OPEN SOURCE SOFTWARE (OSS) IMPLEMENTATION GUIDELINES

MALAYSIAN PUBLIC SECTOR OPEN SOURCE SOFTWARE (OSS) INITIATIVE
COPYRIGHT

- The Copyright to this document is owned by the Government of Malaysia
TABLE OF CONTENTS
# INTRODUCTION

1.1 Motivation For OSS Initiative

1.2 Supplementary Documents

1.3 Purpose Of The Document

1.4 Objectives Of The OSS Implementation Guidelines

1.5 Scope

1.6 Audience

# ADOPTION

2.1 Guidelines For Adoption

2.1.1 Establish An OSS Team

2.1.2 OSS Adoption Considerations In Agency’s ISP

2.1.3 Software Maturity Assessment

2.1.4 Identify And Develop OSS Opportunity Areas

2.1.5 Implement Pilot Projects

2.1.6 Plan For Phased Deployment

2.1.7 Plan For Change Management

# OSS PROCUREMENT

3.1 Guidelines For Procurement

3.1.1 Support For Open Standards

3.1.2 Access To Source Code

3.1.3 Security In IT Solutions

3.1.4 Best Value For Money

3.1.5 Merit-Based Selection

3.1.6 Sourcing Scenarios

# OWNERSHIP

4.1 Guidelines For Ownership

4.1.1 Understanding The Legal Background To OSS

4.1.2 Minimum Standards As Set Down By The OSI

4.1.3 OSS Licenses In General

4.1.4 Implementing/Using OSS Licenses In The GOM

4.1.5 A Summary Of Common And Relevant OSS Licenses

4.1.6 Identify The Salient Considerations In Specific OSS Licensing Scenarios

4.1.7 Likely OSS Acquisition, Modification & Distribution Scenarios

4.1.8 Ownership
## TECHNOLOGY

5. **Guidelines For Technology**
   - **Technology Recommendations**
   - **Adhere To Standards Defined In MYGIFOSS**
   - **Ensure Continuity of Support**

## IMPLEMENTATION

6. **Guidelines For Implementation**
   - **Refer To Technical Implementation Plan, Knowledge Bank And MyGIFOSS**
   - **Implement OSS Within The Six (6) Solution Areas In Phases**

## KNOWLEDGE SHARING

7. **Guidelines For Knowledge Sharing**
   - **Register All OSS Solutions and Projects In The OSCC Knowledge Bank**
   - **Open Access To All Knowledge And Resources**
   - **Share All Modified OSS**
   - **Report All Bugs**
   - **Communicate And Share Information**

## OSS EDUCATION

8. **Guidelines For OSS Education**
   - **Proliferate OSS in IT Labs**
   - **OSS Infrastructure In Learning Institutions**
   - **Develop Education Materials Using OSS**
   - **Reskill Existing IT Coordinator**
   - **OSS At Teacher Training Colleges**

## TRAINING

9. **Guidelines For Training**
   - **Training Roadmap**
   - **Technical Training**
   - **Non-Technical Training**
   - **Train the Trainer**
LIST OF TABLES

Table 1.1 – Strategic Thrusts and Guidelines Matrix 3
Table 2.1 – Roles and Responsibilities of the OSS Team 8
Table 2.2 – OSS Technical Implementation in the Short Term 19
Table 2.3 – OSS Technical Implementation in the Medium Term 19
Table 2.4 – OSS Technical Implementation in the Long Term 20
Table 4.1 – Key Differences between Proprietary Software and OSS 31
Table 4.2 – Guidance in Deciding on the OSS License 40
Table 4.3 – Reference Point as to Whether Sharing Amounts to Redistribution 41
Table 4.4 – Modification Guideline 42
Table 4.5 – Guidance in relation to Specific Licensing Requirements of identified OSS Licenses 43
Table 4.6 – Questions in relation to OSS License’s warranties and indemnities availability 43
Table 8.1 – Available OSS for Education 55

LIST OF FIGURES

Figure 1.1 – Malaysian Public Sector OSS Framework and 7 Strategic Thrusts 2
Figure 2.1 – OSS Team Structure 7
Figure 2.2 – OSS Adoption Considerations in Agency’s ISP 9
Figure 2.3 – OSS Implementation Scenario Matric 12
Figure 2.4 – Identify and Develop OSS Opportunity Areas 12
Figure 2.5 – Type of OSS Solution Areas 18
Figure 3.1 – Workflow for In-House Sourcing for OSS 26
Figure 3.2 – Workflow for External Sourcing for OSS 28
Figure 4.1 – Minimum Licensing Terms Set By The OSI 32
Figure 6.1 – OSS Technical Implementation Plan for the Public Sector 50
Figure 9.1 – OSS Training Roadmap 59

LIST OF APPENDICES

Appendix 1 – Navica OSMM Model 63
Appendix 2 – Cap Gemini OSMM Model 67
1. INTRODUCTION

On 19 June 2002, the Government of Malaysia (GOM) has decided to encourage the adoption of Open Source Software (OSS) in the Public Sector. The Malaysian Administrative Modernisation and Management Planning Unit (MAMPU) of the Prime Ministers Department is entrusted to lead and implement this OSS Initiative.

On 16 July 2004, the *Malaysian Public Sector OSS Master Plan* (OSS Master Plan) was published together with the launching of the Open Source Competency Centre (OSCC). Within the Master Plan, the OSS Framework was designed and developed to provide guidance for its implementation. In line with the framework, seven (7) strategic thrusts were identified that defined the high level implementation action plan of the OSS Initiative:

i. Develop OSS Technical Implementation Plan for the Public Sector

ii. Entrust a governing body to champion, monitor & drive OSS implementation

iii. Train and develop human resource to support OSS implementation

iv. Promote creativity and innovativeness via R&D to harness competitiveness

v. Continuous development of policies & legal direction to encourage utilisation and production of OSS

vi. Provide incentives to prosper the development of OSS solutions

vii. Optimise resources by encouraging smart partnerships with relevant organisations
Figure 1.1 – Malaysian Public Sector OSS Framework And 7 Strategic Thrusts

Figure 1.1 above illustrates the components of the Malaysian Public Sector OSS Framework and its seven (7) Strategic Thrusts.

To support the strategic thrusts, eight (8) policy areas were identified as follows:

- Adoption
- Procurement
- Ownership
- Technology
- Implementation
- Knowledge Sharing
- Education
- Training

For each of the strategic thrusts, guidelines were developed with the objective of creating conducive enabling environment for successful OSS implementation in the Public Sector. The following matrix highlights each strategic thrust with corresponding OSS implementation guidelines.
The vision for OSS in the Malaysian Public Sector is to create and enhance value by using OSS within the Public Sector ICT framework in providing efficient and quality services. The Government of Malaysia (GOM) will implement OSS wherever it indicates to be the most appropriate option.

It is envisioned by the GOM that all Public Sector agencies will align their respective ICT plans, resources and actions towards achieving objectives of the OSS Initiative, that are to:

- Reduce total cost of ownership
- Increase freedom of choice of software usage
- Increase interoperability among systems
- Increase growth of the ICT industry
- Increase growth of the OSS industry
- Increase growth of OSS user and developer community
- Increase growth of knowledge-based society
- Reduce digital divide

Among the steps taken to support this Initiative is the establishment of the Open Source Competency Centre (OSCC). OSCC was established to be the single point of reference, to lead, guide, facilitate, coordinate and monitor the implementation of OSS in the Public Sector. Information and documents related to the OSS Initiative are available at http://opensource.mampu.gov.my and http://oscc.org.my
1.2 Supplementary Documents

This document should be read together with The Malaysian Public Sector OSS Master Plan. The OSS Master Plan charts the overall approach and strategy for the adoption and implementation of OSS and related technologies in the Public Sector.

This document should also be read concurrently with The OSS Technical Implementation Plan and The Malaysian Government Interoperability Framework for Open Source Software (MyGIFOSS).


1.3 Purpose Of The Document

This document serves as a guide for the implementation of OSS in the Public Sector. In addition, this document aims to:

- Create awareness of OSS within the Public Sector
- Provide a consistent approach to the deployment and implementation of OSS in the Public Sector
- Provide a common framework based on best practice principles to support and encourage utilisation of OSS within the Public Sector

1.4 Objectives Of The OSS Implementation Guidelines

The OSS implementation guidelines are designed to assist and guide the stakeholders within the Public Sector in understanding and taking the right direction in the utilisation and production of OSS. These guidelines are intended to provide the right enabling environment for the implementation of OSS in the Public Sector. These guidelines will be reviewed and adjusted from time to time, in order to reflect advances in technology and the requirements of the Public Sector.
1.5 Scope

The scope of the OSS Implementation Guidelines covers the following areas:

- Adoption
- Procurement
- Ownership
- Technology
- Implementation
- Knowledge Sharing
- Education
- Training

1.6 Audience

This document is intended for use by Public Sector agencies to gain understanding on how to implement OSS in their respective agencies. The target audiences for this document are the Head of Departments (HODs), Chief Information Officers (CIOs) and all ICT personnel.
2 ADOPTION

The objective of the OSS Adoption Guidelines is to assist the process of adoption of OSS and its success within the Public Sector. To this end, Public Sector agencies are required to consider adopting OSS in their respective agencies in accordance with the OSS Master Plan, this document and the agencies ICT Strategic Plan (ISP).

Within the OSS Master Plan, potential areas in which OSS can be deployed have been categorised into six (6) solution areas, namely:

- Workload Consolidation
- High Performance Computing
- Distributed Enterprise
- Application Solution
- Infrastructure Solution
- Desktop Solution

Public Sector agencies are required to identify opportunities (opportunity areas) for implementing OSS within the said six (6) solution areas (please refer to Section 2.1.4 for further details).

The selection of opportunity areas for OSS implementation must be based on the guiding principles detailed in the OSS Master Plan, namely:

- Fit for purpose in terms of functionality, the technology platform and user requirements as a whole.
- Least disruptive to operations specifically in relation to the Public Sector agency’s technical infrastructure, users and day-to-day operations.
- Co-exist with other legacy proprietary systems in the current environment.
- Leverage on existing facilities, hardware, software and expertise. Skills of existing resources should be enhanced through OSS training and knowledge.
- Not driven or controlled by hardware or software vendors. Government Agency should seek to gradually achieve independence from hardware and software vendor(s) by cultivating OSS skills and expertise.
2.1 Guidelines For Adoption

The Public Sector agencies are encouraged to consider the following steps in the process of adopting OSS:

- To establish an OSS team
- To consider OSS in agency’s ISP
- To assess software maturity
- To identify and develop OSS opportunity areas
- To implement pilot project
- To plan for phased deployment
- To plan for Change Management

2.1.1 Establish An OSS Team

Public Sector agencies are encouraged to establish an OSS team with appropriate skills and management support. The roles and responsibilities of the team include, among others the following:

- Become the OSS champion
- Secure buy-in from senior management, technical management and financial management
- Perform Change Management programme
- Construct the Term of Reference of agency’s OSS project

The new team setup shall consist of management, end-user and IT technical staff. **Figure 2.1** below depicts the recommended structure of the OSS team.

![Figure 2.1 – OSS Team Structure](image-url)
The responsibilities listed in the following table are just a few of the many functional roles that may exist in the project.

<table>
<thead>
<tr>
<th>No.</th>
<th>Team Members</th>
<th>Roles</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Head of Department/ Deputy Head of Department</td>
<td>OSS Team Leader</td>
<td>The Team leader is expected to become the Project Champion for OSS in the respective agency. This person must be able to identify a need for OSS adoption and has the authority to make all decisions relating to the change to OSS at the agency.</td>
</tr>
<tr>
<td>2.</td>
<td>End Users</td>
<td>User Team</td>
<td>End users form valuable resources in the team - they can be used for many purposes related to the design, construction and delivery of the OSS solution. As members of the team they will act as experts and coaches for the new OSS solution.</td>
</tr>
<tr>
<td>3.</td>
<td>End Users / System Analysts / Web Designers</td>
<td>Process Team</td>
<td>Process team shall be responsible for defining business or operational needs within the six solution areas at the agency. Together with the Technical and End User Team, the Process Team shall be responsible to identify the suitable OSS solution to serve the business or operational requirements. They shall also gather data, conduct Cost Benefit Analysis, perform risk assessment and mitigation plan as well as identify issues regarding the OSS solutions of interest.</td>
</tr>
<tr>
<td>4.</td>
<td>System Analysts/ Programmers/ Database Administrator / System Administrator</td>
<td>Technical and Application Team</td>
<td>Technical personnel must be expert in defining OSS technical components and architecture of any particular OSS solution within the agency.</td>
</tr>
</tbody>
</table>

**Table 2.1 – Roles And Responsibilities Of The OSS Team**

### 2.1.2 OSS Adoption Considerations In Agency’s ISP

In line with the Public Sector vision and direction, each Public Sector Agency should define/redefine its ISP, with due consideration given to the adoption of OSS solution components.

This process of definition/redefinition of ISPs by Public Sector Agencies will take place at various stages of ISP implementation by each Public Sector Agency, including but not limited to the following:

- A Public Sector agency has an ISP but has yet to implement
- A Public Sector agency has an ISP and is in the midst of implementation
- A Public Sector agency has an ISP and has completed implementation
Figure 2.2 illustrates how OSS adoption can take place at the various stages of ISP implementation by Public Sector agencies as identified above.

2.1.3 Software Maturity Assessment

Each agency should also consider the maturity of the OSS before deploying it. It is recommended that the agencies adopt Open Source Maturity Model (OSMM) in order to determine if an OSS is suitable for implementation. This OSMM allows Public Sector agencies to determine if or which OSS is suitable for use within that particular agency. The OSMM is an effective method to prevent interesting but immature products from being implemented at Public Sector agencies. It serves as a useful tool to rationalise the adoption of OSS in the Public Sector.

The OSS is measured based on the following indicators:

i. **Functionality:** How well will the software meet the average user’s requirements?

ii. **Usability:** How good is the User Interface? How easy is the software to use for end-users? How easy is the software to install, configure, deploy, and maintain?

iii. **Quality:** Of what quality are the design, the code, and the tests? How complete and error-free are they?

iv. **Security:** How well does the software handle security issues? How secure is it?
v. **Performance**: How well does the software perform?

vi. **Scalability**: How well does the software scale to a large environment?

vii. **Architecture**: How well is the software designed? How modular, portable, flexible, extensible, open, and easy it is to integrate?

viii. **Support**: How well is the software component supported?

ix. **Documentation**: Of what quality is the documentation for the software?

x. **Adoption**: How well is the component adopted by the community, market, and industry?

xi. **Community**: How active and lively is the community for the software?

xii. **Professionalism**: What is the professionalism level of the development process and of the project agency as a whole?

The above mentioned indicators shall be assigned a weighted score value based on the relative importance of the indicators to the agency’s specific requirements. The value of the final score of the weighted indicators concludes the general maturity level of the product.

The basis of identifying the relevant indicators, assigning weightage to the identified indicators and scoring will need to be based on the OSMM Model adopted. There are various OSMM Models available to assess OSS maturity, of which two (2) have achieved prominence, namely:


Summaries of these models are available in the appendix of this document. Nevertheless, Public Sector agencies are free to create and/or use other OSMM models as appropriate.

### 2.1.4 Identify And Develop OSS Opportunity Areas

In order to realise the full benefits of OSS, implementing agencies should identify the relevant opportunity areas within their respective agency. Each agency’s ICT infrastructure and business requirements have their own unique characteristics. Each in turn would require specific economic and strategic considerations. Hence, it is vital for each agency to identify the correct areas in which OSS can be implemented.
The Public Sector agencies should identify and develop OSS opportunity within the six (6) solution areas:

- **Workload Consolidation** - Workload consolidation applies to projects that involve many applications that reside on multiple servers to move to fewer numbers of servers.
- **High Performance Computing (Clusters)** - High performance computing applies to projects that seek to reduce computing time in computational intensive jobs. Specifically, tasks need to be able to run in parallel in order to achieve this.
- **Distributed Enterprise** - Distributed enterprise applies to projects that seek to optimise infrastructure by deploying client-server or similar models.
- **Application Solution** - Application solutions apply to projects that involve the implementation of a total solution in order to meet business needs. Examples of these are Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), portal and e-commerce.
- **Infrastructure Solution** - Infrastructure solutions apply to projects that consist of basic infrastructure related services. Examples of these are LDAP, mail, DNS, database, high availability, firewall, IDS, file/print, web servers.
- **Desktop Solution** - Desktop solutions apply to projects that are intended at implementing client-related applications for users. Examples of these are office productivity tools, development tools, mail readers.

Within each solution area, there are potential OSS projects that can fall under any of these three (3) project lifecycles:

- **New Projects** - Public Sector agencies that would have a business need but have not defined a solution to meet the need
- **Ongoing Projects** – Public Sector agencies that are in the process of designing/deploying a solution based on commercial closed source software
- **Cut-over Projects** – Public Sector agencies that have already implemented and are currently using a solution based on commercial closed source software

Each combination of a particular solution area and project life cycle phase is defined as a scenario. The scenario provides an opportunity for OSS implementation. The eighteen (18) possible scenarios are shown in the scenario matrix in **Figure 2.3**.
Public Sector agencies can use this matrix to identify and validate OSS opportunity for implementation. For each of the opportunity areas identified, Public Sector agencies are recommended to perform the following tasks to develop the project implementation plan. The diagram below depicts the relevant tasks to be carried out for each project.

![Figure 2.3 – OSS Implementation Scenario Matrix](image)

### OSS Opportunities Scenario Matrix

<table>
<thead>
<tr>
<th>OSS Solution Areas</th>
<th>New Project</th>
<th>On-Going Project</th>
<th>Cut Over Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workload Consolidation</td>
<td>Agencies that seek to optimise large number of servers and reduce costs</td>
<td>• Define business needs</td>
<td>• Define business needs</td>
</tr>
<tr>
<td>High Performance Computing</td>
<td>Agencies that have long, running computing intensive jobs</td>
<td>• Data collection</td>
<td>• Data collection</td>
</tr>
<tr>
<td>Distributed Enterprise</td>
<td>Agencies that have multiple remote or geographically dispersed clients needing to access the same application</td>
<td>• Define possible solutions</td>
<td>• Define possible solutions</td>
</tr>
<tr>
<td>Application Solution</td>
<td>Agencies that have a business need to be fulfilled by IT</td>
<td>• Analyse Cost Benefit Analysis (CBA) projection for each solution</td>
<td>• Analyse Cost Benefit Analysis (CBA) projection for each solution</td>
</tr>
<tr>
<td>Infrastructure Solution</td>
<td>Agencies that require infrastructure solutions</td>
<td>• Produce detailed implementation plan</td>
<td>• Produce detailed implementation plan</td>
</tr>
<tr>
<td>Desktop Solution</td>
<td>Agencies that require client PCs for their users</td>
<td>• Revise the implementation plan</td>
<td>• Perform risk analysis and develop a risk mitigation plan</td>
</tr>
</tbody>
</table>

### Figure 2.4 – Identify And Develop OSS Opportunity Areas
2.1.4.1 Recommended Tasks In Adopting OSS Opportunity For New Project

A Define Business Needs

The first step to a new project implementation is to define the business requirements. It is important that the OSS solution will improve the efficiency and effectiveness of service delivery by the agency.

B Data Collection

The next step is collecting data on the existing environment. This would form the basis for the analytical work. Typical data that should be collected are user functionality requirements, current wide area network utilisation and costs as well as current operational skills.

C Define Possible Solutions

Agencies should define possible platforms and applications that would suit their needs. It is here that they would specify the number of hardware, software licenses and network bandwidth required for the solution to work. Agencies should endeavour to include as much OSS in their solutions as possible. It is important that agencies consider all possible OSS solutions at this point to ensure that none would be missed. Data flows and formats would be defined for interoperability of the various components of the solution. Technical feasibility checks involving operations, application development and user groups should also be carried out on all solutions.

D Analyse CBA Projection For Each Solution

Based on the data gathered, agencies should consider the Cost Benefit Analysis (CBA) projections for each of the defined solutions. In performing CBA projections, agencies should compare cost data of OSS solution against proprietary solution. For example, for Desktop Solution, OpenOffice is compared to Microsoft Office. Solutions that do not meet favourable CBA projections can be discarded.

E Produce Detailed Implementation Plan

Based on the selected solution, agencies should then define an implementation plan detailing the exact steps, timeline and budget to implement the project.
### 2.1.4.2 Recommended Tasks In Adopting OSS Opportunity For On-going Project

**A Define Possible Solutions**

Agencies seeking to migrate an ongoing project to an OSS based system would already have relevant data gathered on the existing environment. Agencies should define possible platforms and applications that would suit their needs. It is here that they would specify the number of hardware, software licenses and network bandwidth required for the solution to work. Agencies should work towards including as much OSS in their solutions as possible. It is important that agencies consider all possible solutions at this point to ensure that none would be missed.

**B CBA Projection For Each Solution**

Based on the data gathered, agencies should consider the CBA projections for each of the defined solutions. In performing CBA projections, agencies compare data of the current proprietary solution against the new OSS solution. For ongoing projects, CBA projections for the current implementation should only include future costs as the cost that they have already incurred may not be recoverable. Solutions that do not meet favourable CBA projections are discarded.

**C Revise The Implementation Plan**

Once a solution has been defined, the project implementation plan would have to be revised to reflect the new solution. Migration activities from the current implementation to the new one should be detailed out.

### 2.1.4.3 Recommended Tasks In Adopting OSS Opportunity For Cut-over Project

**A Redefine Business Needs**

Since the agency has already gone through an ICT implementation, it is presumed that the initial business requirements were met. However, after the exercise, agencies may have found that new requirements may have shifted their needs. Thus, the first step is to redefine the business needs.

**B Data Collection**

The next step is collecting data on the existing environment. This would form the basis for the analytical work. Typical data that should be collected are user functionality requirements, current wide area network utilisation and costs as well as current operational skills.
C Define Possible Solutions

Agencies should define possible platforms and applications that would suit their needs. It is here that they would specify the number of hardware, software licenses and network bandwidth required for the solution to work. Agencies should endeavour to include as much OSS in their solutions as possible. It is important that agencies consider all possible OSS solutions at this point to ensure that none would be missed. Data flows and formats would be defined for interoperability of the various components of the solution. Technical feasibility checks involving operations, application development and user groups should also be carried out on all solutions.

D Analyse CBA Projection For Each Solution

Based on the data gathered, agencies should consider the CBA projections for each of the defined solutions. In performing CBA projections, agencies compare data of the current proprietary solution against the new OSS solution. For cut-over projects, CBA projections for the current implementation should only include future costs as the cost that they have already incurred may not be recoverable. Solutions that do not meet favourable CBA projections are discarded.

E Produce Detailed Implementation Plan

Based on the selected solution, agencies should then define an implementation plan detailing the exact steps, timeline and budget to implement the project.

F Perform Risk Assessment And Develop Risk Mitigation Plan

Since the implementation would replace a current production environment, it may introduce some risk. This risk should be assessed and a risk mitigation plan would need to be developed. The plan should include considerations on applying patches, upgrades and bug fixes, staff and user training requirements, recovery from system failures and other general project implementation issues.

2.1.5 Implement Pilot Projects

2.1.5.1 Objectives of Pilot Projects

OSS pilot project is important in assessing the potential implementation risk associated with a full-fledged OSS deployment exercise.
The objectives of the pilot projects are:

- To develop proof of concept
- To obtain buy-in from key stakeholders within the agency
- To measure the benefits and recommend OSS solutions for roll out

2.1.5.2 Selection Criteria for Pilot Projects

In implementing pilot projects, the agencies should consider the following guidelines:

- **Manageable and feasible projects** – It is advisable to start with small scale and manageable pilot projects, involving a small number of users. The implementation cost must be within the allocated budget.
- **Quick-wins** - Choose pilot projects that are easy to implement with quick-wins and high impact.
- **Short duration** – The pilot projects should be completed within a short time period. It is recommended that the specified time frame for any particular pilot project is carried out within three (3) months period.
- **Easy to replicate** - The pilot projects must be highly visible and have great impact and easily replicable.

2.1.5.3 Solution Areas for Pilot Projects

It is recommended that the pilot projects to be implemented should begin with the following solution areas:

A **Infrastructure Solution**

Begin implementing OSS in secondary applications at the server level (i.e. server applications, operating system, database, backup, recovery and security). This is because many of the changes done at the server side are compatible to the existing environment thus minimising the disruptive effect.

B **Desktop Solution**

Adopt OSS equivalent such as OpenOffice and Mozilla browser as alternative to Microsoft Office and Internet Explorer browser. These applications are able to operate both in Microsoft Windows and Linux operating systems (e.g. Redhat, Novell Linux Desktop, etc.).
C Application Solution

Adoption of specific application solutions to address business needs. Version Control, Portals and Content Management component can be implemented with ease and fairly quickly in order to realise most of the benefits.

2.1.6 Plan For Phased Deployment

It is highly recommended for agencies to adopt OSS in phases. Agencies planning for phased deployment approach shall consider the various implementation factors such as the complexity of the implementation environment and solution maturity. Solutions should be the least disruptive, have the ability to co-exist with other solutions, have the ease of deployment and require minimal training cost. The phases are defined as short term, medium term and long term.

i. Short term (less than 2 years)

The short-term phase focuses on OSS solutions that are mature, widely used and able to operate on existing proprietary operating systems.

ii. Medium term (2 to 5 years)

The medium-term phase focuses on prioritising OSS solution areas through the implementation of complete OSS infrastructure components and of more complex OSS solutions.

iii. Long term (Beyond 5 years)

The long-term plan aims for self-reliance through continuous improvement of OSS solutions that are highly complex and to reduce dependency on proprietary systems in all solution areas.

*Figure 2.5* provides the type of OSS Solution Areas and its related components that can be deployed over the three (3) phases.
Figure 2.5 – Type Of OSS Solution Areas
2.1.6.1 OSS Technical Implementation In The Short Term

With reference to Figure 2.5 above, the following OSS opportunities are recommended to be implemented in the short term, between the zero (0) to two (2) year timeframe:

<table>
<thead>
<tr>
<th>No</th>
<th>OSS Solution Areas</th>
<th>Application Types</th>
<th>Example of OSS Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Desktop Solutions</td>
<td>E-Mail Readers</td>
<td>Kmail, Evolution, Thunderbird, Pine</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Web Browsers</td>
<td>Mozilla, Firefox, Konqueror, Epiphany</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Office Productivity</td>
<td>OpenOffice, KOffice, Abiword, GNUMeric</td>
</tr>
<tr>
<td>2</td>
<td>Infrastructure Solutions</td>
<td>Web Servers</td>
<td>Apache</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Server Operating System</td>
<td>Open BSD, FreeBSD, NetBSD, SuSE Enterprise Edition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>e-Mail Servers</td>
<td>Sendmail, QMail</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Security Tools and DNS</td>
<td>ClamAV AntiVirus, Snort IDS</td>
</tr>
<tr>
<td>3</td>
<td>Application Solutions</td>
<td>Version control</td>
<td>GForge, Concurrent Versions System (CVS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Portal and Content Management</td>
<td>Phpnuke, Mambo</td>
</tr>
</tbody>
</table>

Table 2.2 – OSS Technical Implementation In The Short Term

2.1.6.2 OSS Technical Implementation In The Medium Term

With reference to Figure 2.5 above, the following OSS opportunities are recommended to be implemented in the mid term, between the two (2) to five (5) year timeframe:

<table>
<thead>
<tr>
<th>No</th>
<th>OSS Solution Areas</th>
<th>Application Types</th>
<th>Example of OSS Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Desktop Solutions</td>
<td>Graphic Manipulation</td>
<td>GIMP, Inkscape, Sodipodi, Dia</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Development Tools</td>
<td>Eclipse, Vim</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Client Operating System</td>
<td>Mandriva, SuSE, Ubuntu</td>
</tr>
<tr>
<td>2</td>
<td>Infrastructure Solutions</td>
<td>LDAP and databases</td>
<td>MySQL, Firebird, PostgreSQL</td>
</tr>
<tr>
<td>3</td>
<td>Application Solutions</td>
<td>Knowledge Management</td>
<td>Phpnuke, Zwiki</td>
</tr>
<tr>
<td></td>
<td></td>
<td>e-Learning</td>
<td>Moodle</td>
</tr>
<tr>
<td>4</td>
<td>Workload Consolidation OSS Solutions</td>
<td>Operating system virtualisation and monitoring tools</td>
<td>Nagios, Ethereal, Xen, Linspire</td>
</tr>
<tr>
<td>5</td>
<td>High Performance Computing OSS Solutions</td>
<td>Workload schedules and cluster manager</td>
<td>Oscar, HA-JDBC</td>
</tr>
<tr>
<td>6</td>
<td>Distributed Enterprise OSS Solutions</td>
<td>Groupware collaborative tools</td>
<td>Phpnuke, Mambo, Drupal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Search Management</td>
<td>OpenCMS</td>
</tr>
</tbody>
</table>

Table 2.3 – OSS Technical Implementation In The Medium Term
2.1.6.3 OSS Technical Implementation In The Long Term

With reference to Figure 2.5 above, the following OSS opportunity is recommended to be implemented in the long term, after the five (5) year timeframe:

<table>
<thead>
<tr>
<th>No</th>
<th>OSS Solution Areas</th>
<th>Application Types</th>
<th>Example of OSS Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application OSS Solutions</td>
<td>ERP, CRM and HR</td>
<td>Fisterra, Compiere</td>
</tr>
</tbody>
</table>

Table 2.4 – OSS Technical Implementation In The Long Term

2.1.7 Plan For Change Management

Change Management is critical to the success of any implementation. Since the use of OSS will be new to many users, it is natural for users to fear and resist change. Therefore, Change Management programmes; such as awareness and training programmes, would need to be developed in detail and undertaken, within each implementing agency. It is recommended that Public Sector agencies to consider the following strategies in implementing Change Management programme to further ensure successful adoption and implementation of OSS:

A. **Introduce OSS to enthusiastic users first**

   There will be users who are naturally more inquisitive and open towards trying something new. It is these people who should first be introduced to OSS. Through them it can be proved to others that there is not much difference between using OSS and proprietary software.

B. **Provide extensive OSS support mechanism**

   The second group of users who may be more reserved and less enthusiastic about OSS will need to have greater support facilities in the form of help desks, intranets and experienced local users.

C. **Train people who have the most knowledge of the existing systems and setup**

   The people who know the existing systems and setup may have some amount of authority over the system. They may be reluctant to give this up if the new OSS environment is perceived as very different from the existing one, thus effective change management is essential. They may need to be among the first to be trained in the new systems to maintain their relevance and minimise resistance to change.
D. Make system administrator as effective change agents within the agencies

The system administrator must be among the first to be trained in OSS. The system administrator in particular, needs to have his/her fears allayed at an early stage. They will be the focal point for occurring problems.

Public Sector agencies are also encouraged to adopt Communication Lifecycle Model AIDA (Awareness, Interest, Desire and Action) to create a greater awareness and readiness for change among the staffs.
3 OSS PROCUREMENT

The objective of OSS Procurement Guidelines is to guide Public Sector agencies in the procurement of OSS for implementation. These guidelines will facilitate Public Sector agencies to derive best value for money and all other benefits of OSS. OSS procurement should be based on merits, value for money, transparency, security and interoperability, as well as in accordance with the GOM procurement policies and procedures. In situations where advantages and disadvantages of OSS and proprietary software are equal, preference shall be given to OSS.

Public Sector agencies are required to consider the following five (5) key principles in the procurement of software and such other ICT equipment:

i. **Interoperability**

   Public Sector agencies should only use products for interoperability that support open standards and specifications in all future ICT developments.

ii. **Transparency**

   The move towards greater transparency of IT governance is central to the principle of accountability. Procurement activities of Public Sector agencies must adhere to standard GOM procurement policies and procedures and tender specifications must be free of ambiguity. Access to source code must be available wherever possible.

iii. **Security**

   Potential ICT solutions should be carefully evaluated on case by case basis prior to being accepted as safe and free from security flaws for use in GOM operations.

iv. **Value For Money**

   Public Sector agencies procurement exercises seek to avoid unnecessary public spending. Hence, procurement decisions are always to be based on the best value for money solution.

v. **Merit**

   Public Sector agencies support a level playing field between OSS and proprietary software procurement within the Public Sector. Procurement decisions are to be based on the merit.
Preference will only be given to OSS in the event that both OSS and Proprietary Software, after thorough evaluation in accordance with standard GOM procurement procedures and policies, are considered to be equal in terms of their advantages and disadvantages.

3.1 Guidelines For Procurement

Public Sector agencies are required to follow the OSS procurement guidelines during all procurement activities such as but not limited to software, hardware, IT related products and services acquisitions. The following guidelines are formulated to assist Public Sector agencies in their OSS procurement activities.

In order for the selection exercise to be conducted effectively, the following considerations need to be ensured at the tender specification stage:

- Ensure that OSS solutions are not directly or indirectly excluded in tenders whether by inclusion of any specifications that unjustifiably discriminate against OSS or otherwise.
- Whenever possible, acquire hardware that also supports OSS.
- Whenever possible, avoid lock-in to IT products and services that are not compliant with open standards.

3.1.1 Support For Open Standards

Open Standards are defined as “Specifications for systems that are publicly available and are developed by an open community and affirmed by an internationally recognised standard body”.¹

- When planning for IT investments such as but not limited to software procurement or development, Public Sector agencies must consider Open Standards compliance as a requirement.
- Open Standards relate to file formats, data formats and protocols used with respect to GOM applications and systems.
- Open Standards imply that multiple vendors can compete directly based more on the features and performance of their products.
- Open Standards would result in the GOM’s IT solutions being portable and capable of being replaced with that of another vendor with minimal effort and without major interruption.
- OSS does not necessarily mean Open Standards compliance. Compliancy to Open Standards must be carefully evaluated without any discrimination or bias as to whether the products are OSS or proprietary products.

¹ [http://www.mass.gov/Aitd/docs/policies_standards/openstandards.pdf](http://www.mass.gov/Aitd/docs/policies_standards/openstandards.pdf)
3.1.2 Access To Source Code

- Whenever possible, selected IT solutions should be available in source code form in order to allow inspection such as but not limited to examine the code to verify that it performs as expected and contains no hidden features.
- Whenever possible, selected IT solutions must come with the rights to modify and enhance the source code for various GOM operations purposes.

3.1.3 Security In IT Solutions

- Public Sector agencies should ensure maximum national security is maintained at all times with regard to the GOM’s computer systems.
- Whenever possible, selected IT solutions should have been certified by rigorous security evaluation programmes such as but not limited to Common Criteria for Information Technology Security Evaluation (CCITSE) commonly known as “Common Criteria”.
- While OSS products should not be considered more secure than propriety products, having access to view and modify the source code would eliminate a level of risk posed by potential threats in closed source code such as but not limited to back doors and inherent weaknesses that may be exploited by hackers.
- On the other hand, the OSS development process does not insure security in itself. Careful evaluation must be conducted to ensure that OSS products are safe to be used for GOM operations. If security issues are found, decisions must be made to either fix the issues identified or wholly eliminate the solution as an option.
- Vendors should be required to provide evidence of the security of their proprietary or OSS solutions by providing proof of the solutions as being validated products or in lieu thereof, providing their contractual undertaking as to the security of the same.

3.1.4 Best Value For Money

Whenever possible, Public Sector agencies must choose solutions that can provide best value for money - defined as “optimum combination of whole-life cost and fitness/ability to meet the user’s requirement”.

- Fitness is to be determined by reference to the ability of the solution to meet the requirements identified for a specific procurement, e.g. the ability to modify the solution (which will require the availability of the source code) and/or the freedom to distribute/share the solution without having to pay additional licensing fees. The requirements are to be determined on a procurement per procurement basis.
Evaluations conducted must take into consideration total cost of ownership of owning IT products and services throughout their full life-cycle including but not limited to support, maintenance, integrations, future upgrades and enhancements cost.

- Best value for money does not necessarily mean lowest price among the evaluated options.
- Best value for money does not necessarily mean market domination by the evaluated solutions.

### 3.1.5 Merit-Based Selection

Selection of IT solutions must be based on merit which covers but not limited to performance, reliability, quality, flexibility, and user friendliness.

- Performance, reliability, quality, flexibility, and user friendliness shall be among the criteria to be considered in evaluating procurement options. Reference should be made to benchmark and comparison studies done by independent international bodies or in-house team wherever possible.

### 3.1.6 Sourcing Scenarios

There are two basic sourcing scenarios for the introduction of open source software within the GOM, namely in-house and external sourcing:

- **In-house sourcing:** direct procurement of open source software, using in-house skills for sourcing and deployment
- **External sourcing:** using one or more external solution providers to deliver and deploy OSS-based solutions

There will also be instances when there is an amalgam of both these sourcing scenarios, e.g. where part of the solution identified by an agency is sourced in-house (and perhaps modified further by the agency) while some of the other parts of the said solution are sourced from external solution providers.

#### 3.1.6.1 In-House Sourcing

If an agency has the requisite skills in-house and flexible software procurement guidelines, it can obtain, install and use a substantial number of OSS products that are directly available from various online repositories. Browsing or searching these repositories allows agencies to find and download OSS products to meet their needs.
Direct in-house sourcing enables agencies to bypass formal procurement processes, purchase orders and expense requisitions; the principal costs relate to bandwidth usage and staff time. However, agencies still need to follow the standard risk assessment and change management procedures required by each agency’s policy.

Figure 3.1 below sets out a model workflow for in-house sourcing of OSS products.

The following is an overview of a sample process for the in-house sourcing of OSS (as diagrammatically represented in Figure 3.1 above) that takes into consideration the key principles in the procurement of software as addressed above:

i. **STEP 1**: Formulate agency’s baseline expectations, whether based on user requirements and/or functionality perspective and thereafter locate open source solutions that meet the agency’s baseline expectations. This can be done by reference to OSS repositories such as sourceforge.net, freshmeat.net and OSCC Knowledge Bank. Proceed to check the identified OSS against the agency’s baseline expectations and where necessary modify the baseline expectations.
ii. **STEP 2:** Check the applicable licenses for each of the identified OSS and assess the suitability of the license for the agencies intended purpose as well as the ability of the agency to comply with such license terms and conditions (Please refer to Section 4 on Ownership).

iii. **STEP 3:** Proceed to select the preferred solution based on the ability to meet the agency’s baseline expectations, suitability of the applicable license for the agency’s intended purpose and such other pertinent criteria, e.g. the availability of external support, ease of use, whether it uses a language preferred by the agency (e.g. PHP, python, Java).

iv. **STEP 4:** Review the fitness for purpose and value for money of the selected OSS (Please refer to 3.2.2 above).

v. **STEP 5:** Analyse the quality, security and interoperability of the selected OSS. In relation to quality, the agency may refer to its robustness and ability to recover from crashes, whilst in respect of security, the agency should assess whether there have been any reported hacks or other exploits in relation to the identified OSS. Agencies can also refer to third party sites, e.g. securityfocus.com for objective reports.

vi. **STEP 6:** Where considered necessary, undertake a pilot project. This is an optional step that is to be taken only where considered necessary, e.g. where the deployment of the OSS will affect many (e.g. Open Office), or where the project is complex (e.g. Sugar CRM). As opposed to this, where the OSS selected is tried and tested, e.g. Apache, there is no need to mount a pilot project.

vii. **STEP 7:** Where a pilot project is undertaken and once it has come to a conclusion, review the pilot project results against the agency’s baseline expectations. The results can be derived from such user and functionality tests carried out as well as user feedback.

viii. **STEP 8:** If everything is satisfactory, plan for production and implementation of the OSS as per the agency’s usual procedure.

ix. **STEP 9:** Deploy the production OSS, conduct training of users and obtain external support where necessary.

x. **STEP 10:** Register and upload\(^2\) the OSS (where it has been further modified or developed by the agency) with the OSCC Knowledge Bank.

---

\(^2\) All agencies are required to upload the source and object codes of any OSS solutions that have been customised, modified or developed for or by the agency UNLESS there are valid overriding national interests involved, e.g. security concerns.
3.1.6.2 External Sourcing

Agencies that do not have the in-house technical capabilities to source OSS products directly may prefer to acquire open source solutions through external service providers.

*Figure 3.2* below sets out a model workflow for external sourcing of OSS products where an agency does not have either a proprietary or open source solution in mind.

*Figure 3.2 – Workflow For External Sourcing For OSS*

The following is an overview of a sample process for the external sourcing of OSS that takes into consideration the key principles in the procurement of software as addressed above, where an agency does not have either a proprietary or open source solution in mind (as diagrammatically represented in *Figure 3.2* above):

i. **STEP 1:** The agency will need to plan and prepare the requisite tender specifications without discriminating against OSS.
ii. **STEP 2:** The procurement process is to be in accordance with standard GOM procurement procedures and policies.

iii. **STEP 3:** Evaluate bids in accordance with standard GOM procurement procedures & policies

iv. **STEP 4:** When assessing vendor proposals for OSS based solutions, the agency must assess the suitability of the applicable OSS license to its intended purposes with regard to the solution being procured.

v. **STEP 5:** Where both proprietary and OSS solutions are proposed by vendors responding to the tender, and subsequent to the agency’s thorough evaluation in accordance with standard GOM procurement procedures and policies, both are assessed to be equal in terms of their advantages and disadvantages, preference will be given to the OSS solution.

vi. **STEP 6:** Agencies should insist on the provision of suitable warranties with regard to the compliance of the OSS based solution with the agency’s specifications.

vii. **STEP 7:** Thereafter, the selected vendor will modify, develop, deliver and put its proposed OSS solution into production subject to having successfully completed the relevant user acceptance and final acceptance tests.

viii. **STEP 8:** Register and upload\(^3\) the OSS (where the OSS has been customised or modified) with the OSCC Knowledge Bank.

In situations where an agency, based on its specific requirements, has already predetermined that the solution required to be sourced must be OSS based, the agency needs to ensure that the tender specifications should specifically require vendor proposals to provide OSS or equivalent solutions.

As part of vendor due diligence, agencies should evaluate the financial strength, stability and technical capability of the short listed vendors.

Agencies may also refer to the OSCC for a list of vendors identified as being capable to provide OSS solutions and related services to agencies.

---

\(^3\) All agencies are required to upload the source and object codes of any OSS solutions that have been customised, modified or developed for or by the agency UNLESS there are valid overriding national interests involved, e.g. security concerns.
4 OWNERSHIP

The objective of OSS Ownership Guidelines is to guide Public Sector agencies to manage the licensing and ownership issues related to the modification, development and distribution of OSS in the Public Sector. This section on Ownership sets out the legal background to OSS, identifies the issues that need to be ascertained in assessing the suitability of OSS licenses to a Public Agency’s purposes, details the likely scenarios in which the licensing and ownership issues related to OSS may arise, identifies specific issues that need to be addressed with respect to the same and identifies the process to determine the suitability of an OSS license to an agency’s purposes. OSS ownership should include software licensing that allows rights to use and modify the software. Licensing for software developed within the Public Sector should be compatible with the GPL license, the BSD license or a formulated GOM license based on suitability.

These guidelines relate to the acquisition by the GOM of the legal rights to use, modify and share OSS that is deployed within the Public Sector.

When OSS is being sourced or procured for use in the Public Sector, agencies will need to confirm their ability to comply with the applicable OSS license terms and conditions bearing in mind the agencies intended use of the OSS.

Where agencies are modifying or developing OSS, agencies should consider (where applicable) the most suitable license to apply to the same, whether it is a license compatible with the GPL license, the BSD license, a GOM license or such other OSS preferred license.

4.1 Guidelines For Ownership

This Ownership section must be read in conjunction with the Procurement section and such applicable GOM circulars and guidelines in relation to the acquisition and ownership of intellectual property by and/or for the GOM.

As users of this document will already be familiar with the proprietary software model and its related legal and licensing issues, this section principally focuses on the licensing and ownership issues related to OSS.

4.1.1 Understanding The Legal Background To OSS

In general, OSS refers to software that is licensed to users under certain minimum terms. Pursuant to these terms, users have the right to use, copy, access, modify and distribute the software (whether in original or modified form), without the payment of any royalty or other significant fee.
The alternative to OSS is proprietary software which generally restricts the right to use to the number of “seats” paid for, prohibits copying, modifying and distributing of the software and does not provide access to the source code of the software.

Table 4.1 briefly describes the key differences between proprietary software and OSS.

<table>
<thead>
<tr>
<th></th>
<th>Proprietary Software</th>
<th>Open Source Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intellectual Property Rights</td>
<td>Based on copyright</td>
<td>Based on copyright</td>
</tr>
<tr>
<td>Licensed to users</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Cost</td>
<td>High (i.e. licensing fees)</td>
<td>Low / Nil (i.e. usually no licensing fees)</td>
</tr>
<tr>
<td>Source code</td>
<td>Unavailable</td>
<td>Available</td>
</tr>
<tr>
<td>Distribution</td>
<td>Restricted</td>
<td>Permitted (sometimes with restrictions)</td>
</tr>
<tr>
<td>Modification</td>
<td>Restricted</td>
<td>Permitted</td>
</tr>
<tr>
<td>Warranty</td>
<td>Limited</td>
<td>Limited / Nil</td>
</tr>
<tr>
<td>Indemnity</td>
<td>Limited</td>
<td>Limited / Nil</td>
</tr>
<tr>
<td>Support</td>
<td>Limited</td>
<td>Limited / Nil</td>
</tr>
</tbody>
</table>

**Table 4.1 – Key Differences Between Proprietary Software And OSS**

4.1.2 *Minimum Standards As Set Down By The OSI*

For software to be considered OSS, the license governing the use of that software must be recognised by the Open Source Initiative (“OSI”) as being one of its approved licenses. Please refer to http://www.opensource.org/licenses for a list of the currently approved OSS licenses.

In order for a software license to be approved by the OSI, the said software license must comply with the minimum licensing terms set down by the OSI as embodied in its “Open Source Definition” that can be found at http://www.opensource.org/docs/definition.php

In brief, in order for a software to be considered OSS, the license under which the software is distributed must embody the following rights and obligations. The diagram below illustrates the minimum set of criteria required of OSS licenses and by extension the software that is distributed under the said licenses (see figure 4.1).
The minimum licensing terms set by the OSI*

- Free distribution of the software by users without need to pay the proprietor any royalty or fee
- Source code is made available to the user
- Modifications and creation of derivative works by the user are allowed
- Integrity of source code, i.e. proprietor may require the user to distribute the modified version as the original version plus patches
- No Discrimination against persons/groups who may or may not use the software
- No Discrimination against specific fields of endeavour i.e. can be used for any purpose
- The rights granted to the software must not require the execution of an additional license.
- The license must not restrict other software that it is distributed with
- License must not be technology specific i.e. be technology neutral

* Please note that the list above is not exhaustive and has been abbreviated for your ease of reference. Please refer to OSI’s “Open Source Definition” available at http://www.opensource.org/docs/definition.php for more detail.

**Figure 4.1 – Minimum Licensing Terms Set By The OSI**

Some of the criteria are absolute (e.g. availability of source code, right to modify and create derivative works, the right to redistribute without having to pay royalties) while others are optional (e.g. requiring the software to be distributed as the original version plus patches, requiring the same license be used to distribute modified and derived works).

However there is room for variance in the terms of an OSS license, wherein licensors of OSS are in fact free to impose other terms and conditions upon users of their software so long as they do not conflict with the above said minimum criteria.
4.1.3 **OSS Licenses In General**

A. There are numerous types of OSS licenses\(^\text{4}\) but they basically fall into two (2) categories:

i. **Academic Licenses**

   Generally, academic licenses allow software to be used for any purpose and do not impose an obligation on the user to provide the source code of the software (whether in original, modified or derivative form) when distributing it further. Academic licenses also do not specify the type of license that must govern any further distribution of the software.

   Academic licenses allow anyone to take the software and use it for whatever purpose, whether commercial or otherwise (e.g. modify it and offer it as a proprietary product), without the need to make available the source code to either the original authors of the software, the open source community or even the persons to whom it is redistributed to.

ii. **Reciprocal Licenses**

   Generally, reciprocal licenses allow the software to be used for any purpose, so long as (i) the source code of the said software is made available to the persons it is redistributed to, and (ii) redistribution of the software (whether in original, modified or derivative form) is made under the same license the software was originally distributed under.

   OSS licenses such as the General Public License (“GPL”), Lesser General Public License (“LGPL”) and MIT License (“MIT”), which fall within this category, grant free rights to users to use, copy or modify without payment or restriction. These licenses however prohibit users from distributing the software on any terms other than the original license, and impose this requirement on any programme derived from or based in whole or part on the software.

B. Some software developers offer the same software under more than one license. For example, a software developer may offer its software under a proprietary license (that requires the payment of a license fee) and also under an OSS license (which is usually free). This practice is known as “dual licensing”. The main difference between the two offerings in the prior example is that under the proprietary license, users will be provided support and warranties, whereas neither will be provided for the same software distributed under the OSS license.

\(^{4}\) Currently there are around sixty nine (69) different forms of OSS which have been OSI certified.
4.1.4 Implementing/Using OSS Licenses In The GOM

A. Reasons For Having Such Assessment

As part of the implementation and use of any OSS within the GOM, an assessment of the applicable OSS license(s) must first be conducted in order to decide whether the OSS license(s) in question are indeed suitable for the GOM’s purposes. It is therefore highly advisable that agencies must familiarise themselves at the outset with the terms and conditions of the major OSS licenses and the legal issues relating to the choice of OSS license.

Failure to carry out a proper assessment of the licenses of the OSS chosen may result in adverse consequences being faced by such agencies, e.g.:

i. Breach of the OSS licenses

In the event of non-compliance with or breach of the applicable OSS licensing terms, the agency may lose the right to use such OSS unless and until such breach is rectified.

ii. Different and incompatible OSS licenses

Where several different OSS modules/codes/libraries falling under different OSS licenses are combined and used together, there is a risk that the said OSS licenses contain terms that make them incompatible with one another and therefore unsuitable to be combined and used together (e.g. GPL and Sun licenses). Not all OSS licenses can be adapted and combined without restriction in order to produce or modify software.

iii. Loss of GOM Intellectual Property

Where an agency commissions modifications to an OSS with the intention of keeping such modifications private (i.e. not providing the source code to such parties that the modified OSS may be distributed to), the agency may be forced to provide the source code of modifications made to the said OSS as a consequence of having chosen an OSS falling under a reciprocal license instead of an academic license.

iv. Breach of GOM’s secrets and confidentiality

There may also be situations where there is a need to maintain the GOM’s secrecy and confidentiality which is in conflict with the disclosure requirement under a Reciprocal License. An example of such an instance would be where the source code of an OSS modified by an agency discloses attributes and/or processes related to the GOM that are in fact confidential, but must be revealed to users when distributed under a Reciprocal License (as opposed to an Academic License).
4.1.5 A Summary Of Common And Relevant OSS Licenses

It is important to understand that each OSS license varies in terms of its restrictions and implications. In any OSS use scenario, it is therefore necessary to study the relevant OSS license carefully before using or agreeing to its licensing terms. For general reference purposes a summary of the more common and relevant OSS Licenses to the GOM, is provided below:

A. General Public License (GPL)

The GPL emphasises freedom wherein users are permitted to copy and distribute verbatim copies of the licensed software, and are required to ensure that any derivative products remain free and open for public use. Users are also permitted to examine the source code and improve the software. However if there is redistribution, the improvements and derivative works must be released back to the public in source code form and under the terms of the GPL. For example, after releasing a code as GPL, the GPL can never be removed and any derived work from the GPL must also be GPL.

Since the GPL does not allow users to take improvements and derivative works private, the GPL does not permit the incorporation of GPL software into proprietary software.

An example of software available under the GPL is Linux Kernel. The software can be found at http://www.kernel.org

Please refer to http://www.opensource.org/licenses for the full text of the GPL.

B. Lesser General Public License (LGPL)

The LGPL is a derivative of the GPL and is therefore quite identical to the GPL in many of its provisions. The LGPL was designed for software libraries. However unlike the GPL, LGPL software can be incorporated into proprietary software.

An example of software available under the LGPL is Bonita J2EE Cooperative Workflow System. The software can be found at http://bonita.objectweb.org

Please refer to http://www.opensource.org/licenses for the full text of the LGPL.

C. Berkeley Software Distribution License (BSD)

The BSD is intended to permit the free use, modification and distribution of the University of California software without obligation on the users. This license requires users to put a copyright notice, comply with the conditions (such as the name of authors/
owners must not be used in endorsing products and/or promotions without permission) and the disclaimer of liability.

Basically the BSD allows a person to do absolutely anything with the code (including using it with proprietary software) as long as the above conditions are fulfilled. The BSD is one of the least restraining OSS licenses.

An example of software available under a BSD license is FreeBSD Operating System. The software can be found at http://www.freebsd.org

Please refer to http://www.opensource.org/licenses for the full text of the BSD license.

D. MIT License (MIT)

The MIT license is also known as the X or X11. This license permits redistribution subject to the publication of copyright notice and permission notice (to use, modify, merge, publish, distribute, sub-license, sell etc.).

An example of software available under a MIT license is PuTTY, a free telnet/SSH client. The software can be found at http://www.chiark.greenend.org.uk/~sgtatham/putty

Please refer to http://www.opensource.org/licenses for the full text of the MIT license.

E. Mozilla Public License 1.1 (MPL)

The Mozilla Public License, though similar to the GPL, is often seen as being the middle-ground between the GPL and the BSD. The MPL is subject to the following conditions:

i. all distributed copies (original or modified) must include the source code or advice on how to obtain the source code

ii. all modifications are described in accompanying documentation

iii. any patent rights necessary to operate the software are clearly described in accompanying documentation

iv. all copies of the code, original or modified, have a statement of copyright and an exclusion of warranties attached

v. all modified code must be distributed under the MPL

An example of software available under the MPL is Mozilla Firefox Web Browser. The software can be found at http://www.mozilla.com/firefox
Please refer to http://www.opensource.org/licenses for the full text of the MPL.

F. Apache Software License (Apache)

The Apache license is very different from the GPL and LGPL. This license allows users to do nearly anything with the software licensed under it. The Apache permit users to copy, modify and distribute the covered software in source and/or binary forms provided that all copies, modified or unmodified, are accompanied by a copy of the license.

An example of software available under the Apache license is Apache HTTPD Web Server. The software can be found at http://httpd.apache.org

Please refer to http://www.opensource.org/licenses for the full text of the Apache license.

G. Eclipse Public License (EPL)

The Eclipse Public License is an Open Source Initiative (OSI) approved license that is similar to the Common Public License. Its primary use is for Eclipse software development tools. The Eclipse Public License is not compatible with the GNU General Public License because it has various specific requirements that are not in the GPL.

An example of software available under the EPL is AspectJ Development Tools (AJDT). The software can be found at http://www.eclipse.org/ajdt

Please refer to http://www.opensource.org/licenses for the full text of the Eclipse Public License.

H. XFree86 License (XFree86)

Under this license, the code must be redistributable by users in either source or binary form. Users must be permitted to modify the code, and to distribute modified code or derived code without being required to pay a licensing fee or other financial consideration to the copyright holder. Users must be permitted to distribute binary-only forms of the code if they wish to do so.

An example of software available under the XFree86 license is X Window System. The software can be found at http://www.x.org

Please refer to http://www.opensource.org/licenses for the full text of the X license.
I. Sun Public License Version 1.0 ("Sun License")

Sun Microsystems has introduced the Sun Public License Version 1.0 that is a "Community Source License Agreement". The Sun License contains significant restrictions that make it substantially different from the "classic" open source licenses such as the GPL and BSD-style licenses. The Sun License is a GPL-incompatible license.

The Sun License in some instances requires the licensee to pay Sun a fee. The Sun License also contains restrictions on modifications that do not pass a large set of conformance tests, and purports to treat the source code as "confidential information," even though it is downloadable from the Internet.

An example of software available under the Sun license is SLAMD Distributed Load Generation Engine. The software can be found at http://www.slamd.com

Please refer to http://www.opensource.org/licenses for the full text of the Sun license.

J. The Artistic License

The Artistic license is generally less restrictive than the GPL. However the Artistic license does not allow sale of the software, but allows a software/programme to be added to another software/programme and to be resold as a bundle, i.e. an aggregate software distribution of more than one programme to be sold. The Artistic license also requires that modified software be free, but allows additions to be made to the original software and the addition together with the original software to be taken “private” or allows parts of the Artistic-licensed programme to be placed in the public domain.

Most Artistic-licensed software is now dual-licensed, i.e. offering the choice of the Artistic license or the GPL.

An example of software available under the Artistic license is Perl. The software can be found at http://www.perl.com

Please refer to http://www.opensource.org/licenses for the full text of the Artistic license.

4.1.6 Identify The Salient Considerations In Specific OSS Licensing Scenarios

It should be noted that the following section will seek to identify the salient provisions of an OSS license being considered for use by an agency. However, there may be additional terms and conditions that will also need to be taken into consideration when assessing the suitability/applicability of a particular OSS license.
The salient features of an OSS license can be generally classified in the following categories:

i. Copying and distribution;
ii. Modification;
iii. Licensing; and
iv. Warranties

4.1.6.1 Copying And Distribution

In determining the most acceptable OSS license for an agency’s project, the agency will need to first take into account whether there will be copying and redistribution of the particular OSS.

Below are some of the questions that an agency should ask in relation to the OSS license and its provisions relating to the issue of copying and redistribution:

i. Does it allow free copying and redistribution?

ii. Does it allow a fee to be charged for the physical transfer of the software?

iii. Does it require a copyright notice to be attached (when redistributing)?

iv. Does it require that all notices referring to the license to be kept intact (when redistributing)?

v. Does it require the complete source code or a written offer to provide complete source code at nominal cost be included (when redistributing)?

In addressing/answering the above questions, an agency should start with the end in mind. It is important to decide whether there will be redistribution of the OSS or not, as the strict requirement of complying with a particular OSS license (such as the GPL) is triggered when there is redistribution of the software.
The following table may be referred to for guidance in deciding on the OSS license most suited to an agency’s intentions and needs.

<table>
<thead>
<tr>
<th>COPYING &amp; DISTRIBUTION</th>
<th>GNU GPL</th>
<th>GNU Library GPL (LGPL)</th>
<th>Mozilla Public License 1.1</th>
<th>BSD License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can freely copy and distribute</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Can charge royalty for copying or distribution</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Can charge a fee for physical transfer of software</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Must attach a copyright notice</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Must keep intact all notices referring to license</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Must include complete source code</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 4.2 – Guidance In Deciding On The OSS License

If there is to be redistribution to other government agencies and/or third parties and the agency does not want to reveal or provide the source code, the agency should avoid OSS that are governed by licenses that are reciprocal in nature and instead opt for OSS that are governed by academic licenses such as the BSD license.

It should be noted that distribution to other federal government agencies does not qualify as redistribution, as all such federal agencies are part of the legal entity that is the Government of Malaysia. However, should the distribution be from a federal agency to a state government or entity, it would likely be deemed as redistribution. Thus and in the case of reciprocal OSS licenses, the said OSS that is being redistributed will need to be provided under the terms of the original OSS license and the source code of any modifications must be provided in any further redistribution.

The following table provides a simplified reference point as to whether sharing between various GOM related bodies amounts to redistribution or not. However, it is advised that in major OSS acquisition scenarios, the confirmation of internal legal counsel is obtained.
4.1.6.2 Modification

In determining the most acceptable OSS license for an agency’s project, the agency will also need to take into account whether there will be modification of the particular OSS.

Below are some of the questions that an agency should ask in relation to the OSS license and its provisions relating to the issue of modification:

i. Does it allow modification of the software or parts of the software to form a new work?

ii. Does it allow the free copying and distribution of the modified version of this software?

iii. Does it allow a royalty to be charged for distribution of the modified software?

iv. Does it require that the source code of any modifications must be included if the modified software is distributed?

v. Does it require that interactive programmes must display notices regarding warranty, copyright information, redistribution information, and how to view the license?
In so far as modification of OSS is concerned, all OSS licenses permit the same. Modification only becomes an issue where there is redistribution of the modified OSS as some OSS licenses require the source code of the modified OSS to be provided in any redistribution and may additionally require that the modified OSS is licensed under the initial OSS license. This is typically the case with reciprocal licenses such as the GPL.

The following table provides a guideline as to whether there can be modification and if so, it shows the requirements imposed on such modified OSS application.

<table>
<thead>
<tr>
<th>MODIFICATION</th>
<th>GNU GPL</th>
<th>GNU Library GPL (LGPL)</th>
<th>Mozilla Public License 1.1</th>
<th>BSD License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can modify the program or use it to form a new work</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Can freely copy and distribute the modified version of the software</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Can charge royalty for distribution of modified software</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Must prominently include notice and date of modification on the modified version</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Must include source code of any modifications if distributed</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

**Table 4.4 – Modification Guideline**

4.1.6.3 Licensing

Upon the clarification of the first two salient features of OSS licenses (i.e. redistribution and modification) the agency would have a fair view of the OSS licenses (and consequently the OSS) that are suitable to their intentions and purposes.

Thereafter, the agency should consider the other licensing requirements imposed by the applicable OSS licenses and assess their suitability and appropriateness to the agency, especially with regard to the issue of whether the OSS license requires the modified OSS to be licensed under the original license that it was first distributed under.

Below are some of the important questions that an agency should consider in relation to the OSS license and its provisions in relation to specific licensing requirements:

i. Does it require any OSS or modified work that is distributed, to be strictly licensed under the terms of the original license?

ii. Does it allow the imposition of new license restrictions on distributed/modified copies?

The following table provides guidance in relation to the specific licensing requirements of the identified OSS licenses.
4.1.6.4 Warranties And Indemnities

In general, most OSS does not come with any warranties and/or indemnities, and is usually provided “as is”. The primary reason for this is because most OSS is free and it would therefore be unreasonable to expect any warranties and/or indemnities to be given by the authors and/or distributors of the same.

Below are some of the questions that an agency should ask in relation to the OSS license and the warranties and indemnities available (if at all):

i. Is it provided “AS IS”, i.e. no express or implied warranties?
ii. Does it disclaim liability for damage caused by the software?
iii. Does it permit distributors of the software to provide a warranty in exchange for a fee?
iv. Does it require distributors of the software to include warranty disclaimer?

*Please note that the above is not exhaustive as there may be additional terms and conditions that need to be taken into consideration when assessing the suitability/applicability of a particular OSS.

<table>
<thead>
<tr>
<th>Licensing Requirement</th>
<th>GNU GPL</th>
<th>GNU Library GPL (LGPL)</th>
<th>Mozilla Public License 1.1</th>
<th>BSD License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Must license any modified work distributed under the terms of the original license</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Can impose new license restrictions on distributed/modified copies</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Un Specified</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Warranty Requirement</th>
<th>GNU GPL</th>
<th>GNU Library GPL (LGPL)</th>
<th>Mozilla Public License 1.1</th>
<th>BSD License</th>
<th>MIT License</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provided “AS IS” — No express or implied warranties</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No liability for damage caused by software</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Distributors can provide a warranty in exchange for a fee</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Un specified</td>
<td>Un specified</td>
</tr>
<tr>
<td>Distributors must include warranty disclaimer</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table 4.5 – Guidance In Relation To Specific Licensing Requirements Of Identified OSS Licenses*

4.1.7 Likely OSS Acquisition, Modification & Distribution Scenarios

As can be seen from above, OSS licenses can differ significantly from one another. As such, prior to procuring OSS, the provisions of the applicable OSS license needs to be assessed to its suitability for the agency’s purpose and intent.
Some of the more likely scenarios to be faced by agencies in relation to the acquisition, modification, development and distribution of OSS within the GOM have been identified hereunder, together with illustrations as to the choice of appropriate license for specific purposes.

4.1.7.1 Merely Using Existing OSS Applications/Tools

**Example:** Using Open Office for the generation of documents, spreadsheets and presentations.

**Issues:** None. The mere use of an OSS application such as Open Office (that is under the LGPL) has no impact whatsoever on the output of the same, e.g. the documents, spreadsheets and presentations, and the GOM continues to have full rights in such output.

4.1.7.2 Using Existing OSS Code In The GOM’s Application

**Example:** Where the source code from an OSS module relating to the performance of an authentication function is extracted from the said OSS module and included within the body of a GOM application.

**Issues:** Where the source code is extracted from an OSS module that is governed by a reciprocal license (e.g. GPL) and included in an application owned by the GOM which is then redistributed (e.g. shared with a State Government), the modified GOM application will be deemed to fall under the reciprocal license. This would entail the release of the source code of the modified GOM application to whichever party it is redistributed to. One way of avoiding such an outcome is to only use source code from OSS that is governed by academic type licenses (e.g. BSD) that allow code being taken “private” so long as the minimal requirements specified are adhered to.

4.1.7.3 Modifying Existing OSS Applications/Tools For Internal Use i.e. No Redistribution

**Example:** Where an open source security tool (e.g. NMap, SNORT) is modified by an agency for its own use (i.e. it is not redistributed or shared with another party).

**Issues:** None. Even if the said open source security tool is governed by a reciprocal license (e.g. the GPL), as long as there is no redistribution, the source code of the modifications does not need to be provided to be contributed back to the source from which the open source security tool was obtained.

4.1.7.4 Modifying Existing OSS Applications/Tools And Distributing/Sharing The Modified OSS Applications/Tools Amongst Different Legal Entities In The GOM

**Example:** Where an open source security tool (e.g. NMap, SNORT) is modified by a federal agency and thereafter redistributed or shared with local authorities.
4.1.7.5 Commissioning Modifications To OSS Applications For The GOM’s Use

Example: Where a GOM ministry commissions a third party to modify an existing OSS application for the management of schools (e.g. the Learning Management System (“LMS”)). Upon delivery of the modified OSS application, the GOM ministry makes the modified OSS application available to all schools and public universities.

Issues: First, as the modifications were effected via third party developers, the ownership of the modifications needs to be ascertained (i.e. whether the copyright in the modifications is that of the GOM or of the third party developers). The default position at law, is that copyright vests in the commissioner of the work unless expressly agreed between the parties otherwise. In this scenario, it is presumed that ownership vests in the GOM. Secondly, where the said OSS application is governed by a reciprocal license (e.g. the GPL), as there has been redistribution (i.e. public universities are considered distinct entities from the GOM), the source code of the modified OSS application will need to be provided in any redistribution of the same. Further, if the reciprocal license requires that any further distribution be under the original license (e.g. the GPL) governing the OSS application, the redistribution by the GOM ministry of the modified OSS application will also have to be under that same license (i.e. the GPL). However, where the said open source security tool is governed by an academic license (e.g. the BSD license), the source code does not need to be provided by the federal agency in any redistribution and further, the federal agency may license it on such terms as it wishes to the local authorities and even charge fees for the same.

4.1.7.6 Making Contributions To OSS Projects

Example: An employee of the GOM participates in an open source project and assigns the copyright in his/her contributions to the project owner.

Issues: Any works developed by an employee of the GOM belong to the GOM. The issue here would be whether the GOM employee has obtained authorisation to participate, make contributions and transfer the intellectual property rights of the GOM in his works to the project? Failing such
authorisation, any contributions to the project will be in infringement of the GOM’s intellectual property rights and in breach of the said employee’s terms of service.

### 4.1.7.7 Linking

**Example:** Where the proprietary code in an application developed by an agency links to a library falling under a reciprocal license. The said application is then redistributed or shared with third parties.

**Issues:** Where proprietary code in an application developed by the GOM links (whether statically or dynamically) to a library that is governed by a reciprocal license (e.g. GPL) and is then redistributed (e.g. shared with a State Government), it is arguable whether the application itself will be deemed to fall under the reciprocal license. There are varying views on this issue that have yet to be determined by a court of law. If it does indeed result in the application itself falling under the said reciprocal license, this would require the said agency to release the source code of the said GOM application to whichever party it is redistributed to. One way of avoiding such an outcome is not to link in any way to software governed by reciprocal licenses unless legal clearance has been obtained.

### 4.1.8 Ownership

Ownership in any modifications made to OSS will vest in the author of the modifications. Where the author is an employee or where the author has been commissioned to make the modifications, ownership will be with the employer/commissioner of the work, unless specifically agreed otherwise.

In commissioning third party developers to make modifications to OSS, it should be ensured that ownership of the modifications effected vests in the GOM and is supported by the usual warranties and indemnities in respect of the functionality, fitness for purpose and non-infringement of third party intellectual property rights of the said modifications and where appropriate, the modified OSS in entirety.
5 TECHNOLOGY

The objective of OSS Technology Guidelines is to ensure that OSS technology used comply with worldwide open standards. The technology acquired should be able to be supported by any other party to ensure continuity of support.

Public Sector agencies are required to consider open standards. Open standards promote interoperability between software, computer systems, and also between machines that run them. Public Sector agencies that deploy open standards in its ICT infrastructure solutions are able to take advantage of better and more effective intra and inter agency communication which would result in increased productivity and efficiency.

Public Sector agencies implementing OSS should ensure that the technical specifications for the technology conform to internationally recognised standards to ensure continuity of support.

5.1 Guidelines For Technology

Public Sector agencies are required to follow the following guidelines in order to adhere to the above mentioned objective.

5.1.1 Technology Recommendations

Public Sector agencies must ensure that all OSS related technical specifications for every OSS initiatives implemented must conform to the internationally recognised OSS technology standards. It is recommended that the agencies adopt the following recommendations:

i. It is recommended that hardware servers are able to run on multiple platforms

ii. It is recommended that hardware peripherals (such as printer, scanner, tape backup, external storage) are compatible to OSS based operating system

iii. It is recommended that software is able to run on OSS based operating system

iv. It is recommended that application software is able to support OSS based database software

5.1.2 Adhere To Standards Defined In MyGIFOSS

Public Sector agencies should adhere to the recommended technology standards for OSS implementations contained in MyGIFOSS. MyGIFOSS provides recommendations on the following:

- OSS that is compatible and comparable to commonly used technology, including, brief description, rationale and limitation of each software.
Information access standards and specifications to enable interoperability and access to Public Sector information, including, brief description, rationale and limitations of each standard.

Application notes and guidelines that address OSS technology migration and co-existence with proprietary systems.

*MyGIFOSS* is an evolving document and will be regularly updated to take into account new developments and requirements that are suitable for adoption. Public Sector agencies are advised to always refer to the most current *MyGIFOSS* version when implementing software. Agencies may also propose updates or additions to the *MyGIFOSS* document through the OSCC. *MyGIFOSS* document is available and can be downloaded at [http://opensource.mampu.gov.my](http://opensource.mampu.gov.my)

### 5.1.3 Ensure Continuity of Support

Public Sector agencies are required to set-up a support mechanism to enable users to convey their issues, provide feedback and give suggestions regarding the implemented OSS solution. There are several methods of support:

I  **Refer to In-house Support**

   An agency may opt to in-source the procurement and support of an OSS solution. This method requires the in-house technical staffs to have the necessary skills to deploy and manage the solution on a daily basis.

II  **Refer to OSCC for Advice**

   Public Sector agencies can also seek advice and support from the OSCC on issues arising from OSS implementation.

III  **Refer to Web Support**

   Web support includes frequently asked question, forums, OSS online community resources and email support.

IV  **Engage External Support Vendor**

   In the absence of any existing OSS expertise within the implementing agency, engaging an external vendor to provide support for the OSS is recommended. The external support vendor should provide first level support, second level support and undertakes the necessary collaboration and correspondence with the developer community that originally created the software.
6 IMPLEMENTATION

The objective of OSS Implementation Guidelines is to ensure successful implementation of open source software in the Public Sector. The guidelines also ensure consistency in development of OSS solutions, interoperability and support. OSS implementation should be based on guidelines specified in the Malaysian Public Sector OSS Technical Implementation Plan. The implementation should leverage on the shared knowledge in the Knowledge Bank and with minimal impact to the day-to-day business operations.

- **OSS Technical Implementation Plan**

  Public Sector agencies are required to follow the OSS Technical Implementation Plan in implementing OSS. The plan provides a broad guide on the implementation of ongoing and future OSS projects.

  The document is available for the public in the Knowledge Bank at [http://oscc.org.my](http://oscc.org.my)

- **OSS Knowledge Bank**

  In addition to the OSS Technical Implementation Plan, the Public Sector agencies shall refer to the OSS Knowledge Bank at [http://oscc.org.my](http://oscc.org.my) to leverage on OSS solutions that are already implemented in Government Agencies. Knowledge Bank is a repository aimed to harness and share OSS knowledge, application development knowledge and support among Government Agencies.

6.1 Guidelines For Implementation

Public Sector agencies are required to follow the following guidelines in order to have a more systematic approach in implementing OSS in their respective agencies.

6.1.1 Refer To Technical Implementation Plan, Knowledge Bank And MyGIFOSS

During implementation of OSS, Public Sector agencies need to refer to the Technical Implementation Plan, Knowledge Bank and MyGIFOSS.

Technical Implementation Plan details OSS solution areas that may be implemented over the short, medium and long term. Public Sector agencies can also refer to Knowledge Bank on the availability of similar OSS solutions that have been developed as well as MyGIFOSS that provides OSS technology standards, information access and application notes for guidance in usage and migration. These references provide essential guide for Public Sector agencies to successfully implement OSS.
6.1.2 Implement OSS Within The Six (6) Solution Areas In Phases

With reference to Section 2 on Adoption, all Public Sector agencies are encouraged to implement OSS within the six (6) solution areas in phases; short term, medium term and long term. Refer to Figure 6.1 below:

Figure 6.1 – OSS Technical Implementation Plan For The Public Sector
7 KNOWLEDGE SHARING

The objective of OSS Knowledge Sharing Guidelines is to facilitate and encourage sharing of Open Source Software experience, expertise and resources within the Public Sector. Public Sector agencies implementing OSS must register OSS initiatives, related information and knowledge in the **OSCC Knowledge Bank** at [http://oscc.org.my](http://oscc.org.my). The Knowledge Bank shall serve as a platform for sharing knowledge and experience.

The OSCC Knowledge Bank serves as a medium of sharing OSS knowledge and also as a guide for implementing OSS in the Public Sector. It provides Public Sector agencies with a one stop portal where they can interact and exchange ideas. The resources and features available in the OSCC Knowledge Bank would allow the Public Sector agencies to perform the following functions:

- Register their OSS initiatives at the Knowledge Bank
- Download solutions and information related to OSS
- Request for technical support and OSS solution updates
- Provide feedback to OSCC for continuous improvement of OSS skills, knowledge and solutions
- Engage in online discussion on OSS related issues

7.1 Guidelines For Knowledge Sharing

The establishment of the OSCC Knowledge Bank is intended to cultivate a knowledge sharing culture amongst Public Sector agencies in respect of OSS. These guidelines provide guidance on how to share OSS knowledge by leveraging on the functions, resources and information provided by the OSCC Knowledge Bank.

This section is to be read in conjunction with the Ownership section and the **OSS Knowledge Bank User Manual**. The manual is available for download in the Knowledge Bank at [http://www.oscc.org.my/docs/manual.html](http://www.oscc.org.my/docs/manual.html)

7.1.1 Register All OSS Solutions and Projects In The OSCC Knowledge Bank

All Public Sector agencies are required to register the OSS solutions that have been developed or modified into the OSCC Knowledge Bank.

All Public Sector agencies are also required to upload the OSS solutions that have been developed or modified into the OSCC Knowledge Bank unless there are overriding reasons for not doing so, e.g. security or investment considerations.

Any subsequent modifications to the OSS solutions so registered must also be registered and updated in the OSCC Knowledge Bank upon its production.
Where the original OSS was acquired from external parties and further modified, agencies need to take the following precaution prior to uploading the modified OSS into the OSCC Knowledge Bank:

- Agency must confirm the contractual terms governing the acquisition of the original OSS from the said external party, particularly with respect to whether the modifications to the original OSS are the property of the agency. Please note that the modified OSS may only be uploaded into the OSCC Knowledge Bank where the modifications to the OSS are the property of the agency.

7.1.2 Open Access To All Knowledge And Resources

All information and resources related to the development or modification of OSS solutions registered in the OSCC Knowledge Bank (other than the OSS solution itself) must be uploaded and made available in the OSCC Knowledge Bank. This is necessary in order to allow other developers to view, understand and further develop existing or new OSS solutions.

7.1.3 Share All Modified OSS

All users duly registered in the OSCC Knowledge Bank may register, copy, modify and distribute the OSS in the OSCC Knowledge Bank within the Public Sector agencies, subject to such restrictions indicated in the OSCC Knowledge Bank.

7.1.4 Report All Bugs

The Knowledge Bank provides a Tracker function that stores information on bugs, feature requests, patch submission and other relevant information. When a project is created, the Knowledge Bank will create four (4) types of trackers:

- Bugs: Used for bug tracking
- Support Requests: Users can insert support request and receive support
- Patches: Developers can upload patches to the application
- Feature Requests: Requests for enhancements of the applications

In the event any user finds a problem with OSS applications registered in the OSCC Knowledge Bank, the user must report and enter the findings in the Tracker section within the Knowledge Bank.

Please refer to OSS Knowledge Bank User Manual for further guide on Tracker usage.
7.1.5 Communicate And Share Information

A E-mail

The most effective and efficient communication tool to share OSS knowledge is through e-mail. It is highly recommended for Public Sector personnel to join the OSS mailing list established in the OSCC Knowledge Bank and other OSS communities.

B OSCC Knowledge Bank Forum and Newsgroup

Forum and newsgroup are created in the Knowledge Bank for users to exchange thoughts and discuss topics related to OSS applications. All users are encouraged to capitalise on these facilities made available to everyone.

C OSS Global Online Communities

There is a number of OSS community sites on the web that offer recommendations, bugs fixes and applications that Public Sector agencies’ personnel can leverage upon to facilitate knowledge sharing.
8 OSS EDUCATION

The objective of OSS Education Guidelines is to ensure continuous supply of and development of competent human capital in OSS, hence ensuring continuous support and sustainability of the OSS initiative in the Public Sector. Open Source Software education should be introduced through structured programmes for IT labs in primary, secondary and tertiary education levels.

The guidelines targeting IT labs at primary, secondary and tertiary education levels, is meant to introduce and promote OSS at an early stage to ensure continuous supply of quality human resource. Structured adoption of OSS in the education system can be promoted by:

- Introducing OSS training for IT labs teachers and relevant administrative personnel
- Promoting OSS tools and applications in structured education programmes in all IT labs
- Promoting the use of OSS in learning, teaching and administrative IT infrastructure

8.1 Guidelines For OSS Education

The following guidelines need to be adhered to by the relevant agencies.

8.1.1 Proliferate OSS in IT Labs

Learning institutions shall use OSS as part of teaching and learning tools. Most software currently used in teaching IT at the IT labs, including basic productivity software for teaching computer literacy, compilers for programming courses and relational database management system, are proprietary. However, there are OSS equivalents available that can be suitable replacements.

OSS can be introduced in the six (6) IT knowledge fields within the current school syllabus at the IT labs as follows:

I  Computer Systems

Students should be introduced to open source operating systems such as SuSe, Slackware, Mandrake, Fedora, Debian, Open BSD and Mandriva.

II  Software Applications

Open source educational software should be used for teaching specific subjects or courses in schools, colleges and universities. Table 8.1 represent only a very small sample of OSS available for education. Please refer to MyGIFOSS for other various useful OSS resources.
### III Multimedia

All IT labs in learning institutions are encouraged to use multimedia OSS to enhance their educational content and its delivery. A wide range of multimedia OSS is available, including graphics editors and video players that can serve as tools, such as Mplayer and Xine.

### IV Programming

- Open source programming and development tool should be used in programming subjects and courses.

- There are numerous computer languages available on the Linux platform that can be used for this purpose. The GNU Compiler Collection (GCC) is a collection of programming language compilers that are included in most Linux distributions. It currently supports computer languages such as C, C++ and Java.

### V Information System Management

- Students should be introduced to the concept of information system and hands-on application of simple databases.

- The OSS database systems such as MySQL and PostgreSQL are full-featured and can be used to teach the basics of database systems.

### VI Network and Communication

- Learning institutions should fully utilise the flexibility and effectiveness of networking and communicating using OSS. Open source offers rich and industry standard Internet communication tools such as remote access software, email servers and clients, web browsers and website editors.
There are a number of Open Source browsers available such as Firefox, Mozilla and Konqueror. Apart from a browser, Mozilla also contains email client software and a web authoring tool.

8.1.2 OSS Infrastructure In Learning Institutions

Make OSS infrastructure ubiquitous at four (4) different levels in public schools, colleges and universities by:

- Running OSS servers and providing school-wide or campus-wide services such as Internet access
- Providing OSS operating systems for classroom and administrative computers
- Providing OSS desktop applications for classroom and administrative computers
- Providing OSS training for teachers, lecturers, lab administrators and personnel responsible for the IT infrastructure

Please refer to the Adoption and Implementation section for detailed guidelines on various OSS applications that can be applied at the above-mentioned levels. For a list of equivalent OSS, please refer to *MyGIFOSS*.

8.1.3 Develop Education Materials Using OSS

Learning institutions are encouraged to develop and use courseware which is developed based on OSS. For example, Linux distribution typically comes with an offering of desktop software such as productivity suite, multimedia and Internet tools. With the bundling of this software, teachers and lecturers have access to courseware development tools without having to acquire expensive tools.

8.1.4 Reskill Existing IT Coordinator

Existing IT Coordinators shall be reskilled by providing them appropriate OSS training to build competency and to keep them updated with the open source technology. The IT Coordinators shall provide OSS technical support to students, teachers and lecturers, as well as maintain the IT labs.

8.1.5 OSS At Teacher Training Colleges

Strategically, it is imperative to equip teachers with necessary OSS skills and knowledge at the training colleges. Below are recommended guidelines on how to incorporate OSS knowledge during teachers’ training at the colleges.
8.1.5.1 OSS For Teacher Training Colleges

- All teachers shall acquire ICT literacy skills through the use of OSS as the basis of the computer literacy curricula. Teachers who have already gone through ICT literacy programme shall be retrained in OSS.
- All teachers who teach Information Technology, Computer Studies and Computer in Education (CIE) subjects shall go through more detailed OSS training courses, in which they will be exposed to open source language programming such as PHP, MySQL database course and other courses related to OSS.
- All teachers are encouraged to take short-term ICT courses including those in OSS (for example, Introduction to OpenOffice, PHP Programming Language) to increase their OSS skills.
9 TRAINING

The objective of OSS Training Guidelines is to enhance the knowledge and build the skill of Public Sector personnel to be competent in OSS implementation. Agencies must be committed in educating and reskilling its personnel to ensure their competency in Open Source Software.

Public Sector agencies are required to ensure that their personnel are trained in both technical (i.e. OSS technical support, system and network administrator, programming and desktop) and non-technical areas (i.e. OSS licensing, OSS opportunity identification and OSS software maturity assessment)

The OSS training strategy will be aimed at providing basic skills to end users, programmers and administrators in an establishment where OSS is implemented.

9.1 Guidelines For Training

Public Sector agencies are recommended to follow the following guidelines in order to adhere to the above mentioned objective.

9.1.1 Training Roadmap

It is recommended that Public Sector agencies to train their personnel with OSS basic skills before taking any OSS certification programme. The following diagram shows the OSS training roadmap.
Figure 9.1 – OSS Training Roadmap
9.1.2 Technical Training

A  End User Training

An introductory class to OSS that covers topics such as, Introduction to Linux Desktop and OpenOffice, is highly recommended for agencies personnel to attend. It allows beginners to explore the Linux OS from the basic for the purpose of using the system for Word Processing, Office Automation and Communications.

B  Developer and Administrator Training

Public Sector agencies are recommended to train their developers and administrators with the following types of training:

- Technical Support Course: For example, Linux Fundamental, Installation and OpenOffice
- Administration Course: For example, Linux System and Network Administration
- Developer Course: For example, PHP Programming and MySQL Database.

C  OSS Certification Programme

A certification is recommended to complement the retraining effort. Certification ensures that all technical personnel possess a consistent and sufficient level of skill before they are allowed to implement OSS projects. The followings are the recommended open source certification programmes for the agencies to consider:

- A certification programme recommended for GNU/Linux is Linux Professional Institute (LPI)'s Certification (LPIC). There are three (3) levels of LPI certification (Junior Level Administrator, Intermediate Level Administrator and Senior Level Administrator). Please refer to http://www.lpi.org for information on LPI certificate.
- MySQL Certification Program is a certification programme that provides developers and DBAs with the credentials to prove they have the knowledge, experience and skills to use and manage the MySQL products. Please refer to http://www.mysql.com for information on MySQL certificate.
- Zend PHP Certification is the industry standard in PHP certification. It recognises outstanding PHP expertise and serves as a measure of distinction for PHP developers. Developed under curriculum criteria specified by a commercially neutral advisory board made up of renowned PHP developers worldwide, the Zend PHP Certification Exam ensures those who pass are truly proficient in PHP. Please refer to http://www.zend.com for more information on Zend Certified Engineer (ZCE).
9.1.3 Non-Technical Training

A  OSS Ownership

Users must also receive training on how to manage licensing and ownership issues related to the development and redistribution of OSS within the Public Sector. They must be trained to assess the appropriate licenses to be used under different scenarios as well as to identify and address related issues that may arise from the situation.

B  OSS Opportunity Identification

Public Sector agencies must be trained to identify the relevant OSS opportunity areas in their respective agencies within the six (6) solution areas:

- Workload Consolidation
- High Performance Computing (Clusters)
- Distributed Enterprise
- Application Solution
- Infrastructure Solution
- Desktop Solution

C  OSS Maturity Assessment

Public Sector agencies shall be trained to assess the level of OSS maturity prior to implementation. The objective of this assessment is to ensure that the OSS of interest can fulfil the agencies’ requirements. Please refer to Adoption section for information on software maturity assessment.

9.1.4 Train the Trainer

Besides the abovementioned training areas, another training initiative that is highly beneficial towards a wider use of OSS in the Public Sector is the Train the Trainer course.

The objective of this course is to train the potential trainers at the agencies to teach OSS to the end user. The training can be divided into two (2) parts. For the first part, basic technical OSS training will be covered. For the second part, the main focus of training is to equip the personnel with the necessary skills to set up and maintain the training environment at the agencies, i.e. network, security, users management, backup/restore etc. as well as tips on teaching topics on Open Source Software.
Public Sector agencies shall send suitable personnel that have the following pre-requisites to this course:

i. Experience in delivering IT or similar programmes
ii. Able to use computers in a networked environment
iii. Preferable but not compulsory:

- Knowledge of open-source software and its issues
- Knowledge in installing and setting up Linux in a networked environment

Public Sector agencies could enquire on such a programme either to the OSCC or the Malaysian National Computer Confederation (MNCC).
APPENDIX 1

Navica OSMM Model
NAVICA OSMM

OSMM developed by Navica is designed to enable organisations to evaluate open source products and to assess whether a product can fulfil the organisation's requirements.

The OSMM assesses a product's maturity in three (3) phases:

- Phase 1: Assess vital product elements (software, support, documentation, training, integration, and services) for maturity and assign a maturity score.
- Phase 2: Define a weighting for each element based on the organisation's requirements.
- Phase 3: Calculate the product's overall maturity score.

A graphical representation of the OSMM is presented in figure below. It illustrates the OSMM process and gives a good understanding of the relationships between the three phases.

<table>
<thead>
<tr>
<th>Phase 1: Assess Element Maturity</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define Requirements</td>
<td>Locate Resources</td>
<td>Assess Element Maturity</td>
</tr>
<tr>
<td>Product Software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product Integration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Professional Services</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1 – Open Source Maturity Model (OSMM)

A Phase 1: Assess Element Maturity

The first phase identifies key product elements and assesses the maturity level of each element. Key elements are those that are critical to implementing a product successfully; product software, support, documentation, training, product integrations and professional services.

Each element is assessed and assigned with a score via a four-step process:

Step One: Define Requirements. The purpose of this step is to define the agency's requirements
for a particular element. For example, if an agency wants to implement an open source Web content cache, it must determine what functionality it requires in the software based on the agency’s purpose—if it’s attempting to reduce bandwidth load, reduce response time, or the like. As another example, if an agency is implementing an open source J2EE application server, its training requirements will be vastly different if it already has significant experience with a commercial application server rather than to begin using one for the first time. Defining the requirements for an element is a key step in assessing the usefulness of a product for a particular agency.

**Step Two: Locate Resources.** Locating the resources for an element is challenging, but there are a number of methods to identify resources that can assist an agency in implementing open source software. As an example, product forums can be searched to locate a service provider that can supplement an agency’s own personnel resources.

**Step Three: Assess Maturity.** This is the key activity in determining how useful a product element is to the agency. Determining where the element lies on the maturity scale (from nonexistent to production-ready) allows an agency to determine how likely the product will be to satisfy its requirements.

**Step Four: Assign Maturity Score.** After the maturity assessment is complete, a maturity score between 0 and 10 is assigned to document how well the product element meets the agency’s requirements. The score serves as a concrete output of Step Three: It documents the consensus of the agency. Assigning a score also compels the agency to crystallise its judgement.

Element scores are also helpful when comparing different products— it’s easy to compare, say, the training maturity of two different open source content management systems in light of the agency’s needs. This can help the agency as a decision tool, enabling it to select one product or another based on the specific requirements of the agency.

Finally, the maturity score serves as an input into improving the element’s maturity. If a product’s overall maturity score is satisfactory, but one element’s maturity score is low, the agency can take steps to improve that element’s maturity.

**Phase 2: Assign Weighting Factors**

The OSMM assigns a weighting to each element’s maturity score. Weighting allows each element to reflect its importance to the overall maturity of the product. For example, the heaviest weighting is typically assigned to the product software, whereas other elements have lower weighting factors to reflect the fact that they are less critical than the software itself in determining overall product maturity.
Agencies might choose to adjust the default weighting factors based on their specific needs. For example, if an agency is stretched very thin in terms of personnel, it might plan to have an open source product implemented by a professional services firm. In that case, it might increase the weighting factor for professional services to reflect the relative importance of professional services. This allows the OSMM the flexibility to apply to every agency’s situation. A product’s maturity score will reflect the agency’s specific needs and resources.

The only requirement for adjusting the maturity weighting is that the element scores must sum to 10, since the final step of the OSMM is to create an overall maturity score that is normalised to a 100 point scale.

**C Phase 3: Calculate the Product’s Overall Maturity Score**

After each element has been assessed and assigned a weighting factor, the overall product maturity score is calculated. The element scores are summed to give an overall product maturity score on a scale of 1 to 100.
APPENDIX 2

Cap Gemini OSMM
CAP GEMINI OSMM

The Cap Gemini OSMM model in close cooperation with the customer allows to deliver the following:

a. **Determine** the maturity of an Open Source product,

b. **Access** an Open Source product’s match to the business requirements,

c. **Compare** Open Source products with commercial alternatives,

d. **Show** the importance of an Open Source Partner (OSP).

This model allows you to determine if or which open source product is suitable using just seven clear steps.

**Figure 1 : The Seven (7) Steps**

The OSMM describes how to determine product and application indicators, how to score based on these indicators and how the selection of a product is achieved.
Product indicators are grouped into four (4) different groups.

- **Product** - The Product group focuses on the ‘internals’ of the product, things like the development and stability of the developer group or the purpose of the product.

- **Integration** - The Integration group measures the options to link the product to other products or infrastructure. In addition it is also a measure for the product’s modularity.

- **Use** - The Use group tells us something about the way in which the user is supported in everyday use of the product. For instance, by reviewing the number of support options made available to the user.

- **Acceptance** - The Acceptance group is all about the way the product is received in the user community, as this is largely indicative of the product’s ability to grow and become a prominent product.

An Open Source product cannot (just as any other products) be introduced into a working environment based solely on a measurement of its strengths and weaknesses. To properly assess the product, one must also take into account several environmental aspects and naturally the present and future demands of the user by defining the following application indicators:

- **Usability** – The intended user audience, the experience of that group.

- **Interfacing** – Required connectivity, which standards are applicable. How does this fit into the organisation?

- **Performance** – The expected load and processing capability. The performance demands that must be met.

- **Reliability** – What level of availability should the product deliver?

- **Security** – What security measures are required, what restrictions are imposed onto the product?

- **Proven technology** – Does the product use technology that has proven itself in daily production?

- **Vendor independence** – What level of commitment between supplier and user does the product demand?

- **Platform independence** – Is the product available for particular ICT environments only, or does the product allow a wide range of platforms?

- **Support** – What level of support is required?

- **Reporting** – What reporting facilities are required?

- **Administration** – Does the product allow the use of existing maintenance tools, what are the demands for operational management?

- **Advice** – Does the client require validation / recommendation by independent parties, if so, what is required?

- **Training** – Required training and facilities.

- **Staffing** – Is product expertise bought, taught or hired?

- **Implementation** – What is the preferred implementation scenario?
These product and application indicators receive a score valued between one and five. One is poor, five is excellent and three is average. All the scores are summed to produce a product score. The value of this score tells us something about the general maturity of the product. Each of these groups consists of a number of indicators, which together form the product score.

**EXAMPLE:**
To show how the OSMM calculates an example is presented below. In this example, we will compare two non-existing products, A and B. First the product indicators are discussed, followed by the application indicators.

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Immature</th>
<th>Mature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>The project has just started. The stability of the developers group and need for the product is unclear.</td>
<td>The project has been active for some time. The need for the project stability project are no longer issues.</td>
</tr>
<tr>
<td>Licensing</td>
<td>Not fully described or clearly unuitable for the product.</td>
<td>One of the &quot;standard&quot; licences (<a href="http://www.opensource.org/licenses">www.opensource.org/licenses</a>) offers clear options for choosing the licence type, which is supported by the user community. Often allows both commercial and Open Source variants to co-exist.</td>
</tr>
<tr>
<td>Human hierarchies</td>
<td>Original founder is lead developer and solely responsible. Development depends on a single person.</td>
<td>Large community, multiple leaders who coordinate. Separation of development and maintenance.</td>
</tr>
<tr>
<td>Selling points</td>
<td>Only enthusiasm.</td>
<td>Commercial issues like security or maintainability.</td>
</tr>
<tr>
<td>Developer community</td>
<td>Small tight-knit group.</td>
<td>Very active developers community, several hand-offs have taken place. Documented procedures to becoming a member.</td>
</tr>
<tr>
<td>Integration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modularity</td>
<td>No modules, still single product functionality is offered on a &quot;take all or nothing&quot; basis.</td>
<td>Product has been separated into smaller pieces of functionality. Users can select which parts are required. Allows tailoring of the product to a particular situation.</td>
</tr>
<tr>
<td>Collaboration with other products</td>
<td>Not in focus yet. Product development is still firmly centred on core functionality.</td>
<td>Product is close to completion. Attention is shifting to linking the product to other products.</td>
</tr>
<tr>
<td>Use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standards</td>
<td>Uses proprietary protocols or uses dead technologies.</td>
<td>Uses current accepted protocols models. Deals with issues surrounding standards, integration and etc.</td>
</tr>
<tr>
<td>Support</td>
<td>Just within the own community and then only provided by a small minority within that community.</td>
<td>Besides community support, professional support can be purchased. An SLA can be negotiated. The community itself is active and questions draw responses from a wide section of the community.</td>
</tr>
<tr>
<td>Acceptance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ease of deployment</td>
<td>Little to no training facilities or courses. Documentation is poor, particularly with regard to maintenance.</td>
<td>Training or courses available. In addition to well written documentation lots of HowTo55 of users detailing particular situations. Within the group knowledge about maintaining the product is readily available.</td>
</tr>
<tr>
<td>User Community</td>
<td>Small group, possible with a high proportion of &quot;lurkers&quot;.</td>
<td>Large group that often has divided itself into sub-groups. Each group has a specific focus. Traffic in general is best described as high-volume.</td>
</tr>
<tr>
<td>Market penetration</td>
<td>Few references, just local promotion. Low exposure is the reason the product is not widely known.</td>
<td>Often mentioned by others (for example Gartner, ZDNet, Netcraft, IDC). Multiple cases of successful implementation across a range of companies. Well known.</td>
</tr>
</tbody>
</table>

*Figure 2: The Four (4) Different Groups Of Product Indicators*
### Product Indicators

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Product A</th>
<th>Product B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Licensing</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Human hierarchies</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Selling points</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Developer community</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td><strong>Integration</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modularity</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Collaboration with other products</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Standards</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td><strong>Use</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Ease of deployment</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td><strong>Acceptance</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User community</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Market penetration</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td>23</td>
<td>27</td>
</tr>
</tbody>
</table>

*Figure 3: Product Indicators*

### Application Indicators

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Priority (P)</th>
<th>Product A</th>
<th>Product B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usability</td>
<td>5</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>Interfacing</td>
<td>4</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>Performance</td>
<td>3</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>Reliability</td>
<td>4</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Security</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Proven technology</td>
<td>5</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>Vendor independence</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Platform independence</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Support</td>
<td>3</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>Reporting</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Administration</td>
<td>3</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>Advice</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Training</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Staffing</td>
<td>5</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Implementation</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

*Figure 4: Application Indicators*