Prepared By Dr. N.THENMOZHI M.C.A., M.S., M.Phil., Ph.D.,

Govt, Arts College (Autonomous), Coimbatore-18

Department of Information Technology

OPEN SOURCE TOOLS (18MIT33C) -----II MSc

UNIT-I: Introduction to OSS :Introduction - Need for Open Source Applications - Advantages of Free Open Source Software - Disadvantages of Free Open Source Software - History - Meaning and Extraction of the Terms Free Software and Open Source Software - Free Software Foundation and Open Source Initiative Presentation- Free Software and Open Source Software Licenses Comparison - Licensing - Familiar Licenses - Free and Open Source Software (Game Theory) - Security and Reliability - Economical Aspects and Adoption - Applications of Open Source Software – Open source grid computing : Open Grid Service Architecture – OGSI – Security Issues – Globus Toolkit – Open source cloud: Introduction-FOSS cloud software environment.

UNIT-II: Open Source OS Linux: Linux Basics: Introduction - Kernel/User Mode – Process – Advanced Concept-Scheduling – Personalities- Cloning - Signals - Development with Linux - OSS Installation. Linux shell Commands – Vi Editor - Shell programming: Shell Syntax - Variables – conditions – control structures – functions – commands – command execution.

UNIT-III: PHP: Introduction – Identifier -Variables - Constants – Data types – Operators - Statements – loops. Advanced PHP –Arrays – Get and Post – Object oriented concepts – Strings –File handling and data storage. MySQL Databases – Setting –Starting, terminating and writing own SQL programs – record selection technology, strings functions, date and time – starting query – generating summary – working with metadata – using sequences – PHP and MySQL databases.

UNIT-IV: PERL: Introduction – advantages –working environment of Perl – variables – strings –statements –subroutines – files –packages and modules – Object-Oriented PERL.

UNIT-V: Ruby on Rails: Welcome to Ruby –Conditions, methods, loops and blocks - classes and objects. Welcome to rails: Connecting to databases – working with databases.

TEXT BOOKS

1. M.N. Rao, “Fundamentals of open source software”, PHI Learning Private Limited, 2015.
2. Neil Matthew and Richard Stones, “Beginning Linux Programming”, 4th Edition, WROX, 2011.
3. Steven Holzner, “Beginning ruby on rails”, Wiley publishing, Inc, 2007.

REFERENCE BOOKS

1. Chris DiBona, Danese Cooper and Mark stone O Reilly,”Open Sources 2.0 – The Continuing Evolution”, First Edition, 2005.
2. Elliot White III, Jonathan.D.Eisenhamer, “PHP 5 in practice” pearson Education,2007.
3. Paul Du Bois,O Reilly Publishers,”My SQL- Cookbook”,Second Edition,2010.
4. Larry Wall, Tom Christiansen, Jon Orwart- O Reilly, “Programming PERL”,3rd Edition, 2010.
5.Yashavant P. Kanetkar,” Unix Shell Programming”, BPB publications, 2003. *2018-2019*

UNIT-I: Introduction to OSS :Introduction - Need for Open Source Applications - Advantages of Free Open Source Software - Disadvantages of Free Open Source Software - History - Meaning and Extraction of the Terms Free Software and Open Source Software - Free Software Foundation and Open Source Initiative Presentation- Free Software and Open Source Software Licenses Comparison - Licensing - Familiar Licenses - Free and Open Source Software (Game Theory) - Security and Reliability - Economical Aspects and Adoption - Applications of Open Source Software – Open source grid computing : Open Grid Service Architecture – OGSI – Security Issues – Globus Toolkit – Open source cloud: Introduction-FOSS cloud software environment.

1. AN INTRODUCTION TO OSS

1.1 INTRODUCTION

Open Source Software's (OSS) are user-friendly and are available as commercial and free licensed software. Therefore, these are easy to use by anyone, anywhere. Open source is easily modifiable as its core code is publicly accessible. As OSS is originated from the framework of software devel Mathematics opment, it designates a set of values and initiatives, those that embrace, celebrate open exchange, collaborative participation, rapid prototyping, clearness, and community development. It is widely accepted by a number of people and organizations. The software allows to outlook the code, but it does not allow the conversion or modification of the code. In fact, this software is known as source-available or shared sources.

As mentioned earlier, OSS is software whose source code is accessible for alteration or enrichment by other programmers. Source code is the part of software which is mostly used in the development of the software package, and is accessed by the programmers. Programmers can enrich the performance of that program by including features, or by modifying, or by fixing add-on's to make the system function more effectively than older versions.

Generally, OSS implements different levels of development which make them suitable for building minor and comprehensive real-world systems. Simultaneously, the field of knowledge-based systems has refined a huge frame of dynamic learning algorithms for distinct purposes. Indeed, the possibility of these methods is not thoroughly utilized, and as a result, current implementations are not freely pooled which results in the software with low usability and feeble interoperability. This condition can be effectively enhanced by the rising motivation for the investigators to broadcast their program covered by an open source model. In addition to that, they outline the problems that the authors face when trying to publish algorithmic implementations of learning methods, i.e., system, machine, etc. In this approach, to believe that an analysis of software service is accompanied by some journals, can be extremely profitable to both the system knowledge and the general scientific community.

1.2 NEED FOR OPEN SOURCE APPLICATIONS

- The manipulators are well-maintained to have entry to the software source code. Additionally, users are encouraged to submit add-ons, improvements, bug reports, documentation, etc. to the software. When a number of programmers view the code, they identify all the errors and recommend how to debug them. Generally, some programmers have advanced programming skills and adding to this, the individual end-user's machine

contributes an additional testing situation, which offers the ability to formulate new applications and to fix some existing bugs.

- The earliest version of the software should be discharged as soon as possible, in order to be followed in an iterative manner with version control. This eventually leads to a much effective software package. Source code changes should be surely unified continuously, to avoid the overhead of fixing a huge number of errors at the termination of the project life cycle which leads to the software failure.
- Two versions of the software are present–one is the buggier version with more features and the second is an additional constant version with lesser characteristics. The buggier version is developed for end-users, who need the instant use of the current functionalities and are amenable to obtain the hazard of using the code which is not tested intensely. The end-users are also called co-developers, because they report errors and also suggest how to fix those errors.
- End-user also takes part in developing the stable version. The regular software structure should be standard. This standard software allows both, parallel and distributed development of independent components of the software.
- On the basis of user needs, an essential decision-making structure may be either official or relaxed.

1.2.1 Advantages of Free Open Source Software

- No royalty for licensing Free Open Source Software.
- Allow greater opportunities for software application architectures.
- Incorporate tools for any system in a real world environment.
- Easy to manage.
- Easy to understand for the developers.

1.2.2 Disadvantages of Free Open Source Software

- Free Open Source Software supports only single or multiple sources, but not parallelized environments.
- The business logic should be known to modify the configuration or to generate code modifications to compensate exclusive workflows.
- End-user considers ownership than proprietary licenses, which includes rising of upgrades, enrichments, configuration and software service.
- Hard to employ vendor service with real time practice in guiding the application.
- Hard for smaller companies to maintain the services individually.
- Require developed professional support on licensing problems and in making of agreements.
- Regular preparation and development of training information related to specific applications.
- There are no measures at the time when the information is overloaded.

1.3 HISTORY

During its nascent stages, software was considered as a type of mathematics. After some time, it became clear to improve mechanized pre-processing of data for governments and many

organizations. Firms started improving personal and proprietary systems. Opening the lock-in cycle and frequent fall of technology asset continued for a number of decades.

The thought of Free Open Software started with the improvement in the beginning of the 1980's, with the idea of 'free software' by Richard Stallman. Stallman wanted to build the GNU project and also Free Software Foundation to advance his vision which took a number of decades to achieve.

In the early 2000's, many people realized that the free software was centralized by the sharing of data all around the world. In the late 1990's, Eric S. Raymond along with others developed the term open source as a business-friendly word compared to free software. Open source has a complete meaning if the licenses were not strict and allows modifications. The Open Source Initiative (OSI) was launched to present a central certification organization for the several types of licenses that satisfy the definition of open source. The commercial open source software had preferred this term effectively, by leading the community to unite around the term Free Open Source Software which will remind the primary visions of Stallman and Raymond together. The term, Free Open Source Software was mentioned for the first time in the newsgroup.

1.4 MEANING AND EXTRACTION OF THE TERMS FREE SOFTWARE AND OPEN SOURCE SOFTWARE

From the GNU project, the definition of the term free software has been evolved. Free Software Foundation (FSF) maintained a competitive definition which determines the appropriate means of understanding the sense of using the term free . Free software is an element of freedom, but not the cost. According to FSF definition, if any program has four freedoms, then that program is said to be a free software . The four freedoms are as follows:

Freedom 0: There should be freedom to execute the program for any idea.

Freedom 1: Entry to the source code is a prerequisite. There should be the freedom to review the working procedure of the program.

Freedom 2: There should be liberty to redistribute models. Therefore, you can assist others.

Freedom 3: Entry to the source code is a prerequisite. The liberty to enhance the program and send your enhancements to the community. Therefore, the community gets benefitted.

An additional frequently used term is Open Source Software (OSS), which is managed by the Open Source Initiative (OSI). OSI declares that open source not only signifies the accessing to source code, but also sharing words of Open Source Software must allow rights to redistribute the software as a module. OSI states that modified source code must be an essential with the primary source code and the licence cannot be particular to a software. In 1998, the term Open Source Software was used by the free software community for the first time.

1.5 FREE SOFTWARE FOUNDATION AND OPEN SOURCE INITIATIVE PRESENTATION

On September 27, 1983, Richard Stallman sent a primary declaration of the GNU project to the newsgroups. The declaration consisted about clarification of writing GNU. GNU is expressed as a UNIX-friendly method containing kernel, which needs additional services to write and execute C programs, which will be prearranged at no cost that can utilize it.

Richard Stallman's intention was to share any program with neighbors who are fond of it, too. The term GNU was selected, since it met some of the needs. Before FSF was established by

Stallman, he prepared required steps to defend his upcoming work from his existing employer demands. Then, he left his job in January, 1984 at MIT lab. Writing of non-proprietary software was started by Richard Stallman. Later, the GNU manifesto was published by Stallman. The GNU manifesto is a manuscript that strengthens the primary announcement. Stallman supplemented the data on GNU, ways of granting the project and advantages from the project.

Free Software Foundation (FSF) was established in 1985 to support thoughts specified in the Software (OSS). On non-proprietary software market, the OSI becomes an option to FSF even though FSF challenger allows developers to utilize non-proprietary software.

1.5.1 Free Software and Open Source Software Licenses Comparison

The two slogans with different activities and philosophies are– free software and open source. The aim is to distribute freely and cooperatively in case of free software movement. Non-free software is unfriendly, because it does not allow the user to access the software at his/her liberty. The open source movement encourages a technically superior improvement form that regularly obtains technically superior results.

1.5.2 Licensing

The open source license allows the software to be freely usable, editable and sharable. Freedom from the privileges is permitted to all the users by all the open software licenses. If the software license is not consistent, then the association of coding, i.e., combining the source code or straight connection of binaries is a risky task. This problem can be avoided by connecting programs indirectly. Most of the software comes under the licenses which are discussed in this chapter.

1.5.3 Familiar Licenses

- GNU General Public License
- GNU Lesser General Public License
- BSD 2-Clause Simplified, or FreeBSD licence
- BSD 3-Clause New, or Revised license
- Mozilla Public License
- MIT License
- Apache License
- Eclipse Public License

The FSF, i.e., Free Software Foundation and the OSI, i.e., Open Source Initiative both, report set of licenses that they satisfy with their own definitions. The OSI list includes the licenses that are submitted and authorized. All open source licenses that satisfy the Open Source Definition are called Open Source Software. Licensed software that does not satisfy the free software definition cannot be considered as free software.

1.5.4 Free and Open Source Software (Game Theory)

A software is a collection of ideas with architecture that designs the entire process. It is possible to differentiate source code files and compiled files (binary). Any source code that satisfies definite qualifications is copyrighted in most of the states. Copyright comprises the

privilege to utilize the software and to build this software accessible to others. It also promotes the privileges to make imitative mechanisms and reallocate. In 90% of the states, the copyright of a work is manageable. Therefore, it is achievable to differentiate between the author and the copyright holder. The privileges are limited by default.

The authors can decide to maintain their work for themselves and not to share it with others. The authors correspondingly the copyright holders can select to share their work. If a software author developer desires to make his software accessible, then the author must particularly allow him by giving permissions to do that task in the form of license. The authors are differentiated by applying and obtaining copyright as copyright law. It is also achievable to identify the number of individual users. The count of use cases by individual user is also being calculated. Free and open source licenses are related to sharing. Therefore, we can make the software available to specific people. The copyright grants two differentiable use cases:

- The author or the copyright holder, can trade licenses to utilize a software, i.e., allows the privileges to utilize the software by the law of a state over a contract known as End-User License Agreement (EULA). The EULA is also known as Software License Agreement (SLA).
- The author or the copyright holder can distribute the software under a free software license and/or open-source software license. The basic similarity between them is to allow the irreversible right to utilize, reallocate, change and reallocate the change of software in a definite version. To alter software, a duplicate source code is required. The software is generally shared without any responsibilities. In fact, several recommend extra support for fee. Free and open source licenses include the information about software sharing.

Naming

Using of the term Open Source Software than using free software is prescribed by Free Source Foundation (FSF). The FSF records that Open Source has definitely a particular definition, i.e., you can view the source code. But, Stallman declares that the term free software points towards two distinct meanings, in which the former meaning is persistent with the Free Source Foundation definition of free software.

1.6 SECURITY AND RELIABILITY

Security is the capability of a structure to supervise, defend and share information. It is a complex task. FOSS systems are frequently higher than proprietary software. As long as possible, the general method used by developers is to hide the bugs. The bugs will not be determined even by testers, if they hide the bugs. Therefore, all the systems are attacked. There is no difficulty, if the software developers determine the attacks before the public who are fond of hacking. In most of the cases, such attacks are detected after hacking. The tests are partial because the developer and the tester are same. But, in case of FOSS, we cannot find such type of disadvantages because the entire source code is accessible to all the users. Several people can inspect the code at a time. Therefore, the bugs are identified more rapidly. Regularly bugs are reconstructed, even without the assistance of developers' team.

The level of excellence or quality of the developed software directly depends on the experience, capability and skilled methodologies of the programmer. By performing testing, peer reviews beta and alpha version; we can improve the security and reliability of the code. No software is 100% resistant from security vulnerabilities, but free open source software delivers

better security performances. There are three reasons why FOSS offers better security. They are as follows:

1. The source code is accessible to the entire community. This accessibility helps them to discover and fix the security vulnerabilities.
2. The FOSS applications mainly focus on the functionality and the robustness rather than ease of use. Before adding new features to these FOSS applications, the security considerations are considered initially. If the security principles are not satisfied, then the feature cannot be considered.
3. FOSS structures are more secured. They have well-permissioned structure, depending on the UNIX form with well-structured network. These models are vital because several users share one server.

1.7 ECONOMICAL ASPECTS AND ADOPTION

Free software played a vital role in the advancement of the networks, World Wide Web and the framework of companies. Free software grants users to assist in improving the programs they use. It is completely a public product rather than a private product. Companies that provide free software can improve commercial novelty.

The economic feasibility of the free software has been accepted by huge corporations like IBM, Sun Microsystems and Red Hat. Several companies which are in non-IT sector, select free software for browsing information about sales because of the lesser primary asset and the capability to freely adapt the application packages.

In the free software business model, dealers may charge a bill for allocation and support through payment. Proprietary software uses a distinct business model where a proprietary software customer has to pay bills for a license to utilize the software themselves. Frequently, some level of assistance is included while buying the proprietary software, but additional services are usually accessible for an extra payment. A number of proprietary software dealers will also customize software for a payment.

Free software is regularly available at no cost and can affect permanently at lower costs compared to the proprietary software. In free software, businesses can make software to their specific requirements by modifying the software themselves or by employing programmers to change it. Free software has no assurance and does not assign legal responsibility to anyone. However, warranties are allowed between any two parties upon the condition and its utilization. Such an agreement is made individually from the free license software.

A statement by Standish Group predicts that acceptance of free software has caused a fall in income to the proprietary software industry. Irrespective of this, Eric S. Raymond disputes that the term free software is unclear, and therefore, a threat to the business community. Raymond popularizes the term Open Source Software as friendly which is an option for the business.

1.8 APPLICATIONS OF OPEN SOURCE SOFTWARE

FOSS is applied in the fields of artificial intelligence, CAD (especially, in the field of Electronic Design Automation), statistics, surveys, computer simulation, finance, integrated library system, mathematics and sciences like bioinformatics, grid computing, molecular dynamics, molecule viewer, nanotechnology, microscope image processing and plotting.

1.8.1 Science

We can examine and present the scientific investigation result under this group. These are the free open source software under science category.

- Astronomy software
- Chemistry software
- GIS software
- Linguistic software
- Mathematics software
- Physics software
- Plotting software

Applications of FOSS in the Different Fields of Science

Nanotechnology: It includes the free open source software like Ninithi Software. It is a software which is used for visualization and analysation of allotropes like fullerene, carbon nanotube, graphene nanoribbons.

Bioinformatics: BioJava, BioPerl, BioPython, BioPHP, Visomics, EMBOSS , etc., are the freely Available Bioinformatics software.

Grid computing (P-GRADE portal): It is a grid portal program that facilitates the creation of workflows, execution of workflows and monitoring of workflows.

Molecular dynamics: GROMACS, NAMD and LAMMPS are the different freely available software in the field of molecular dynamics.

Molecular viewer: Avagadro, Jmol, PyMOL, RasMol and BALLView are the different freely available software in the area of molecular viewer.

Geographic Information Systems (GIS): The various GIS software that are available freely are GeoServer, GeoTools, GeoNetwork open source, GeoMapping Tools, Mapserver, MapWindow GIS and Whitebox Geospatial Analysis Tools.

1.8.2 Mathematics

The open source software included under mathematics is utilized for higher order calculations of mathematics. This software plays a vital role in the area of mathematics.

Computer algebra systems: A computer algebra system is a kind of software that is utilized for manipulating the mathematical formulae. It systematizes repeated and occasionally challenging algebraic manipulating responsibilities. An axiom is an all-purpose computer algebra system.

Numerical Analysis: Numerical analysis is a field of mathematics using which we can create an algorithm and analyzes an algorithm for getting numerical approximations to the continuous variable problems.

Statistics: Collation and interpretation of numerical information from data can be studied using statistics.

Multipurpose mathematics software: Multipurpose mathematics software was developed with an intention of generating a math platform. This software is compared with proprietary software like Matlab and Mathematica.

1.8.3 Computer Simulation

Blender: The code is written in python and C++. It is a 3D modelling software. Using Blender, we can do different tasks, like gaming and animations.
SimPy: Based on Python, SimPy is developed. It is a simulator of event-based type.
Flightgear: Flightgear is developed for operating systems, like Windows, Mac and Linux. It is an open source simulator.

1.8.4 Statistics

The free statistical software are as mentioned below:

- Free Bayesian software
- Free Econometrics software
- Free Data Analysis software
- R (programming language) software
- Plotting software
- Web Analysis software

1.8.5 Surveys

Lime Survey is a free open source online survey function, which is developed in PHP. It permits users to create and publish online surveys without performing any coding.

1.9 RECAPITULATION

Open Source Software's (OSS) are user-friendly and are available as commercial and free licensed software. So, it is easy to use by anyone from anywhere. Open source can be easily modifiable as its core code is publicly accessible. The regular software structure should be standard. Standard software allows both parallel and distributed development on independent components of the software. OSS is easily understandable for developers. The term Free Open Source Software was mentioned for the first time in the newsgroup. According to FSF definition, if any program has four freedoms, then that program is said to be free software.

Richard Stallman on September 27, 1983 has sent a primary declaration of GNU project to the newsgroups. The GNU manifesto is a manuscript that strengthens the primary announcement. Stallman has supplemented the data on GNU, ways of granting to project and advantages from the project. The two slogans with different activities and philosophies are free software and open source. Open Source license allows the software to be freely usable, editable and sharable. A collection of ideas with architecture that is going to design the entire process is said to be software. Using of the term Open Source Software than using Free Software is prescribed by the FSF.

Security is the capability of a structure to supervise, defend and share information. It is a complex task. Free software played a vital role in the advancement of the networks, World Wide Web and the framework of companies. Free software is regularly available at no cost and can affect permanently at lower costs compared to proprietary software. FOSS is applied in the fields of artificial intelligence, CAD (especially in the field of Electronic Design Automation), statistics, surveys, computer simulation, finance, integrated library system, mathematics and sciences, like

bioinformatics, grid computing, molecular dynamics, molecule viewer, nanotechnology, microscope image processing and plotting.

REVIEW QUESTIONS

1. Explain the needs of open source applications.
2. List the advantages and disadvantages of free open source software.
3. Explain the origin of free open source software.
4. Elucidate the meaning and extraction of the term free software and open source software.
5. Compare and contrast the free software and open source software licenses.
6. What is meant by licensing? List all the familiar licenses.
7. Define game theory and discuss it.
8. Explain the security and reliability of free open source software.
9. List and explicate the applications of open source software in the following fields: (a) Science (b) Mathematics (c) Statistics (d) Surveys

1.10 Open Source Grid Computing

1.10.1 INTRODUCTION

Grid computing integrates computers from several administrative domains to execute some specific task. Grids are a part of distributed computing in which the 'super virtual computer' is the combination of multiple networked loosely coupled systems which act together to execute huge tasks. Open source grid computing offers hardware and software development services for the customers who design, build and deploy high performance computing applications on an open source systems. It includes device driver development, firmware development, kernel modification, distributed file system development, and system software design.

1.10.2 OPEN GRID SERVICE ARCHITECTURE (OGSA)

Open Grid Service Architecture (OGSA) is a collection of standards that defines the approach in which the information is distributed among different components of huge, and heterogeneous Grid systems. In this perspective, a Grid system is an extensible Wide Area Network (WAN) that maintains resource sharing. OGSA illustrates a service oriented infrastructure for a Grid computing platform designed for business and scientific purpose. OGSA is a trademark of the Open Grid Forum. OGSA was enhanced within the Open Grid Forum (OGF). It is also called the Global Grid Forum (GGF).

1.10.2.1 OGSA Definition

Based on OGSA standards, it is defined as follows:

- An architectural method in which the GGF's working group assembles requirements and manages a group of informational documents that explains the architecture.
- A group of normative requirements which outlines the document that contains exact requirements for satisfying hardware or a software component.
- Software components that hold the OGSA requirements and outlines.
- Facilitates the deployment of Grid solutions that are interoperable irrespective of the the dependency on the handling of multiple resources.

1.10.2.2 Capabilities of an OGSA

- Information services
- Data services
- Infrastructure services
- Security services
- Execution management services
- Self-management services
- Resource management services

In late 2006, a modernized version of OGSA and different related documents were published. The OGSI, i.e., Open Grid Services Infrastructure is associated to OGSA, as it was basically intended to form the basic OGSA plumbing layer. It was outdated by a Web Services Resource Framework (WSRF) and WS-Management. Figure 10.1 shows the open grid services architecture.

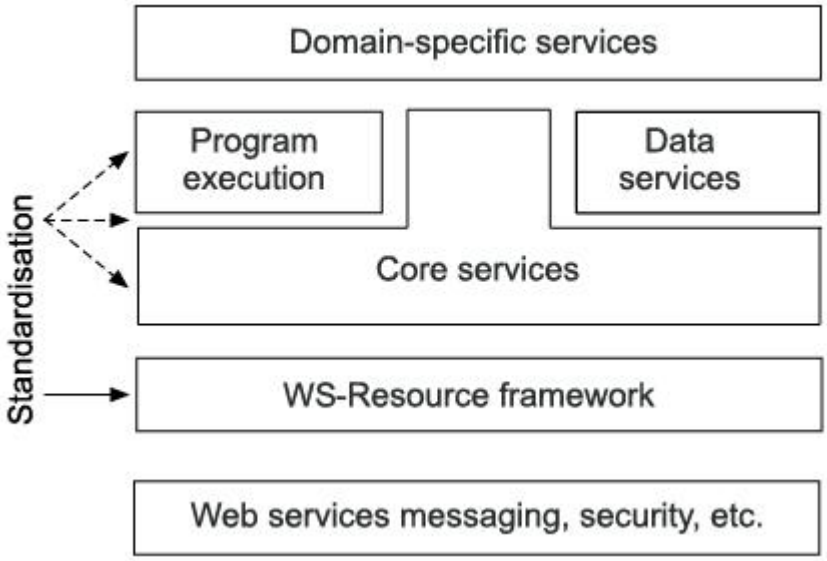

Figure 1 : Open Grid Service Architecture.

1.10.2.3 Description

OGSA is a shared communication and computing architecture based service which assures interoperability of heterogeneous systems. Therefore, dissimilar resources can communicate and distribute information. OGSA is based on different web service technologies such as the WSDL, i.e., Web Service Description Language and SOAP, i.e., Simple Object Access Protocol. But, it targets to be fundamentally independent of transport level management of data. It can be defined as a fine-tuning of web service architecture, exclusively designed to maintain Grid requirements. The idea of OGSA has resulted from the work presented in the Globus Alliance paper related to the Grid Physiology by Ian Foster in 2002. It was enhanced by Global Grid Forum (GGF) working groups which results in a document, named 'The Open Grid Service Architecture'. The GGF made some use case scenarios to be available.

1.10.3. OPEN GRID SERVICE INFRASTRUCTURE (OGSI)

The Global Grid Forum (GGF) reported Open Grid Services Infrastructure (OGSI) in June, 2003. It was planned to offer an infrastructure layer for the Open Grid Service Architecture (OGSA). OGSI obtains the statelessness problems into account by necessarily extending web services to hold Grid computing resources. Establishing on both Grid and web service technologies, the OGSI describes the procedure for creating, managing and exchanging data among entities known as Grid services.

1.10.3.1 Grid Service

In brief, a Grid service is described as a web service that adapts to a set of interfaces and behaviors which defines the client interaction with a Grid service. These interfaces, behaviors and other OGSI mechanisms coupled with the Grid service creation and discovery, presents a fault-resilient, secured management and long lived state which is usually required in sophisticated distributed applications. It introduces a set of Web Service Definition Language (WSDL) conventions that we apply in Grid service specification. These conventions are incorporated in WSDL 1.2. The Grid service defines the service data which offers a standard way of presenting and querying Meta data along with state data from a service instance. It introduces a series of Grid service core properties, such as:

- Defines a Grid service description and instance, as an organizational principle.
- Defines the way of modeling time by OGSI.
- Defines the Grid Service Handle (GSH) and Grid Service Reference (GSR) constructs which are used to refer Grid service instances.
- Defines a general approach for transferring fault information from operations. This procedure defines a fundamental XML schema definition and related semantics for WSDL error messages to support a regular analysis. The method basically defines the basic format for fault messages, without altering the WSDL fault message model.
- Define the Grid service instance life cycle.

1.10.3.2 WSDL Extensions and Conventions

OGSI depends on the web services. In specific, it uses WSDL as the procedure to define the Grid service public interfaces. However, WSDL 1.1 is incomplete in two critical areas as follows:

1. Lack of interface (portType)
2. Lack of open-content

These deficiencies have been referenced by the W3C Group because WSDL 1.2 is ‘work in development’. OGSI cannot directly integrate the entire WSDL 1.2. Instead, OGSI defines WSDL 1.1 extension, separated to the wsdl:portType element which provides the minimal essential extensions to WSDL 1.1. We can define an individual namespace with the prefix gwsdl, to isolate the adaptations to WSDL 1.1. In specific, gwsdl adds the following new constructs to the wsdl:portType element to maintain the open content model and portType extension.

1.10.3.3 Service Data

The procedure for stateful web services initiated in OGSI recognized the need for a regular process to reveal state data to service requestors of a service example for querying, updating and change notification. As this concept is applicable to the entire web services, including those that are used in the external context of Grid applications, a common approach is proposed to expose web service state data called ‘service data’.

1.10.4. SECURITY ISSUES

Security issues related to traditional and Grid systems are as follows:

Traditional systems: The security issues include defending a system from its users and specific user from negotiation.

Grid systems: The security issues for a Grid system are as follows:

1. Defending applications and data from the system where computations are executed.
2. Stronger verification is necessary.
3. Defend local implementation of remote systems.
4. Dissimilar admin domains/security policies.

1.10.4.1 Authentication

Authentication is a procedure of validating the uniqueness of a participant to perform a function or request. A principal is an entity whose identity is verified by the restricted user or user who has logged into a remote system. Traditional systems validate client to defend server. In case of Grid system, a mutual authentication is needed to ensure that resources and data should not be supplied by an attacker.

Authentication Methods

Password-based authentication: There are two methods in case of password-based authentications which are as follows:

- Send unencrypted passwords: It is suitable only when the messages cannot be read by entrusting processes while on the network.
- Prove knowledge of a password: In place of sending a password over the network, send password as an encryption key. Encryption should be done for a known value which should be non-repeating.
-

Kerberos authentication : It is well-suited for frequent authentication. It is centrally administered and requires trusted, online certification.

Secure Sockets Layer (SSL) : SSL can be deployed in any browser, i.e., widely deployed. In this, client authenticates the server identity and sends a session key from client to server for setting up an encrypted connection. The server consists of a certificate with a public key. If the client has a certificate, then it can authenticate itself to the server.

Symmetric crypto system: It uses the same key for both encryption and decryption. Data Encryption Standard (DES), RC4, RC5 and triple-DES are the examples. With static keys, user needs dissimilar key for every service provider. Service provider maintains key for every user.

Asymmetric cryptography: It is also called as Public Key Cryptography (PKI). Digital signature algorithm is an example for asymmetric cryptography. It uses a pair of keys for both encryption and decryption. The public key is published and available to anyone, whereas the private key is a secret key and known to only single party. It has an advantage of disseminating public key freely and a disadvantage of worse performance compared to symmetric encryption.

Authentication and key distribution protocol: It is applied to symmetric encryption systems and performs better when compared to the systems which are using public key or asymmetric cryptography.

1.10.4.2 Additional Security Issues

Assurance: It is a form of authorization which is used to validate service provider. It will check whether the requirements of the client, such as performance, security and reliability are met or not.
Accounting: It is a means of tracking, limiting or charging for consumption of resources. It is critical for fair allocation of resources. Accounting is critical because it needs a payment and incentives to make use of resources.

Audit: It records operations performed by a system and integrates actions with principals. In the Grid, the audit mechanism must be distributed.

1.10.4.3 Security Technologies

IPSec and IPv6: IPSec and IPv6 act as a transport layer protection for confidentiality and integrity. When communication is established between the two network hosts, it uses key distribution to exchange keys for symmetric encryption. Key distribution may use kerneros. The keys are connected to hosts, but not with applications or users.

Virtual Private Networks (VPNs): VPNs operate at the transport layer. It provides communication only between participating nodes. It uses transport layer confidentiality and integrity. The features of VPNs are as follows:

1. Authenticate end users
2. Recognize application level objects that need protection
3. Maintain security policies that differentiate users and application objects

Firewalls: Firewalls prevent several attacks on hosts within the organization. It provides an obstacle at the organizational network boundary. Grid application often requires communication through the firewall. Firewalls need to combine IPsec along with VPN technologies at network boundaries with firewalls.

1.10.5. GLOBUS TOOLKIT

Grid computing has an open source Toolkit known as a Globus Tool Kit which is offered by the Globus Alliance. The standard implementation of the Globus Toolkit is performed on the basis of specific standards which are as follows:

- Job Submission Description Language (JSDL)
- Open Grid Services Architecture (OGSA)
- Grid Security Infrastructure (GSI)
- Distributed Resource Management Application API (DRMAA)
- Open Grid Services Infrastructure (OGSI) which is fundamentally planned to produce the central ‘plumbing’ layer for OGSA, but has been outdated by WSRF as well as WS management.

- Web Services Resource Framework (WSRF)

The open source Globus Toolkit is a central technology for the 'Grid' which allows the people to distribute the computing resources, databases and other tools in the online securely across commercial and geographic boundaries. The toolkit comprises software services and libraries for resource supervision, discovery, security and file management.

The Globus Toolkit includes software for the purpose of security, resource management, data management, fault detection, portability and communication. The toolkit is packed as a group of components that can be utilized either individually or combined to extend the applications. All the organizations have individual operational modes and association between multiple organizations is delayed by incompatible resources, such as, networks, computers and data archives. The toolkit was considered to eliminate complexities that stop the collaboration. Its central services, protocols and interfaces facilitate the users to access remote resources.

The Globus Toolkit has developed on the basis of open source strategy analogous to the Linux operating system and different from proprietary effort in resource sharing software. This promotes wider and faster adoption which leads to better technical innovations, as the open source community offers persistent improvement to the product.

1.10.5.1 Versions

The Globus Toolkit version plan is analogous to the technique used for the Linux kernel prior to 3.0 versions of the Globus Toolkit which are labeled with a version number, comprises three parts, i.e., major, minor and point.

- The major release number stands for the main state of the Globus Toolkit. It is increased when the considerable architectural shifts take place in Toolkit development.
- The change in the minor release number specifies that several new features have been included in the Globus Toolkit. API changes along with binary incompatibility may occur linking minor releases.
- The point release number alters when the Globus Toolkit components have been renewed. Public interfaces may be supplemented, but not detached between point releases of the constant release version. An installation can be promoted to a new point release through native packages with the prospect that all services along with libraries should persists to work as earlier versions, unless innovative features are enabled.

1.10.5.2. Stable and Development Releases

Stable and development are the two series in the Globus Toolkit release streams. Stable distributions include a minor version number as even number, whereas development distributions include an odd version number. Globus-4.0.1, Globus-4.0.0, Globus-3.2.1 are the examples for stable releases. Globus-3.9.5, Globus-3.9.4, Globus-3.3.0 are the development releases. Releases including under development series are used to distribute novel features in a timely approach. Development distributions can act in irregular ways and should not be applied in production settings. Unlike stable releases, there is no declaration of binary compatibility between point versions of development releases.

1.10.5.3. Globus Toolkit Version

Globus Toolkit version includes the following:

- Version 1 is essentially a research prototype which is not broadly used.

- Version 2 is extensively used in non-web-based services.
- Version 3 is a web-based service. But it is not widely accepted.
- Version 4 is a web-based service and widely used to acquire new software difficulties.
- Version 5 returns to a non-web-based approach of version 2.

1.10.5.4. Globus Toolkit4 (GT4) Architecture

Globus Toolkit 4 is a collection of many software components which are divided into following five categories.

1. Security: The connections should be secured based on the Grid Service Infrastructure (GSI). Information services:
2. The information services are also called as Monitoring and Discovery Services (MDS), comprises a collection of components to discover and supervise resources in a virtual organization. GT4 also comprises a non-web service version of MDS2 for legacy purposes. This component will be disappearing in forthcoming releases of the Toolkit.
3. Execution management: It deals with the initiation, monitoring, coordination and management of executable programs in a Grid.
4. Data management: The Common Runtime components present a set of basic libraries and tools which are required to build web services along with non-web services.
5. Common runtime: The Common Runtime components offer a set of fundamental libraries along with tools which are necessary to construct both web services and non-web services.

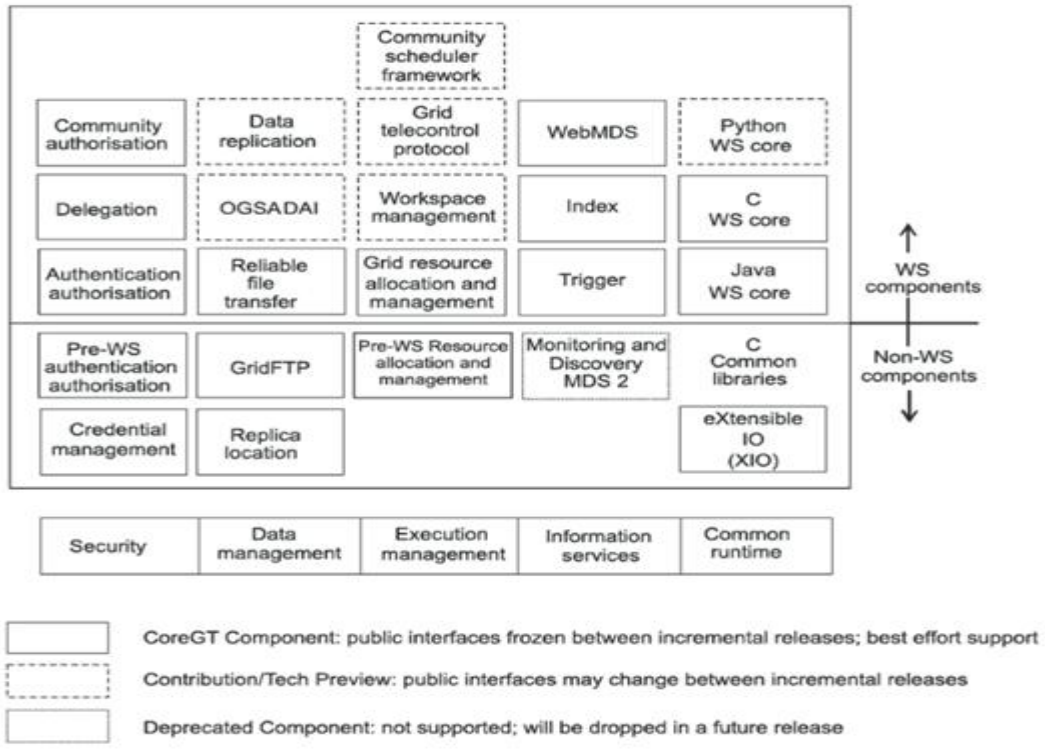

Figure 2 : GT4 Architecture

1.10.5.5 Globus Toolkit Programming Model

In the Globus Toolkit, the library functions are actually used. The truth that we use an asynchronous programming model regularly seems strange to people rising C applications with the Globus Toolkit. It includes the following:

Include Headers

The entire code that formulates calls to functions in the GridFTP client library must include the following header: #include _client globus_ftp.h

Module Activation/Initialization

In every Globus Toolkit C code, you must call the module activate/deactivate for all the modules that perform direct calls. The module determines the activation and deactivation of its own dependencies.

Handle Setup

Every function call is an entirely encapsulated GridFTP session. In a GridFTP session, the control connection is created, and authentication is performed. If necessary, a data channel is set in a GridFTP session, and then the required data will be transferred.

Check Features

Verification of the server compatibility related to the feature is a good practice. Verification of a feature comprises four steps. In the first step, init function is called. The second step, requires to invoke the feature function which sends the FTP FEAT command to the server and loads the structure along with the results. The third step includes accessing the results by means of is_feature_supported by listing the features of interest. In the fourth step, call features_destroy to free the structure memory is needed.

Set Operation Attributes

After knowing that what are all the features that are supported by the server's one is using, one can configure any essential attributes. The file system functions may not use attributes, however the data movement operations often use attributes. For a data movement operation, if one does not denote any attributes, the default will be the standard RFC959 stream mode.

The operation attributes can be divided into two rough categories. The first category is the data movement and the second category is the security. All the function calls enclose the same form

The set variant changes the attribute value and the get variant returns the or image. Mode: GridFTP supports two modes. The first mode is the stream mode and the second mode is the extended block mode. In stream mode, the file is navigated by sending the bytes as an ordered sequence of bytes. An extended block mode is a GridFTP extension which sends the data in blocks with eight bits of the flag and 64-bit offset. This format allows out-of-order reception of the data since the transmitting offset is specified. With out-of-order data, we can now send the data in multiple paths. In the existing version of the server, the parallelism is invoked. It involves only a single source and destination network endpoint but includes multiple TCP streams.

Parallelism

Parallelism indicates the number of TCP streams that should be opened between every network endpoint.

Module Deactivation and Clean Up

Once the work is completed, the cleanup remains. One should remind about the cleaning of the buffer that have utilized by means of the statement as follows: globus_error_get() globus_print_friendly() These statements destroy all the tasks that have initiated and then deactivate all the modules by means of the statement as follows: globus_module_deactivate_all()

REVIEW QUESTIONS

1. Define OGSA and explain its features.
2. Sketch and explain the OGSA architecture.
3. Explain OGSI and its services.
4. What are all the security issues in Grid computing?
5. What is meant by authentication and explain the different authentication methods used for verification?
6. Explain the additional security issues in Grid computing.
7. List and explain the security technologies in Grid computing.
8. What is meant by Globus Toolkit and explain its standards?
9. Mention the Globus Toolkit versions.
10. Explain the stable and development releases of a Globus Toolkit.
11. Sketch and explain the GT4 architecture.
12. List and explain the GT4 components.
13. Explain the Globus Toolkit programming model.

1.11. Open Source Cloud

1.11.1 INTRODUCTION

Cloud computing is generally defined as a computer network which includes, computing hardware machine or group of computing hardware machines commonly referred as a server or servers connected to a network , such as the Internet , an intranet , a Local Area Network (LAN) or Wide Area Network (WAN) . Any user who has a license to access the server can use the server's processing power to run an application, store data, or perform any other computing task. Previously users use to run their applications on their own computers, but these days the individual runs the application from anywhere in the world, as the network of servers provides the processing power to the application and the servers are also connected to the Internet or other connection platforms to be accessed from anywhere.

The relationship between open source and cloud computing is symbiotic. Open source presents a fundamental foundation for cloud computing. Software as a Service (SaaS) providers, such as Google depends profoundly on the operating systems (open source), such as Linux to execute their functions. Linux machines were first offered by the Amazon cloud which persists to present new services on Linux first. Moreover, Open Source Software has low-friction licensing which is a usual fit for cloud computing. Business dealers often hesitate to completely support their products in a cloud computing platform. Placing up free piles of open source software and formulating them accessible is a powerful model for computational advancements.

The significant cloud computing vendors, such as Amazon, Microsoft and Google have not encouraged open standards for their own services. These vendors do not discharge the code

for their services. Therefore, it is a complex task to set up an accurate equivalent code. For example, Open Stack and Eucalyptus has made some efforts to build an open cloud standard but they have not achieved wide acceptance yet.

1.11.2 FOSS CLOUD SOFTWARE ENVIRONMENTS

Cloud computing is the ability to execute a program on several computers at the same time. It illustrates distinct theories, which involves a huge number of computers linked through a network.

1.11.2.1 Characteristics of a Cloud

- Self-service based on requirement
- A wide network access
- Resource sharing
- Rapid elasticity

1.11.2.2 Why Open Source?

- User driven solutions to the existing difficulties.
- Fewer obstacles for participation.
- User dependent, i.e., one user helps another user.
- Open data, standards and APIs.

1.11.2.3 Open Source Clouds

The open source cloud is a software that facilitates to construct virtualization and services of cloud in a private or a public cloud. The open source cloud is a shared server design that offers virtualization and cloud services, terminal server, Virtual Desktop Infrastructure (VDI) and Software as a Service (SaaS) based on Windows or Linux. The open source cloud makes the virtual machines to be accessed internally or from the internet. The FOSS Cloud envelops all the characteristics of an open source IT platform. An open source cloud is licensed under the European Union Public License (EUPL). OVF, i.e., Open Virtualization Format is an open pattern for wrapping and sharing virtual applications or more generally software to work on virtual machines. Eucalyptus, Open Nebula and Open Stack are the open source clouds.

REVIEW QUESTIONS

1. Define and explain open source cloud platform.
2. List and explain the characteristics of a cloud.
3. List all the open source clouds and explain.