a separate execution flow, releasing the *main* execution flow to continue accepting new clients. In other words: once the new client's connection is accepted by main server loop, the client's socket is delegated to a separate Thread for handling. The thread is then started with *handle_client* method and *client socket* parameter. With the model like that we create a new thread each time new client connects (guaranteeing each client gets handled immediately), therefore it is also referred as **thread-per-socket**.
 - * As you may have noticed the amount of the threads created is defined by amount of incoming clients, therefore it can vary from zero to a huge number (in case of that many incoming clients).
- Producer-Consumer pattern
 - Indirect delegation using queue of client connections. The threads try to get Each time we accept new client's connection, we put it into client connections queue. Handling the client's connection is also done in a separate thread. The threads however are not created *on-demand*, instead a predefined set of threads is created once the application is started. With this model the amount of threads is not changed, the amount of clients that are handled simultaneously is defined by a number of predefined threads and the size of the queue. The model is therefore also referred as **pre-fork**.

```

import logging
logging.basicConfig ( level=logging.DEBUG,
format='%(asctime)s (%(threadName)-2s) %(levelname)s %(message)s ',) from socket import socket
, AF_INET, SOCK_STREAM, SHUTWR from socket import error as soc_err from socket import timeout as soc_err_timeout
from time import sleep from threading import Thread DEFAULT_CLIENTS = 5
DEFAULT SERVER HOST = '127.0.0.1'
DEFAULT SERVER PORT = 7777
DEFAULT STAY CONNECTED SEC = 5
DEFAULT TIMEOUT SEC = 15
DEFAULT RECV BUFSIZE = 1

def connect _toserver ( socket _addr , timeout , stay _connected ):
    s = socket (AF_INET,SOCK_STREAM) s . settimeout
    ( timeout ) try :
        logging . debug( 'Trying to connect to %s:%d ' % socket _addr ) s . connect ( socket _addr )
        logging . info ( 'Connected , waiting for service ...' )
        # Just let's say if received at least something let's count it
        # server dealing with us m = s . recv (DEFAULT
        RECV_BUFSIZE)
        if len (m) > 0:
            logging . info ( 'Service available ! \n'
            'Keeping it busy for %d sec ... ' % stay _connected ) # Simulateserviceusage
            by waiting for a while before disconnecting sleep ( stay _connected ) logging . info ( ' Finished using a service
            ... ' ) s . shutdown(SHUTWR) except soc_err timeout : logging . error ( 'Timeout waiting for service ! ' )
            except soc_err as e:
                if e . errno == 107:
                    logging . error ( ' Server closed connection , error %s ... ' % str (e )) else :
                        logging . error ( 'Can\'t connect to %s:%d , error %s '\
                        '' % ( socket _addr+str (e ), ))
            except KeyboardInterrupt :
                logging . info ( ' Ctrl+C issued , terminating ' )
            finally :
                s . close ()
                logging . info ( ' Disconnected ... ' )
if __name__ == '__main__':
    logging . info ( ' Application start ' )
    logging . info ( '%s clients will connect to %s:%s ... ' %
    (DEFAULT_CLIENTS, DEFAULT_SERVER_HOST, DEFAULT_SERVER_PORT)) logging . debug( ' Connection timeout %s sec ' % DEFAULT_TIMEOUT_SEC) logging . debug( ' Stay connected for %s sec ' % DEFAULT
    STAY_CONNECTED_SEC) threads = [] # Prepareclients for i in range (DEFAULT_CLIENTS):
        t = Thread( target=connect _toserver , args=(DEFAULT_SERVER_HOST, int (DEFAULT_SERVER_PORT)),
        int (DEFAULT_TIMEOUT_SEC),\ int (DEFAULT_STAY_CONNECTED_SEC)))
        threads . append(t)
        logging . debug( ' Client threads created ... ' )
map(lambda x: x.start () , threads ) # Start all
        logging . debug( ' Client threads started ... ' )
map(lambda x: x.join () , threads ) logging . debug( ' Client threads dead ... ' ) # Wait
for all logging . info ( ' Terminating ... ' )

```

1
2
3
4
5
6
7
8
9
10
11
12
13

```
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
```

Figure 4.5: Clients simulator (using threads) (simulating many connections.py)

```
56:54,257 (MainThread) INFO Application start
56:54,261 (MainThread) INFO 5 clients will connect to 127.0.0.1:7777 ...
56:54,261 (MainThread) DEBUG Connection timeout 15 sec
56:54,262 (MainThread) DEBUG Stay connected for 5 sec
56:54,264 (MainThread) DEBUG Client threads created ... 56:54,268 (Thread-1) DEBUG Trying
to connect to 127.0.0.1:7777 56:54,268 (Thread-1) INFO Connected, waiting for service ...
56:54,268 (Thread-1) INFO Service available! Keeping it busy for 5 sec ...
56:54,269 (Thread-2) DEBUG Trying to connect to 127.0.0.1:7777
56:54,269 (Thread-3) DEBUG Trying to connect to 127.0.0.1:7777 56:54,270 (Thread-3) INFO
Connected, waiting for service ... 56:54,270 (Thread-4) DEBUG Trying to connect to 127.0.0.1:7777
56:54,270 (Thread-4) INFO Connected, waiting for service ...
56:54,270 (MainThread) DEBUG Client threads started ... 56:54,271 (Thread-5) DEBUG Trying
to connect to 127.0.0.1:7777 56:54,276 (Thread-2) INFO Connected, waiting for service ...
56:54,278 (Thread-5) INFO Connected, waiting for service ...
56:59,270 (Thread-1) INFO Finished using a service ...
56:59,271 (Thread-1) INFO Disconnected ...
56:59,271 (Thread-2) INFO Service available! Keeping it busy for 5 sec ...
57:04,272 (Thread-2) INFO Finished using a service ...
57:04,273 (Thread-2) INFO Disconnected ...
57:04,273 (Thread-3) INFO Service available! Keeping it busy for 5 sec ...
57:09,279 (Thread-3) INFO Finished using a service ...
57:09,279 (Thread-3) INFO Disconnected ...
57:09,280 (Thread-4) INFO Service available! Keeping it busy for 5 sec ...
57:09,291 (Thread-5) ERROR Timeout waiting for service!
57:09,292 (Thread-5) INFO Disconnected ...
57:14,283 (Thread-4) INFO Finished using a service ...
57:14,284 (Thread-4) INFO Disconnected ...
57:14,284 (MainThread) DEBUG Client threads dead ...
57:14,284 (MainThread) INFO Terminating ...
```

5

10

15

20

25

```

Application started Socket
127.0.0.1:7777 is in listening state
Falling to serving loop, press Ctrl+C to terminate ...
Awaiting new clients ...
New client connected from 127.0.0.1:40366
Local end-point socket bound on: 127.0.0.1:7777
Service offered, client 127.0.0.1:40366 is using a service ...
Client 127.0.0.1:40366 finished using service
Client 127.0.0.1:40366 disconnected Awaiting new clients ...
New client connected from 127.0.0.1:40367
Local end-point socket bound on: 127.0.0.1:7777
Service offered, client 127.0.0.1:40367 is using a service ...
Client 127.0.0.1:40367 finished using service
Client 127.0.0.1:40367 disconnected Awaiting new clients ...
New client connected from 127.0.0.1:40368
Local end-point socket bound on: 127.0.0.1:7777
Service offered, client 127.0.0.1:40368 is using a service ...
Client 127.0.0.1:40368 finished using service
Client 127.0.0.1:40368 disconnected Awaiting new clients ...
New client connected from 127.0.0.1:40369
Local end-point socket bound on: 127.0.0.1:7777
Service offered, client 127.0.0.1:40369 is using a service ...
Client 127.0.0.1:40369 finished using service
Client 127.0.0.1:40369 disconnected Awaiting new clients ...
New client connected from
127.0.0.1:40370 Local end-point socket bound on:
127.0.0.1:7777 Client 127.0.0.1:40370 left before server could handle it
Client 127.0.0.1:40370 disconnected Awaiting new clients ...

^CCtrl+C issued closing server ...
Terminating ...

```

Figure 4.6: Client Simulator output

Figure 4.7: Single threaded servers output

1. accept client's connection (creating client's socket)
2. receive the client's request (reading it from the client's socket)
3. prepare the response (processing clients request)
4. send the response to client (writing it to the client's socket)
5. close client's connection

Table 4.1: Client's Connection Handling Sequence

* As *pre-fork* model is having fixed amount of threads associated through a queue, it is easily portable from threading to multiprocessing - having predefined number of processes started early to handle the upcoming client requests.

Next we will take our single threaded server and make two improved versions out of it:

- using delegation pattern (illustrated in File [*multithreaded server delegation pattern.py*])
- using producer-consumer pattern (illustrated in File [*multithreaded server prefork .pattern.py*])

Let's now measure performance of those versions having simulator settings as follows:

Global Variables:

- DEFAULT TIMEOUT SEC = 5
- DEFAULT STAY_CONNECTED SEC = 30

The simulators output looks same for both server implementations: all the client got service in time (refer to Figure 4.10 and 4.11). The output of the *thread-per-socket* model is illustrated in Figure 4.8 and the *pre-fork* model in Figure 4.9. We can see from server logs that in the first model the server created additional threads each time client was connected. In second model, eight threads were created at early point of application startup. Each thread was assigned to check for new sockets in the queue (which is empty in the beginning). Each time the main server loop accepts new connection, the corresponding socket is put into queue and then the waiting threads are invoked.

Question: What will happen if we set the following global variable in the *pre-fork* server's code (simulator stays with the same settings):

-
- PREFORK_THREADS = 4

Thread-per-request using different thread models

In our previous examples, we did not apply actual load (no request/response were sent), we were just simulating load by keeping the connection (that was opened by a client) alive. In our real applications (like

```
18:21:29,396 (MainThread) Application started
18:21:29,396 (MainThread) Socket 127.0.0.1:7777 is in listening state
18:21:29,396 (MainThread) Falling to serving loop, press Ctrl+C to terminate ...
18:21:29,396 (MainThread) Awaiting new clients ...
18:21:41,577 (Thread-1) New client connected from 127.0.0.1:40429
18:21:41,577 (Thread-1) Local end-point socket bound on: 127.0.0.1:7777 18:21:41,577 (MainThread) Awaiting new
clients ...
18:21:41,578 (Thread-1) Service offered, client 127.0.0.1:40429 is using a service ...
18:21:41,580 (Thread-2) New client connected from 127.0.0.1:40430 18:21:41,580 (MainThread) Awaiting new
clients ...
18:21:41,580 (Thread-2) Local end-point socket bound on: 127.0.0.1:7777
18:21:41,581 (Thread-2) Service offered, client 127.0.0.1:40430 is using a service ...
18:21:41,584 (Thread-3) New client connected from 127.0.0.1:40431 18:21:41,584 (MainThread) Awaiting new
clients ...
18:21:41,584 (Thread-3) Local end-point socket bound on: 127.0.0.1:7777
18:21:41,584 (Thread-3) Service offered, client 127.0.0.1:40431 is using a service ...
18:21:41,593 (Thread-4) New client connected from 127.0.0.1:40432 18:21:41,593 (MainThread) Awaiting new
clients ...
18:21:41,593 (Thread-4) Local end-point socket bound on: 127.0.0.1:7777
18:21:41,594 (Thread-4) Service offered, client 127.0.0.1:40432 is using a service ...
18:21:41,594 (Thread-5) New client connected from 127.0.0.1:40433 18:21:41,594 (MainThread) Awaiting new
clients ...
18:21:41,595 (Thread-5) Local end-point socket bound on: 127.0.0.1:7777
18:21:41,595 (Thread-5) Service offered, client 127.0.0.1:40433 is using a service ...
18:22:11,608 (Thread-1) Client 127.0.0.1:40433 finished using service
18:22:11,610 (Thread-2) Client 127.0.0.1:40433 finished using service
18:22:11,612 (Thread-3) Client 127.0.0.1:40433 finished using service
18:22:11,624 (Thread-5) Client 127.0.0.1:40433 finished using service 18:22:11,624 (Thread-4) Client 127.0.0.1:40433
finished using service 18:22:18,012 (MainThread) Ctrl+C issued closing server ...
18:22:18,013 (MainThread) Terminating ...
```

Message Board and File Server), we have actual requests like “publish message”, “get messages” or “upload file”. Hence the client handling sequence on server side is as shown in Table 4.1.

Let's suppose we have a separate thread taking care of the client's connections. Once the client is connected (step 1) the corresponding socket is either delegated explicitly to a thread to handle or it is stored in a queue

15

20

25

30

Figure 4.8: Log output of Multi-threaded server using thread-per-socket

```
12:07:21,206 (MainThread) Application started
12:07:21,207 (MainThread) Socket 127.0.0.1:7777 is in listening state
12:07:21,207 (MainThread) Falling to serving loop, press Ctrl+C to terminate ...
12:07:21,207 (MainThread) Awaiting new clients ...
12:07:28,237 (MainThread) Added item <socket._socketobject object at 0x7f58f1406c20>
12:07:28,238 (MainThread) Queue is no more empty, notifying all waiting threads 12:07:28,238 (MainThread) Awaiting new clients ...
12:07:28,238 (Thread-1) Got item <socket._socketobject object at 0x7f58f1406c20>
12:07:28,238 (Thread-1) New client connected from 127.0.0.1:42769
12:07:28,238 (Thread-1) Local end-point socket bound on: 127.0.0.1:7777
12:07:28,238 (Thread-1) Service offered, client 127.0.0.1:42769 is using a service ...
12:07:28,245 (MainThread) Added item <socket._socketobject object at 0x7f58f1406c90>
12:07:28,245 (MainThread) Queue is no more empty, notifying all waiting threads 12:07:28,245 (MainThread) Awaiting new clients ...
12:07:28,245 (Thread-2) Got item <socket._socketobject object at 0x7f58f1406c90>
12:07:28,245 (Thread-2) New client connected from 127.0.0.1:42770
12:07:28,245 (Thread-2) Local end-point socket bound on: 127.0.0.1:7777
12:07:28,246 (Thread-2) Service offered, client 127.0.0.1:42770 is using a service ...
12:07:28,249 (MainThread) Added item <socket._socketobject object at 0x7f58f1406d00>
12:07:28,249 (MainThread) Queue is no more empty, notifying all waiting threads 12:07:28,249 (MainThread) Awaiting new clients ...
12:07:28,250 (Thread-3) Got item <socket._socketobject object at 0x7f58f1406d00>
12:07:28,251 (Thread-3) New client connected from 127.0.0.1:42771
12:07:28,251 (Thread-3) Local end-point socket bound on: 127.0.0.1:7777
12:07:28,251 (Thread-3) Service offered, client 127.0.0.1:42771 is using a service ...
12:07:28,253 (MainThread) Added item <socket._socketobject object at 0x7f58f1406d70>
12:07:28,253 (MainThread) Queue is no more empty, notifying all waiting threads 12:07:28,253 (MainThread) Awaiting new clients ...
12:07:28,254 (Thread-4) Got item <socket._socketobject object at 0x7f58f1406d70>
12:07:28,254 (Thread-4) New client connected from 127.0.0.1:42772
12:07:28,254 (Thread-4) Local end-point socket bound on: 127.0.0.1:7777
12:07:28,254 (Thread-4) Service offered, client 127.0.0.1:42772 is using a service ...
12:07:28,258 (MainThread) Added item <socket._socketobject object at 0x7f58f1406de0>
12:07:28,258 (MainThread) Queue is no more empty, notifying all waiting threads 12:07:28,258 (MainThread) Awaiting new clients ...
12:07:28,258 (Thread-5) Got item <socket._socketobject object at 0x7f58f1406de0>
12:07:28,258 (Thread-5) New client connected from 127.0.0.1:42773
12:07:28,258 (Thread-5) Local end-point socket bound on: 127.0.0.1:7777
12:07:28,259 (Thread-5) Service offered, client 127.0.0.1:42773 is using a service ...
12:07:58,270 (Thread-1) Client 127.0.0.1:42773 finished using service
12:07:58,277 (Thread-2) Client 127.0.0.1:42773 finished using service
12:07:58,284 (Thread-4) Client 127.0.0.1:42773 finished using service
12:07:58,284 (Thread-3) Client 127.0.0.1:42773 finished using service 12:07:58,287 (Thread-5) Client 127.0.0.1:42773 finished using service 12:08:09,075 (MainThread) Ctrl+C issued closing server ...
...
```

10

15

20

25

30

35

40

45

Figure 4.9: Log output of Multi-threaded server using pre-fork

```
18:21:41,569 (MainThread) INFO Application start
18:21:41,570 (MainThread) INFO 5 clients will connect to 127.0.0.1:7777 ...
18:21:41,570 (MainThread) DEBUG Connection timeout 5 sec 18:21:41,570 (MainThread) DEBUG
Stay connected for 30 sec
18:21:41,570 (MainThread) DEBUG Client threads created ... 18:21:41,575 (Thread-1) DEBUG Trying to
connect to 127.0.0.1:7777 18:21:41,576 (Thread-1) INFO Connected, waiting for service ...
18:21:41,577 (Thread-1) INFO Service available! Keeping it busy for 30 sec ...
18:21:41,579 (Thread-2) DEBUG Trying to connect to 127.0.0.1:7777 18:21:41,580 (Thread-2) INFO Connected,
waiting for service ...
18:21:41,580 (Thread-2) INFO Service available! Keeping it busy for 30 sec ...
18:21:41,583 (Thread-3) DEBUG Trying to connect to 127.0.0.1:7777 18:21:41,584 (Thread-3) INFO Connected,
waiting for service ...
18:21:41,584 (Thread-3) INFO Service available! Keeping it busy for 30 sec ...
18:21:41,591 (Thread-4) DEBUG Trying to connect to 127.0.0.1:7777 18:21:41,592 (Thread-4) INFO Connected,
waiting for service ...
18:21:41,593 (MainThread) DEBUG Client threads started ...
18:21:41,594 (Thread-4) INFO Service available! Keeping it busy for 30 sec ...
18:21:41,594 (Thread-5) DEBUG Trying to connect to 127.0.0.1:7777 18:21:41,594 (Thread-5) INFO Connected,
waiting for service ...
18:21:41,595 (Thread-5) INFO Service available! Keeping it busy for 30 sec ...
18:22:11,607 (Thread-1) INFO Finished using a service ...
18:22:11,608 (Thread-1) INFO Disconnected ...
18:22:11,609 (Thread-2) INFO Finished using a service ...
18:22:11,610 (Thread-2) INFO Disconnected ...
18:22:11,611 (Thread-3) INFO Finished using a service ...
18:22:11,612 (Thread-3) INFO Disconnected ...
18:22:11,623 (Thread-5) INFO Finished using a service ...
18:22:11,624 (Thread-5) INFO Disconnected ...
18:22:11,624 (Thread-4) INFO Finished using a service ...
18:22:11,624 (Thread-4) INFO Disconnected ...
18:22:11,624 (MainThread) DEBUG Client threads dead ...
18:22:11,625 (MainThread) INFO Terminating ...
```

5

10

15

20

```

12:07:28,236 (MainThread) INFO Application start
12:07:28,236 (MainThread) INFO 5 clients will connect to 127.0.0.1:7777 ...
12:07:28,236 (MainThread) DEBUG Connection timeout 5 sec 12:07:28,236 (MainThread) DEBUG
Stay connected for 30 sec
12:07:28,237 (MainThread) DEBUG Client threads created ... 12:07:28,237 (Thread-1) DEBUG Trying to
connect to 127.0.0.1:7777 12:07:28,239 (Thread-1) INFO Connected, waiting for service ...
12:07:28,239 (Thread-1) INFO Service available! Keeping it busy for 30 sec ...
12:07:28,244 (Thread-2) DEBUG Trying to connect to 127.0.0.1:7777 12:07:28,245 (Thread-2) INFO Connected,
waiting for service ...
12:07:28,246 (Thread-2) INFO Service available! Keeping it busy for 30 sec ...
12:07:28,248 (Thread-3) DEBUG Trying to connect to 127.0.0.1:7777
12:07:28,253 (Thread-4) DEBUG Trying to connect to 127.0.0.1:7777 12:07:28,253 (Thread-3) INFO Connected,
waiting for service ...
12:07:28,254 (Thread-3) INFO Service available! Keeping it busy for 30 sec ...
12:07:28,254 (Thread-4) INFO Connected, waiting for service ...
12:07:28,255 (Thread-4) INFO Service available! Keeping it busy for 30 sec ...
12:07:28,257 (MainThread) DEBUG Client threads started ... 12:07:28,257 (Thread-5) DEBUG Trying to
connect to 127.0.0.1:7777 12:07:28,258 (Thread-5) INFO Connected, waiting for service ...
12:07:28,259 (Thread-5) INFO Service available! Keeping it busy for 30 sec ...
12:07:58,267 (Thread-1) INFO Finished using a service ...
12:07:58,268 (Thread-1) INFO Disconnected ...
12:07:58,276 (Thread-2) INFO Finished using a service ...
12:07:58,276 (Thread-2) INFO Disconnected ...
12:07:58,283 (Thread-4) INFO Finished using a service ...
12:07:58,283 (Thread-4) INFO Disconnected ...
12:07:58,284 (Thread-3) INFO Finished using a service ...
12:07:58,284 (Thread-3) INFO Disconnected ...
12:07:58,287 (Thread-5) INFO Finished using a service ...
12:07:58,287 (Thread-5) INFO Disconnected ...
12:07:58,288 (MainThread) DEBUG Client threads dead ...
12:07:58,288 (MainThread) INFO Terminating ...

```

Figure 4.10: Simulator output testing Multi-threaded server (thread-per-socket)

15

20

25

30

Figure 4.11: Simulator output testing Multi-threaded server (pre-fork)

for a thread to pickup later. In both scenarios the steps 2,3,4 and 5 are done by a separate thread and only step 1 is left for main thread. We assume that the steps 2,3 and 4 are the most time consuming and therefore it is good to handle them in a separate thread. The actual request handling (in step 3) may be very fast, but slow network might be slowing down the whole client handling (steps 2 and 4). In other scenarios, the request handling (step 3) might be a CPU intensive task and the actual request/response data might be very tiny. In this case, we will have a very fast send/receive (steps 2,4) and slow request handling (step 3). Example would be an integer prime factorization task[1], where clients request integer to factorize and the server that does the factorization. In this example most of the time is spent on actual computing and not on I/O (sending and receiving a couple of integers is doable fast even in slow network connections). Therefore we can assign the request handling to a separate execution flow as opposed to connection handling.

Next, we demonstrate the code examples of the corresponding pattern implementation compared to single threaded implementation. File [*singlethreaded_server_heavy_request.py*] illustrates the code of a simple request handling server. The request/response payload we simulate with fixed size random string (36 byte UUID). As you can see the request and response here are short in size therefore the time spent on delivery will be minimal. In *handle request* method we simulate the load with intentionally added *sleep*, hence the time spent in request handling requests may vary (dependent on DEFAULT REQ PROCESS DELAY global variable). There is no real work done by server with the requests in the *handle request* method, once the *sleep* method returns, the server replies with the same string in its response. File [*simulating many requests.py*] illustrates the code of corresponding simulator. Here again we create multiple threads that start sending the requests simultaneously and they leave in case the server could not handle request in reasonable time.

We did run the client request simulator having the following settings:

- DEFAULT CLIENTS = 5
- DEFAULT TIMEOUT SEC = 10

The output of the server is illustrated in file [*singlethreaded_server_heavy_request-log.txt*], simulator output is illustrated in file [*singlethreaded_server_heavy_request_simulator-log.txt*]. As you can see only 2 requests were

successfully processed by a server and 3 clients left with no response received.

In order to make a server able to process the requests in time we indeed will apply the same threading patterns: either delegation or producer-consumer. This time the separate thread executes the *handle request* method with request as an argument.

Combining thread-per-socket and thread-per-request

Several patterns can be combined in one server application, for example the pipeline model can be employed if separate the tasks (receiving, processing, sending), therefore we have a fixed number of workers assigned to groups “receivers”, “handlers” and “senders”. Each task assumes some input and output, and we use three different queues to collect “connections”, “requests” and “responses”. The worker dependent on his group is responsible for taking input task from one queue and putting a result into second one.

The corresponding model is implemented in [*multithreaded server combined pipeline.py*]

The corresponding output is shown in file [*multithreaded server-combined pipeline-log.txt*] the corresponding simulator output is shown in file [*multithreaded server combined pipeline_simulator-log.txt*].

Another combination could be the fixed number of group managers and on-demand workers. In this case there are four fixed threads (main, request-manager, receiver-manager, sender-manager), each manager can create additional threads for handling the corresponding tasks. The example code is shown in file [*multithreaded server_combined delegation.py*].

The corresponding output is shown in file [*multithreaded server combined delegation-log.txt*] the corresponding simulator output is shown in file [*multithreaded server combined delegation simulator-log.txt*].

Combining threading and multiprocessing

Considering previous example it is clearly visible the two kinds of thread jobs:

- an I/O job - reading/writing to sockets
- processing job - handling requests (which may CPU intensive task)

As we already know Python’s threads are good for handling I/O but for parallelizing CPU intensive tasks they do not suit very well. The threads are virtual and in fact only one OS-thread is created for Python interpreter itself (many-to-one model). Therefore the Python threads will never be scheduled to different CPU cores, and in order to benefit from multi-core CPU the Python multiprocessing module has to be used. In our previous example we can just replace the request handling from Thread based to Process based producer-consumer pattern. The file [*multiprocess server_combined_pipeline.py*] illustrates the modified code of combined pipeline (multiple producer-consumers attached) with Process based request handling. In this example the request-handler threads are running in separate processes, the request-receiver threads are still in the main process. Passing the input/output tasks between the thread groups is done using the Queue (this time using multiprocessing Queue which supports interprocess communication). Having producer-consumer model for multiprocessing is therefore obvious:

- Producer-Consumer model has a fixed set of producers and consumers (fixed amount of jobs)
- Creating a new Processes on demand is expensive, it is better to create a fixed amount of processes at the beginning and then to reuse them

-
- The number of Processes in a multi-process application is usually not exceeding the number of CPU cores (or CPUs)
 - Tasks are not assigned explicitly, but through a queue
 - Porting the Python code from producer-consumer multi-threading to multi-processing is just a matter of replacing the queue implementation

The corresponding output is shown in file [*multiprocess server combinedpipeline-log.txt*] the corresponding simulator output is shown in file [*multiprocess server combined pipeline_simulator-log.txt*].

Examples of object oriented implementation

The object oriented design of the server side application can be applied even before introducing threads. In case we have server implementation like the one shown in Figure 4.4. The corresponding handler method is always a subject of corresponding client's socket. Therefore the socket can be isolated completely inside a *ClientHandler* class. What remains is the server initialization and the main serving loop. In fact, both are subjects of one running server, therefore it is wise to isolate them into a separate *Server* class. The example of OOP implementation is shown in Figure 4.12

Implementing the multi threaded version of the same server using OOP is even more simpler. Our *ClientHandler* class suits perfectly for thread-per-socket model (delegation) as it is already a subject of a connected client (through client's socket). Now lets just make it live independent of the master thread once it is created. In the Figure 4.13 we see the changes made: the *ClientHandler* becomes the subclass of Python's *Thread* class overriding its *run()* method. The Server class we do not need to touch, thanks to OOP design for the Server creating the ClientHandler remains the same, just this time handling the client is done by a separate thread.

Concerning the client side there will be no changes.

```

from socket import AF_INET, SOCK_STREAM, socket, SHUTWR
from socket import error
as soc_err

class ClientHandler():

    def __init__(self, client_socket, client_addr):
        self.client_socket = client_socket
        client_socket.setsockopt(client_address=client_addr)

    def handle(self):
        try:
            print('New client connected from %s:%d' % self._client_address)
            print('Local end-point socket bound on :%s:%d' %
                  '%s' % self._client_address, self.client_socket.send('1'))
            self.client_socket.shutdown(SHUTWR)
            print("Service offered, client %s:%d is using a service ..." %
                  '%s' % self._client_address)
            self.client_socket.recv(1)
            print('Client %s:%d finished using service' % self._client_address)
        except soc_err as e:
            if e.errno == 107:
                print('Client %s:%d left before server could handle it' %
                      '%s' % self._client_address)
            else:
                print('Error: %s' % str(e))
        finally:
            self._client_socket.close()
            print('Client %s:%d disconnected' % self._client_address)

class Server():

    def listen(self, sock_addr, backlog=1):
        self.sock_addr = sock_addr
        self.backlog = backlog
        s = socket(AF_INET, SOCK_STREAM)
        s.bind((self.sock_addr))
        s.listen(self.backlog)
        print('Socket %s:%d is listening state' % self._getsockname())

    def loop(self):
        print('Falling to serving loop, press Ctrl+C to terminate ...')
        try:
            while 1:
                clientsocket = None
                print('Awaiting new clients...')
                clientsocket, client_addr = self.s.accept()
                c = ClientHandler(client_socket, client_addr)
                c.handle()
        except KeyboardInterrupt:
            print('Ctrl+C issued closing server ...')
            finally:
                if clientsocket != None:
                    clientsocket.close()
                self.s.close()

    if __name__ == '__main__':
        print('Application started')
        s = Server()
        s.listen(('127.0.0.1', 7777))
        s.loop()
        print('Terminating ...')

```

```
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
```

Figure 4.12: Single threaded server OOP implementation

```

import logging
logging.basicConfig( level=logging.DEBUG,\\ format='%(asctime) s (%(threadName)-2s )\\n%(message)s' )
from socket import AF_INET, SOCK_STREAM, socket, SHUTWR from socket import error
as soc_err from threading import Thread

class ClientHandler(Thread):
    def __init__(self, client_socket, client_addr): Thread.__init__(self)
    self.client_socket = client_socket
    self.client_address = client_addr

    def run(self):
        self.handle()

    def handle(self):
        self.start()

    def __handle(self):
        try:
            logging.info('New client connected from %s:%d'\\n' % self.client_address)
            self.client_socket.send('1') self.client_socket.shutdown(SHUTWR) logging.debug('Service offered, client %s:%d is using a'\\n' % self.client_address')
            self.client_socket.recv(1) logging.info('Client %s:%d finished using service'\\n' % self.client_address)
        except soc_err as e:
            if e.errno == 107:
                logging.warn('Client %s:%d left before server could handle it'\\n' % self.client_address)
            else: logging.error('Error: %s' % str(e))
        finally:
            self.client_socket.close()
            logging.info('Client %s:%d disconnected' % self.client_address)

class Server():
    def listen(self, sock_addr, backlog=1):
        self.sock_addr = sock_addr
        self.backlog = backlog
        self.s = socket(AF_INET, SOCK_STREAM)
        self.s.bind(self.sock_addr)
        self.s.listen(self.backlog)
        logging.debug('Socket %s:%d is in listening state'\\n' % self.s.getsockname())
    def loop(self):
        logging.info(' Falling to serving loop, press Ctrl+C to terminate... ')
        handlers = []
        try:
            while 1:
                clientsocket = None
                logging.info('Awaiting new clients... ')
                clientsocket, client_addr = self.s.accept()
                c = ClientHandler(clientsocket, client_addr)
                handlers.append(c)
                c.handle()
        except KeyboardInterrupt:
            logging.warn('Ctrl+C issued closing server... ')
        finally:
            if clientsocket != None:
                clientsocket.close()
            self.s.close()
            map(lambda x: x.join(), handlers)
    if __name__ == '__main__':
        logging.info('Application started')
        s = Server()
        s.listen(('127.0.0.1', 7777))
        s.loop()
        logging.info('Terminating... ')

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

Figure 4.13: Multi threaded server OOP implementation

Shared state

In order to get the idea of the shared state, we should first make sure we understand the meaning of the state in general. It's easy to get it if you imagine a white-board drawing application (like famous windows paint application). Eventually, it gives you the drawing facility - white pane and the tools applicable for drawing. Considering the implementation, the drawing itself is just a matrix of RGB pixel values (the triples (0-255,0-255,0-255)) that are representing the colors. The process of drawing is then reduced to alternation of pixel values by coordinates in the picture. Speaking about the state here - the whole drawable area is the state, all the pixel values contribute to it. Once, we change at least one value - the state of the whole drawable area gets changed.

Everything looks easy once we are sure that there is only one entity alternating the drawable area at each separate moment (like one user is drawing in windows paint). The things get more complicated when we allow many users to take part in the drawing process in our drawing application. Hence, we have to take into account the following:

1. Real-time drawing. The connected users want to see all changes immediately. Once some changes are introduced by one user or many users, the others should be notified immediately.
2. Collision resolving. Relevant actions need to be taken, when concurrent modification of the same pixel region by different users occurs at the same time.

Collision resolving in distributed systems shared state can roughly be divided into two:

- centrally coordinated – most natural in client-server applications, where the application server takes the role of making decisions on collision resolution with certain policy
- collaborative coordination – certain consensus algorithms are used for reaching collaborative agreements on shared state

In any case the collision resolving leads to the problem of global ordering of distributed events and how, based on local states to determine the global state? This in turn gives rise to the problem of synchronization of physical clocks of distributed system components.

Short-lived vs. Long-lived TCP connections

Regarding request-response protocols, we have seen two delivery strategies when the application layer protocol uses TCP for transport:

- Each request-response transaction is a separate TCP connection
 - the connection gets closed after response is received
 - the new request means new TCP connection
 - in case of short requests/responses the average lifetime of the TCP connection is insignificant
 - therefore the name "short-lived" TCP connections

- Many request-responses interactions are done using the same connection
 - connection usually being established at application's startup
 - connection stays open until application terminates
 - the request-response interactions use only one existing connection
 - the application therefore never creates more than one connection
 - average connection life-time is defined by application's runtime and may be long
 - therefore the name "long-lived" TCP connections

Next let's consider the HTTP and SSH protocols and how the corresponding TCP connections looks like.

[class demonstration HTTP vs. SSH]

```
while [ true ]; do netstat -tpn | grep -E ':22 |:80'; sleep 1; clear; done;
```

In Linux, execute the following command:

¹

This one will loop forever (kill using Ctrl+C) and each iteration it will print the active TCP connection on ports 80 and 22. The delay between iterations is set to one second, and before printing new results we clean the previous.

Next let's visit some news portal like *postimees.ee*.

Question:

- How many connections were required to load the title page ?
- How long did they stay open?

Next let's connect to remote shell of the host *newmath.ut.ee* (using UT credentials). Let's take a look at the TCP connections created and answer the same questions.

Advantages and Disadvantages

- Short lived TCP

- Advantages

- * Connection is made only when we need to send/receive the data
- * The send/receive tasks can be balanced to multiple connections
- * Good for load balancers. In case of HTTP, the static read-only content gets distributed faster as each HTTP site usually has a number of replicas behind the load balancer proxy.
- * Good for network hand-overs. In case we change the Wi-Fi network while loading HTTP resources like *postimees.ee* only a small number of requests may get lost (and even they can be re-requested once we get connected to a new Wi-Fi network).

- Disadvantages

- * Transport layer overhead. In case of HTTP the TCP handshake happens before each HTTP request. In case we

implement sessions on top of HTTP, then the session metadata is added to each request, increasing the protocol overhead.

- * Client centric: client has to contact the server before server can send any updates. No way server can initiate the connection with the client.
- * Varying number of connections. We do not know how many connections the application will need. Each connection needs a TCP port and we only have 2^{16} ports (some of them are already reserved by services).
 - For the routers serving private network it is even more worse. The ports opened for outgoing connections by individual machines in private network are reduced on one external IP of the router. Many machines in private network opening many outgoing connection might easily exhaust the router's ports.

- Long-lived

- Advantages

- * No TCP overhead - only one TCP handshake is required
 - * No protocol overhead. One connection is always one user, no additional session metadata required.
 - * Server always has a number of client connections idling (even if clients do not actively use them). Server may use those idling connections for sending notifications to clients.
 - * Less connections needed. One application → one connection, or one application+user → one connection.
 - Less ports required –

- Disadvantages

- * Connection is kept open all the time, even we do not transfer any data
 - * Impossible to load-balance the single session (one established connection)
 - We can only load balance many different connections (many individual sessions)
 - * Impossible to handover the established TCP connection, as the source IP of the client changes when we join to new IP network. The established TCP pipe is still bound to old IP. We cannot interchange the endpoints of established TCP, but we have to disconnect and re-establish TCP pipe.

Make clients get the updates immediately

In this section, we demonstrate the message board implementations using long-lived TCP connections, which allows the clients to get notifications once the server gets new message published. As a result, the clients fetch the new messages almost immediately after the message get published.

The implementation of the Message Board using short-lived TCP that we have tried in 2 (TCP Application) was fetching the messages from the server manually by the user. The protocol allowed only one request-response transaction per connection (each new transaction was done using separate connection).

Heavily modified version of Message Board is illustrated in files [*long lived tcp message_board server.py*] and [*long lived tcp message board client.py*]. The new version uses long-lived TCP connections, client gets connected early at application startup. The connection stays open till client application terminates it. Request-response transactions are done over the same connection (only one connection is required). The client application is interactive: user may publish the message multiple times per session. The new messages are automatically fetched from the server once the corresponding notification is received. The server is sending the notifications to connected clients once

new message is published. The server keeps track of fetched messages for each client to avoid double-sending the messages.

Concurrent modification

Imagine the simple “guess word” game where multiple users guess word by proposing letters one by one. Usually in such game, the players propose letters one after another and the moderator ensures that they never propose letters simultaneously. Next, we demonstrate network multi-player implementation of this game. The code may be found in files [`shared_state_gessword_game server.py`] and [`shared state gessword game client.py`]. As you can notice, we mostly reused the network code of the previous example, as the way long-lived TCP connection are handled here is mostly the same. The connection is established once the user joins the game, and the user may join even in the middle of game session. Next, during the game process user can see immediately how the letters are guessed by the other players. User can propose his guesses too. In case two users propose the same letter as a guess - the server scores the fastest. There is no strict ordering how the players are guessing - the fastest one may score more points. Once the game is over - the users have to propose new word to guess. The server starts new game using the first proposed word. The key element of the server code is the *Game* structure containing all the routines for handling concurrent actions.

Chapter 5

Remote Procedure Calls (RPC) and Distributed Objects (DO)

Remote Procedure Calls

During this lecture we will introduce Remote Procedure Calls (RPC) and Distributed Objects (DO). Therefore, we will divide our lecture into two main sections one for RPC and the second one is for DO.

26. RPC is a powerful technique for constructing distributed, client-server based applications. The RPC philosophy relies on extending the conventional or local procedure of calling. As a result, the called procedure does not exist in the same address space where the calling procedure is. Therefore, the two processes can be on the same system or on different systems that are connected via the network connection. Moreover, the use of RPC allows programmers of distributed applications to avoid details of the interface with the network. Thanks to the transport independence of RPC, it isolates the application from the physical and logical layer of the data communication mechanism. Hence, RPC allows the application to use a variety of transports and enhance the client/server model computing power.

How Does it Work?

From the programming perspective RPC is equivalent to a function call. This means that when an RPC is made the calling arguments are passed to the remote procedure and the caller waits for a response to be returned from the remote procedure.

The Figure 5.1 illustrates the flow of activity occurring when an RPC call is made between two network systems. As you can see the client makes a procedure call that sends a request to the server and waits. The thread is blocked from the processing till either a reply is received or time-out occurs. When the request arrives the server calls a dispatch routine to perform the requested service. Next, a return reply is sent to the client. The client program continues after the RPC call is completed.

The entire trick in making the remote procedure calls work is in the creation of **stub function**, which gives the user the illusion as if the call is made locally. However, on client side the stub function contains a protocol of sending and receiving messages over the network.

Breaking down all the steps in the RPC procedure (Figure 5.2) will result in following:

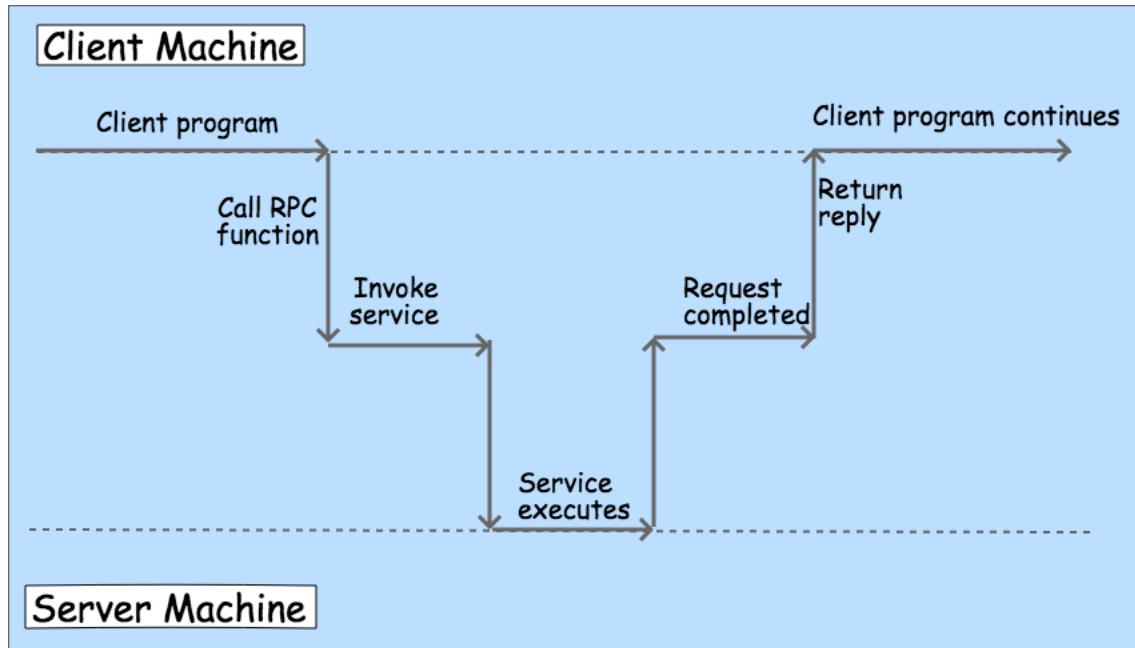


Figure 5.1: RPC Mechanism

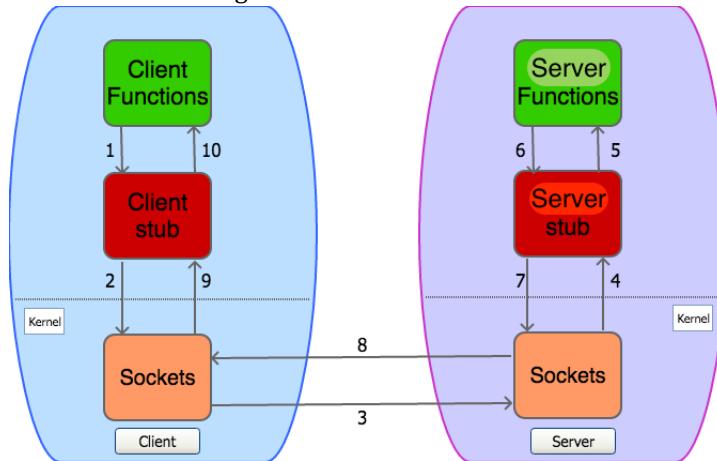


Figure 5.2: RPC Execution Steps

1. The client calls a local procedure (client stub). We described it above as a stub function. The client stub processes the parameters to the remote procedure and builds one or more network messages. The process of packaging all the arguments into network message is called **marshalling** (assembling) – it requires serializing all the data elements into a flat array-of-bytes format.
2. The client stub sends the network messages to the remote server through the kernel using sockets interface.
3. The sockets transfer the message to the remote system using some protocol.
4. The server stub (or **skeleton**) receives the messages on the server side. It **unmarshals** the arguments from the messages and preprocesses them (data conversion) if needed.
5. The server stub calls the server function and parses it with the received argument.

6. The moment the server function has finished, it returns to the server stub with the outputs.
7. The server stub preprocesses (data conversion) the outputs and marshals them into one or more messages to send them to the client stub.
8. The messages are sent back across the network to the client stub.
9. The client stub reads the messages from the local kernel.
10. The client stub preprocesses the returned data (if needed) and returns the results to the client function.
11. The client code continues its execution.

Programming RPC

The RPC was initially attributed to OS and it was the OS who handled them. For this reason, many popular programming languages were not designed with a built-in syntax for RPC. Hence, to enable the use of RPC using programming languages a separate compiler to generate the client and server stub functions was introduced. This compiler takes its input from a programmer-specified definition such as an **interface definition language (IDL)**.

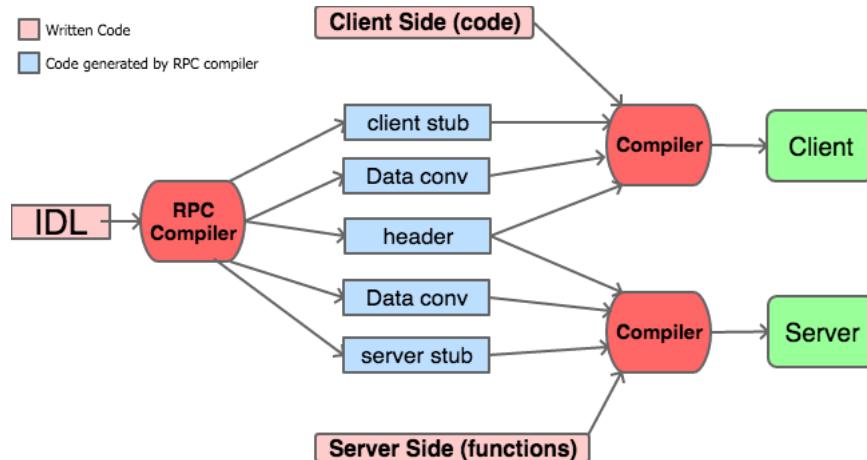


Figure 5.3: RPC Compilation Steps

The IDL usually looks similar to function prototype declaration (create enumeration of the set of functions alongside with input and return parameters). The moment RPC compiler is in action, the client and the server programs can be compiled and linked with the suitable stub functions (Figure 5.3). Note that the client side has to be modified to initialize the RPC mechanism in order to handle RPC failures in case these happen.

RPC Advantages

The advantages of using RPC can be resumed as follows:

1. The system can be independent of transport provider. Since the generation of server stub code is done automatically, it makes it available over any transport layer protocol (UDP or TCP).
2. No worries on getting a unique transport address any more. The server can bind to any available port and then register that port with an RPC name server. The client will contact this name server to find the port

number that corresponds to the program it needs. All this will be invisible to the programmer.

3. Applications on client side only need to know a single transport address: the one of the name server that is responsible for telling the application where to connect for a given set of server functions.
4. The function-call model can be used instead of the send/receive (read/write) interface provided by sockets. Users do not have to deal with marshalling parameters neither actions of parsing them out on the other side.

RPC Disadvantages

Despite the many advantages that RPC gives to us there are still tasks where RPC will introduce rather regression in performance. In sample codes below we will discover which sort of tasks the RPC underperforms compared to other application layer protocols on top of TCP or UDP.

RPC Implementations

RPC implementations are usually platform independent and even language independent. Client may be implemented in Python on Windows and the server in C on Unix, *etc.* Here we mention a couple of corresponding libraries:

- XML-RPC, and it's successor SOAP (using HTTP based transport of XML encoded calls)
- JSON-RPC and it's derivative JSON-WSP (using HTTP based transport of JSON encoded calls)
- ICE framework (full-scale communication engine providing RPC on top of proprietary protocols)
 - supports communication over TCP, UDP, TLS and WebSockets
- CORBA, classics of RPC using broker architecture

There are number of RPC implementations that are platform specific but still support different languages:

- D-Bus, interprocess communication framework for Linux (`libdbus`), and it's several ports:
 - GDBus, glib based implementation used in GTK+ applications (in GNOME desktop)
 - QtDBus, Qt based implementation used in Qt applications
 - sd-bus heavily rewritten libdbus aiming at performance boost (as part of systemd framework)
 - kdbus another replacement of libdbus
 - MSRPC Microsoft implementation of RPC (first used in Windows NT to support Windows Server domains)
 - DCOM higher level framework on top MSRPC similar to CORBA (provides communication between software products of Microsoft)
 - .NET Remoting and it's successor WCF, .NET API for building distributed application on Windows platform
- Language specific:
- Java RMI, pure Java implementation of RPC

- RPyC pure Python implementation of RPC
- DRb, Ruby ...
- *etc* ...

Other frameworks that are not focused on but offer the RPC:

- Google Web Toolkit, web development framework offering wide range of features for web-devs, including asynchronous RPC
- Apache Avro, RPC framework for Apache's Hadoop and it's derivatives like Apache Spark.

As you can see the RPC is applied a lot, and its original was used as a base for many frameworks we use nowadays, let's proceed with example application using RPC.

RPC Application

In our example application we will rely on XML-RPC implementation for Python:

- The *SimpleXMLRPCServer* module, providing server side bindings[3]
- The *xmlrpclib* module, providing client side bindings [6]

Both are based on the same generic specification of XML-RPC [8] which was aiming to achieve simplicity of RPC usage in particular. Adding the Python features made it even more simple – due to Python's dynamic nature. As a result, the XML-RPC Client in Python is skipping the IDL part completely (as it is done automatically in runtime).

Let's consider the example of client/server application we used to reference a lot: the Message Board. This time we design it to rely on RPC features and implement both server and client using Python's XML-RPC modules.

The design

The implementation is shown in Figures 5.4 and 5.5. The full-length code is illustrated in files [*rpc mboard _server.py*] and [*rpc mboard client.py*].

Distributed Objects

We know the objects in terms of OOP paradigm are the class instances, and by default are referred from the scope of the same application runtime (from the same address space). The concept "remote objects" or "distributed objects" allows objects to be referred from outside of the current application's address spaces. Moreover the objects can be distributed across the different application running on different machines. This paradigm is the further developed concept of RPC, and is just adding objects support to existing features of RPC. Therefore Distributed Objects are often referred as "Second Generation" RPC.

With distributed objects we are extending the basic concepts of object-oriented programming to the distributed systems world. With this move certain things certainly become more complex. What it means is summarised in the following table:

```

from SimpleXMLRPCServer import SimpleXMLRPCServer from
SimpleXMLRPCServer import SimpleXMLRPCRequestHandler from time import
time class MessageBoard():

    def __init__(self):
        self.m_board = {} # For storing published messages self.m
        uuid = 0 # For generating unique IDs

    def get_uuid(self): uuid = self.m
        uuid += 1
        self.m.uuid += 1
        return uuid

    def publish(self, msg, source=(None, -1)):
        ip, port = source
        t = time()
        self.get_uuid()
        self.m_board[uuid] = (uuid, t, ip, port, msg)
        return uuid

    def last(self, n=0):
        ids = map(lambda x: x[:2], self.m_board.values())
        ids.sort(key=lambda x: x[1])
        return map(lambda x: x[0], ids[n*-1:])

    def get(self, m_id):
        return self.m_board[m_id][1:]

# Restrict to a particular path class
MboardRequestHandler(SimpleXMLRPCRequestHandler):
    rpc_paths = ('/RPC2')

if __name__ == '__main__':
    mboard = MessageBoard()
    server_sock = ('127.0.0.1', 7777)

    # Create XML-RPC server
    server = SimpleXMLRPCServer(server_sock,
                                requestHandler=MboardRequestHandler)
    server.register_introspection_functions()

    # Register all functions of the Mboard instance
    server.register_instance(mboard)

    try:
        server.serve_forever()
    except KeyboardInterrupt:
        print('Ctrl+C issued, terminating ...')
    finally:
        server.shutdown() # Stop the serve-forever loop
        server.close() # Close the sockets
        print('Terminating ...')

```

1
2
3
4
5
6
7

```
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54
```

Figure 5.4: Message Board Server implementation using Python's SimpleXMLRPCServer module

```

from xmlrpclib import ServerProxy

def mboard_client_main(args):
    m = args.message
    if len(args.message) > 0
        else"
            # Message to publish if m == '-'
            :
                LOG.debug('Will read message from standard input ...')
                # Read m from STDIN m =
                stdin.read()
                LOG.debug('User provided message of %d bytes' % len(m))

            # Processing arguments
            # 2.) If -l was provided
            # Parse integer n = int(args.last) # Last
            n messages to fetch n = n if n > 0 else
            0
            LOG.debug('Will request %s published messages'\
                      '' % ('all' if n == 0 else ('last %d' % n)))
            # RPC Server's socket address server =
            (args.host,int(args.port))
            try:
                proxy = ServerProxy("http://%s:%d" % server)
            except KeyboardInterrupt:
                LOG.warn('Ctrl+C issued, terminating') exit(0)
            except Exception as e:
                LOG.error('Communication error %s' % str(e)) exit(1)
            LOG.info('Connected to Mboard XMLRPC server!')
            methods = filter(lambda x: 'system.' not in x, proxy.system.listMethods())
            LOG.debug('Remote methods are: [%s]' % ','.join(methods))

            ids = []
            msgs = []

            try:
                if len(m) > 0:
                    proxy.publish(m)
                    LOG.info('Message published')
                    ids += proxy.last(n)
                    msgs += map(lambda x: proxy.get(x), ids)
                except Exception as e:
                    LOG.error('Communication error %s' % str(e)) exit(1)
                except KeyboardInterrupt:
                    LOG.debug('Ctrl+C issued ...')
                    LOG.info('Terminating ...') exit(2)

....
```

```
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51
```

Figure 5.5: Message Board Client implementation using Python's xmlrpclib module

Objects	Distributed objects	Description of distributed object
Object references	Remote references	Globally unique reference for a distributed object; may be passed as a parameter

Interfaces	Remote interfaces	Provides an abstract specification of the methods that can be invoked on the remote object; specified using an interface definition language (IDL)
Actions	Distributed actions	Initiated by a method invocation, potentially resulting in invocation chains; remote invocations use RMI
Exceptions	Distributed exceptions	Additional exceptions generated from the distributed nature of the system, including message loss or process failure
Garbage collection	Distributed garbage collection	Extended scheme to ensure that an object will continue to exist if at least one object reference or remote object reference exists for that object, otherwise, it should be removed. Requires a distributed garbage collection algorithm.

In short we can state, that with object-oriented programming we are concerned with objects, classes and their inheritance; while with distributed object programming we deal with encapsulation, data abstraction and design methodologies.

With distributed objects certain complexities are added, in particular:

- **Inter-object communication** is to be handled. Usually this is done with the help for remote method invocation (RMI), but other communication paradigms can be used as well.
- **Additional services** come into play. Most obvious examples are naming, security and transaction services.
- **Life-cycle management** is needed for creation, migration and deletion of distributed objects.
- **Activation** and **deactivation** of distributed objects are needed to be handled as the number of distributed objects can become very large. Also node availabilities needs to be handled.
- **Persistence** is an important property that needs to be handled in case of distributed objects – the way of object survival in case of server restarts, as an example, but also across all its cycles like activation and deactivations, system failures etc.

Distributed Objects Frameworks

As the whole paradigm is just a next version of RPC, the most of the frameworks we listed in RPC section are actually supporting the Distributed Objects as well: CORBA, DCOM, JavaRMI, DRb etc.

In our sample application we will rely on pure Python implementation of Distributed Objects paradigm:

- **Pyro** - Python Remote Objects framework aiming at the simplicity of implementing distributed applications, therefore most of the initialization related routines are done automatically.

Thanks to Python's dynamic nature the whole development is even easier.

Pyro

27. Pyro is a framework that allows the users on a network to share nearly every resource and responsibility that might be distributed between different parties.

Pyro is capable of handling many resources. For example, we can resume its responsibilities into the following categories:

- Computational resources (Hardware); • Informational resources (data management);
- Business logic expertise.

How does it work?

Pyro permits one application to serve objects to another application. In most cases, these applications are running on top of a network (loopback on one machine works also). Pyro gives the capability to make a remote objects on a **Pyro server** to become attached to a proxy object on a **Pyro client**. From the moment these two objects are attached, the client application is able to treat the remote object as if it was a true local object.

Besides, once a specific object is being served by a certain machine - one or multiple machines can use it. Plus, the served machine can simultaneously serve other objects as well. Moreover, the Pyro client is capable of getting connected to many machines, where each one has its own resources.

Another interesting aspect of Pyro is the usage of the **Pyro Name Server**, which can run anywhere as far as it is accessible via TCP/IP by all the Pyro clients and servers. In case we are on a local LAN, broadcast is used to find a Name server node without specifying any extra parameters. Note that, if you have a firewall in your network then you have to use some IP addresses or domains to find the Name server. After all, using Pyro does not require Name Server. Moreover, you can create your own by using the Pyro libraries.

Pyro Application

Here we demonstrate a simple chatting application. You can notice the client side being very minimalistic, as all the objects are created on server side and then exposed to the client using Pyro URI. The server code is also minimal, containing only the chat logics, and no protocol or network related code at all (as it is being handled by Pyro). The code is illustrated in Figures 5.6 and 5.7.

From objects to components

With the development of RMI the practical implementation by practitioners designing real distributed applications noticed certain patterns that happen again and again with every project they developed. They have come up with an idea of: **component-based approaches** – a natural evolution from distributed object computing. We will first look what are the concerns with RMI and then will see the concepts on how the component-based approach can solve these.

Issues with object-oriented middleware

The first concern with object-oriented approach is that object interfaces do not describe what the implementation of an object depends on [2]. We can therefore state this issue as:

Implicit dependencies – internal (encapsulated) behaviour of an object is hidden – think remote method invocation or other communication paradigms... – not apparent from the interface

- there is a clear requirement to specify not only the interfaces offered by an object but also the dependencies that object has on other objects in the distributed configuration

The second concern is with the need to master many low-level details. The issue is within the

Interaction with the middleware – too many relatively low-level details associated with the middleware architecture

Therefore we have a clear need to:

- simplify the programming of distributed applications,
- to present a clean separation of concerns between code related to operation in a middleware framework and code associated with the application,
- to allow the programmer to focus exclusively on the application code.

Lack of separation of distribution concerns: Application developers need to deal explicitly with nonfunctional concerns related to issues such as security, transactions, coordination and replication – largely repeating concerns from one application to another

- the complexities of dealing with such services should be hidden wherever possible from the programmer

No support for deployment: objects must be deployed manually on individual machines – can become a tiresome and error-prone process, in particular with large-scale deployments with lots of objects spread over different sites with a large number of (potentially heterogeneous) nodes

- Middleware platforms should provide intrinsic support for deployment so that distributed software can be installed and deployed in the same way as software for a single machine, with the complexities of deployment hidden from the user

To resolve the issues above **component based middleware** has been emerging as a new general paradigm for designing distributed systems.

Essence of components

Software components can be defined as follows:

Software component is a unit of composition with contractually specified interfaces and explicit context dependencies only

With component based middleware all dependencies are also represented as interfaces. Each component is specified in terms of a contract, which includes a set of provided interfaces and a set of required interfaces. The provided interfaces are the ones that the component offers as a service to other components. The required interfaces are the dependencies that this component has in terms of other components that must be present and connected to this component for it to function correctly. Therefore, every required interface must be bound to a provided interface of another component. This makes up a software architecture consisting of components, interfaces and connections between interfaces.

Many component-based approaches offer two styles of interface:

- Interfaces supporting remote method invocation, as in CORBA and Java RMI,

- Interfaces supporting distributed events (as described in [2] Chapter 6).

Component-based system programming is concerned with (a) development of components as well as with (b) composition of components. In general terms, with this paradigm shift the programmer is moving from software development to software assembly.

Components and distributed systems

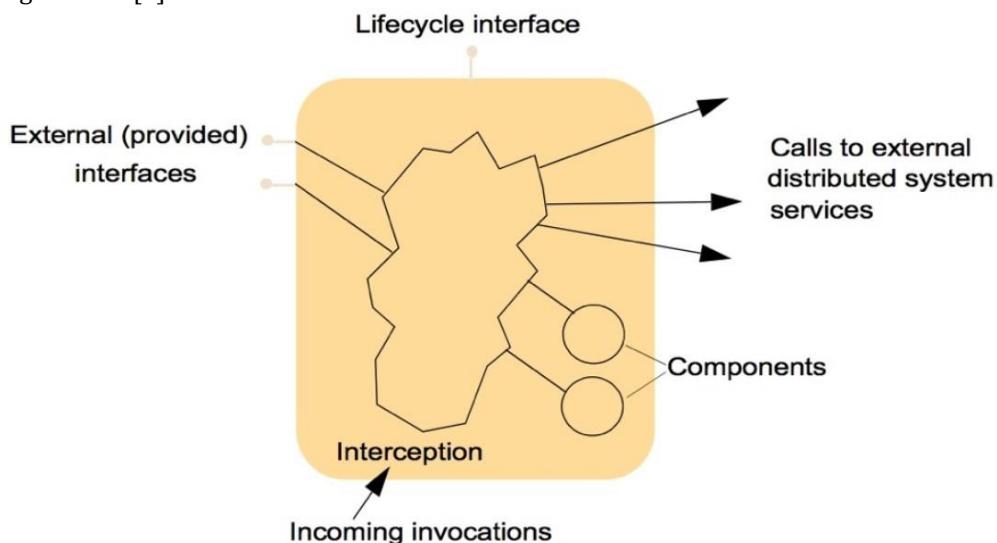
Containers:

Containers support a common pattern often encountered in distributed applications, which consists of:

- a front-end (perhaps web-based) client,
- a container holding one or more components that implement the application or business logic,
- system services that manage the associated data in persistent storage.

Components deal with application concerns, container deals with distributed systems and middleware issues (ensuring that non-functional properties are achieved).

Figure 8.12 [2] The structure of a container



The container does not provide direct access to the components but rather intercepts incoming invocations and then takes appropriate actions to ensure the desired properties of the distributed application are maintained. Middleware supporting the container pattern and the separation of concerns implied by this pattern is known as an *application server*.

This style of distributed programming is in widespread use in industry today: – we list here some of the application servers that are around:

Technology	Developed by	Further details
WebSphere Application Server	IBM	www.ibm.com
Enterprise JavaBeans	SUN	java.sun.com

<i>Spring Framework</i>	SpringSource (a division of VMware)	www.springsource.org
<i>JBoss</i>	JBoss Community	www.jboss.org
<i>CORBA Component Model</i>	OMG	[Wang <i>et al.</i> 2001] JOnAS]
<i>JOnAS</i>	OW2 Consortium	jonas.ow2.org
<i>GlassFish</i>	SUN	glassfish.dev.java.net
<i>zope.component</i>	zope.org	zopecomponent.readthedocs.io/en/latest/

Support for deployment

Component-based middleware provides support for the **deployment of component configuration**.

- Components are deployed into containers,
- Deployment descriptors are interpreted by containers to establish the required policies for the underlying middleware and distributed system services.

Container therefore includes a number of components that require the same configuration in terms of distributed system support. Deployment descriptors are typically written in XML with sufficient information to ensure that:

- components are correctly connected using appropriate protocols and associated middleware support,
- the underlying middleware and platform are configured to provide the right level of support to the component configuration,
- the associated distributed system services are set up to provide the right level of security, transaction support, etc.

Python implementations of component-based architectures

One of the component-based architectures based entirely on python is zope, which seems actually just a web-server written in python. But actually, it is based on a full component-based architecture setup, **zope Application Server** implementd in *zope.component*.

Read more about zope at <https://zopecomponent.readthedocs.io/en/latest/>.

There is a small tutorial on how to write your own first python component-based architecture application at <https://regebro.wordpress.com/2007/11/16/a-python-component-architecture/>. An extensive book on *zope.component* architecture can be found at A Comprehensive Guide to Zope Component Architecture A

Comprehensive Guide to Zope Component Architecture .

| Note: installing *zope.component* on Linux system can be done with the command: |

\$ pip install zope.component

```

import Pyro4

@Pyro4.expose class User( object ): """UserclassextendingPyROobject.
    instances of this one can be exposed over network"""
    """
def __init__( self , name , chat ):
    self . name = name
    self . private msgs = [ ] # for receiving private messages
    self . last common = 0 # for knowing last viewed public message
    self . chat = chat # parent chat

def post( self , msg ):
    """Public post"""
    self . chat . post( msg , self . name )

def post_private( self , msg , to ):
    """Private post"""
    self . chat . post_private( msg , to , self . name )

def get( self ):
    """Get all public posts for me, not viewed"""
    msgs = self . chat . get( self . last common )
    self . last common += len( msgs )
    return msgs

def get_private( self ):
    """Get all private posts for me, not viewed"""
    msgs = [ m for m in self . private msgs ]
    self . private msgs = []
    return msgs

def whoami( self ):
    return self . name

@Pyro4.expose class Chat( object ): """ChatclassextendingPyROobject.
    instances of this one can be exposed over network"""
    """
def __init__( self ):
    self . users = {}
    self . msgs = []

def register( self , me ):
    """Register user by name, return PyRO objects URI"""
    user = User( me , self )
    self . users[ me ] = user
    u_uri = daemon.register( user )
    return str( u_uri )

def post( self , msg , name = 'Unknown' ):
    """Post public message"""
    self . msgs . append( 'From %s : %s' %
        ( name , msg ) )

def post_private( self , msg , to , name = 'Unknown' ):
    """Post private message"""
    if to not in self . users . keys():
        return
    self . users[ to ]. private msgs . append( 'From %s : %s' %
        ( name , msg ) )

def get( self , idx = 0 ):
    """Get all public messages starting from idx"""
    return self . msgs[ idx : ]

def get_users( self ):

```

```
"""Get all users online"" return self.users.keys
0

if __name__ == '__main__':
    # make a Pyrodaemon
    daemon = Pyro4.Daemon(host='127.0.0.1', port=7777) # Create chat,
    expose to network using PyRO chat = Chat()
    uri = daemon.register(chat)
    print "The chat URI is : ", uri #
    Serves forever ... daemon.requestLoop()
```

2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```

import logging
FORMAT = '%(asctime)-15s %(levelname)s %(message)s'
logging.basicConfig(level=logging.DEBUG,format=FORMAT) LOG =
logging.getLogger() import Pyro4 from argparse import ArgumentParser
from sys import exit

# Constants -----
__NAME = 'PChat'
__VER = '0.2.0.0'
__DESC = 'Simple Chat Client (Pyro version)'
__BUILT = '2016-10-27' __VENDOR = 'Copyright (c) 2016 DSLab'

# Private methods -----
def __info():
    return '%s version %s (%s) %s' % (__NAME, __VER, __BUILT, __VENDOR)

if __name__ == '__main__':
    parser = ArgumentParser(description=__info(),
                           version = __VER)
    parser.add_argument('-u','--uri',\
                       help='Chat server Pyro URI', required=True)
    parser.add_argument('-n','--name',\
                       help='Username', required=True)
    args = parser.parse_args() name =
    args.name chat_uri = args.uri

    try:
        chat = Pyro4.Proxy(chat_uri)
    except Exception as e:
        LOG.error('Cannot connect to chat by URI: %s' % chat_uri) exit(1)

    try:
        my_uri = chat.register(name) me =
        Pyro4.Proxy(my_uri)
    except Exception as e:
        LOG.error('Cannot connect to my remote object by URI: %s' % my_uri) exit(2)

    while True:
        LOG.info('My messages:\n' '\n'.join(me.get())) input_msg = raw_input('Message
to send: ')
        if len(input_msg) > 0: me.post(input_msg)

```

Figure 5.6: Simple Chat server using Pyro

```
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48
```

Figure 5.7: Simple Chat client using Pyro

Chapter 6

Indirect communication

In this lecture, our objective is to understand one of the communication paradigm which is indirect communication. Hence, we will try to go through the following key topics:

- key properties of indirect communication
- space and time uncoupling
- group communication
- publish and subscribe systems
- message queues

28. *Indirect communication is defined as communication between entities in a distributed system via an intermediary with no direct coupling between the sender and the receiver or receivers*

Before we proceed with indirect communication techniques, lets remind what techniques of direct communication we did cover up to now:

Direct communication techniques:

With direct communication we have two distinguishable communication parties irregardless of how the communication is organized. This is the unbreakable limit of the direct communication - the explicit delivery between two parties, hence all these techniques are so “end-to-end” pattern oriented. Surely we can have multi-party communication pattern organized by one of the up-mentioned techniques, but it will be just several “end-to-end” patterns incorporated.

Disadvantages of direct communication techniques

Think of an example application of message board. In order for the application to function the server has to be reachable. Eventually if the server is not reachable - we cannot publish neither fetch any messages. But why is it that critical? Why can't we just leave the message as “pending for publish” (server would then publish the message once it is back on-line). We could implement it through client retrying the publish attempt till the server is finally reachable and can process the request. However we will fail if the client will also go off-line before the server is back, leaving the goal then not reached – the critical message was not published. In order for “pending for publish” to work we would want some intermediary between the message board and the client – the third party who can keep pending requests alive in case either sever or client happen to be off-line.

104

Another issue is that in case of direct communication the message board server has to be specified explicitly before we can publish or fetch messages. The first step we do is “connect” to the server (and even in case of UDP the address of the server typically has to be given). Can we do it more conveniently for the client side? What if the client does not specify any server and just issues the publish or fetch. Here we have multiple options how to achieve it:

- Static list of servers

- Predefined list of Message Board servers hard-coded in each client
- DNS name for Message Board server with multiple matching IP addresses
- Dynamic list of servers
 - Broadcast/Multicast announcement of Message Board server presence
 - Message Board server deployed on host with dynamic DNS client (updating DNS name each time the IP address changes)
 - Message Board server implemented using RCP and is exposed using RCP name server

Even if we have something like that we will solve only the issue of server's reachability (eventually at least one server will be reachable in case the client is on-line). However we wanted also the servers to be interchangeable – the client should not publish to a particular server, instead the client can publish to any server. In this case we need more intelligent servers, that should be able to synchronize the published content. Now this means that the servers have to communicate with each other as well, and here we are back with the reachability issue (for servers this time). How will the servers all know about each other's presence (in order to synchronize)? Having a finite number of servers on static IPs reachable all the time? Who wants a service taking N fixed hosts that will run forever (burning kilowatts of energy even if not used)? We are in the era of minimizing costs of everything, including the infrastructure and power consumption. Our services have to be dynamic and corresponding features became a "must have" for good service:

- Automatically increase the number of servers in case of growing demand
- Reduce the number of servers in case of demand decreasing

Coming back to our example, in order to bring the up-mentioned "dynamic features" to our architecture, we would need to have a sort of announcement or discovery of dynamic servers. Both clients and servers have to be able to understand that a new server has joined or one has left. We could rely on dynamic DNS, however this would only help in case the servers are in public space. What if we want to host the servers also in private space (without the need for manual configuring the Proxy or NAT)? Indeed for this case we would again refer to three-component pattern: where there is a third party helping the clients and servers find each other. This third-party has however no other task but providing the intermediate discovery and communication for the other architecture components. Once we have introduce an intermediate (we can call it broker) the clients are not anymore contacting the servers directly, but contacting the broker to publish or fetch the messages. The servers do not interact with clients directly, but are interacting with the broker (processing the pending requests).

With architecture like this it looks like we can achieve the specified goals, barely introducing a third party between clients and servers, hence we end up with the paradigm of "indirect communication".

Indirect communication

Indirect communication is good for cases when the users connect and disconnect very often or in the case of event dissemination where receivers may be unknown and change often. In addition, it is suitable for the

scenarios where we have so many users involved or the change is anticipated. However, it has also a few disadvantages: for example performance overhead due to the level of indirection, difficulties in management because of lack of direct coupling and finally it is hard to achieve real time properties or security. Next let's discuss the properties of indirect communication and defining the terms.

Key properties

Most of the cases described in previous sections have a certain name in the scope of indirect communication, let's list them once again from the term definition perspective.

Space uncoupling

Space uncoupling means that the sender does not know or need not to know the identity of the receiver and vice versa.

This is exactly the case where client can submit the request without caring which server will process it (in particular – the client does not have to mention the server's IP or Host name). And the other way round – the server does not care about clients' transport endpoint, server is rather request-oriented as opposed to connection-oriented. In this case we do not address the request explicitly to an addressee — to whom then should we send? The options are: to send it to everyone (group communication using multicast or broadcast) or to delegate it to an intermediate party (message bus, broker).

Time uncoupling

In time uncoupling, the sender and the receiver can have independent lifetimes. In other words the client can submit the request without making sure if the server is on-line or not. The server does not have to stay on-line 24/7.

This is the case with “pending requests” discussed above (which is why it is well suitable with “requestresponse” kind of protocols). It is not critical to submit the request to the server explicitly, request may be delegated to an intermediate layer (message bus, broker) in the form of a message. Broker then delegates the message further to a server, and after the request is processed by the server the response is delegated to the broker in the same way. In order to get a response the client has to contact the broker. In case the server is off-line at the time the client submits the request, the broker will keep the request till the server gets back on-line to process it. In case the client is off-line by the time the server sends the response, the broker will keep it till the client is back on-line to receive the response.

Space and time coupling

The architectures can vary based on the features incorporated. Here we show some of them classified into two groups based on implemented features, refer to Table 6.1:

Asynchronous communication

It looks like the paradigm of “asynchronous communication” is very close to the concept of “time-uncoupling”. Asynchronous communication allows the sender to send a message and then continue (no blocking). The response is then handled in a separate (receiver) thread, and it does not necessarily have to happen right away.

	Time-coupled	Time-uncoupled
Spacecoupled	Typical direct communication: client and server both have to be on-line and reachable. Client refers to a server explicitly, establishing a TCP connection or sending an UDP packet. Examples: FTP, HTTP, SMTP, IMAP, etc. (most of the well known protocols).	Explicit communication (no intermediates) between sender and receiver and yet one of them may be off-line during communication. Example: an off-line “state file” that can be read/written by several processes in order to exchange the messages. In this case a message can be placed for “future processes” to consider.
Spaceuncoupled	A sort of communication where sender/receiver are not addressed explicitly, but still both have to be on-line and reachable. Example: IP broadcast or multicast, here the packet is sent to “everyone” assuming the actual receiver is somewhere among the on-line peers.	Typical indirect communication that we discuss in this lecture. Sender/Receiver do address each other directly, also they may be off-line during communication. Examples: Group Communication, Publish-Subscribe, Message Queues etc.

Table 6.1: Space and Time uncoupling in distributed systems

after sending the request. The sender may even be off-line for a while between sending the request and receiving the response, which may lead to a conclusion that asynchronous communication is time-uncoupled. In fact it is not completely the case, because not all of the properties of “time-uncoupling” are preserved, like shown in Table 6.2.

Therefore we may say the asynchronous communication is not completely time-uncoupled, but “loosely” time-coupled.

Next let's discuss the corresponding technologies of the indirect communication and try out some frameworks by implementing our famous message board application on top of them.

Group communication

One of advantage of group communication is in offering service whereby a message is sent to a group and then this message is delivered to all the members. In addition, it is characterized by the fact that the sender has no awareness of the identity of the receivers and it represents an abstraction over multicast communications. Besides, from an implementation point of view it gave you the ability to do it via IP multicast or any equivalent overlay network by managing group membership, detecting failures, providing reliability and ordering guarantees.

- JGroups, Appia, NeEM - Java implementations of reliable multicast communication
- Spread Toolkit - platform neutral group communication framework, supports C/C++, Java, Python
- PGM experimental protocol

Asynchronous Communication	Time-uncoupled
<p>The request/response (or send/receive) activities are time-uncoupled in the scope of one session. Receive action does not necessarily occur after sending. Regarding the overall existence of sender and receiver, they still have to meet in the same time slot for the transfer to happen. If one of them is off-line the message delivery is impossible.</p> <p>Example:</p> <ol style="list-style-type: none"> 1. Server is on-line, Client is on-line 2. Client sends message to the Server and continues with other routines (not waiting for a reply) 3. Server receives the request and processes it 4. Server sends reply to Client triggering Client's "on-receive" callback 	<p>It should be possible to deliver the message even if one of the endpoints is off-line. Example:</p> <ol style="list-style-type: none"> 1. Server is off-line, Client is on-line 2. Client sends message to the Server and goes off-line 3. Server comes on-line and still receives the message from the Client who is already off-line. 4. Server processes the request and replies to the Client even if the Client is off-line 5. Client should receive the reply once it is back on-line (even if the Server is off-line by that time).

Table 6.2: Asynchronous Communication (direct) and Time-uncoupling (indirect)

Publish-subscribe systems

First let's discuss the Publish-Subscribe systems (also referred as Distributed event-based systems). As have already seen in previous lectures, many protocols map naturally onto request-response or remote invocation paradigm (we had our MessageBoard application employing a protocol which was very easily replaced with XML-RPC calls). Moreover, both request-response as well remote invocation paradigms are in fact "event driven", as the communication is triggered by certain kind of event. Consider the MessageBoard application, where we had all the protocol built around "publish message" event, which we say is primal. There was secondary event though "get messages", and we call it secondary because it was only needed for the cases where the Server couldn't notify the Clients about new published messages (Lecture #3). In fact we solved this issue in (Lecture #4) evolving the long-lived TCP with asynchronous notifications: all the Clients were notified by Server in case new message was published. By introducing the design like that we did in fact use the "publish-subscribe" (or "event driven") paradigm. In our case we had only one kind of event "publish message" and all the Clients were automatically subscribed to the "publish event" one once they got connected. General definition of the publish-subscribe system says:

29. *A publish-subscribe system is a system where publishers publish structured events to an event service and subscribers express interest in particular events through subscriptions which can be arbitrary patterns over the structured events.*

There is a number of frameworks implementing publish-subscriber pattern and offering corresponding API. In case the framework supports “distributed events” (publish-subscribe over network), it provides a middleware service in addition to API library. The middleware is dealing with all the network related tasks, the API refers to middleware in case event has to be delivered to a remote host. Next we will list some of the frameworks and focus on Python implementations in particular. In the end we will show the MessageBoard application ported onto one of them.

Distributed events frameworks

Also called “distributed publish-subscribe”, “event service” or “broker network”. As we have said before the idea is quite simple: a “publish-subscribe” paradigm is in fact “event driven”. We can emit events and we can subscribe for certain events, assuming once event is emitted a certain action will be triggered.

- Centralized (Broker Service)
 - CORBA Event Service
 - OSE Framework
- Distributed (Broker Network)
 - Apache Kafka
 - DDS (Data Distribution Service)

None of the above mentioned are in fact pure Python, and rather platform and language neutral middlewares with Python interfaces.

Message queues

Also called “message brokers”, middleware is build around central entity storing the queues that can be shared/subscribed by different entities. Allows building different communication paradigms, which are well illustrated by following link (RabbitMQ tutorial):

[<https://www.rabbitmq.com/getstarted.html>]

- Python supported frameworks:
 - snakemq - implements own message brokers (not compatible with AMQP neither ZeroMQ)
 - pika python library (AMQP client library, compatible with RabbitMQ server)
 - pyzmq python library (ZeroMQ python bindings)
 - celery (distributed task queue implementation, compatible with multiple message brokers)

Bibliography

- [1] Connelly Barnes. Integer factorization algorithms. <http://www.connellybarnes.com/documents/factoring.pdf>, 2004. [Online; accessed 18-October-2016].
- [2] George F Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed systems: concepts and design*. Pearson education, 2005.
- [3] Python Software Foundation. Python basic xml-rpc server. <https://docs.python.org/2/library/simplexmlrpcserver.html>, 2016. [Online; accessed 27-October-2016].
- [4] Python Software Foundation. Python higher-level threading interface. <https://docs.python.org/2/library/threading.html>, 2016. [Online; accessed 18-October-2016].
- [5] Python Software Foundation. Python process-based threading interface. <https://docs.python.org/2/library/multiprocessing.html>, 2016. [Online; accessed 18-October-2016].
- [6] Python Software Foundation. Python xml-rpc client api. <https://docs.python.org/2/library/xmlrpclib.html>, 2016. [Online; accessed 27-October-2016].
- [7] Cameron Hughes. *Parallel and distributed programming using C*. Addison-Wesley, Boston, 2004.
- [8] Inc. Scripting News. Xml-rpc. <http://xmlrpc.scripting.com/>, 2016. [Online; accessed 27-October-2016].

E-Commerce



E-COMMERCE
FUNDAMENTALS



GraphicEra

Deemed to be University

Accredited by NAAC with Grade A

NBA Accredited Programs in ECE, CSE & ME
Approved by AICTE, Ministry of HRD, Govt. of India

DEHRADUN, UTTARAKHAND, INDIA

1. Introduction	10
Buyers and sellers	10
E-commerce	11
E-commerce statistics	12
2. What Is E-commerce?	16
What is e-commerce?	16
The history of e-commerce	18
How to get started?	19
3. Creating a Business Plan	22
What is a business plan?	22
Essentials	23
Business name	23
Your website	24
Logo	25
Product information	26
Product to sell	26
Place to sell	27
Investment	28
Web domain	28

Hosting	
28	
Technical support	
29	
Product related costs	
29	
Other things in the business plan	30
Predicting profit	
30	
Pricing	30
Special offers	
31	
Plan for promotion	
31	
Determining goals	
33	
Benefits of having a business plan	34
Organization	
34	
Time management	
34	
Asset distribution	
34	
Unified approach	34
How to use a business plan?	35
4. Legal Guide for E-commerce	37
Trademark registration	
37	
Terms of use	
38	Refund
	policy
40
Privacy policy	
41	

5. Having an E-commerce Website	45
Website
45	
Store
46	
Product information
46	
Product name
46	
Product image
46	
Product description
47	
Buttons and fields
48	
Order information
49	
Shopping cart
50	
Hosted shopping cart
50	
Licensed shopping cart
50	
Shopping cart abandonment
50	
Shipping
53	
Shipping yourself.....
54	
Fulfillment warehouse
54	
Drop shipping
54 Payment gateway	
.....	55
Website security
57	

PCI standards	57
SSL certificate	
57	
6. Choosing an E-commerce Platform	59
Shopify	
59	
BigCommerce	
60	
Magento	
61	
WooCommerce	
63	
Volusion	
65	
7. How to Promote Your Business	69
Leverage the power of Facebook	
69	
Facebook profile	69
Facebook group	69
Facebook page	
70	
Facebook ads	71
Spread the news on Twitter	
71	
Network on Pinterest	
73	
Explore the power of video	
74	
Embrace the world of social media	75
Get on Google	
76	
Check your website again	77

SEO	
77	
Website analytics	
78	
Mobile-friendliness	
78	
Write a blog post	
78	
Share on your blog	
78	
Write a guest post	
78	
Interview an influencer	
79	
Start building your mailing list	79
8. Strategies to Increase Sales	82
Social media marketing	
82	
Choose the networks	
82	
Create a plan	
82	
Be active on social media	
83	
Explore paid options	84
Analyze the performance	
85	
Email marketing	
85	
Subscribers	
86	
Customers	
86	
Analyze the abandoned cart	87

9. E-commerce SEO	90
Keyword research	90
Site structure	91
Structure	91
Friendly URLs.....	92
Internal links	93
Structured data markup helper	93
On-site SEO	94
Title tag	94
Product pages	95
User-generated content.....	99
Site search	101
Link building for e-commerce	102
Influencer outreach	102
Partnership	102
Affiliate marketing	103
10. Common Issues with E-commerce SEO	105
Duplicate content	105
Noindex tag	105
Canonical tag	105
Content	106

Pagination
106	
Series
107	
Load more
107	
Scrolling
107	
Site speed
107	
11. Mobile SEO for E-commerce	110
Have a mobile-friendly configuration	110
Increase site speed
111	
Video format
111	
Design with mobile users in mind	112
12. Exploring Online Marketplaces	115
What is an online marketplace?	115
Online marketplace vs. e-commerce website.....	116
Hosting
116	
Payment
116	
Competition
117	
Customers
117	
Selling on Amazon
117	
What to sell?
118	
How to start?
	119

Shipping and getting paid	121
Selling on eBay	121
Start selling	122
An eBay store	123
Other marketplaces.....	
125	
13. Understanding Customers.....	128
Types of customers	128
Target group	129
Potential customers	129
First-time customer	130
Recurring customer	130
Buyer persona	
131	
User experience	132
Understanding the buying process	133
What is the first thing potential customers will notice?	133
Who are your customers?.....	133
Why do they buy from you?	133
How do they buy?	
134	
14. E-commerce Glossary	136
15. Questionnaire	144
Questions	144

Answers
157	
16. Conclusion	159
Advantages and disadvantages
159	
Consistency and dedication
159	
E-commerce and online marketing
160	

1

Introduction

1. Introduction

The internet has changed people's lives in numerous ways. From the way we look for information to the way we buy things. Nothing can stop the fast development of this immersion into the new technologies that we have now reached the point where we can no longer even imagine our lives without the internet.

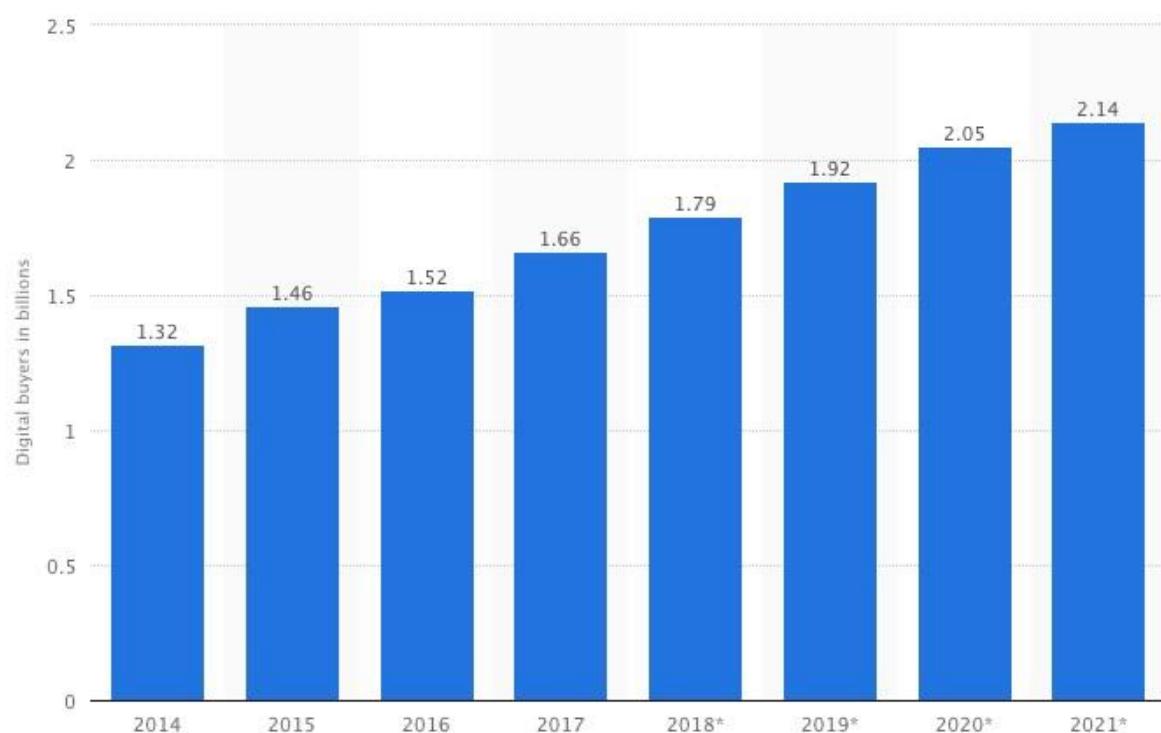
In such a reality, people are trying to adapt their lives and to get the best out of it. They want to improve their lives and become more efficient. Why would they go to the restaurant for dinner if they can order food in? Why would they spend time shopping when they can order everything they need in less than ten minutes? There is a constant trend to use the internet, not only to find out information but to get things done.

Buyers and sellers

On the one hand, we have buyers, people who try to make their lives easier and better by buying online. Regardless if we talk about making a grocery shopping quick and easy, or ordering an item from abroad. The internet enables fulfilling those needs and the ability to obtain something you could not even imagine two decades ago. Besides convenience, online shopping also provides a great selection of products and services, as well as competitive prices. This is something that provides an incredible advantage over any local retail.

Due to the growing expansion of the internet and the fact that new generations are growing up with the internet experience being part of their lives from the early days, the number of digital buyers is constantly on the rise. From 1.32 billion digital buyers in 2014, this number is expected to grow to more than 2 billion by 2020.

Number of digital buyers worldwide from 2014 to 2021



© Statista 2017

The estimated number of digital buyers worldwide

Image: <https://www.statista.com/statistics/251666/number-of-digital-buyers-worldwide/>

On the other hand, we have people who create or simply resell the products through the usage of the internet. These sellers are turning to the internet as a tool to run their business and ultimately earn their living. Some are choosing the existing platforms specialized in the sale of goods, such as eBay, while others decide to create their online stores. Either way, the goal is to reach consumers and encourage them to buy. As a seller, your goal is first to reach the digital buyers, and then persuade them to buy from you, using a series of promotional activities, online interaction, etc.

E-commerce

Buyers and sellers together make up e-commerce, short for electronic commerce. E-commerce is the transaction between a buyer and a seller, which is done through online technologies. It is the process of using the internet to buy or sell online.

Despite the fact that e-commerce is something available to anyone and in spite of how easy it might be to start selling products and services online, there are many struggles to make this business viable and actually capitalize your efforts in the digital world. As the market gets saturated, day by day, it becomes even more difficult to place your products on that market.

A modern consumer is overwhelmed with numerous advertising messages each day. He is also mobile, which means easy access to the information and ability to buy on the go. A modern consumer is also doubtful and equivocal. Faced with plenty of choices and alternatives, a modern consumer needs to make decisions on a daily basis. All of this shapes the approach of modern businesses whose goal is to reach that modern consumer and persuade him that theirs is the product they should buy. It is not an easy task, but with the right strategy and tools, you should be able to introduce and position your company on the e-commerce market.

E-commerce statistics

Getting started with e-commerce is a process that requires learning about different aspects that influence setting up and running an e-commerce website. It is also helpful to have in mind the e-commerce statistics that provide a glimpse into the importance of this industry and the trends that are tremendously shaping the way strategy is implemented.

- Millennials and Gen Xers are similar in their likelihood to buy online—more than 90% of internet users in both age groups have made a digital purchase in the past year. But millennials are more likely to research products or make a purchase via their smartphone. ([eMarketer](#))
- In 2015, the global population amounted to around 7.3 billion people, of which 1.4 billion people purchased goods and/or services online at least once. In total, they spent \$2,272.7bn online, which results in an average spending per e-shopper of \$1,582. ([EcommerceWiki](#))
- Worldwide retail sales—including in-store and internet purchases—will surpass \$22 trillion in 2015, up 5.6% from 2014. Retail ecommerce sales, those purchased over the internet, will make up 7.4% of the total retail market worldwide, or \$1.671 trillion. By 2019, that share will jump to \$3.578 trillion, yet retail ecommerce will account for just 12.8% of retail purchases. ([eMarketer](#))
- About 71% of consumers are shopping online to find the best price. ([IBM](#))

- 53% of global internet users have made an online purchase in 2016. ([SocialMediaToday](#))
- Primary reason for digital shoppers in the United States to abandon their carts is the cost of shipping. ([Statista](#))
- Nine out of 10 of the survey participants said free shipping was the No. 1 incentive when asked what would make them shop online more often. ([MarketingLand](#))
- Average e-commerce conversion rate vary from 3% to 4%. ([SocialMediaToday](#))
- 42 percent of online shoppers worldwide stated that they preferred to pay via credit card, while 39% preferred PayPal. ([Statista](#))
- In 2017, global B2C e-commerce sales are expected to reach 2,143 billion U.S. dollars. ([Statista](#))
- Customer satisfaction is the number one success metric for marketers today. ([Salesforce](#))
- 39% of people will stop engaging with a website if it takes too long to load. ([SocialMediaToday](#))
- Eighty-eight percent of shoppers characterize detailed product content as being extremely important. ([Forbes](#))
- Over 90% of people buying on Amazon wouldn't purchase an item with less than three stars. ([Forbes](#))
- One-click checkouts and e-wallets will become a game-changer in 2017. ([HuffingtonPost](#))
- 75% of people participate in webrooming to find the lowest price, while 72% do it to compare the products. ([RetailPerceptions](#))
- The main reasons why customers webroom over showroom is because they don't want to pay for shipping (47%) and because they like to go to a store and touch and feel the product before they buy it. ([Shopify](#))

- Social commerce accounted for 5% of ecommerce. But predictions are healthy and show a 25% growth rate over the next five years. Mobile ads are the backbone of this growth. ([HuffingtonPost](#))
- Mobile remains a powerful upward force for ecommerce. US retail mcommerce will climb 43.2% in 2016, paced by a 78.3% jump in sales via smartphones. ([eMarketer](#))
- In 2015, mobile commerce accounted for 30 percent of all U.S. e-commerce in 2015, and U.S. retail e-commerce dollars are projected to grow to almost 482 billion in 2018. ([PointSource](#))
- 56% of smartphone or tablet users intend to utilize their devices to search for and/or buy gifts this holiday season. ([Invesp](#))

With the analysis of the data from numerous researches on the state of e-commerce, the following can be concluded:

- E-commerce market is expanding each year
- Mobile commerce is increasing its share in the overall e-commerce
- It is important to understand the profile of digital buyers
- Digital buyers want quick and easy way to buy online
- They also want the best price and free shipping
- Webrooming is slowly taking over showrooming

2

What Is E-commerce?

2. What Is E-commerce?

As this e-commerce trend is slowly taking over, changing the habits and the mindset of online users, increasing the number of digital buyers, there comes the time to explore this segment of the digital world and explore the opportunities it provides.

Regardless if you are simply starting your business and you want to focus on e-commerce, or if you already have an established business you want to transfer to the digital realm, you will need to understand the essence of the process and practices that can help with taking the most out of it.

What is e-commerce?

The term e-commerce stands for electronic commerce. It is usually abbreviated as e-commerce, but it is sometimes used as ecommerce, eCommerce or simply as EC. The terms e-business or etailing are sometimes used as synonyms. The term mCommerce is gradually taking its place as a part of e-commerce that is focused on transactions via mobile devices.

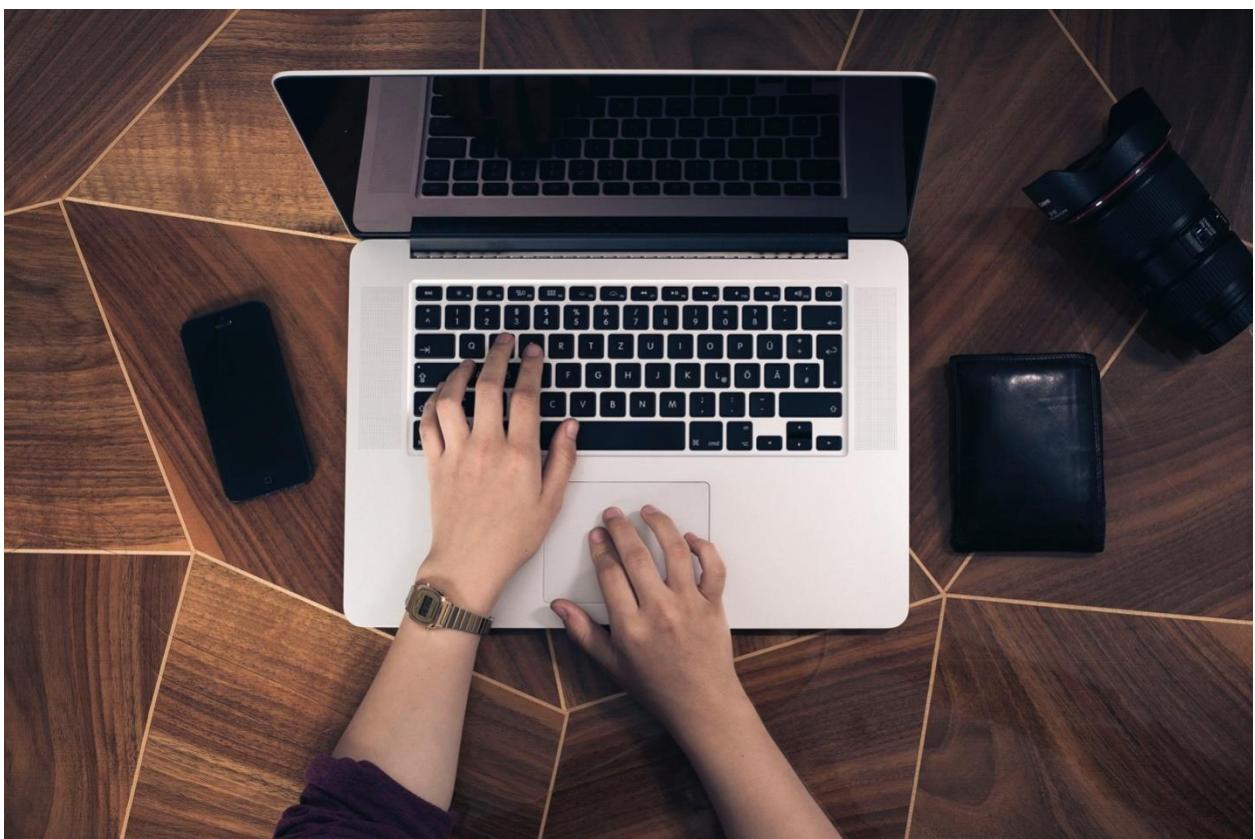


Image: <https://unsplash.com/photos/67I-QujB14w>

E-commerce is the process of buying and selling online. It is a business transaction where the funds and data are transferred through an electronic network. This way, the sales of all kinds of

products, both physical and digital, and services are arranged between two participants in the transaction. Depending on those participants, e-commerce includes several types of transactions:

- Business to business (B2B)
- Business to customer (B2C)
- Customer to customer (C2C)

E-commerce has enabled companies and individuals to join this vast marketplace and develop their business using a sales model that brings significant advantages for both parties. The main benefit of this business model is convenience. E-commerce is up and running all the time, 24 hours a day, seven days a week. This means that buyers can obtain products whenever they find it convenient, without thinking about office hours, how to get to the location and similar things they would otherwise need to consider. Another edge that e-commerce has over brick-and-mortar stores is the selection. As opposed to physical shops, e-commerce offers a much greater diversity of products. The fact that you can always order products from different cities or even countries means that the selection is huge. For business, these benefits significantly increase their reach because they are open to customers at any moment, accepting orders with no geographical limitations.

Despite these benefits, e-commerce also has a couple of disadvantages for customers. Firstly, they are unable to see and touch the product. Although the images are quite helpful, you simply cannot get the whole picture as you would when you see the product in person. This can sometimes lead to customers feeling disappointed when they receive the product that does not fulfill the expectations. Buying online also deprives customers of instant gratification. When you buy a product in a physical store, you can instantly use it, test it and see the benefits of that particular product. On the other hand, with e-commerce, you have to wait for the product to be shipped and delivered to you, which can take anywhere from a couple of hours to a couple of weeks.

E-commerce store is an online shop. From a business perspective, e-commerce is the place to advertise products and services, and this is most commonly done through a website.

If you design your website, you will have your own webshop. You will have to take care of site maintenance and management of the store, which requires a more significant investment. On the other hand, there is always an option to join the online retailers and sell your products directly through their website, without even having your own. While this is an easier solution regarding management and having in mind the fact that these sites offer a large community base which can find your products more easily, it still is a more competitive surrounding, as opposed to having your own online store.

Anyone can join the world of e-commerce because the limitations typical for running a traditional store are simply non-existent. There are no costs of building lease, employees, or comprehensive

administration. Instead, all you need is a website and instructions on how to start. This means that an online store is easier and cheaper to set up than a traditional store, which is the main reason why a lot of people are turning to this activity as their way to earn their living.

Besides the actual presentation of products on your website, you will also need to understand the fundamental processes involved in e-commerce, such as payment processing, shipping, buying process, etc.

The history of e-commerce

The first origin of e-commerce is the process of sharing electronic documents and arranging sales, which can be traced back to 1970s. A couple of commercial sales websites were founded in the early 1990s, but the beginnings of what we now know as e-commerce started with the introduction of Amazon and eBay in 1995. These websites completely revolutionized the ecommerce industry and connected buyers and sellers worldwide. Interestingly, Amazon, which now takes up a large percentage of the e-commerce market share, first reported the annual profit in 2003, eight years after its creation.

Over the years, the development of other marketplaces was initiated globally. The ways of ecommerce evolved as the internet itself did. The first obstacle for the safe online transaction was resolved with the introduction of security protocol in 1997.

The first company that recorded a million dollars in online sales was Dell.com in 1997. The success of their business was mainly attributed to the fact that the company enables customers to browse the website and assemble PCs by choosing pieces. The company did not use intermediaries but directly sold to the customers, while providing them with full control over the buying process, which is what made this business model so successful.

PayPal was introduced in 1998, and it was acquired by eBay in 2002. The company is now one of the most popular methods for online payments, with over 200 million users worldwide.

After the 2000s, we have witnessed a massive expansion of the internet usage on the global level, which has also led to the further development of e-commerce. As online security was a burning issue, the Payment Card Industry Security Standards Council (PCI) was formed in 2004, to regulate business compliances regarding security requirements.

E-commerce slowly became more available to the online users as a mean of setting up and running a business, which eventually led to the new e-commerce solutions and software, such as Shopify (introduced in 2004), WooCommerce (introduced in 2011), BigCommerce (introduced in 2015). Once a place for big players with their huge investments, e-commerce is now available to anyone anywhere in the world. Other online marketplaces were introduced, following the example of Amazon and eBay. Some of those are Etsy, Bonanza, and Craigslist.

Finally, as the mobile is in full swing, mobile e-commerce is gaining more and more importance for businesses. It is the perfect place to reach digital buyers who will most likely turn to the internet not only to buy but to get recommendations, suggestions, and reviews about the products and services they are interested in.

How to get started?

If you have decided that e-commerce is an area you would like to explore as an opportunity to earn the profit, you will need to start with developing this plan in details.

Firstly, start by deciding which platform you are going to use. The choice is basically between having your shop on your website or selling on marketplaces. And this choice depends on your business. For example, if you are a small-sized company, with limited resources for web development, starting with marketplaces is perhaps the better alternative. In case you have enough resources, having your own shop is recommended. However, even if you do decide to have your online store, nothing is stopping you from joining the marketplaces as well. In fact, it is advisable to use everything that is available to you to start making the profit through ecommerce.

You will also need to think about the strategy you are going to use, which will include a set of actions you will use to present and promote the products and services to the online users. The next step in getting started is choosing the online channels you will use. Have in mind that social aspect has a major role in online marketing in general, which means that your ecommerce website will also find this approach profitable and worthy of exploring.

Besides the presentation and promotion of products, you will also need to think about the legal aspect, how to handle shipping and payments, and finally about customer care. Since all of these aspects directly influence the performance of your e-commerce, you should consider this a serious business endeavor.

We are all witnessing the extraordinary growth of electronic commerce market and sales realized in the digital world, but if you want to become a part of that world, you will have to get acquainted with rules that are sometimes different from running a traditional commerce business. If you have an idea about what and how to sell, you are to a good start. But to shape this idea into a concrete strategy, a strategy that will yield good results, you should start working on a business plan.

3

Creating a Business Plan

3. Creating a Business Plan

It all starts with an idea. You decide to open an online shop, with a particular idea in mind. However, to make this idea concrete, to turn it into a business which will eventually bring profit, you will need to organize it properly. Even if you are completely new to the digital world, it is essential that you catch up with the trends and practices that will enable you to run your business successfully. If the assets are limited, and you are unlikely to afford an expert to help you with setting up, you should explore a DIY method, where you would study and explore each aspect of running an e-commerce website.

To make sure you keep your ideas and thoughts organized, start creating a business plan.



Image: <https://pixabay.com/en/plan-objective-strategy-goal-2372176/>

What is a business plan?

A business plan is a document which highlights business goals and strategies on how to reach those goals. It also defines the key aspects of starting and running a business. Essentially, it is a roadmap that will include where you want to go and how to get there. It is a way to describe where you see your business with details about the actions that are needed to reach that destination.

There is no need for a business plan to be long or exhaustive. With this DIY method, you will be creating this plan for yourself, and you need the essentials to keep everything organized. It is likely that no one will analyze that plan, except for you and people you get involved in the project, so make it concrete and spot on. Focus only on those segments that will help you with defining your vision and determining a set of actions you will take.

Try to group your ideas into these three groups.

Essentials

This part of a business plan should contain basic information about your business. These include:

Business name

The business name is the name you choose for your business. It is the name that should eventually become recognizable, distinctive and associated with your business. It will become a part of your brand. This is why determining a business name is a crucial decision in the business plan. It requires a lot of thinking, piling up the ideas and eventually choosing the right one.

When it comes to making a decision on which business name to choose, it is helpful to know characteristics of a good business name. A good business name should:

- Be easy to spell
- Be relevant and associated with your products
- Trigger emotions
- Contain keywords
- Be clear and professional
- Be powerful and have deeper meaning
- Be unique

A good business name should not:

- Overuse keywords
- Have irrelevant keywords
- Have negative connotations
- Infringe upon anyone's rights
- Be offensive
- Be difficult to spell

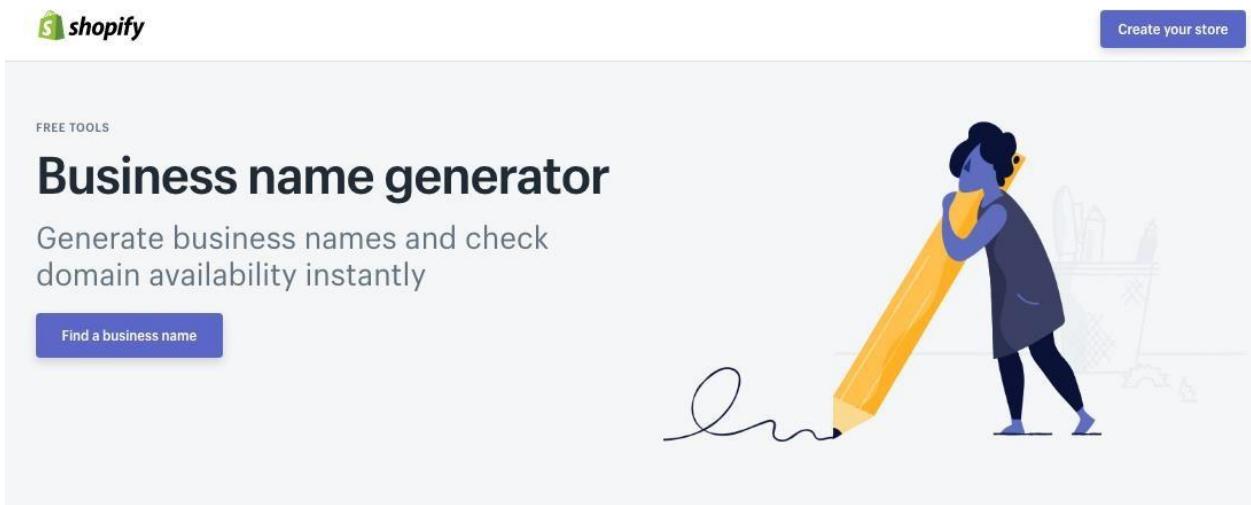
Make sure you choose a perfect name by following these steps:

- Know the characteristics of a good business name
- Create a list of names/keywords that can be associated with your business
- Expand the list by adding synonyms or related phrases
- Combine the words and phrases
- Start narrowing down the list
- Check the domain name availability
- Request feedback from employees, business partners or analyze the results of A/B testing

The final step is when you choose among the selected few. You might want to consider the opinion of the people who are directly related to your business. Alternatively, you could use A/B testing where you would create two identical landing pages. You could use a tool such as [LeadPages](#) or [Unbounce](#) for this purpose. You would then share the page, gain traffic and examine the performance of two or three name variations.

Your website

Ideally, your business name should match your website name. Use a tool, such as this [business name generator](#), and check if a particular website domain name is available. The tool will also show plenty of alternatives, so even if the name is not available, you will get suggestions on which modifications of that name are available.



Enter a word that you want your business name to include

Generate names

Please don't infringe on other brands' trademarks with this tool

Image: <https://www.shopify.com/tools/business-name-generator>

The main reasons why the website domain should match your business name are the users and the search engines. Firstly, users will expect to see your site once they type your company name into the search bar. In case a website with a different name comes up, as a result, they might not consider it to be related. You should be aware of this expectation when making a decision on choosing a business name and registering a website. And secondly, this will also be good for SEO because your business name will be a part of each URL created on your website.

Logo

Depending on the scale of your business and the budget you have, you could hire a professional designer who would design a logo once you have a business name. Apart from a business name, it is helpful to share some keywords and topics that are associated with the brand, the story behind the name, etc. This helps with the process of creating a logo.

If you prefer DIY method, there are tools to help you out:

[Logomakr](#)– The tool offers low quality + CC license with free logo, while you will need to pay for high-resolution photo.

[LogoShuffle](#)– Provide your business name, slogan, and keywords, select the icons, font categories and change colors to get logo mockups. Shuffle the ready-made designs, and once you find the one, you can buy it.

Tailor Brands— You will provide a logo name and a short description after which you will be offered a series of example pairs where you will choose the one you prefer. This will enable the tool to create a logo tailored to you, after which you can purchase the logo.

Online LogoMaker— Add text, choose symbols or upload an image, and get started creating your logo. A free logo includes a low-resolution image, while payment is required to download a full resolution logo.

Other business details can include:

- Social media profiles
- Blogs you can cooperate
- List of competitors
- List of keywords you want to associate with your business

Product information

The next part of the business plan is focused on your products. Start with defining what you are selling and how you are going to sell.

Product to sell

When you start with developing this business plan, you already have in mind the exact product you are selling. However, what you need to define here are the following things:

Produced product or sourced externally

If you are creating products, you will need to provide information about the supplies needed, the way you will obtain supplies and the time required for making those products. On the other hand, if you are going to buy products to sell them in your store, you will need to focus on choosing suppliers. The goal is to get the best product for the best possible price. Planning these activities in advance helps you stay organized and prepare everything to quickly implement the plan.

Digital or physical products

In most cases, you will either sell digital or physical products. However, when it comes to books or music, you can choose which one to sell, a digital format which is downloadable and instantly available, or a physical product. You can always offer both formats.

Place to sell

Since e-commerce is focused on sales that are conducted online, the primary place to sell products is the internet. However, this does not stop you from selling off-line as well. Choosing between a physical or a virtual store is simply a matter of how you want to run your business.

The main difference is that virtual store usually requires less paperwork and administration. This is why it is easier to set up an online store.

The main convenience an online store offers is that it can be accessed anytime anywhere. There are no limitations regarding space which means that the customer base is much wider than when it comes to a physical store. For customers, this means an enormous amount of time saved they would otherwise need to spend looking for items and exploring different shops all across town. Searching for the desired product has never been easier, with the use of search option offered by every virtual store, allowing a quick and easy way to look for an item based on the keywords, product name, etc.

In terms of business owners, a virtual store primarily represents a smaller investment. It is much cheaper to run a virtual shop because there are no rent costs, no utility bills, etc. The only costs involve web hosting and technical support if you need it. With virtual stores, you can do most of the work related to management and handle the sales, which means that the costs of hiring employees can also be avoided.

You can always opt for an option to run both virtual and physical store. Since there is a huge number of consumers who still prefer that old-school method and are in fact willing to pay more if they are able to feel the actual product before buying, you could open a physical store alongside a virtual one. Being aware of additional costs that a physical store can impose is a good start to evaluate if you want to run this kind of business as well.

When we focus on the virtual realm, there are a couple of ways you can sell online. Firstly, you can have your own website. Alternatively, you could join marketplaces and sell through their website. This choice is also something to include in the business plan. If you are planning on using several marketplaces, provide all of this information in the business plan, to help you further along with the planning and promoting products.

Investment

Opening any business requires an investment. Even though running an e-commerce business does offer this advantage of being the one that needs a small investment, you will still need a certain amount of assets to open and start running a virtual store. It is helpful to determine the assets you can invest, and then compare those to the actual costs you will have when opening up an e-commerce business.

To help you with calculating the investment needed, let us focus on the costs you will have when opening this kind of business.

Web domain

The domain is the first thing you will have to purchase when setting up a business. The actual costs may vary depending on the provider, as well as on the type of domain extension you choose (.com, .net, .shop, etc.). The domain is bought yearly, but you can choose between one year, two

year or even five-year period. Domain name is sometimes free with a hosting plan, or it can start from \$1 for a yearly plan (usually for first-time customers only). The regular price is usually between \$10 and \$15 per year. If a domain name is not available, sometimes it might be offered for an auction, which means that the price can be much higher. This is usually what happens with popular domain names.

Hosting

Hosting is an online storage for your website, and it is the place where your website will be built on. There are many hosting providers on the market, offering hosting packages anywhere from \$1 to \$3 per month for first-time customers and basic plans. For e-commerce, you will need a hosting that is flexible and can grow as your business grows. When starting your business, you might be prone to choosing a simpler solution and a plan with some limitations. These plans are usually cheaper, which is great if you are in the phase where you are still testing the market and exploring the potential of your business. However, the plan you choose has to be easy to upgrade as soon as you recognize that your business needs a more powerful solution, in terms of performance, bandwidth, storage, etc. The average hosting package is between \$10 and \$20 per month, depending on the type of plan and hosting provider you choose. More advanced solutions such as a virtual private server, a dedicated server, etc. are more costly and will require greater investment, but surely are something to think about if you are thinking about a big online business.

Technical support

Although running an online store is a more convenient method of commerce in general, it still requires a bit of know-how, especially when it comes to setting up a website, online payments, etc. The costs here can range a lot, depending on how simple or complex the system is, as well as on what kind of help you will need.

You might be able to handle some portion of the tasks, such as setting up and optimizing a website, but you might want to hire a professional to handle sections with online payments (credit card payment, PayPal integration, etc.). In case you are not able to set up a website yourself, the costs here will be greater. Additional costs can include:

- Website optimization expert to improve performance of your website
- Photographer for product photos
- Content writer for product description/blog content
- Administrator for handling management and customer support
- Marketing expert to work on promoting your products

In case you opt for a marketplace instead of your own website, you will obviously avoid a part of these costs, such as the domain and hosting. Still, selling through marketplace usually requires a certain fee, that is often calculated as the percentage deducted from the sales.

Product related costs

This section of the investment can vary hugely from business to business. For starters, you might sell products by acquiring them directly from the manufacturer. In this case, you usually do not have to buy the actual products, but instead, you will simply purchase them once you receive an order.

On the other hand, if you make your own products, you will probably need some kind of materials to make them. These costs should all be included in the calculation of your business investment.

When it comes to offering services, you will think about the following:

- Who will be handling the service (this person's time should be paid accordingly)
- Costs related to providing the service (if you need to drive yourself to a location to provide the service, the costs of transport should be included in the price of the service)

Create a business plan that features all these fields and then fill in the expenses you expect to have. Sometimes it is not possible to predict the exact costs but an approximate value. Then it is always better to calculate a higher value, just in case. Once you have all the values, the sum of those is the investment you will need to start your business.

These three elements make up a core of the business plan. These are the base from which you will form a strategy later on. For example, having determined a business name, you can then work on social media marketing strategy, by creating online profiles with this name. While a lot of information is included in these three parts, there are still more aspects of e-commerce website you should have in mind.

Other things in the business plan

Predicting profit

Each e-commerce has a goal of acquiring profit through sales. This is the primary goal you want to achieve, and we will focus on different methods for achieving this goal later on, but when setting up a business and defining a business plan, it is helpful to try to predict the profit over a given period.

Since each business requires an investment, the first income you acquire from sales will be enough to cover this investment, and it might be a while before you start gaining profit.

Here is how to predict the profit:

- Choose a period you want to focus on
- Predict the number of sales you can expect (based on the market research, previous experience, etc.)
- Multiply the predicted number of sales with the product price
- Predict the recurring purchase trend, and add this to the previous sum
- Subtract the investment needed for the same period from the total predicted revenue

Pricing

Another part of the business plan you will work on is pricing for the products or services that you offer. Since the price essentially should cover the costs needed for that product or service to be provided, plus some profit for you, a good starting point is predicting the costs to make (or buy) an individual product or to provide a particular service. It is how you get an estimate on how much to charge. For example, if the product costs \$10 to make, and you want to earn \$5 from each sale, the product price could be \$15. However, you will also need to consider other costs as well. Even though the cost of website hosting is not directly part of the product cost, it is an expense you will have to pay from the revenue you obtain. Thinking about overall predicted costs might give deeper insights into how to form the product selling price. So, you might want to go up from \$15 to \$18.

The second thing you should consider when determining the pricing is the situation on the market. Make sure you explore the internet and look up businesses that offer the same products or services. Take a look at their pricing, and check out if there is a space to modify your price further. For example, if you explore the same product category and the cheapest product you were able to find is sold for \$25, you might want to increase the price from \$18 to \$23 or even \$25 to match your competitors.

Special offers

With pricing, you determine the regular prices you will feature in your e-commerce. Still, special offers are usually a part of the company strategy, which is why you could plan and predict those offers with the initial business plan. Here are a couple of options:

- First-time offer – Offering a special discount to first-time customers only
- Newsletter discount – Sending out discount codes to your subscribers
- Bundles – When selling product, you might want to provide a special (discounted) price for buying a bundle of products

- Seasonal offers – Holiday seasons are popular times to offer discounts and special offers, so you might want to plan this for Christmas, Easter, etc.

Plan for promotion

So far, the business plan was focused on what to sell, where to sell and how to determine costs and pricing. Finally, the business plan should also include a section on how you are going to promote the products. Since promotion includes an important part of your strategy and it helps with estimating profit, it is very helpful to plan online promotion in advance.

Channels for promotion

Channels for promotion include the platforms you will use to advertise your products and try to reach the online customers. Unlike traditional marketing channels used for reaching customers, e-commerce is focused on the online media, and the usage of the following channels:

- Website
- Third-party websites (gaining inbound links)
- Social media
- Paid advertising

There are numerous opportunities within these channels. For instance, you can advertise products on your site; you could pay to have banners on other sites, etc. Regarding social media, you can choose a format (text post, image, video) and then promote it through different social networks such as Facebook, YouTube, Twitter, etc.

A good thing with planning channels for promotion is that this helps you determine online media you want to focus on. If you plan on using Facebook ads to promote your business, you will need to have a Facebook profile and a Facebook page first. The same goes for all other channels for promotion. Planning also gives you estimates on the budget you will need for this type of promotion. Let us say you plan to use paid advertising with Google AdWords. These campaigns require a budget, and you will be able to plan it within the business plan.

Promotional activities

Besides a direct promotion, which can range from promoted posts to paid video adverts, other promotional activities that are not as direct when it comes to promoting the actual product, but still are efficient at raising awareness of your brand. Those include promotional activities such as:

- Special offers
- Discounts

- Loyalty club
- Contests
- Giveaways
- Collaboration with other brands

With these activities, you award your online followers, regardless if those are your subscribers, your previous customers or perhaps even your social media followers. Your goal is to organize an activity that will increase engagement of those interested in your brand while awarding one (or several) of those based on certain criteria (randomly chosen winner, the one who gets most votes, the most active contributor, etc.)

Despite the fact that the promotion of the actual product is not a direct goal, these activities are still called promotional because they help with:

- Placing your brand on the market
- Raising awareness of what you do
- Increasing interest in your brand
- Engaging those interested in your brand
- Improving the level of loyalty with rewards, benefits, etc.

Since promotional activities improve your online influence, try planning them in advance inside a business plan. The main reason why it is good to prepare them in advance is the fact that you will be able to define the budget for each of these activities. Furthermore, you will be able to plan the exact period when you will be organizing each activity. You do not want to host several of those at the time, nor do you want to overlap with other activities you plan for your business. For example, when introducing a new product in your store, you might not want to offer a discount for that product, but instead, you could host a contest or a giveaway.

To successfully plan promotional activities, think about these:

- Type of the activity you want to organize
- Goals you want to achieve (promotion of the products, introducing new products, engaging previous customers, etc.)
- Period during which the promotion is going to be active
- Terms and conditions for participation
- Cost related to organization

Determining goals

The sale is the most obvious and the most common goal of each e-commerce website. However, this does not have to be the only goal. Here are some example goals you could have in mind:

- Sell 50 products during the first month of running the business
- Increase the number of sales by 10% each month
- Gain 10 recurring customers
- Work with affiliates to increase the number of new customers

These are only some ideas, and the numbers are used just as an example. You should define a goal based on your own business and based on the predictions related to the business success. If possible, always add a quantifier (such as 50, 10%, etc.) to make those goals easily comparable to the results. This also makes goals more specific, allowing you to plan actions that will lead you towards achieving those goals.

Your business plan should contain several goals that you want to achieve. Once you have those goals, try to predict actions that will help you achieve them, as well as the time interval need for that. For example, if you set up a goal to increase the sales for 10% each month, you obviously need to work on promoting your product and reaching new customers. You will plan actions such as social media promotion, search engine ads, organizing a giveaway in collaboration with a blogger or influencer, etc. As you can see, different segments of online marketing will directly affect the ability to achieve e-commerce goals.

Benefits of having a business plan

Having a plan of action for any business is always recommended. Here is why:

Organization

A business plan helps you stay organized. When you have a business plan as a reference, you will always know what your next steps are and what you need to complete preparation. Being organized helps you save time and effort by determining a roadmap you will follow.

Time management

We all know that time is of crucial importance, especially when starting up a business as you will soon become overwhelmed with the number of tasks you will face with. However, if you have a business plan, you will be able to organize your time much more efficiently.

Asset distribution

Since a business plan has the information about the estimated costs, this is an excellent reference when thinking about your available assets and how you are going to distribute those.

Unified approach

Finally, a major benefit you will see with the business plan is the fact that you will recognize activities that will require different departments to work together. For example, collaborating with social media department on a promotional activity can yield great results for your business. Smaller companies might not have departments, but different persons being in charge of those tasks, or even perhaps one person in charge of various sectors. This is simply a matter of organizing and distributing work, but the point is that e-commerce is and should be seen as a part of global business running. Combining the power of sectors such as online marketing brings more benefits for your business by joining the forces to work on achieving the common goal.

How to use a business plan?

Once you finish the plan of activities, once all of the fields are filled in, you will have a very valuable document about your business and its future. All of these ideas gathered together represent a roadmap that will show you the way through many different situations that might shake your confidence and make you doubt your idea.

You will use the plan as a reference. It will be your guide and a resource you will turn to when you want to plan the next action. Check your business plan to explore other options you can grow your business.

And as your business grows, so should the business plan. Apart from that initial information gathered inside the plan, think of this document as something that will evolve and change as your business takes new turns.

Allow the opportunity for this plan to be flexible and to adapt as you change your business strategy based on the market situation. Update the plan with new trends, ideas and practice and how these can reflect your business approach. This way, you can create a document that follows your vision but still keeps the pace with the opportunities your company might be facing as the situation on the market changes.

The purpose of designing a business plan is to provide insights into how you will organize different aspects of your business. It is not necessary to go too deep into the matter and be too elaborate and thorough. For example, rather than planning a promotional activity for Christmas with all the details, a business plan could only acknowledge the fact that you will organize one and perhaps the budget you can allocate to this activity. The idea is only to plan the methods and activities you will use, as well as to predict the budget you will need for conducting those.

A good business plan is supposed to keep you organized and help you stay on track. The descriptions and details about each segment can be quite scarce, but enough for you to make sense of those. This will help you to stay focused and never lose track of what you want to do with your business and what kind of strategy you will use to achieve your goals. It will also help with setting up a time reference because some segments of the business plan are crucial before being able to proceed to the next phase. For example, planning and predicting the time needed to develop a website has an impact on when your online store will be up and running.

4

Legal Guide for E- commerce

4. Legal Guide for E-commerce

Although establishing your online business is an amazing opportunity to grow your business in the digital world, there are some issues you have to be aware before you even set up an online store. The legal aspect of running an e-commerce website helps you define your business and your relationship with the customers. The goal is to enable seamless cooperation and provide documents that regulate any issues that might occur along the way. This makes doing business much easier for you, but it also helps your customers to feel safe and confident about your business.

In the first place, you should have in mind that the laws differ from country to country. To run a successful business online, it is recommended that you explore the laws that are valid in your country and how each of the following issues is resolved based on the applicable laws. It is also helpful to have some parts of regulations available as documents for your customers. Not only do these help protect you and your business, but they will also contribute to establishing a better relationship with the customers. Your company will instantly appear more legitimate and professional, enabling customers to feel secure when buying from you.

Most of the documentation here follows a certain standardized form, which has to state obligations and rights of both parties involved, your business on one side and a customer on the other.

Trademark registration

A trademark is an important part of owning a business because it protects the brand names and logos. It helps you promote your brand and establish a recognizable name on the market. Customers can identify a certain trademark with a certain brand, gaining you credibility and reputation.

Simply choosing a name is not enough because you have to make sure that you:

- Conduct a trademark search – This way you check if anyone is using the same name (regardless if they have registered the name or not). If the name you are planning to use is already a registered trademark, you should avoid possible trademark infringements.

Determine the name confusion – You should evaluate the likelihood of the name becoming recognizable and distinguished on the market. Analyze the industry and competitors to make this decision.

- Decide to register a trademark – If you want to protect your business and your brand, registering a trademark is a recommended option. Your goal is to make this name recognizable and trustworthy among customers, so you do not want to risk anyone taking the name to enjoy the benefits or possibly damage your reputation.

A trademark can be designated by the following symbols:

- TM – "TM" in superscript symbolizes an unregistered trademark, a mark used to promote or brand goods.
- SM – The letters "SM" in superscript stand for an unregistered service mark, a mark used to promote or brand services.
- [®] – The letter "R" surrounded by a circle is used for a registered trademark.

Countries offer formal trademark registration, which is used to protect the brand and the name associated with the brand. You should explore the regulations and the conditions for trademark registration, or you should hire legal help from someone who specializes in the legislation for the country where you do business.

Terms of use

Terms of use, also known as terms of service or terms and conditions, is a document that regulates the rules the users must agree to if they want to use the service. It is used for legal purposes, especially in cases when the personal data of the users are being stored, such as the case with an e-commerce website. Each customer agrees to accept the terms of use that are provided by the site owners. Besides being legally binding for both parties, the terms of use can also be subject to change, which is something that is also highlighted as a part of the document. As a website owner, your responsibility is to make the terms of use available to the users who want to review the file.

This document is often designed based on a certain template, as you will notice that most terms of use have similar sections as the part of the document. Those parts include:

-

Company information

The first part of the document should provide the information about your company, as the purpose of the document is to regulate the relationship between your company and the online user.

- User rights and responsibilities

The second part usually defines the user rights when it comes to the website, products, or services you provide. This section should also focus on user responsibilities and any actions that may be prohibited. Defining a proper way to use the website or service also helps to prevent misuse. This section may also highlight the existence of a separate document that regulates the usage of the personal data (Privacy Policy).

- Additional business information

Sometimes, it might be helpful or necessary to give more information about providing the service, as well as possibilities that could lead to modifying or terminating the services due to a certain reason.

- Disclaimer and limitation of liability

In this section of the terms of use, you clarify the legal liability for the damages incurred by the users. This particular section helps you protect your business from any loss or damage that cannot be foreseen or are caused due to no fault of your own.

- About the terms

The final part is where you state the fact that the terms of use may be subject to change, of which the user may or may not be informed. It is also in this section that you state if the document has been officially registered and in accordance with particular laws (usually those applicable in the country your business is founded). In the case of any legal issues that may occur, it is helpful to state the legal entity that will be in charge of solving those disputes. This is especially important when working with customers on an international level, as it is impossible to be aware of the legal regulations for numerous countries. You can also provide information on how to get in touch, in case the user has a question or an issue related to the document.

The terms of use should regulate all aspects of doing business with customers, including all the phases where an issue might occur. When running an e-commerce business, you should think about the terms related to the following segments:

Terms of payments

- - Terms of delivery
 - Terms of shipping
 - Terms of refunds
 - Terms of use of your website

Each of these segments is an area where issues might occur, which is why it is helpful to have the terms ready to regulate all of the transactions. For example, if there is a problem with the delivery, you, as well as the customer, can always refer to the terms of delivery. In essence, a customer must accept the terms of use before the transaction is initiated, which means that all of those conditions are applicable from that moment onwards.

Refund policy

During an e-commerce transaction, the money is paid upfront, and in return, the product is sent and delivered within the specified timeframe. In this process, it is crucial to realize that refunds are sometimes required. It is a part of doing business online, and having a clear refunds policy on your website allows you to provide a bit of guarantee for the customers. When you define the refunds policy, you determine the conditions under which the product can be returned. In the case of providing a service, it is also possible to issue a refund, but most service providers actually enable trial period instead. The period of 7 to 30 days is commonly used as a test period, during which the customer has the right to change his or her mind.

If the customers are eligible, they have the right to return the product and receive a full or partial refund.

When you provide refunds policy on your website and asking to accept those terms before initiating the transaction, you get customers to read and accept these terms. This way, they are familiar with the conditions that apply when a customer wants to claim a refund, regardless if that claim is due to the product damage, failing to fulfill the expectations, etc.

Handling refunds requests promptly is a way to establish a good relationship with the customers and to gain a good reputation. Take every refund request with seriousness and treat customers fairly. If they are eligible for a refund based on the conditions of the refunds policy, you are then obliged to issue a refund. Sometimes a customer might ask for a refund even though conditions are not fulfilled for that refund to be issued. For instance, a customer might want to return the product after three months when the refunds policy states that the product may be returned within 14 weeks. In these situations, you could always refer the customer to the refunds policy document.

Regulations defining refunds are usually part of the terms of use, but they are a significant segment of the document. You should have this legal aspect in mind when creating the terms of use for your e-commerce website.

Privacy policy

Another legal document, binding for you as a product/service provider and a customer, is used to define terms of gathering and managing customers' data. Privacy policy is used as an official statement to protect customers' privacy and any personal information gathered on the website.

Personal information can be anything that defines a user, such as a name, address, the date of birth, contact information and even credit card information. These data are often provided by the users themselves when they fill in a form, but they can also be gathered through the use of third-party integrations, such as the usage of the cookies.

The important thing is to understand that as a business you can store and manage the data provided by the users, but you need to provide the information on how these data will be used. For example, the customers have the right to know if the data is kept confidential or shared with your partners, and this is something you should state in the privacy policy document.

Privacy policy should be based on the applicable law, which is why the geographical aspect and legal jurisdiction have an important role when creating the document. The content shared through the document should thus be in accordance with the applicable regulations in the county where your business is set up. Have in mind that personal data protection is a very hot topic in today's reality when the user data have a great value for businesses and establishing the relationship with the customers.

Besides the data protection, where you agree not to disclose personal information about the users who give you the right to use the data for your own business, there is another segment related to privacy, and that is anti-spam. Since email marketing is a very powerful method for reaching the customers and promoting your business, it is essential that you follow anti-spam regulations and be extra careful when using the user information to contact the customers.

Typically, a privacy policy document can include:

- The kind of information you collect

This information is required by the customer, and it is usually the name, the email address, etc.

- How you collect the information

You should inform the customers how you collect and store the data. You could also mention the usage of cookies, in which case some data may be left on the user's computer to track the habits and personalize the online experience.

- What you will do with the information

The customers have the right to know in what kind of purposes will the data be used, for example, for contacting the customers in the future, sending promotional announcements, etc. This section should also state whether you plan on sharing or even selling the information about the customers.

- Reviewing and deleting information

The customers should have insights into the information collected about them, as well as the instructions on how to change or delete them. The customer can send a request for his or her data to be removed from your database in which case you have to act upon this request.

- Time period for storing information

Information is often stored much later after the transaction is over. This is something you should inform the customers about.

- About privacy policy

At the end of the document, you will provide information about the effective date of the privacy policy and any further information about your business. Since the document might be changed over time, you could state this information here, or you could even provide information about previous updates of the document.

When you are selling online, it is essential to be aware of all the regulations and laws that are applicable in the country where you do business. Not only is it going to be helpful in the long run and assist with successfully running a business, but being familiar with the valid regulations will also spare you any issues and disputes that might occur with customers.

The documents that regulate the usage of your e-commerce website should be accurate, current, valid and concise. The customers should find them easily on your website and review them before deciding to do business with you. These documents keep your business credible in the mind of the customers while building trust in your brand.

Due to the legal character of these documents, it is recommended that you seek legal advice on the matter in the particular country. To help you with some general guidelines typical for some countries, you could refer to this online resource: [A Legal Guide to Ecommerce](#). You will find general laws and regulations valid in Australia, Canada, Singapore, United States of America, United Kingdom and New Zealand.

The screenshot shows the Shopify website's navigation bar at the top, featuring the Shopify logo, 'WAYS TO SELL', 'PRICING', 'BLOG', 'RESOURCES', and 'Help Center'. Below the navigation is a large dark blue header section for the 'A Legal Guide to Ecommerce'. On the left side of this header is a light blue graphic of a building facade with five small international flags (Canada, USA, Australia, Turkey, and the UK) above the windows. To the right of the graphic, the title 'A Legal Guide to Ecommerce' is displayed in white, bold, sans-serif font. Below the title, the text '6 chapters' is shown in a smaller white font. Further down, a descriptive paragraph in white text reads: 'An online shop is an exciting way to grow your business and this guide will teach you how to maximize the potential of your brand and comply with laws applicable to you.' At the bottom right of the header, there are social media sharing icons for email, Pinterest, Facebook, Twitter, and LinkedIn.

5

**Having an E-commerce
Website**

5. Having an E-commerce Website

When you decided to explore the possibilities of an e-commerce business, you will start with thinking about how you are going to make your products or services available to the internet users. Joining the market places is certainly one option you could explore, but first, we will focus on having your own website with an online store. For big businesses, having their own ecommerce website is a crucial step. Alternatives do exist and offer plenty of exciting opportunities, but having your website helps you with creating a compelling presentation, a brand and an image that you want your business to be identified with.

Regarding having an e-commerce website, you will need to explore different solutions from website hosting to e-commerce integration, shopping cart, and payment gateways.

Website

First things first, you will need to create a website. The most common choice is choosing one of the CMS (content management system) platforms. Platforms such as WordPress, Joomla, and Drupal, host a large percentage of websites. All of these platforms are free. They are mainly popular because they are quite simple and straightforward. You do not need any coding experience because all the features are already built-in or available as plugins and extensions. Other things that can be very helpful is provided documentation, resources, as well as the community of those using the same platform to exchange experiences. Furthermore, CMS platforms are usually up-to-date with the current industry standards, which means that they are SEO and mobile-friendly.

In e-commerce business, the website is your place to sell, which makes it the most important part of your e-commerce business. It is the place where your customers will get to learn about your products, to interact with you, get feedback and eventually buy. Therefore, your website should be:

- Engaging for the users to start interaction
- Helpful for them to quickly find information
- Responsive for both desktop and mobile users
- Effective to persuade visitors to buy

Your website should have:

- Homepage
- About us page

- Terms and conditions
- Privacy policy
- Contact us page
- Blog
- Shop

All of the pages, apart from the shop, are created on the actual platform you choose. When creating each of them, think about best SEO practices and how to optimize each page to make it SEO-friendly.

Store

When it comes to the actual store, this is the e-commerce section of the website which requires e-commerce platform integration. Choosing one over the other platform is a matter of preferences, business needs, and budget. In the following chapter, you will find suggestions on the most popular e-commerce platforms at the moment and what kind of features you can expect from each of them.

To make the store fully functional, you will have to integrate specific features that enable the online buying process. Sometimes, these features are already available as a part of the platform itself, but some platforms will require additional installations and integrations to make everything work.

Product information

Product information has a huge impact on the performance of your website, on your ability to engage the visitors, to convert them into customers and eventually complete a sale. The way you present the product can make or break your business success.

Product name

Choose a descriptive product name that the customers can easily find and relate to the actual product. Avoid using too many random capital letters or numbers. Although this might be helpful with tagging product for your purposes, the customers can only find it confusing.

Product image

Images are necessary for an e-commerce website. It is enough for customers that they are unable to feel the actual product, but not seeing it as well means that they will probably be reluctant to buy at all. Product images should be very vivid and depict the features correctly. Ecommerce websites usually provide a couple of images for a product (for example, product in a box, unboxing, product in use, etc.).

Product description

The description should provide more information about the product or service. It is a piece of text that provides more details, highlights particular features or explains how the product is applied, used, etc. The purpose of this text is for the viewer to get more familiar with the product features and how buying this product can be useful and practical. The length of the text should be optimal because you do not want to overwhelm the customers with irrelevant text. Since this text is also helpful in terms of SEO, it can be quite useful to have more content in the product description. After all, this is the segment based on which the search engines are going to index and rank the page. However, always have in mind that the page (and its content) should be user-friendly. If the content is completely redundant and repetitive, you should not add it just for the sake of the search engines because this can have a negative influence on user experience.

The screenshot shows a product page for an Adidas Cloudfoam Groove Mens Trainers. At the top, there's a navigation bar with a search bar, categories like MENS, LADIES, KIDS, FOOTBALL SHIRTS, ACCESSORIES, SPORTS, BRANDS, and VIEW ALL DEALS, and links for WISHLIST and BAG IS EMPTY. Below the navigation, the breadcrumb trail shows the user is at Home / adidas / View All adidas / Cloudfoam Groove Mens Trainers. A large blue and white shoe image is the central focus. To the left, there's a sidebar with a 'BIG BRAND SALE' banner showing three smaller shoe images. To the right, the product details are listed: 'adidas Cloudfoam Groove Mens Trainers' at '50,40 € 71,99 €', color 'Blue/Blue/Whit', and a size guide. The main product image has a '360 View' and 'Zoom' option. Below the price, there are size and quantity selection boxes, and a prominent green 'Add to bag' button. A link to 'Sign in to add to wishlist' is also present. At the bottom, there's a 'Find similar items here:' section and a detailed 'Product Info' box containing a bulleted list of product features.

Image: <http://rs.sportsdirect.com/adidas-cloudfoam-groove-trainers123023?colcode=12302318>

Additionally, it is recommended not to use the same product description for the multiple products. Despite the fact that it might be easier, and sometimes logical to do so (for example, you sell several types of running shoes with similar properties and design), you should avoid this practice. Firstly, because of your customers, but more importantly, because of the search engines. Search engine crawlers will use your website content (including product description) to understand how to index it and when to show such content as a response to a user query. Using repetitive product description will make this job more difficult for them.

Buttons and fields

Besides the actual products, you will probably use additional clickable buttons and fields to make the purchase process simpler. Some of the common items you will have to include are the following:

- Product size
- Product length
- Quantity
- Color
- Size guide
- Store locator
- Add to bag

FOSSIL

[WOMEN](#) [MEN](#) [WATCHES](#) [BAGS](#) [WALLETS](#) [JEWELRY](#) [SMARTWATCHES](#) [GIFTS](#) [SALE](#) [OUTLET](#)

Home > Neely Three-Hand Luggage Leather Watch



NEELY THREE-HAND LUGGAGE LEATHER WATCH

ES4255P
★★★★★ 4.0 (2) [Read Reviews](#) [Write a Review](#)

\$105.00

7 COLORS



[ADD TO BAG](#) [ENGRAVE ME - FREE](#)

[Check Store Availability](#)
[View Shipping and Return information](#)

PRODUCT DETAILS

Collection: Neely
Movement Type: Quartz
Case Size: 34mm
Strap Material: Leather
Water Resistant: 3 ATM

Image: <https://www.fossil.com/us/en/products/neely-three-hand-luggage-leather-watch-skues4255p.html>

Those and similar information can be provided about the product if you believe that this information would benefit the customers and provide a better user experience.

Finally, the button that directly leads the customer to the next step, to the shopping cart, is the most prominent one. This is the reason why it is usually in a contrasting color. It is an action button that invites the customer to complete the shopping, usually using the verbs such as *buy, add to cart, order, etc.*

Order information

Order information is another field that can be used to improve the customer experience. With this option, you enable customers to provide additional information about the order, as well as to monitor the status of their order. These features are commonly used as integration with the e-commerce platform.

- Order notes which can be customized – This way the users write comments or specify any information related to their order.
- Tracking the order – An additional integration can enable customers to track their order status, from the moment they have paid for until the item is delivered.

Shopping cart

A shopping cart is a software that allows the internet users to browse and eventually buy the product by putting it in a virtual basket. Adding a product to cart means that the customer is interested in buying that product, but he or she will continue purchase once they are finished with browsing the website. Besides storing product information, shopping cart also represents a direct link to the checkout process, where the user actually starts the process of purchasing.

There are two types of shopping carts you can use for the website:

Hosted shopping cart

This cart is provided by the third-party company that provides the solution that is fully hosted on their own servers. They are in charge of system maintenance as well as upgrades. While these might be advantages especially desirable by beginners in e-commerce, a hosted shopping cart involves customers being directed to another domain for payment processing. This is a drawback since changing domain might create a bit of distrust among the customers. Still, this solution is a recommended option for beginners because it is cheaper than the second solution and it demands less time for management.

Licensed shopping cart

On the other hand, having a licensed shopping cart allows more flexibility and customization options, allowing you to tailor the shopping cart based on your needs. Although this is a great plus, a licensed shopping cart comes with higher costs, and it usually requires more technical knowledge for implementing and fixing potential issues.

Shopping cart abandonment

As shocking as it might be, but over 69% of customers will put something in a cart on your website and leave without purchasing ([Source](#)). It is an extraordinary percentage of lost sales, and this evidently has an impact on your business. Just imagine how your revenue would spike if you could actually retain these customers.

The most common reason for abandoning the shopping cart is the extra costs. In fact, over 60% of customers have abandoned the shopping cart due to costs such as shipping cost, tax, fee, etc. More than one third of customers (35%) say that they have abandoned the cart because they were required to create an account. This is an extra step for them, and if they were in a rush to complete the purchase, this would only slow them down, especially if they were on their mobile device. Too long or complicated checkout process, inability to calculate total order cost up-front or website errors are other common reasons for the customers to abandon their carts.

Reasons for abandonments during checkout

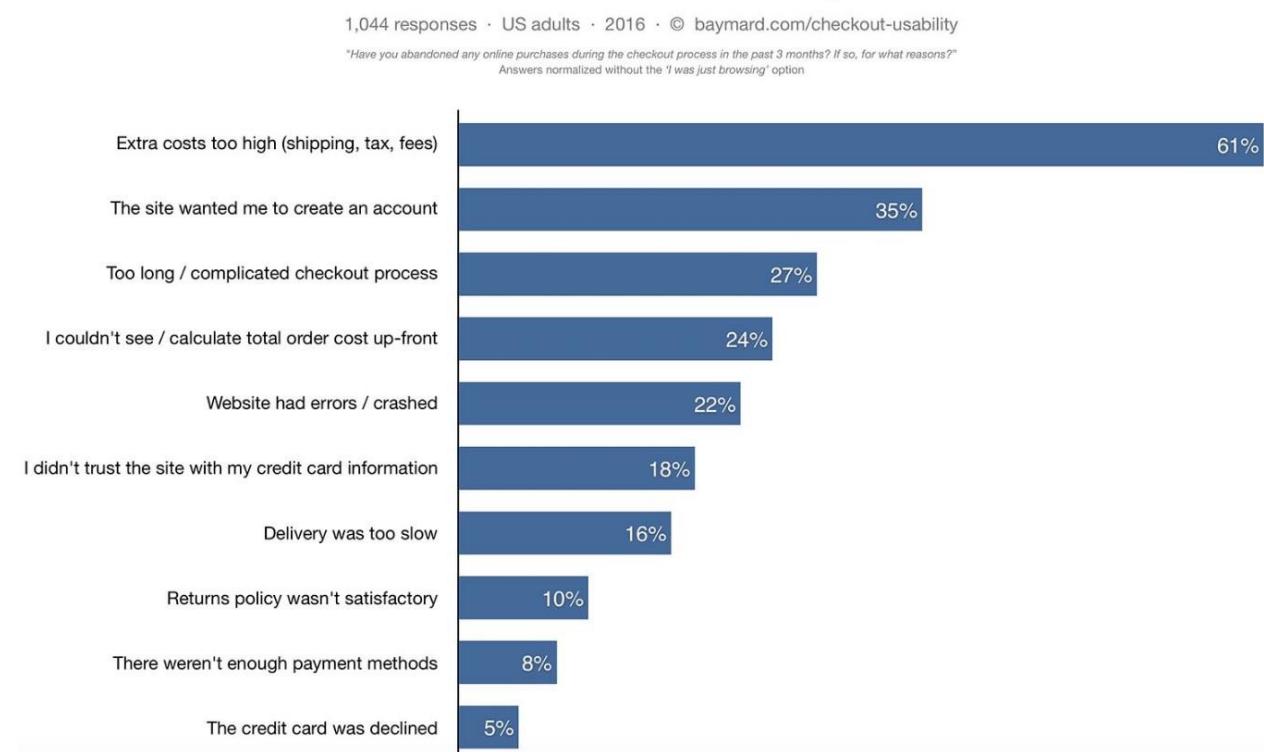


Image: Reasons for abandonments during checkout <https://baymard.com/lists/cart-abandonment-rate>

Being aware of all the issues that can be causing shopping cart abandonment helps you optimize your website and the buying process in a way that you maximize the percentage of completed purchases. For starters, allow users to buy without registering an account. Instead, enable the customers to process through the checkout with their email address. Not only are you going to save them time, but you are also going to obtain their email address early on before they go on to the checkout, where they might still abandon the cart for some other reason.

FEEL UNIQUE

The screenshot shows a sign-in page for the website <http://eu.feelunique.com/>. At the top, there is a horizontal progress bar with four steps: 1. Sign In (highlighted with a green circle), 2. Order Summary, 3. Secure Payment, and 4. Order Complete. Below the progress bar, the word "Sign In" is displayed in a large, bold font. The main form area contains fields for "EMAIL ADDRESS" and "PASSWORD", each with a password input field and a "Forgot your password?" link. There are also two radio button options: "New to feelunique.com" (unchecked) and "I am a returning customer" (checked). At the bottom of the form is a "Sign In" button with a lock icon. Below the form, there is a section titled "Or, you can Sign-in with" followed by icons for Facebook, Google+, Twitter, Instagram, and Pinterest.

Image: <http://eu.feelunique.com/>

The shopping cart page contains the following information:

- Product list

All of the items that have been put in the cart are visible here. The list can be edited, so the customer can increase the quantity of the products or remove the products from the cart.

- Price

Next to each product, there is a price, and the page will also contain the total price. Apart from the actual product price, the price can include taxes, fees or shipping costs.

- Discount code

Some e-commerce businesses offer discount codes, and you can provide them on this page. The price will automatically be recalculated.

The screenshot shows the FEEL UNIQUE website's shopping cart page. At the top, there's a navigation bar with categories like Women, Men, Brands, Offers, New, The Lounge, Summer Sale, and Summer Shop. Below the navigation is a search bar with placeholder text "I'm looking for..." and a "Search" button. A promotional banner at the top features icons for free delivery, beauty favourites, free samples, and official retailer status. The main content area is titled "My Shopping Bag". It lists a single item: "Armani Code for Men Eau De Toilette Spray 75ml" with a quantity of 1, price of £52.80, and a total of £52.80. There are "Remove" and "Edit" buttons next to the item. Below the item, there's a section for entering a promotion code with an "Apply" button and a help icon. To the right, there's a summary table with Subtotal (£52.80), Delivery (£5.00), and a highlighted Total of £57.80. At the bottom of the bag section are "Keep Shopping" and "Continue Checkout" buttons. To the right of the bag, there's a sidebar for "COMPLIMENTARY BEAUTY SAMPLES" showing two sample products with "ADD" buttons. Further down is a "LAST MINUTE TREAT" section featuring "Bardou Dry Shampoo" at a price of "only £15.00" with an "ADD TO BAG" button.

Image: <http://eu.feelunique.com/>

Shipping

During this phase, the customer will also learn about shipping costs and shipping methods that are available. Since shipping costs are the most common reason for shopping cart abandonment, you could think about offering free shipping. A commonly used method is to provide free shipping on purchases that are over a certain limit. For example, if the customers buy products in the total

worth of over \$50, they are eligible for free shipping. Not only do you provide a special perk for your customers, but this practice also encourages customers to spend more in your store. Have in mind that free shipping is free for the customer, but your business will have to cover the costs of shipping which depend on the packaging, carrier, insurance, tracking service, etc. All of these influence the shipping costs, and you have to calculate whether you are able to offer free shipping to your customers.

Shipping is another major distinction between an offline and online store. Customers are not able to see the products, nor to collect them, but instead, the seller is responsible for delivering the product to the customers. However, the situation is far from simple. In fact, shipping can become quite complicated for e-commerce businesses which is why you have to come up with a plan and strategy on how to handle shipping.

There are several shipping models you can use in your e-commerce business. Choosing which one of these to use is based on the type and size of the business you run.

Shipping yourself

The first option is the situation where you send products from home. You are in charge of writing labels and packaging the product. When it comes to sending the packages, you can walk to the post office yourself and send the package or you can have a courier pick up the packages from you and deliver them to the customers.

Fulfillment warehouse

This option refers to using a warehouse to accept and process all the orders. You have the full control of the inventory, and you are able to create a much more consistent shipping strategy. This allows delivering products more quickly and more efficiently.

Both of these situations are shipping models where you send the actual product. However, there is a third option which is called dropshipping.

Dropshipping

This model is a business model where you do not keep the product in stock. Instead of doing so, you collaborate with the suppliers who actually send the product themselves on your behalf, with your own packaging and brand. As a seller, you purchase the product, but you never actually handle it, because the supplier will be responsible for fulfilling orders and delivering the products to the customers. The benefits of this model are lower investment, flexible location and a wide selection of goods. However, it can be more difficult to calculate shipping costs due to a large number of suppliers, and it also might lead to inventory issues and delivery errors.

Instead of offering free shipping, you can use carrier services, in which case you would provide this as an optional shipping plan. When you connect your checkout page with services such as FedEx, you can offer real-time rates to your customers. Customers are also presented with the

time needed for the delivery using each of the services, as well as the price for each service, which is then added to the total price in the shopping cart.

The screenshot shows the Sephora website's shipping information page. At the top, there is a black header bar with "Free 3-Day Shipping over \$50" on the left, and "Track Order", "Find a Store", and "US" on the right. Below the header is the Sephora logo and a search bar. To the right of the logo are links for "Hi, Beautiful", "Sign In or Register", a heart icon, and a shopping bag icon with a red notification dot. A navigation bar below the header includes "SHOP CATEGORIES", "NEW", "BRANDS", "GIFTS", "COMMUNITY", "HOW-TOS", and "STORES & SERVICES". Under "SHOP CATEGORIES", there are links for "Customer Service Help", "Shopping Sephora.com", "Orders & Returns", and "Shipping Information (US)". The main content area is titled "SHIPPING INFORMATION (US)" and states: "We offer FREE Standard 3 Day Shipping on all U.S. merchandise orders \$50 and over (excluding taxes)." It then lists shipping options with their respective costs:

Standard 3 Day Shipping (Orders \$50 and over)	Standard 3 Day Shipping (Orders under \$50)	FLASH 2 Day Shipping*	2 Day Shipping	1 Day Shipping
FREE	\$5.95	FREE (after enrollment)	\$10.95	\$16.95

Below the table, it says "Need it now? Shop eGift Cards." and features a graphic for "FLASH 2-DAY SHIPPING" with a truck icon and the text "FREE shipping for one year. No minimums. Just \$10." The graphic has a red and white striped border.

Image: <http://www.sephora.com/shipping-information>

Another alternative when charging shipping is to offer flat rates. Defining flat rates means that you determine the shipping price that will be applicable for all the products on your website. This price can sometimes be over or under the actual shipping price, but it is a great way to predefine the shipping strategy and make sure you charge all the customers the same. Additionally, when forming flat rates, you can define different rates for certain package weight. This means that there would be one price for all the packages up to 1 kilogram, then you would change the price for the packages up to 2 kilograms, and so on.

Payment gateway

The buying process ends with the payment gateway. A payment gateway is a service that processes payment for an online business. It is integrated into your website as an additional software. This software facilitates the transaction by transferring the information from the customer to the bank. This service basically handles and authorizes the payment between the seller and the buyer. Once the checkout process is complete, and the customer is ready to buy, the payment gateway is activated which is where the customer will make the transaction safely.

PAYMENT METHODS

Bank card (VISA, MasterCard, AMEX, Discover, DinersClub, JCB)

PayPal

IN Discount

* Name on card *

* Expiration date

*Security code

MM * YY *

CVV * What is it?

The process goes like this. Web browser encrypts the data which is sent to the payment processor used by the bank that handles the payment. The transaction data is sent to the card association to the bank issuing that card to authorize the transaction. Once this authorization is received, the payment can be processed, and the approved response is generated. This process actually happens in a few seconds.

Commonly used payment gateways used by e-commerce businesses include:

- [PayPal](#)
- [Stripe](#)
- [2Checkout](#)

The benefit of using such software is the that the process filters out possible frauds. This provides an additional level of security to the customers. Online payment processing can be a little intimidating for them, especially if it is the first time they are buying from you. The customers can be scared to share sensitive information such as the credit card number with someone they are not familiar with. Using these services can increase their trust because these services are trusted payment gateways in the online world, and the customers are willing to share their data this way. In fact, they most probably have done so already through previous payments. In this case, your website does not actually collect any data related to payment, such as the credit card number because the service will do this process for you. An additional perk of these services is that they can calculate taxes which are applicable in particular countries and they also convert currencies based on the valid rates.

These services usually offer the service for free, and you pay based on the number of transactions. This fee can vary, and it depends on a number of factors, such as the number of monthly transactions, the membership plan you are using, your location, etc. To calculate these costs and

to add them to your business plan as a fixed expense that you will have, it is recommended first to choose the service and then look into the estimated rates for your business.

Website security

The final part of designing an e-commerce website is website security. Since you expect the customers to buy from you, during which they will go through the payment process and payment gateway, you need to think about security on your website.

PCI standards

Firstly, there is something called Payment Card Industry (PCI) Security Standards, which is how industry standards are defined to safeguard payment data before, during, and after the purchase. These standards have to be followed on your server, in the shopping cart, and throughout the payment gateway process. Since you are using third-party services for all these aspects of your business, you have this part of website security covered because those services are already PCI compliant.

SSL certificate

SSL (Secure Sockets Layer) is the security standard that establishes an encrypted link between a web server and a browser. This means that all the data that passes between these two is encrypted and it will remain private and integral. This protocol is used by HTTPS pages to encrypt communication and keep the data protected, which is why it is recommended to have SSL certificate activated on your e-commerce website. Furthermore, customers are more likely to trust HTTPS websites which will encourage them to complete their purchase.

6

Choosing an E-
commerce Platform

6. Choosing an E-commerce Platform

Once you have finished planning, the time comes to start working on bringing your e-commerce website to life. You have a vision of what you want to do and you have your goals in mind. Now it is time to realize this vision by creating a store that perfectly reflects your idea.

To begin with, you will consider using one of the e-commerce platforms as a website integration that will enable you to take advantage of numerous features to set up and manage an online store.

Shopify

[Shopify](#) currently hosts over 400,000 active online shops. The platform is constantly evolving with new apps and extensions, growing in popularity, especially among beginners. The platform offers:

- Customization of the store with 100+ templates and 1500+ apps
- Unlimited bandwidth, products, and inventory data
- Manual order creation
- Discounts codes and gift cards
- Website and blog
- Tracking sales and grow trends
- Free SSL certificate
- 24/7 customer support
- Free 14-day trial

There are three pricing plans, starting from Basic Shopify, available at \$29 per month. For a growing business, you might want to consider Shopify plan (\$79 per month) or even Advanced Shopify (\$299 per month) which includes more advanced features, such as reports. Shopify Plus is an enterprise-grade solution but to find out more, you will have to send an inquiry. Finally, if you want to sell on Facebook exclusively, you might want to explore options offered by Shopify Lite, available at \$9 per month.

Ability to connect with a Facebook page and sell on this social network, as well as a mobilefriendly shopping cart, are the features that certainly make Shopify a platform suitable for reaching online users. The process of using the platforms is also pretty straightforward with simple design and integration of 70 external payment gateways, making accepting payments easy. SEO tools and secure hosting are also benefits offered by the platform. The main drawback is the fact that running the store might become a bit costly, with additional extensions which are paid separately,

or when you do not use Shopify Payment, in which case additional transaction fee is applied for every sale.

The image is a screenshot of the Shopify website's homepage. At the top, there is a navigation bar with the Shopify logo, links for 'WAYS TO SELL', 'PRICING', 'BLOGS', and 'RESOURCES', and buttons for 'Help Center', 'Log in', and 'Get started'. Below the navigation, the main headline reads 'The ecommerce platform made for you'. A sub-headline states 'Whether you sell online, on social media, in store, or out of the trunk of your car, Shopify has you covered.' There is a text input field for 'Enter your email address' and a 'Get started' button. Below the input field, a small note says 'Try Shopify free for 14 days. No risk, and no credit card required.' The main visual features a woman in a white top holding a tablet, standing next to a peach-colored dress hanging on a hanger. To her left is a potted plant. In the bottom left corner of the image, there is a caption: 'Jaya Kahlon of Smoke & Ash uses Shopify to sell online and sell in person'. There are also navigation arrows at the bottom right of the image.

The ecommerce platform made for you

Whether you sell online, on social media, in store, or out of the trunk of your car, Shopify has you covered.

Enter your email address

Get started

Try Shopify free for 14 days. No risk, and no credit card required.

Jaya Kahlon of Smoke & Ash uses Shopify to sell online and sell in person

Image: <https://www.shopify.com/>

BigCommerce

[BigCommerce](#) offers a variety of solutions suitable for businesses of all sizes. The platform allows creating stores with engaging features to help increase sales. In fact, the solution is perfect for those who are not much tech savvy and are reluctant to tamper with the code themselves. Highly customizable platform with a great selection of e-commerce templates, reliable hosting, and seamless migration process are some of the features that make BigCommerce a desirable solution for an e-commerce business. Other benefits of using this software include:

- Single-page checkout
- Coupons, discounts, and gift cards
- Professional reporting tools
- Product rating and reviews
- Integration with Facebook, eBay and Amazon, Pinterest, Google Shopping
- No transaction fees

- Unlimited products, file storage, and bandwidth
- Unlimited staff accounts

There are not many plugins because the platform itself is very customizable and already provides a set of integrations to help you optimize and boost the performance of the online store. The plans start with the Standard plan (available at \$29.95 per month), the most popular Plus plan (\$79.95 per month) and Pro plan (\$249.95 per month). Finally, there is the most advanced solution called Enterprise, but to get more details, you will need to get in touch with the customer support. There is a 15-day free trial period.

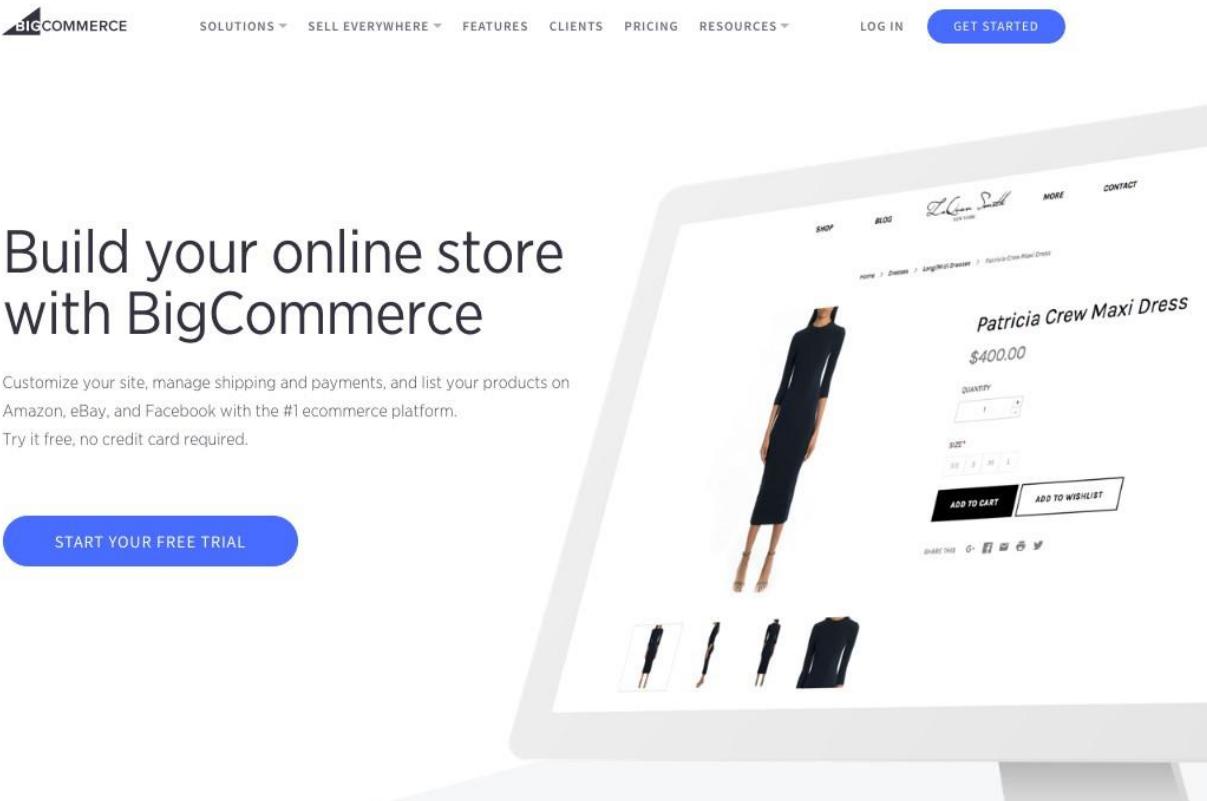


Image: <https://www.bigcommerce.com/>

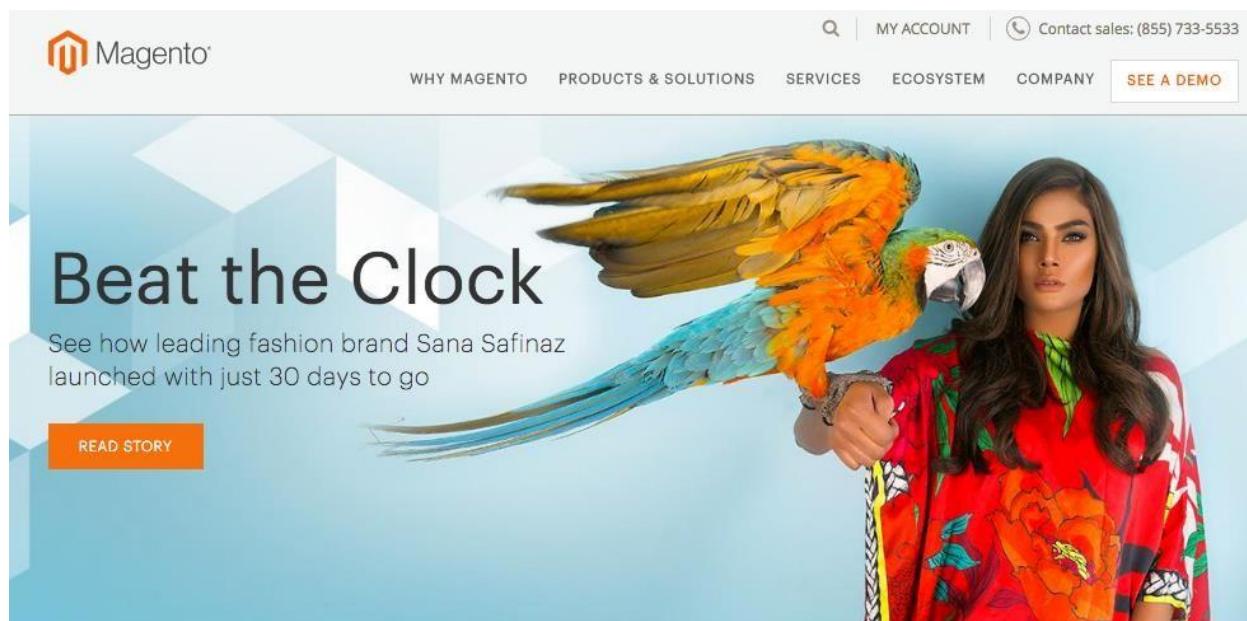
Magento

Another established name in the world of e-commerce is [Magento](#), offering solutions for small businesses, as well as advanced technology and reliability recognized by some of the top players on the market, including Coca Cola, Charlotte Tilbury, Burger King, etc. Over 250,000 merchants are using the platform to grow their businesses and increase sales. Some of the available features the software comes with are:

- Flexibility to customize the platform to provide branded experience
- Open source platform

- Thousands of plugins and extensions
- Extensive documentation and resources
- Tools to attract and retain customers
- Product bundling, gift with purchase, and customer reviews
- Free Magento social integration to catalog products on social media and create advertisements

For beginners, Magento offers a free Magento Community Edition platform. It is a good way to experiment with the features and learn how to integrate and grow your e-commerce business with this platform. There is no clear pricing on the website regarding solutions for growing businesses and large enterprises. The best way to find out more is to schedule a free product demo.



Reach for the Sky

Sell on mobile. Sell in store. Sell on a plane.
Magento powers amazing omnichannel experiences, wherever your customer is.

Image: <https://magento.com/>

WooCommerce

[WooCommerce](#) is an ideal e-commerce solution if your blog is hosted on WordPress platform. The software is specifically designed with WordPress in mind, providing hundreds of free and paid extensions to help you fully customize the store based on your requirements. The software seamlessly integrates with WordPress, and it offers easy migration from other platforms. Features provided by WooCommerce include:

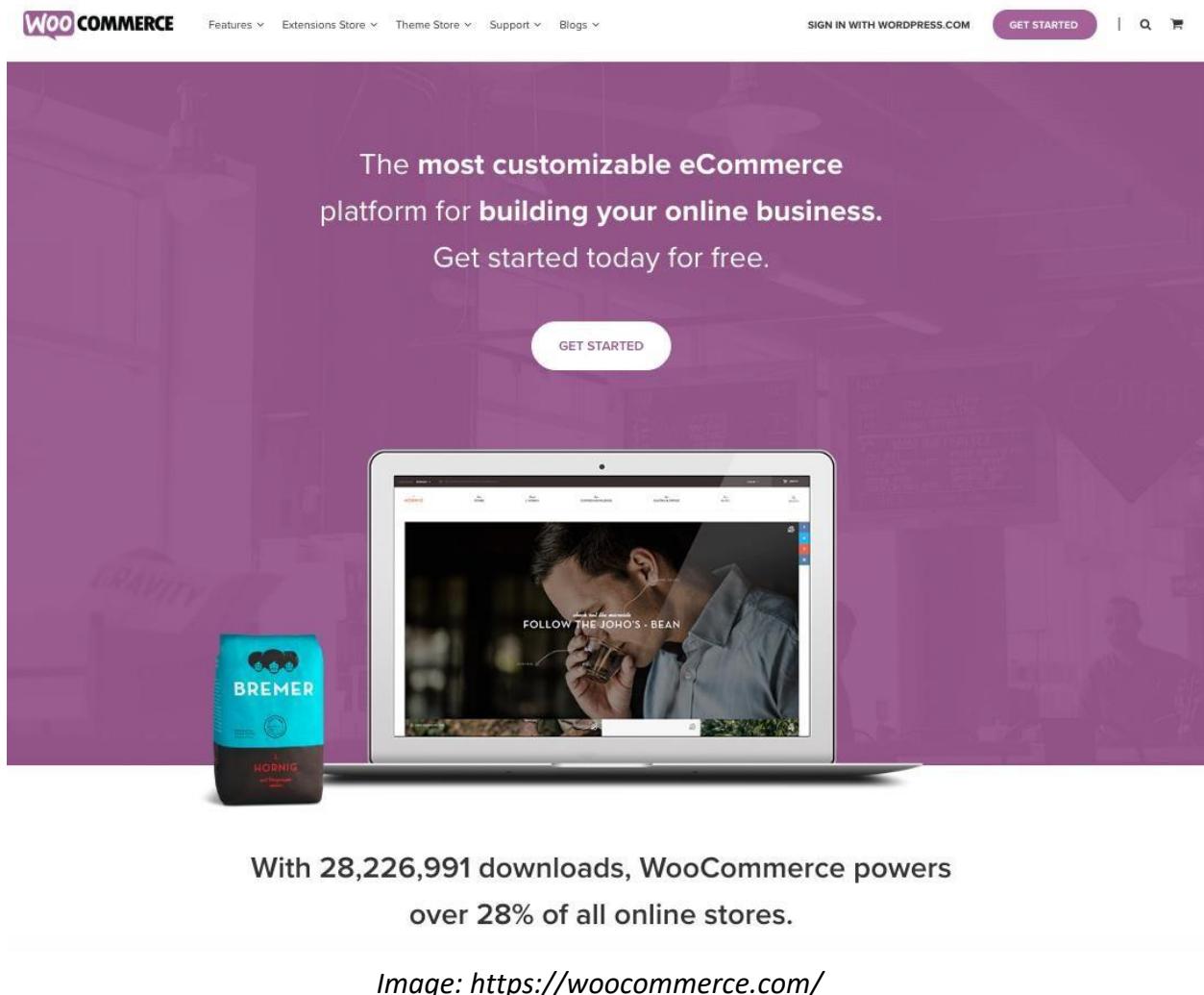
- Selling both physical and digital products worldwide
- Mobile-friendly platform
- Marketplace with free and paid extensions
- Comprehensive documentations related to setup, extending, theming, etc.
- Unlimited number of products
- Secure code
- Pre-installed payment gateways
- Geo-location support making shipping and tax calculation simpler
- SEO-friendly platform
- Discount coupons and codes
- Built-in analytics + 3rd party integrations

WooCommerce offers a self-hosted solution, which means you retain full ownership of the information and data shared through your e-commerce store. There are no transaction fees, and you will have support available for all the paid products provided by WooCommerce. Getting started is free. You will log in to your WordPress account and go from there. To get the most out of your store, you will probably need to use some of the extensions, such as:

- WooCommerce Subscriptions – If you are going to use recurring billing to your membership website(from \$199)
- Products Add-Ons – To create additional options for products, including select boxes, text areas, custom price input, etc.(from \$49)
- WooCommerce Bookings – It enables appointment bookings on your own website (from \$249)
- Stripe – It allows accepting different credit cards directly in your store (free)

- Product bundles – It is used to create customizable offers and product bundles (from \$49)

The purpose of these extensions is to provide advanced features and functionalities for running an e-commerce business, which is why it is worth exploring the features and how those can help with optimizing and improving your store and customer experience.



Volusion

Besides an e-commerce platform, [Volusion](#) is an all-in one solution with options to grow your business with built-in SEO management, newsletters, and CRM system. This provides an interesting option for hosting an e-commerce website even if you are only starting out and exploring the market. You can upgrade at any moment enabling you to unlock more features you might need as your business grows.

All plans include features such as:

- Free, responsive themes
- Securely accepting payments
- Built-in SEO tools
- Social media integrations and reporting tools
- Expert support plus helpful resources
- Inventory management

There are four plans, starting from Mini (available at \$15 per month) with a limit of 100 products and 1GB of bandwidth. The Plus plan is a more advanced (priced at \$35 per month) with 1000 products and 3GB of bandwidth. In case you need a solution that goes beyond these limitations, you can choose Pro plan (the price is \$75 per month) with 10,000 products, 10 GB of bandwidth and priority support, or Premium plan (for \$135 per month) with unlimited products, 35GB of bandwidth and dedicated account manager. A free trial is available during the 14-day period. Advanced features you can unlock with more expensive plans include:

- Rating and reviews
- Newsletters
- Abandoned cart reports
- eBay and Amazon integration
- API access + batch order processing
- Deal of the day + customer loyalty plan

With comprehensive website builder and responsive themes, the platform allows creating highquality e-commerce websites with no coding experience. You will have full control of managing the content, with customization options to help you explore the benefits of your content. It is quite easy to connect the platform with the most popular tools for an e-commerce business that provides additional functionalities. Finally, the built-in SEO management and reporting features are powerful options to explore the influence of your website and your content, which helps with running your online business.

Everything You Need to Open a Successful Ecommerce Store

Shoppers spent more than \$26 billion and placed over 185 million orders on Volusion ecommerce websites.

GET STARTED

👉 Try it free for 14 days. No credit card required. Cancel anytime.

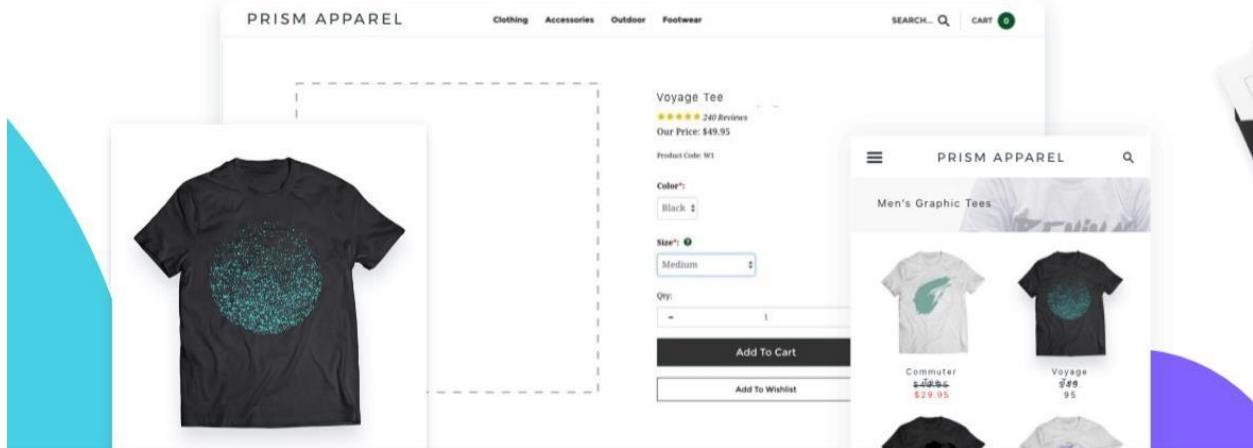


Image: <https://www.volusion.com/>

You will notice a lot of similarities between platforms we have mentioned here, and chances are you will find more platforms that provide a solution for running an e-commerce website. The choice should be based on the needs of your company and the budget you can provide for the platform integration. When it comes to choosing different plans, try using the solution that is enough for your current needs, but always make sure that there is a possibility to upgrade to a more compressive solution once you need it.

7

How to Promote Your Business

7. How to Promote Your Business

Having gone through all of the work of establishing your e-commerce business online, you will probably be eager to see the results. Your store is up and running, and all you want to see is sales being completed. However, this is not as simple. Setting up an online store is the first part, but getting the first sales is another segment of the e-commerce business you will have to work on.

Again, it is very useful to have a plan, a sort of a strategy on how you are going to promote your business to the online users. There are slim chances of online customers finding you on their own if you are only starting out. Your website will probably need some time to become indexed by the search engines, and you will probably struggle to get visits at first, let alone to realize actual sales.

At this point, you will probably start wondering how to make the first sale. What can you do to improve chances of starting to sell in your store?

Well, it might be a while before you see the first sale, but instead of waiting for this to happen on its own, there are online media and tools you can use to increase the reach of the products and ultimately increase the chances of selling. The following set of strategies will help you get through this first phase and make the first sale.

Leverage the power of Facebook

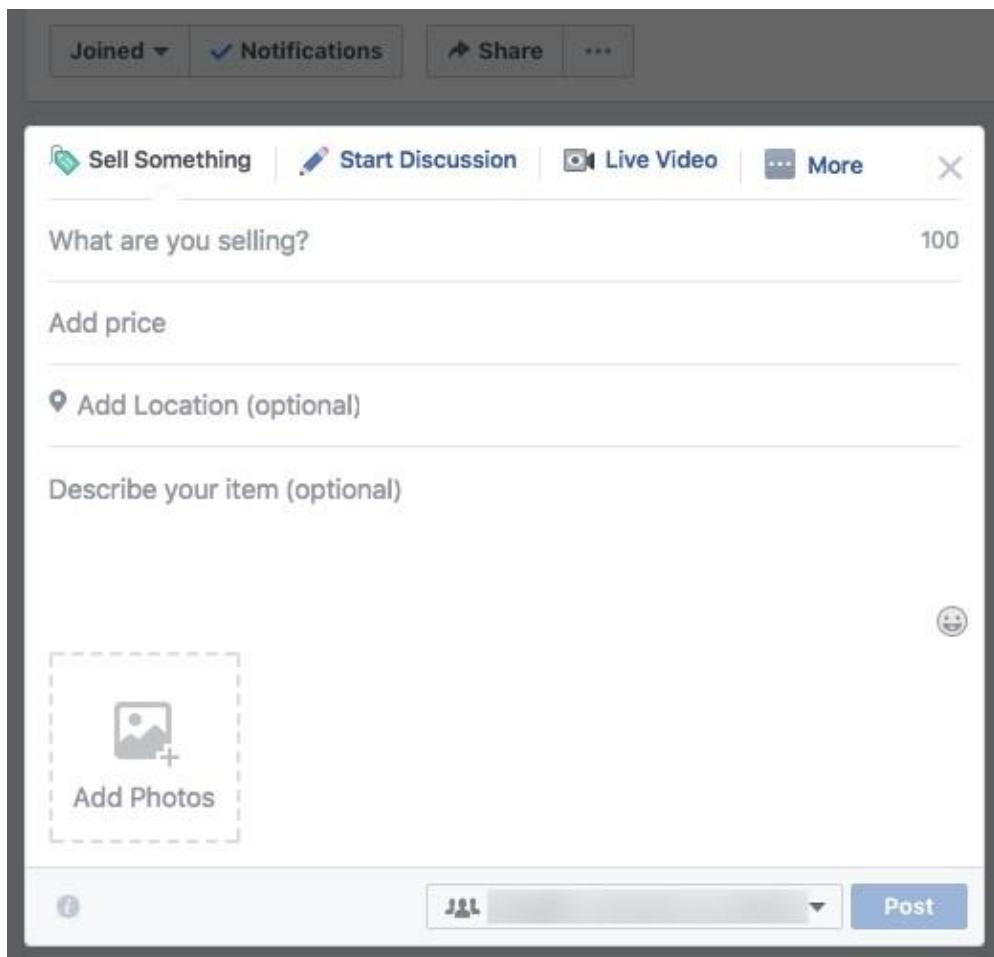
This network is such a powerful tool for promoting a business. Regarding promoting your business, there are four options to help you make the first sale.

Facebook profile

Although you will not use this option as much later on, in the first few months of starting a business, you could think about sharing your products on your own Facebook profile. This means that this post can be seen by your Facebook friends, as well as by your followers if the post is public. Obviously, your target group goes beyond your Facebook friends, but it is a good way to start spreading the word about your business.

Facebook group

Facebook groups are places for people of the same interests to hang out, exchange experience and share stories. Some groups also support selling feature, which you will quickly spot if you see the “Sell Something” option at the top.



You have to be very careful to follow the rules of the group when sharing the links of your products. Of course, you do not want to share too frequently and risk being marked as a spammer. In general, this option is a good way to reach people of the same interests, and it can help you with increasing the reach of your products.

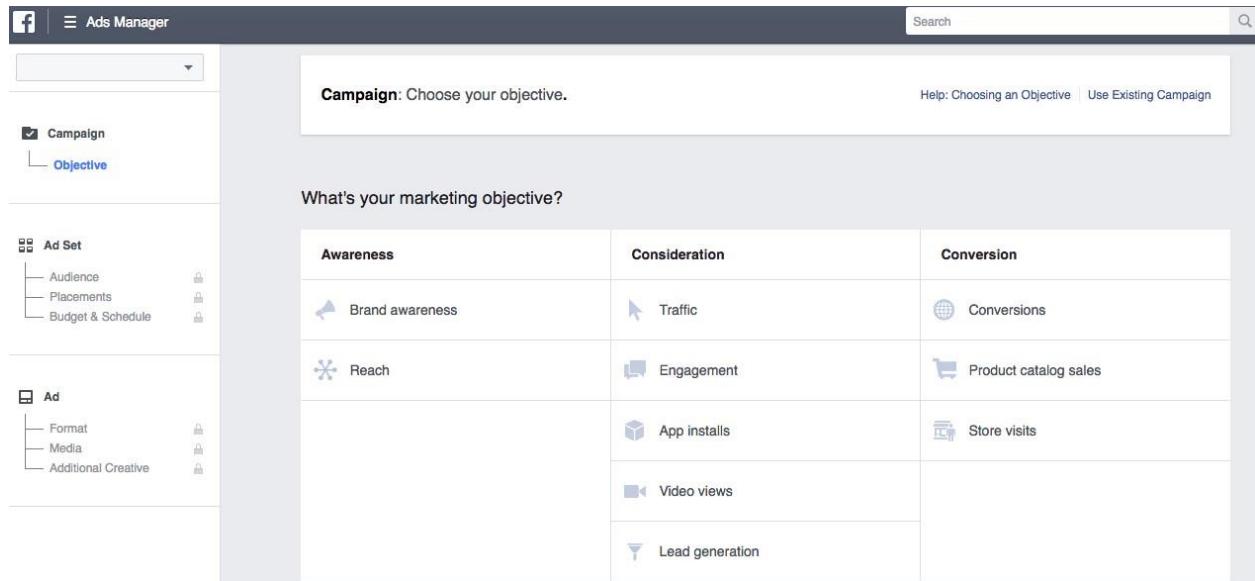
Facebook page

When you create a business, you should create a Facebook page as well. A page is your official presentation on this social network. The page should have photos, website address, contact information and other information about your business such as product information, hours, etc. You will then try to attract followers and start being active on the page by posting updates regularly. Even though having a Facebook page is recommended for all the businesses, to start getting conversion and sales from the page will probably take you some time. The page will start with zero followers, but you will slowly build your following through all sorts of interaction, viral posts, collaborations, recommendations, or ultimately paid advertising.

Facebook ads

The final option you have when starting promotion of your business on Facebook is paid ads.

You will need to have a Facebook profile, which you will use to access Facebook Ads Manager. You will start creating a campaign, through which you can achieve different goals such as brand awareness, increasing traffic, generating store visits, etc.



The screenshot shows the Facebook Ads Manager interface. On the left, there's a sidebar with navigation links: 'Campaign' (selected), 'Objective' (under Campaign), 'Ad Set' (with Audience, Placements, Budget & Schedule), and 'Ad' (with Format, Media, Additional Creative). The main area is titled 'Campaign: Choose your objective.' It includes a search bar and links for 'Help: Choosing an Objective' and 'Use Existing Campaign'. Below this, a section asks 'What's your marketing objective?' and lists several options in a grid:

Awareness	Consideration	Conversion
Brand awareness	Traffic	Conversions
Reach	Engagement	Product catalog sales
	App installs	Store visits
	Video views	
	Lead generation	

Following the instructions in the manager, you will go through a set of options to customize your campaign by choosing ad format, budget and schedule, placements, audience, etc. You will have full control over the budget. You will also use targeting options to choose the audience for your ads which is how you make sure that your ad is shown to the people who are generally interested in what you are selling.

When starting out, this option can be very helpful. With a Facebook page, you need to wait for people to like your page through organic reach, while your ads will instantly be served to the target group as soon as the campaign goes live.

Spread the news on Twitter

Here is another network that can help with promoting your business and spreading the news about your products. Start by setting up an account for your business. On Twitter, there is no distinction between a profile and a page, but instead, you have an account. You can have your private account, and then you can have an account of the business you manage. For the account of your business, you will need a profile photo which is preferably the one used in other profile as well, to help with recognizing your brand and promoting logo. You will also need a bio where you can share your mission, type of business you run or location. Just remember to keep it short, since there is that 140-character limit. You should also add a cover photo and the website URL.

You start the interaction on Twitter by tweeting the status updates to your followers. Have in mind that following someone on Twitter does not establish a mutual connection. When you start

following an account, you can see their updates, but they cannot see yours until they follow you back.

In terms of posting status updates, try to be active regularly and offer great content. Do not make it all about sales. Instead, try sharing some useful content, practical information, problem-solving articles or videos, etc. This will make your audience appreciate your content more, and it will help you build trust. Once your first sale is realized, Twitter is a great place to interact with the customers and ask them for a feedback about the product, about the purchase, etc. Not only is this going to help with improving the business-to-customer relationship, but it will also be a great promotion for your business because other potential customers will see your professional relationship and positive feedback others shared about their purchase.

Additionally, you can use Twitter ads as an extra option to reach more people with your tweets.



The image shows a screenshot of the Twitter advertising landing page. At the top, the text "Advertise on Twitter" is displayed in blue, followed by the subtext "Solutions to help you achieve your business goals". Below this, there are two buttons: "Get started" on the left and "Don't have a Twitter account? Sign up" on the right. To the right of the text, there is a graphic featuring a smartphone, a computer monitor displaying a dashboard with graphs and charts, and a laptop, all set against a background of abstract blue shapes representing data or social connections.

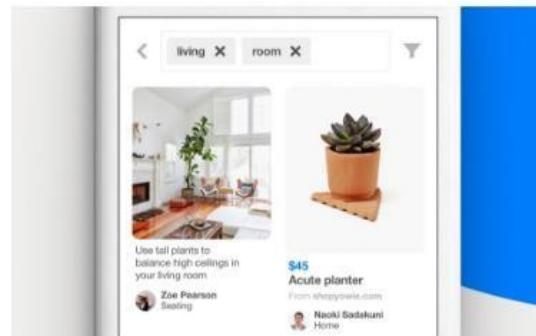
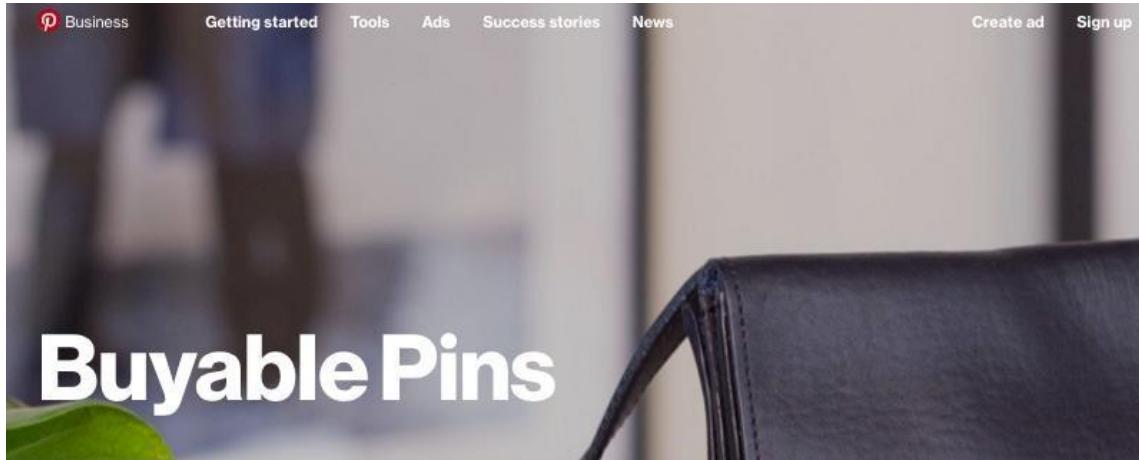
<p>Reach potential customers</p> <p>Get your messages in front of people not yet following you by promoting your Tweets</p>	<p>Gain more followers</p> <p>Quickly grow your community of high value followers and drive word of mouth by promoting your account</p>	<p>Measure results in real time</p> <p>Track the growth of your follower base and see how people engage with every single Tweet</p>
--	--	--

Network on Pinterest

Pinterest slowly grew from a social network to a powerful tool for online store owners. The platform is primarily focused on visual content, which is why you will need lots of inspirational and creative images to promote your products on Pinterest. It is recommended to have your personal account created on Pinterest, but you should create a business account as well. This way you can take advantage of both. Through your personal account, you can have a more personal relationship with online users and share occasional updates, while you will use the power of the business account to actively work on your online promotion through this network.

People use Pinterest to discover content, to share projects and save ideas. These ideas are organized in boards which are created as a single topic with resources shared as pins. Start by sharing amazing images and high-quality content. Interact with other users by sharing comments, liking pins, etc. A very helpful option with spreading the word on Pinterest is to have the “Pin it” button on your website.

In terms of paid promotion, the network offers Buyable Pins where users can see product information and buy directly without leaving the app, or Promoted Pins, pins that are more likely to be seen as Pinterest users than regular pins.



Explore the power of video

As a content format, video is slowly taking over. It has become a number one format to use in content marketing and some even go as far as to say that video represents the future of marketing. When it comes to sales, studies show that *“after watching a video, 64% of users are more likely to buy a product online”* ([Source](#)).

Besides promotional videos, which is probably the first thing that comes to mind when you think about using video for promotion, there are other options you can explore. Product reviews, unboxing, live chats, webinars, and Q&A sessions, are all some ideas how you can use this video format to create content that is primarily addressed to your customers. Rather than directly trying to sell, focus on content that is creative, engaging and helpful to gain trust and credibility.



Image: <https://pixabay.com/en/camera-photography-lens-equipment-690163/>

As you are waiting for that first sale to be finalized, start exploring the video format and networks such as YouTube, where you can share your content to gain more reach.

Embrace the world of social media

Social media world is here to stay, and if you are planning to make it work for your business, you have to be a part of that world. Despite the fact that you might not want to spend an hour or two of your day reading tweets or YouTube comments, chances are, at some point, you will probably have to do so.

Social media is a great place to get reviews from the clients, to share the new products, to help customers complete their purchase, to get feedback or to ultimately solve issues that customers might have. All of this means that social media is a place where you get to connect with your customers and to establish a good relationship. This relationship is very useful for building the reputation and creating a reputable brand on the market.



Image: <https://pixabay.com/en/tree-structure-networks-internet-200795/>

When you start your e-commerce business, try to explore the world of social media and how you can benefit from it. Since businesses are all different, with different goals and needs, there is no unified approach. There is no one single roadmap to follow that will surely bring the good results. Instead, social media marketing is a personalized and customized approach with actions that are designed to promote your business through social media and get visitors to your website where you can convert them into customers.

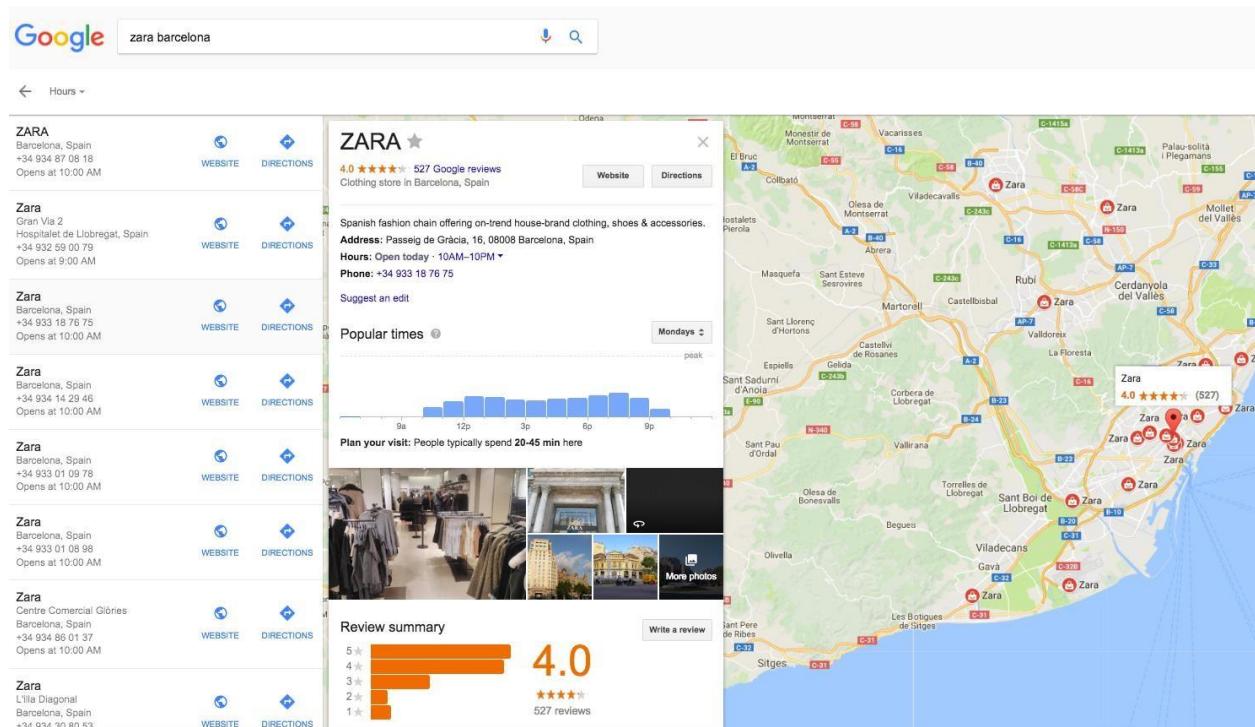
Here is how you can start:

- Choose the networks you want to be active on
- Create profiles on those networks
- Customize each profile
- Start sharing content through these networks
- Promote your online store
- Explore paid advertising if available

- Interact with your followers
- Define and implement social media marketing strategy
- Keep an eye on the statistics
- Analyze the approach and make necessary changes

Get on Google

There is an easy way to get on Google, even if you are still in the initial phase of building your online presence. You should use [Google My Business](#) tool. This option helps you to get your local business listed on the map, with all the necessary information such as the company name, price range, direction, website, hours, reviews, etc. All of this can help you get your business on Google quickly, which means your business is likely to be shown in local Google searches.



While the process of getting in the search results might be more complicated, Google My Business account ensures that your business will be shown as a suggestion when the local area is explored. In case you have an offline store as well, this will help you a lot when getting started with e-commerce.

Check your website again

During this initial phase, you should go ahead and recheck your website. Even though you might have done everything right when setting up the site, errors can still slip through, so make sure that everything is just fine with your website.

SEO

Check if your site is SEO-friendly. Besides checking all of the aspects manually (image optimization, content optimization, detecting any external or internal link errors, etc.), you can always conduct an SEO audit of your website to identify any weak spots. Some of the SEO audit tools include:

- [Site Auditor](#)
- [Seoptimer](#)
- [Website Grader](#)

Website analytics

Analyze your site analytics and if you notice any unusual data, such as a high bounce rate or short average session duration. All of these can be indicators that you need to improve your website performance.

Mobile-friendliness

A website that is not mobile-friendly will hardly have any chance of succeeding nowadays, so use this [Google's mobile-friendly test](#) to check how your site performs on mobile devices.

Write a blog post

Here is another creative thing you can do while you wait for that first sale to hit. Write a blog post. If you have been exploring the world of online marketing, you probably know that content marketing holds that special place nowadays. It is an amazing way to reach consumers. Unlike advertising and promotional campaigns, content marketing uses that subtle approach where you offer quality and help through content in order to get trust and encourage interest in your brand. Indirectly, you will be promoting your brand as well.

When writing a blog post for an e-commerce website, make sure the topic is directly related to the type of products you are selling. Your goal with content marketing is to create content that will attract people who could also be interested in what you are selling. This is why choosing a niche is very important.

When it comes to content creation and blog posts, you have a couple of options:

Share on your blog

Create your blog and start sharing content that will be helpful to your target group. This content will also help with on-site SEO.

Write a guest post

Instead of publishing the post on your blog, try to find blogs or websites from the same niche that would be interested in publishing your post. Of course, you would have the link back to your website. This is great for those visitors who would like to know more about you, and it is good for off-site SEO.

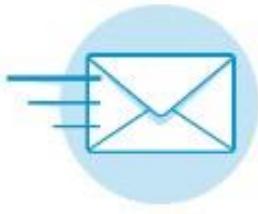
Interview an influencer

The post you can create can be a product of collaboration. The major benefit of working with influencers is the fact that you will be sharing expert advice and you can count on the influencer sharing this post with the followers on social media.

Start building your mailing list

It is never too early to start with building a mailing list. That “Sign up for the newsletter” button should be on your website from the beginning. This is certainly a start, but to really work on building your mailing list, try creating something that you will share exclusively with the subscribers. It can be free content (an ebook, a video course, templates, etc.), free access to members’ area, exclusive access to limited products, coupons or discounts, etc. All of these will give that special character to the subscription, and they will make people really want to leave their email to gain access to this special benefit.

Have in mind that email marketing results are impressive, and your mailing list will become one of the powerful tools in your online marketing strategy. Even if you still do not see any sales, you should still work on creating a mailing list. At one point, you could try to convert those visitors through an email campaign.



Sign up

AND GET 10% OFF YOUR FIRST ORDER!

Email

Sign Up!

No thanks

The primary goal with all of these strategies is to work on spreading the news about your ecommerce, which is essential in the beginning because you have yet to establish your brand and your position on the market. Ideally, you should work on all of these aspects simultaneously, so that you can take advantage of those and really give your business that first kick.

8

Strategies to Increase
Sales

8. Strategies to Increase Sales

As you can notice, there are a lot of things to do while you are waiting for the first sale to happen, but even when it does happen, it is not over. There are plenty of activities that will help you improve your business and increase profit. The work on promoting the products and reaching customers is far from over. Actually, these activities go in a circle.

Your goal is to sell products consistently and to increase the customer base gradually. The activities that help you get your first sale are helpful later on as well. For each of those, you get to define a set of tactics and strategies to use in order to maximize the effect of those activities. As your business grows, so will your approach to using different tactics to promote it with online marketing.

With this in mind, here are a couple more tips to make online marketing a part of your routine:

Social media marketing

You should actively be using at least one social media to interact with your online customers. Social networks enable users to find you, to interact with you and learn more about your products, special offers, etc.

To make the most out of social media, follow these tips:

Choose the networks

There are hundreds and thousands of social media out there, but in terms of business, you will probably end up using only a selected few. Choose the networks that are most likely to help you reach customers and promote your products. At first, you might create profiles on multiple networks, but after a while, you will begin to notice that only some of them actually bring benefits. This makes them worth the time and effort, and that is why you will keep posting on those, while you might reduce the activity on the networks that are not converting.

Create a plan

Publishing as an e-commerce business on social media usually entails a particular plan. It is not enough to post product links or blog articles randomly, but instead, try to make posting activity organized and coherent. Have in mind that social media users love helpful and engaging posts, which are likely to perform better. This means that providing great content through social media is very important, while promotional posts and advertisements of products are activities you should occasionally do and if possible, discreetly. For example, if you want to promote new tennis gear in your online shop, try writing a blog post about tennis gear and how it can influence a player's game. Instead of directly sharing product links, you will create a useful article with links to products for those interested in buying tennis gear. Ideally, you want to mix between helpful

and promotional content, but this ratio should be 90:10 for example. So, more great content, and less direct advertising.

To follow these guidelines, create a plan on what you will post. Your plan can also include the times when you will post on social media. This kind of planning is usually based on customers' behavior research or the engagement statistics from the previous period.

Try creating a calendar to organize everything and make sure your offer diverse content to your customers. Using social media management dashboards and scheduling tools can help you with keeping everything organized. Helpful tools for this purpose include:

- [Buffer](#)
- [Everypost](#)
- [SocialOomph](#)
- [HootSuite](#)

Be active on social media

While customization of the profiles and regular status updates are helpful in terms of establishing the presence on social media, being active is another major component you should not neglect. You must not forget that social networks are all about people, connections, and communication they can establish through these networks.

When users interact with your business profile, regardless if they leave a comment or send a private message, they expect real people to answer their questions. They do not want a template that will be copied and pasted all over again. You should always try to make it personal and to make the customers feel like they get the attention they deserve. It is also helpful if you are monitoring social network notifications regularly and if you are able to respond rather quickly.

Clothing Store in London, United Kingdom

Community See All

Invite your friends to like this Page

4,882,305 people like this

4,777,601 people follow this

and 1 other like this

About See All

Typically replies instantly

Send Message

asos.do/p3RFnb

Clothing Store · Clothing (Brand)

ASOS

SALE

ASOS

ASOS is active now. Start a conversation.

4.8M people like this including
and 1 friend

Explore paid options

Most social networks offer paid options to help with reaching more users. Organic reach is usually limited to your own followers, and even in that case, not all of them is going to see the posts you share. To make sure you increase the visibility of your post, try using paid advertising as an option and see how the ad performs. If you notice that you can make your investment worth, if you see real profit from the social media ads, then this is something that should be integrated into your social media marketing strategy.

facebook business

Search Facebook Business

Resources Support

Adverts Pages Inspiration Success Stories News

Create Advert Create a Page

Make meaningful connections with people to grow your business.

Learn about Ads Create an Ad

Image: <https://www.facebook.com/business>

The major benefit of paid advertising on social media are:

- Reaching more people
- Increasing brand awareness
- Using advanced targeting options (target based on location, gender, based on the interest, target members of specific groups, etc.)
- Scheduling ads (choose the time during which the ad will be active)

Analyze the performance

Once you analyze the performance of the accounts, you can see the whole picture. This can help you determine whether you want to be active on a certain network, whether you need to change your approach to social media marketing, etc. Pay attention to these things:

- The number of followers for each network
- Engagement level (comments, shares, likes)
- Negative statistics (unlikes, marked as spam, reported post, etc.)
- Click-through rate
- Organic vs. paid reach
- The number of conversions (sales, leads, etc.)

Finally, it is important to have in mind that social media marketing is a whole segment in online marketing, and as such, it should be studied in more details. These are quick suggestions on how to use social media to keep your business profiles active, but to get the full potential of having and implementing a social media strategy, you should explore the topic in more details. For starters, check out this [Social Media Marketing Course](#).

Email marketing

Collecting emails on your website is an excellent way to build a list of those interested in your business. On one side you have those interested, while on the other you have those who already are your customers, which means that they have completed at least one purchase on your website. As a result, this means that you will have to use at least two different strategies to communicate with those users effectively.

You can divide the strategies like this:

Subscribers

This first group includes the users who subscribe to your blog updates or provide their email address so that they could redeem an offer you have provided, such as a coupon, or a free template. When you use email marketing, your goal is to get those subscribers interested in your business even more. Here is what you can do:

- Do not send a promotional email as soon as the subscriber is on the list
- Instead, start building this relationship gradually
- Offer even more great content to the subscribers
- Share some helpful content from your blog
- Use personalized approach
- Customize each email campaign
- Make sure your email message is mobile-friendly
- Then share a promotional email where you would feature some of your products •
Offer exclusive discounts for subscribers (or first-time customers) only

Customers

Customers are those who have completed at least one purchase. This means that they were persuaded to try out your products. However, this is not to say that your relationship is over. The same way you nurture the relationship with the subscribers, it is necessary to nurture the relationship with the customers.

Some of these tactics can be helpful:

- Ask for their impression related to the product, or any feedback they would like to share
- Ask if the customer was satisfied with the purchase process, customer service, etc.
- Suggest that they share the experience (or link) with friends
- Offer exclusive discounts for recurring customers
- Send customized email with product suggestions based on the previous purchase
- Share product suggestions at the perfect time (for example, if you sell printer ink cartridges, and you know that one could last approximately a month, you could send

a reminder email next month, just in time when the person would actually need to buy a new one).

Analyze the abandoned cart

We have already mentioned that an astonishing number of products placed in carts is never actually purchased, but instead, the cart is abandoned. This abandonment can happen anywhere through the checkout process, but it most commonly does when unexpected shipping costs are calculated (based on the studies).

Nowadays, most e-commerce businesses use email as the first piece of information they require when the person starts the checkout process. Some require creating an account, but more and more business are deciding to skip this step to speed up the process and make purchase quicker and easier for the customer. Asking for an email address early on guarantees that you get priceless information to contact the person even if they abandon the cart along the way.

This kind of a situation requires a completely new email marketing approach where you will have to define how you address your customers (or potential customers). Obviously, there was a reason for that person to abandon the cart. They might be surprised by the shipping costs, but they also might be uncertain whether they want to buy the product from you or not. Or perhaps they were simply in a rush and had to quit the browser before they were able to complete the purchase.

When sending an email to the email addresses you have in your abandoned cart list, you should:

- Remind the person they have the products in the cart
- Ask for feedback about the purchase process (a poll with ready-made answers is much quicker than for the users to write the feedback themselves)
- Share helpful resources from your blog or website
- Offer a special discount if they complete the purchase they have started within a certain time limit

In conclusion, we have to highlight the importance of the user in e-commerce business. All of the work you will have when trying to make the first sale and beyond is focused on reaching new users and converting those interested into your customers. The users will become the center of your strategy and your ultimate goal will be understanding users and which practices would be the most effective in order to turn them into your customers.

As you will soon realize, a significant part of an e-commerce business is developing and nurturing that business-to-customer relationship. Both social media marketing and email marketing are helpful strategies to do so, which is why you should explore them in greater details.

9

E-commerce SEO

9. E-commerce SEO

If you want to run a successful online business, you know that SEO is everything. Search engine optimization helps you increase the performance of your website in order to make it more user-friendly and discoverable in the search engines. As a result, you can expect more sales through the website which is SEO-friendly because it is more likely to be shown in the search engine result pages.

Keyword research

Every SEO strategy and planning starts with the keyword research. Keywords make a base for the website optimization, content creation and even for the paid search engine ads. This is why they are so important part, and it is crucial that you start with keyword research.

Keywords are basically words or phrases you want to position for. This means that once that word or phrase is typed into the browser, you want your site to show at the top of the search engine result page. Being positioned at the top ensures that the users find your website and click on the link. According to numerous studies, the top results receive the highest number of visits, as opposed to the results shown below, while the results on the second, third and subsequent pages of the search result received hardly any visits.

In a nutshell, the process of keyword research works like this:

- Write down all the words and phrases you think are associated with your business
- Add synonyms and related phrases
- Explore the competitors and the keywords they are using
- Use a tool to check the popularity and competitiveness of those keywords
- Narrow down the list based on popularity and competitiveness
- Use a tool to get keyword phrase combinations
- Make a final selection by choosing a certain number of keywords (for example, between 10 and 30)

Since you will need a tool to help you with keyword research, here are a couple of suggestions:

- [Google Keyword Planner](#)
- [Keyword Match Type Tool](#)
- [Keyword Explorer](#)
- [Keyword Tool by WordStream](#)

- [SEMrush](#)

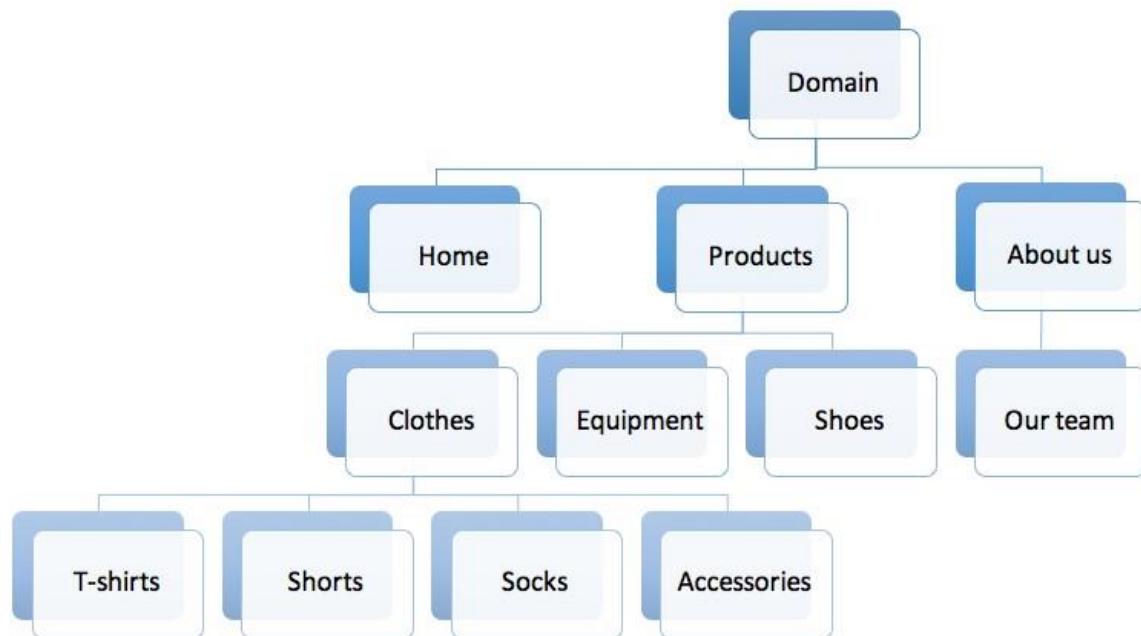
Site structure

Site structure or site architecture is the way your website is built. In general, you should always have in mind SEO-friendliness as a factor when creating a website, when designing the menus, categories, etc. Everything should be well-organized and follow a logical structure, which is also something that the search engine crawlers will be able to follow.

Structure

E-commerce websites usually have lots of pages. These are typically product pages, and there can be thousands of them. If not organized properly, these pages can turn into a huge mess, in which case it would be impossible to index the pages accurately and determine the relationship between them. Therefore, thinking about site structure is crucial.

Start with the domain and go from there. You will have a couple of pages branching, such as home, about us, contact, products, etc. Now, each of these can branch into other submenus. You want to focus on products, or services if that is what you are selling. If you have a lot of products, try to organize them in categories. For example, if you sell tennis equipment, you can have categories clothes, equipment, shoes, etc. Then, inside each of these categories, you will further classify the products. Speaking of clothes, you maybe have T-shirts, shorts, socks, accessories, etc.



A structure like this one makes pages more organized and easier to find, for both the users and the search engines.

Friendly URLs

Another element of the website structure that has the influence on website ranking and indexing process is a friendly URL structure. A URL helps search engine crawlers to understand context and index those URLs properly. A good SEO-friendly URL usually:

- Is readable
- Is short
- Matches the title
- Has keywords or words that describe content
- Does not use the stop words (*and, or, but, the*, etc.).

When creating URLs for an e-commerce website, you should not use numbers or other abbreviations for the product page name. Instead, use the product name and make this also a part of the URL.

So, following the example above, instead of: <https://yourdomain.com/products/shoes/90087>

You could use: <https://yourdomain.com/products/shoes/barricade-classic-bounce-shoes>

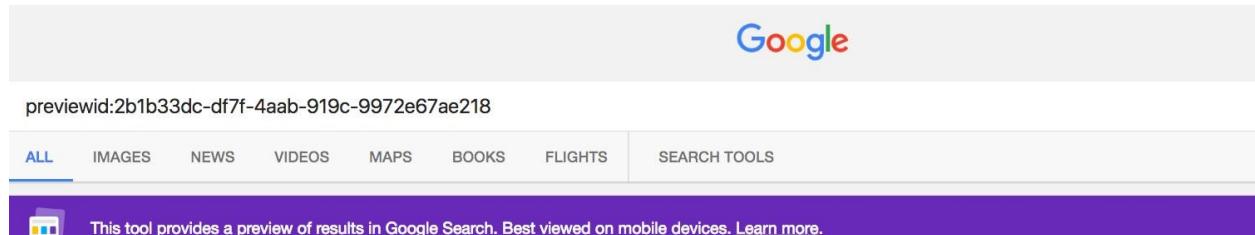
Internal links

Internal links are the links that point to other pages of your website. In terms of SEO, internal links help website crawlers to discover new content on your site. It works like a giant spider web. A search engine crawler gets to one page of your website, and then from that point, it crawls to the other pages. Without the internal links, the crawlers would consider this one page independently and would neglect the other pages.

When you create a good structure of your e-commerce, you do a pretty good work in arranging internal links. The navigation is created through the structure, so every product page links to the other internal page which actually has higher priority. For example, the product page links back to the category page, and the category page links to the menu.

Structured data markup helper

Website content is presented in a way search engine crawlers see it. Google, for example, needs to understand the content in a better way, so that it can present this content in the search results more effectively. Apart from the search results, Google can actually present content in a useful way in Gmail as well. This is done through rich snippets. These snippets improve user experience with enriched information and presentation that is more engaging.



To take advantage of this feature, all you need to do is tell Google about the data on your website. In terms of e-commerce, you will use “Product” as the data type.

A screenshot of the Structured Data Markup Helper tool. The interface has a header with tabs for Website and Email. A red circle highlights the "Enter Page" button. Below the tabs, there's a text input field with the placeholder "Enter a URL or paste HTML...". To the right of the input field are three buttons: "Tag Data", "View HTML", and "Start Tagging". The main content area is titled "Structured Data Markup Helper" and contains a list of data types: Articles, Book Reviews, Events, Local Businesses, Movies, Products, Restaurants, Software Applications, TV Episodes, and TV Episodes with Ratings. Below this, there are tabs for "URL" and "HTML", and a text input field containing "url".

Image: <https://www.google.com/webmasters/markup-helper/?hl=en>

Provide the URL or HTML of the page you wish to markup and once you finish it, you will get schema.org markup which you will add to the pages on your website. The same process can be done for email, so that Gmail presents the data in a new way.

On-site SEO

On-site SEO is the process of optimizing website pages to make the website more SEO-friendly, and thus increase the performance of its pages in the search engine results. This process requires a series of actions to enhance the performance. You need to follow the guidelines related to

content optimization, image optimization, etc. This is what is typical for most websites, but for an e-commerce website, on-site SEO also has to focus on product pages and their optimization.

Title tag

Title tag represents the title of the page. It is marked with an H1 tag (heading 1), and it has the greatest importance for the search engine crawlers. Typically, you should use keywords in the title tag. When it comes to an e-commerce website, your goal should be to aim for a more specific keyword phrase (a long-tail keyword). These actually contain words like *buy*, *online*, *cheap*, *best*, etc. These words help you create long-tail keywords that perform better as they will attract the right traffic. For example, if you use keyword “*buy tennis equipment*” in the title tag, the page should appear when someone uses this phrase in the search query. And someone who conducts this kind of query is usually interested in completing the purchase. Of course, these do not apply to product pages, which should have product names instead.

Product pages

Product pages are an essential part of e-commerce, but at the same time, they are what makes e-commerce SEO different from the usual website optimization. Unlike content articles, which provide a lot of content, which is optimized and thus accessible and indexable by the search engines, product pages might become a challenge. Firstly, they do not have as much content. The main content on these pages is actually product description.

Secondly, product pages usually focus on visual content, such as images and videos. Visual content is more successful at grasping the attention of the visitors. Additionally, the features and usefulness of the product are much better illustrated through the visual content, which makes it more effective in terms of persuading customers to buy.



Image: <https://www.lilgadgets.com/products/connect-pro>

Product description

Google uses content to learn what your page is about. When you provide this content, you instruct Google how to index your page and when to show it in the search results. This is why it is essential for the content to be relevant to the actual product. You will also need to use keywords, but always make sure that you use them naturally in the text, along with the synonymous phrases.

Regarding content length, the industry statistics show that “*long form content of over 1,000 words consistently receives more shares and links than shorter form content*” ([Source](#)). Even though this might be a difficult goal to achieve if you have thousands of product pages, you could at least provide lengthy content for the most important pages, such as the main category page. Try creating more content to enrich those pages and make them more SEO-friendly.

Product images and videos

Since they are an essential part of an e-commerce website, product images and videos should be SEO-friendly as well. When it comes to the users, the most obvious requirement is that the visual content is relevant and of high-quality. On the other hand, the search engine crawlers are unable to see content this way. Instead, they try to understand what the content is all about. How do they do that? Using HTML tags.

- Name – Start by file name. Always make sure the file name is descriptive and, if possible, use keywords in the file name.
- Size – High-resolution images are a fantastic presentation for the website visitors, but they might cause optimization problems for your website, such as decreasing site speed. Therefore, make sure to use a tool to change the format or compress the file when uploading images and videos to your server. These will help you preserve the quality and reduce the file size.
- Alt text – Alt text or alternative text is a phrase or a part of the text that is displayed when the image file cannot be uploaded and shown to the viewer. Alt text enables search engines to understand what the image file is about because the search engines on their own cannot see the images. As a result, this helps them index the visual files correctly.

Related products

Related products feature is great for two reasons. Firstly, it shows similar products to the customers, which can increase the chance of them finding new products on your website. The other reason why you should have this feature is the fact that the related products are actually links to other product pages. Like mentioned in the part about the site structure, for search engines crawlers it is essential that you interlink pages and make logical connections. This will provide the search engines with additional context to understand and index the product pages.

Apartment / Lighting + Candles / Mandy Table Lamp



Online Only

Mandy Table Lamp

\$69.00

***** | No Reviews Yet

Color: White



Size: ONE SIZE

Qty: 1

* Ship to Me

In-Store Pickup

Please enter a location to see the nearest stores.

[Change Location](#)[Add to Bag](#)[Add to Wish List](#)[Details](#)[Shipping + Returns](#)

Reviews

*****:

No Reviews Yet

[Write a Review](#)

If You Like This, You Might Be Into These



Image: <https://www.urbanoutfitters.com/shop/mandy-table-lamp?category=table-desklamps&color=010>

Finally, e-commerce websites with lots of products can have thousands of product pages. Not only is it a lot of work to optimize each page, but additional problems might occur due to the fluctuation of products. When you have lots of products, it usually means that you will often get new products, some of the old ones will be discontinued, etc. What it all means is that you will

need to develop a system to track all of these pages and keep an eye on changes. Maintenance of an e-commerce website can be an arduous task, and the best way to make it work is to keep everything organized. Start with a great website structure, which is yet another reason why you need to assign product categories and subcategories.

Product page issues

In terms of products, we can differentiate between two types of e-commerce websites. First, there are those that sell a limited number of products, which are usually pretty constant. This means that they are unlikely to update product pages often. Websites that offer services also do not change their pages frequently. The offer is usually the same and rarely changes. In these cases, website maintenance is somewhat simplified, because once you create and optimize the product pages, that is pretty much it.

On the other hand, we have more dynamic types of websites. These are the sites that have a lot of products, sometimes even hundreds or thousands. Apart from this large amount of products (and thus product pages), there is a need to keep an eye on the product inventory because some of the items might be sold out, some might be seasonal, so they are only sold during a particular period of the year, etc.

Since these websites are dynamic, the product page information change frequently. Unfortunately, this leads to a lot of product page issues which can result in broken pages, incorrect or obsolete information, etc.

- New products

Every time there is a new product added to your website, think about your site structure. You have to add the product page to the right category, and you have to optimize it based on the guidelines above. You could also add “New” sign to the product page or even create a special category with the newest items.

- Out-of-stock products

If the product is out-of-stock, but it will be back soon, you should just leave those pages up with a message that the product is currently unavailable. This way you keep the ranking of that page. As for the users, it is helpful if you could provide the date when the product will become available (they can leave their email to get the notification), or you could offer alternatives, such as the products in a different color or a similar model.

- Expired products

When the product is expired and will no longer be available, you have two options. You can either delete the page, in which case some users might be arriving at the 404 error page. Alternatively, you could use 301 redirects. In this case, the users will automatically be redirected to the new

page. The good thing about 301 redirects is that they pass link juice from the old page to the new one. So, if the expired product page was ranking really well, this will transfer to the new page. You could redirect to a new product page (for example, a new model of the product) or the product category page.

- Product variations

When you have product variations, such as different sizes, colors, etc. you might have different URLs for those pages. However, you should always use the rel=canonical tag for the original product pages. This way the search engines will know which pages to prioritize, and you will avoid the risk of duplicate content because the description is likely to be similar or perhaps even the same.

- Seasonal products

Since seasonal products are available for a limited time only, try introducing these pages earlier with “Coming soon” label. This will give you more time for the product pages to be live on the website allowing the search engines to index them early on. Labels such as “Limited series” show to the users that the products will be available for the limited time only, urging them to buy if they are interested. If each year, you bring out new products, use the year in the URL to make difference between pages. As for the old product pages, use a 301 redirect to redirect the users to the product category.

Due to this dynamic process of product page updates, you also have to update your website sitemap regularly. Since the sitemap holds all of the website files together, enabling the search engine crawlers access to read and index them, you need to keep it updated as the pages and links on the site change.

User-generated content

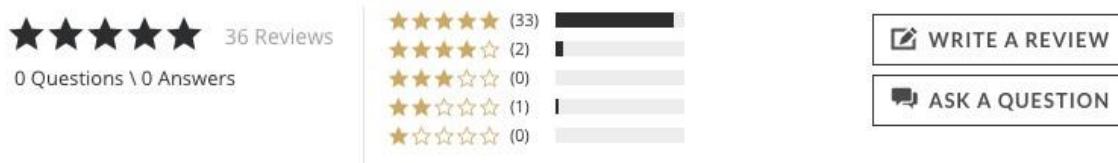
User-generated content is an excellent way to add unique content to your website (good for SEO), and it is also encouraging for the future customers. Therefore, this content brings freshness to your site, and it helps you with achieving conversions. There are two essential steps to help you with user-generated content:

- Obtain great reviews and testimonials by providing amazing products and excellent service
- Encourage the customers to provide their review

When it comes to how to get reviews, try these:

- Schema.org – This review markup enables you to get stars on product pages

- Integrate social media – Plugins for social media integration allow providing feedback through social media accounts
- Install a plugin – You could install a plugin that will enable customers to leave their review. The review can feature customer's name (or any other customer data if relevant, such as age, etc.), a star review and a written review. The date when the review was posted could also be helpful for the future customers.
- Make review process easy – Make this process quick and easy for customers to encourage more reviews



[Reviews \(36\)](#) [Questions \(0\)](#)

Frank P. Verified Buyer 05/02/17

Ratio 8 is the best pour over coffee maker period

I purchased the Ratio 8 and after several tries at weighing the coffee grinds and measuring the water for different carafe volumes I have enjoyed the best coffee ever. Jake and I have had several discussions and the customer service is always beyond my expectations. I am waiting to order the new thermal carafe and until then I rinse the glass cara...[Read More](#)

[Share](#) | [Was This Review Helpful?](#) 0 1

Daniel Walbert Verified Buyer 03/08/17

Warranty and Repair

Where do I begin? The team at Ratio took wonderful care of me when my Ratio broke during a cross country move. They effectively rebuilt it and even upgraded the software to the latest version. The team was also incredibly gracious and kind to me throughout the entire process of shipping and repairing the Ratio. The soft skills and customer care we...[Read More](#)

[Share](#) | [Was This Review Helpful?](#) 2 0

Image: <https://ratiocoffee.com/product/ratio-eight/>

Site search

You should not underestimate the power of site search. This option is very useful in ecommerce, and a lot of website visitors will use it to look for the products. This means that you have to optimize the internal site search results, so that even the users who type synonyms, misspell the product name or use spacing errors are eventually taken to the appropriate product page. This way you significantly reduce the chances of losing sales due to incorrect search results and because users were unable to find the right product using the internal search results.

The screenshot shows the Sephora website's search interface. At the top, there's a navigation bar with links for NEW, BRANDS, GIFTS, COMMUNITY, HOW-TOS, and STORES & SERVICES. On the right side of the header, there are links for 'Hi, Beautiful' (with options to 'Sign In or Register'), a heart icon, and a shopping cart icon with a '1' notification. Below the header, a secondary navigation bar includes links for PRODUCTS (95), VIDEOS (129), PHOTOS (1588), ARTICLES (93), and ADVICE (178). The 'PRODUCTS' link is underlined, indicating the current category. A search bar at the top of the main content area contains the query 'urban decay'. Below the search bar, a message in a red box states: 'We could not find an exact match for "urban decay" and have suggested "urban decay" as a possible alternative.' The main content area displays four product cards for Urban Decay items:

- URBAN DECAY Naked Heat Palette** (\$54.00)
NEW |
A compact of eyeshadow with warm-toned shades.
★★★★★
\$54.00
- URBAN DECAY All Nighter Long-Lasting Makeup Setting Spray** (\$15.00 - \$31.00)
NEW |
A black spray bottle with white text.
★★★★★
\$15.00 - \$31.00
- URBAN DECAY 24/7 Glide-On Eye Pencil** (\$20.00)
NEW |
A pencil with a red-to-gold gradient color.
[36 more colors]
★★★★★
\$20.00
- URBAN DECAY Naked3 Palette** (\$54.00)
NEW |
A compact of eyeshadow with neutral-toned shades.
★★★★★
\$54.00

Below the products, there are sorting and viewing options: 'sort by relevancy', 'view 60 per page', and page navigation buttons (1, 2, >).

Image: www.sephora.com

Analysis of the search results can also help you find out what the users are looking for on your website, which products are popular and how users interact with your website in general (for example, they might explore the category page first, etc.). This can help you understand the behavior of your customer in a better way, and it also shows you if you need to optimize or adapt certain pages to increase their performance even more.

Link building for e-commerce

Link building is an essential activity of off-site optimization. The process includes a series of activities conducted to get more links to point to your website. The main benefit of these links is increased exposure. When a link to your site is placed on another site, you increase the chance of online users learning about your products. In terms of search engines, links are seen as a vote of trust, and thus they help with deserving better ranking in the search engine result pages. This is why gaining quality links (i.e. links from the authority websites) can be very good for your SEO.

In SEO, links are usually deserved with exceptional quality and relevance. These are so-called editorial links, and they are the most valuable links a site can get. When it comes to ecommerce websites, things can be a bit different, because an editorial link can be harder to obtain, especially on its own. Product pages are usually not seen as quality content, but instead, they are often considered commercial content. Here are a couple of ways to get links for an ecommerce website:

Influencer outreach

This strategy includes contacting influencers in the industry and asking them to share the review or information about the product. This can help create a buzz about your products, and get exposure online. To sum up the process of reaching out:

- Find the influencers
- Choose the influencer
- Create an outreach email with an offer for the influencer
- Conduct a campaign together (creating content, organizing giveaway, etc.)
- Track social shares, visits, etc.
- Evaluate the performance of the campaign and determine its efficiency in achieving goals

Partnership

A partnership can be a very lucrative collaboration for business. The idea with this approach is to find another business that is somehow related to you, but it is not your competitor. For example, if your e-commerce website sells tennis equipment, your partner can be a local tennis club. This way, they could promote your brand, and in return, you could offer a discount or special offers for their members.

Affiliate marketing

Online marketing includes several segments, and one of those is affiliate marketing. With this type of online marketing, you become a merchant, and you create special offers for affiliates who

decide to join your affiliate program. Once they join, the affiliates get unique codes that track sales originating from their campaigns. They can distribute these links in any way they see fit. For example, they might share an article on their blog. They might share on social media, or they might even send a newsletter to their subscribers. The point is that they are in charge of promoting your product (and distributing links online) and in exchange, they get a commission, which is usually a certain percentage of the sale. It is a very profitable strategy for merchants because they do not invest anything and yet, have so much to gain.

What we can conclude is that optimizing an e-commerce website does have some distinctive elements, such as product pages, but in the core of each website optimization should be the user. Everything you do here is to improve the website performance for the users. This refers to the entire structure, link building, and even mobile optimization. All of this provides better user experience, and this results in more visits and more interest in your brand. Ultimately, this leads to more purchases and more exposure through recommendations.

Think about your potential customers and how they would browse, what kind of information they would need, etc. All of this helps you learn how to design your website and present products.

In the end, do not forget about the search engines. Even though we are way past this phase in marketing when optimizing for the search engines was all marketers were focused on, it is still essential to enable the crawlers to access and understand the content of your website, so that they could index and rank it successfully. However, this user experience component is believed to have influence here as well, because the crawlers are now learning to evaluate how users interact with your website, how long they do so, etc. This is supposed to show them how exactly your website is helpful and providing better user experience ultimately means that your website will probably rank better in the long run.

10

Common Issues with E-commerce SEO

10. Common Issues with E-commerce SEO

Like said, e-commerce websites often have lots of pages, which requires a lot of time for maintenance and administration, but an error here and there is bound to happen. The best way to be prepared for those is to learn about common issues most e-commerce websites face concerning SEO.

Duplicate content

This is the number one problem e-commerce sites face. First of all, you need to know that Google hates duplicate content, and this can lead to penalties for your website, which means you might lose ranking. Duplicate content confuses the search engines. They are not able to determine which page is more relevant and which should come first in the search results because they are the same. They might have a different URL or the title, but if the content on the page is identical, this is an issue for the search engine crawlers. Duplicate content is caused by:

- Product variations
- Copying short description similar for multiple products
- Using manufacturer's description as other sellers might do the same so the search engines will find the same content on different websites

The best way to deal with this issue is to create unique content. Content helps with optimizing pages and increasing traffic, and it also provides useful information for the customers. However, it can be rather difficult to create the amount of content needed for a large ecommerce website, in which case, there are two more solutions for fixing this issue of duplicate content:

Noindex tag

If there are pages with similar content that cause a duplicate content issue, use noindex tag for a page that does not bring traffic and does not have to show in the search results. Using this tag, you instruct the search engines not to index the particular page, and thus they will not see the duplicate content.

Canonical tag

Another way to solve the problem is by using rel=canonical tag. In this case, you tell the search engines that a copy or a variation of the page does exist, and the search engines know not to treat this page as unique. So even if the pages have similar or identical content, search engines will still know what to do and how to prioritize them.

Content

Content, in general, can be an issue. Besides the duplicate content, there is an issue of “thin content”. Thin content is the situation when a page has a minimal amount of content. One of the main reasons why this issue occurs for e-commerce websites is the fact that there is a need to create a huge amount of content for rather similar products, with the similar features and properties. This can be quite a challenge. In general, longer content (1000+ words) ranks better, which is another reason why content on the e-commerce website can be considered thin. Hardly anyone will write that much content for each product. Just imagine having 1000 words written, or even 1500, for every product. It might be doable with three, four or even ten products. But e-commerce websites can have hundreds of product pages.

How to fix this problem? Start by identifying the pages with thin content and then determine how to improve them. You could use a template. Of course, you will have to pay attention to avoid all of that duplicate content issue, but you could create a sort of a template to make things easier for you.

- Intro – Describe the product
- List of the features – It is best to use bullets
- Product description– Describe when to use the product, how, benefits, award, etc.
- Summary – A short conclusion about who and how will benefit from this product

Your aim should be at least 300 words, but whenever there is a chance to make this description more lengthy, you should go for. It is a general rule to create more content for the most important pages (those that bring traffic, that rank well, etc.).

Pagination

Pagination does not necessarily have to be an issue, but it frequently is for e-commerce websites. When there are a lot of products inside a category, those are sorted using pagination. The number of products shown on each page is usually between 20 and 70, even fewer for the mobile version of the pages.

Series

Adding the tags rel=next and rel=prev is the way to create a series of pages. When you use these tags, the search engine can understand the connection between these pages. You should use noindex tag for all other pages, so the search engine will only index the first one.

Load more

This option shows a limited number of products with the “Load more” button. There is no need to click on separate pages, but the products are shown once the user clicks on the button. This

solution reduced the loading time of the page because the products are not loaded all at once but in segments, as the users click on the button.

Scrolling

Scrolling is similar to load more, but it is infinite. This means that a limited number of products is shown, but as the user scrolls down the page, the content is automatically loaded. This way, the users cannot reach the footer until all of the products are loaded.

Choosing the type of pagination for your website is based on the type of products you sell, but also based on the number of products you have within each product category. What you need to pay attention is not to prolong the loading time or affect your SEO.

Site speed

Another issue that can harm your SEO is site speed. Search engines use site speed as one of the factors that are integrated as a part of their algorithm used to rank websites, which is why this element is crucial for successful website optimization. The most common reasons why the sites load slowly are:

- Large images
- Slow hosting or server
- Messy website code

Address each of these issues to improve the website speed. You could also run a check of your site to get additional suggestion on how to make your website pages faster: [PageSpeed Insights](#).

PageSpeed Insights

Enter a web page URL

Mobile

Desktop

Needs Work

70 / 100

This page is missing some common performance optimizations that may result in a slow user experience. Please investigate the recommendations below.

Possible Optimizations

Reduce server response time

► [Show how to fix](#)

Optimize images

► [Show how to fix](#)

Eliminate render-blocking JavaScript and CSS in above-the-fold content

► [Show how to fix](#)

Leverage browser caching

► [Show how to fix](#)

Minify CSS

► [Show how to fix](#)

Think about all of these issues as a way to further optimize your website. Even though they might be common obstacles, there are ways you can solve them to make sure the performance of your website stays unharmed. You should definitely address these issues as ones of high priority. Do not let anything slow you down, and anything that might harm the user experience should be something you should work on. The bottom line is that reaching more users is what you should be focused on in order to increase sales and fixing these issues can certainly help you achieve this goal.

11

Mobile SEO for E-
commerce

11. Mobile SEO for E-commerce

The whole world is going mobile. So should your e-commerce website. Mobile optimization is something you should focus on with your e-commerce to make the most out of your performance. The number of customers buying from their phones is growing gradually. Even though they mostly use it for browsing, a significant rise in the number of purchases is also noticed.

One of the ways to address the issue of a mobile-friendly site was to have a mobile version of the website. This means that the desktop version would be a primary asset, while the mobile version would be loaded for the mobile users. This practice is no longer recommended for two reasons. Firstly, it gives precedence to the desktop version. However, the search engines now consider mobile-friendliness as one of the most important factors for ranking, and actually, give priority to the mobile-friendly websites in the search results. Secondly, having this separate mobile version can somewhat limit the user experience, because the site might not be fully recreated for mobile.

Instead of this solution, there are different strategies to optimize a website for mobile devices.

Have a mobile-friendly configuration

Nowadays, when designing an e-commerce website, or any website for that matter, you need to use a responsive design. What this means is that you will be creating one website. Still, this site will automatically adjust regardless of the type of device you use to access it. So a responsive design resizes to provide a mobile-friendly experience. There is no need to zoom in to see the text because the content will be adapted to the small screen.

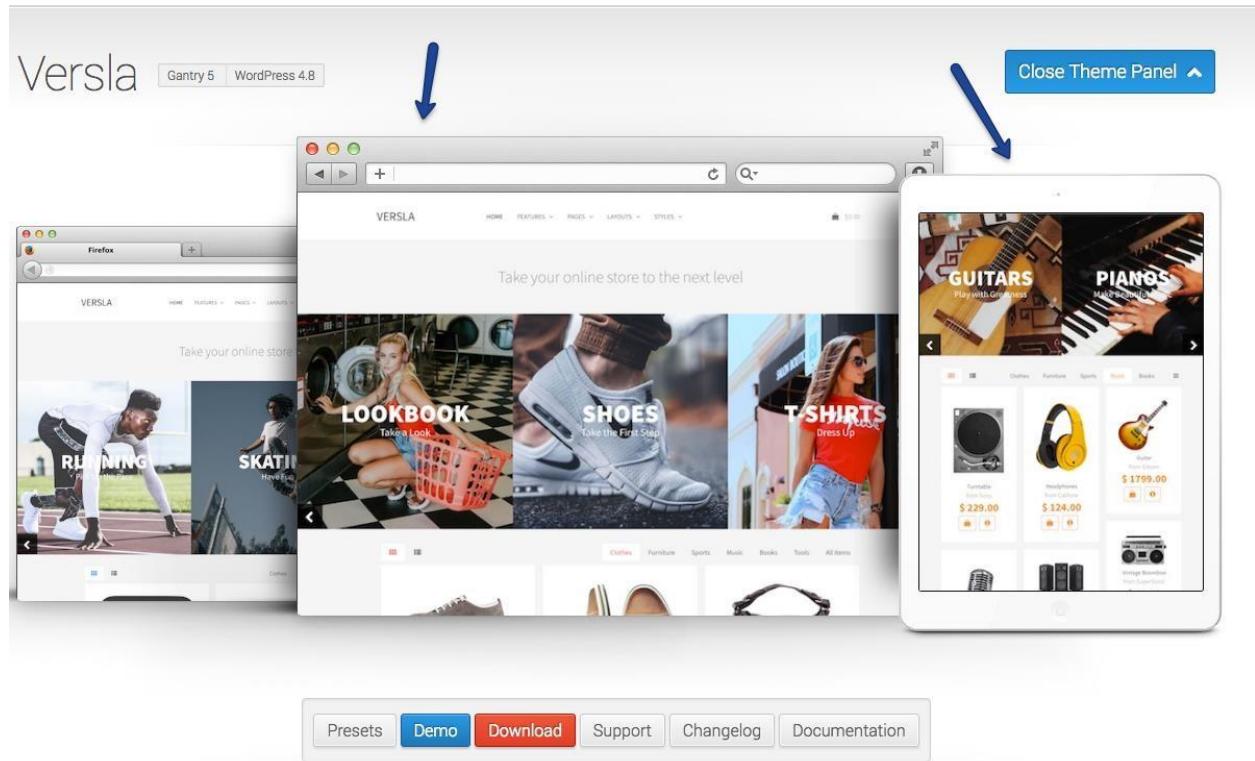


Image: <http://www.rockettheme.com/wordpress/themes/versla>

Increase site speed

A user accessing your website via a mobile device often has a slower internet connection than the desktop users because 3G network takes more time to load content. This means that site speed is even more important concerning mobile users. Browsing and buying online through a mobile device is more “on-the-go” action, and it is expected to be done rather quickly. Therefore, your website needs to load fast to provide good user experience and to keep the users on track until they have eventually completed the purchase.

Try to optimize the site speed using these practices:

- Reduce the number of redirects
- Use tools such as [PageSpeed Insights](#) to detect issues that reduce the speed of the mobile site
- Optimize images

Video format

Mobile devices have certain limitations when it comes to how they access and present content of the internet, and flash videos are part of these limitations. Most mobile devices will not

support this format. Not only does this lead to a poor user experience, but it also affects sales negatively.

Having in mind that videos are a prevalent type of content and can increase the performance of your website, you should not avoid using them. To fix the issue, use HTML5 video instead. It will ensure that the video is played correctly on mobile devices, thus not harming the mobile user experience.

Design with mobile users in mind

When you are creating a mobile-friendly website, have mobile users in mind and how they interact with the site. Make options tappable, not clickable. Mobile users tap with their fingers which are less precise than a mouse cursor, which means that the buttons should be bigger on a mobile website to make them more responsive.

It is also helpful to have the most important things centrally located. This ensures the highest level of visibility, and this is particularly important when it comes to CTA buttons.



HubSpot mobile homepage

Image: <https://www.hubspot.com/>

The less they need to scroll to find the button, the more likely they are to tap on it. Therefore, the most important options should be at the beginning of the page, to make them visible as soon as the user gets to the page, before scrolling down to see the rest of the content.

Based on the current trends and where they are taking us, it is expected that we will soon witness the rise of mobile-first websites. This means that in the future, we will be designing websites for mobile first, and then we will adapt them to a desktop view. It is only natural to expect this, as the number of mobile users is constantly on the rise.

For e-commerce businesses, it means that mobile optimization, as well as overall mobile strategy including promotion, content creation, and mobile website analytics, is going to become an essential part of e-commerce SEO.

12

Exploring Online Marketplaces

12. Exploring Online Marketplaces

Although e-commerce as an industry is associated with having an online store, there is actually more to the story. On the one hand, there are sellers that create their online stores hosted on their websites. However, there is always an option of using online marketplaces for selling products. In fact, some sellers choose to pursue sales success on marketplaces exclusively. Regardless if you choose an online shop or online marketplaces, knowing the benefits and possibilities online marketplaces provide is certainly helpful.



Image: https://unsplash.com/photos/z55CR_d0ayg What

is an online marketplace?

An online marketplace, or simply a marketplace, is a type of e-commerce website specialized in the sales of product or services provided by the multiple third parties. In this case, there is no single seller behind the marketplaces, but instead, the marketplace merely becomes a platform that is supposed to unite multiple sellers and buyers, usually on the global level. In a way, the marketplace is an intermediary between sellers (also called merchants) and buyers (also known as customers).

Online marketplace vs. e-commerce website

Even though an online marketplace is a type of e-commerce, there are still differences between this kind of business model and an e-commerce site, which has been the topic of this ebook so far.

Hosting

The first distinction is the hosting of the online store. With an online marketplace, you just register an account on the platform, and you can start selling online. You will usually have to go through a particular process to verify your account, integrate a credit card, or provide other details about your business, but once that is all over, you can start creating product pages. After that, the pages will soon be published, and the products (or services) will be available to the customers. There is no process of buying a domain, a hosting plan, designing a website or integrating payments. Also, there are no costs associated with these, so site maintenance costs are non-existent when it comes to the online marketplaces, as opposed to an e-commerce website.

Payment

Despite the fact that there are no hosting costs, online marketplaces still have to earn their income to be able to provide this platform, and they do this through fees. Each online marketplace has its own payment system, which collects some kind of fee for using the service. This fee can either be paid in advance (as a sort of membership), or introducing fees for different services which include some of these:

- Sellers' fee
- Buyers' fee
- Bank transfer fee
- Chargeback/Return fee
- Currency conversion fee
- Technology support

The marketplace charges its usage through these payments. The fees and other rates are usually available online, so you can see the exact terms before you join a marketplace.

Competition

Another significant distinction between using an online marketplace as opposed to an ecommerce website is direct competition. You always have competitors in the e-commerce business, but when selling on an online marketplace, your competitors are selling on the same

platform. You use the same product page design available on the marketplace, and you are usually part of the same payment system.

It might become harder to differentiate yourself from the competitors this way, but you still have to try to do so. For starters, follow the best practices for optimizing a product page. Even though it is not your website, product page guidelines are still valid. So think about keyword usage, elaborate description, competing images, etc. Make your product pages distinguished and recognizable using optimization methods and branding (add the logo to the images, or use a unique seller's name).

Customers

Regarding customers, there are also differences. An online marketplace usually has its base of clients. A marketplace is usually popular among online users and thus attracts them with its reputation. This means that once you start selling through a marketplace, you already have a certain base of online customers you will reach. They use the website's search engine to look for products and browse based on different criteria using available filters.

This is an entirely new way of reaching customers because with an online store you try to reach consumers through a search engine. It is more difficult to build trust, especially having in mind that the customers are supposed to provide you with sensitive details, such as credit card information. When they buy through a marketplace, they usually already have an account (or are required to create one for the first purchase). They have trust into the whole payment system due to the reputation of the online marketplaces.

Selling on Amazon

One of the most popular marketplaces is Amazon. It offers two types of plans. The Individual plan comes with no monthly fees, but instead, the platform collects \$0.99 per each item sold, plus referral and variable closing fees. The plan is perfect for the users selling fewer than 40 items per month. On the other hand, there is a Professional plan for those selling more items per month. The Professional plan includes a monthly subscription fee, currently priced at \$39.99, plus referral and variable closing fees.



Why Sell on Amazon?

Since 2000, Selling on Amazon has been helping individuals and businesses increase sales and reach new customers. Today, more than 40% of Amazon's total unit sales come from third-party selection. Consider all the benefits of Selling on Amazon, then choose a selling plan and find out how selling on Amazon works on the [How it Works page >](#)

All sellers can list products in more than 20 categories. Professional Sellers can apply to sell in at least 10 additional categories. [Learn more about product categories >](#)

[Sell as a Professional >](#)

You plan to sell more than 40 items a month
\$39.99/month + other selling fees
[What can I sell as a Professional?](#)

[Sell as an Individual >](#)

You plan to sell fewer than 40 items a month
\$0.99 per sale + other selling fees
[What can I sell as an Individual?](#)

To calculate the exact fees and costs of selling through Amazon, check out this page for [Amazon selling fees](#).

What to sell?

Before you decide to start selling on Amazon, you should decide what you are selling. Products on Amazon are divided into two categories. Open categories include the products that can be listed without any permission from Amazon. The products within this category can be used, but some categories within this group only allow new products. Closed categories on Amazon require approval.

When a category requires approval, it means that the product can be listed only with the permission from Amazon and this permission can be requested only by sellers with a Professional selling plan. The reason why there are the limitations here is the fact that Amazon needs to make sure that the products meet the standards. Some categories have special requirements, and some are not even accepting new sellers at all. To make sure where you stand when it comes to selling on Amazon, check this link for more information: [Amazon Categories](#).

Product Category	Types of Products	Conditions Allowed	Approval Required
 Amazon Device Accessories	Amazon Devices Accessories	New, Refurbished, Used	No
 Amazon Kindle	Kindle Devices Accessories	Used only	No
 Automotive & Powersports	Parts, Tools & Equipment, Accessories	New, Refurbished, Used, Collectible	Approval required. Available to Professional sellers only. Requirements Contact us
 Baby Products (Excluding Apparel)	Nursery, Feeding, Gear	New only	No, but may be required for holiday selling!
 Beauty	Fragrance, Skincare, Makeup, Hair Care, Bath & Shower. See also Health & Personal Care.	New only	Approval required. Available to Professional sellers only. Requirements Contact us
 Books	Books, Calendars, Card Decks, Sheet Music, Magazines, Journals, Other Publications	New, used	No, but all media items must ship within two business days of the date the order confirmation is made available to you.
 Business Products (B2B)	Business-relevant products across multiple categories. Special pricing features to target business customers.	New, Refurbished, Used	Available to Professional sellers only. Learn more
 Camera & Photo	Cameras, Camcorders, Telescopes	New, Refurbished, Used	No
 Cell Phones	Phones	New, Used, Refurbished, Unlocked	No, but must meet certain requirements (Seller Central sign-in required)
 Clothing & Accessories	Outerwear, Athletic Wear, Innerwear, Belts, Wallets	New only	Approval required. Available to Professional sellers only. Requirements Contact us

How to start?

Visit the Amazon homepage and find the “Sign in” button. The drop down menu will also have an option to click on if you are a new customer, which will take you to the page where you will create your own Amazon account.

You will need:

- Name
- Email address
- Password



The image shows the 'Create account' page from the Amazon website. At the top is the Amazon logo. Below it is a large input field for 'Your name'. Underneath are fields for 'Email' and 'Password' (with a note 'at least 6 characters'). There is also a field for 'Re-enter password'. A yellow button at the bottom right says 'Create your Amazon account'. Below the button, a note states: 'By creating an account, you agree to Amazon's Conditions of Use and Privacy Notice.' At the very bottom left, there is a link 'Already have an account? Sign in ›'.

Once you are registered, it is time to list the products. There are two ways to do so:

- List products that are already on Amazon – In this case, the description is already available, so you can only specify how many products you have to sell, condition and shipping options
- List products that are not yet on Amazon – You will need to list all product attributes (title, description, etc.)

When you finish creating the listing, your products become visible to the Amazon customers. One way to make sure you increase the visibility of your products is to provide:

- Complete and detailed description
- High-quality images
- Accurate description and product details

Shipping and getting paid

As soon as a customer places an order, Amazon notifies you. There are two ways to handle shipping.

Ship yourself

In this case, you will be handling shipping yourself. There will be shipping costs involved, which are usually added to the product price so that the customers will be charged more.

Fulfillment by Amazon (FBA)

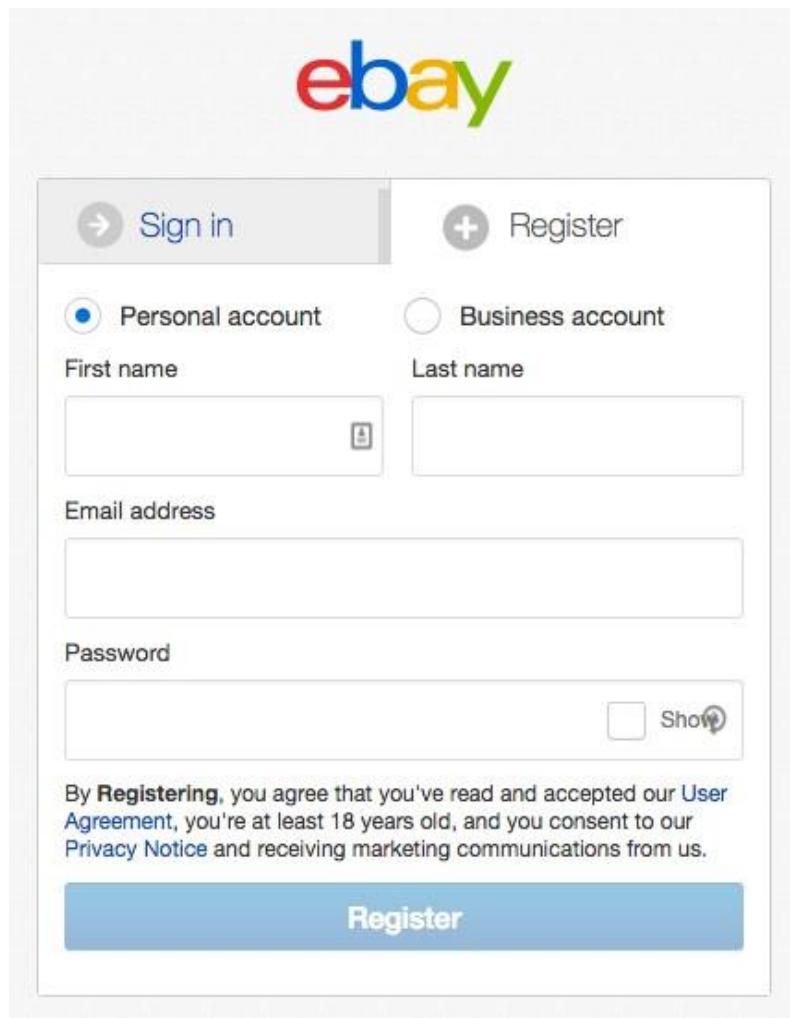
The second option is using Amazon's service which is trusted by the customers and offers perks such as free two-day shipping, free shipping or other benefits. Amazon has fulfillment centers where your inventory is stored and managed online. With FBA, Amazon handles picking, delivery, customer service and returns, which is why this service is top-rated among customers who want a trustworthy handler to be responsible for their deliveries. Products that are part of FBA program have the Prime logo next to it. This service does increase your monthly budget for selling on Amazon because you will have to pay for the storage space and the orders Amazon fulfills for you.

The final part of the process is getting paid. Amazon sends payments to your bank account which is linked to the Amazon account. Payments are sent at regular intervals.

Selling on eBay

eBay is another popular marketplace with a huge base of online shoppers you can reach as soon as your product is listed on the website. You start by joining the platform using the option "Register". You will need to provide your name, email address, and a password. Besides creating an account, you will need to meet these requirements to start selling on eBay:

- Create a seller account
- Confirm your phone number to verify your identity
- Provide a valid credit card, debit card, or bank account
- Provide information on how you will pay seller fees
- Select the payment methods you will accept
- Make your Feedback profile public



Start selling

Determine which the products you would like to sell and make sure those are legal to sell on eBay (here is a [list of prohibited items](#)). You will then choose the selling format (Auction or Buy It Now). Select a category, upload the listing and determine the price.

There are a couple of ways you can improve your chances of selling through eBay:

- Provide high-quality images of the product
- Use listing upgrades to customize the appearance of your listing
- Set up your Q&A
- Communicate with the buyers
- Leave the buyer feedback
- Have a clear returns policy and issue refunds promptly

- Send products immediately with appropriate packaging to protect the product during transport
- Use selling tools to save time and automate seller tasks
- Maintain buyer satisfaction to get great feedback

It is also helpful to check out these documents to further explore your possibilities of becoming a seller on eBay.

- [Sell an item - Getting started](#)
- [Seller's checklist](#)
- [Tips for successful selling](#)
- [The rules for sellers](#)

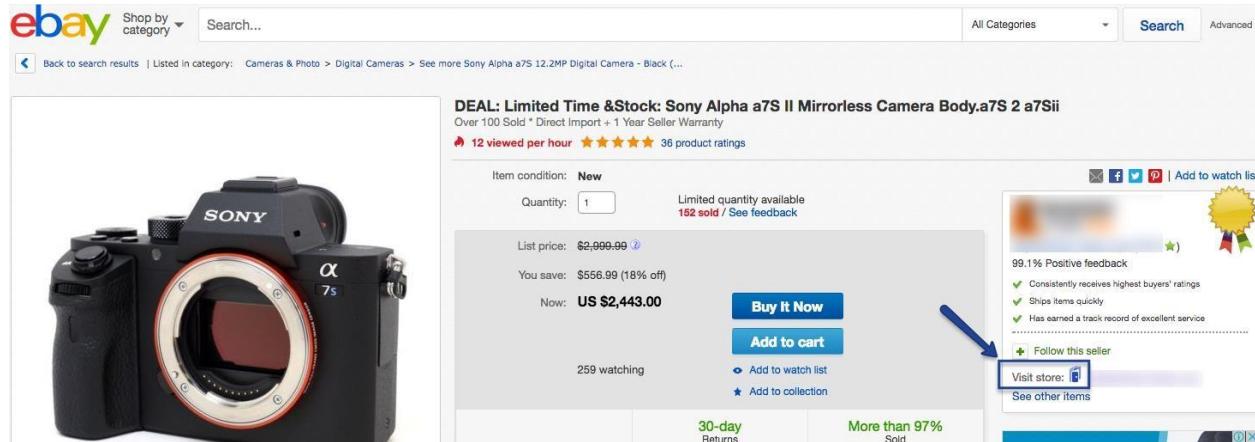
eBay allows listing up to 20 items a month for free, after which you will pay a £0.35 fee for extra items you list. This fee excludes Classified Ads listings and items listed in the Cars, Motorcycles & Vehicles category.

When you sell an item through the platform, you pay 10% of the final transaction value, including postage, but this final value will never be over £250. There are optional listing upgrades which require an extra fee.

When it comes to an eBay store, the basic shop subscription starts at £19.99 per month, with 100 listings available and 8% final value fee (instead of 10%).

An eBay store

An eBay store is a perfect solution if you want to use this marketplace as your go-to ecommerce solution. Unlike regular seller's account, the store offers marketing and merchandising tools, customization features and branding options to maximize the success of your business. Store sellers can even create customized headers below the standard eBay store headers and completely customize the store home page using HTML. A "Visit store" tag next to a seller's user data indicates that the seller has an eBay Store.



The requirements you will need to meet for opening an eBay store are the following:

- Having an eBay seller's account
- Using a verified PayPal account
- Not having “below standard seller performance” rating which can be visible in your Seller Dashboard

If you fulfill these conditions and want to set up an eBay store, you can do so using “Subscriptions” option in your Account tab. You will then click on the option “Manage My Store” to start designing the store and listing items.

eBay stores have different levels which affect subscription price and other fees. To calculate the monthly subscription value, you can provide more information about your average monthly sales and system will show estimates for the store subscription.

The screenshot shows the eBay Fee Illustrator page. On the left, there's a form for entering average monthly sales data, including fields for Auctions and Fixed price*. To the right is a chart titled "eBay Store Subscriptions" comparing fees across four store levels: No Store, Basic, Premium, and Anchor. The chart includes categories like Subscription fee, Free-insertion-fee listings, Auction insertion fees, Fixed price insertion fees, Auction final value fees, and Total estimated fees.

	No Store	Basic	Premium	Anchor
Subscription fee	-	-	-	-
Free-insertion-fee listings	-	-	-	-
Auction insertion fees	-	-	-	-
Fixed price insertion fees	-	-	-	-
Auction final value fees	-	-	-	-
Fixed price final value fees	-	-	-	-
Total estimated fees:	-	-	-	-

Image: <http://www.fees.ebay.com/feeweb/feeillustrator>

Other marketplaces

These two platforms are considered the leaders in the online marketplace industry, mainly due to their immense popularity worldwide and an impressive number of buyers. However, the possibilities for using marketplaces for an e-commerce business do not end here because there are many more marketplaces you can explore.

- [Etsy](#)— Etsy is a peer-to-peer website, which is mainly focused on handmade items, vintage products, and supplies.
- [Bonanza](#)— A marketplace offering a variety of products from women's fashion to collectibles and art.
- [CrateJoy](#)— If you sell subscription boxes, this is the perfect website for you.
- [Vide Dressing](#)— This platform is specialized in selling high-end fashion.

- [Flipkart](#)– This e-commerce platform is India's biggest online store selling a great variety of products, such as fashion, books, sports equipment, home decor, furniture, etc.
- [Alibaba](#)–As a top marketplace in China, Alibaba represents in the Asia market, what Amazon is for the US.

The bottom line is that marketplaces offer a lot of possibilities, especially when it comes to opportunities to reach customers. Once you join a platform, your products are instantly available to a huge community of buyers, while your own e-commerce website needs more time to get visitors and actually convert them.

Apart from fierce competition, there are those additional subscription costs and other fees you will have to pay to be able to offer your product on a marketplace. However, this can become quite a cost-effective solution for you because in reality, running your own e-commerce website is not free either. One way to decide if you want to work on marketplaces, and which one to choose is to consider your own business and product you have to offer. Analyze both advantages and disadvantages and how those would reflect your business. Explore possibilities and limitations that come with this strategy to determine if this is an approach you want to pursue.

13

Understanding Customers

13. Understanding Customers

E-commerce is an industry which revolves around people. Those people are online users, and they are often referred to as customers. They can be both previous and future customers, as well as those who are only interested in your brand. However, to make sense of the approach and to fully understand the relationship with the online users and how that relationship can help in growing your e-commerce business, it is best to segment the online users into several groups of customers.



Image: <https://unsplash.com/photos/yqBKaF1KecM>

Types of customers

Being aware of the different customers you will encounter, helps you with defining your goals and actions in a way to get the most out of the relationship and keep growing the customer base. Different customers require a different tactic or an approach. Understanding customers and their needs help you optimize and provide a better experience for them, which, in the long run, has tremendous benefits for your online business, starting from word-of-mouth marketing, to recurring sales.

Target group

Target group represents the group of people you plan on reaching with online promotion of your business. Those are the online users who are potentially interested in your business due to some reason, such as the fact that they are within a particular age group, that they are male or female, or even based on the fact that they have searched for a related topic recently. When you first start setting up your business and plan promotional activities, considering the target group is helpful because it allows you to:

- Create highly targeted campaigns which perform better
- Design your website (product and/or service) in a way that it appeals to the target group
- Shape your entire business strategy with those users in mind
- Anticipate the need the online users will have and provide a solution (answer or help).

When it comes to defining the types of consumers, the target group is the largest group.

Potential customers

Unlike the target group, the potential customers are within a smaller group which includes people interested in your brand. Potential customers are members of the target group who are more likely to become your customers due to a particular action such as:

- Visiting your website
- Signing up for the newsletter
- Placing the products in the cart
- Sharing your site link on social media
- Sending an inquiry through your website form
- Commenting on your website/blog content
- Following you on social media
- Interacting with your brand on social media (liking, commenting, sharing, etc.)

In general, potential customers show greater interest in your brand. For you as a business, this means an opportunity. It is an opportunity to start the interaction and establish a certain relationship with the potential customers with one goal in mind - turning these potential customers your first-time customers.

First-time customer

When someone buys from you for the first time, that person becomes your first-time customer. Once you recognize a potential customer, your goal is to complete a set of actions that will help you convert that visitor. To improve the chances of achieving this goal, you could use options such as lead nurturing, interaction with the customers through social media, etc. Offering special discounts or free trials for the first-time customers is also proven to be quite useful.

Processing the sale does not mean that the process is over. Keeping the customers loyal to your brand demands as much work as gaining the new ones. In fact, it is generally known that it can be more cost-effective to keep the existing customers than to organize activities that will enable you to attract new ones who have not had prior contact with your brand.

Finally, great experience through exploring your products and completing the purchase is what will leave a positive impact on the first-time customer and eventually help you turn a first-time customer into a recurring customer.

Recurring customer

A recurring customer is a customer that has completed several purchases. When we consider the whole customer group, this one is the smallest. It includes a portion of the first-time customers, because not all of them will go back and buy from you again. The obstacles that might reduce the chances of gaining recurring customers include:

- Poor website performance
- Low-quality product
- Not meeting customers' expectations (regarding quality, size, durability, functionality, etc.)
- Negative experience with customer service
- No efforts in engaging and keeping the customers loyal

Maintaining good customer relationship, apart from providing quality product or service, is a crucial factor for gaining recurring customers. The most commonly used channel of communication for this purpose is an email campaign. These types of campaigns provide great targeting options, customization, and personalization which create a much deeper bond with the customers, and thus yield great results. When it comes to email campaign ideas to engage previous customers, those include:

- Discounts for recurring customers or a certain renewal period
- Promo email for the product the customer has already purchased

- Product suggestion email based on the previous purchase
- Free shipping with a minimum purchase value

There are many more options you could consider here, which are all the topics you will focus on when you explore email marketing in more details and how this section of online marketing helps with promoting your business and generating sales.

The main reason why gaining recurring customers is often considered easier than acquiring new ones is the fact that you already have established the connection. You already have the customer's email address which enables you to initiate the next phase. On the other hand, with gaining new customers, you have to use different methods to reach your target group, which often includes a set of actions and efforts from your end to promote your business in the online world. Frequently, these methods will require payment, such as the case with search engine ads, social media promoted posts, etc.

Buyer persona

This term is often used in e-commerce, and it represents your ideal customer. Determining a buyer persona helps you with marketing, sales, product development, tailoring content, etc. Ultimately, this has a significant influence on your business and strategy you will use.

The representation of an ideal customer helps you get a deeper understanding of the persons you want to attract. The best way to shape your buyer persona is to analyze your insights. The information which can help you divide and categorize customers include:

- Demographics
- Location
- Behavior
- Patterns
- Goals

Try to analyze each of these areas and find as many details as you can about your customers. Collect all the details and try to detect the characteristics of a buyer persona, i.e. characteristics of those online users who are most likely to become your customer. These can include people within the age group, people who visit your blog, etc. Understanding a buyer persona helps to:

- Relate with customers
- Recognize their needs
- Organize your campaigns

- Improve your approach
- Create an effective online strategy

Opposite to a buyer persona, there is a negative persona. These are the customers you do not want. When you analyze the customers to identify the buyer person, you will also find out which type of customers you do not want, i.e. customers who are unlikely to buy from you. Either the product is too advanced for them, or they are a type of person who would not benefit from such product. Learning about the negative persona can help you narrow down the group of potential customers and focus only on real buyer persona.

User experience

Person's emotions and attitudes about buying and using a certain product or service are feelings collectively known as user experience (commonly abbreviated as UX). User experience includes all the emotions that start as a part of the initial interaction of the person with the brand and last until the purchase is complete. In fact, they go beyond the completed purchase, as the attitudes linger in customer's mind much later than that. These attitudes also affect the following purchases or the recommendations the customer is likely to share with other online users.

For a business, it is helpful to understand the user experience, as this gives opportunities for improvement and development. It is also a chance to identify pitfalls that might be preventing your business from going ahead.

Explore the user experience through these methods:

- Website user experience – Analyze session duration, page views, bounce rate, etc. to see if your website is user-friendly.
- Feedback from potential customers – Try to obtain feedback from potential customers to see what might be stopping them from completing the purchase, if they have any doubts, etc.
- Feedback from customers – Customers can share their impressions about the buying process and what triggered them to complete the purchase.
- Rating – While feedback demands more work to come up with, a rating is a quicker way for customers to tell you about the user experience. They can rate products, buying process, customer service, etc.

Understanding the buying process

To help you optimize user experience and improve the overall success of your business, you need to understand the buying process.

What is the first thing potential customers will notice?

They say that the first impression is the most lasting. You should have this in mind when you design your website, your promotional campaigns and in general when presenting your brand on the online market. Think about how the online visitors would feel:

- What will they see?
- What will intrigue them?
- Will there be enough information?
- Will they need to learn more?
- Does the landing page (or product page, or promotion page) fit the company image?

These are only some of the questions you can analyze to figure out the impression you leave. You could always conduct a survey to find out more details.

Who are your customers?

Knowing more about your customers helps with understanding your buyer persona, user experience and ultimately with understanding the entire buying process. The goal of all this is to make necessary adjustments and improvements to make your strategy even better.

Why do they buy from you?

Try to determine why do people buy from you. Or why they do not. This means that you should check out the website analytics and explore user behavior and spot any patterns. For example, you might have a landing page with a video which is converting more visitors than any other page on your website. So the reason for the conversions is on this page. You might also notice a large abandoned cart rate on the checkout page. This would mean that there is something that is driving them off, such as unexpected shipping costs or long delivery time. The answer to improving your strategy is to fix these issues you were able to identify as the reasons why the customers are not buying from you.

How do they buy?

Learning how people buy from you is also helpful in understanding the process. There are several aspects to consider here:

- Payment method the customers like to use as their preferred method
- The time needed for them to make a decision to buy, which can be immediately, or after some thinking
- Redeeming an offer is another way to buy from you, which is an excellent method for encouraging purchase, especially if the offer is time-sensitive

The ultimate goal is to find out more about the customers to improve your business. Your business revolves around the customers and your relationship with them. Learning more about them and how they behave, what affects their buying habits and how they go through the buying process is priceless information for a business looking for opportunities to grow its reach and success in the e-commerce world.

14

E-commerce Glossary

14. E-commerce Glossary

1. 301 redirect – 301 redirect is used for permanently moved pages to redirect traffic to the new page. Most of the link juice from the old page is passed on to the new one with this redirect.
2. 404 error page – This is the page that is displayed when the page could not be found. It is a standardized response, but the page could be customized with custom text, links to other pages or the homepage, etc.
3. A/B testing – It is the process of creating two versions of website pages and comparing their performance.
4. Ad format – Ads can be created in different formats, including text, image, video, and link.
5. Affiliate marketing – It is a segment of online marketing where merchants create an affiliate program for affiliates to join. Affiliates are paid a commission based on their success in promoting the merchant and referring customers.
6. API – Application Programming Interface (API) is a set of functions, procedures, and tools used for creating an application or software.
7. Bandwidth – Website bandwidth refers to the amount of information that a website can handle.
8. Bounce rate – Bounce rate is a metric that shows the percentage of visitors who left the website after viewing only one page.
9. CC License – A Creative Commons (CC) license is a public copyright license that allows free distribution of content. This way the author of the content gives the right to share, use, and alter the published content.
10. CMS – Content Management system (CMS) is an application used for creating and managing digital content. It is a platform on which you create a website.
11. Conversion – A conversion is an action that you mark as desirable. For example, when a site visitor becomes a customer, you achieve a conversion since the desired action, in this case a sale, is complete.
12. Cookies – A piece of data from a website stored on the user's computer.

13. Crawler – A search engine crawler is a program that browses the internet to collect data and create the index.
14. CRM system – Customer relationship management (CRM) system is a software used for management and analysis of the customer-related data.
15. CTA button – Call-to-action (CTA) button is the button that you want the users to click on to achieve a conversion. A CTA invites the users to click on it.
16. Dedicated server – Dedicated server is a rented server where you host your website. It is assigned only to your company, as opposed to a shared server where multiple websites can be hosted on one server.
17. Domain extension – A domain extension is an internet category added to the domain name after the dot (.com, .org, .edu, .net, etc.)
18. Email marketing – Email marketing is a type of online marketing focused on promoting your brand through email campaigns.
19. External link – External links are links pointing to a source that is not hosted on the same domain. When you link to another website from your own, this is considered an external link.
20. Flash video – This is a file format for delivering digital video content.
21. Follower – A person who chooses to follow your online activity, through social media, blog, forum, etc.
22. Google AdWords – It is the program provided by Google used for creating and managing search engine ads.
23. Guestpost – Guest post is an article designed to appear on another website instead on the author's site. It is a strategy in online marketing which is often used for increasing traffic and influence in the online community.

24. Hosting Website hosting or web host is a service enabling you to create the website and make it available on the internet.
25. HTML – Hypertext Markup Language (HTML) is a standardized system for tagging text files on the web pages.
26. HTML5 – This is the latest version of the Hypertext Markup Language (HTML) used for presenting digital content.
27. HTTPS – Hyper Text Transfer Protocol Secure (HTTPS) is a protocol that sends data from the browser to the website. It is a more secure version of the HTTP protocol that is still used.
28. Inbound link – An inbound link is a link published on a third party website that links to yours.
29. Index – To index a website means to collect information about the website and make them available once the relevant search query is performed in the search engines.
30. Influencer – An influencer is an individual who has the authority, knowledge, and position to persuade people. Influencers have a significant following on social media as well as on their website/blog.
31. Internal link – This is a type of link that points to a page that is hosted on the same domain. A link from one product page to a related product page is an example of an internal link.
32. Keyword – A word or a phrase that is considered to have great significance for your business is called a keyword. Keywords are determined through keyword research, and they are used when optimizing the website.
33. Keyword competitiveness – This is the level of how difficult it would be to rank for a particular keyword.
34. Keyword popularity – This is a metric showing how popular a keyword is. More popular keywords are more difficult to position for.
35. Landing page – A landing page is usually an entry page. It is designed with a particular campaign in mind, and the purpose of the page is to engage and convert the visitors.
36. Link juice – This term is used to describe strength or the reputation one web page passes on to the page it is linking to in terms of SEO. It is believed that if a page is

ranked well, when the link is placed on that page, this rank or popularity will pass on to the new page as well.

37. Load time – Website page load time is the time needed for the content of the page to be downloaded and displayed to the online users. This is one of the metrics that affect the search engines when ranking the websites.
38. Long-tail keyword – It is a keyword phrase which is more specific than a single keyword. Long-tail keywords are used to narrow down the target group, and they are often less competitive.
39. Mailing list – A mailing list is a list of email addresses of your subscribers or previous customers.
40. Marketplace – A marketplace is the place where the things are sold. It connects sellers and buyers.
41. Niche – In marketing, a niche is a part of the online market which is focused a particular topic, product, or service.
42. Online marketing – Online marketing, or internet marketing, is the process of advertising products or services on the internet using online channels such as search engines, email, social media, etc.
43. Organic reach – Organic reach is the portion of online audience you reach without paid distribution.
44. Page view – The act of an online user visiting a page is referred to as page view.
45. Paid reach – This is the total number of people you reach through paid distribution using online advertising methods, such as social media ads, search engine ads, etc.
46. Payment Card Industry Security Standards Council (PCI) – This is a global forum formed to manage the evolution of Payment Card Industry Security Standard.
47. Penalty – When a website is believed to use suspicious practices and methods to mislead the internet crawlers and thus increase its ranking, this site is given a penalty by the search engines. This has an adverse impact on the ranking of that website.
48. Ranking In terms of SEO, ranking is referred to as a position a certain website has in the search engine result pages.

49. Registered trademark – It is a symbol signifying that a product or service has been registered with a national trademark office.
50. Rel=canonical tag – This is a part of HTML that is used to resolve the duplicate content issue. With this tag, the webmaster specifies which page is the preferred version.
51. Schema.org markup – It is a code placed in your website code, that enables search engines to understand the site in a better way.
52. Search engine ad – A search engine ad is a part of advertising message shown in the search engine result pages, usually above the organic results.
53. Search query – A search query is a term the online user types in the search engine.
54. Security protocol – This protocol is conducted regarding security-related functions. It is a method used to establish a secure transfer of the data from the web browser to the user and vice versa.
55. Segment – In online marketing, to segment means to create categories or groups based on particular features. For example, you segment a mailing list to create several groups of subscribers (those that already bought from your, those that live in a particular location, etc.)
56. SEO – Search engine optimization (SEO) is the process of conducting a series of actions to improve the site's ranking the search engine results.
57. SEO audit – SEO audit is the process of analyzing the website performance by examining the SEO elements such as title tag, image optimization, site speed, etc.
58. Session duration – This is the metric in website analytics showing the duration of the visit, usually in seconds.
59. Showrooming – Showrooming is the act of visiting a shop to see the product before buying it online, usually at a lower price.
60. Site speed – This metric is used to measure how fast a website page loads. It is one of the factors that affect ranking.

-
61. Sitemap A sitemap is a list of all the pages of a website. Search engine crawlers use it to detect links and figure out the site structure.
 62. Social media marketing – Social media marketing (SMM) is the process of using social networks to promote your business, connect with the customers, etc. It includes both organic and paid reach.
 63. Spam – Spam is an irrelevant or unsolicited message. This term is used in reference to email messages, but some people use it in a more general context, so any type of irrelevant and poor-quality content is considered spam.
 64. Strategy – A strategy is a plan of action, a roadmap that shows you the goals you plan to achieve, as well as how to achieve them.
 65. Subscriber – The person who willingly decides to follow your email updates via newsletter is called a subscriber.
 66. Targeting options – Targeting options are used for targeting content so that it reaches the users who are most likely interested in such content. These options can include targeting by location, age, gender, behavior, etc.
 67. Template – A website template is a predesigned set of webpages that are used to create a website which is further customized with custom texts, images, and videos. A template represents the layout of the site.
 68. Trademark infringements – It is an unauthorized use of a trademark.
 69. Traffic – Traffic, or website traffic, is the amount of data sent and received on the site. A measurement used to track traffic is called a visit, or sometimes a session.
 70. Transaction – Transaction in e-commerce is an agreement between the seller and buyer which finalizes the purchase process.
 71. URL – Uniform Resource Locator (URL) is the web address of an online resource. It points to a web page.
 72. Virtual private server – Virtual private server (VPS) is a service provided by hosting providers giving the owner of VPS access to the virtual machine used to operate the system.

73. Visibility In terms of online marketing, visibility is often referred to as the likelihood of the website being shown in the search engine results and thus being visible by the online users.
74. Webrooming – Webrooming is the term used for researching products online before buying them in a physical store.
75. Website optimization – It is the process of modifying and improving the website to increase its ranking. Website optimization is frequently called search engine optimization (SEO).
76. Word-of-mouth – In marketing, word of mouth (WOM) is the process of passing information about a certain topic, product, or business, from one person to another. It is an excellent way to get your products promoted and recommended without any promotional action from your end. In the online world, this practice is sometimes called electronic word of mouth (eWOM).

15

Questionnaire

15. Questionnaire

Questions

1. E-commerce is:

- a) Short for email commerce
- b) The transaction between a buyer and a seller in a physical store
- c) The transaction between a buyer and a seller which is done through online technologies
- d) Software for browsing the internet

2. A business plan is:

- a) A document explaining what e-commerce is
- b) A document with business goals and plans on how to reach those goals
- c) Universal guidelines for setting up an online store
- d) An analysis which includes website analytics and reports for the previous period

3. A good business name should:

- a) Have as many keywords as possible
- b) Have negative connotations
- c) Be difficult to spell
- d) Contain keywords and be relevant to the products

4. Which of these is a type of transaction in e-commerce business?

- a) Business to business (B2B)
- b) Business to customer (B2C)
- c) Customer to customer (C2C)

d) All of the above

5. What is a disadvantage customers face when buying online?

- a) No instant gratification
- b) Convenience
- c) Diversity
- d) No geographical limitations

6. When you determine to set up an e-commerce business you can:

- a) Create your own web shop
- b) Create your own web shop and join the online marketplaces
- c) Join the online marketplaces
- d) Join social media

7. What should product page contain?

- a) Product name, images, and description
- b) Terms and conditions
- c) Privacy policy
- d) Blog

8. What is the number one reason for shopping cart abandonment?

- a) Too long checkout process
- b) Website errors
- c) Requirement to create an account
- d) Extra costs such as shipping, tax, etc.

9. What happens when the products are shipped through fulfillment warehouse?

- a) You send the product yourself from home
- b) The supplies directly send the products to the customers
- c) You use a warehouse to store the inventory and ship products
- d) You hire a carrier service

10. Which of these is used to promote your e-commerce business on Facebook?

- a) Facebook page, groups, and ads
- b) Facebook messenger
- c) Google AdWords
- d) “Pin it” Button

11. What are the essentials for a business plan?

- a) Business name, website URL, and logo
- b) Content writer, marketing expert, and administrator
- c) Products and places to sell
- d) Web domain and hosting

12. Which of these is a payment gateway?

- a) CMS
- b) Stripe
- c) FedEx
- d) Unbounce

13. Which of these is not a channel for online promotion of e-commerce? a)

- Website
- b) Social media
- c) Paid advertising
- d) Web hosting

14. What does a customer have to do before the e-commerce transaction is completed?

- a) Accept the terms and conditions
- b) Return the product
- c) Participate in a giveaway
- d) Register a trademark

15. What are the benefits of having a business plan?

- a) Paid advertising and Facebook page
- b) Organization and time management
- c) Business name, website URL, and logo
- d) Special offers, discounts, and giveaways

16. What should a privacy policy document contain:

- a) Information about the refunds policy
- b) Information on what user data you collect, how you collect them and how the user data will be used
- c) Information about shipping and deliveries
- d) Product information and details

17. Which of these are promotional activities you can organize on your website to promote your e-commerce business?

- a) Paid advertising and Facebook page
- b) Organization and time management
- c) Business name, website URL, and logo
- d) Special offers, discounts, and giveaways

18. What does an SSL security standard do?

- a) It creates landing pages
- b) It establishes an encrypted link between web server and a browser
- c) It defines industry standards for payment gateway process
- d) It helps to create multiple shipping options

19. When choosing an e-commerce platform, you should look for:

- a) A platform that is enough for your business needs but can easily be upgraded to a more advanced solution once you need it
- b) The cheapest platform you can find on the market
- c) The platforms that come with free integrations
- d) The online marketplaces you can join

20. Being active on Twitter can help you:

- a) Monitor sales
- b) Optimize blog article
- c) Improve client-to-customer relationship and promote your business
- d) Schedule the activities for all social networks

21. The letters "SM" in superscript next to the brand name stand for:

- a) Social media
- b) Search engine marketing
- c) Unregistered service mark
- d) Services and products

22. Which of these content formats is rapidly growing in popularity? a)

- Image
- b) Video
- c) Ebook
- d) Blog article

23. What is Google tool that can directly help you gain visibility in the local search results? a)

- Google Analytics
- b) Google Insights
- c) Google My Business
- d) Google Trends

24. A very effective way to build an email list is:

- a) Having an unsubscribe button in a newsletter
- b) Offering exclusive content or access to the subscribers
- c) Using social media
- d) Conducting an SEO audit

25. How to make sure you get the most out of social media marketing?

- a) Conduct an SEO audit
- b) Have a subscribe button on your website and an effective landing page
- c) Create a social media strategy and be active on social media
- d) Use search engine marketing ads

26. How can you nurture the relationship with the customers?

- a) Use search engine ads
- b) Offer special discounts for recurring customers
- c) Organize a giveaway on Facebook
- d) Use Twitter ads and Promoted pins

27. What is the tool that can help you conduct an SEO audit of a website?

- a) Seoptimer
- b) SocialOomph
- c) Google My Business
- d) Buffer

28. When you analyze the performance on social media, you should pay attention to:

- a) The schedule for posting the ads
- b) Engagement level, click-through rate, and organic vs. paid reach
- c) Your website traffic
- d) Visibility in the search engines

29. What is keyword research?

- a) The process of discovering the keywords you will use in online marketing

- b) Popular words and phrases on the internet
- c) Words used in paid ads exclusively
- d) The process of creating alt text and title tag

30. Which part of the product page you should optimize?

- a) Product description
- b) Product title
- c) Product images and videos
- d) All of them

31. Site structure represents:

- a) The process of keyword research
- b) The way your website is built
- c) The paid results in the search engines
- d) Site search and site speed

32. A friendly URL:

- a) Isn't related to the page content
- b) Uses words such as *and, but, the*
- c) Is readable and matches the title
- d) Has only keywords and does not match the title

33. The focus of on-site SEO are:

- a) Links
- b) Website pages

- c) Social media profiles
- d) Email campaigns

34. How is title tag marked?

- a) HTML
- b) URL
- c) H1
- d) SEO

35. What to do with the page of an expired or discontinued product?

- a) Use 301 redirect
- b) Leave the page up
- c) Use HTML title tag
- d) Optimize it for mobile devices

36. What is an online marketplace?

- a) An e-commerce website specialized in the sales of products in mobile apps
- b) An e-commerce website specialized in the sales of products offline
- c) An e-commerce website specialized in the online sales of products provided by multiple third parties
- d) An e-commerce website specialized in the sales of products on social media

37. Which tag should you use for product variations?

- a) Title tag
- b) Rel=canonical
- c) Noindex

d) HTML

38. Link building helps with:

- a) Optimizing the website
- b) Improving ranking in the search engine result pages
- c) Optimizing search engine ads
- d) Understanding the best practices in SEO

39. Which option(s) you can use for shipping products when selling on Amazon?

- a) Only you can ship products
- b) Only Amazon can ship products
- c) You can ship yourself or use the service fulfillment by Amazon
- d) Only special agents can ship products

40. What is a buyer persona?

- a) A person who is unlikely to buy from you
- b) A person who follows you on social media
- c) A first-time customer
- d) An ideal customer

41. Which of these is a link building strategy?

- a) Mobile optimization
- b) Site structure
- c) Product page description
- d) Influencer outreach

42. How can the problem of duplicate content be solved?

- a) Noindex and canonical tag
- b) HTML tag
- c) Title tag
- d) Site structure and URL optimization

43. In affiliate marketing:

- a) Merchants pay the affiliates in advance
- b) Merchants do not pay the affiliates
- c) Affiliates earn fix monthly salary
- d) Affiliates are paid on commission

44. Which of these is used as a type of website pagination?

- a) Site speed
- b) Sitemap
- c) Scrolling
- d) Structure

45. The potential problems that can reduce the site speed are:

- a) Large images and slow server
- b) Product description and title
- c) Product variations and tags
- d) No internal links and social media integration

46. Which of these can have a negative influence on gaining recurring customers?

- a) No efforts in engaging and keeping the customers
- b) Free shipping with a minimum purchase value
- c) Placing the products in the cart
- d) Sending an inquiry through your website form

47. What is essential for mobile SEO for e-commerce?

- a) Having lots of pages
- b) Using a responsive design
- c) Designing two versions where desktop would be a primary asset
- d) Using lots of redirects

48. What are the main differences between an online marketplace and an e-commerce website:

- a) Hosting and payments
- b) Keywords
- c) SEO audit
- d) There are no differences

49. You can identify potential customers based on one of these actions. Which one?

- a) Being within a certain age group
- b) Buying from your competitors
- c) Placing the products in the cart
- d) Accessing the website via a mobile device

50. Understanding a buyer persona helps you:

- a) Provide free shipping

- b) With a responsive design
- c) Start selling on online marketplaces
- d) Create an effective online strategy

Answers

1. c 18. b 35. a

2. b 19. a 36. c

3. d 20. c 37. b

4. d 21. d 38. b

5. a 22. b 39. c

6. b 23. c 40. d

7. a 24. b 41. d

8. d 25. c 42. a

9. c 26. b 43. d

10. a 27. a 44. c

11. a 28. b 45. a

12. b 29. a 46. a

13. d 30. d 47. b

14. a 31. b 48. a

15. b 32. c 49. c

16. b 33. b 50. d

17. d 34. c

16

Conclusion

16. Conclusion

The internet has opened so many opportunities for doing business online, and e-commerce is one of the most popular ones. Not only does it require low investment, it actually is a type of business that does not require a full-time commitment, especially if you are selling a limited number of products. You just make an online presence and promote the business here and there using both free and paid methods. However, as each business, it has its positive and negative sides. If you want to achieve success, you will have to create an organized strategy which is based on realistic goals and comprehensive analysis of the market.

Advantages and disadvantages

Website maintenance, processing orders, customer services and website analytics are all the tasks you will have to handle once the store is live and running, but this is not a 9 to 5 work. You do not have to be there all the time, sitting by your desk, bound by the fixed working hours.

Instead, e-commerce offers lots of flexibility or the merchants, and this is one of the main characteristics that bring this profession into the list of top desired ones for people nowadays. It brings a level of freedom to manage your own time and work flexible hours, which is a priceless thing to have in life.

Even though this is an advantage a lot of work professionals can only dream of, running an ecommerce business also requires being online and available all the time. You always have to monitor the notifications and be ready to help customers. Since e-commerce is not restricted by location or limited work hours, you can accept the orders from anywhere at any time. For you, this means that keeping an eye on the activity and customer support emails is going to be a requirement all the time. And this is important if you want to have a successful business because good customer support is one of the best ways to attract and keep customers loyal.

Consistency and dedication

Running an e-commerce business requires a lot of consistency and dedication from your end to create a successful business. You always need to keep it professional and use an approach that is in accordance with your business goals. The first part is the initial one where you are focused on planning and setting up the online business. Besides planning, which will actually be one of the main tasks, you will also have to focus on website development, product page creation, and optimization, etc. Before you make products and services available to the customers, you will also have to do a lot of testing of the website. You should especially focus on testing of the integrations such as checkout page and payment gateways. It is crucial that everything works fine on your website because any loading error will probably turn away the potential customers.

The second part of the work is done once the store is published online and these tasks are focused on website maintenance, product promotion, and customer service. Consistency in the approach you use is necessary here as well because each social media post, each newsletter, and each ad represents your company and adds up to the public image you create for the business you run. If you want to be taken seriously, if you want to build a respectable online business, you have to be professional and dedicated to representing your company in the best possible way. After all, every online activity will be scrutinized by potential customers, and you do not want to give them any reason to doubt your credibility.

E-commerce and online marketing

Throughout the ebook, the most important tasks and goals of an e-commerce website have been highlighted. Online marketing, or one of the segments of online marketing, is mentioned on several occasions. This is because having an e-commerce business requires the usage of online marketing techniques and strategies.

You will need SEO to optimize your website. You will have to learn about website analytics to be able to analyze and improve the performance of your website. Promotion of the e-commerce website cannot be done without search engine marketing, social media marketing, email marketing and even affiliate marketing.

This means that e-commerce is a part of the system. It is an online business, and as such, it is dependent on the online marketing strategies that can bring success. To make the most out of these strategies and to be sure you are doing all that is possible to promote your e-commerce business, you should explore the topic of online marketing in more details. The focus of this ebook was to show you different aspects of running an e-commerce business, and online marketing and its types are mentioned to a limited extent, which does not mean you should stop there.

For example, if you have learned that email marketing can help you with promoting your ecommerce, find relevant resources on email marketing and then explore strategies that can be implemented to your business.

Finally, it is necessary to highlight the influence of mobile devices once again. There is no doubt that mobile user experience is one of the primary goals you need to think as the statistics in favor of mobile users are keep growing. This affects the creation of a sub-sector called mcommerce. It is left to be seen how this trend will affect selling online, but what is evident now is that mobile users comprise a significant portion of overall traffic, and as such, they cannot be ignored. You must not neglect the traffic that could help you increase the number of buyers, because, after all, that is what you are trying to do with various promotion strategies.

The bottom line is that the internet changed the idea of shopping forever. So many limitations are now extinct because you can buy goods from anywhere in the world. Even so, some

boundaries are part of this online experience such as inability to touch and feel the product before it is delivered. All of this changes the way buyers think about shopping, and this is something you, as a business owner, need to think about.

This industry is all about consumers. Understanding their needs and expectations, together with defining your business goals and strategies is going to help you with running a successful ecommerce business. Have in mind that the online world is a very dynamic environment, and to be on top of your game, you need to make industry research and website analysis your top priorities.

LECTURE NOTES
ON
MOBILE COMPUTING



GraphicEra

Deemed to be University

Accredited by NAAC with Grade A

NBA Accredited Programs in ECE, CSE & ME
Approved by AICTE, Ministry of HRD, Govt. of India

DEHRADUN, UTTARAKHAND, INDIA

PECS5301 MOBILE COMPUTING (3-0-0)

Module - I 10 Hrs

Introduction to Personal Communications Services (PCS) : PCS Architecture, mobility management, Networks signaling, Global System for Mobile Communication (GSM) System overview : GSM Architecture, Mobility management, Network signaling.

General Packet Radio Services (GPRS): GPRS Architecture, GPRS Network Nodes, Mobile Data Communication; WLANs (Wireless LANs) IEEE 802.11 standard, Mobile IP. **Module - II 14 Hrs**

Wireless Application Protocol (WAP): The Mobile Internet standard, WAP Gateway and Protocols, wireless mark up Languages (WML), Wireless Local Loop (WLL) : Introduction to WLL Architecture, wireless Local Loop Technologies.

Third Generation (3G) Mobile Services: Introduction to International Mobile Telecommunications 2000 (IMT 2000) Vision, Wideband Code Division Multiple Access (WCDMA), and CDMA 2000

Module - III 12 Hrs

Global Mobile Satellite Systems ; case studies of the IRIDIUM, ICO and GLOBALSTAR systems. Wireless Enterprise Networks : Introduction to Virtual Networks, Blue tooth technology, Blue tooth Protocols.

Server-side programming in Java, Pervasive web application architecture, Device independent example application. **Text Books:**

1. Mobile Communication: J. Schiller, Pearson Education 2.
Mobile Computing: P.K. Patra, S.K. Dash, Scitech Publications.

3. Mobile Computing: Talukder, TMH, 2nd Edition. **Reference Books:**

1. Pervasive Computing: Burkhardt, Pearson Education.
2. Principles of Mobile Computing: Hansmann, Merk, Springer, 2nd Edition.
3. Wireless Communication & Networking: Garg, Elsevier
4. Third Generation Mobile Telecommunication Systems: P. Stavronlakis, Springer.
5. The Wireless Application Protocol: Sandeep Singhal, Pearson Education

Module-I

PCS (Personal Communication System)

PCS stands for Personal Communication System. The objective of PCS is to enable communication with a person at any time, at any place & in any form. It also manages their individual call services according to their service by providing unlimited reachability & accessibility.

Sprint was the first company to set up a PCS network, which was a GSM-1900 network in the Baltimore-Washington metropolitan area in the USA. PCS promises to provide a wide range of locations and equipment-independent services to a large number of users. According to the definition given by the US Federal Communications Commission (FCC), PCS is the system by which every user can exchange information with everyone, at anytime, in any place, through any type of services, using a single personal telecommunication number (PTN).

Key factors of PCS are:

1. Reachability with respect to Location (Home, office, in public, in transit)
 1. Accessibility with respect to Device (Cellular phone, wired phone, fax etc.)
 2. Management of Service **Architecture**

Architecture consists of two parts

Radio Network

PCS users carry mobile stations (MS) to communicate with a BS in a PCS n/w. MS is also referred to as handset or mobile phone. The radio coverage of a base station is called cell. In GSM n/w each cell is controlled by BSC which are connected to MS through BS. The BSCs are connected to MSC by landlines.

Wireline Transport Network

An MSC is a telephone exchange configured specially for mobile applications. It interfaces the MSC (via BS) with PSTN. MSCs are also connected with mobility database to track the location of MS and roaming management. The databases are HLR & VLR. HLR contains the authentication information like IMSI (International Mobile Subscriber Identity), identification information like name, address of the subscriber, billing information like prepaid or postpaid, operator selection, denial of service to a subscriber etc. VLR gives information about the location area of the subscriber while on roaming and power off status of the handset.

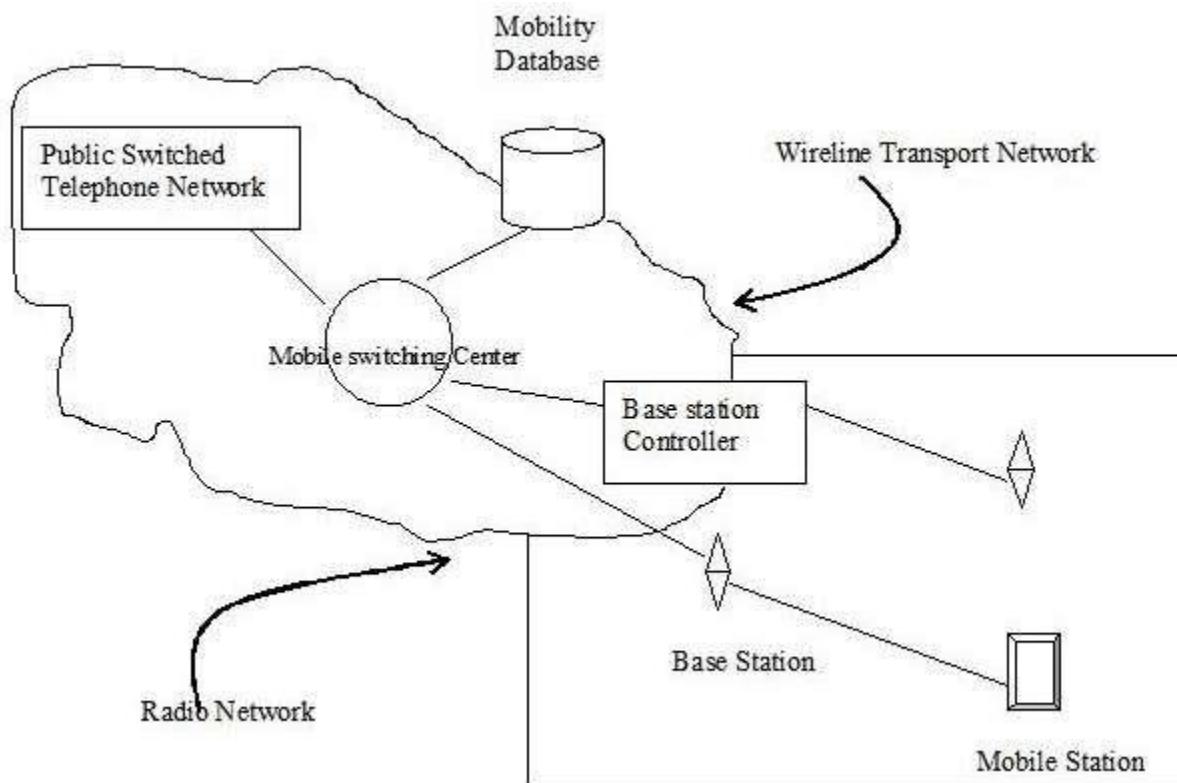


Fig 5.1 PCS network Architecture

5.4 Mobility Management

Mobility mgmt function handles the function that arises due to mobility of the subscriber.

Main objective of MM is location tracking & call set up. There are two aspects of mobility in a PCS n/w.

HANDOFF: When a mobile user is engaged in conversation, the MS is connected to BS via radio link. If the user moves to the coverage area of another BS, the radio link to old BS is disconnected and radio link to new BS is established to continue conversation. This process is called automatic link transfer or handoff. Depending on the mobility of MS, the handoff is divided into two categories:

Inter-BS Handoff/ Inter Cell Handoff:

Here MS usually moves from one BS to another BS under one MSC.

Action taken for communication:

1. The MS momentarily suspends conversation & initiates the hand-off procedure by picking a channel in new BS. Then it resumes the conversation in old BS.
2. When MSC receives that signal, it transfers the information to the new BS & sets up new conversation path to MS through that channel.
3. After MS has been transferred to new BS, it starts the conversation channel with new BS & then MSC disconnects the link with old BS.

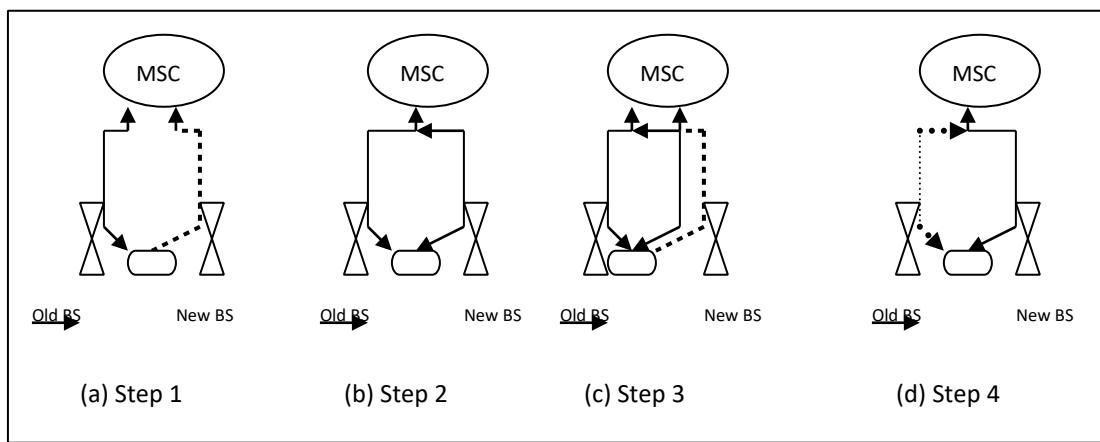


Fig 5.2 Inter-BS link Transfer

Inter-System Handoff/Inter-MSC Handoff

MS moves from one BS to another connected to two different MSCs.

Action taken for communication:

1. MSC1 requests MSC2 to perform handoff measurement on the call in progress.

2. MSC2 then selects a BS by interrogating the signal quality and sends the information to MSC1.
3. Then MSC1 asks MSC2 to setup a voice channel.
4. Assuming that a voice channel is available in BS2.MSC2 instructs MSC1 to start radio link transfer.
5. MSC1 sends the MS a handoff order. Now MS can access BS2 of MSC2.MSC2 informs MSC1 that handoff is successful.MSC1 then connects call path to MSC2.
6. In the intersystem handoff process, anchor MSC is always in call path before & after handoff.

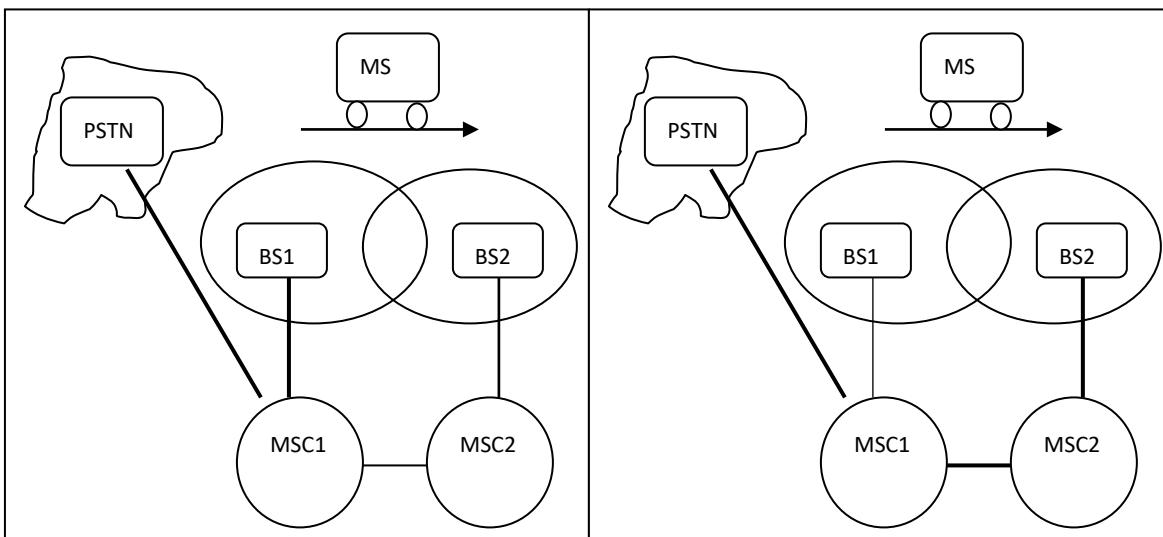


Fig 5.3 Inter System Handoff

Path Minimization- When MS moves to MSC3, MSC2 may be removed from the call path. The link between MSC1 and MSC2 is disconnected and MS connects to MSC3 directly. This process is called path minimization.

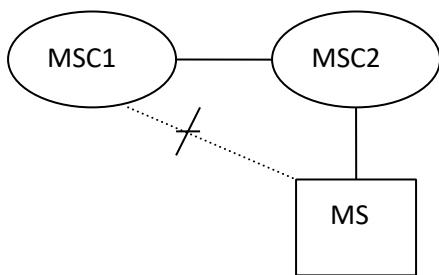


Fig 5.4 Handoff Forward

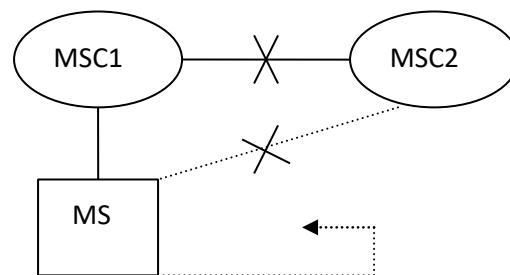


Fig 5.5 Handoff Backward

Fig

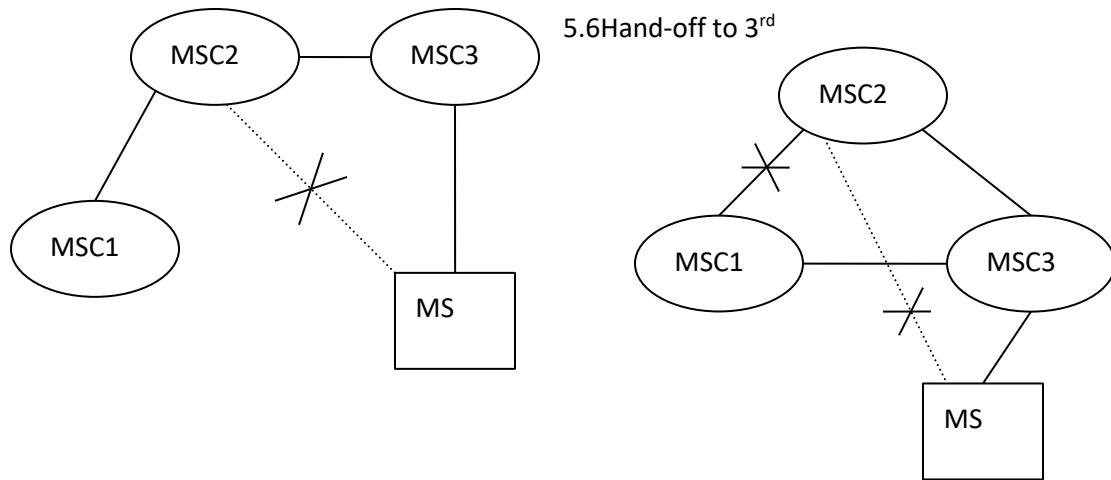


Fig 5.7 Path minimization

ROAMING: When a mobile user moves from one PCS system to another, then the system should be informed of the current location of the user. Otherwise it is impossible to deliver services.

Two basic operations are performed under roaming management.

1. **Registration (location update):** Where MS informs the system its current location.
2. **Location tracking:** Process during which a system locates MS. Location tracking is required when n/w attempts to deliver call to a mobile user.

The roaming management follows a two level strategy where two tier systems of home and visited databases are used. When a user subscribes to the services of a network, a record is created in the system's database called HLR. This is referred to as home system of the mobile user. HLR is a n/w database, where MS's identity, profile, current location & validation period is stored.

When the mobile user visits a new network other than home system, a temporary record for the mobile user is created in the VLR of visited system. VLR temporarily stores information for visiting subscribers so that corresponding MSC can provide service.

Registration Procedure includes following steps:

1. When mobile user enters into new PCS n/w, it must register in VLR of new system.
2. The new VLR informs mobile user's HLR regarding the current location & address of user. The HLR sends an acknowledgement which includes MS's profile, to the new VLR.

3. New VLR informs MS about successful registration.
4. HLR sends a deregistration message to cancel the location record of MS in old VLR. The old VLR acknowledges the deregistration.

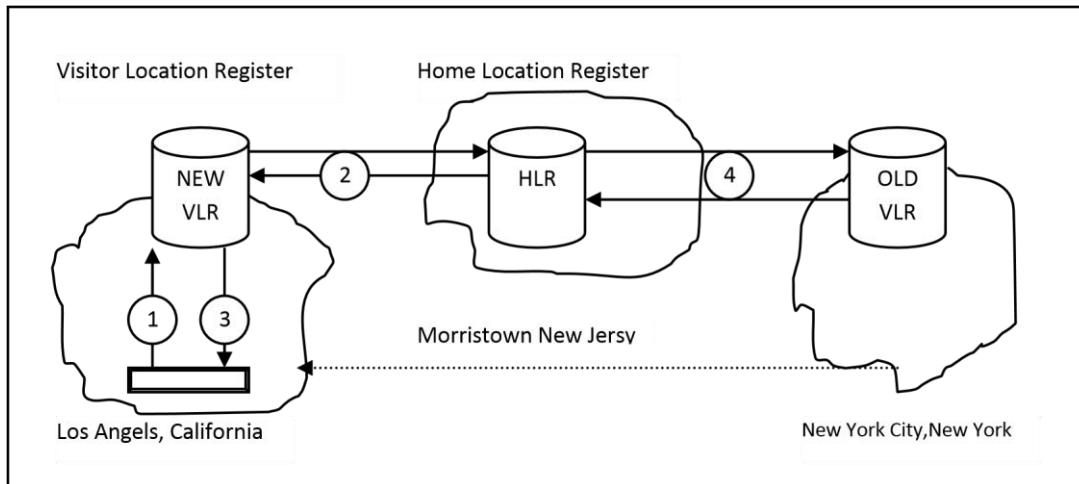


Fig 5.8 MS registration Process

To originate a call, MS first contacts with MSC in the new PCS n/w .The call request is forwarded to VLR for approval. If it is approved, MSC sets up the call to the user following the standard PSTN procedures.

1. If a wireline phone attempts to call a mobile subscriber, the call is forwarded to switch called the originating switch in PSTN. The switch masses a query to HLR to find current VLR of MS. The HLR queries the VLR in which MS resides to get a communicable address.
2. The VLR returns the address to switch through HLR.
3. Based on address, a communication link is established between MS through visited MSC.

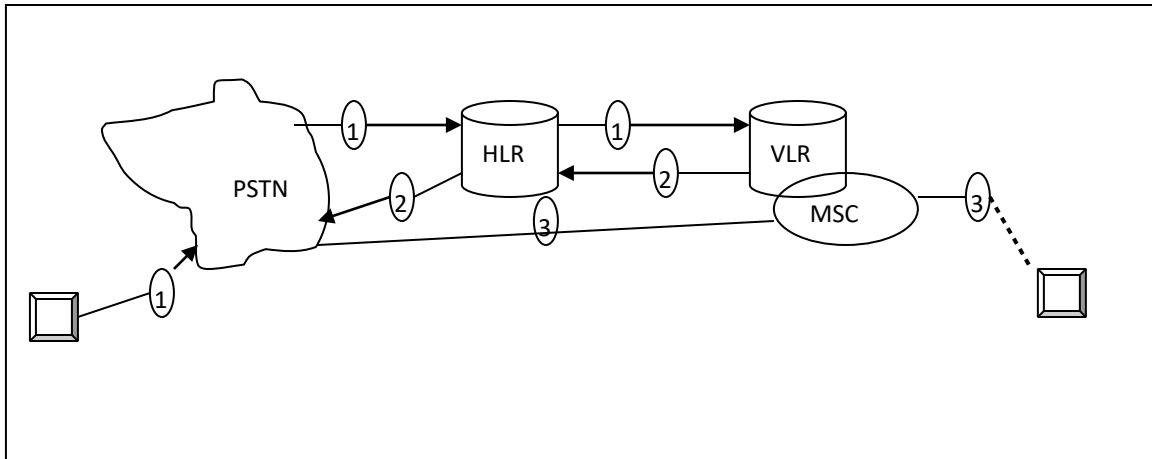


Fig 5.9 Call Delivery Procedure

GSM(Global System for Mobile Communication)

GSM is the most popular standard for mobile phones in the world. GSM (Global System for Mobile communication) is a digital mobile telephony system that is widely used in Europe and other parts of the world.

In 1982, the European Conference of Postal and Telecommunications Administrations (CEPT) created the Groupe Spécial Mobile (GSM) to develop a standard for a mobile telephone system that could be used across Europe. In 1989, GSM responsibility was transferred to the European Telecommunications Standards Institute (ETSI) and phase I of the GSM specifications were published in 1990.

The first GSM network was launched in 1991 by Radiolinja in Finland with joint technical infrastructure maintenance from Ericsson. The proposed GSM system had to meet certain business objectives:

- Support for International Roaming
- Good Speech Quality
- Ability to support handheld terminals
- Low terminal and service cost.
- Spectral Efficiency

GSM uses a combination of FDMA and TDMA. The GSM system has an allocation of 50 MHz bandwidth in the 900 MHz frequency band. Using FMA, this band is divided into 124 channels each with a carrier bandwidth of 200 KHz. Using TDMA, each of these channels

is further divided into 8 time slots. Therefore with combination of FDMA and TDMA we can realize a maximum of 992 channels for transmit and receive.

Cell: Cell is the basic service area: one BTS covers one cell. Each cell is given a Cell Global Identity (CGI), a number that uniquely identifies the cell.

Location Area: A group of cells form a Location Area. This is the area that is paged when a subscriber gets an incoming call. Each Location Area is assigned a Location Area Identity (LAI). Each Location Area is served by one or more BSCs.

GSM Architecture

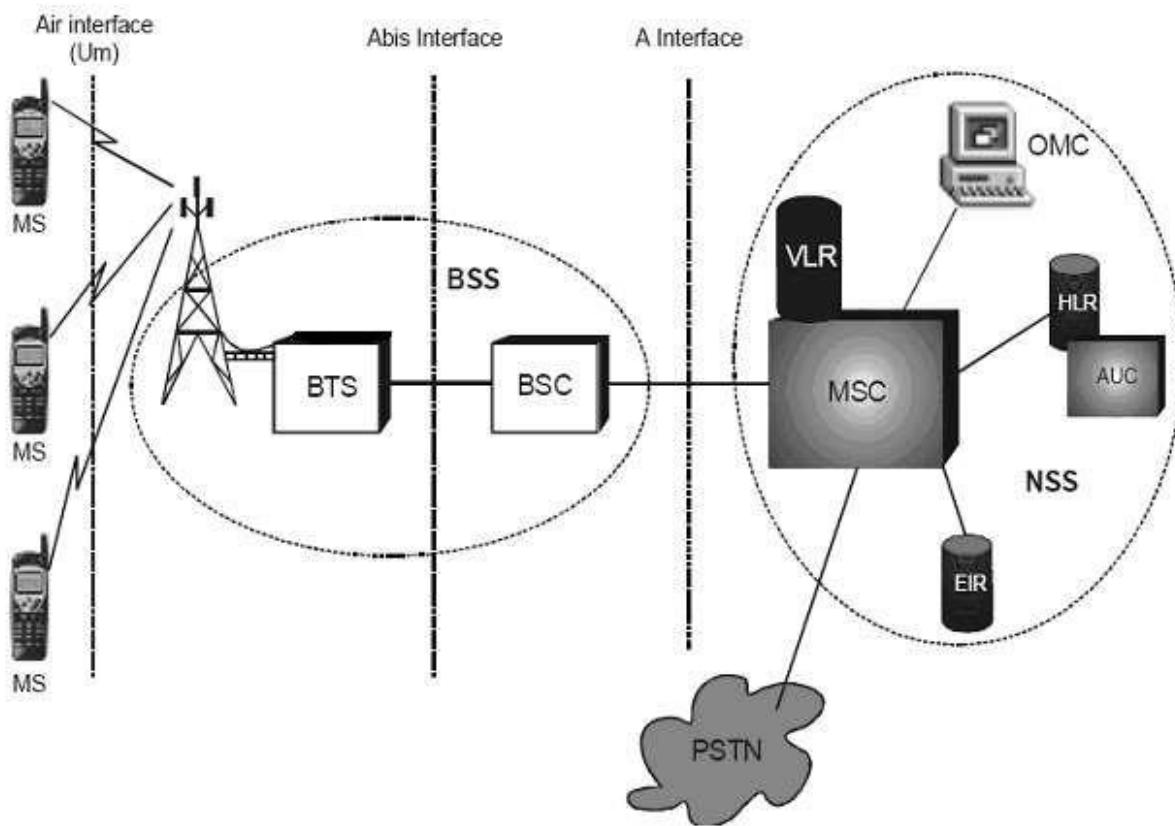


Fig 6.1 GSM Architecture overview

Abbreviations

MSC : Mobile switching center

BSC : Base station controller

BTS : Base transceiver station

TRX : Transceiver.

MS : Mobile station

OMC : Operations and Maintenance centre.

PSTN : Public switched telephone network.

BSS : Base station sub-system.

HLR : Home location register

VLR : Visitor locations register

AUC : Authentication centre

EIR : Equipment Identity Register.

GSM network can be divided into 4 groups.

MS (Mobile Station)

An MS is used by a mobile subscriber to communicate with the mobile network. Several types of MSs exist, each allowing the subscriber to make and receive calls. Manufacturers of MS offer a variety of design and features to meet the need of different market.

The mobile station consists of:

- Mobile Equipment (ME)
- Subscriber identity module (SIM)

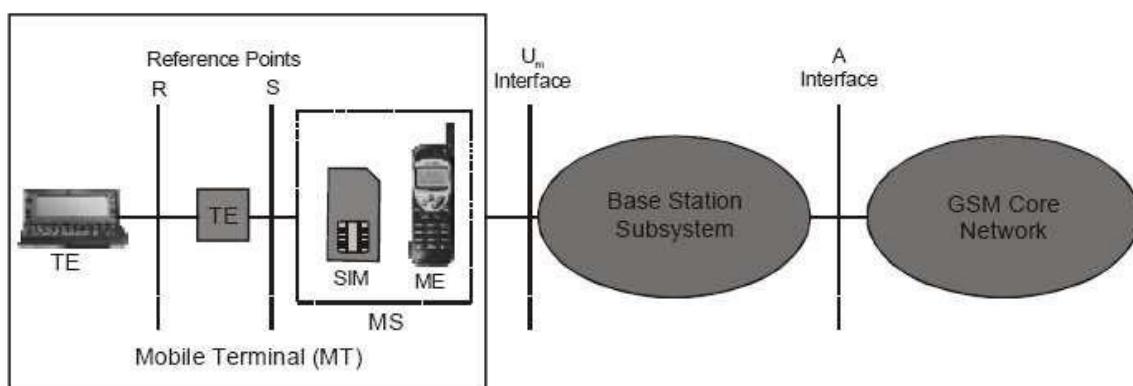


Fig 6.2 GSM Mobile Terminal

ME (Mobile Equipment)

“Cellular phone without SIM card”

The mobile equipment has a unique international mobile equipment identity (IMEI) which is used by EIR. The numbers of GSM terminal types are defined within the GSM specification. They are distinguished primarily by their power output rating. The range or coverage area of an MS is dependent on the output power capabilities and consequently

different ranges. For example, hand held MSs have a lower output power and shorter range than car-installed MSs with a roof mounted antenna

SIM (Subscriber Identity Module)

SIM card used in phones are smart processor cards. It possesses a processor and a small memory. The SIM stores permanent and temporary data about the mobile, the subscriber and the network. It contains **a serial no, PIN, PUK (Pin Unblocking Key), an authentication key (Ki), IMSI (International Mobile Subscriber Identity).**

The SIM can be plugged into any GSM mobile terminal. This brings the advantages of security and portability for subscriber. Example: Subscriber A's mobile terminal may have been stolen. However, A's own SIM can be used in another person's mobile terminal and the calls will be charged to subscriber A.

Functions of MS

Function of MS is transmission of signal from MS to BTS (using uplink) and reception of signal from BTS to MS (using down link).

BSS (Base Station Subsystem)

BSS contains two components:

- BTS
- BSC

BTS (Base Transceiver Station)

It comprises all radio equipments (e.g.: antenna, signal processing & amplifier required for transmission). It is placed in the center of a cell. Its transmitting power defines the size of a cell. It is connected to MS via Um interface and connected to BSC via Abis Interface. It manages the radio resources for BTSs. It handles & handover the radio frequency, radio channel set up from one BTS to other.

BSC (Base Station Controller)

It connects the BTS and MSC of NSS. It manages radio resources for one or more BTS. It handles and Handover the radio frequency, radio channel setup from one BTS to another.

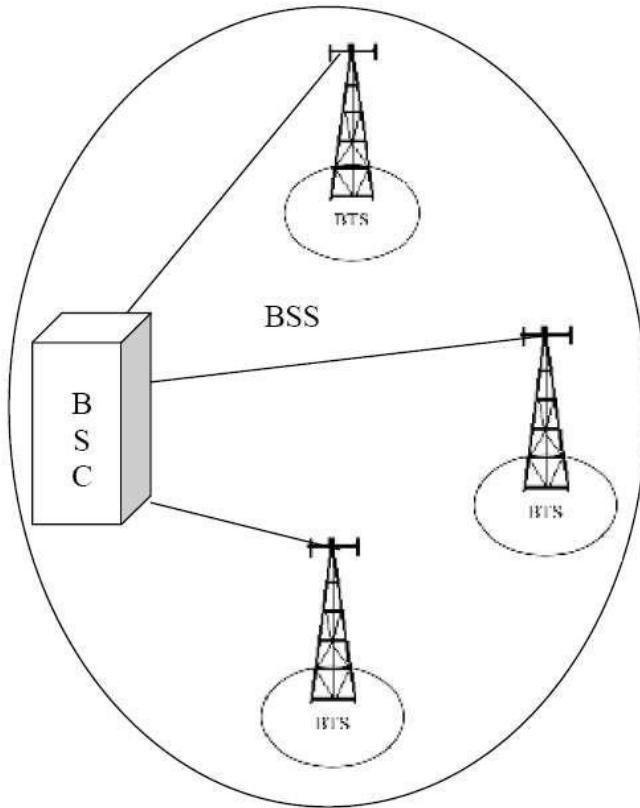


Fig 6.3 BSC & BTS Arrangement in GSM System

NSS (Network Switching Subsystem)

The NSS combines the call rotating switches (MSC and GMSC) with data base registered required to keep track of subscriber's movements and use of the system. Key elements of NSS are:

- MSC
- VLR
- HLR

MSC (Mobile Switching Centre)

The mobile-services switching centre is an exchange which performs all the switching and signaling functions for mobile stations located in a geographical area designated as the MSC area. These are high performance digital ISDN switches. It is used for connection between mobile phone to mobile phone within same network. It is used for connection between mobile phone to fixed phone within a network. It manages BSC within a geographical area.

GMSC (Gateway MSC)

Connection for another network MSC handles all the signaling needed for connection set up and connection release.

HLR (Home Location Register)

The HLR is a centralized network data base that stores and manages all mobile services belonging to a specific operator. It acts as a permanent store for a person's subscription information until that subscription is cancelled. It provides call routing and roaming facility by combining with MSC and VLR. It is considered as a Database which stores the information about the subscriber within covering area of MSC. Information includes current location of the mobile & all the service providing information, when a phone is powered off this information is stored in HLR. It is also a database but contains a temporary copy of some of important information stored in HLR. If a new MS user comes into location area, then VLR will provide relevant information by bringing it from HLR.

VLR (Visitor Location Resister)

It is a temporary storage device of GSM network. It stores subscribers' subscription information for MS which are within the particular MSC service Area. There is one VLR for each MSC service area

OSS (Operation and Support Subsystem)

It contains necessary function for network operation and maintenance.

Key Elements are

- OMC
- EIR
- AUC

OMC (Operation and maintenance center)

It is connected to different components of NSS & to BSC. It controls the traffic load of BSS.

EIR (Equipment Identity Register)

A database that contains a list of all valid mobile equipment within the network where each MS is identified by IMEI (International Mobile Equipment Identity).EIR contains a list

of IMEI of all valid terminals. An IMEI is marked invalid if it is stolen. EIR allows the MSC to forbid calls from this stolen terminal. The equipment identification procedure uses the identity of the equipment itself (IMEI) to ensure that the MS terminal equipment is valid.

AUC (Authentication Center)

It is defined to protect user identity & transmission. It is a protected database and stores a copy of secret information stored in SIM card .These data help to verify user's identity.

Network Signaling

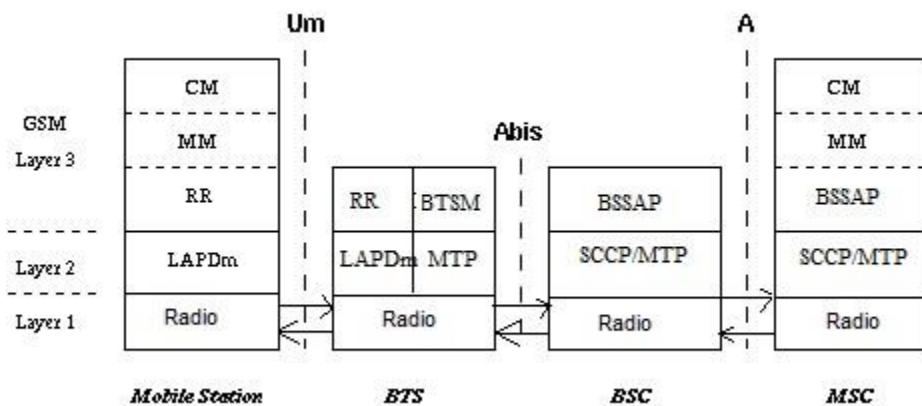


Fig 6.4 Network Signaling

Abbreviations

LAPD Link Access Procedure D-Channel Managed

RR: Radio Resource

MM: Mobility Management

CM: Call Management

BTSM: BTS MAnagement

BSSMAP: BSS Application Protocol

SCCP: Signaling Connection Control Part

The signaling protocol in GSM is structured into 3 layers.

- Layer1|Physical Layer
- Layer2|Data Link Layer
- Layer3|Network Layer

MS ↔ BTS

The physical layer between MS & BTS is called Um interface. It performs following functions

- Full or half duplex access.
- Provides TDMA, FDMA, and CDMA.
- Framing of data.

The data link layer controls the flow of packets to and from network layer and provides access to various services like:

Connection: Provides connection between two terminals.

Teleservices -Services offered by a mobile network to users like: MMS, SMS, etc

The data link layer present between MS & BTS is LAPDm (Link Access Protocol managed). LAPDm protocol describes the standard procedure in GSM for accessing Dchannel Link.

Its functions are:

- Dataflow control.
- Acknowledged / unacknowledged data Transmission.
- Address and sequence no. check.
- Segmentation.

The network layer has 3-sublayers

CM (Call Management)

Supports call establishment, maintenance, termination.

It supports SMS.

Support DTMF (Dual Tone multiple frequency) signaling.

MM (Mobility Management)

Control the issue regarding mobility Management, location updating & registration.

RRM (Radio Resource Management.)

It manages radio resources such as: frequency assignment, signal measurement.

BTS □□ BSC signaling protocols

The physical layer between BTS & BSC is called Abis interface, where voice is coded by using 64kbps PCM. The connection between BTS and BSC is through a wired network. The data link layer is LAPDm. Network Layer protocol is called BTS Management which interact with BSSAP.

BSC □□ MSC signaling protocol

Physical layer between BSC & MSC is called U interface. Data link layer protocol between BSC & MSC is MTP (Message Transfer Protocol) & SCCP (Signaling Connection Control Protocol). MTP and SCCP are part of the SS7 (Signaling System No7) used by interface A.

NETWORK layer protocols at the MSC are CM, MM and BSSAP (Base Subsystem Application Part).

GSM INTERFACES Um Interface (MS to BTS)

The Um radio interface (between MS and base transceiver stations [BTS]) is the most important in any mobile radio system. It addresses the demanding characteristics of the radio environment. The physical layer interfaces to the data link layer and radio resource management sublayer in the MS and BS and to other functional units in the MS and network subsystem (which includes the BSS and MSC) for supporting traffic channels. The physical interface comprises a set of physical channels accessible through FDMA and TDMA.

Abis Interface (BTS to BSC)

The interconnection between the BTS and the BSC is through a standard interface, Abis. The primary functions carried over this interface are traffic channel transmission, terrestrial channel management, and radio channel management. This interface supports two types of communications links: traffic channels at 64 kbps carrying speech or user data for a full- or half-rate radio traffic channel and signaling channels at 16 kbps carrying information for BSC-BTS and BSC-MSC signaling. The BSC handles the LAPD channel signaling for every BTS carrier.

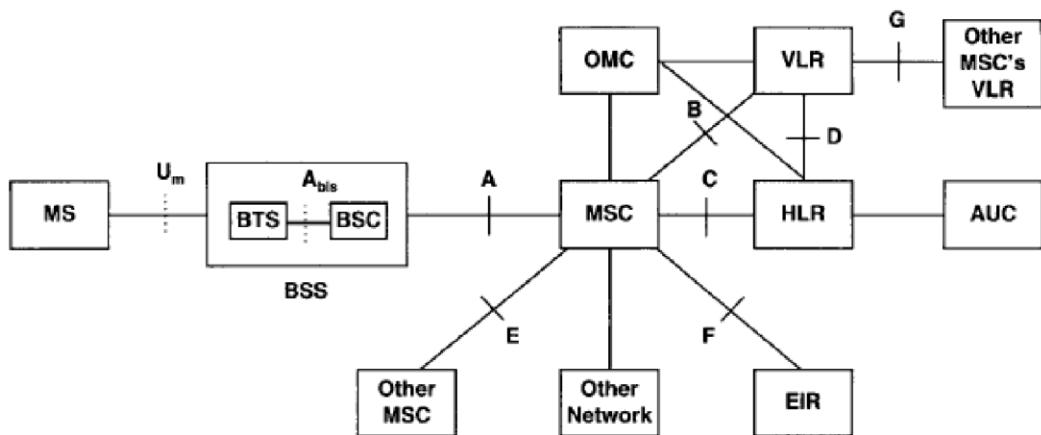
There are two types of messages handled by the traffic management procedure part of the signaling interface—**transparent** and **nontransparent**. Transparent messages are between the MS and BSC-MSC and do not require analysis by the BTS.

Nontransparent messages do require BTS analysis.

A Interface (BSC to MSC)

The A interface allows interconnection between the BSS radio base subsystem and the MSC. The physical layer of the A interface is a 2-Mbps standard Consultative Committee on Telephone and Telegraph (CCITT) digital connection. The signaling transport uses Message Transfer Part (MTP) and Signaling Connection Control Part (SCCP) of SS7. Error-free transport is handled by a subset of the MTP, and logical connection is handled by a subset of the SCCP. The application parts are divided between the BSS application part (BSSAP) and BSS operation and maintenance application part (BSSOMAP). The BSSAP is further divided into Direct Transfer Application Part (DTAP) and BSS management application part (BSSMAP). The DTAP is used to transfer layer 3 messages between the MS and the MSC without BSC involvement. The BSSMAP is responsible for all aspects of radio resource handling at the BSS. The BSSOMAP supports all the operation and maintenance communications of BSS.

Figure shows the various interfaces between the GSM entities.



GSM Channels

GSM has been allocated an operational frequency from 890 MHz to 960 MHz. GSM uses the frequency band 890 MHz-915 MHz for uplink (reverse) transmission, and for downlink (forward) transmission, it uses the frequency band 935 MHz-960 MHz. The available 25-MHz spectrum for each direction is divided into 124 Frequency Division Multiplexing (FDM) channels, each occupying 200 kHz with 100 kHz guard band at two edges of the spectrum as shown in fig.

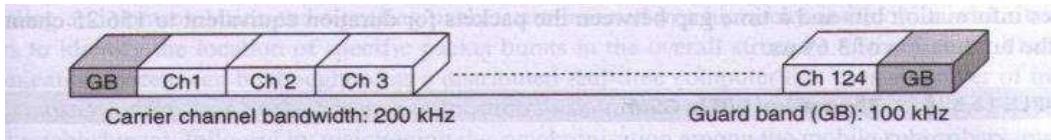


Table 11.2 Logical Channels in GSM

Channel type	Channel group	Channel	Direction
Control Channel (CCH)	Broadcast Channel (BCH)	Broadcast Control Channel (BCCH)	Downlink
		Frequency Correction Channel (FCCH)	Downlink
		Synchronization Channel (SCH)	Downlink
	Common control Channel (CCCH)	Paging Channel (PCH)	Downlink
		Random Access Channel (RACH)	Uplink
		Access Grant Channel (AGCH)	Downlink
	Dedicated control Channel (DCCH)	Standalone Dedicated Control Channel (SDCCH)	Uplink and Downlink
		Slow Associated Control Channel (SACCH)	Uplink and Downlink
		Fast Associated Control Channel (FACCH)	Uplink and Downlink
Traffic Channel (TCH)	Traffic Channel (TCH)	Half-rate Traffic Channel (TCH/H)	Uplink and Downlink
		Full-rate Traffic Channel (TCH/F)	Uplink and Downlink

The logical channels in the GSM network are divided into two principal categories: Control Channels (CCHs) and Traffic Channels (TCHs). Control channels carry signaling and synchronizing commands between the base station and the mobile station. Traffic channels carry digitally encoded user speech or user data and have identical functions and formats on both the forward and reverse link. GSM system uses a variety of logical control channels to ensure uninterrupted communication between MSs and the BS.

GSM Control Channels

There are three classes of control channels defined in GSM: Broadcast Channels (BCH), Common Control Channels (CCCH) and Dedicated Control Channels (DCCH). Each control channel consists of several logical channels that are distributed in time to provide the necessary GSM control functions.

I. Broadcast Channel (BCH)

The BCH channels are broadcast from the BTS to MSs in the coverage area of the BTS and are one way channels. The broadcast channel operates on the forward link of a specific ARFCN within each cell and transmits data only in the first time slot of certain GSM frames. The BCH provides synchronization for all mobiles within the cell and is

occasionally monitored by mobiles in neighboring cells. There are three separate broadcast channels:

1. **Broadcast Control Channel (BCCH):** This channel is used by BTS to broadcast system parameters such as frequency of operation in the cell, operator identifiers, cell ID and available services to all the MSs. Once the carrier, bit, and frame synchronization between the BTS and MS are established, the BCCH informs MS about the environment parameters associated with the BTS covering that area such as current a channel structure, channel availability, and congestion. The BCCH also broadcasts a list of channels are currently in use within the cell.
2. **Frequency Correction Control Channel (FCCH):** This is used by the BTS to broadcast frequency references and frequency correction burst of 148 bits length. An MS in the coverage area of a BTS uses broadcast FCCH signal to synchronize its carrier frequency and bit timing.
3. **Synchronization Channel (SCH):** This channel is used by the BTS to broadcast frame synchronization signals containing the synchronization training sequences burst of 64 bits length to all MSs. Using SCH, MSs will synchronize their counters to specify the location of arriving packets in the TDMA hierarchy. SCH is broadcast in Time Slot 0 of the frame immediately following the FCCH frame and is used to identify the serving base station while allowing each mobile to framesynchronize with the base station.

II. Common Control Channels (CCCH)

The Common Control Channels (CCCH) are one-way channels used for establishing links between the and the BS for any ongoing call management. CCCHs are the most commonly used control channel and are used to page specific subscribers, assign signaling channels to specific users, and receive mobile requests for service. There are three CCCH logical channels:

1. **Paging Channel (PCH):** This is a forward link channel and is used by the BTS to page or notify a specific individual MS for an incoming call in the cell. The PCH transmits the IMSI of the target subscriber, along with a request for acknowledgment from the mobile unit on the RACH.

2. **Random Access Channel (RACH):** This is a reverse link channel and is used by the MS either to access the BTS requesting the dedicated channel for call establishment or to acknowledge a page from the PCH. The RACH is used with implementation of a slotted-ALOHA protocol, which is used by MSs to contend for one of the available slots in the GSM traffic frames. The RACH is implemented on the short Random Access Burst (RAB).

III. Dedicated Control Channels (DCCH)

Dedicated Control Channels (DCCH) are two-way channels having the same format and function on both the forward and reverse links, supporting signaling and control for individual mobile subscribers. These are used along with voice channels to serve for any control information transmission during actual voice communication. There are three DCCH logical channels:

1. **Stand-alone Dedicated Control Channel (SDCCH):** This is a two-way channel allocated with SACCH to mobile terminal to transfer network control and signaling information for call establishment and mobility management. The SDCCH ensures that the mobile station and the base station remain connected while the base station and MSC verify the subscriber unit and allocate resources for the mobile. The SDCCH is used to send authentication and alert messages as the mobile synchronizes itself with the frame structure and waits for a TCH.
2. **Slow Associated Control Channel (SACCH):** This is a two-way channel associated with a TCH or a SDCCH and maps onto the same physical channel. The SACCH is used to exchange the necessary parameters between the BTS and the MS during the actual transmission to maintain the communication link. Each ARFCN systematically carries SACCH data for all of its current users. The gross data rate of the SACCH channel is half of that of the SDCCH. On the forward link, the SACCH is used to send slow but regularly changing control information to the mobile subscriber. The reverse SACCH carries information about the received signal strength and quality of the TCH.
3. **Fast Associated Control Channel (FACCH):** This is a two-way channel used to support fast transitions such as a hand-off request in the channel when SACCH is not adequate. The FACCH is physically multiplexed with the TCH or SDCCH to provide additional support to the SACCH. FACCH is not a dedicated control

channel but carries the same information as SDCCH. FACCH is a part of the traffic channel, while SDCCH is a part of the control channel.

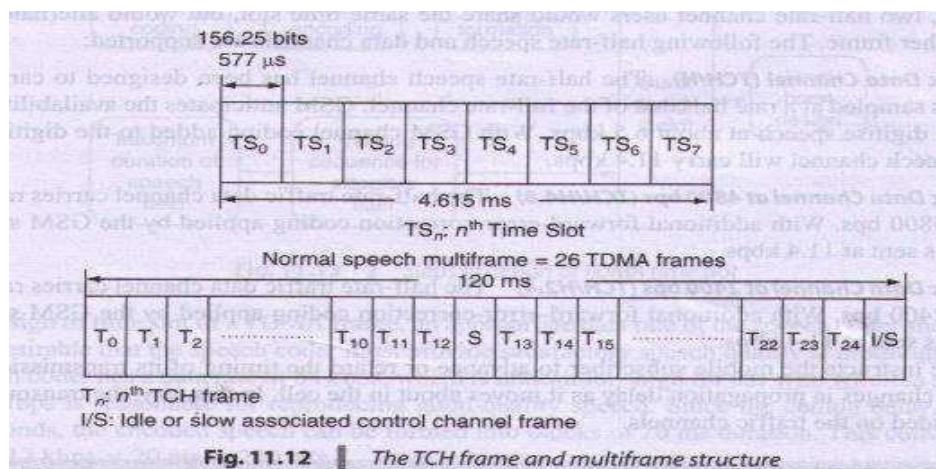
Control information in GSM is mainly on two logical channels—the Broadcast Channel (BCCH) and the Paging Channel (PCH). The broadcast information is transmitted first, followed by paging information. **Figure 11.11** shows the structure of a GSM logical control channel.



6.6.3 GSM Traffic Channels

Voice channels are called Traffic Channels (TCH) in GSM. Traffic channels are two-way channels carrying the voice and data traffic between the MS and BTS. Traffic channels can carry digitally encoded user speech or user data and have identical functions and formats on both the forward and reverse link. One RF channel is shared by eight voice transmissions using TDMA. GSM works out to 25 kHz per voice channel.

Figure 11.12 illustrates how the TCH data is transmitted in consecutive frames.



Frames of TCH data are broken up every thirteenth frame by either Slow Associated Control Channel Data (SACCH) or idle frames. Each group of twenty-six consecutive

TDMA frames is called a multiframe. For every twenty-six frames, the thirteenth and twenty-sixth frames consist of Slow Associated Control Channel (SACCH) data, or the idle frame, respectively. The twenty-sixth frame contains idle bits for the case when fullrate TCHs are used and contains SACCH data when half-rate TCHs are used. TCH logical channels are implemented over the normal burst. There are two types of TCH channels:

Full-rate traffic channel (TCH/F): This channel uses a 13 kbps speech-coding scheme and 9.600 bps, 4.800 bps, and 2.400 bps data. After including signaling overhead, each full-rate traffic channel has a gross bit rate of 22.8 kbps for the network. When transmitted as full-rate, user data is contained within one time frame.

Half-rate traffic channel (TCH/H): This channel uses 16 time slots per frame that has a gross bit rate of 11.4 kbps. The half-rate TCH supports 4800 bps and 2400 bps rate only. When transmitted as half-rate, user data is mapped onto the same time slot, but is sent in alternate frames. That is, two half-rate channel users would share the same time slot, but would alternately transmit during every other frame.

Mobility Management in GSM

Mobility Management function handles the function that arises due to mobility of the subscriber.

Main objective of MM is location tracking & call set up. The current location of an MS is maintained by a 2-level hierarchical strategy with HLR & VLR. When an MS visits a new location it must register in the VLR of visited location. The HLR must be informed about the registration. The registration process of MS moving from one VLR to another VLR follows following steps.

STEP-1. MS periodically listens to the BCCH (Broadcast Control Channel) broadcast from BSS. If the MS detects that it has entered into a new location area, it sends a registration message to the new VLR by using SDCCH (Standalone Dedicated Control Channel) channel.

SDCCH: Used only for signaling & short message.

BCCH: Provides system information.

STEP-2. The new VLR communicates with old VLR to find HLR of MS. The new VLR then performs authentication process.

STEP-3. After MS is authenticated, new VLR sends a registration message to HLR. If the registration request is accepted, the HLR provides new VLR with all relevant subscriber information.

STEP-4. The new VLR informs the MS of successful registration.

STEP-5. Then the HLR sends a deregistration (Cancellation) message to old VLR. The old VLR cancels the record for MS & sends an acknowledgement to the HLR regarding cancellation.

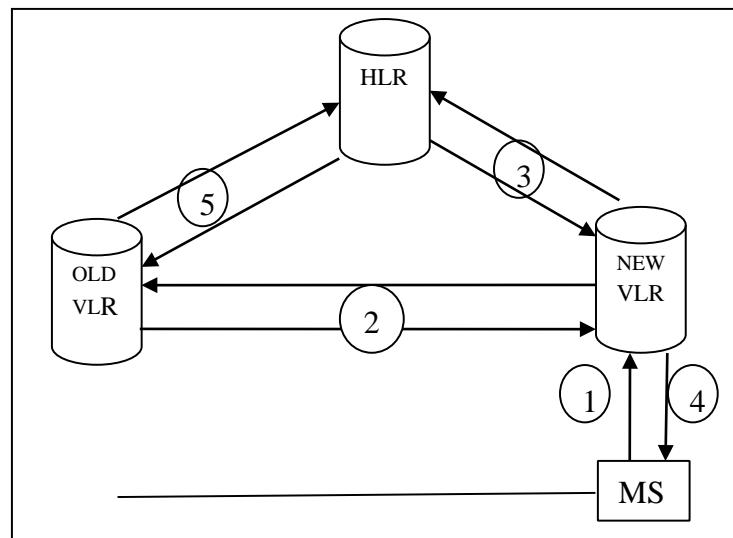


Fig 6.5 GSM Mobility Management

GSM Call Origination

1. MS sends the call origination request to MSC.
2. MSC forwards the request to VLR by sending **MAP_SEND_INFO_FOR_OUTGOING_CALL**.
3. VLR checks MS's profile & sends an ACK to MSC to grant call request.
4. MSC sets up communication link according to standard PSTN call set up procedure.

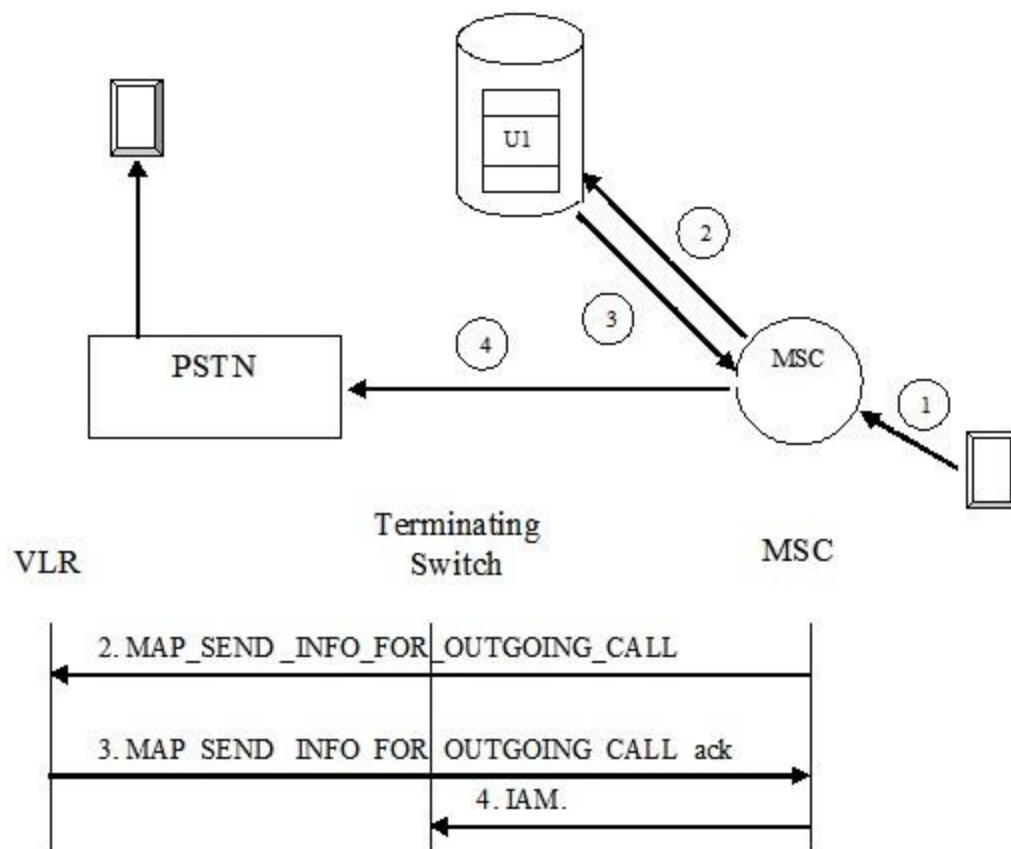


Fig 6.9 Call Origination Operation

Call Termination

When mobile station number is dialed by PSTN user, call is routed to GMSC by IAM (Initial Addressing Message) message.

1. To obtain routing information, GMSC interrogates HLR by sending **MAP_SEND_ROUTING_INFORMATION** to HLR.
2. HLR sends a **MAP_PROVIDE_ROAMING_NUMBER** message to VLR to obtain MSRN (MS Roaming Number). The message consists of IMSI, MSC number etc.
3. The VLR creates the MSRN by using MSC number stored in VLR record of MS. The MSRN no is sent back to GMSC through HLR.
4. MSRN provides address of target MSC where the MS resides. Then a message is directed from GMSC to target MSC to set communication link.

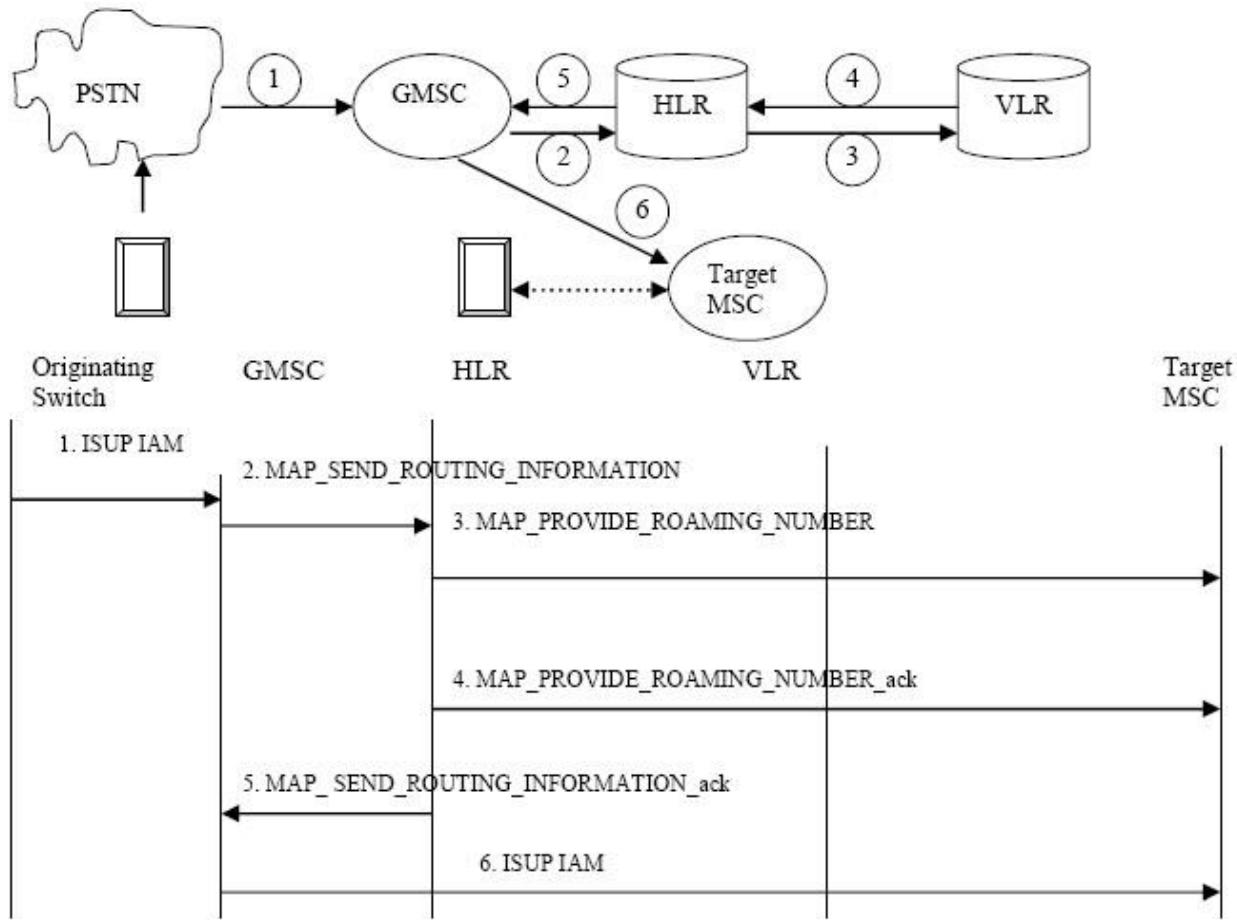


Fig 6.10 Call Termination Operation

GPRS (General Packet Radio Service)

GPRS is a packet oriented mobile data service available to users of the 2G cellular communication systems global system for mobile communications (GSM), as well as in the 3G systems. In the 2G systems, GPRS provides data rates of 56-114 kbit/s.

GPRS is a mechanism to transport high speed data over GSM. GPRS has the ability to offer data in small packet at speed of 14.4Kbps to 171.2Kbps. GPRS is a speed enhanced data transmission service designed for GSM system. Speed enhanced data transmission takes place by packetizing of data & simultaneous transmission of packets over different channels. GPRS standard is defined by ETSI (European Telecommunication Standard Institute). GPRS is a packet oriented service for mobile data transmission and their access to internet. It uses unused slots and channels in TDMA mode of a GSM for packetized transmission from a mobile station.

GPRS upgrades GSM data services providing:

- Multimedia messaging service (MMS)
- Short message service (SMS)
- Push to talk over cellular (PoC/PTT)
- Instant messaging and presence—wireless village
- Internet applications for smart devices through wireless application protocol (WAP)
- Point-to-point (PTP) service: inter-networking with the Internet (IP)
- Future enhancements: flexibility to add new functions, such as more capacity, more users, new accesses, new protocols, new radio networks.

Architecture

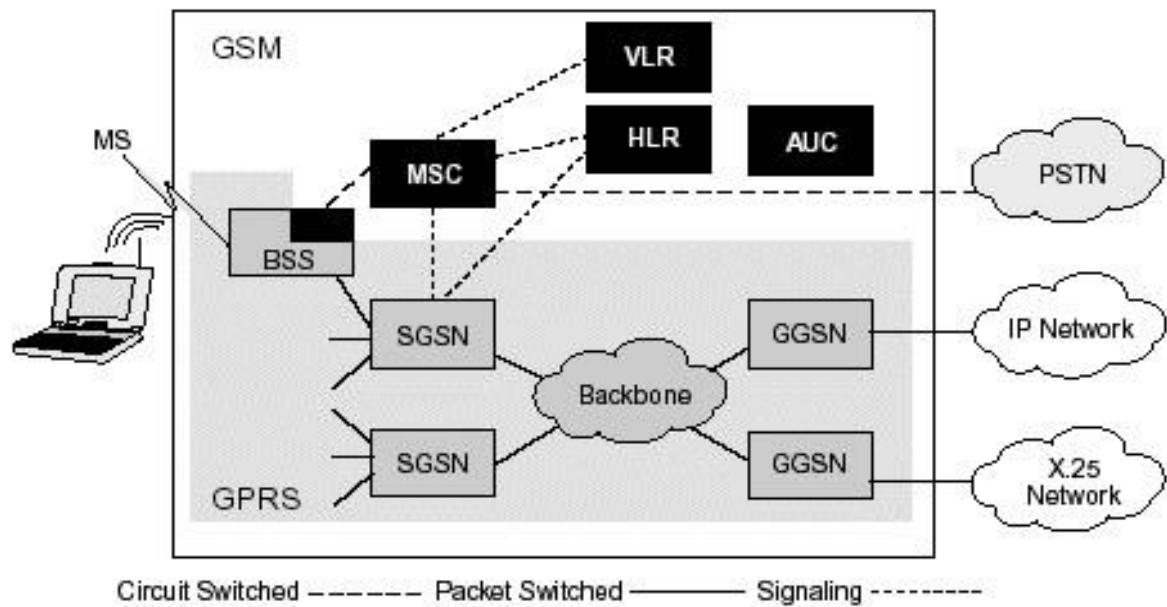


Fig 7.1 GPRS Architecture

GPRS uses GSM architecture for voice. GPRS supports a class of network nodes to offer packet data. These nodes are called as GPRS support nodes. GSN are responsible for delivery & routing of data packets between MS and external packet data network (PDN). An MS having GPRS capability stores CKSN (Cipher Key Sequence No) similar to cipher key stored in SIM & GSM. It also stores a TLLI (Temporary logical link identity) similar to TMSI. BSS system existing in the network needs enhancement to recognize and packet data. BTS also needs to be upgraded to support packet data transportation.

HLR needs enhancement to register GPRS user profile and respond to queries originating from GSN.

The GPRS system brings some new network elements to an existing GSM network.

These elements are:

Packet Control Unit (PCU)

The PCU separates the circuit switched and packet switched traffic from the user and sends them to the GSM and GPRS networks respectively. It also performs most of the radio resource management functions of the GPRS network. The PCU can be either located in the BTS, BSC, or some other point between the MS and the MSC. There will be at least one PCU that serves a cell in which GPRS services will be available. Frame Relay technology is being used at present to interconnect the PCU to the GPRS core.

Serving GPRS Support Node (SGSN)

The SGSN is the most important element of the GPRS network. The SGSN of the GPRS network is equivalent to the MSC of the GSM network. There must at least one SGSN in a GPRS network. There is a coverage area associated with a SGSN. As the network expands and the number of subscribers increases, there may be more than one SGSN in a network.

Gateway GPRS Support Node (GGSN)

The GGSN is the gateway to external networks. Every connection to a fixed external data network has to go through a GGSN. The GGSN acts as the anchor point in a GPRS data connection even when the subscriber moves to another SGSN during roaming. The GGSN may accept connection request from SGSN that is in another PLMN. Hence, the concept of coverage area does not apply to GGSN. There are usually two or more GGSNs in a network for redundancy purposes, and they back up each other up in case of failure.

Border Gateway

The Border Gateway (BG) is a router that can provide a direct GPRS tunnel between different operators' GPRS networks. This is referred to as an inter- PLMN data network. It is more secure to transfer data between two operators' PLMN networks through a direct connection rather than via the public Internet. The Border Gateway will commence operation once the GPRS roaming agreements between various operators have been

signed. It will essentially allow a roaming subscriber to connect to company intranet through the Home GGSN via the visiting PLMN network.

Charging Gateway

GPRS users have to be charged for the use of the network. In a GSM network, charging is based on the destination, duration, and time of call. However, GPRS offers connectionless service to users, so it is not possible to charge subscribers on the connection duration. Charging has to be based on the volume, destination, QoS, and other parameters of a connectionless data transfer. These GPRS charging data are generated by all the SGSNs and GGSNs in the network. This data is referred to as Charging Data Records or CDRs. One data session may generate a number of CDRs, so these need to be collected and processed. The Charging Gateway (CG) collects all of these records, sorts them, processes it, and passes it on to the Billing System. Here the GPRS subscriber is billed for the data transaction. All CDRs contain unique subscriber and connection identifiers to distinguish it. A protocol called GTP' (pronounced GTP prime) is used for the transfer of data records between GSNs and the Charging Gateway.

GPRS interfaces

The GPRS system introduces new interfaces to the GSM network. Figure 4 illustrates the logical architecture with the interfaces and reference points of the combined GSM/GPRS network.

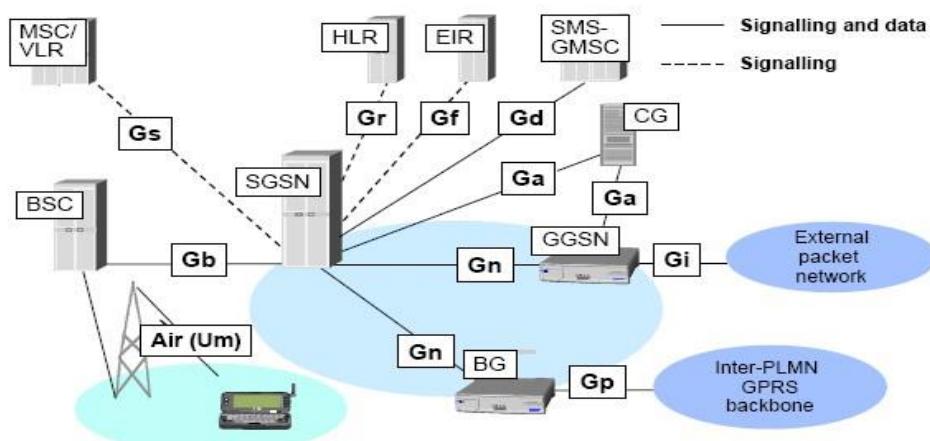


Figure 4. GPRS interfaces

The interfaces used by the GPRS system are described below:

- **Um** between an MS and the GPRS fixed network part. The Um is the access interface the MS uses to access the GPRS network. The radio interface to the BTS is the same interface used by the existing GSM network with some GPRS specific changes.
- **Gb** between a SGSN and a BSS. The Gb interface carries the GPRS traffic and signalling between the GSM radio network (BSS) and the GPRS network. Frame Relay based network services is used for this interface.
- **Gn** between two GSNs within the same PLMN. The Gn provides a data and signalling interface in the Intra-PLMN backbone. The GPRS Tunnelling Protocol (GTP) is used in the Gn (and in the Gp) interface over the IP based backbone network.
- **Gp** between two GSNs in various PLMNs. The Gp interface provides the same functionality as the Gn interface, but it also provides, together with the BG and the Firewall, all the functions needed for inter-PLMN networking, that is, security, routing, etc.
- **Gr** between an SGSN and the HLR. The Gr gives the SGSN access to subscriber information in the HLR. The HLR can be located in a different PLMN than the SGSN (MAP).
- **Ga** between the GSNs and the CG inside the same PLMN. The Ga provides a data and signalling interface. This interface is used for sending the charging data records generated by GSNs to the CG. The protocol used is GTP', an enhanced version of GTP.
- **Gs** between a SGSN and a MSC. The SGSN can send location data to the MSC or receive paging requests from the MSC via this optional interface. The Gs interface will greatly improve the effectiveness of the radio and network resources in the combined GSM/GPRS network. This interface uses BSSAP+ protocol.
- **Gd** between the SMS-GMSC and an SGSN, and between SMS-IWMSC and an SGSN. The Gd interface is available for more efficient use of the SMS services (MAP).
- **Gf** between an SGSN and the EIR. The Gf gives the SGSN access to GPRS user equipment information. The EIR maintains three different lists of mobile equipment:

black list for stolen mobiles, grey list for mobiles under observation and white list for other mobiles (MAP).

- **Gc** between the GGSN and the HLR. The GGSN may request the location of an MS via this optional interface. The interface can be used if the GGSN needs to forward packets to an MS that is not active.

There are two different **reference points** in the GPRS network. The Gi is GPRS specific, but the R is common with the circuit switched GSM network:

- **Gi** between a GGSN and an external network. The GPRS network is connected to an external data networks via this interface. The GPRS not a standard interface, but merely a reference point.
- **R** between terminal equipment and mobile termination. This reference point connects terminal equipment to mobile termination, thus allowing, for example, a laptop-PC to transmit data over the GSM-phone. The physical R interface follows, for example, the ITU-T V.24/V.28 or the PCMCIA PC-Card standards.

GPRS Network Protocol

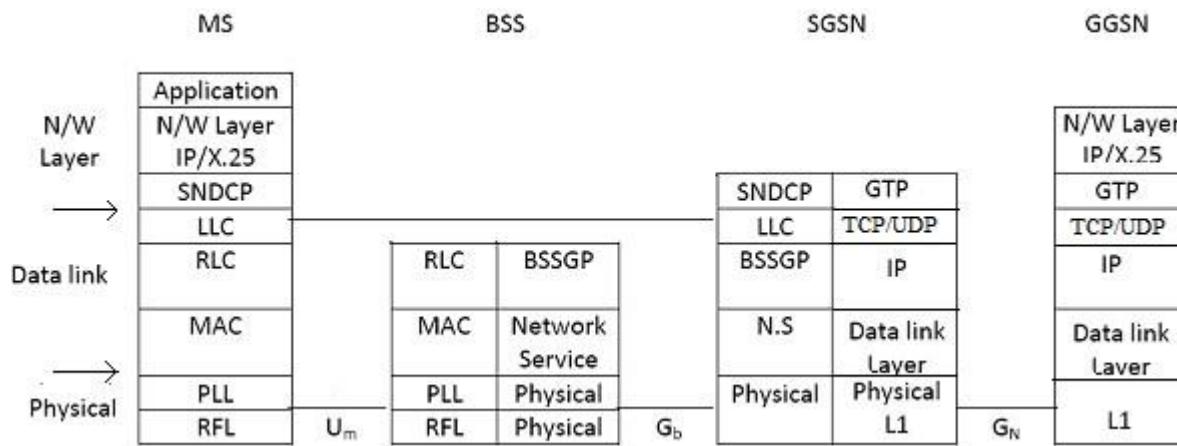


Fig 7.2 GPRS Networking Protocol

Abbreviations:

SNDCP: Subnetwork Dependent Convergence Protocol

LLC: Logical Link Control

RLC: Radio Link Control

MAC: Medium Access Control

PLL: Physical Link Layer

RFL: Radio Frequency LAYER

BSSGP: BSS GPRS Protocol

GTTP: GPRS Tunneling Protocol

TCP: Transmission Control Protocol

IP: Internet Protocol

Transmission protocols in the Um interface A. Physical layer

The physical layer can be divided into the Radio Frequency (RF) layer and the Physical Link layer.

The Radio Frequency (RF) is the normal GSM physical radio layer. It performs the modulation of physical waveform based on sequence of bits received from PLL. RF layer specifies:

- the carrier frequency characteristics and GSM radio channel structures
- the radio modulation scheme used for the data
- the radio transmitter and receiver characteristics as well as performance requirements.

The Physical Link layer supports multiple MSs sharing a single physical channel and provides communication between the MSs and the network. Network controlled handovers are not used in the GPRS service. Instead, routing area updates and cell updates are used. The Physical Link layer is responsible for:

- a. Forward Error Correction (FEC) coding, allowing the detection and correction of transmitted code words and the indication of incorrectable code words
- b. the interleaving of one RLC Radio Block over four bursts in consecutive TDMA frames.

B. Medium Access Control (MAC)

The Medium Access Control (MAC) protocol handles the channel allocation and the multiplexing, that is, the use of physical layer functions. The RLC and the MAC together form the OSI Layer 2 protocol for the Um interface. The GPRS MAC function is responsible for:

- Providing efficient multiplexing of data and control signalling on both the uplink and downlink. This process is controlled by the network. On the downlink, multiplexing is controlled by a scheduling mechanism. On users (for example, in response to a service request).
- Mobile originated channel access, contention resolution between channel access attempts, including collision detection and recovery.
- Mobile terminated channel access, scheduling of access attempts, including queuing of packet accesses.
- Priority handling.

C. The Radio Link Control (RLC)

The Radio Link Control (RLC) protocol offers a reliable radio link to the upper layers. Two modes of operation of the RLC layer are defined for information transfer: unacknowledged and acknowledged. The RLC layer can support both modes simultaneously.

The RLC function is responsible for:

- Providing transfer of Logical Link Control layer PDUs (LLC-PDU) between the LLC layer and the MAC function.
- Segmentation and reassembly of LLC-PDUs into RLC Data Blocks. See Figure 2.
- Backward Error Correction (BEC) procedures enabling the selective retransmission of uncorrectable code words. This process is generally known as Automatic Request for Retransmission (ARQ).

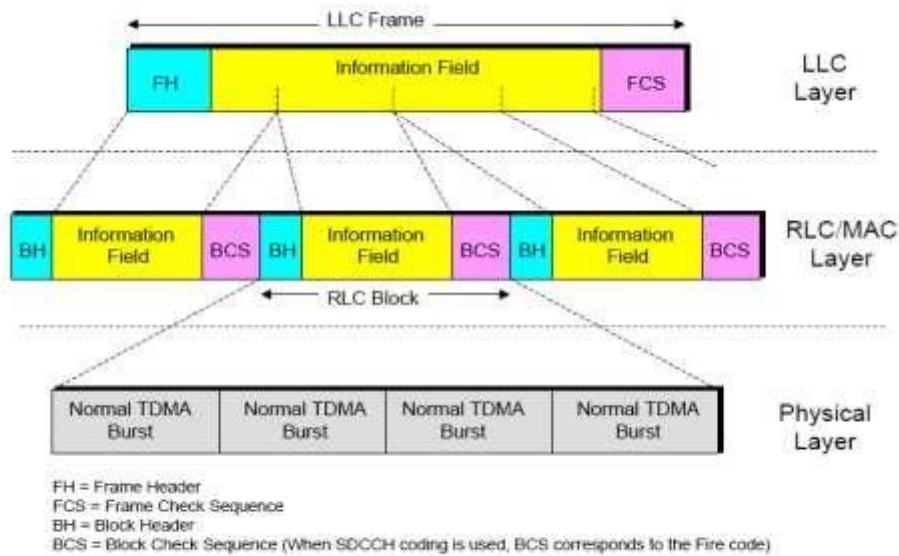


Figure 2. Segmentation of LLC-PDUs into RLC data blocks

D. Logical Link Control (LLC)

The Logical Link Control (LLC) layer offers a secure and reliable logical link between the MS and the SGSN for upper layer protocols, and is independent of the lower layers.

The LLC conveys signalling, SMS, and Subnetwork Dependent Convergence Protocol (SNDCP) packets. SNDCP exists between the MS and the SGSN and provides a mapping and compression function between the network layer (IP or X.25 packets) and the lower layers. It also performs segmentation, reassembly, and multiplexing.

Two modes of operation of the LLC layer are defined for information transfer: unacknowledged and acknowledged. The LLC layer can support both modes simultaneously.

- In acknowledged mode, the receipt of LLC-PDUs is confirmed. The LLC layer retransmits LLC-PDUs if confirmation has not been received within a certain timeout period.
- In unacknowledged mode, there is no confirmation required for LLU-PDUs.

Signalling and SMS is transferred in unacknowledged mode. In unacknowledged mode, the LLC layer offers the following two options:

- Transport of "protected" information means that if errors occur within the LLC information field, the frame will be discarded.

- Transport of "unprotected" information means that if errors occur within the LLC information field, the frame will not be discarded.

E. SNDCP (Subnetwork Dependent Convergence Protocol)

This is a convergence protocol used to transfer data packet and voice data between SGSN & MS through IP/X.25. Network layer protocols are intended to be capable of operating over a wide variety of subnetworks and data links. GPRS supports several network layer protocols providing protocol transparency for the users of the service. To enable the introduction of new network layer protocols to be transferred over GPRS without any changes to GPRS, all functions related to the transfer of Network layer Protocol Data Units (N-PDUs) are carried out in a transparent way by the GPRS network. This is one of the requirements of SNDCP. Another requirement of the SNDCP is to provide functions that help to improve channel efficiency. This is achieved by means of compression techniques. The set of protocol entities above the SNDCP consists of commonly used network protocols. They all use the same SNDCP entity, which then performs multiplexing of data coming from different sources to be transferred using the service provided by the LLC layer.

A. Physical Layer Protocol

Several physical layer configurations and protocols are possible at the Gb interface and the physical resources are allocated by Operation & Maintenance (O&M) procedures. Normally a G703/704 2Mbit/s connection is provided.

B. Network Services layer

The Gb interface Network Services layer is based on Frame Relay. Frame Relay virtual circuits are established between the SGSN and BSS. LLC PDUs from many users are statistically multiplexed onto these virtual circuits. These virtual circuits may traverse a network of Frame Relay switching nodes, or may just be provided on a point to point link between the BSC and the SGSN (if the BSC and SGSN are co-located). Frame Relay is used for signalling and data transmission over the Gb interface.

C. Base Station System GPRS Protocol (BSSGP)

The Base Station System GPRS Protocol (BSSGP) transfers control and signaling information and user data between a BSS and the SGSN over the Gb interface.

The primary function of BSSGP is to provide Quality of Service (QoS), and routing information that is required to transmit user data between a BSS and an SGSN.

The secondary function is to enable two physically distinct nodes, the SGSN and BSS, to operate node management control functions. There is a one-to-one relationship between the BSSGP protocol in the SGSN and in the BSS. If one SGSN handles multiple BSSs, the SGSN has to have one BSSGP protocol device for each BSS.

A. Layer 1 and layer 2

The L1 and the L2 protocols are vendor dependent OSI layer 1 and 2 protocols that carry the IP datagrams for the GPRS backbone network between the SGSN and the GGSN.

B. Internet Protocol (IP)

The Internet Protocol (IP) datagram in the Gn interface is only used in the GPRS backbone network. The GPRS backbone (core) network and the GPRS subscribers use different IP addresses. This makes the GPRS backbone IP network invisible to the subscribers and vice versa. The GPRS backbone network carries the subscriber IP or X.25 traffic in a secure GPRS tunnel. All data from the mobile subscribers or external networks is tunnelled in the GPRS backbone.

C. TCP or UDP

TCP or UDP are used to carry the GPRS Tunnelling Protocol (GTP) PDUs across the GPRS backbone network. TCP is used for user X.25 data and UDP is used for user IP data and signalling in the Gn interface.

D. GPRS Tunnelling Protocol (GTP)

The GPRS Tunnelling Protocol (GTP) allows multi-protocol packets to be tunnelled through the GPRS backbone between GPRS Support Nodes (GSNs). GTP is defined both for the Gn interface, which is, the interface between GSNs within the same PLMN, and the Gp interface between GSNs in different PLMNs. The UDP/IP and TCP/IP are examples of paths that may be used to multiplex GTP tunnels. The choice of path is dependent on whether the user data to be tunnelled requires a reliable link or not. Two modes of operation of the GTP layer are therefore supported for information transfer between the GGSN and SGSN.

- unacknowledged (UDP/IP)
- acknowledged (TCP/IP).

A UDP/IP path is used when the user data is based on connectionless protocols, such as IP. A TCP/IP path is used when the user data is based on connection oriented protocols, such as X.25. The GTP layer can support both modes simultaneously.

Wireless LAN

WLAN is a LAN without wires. Mobile users can access information and network resources through LAN. The goal of WLAN is to replace office cabling to enable quicker access to internet and to higher flexibility communication. It is implemented as an extension to a wired LAN within a building or campus.

Wireless LAN Application

There are many area and applications of wireless LAN. Wireless LAN is best suited for dynamic environment. The applications are as follows:

- **Cross Building Interconnect** – Wireless can be used to connect LANs in nearby buildings. Here a point-to-point wireless link is used between two buildings. The devices connected are bridges and routers.
- **Nomadic Access** – It provides a wireless link between a LAN hub and a mobile data terminal equipped with an antenna such as laptop or notepad computer. Nomadic access is also useful in an extended environment such as a campus or a business operating out of a cluster of building.
- **Ad Hoc Networking** – An ad hoc network is a peer-to-peer network setup temporarily to meet some immediate need. For example, a group of employees, each with a laptop computer, may convene in a conference room for business. The employees link their computers in a temporary network just for the duration of the meeting.

Wireless LAN Requirements

A wireless LAN must meet the same sort of requirements typical of any LAN including high capability, ability to cover short distances and broadcast capability. There are also a number of requirements specific to wireless LAN environment. The following are the most important requirements:

- **Throughput:** The medium access control protocol should make as efficient use as possible of the wireless medium to maximize capacity.
- **Number of nodes:** Wireless LAN may need to support hundreds of nodes across multiple cells.

- **Connection to backbone LAN:** In case of Wireless LAN, an interconnection structure is required
- **Service area:** A typical coverage area for a wireless LAN has a diameter of 100 to 300 m.
- **License free operation:** Users would prefer to buy and operate wireless LAN products without having to secure a license for the frequency band used by the LAN.
- **Handoff/roaming:** The MAC protocol used in the wireless LAN should enable mobile stations to move from one cell to another.
- **Dynamic configuration:** the MAC addressing and network management aspects of the LAN should permit dynamic and automated addition, deletion and relocation of end systems without disruption to other users.
- **Battery power consumption:** Mobile workers use battery powered workstations that need to have a long battery life when used with wireless adapters. Wireless LAN implementations have features to reduce power consumption while not using the network, such as a sleep mode. **Wireless LAN Advantages**
- **Mobility:** Productivity increases when people have access to data and information from any location.
- **Low Implementation Cost:** WLANs are easy to set up, relocate, change and manage.
- **Installation Speed and Simplicity:** Installing a WLAN can be fast and can eliminate the need to install cable through walls.
- **Network Expansion:** Wireless Technology allows the network to reach where wires can not reach.
- **Reliability:** WLAN is resistant to different types of cable failures.
- **Scalability:** WLAN can be configured in a variety of topologies to meet the needs of specific applications and installations.
- **Usage of ISM band:** WLAN operates in the unregulated ISM bands available for use by anyone.

Wireless LAN Technology

Wireless LANs are generally categorized according to the transmission technique that is used. Current wireless LAN products fall into one of the following categories:

- **Infrared (IR) LANs:** An individual cell of an IR LAN is limited to a single room, because infrared light does not penetrate opaque walls. Three transmission techniques are used for IR data transmission. (i). Direct beam IR can be used to create point-to-point links. In this mode the range depends on the emitted power and on the degree of focusing. (ii) An omni directional configuration involves a single base station that is within line of sight of all other stations on the LAN. Typically this station is mounted on the ceiling. (iii) In diffused configuration, all of the IR transmitters are focused and aimed at a point on a diffusely reflecting ceiling. IR radiation striking the ceiling is reradiated omnidirectionally and picked up by all of the receivers in the area.
- **Spread Spectrum LANs:** This type of LAN makes use of spread spectrum transmission technology. In most cases, these LANs operate in the ISM frequency bands.

Spread spectrum LAN makes use of a multiple cell arrangement. Within a cell, the topology can be either hub or peer to peer. In a hub topology, the hub is typically mounted on the ceiling and connected to a backbone wired LAN to provide connectivity to stations attached to the wired LAN and to stations that are part of wireless LANs in other cells. A peer-to-peer technology is one in which there is no hub. A MAC algorithm such as CSMA is used to control access.

- **Narrowband microwave:** These LANs operate at microwave frequencies but do not use spread spectrum. The term narrowband microwave refers to the use of a microwave radio frequency band for signal transmission with relatively narrow bandwidth.

Types of WLAN

- **IEEE 802.11:** In June 1997, IEEE finalized the initial specification for WLAN. It specifies 2.4 GHz frequency band with data rate of 2Mbps. This standard evolved into many variations using different encoding technologies.
- **HYPER LAN:** It began in Europe in 1996 by ETSI (European Telecommunication Standard Institute). ETSI broadband radio access network group. The current version Hyper LAN/1 works at 5 GHz frequency band and offers up to 24 Mbps bandwidth.

- **BLUETOOTH:** It is promoted by big industry leaders like IBM, ERICSON, NOKIA. It was named after Harold Bluetooth, king of Denmark. It offers 1Mbps data rate at 2.2 GHz band. It is also known as PAN (Personal area network).
- **MANET:** It is a working group to investigate and develop the standard for mobile ad-hoc network (MANET).

IEEE 802.11

The IEEE standard 802.11 specifies the most famous family of WLANs in which many products are available. The number in the standard indicates, it belongs to the group of 802.X LAN standards. The primary goal of the standard was the specification of a simple and robust WLAN which offers time bounded and asynchronous services.

Architecture

The smallest building block of a WLAN is a basic service set (BSS) which consist of some number of stations executing the same MAC protocol. A BSS may be isolated or it may connect to a backbone distribution system (DS) through an access point (AP). The AP functions as a bridge and a relay point. In a BSS, client stations do not communicate directly with one another. If one station in BSS wants to communicate with another station in the same BSS, the MAC frame is first sent from originating station to the AP and from AP to destination station.

When all the stations in the BSS are mobile stations, with no connection to other BSSs, the BSS is called an independent BSS (IBSS). An IBSS is an ad hoc network.

An extended service set (ESS) consists of two or more basic service sets interconnected by a distribution system. The distribution system is a wired backbone LAN but can be any communication network. The ESS appears as a single logical LAN to the logical link control level.

An access point is implemented as part of a station. The AP is the logic within a station that provides access to the DS by providing DS services in addition to acting as a station.

A Wireless networks can exhibit two different basic system architecture. WLAN are of 2 types:

- Infrastructure mode
- Ad-hoc mode

Infrastructure Mode

Here, MSs are connected with BS or access point. This is similar to star network communication takes place between wireless nodes and access point but not directly between wireless devices. Here access points acts as a bridge to other network.

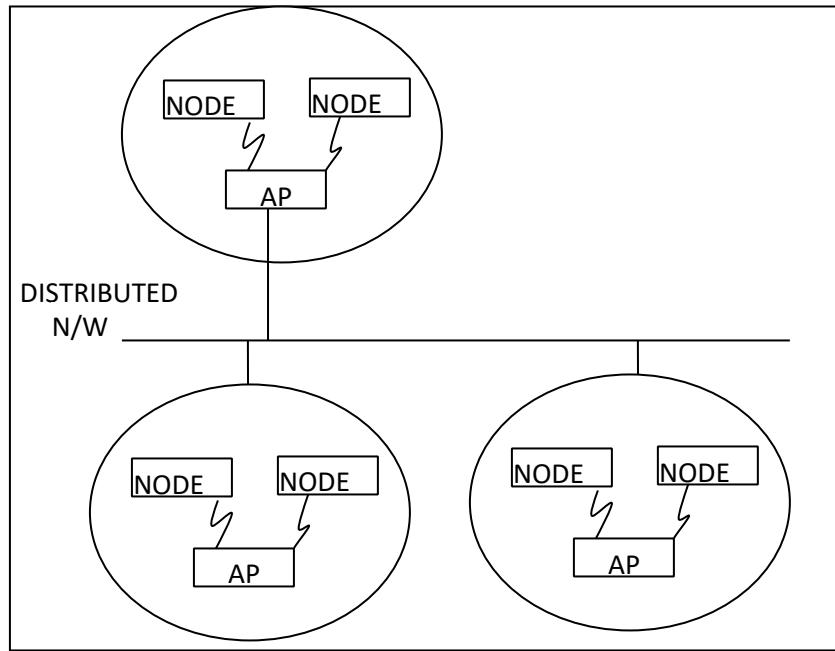


Fig 9.1 WLAN in Infrastructure Mode

Ad-hoc Mode

In ad-hoc mode there is no access point. A number of MS can communicate directly with each other. Nodes can communicate if they can reach each other physically i.e. they are given each other radio range.

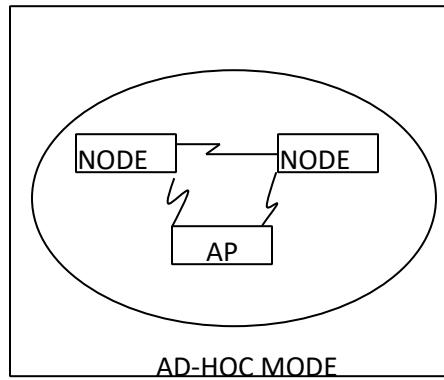


Fig 9.2 WLAN in Ad-hoc mode

IEEE 802.11 Services

IEEE 802.11 defines the following services that need to be provided by the Wireless LAN.

- **Distribution:** It is the primary service used by stations to exchange MAC frames when the frame must traverse the Ds to get from a station in one BSS to a station in another BSS.
- **Integration:** This service enables transfer of data between a station on an IEEE 802.11 LAN and a station on an integrated IEEE 802.x LAN. The term integrated refers to a wired LAN that is physically connected to the DS and whose stations may be logically connected to an IEEE 802.11 LAN via integration services. The integration service takes care of address translation, media conversion logic required for exchange of data.
- **Association:** Establishes an initial association between a station and an AP. Before a station can transmit or receive frames on a wireless LAN, its identity and address must be known. For this purpose, a station must establish an association with an AP within a particular BSS. The AP can then communicate this information to other APs within the ESS to facilitate routing and delivery of address frames.
- **Reassociation:** Enables an established association to be transferred from one AP to another, allowing a mobile station to move from one BSS to another.
- **Disassociation:** A notification from either a station or an AP that an existing association is terminated. A station gives this notification before leaving an ESS or shutting down.
- **Authentication:** Used to establish the identity of stations to each other. This authentication service is used by stations to establish their identity with stations they wish to communicate with.
- **Deauthentication:** This service is invoked whenever an existing authentication is to be terminated.
- **Privacy:** Used to prevent the contents of message from being read by other than the intended recipient. The standard provides for the optional use of encryption to assure privacy.

Protocol Architecture

The protocol architecture of 802.11 consists of two layers physical layer and MAC layer. The physical layer is subdivided into physical layer convergence protocol (PLCP) and

physical medium dependent sub layer (PMD). The basic task of MAC layer comprise medium access, fragmentation of user data, encryption. The PLCP layer provides carrier sense signal called clear channel assessment. It delivers the incoming frame from wireless medium to MAC protocol Data unit (MPDU) for data transfer. PMD layer handles modulation and encoding or decoding of signals. It provides the actual transmission and reception of physical layer entity between MS through wireless medium.

802.11 MAC Management			MAC Layer		Data Link Layer
Frequency	Direct	Infrared	PLCP layer	sub	Physical Layer
Hopping	Sequence		PMD	Sub Layer	

Fig 9.3 IEEE 802.11 protocol architecture

Physical Layer

IEEE 802.11 supports three different physical layers: one layer based on infra red and two layers based on radio transmission. All PHY variants include the provision of the clear channel assessment signal (CCA). This is needed for the MAC mechanisms controlling medium access and indicate if the medium is currently idle. The PHY layer offers a service access point (SAP) with 1 or 2 Mbits/transfer rate to the MAC layer.

Frequency hopping spread spectrum

Frequency hopping spread spectrum (FHSS) is a spread spectrum technique which allows for the coexistence of multiple networks in the same area by separating different networks using different hopping sequences.

The standard specifies Gaussian shaped FSK (frequency shift keying), GFSK, as modulation for the FHSS PHY. For 1 Mbit/s a 2 level GFSK is used, for 2Mbit/s a 4 level GFSK is used.

Figure shows a frame of the physical layer used with FHSS. The frame consists of two basic parts, the PLCP part (permeable and header) and the pay load part. While the PLCP part is always transmitted at 1Mbit/s, payload can use 1 or 2 Mbit/s.

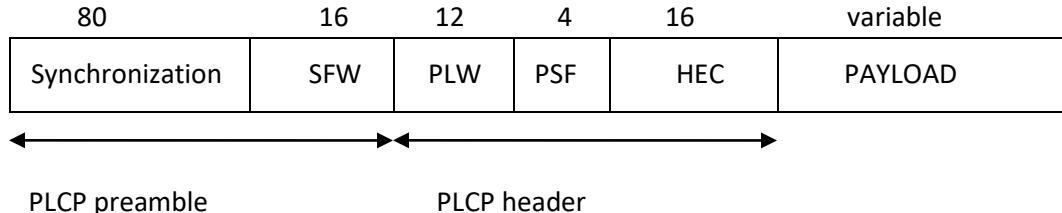


Fig 9.4 IEEE 802.11 PHY frame using FHSS

The field of the frame fulfills the following functions:

Synchronization: The PLCP permeable starts with 80 bit synchronization, which is a 010101....bit pattern. This pattern is used for synchronization of potential receivers and signal detection by the CCA.

Start frame delimiter (SFD): The following 16 bits indicate the start of the frame and provide frame synchronization. The SFD pattern is 0000110010111101.

PLCP_PDU length word (PLW): This first field of the PLCP header indicates the length of the payload in bytes including the 32 bit CRC at the end of the payload. PLW can range between 0 and 4095.

PLCP signaling field (PSF): This 4 bit field indicates the data rate of the payload following. All bit set to zero (0000) indicates the lowest data rate of 1Mbit/s, 2 Mbit/s is indicated by 0010 and the maximum is 8.5 Mbit/s (1111).

Header error check (HEC): PLCP header is protected by a 16 bit checksum

Direct sequence spread spectrum

Direct sequence spread spectrum (DSSS) is the alternative spread spectrum method separating by code and not by frequency. IEEE 802.11 DSSS PHY uses the 2.4 GHz ISM band and offers both 1 and 2 Mbit/s data rates. The system uses differential binary phase shift keying(DBPSK) for 1 Mbit/s transmission and differential quadrature phase shift keying(DQPSK) for 2 Mbit/s as modulation schemes.

Figure shows a frame of the physical layer using DSSS. The frame consists of two parts, one the PLCP part and the other Payload part while the PLCP part is always transmitted at 1Mbit/s,payload,.i.e MAC data can use 1 or 2 Mbit/s.

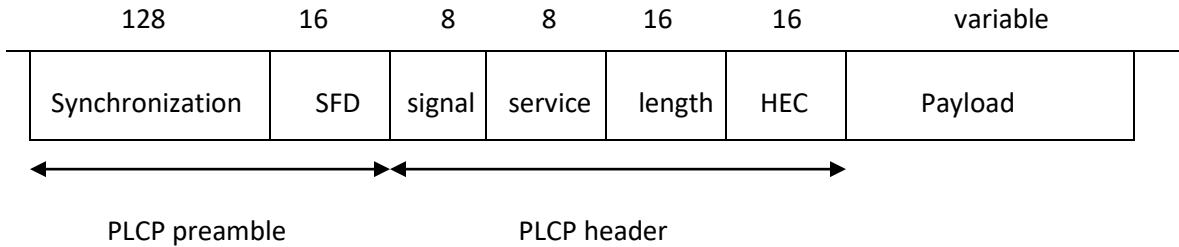


Fig 9.5 IEEE 802.11 PHY frame using DSSS

The field of the frame fulfills the following functions:

- **Synchronization:** The first 128 bits are used for synchronization, energy detection, frequency offset compensation. The synchronization field only consists of scrambled 1 bit.
 - **Start frame delimiter (SFD):** This 16 bit field is used for synchronization at the beginning of the frame and consists of the pattern 1111001110100000.
 - **Signal:** The values in this field indicate the data rate of the payload. The value 0x0A indicates 1 Mbit/s, 0x14 indicates 2 Mbit/s. Other values have reserved for further use i.e. higher bit rates.
 - **Service:** This field is reserved for future use; however, 0x00 indicates an IEEE 802.11 complaint frame.
 - **Length:** 16 bits are used for length indication of the payload in microseconds.
 - **Header error check (HEC):** Signal, service and length fields are protected by this checksum using the ITU-T CRC-16 standard polynomials.

Infra-red

The PHY layer based on infra-red (IR) transmission uses near visible light at 850-950 nm. Infra- red light is not regulated apart from safety restrictions (using lasers insteaded of LEDs).The standard does not require a line of sight between sender and receiver, but should also work with diffuse light. This allows for point-to-multipoint communications. The maximum range is about 10m if no sunlight or heat sources interfere with the transmission. Typically such a network will only work in buildings, e.g.-class rooms, meeting rooms etc.

Frequency reuse is very simple-a wall is more than enough to shield one IR based IEEE 802.11 network from another.

MAC Layer

MAC layer is responsible for controlling the access medium, roaming, authentication, power conservation etc. The basic services provided by MAC layer are mandatory asynchronous data service and optional time bound service. Asynchronous data service is provided in ad-hoc network medium and both services are available in infrastructure mode.

Access mechanisms defined under IEEE 802.11 are as follows:

- Carrier Sense Multiple Access with collision Avoidance
- Request to Send/ Clear to Send

These methods are called as Distributed Coordination Function (DCF) and offers asynchronous service.

Access mechanism using CSMA/CA

The mandatory access mechanism is based on carrier sense multiple access with collision access CSMA/CA. It employs a random back off mechanism with carrier sense and collision avoidance to reduce the probability of collision between 2 frames. The mechanism behind CSMA/CA is as follows:

- When a wireless station wants to communicate, it first listens to its media to check if it can sense radio wave from any other station.
- If the medium is free for a specified time then the station is allowed to transmit. This time interval is called Distributed Inter Frame Space (DIFS).
- If the current device senses carrier signal of another device on the same frequency, it does not transmit and initiates a random timeout.
- After the timeout has expired, the wireless station again listens to the radio spectrum and if it still senses another station transmitting, initiates another random time out.
- When it does not sense another wireless station transmitting, it starts transmitting its own carrier signal to communicate with other station.
- The receiving station checks the CRC of the received packet and sends an acknowledgement. Receipt of the acknowledgement indicates to the transmitter

that no collision occurred. If the sender does not receive the acknowledgement then it retransmits the fragments until it receives acknowledgement.

Access mechanism using RTS/CTS

This mechanism is used to solve the problems of hidden terminals. This problem occurs if one station can receive two others, but those stations can not receive each other. The two stations may sense the channel is idle, send a frame and cause a collision at the receiver in the middle. In order to reduce the probability of two stations colliding because they can't sense each other's presence, the standard defines a mechanism called Virtual Carrier Sense (VCS).

- A station wants to transmit a packet first transmits a short control packet called RTS (Request to send) which includes source, destination and duration of transaction.
- The destination station after receiving this request packet responds with a response control packet called CTS (CLEAR TO SEND), which includes same duration information.
- All stations receiving either RTS or CTS set their virtual carrier sense indicator called Network Allocation Vector (NAV) for the given duration and uses this information together with physical carrier sense when sensing the medium. This mechanism reduces the probability of a collision on the receiver side by a station.
- The transmitter station when receives the CTS after sending a RTS; it reserve the medium busy until the end of transaction.

MAC Frames

Figure shows the basic structure of an IEEE 802.11 MAC frame.

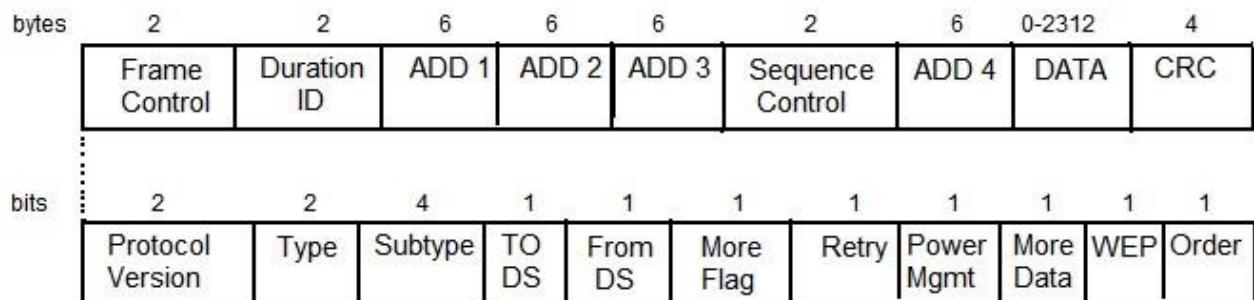


Fig 9.6 IEEE 802.11 MAC packet structure

- **Frame Control:** It is of two bytes and contains several sub-fields.
 1. **Protocol Version:** It is of two bit and indicate current protocol version.
 2. **Type:** This field determines function of a frame. For example: management (00), control (01), data (10)
 3. **Subtype:** If the above frame type is again sub-divided into various types. Ex- RTS is a control frame with sub type 1011.
 4. **To DS/From DS:** Frames can be transmitted between MS and AP. This field contains the address of source and destination Distributed system.
 5. **More Fragment:** This field is set to 1 if frames are of type data and management.
 6. **Retry:** If current frame is a retransmission of a earlier frame then this bit is set to 1.
 7. **Power Management:** This field indicates the power of a station after successful transmission of a frame .If one station is in power same mode.
 8. **More data:** This field is used to indicate a receiver that a sender has more data to send.
 9. **Wired Equivalent Privacy (WEP):** This field indicates the standard security mechanism of 802.11.
 10. **Order:** If this bit is set to 1 then the received frame must be processed in strict order.
- **Duration ID:** Indicating the period of time in which the medium is occupied for frame transfer.
- **Address 1 to 4:** The four address field contains IEEE802.MAC address.
- **Sequence Control:** A sequence no is maintained to filter duplicate frames.
- **Data:** MAC frame may contain arbitrary data, which is transferred from sender to receiver.
- **Checksum (CRC):** 32 bit checksum is used to protect the frames.

MAC frames are of various types

Control Frame: Control frames assist in the reliable delivery of data frames. There are six types of control frame:

- **Power save-Poll (PS-Poll):** This frame is sent by any station to the station that includes the access point. Its purpose is to request that the AP (access point) transmit a frame that has been buffered for this station while the station was in power saving mode.
- **Request to Send (RTS):** This is the first frame in the four way frame exchange. The station sending this message is alerting a potential destination, and all other stations within reception range that it intends to send a data frame to that destination.
- **Clear to Send (CTS):** This is the second frame in the four way exchange. It is sent by the destination station to the source station to grant permission to send a data frame.
- **Acknowledgement:** Provides an acknowledgement from the destination to the source that the immediately preceding data, management, or PS-poll frame was received correctly.
- **Contention-free (CF)-End:** Announces the end of a contention-free period that is part of the point co-ordination function.
- **CF-End + CF-Ack:** Acknowledges the CF-End. This frame ends the contentionfree period and releases stations from the restrictions associated with that period.

Data Frames: There are 8 data frame sub-types, organized into two groups. The first 4 subtypes define frames that carry upper level data from the source station to the destination station. The 4 data-carrying frames are as follows:

- **Data:** This is the simplest data frame. It may be used in both a contention period and a contention-free period.
- **Data + CF-Ack:** May only be sent during a contention-free period .In addition to carrying data, this frame acknowledges previously received data.
- **Data + CF-Poll:** Used by a point coordinator to deliver data to a mobile station and also to request that the mobile station send a data frame that it may have buffered.

- **Data + CF-Ack + CF-Poll:** Combines the functions of the Data + CF-Ack and Data + CF-Poll into a single frame.

MAC Management

- **Synchronization:** Function to support finding a wireless LAN, synchronization of internal clock.
- **Power Management:** Function to control transmitter activity for power conservation without missing a frame.

The basic idea of power management is to switch off transceiver whenever it is not needed. The idea of power saving includes two states for a station: sleep and awake, and buffering of data in senders. If a sender intends to communicate with a power saving station it has to buffer data if the station is asleep. The sleeping station has to wake up periodically and stay awake for a certain time. During this time, all senders can announce the destinations of their buffered data frames. If a station detects that it is a destination of a buffered packet it has to stay awake until transmission takes place.

- **Roaming:** Moving between access points is called roaming. Steps for roaming between access points are:
 1. A station decides that the current link quality to its access point AP1 is too poor. The station then starts scanning for another access point.
 2. Scanning involves the active search for another BSS and can be used for setting up a new BSS. Scanning is of two types. Passive scanning means listening into the medium to find other networks. Active scanning comprises sending a probe on each channel and waiting for a response. Probe response contains the necessary information to join the new BSS.
 3. The station then selects best access point for roaming based on signal strength and sends an association request to the selected access point AP2.
 4. The new association point AP2 answers with an association response. If the response is successful, the station has roamed to the new access point.
 5. The access point accessing an association request indicates the new station in its BSS to the distribution system. The DS then updates its database, which contains the current location of the wireless station.

Internet Protocol

The **Internet Protocol (IP)** is a protocol used for communicating data across a packetswitched internetwork using the Internet Protocol Suite, also referred to as TCP/IP. IP is the primary protocol in the Internet Layer of the Internet Protocol Suite and has the task of delivering distinguished protocol datagrams (packets) from the source host to the destination host solely based on their addresses.

The Internet Protocol (IP) is the protocol by which data is sent from one computer to another on the Internet. Each computer (known as a host) on the Internet has at least one IP address that uniquely identifies it from all other computers on the Internet. When you send or receive data (for example, an e-mail note or a Web page), the message gets divided into little chunks called packets. Each of these packets contains both the sender's Internet address and the receiver's address. Any packet is sent first to a gateway computer that understands a small part of the Internet. The gateway computer reads the destination address and forwards the packet to an adjacent gateway that in turn reads the destination address and so forth across the Internet until one gateway recognizes the packet as belonging to a computer within its immediate neighborhood or domain. That gateway then forwards the packet directly to the computer whose address is specified.

Because a message is divided into a number of packets, each packet can be sent by a different route across the Internet. Packets can arrive in a different order than the order they were sent in. The Internet Protocol just delivers them. It's up to another protocol, the Transmission Control Protocol (TCP) to put them back in the right order.

IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (The reason the packets do get put in the right order is because of TCP, the connection-oriented protocol that keeps track of the packet sequence in a message.) In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Network Layer.

IPv4

The Internet Protocol version 4 (IPv4) is the delivery mechanism used by the TCP/IP Protocols. IPv4 is an unreliable and connectionless datagram protocol-a best-effort delivery service. The term best-effort means that IPv4 provides no error control or flow

control (except for error detection on the header). IPv4 assumes the unreliability of the underlying layers and does its best to get a transmission through to its destination, but with no guarantees.

IPv6

The Internet Engineering Task Force (IETF) introduced a specification in 1995 for a next generation IP, known as IPng. This specification was turned into a standard in 1996 known as IPv6. IPv6 provides a number of functional enhancements over the existing IP. It was designed to accommodate high speed networks, support of mix of data stream including graphics and video etc. Ipv6 uses 128-bit address to specify source and destination.

Mobile IP

It is an Internet Engineering Task Force (IETF) standard communications protocol that is designed to allow mobile device users to move from one network to another while maintaining a permanent IP address.

A standard that allows users with mobile devices whose IP addresses are associated with one network to stay connected when moving to a network with a different IP address.

Mobile IP was developed to enable computers to maintain Internet Connectivity while moving from one Internet attachment point to another. When a user leaves the network with which his device is associated (home network) and enters the domain of a foreign network, the foreign network uses the Mobile IP protocol to inform the home network of a care-of address to which all packets for the user's device should be sent.

Goals of mobile IP

The major goals of mobile IP were as follows:

- To continue to work with the exiting TCP/IP protocol suite.
- To provide Internet wide mobility, allowing a host the same IP address, called 'home address'.
- To optimize local area mobility without sacrificing performance or functionality of the general case.
- To leave the transport layer and higher protocols untouched.

- To ensure that no application needs to change in order to run on or to be used from mobile hosts(MHs)
- To ensure that the infrastructure, that is , non-MH, routers, routing protocols, etc are not changed either.
- To see the mobility is handled at the network layer.
- To ensure that the solution scales well and minimizes potential points of failure, and
- To ensure minimum power consumption, since mobile nodes are likely to be battery powered.

Operation of Mobile IP

A mobile node can have two addresses - a permanent home address and a care-of address (CoA), which is associated with the network the mobile node is visiting. There are two kinds of entities in Mobile IP:

- A home agent stores information about mobile nodes whose permanent home address is in the home agent's network.
- A foreign agent stores information about mobile nodes visiting its network. Foreign agents also advertise care-of addresses, which are used by Mobile IP. When the mobile node moves its attachment point to another network, that network is considered as a foreign network for the host.

A node wanting to communicate with the mobile node uses the permanent home address of the mobile node as the destination address for sent packets. Because the home address logically belongs to the network associated with the home agent, normal IP routing mechanisms forward these packets to the home agent. Instead of forwarding these packets to a destination that is physically in the same network as the home agent, the home agent redirects these packets towards the foreign agent. The home agent looks for the care-of address (CoA) in a special table known as a binding table, and then tunnels the packets to the mobile node's care-of address by appending a new IP header to the original IP packet, which preserves the original IP header. The packets are decapsulated at the end of the tunnel to remove the IP header added by the home agent, and are delivered to the mobile node.

When acting as transmitter, mobile node simply sends packets directly to the other communicating node through the foreign agent, without sending the packets through the home agent, using its permanent home address as the source address for the IP packets. This is known as triangular routing. If needed, the foreign agent could employ reverse tunneling by tunneling the mobile node's packets to the home agent, which in turn forwards them to the communicating node. This is needed in networks whose gateway routers have ingress filtering enabled and hence the source IP address of the mobile host would need to belong to the subnet of the foreign network else the packets will be discarded by the router.

When IP datagrams are exchanged over a connection between the mobile node and another host following operation takes place

1. Server X transmits an IP datagram destined for mobile node A, with A's home address in the IP Address. The IP datagram is routed to A's home network.
2. At the home network, the incoming IP datagram is intercepted by home agent. The home agent encapsulates the entire datagram inside a new IP datagram and retransmits the datagram. This datagram is routed to the foreign agent.
3. The foreign agent strips off the outer IP header, encapsulates the original IP datagram in a network level PDU and delivers the original datagram to A across the foreign network.
4. When A sends IP traffic to X, it uses X's IP address. Each datagram is sent by A to a router on the foreign network for routing to X.
5. The IP datagram from A to X travels directly across the internet to X, using X's IP address.

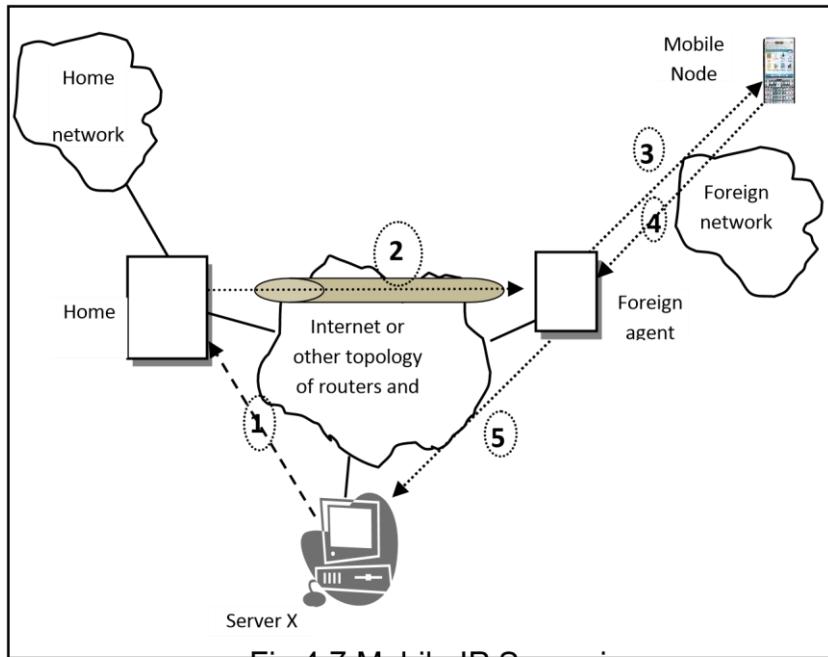


Fig 4.7 Mobile IP Scenarion

To support the operations given in the figure, Mobile IP includes three basic capabilities:

Discovery: A mobile node uses a discovery procedure to identify prospective home agents and foreign agents.

The mobile node is responsible for an ongoing discovery process. It must determine the case(in home network or in foreign network) in which IP datagrams may be received without forwarding. As handoff from one network to another occurs at physical layer, a transition from home network to foreign network can occur at any time without notification to the network layer. For discovery a router can act as an agent to issue router advertisement ICMP message. The router advertisement message includes the IP address of the router and router's role as an agent. A mobile node listens for these agent advertisement messages. Because a foreign agent could be on mobile node's home network, the arrival of an agent advertisement does not necessarily tell the mobile node that it is on a foreign network. The mobile node must compare the network portion of the router's IP address with the network options of its own home address. If these network portions do not match, then mobile node is on a foreign network.

Registration: A mobile node uses an authenticated registration procedure to inform its home agent of its care of address.

Once a mobile node has recognized that it is on a foreign network and has acquired a care-of-address, it needs to alert a home agent on its home network and request that the home agent forward its IP traffic. The registration process involves four steps:

1. The mobile node requests the forwarding service by sending a registration request to the foreign agent that the mobile node wants to use.
2. The foreign agent relays this request to the mobile node's home agent.
3. The home agent either accepts or denies the request and sends a registration reply to the foreign agent.
4. The foreign agent relays this reply to the mobile node.

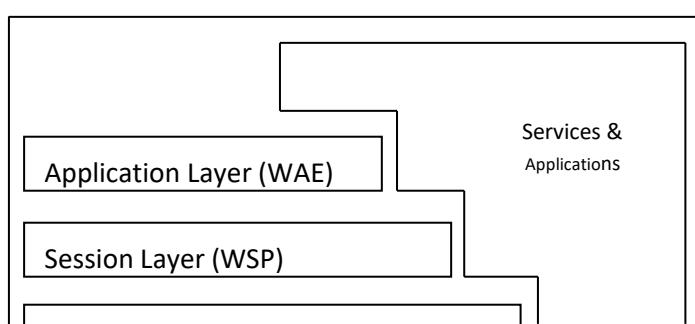
Tunneling: It is used to forward IP datagrams from a home address to care-of-address.

Once a mobile node is registered with a home agent, the home agent must be able to intercept IP datagrams sent to the mobile node's home address so that these datagrams can be forwarded via tunneling. The home agent needs to inform other nodes on the same network that IP datagrams with a destination address of the mobile node should be delivered to this agent. To forward an IP datagram to a care-of-address, the home agent puts the entire IP datagram into an outer datagram by making encapsulation.

MODULE-II

WAP (Wireless Application Protocol)

WAP is a universal open standard developed by the WAP forum to provide mobile users of wireless phones and other wireless terminals. WAP standard represents the first successful attempt to establish a broadly accepted environment for delivering information, data and services to both enterprise and consumer users over wireless networks. WAP is based on existing internet standards such as IP, XML, HTML and HTTP. It also includes security features.



WAP Architecture

The WAP standard is a set of standards which together define how wireless data handset communicates with the wireless network and how contents and services are delivered and executed to their handsets. Using these standards the handset can establish a connection to a WAP data infrastructure, request content and services from infrastructure. WAP is based on layered architecture. The WAP stack is similar to the OSI network model. The architecture consists of a set of services encompassing network protocol, security and application environment. The WAP standard is layered i.e. one layer rests on top of another and therefore depend on another to provide services. Each layer in the network infrastructure are exposed to the interface layer above it and also exposed an interface directly to application and services. These layers are symmetric i.e. they run both on the client device and network infrastructure.

Fig 11.1 WAP Protocol Stack

Components of WAP Standard Bearer Adaptation

Wireless networks employ a variety protocols for exchanging messages, packets, or frames to and from the client device. Dozens of these network protocols, also known as bearer protocols, exist. Each bearer protocol is associated with a particular type of network infrastructure, and each type of infrastructure typically associated with a particular set of suppliers or with particular regions of the world. Examples of various bearers are AMPS, CDPD, CDMA, GSM etc

WDP (Wireless Datagram Protocol)

The transport layer protocol in the WAP architecture is referred to as WDP. This layer operates above the bearer services and offers a consistent service to upper layers.

WDP offer similar functions like UDP through T-SAP.

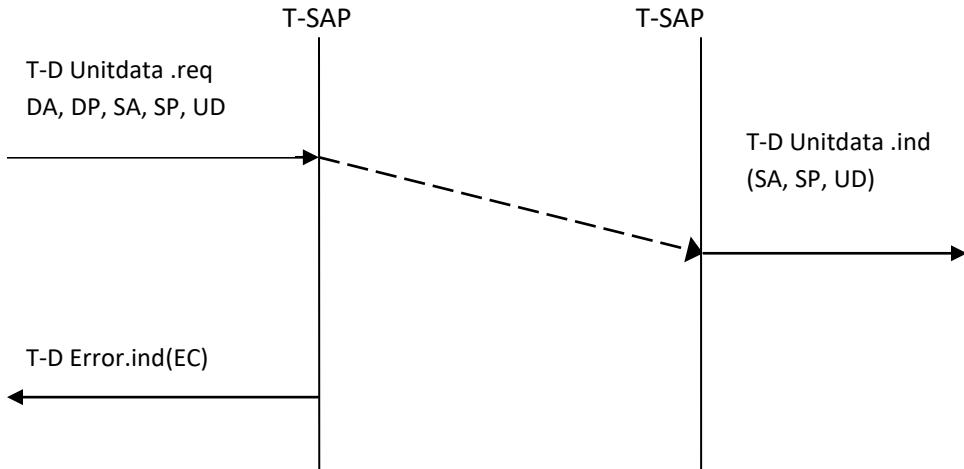


Fig 11.2 WDP Service Primitives

WDP offers source and destination port numbers used for multiplexing and demultiplexing of data. The service is initiated by sending a datagram. T-D Unitdata.req with Destination Address (DA), Destination Port (DP), Source Address (SA), Source Port (SP) and user data (UD). DA and SA are unique address of receiver and sender which may IP address or PSTN number. T-D Unitdata.ind service indicates the reception of data. Here DA, DP are optional. If that request can't be fulfilled by WDP, then an error is indicated with T-D Error.ind service. An error code (EC) is returned indicating reason for the error to higher layer. If any error occurs due to unsuccessful transmission of WDP data grams then WDP uses an error control mechanism known as WCMP (Wireless Control Message Protocol). WCMP is used by WDP nodes to report errors.

WTLS (Wireless Transport Layer Security)

WTLS is a security protocol based on TLS. If requested by an application, a security service (WTLS), can be integrated into the WAP architecture on top of WDP.

WTLS can provide different levels of security for privacy, authentication, and data integrity. WTLS takes into account the low processing power and very limited memory capacity. WTLS takes into account the low processing power and very limited memory capacity of the mobile devices for cryptographic algorithms. WTLS supports datagram and connection oriented transport layer protocols.

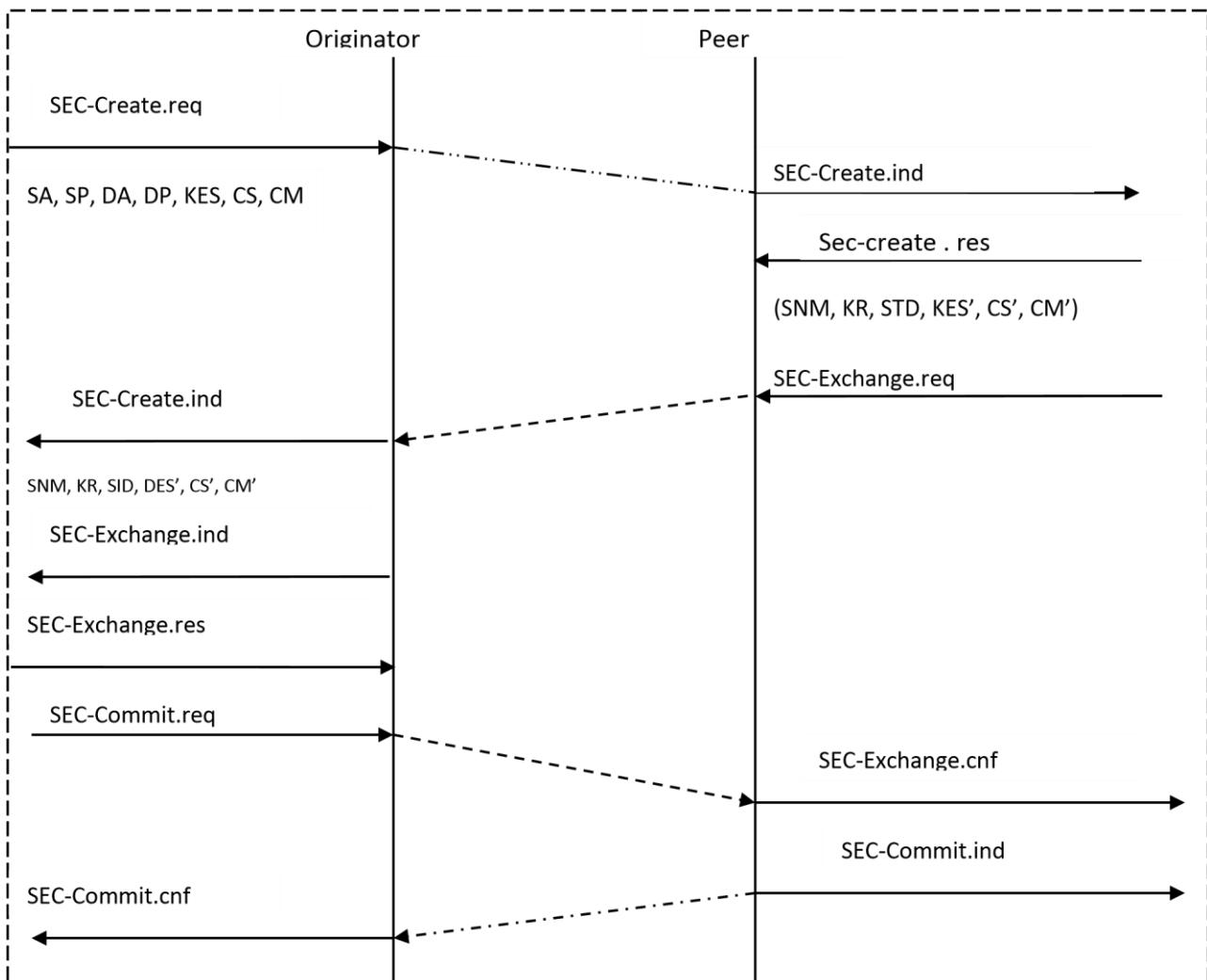


Fig 11.3 Establishment of Secure Session in WTLS

WTLS divides the whole transaction by two sections: originator and peer. First step is to initiate the session with SEC-Create.req. Parameters are SA, SP of the originator and DA, DP of peer. The originator proposes a key exchange suite KES (e.g. RSA), a cipher suit (CS) (for ex: DES), a compression method CM. The peer answers with parameter for SNM (Sequence Number Mode), KR (Key Refresh) cycle (i.e how after keys are refreshed within this secure session), the SID (session identifier) which is unique with each peer and DES', CS', CM'. Peer issues a SEC-exchange primitive. This indicates that peer wishes to perform public key authentication with the client i.e peer request a certificate from the originator.

In the first step of secure session, the negotiation of the security parameters is indicated on the originator's side followed by the request for a certificate. The originator answers with its certificate and issues a SEC-Commit.req primitive. This primitive indicates that the

handshake is completed for originator's side and originator want to switch into new transaction .The certificate delivered to the peer side and SEC-Commit is indicated. This concludes the full handshake for secure session setup. After setting up a secure connection between two peers, user data can be exchanged. This is done using SEC-Unitdata primitive.

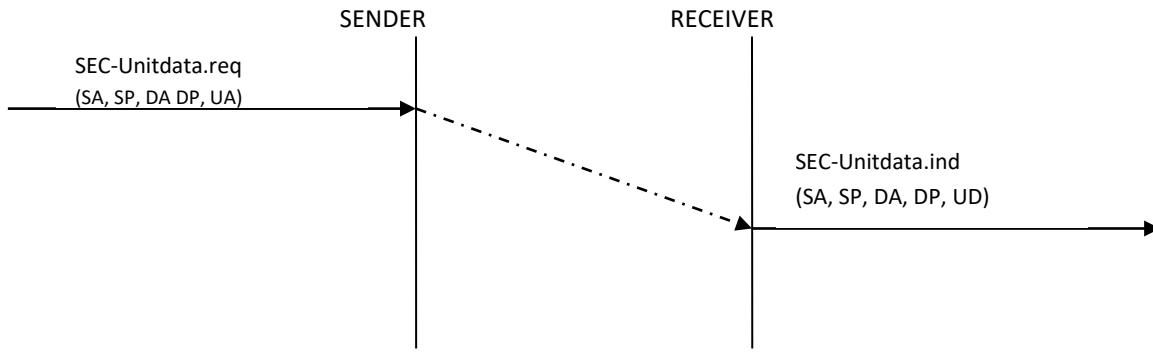


Fig 11.4 WTLS Datagram Transfer

WTP (Wireless Transaction Protocol)

WTP runs on top of a datagram service and provides transaction oriented protocol that is suitable for implementation in thin clients such as mobile phones. WTP offers following advantages:

- Improved reliability over datagram services
- Improved efficiency over connection oriented services.
- Support for Transaction oriented services.

WTP supports three classes of transaction service:

- Class 0: Unreliable with no result message
- Class 1: Reliable with no result message
- Class 2: Reliable with one result message

WTP achieves reliability using duplicate removal, retransmission, acknowledgements and unique transaction identifier. WTP allows for asynchronous transaction, abort of transaction, concatenation of messages and can report success or failure of reliable messages. For reliable transmission of messages three service primitives are offered by WTP are as follows:

- TR-invocation: It is used to initiate a transaction.
- TR-result: It is used to send back a result of a previously initiated transaction.
- TR-abort: It is used to abort a normal transaction.

WTP Class-0 (Unreliable with no result message)

It offers an unreliable transaction service without a result message. The transaction is stateless and can not be aborted. The service is requested with TR-Invoke. The parameters used in the request are Source Address (SA), Destination Address (DA), Source Port (SP), and Destination Port (DP).

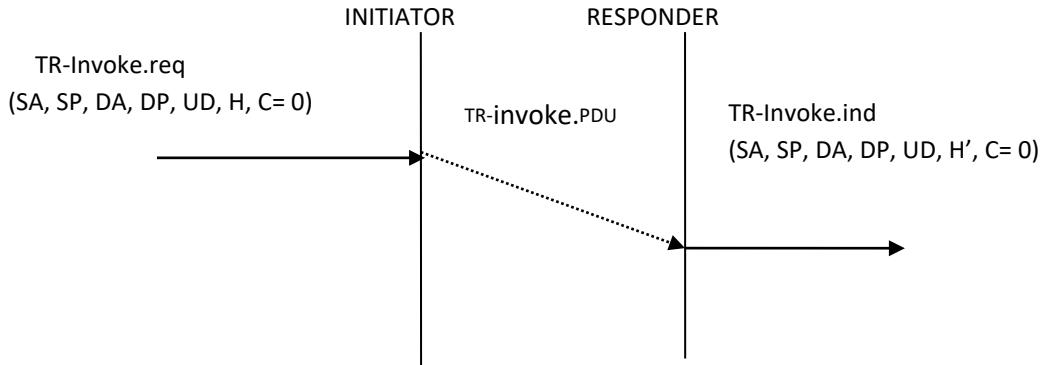


Fig 11.5 Class 0 Basic Transaction

The WTP entity at the initiator sends an invoke PDU which the responder receives. The WTP entity at the responder then generates a TR-Invoke.ind primitive with same parameters. In this class responder does not acknowledge the message and initiator does not perform any retransmission.

WTP Class-1(Reliable with no result message)

It offers reliable transaction service with no result message. The initiator sends an invoke PDU after a TR-Invoke.req from a higher layer. This time class equals to 1 and no user acknowledgement has been selected. The responder signals the income invoke PDU by a TR-Invoke to the higher layer. This specification allows the user on the responder side to send acknowledgement. For initiation, transaction ends with reception of acknowledgement. The responder gives the transaction state for some time to be able to retransmit the acknowledgement if it receives the same invoke PDU again.

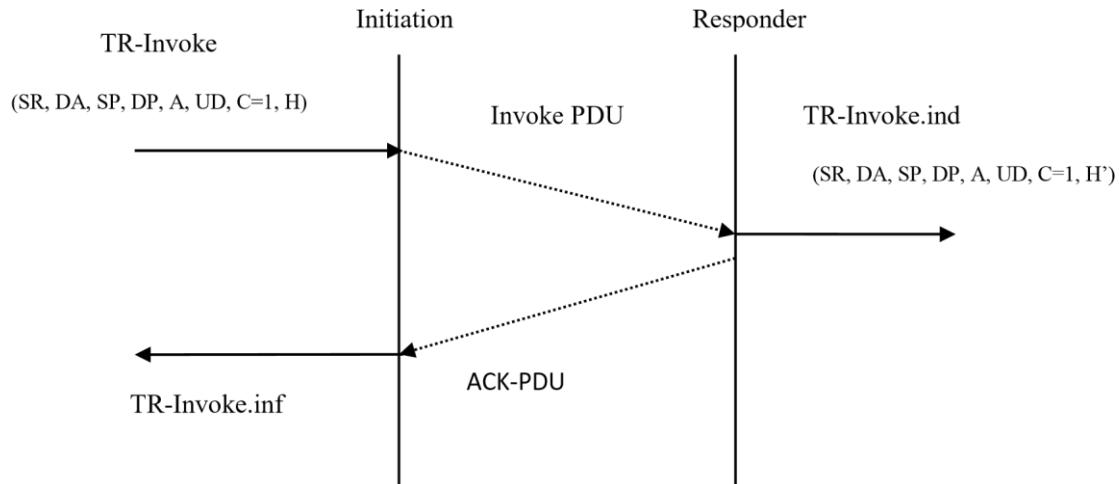
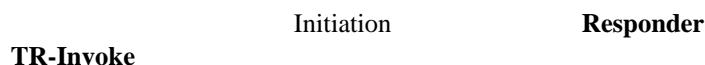


Fig 11.6 Class 1 Basic Transaction

WTP Class-2(Reliable with one result message)

Class 2 transaction service provides the reliable request/response transaction. A user on initiation side sends an invoke PDU to the responder. The WTP entity on responder's side indicates the request with TR-Invoke.ind. The responder now waits for processing of the request. The user on responder side can give the result to the WTP entity on the responder side using TR-result-request. The result PDU can now send back to initiator which implicitly acknowledges the invoked PDU. The initiator can indicate successful transmission of invoke message and result with 2 services TR-Invoke.cnf and TRResult.ind. A user may respond to this result with TR-Result.res. An acknowledgement PDU is then generated which finally triggers the TR-Result.cnf primitive on the responder's side



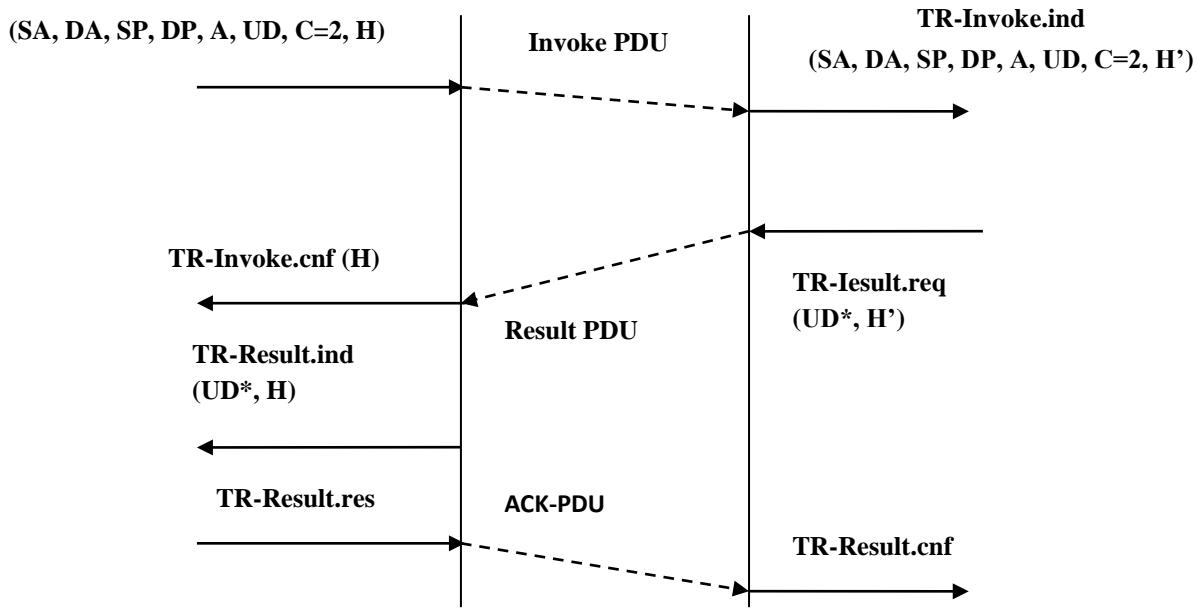


Fig 11.7 WTP Class 2 Transaction

WSP (Wireless Session Protocol)

The WSP provides a connection oriented service on top of WTP. It provides a consistent interface between two session services (client and server). WSP protocol is based on concept of request and a reply. Each WSP protocol data unit consists of a body (containing WML, WML script) and a header (containing information about the data). WSP provides applications with in interface for two session services. The connection oriented session services operates above WTP and connectionless session service operates above WDP. WSP offers following general features need for content exchange:

Session Management: WSP introduces a session that can be established form a client to a server and may exist for a long time. It is responsible for suspending and resuming a session of a mobile during its operation.

Capability Negotiation: Client and server can agree upon a common level of functionality to set upon a common level of functionality to set the capacity of a mobile depending upon parameters such as client size, server size and maximum request it can handle.

Content Encoding: It defines encoding for transferring the contents.

Session establishment involves the exchange of S-Connect primitives. A WSP user acting as a client requests a session with a WSP user acting as a server by issuing an S-Connect.req. Parameters are Server Address (SA), Client address (CA), client header

(CH), requested capabilities (RC). The client transfers the PDU to server where SConnect.ind primitive indicates a new session. The server accepts the new session and answers with S-Connect.res. The parameters are Server Header (SH) and negotiated capabilities (NC). The PDU is then transferred to client. S-Connect.cnf confirms the session establishment.

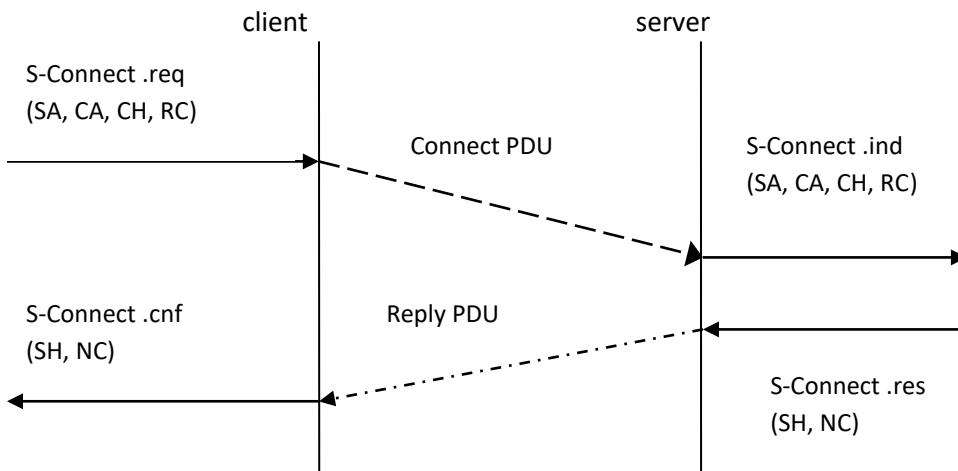


Fig 11.8 WSP Session Establishment

Terminating a session is done by using the S-Disconnect.req service primitive. These primitive aborts all current method used to transfer data. Disconnection is indicated on both sides using S-Disconnect.ind. The reason for disconnection can be network error, protocol error, congestion, maximum packet size exceeded etc.

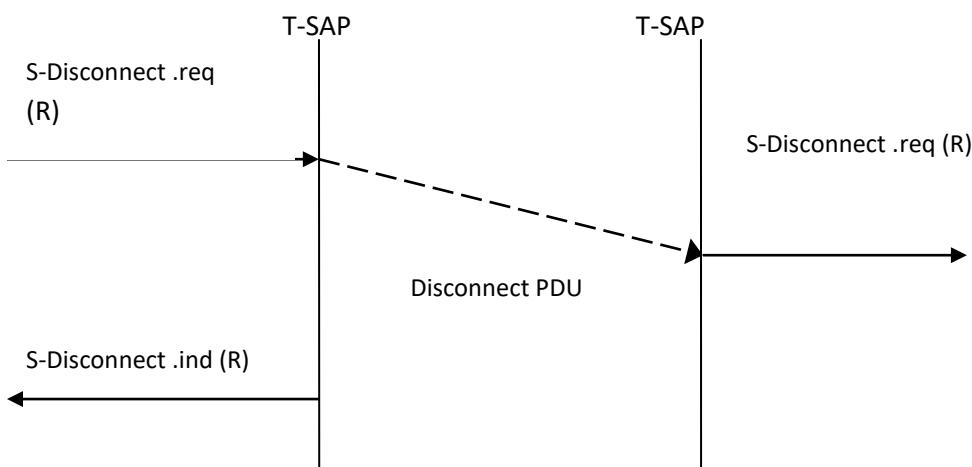
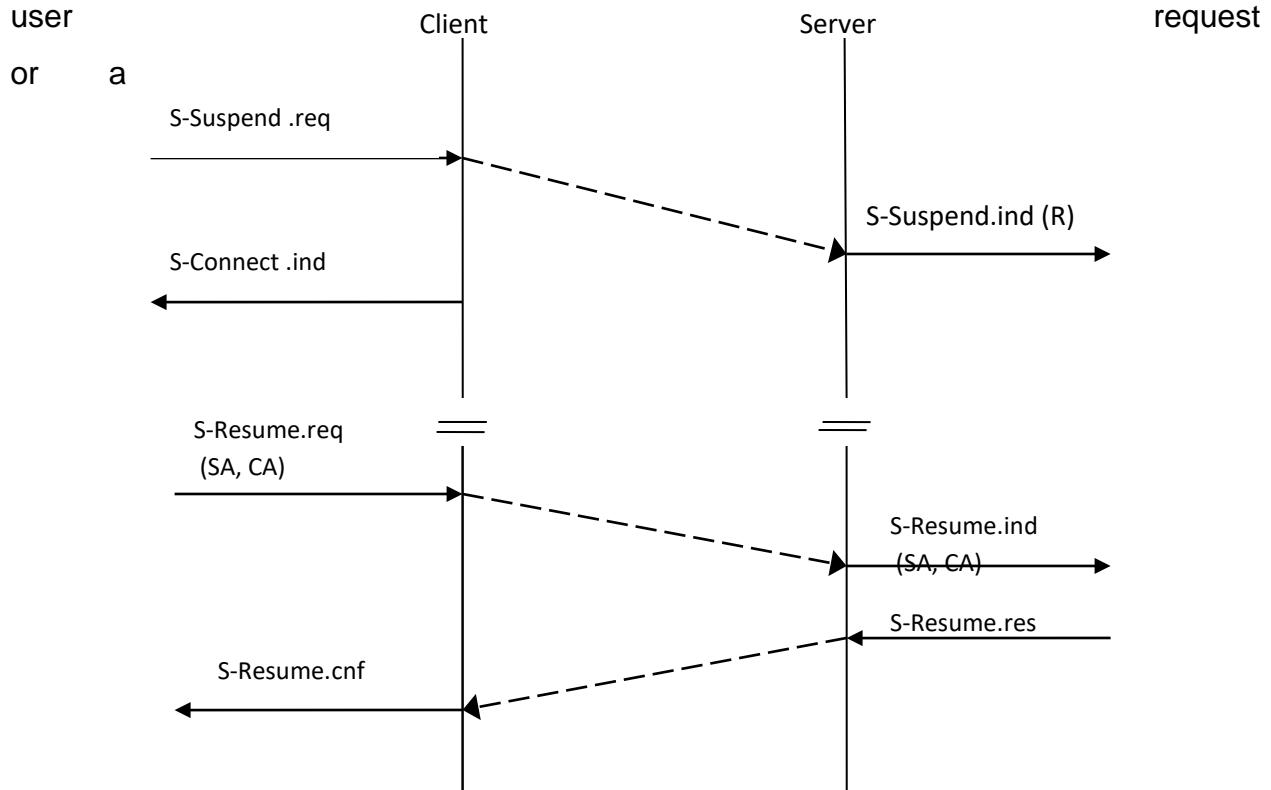


Fig 11.9 WSP Session Termination

WSP Supports session suspend and resume. If a client notices that the bearer network will be unable due to roaming to another network or the user switches off the device. The

client can suspend the session. Session suspension will automatically abort all data transmission and freeze the current state of the session on the client and server side. A client suspends a session with S-Suspend.req, WTP transfers the suspend PDU to sever with class 0 transaction. SP will signal the suspension with S-Suspend.ind on the client and server side. The only parameter is the reason R for suspension. Reasons can be a



suspension initiated by the service provider. A client can later resume a suspended session with S-Resume.req. The parameters are SA, CA (Client Address).

Suspend PDU

Resume PDU

Resume PDU

Fig 11.10 WSP session suspension and resume

WAE (Wireless Application Environment)

The objective of WAE is to provide an interoperable environment to build wireless services among operators and service providers. It offers a framework for the integration of different World Wide Web and mobile telephony application. The major elements of WAE are:

Wireless Telephony Application (WTA): A collection of telephonic features for call mechanism. Using WTA, application developers can use micro browser to originate telephone calls.

Content Generator: Application or services on origin servers that produce standard content formats in response to requests from user agents in the mobile terminal.

User Agent: The user agent signifies an agent who works on behalf of the user. In WWW and WAE content user agent is the user facing browser software.

Wireless Markup Language (WML): It is a mark-up language optimized for use in wireless devices similar to HTML.

The goal of WAE is to minimize over the-air traffic and resource consumption on handheld device. This goal can be implemented using WAE logical model. In logical model, a client uses an encoded request for an operation on a remote server. Encoding is necessary to minimize data sent over the air and to save resources on the handheld device. Decoders in a gateway translate the encoded request into a standard request as understood by origin server. The gateway transfers this request to the origin server. Origin server will respond to the request. The gateway encodes the response and transfers the encoded response with the content to the client.

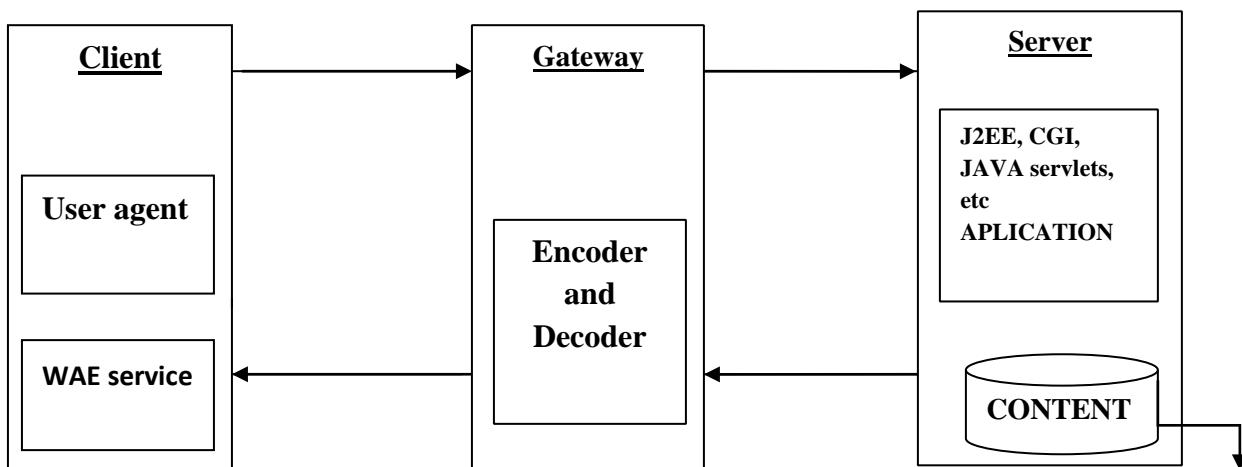


Fig 11.11 WAE Logical Model

WML (Wireless Markup Language)

WML was designed to describe content and format for presenting data on devices with limited bandwidth, limited screen size, and limited user input capability. It is designed to work with telephone keypads, styluses, and other input devices common to mobile, wireless communication. WML permits the scaling of displays for use on twoline screens found in some small devices, as well as the larger screens found on smart phones.

For an ordinary PC, a web browser provides contents in the form of web coded with the Hypertext Markup Language (HTML). To translate an HTML-coded webpage into WML with content and format suitable for wireless devices, much of the information, especially graphics and animation, must be stripped away. WML presents mainly textbased information that attempts to capture the essence of web page and that is organized for easy access for users of mobile devices.

Important features of WML include the following:

- **Text and image support:** Formatting and layout commands are provided for text and limited image capability.
- **Deck/card organizational metaphor:** WML documents are subdivided into small, well-defined units of user interaction called cards. Users navigate by moving back and forth between cards. A card specifies one or more unit of interaction (a menu, a screen of text, or a text entry field). A WML deck is similar to an HTML page in that it is identified by a web address (URL) and is the unit of content transmission.
- **Support for navigation among cards and decks:** WML includes provisions for event handling, which is used for navigation or executing scripts.

In an HTML-based web browser, a user navigates by clicking on links. At a WMLcapable mobile device, a user interacts with cards, moving forward and back through the deck.

WML is tagged language, similar to HTML, in which individual language elements are delineated by lowercase tags enclosed in angle brackets. Typically, The WML definition of a card begins with the non visible portion, which contains executable elements, followed the visible content. As an example, consider the following simple deck with one card form :

```
<wml>  
    <card id='card1'>  
        <p>  
            Hello WAP World.  
        </p>  
    </card>  
</wml>
```

The tags `<wml>`, `<card>`, and `<p>` enclose the deck, cards, and paragraph, respectively. Like HTML, most elements end with a terminating tag that is identical to the starting tag with the addition of the character “\”. When a wireless device receives this code, it will display the message “hello WAP world” on the terminal’s screen.

WML Script

WML Script is a scripting language with similarities to JavaScript. It is designed for defining script-type programs in a user device with limited processing power and memory.

WLL (Wireless in Local Loop)

Wireless local loop (WLL) provides two-way calling services to the stationary or “fixed” users, which is intended to replace its wireline Counterpart. It is a system that connects a subscriber to PSTN using wireless technology and use radio signals to provide standard telephone service. It is a broadcast connection system that uses high frequency radio links to deliver voice and data without fiber optic cables.

In telephony, loop is defined as the circuit connecting a subscriber’s station (e.g., telephone set) with the line terminating equipment in a central office (a switch in the telephone network). The trunks start from the central office in the loop, and are broken down into several smaller bundles of circuits after some distance from the central office. These circuits are eventually separated into individual drops for the residence houses. The central office switch is typically the first point of traffic concentration in the public switched telephone

Network (PSTN). Newer installations use fiber optics to connect residential neighborhoods or business campuses to the central office and statistical multiplexers to concentrate traffic.

Architecture

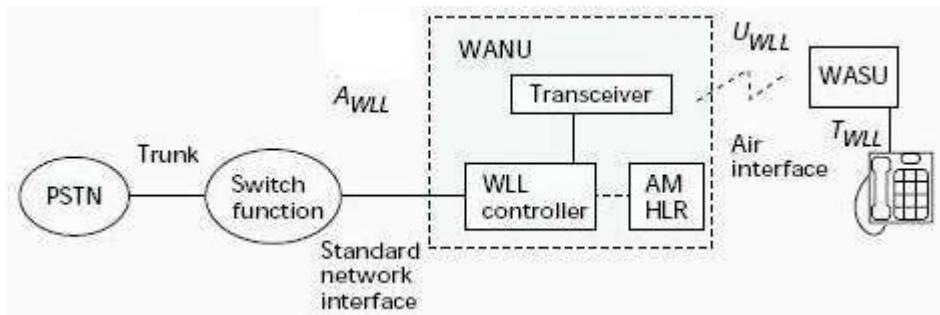


Fig 12.2 WLL Architecture

A simplified version architectural reference model for WLL is shown in Fig. In this figure, the wireless access network unit (WANU) consists of the base station transceivers (BTS) or radio ports (RP), the radio controller (RPCU), an access manager (AM), and home location register (HLR), as required. The interface between the WANU and the switch is called A_{WLL} . The air interface between the WANU and the user side is called U_{WLL} . The WANU should provide for the authentication and privacy of the air interface, radio resource management, limited mobility management, and over-the-air registration of subscriber units (SUs). It may also be required to provide Operation and Maintenance (OAMP), routing, billing and switching functions as appropriate or necessary. The WANU also provides protocol conversion and transcoding of voice and data. The wireless access subscriber unit (WASU) provides an air interface toward the network and a “traditional” interface T_{WLL} to the subscriber. This interface includes protocol conversion and transcoding, authentication functions, local power, OAMP, dual tone multi frequency (DTMF), dialtone. In this reference model the switching fabric (SF) can be a digital switch with or without Advanced Intelligent Network (AIN) capability, an ISDN switch, or a mobile switching center (MSC).

It consists of 3 major components.

WANU (Wireless Access Network Unit)

It consists of several BST or radio parts, a RPCU (Radio port control unit), an AM (access Manager, Responsible for working of RPCU), a HLR. It is responsible authentication, air registration of Subscriber. It may also be required to provide OAM, routing, billing, switching, protocol conversion Trans-coding of voice and data.

WASU (Wireless access subscriber unit)

It provides an air interface “Uwll” towards n/w and “Pwll” interface towards subscriber. It is responsible for voice trans-coding authentication and signaling function.

SF (Switching Fabric)

It is associated with a switch that can be a digital switch, MSC, ISDN switch. The transmission between WANU and SF can be leased line, microwave or optical fiber.

Switch is connected to WANU through “Awll” interface.

The Wireless Local Loop Technologies

The WLL systems are typically based on one of the following four technologies

Satellite-Based Systems: These systems provide telephony services for rural communities and isolated areas such as islands. These systems can be of two types:

- Technology designed specifically for WLL applications
- Technology piggybacked onto mobile satellite systems as an adjunct service

Of these, the former offers quality and grade of service comparable to wireline access, but it may be expensive. The latter promises to be less costly but, due to bandwidth restrictions, may not offer the quality and grade of service comparable to plain old telephone service (POTS).

An example of a satellite based technology specifically designed for WLL is the HNS telephony earth station (TES) technology. This technology can make use of virtually any geostationary earth orbit (GEO) C-band or Ku-band satellite. Satellite technology has been used to provide telephony to remote areas of the world for many years. Such systems provide an alternative to terrestrial telephony systems where land lines are not cost effective or where an emergency backup is required. There are many proposed systems for mobile satellite service, including the Inmarsat International Circular Orbit (ICO) system, Iridium, Globalstar, Odyssey, American Mobile Satellite Corporation (AMSC), Asia Cellular Satellite (ACeS) and Thuraya mobile satellite system. These systems are specialized to support low-cost mobile terminals primarily for low bit rate voice and data applications.

Cellular-Based Systems: These systems provide large power, large range, median subscriber density, and median circuit quality WLL services. Cellular WLL technologies are primarily used to expand the basic telephony services. Typically, they operate in the mobile frequency bands at 800-900 MHz, 1.8- 1.9 GHz, and sometimes at 450 MHz or 1.5 GHz .

This approach offers both mobility and fixed wireless access from the same cellular platform. Cellular systems are optimized for high-tier coverage. Support for mobiles traveling in excess of 100 mph and cell sizes up to 10 mi in radius is required. To achieve the above goals, extensive signal processing is required, which translates into high delay, high overhead and low user bandwidth. These systems are not well suited to deployment indoors and in picocells. Additional complexity of the air interface with the same low user bandwidth is required.

Low-Tier PCS or Microcellular-Based Systems: These systems provide low power, small range, high subscriber density, and high circuit quality WLL services. These technologies are considered to facilitate rapid market entry and to expand the capacity of the existing infrastructure. They are typically operated at 800 MHz, 1.5 GHz, 1.8 GHz, and 1.9 GHz frequency bands.

Compared with the cellular-based WLL, more base stations are required to cover the same service area. Operators may consider low-tier WLL technologies when an existing infrastructure is in place to support backhaul from many base stations to the switch, or when wireline-like services and quality are essential.

Fixed Wireless Access Systems: These systems are proprietary radio systems designed specifically for fixed wireless applications, which may or may not be extensible to PCS or cordless. The primary disadvantage of the cellular approach is its limitation on tollquality voice (new toll-quality vocoders designed for cellular technologies may eliminate this problem), and signaling transparency. The primary disadvantage of lowtier PCS and microcellular approaches is their range. Nonstandard fixed wireless access (FWA) technology can address these issues and become more efficient. FWA systems for zonal areas are designed to cover the local telephone area directly from the PSTN switches. The systems for rural areas provide connection at the remote ends of rural links to the end users.

Examples of WLL Products HNS Terminal Earth Station Quantum System

The HNS terminal earth station (TES) product is used for the Intelsat network to provide remote access telephone service. In total there are approximately 100 TES networks worldwide, in addition to the Intelsat network, and more than 10,000 remote site stations. The TES system is a satellite based telephony and data communications network providing mesh connectivity between multiple earth stations. The system provides call-by-call demand-assigned multiple access (DAMA) circuits and preassigned circuits, via single hop single channel per carrier (SCPC) communications paths between earth stations. It supports both public and private networks, and is capable of operating with any telephony interface from individual subscribers to toll switches and major gateways.

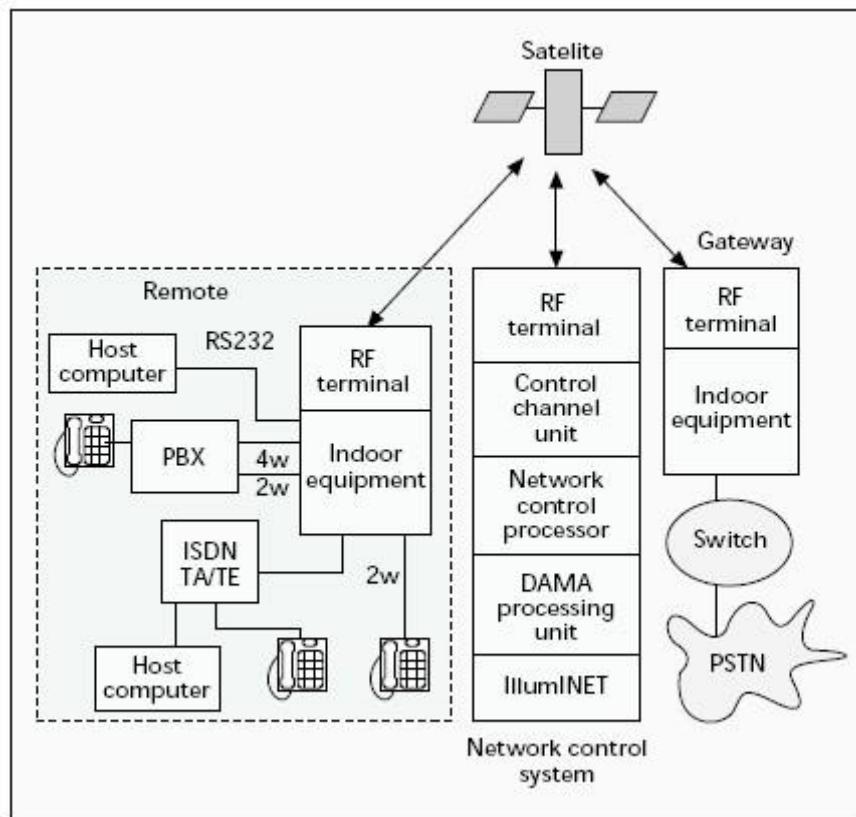


Fig 12.3 TES quantum topology.

Lucent Wireless Subscriber System

Lucent Wireless Subscriber System (Lucent WSS) is a cellular-based WLL. This system typically supports 800-5000 subscribers per switch controlled area (the capacity can be

enhanced with a more powerful processor).

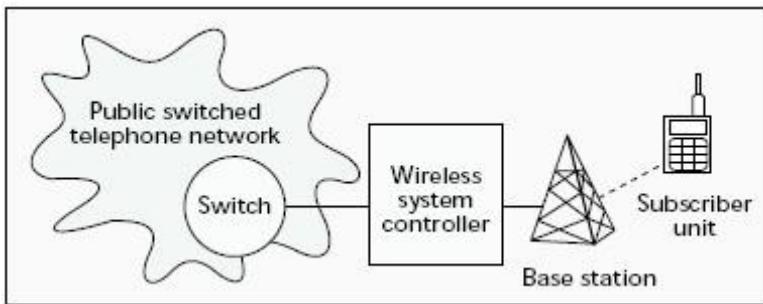


Fig 12.4 The Lucent wireless subscriber system architecture

HNS E-TDMA

E-TDMA is an extension to the IS-136 TDMA standard that provides support for WLL with increased capacity and improved network performance while maintaining the large coverage area feature of other cellular standards.

E-TDMA offers a choice of subscriber unit platforms including single subscriber units (SSU) and multiple subscriber units (MSU) capable of supporting up to 96 lines, depending on the subscriber traffic load and MSU provisioning. The single subscriber unit supports high capacity digital voice, fax, and data transparently using a standard RJ-11 interface, and enables multiple terminal connections as simple extensions on a single access unit or per directory number. Such units are appropriate for locations with low population densities such as residences and small businesses. Multiple subscriber units provide access to the WLL system in areas of high population densities such as hotels and apartment buildings. MSU and radio resources are allocated on a call-by-call basis, thereby reducing the required hardware. Operations and maintenance of both SSUs and MSUs are supported by a full set of remote terminal diagnostic protocols to assess performance and respond to end-user issues. Over the-air activation protocols are also supported to improve system installation. The system supports software downloads to subscriber units to take full advantage of system upgrades remotely.

The PACS WLL System

The ANSI standard personal access communications system (PACS) WLL system is a low-tier PCS-based WLL. The low power PCS technology supports high circuit quality (32 kb/s voice coding) and low latency data with high user bandwidths. PACS is designed to

cover a broad range of venues not optimally served by typical cellular systems, including high density WLL. PACS has features that set it apart from other PCS and cellular standards.

For example, PACS supports both public and private key authentication and privacy. It operates in both FDD and TDD mode, and can interoperate across a wide variety of public and private networks in licensed and unlicensed spectrum. It has a rich suite of data protocols, including packet and circuit protocols. It spans a wide range of venues from large outdoor microcells to indoor picocells.

There are two types of user terminals in PACS/WLL: portable handset (subscriber unit) and fixed access unit. The fixed access units convert the radio signal to an RJ-11 interface signal to the customer premises equipment.

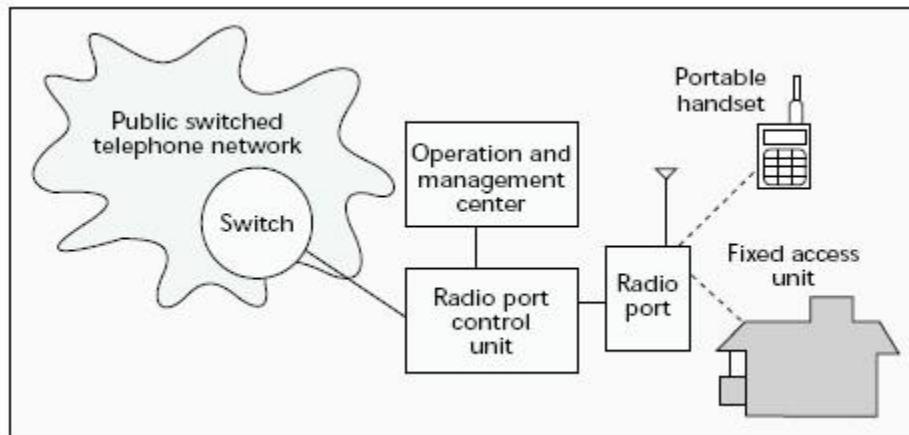


Fig 12.5 The PACS/WLL architecture.

Qualcomm QCTel

Qualcomm's QCTel CDMA WLL System is an FWA WLL. A basic six-sector QCTel system may support 24,000 subscribers. The QCTel technology supports 8 kb/s voice and up to 7.2 kb/s data rate. QCTel supports limited mobility, and the subscriber unit can be a portable handset.

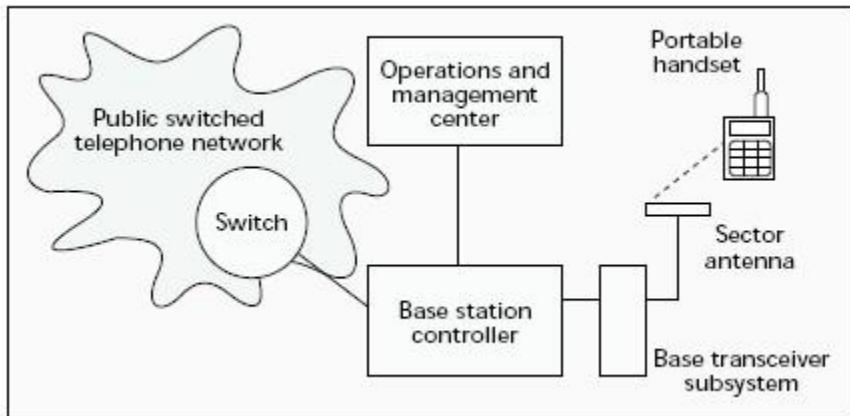


Fig 12.6 The Qualcomm QCTel architecture.

IMT 2000 (International Mobile Telecommunications-2000)

IMT 2000 known as 3G or 3rd Generation, is a family of standards for mobile telecommunications defined by the International Telecommunication Union, which includes EDGE, UMTS, and CDMA2000 as well as DECT and WiMAX. Services include wide-area wireless voice telephone, video calls, and wireless data, all in a mobile environment. 3G allows simultaneous use of speech and data services and higher data rates (up to 14.0 Mbit/s on the downlink and 5.8 Mbit/s on the uplink). Thus, 3G networks enable network operators to offer users a wider range of more advanced services while achieving greater network capacity through improved spectral efficiency. The International Telecommunication Union- Radio communication developed 3G specification to facilitate a global wireless infrastructure. International Mobile Telecommunications-2000 (IMT 2000) is the global standard for 3rd generation mobile communication. IMT 2000 is a standard name used for all 3G systems. Earlier the name was Future Public Land Mobile Telecommunication System (FPLMTS). The number 2000 in IMT 2000 indicates the start of the system in the year (2000+x) and spectrum used (around 2000 MHz). IMT 2000 provides a framework for worldwide access by linking the systems in satellite based networks

Vision

- Common worldwide spectrum
- Multiple radio environment (LAN, satellite, cordless, cellular)
- Worldwide roaming capability
- High quality, enhanced security and performance

- Small terminal for worldwide use
- Integration of satellite and terrestrial system

Services provided

High bearer rate capabilities

- 2 Mbps for fixed environment
- 384 Kbps for indoor and outdoor pedestrian environment
- 144 kbps for vehicular environment

Standardization work

- Europe- ETIS –UMTS (Universal Mobile Telecommunication System)
- Japan- ARIB (Association of Radio Industries and Business) – WCDMA
- USA- TIA(Telecom Industry Association) – CDMA 2000

IMT 2000 Family

ITU standardized five groups of 3G radio access technology

- IMT-DS: Direct spread technology comprises wideband CDMA systems (WCDMA). It is used by European provider. It is specified for UTRA FDD system
- IMT-TC: It stands for time code and specified for UTRA FDD system. It uses time division CDMA.
- IMT-SC: It is a single carrier technology originally promoted by the Universal Wireless Communications Consortium. It is now integrated to the 3GPP efforts. This technology is implemented in DECT.
- IMT-FT: A frequency time technology. It is an enhanced version of the cordless telephone standard DECT.
- IMT-MC: CDMA 2000 is a muti-carrier technology standardized by 3GPP2, which was formed shortly after3GPP to represent the second main stream in 3G technology.

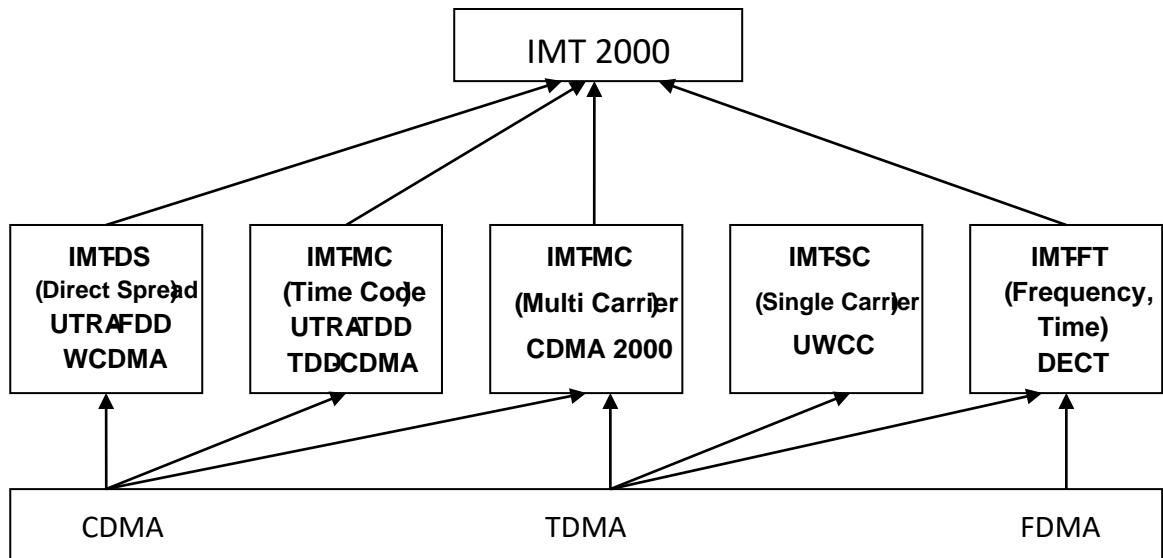


Fig 14.1 IMT 2000 Family

WCDMA

Wideband Code-Division Multiple-Access (W-CDMA) is one of the main technologies for the implementation of third-generation (3G) cellular systems. It is base on radio access technique proposed by ETSI Alpha group and the specifications was finalised 1999. WCDMA supports data rates of 2Mbps or higher for short distances and 384 kbps for long distances. It is also referred to as UTRA-FDD (universal terrestrial radio accessfrequency division duplex). WCDMA access is either FDD or TDD. FDD separates reverse-link AND FORWARD LINK frequencies. These are 1.920-1.980 GHz for uplink and 2.110-2.170 GHz for downlink and each uses a 5MHz bandwidth. It is wider as compared to the 1.25 MHz of the IS-95 SYSTEM.

The key features of WCDMA processing units are as follows:

- Asynchronous base stations
- Employing the same direct sequencing FDD made 1 transmission, same canalization codes (OVSF), same modulation (QPSK), and same carrier modulation (QPSK) for both uplink and downlink.
- Chipping rate of 3.84 MHz
- Multi-rate transmission of signals by spread factor control
- Use of variable data rates.

Protocol Architecture

Protocol architecture consist of 3 layer

- L1-Physical layer
- L2-Datalink layer
- L3-Network layer

The physical layer of WCDMA uses DSSS technique. Physical layer supports 2 modes operation TDD & FDD. Physical layer offers different transport channels to MAC. A transport channel is characterized by how info is transferred over radio interface. MAC offers different logical channel to RLC sub layer of layer2 (L2). A logical channel is characterized by type of information transferred. L2 is split into following sub layer: RLC, MAC, PDCP, BMC

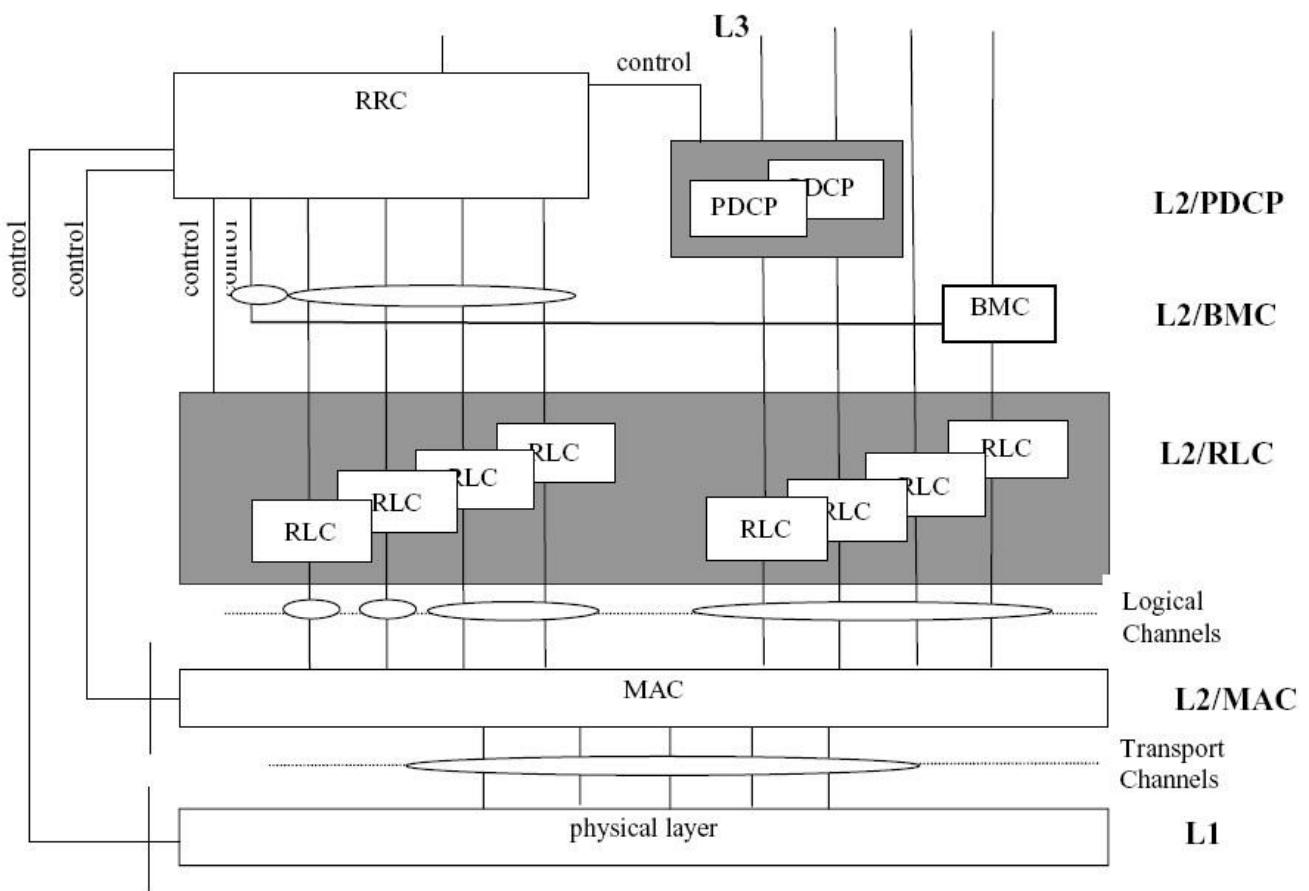


Fig 14.2 WCDMA Protocol Architecture

Logical Channels

MAC layer provides data transfer services on logical channel. A set of logical channel type is defined for different kinds of data transfer services offered by MAC. Each logical channel type is defined by the type of information that is transferred. Logical Channels are classified into two groups:

- Control channel- These channels are used to transfer control information
- Traffic channel:-These channels are used to transfer user information

Table Logical Control Channels

Broadcast Control	Downlink channel for broadcasting system control
Channel(BCCH)	information
Paging Control channel (PCCH)	Downlink channel that transfers paging information and is used when: <ul style="list-style-type: none">• Network does not know the location cell of the mobile station• The mobile station is in the cell connected state(utilizing sleep mode procedures)
Common control channel (CCCH)	Bidirectional channel that transfers control information between network and mobile stations. This channel is used: <ul style="list-style-type: none">• By the mobile stations having no RRC connection with the network• By the mobile stations using common transport channels when accessing a new cell after cell reselection
Dedicated Control channel (DCCH)	Point-to-point bidirectional channel that transfers control information between a mobile station and the network. This channel is established through RRC connection set up procedure

ODMA Common Control Channel (OCCCH)	Bidirectional channel for transmitting control information between mobile stations
ODMA Dedicated Control Channel (ODCCH)	Point-to-point bidirectional channel that transmits dedicated control information between mobile stations. This channel is established through RRC connection setup procedure

Table Logical Traffic Channels

Dedicated Traffic Channel (DTCH)	Point-to-point channel, dedicated to one mobile station for the transfer. DTCH can exist both in uplink and downlink.
ODMA Dedicated Traffic Channel	Point-to-point channel, dedicated to one mobile station for the transfer. DTCH can exist both in uplink and downlink. An ODTCH exists in relay link. A point-to-multipoint unidirectional channel for transfer of dedicated user information.

Transport channels

A transport channel is defined by how and with what characteristics data is transferred over the air interface. There exist two types of transport channels:

- Dedicated channels
- Common channels

There is one dedicated transport channel that is dedicated channel (DCH). It is a downlink or uplink transport channel. The DCH is transmitted over the entire cell or over only a part of the cell using beam-forming antennas.

The common transport channels are mentioned in the table below.

Table Common Transport Channels

Broadcast channel (BCH)	Downlink transport channel that is used to broadcast system and cell-specific information. The BCH is always transmitted over the entire cell with a low fixed bit rate.
Forward access channel (FACH)	Downlink transport channel. The FACH is transmitted over the entire cell or over only a part of the cell using beam-forming antennas. The FACH uses slow power control.
Random Access Channel(RACH)	Uplink transport channel. The RACH is always received from the entire cell. The RACH is characterized by a limited size Data field , a collision risk and by the use of open loop power control
Common packet channel(CPCH)	Uplink transport channel. The CPCH is a contentionbased random access channel used for transmission of bursty data traffic. CPCH is associated with a dedicated channel on the downlink, which provides power control for the uplink CPCH.
Downlink shared channel(DSCH)	Downlink transport channel shared by several mobile stations . The DSCH is associated with a DCH
Paging channel(PCH)	Downlink transport channel. The PCH is always transmitted over the entire cell. The transmission of the PCH is associated with the transmission of a physical layer signal, the paging indicator, to support efficient sleep mode procedures.

Physical Channel

The transport channels are mapped on the physical channels. Physical channels are merged in physical layer which consist of radio frame and time slot. The length of radio

frame is 10ms and one frame consists of 15 time slot. Time slot is unit which consists of fields consisting of bits. A time slot is a unit which consist of field containing bits. The number of bits per time slot depends on the physical channel. Normally physical layer consists of 2 uplink and one downlink channel.

Uplink Physical Channels

There are two uplink dedicated and two common physical channels:

- The uplink dedicated physical data channel (uplink DPDCH) and the uplink dedicated physical control channel (uplink DPCCH)
- The physical random access channel (PRACH) and physical common packet channel (PCPCH)

Dedicated Physical Data Channel (DPDCH): It is used to carry dedicated data generated at L2 and above. There may be zero or several uplink DPDCH.

Dedicated Physical Control Channel DPCCH): The uplink DPCCH is used to carry control information generated by Layer-1. Control information consists of pilot bits, transmit power control, feedback information etc.

Physical Random Access Channel (PRACH): It is used to carry the RACH. The random-access transmission is based on a slotted ALOHA approach with fast acquisition indication.

Physical Common Packet Channel (PCPCH): The PCPCH is used to carry the CPCH transport channel. The CPCH transmission is based on DSMA-CD approach with fast acquisition indication.

Downlink Physical Channels

There is one downlink dedicated physical channel, one shared and five common control channels:

- Downlink Dedicated Physical Channel (DPCH)
- Primary downlink Shared Channel (DSCH)
- Primary and secondary common pilot channels (CPICH)
- Primary and secondary common control physical channels (CCPCH)
- Synchronisation Channel (SCH)

Downlink Dedicated Physical Channel (DPCH): On the DPCH, the dedicated transport channel is transmitted time multiplexed with control information generated at layer 1.

Dedicated Shared Channel (DSCH): It is always associated with a downlink DPCH. It is shared by users based on code multiplexing.

Common Pilot Channel (CPICH): It is a downlink physical channel which carries a predefined bit/symbol sequence. There are two types of common pilot channel Primary and Secondary CPICH

- Primary CPICH: There exist one primary CPICH per cell. It is broadcasted over the entire cell
- Secondary CPICH: There may be zero, one or more secondary CPICH. They are broadcasted over only a part of the cell

Common Control Physical Channel (CCPCH): It is a downlink physical channel used to carry the BCH. There are two types of common pilot channel Primary and Secondary CPCCH

- Primary CPCH: It is used to carry the BCH.
- Secondary CPCH: It is used to carry FACH and PCH.

Synchronisation Channel (SCH): This channel is used for searching the cell. The SCH consists of two sub channels:

- Primary SCH consists of a modulated code of length 256 chips.
- Secondary SCH consists of repeatedly transmitting a length 15 sequence of modulated codes of length 256 chips.

CDMA 2000

CDMA2000 is a family of 3G mobile technology standards, which use CDMA channel access, to send voice, data, and signaling data between mobile phones and cell sites. It is also known as also known as IMT Multi-Carrier (IMT-MC) . Cdma2000 specification was developed by the Third Generation Partnership Project 2 (3GPP2), a partnership consisting of five telecommunications standards bodies: ARIB and TTC in Japan, CWTS in China, TTA in Korea and TIA in North America. The set of standards includes:

CDMA2000 1X, CDMA2000 EV-DO Rev. 0, CDMA2000 EV-DO Rev. A, and

CDMA2000 EV-DO Rev. B. All are approved radio interfaces for the ITU's IMT-2000.

CDMA2000 can support mobile data communications at speeds ranging from 144 Kbps to 2 Mbps. Versions have been developed by Ericsson and Qualcomm. As of March 2006, the CDMA Development Group reports more than 250,300,000 subscribers worldwide.

Protocol Architecture

In CDMA 2000, four different protocols are specified.

1. Physical layer-Layer1
2. MAC sub layer-layer2
3. Link Access Control (LAC) sub layer (Layer 2) **CDMA 2000 Physical layer (Layer 1)**

The physical layer is responsible for:

1. Transmitting and receiving bits over the physical medium, which is the air. The bits have to convert into waveforms by modulation.
2. Carrying out coding functions to perform error control functions at the bit and frame levels.

Cdma2000 accepts both signal carrier and multiple career implementations. It also has proposed two kinds of multiplexing: FDD and TDD. The physical layer channels for both FDD and TDD are the same.

Physical Channels:

Physical channels are distinguished in two groups: dedicated and common channel.

Dedicated Physical Channel (DPHCH)

1. **Forward dedicated Physical Channel (F-DPHCH):** There are 4 dedicated channels.

- - Fundamental channel (F-FCH): Provides for transportation of dedicated data.
 - Supplemental Channel (F-SCH): Allocated dynamically to supply a required data rate.
 - Dedicated control channel (F-DCCH): Used to transport mobile specific control information.
 - Dedicated auxiliary pilot channel (F-DAPICH): Used with antenna beam forming and beam-steering to increase coverage or data rate of a desired user. This channel is optional.
- 2. **Reverse Dedicated Physical Channel (R-DPHCH):** There are 3 dedicated channels.
 - Fundamental channel (R-FCH): Same function as F-FCH.
 - Supplemental Channel (R-SCH): Same function as F-SCH.
 - Dedicated control channel (R-DCCH)

Common Physical Channel (CPHCH)

1. Forward Common Physical Channel(F-CPHCH)

- **Pilot channel (F-PICH):** Carries the Pilot symbol and provides capabilities for channel estimation and coherent detection and soft handoff.
- **Common Auxiliary Pilot Channel (F-CAPICH):** Provides a fine tuning on coherent detection and hand off.
- **Sync Channel (F-SYNC):** Provides the mobile station with system information and synchronization.
- **Common Assignment Channel (F-CACH):** Support the reservation access mode on the R-EACH (Enhanced Access Channel).The message that assigns the
- **R-CCCH** is transmitted on the F-CACH

□

- Paging Channel (F-PCH):** It can enable paging functions, also provides a means for short burst data communications. Each mobile is assigned an 80ms slot and decodes periodically to receive page messages. Two channels FBCCH and F-CCCH can substitute it
- **Broadcast Control Channel (F-BCCH):** Serve to broadcast system-specific and cell-specific overhead information.
- **Common Control Channel (F-CCCH):** It provides a means for paging functions and support different data rates for short burst data communications.
- **Quick Paging Channel (FQPCH):** The idea of having F-QPCH is to decrease the time and mobile station needs to monitor the F-PCH or FCCCH. The period at which the mobile station must decide F-PCH or FCCCH as short as 1.28 ms.
- **Common power control of the (F-CPCCH):** Serve two purposes. To allow power control of the R-RCCCH and R-PICH works during the reservation access. To control the P-; ICH when the mobile station is in the traffic state.
- **Packet Data Control (F-PDCH):** A shared packet data channel that supports high-speed operation traffic. Access to this channel is handled through MAC layer scheduling.

2. Reverse Common Physical Channel

- **Access Channel(R-ACH):** Used for mobile stations communications messages to the base station for backward compatibility reasons.
- **Common Control Channel (RCCCH):** To transport control information.
- **Enhanced Access Channel(R-EACH):** An enhanced access product relative to that of the R-ACH.
- **Dedicated Control Channel(R-DCCH):** Same function as F-DCCH.
- **Pilot Channel(R-PICH):** Provides the signal for coherent detection.

□

- **Channel Quality Indicator Channel(R-CQICH):** A support channel for adoptive coding and modulation over the F-PDCH.
- **Acknowledgement channel(R-ACKCH):** Check whether the CRC of the decoded packed has passed or failed.

CDMA 2000 MAC Sub layer (Layer 2)

In the MAC sub layer, there are four different entities. Radio Link Protocol (RLP), signalling Radio Burst Protocol (SRBP), Common Channel Multiplex Sublayer and Dedicated Channel Multiplex sub layer.

- The radio link protocol (RLP) handles user packet data. RLP is performing in the Dedicated Channel Multiplex Sub-layer.
- The Signalling Radio Burst Protocol (SRBP) handles common-channel signalling using radio burst techniques. The SRBP is performing in the common channel Multiplex sublayer.
- The Common Channel Multiplex Sublayer performs the mapping between the logical common channels.
- Dedicated channel multiplex sub layer performs the mapping between the logical dedicated channel (i.e. , those channels are dedicated to specific users) and the physical dedicated cahnnels.

The primary function of the MAC sublayer is to multiplex logical channels before sending and to de-multiplex physical channels into different logical channel after receiving. The two multiplex sublayers of the MAC as mentioned above handle these two functions.

The dedicated channels can be used for both signalling and user data; common channels are only used for signalling. The same arrangement of channels appears in WCDMA. However, the transport channels used in WCDMA for exchanging information between physical layer and logical channels reply MAC layer directly as a means to exchange information between Layer 1 and Layer 2.

□

Primitives

The messages sending and receiving between layers/sub layers are primitives, a form of these communication messages. Two widely used types of primitives are:

Request primitives: A service requester (MS) uses request primitives to request a service or a resource

Indication Primitives: A service provider uses indication primitives to indicate an event requested by service requester has occurred.

Logical channels

The multiplex sub layers, both common channels and dedicated channels, are responsible for the mapping between logical channels and physical channels. The forward-dedicated traffic channel (F-DTCH) Logical Channel Data (common or dedicated) should be reliably delivered from end to end. In executing reliable delivery, the MAC sub layer assembles data received from higher layers and passes the assembled data to the physical layer for transmission. The MAC sublayer also receives data from the physical layer, disassembles the data, and passes the disassembled data to higher layers.

SDU

On the transmit site, the MAC sub layer assembles data blocks received from a higher layer into an SDU and delivers the SDU to the physical layer for transmission. The MAC sub layer receives an SDU, disassembles the SDU into data blocks and delivers them into higher layers.

Adding one or more data blocks with a header can assemble another SDU. All SDUs can be sent either by common channels or dedicated channels.

Multiplex Sublayer's Interaction

Multiplex sub layers can interact not only with physical layer (Layer 1) below, but can interact with four entities above it, RLP and voice service on the dedicated channel side, LAC (Link Access Protocol), and SRBP on the common channel side.

RLP Layer

RLP controls the process of user packet data that travels on the dedicated user channels.

The RLP layer is a Layer 2 protocol that responds for the delivery and receipt of user packet data. An important function of Layer 2 entity is to control packet errors introduced by the physical layer. There are several techniques to control packet data errors.

1. Positive acknowledgement (ACK): Acknowledgement of receiving successfully
2. Negative acknowledgement (NAK): Acknowledgement of receiving unsuccessfully
3. Retransmission: Retransmit when neither an acknowledgement nor a NAK is received.

Other data link control protocols in Layer 2 are:

1. Logical Link Control (LLC): for operating over a LAN using IEEE 802 standards
2. Link Access Protocol: balanced for connecting a device to a packet switched network using the X.25 standard

Three classes of frames on RLP

1. Control frames: carrying control information that have the highest priority
2. Retransmitted data frames: retransmit old data frames
3. New data frames: with lowest priority

Three service types of RLP

RLP 1: Implement packet data services over IS-95A traffic channels, with a data rate of 9.6 or 14.4 kbps

RLP 2: implement packet data service over IS-95B traffic channels, which are fundamental and supplemental code channels.

RLP 3: Implements packet data service over CDMA 2000 traffic channels with a data rate up to 2 Mbps

SRBP (Signalling Radio Burst Protocol) Layer

Functions of SRBP:

- A. The SRBP controls the process of signalling messages that travel on the common signalling channels in the physical layer. There are six forward common signalling channels: F-SYNCH, F-CPCCH, F-CCCH, F-PCH, F-CACH and FBCCH. There are reverse common signalling channels: R-ACH, R-EACH and RCCCH. SRBP is the entity that computes and generates parameters, such as the power level of each successive access probe, the randomization delay of each access sub-attempt for the transmission and reception of common signalling messages.
- B. The SRBP also assembles SDUs for the physical layer to transmit on the physical channels and pass received SDUs from the physical layer to the LAC sub-layer.

MODULE-III

Global Mobile Satellite Systems

Satellite is an object which has been placed into orbit by human endeavor. Such objects are sometimes called artificial satellites to distinguish them from natural satellites such as the Moon.

In general, a satellite is anything that orbits something else, as, for example, the moon orbits the earth. In a communications context, a satellite is a specialized wireless receiver/transmitter that is launched by a rocket and placed in orbit around the earth. There are hundreds of satellites currently in operation. They are used for such diverse purposes as weather forecasting, television broadcast, amateur radio communications, Internet communications, and the Global Positioning System, (GPS).

The first artificial satellite, launched by Russia (then known as the Soviet Union) in the late 1950s, was about the size of a basketball. It did nothing but transmit a simple Morse code signal over and over. In contrast, modern satellites can receive and retransmit thousands of signals simultaneously, from simple digital data to the most complex television programming.

Application

1. Traditional:
 - i. Weather satellite ii. Radio & TV broadcast iii. Military application
2. Telecommunication:
 - i. Global telephone communication ii. Global mobile communication iii. Connection for communication in remote area

Types of Satellite Systems

There are four general system designs, which are differentiated by the type of orbit in which the satellites operate: Geostationary Orbit (GEO), Low-earth Orbit, Medium-earth Orbit (MEO), and Highly Elliptical Orbit (HEO). Each of these has various strengths and weaknesses in its ability to provide particular communications services.

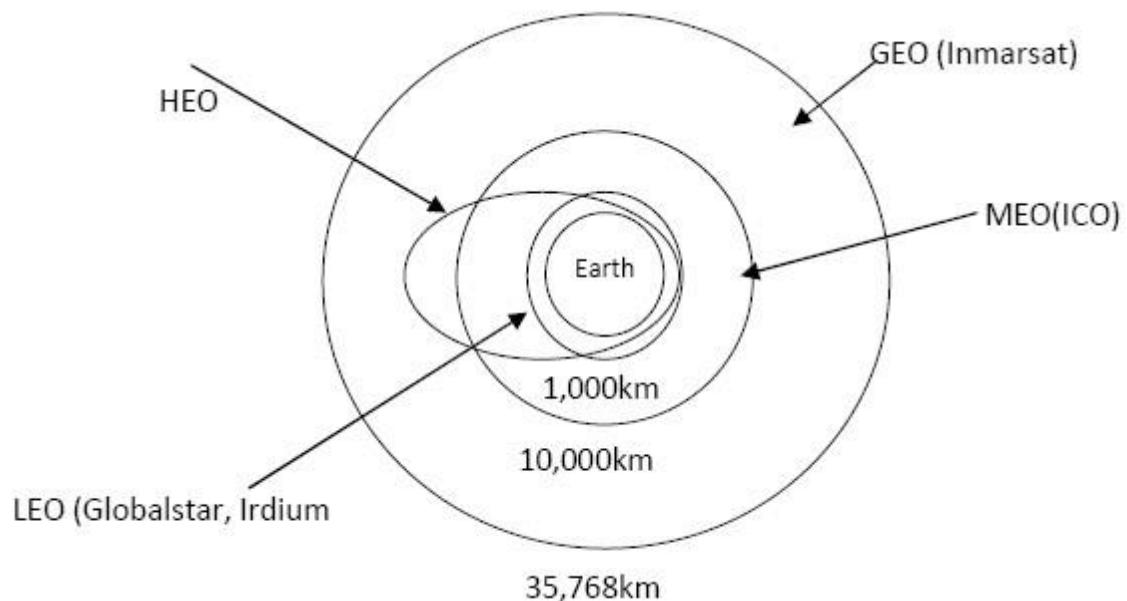


Fig 8.2 Orbits of Different Satellites

Geostationary (GEO)

GEO systems orbit the Earth at a fixed distance of 35,786 kilometers (22,300 miles). The satellite's speed at this altitude matches that of the Earth's rotation, thereby keeping the satellite stationary over a particular spot on the Earth. Examples of GEO systems include INTELSAT, Inmarsat, and

Geostationary satellites orbit the Earth above the equator and cover one third of the Earth's surface at a time. The majority of communications satellites are GEOs and these systems will continue to provide the bulk of the communications satellite capacity for many years to come.

Medium Earth Orbit (MEO)

Stands for medium earth orbit or ICO (intermediate circular orbit)

MEO systems operate at about 10,000 kilometers (between 1,500 and 6,500 miles) above the Earth, which is lower than the GEO orbit and higher than most LEO orbits. The MEO orbit is a compromise between the LEO and GEO orbits. Compared to LEOs, the more distant orbit requires fewer satellites to provide coverage than LEOs because each satellite may be in view of any particular location for several hours. Compared to GEOs, MEOs can operate effectively with smaller, mobile equipment and with less latency (signal delay). Typically, MEO constellations have 10 to 17 satellites distributed over two or three orbital planes

Low Earth Orbit (LEO)

LEO systems fly about 1,000 kilometers above the Earth (between 400 miles and 1,600 miles) and, unlike GEOs, travel across the sky. A typical LEO satellite takes less than two hours to orbit the Earth, which means that a single satellite is "in view" of ground equipment for only a few minutes. As a consequence, if a transmission takes more than the few minutes that any one satellite is in view, a LEO system must "hand off" between satellites in order to complete the transmission. In general, this can be accomplished by constantly relaying signals between the satellite and various ground stations, or by communicating between the satellites themselves using "inter-satellite links."

Highly Elliptical orbit (HEO)

The satellites orbit the Earth in an elliptical path rather than the circular paths of LEOs and GEOs. The HEO path typically is not centered on the Earth, as LEOs, MEOs and GEOs are. This orbit causes the satellite to move around the Earth faster when it is traveling close to the Earth and slower the farther away it gets. In addition, the satellite's beam covers more of the Earth from farther away, as shown in the illustration.

Example- Ellipso, Pentriad

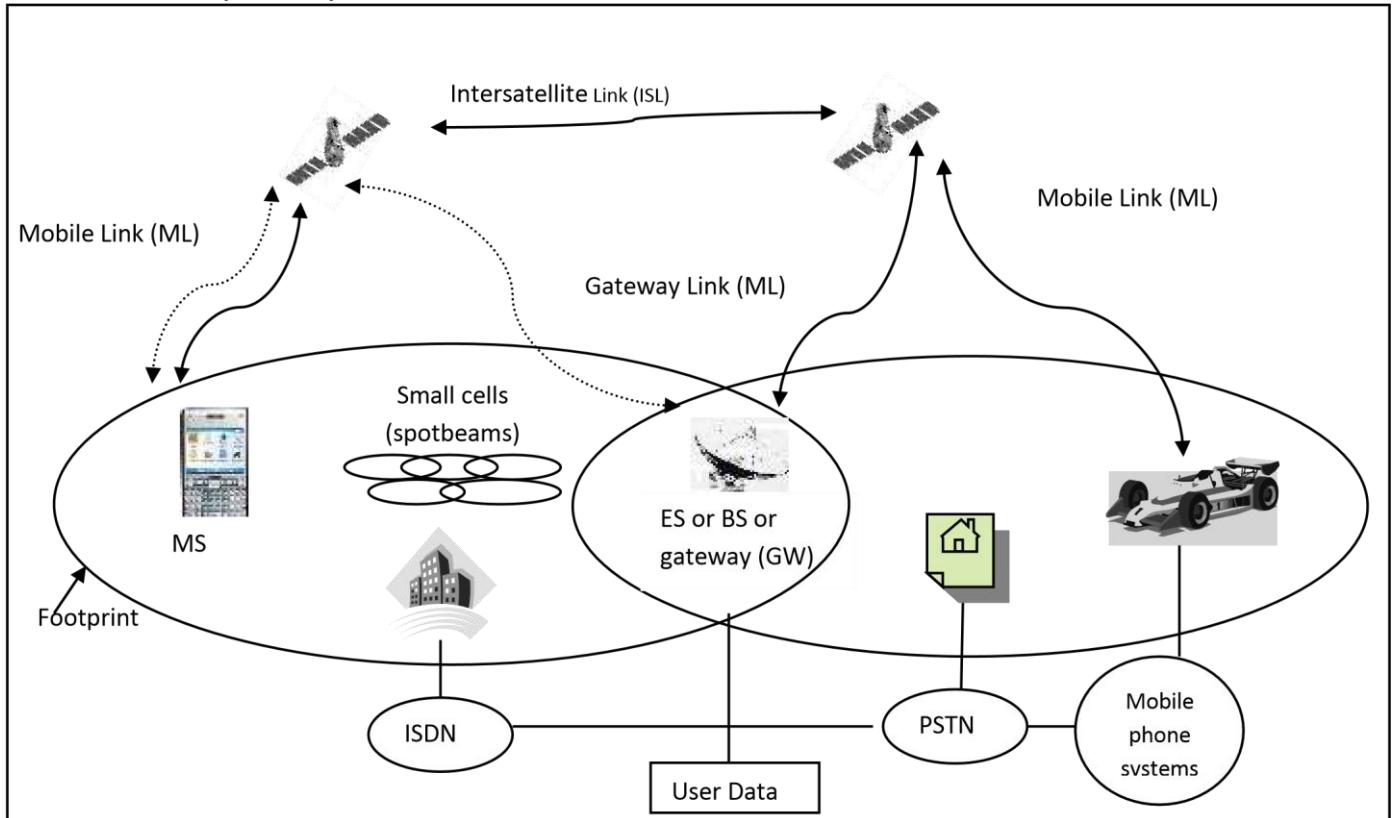


Fig 8.3 A typical Satellite System

CASE STUDIES

IRRIDIUM

It is a satellite based wireless personal communication network designed to permit any type of telephone transmission (voice and data). It is an extension of existing wireless system to provide mobile service to remote areas that are not covered by terrestrial cellular services. It is a LEO satellite.

CONCEPT

The concept of using a constellation of LEO satellite to provide telecommunication services to mobile users was originated by Motorola in 1987. The initial proposal was for 77 satellites in constellation and the system was called IRRIDIUM which has 77 electrons in its orbit. Later on the no of satellite was reduced to 66 which were adequate to provide target services. The 66 satellites are grouped in 6 orbital planes and there are 11 active satellites in each plane with uniform nominal spacing of 32.7° . The satellites have circular orbit at an altitude of 783 km. The satellite travels one side of the earth crossing over near the North Pole and then traveling down the other side. The adjacent planes are called **co-rotating plane** excluding the first and last which is known as **counter-rotating plane**. The angle between co-rotating planes is 31.6° . The angle between counter-rotating planes is 22° . In iridium system, each satellite is equipped with 4 two way communication, 1 each with its neighbors in the same plane and those with in the adjacent planes. Each iridium satellite uses 48 beam antenna patterns and each beam has a minimum diameter of 600 km. In satellite system beams are equivalent to cells associated with mobile system.

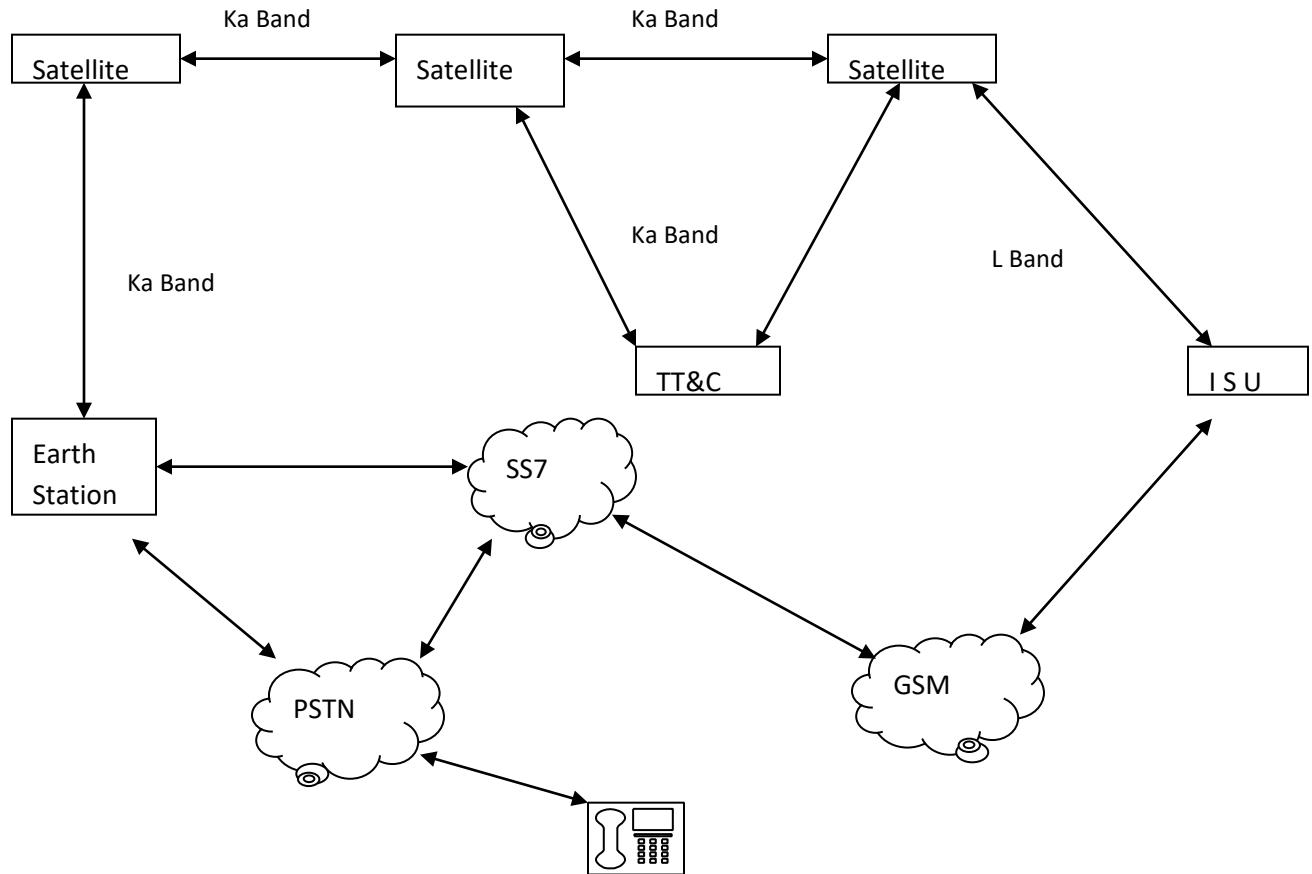


Fig 8.5 Network Architecture for Iridium System

GLOBALSTAR

Globalstar is a low Earth orbit (LEO) satellite constellation for satellite phone and low-speed data communications. The Globalstar project was launched in 1991 as a joint venture of Loral Corporation and Qualcomm.

It is a Global satellite based system on 48 LEO satellites. Initial service was scheduled in late 1999. It doesn't use inter-satellite links but depends on a large number of interconnected earth stations or Gateways for efficient call routing but delivery of the terrestrial network. It is designed to complement the terrestrial mobile network to provide telephony and messaging services to subscribers in locations that are not covered, inadequately covered by wire line/wireless network.

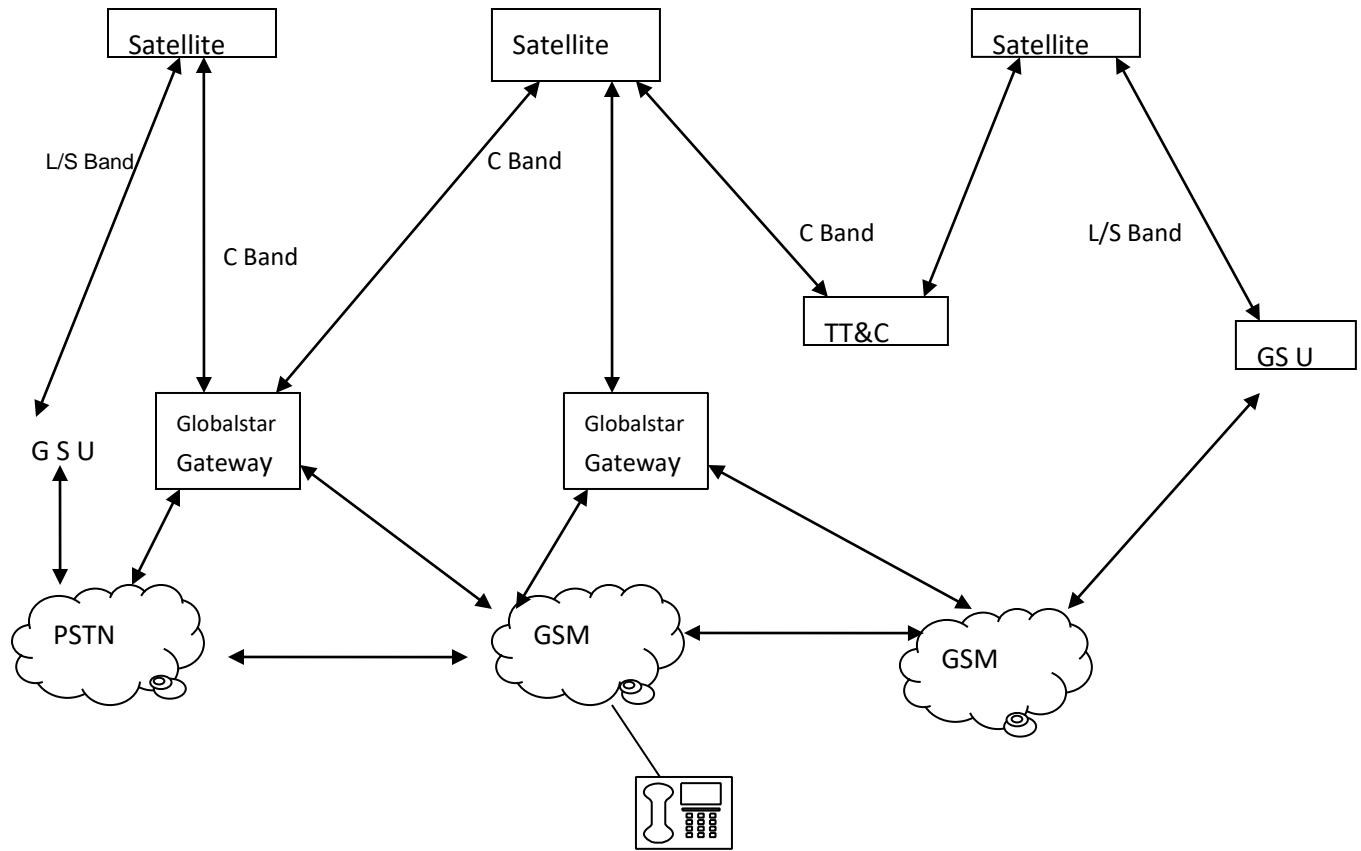


Fig 8.6 Network Architecture for Globalstar System

Concept

Global star constellation of 48 LEO satellites is designed to orbit at an altitude of 1414km above the earth surface in 8 orbital planes inclined at 52° . Each plane is occupied by 6 satellites and nominal weight of each satellite is 450 kg and working life is 7.5 years. Each satellite supports a 16 beam antenna pattern with an average beam diameter of 2250km. In the absence of ISL Globalstar makes maximum use of international terrestrial network (both wireline/wireless). Calls from the subscriber are routed via a satellite to the nearest earth station or gateway and from there they'll be routed over the existing network. A globalstar gateway is designed to serve an area of 3000 km in diameter. Globalstar uses CDMA technology for service link and FDMA/FDD for gateway link.

ICO

The ICO is a medium earth orbit (MEO) mobile satellite system, which is designed primarily to provide services to handheld phones. ICO will use TDMA as the radio transmission technology. ICO has submitted this proposal to the ITU-R evaluation/selection process as a potential candidate RTT for the satellite component of IMT 2000- the third generation mobile telecommunication system being specified by the ITU. The ICO system is planned to go in service in August 2000. The system is designed to offer digital voice, data, facsimile and short targeted messaging services to its subscribers. ICO's primary target customers are users from the existing terrestrial cellular system who expect to travel to locations in which coverage is unavailable or inadequate.

Concept

ICO system is designed to use a constellation of 10 MEO satellites in intermediate circular orbit (ICO), at an altitude of 10,355 km above the earth's surface. The nominal weight of these satellites at launch is less than 2000 kg. The satellites with an expected life of 12 years are arranged in two planes with five satellites (one and one spare) in each plane: orbital planes inclined at 45 degrees relative to the equator. Each satellite has antennas to provide 163 transmit and service link beams. The orbital configuration provides coverage of earth's entire surface at all times and ensures significant overlap so that two or more satellites are visible to the user and the satellite access nodes (SAN) at any time. Further, at least one of the satellites appears at a high elevation angle, thereby minimizing the probability of blocking due to shadowing effects.

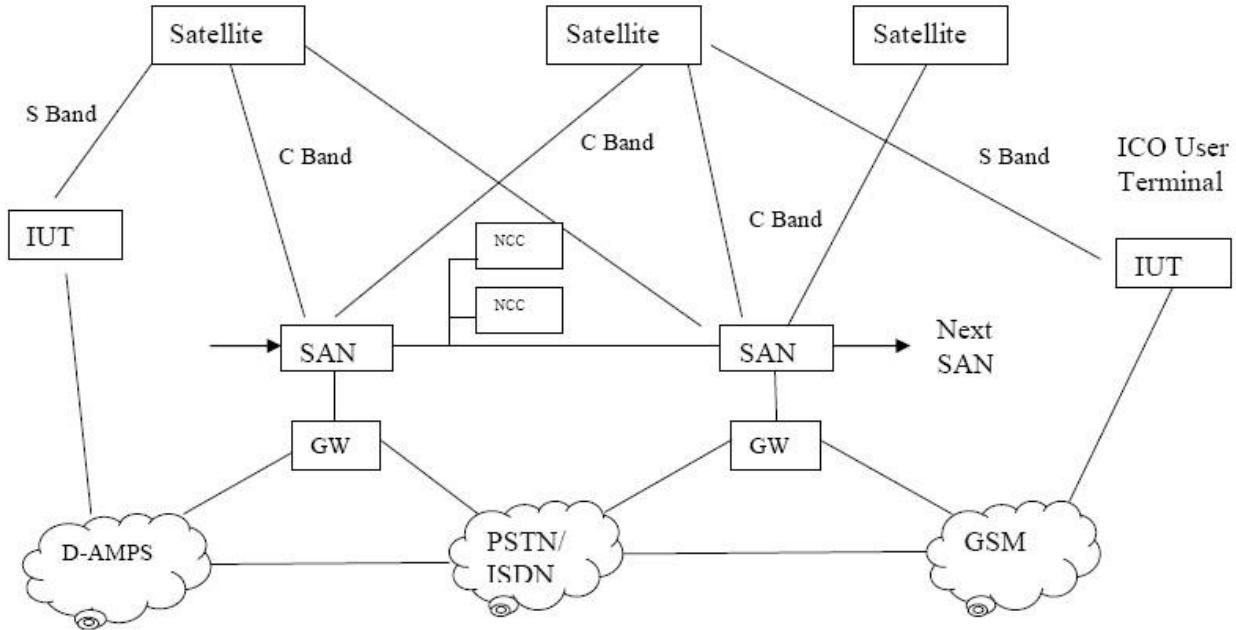


Fig 8.7 Network Architecture for ICO System

VPN (Virtual Private Network)

A VPN is a private data network that makes use of the public telecommunication infrastructure, maintaining privacy through the use of a tunneling protocol and security procedures. A communications network tunneled through another network, and dedicated for a specific network. A virtual private network can be contrasted with a system of owned or leased lines that can only be used by one company. The main purpose of a VPN is to give the company the same capabilities as private leased lines at much lower cost by using the shared public infrastructure. Phone companies have provided private shared resources for voice messages for over a decade. A virtual private network makes it possible to have the same protected sharing of public resources for data. Companies today are looking at using a private virtual network for both extranets and wide-area intranets.

A VPN works by using the shared public infrastructure while maintaining privacy through security procedures and tunneling protocols such as the Layer Two Tunneling Protocol (L2TP). In effect, the protocols, by encrypting data at the sending end and decrypting it

at the receiving end, send the data through a "tunnel" that cannot be "entered" by data that is not properly encrypted.

Working of VPN

Routers with VPN software works like normal router, filtering packets. To achieve the desired effect of secured private network on the public network VPN performs the following functions:

- A router implementing VPN software is configured such that
 - It rejects all incoming packets except those that arrive from a router which belongs to the authorized network.
 - It rejects all outgoing packets except those which belong to the authorized network.
- VPN software encrypts each outgoing packet before transmission. Received packets are decrypted

Goals of VPN

The design goals for a VPN are as follows:

Confidentiality

Confidentiality protects the privacy of information being exchanged between communicating parties. Towards this end, every VPN solution provides encryption of some sort.

Integrity

Integrity ensures that information being transmitted over the public Internet is not altered in any way during transit.

Authentication

Authentication ensures the **identity** of all communicating parties.

Types of VPN

Remote-Access VPN

Also called a **virtual private dial-up network (VPDN)** is a user-to-LAN connection used by a company that has employees who need to connect to the private network from various remote locations. Typically, a corporation that wishes to set up a large remote access VPN will outsource to an **enterprise service provider (ESP)**. The ESP sets up a **network access server (NAS)** and provides the remote users with desktop client software for their computers. The telecommuters can then dial a toll-free number to reach the NAS and use their VPN client software to access the corporate network.

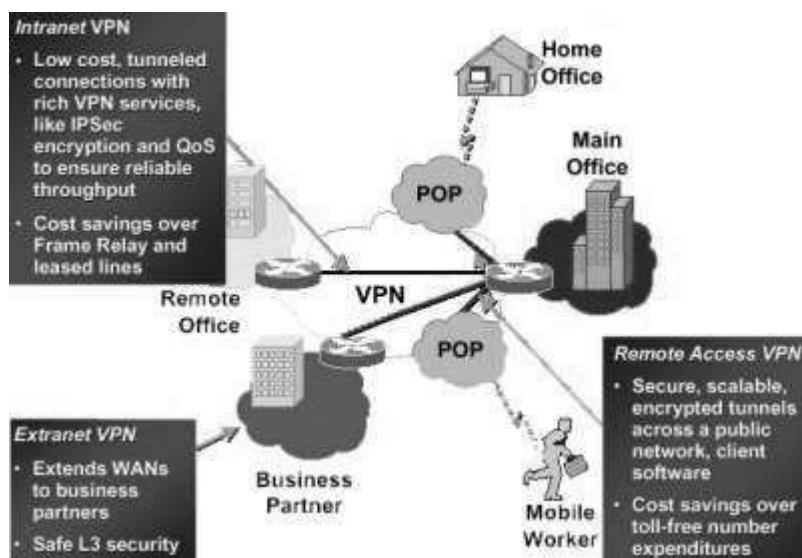


Fig 13.1 Remote Access VPN

Site-to-Site VPN

Through the use of dedicated equipment and large-scale encryption, a company can connect multiple fixed sites over a public network such as the Internet. Site-to-site VPNs can be one of two types:

- Intranet-based** - If a company has one or more remote locations that they wish to join in a single private network, they can create an intranet VPN to connect LAN to LAN.

- **Extranet-based** - When a company has a close relationship with another company (for example, a partner, supplier or customer), they can build an extranet VPN that connects LAN to LAN, and that allows all of the various companies to work in a shared environment.

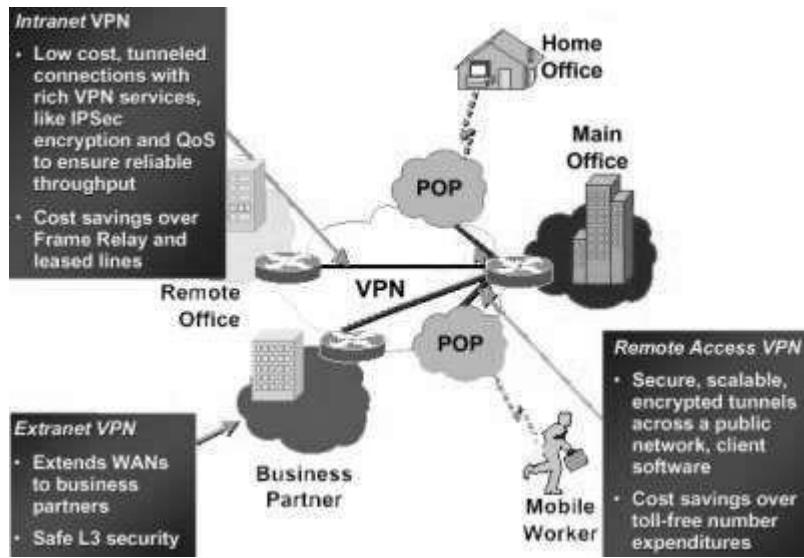


Fig 13.2 Site-to-Site VPN

VPN Protocols

Three protocols are widely used to provide authenticity, confidentiality and integrity.

These are as follows:

Internet Protocol Security (IPSec)

It is a set of authentication and encryption protocols, developed by the Internet Engineering Task Force (IETF) and designed to address the inherent lack of security for IP-based networks. It is designed to address data confidentiality, integrity, authentication and key management, in addition to tunneling. IPSec encapsulates a packet by wrapping another packet around it. It then encrypts the entire packet. This encrypted stream of traffic forms a secure tunnel across an otherwise unsecured network.

Point to Point Tunneling Protocol (PPTP) and Layer 2 Tunneling Protocol (L2TP)

PPTP is a tunneling protocol which provides remote users encrypted, multi-protocol access to a corporate network over the Internet. Network layer protocols are

encapsulated by the PPTP protocol for transport over the Internet. PPTP can support only one tunnel at a time for each user. L2TP (a hybrid of PPTP and another protocol, L2F) can support multiple, simultaneous tunnels for each user. L2TP will be incorporated in Windows 2000 and can support IPSec for data encryption and integrity

Socks5

SOCKS version 5 is a circuit-level proxy protocol that was originally designed to facilitate authenticated firewall traversal. It provides a secure, proxy architecture with extremely granular access control, making it an excellent choice for extranet configurations. SOCKS v5 supports a broad range of authentication, encryption, tunneling and key management schemes, as well as a number of features not possible with IPSec, PPTP or other VPN technologies. SOCKS v5 provides an extensible architecture that allows developers to build system plug-ins, such as content filtering and extensive logging and auditing of users. When SOCKS is used in conjunction with other VPN technologies, it's possible to have a more complete security solution than any individual technology could provide.

Bluetooth

Bluetooth is an open specification for a radio system that provides the network infrastructure to enable short range wireless communication of data and voice. It comprises of a hardware component and a software component. The specification also describes usage models and user profiles for these models. Bluetooth was the nickname of a Danish king Harald Blatand, who unified Denmark and Norway in the 10th century. The concept behind Bluetooth wireless technology was unifying the telecom and computing industries. Bluetooth technology allows users to make ad hoc wireless connections between devices like mobile phones, desktop or notebook computers without any cable. Devices carrying Bluetooth-enabled chips can easily transfer data at a speed of about 720 Kbps within 50 meters (150 feet) of range or beyond through walls, clothing and even luggage bags.

Bluetooth Protocol

Bluetooth protocol uses a combination of circuit and packet switching. The channel is slotted and slots can be reserved for synchronous packets. Bluetooth protocol stack can

support an asynchronous connection-less (ACL) link for data and up to three simultaneous synchronous connection-oriented (SCO) links for voice or a combination of asynchronous data and synchronous voice (DV packet type). Each voice channel supports a 64 Kb/s synchronous channel in each direction. The asynchronous channel can support maximum of 723.2 Kb/s uplink and 57.6 Kb/s downlink (or vice versa) or 433.9 Kb/s symmetric links. The stack primarily has a baseband for physical layer and link manager and controller for link layer. The upper layer interface depends on how these two layers are implemented and used with applications. The stack is shown below.

Piconet and Scatternet

Bluetooth supports both unicast (point-to-point) and multicast (point-to-multipoint) connections. Bluetooth protocol uses the concept of master and slave. In a master-slave protocol a device cannot talk as and when they desire. They need to wait till the time the master allows them to talk. The master and slaves together form a piconet. The master determines the hopping pattern in the piconet and the slaves have to synchronise to this pattern. Two additional types of devices are also present: Parked (P) and Standby (SB). Parked devices cannot actively participate in the piconet but are known and can be reactivated within some milliseconds. Devices in stand-by do not participate in piconet. Each piconet has exactly one master and up to seven simultaneous slaves. More than 200 devices can be parked. Several of these **piconets** can be linked together to form a larger network in an ad hoc manner. The topology can be thought as a flexible, multiple piconet structure. This network of piconets is called scatternet. A scatternet is formed when a device from one piconet also acts as a member of another piconet. In this scheme, a device being master in one piconet can simultaneously be a slave in the other one.

Bluetooth protocol is a combination of different protocols. The Bluetooth Core protocols plus the Bluetooth radio protocols are required by most of Bluetooth devices, while the rest of the protocols are used by different applications as needed. At the physical layer Bluetooth uses spread spectrum technologies. It uses both direct sequence and frequency hopping spread spectrum technologies. Bluetooth uses connectionless (ACL-Asynchronous Connectionless Link) and connection-oriented (SCO-Synchronous

Connection-oriented Link) links. Together, the Cable Replacement layer, the Telephony Control layer, and the Adopted protocol layer form application-oriented protocols enabling applications to run over the Bluetooth Core protocols.

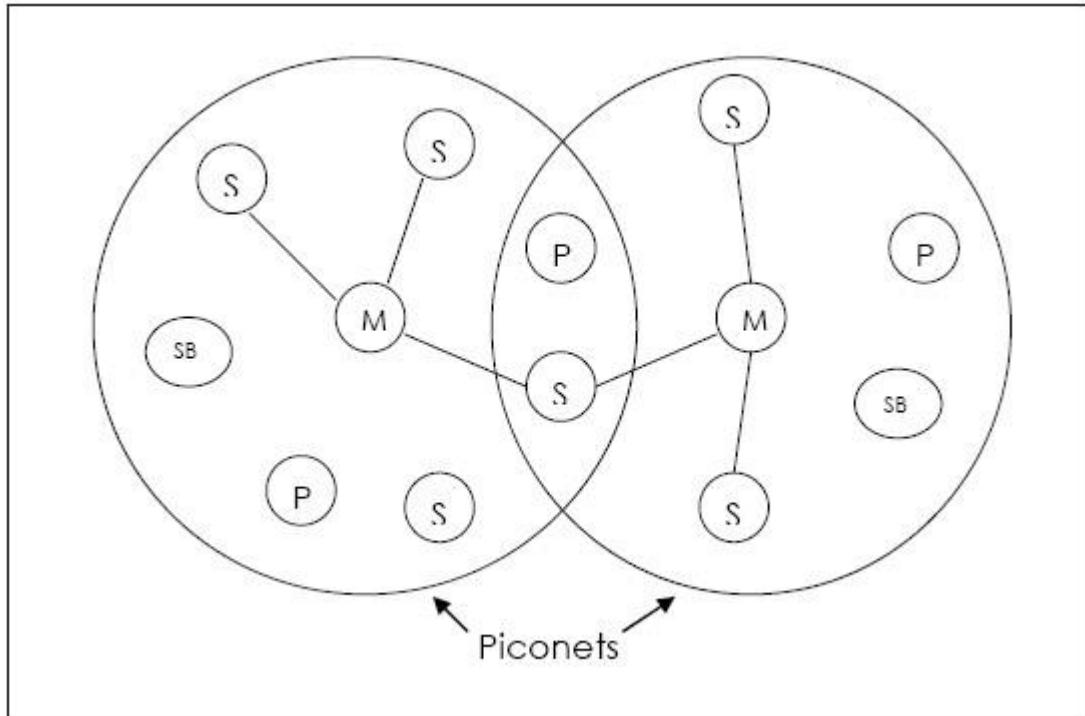


Figure 10.1 Bluetooth scatternet as a combination of Piconets

Bluetooth Protocol Stack

Bluetooth protocol stack can be thought of combination of multiple application specific stacks as depicted in Figure 4.2. Different applications run over one or more vertical slices from this protocol stack. These are RFCOMM (Radio Frequency COMMunication), TCS Binary (Telephony Control Specification), and SDP (Service Discovery Protocol). Each application environments use a common data link and physical layer. RFCOMM and the TCS binary (Telephony Control Specification) protocol are based on the ETSI TS 07.10 and the ITU-T Recommendation Q.931 respectively. Some applications have some relationship with other protocols, e.g., L2CAP (Logical Link Control and Adaptation Protocol) or TCS may use LMP (Link Manager Protocol) to control the link manager.

Bluetooth protocol stack can be divided into four basic layers according to their functions. These are: **Bluetooth Core Protocols** this comprises of Baseband, Link Manager Protocol (LMP), Logical Link Control and Adaptation Protocol (L2CAP), and Service Discovery Protocol (SDP).

Baseband

Baseband is the physical layer of the Bluetooth which manages physical channels and links apart from other services like error correction, data whitening, hop selection and Bluetooth security. Baseband lies on top of Bluetooth radio in Bluetooth stack and essentially acts as a link controller and works with link manager for carrying out link level routines like link connection and power control. Baseband also manages asynchronous and synchronous links, handles packets and does paging and inquiry to access and inquire the Bluetooth devices. Baseband transceiver applies a time-division duplex (TDD) scheme.

ACL and SCO links

Baseband handles two types of links: SCO (Synchronous Connection-Oriented) and ACL (Asynchronous Connection-Less) link. The SCO link is a symmetric point-to-point link between a master and a single slave in the piconet. The master maintains the SCO link by using reserved slots at regular intervals (circuit switched type). The SCO link mainly carries voice information. The master can support up to three simultaneous SCO links while slaves can support two or three SCO links. SCO packets are never retransmitted. SCO packets are used for 64 kB/s speech transmission.

The ACL link is a point-to-multipoint link between the master and all the slaves participating on the piconet. In the slots not reserved for the SCO links, the master can establish an ACL link on a per-slot basis to any slave, including the slave already engaged in an SCO link (packet switched type). Only a single ACL link can exist. For most ACL packets, packet retransmission is applied.

Logical Channels

Bluetooth has five logical channels which can be used to transfer different types of information. LC (Control Channel) and LM (Link Manager) channels are used in the link

level while UA, UI and US channels are used to carry asynchronous, isosynchronous and synchronous user information.

Bluetooth Addressing

There are basically four types of device addresses in Bluetooth:

- BD_ADDR: 48 bit Bluetooth device address. It is divided into LAP (Lower Address Part of 24 bits), UAP (Upper Address Part of 8 bits) and NAP (Nonsignificant Address Part of 16 bits).
- AM-ADDR: 3 bit active member address. The all zero AM_ADDR is for broadcast messages.
- PM_ADDR: 8-bit member address that is assigned to parked slaves.
- AR_ADDR: The access request address is used by the parked slave to determine the slave-to-master half slot in the access window it is allowed to send access messages.

Bluetooth packets

The data on the piconet channel is conveyed in packets. The general packet is shown below:

ACCESS CODE (72)	HEADER (54)	PAYLOAD (0-2754)
------------------	-------------	------------------

Fig 10.2 Bluetooth Packet

Access code is used for timing synchronization, offset compensation, paging and inquiry. There are three different types of Access code: Channel Access Code (CAC), Device Access Code (DAC) and Inquiry Access Code (IAC). The channel access code identifies a piconet (unique for a piconet) while DAC is used for paging and its responses. IAC is used for inquiry purposes. The header contains information for packet acknowledgement, packet numbering for out-of-order packet reordering, flow control,

slave address and error check for header. The packet payload can contain either voice field, data field or both. The packet can occupy more than one slot (multislot packets) and can continue transmission in the next slot. The payload also carries a 16-bit CRC for error detection and correction in the payload. SCO packets do not include CRC.

Flow control and synchronization

Bluetooth recommends using FIFO queues in ACL and SCO links for transmission and receive. Link Manager fills these queues and link controller empties the queues automatically.

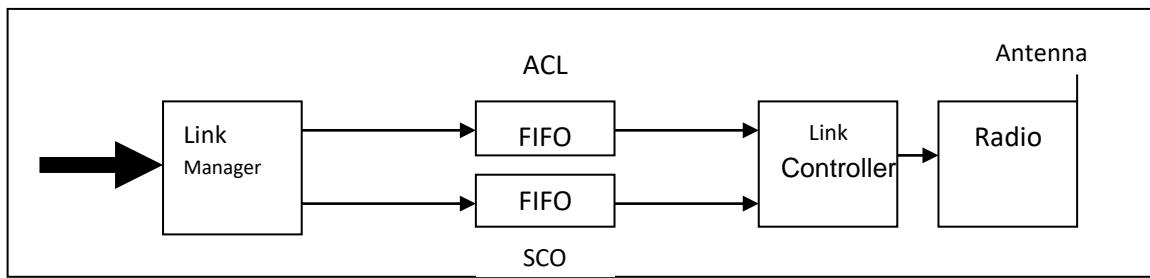


Fig 10.3 Flow Control and Synchronization

If these FIFO queues are full, flow control is used to avoid dropped packets and congestion. If data cannot be received, a STOP indication is transmitted inserted by the Link Controller of the receiver into the header of the return packet. When the transmitter receives the STOP indication, it freezes its FIFO queues. If receiver is ready it sends a GO packet which resumes the flow again.

Controller States

Bluetooth controller operates in two major states: **Standby** and **Connection**. There are seven sub states which are used to add slaves or make connections in the piconet. These are **page**, **page scan**, **inquiry**, **inquiry scan**, **master response**, **slave response** and **inquiry response**.

The **Standby** state is the default low power state in the Bluetooth unit. Only the native clock is running and there is no interaction with any device whatsoever. In the **Connection** state, the master and slave can exchange packets using the channel

(master) access code and the master Bluetooth clock. The hopping scheme used is the channel hopping scheme.

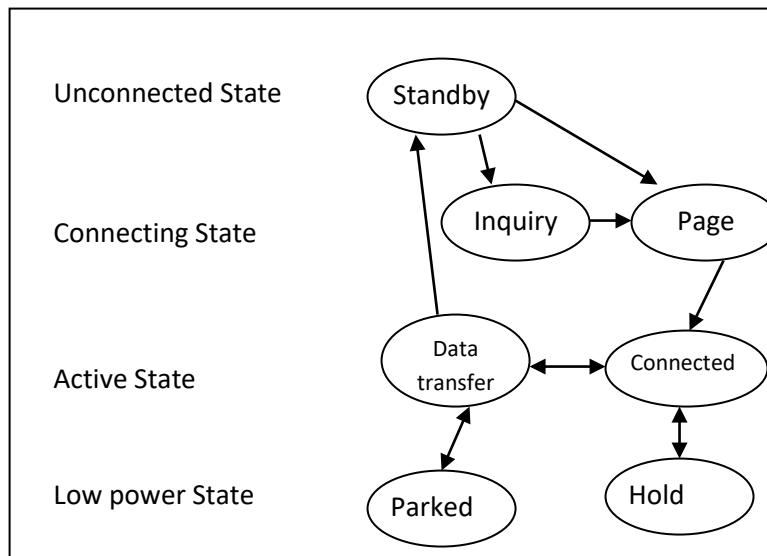


Fig 10.4 Bluet0oth Controller States

A connection between two devices occurs in the following fashion. First master uses the GIAC and DIAC to inquire about the Bluetooth devices in the range. If any nearby Bluetooth device is listening for these inquiries, it responds to the master by sending its address and clock information to the master. After sending the information, the slave may start listening for page messages from the master. The master after discovering the in range Bluetooth devices may page these devices for connection setup. The slave in page scan mode if paged by this master will respond with its device access code (DAC). The master after receiving the response from the slave may respond by transmitting the master's real time clock, master's BD_ADDR, the BCH parity bits and the class of the device (FHS packet). After slave has received this FHS packet, both enter into **Connection** state.

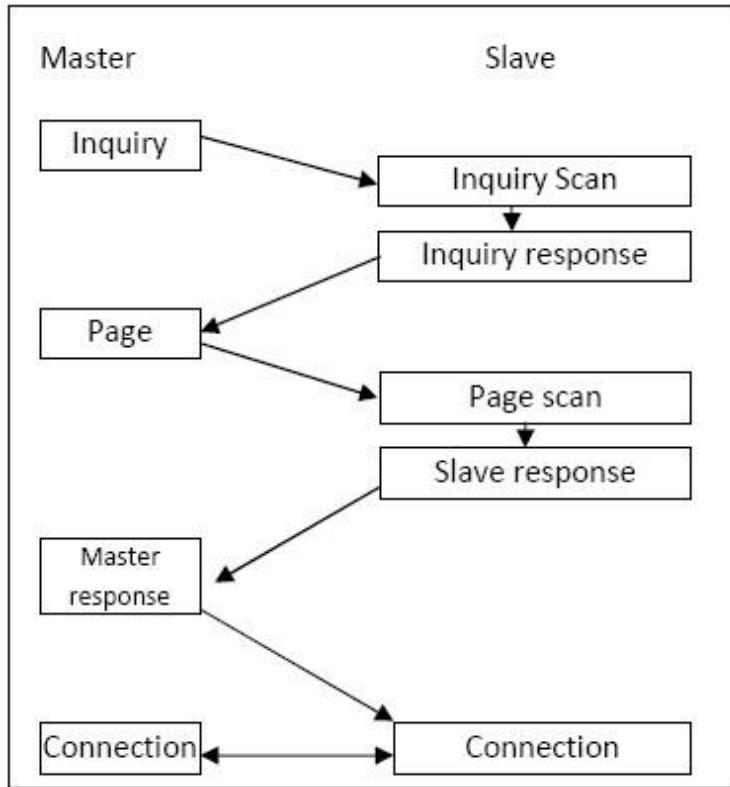


Fig 10.5 Connection setup between master and slave

Link Manager Protocol (LMP): Link Manager is used for managing the security, link set-up and control. It talks to the other link manager to exchange information and control messages through the link controller using some predefined link-level commands. When two Bluetooth devices come within each other's radio range, link managers of either device discover each other. LMP then engages itself in peer-to-peer message exchange. These messages perform various security functions starting from authentication to encryption. LMP layer performs generation and exchange of encryption keys as well. This layer performs the link setup and negotiation of baseband packet size. LMP also controls the power modes, connection state, and duty cycles of Bluetooth devices in a piconet.

Functions of Link Manager

- Administration and control of all link actions.
- Monitoring state changes
- Monitoring transmission modes

- Monitoring Synchronization
- Device Pairing
- Handling data encryption
- Handling device authentication

The Host Controller Interface (HCI)

The HCI provides a command interface to the base-band controller, link manager and access to the hardware status and control registers. The interface provides a uniform method of accessing the Bluetooth baseband capabilities. The Host control transport layer abstracts away transport dependencies and provides a common device driver interface to various interfaces.

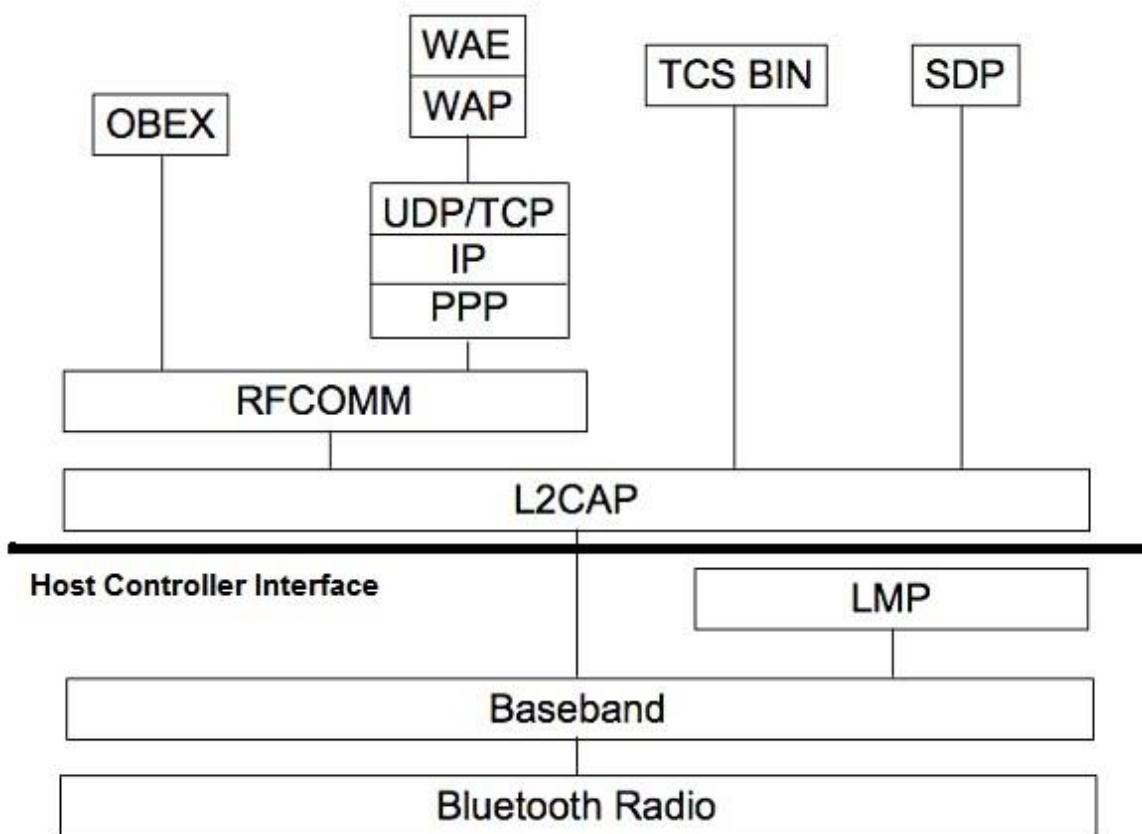


Figure 10.6 Bluetooth Protocol stack

Information Exchange and Request

A Bluetooth link manager can request from other link manager the clock offset (master requesting the slave to tell it the current clock offset stored by it which slave itself got from master during some packet exchange), slot offset (slot offset is the time in microseconds between the start of the master's transmission slot in the piconet where the PDU is transmitted and the start of the master's transmission slot where the BD_ADDR device in the PDU is master. It is useful in master-slave switch and interpiconet communications), timing accuracy (clock drift and jitter), link manager version and information about supported features like support for authentication, SCO packets etc.

Logical Link Control and Adaptation Protocol (L2CAP)

L2CAP provides connection-oriented and connectionless data services to upper layer protocols with protocol multiplexing capability, segmentation and reassembly operation, and group abstractions. L2CAP permits higher level protocols and applications to transmit and receive L2CAP packets up to 64 Kilobytes in length. L2CAP only supports ACL links. L2CAP uses the concept of channels to establish different pathways between different applications on Bluetooth devices. These channels are identified by Channel Identifiers (CIDs) which represent a logical end point of a connection for each application on a device.

Segmentation and Reassembly

Segmentation and Reassembly (SAR) operations are used to improve efficiency by supporting a maximum transmission unit (MTU) size larger than the largest Baseband packet. This reduces the overhead by spreading the network and transport packets used by higher layer protocols over several baseband packets. L2CAP segments higher layer packets into `chunks' that can be passed to the Link Manager for transmission and reassembles those chunks into L2CAP packets using information provided through HCI and from packet header

Service Discovery Protocol (SDP)

The Service Discovery Protocol (SDP) enables a Bluetooth device to join a piconet. Using SDP a device inquires what services are available in a piconet and how to access them.

SDP uses a client-server model where the server has a list of services defined through service records. One service record in a server describes the characteristics of one service. In a Bluetooth device there can be only one SDP server. If a device provides multiple services, one SDP server acts on behalf of all of them. Similarly multiple applications in a device may use a single SDP client to query servers for service records.

Cable Replacement Protocol

This protocol stack has only one member viz., Radio Frequency Communication (RFCOMM).

RFCOMM: This is a serial line communication protocol and is based on ETSI 07.10 specification. The "cable replacement" protocol emulates RS-232 control and data signals over Bluetooth baseband protocol. It provides a reliable data stream, multiple concurrent connections, flow control and serial cable line settings.

Telephony Control Protocol

This comprises of two protocol stacks viz., Telephony Control Specification Binary (TCS BIN), and the AT-Commands.

Telephony Control protocol Binary: TCS Binary or TCS BIN is a bit-oriented protocol. TCS BIN defines the call control signaling protocol for set up of speech and data calls between Bluetooth devices. It also defines mobility management procedures for handling groups of Bluetooth TCS devices.

AT-Commands: This protocol defines a set of AT-commands by which a mobile phone can be used and controlled as a modem for fax and data transfers. AT (short form of attention) commands are used from a computer or DTE (Data Terminal Equipment) to "control a modem or DCE (Data Circuit terminating Equipment). AT-commands in Bluetooth are based on ITU-T Recommendation.

Adopted Protocols

PPP, TCP/IP: These are standard Internet protocols defined by IETF. These are used as the lower layer protocols of the WAP stack.

OBEX: It is a session protocol defined by IrDA. This protocol is also utilized by bluetooth thus enabling the possibility for application to use either the Bluetooth radio or IrDA technologies.

WAP/WAE: Bluetooth may be used as a bearer technology for transporting between a WAP client and a nearby WAP server. WAP operates on top of the bluetooth stack using PPP and the TCP/IP protocol suite.

REFERENCES

1. Mobile Cellular Telecommunications – William C.Y.Lee (Mc Graw Hill)
2. Mobile and Personal Communication System – Raj Pandya (PHI)
3. Mobile Computing—Raj Kamal(Oxford)
4. Mobile Computing—Asoke Talukdar(TMH)
5. Mobile Communication – J.Schiller(Pearson)
6. Wireless Communication & Network—William Stallings (PEARSON)
7. Wireless Communication—Rappaport(PHI)
8. The Wireless Application Protocol – Sandeep Singhal(PHI)
9. Data Communication and networking—Forouzan(TMH)
10. Wireless and Mobile Systems—Agrawal(Thomson)
11. Wireless and Mobile Network Architecture—Lin (WILEY)
12. High speed Networks and Internets – Stallings (Pearson)
13. Wireless Communication and Networking—Garg (Elsevier)
14. Wireless Networking – Kumar (Elsevier)
15. IS-95, CDMA and CDMA 2000 – Garg (Pearson)
16. Computer Networks – Tanenbaum (PHI)
17. Professional WAP – Charles Arehart (Wrox Press)
18. Data and computer communications-Stalling (Pearson)