Australian Government
Department of Finance and Deregulation
Australian Government Information Management Office

A Guide to Open Source Software for Australian Government Agencies

June 2011

Version 2.0

AUSTRALIAN GOVERNMENT INFORMATION MANAGEMENT OFFICE (AGIMO)

A Guide to Open Source Software for Australian Government Agencies

June 2011

Version 2.0

AUSTRALIAN GOVERNMENT INFORMATION MANAGEMENT OFFICE (AGIMO)

Contact us

Enquiries regarding the licence and any use of this document are welcome at:

Assistant Secretary
Governance and Policy Branch
Australian Government Information Management Office
Department of Finance and Deregulation
John Gorton Building
King Edward Terrace Parkes ACT 2600
Email: aga@finance.gov.au

Foreword

I am delighted to be publishing Version 2 of the Guide to Open Source Software for Australian Government Agencies.

Open source software is an alternative to proprietary software that provides users with the ability to view, copy, modify and distribute the software, subject to licensing conditions. Open source software can offer benefits to both the Australian Government and wider community, such as improving interoperability and possible cost savings.

Under the Australian Government's **Open Source Software Policy** (AGIMO Circular 2010/004 released in January 2011), agencies must actively and fairly consider open source software in all their information and communications technology (ICT) software procurements. As a result, the Guide has been updated to reflect the policy and the increasing maturity of open source software.

The guide provides practical information to assist agencies assess open source software solutions, including the key issues to consider when procuring open source software. It also provides information on the types of open source software licences, licensing risks and risk mitigation techniques.

This document is a companion document to the 2007 publication *A Guide to ICT Sourcing for Australian Government Agencies* (second edition). The Department of Finance and Deregulation would like to thank the Australian Taxation Office and the Australian Government Open Source Software Community of Interest for their assistance in developing Appendix 1: Australian Government Open Source Software Licensing Risk Framework.

Ann Steward
Australian Government Chief Information Officer
Australian Government Information Management Office
Department of Finance and Deregulation

Contents

one Introduction

one

one Introduction

The Guide to Open Source Software for Australian Government Agencies provides an introduction to open source software. It includes background information on the benefits and risks of using, modifying, distributing and developing open source software and guidance to assist agencies understand, analyse, plan for and deploy open source software.

1.1 Intent

The guide is a stand-alone reference document on open source software; however, agencies are encouraged to read it alongside **A Guide to ICT Sourcing for Australian Government Agencies** (Guide to ICT Sourcing). This guide is not a substitute for legal or procurement advice. Any decisions on the use of software, including open source software, or associated services should be made according to the **Commonwealth Procurement Guidelines** and the Australian Government's **Open Source Software Policy**

Agencies should be aware that this guide is focused on open source software. It does not provide a complete picture of the benefits and risks of using proprietary software solutions.

1.2 Audience

Although this guide can be considered general background reading for anybody who is interested in open source software within government, the primary audiences for this guide are project managers and procurement teams who are sourcing software to meet business requirements. Agency personnel who influence the selection of software may also find this guide useful.

The Australian Government ***Open Source Software Licensing Risk Framework*** is designed for ICT specialists.

two What is open source software?

two

two What is open source software?

Open source software is a popular term in the information and communications technology (ICT) industry, but it can mean different things to different people. This section defines open source software and highlights its benefits.

Agencies should keep in mind that open source software is not intrinsically of higher or lower quality than proprietary software. It is not inherently more or less secure, and it does not necessarily have a higher or lower total cost of ownership.

2.1 Definition of open source software

The Open Source Initiative (OSI),[1] an organisation established to promote open source software, has developed an Open Source Definition (OSD) as follows:

The Open Source Definition

Introduction
Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution
The licence shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The licence shall not require a royalty or other fee for such sale.

2. Source Code
The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works
The licence must allow modifications and derived works, and must allow them to be distributed under the same terms as the licence of the original software.

4. Integrity of The Author's Source Code
The licence may restrict source-code from being distributed in modified form *only* if the licence allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The licence must explicitly permit distribution of

[1] Open Source Initiative: http://opensource.org

software built from modified source code. The licence may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups
The licence must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavour
The licence must not restrict anyone from making use of the program in a specific field of endeavour. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of Licence
The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional licence by those parties.

8. Licence Must Not Be Specific to a Product
The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's licence, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. Licence Must Not Restrict Other Software
The licence must not place restrictions on other software that is distributed along with the licensed software. For example, the licence must not insist that all other programs distributed on the same medium must be open-source software.

10. Licence Must Be Technology-Neutral
No provision of the licence may be predicated on any individual technology or style of interface.

Misconceptions

Although open source software often involves a distinctive development and distribution model, it may also be bundled and sold as part of a package with proprietary software. Software can be offered under both open source and proprietary licences. Where software is dual-licenced, agencies should choose the arrangement that best matches their requirements and provides value for money.

Open source software is sometimes confused with public domain software, shareware, community source software and freeware.[2] In addition, open source software is often linked with open standards; however, not all open source software products use open standards.

[2] Definitions of public domain software, shareware, community source software and freeware are available at Appendix 3: Definitions. Note: freeware is not the same as free software. For a definition of free software, see the Free Software Foundation's website: http://www.gnu.org/philosophy/free-sw.html.

Another common misconception about open source software is that it can always be obtained free of financial cost. When open source software is labelled as 'free', that word refers to the ability of people to read, modify and redistribute the source code of the software, not the cost of the software.[3] The definition of open source software does not preclude people from selling the software. However, despite this, open source software is usually available free of upfront costs, although agencies still need to be aware of the total cost of ownership (TCO).

2.2 Development and support of open source software

There are three broad models for open source software development and support:

- ***Volunteer community.*** A large proportion of open source software is developed by a community of skilled people who usually communicate online. In this model, there is no specific corporation managing the development process. Support is available through the members of the community, who have forums and other feedback mechanisms to receive requests from users. There is generally no service level agreement available from the community. Popular packages such as the Apache web server and the Linux operating system have been developed using this model.
- ***Corporate-backed community.*** Some commercial organisations provide support for open source software. The commercial organisation may choose to create its own community to develop the open source software or they may choose to leverage off an existing product created by a volunteer community. The commercial organisation usually provides support to a defined service level agreement. More than one organisation can provide support for a product, leading to competition based on the quality and price of the service. For example, Oracle's and IBM's web servers are both based on the community-developed Apache.
- ***Commercial open source.*** Some open source software is developed or supported by a single corporation. Sun Microsystems (now owned by Oracle) provides the OpenSolaris operating system under this model.

2.3 Benefits of open source software

Open source software has a number of potential benefits. These benefits are not applicable in every instance; however, they can be seen as general characteristics of open source software. Some of these benefits can be realised only when agencies contribute back to the community. In some cases there are risks associated with the benefits, as discussed in **Section 4.**

[3] All software is written in source code. Source code refers to the underlying, human-readable programming instructions written by software developers. Source code is used to specify actions to be performed by the computer. In most circumstances, programming instructions are compiled into binary code, the machine-readable code that actually runs or executes on a computer or is interpreted by another platform.

Open source software:

- **Usually has no upfront payment.** The lack of upfront payment may seem to benefit agencies financially; however, as with all software, agencies should consider the total cost of ownership, including all support services that will be required to operate the software over its lifespan.
- **Encourages a competitive market for support services.** Because the source code is available, it is possible for any software organisation to provide support for an open source product. In addition, customers are able to support the software themselves.
- **Encourages a collaborative approach.** Open source software encourages an open exchange of ideas, where any user of the software can contribute ideas to improve it. This tends to promote a collaborative approach that may foster innovation.
- **Places fewer restrictions on the users of the software.** Most open source software licences place fewer restrictions on the users of the software and emphasise respect for the privacy of the users. However, agencies should ensure that they understand the obligation for reciprocity that is included in many open source licences.
- **Provides the opportunity for users to take direct control of the maintenance and support of the software.** This may be a benefit to agencies that possess the appropriate skill base.
- **Allows the opportunity to try the software before committing to it.** This will enable agencies to test the viability of the software before fully committing to it.
- **May reduce vendor lock-in.** As the source code is publicly available, most licences will allow any individual or group to further develop the software without the obligation to support other users, even if the original community discontinues development. Commercial organisations may provide support for an open source package, if there are enough users willing to pay for that service.
- **Allows users to view and modify the source code.** The ability of users to scrutinise and change the source code of open source software may lead to increased stability and security. It also allows agencies to tailor the software to their own needs.
- **Allows users to take advantage of the improved functionality of new releases more rapidly.** Many new open source software communities follow the maxim of release early, release often', meaning that users can quickly gain extra functionality for the software.
- **Increases interoperability.** Many open source software packages use open standards, which tend to lower the costs of integration and improve interoperability.[4]
- **Usually is modular.** Open source software packages are generally modular, which means that changes to one part of the source code is less likely to affect the rest of the software package.

[4] Agencies should be aware that not necessarily all open source software solutions will use open standards.

three Australian Government Open Source Software Policy

three

three Australian Government Open Source Software Policy

This section describes the principles that underpin the Australian Government's policy in regard to the procurement of open source software and suggests ways that consideration of open source software can be incorporated into procurement processes.

In January 2011, the Australian Government released a policy requiring agencies to consider open source software for all software procurements. The **Open Source Software Policy,** which is available from the Department of Finance and Deregulation website, will apply to any ICT procurement activity initiated after 1 March 2011.

3.1 Principles

The policy directs agencies to comply with three core principles.

Principle 1: Australian Government ICT procurement processes must actively and fairly consider all types of available software.

Australian Government agencies must actively and fairly consider all types of available software (including but not limited to open source software and proprietary software) through their ICT procurement processes. It is recognised there may be areas where open source software is not yet available for consideration. Procurement decisions must be made based on value for money. Procurement decisions should take into account whole-of-life costs, capability, security, scalability, transferability, support and manageability requirements.

For a covered procurement (over $80K), agencies are required to include in their procurement plan that open source software will be considered equally alongside proprietary software. Agencies will be required to insert a statement into any Request for Tender that they will consider open source software equally alongside proprietary software. Tender responses will be evaluated under the normal requirements of the Commonwealth Procurement Guidelines. For a non-covered procurement (below $80K), agencies are required to document all key decisions, as required by the Commonwealth Procurement Guidelines. This includes how they considered open source software suppliers when selecting suppliers to respond to the Select Tender or Request for Quotation.

Principle 2: Suppliers must consider all types of available software when dealing with Australian Government agencies.

Australian Government agencies will require suppliers to consider all types of available software (including but not limited to open source software and proprietary software) when responding to agencies' procurement requests.

Agencies are required to insert this requirement into their tender documentation. Suppliers will need to provide justification outlining their consideration and/or

exclusion of open source software in their response to the tender. Agencies will determine compliance with this requirement when assessing tender responses.

Principle 3: Australian Government agencies will actively participate in open source software communities and contribute back where appropriate.

The Australian Government, through AGIMO, will actively seek to keep up-to-date with international best practice in the open source software arena, through engaging with other countries and organisations. Australian Government agencies should also actively participate in open source software communities and contribute back where appropriate.

3.2 Compliance

The policy suggests sample draft clauses designed to assist agencies in complying with the policy. Agencies may choose to draft their own clauses.

The policy provides the following sample clauses:

- **for inclusion in procurement plan/procurement documentation**

  [Agency Name] will actively and fairly consider all types of available software for ICT software procurements. Open source software will be considered equally alongside proprietary software.

- **for inclusion in request for quote/select tender checklists**

  Have you considered all types of available software (including but not limited to open source software and proprietary software)?

- **for inclusion in requests for tenders for covered procurements**

  [Agency Name] encourages suppliers to submit and/or develop open source software for this tender. When responding to this tender, suppliers must demonstrate a willingness to actively consider open source software throughout all stages of procurement, solution design and implementation in order to produce a product that demonstrates value for money and is fit for purpose. This may include incorporating open source software components together with proprietary software components.

  In evaluating the tender, *[Agency Name]* will consider open source software equally alongside proprietary software.

- **for inclusion in request for tender assessment checklists**

  Has the supplier sufficiently demonstrated that they have considered all types of available software (including but not limited to open source and proprietary software)?

Agencies are also encouraged to include a definition of open source software in their procurement documentation.

four Procurement of open source software

four

four Procurement of open source software

This section uses the Department of Finance and Deregulation's four-phase ICT sourcing lifecycle to identify issues that agencies should consider when procuring open source software. Further details on the four-phase ICT sourcing lifecycle can be found in **A Guide to ICT Sourcing for Australian Government Agencies** (Guide to ICT Sourcing).

The following sub-sections identify the common issues in software procurement and the specific issues that should be considered when procuring open source software. It is important for agencies to understand the range of different software options available. Agencies need to ensure that they comply with the procurement procedures outlined in the Commonwealth Procurement Guidelines, any relevant agency Chief Executive Instructions and any relevant whole-of-government ICT policies.

4.1 Common issues in software procurement

In many aspects, procuring open source software is similar to procuring proprietary software. Agencies must consider the following when procuring either open source software or proprietary software:

- **Applicability of the *Commonwealth Procurement Guidelines*.** Agencies must always follow the *Commonwealth Procurement Guidelines* when selecting a software solution.
- **Total cost of ownership.** When considering value for money, agencies need to take into account the total cost of ownership (TCO), also known as the whole-of-life costs, for use of the software. Even software that can be downloaded and used without cost may have downstream support, maintenance and exit costs. Agencies may need to purchase services for maintenance, support and deployment, and they may also have costs involved with installation, system integration, data conversion and testing. Agencies may also need to pay a developer to modify or integrate the software. Refer to the Guide to ICT Sourcing for more information.
- **Matching support and maintenance arrangements to the agency's requirements.** Agencies should ensure that the risk profile of their service level agreement for support and maintenance is appropriate for the business criticality of the software. Most agencies will incur some combination of internal staff charges and external support and maintenance charges for either proprietary or open source software.
- **Matching product innovation, maturity and roadmap to the agency's requirements.** There are variations in the stability, innovation and maturity of both open source and proprietary software packages. Agencies need to take these differences into account when procuring software.
- **Aligning with the agency's strategy and architectures.** The strategy and architectures of an agency may dictate certain principles, standards and technologies that need to be taken into account when considering new software. Consistent application of an agency's strategy and architectures helps to reduce staff training and ICT support costs.

4.2 Four-phase ICT sourcing lifecycle

The Guide to ICT Sourcing divides the sourcing lifecycle into four phases. The key issues to consider for open source software in each phase are:

- **Phase I—Case for change**
  - Agencies should clarify their business need using their strategy and architectures to define their business case for change. This may include identifying any need for innovation, maturity, support, and integration with existing software or systems.
- **Phase II—Decide sourcing strategy**
  - Agencies should decide whether there is any justification for limiting their software selection to specific technologies, packages or software models. It should be noted that an approach to an open market will provide the most objective evidence of available options. Agencies should consider the market conditions and TCO, especially support and transition costs.
  - Agencies should also consider any whole-of-government ICT policies that may influence their decision making, for example, the **ICT Customisation and Bespoke Development Policy.**[5]
  - Agencies should be aware that open source software can be sourced ‹in house' by downloading open source software from various online repositories. The benefits and risks of 'in house' sourcing should be assessed, including the TCO.
- **Phase III—Undertake procurement**
  - Agencies' procurement processes must be compliant with the **Open Source Software Policy.**
  - Agencies should ensure that there is a software licence management framework, especially if they choose to procure open source software. See **Appendix 1: Australian Government Open Source Software Licensing Risk Framework** for more information.
  - Agencies should be aware that it may be necessary to procure support services separately for open source software.
- **Phase IV—Transition and manage**
  - Agencies should continue to manage their software against the licence conditions.
  - Agencies should also keep up to date on the changing software industry landscape.

[5] The ICT Customisation and Bespoke Development Policy is focused on strengthening governance around customisation and bespoke development. Agencies can still customise or bespoke develop software provided they comply with the requirements of the strengthened governance arrangements. Within the purview of the ICT Customisation and Bespoke Development Policy, open source software that is not customised – including commercial software licensed under an open source software licence – is considered off-the-shelf software.. In this instance, customisation is any deviation from the available versions of the open source software.

five Comparing open source and proprietary software

five

five Comparing open source and proprietary software

This section highlights some of the key issues that agencies should consider when comparing open source software to proprietary software.

Agencies need to understand the opportunities and risks associated with the different software options. Implementing open source software does not necessarily expose an agency to greater risk than implementing proprietary software; however, there may be a change in the risk profile. Some of the factors that will affect the risk profile are:

- **How the agency is using the software.** An agency may use the software as supplied, modify it, distribute it or use it as a component of another software implementation.
- **The business alignment of the initiative.** The Guide to ICT Sourcing provides a framework for assessing initiatives as vital, duty-bound, or discretionary and support. This is based on the relevance of the initiative to the agency's core business.

5.1 Key issues

Agencies need to consider the following when procuring open source or proprietary software solutions.

- **Access to source code.** By definition, open source software makes the source code available to anyone for viewing, vetting and modification. Proprietary software generally restricts access to and modification of its source code.
- **Capital expenditure.** Although open source software usually has no upfront cost, the TCO is unlikely to be nil, even if an agency provides in-house support.[6] Proprietary software generally includes an upfront fee, unless the proprietary software is provided as a service.[7] Agencies should consider the TCO for both proprietary and open source solutions. Considerations include acquisition, deployment, integration, support and maintenance, training and exit costs. However, there may be an opportunity to leverage an agency's existing software investments, for example, if an agency's software uses a particular standard, the cost of integration may be reduced by integrating software that supports the same standard.
- **Customisation.** Agencies should consider whether they need to customise the software and whether there are any applicable whole-of-government ICT policies (for example, the **ICT Customisation and Bespoke Development Policy**). Customisation of open source software can be undertaken either by the agency or by a third party. If agencies choose to customise the software, they should consider the cost of future support, maintenance and upgrades. Agencies should also consider any licensing obligations. If agencies customise open source

[6] While most open source software can be readily downloaded and used without paying a licence or acquisition fee of any kind, this is not an inherent characteristic of all open source software.

[7] Agencies should note that alternative sourcing models such as cloud computing also do not have capital expenditure. For further information on cloud computing, see the Australian Government's Cloud Computing Strategic Direction paper: http://www.finance.gov.au/e-government/strategy-and-governance/cloud-computing.html

software and do not contribute the modified product back to the open source software community, this is called code forking. Code forking is discussed in further detail in the next section.

- **Development/Governance.** Open source software is generally developed by communities of developers who work together online.[8] These communities may also be supported by commercial organisations. An open source software community with an active and diverse membership, a broad user base, a good governance structure and regular updates is more likely to be responsive to user requests. The corporate history and product roadmap of proprietary software vendors may give agencies an indication of the quality of the vendor. Before an agency commits to using any software package, it should carefully assess the credentials and resources of the developers. The agency should consider whether appropriate development of the software will continue during the expected lifespan of its use by the agency.
- **End user.** The training necessary for end users should be considered whenever a new software purchase is made or an upgrade is obtained.[9]
- **Innovation.** The nature of open source software allows agencies to contribute back to the product, which can aid innovation. However, this may affect the TCO of the product, as agencies will need to factor in the cost of contributing back (i.e. staff costs). Historically, proprietary software relies on the vendor to drive innovation.
- **Intellectual property.** There is a specific exemption for software governed by open source licences in the Australian Government's *Statement of Intellectual Property Principles for Australian Government Agencies*. This exemption allows the Commonwealth to retain intellectual property in products governed by open source licences.[10]
- **Liability**. Agencies need to be aware of any liability they may face when modifying and distributing software. Any liability that agencies may face is generally listed in the software licence conditions under disclaimer of liability or disclaimer of warranty.
- **Licence obligations.** Agencies should be aware of their licensing obligations, including the possibility of the software being dual licensed. Some open source software licences may oblige agencies that modify and distribute the software to contribute all changes back to the open source software community. Proprietary software also comes with its own set of licensing obligations.
- **Lock-in.** Agencies should be aware of the risks of being locked-in to one type of software. Open source software may align to open industry standards, which can improve interoperability and reduce vendor lock-in. **A Guide to ICT Sourcing for Australian Government Agencies** provides a detailed review of this topic. Agencies should also consider the possibility of being locked in due to a lack of support options.

[8] Many open source software products are available on SourceForge (sourceforge.net).
[9] Some open source software products will come with an option to configure them to a more familiar interface.
[10] The Statement of Intellectual Property Principles for Australian Government Agencies is available for download from www.ag.gov.au/www/agd/agd.nsf/Page/CopyrightStatement_of_Intellectual_Property_Principles_for_Australian_Government_Agencies.

- **Maturity and portability.** Agencies should ensure that they evaluate the maturity of any software product they are procuring. This includes considering the risks of having to change to a different product in the future.
- **Release management.** Open source software generally has an increased number of new releases that may have a negative impact in terms of greater requirements for integration testing, release management, bug fixes, and the associated risk management and support tasks.
- **Reliability.** Agencies should evaluate the reliability of any software product they are procuring. Commonly used open source software products may be more reliable as the community works to select the best improvements and offer them in the next release.
- **Restrictions on use.** There are typically few or no restrictions on the use of open source software. However, agencies should ensure that they understand the licence conditions before modifying the software. Agencies should also check the support arrangements. Proprietary software will usually have some restrictions on its use, which may include the requirement to pay additional licensing or support costs if there is a change in how the software is to be used.
- **Re-Use.** Open source software may encourage re-use through the community creation of solutions specifically for government use. However, agencies need to ensure that they have the appropriate governance structures in place for any shared solutions. The **ICT Customisation and Bespoke Development Policy** provides governance principles for cross-agency solution sharing.
- **Security.** Open source software allows agencies the opportunity to examine the source code, which may assist in assessing security risks. All software should be scrutinised for its security, governance and deployment arrangements, particularly if it will be used in a high-security area. The Defence Signals Directorate's Evaluated Products List provides a list of products that are certified for specific purposes and specific security levels.[11]
- **Support and maintenance.** Open source software offers the following options for support and maintenance:
  - In-house: Support and maintenance can be provided in-house by the agency.
  - Community: Free support can be provided from the open source software community.
  - Commercial: Support can be procured from a commercial organisation.

  When an agency acquires an open source software solution through an external service provider, it is generally purchasing services and receiving the related software free of charge. There is usually a competitive market for commercial support services for open source software. However, agencies need to assure themselves of the capacity and capability of any organisation claiming to offer support services. Some open source software products may depend on key individuals within a community or a specific vendor to support the product. Open source software that is in widespread use is likely to have more competitive support services. Support and maintenance for proprietary software is generally provided

[11] http://www.dsd.gov.au/infosec/epl/index.php

by the vendor or authorised partners, with a certain amount of first level support usually being provided in-house.

- **Warranties.** Open source software that is downloaded free generally does not offer warranties. However, open source software that is procured from a commercial vendor will generally come with similar warranties to proprietary software.

5.2 Beyond use: code forking and reciprocity

This section gives further information on two issues that apply when modifying or developing open source software.

Code forking

Code forking occurs when agencies make changes to the code of open source software without publishing the code back to the software's development community. The fork is the split between the agency's version of the software and the version published by the community. Any further changes made by either the agency or the community will increase the fork. This can make it difficult for the agency to upgrade to a new published version, as the agency would have to reapply all its changes. This risk may be mitigated by contributing modified source code back to the open source software community.

Code forking is similar to customising proprietary software packages. Customising commercial products can also create a future liability for the agency, as upgrading to the next supported version of the package may be more expensive and time consuming due to the customisation.

- The benefits, costs and risks of customising should be included in the business case for any software initiative. Agencies should be aware of whole-of-government ICT policies that may govern their ability to customise software, such as the **ICT Customisation and Bespoke Development Policy.** Agencies should also ensure that they have the appropriate skill base to manage the development and ongoing maintenance of the forked software.

Agencies working with open source software have the option to publish changes back to the development community. Depending on the licence, they may also be obligated to publish any changes that have been distributed. Should these changes be accepted by the community and integrated into the base product, alignment is maintained with the published version. Agencies need to consider the implications of contributing modified source code back to the community.

Reciprocity

International precedent strongly suggests that the copyright attached to open source software is legally enforceable in Australia. If agencies do not follow the terms and conditions of a software licence, they risk being in breach of copyright, which may involve prosecution.

Some open source software licences include the concept of reciprocity. A highly reciprocal licence will require agencies to make any modified code publicly available, generally

under the same licence. Low reciprocity or permissive licences do not oblige agencies to contribute back any changes. Reciprocity is triggered when a derived work is distributed. Agencies that use the open source software without modifying it are unlikely to trigger a reciprocity provision.

The current law is unclear as to the boundaries of distribution. If the modified source code is used only within one agency, it is unlikely that reciprocity will be triggered. It is strongly recommended that agencies seek legal advice whenever they seek to modify open source software. Agencies will need to consider the implications of publishing the whole of the derived work.

Agencies are encouraged to set up strong governance to track all instances of open source software in their systems. Without strong governance, agencies run the risk of being unaware of licensing risks.

Further information on reciprocity is available in **Appendix 1: Australian Government Open Source Software Licensing Risk Framework.**

Appendix 1: Australian Government Open Source Software Licensing Risk Framework

Appendix 1: Australian Government Open Source Software Licensing Risk Framework

1. Overview

Open source software is licensed under conditions that allow users to view, use and modify the source code. It is generally available free of any fee for access or use. These characteristics differentiate it from proprietary software, which usually disallows access to source code and is available only on payment of a fee.

Open source software has many potential benefits for Australian Government agencies. To access those benefits, agencies must understand the issues posed by open source software licences in respect to the use, deployment, development, modification and/or distribution of the software.

The purpose of the *Open Source Software Licensing Risk Framework* (Licensing Risk Framework) is to provide a high-level overview of the key issues that Australian Government agencies need to consider when identifying, assessing and managing risk and compliance issues to do with open source software, particularly in situations where open source software might be modified, developed or distributed. Attachment A to the framework provides a list of assumptions behind the creation of the framework.

Agencies that are considering modifying open source software are advised to consult their legal departments to clarify their licence obligations.

2. Background to the framework

The use of open source software has no more inherent risks than the use of proprietary software; in fact, open source software may have significant advantages. The Australian Government **Open Source Software Policy** requires agencies to consider both types of software when procuring software to meet their business needs.

Increasingly, Australian Government agencies are choosing to procure open source software for application in one or more of these scenarios:

- **use** without modification, similarly to proprietary software
- **customisation** through external contractors or inhouse developers
- **integration** with other software (including open source, proprietary or custom developed software)
- **bespoke development**.

Agencies need to fully understand and comply with the terms and conditions of the licences that govern the software that they use. This may involve reviewing and adapting the licensing compliance methods that they have established, to ensure that they are equally effective in managing both proprietary and open source software.

The Licensing Risk Framework is designed to advise agencies about risks and issues they may face when using, customising, integrating or developing open source software. It is not legal advice.

3. Outline of the framework

The Licensing Risk Framework will assist agency staff members to identify, understand and mitigate potential risks and issues involving open source software. Staff members who may be interested in this framework include:

- technical staff who are involved in building software solutions
- project managers who manage project delivery and project risks
- agency policy makers.

The framework is presented in four parts:

- **Part 1: Introduction to licensing risk.** This section briefly summarises the purpose of the Licensing Risk Framework and outlines why the Australian Government needs a framework to address the risks involved in open source licensing. All relevant staff members are encouraged to read this section.
- **Part 2: Anatomy of licensing risk for technical staff.** This section introduces the concept of reciprocity and describes how reciprocity applies to both the licensor and licensee. It provides a model that enables an agency to assess its licensing risk in its specific circumstances.
- **Part 3: Anatomy of licensing risk for project managers**. This section recommends how licensing risk should be managed throughout the term of a project, including those cases in which software development is outsourced.
- **Part 4: Attachments**. The attachments provide more detailed information on the assumptions on which the framework is based, technical information on particular aspects of open source software licensing and an example of an agency's approach to identifying licensing risk, based on a risk matrix.

4. Australian Government Open Source Software Licensing Risk Framework

PART 1:

Introduction to licensing risk

This part:

- outlines the importance of being aware of open source licensing risks
- offers an approach to identifying and managing the risks related to open source software licensing.

Agencies need to comply with the provisions of both open source and proprietary software licences. While proprietary software licences attract the standard protections available under copyright law, there is some ambiguity about how open source licences can be legally enforced. To date, this has not been tested in Australian courts. However, in Europe and the United States, the Free Software Foundation has successfully taken legal action against distributors that have failed to comply with open source licensing provisions.[12]

A breach of an open source licence will occur if software covered by an open source licence is used contrary to the terms of the licence. Any breach may have far-reaching consequences. For example, a breach of the GNU General Public Licence V2 immediately terminates the licence, after which only the copyright holder can reinstate the licensee's rights. Without a valid licence, the licensee must immediately cease using or distributing the software.[13]

Breaches of licence provisions are not always intentional; they may be due to a lack of governance in the tracking the use of open source software within an agency. In addition, agencies may not be aware of all the actions that may lead to a breach of an open source licence.

Therefore, as is the case for proprietary software, agencies must ensure that they have the appropriate governance in place to track, monitor and evaluate all instances of open source software. Given that open source software can generally be acquired without incurring any licence fee, traditional mechanisms for assessing and approving procurement may need to be supplemented with mechanisms for acquisition and deployment. Staff members who use the software should be informed of the licensing obligations.

[12] The Free Software Foundation is a not-for-profit organisation that advocates and educates on the value of free software.
[13] It is possible in some circumstances that use for the purposes of the Commonwealth could continue under special rights contained in the *Copyright Act 1968*, but there are other implications of this and it is the exception rather than the rule.

Identifying licensing risks

Generally, when dealing with open source software, agencies need to be aware of the:

- **Type of licence.** Open source software licences may be described in terms of their reciprocity (high, medium or low) or restrictiveness (restrictive, restrictive hybrid or permissive). Typically, licences will be both highly restrictive and reciprocal, or both permissive and lowly reciprocal. Reciprocity and restrictiveness are further defined in Part 2 of this framework.
- **Intended use.** This includes whether the agency intends to modify the open source software product (that is, create a derived product) and to distribute any modified versions of the product (derived works).

The type of licence can affect what an agency can do with the software. For example, an agency that modifies and distributes an open source product with a restrictive licence will trigger an obligation to publicly release the modified code back to the open source community. However, agencies that use open source software without modification are unlikely to trigger any licensing compliance issues. If the public release of modified code is not acceptable to an agency, the agency should consider not using open source code and tools that are subject to a restrictive licence.

Generally, an obligation to publicly release the modified code is triggered only if all three of these conditions apply:

- the licence contains a reciprocity provision
- a derived work has been produced
- that derived work has been distributed.

When agencies modify or develop open source software (for example, by using libraries or tools), they need to be aware of, and manage, the associated licensing risks and issues. Figure 1 provides a general overview of the potential risks applicable in different licensing scenarios.

Figure 1: Risk matrix for open source software use

Intended use	**Licence Type 1** High reciprocity	**Licence Type 2** Medium reciprocity	**Licence Type 3** Low reciprocity
Distribution of derived work	Very high risk potential	High risk potential	Low risk potential
No distribution of derived work	Medium risk potential	Low risk potential	Very low risk potential

Agencies can integrate a similar risk matrix into their corporate risk management framework, to help determine their individual risk appetite for initiatives that involve open source software. This will ensure that agency staff members are aware of the risk level acceptable to their agency.

Attachment E provides an example of the procedure for applying such a matrix.

After agencies have identified the risks associated with a project, they must ensure that their policies, procedures and tools are adequate to manage the risks. Effective governance of source code management is critical to managing open source software licensing risk. Project managers should also ensure that they are fully aware of all open source software within their project, and that components used in the project carry mutually compatible licences.

There are additional licensing risks when agencies engage external contractors for software development. When external contractors are engaged, agencies have limited control and day-to-day oversight over vendor activities, including the selection, use and management of open source software. Therefore, to mitigate this risk, agencies should stipulate in the contract that copyright of any code developed under restrictive open source licences lies with the agency.[14]

Mitigating risk

Table 1 describes the risk mitigation techniques that agencies may choose to follow in order to ensure compliance with software licences in general, and especially with open source licences.

[14] The *Statement of Intellectual Property Principles for Australian Government Agencies* specifically excludes ICT products governed by open source licences from the general advice to allow vendors to retain the intellectual property in software developed under contract for agencies.
http://www.ema.gov.au/www/agd/agd.nsf/Page/Copyright_CommonwealthCopyrightAdministration_StatementofIPPrinciplesforAustralianGovernmentAgencies

Table 1: Risk mitigation

	Mitigation practices for use within the agency	Mitigation practices for circulation outside of the agency
Use as is (use without modification)	• Read the licence to ensure compliance with all the terms and conditions.	• Read the licence to ensure compliance with all the terms and conditions. • Ensure that the licence allows the agency to act as a distributor. • Consider the reputational and liability risks in being seen as a distributor of the software.
Derive (create a customised or modified version in-house)	The above action plus: • Ensure that the licence conditions allow for derivation. This is generally true for open source software, but generally limited for proprietary software. • Ensure that the agency has access to the skills necessary for the development, integration, support and maintenance over the intended lifetime of use of the software in the agency, in proportion to the business criticality of its role. • Ensure that the agency has considered the implications of deviating from the baseline product, including the likely additional time, costs and risks of reapplying agency-specific changes to future baseline upgrades. • Ensure that there is a clear understanding of the legal boundary of the agency and the software's intended use. Without that understanding, an agency may accidentally distribute its derived code, which may trigger an obligation for the agency to publicly release the derived code.	The above actions plus: • Ensure that the licence conditions allow for derivation. This is generally true for open source software, but generally limited for proprietary software. • Ensure that the agency has access to the skills necessary for the development, integration, support and maintenance over the intended lifetime of use of the software in the agency, in proportion to the business criticality of its role. • Ensure that the agency has considered the implications of deviating from the baseline product, including the likely additional time, costs and risks of reapplying agency-specific changes to future baseline upgrades. • Ensure that there is a clear understanding of the legal boundary of the agency and the software's intended use. • Ensure that the licences for all components are compatible. • Consider the possibility of the agency having to publish all source code that is being integrated with open source components, or immediately stopping the distribution of the software if it is found to be in breach of licence conditions. • Consider the reputational risks to the agency due to the quality of the published software.

	Mitigation practices for use within the agency	Mitigation practices for circulation outside of the agency
Bespoke development (develop a new solution from scratch, either in-house or by use of contractors)	All of the above actions plus: • Consider the risks of open sourcing against the benefits. • Ensure that the arrangements will comply with any licence arrangements for development tools and/or licensed components. For proprietary software, this may involve a review of provisions relating to the use of runtime environments and libraries. For open source restrictive licence development tools and/or licensed components, this will involve making sure that the contractual and other arrangements accord with all obligations under relevant licences. Where external contractors are involved, this will typically mean that either the contract must provide that your organisation owns all intellectual property or that the code is licensed under an open source licence that is compatible with all other relevant tools and components forming part of the bespoke development.[1]	All of the above actions plus: • Consider the risks of open sourcing against the benefits. • Consider the best type of open source licence to govern the package. – If components/tools governed by a restrictive licence are used, the package will need to be governed by a compatible licence. – The type of licence used may affect the likelihood of an external community forming to further develop/support the open source initiative that you are starting. – Consider potential issues that may arise with a community; for example, expectations of what ongoing support or assistance might be provided by government to the community; and whether or not parts of the community may take the software in different directions (forking). – If a permissive licence is being considered, you need to consider whether any potential issues would arise if external parties were to exercise their right to include the code in commercial products. • Ensure that the arrangements will comply with any licence arrangements for development tools and/or licensed components. For open source restrictive licence development tools and/or licensed components, this will involve making sure that the contractual and other arrangements accord with all obligations under relevant licences. Where external contractors are involved, this will typically mean that either the contract must provide that your organisation owns all intellectual property or that the code is licensed under an open source licence that is compatible with all other relevant tools and components forming part of the bespoke development.[1]

[1] See the *Statement of Intellectual Property Principles for Australian Government Agencies*, www.ag.gov.au/www/agd/agd.nsf/Page/Copyright_CommonwealthCopyrightAdministration_StatementofIPPrinciplesforAustralianGovernmentAgencies

PART 2:

Anatomy of licensing risk for technical staff

This part:

- describes the concepts of reciprocity and restrictiveness, and the conditions that trigger reciprocity
- provides an open source licensing risk model for agencies to assess their licensing risks
- emphasises the need for agencies to consult with their legal departments before modifying open source software.

Each open source software licence imposes a specific set of requirements and limitations on developers wanting to modify and/or redistribute the licensed software. For links to open source licences, including an overview of the concept of dual licensing, see Attachment D. Generally, when an agency is using an unmodified open source software product, regardless of what open source licence is attached to the product, the risks are comparable to the risks incurred when an agency is using a proprietary product. However, compliance with open source licences may become complicated, particularly when a developer intends to create a new application incorporating code from a variety of sources. Whenever an agency creates a derived work and distributes that work, the agency will need to comply with any reciprocity provisions that exist in the relevant open source licence.

Industry views on what constitutes a derived work are available in Attachment B, while an analysis of what constitutes distribution is in Attachment C. Due to a lack of legal precedent in this area, the definitions and boundaries of these terms are not clear. Agencies that are considering modifying open source software should seek legal advice.

Restrictiveness and reciprocity

Reciprocity refers to the obligation to make available to the development community any changes that are made to the source code of licensed open source software. The obligation is generally triggered when all of the following provisions are met:

- The licence contains a reciprocity provision
- A derived work has been produced
- That derived work has been distributed.

Open source licences can be divided into three broad categories, based on the level of reciprocity obligations they contain:

- **Permissive** licences contain no reciprocity provisions. They aim to encourage the widespread use of the software by placing few barriers to its use.
- **Restrictive** (sometimes called copyleft) licences contain strong reciprocity provisions (to ensure the freedom of the software cannot be compromised). They aim to encourage the continued growth of the software by ensuring that those who use it share their changes with the development community.
- **Hybrid restrictive** licences contain reciprocity provisions that have some exceptions or apply only in some circumstances.

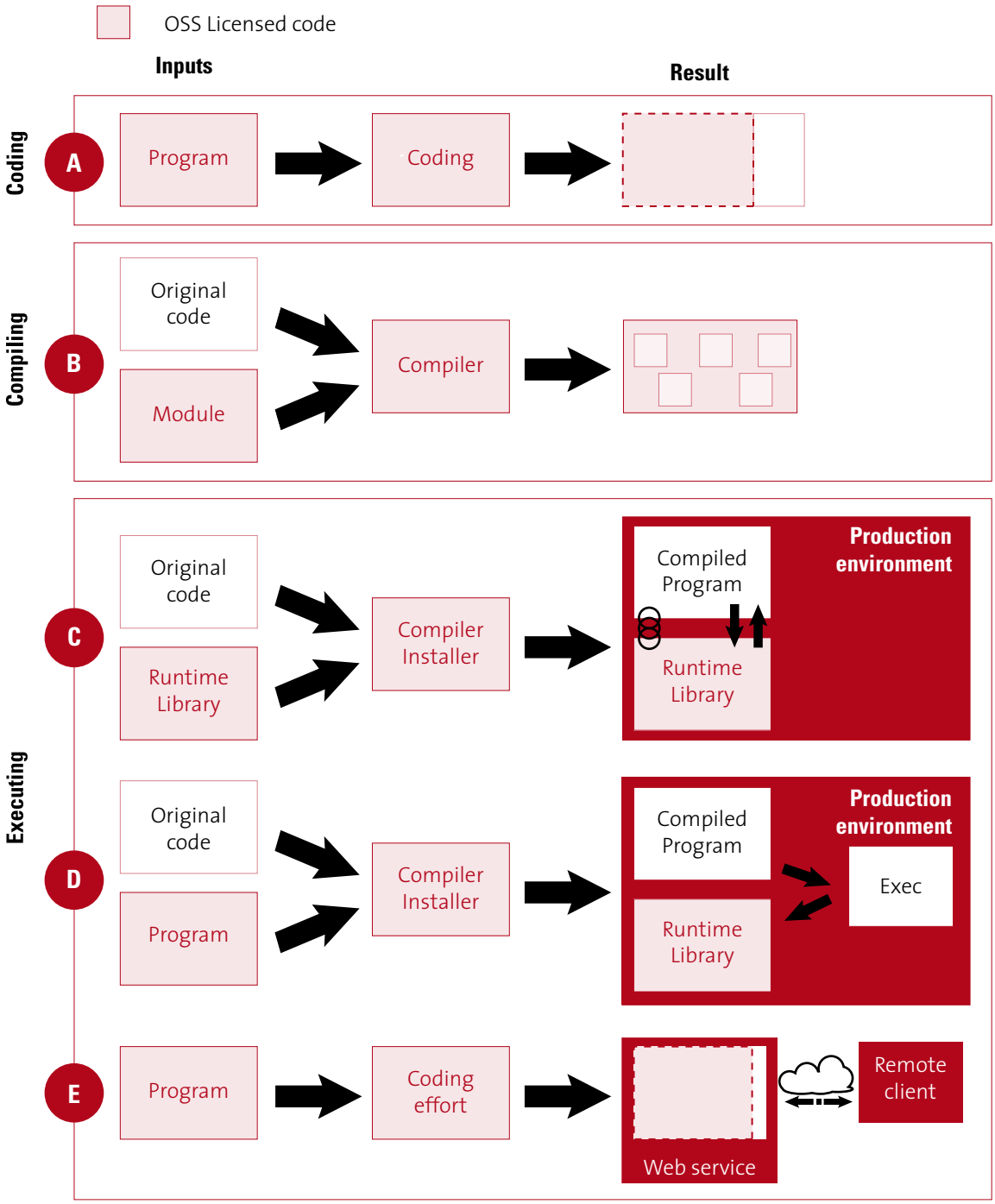
A more restrictive licence will tend to mean that agencies are restricted in the choice of licence they can use for the derived software. A highly restrictive licence will mean that all derived software must come under the same licence as the original source. A permissive licence allows for a greater freedom in how the original components of a derived work can be licensed.

Derived works

Even under a restrictive licence, reciprocity is not triggered until derived work is distributed. Combining open source code and other code is called deriving.

To clarify the range of circumstances in which modifying source code creates a derived work, Figure 3 illustrates five scenarios in which a software developer incorporates open source software into an agency's ICT project. The list of scenarios is not designed to be exhaustive; for further information on the definition of derived work, see Attachment B.

Figure 2: Scenarios for developing software with open source software (OSS) components



Note: The scenarios assume that the original open source software is licensed under a restrictive licence.

The five scenarios are:

- Case A—200 lines of source code are added to extend the software's functionality.

- Case B—The custom source code is compiled with open source software licensed code (a static library) into a single executable.
- Case C—A custom-built proprietary program interacts with a runtime library licensed under an open source software licence.
- Case D—This is similar to Case C. The first program invokes the second program via a system-level command, called an exec command, to a third party, namely the operating system.
- Case E—This is similar to Case A. The functionality of an open source software based solution is extended. These improvements are implemented as an online service over a network.

Open source software licensing risk model

A licence is a legal agreement between two parties.[15] Generally, the parties to an open source software licence are the copyright holder/licensor (typically, one or more software developers) and the licensee. In the case of the Licensing Risk Framework, the licensee is generally the agency.

Having chosen a type of open source licence that suits their purposes, the licensor has the legal right to enforce their rights as outlined by that licence. Any licensee that does not follow the terms and conditions of the licence risks being subjected to legal action. Historically, where the General Public Licence (GPL) and the Lesser General Public Licence (LGPL) are involved, the Free Software Foundation has instigated compliance action in the event of a reported breach.[16]

If all of the following conditions are met, reciprocity is triggered, and the agency must publicly release the source code of the derived product, regardless of whether the agency intended the trigger events to occur or abandons the project after the event:

- The open source software component is subject to a non-permissive open source licence.
- The open source software component has been used by the agency to create a derived work.
- The derived work has been then distributed or conveyed to others.

Once reciprocity has been triggered, the agency must make the source code of the derived work available and licence the derived work under the applicable open source software licence. Failure to release the source code under the relevant open source software licence would be a breach of the agency's legal obligations governing the open source software component of the derived work.[17]

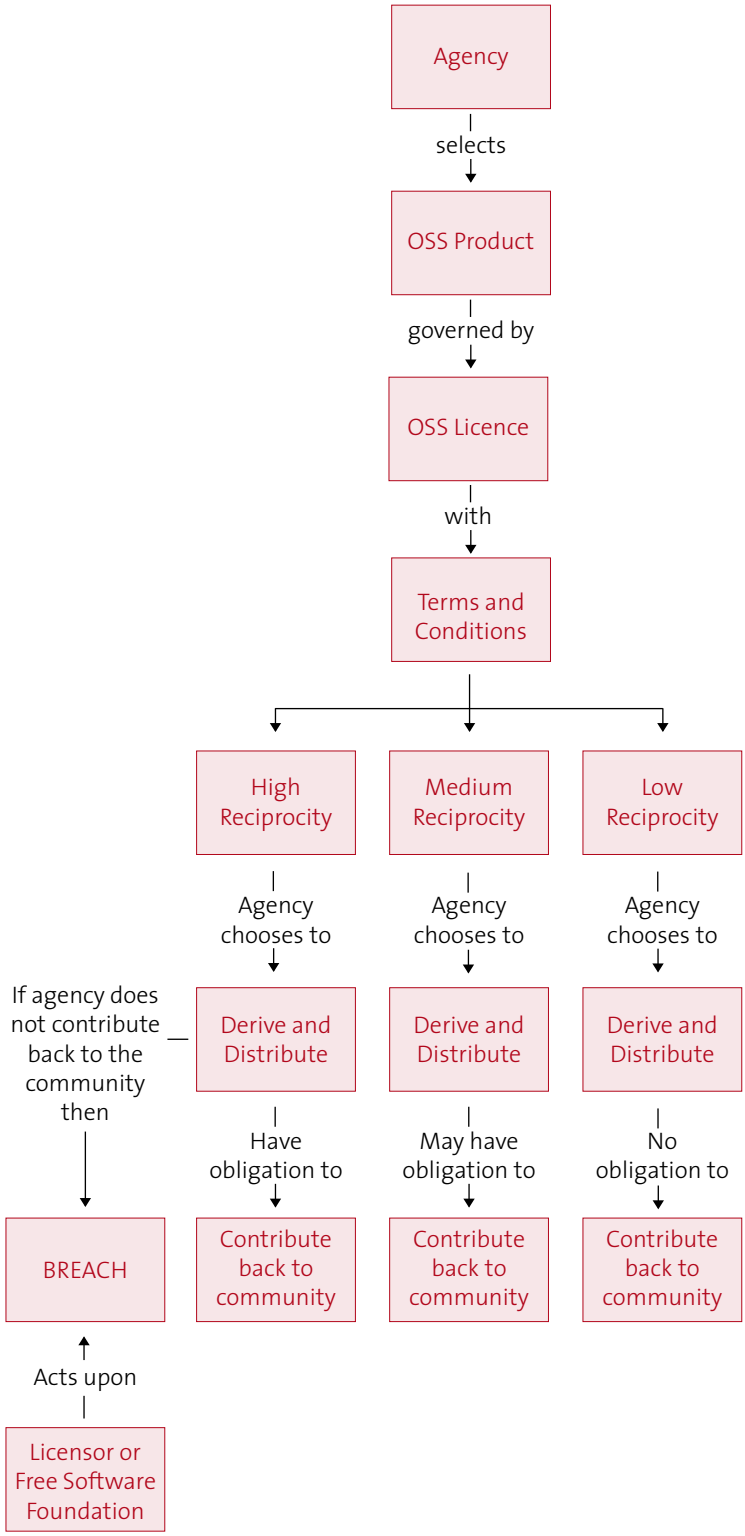
[15] Agencies should refer to their Chief Executive Instructions as well as the *Financial Management and Accountability Act 1997* to determine what delegations are required to enter into such arrangements.

[16] In many cases, open source software developers turn over their copyright ownership to the Free Software Foundation, the rationale being that the foundation is best positioned to enforce the licence should a licensing breach occur. The foundation has a proven track record in mounting successful legal action before United States and European courts to enforce the GPL and LGPL licences. Even where developers retain the copyright, the foundation may coordinate and fund efforts to enforce compliance.

[17] However, the relevant Australian Government agency does not have any obligation to maintain the software.

Having determined whether it is creating derived works and what constitutes distribution of those derived works (see attachments B and C), an agency should be able to determine whether its actions will trigger any reciprocity provisions. Agencies are recommended to seek legal advice regarding reciprocity. The agency can then use a model for working through the concepts involved in open source software licensing, such as the example in Figure 3, as part of its strategic approach to identifying and managing risks. **Attachment E** provides an example of a risk treatment matrix and assessment procedure, based on the fairly comprehensive approach adopted by the Australian Taxation Office.

Figure 3: Open source software licensing risk model



Key points

In developing a risk-management strategy, it is important for the agency to note that:

- The agency selects the open source software product.
- The open source software product is governed by an open source software licence.
- The licence comprises terms and conditions, which can be classified by the degree of reciprocity.
- The terms and conditions may specify obligations for agencies. Those obligations are triggered by the act of distributing derived work. More highly reciprocal licences will trigger the obligation more readily.
- Agencies that choose an open source software product with a high-reciprocity licence are obliged to contribute back to the open source software community any work that they derive from that product and distribute.
- Agencies that choose an open source software product with a medium-reciprocity licence may have an obligation to contribute back to the community if they derive and distribute.
- Agencies that choose an open source software product with a low-reciprocity licence will have no obligation to contribute back to the community even if they derive and distribute.
- Agencies must accept the licence attached to pre-existing open source software that they use. Agencies may choose the licence for open source software that they create, except where the creation is actually derived from other open source software. In that case, the degree of reciprocity in the terms and conditions of the licence for the pre-existing components may influence the choice of licence for the derived components.
- The Free Software Foundation has successfully acted on breaches of licence conditions on behalf of licensees.

PART 3:

Anatomy of licensing risk for project managers

This part:

- informs project managers of specific issues they need to consider when managing open source software projects
- reminds project managers of the necessity to consult with their legal department when considering open source software.

Use of the framework

It is difficult to generalise about open source software licensing. Therefore, project managers may find it hard to determine the best approach to identify and mitigate licensing risks. It is important for project managers to work in concert with technical staff to identify, scope and manage licensing risk throughout the development lifecycle. Project managers and technical staff should begin by reading the general overview of open source licensing risks of Part 1 of this framework.

A project manager should use the Licensing Risk Framework to arrive at informed decisions about:

- the appropriateness and merits of using component-based open source software rather than source code written in-house
- the likely risk of being obliged to make any newly created source code publicly available under any of the relevant open source software licences.

Key points

In particular, project managers should pay close attention to the following points when managing open source software projects:

- Ensure that the agency has copyright over any code that has been written for the project by staff, contractors or consultants.
- Where software development is outsourced, ask the vendor about their compliance procedures. In particular, agencies must ask what mechanisms they have in place to aid the agency with compliance. It is also advisable to ask the vendor if they will formally indemnify the Commonwealth in case the agency is found to be in violation of any licence.
- Search the codebase for code from external projects and note the licence used, the version of the code and the contact details (usually a website) for the code's authors. Keep this list up to date in the agency's source repository, and use the Open Source Initiative and Free Software Foundation websites to verify compatibility between licences.

- Choose an appropriate licence for the release of the derived code, making sure that it is compatible with existing licences and that it meets the agency's business needs. For example, the GPL may be appropriate if the agency wants others to contribute back any enhancements they make to the project.

Most importantly, the agency's legal department should be consulted before any decisions are made.

PART 4:

Attachments

The following attachments provide more technical guidance on the issues described in the framework.

- **Attachment A** outlines the assumptions on which the framework is based.
- **Attachment B** provides advice on how to identify a derived work.
- **Attachment C** provides advice on distributing or conveying a derived work.
- **Attachment D** gives links to different types of open source software licences.
- **Attachment E** provides an example, drawn from Australian Taxation Office documentation, of an open source software treatment matrix and assessment procedure.

Attachment A: Assumptions and risk factors

The Open Source Software Licensing Risk Framework is based upon the following assumptions:

- **Individual Australian Government agencies are responsible for managing their own legal risks, including those involving software development; however, the Commonwealth has an interest in ensuring that such measures are well informed.** The risks for agencies will vary mainly in the extent to which a particular agency uses open source software for software development.
- **Understanding the likelihood of enforcement is more useful to the analysis of open source software licensing risk than the abstract legal questions of the meaning of key terms such as derived works.** Although there are various expert opinions on open source licensing, currently, there has been few court cases.
- **Where there is any uncertainty about whether an open source software licensing right or obligation applies, an agency should adopt the most conservative position.** For example, there is some uncertainty about whether dynamically linking GPL-licensed code to agency-developed code creates a combined derivative work to which the GPL applies. The Free Software Foundation is emphatic that it does, while the Open Source Initiative thinks the situation is unclear. The conservative assumption is to treat the package as a combined derivative work to which the GPL applies.
- **The open source software community provides quality information sources that can provide sound guidance on many complex issues.** For example, in the absence of a clear judicial ruling to the contrary, it would be unwise to adopt a practice in opposition to advice given from the Free Software Foundation or the Open Source Initiative. Similarly, due regard should be had to information generally regarded within the open source software community as authoritative or reliable. This guide is about appropriate approaches to managing open source software associated risks, rather than about what may or may not be legally arguable. Consistent with the conservative approach above, this guide draws heavily on reliable information from the open source software community about what is considered best practice and the safe approach to any area of potential controversy.

Research undertaken by the open source software community suggests that many organisations using open source software fail to manage the end-to-end risk from the point of acquisition to an application's retirement. These risks are exacerbated by a number of factors:

- **Open source software awareness within agencies.** Typically, software developers are not fully aware of software licensing issues; consequently, they may inadvertently overlook licensing implications when using open source software. Furthermore, they may not realise that the risk relates to the type and specific provisions of the open source software licence governing each open source software component used in a project. There is also a level of uncertainty about licence interaction, scope change, change of use and other factors that might trigger unintended licence breaches, and the potential consequences of a breach.

- **Licence and source code management issues.** Open source software is very easy to download from the Internet and thus can be difficult for an organisation to track and manage.
- **Distribution of open source software–based solutions.** The risk relates the use to which the software developed under a project will be put and, in particular, whether that use might involve the developed software being distributed or conveyed outside the developing agency. Open source software is used in a number of contexts, whether solely for internal use or passed on, in whole or in part, for use by other agencies or even by members of the community. Certain licences require explicit actions if and when software constituting a derived work is distributed.
- **Software development scenarios.** If and when agencies engage external contracting houses to undertake software development, agency staff have limited control and day-to-day oversight over vendor practices, including the selection, use and management of open source software.

Attachment B: Identifying a derived work

Although there is no definitive guide on how to identify a derived work, the Free Software Foundation (FSF) and Linus Torvalds, the developer of the Linux kernel, are respected industry authorities on the subject.

The following quote is an excerpt from Heather Meeker's book *The Open Source Alternative: Understanding Risks and Leveraging Opportunities*.[18] This analysis may assist agencies in determining whether their activities will lead to the creation of a derived work. Agencies should still seek legal advice if it seems that a derived work may result from the agency's project.

> ***FSF: The boundary is identical to what composes a derived work under copyright law.*** *This position is based on the theory that the GPL can control only what is controllable under copyright law and also on some of the phrases in GPL2 used to define the scope of a work based on the Program.*
>
> ***FSF**: **Any linking (dynamic or static) to GPL code is a derived work within the boundary.** This position is based on the FSF FAQ on GPL2 and FSF comments on the GNU Lesser General Public Licence (LGPL). The LGPL explicitly allows linking to LGPL code, in which case linked code is outside the boundary. LGPL must be different from GPL or there would not be two different licences. Thus, linking brings code within the GPL boundary. In truth, the FSF position is not quite this clear-cut. However, many in the industry use this rule because it represents a safe position: If you assume that all linked code creates a derived work, you will likely not run afoul of the FSF position.*
>
> ***FSF**: **User space is outside the boundary of the kernel.** The FSF recognizes an exception for interaction between user space and kernel space. This special exception is expressly allowed in Section 3 of GPL2, which implies that the FSF considers this to create a derived work but has allowed an exception for it.*
>
> ***Linus Torvalds**: **User space is outside the boundary of the kernel.** Torvalds and the FSF are in agreement on this point: Each recognizes that interaction between user space and kernel space does not create a derived work.*
>
> *"The 'user program' exception is not an exception at all, for example, it's just a more clearly stated limitation on the ‹derived work' issue. If you use standard UNIX system calls (with accepted Linux extensions), your program obviously doesn't 'derive' from the kernel itself.*
>
> *Whenever you link into the kernel, either directly or through a module, the case is just a lot more muddy. But as stated, by default it's obviously derived—the very fact that you need to do something as fundamental as linking against the kernel very much argues that your module is not a stand-alone 'thing', regardless of where the module source code itself has come from."*[19]

[18] Heather Meeker, *The Open Source Alternative: Understanding Risks and Leveraging Opportunities*, John Wiley & Sons, 2008. Copyright © 2008 by John Wiley and Sons, Inc.
[19] Quote originally from Linus Torvalds.

FSF: Linking to standard language routines does not create a derived work. *The FSF views standard system libraries such as Java standard classes as an exception. However, if this is so, it is unclear why the standard C libraries (glibc) are licensed under LGPL.*

FSF: Software that interacts via communications protocols such as pipes and sockets is not a derived work.

The FAQ on the GPL2 (version 2) says:

"What constitutes combining two parts into one program? This is a legal question, which ultimately judges will decide. We believe that a proper criterion depends both on the mechanism of communication (exec, pipes, RPC, function calls within a shared address space, etc.) and the semantics of the communication (what kinds of information are interchanged).

If the modules are included in the same executable file, they are definitely combined in one program. If modules are designed to run linked together in a shared address space that almost surely means combining them into one program.

By contrast, pipes, sockets and command-line arguments are communication mechanisms normally used between two separate programs. So when they are used for communication, the modules normally are separate programs. But if the semantics of the communication are intimate enough, exchanging complex internal data structures, that too could be a basis to consider the two parts as combined into a larger program."

FSF: Software that interacts via an exec statement is not a derived work.
See FAQ quoted earlier.

FSF: Clean integration (e.g. data sharing). *The FSF's overall position on linked code has more to do with the 'intimacy' of integration between modules than their method of integration: dynamic link, static link, or otherwise. One useful approach is to focus on the spirit of the GPL rather than its letter, or the exact words of any extrinsic commentary. The spirit of the GPL is to allow licensees to freely use and modify code. The whole question of the border dispute arises because segregating code into linked files is a way to hide functionality in proprietary modules. Any programmer worth his or her salt can move any key functionality into a separate file and obfuscate it in binary form. This violates the spirit of the GPL. Therefore, a company considering distributing a proprietary module should always ask, 'How does this affect my licensees?' If the existence of the proprietary module means the licensee cannot effectively modify the GPL code, then the spirit of the licence has not been served. However, if the interface between the proprietary module and GPL code is simple, clearly described, and creates a true 'black box,' then the spirit has been served. (A black box means that the programmer modifying the GPL code does not need access to the proprietary code. In other words, the programmer does not need to see the workings of the proprietary module, so it functions as a black box; the interface is all that matters.) This approach is attractive both because it bears directly on risk assessment (by irritating the least number of licensees who want to modify the GPL code) and*

because it is based on sound engineering principles. Black boxes are good design. Every engineer understands that, without a complex explanation of circuit splits and copyright law.

Anything with a GPL header must be covered by GPL. *This statement is heard often, but what it means is not always clear. In a sense it is a truism, because the header is often the file that indicates licence terms. However, more often it means that any module linked to GPL code must be under GPL, because a link requires a header to connect the two linked files. Because this line of demarcation seems more an industry adage than a reasoned opinion, I leave it aside in favour of the more detailed cases discussed earlier.*

Attachment C: Distributing and conveying under the General Public Licence

This attachment provides an overview of what constitutes distributing or conveying under the General Public Licence (GPL).[20] This analysis may be used indicatively to determine what constitutes distributing under other open source licences. It is not legal advice.

Any code that is subject to the GPL may be freely modified by an agency for its own use. However, the agency may only convey or distribute GPL code that it has modified (within the meaning of the GPL) to another person if it complies with relevant obligations set out in the GPL. Generally, this includes the obligation to:

- license the modified code on the terms of the GPL
- ensure that the fact that original work has been modified is clearly notified in each changed file
- provide, or offer to provide, the source code of the modified work.

Definitions

There are currently several versions of the GPL, which use slightly different definitions. GPL 2.0 refers to distribution, but does not contain a definition of 'distribute'. GPL 3.0 uses the word 'convey'.

The GNU website provides a list of **Frequently Asked Questions** that gives guidance on the difference between conveying and distribution, as well as examples of both conveying and distribution.

Scenarios involving Australian Government entities

The following is a summary of scenarios in which a copy of a modified work subject to the GPL may be passed between government entities, indicating whether the scenario would be considered conveying for the purposes of the licence.

[20] The GPL is a free, copyleft licence that is © 2007 Free Software Foundation, Inc: http://fsf.org/.

Table 2: Scenarios for distributing or conveying derived work between government entities

Receiving agency type	Supplying agency type		
	Departments of state Departments of the parliament Non-statutory prescribed agencies	**Statutory prescribed agencies**	**CAC Act bodies[1]**
Departments of state Departments of the parliament Non statutory prescribed agencies	Not conveyed	Seek specific legal advice	Assume conveyed
Statutory prescribed agencies	Seek specific legal advice	Seek specific legal advice	Assume conveyed
CAC Act bodies[1]	Conveyed	Assume conveyed	Assume conveyed
State or territory governments Foreign governments	Conveyed	Conveyed	Conveyed

[1] Bodies established under the *Commonwealth Authorities and Companies Act 1997*: includes government business enterprises, statutory corporations subject to some CAC Act provisions, and Commonwealth-controlled *Corporations Act 2001* companies.

To provide a copy of a modified work subject to the GPL for use within the legal entity that made the modification is not considered conveying (or distributing) the modified work within the meaning intended by the GPL. Providing a copy of a modified work for use by a related legal entity will be considered conveying the modified work, unless the laws of the relevant jurisdiction prevent that. As there does not appear to be any Australian law that would override this position, it is best to assume that the provision of copies between related legal entities will be considered conveying those copies.

Commonwealth departments of state, departments of the parliament and non-statutory prescribed agencies are all part of a single legal entity. Commonwealth authorities within the meaning of section 7 of the *Commonwealth Authorities and Companies Act 1997* (the CAC Act), statutory corporations that are subject only to certain CAC Act provisions and Commonwealth-controlled companies are separate legal entities. Statutory prescribed agencies consist of a range of agencies that can have unique characteristics, so agencies should seek legal advice to determine how the law applies in each case (the safe position is to assume that conveying or distributing might be involved until such time as it is established that it is not).

The Department of Finance and Deregulation publishes a flipchart that lists Australian Government entities by type; the flipchart can be downloaded from **http://www.finance.gov.au/publications/flipchart/index.html.**

Attachment D: Open source software licences

Software may be offered under multiple licences. The most common occurrence of this is dual licensing, which can allow a product to be simultaneously licensed under an open source licence and a proprietary licence. It can also allow a product to be licensed under two open source licences in order to ensure licence compatibility when code from different projects is combined. In addition, this allows users to pick their preferred licence.

- The Software Freedom Law Center has an overview document that provides a primer to open source software. This overview document includes a summary of open source software licence types: **http://www.softwarefreedom.org/resources/2008/foss-primer.html**.
- A list of open source licences managed by the Open Source Initiative is available at **http://www.opensource.org/licenses**.
- A list of licences that qualify as free software licences as per the Free Software Foundation definition is available at **http://www.gnu.org/licenses/license-list.html**.
- Various products that agencies may use to monitor their compliance with open source software licences are available. The Linux Foundation offers such a program at **http://www.linuxfoundation.org/programs/legal/compliance**.

Attachment E: Example of the application of a risk management matrix for open source software

The following open source software treatment matrix and accompanying sample assessment procedure were taken from Australian Taxation Office documentation to provide an example of a possible way to mitigate the risks presented by open source licences. They may be adopted and altered by other agencies.

Matrix of licence types

	Restrictive	Restrictive-Hybrid	Permissive
Description	Applies to whole applications or to component software. Reciprocal obligations arise if a derived work is created and then distributed. The GPL is incompatible with certain other licences, so code mixing should be avoided in such cases.	Applies to whole applications or to component software. Accommodates the linking of source code libraries with proprietary code without derived works being created.	Applies to whole applications or to component software. Earliest licence type. Carries obligations related to labelling and attribution of creator's work on source code.
Treatment for internal use only	The use of unmodified applications is permissible. Prohibit the creation of derived works involving the software covered by this licence. Allow exceptions only if the preconditions listed in the Exceptions section are mostly/fully met.	Creating a solution based on either dynamic or static linking to open source software components is permitted but only if the preconditions listed in the Exceptions section are mostly/fully met.	Permit use of this software but ensure that client leaves any labelling crediting the original author (an 'advertising clause') intact.

	Restrictive	Restrictive-Hybrid	Permissive
Treatment if intending to distribute	Unmodified applications permissible. Prohibit the creation of any derived works. No exceptions allowed.	Derived works not permitted. Check the licensing terms to see whether static linking constitutes derived work. Creating a solution based on dynamic file linking to open source software components is permitted. Contact the licensor to confirm the agency's intended use and that the design of the solution is consistent with their understanding of the licence.	Permit use of this software but ensure that client leaves intact any warranty and labelling crediting the original author (an 'advertising clause'). Contact the licensor to confirm the agency's intended use and that the design of the solution is consistent with their intent in using the licence.

Sample assessment procedure

1. Classify the open source software licence as restrictive, restrictive hybrid permissive.

2. In the case of the restrictive or restrictive hybrid licence types, establish whether an intended use and proposed solution constitute a derived work. Note: This is dependent on the response to Step 1.

3. Determine whether the proposed solution involves distribution of a solution. The final determination will depend on the terms of the licence.

4. Consult the cell in the risk treatment matrix that corresponds to the results of the preceding steps. The treatments listed should be strictly observed and considered as default positions that constitute an acceptable level of risk to the agency. That said, a staff member or sponsor may assert that their circumstances are special, and that an exception to these treatments is necessary and warranted. The requirements in the **Exemptions** section below should be addressed before exemptions are considered.

Exemptions

No exemptions should be granted for any solution based on the *restrictive licence classification type* if the solution constitutes a derived work *and* involves distribution. This treatment will protect the agency's intellectual property by avoiding the obligation to make publicly available the source code for the entire solution, including any proprietary code developed by the agency.

An exemption to the nominated risk treatments should *only* be considered if *all* of the following conditions are met.

- The appropriate decision maker accepts the risk for managing software and licensing obligations while the software operates in the agency's computing environment. To be adequately managed, *all* of the open source software licences associated with a given ICT project should be traceable and auditable at any time.
- The client demonstrates that they have a clear understanding of the need for and commitment to complying with conditions of use for the software to address licensing risk. A simple checklist of expectations will be issued to the client at an appropriate time.
- The client or business owner accepts these conditions of use in writing and is prepared to be audited if and when that is deemed necessary.

Appendix 2: Links to other resources

Appendix 2: Links to other resources

Any reference to any specific commercial product, process or service by trade name, trademark, manufacturer, or otherwise does not constitute or imply its endorsement, recommendation or favouring by the Department of Finance and Deregulation.

Australian Government	Description
Architecture	The Australian Government Architecture (AGA) aims to assist in the delivery of more consistent and cohesive service to citizens and support the more cost-effective delivery of ICT services by government. Source: **http://www.finance.gov.au/e-government/strategy-and-governance/australian-government-architecture.html**
Commonwealth Procurement Guidelines	The *Commonwealth Procurement Guidelines* establish the core procurement policy framework and articulate the Australian Government's expectations of all departments and agencies subject to the FMA Act, and their officials, when performing duties in relation to procurement. Some Commonwealth Authorities and Companies Act 1997 (CAC Act) agencies are also subject to the guidelines. Details are available at **http://www.finance.gov.au/procurement/procurement-policy-and-guidance/procurement-policy-faqs.html.** Source: **http://www.finance.gov.au/publications/fmg-series/procurement-guidelines/index.html**
FMA legislation	The *Financial Management and Accountability Act 1997* (FMA Act) sets out the financial management, accountability and audit obligations of agencies (including departments) that are financially part of the Commonwealth (and form part of the General Government Sector). Source: **http://finance.gov.au/financial-framework/fma-legislation/index.html**
ICT customisation and bespoke development	The *ICT Customisation and Bespoke Development Policy* is a whole-of-government policy, which aims to increase ICT governance and reduce customisation and bespoke development within FMA Act agencies. Source: **http://www.finance.gov.au/e-government/strategy-and-governance/docs/ICT_Customisation_and_Bespoke_Development_Policy.pdf**
ICT procurement	A Guide to *ICT Sourcing for Australian Government Agencies* is a guide for Australian Government agencies that are dealing with ICT sourcing issues. Source: **http://www.finance.gov.au/procurement/index.html, http://www.finance.gov.au/procurement/ict-procurement/index.html, http://www.finance.gov.au/publications/guide-to-ict-sourcing/index.html**
Intellectual property principles	The Attorney-General's Department has published a *Statement of intellectual property principles for Australian Government agencies*, which includes guidance on software development. Source: **http://www.ag.gov.au/www/agd/agd.nsf/Page/Copyright_CommonwealthCopyrightAdministration_StatementofIPPrinciplesforAustralianGovernmentAgencies**

Australian Government	Description
Security requirements	The Attorney-General's Department has published the Protective Security Policy Framework, which outlines mandatory security requirements and links to protocols and guidelines. The Defence Signals Directorate has published many security resources, including the Evaluated Product List and the Information Security Manual. Source: **http://www.ag.gov.au/www/agd/agd.nsf/Page/Protective_Security_Policy_Framework** **http://www.dsd.gov.au/**
SourceIT	The SourceIT contracting framework is a legal framework established by the Australian Government to provide standard terms and conditions for the purchase of ICT goods and services. The SourceIT templates are designed to cater for simple procurement of hardware acquisition and support, licence and support of commercial-off-the-shelf software, licence of commercial-off-the-shelf software (without support) and ICT consultancy services. Source: **http://www.finance.gov.au/procurement/ict-procurement/contract-framework/sourceit-model-contracts/index.html**

Other Government	Description
EU: Guideline for Public administrations on Procurement and Open Source Software	This document contains guidelines for the procuring of Open Source Software in the European Union and template texts for tenders. Source: **http://www.osor.eu/studies/expert-guidance/guideline-for-public-administrations-on-procurement-and-open-source-software-2010**
Open Source, Open Standards and Re-Use: Government Action Plan	This document contains the UK strategy for open source software. Source: **http://www.cabinetoffice.gov.uk/resource-library/open-source-open-standards-and-re-use-government-action-plan**

Open source software products/ repositories	Description
Apache	A not-for-profit corporation that manages a number of open source software development projects. Examples include the Apache HTTP Server and Tomcat Java Servlet and JSP engine. Source: **http://www.apache.org/**
Drupal	An open source content management system. Source: **http://drupal.org/**
ELGG	An open source social networking engine that allows organisations to create their own social networking sites. Source: **http://www.elgg.org/**
Freshmeat	A catalogue of applications and other software. It includes a range of Unix and cross-platform applications, mostly distributed with open source licences. Source: **http://freshmeat.net/**
GNU	A collection of open source libraries, applications and developer tools. They are commonly used with a Linux kernel to give an open source Unix-like operating system. Source: **http://www.gnu.org/**
Joomla	An open source content management system. Source: **http://www.joomla.org/**
LibreOffice	An open source software office suite based upon OpenOffice. Source: **http://documentfoundation.org/** **http://www.libreoffice.org/**
Linux	A family of Unix-like operating systems, which provides the basis for interfaces, libraries and utilities to build complete operating systems. Distributions of Linux include Ubuntu, Debian, Fedora and Kubuntu. Source: **http://www.linux.com**
Moodle	An open source course management system that allows educators to create virtual learning environments. Source: **http://moodle.org/**
OpenOffice	An open source software office suite that is available in many languages and works on a range of operating systems. Source: **http://www.openoffice.org/**
Open Source Windows	A list of open source software that can be used with the Windows operating system. Source: **http://www.opensourcewindows.org/**
SourceForge	A large repository of open source software and development tools. Source: **http://sourceforge.net/**

Open source software groups	Description
Free Software Foundation	Group that manages directories of information for the free software community. It also provides licences for free software developers to share their code, including the GNU General Public Licence. Source: **http://fsf.org**
Open Source Industry Australia (OSIA)	The national industry body for open source within Australia. Source: **http://www.osia.net.au**
Open Source Initiative (OSI)	A not-for-profit corporation formed to educate about and advocate for the benefits of open source software and to build bridges among different constituencies in the open source community. It also provides licences for open source software. Source: **http://www.opensource.org**
Sydney Moodle User Group	The Sydney Moodle User Group (SMUG) provides ongoing support to individuals and organisations using Moodle as an e-learning platform. Source: **http://www.moodleusergroups.org/**

Appendix 3: Acronyms and definitions

Appendix 3: Acronyms and definitions

	Definition
Community source software	Community source software is a subset of open source software. Community source software is not made publicly available in the same way as open source software. It will only remain available to the specific community of developers who created the software.
Derived work	Derived work refers to a work including or based upon one or more pre-existing original works, such as a modified, adapted or extended version or a translation, condensation or any other form in which a work may be recast, transformed or adapted.
Distribute (or convey)	Distribute refers to the act of making a copy of software licensed under the GPL available to a third party (or the public in general). Other terms for the same concept include convey, propagate and make available to the public. In this framework, the word distribute is used to refer to the act of making relevant software available to anyone else in a manner that triggers an obligation to release the source code.
Dynamic linking	Dynamic linking is a mechanism available to one or more software programs during the operation of a computer system that allows them to request the services of (invoke) a separate executable (a library) to undertake more specialised tasks on their behalf. The benefit of this approach is the re-use of a library's functionality. The library is not included within the program.
Freeware	Freeware is free to use and distribute, but not necessarily free to modify. Users may use the software, but may not access the source code to modify it or for any other reason.
GPL	General Public Licence is a model licence for open source software
ICT	Information and communications technology
Licensing Risk Framework	*Open Source Software Licensing Risk Framework*
Open source software	Open source software (OSS) is software that is freely available to use, modify and distribute. Open source software is subject to specific licensing conditions that may obligate organisations to openly distribute any modifications.
Open standards	Open standards are a detailed, descriptive overview of a process, protocol or format. They are formulated through stakeholder consensus. They must be openly published and there should also be no legal or intellectual property restrictions. Open standards are generally defined by focus groups within standards organisations.
Proprietary software	Proprietary software is licensed for use under specific terms set by the copyright owner of the software. It usually involves an upfront free for use and may or may not include access to the source code. Proprietary software usually does not provide any right for the user to modify the software or redistribute it to any other party.

	Definition
Public domain software	Public domain software is not subject to copyright. There are no restrictions on the use, modification and distribution of public domain software.
Shareware	Shareware software is free to distribute. Its use is generally restricted in some way: for example, having limited functionality, working only for a limited time or including advertising in its interface. Generally, the source code is not available. There is usually an option to use the software under a proprietary licence that removes the restrictions.
TCO	Total cost of ownership