X.500 Directory Service

Objectives:

- Explain case study “the X.500 Directory Service”

The X.500 is a directory service that is similar to a conventional name service which is mainly used for satisfying descriptive queries. It is designed to determine the names and attributes of system resources. The individuals and organizations can use a directory service to get detailed information about themselves and the resources that they want to use in the network. The individuals are allowed to search the directory for specific information with partial knowledge of its name or structure, for example, a telephone directory has White pages for specific information about users and Yellow pages have detailed information. The X.500 Directory Service is defined by ITU and

ISO standards organizations to meet users' and organizations requirements for accessing information about hardware and software services and devices. It is designed like a general-purpose directory service that is used in LDAP and DCE directory services.

In X.500, the data is stored in a tree structure with named node like other naming services where a wide range of attributes are stored at each node in the tree. These attributes can be accessed by searching for entries with any combination of attributes not just by name. It is also called Directory Information Tree (DIT) and the directory structure with data and nodes is called Directory Information Base (DIB). Each tuple in a DIB consists of a pair of names and a set of attributes. Generally, organizations have one or more directory servers to which clients can establish a connection for accessing directory information by issuing access requests. If the data that is required is not available in the segment of the DIB of a requested server, then the query is redirected to another server. The servers in X.500 are also called Directory Service Agents (DSAs), and their clients are termed as Directory User Agents (DUAs). The typical architecture of X.500 is shown in Figure 1, which represents DUAs interacting with DSAs to fulfill the request.

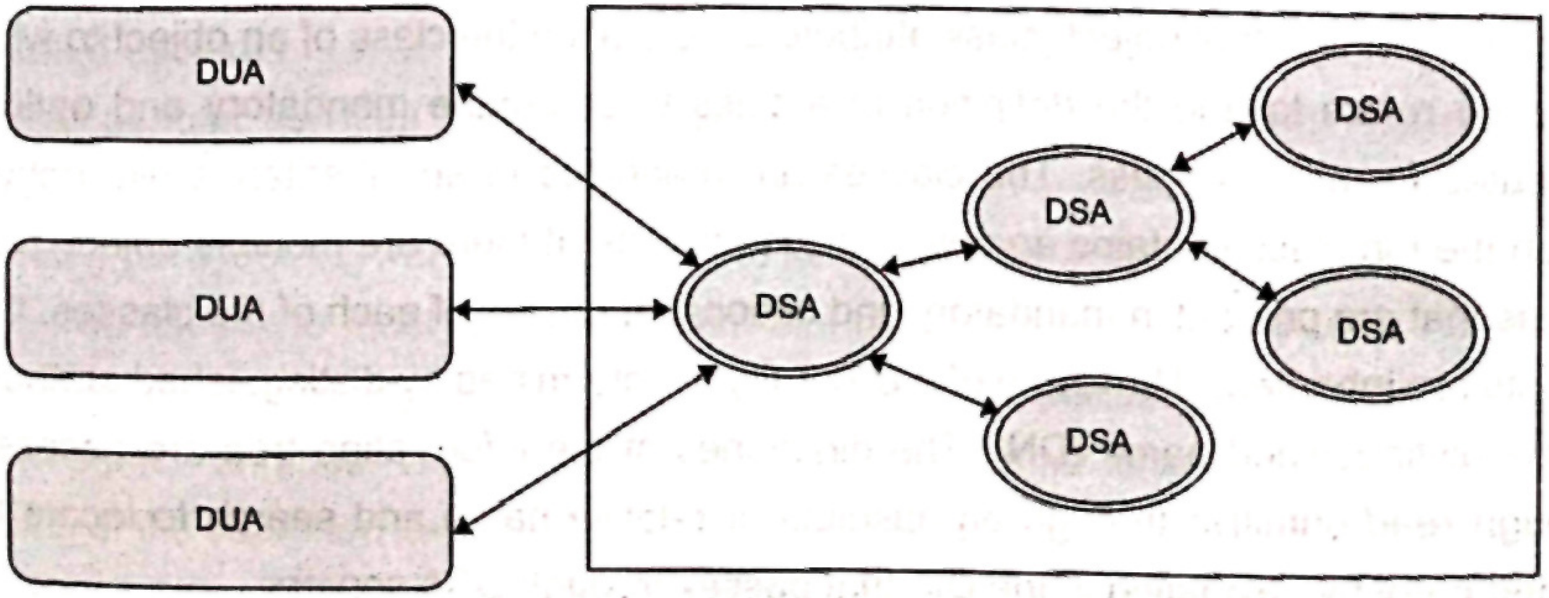

Figure 1. Architecture of X.500

The name servers maintain the entry path to DIT from the root of the tree with full or absolute names. DUA can establish a path from the base node to the named entry using a context that includes a base node. Figure 2 shows Directory Information Tree for Mumbai University in India with associated DIB entries.

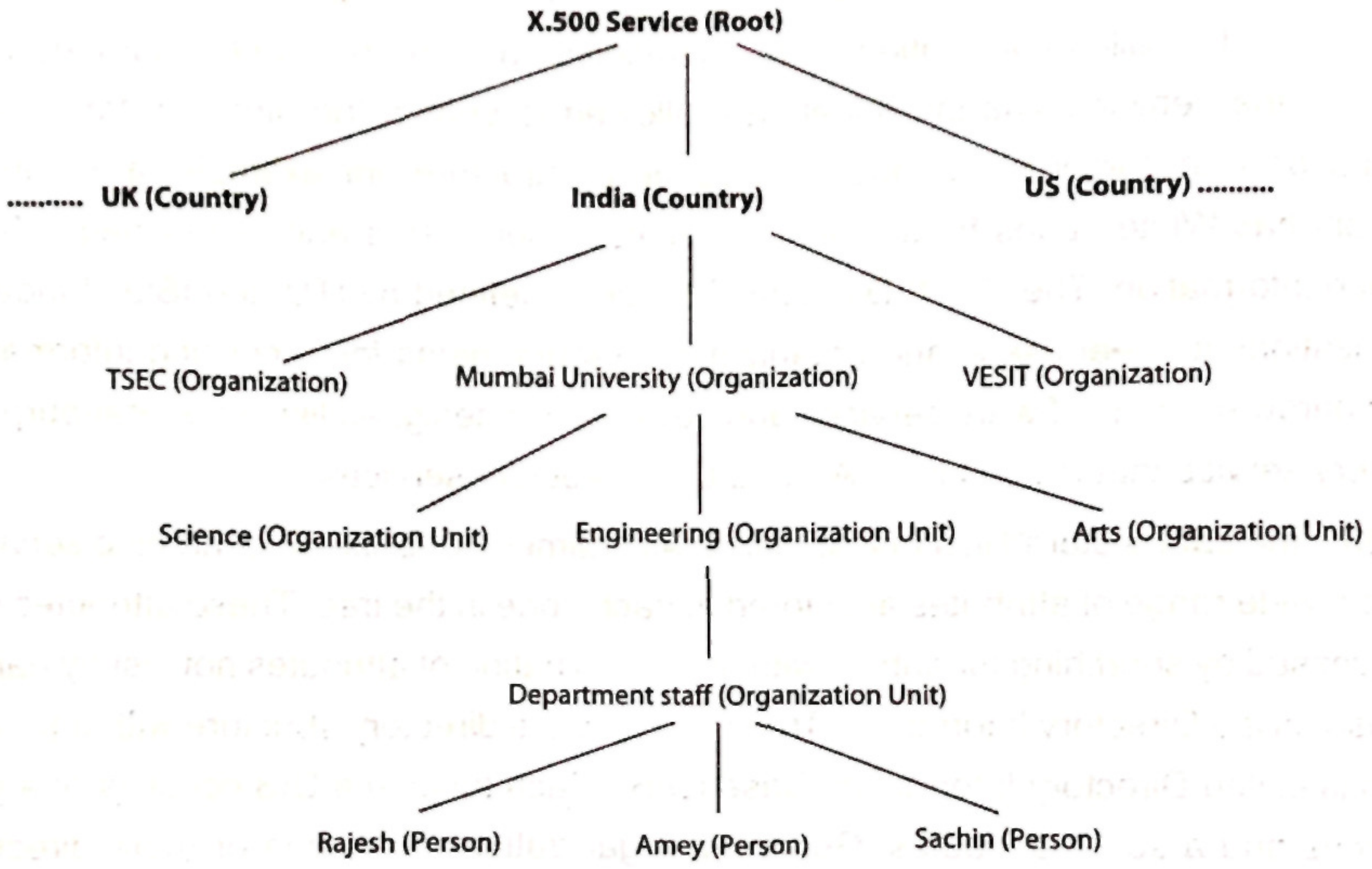

Figure 2. Directory Information Tree for Mumbai University

The DIB and the DIT have flexible data structures for the entries. A set of attributes in each DIB entry has a type and one or more values. The type is defined by a type name (for example, _name, country's name, telephone number, etc.). The attribute has a type definition for each distinct type name that includes syntax definition in the ASN.1 notation with type description.

Each entry in DIB has object_class attribute to determine the class of an object to which an entry refers to and the definition of a class to determine mandatory and optional attributes for a given class. The classes are organized in an inheritance hierarchy in which the top class contains an object_class attribute. If there are multiple object_class values that are present in mandatory and optional attributes of each of the classes, then objects are inherited. The name of a DIB entry is determined by distinguished attributes with a distinguished name (DN). The directories of the information tree are accessed through read primitive through an absolute or relative name and search to locate the named entry by navigating in the DIT that passes through DSA servers.

To retrieve the required attributes, it also uses a filter expression to evaluate for every node that is present below the base node which specifies a search criterion. The filter expression has only two values, namely TRUE and FALSE, where the search command is used for returning a list of names (domain names) for all of the entries that are present below the base node with the TRUE value. The DSA interface supports different operations for adding, deleting and modifying entries along with access control to decide that access to which parts of the DIT may be restricted to certain users or classes of the user. DIT supports both caching and replication at the level of individual DIB along with the consistency of redundant operations.

The Lightweight Directory Access Protocol (LDAP) is the implementation of X.500 in UNIX as directory services that communicate directly over TCP/IP instead of higher layers of the ISO protocol stack which is described in RFC 2251. LDAP uses simplified interfaces of X.500 which provides relatively simple APIs for textual encoding. Although the LDAP specification is based on X.500, it is widely adopted over X.500 for intranet directory services. It also provides secure access to directories and data through an authentication mechanism. For example, Microsoft's Active Directory Services (ADS) provides the LDAP interface for connecting intranet nodes through a directory service that provides additional features such as group management, access control, authentication and authorization mechanisms for workstations.

References:

1] George Coulouris, Jean Dollimore, Tim Kindberg, “Distributed Systems: Concepts and Design”, 4th Edition, Pearson Education, 2005.
2] S. Tanenbaum and M. V. Steen, “Distributed Systems: Principles and Paradigms”, Second Edition, Prentice Hall, 2006.