U.S. Department of Transportation
Federal Highway Administration
Research and Innovative Technology Administration

Policy Analysis and Recommendations for the Open Source Application Development Portal (OSADP)

www.its.dot.gov/index.htm

Final Report —June 2012
FHWA-JPO-12-031

Produced by the John A. Volpe National Transportation Systems Center
U.S. Department of Transportation
Research and Innovative Technology Administration
ITS Joint Program Office

Notice

Acknowledgements

The Volpe Center team would like to acknowledge the leadership of Walter During, P.E., of the Office of Transportation Management (HOTM) within the Office of Operations, Federal Highway Administration, U.S. Department of Transportation, in providing the guidance necessary to conduct the analysis that forms the basis for this document.

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA-JPO-12-031		

4. Title and Subtitle	5. Report Date: June 2012
Policy Analysis and Recommendations for the Open Source Application Development Portal (OSADP)	6. Performing Organization Code:

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. Performing Organization Report No.
Suzanne Sloan, Alan Chachich, Ingrid Bartinique, Linda Sharpe	

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. Work Unit No. (TRAIS)
U.S. Department of Transportation Research and Innovative Technology Administration John A. Volpe National Transportation Systems Center Cambridge, MA 02142	N/A
	11. Contract or Grant No. HW4A3

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
U.S. Department of Transportation Federal Highway Administration/ Research and Innovative Technology Administration's Intelligent Transportation Systems Joint Program Office 1200 New Jersey Avenue, S.W. Washington D.C. 20590	Policy Analysis Report, 2011-2012
	14. Sponsoring Agency Code N/A

15. Supplementary Notes **N/A**

16. Abstract

This white paper addresses the policy and institutional issues that are associated with the development of an open source applications development portal (OSADP), part of a larger research effort being conducted under the ITS Program's Dynamic Mobility Applications (DMA) program. It provides the first analysis of the types of policies that are typically established with an open source portal but are specific to the policy and institutional needs and risks associated with the DMA research objectives. This analysis is provided at this point to inform portal development efforts. However, policy analysis is typically iterative—policy decisions will impact technical choices which may influence the role and/or structure of the OSADP. As a result, policy will need to be examined and modified, as necessary, during the building and testing of the OSADP as well as during subsequent OSADP operations.

17. Key Words	18. Distribution Statement
Open source development, open source portal, dynamic mobility applications, governance, intellectual property, open source licenses, licensing options, procurement mechanisms, Federal policies, security, privacy, liability, connected vehicle	(Remove; Insert Information Here or leave blank)

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
N/A	N/A	154	N/A

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

Table of Contents

List of Tables

List of Figures

Executive Summary

This report analyzes the policy and institutional issues that are associated with the development of an open source applications development portal (OSADP), part of a larger research effort being conducted under the Intelligent Transportation Systems (ITS) Program's Dynamic Mobility Applications (DMA) program.[1] It provides the first analysis of the types of policies that are typically established with an open source portal but are specific to the policy and institutional needs and risks of the DMA program's research objectives, as they are known in 2012. This analysis is provided at this point to inform portal development efforts. It is expected that this policy analysis may impact technical decisions, which may influence the role and/or structure of the OSADP. As a result, the recommendations included in this report will need to be examined and modified, as necessary, during the building and testing of the OSADP as well as during subsequent OSADP operations.[2]

ES.1 Basis for Policy Recommendations

The basis for identifying risks and determining policy and institutional requirements are threefold:

- Technical documents that describe the OSADP technical requirements:
  - *A Concept of Operations (ConOps) – Dynamic Mobility Applications Open Source Application Development Portal* (August 2011)
  - A System Requirements document titled, *SyRS–Dynamic Mobility Applications Open Source Application Development Portal, version 3.0* (October 2011)[3]
  - An architecture and high level design document titled, *Dynamic Mobility Applications Open Source Application Development Portal* (May 31, 2012)
- Conversations with the DMA technical team and review of a range of additional case studies of other open source portals
- A companion report that identifies the critical policy issues: *Identification of Critical Policy Issues for the Mobility Program*[4]

[1] Information on the DMA program and research can be found at: www.its.dot.gov/dma/index.htm. Together with the Data Capture and Management (DCM) program, these two programs form the basis if the ITS Program's connected vehicle Mobility research (also known as the Mobility Program).

[2] An important caveat — there are areas of recommendations that will require further discussion with the technical team before implementation. In some cases, choices need to be made or further review by legal counsel is advised.

[3] Full documents titles are: Task 3.3: Concept of Operations – Dynamic Mobility Applications Open Source Application Development Portal, Final Draft Document, Version 3.3.3 – August 5, 2011; TASK 4.0: SyRS – Dynamic Mobility Applications Open Source Application Development Portal, version 3.0 – October 2011; and Dynamic Mobility Applications Open Source Application Development Portal – Task 6.1a: Architecture and High-Level Design and Task 6.1b: List of Requirements included in the Initial Architecture and High-Level Design (May 31, 2012)

[4] This critical issues white paper will be published in May 2012 with publication number FHWA-JPO-12-035.

ES.2 Institutional Issues and Risks and Policy Options

The OSADP is more complex than typical open source portals due to the expected diversity and breadth of the application bundles that are envisioned for development within the portal. This report explores the risks and describes options for mitigation (policies, technical designs, and other decisions) to guide development of an OSADP policy foundation that meets DMA program goals. Table ES.1 summarizes the key risks and lists policy options for resolving or mitigating risks as well as to form the basis for optimizing opportunities and use of the OSADP.

Table ES.1: Description of Key Risks and Options

Institutional Issues and Risks	Mitigating Strategies and Policy Options
Ineffectiveness of the OSADP/User Policies: • Policies lead to an overly restrictive or bureaucratic structure that does not support participants in working collaboratively toward goals (a greater risk when procurement is based on traditional contracted development that require strict deadlines and/or demand strict accountability which may hinder enhancement or innovation from outsiders) • Policies result in an overly unstructured or chaotic environment that, due to too little process, result in unusable products (a greater risk when using rapid, consensus-driven development with minimal definition of detailed system requirements and maximum communication among contributors) • Non-use or less-than-expected use of the OSADP • Little or no management of resources, unclear priorities, and little or no transparency on products, projects, and processes • User misbehavior and misconduct that drives away other developers • Overall higher costs and/or overall higher commitment of Federal staff and resources due to iterative processes associated with Agile or rapid development	➲ **Develop clear and appropriate policies, operating procedures, and rules at key levels that can be used for effective oversight. For the OSADP, oversight and governance will be applied at three levels:** • Program Oversight/Governance • Portal Oversight/Governance which can be: o Centralized o Decentralized o Federated • Project Oversight/Governance – "Benevolent Dictator (BD)" or Group Decision-Making Model ➲ **Ensure that the Portal Oversight team includes input and feedback from users to ensure that the Portal has user-based policies that account for risks** ➲ **Develop a management plan and a communications and outreach plan for the OSADP**
Lack of protection of Intellectual Property (IP): • Infringement on IP rights or patents with use of source code that contains IP, both known and unknown • Inability to offer open source applications as free and open software and/or inability to commercialize	➲ **Work with developers to ensure proper licensing of products/code.** Options include a range from true "restrictive" licenses that protect the open nature of the code or software in perpetuity to more permissive licenses that allow for commercialization of enhancements and modifications

Table ES.1: Description of Key Risks and Options (continued)

Institutional Issues and Risks	Mitigating Strategies and Policy Options
Procurement and/or Development Strategies are not Aligned with Program Goals: • The procurement mechanism may not be aligned with the goals for developing an application; and/or the procurement processes and contract terms are obstacles to rapid, iterative, collaborative development • The procurement mechanism results in stand-alone projects that prevent a broader range of developer creativity being applied	➲ **Use of appropriate procurement and development strategies. Options include:** • V Model Development • Agile Development • Open Source Development
Exposure of Personal Information or Violation of Privacy: • Exposure of personally-identifiable information (PII) because of datasets introduced into the OSADP from the RDE or from other external sources • Exposure of PII associated with the project managers, programmers, and collaborators who register for greater access within the OSADP and/or with its development projects	➲ **Implementation of privacy policies, controls, and technologies.** Options include: • Use of Federal policies for establishing controls • Investigation of Privacy Enhancing Technologies (PETs) with the OSADP
Exposure to Liability: • Product liability when an application fails due to errors or inaccuracies • Errors or inaccuracies introduced due to poor security, malicious actors, and/or malware	➲ **Options include use of accepted industry practices such as:** • Use of Federal policies for security • Quality control/testing of the applications before release into the repository • Inclusion of product warranties and terms of use that describe limitations to users
Inability of agencies to adopt the open source applications because of infringement fears, lack of support, or local laws.	• **Assurances of proper licensing** • **Facilitation of the development of a vendor community** • **Outreach to States through the National Association of State chief Information Officers (NASCIO) or other organizations to support adoption**
Lack of interest by software development community in using the OSADP	• **Implementation of strategies for attracting developers to the OSADP. Options include:** • Require use of the OSADP in all applications developed using Federal funds • Outreach to stakeholders with information about the tools and opportunities associated with the OSADP

ES.3 Description of Recommendations

Using the options listed in Table ES.1 and based on discussions with Mobility technical team members as well as a review of the OSADP documents listed on page 5, the following six broad recommendations are put forth as proposed strategies. Chapter 3-6 describe how the recommendations were derived and provide a list of steps and actions for implementation.

Recommendation 1: Establish Governance Boards

Three levels of governance and associated roles and responsibilities are recommended, as follows:

Program-level Oversight Team: Roles/Responsibilities

- ❑ Establishes the Portal Oversight Team
- ❑ Works with the Portal Oversight team members to establish policies for a range of policies and processes (i.e., security, privacy, acceptance of new project, user access, application release, managing licensing and IP, among others) and rules of operation. Collectively, these two groups decide where/how flexibility can be tolerated.
- ❑ Responsible for financial resource commitments and conflict resolution
- ❑ Responsible for decisions regarding upgrade and maintenance

Recommendation:

- ***In the Research phase, members of this group should include the Federal DMA program managers.***

The Program-level team is constituted first and establishes the policy foundation for and focus of the Portal-level team. Together, these groups define roles and responsibilities, policies and processes, and standard operating procedures. The Program-level team remains available for critical decisions, assurance of continued funding, conflict resolution, and oversight of the timeline and progress. **The OSADP is being developed and operated under a Federal program, therefore, Federal policies for security, privacy, data release, and others will apply.** If the OSADP transitions to use beyond Federal research, the ultimate owners/operators will take on these roles and responsibilities.

The Portal-level team implements and monitors the day-to-day operations. Their authority is derived from the Program-level team and includes the ability to decide on new projects or release of applications, based on the overall policy set by the Program. This team also plays an active role in making recommendations to the Program-level decision makers regarding portal changes, upgrades, maintenance, or other modifications.

Portal-level Oversight Team: Roles/Responsibilities

- ❑ Establishes standard operating procedures for users
- ❑ Develops criteria for accepting new application development efforts and for releasing new applications into the repository
- ❑ Oversees/monitors operations and supports Project Managers
- ❑ Responsible for security and risk monitoring and response plans
- ❑ Active management includes review of new projects, licensing, validation and verification/testing of applications before release into repository

Recommendation:

- ***In the Research phase, members of this team are expected to include the OSADP contracted managers who will develop the Portal, its SOPs, and criteria based on user and stakeholder needs and feedback.***

Project-level roles and responsibilities are determined through discussions with the Portal-level decision-makers and are based on the policy directions established with the Program-Level team. Chapter 3 describes the different levels of governance and provides more details on the recommendations. Appendix G provides a table that lists the roles and responsibilities throughout this report and recommends the personnel who might act in these roles.

Next Steps:

- ***Establish a small Program-level oversight team comprised of the Federal program managers. Have this group establish the Portal-level oversight consisting of the portal managers.***
- ***Have the Program-level team establish objectives and metrics for the Portal-level team to achieve for risk acceptability, daily operations, and decision criteria (policy foundation).***
- ***Have the Portal-Level team develop user rules, standard operating procedures, and project acceptance/application release criteria.[5] Document these policies and processes and incorporate into the Portal for transparent access for users.***
- ***Once established, have the teams define roles and responsibilities for ongoing operations.***

Recommendation 2: Form of Oversight, Decision-Making, and Governance

It is recommended that the Portal-level decision-making begin as centralized (Portal-level team makes all decisions) and transition to a "federated" structure once standard policies and operating procedures are in place (project teams will then assume decision-making and oversight efforts of monitoring for risks, establishing and implementing policies on openness and collaboration, developing licensing terms and restrictions, etc. that are specific to their projects). Depending on the structure of each project, the portal may eventually host a range of projects

[5] See Chapter 4 of the policy report, *State-of-the-Practice and Lessons Learned on Implementing Open Data and Open Source Policies*, FHWA-JPO-12-030, for descriptions of a range of user policies and controls.

that employ a range of decision-making and oversight models such as "benevolent dictator"[6] through group decision making models[7]. These structures are described in greater detail in Chapter 3.

Next Steps:

- ***Develop a transition plan and timeline for evolving Portal-level oversight from centralized to federated, based on user scenarios and anticipated risks (see risk tables in section ES.1).***
- ***Establish a set of procedures for the Portal-level team to follow when accepting a new project and working with the project lead(s) to tailor decision-making and oversight metrics in a manner that is specific to the project, its goals, and the level of new risks it introduces (for instance, risks in security, privacy, liability, or protection of intellectual property, among others).***

Recommendation 3: Develop a Comprehensive License Strategy

A comprehensive strategy for OSADP licensing will address processes and roles for applying "inbound" and "outbound" licenses; will review and determine the appropriate range of licenses acceptable to the DMA Program; and will establish processes for addressing exceptions.

Inbound licenses are determined by the owner of the intellectual property that is being brought into the OSADP. As part of both program-level and portal-level governance, processes will need to be established for reviewing the terms of inbound licenses and deciding whether those terms align with the DMA Program's open source approach (and thus whether the intellectual property will be allowed within the OSADP). An accompanying recommendation is for the staff that review the inbound terms to be cautious about accepting inbound products with patents. This is a highly controversial practice and is currently posing challenges to the US DOT and State and local-level transportation agencies in fully embracing ITS.

Outbound licenses or the license terms accompany the source code and/or application to the release repository. In the repository, the source code becomes available, under both the original inbound license terms and the new the outbound license terms—assuming new intellectual property has been added—for further enhancements. Similarly, applications are released for transition with their own package of licenses that guide user terms of use and commercialization. This report recommends three "outbound" license options:[8]

[6] This term is associated with open source development projects and refers to a decision-making structure in which one person is typically in charge of all decisions.

[7] This term is also associated with open source development projects. There are a range of models that describe different approaches to providing project members with voting rights or tallying votes based on different calculations. More detail is found in Chapter 3.

[8] These recommendations require review by the US DOT's legal counsel and Acquisitions officers.

- The MIT License (MIT/X11)
- The Berkeley Source Distribution (BSD-new)
- The Apache 2.0 license

The advantages and limitations associated with these and other license type is detailed in Appendix E. In choosing thee three options, our recommendation reflects implementation of a policy that is flexible and supports both:

- **Open source development**—development of applications that are either Incentivized through challenge grants or requested by project lead(s) who seek to have collaborative development (see recommendations below on procurement and development strategies)
- **Open source release** —release of new applications as free and open software or release and availability of the source code for further modifications and enhancements). This is likely to occur with projects that are funded with Federal dollars.

There is one additional option and that is to place works in the public domain. To do so, the first requirement is that the final application or product be free of any licenses on the original source code or other features; and that the developers and contributors agree. This agreement may be stipulated as part of a Federal contract that procures development and claims full ownership of the source code, application, and other documents. It can also be done by publishing the patentable information as '"prior art".

While open source development and open source release are aligned well with the overall goals of the DMA Program, there is moderate probability that accommodations will be needed for protecting inbound intellectual property. Hence, a range of licenses is recommended for the OSADP. It is also recommended that a process be developed for new project developers to work with the Portal-level board to petition for use of additional licenses that are likely to be more restrictive. Such a petition process is likely to involve the Program-level governance board as well as the Legal Policy team who will analyze the impact of introducing a more restrictive license option and determine if fulfilling such a request meets the objectives of the program. Chapter 4 describes the licensing processes and the various license options.

Next Steps:

- ***Establish a comprehensive license strategy by:***
  - ***Working with US DOT legal counsel to determine whether the appropriate level of open source intellectual property expertise can be made available to the DMA Program.***
  - ***In concert with the development of program-level governance, establish a set of processes and procedures that guide how and when licensing arrangements will take place.***
  - ***Ensure that the licenses and other considerations recommended in this report are aligned with US DOT policies.***
  - ***Based on these decisions convene a public webinar or workshop to describe the terms and receive feedback on whether such terms and processes will work for developer(s).***

Recommendation 4: Analyze Risks with Applications Procurement and Development Processes to Ensure Flexibility in the OSADP Design and Policies

Criteria for accepting new projects for the OSADP must include the identification of risks. Key risks include:

- **Intellectual Property Infringement:** These risks include conflicts with intellectual property particularly when patents are unknown or not stated upfront as a project begins. If intellectual property rights are known at the beginning of a project, inbound licensing is the appropriate mitigation. If no prior rights or terms of use are described, the Project-level governance board will need to work with the Legal Policy team to determine acceptance.[9]
- **Sensitivity of Code or Data:** These risks require that the OSADP provide greater protection for known intellectual property or sensitive data sets (those with some PII or those that can be linked with PII by associating the data with other datasets).[10]
- **Level of Adaptability Needed in Development:** These risks include cost and schedule risks that result due to the level of (or lack of) definition of application requirements. Greater adaptability in development (and thus potentially in OSADP policies) is needed when:
    - Application requirements are unknown and flexibility is needed to incorporate new requirements as anew information or ideas arise
    - A quickly evolving market or market demand requires a faster development process.
- **Level of Innovation:** These risks result from the complexity of an application that may require more iterative and longer development processes and/or suggest a higher need for more broad-ranging collaboration, and thus may require greater accommodations of OSADP policies.

Until the actual applications are known, a comprehensive risk analysis is not possible. At a general level, though, there are a range of policy and technical options for mitigating these risks, including a thorough understanding of the impact of choosing one procurement mechanisms and development process over another with any given application.[11] Chapter 5 provides descriptions of procurement mechanisms (traditional contracts, grants, and cooperative agreements as well as challenges and competitions) and development strategies (systems engineering or "V" model; agile development; and open source development). The chapter also

[9] NOTE: This issue is of broader concern for the entire connected vehicle program, and is under analysis with the Legal Policy team at this time —June 2012—and will result in a white paper that establishes the policies and guidance for DOT-funded projects. There are two potential issues that are being addressed: (1) Infringement of unknown patents/rights despite due diligence on the part of the project leads and the DMA team; and (2) Purposeful neglect to describe in order to exercise rights and demand compensation after adoption.

[10] There are technical mitigation options such as firewalls and added-layers of user controls; however, a policy decision is needed with the procurement decision regarding the management and access to such data by Portal managers and others outside of the project team.

[11] A report providing a preliminary analysis on the DMAs is under development.

describes the association between risks and the potential mitigation of risks with appropriate procurement and development strategies.

Next Steps:

- ***Develop a checklist of information that is needed from project leads before accepting a project for procurement or into the OSADP***
- ***Analyze the potential applications to describe their risks and choose appropriate procurement and development strategies***
- ***Work with the Legal Policy team to develop guidelines for accepting source code, data sets, or other software with no associated intellectual property licenses or terms***

Recommendation 5: Encourage Effective Use of the OSADP and Adoption of New Applications

Two risks that are associated with any open source portal are (1) the potential lack of interest by developers in using the portal and/or (2) the risk that applications developed within the portal will not be adopted for use.

With regard to use of the portal, there are two approaches that underpin success:

- Ensuring that the portal has transparent policies and useful tools
- Ensuring that developers are aware of the portal and its opportunities

The development of support for the applications after they are released is a critical element in adoption. While some States have laws or IT governing boards that provide disincentives against or prohibit adoption of open source applications and systems, more and more States and cities are turning to open source applications as a way of reducing initial investment costs and providing a more open and collaborative form of government.

To encourage adoption, particularly by the public sector, a strong vendor community that is capable of supporting maintenance, upgrades, and recovery (in the event of failures), is critical. Such a community is best developed simultaneous with the OSADP and requires transparency with applications development to establish the learning and training for their workforce.[12]

In both instances, a focused outreach effort to create awareness is important. The recent ITS Connected Vehicle Technology Challenge provides an example of the difficulties and successes associated with outreach to a development community beyond the transportation community. The lessons learned are captured in a document titled, *Connected Vehicle Technology Challenge: Communications Assessment* and contains new ideas for outreach.[13]

[12] A useful example is the commercial vendor community associated with support of LINUX systems.
[13] Available from the ITS JPO.

Next Steps for Attracting Developers and Encouraging Adoption:

- ***Ensure that OSADP policies and the portal itself support openness and transparency to the extent possible, given intellectual property concerns.***
- ***Ensure that the OSADP is well-organized and has a range of tools to support an active community.***
- ***Engage the user community throughout the software development process and potentially establish them as lead adopters.***
- ***Understand the challenges to adoption faced by the user community including State and local laws that may prohibit the use of open source software and/or policies by State IT governance boards who view open source software as unproven and costly. In particular, work with NASCIO on which States face such challenges.***
- ***Facilitate development of a vendor community by:***
  - ***Planning for and supporting development of a range of proper documentation that will guide the user.***
  - ***Planning for and engaging the vendor industry that will integrate the open source software into their service offerings, which will support the user community in installation and in receiving regularly scheduled fixes and maintenance.***

Recommendation 6: Future Transitioning of the OSADP

It is expected that if the OSADP were to transfer out from Federal funding and oversight, the owners/operators of the OSADP would inherit the roles at the Program-level and Portal-levels. To anticipate the policy support needed to transition the OSADP from Federal oversight, further research and analysis is needed.

Next Steps:

- ***Perform research to identify the value and uses of an OSADP:***
  - ***Survey a variety of types of organizations who might wish to assume ownership and operations and identify their purpose and potential uses as a means of deriving the value proposition***
  - ***Identify the factors and characteristics that are attractive to organizations other than the DOT, and identify the factors/characteristics that make the OSADP, in its current form, less attractive to potential new owners/operators.***

Introduction

This report analyzes the policy and institutional issues that arise with the development of an open source applications development portal (OSADP) for the Dynamic Mobility Applications (DMA) program. A recent Concept of Operations (ConOps) and System Requirements (SysRS) document for the DMA OSADP[14], have been developed, reviewed by stakeholders, and approved. Using this version, a policy analysis was performed to identify issues and risks; results for the basis for the draft policy guidance and options provided in this report.

This report is intended for multiple uses:

1.) **To create a structured, policy-focused basis for working with the DMA portal developers regarding policy implementation and making decisions about the policy options and recommendations**. As noted in a previous white paper that identified critical issues, portal development and operations will need policies for the following:[15]

   a. **Decision-Making, Governance, and Oversight structures** for the web-based portal, the applications development environment (ADE), the community environment, and the release repository. For the OSADP, decisions and oversight occur at three levels—Program-level, Portal-level (system), and Project-level with variances at each level for roles and responsibilities, user access and controls, rules for conduct; but with one comprehensive policy on language, security, and privacy.

   b. **Intellectual property protection and licensing options** that protect original creations work and provide mitigation against infringement and liability. There are two types of licensing activities:

      i. "Inbound licensing" or the license terms that come with the original source code, algorithms, and documents that both "seed" the portal and form the basis for new application development or enhancements of existing applications. The terms established with inbound licenses often carry through as part of the license terms when releasing the source code and/or commercializing the application.

      ii. "Outbound licensing" or the license terms that accompany the source code and/or application to the release repository. In the repository, the source code becomes available, under both the original inbound license terms and the new outbound license terms—assuming new intellectual property has been added—for further enhancements. Similarly, applications are released for transition with their own package of licenses that guide user terms of use and commercialization.

[14] *Task 3.3: Concept of Operations – Dynamic Mobility Applications Open Source Application Development Portal,* Final Draft Document, and Version 3.3.3 – August 5, 2011 ; *TASK 4.0: SyRS – Dynamic Mobility Applications Open Source Application Development Portal,* version 3.0 – October 2011; and Dynamic Mobility Applications Open Source Application Development Portal – Task 6.1a: Architecture and High-Level Design and Task 6.1b: List of Requirements included in the Initial Architecture and High-Level Design (May 31, 2012). This OSADP policy report was also informed by: Task 3.1: *Open Source Development Web Resources Scan Assessment Report,* February 28, 2011, SAIC.

[15] Identification of Critical Policy Issues for the Mobility Program, FHWA-JPO-12-035, p.11.

c. **DMA procurement and development strategies**. The decision on whether to use traditional contracted development or to use an alternative mechanism; and the decision on the development process for an application result in different needs, risks, and requirements for the OSADP. For instance, an agile development process may require more collaboration which requires broader membership access (and may bring along associated risks of exposure of sensitive or competitive data or may require more OSADP tools for managing and tracking changes). Or, a traditional contract may require greater levels of firewalls or stricter user access policies so that the development process is not influenced by outsiders or be subject to schedule delays. Understanding such risks and how to best apply procurement and development strategies will facilitate the OSADP's design, governance structures, and range of policies for accommodating a wider range of applications (initial review of the potential Federally-funded applications[16] suggests a broad diversity), collaborative projects, datasets and source code, programming languages, and/or license arrangements.,

2.) **To describe, for a broader audience of external stakeholders, developers, and vendors, the benefits and opportunities associated with the OSADP** as a means of facilitating use of the OSADP, adoption of the new applications, and development of a vendor support community to support the adopters of the applications.

3.) **To describe, for a broader audience of stakeholders (both internal to the US DOT and external), the identification, analysis, and decision path for implementing policy options.** This report then forms the basis for engaging stakeholders in discussion about each option's strengths and advantages or limitations and impacts. To support informed conversation, a set of Appendices is provided that includes materials that offer a more detailed background to the issues, options, and recommendations described in this report.

Three additional white papers inform the background for this white paper:

- ***The Role of Free and Open Source Software (FOSS) and Open Data in the ITS Data Capture and Management and Dynamic Mobility Applications Program,*** June 2011 (Draft).
- ***Identification of Critical Policy Issues for the Data Capture and Management (DCM) and Dynamic Mobility Application (DMA) Programs,*** FHWA-JPO-12-035.
- ***Industry Options***: ***State-of-the-Practice Policies and Lessons Learned on Open Data and Open Source,*** FHWA-JPO-12-030.

The structure of this white paper is the following:

Chapter 1: Policy on Open Source Approach summarizes the basis for the decision for and advantages of an open source approach.

[16] The DMA applications are actually "bundles" of applications—separate applications fused together and enhanced with more complex algorithms that utilize multiple sources of data in transformative ways. For further description of the bundles of applications, see: http://www.its.dot.gov/dma/pdf/MAP-HP%20V5.3%20F.pdf.

Chapter 2: Open Source Portal Policies and Risks is a short description of the typical portal policies and the risks and challenges with developing, hosting, and support an open source portal.

Chapter 3: Oversight and Governance Options describes Program-, Portal- (System), and Project-level options for decision-making, oversight, and governance.

Chapter 4: Protection and Use of Intellectual Property describes the role and considerations for choosing among license arrangements.

Chapter 5: Impact of DMA Procurement and Development Decisions presents four procurement strategies and three development paths and notes the implications and impacts associated with the options.

Chapter 6: Encouraging Portal Use and Supporting Application Adoption recommends steps to lay the groundwork for attracting developers and supporting user adoption.

Chapter 7: Summary and Next Steps gathers the set of options presented in previous chapters to identify the choices and discussions needed in the near-term to support Portal development as well as those needed in the longer-term that will support operations. After this summary, the chapter identifies remaining gaps and questions, and proposes a set of next steps.

A set of appendices provide background and explanatory materials:

Appendix A: Primer on Licensing Arrangements for the OSADP supports Chapter 4 and describes, in greater detail, the path for identifying a set of recommended licenses for the OSADP.

Appendix B: Conventional Software Licensing Terms Under US Law provides a set of definitions that are typically associated with license grants, use restrictions, warranties, indemnification provisions, and provisions limiting the liability of the parties. This background substantially informs the analysis and recommendations for Chapter 3.

Appendix C: How Software Is Programmed is both a set of definitions and a description of the programming function. The significance of these definitions is related to the licensing and the opportunity for a community user group to form a consortium to maintain an application, once released. If this opportunity is envisioned for an application or the DMA Program decides to encourage this opportunity, a legal framework will be needed that is dependent upon the license terms. This background informs Chapter 4 and Appendix A.

Appendix D: Common Restrictive and Permissive Open Source Licenses and How to Choose Among Them provides a greater level of detail on the license options but also on the complexity associated with combined and recombined open source software, which requires faithful observance of all licenses. This background substantially informs some of the considerations discussed in Chapter 3.

Appendix E: Additional Considerations for Program Level Governance Policy Decisions presents additional criteria for designating a project for the OSADP and releasing an application to the repository.

Relationship to other Connected Vehicle Mobility Policy Reports

This report is one in a series of six policy reports that describe and analyze the policy issues associated with connected vehicle mobility. The series includes:

- Two foundational reports that identify the critical issues and describe the best practices and lessons learned from government, industry, and academia*:*
    - *Identification of Critical Policy Issues for the Mobility Program, FHWA-JPO-12-035*
    - *State-of-the-Practice and Lessons Learned on Implementing Open Data and Open Source Policies, FHWA-JPO-12-030*
- Four reports that analyze the specific policy issues in context of the goals of the DMA and DCM programs:
    - *Policy Analysis and Recommendations for the Open Source Applications Development Portal (OSADP) (this report), FHWA-JPO-12-031*
    - *Policy Analysis and Recommendations for Development of the Dynamic Mobility Applications, FHWA-JPO-12-033*
    - *Policy Analysis and Recommendations for the DCM Research Data Exchange, FHWA-JPO-12-036*
    - *Privacy and Security Analysis and Recommendations for the DCM and DMA Programs, FHWA-JPO-12-032.*

1. Why Choose an Open Source Approach?

1.1 The Open Source Vision

In a 2010 Vision document, the ITS Program stated that a tenet of the research program is to broadly share Federally-funded foundational research to spur innovation and facilitate the rapid development of applications that can be commercialized and readied for broad deployment. In support of this commitment, the Vision recognized a rationale for applying an open source approach and for investing in a prototype open source development environment. With active investment in open source research and development activities, the DMA Program could: [17]

> *"...achieve an over-arching goal of maintaining a feasible evolutionary path from current technologies and practices to reach the desired end-state."*
>
> *"...promote the highest level of collaboration and preservation of intellectual capital generated from Dynamic Mobility Applications-funded efforts... [and] ...engage partners from academia and industry who may not be directly involved in funded applications development and testing."*
>
> *"Supporting an open source development environment for collaborating researchers requires both web-based tools as well as clear rules of engagement to support collaboration among Dynamic Mobility Applications-funded development activities"*
>
> *"Without a such an investment, it is envisioned that the public and private sectors will bear higher costs of uncoordinated, proprietary and duplicative mobility applications research and testing, higher costs for the commercialization and integration of non-interoperable or proprietary technologies and control systems, and slowed progress towards a less desirable and ad hoc end-state."*

The investment in an open source portal is envisioned to result in the following:

- Software and algorithms developed as a part of the program will be broadly available as part of the technology transfer element of the program. This wide provision seeks to engage the broadest range of public sector and private sector organizations.
- Modifications to basic algorithms and source code may be returned to the development environment under open source licensing, applications based on software or algorithms developed in the Dynamic Mobility Applications program may be commercialized and marketed.
- OSADP research activities will provide an opportunity to evaluate portal policies and processes to ensure that they support operations as intended throughout the course of the program.[18]

[17] The remainder of this page is a set of quotes and paraphrased content from the DMA Vision document located at: http://www.its.dot.gov/dma/dma_vision2.htm.

[18] This report is the first attempt to define the rules of engagement and to recognize success/evaluation factors.

1.2 Policy on Open Source Approach

Through discussion with stakeholders, past experiences with software development, and exchange of ideas, the US DOT "*...believes that the rate of innovation, the quickest path to deployment, and the greatest public good will be achieved by promoting a collaborative research environment where data sets, algorithms, and software are shared. To achieve this, the [program] is adopting an open data/open source approach for all data, algorithms, and source code developed using Federal...funds*." [19]

The US DOT has adopted five principles associated with using an open source/open data approach.[20]

1. *The Mobility Program is adopting an open source approach for all data, algorithms, and source code developed using Mobility Program funds.*
2. *The Mobility Program will allow use of pre-existing proprietary data and software in Mobility Program funded research, testing and demonstration projects. These proprietary elements can remain proprietary.*
3. *Cases involving Mobility Program funded changes to proprietary software will be handled on a case by case basis, as will co-funded projects where private sector partners provide partial funding for the project.*
4. *The Mobility Program intends to utilize one or more FOSS licenses that allow open sourced code to be incorporated into proprietary products without requiring that the modified software also be licensed as FOSS.*
5. *The Mobility Program envisions selecting or developing licenses that require licensees to provide attribution to the previous developers, acknowledgement that the material is provided as is, and that the developers are not liable for any damages caused by use of the material, and a requirement that any derived work is clearly identified as such, so that it is not confused with the original work.*

Federal Support for Open Source Policies

The decision to commit to an open source approach is supported by the Federal government's *Open Government Directive*[21] which promotes mechanisms, tools, and methods for more progressively supporting the principles of transparency, participation and collaboration in supporting citizens.

As an overall policy, implementing an open source approach to application development and building a portal to support this approach aligns with more than just the Open Government Directive. The approach also aligns with other connected vehicle research efforts on open data environments that are envisioned to provide new and multiple sources of open data in support of the new applications. It further fulfills a key public-sector stakeholder requirement—the need

[19] The Role of Free and Open Source Software (FOSS) and Open Data in the ITS Data Capture and Management and Dynamic Mobility Applications Program— June 8, 2011.
[20] Ibid, page iii.
[21] Memorandum 10-06 dated December 8, 2009, located at: http://www.whitehouse.gov/open/documents/open-government-directive.

to be provided with innovations in software and systems that are free or of little cost and thus need not compete with other infrastructure and operational priorities.

Two additional initiatives, the ***Federal Open Technology Report Card*** (2011) and the ***Implementation Plan to Reform Federal Information Technology Management*** (2011) also promote the use of open source technologies in the Federal government. The first document evaluates key indicators of open government, including open source technology practices. It also ranks 15 Cabinet departments and agencies on their use of open source, open formats, and open technologies. The latter document advises on the use of open technologies to achieve efficiency, transparency, and collaboration.

These documents and other examples are documented in the ***Open Source Development Web Resources Scan Assessment Report***[22], a research effort that identified the major capabilities and features that exist in portal technologies. The scan illustrates that open technologies: have reached maturity through successful implementations throughout the world; may result in added security given their tendency toward multi-level environments; may result in fewer coding errors and bugs, as multiple programmers review each others' work; and are likely to support greater cross-agency and cross-system sharing, if not also greater citizen participation.[23]

Recently, a number of governments and States have embraced open source development and open data initiatives as a means of cost reduction[24] and to provide more data back to citizens as a way to spur innovations. Some examples include cities in Colorado, the District of Columbia and San Francisco for open data portals[25], or Colorado's Department of Transportation website.[26]

To implement an open source policy, the Mobility team has developed a set of bounding statements that provide a definition for the level of openness for the OSADP—

- The first is a policy statement—*the Mobility team has decided that there will be no proprietary source code or PII data accepted into the OSADP*; that the source code that "seeds" the portal will be "free and open". Thus all resulting enhancements and modifications can be released with licenses that retain the open source terms of use.
- The second statement is derived from an analysis of the types of applications envisioned by the DMA program—there may be a need to accommodate some proprietary code or sensitive/confidential data as limited exceptions. For instance, the cooperative automated cruise control's source code is likely to be proprietary; the public sector "R.E.S.C.U.E.M.E." algorithms and applications will likely require sensitive and potentially PII-based data in the development and testing; and the freight applications will likely face the potential of starting with proprietary source code and/or working with

[22] Task 3.1: Open Source Development Web Resources Scan Assessment Report, February 28, 2011, SAIC.
[23] Ibid., p.19
[24] See a NASCIO sponsored report at: http://www.nascio.org/committees/clc/best_practices/gov-perfect-storm.pdf
[25] News release at: http://www.govtech.com/e-government/Colorado-IT-Officials-Launch-Open-Data.html
[26] New of their award is located at: http://www.nascio.org/awards/nominations2011/2011/2011CO9-Colorado%20Nomination_CDOT%20Website.pdf.

sensitive/competitive data. As a result, developers may need to request exceptions that will require owners of the code or data to agree to uses and limitations. Because it is not currently known whether this type of data will be needed, the privacy impact analysis that will be conducted on each mobility application will examine needs and determine whether such exceptions are valid as well as the conditions for permission.

This decision on the level of openness is critical for the OSADP as it impacts the types of risks, licensing strategies, user access policies, security, governance, and other policies that will be implemented with an operational portal. Importantly, these policies are not mutually exclusive. This report assumes that the opportunity to attract fully open source code to the OSADP will, in fact, be the predominant direction, but has built in flexibility to some of the policies to accommodate data and source code of a proprietary or sensitive nature.

1.3 What Elements and Technologies Comprise an Open Source Portal?

An open source portal is basically a web portal that allows for collaboration on projects through *"...community interaction...while respecting and maintaining effective control by the project's leaders over process, architecture, participation, and quality*."[27] As noted in the ***Web Resources Scan Assessment Report***, two key decisions in the development of a portal are:

- **What web resources will be made available?**[28] A minimum of resources typically includes:
  - Hosting options
  - Security
  - Storage and backup
  - Operations and maintenance
  - Configuration management
  - Bug reporting
  - Documentation
  - User accessibility
  - Collaboration
  - Recognition of contributors

  How these resources form an open source portal is based on the definition of use cases that are defined by user requirements and captured in the ConOps and SysRS documents. These technical requirements are addressed from a policy perspective in Portal (system) and project-level governance policies.

[27] *Beyond Code: Content Management and the Open Source Development Portal (Position Paper)*, Halloran, T.J., William L. Scherlis, and Justin R. Erenkrantz. Proceedings of the 3rd Workshop on Open Source Software (2003) , p. 69. Abstract located at: http://www.mendeley.com/research/beyond-code-content-management-open-source-development-portal/. Full paper located at: http://www.erenkrantz.com/Geeks/Research/Publications/ContentManagement.pdf.

[28] Task 3.1: Open Source Development Web Resources Scan Assessment Report, February 28, 2011, SAIC. p.4-7.

- **What web technologies will be used to develop the portal?** There is a wide range of technology options that can form the basis for an open source portal.[29] The decision on which one to use is based on how collaborative and open an environment is desired. Some of these technologies are better choices when desiring full member participation; others are better suited to encouraging use and adoption of the applications, but not allowing full participation. For the DMA OSADP, the defining factors should include an ability to:
    - Support a wide and diverse range of source code, algorithms, documentation, and applications development; thus a wide and diverse set of communities will likely create their own unique policies on access/openness, collaboration, and rules of conduct associated with their specific projects.
    - Host a variety of programming languages.
    - Establish user controls/user access at different levels, given the potential data sensitivity associated with some of the applications or the potential proprietary nature of some of the potential source code.

  From a technical perspective, the ConOps and SysRS establish that the OSADP is configured to accommodate this diversity. From a policy perspective, this report offers a range of policies to support this diversity and flexibility and address the resulting risks and challenges.

Choosing the Technologies and Features for the DMA OSADP

Early on, both the policy and technical development team recognized that the Department of Defense's Forge.mil[30] offered a Federal example for portal structure that could accommodate this wide array of needs. SAIC's ***Open Source Development Web Resources Scan Assessment Report*** presents a number of tables that compare Federal and non-Federal web resources and web technologies. Forge.mil consistently ranked as a portal that offered a wide array of capabilities, with SourceForge and JavaForge (both non-Federal) offering examples of open, non-restrictive portals that can accommodate a wide array of applications, programming languages and tools, and fostering collaboration among researchers. [31]

Importantly, being a Federal portal that has been in operation for a while, the DoD operators of Forge.mil were able to provide some insights into how and why decisions were made and how the structure could be modified to fit the DMA program needs. Two lessons learned stand out for consideration for the DMA OSADP:

- To develop environments and governing policies in a way where policies that support one or more areas of applications development do not comprehensively apply across the portal and potentially restrict other development communities; and

[29] See p. 6 in the Task 3.1: Open Source Development Web Resources Scan Assessment Report for a long, but not exhaustive list of options.
[30] http://forge.mil/
[31] See p. 4 and tables 1-3 on pages 6-12 in the Task 3.1: Open Source Development Web Resources Scan Assessment Report.

- To carefully consider the impact of using traditional government acquisition policies which are designed to facilitate procurements with a single person or entity and with a designated timeline as opposed to open source, iterative, and collaborative development. To meet the DMA program's goals of implementing an open source approach, the first step may be the engagement of the legal and procurement staff on questions and requirements noted throughout this document. Importantly, such changes in procedure and application of the Federal Acquisition Regulations (FAR) have been executed successfully with other projects, including some US DOT projects, and should be available to the DMA program in establishing the OSADP.[32]

The choice of which technologies to use to develop the OSADP and which features are offered are defined best by user requirements[33]. The ConOps describes the user needs, developed in a user workshop in January 2011, and user scenarios which translate into a set of features and form the basis of the technology choices.

From a policy perspective, the key considerations in choosing technologies are:

- How well the choices support the DMA program goals and the OSADP project goals. Key evaluation factors include:
  - Ability to maximize productivity;
  - Reduced costs and duplicative efforts;
  - Improved collaboration and greater innovation;
  - Increased agility/flexibility in development and more secure, error-free development; and
  - Development and enhancement of applications that can be offered to users under open source licenses in order to improve mobility, safety, and the environment. [34]
- How well the policy choices mitigate against the risks that are inherent with open source portals.

Further detail on these risks and options for preventing or mitigating them is provided in Chapter 2.[35]

[32] From discussions with the Forge.Mil support team at Hanscom Air Force Base in April 2011; based on an internal presentation to the US DOT on April 20, 2011.

[33] Defined in the OSADP Concept of Operations, 11-17

[34] Summary of the benefits sought from an OSADP from the OSADP Concept of Operations, p.8-9. Task 3.3: Concept of Operations – Dynamic Mobility Applications Open Source Application Development Portal, Final Draft Document, Version 3.3.3 – August 5, 2011 (publication number and web page forthcoming).

[35] The Critical Issues Summary white paper also listed security as a risk. For the purposes of this white paper, we recognize security as predominantly a technical risk that is solved through technical solutions, but acknowledge that strong policies supplement technical solutions and mitigate against the risk of malicious actors introducing viruses, malware, and/or errors into application code. Policies include ensuring that applications are tested and evaluated before implementation; examining user registration information; monitoring the user communities for behavior or products that are suspicious; and implementation and enforcement of user access control policies.

2. Risk Assessment

2.1 Risks – Program, Portal, and Application Risks

Chapter 1 noted that there are risks that occur for both the DMA program and with the OSADP. Table 2.1 below provides an organized and more detailed description of risks in the first column and describes policy options and mitigation strategies in the second column.

Table 2.1: Description of Key Risks and Options

Institutional Issues and Risks	Mitigating Strategies and Policy Options
Ineffectiveness of the OSADP/User Policies: • Policies lead to an overly restrictive or bureaucratic structure that does not support participants in working collaboratively toward goals (a greater risk when procurement is based on traditional contracted procurements that require strict deadlines and/or demand strict accountability which may hinder enhancement or innovation from outsiders) • Policies results in an overly unstructured or chaotic environment that, due to too little process, result in unusable products (a greater risk when using rapid, consensus-driven development with minimal definition of detailed system requirements and maximum communication among contributors) • Non-use or less-than-expected use of the OSADP • Little or no management of resources, unclear priorities, and little or no transparency on products, projects and processes • User misbehavior and misconduct that drives away other developers • Overall higher costs and/or overall higher commitment of Federal staff and resources due to iterative processes associated with agile or rapid development	➲ **Develop clear and appropriate policies, operating procedures, and rules at key levels that can be used for effective oversight. For the OSADP, oversight and governance will be applied at three levels:** • Program Oversight/Governance • Portal Oversight/Governance which can be: o Centralized o Decentralized o Federated • Project Oversight/Governance – "Benevolent Dictator (BD)" or Group Decision-Making Model ➲ **Ensure that the Portal Oversight team includes input and feedback from users to ensure that the Portal has user-based policies that account for risks** ➲ **Develop a management plan and a communications and outreach plan for the OSADP**
Lack of protection of Intellectual Property: • Infringement on intellectual property rights or patents with use of source code that contains intellectual property, both known and unknown • Inability to offer open source applications as free and open software and/or inability to commercialize	➲ **Work with developers to ensure proper licensing of products/code.** Options include a range from true "restrictive" licenses that protect the open nature of the code or software in perpetuity to more permissive licenses that allow for commercialization of enhancements and modifications.

Table 2.1: Description of Key Risks and Options (continued)

Institutional Issues and Risks	Mitigating Strategies and Policy Options
Procurement and/or development strategies are not aligned with Program Goals: • The procurement mechanism may not be aligned with the goals for developing an application; and/or the procurement processes and contract terms are obstacles to rapid, iterative, collaborative development • The procurement mechanism results in stand-alone projects that prevent a broader range of developer creativity being applied • Unclear specifications results in a procurement of the wrong application or system.	➲ **Use of appropriate procurement and development strategies that include Federal oversight and opportunities for stakeholder review.** Options include: • V Model Development • Agile Development • Open Source Development
Exposure of Personal Information or Violation of Privacy: • Exposure of personally-identifiable information (PII) because of datasets introduced into the OSADP from the RDE or from other external sources • Exposure of PII associated with the project managers, programmers, and collaborators who register for greater access within the OSADP and/or with its development projects	➲ **Implementation of privacy policies, controls, and technologies.** Options include: • Use of Federal policies for establishing controls • Investigation of Privacy Enhancing Technologies (PETs) with the OSADP
Exposure to Liability: • Product liability when an application fails due to errors or inaccuracies • Errors or inaccuracies introduced due to poor security, malicious actors, and/or malware	➲ **Options include use of accepted industry practices such as:** • Use of Federal policies for security • Quality control/testing of the applications before release into the repository • Inclusion of product warranties and terms of use that describe limitations to users
Inability of agencies to adopt the open source applications because of infringement fears, lack of support, or local laws	➲ **Assurances of proper licensing; use of standard, proven licenses** ➲ **Facilitation of the development of a vendor community** ➲ **Outreach to stakeholders to include them in setting policies and to facilitate adoption**
Lack of interest by software development community in using the OSADP	➲ **Implementation of strategies for attracting developers to the OSADP.** Options include: • Require use of the OSADP in all applications developed using Federal funds • Outreach to stakeholders with information about the tools and opportunities associated with the OSADP

It should be noted that while a risk may be primarily associated with a single issue area – for example, with intellectual property – its mitigation may require actions in other areas – for example, through procurement strategies.

2.2 OSADP Policy and Process Requirements

The OSADP is expected to comprise of four elements that represent the main technical functionalities of the portal. These elements are listed in Table 2.2 on the next page with a description of the types of policies and processes necessary to support each element and which are further defined in the following chapters. Figure 2-1 below is taken from the OSADP Architecture document and graphically represents the four key elements of the OSADP.

Figure 2-4: DMA OSADP System's Operational View

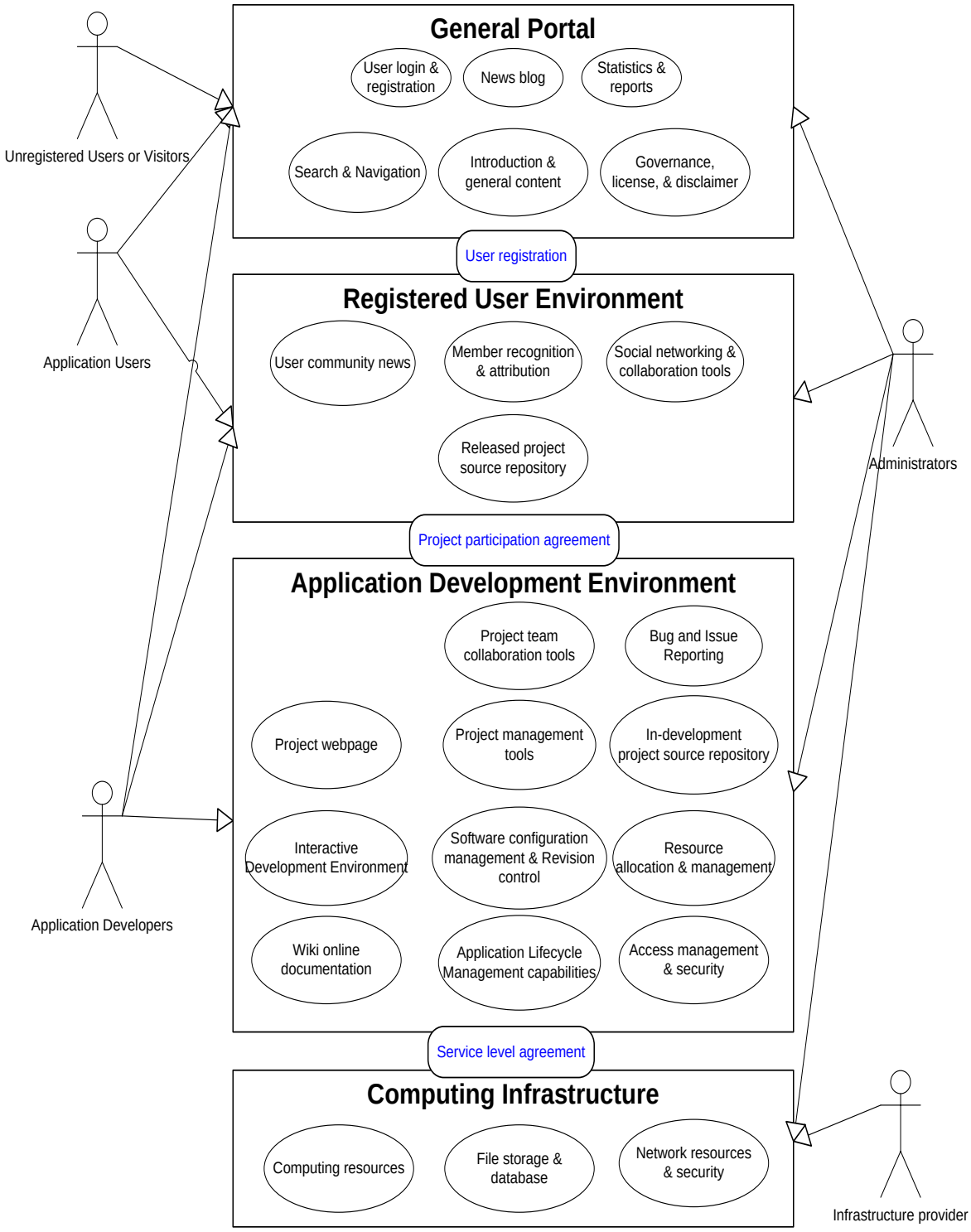


Table 2.2: OSADP Element and Required Policies and Processes

OSADP Element	Policies and Processes
General Portal: A **web-based portal as the primary access** for all Internet users	• Portal user agreement/ user registration is located at this level • Policy on what information is appropriate to be solicited from the user as part of the registration process is needed before registering users • Policy on the use and treatment of personal information gathered at registration is needed before registering users • Language policy must be transparent at this level • Program-level governance policies will set the overall rules that need to be transparent at this level; governance policies will also assign the roles and responsibilities to the Portal-level board and offer options for tailoring Project-level governance
Registered User Environment: A **community environment** that allows for communication, exchange, and collaboration among the registered users. This community is protected and user registration is required to access it. Includes a released project source repository which hosts released and licensed source code, algorithms, and associated documents and artifacts. It is the access point for users to obtain the completed files and the primary technical environment for a sustained community of users.	• Project-level oversight policies and rules of conduct are needed • User registration policies and user access policies must be transparent for users for this function • Policies that support transition of products to commercial use and/or public domain adoption • Policies that facilitate vendor community development and involvement in maintaining products after release • "Forking" policies that stipulate terms associated with "spawning" a new project with a new governance structure (described in Chapter 3)
Application development environment (ADE): An environment that enables software developers and registered users to participate in building mobility applications.	• A process for review and acceptance of licensing terms for "inbound" source code, algorithms, documents, and other intellectual property is needed for the ADE to function, including review of terms of use offered by the owner of the intellectual property that will be established as part of the project-level governance • A user access policy in line with the user groups identified in the logical architecture is needed for the ADE to function • Oversight policies and rules of conduct/operation are needed for the ADE to function • Contribution agreements for recognizing individual contributors participating in development and enhancement activities will need to be available as projects move forward • Licensing processes for completed files and applications that are "outbound" or available for enhancements or commercialization are needed along with clearly assigned roles and responsibilities for licensing is required for an effective repository

Table 2.2: OSADP Element and Required Policies and Processes (continued)

OSADP Element	Policies and Processes
Computing Infrastructure	• Portal policies are needed for security, protection of privacy, system performance, system operations, upgrades/repairs, recovery plans, user authentication, and firewalls associated with project governance needs.

3. Oversight, Decision-Making, and Governance Policy Options

3.1 Critical Questions

As noted throughout chapters 1 and 2, governance is a critical element for any open source portal. The OSADP ConOps and the SysRS documents identify the need for oversight and governance throughout. The objective of this chapter is present governance in a manner that:

- Responds to the set of questions provided in the Critical Issues Summary white paper;
- Aligns with the needs and requirements described in the ConOps and SysRS documents; and
- Clarifies what policies are needed and when, and addresses the risks mentioned in chapter 2.

To meet these three objectives, this chapter defines oversight and governance and identifies its role for open source portals (section 3.2), and then walks through each level to identify what steps need to be taken for implementation (sections 3.3-3.5). To address some additional questions not posed by previous reports, a section for Other Considerations (section 3.6) is included before the Summary (section 3.7), which synopsizes the next steps for action.

3.2 What is Governance and Why is it Important?

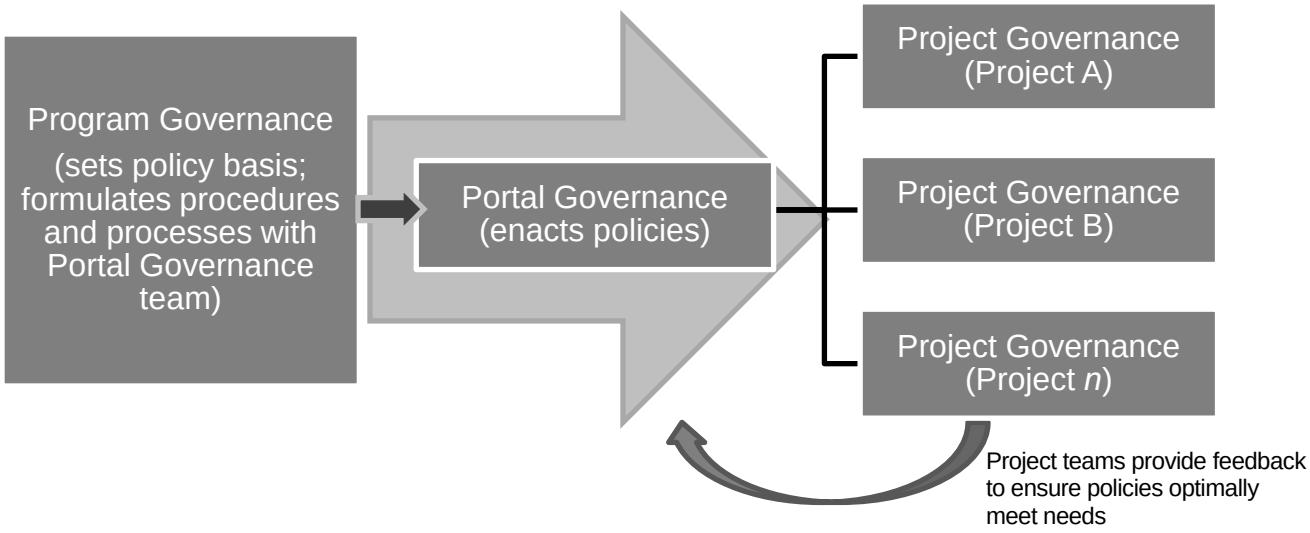
Governance provides a framework for decision-making, oversight, and management of any enterprise in which multiple individuals and organizational entities participate. A governance framework specifies the roles and responsibilities of participants; and the processes by which decisions are made. It defines the structure for collective production by defining end goals, allocating resources, setting priorities, monitoring progress, and determining the conditions for starting and ending programs, projects, and processes.

It should be recognized that the governance policies selected for the OSADP will influence how well the Mobility Program will be able to meet its goals. Governance policies will support the (system engineering) requirements of multiple, interrelated complex development efforts by varied actors, enforce the accountability required by the expenditure of scarce public funds, and allow maximal opportunity for innovative solutions to emerge. Governance policies will set the framework for communication among participants and to provide a structure that will foster the creation of products desired by DMA sponsors, by the transportation system operators, and by the general public.

OSADP governance policies are required at three levels:

- Program-Level Governance;
- Portal-Level Governance (or system governance); and
- Project-Level Governance.

Figure 3-1: Relationship of Governance Structures



Program-level governance determines the overall operations associated with the OSADP and the resources that will need to be committed in support of the work stream; establishes the governance structures for the Portal level; and determines the allowable range of governance structures for the Project level. Program-level governance will establish the roles and responsibilities at all levels and will identify the types of representatives that should fill them. At this juncture, this level maps to the current DMA Program management level, but requires the active assignment of specific roles and responsibilities.

Portal-level governance is concerned with customer satisfaction, ease of use, and user experiences; daily operations and functional reliability; and risk monitoring and mitigation. Portal-level governance functions are associated with content management and change control, access management, security and monitoring, and other Portal functions such as troubleshooting, managing downtime, performing backups and patches, and recommending and providing upgrades. The Portal manager plays an important role in implementing governance policies.

Project-level governance relates to the decisions on roles, responsibilities, and decision-making processes that are made by Project members themselves (the OSADP sub-community organized around the specific application bundle development project). The idea is that Projects–particularly those that are being run as open source–are self-governing. The ideal may be tempered by decisions related to Project governance that are made during the Federal procurement process. In particular, the vision for the OSADP is that it can host projects developed under varying management and governance procedures, including traditional structured development models. *Our analysis indicates that, for the first-generation development of the original Mobility applications, no project using Portal resources is likely to take the form of the pure open source communities such as the Linux community. Only after applications are released to the repository and user community's form and may want to enhance the original application, will pure open source project governance apply.*

In the case of procured projects, software development models will not include open source development; roles and responsibilities will reflect corporate policies and job descriptions, if the project involves a single developer. If the project comprises a team formed of a prime contractor and sub-contractors, or a team made up of members from more than one entity, the roles and responsibilities will reflect the terms of the contract agreement among them. Projects that are the result of a challenge AND follow the open source development model may be largely but not entirely self-governing, in that decisions at the Program level may insert roles for federal team members and responsibilities in relation to decision making.

Community building at the Portal- and Project-levels will occur through the OSADP mechanisms provided to support collaboration (community chat rooms, email, etc.) and recognition (contributor agreements). It is at the Project-level that each project's community will make decisions about who can use which mechanisms and, if appropriate (i.e., for non-procure projects), how to recognize contributors (informs the content of the contributor's agreement).

The challenge is to set up an overall program- and portal-level governance framework that will accommodate multiple models of project-level governance and that does not assume that project-level governance must conform to any one model.

This chapter emphasizes that while some issues related to governance are specified in the licensing agreements that will drive how applications can be used, governance is concerned with broader issues of decision-making and oversight.

3.3 Program Level Governance Policy Decisions

Program-level governance sets the initial conditions for the OSADP to begin accepting users and projects in the context of the overall success factors of the program. Program-level governance also defines the institutional context in which the OSADP will operate.

As familiarity with the goals of the Mobility Program is essential to productive and efficient high-level governance decisions, and with the wide variety of complex of applications and application bundles, we recommend that a subset of the members of the current Federal management structure for DMA Program should be designated as the program governance policy-making group for the OSADP. With that designation, the members will need to make preliminary decisions in support of the OSADP development. Early-stage governance decisions include:

1. The **structure of the Portal-level team** that will govern day-to-day operations of the Portal. The structure includes **who will perform** Portal-level governance and **how that team will make decision**s. Our recommendation is that this structure accommodate two distinct areas of responsibility:
    - **Policies related to the use of the Portal** by the public, developers, Federal managers, and transportation system operators; and
    - **Policies related to the technical oversight of the Portal infrastructure service provider.**

2. **Policies that will set the decision rules for granting access permissions to users** based on their roles within the four tiers of the OSADP logical architecture.[36] The decision rules should build in flexibility so that decisions on access to the second and third levels (the Registered User Environment and the Application Development Environment) can be made on an application-specific basis.
3. **General decision criteria for releasing an application to the Repository**. This report assumes that all applications that are completed and successfully tested will be released through the repository using outbound open source licenses packaged with either open source or other inbound licenses. Additional release policies are needed and primarily relate to testing:
    - What constitutes acceptance testing; [37]
    - What testing capabilities and resources should be included in the ADE tool suite[38], and whether acceptance testing should use those Portal capabilities or another, independent alternative;
    - Who should design the acceptance testing; who should conduct it; who should review the results; and who has authority to say whether the application has passed the test;[39]
    - Which applications should undergo field testing and whether product acceptance should be based on bench-top validation testing alone, or instead should include field testing results. (Additional considerations for developing governance decisions are included in Appendix E.)
4. **Policies that determine the general decision criteria for designating a development effort as a project on the OSADP.** The OSADP ConOps assumes that for all projects awarded through procurement, "...*all engineering and system development will take place in the OSADP environment*."[40] Our analysis indicates that this statement will be true only if Program-level governance decisions make it so. (Additional considerations for developing governance decisions are included in Appendix E.)
5. **The overall strategy for managing licensing and intellectual property**, as discussed in Chapter 4 of this report.
6. **Definition of a plan for attaining Federal inputs from and/or aligning OSADP practices with Federal policy** with US DOT counsel, US DOT Procurement Office, and the US DOT's CIO, at a minimum. These inputs would include, for example:

[36] As illustrated in Figure 2 the most recent version of the OSADP ConOps p.4.
[37] The 9/4/2011 draft OSADP Requirements document speaks in an operational to the testing of modules [US3.7, p. 56] but does not describe testing of the application once the modules are integrated, or acceptance testing, as necessary for release to the Portal. [Also see Figure 9, p. 53.]
[38] The 9/4/2011 draft Requirements document refers to "a suite of development applications tools such as software compilers for various programming languages, software version control, bug and issue trackers, and release management tools." (p. 11)
[39] The 9/4/2011 draft OSADP Requirements document assumes that the evaluation of test results falls to the Project Manager, rather than the DMA Program. UC3.8, p. 56.
[40] Final draft OSADP ConOps, p. 6.

- US DOT counsel's determination of the appropriate terms for outbound and inbound licenses; and
- If using challenge grants in application procurement, US DOT Procurement Office's involvement in clarifying the options and requirements associated with deciding which challenge authority would permit the DMA Program make participation in a challenge permissible for non-US citizens, corporate entities, and non-permanent residents.[41]

7. **Definition of standards for user conduct.** In particular, the Forge.mil User Agreement[42] provides an example of how a Federal entity identifies its authority to set boundaries by identifying the legal and procurement documents that provide the right to set such boundaries including conditions on use of service, monitoring, appropriate conduct, obligations of the user regarding representation (warrants), limitations on liability, and other conditions of use.
8. **Definition of a process for monitoring the evolving requirements** of stakeholder communities for potential changes to the governing framework.
9. **Development of plans to coordinate management and financial resources** among the DMA application bundles.

With these decisions, program-level governance sets the initial conditions for the OSADP to begin accepting users and projects in the context of the overall success factors of the program.

3.4 Portal-Level Governance Policy Decisions

Once the conditions for the start-up of the OSADP have been set at the program level, the portal-level governance structure will need to be established and in place for the launch of OSADP operations. Portal governance policies structure how the portal meets the needs of the users, the operational needs, and the business needs. Portal governance determines how website developers, administrators, interface designers, content creators, business marketing, portal users, and IT support will interact to ensure the efficient and successful operation of the portal.[43] Portal-level governance includes both governance policy-making and the day-to-day operations management that will enable all of the parts and stakeholders to work productively together.

As noted on the previous page, determining the make-up of the portal-level governance and operations group is one of the first decisions of the program-level governance policy-making

[41] With regard to the rules and requirements for challenges, see the OMB Executive Memorandum *Prize Authority in the America COMPETES Reauthorization Act*, at http://www.cio.gov/documents/Prize%20Authority%20in%20the%20America%20COMPETES%20Reauthorization%20Act.pdf; OMB Memorandum M-10-11, *Guidance on the Use of Challenges and Prizes to Promote Open Government,* at http://www.whitehouse.gov/sites/default/files/omb/assets/memoranda_2010/m10-11.pdf; and E. Halchin, Other Transaction (OT) Authority, Congressional Research Service, 1/29/2010.

[42] http://forge.mil/UserAgreement.html.

[43] State-of-the-Practice and Lessons Learned on Implementing Open Data and open Source Policies, June 2012. FHWA-JPO-12-030.

body. Because there are multiple application bundles with varying requirements for development we recommend that Program level decisions will be made by a body with representatives from more than one application bundle.

We recommend that portal-level governance policies be developed for two distinct areas of the Portal:

- Users and content areas. These include the first three tiers of the OSADP logical architecture defined in the ConOps: Public Portal (website), Registered User Environment, and Application Development Environment.
- Technical infrastructure which includes the last tier: Computing Infrastructure.

Portal governance options reside along a continuum from fully-centralized to fully-decentralized; most portals reviewed in this analysis appear to opt for a compromise, sometimes called "federated." Descriptions of these three governance models follow:

Centralized

The centralized portal governance model follows a typical top-down organizational structure, where one person or small group controls all final decisions, sets rules, and enforces processes. This was once the dominant model for businesses of many types, although it has fallen out of favor in large organizations due to the resource required to sustain it, and the negative impact that a single individual can have.

Decentralized

The decentralized portal governance model has no central command structure. All rules and decisions are made collectively by self-defined groups with common interests. This model offers freedom, but provides little consistency, guidance, or support.

Federated

The federated governance model retains a strong central entity, but with numerous loosely connected entities beneath it. In this model the central authority controls only those roles and process that benefit all stakeholder groups (e.g. portal policies and procedures).The smaller units are then provided the freedom to determine their own needs, structure and design. Most portals follow use some form of federated governance structure[44].

It is recommended that the Portal-level governance begin as centralized (Portal-level team makes all decisions) and transition to a more "federated" structure once standard policies and operating procedures are in place (project teams will assume governance/oversight efforts of monitoring for risks, establishing and implementing policies on openness and collaboration, developing licensing terms and restrictions, etc. that are specific to their projects).

[44] Roth, Craig. *Website Governance: A How-to Guide.* http://www.craiglroth.com/Opinions%20In%20Depth%20-%20web%20governance.pdf and Behl, Pardeep. *Winning Strategies for Portal Governance*. http://www.ibm.com/developerworks/websphere/library/techarticles/0904_behl/0904_behl.html.

It is recommended that, initially, a centralized governance approach be followed for the user and content tiers of the portal (General Portal, Registered User Environment, and Application Development Environment) but that the centralized approach transition to a federated approach as the OSADP matures, policies are proven successful, and the user communities assume greater control over operations of their project development communities. A centralized approach, however, is recommended for the technical (computing) infrastructure because it results in greater control for the Portal manager for taking actions that minimize risk to security and operations. This is based on lessons learned from such entities as SourceForge, for example, which has experienced several attacks, the most notable of which in early 2011 prompted the site to invalidate all user passwords as a precaution.[45] Although the site experienced no data loss from the attack, the incident did cause user inconvenience in the form of manual password resets and service disruptions associated with verifying data integrity. Frequent hacking or denial-of-service attacks could erode developer confidence in the portal, causing decreased usage and interest. Thus, a strong, centralized approach makes sense for the Computing Infrastructure element.

Because the open source model for software development is a relatively new direction for Federally-sponsored software development and because some of the practices are unfamiliar to Federal program and procurement managers, the collaborative development of portal- and project-level governance processes will be critical in gaining acceptance and in maximizing the success of the projects. These processes should be written with the foremost goal of encouraging maximum productive participation and collaboration in the portal's space by all stakeholders, but especially by developers, Federal program staff, and DMA users.

Portal-level governance will specify:

- The roles and implementation mechanisms that will be required to manage the portal throughout development and operations and updates. Roles and responsibilities are detailed in Appendix G. Note that these functions might be served by one individual or that one individual might serve in multiple roles.
- Processes that are identified, defined, and mapped to defined roles, include:
    - Prioritization and Release strategy at both the Portal-level and at the Project-level
    - Site Brand Management and User Experience
    - Communication and Rules of Engagement
    - Site Policies and Compliance
    - Site Taxonomy
    - Content Management[46]
    - Documentation.

As not all potential participants may be cooperative or benign, the definition of the processes must be done with consideration of how to reduce risks from malicious or incompetent actors, from infringement of intellectual property rights, or from other risks outlined in Chapter 2. The

[45] http://sourceforge.net/blog/sourceforge-attack-full-report/

combination of the need for openness with the need to mitigate risks means that considerable effort will be needed to produce the portal-level governance policies for security. One lesson learned from the open source community is the value of having a multitude of OSADP users watch for and report suspicious activity. Thus, the policies associated with the registered user environment and ADE should encourage observation and reporting in addition to having a strong, centralized governance of the computing infrastructure; and tools should be provided to assist in the reporting. As will be noted later in this document, because the OSADP is initially launched with Federal ownership and financing, the OSADP will need to adhere to the NIST guidelines for security and privacy, including the step of having the OSADP certified and accredited by the FHWA's CIO.[47] The NIST guidelines provide a roadmap for the OSADP technical teams and portal developers.

3.5 Project-Level Governance Policy Decisions

Project-level governance is a framework for decision-making and management of a project. While program-level governance determines the overall rules of engagement for the OSADP, individual project-level governance will tailor and enhance rules, roles and responsibilities to match the needs of the particular project.

The governance structure of a project describes the roles and responsibilities of the participants, with a particular emphasis on how decisions are made. In the world of open source software development, project governance establishes the rules by which collaborators may contribute to a project, how contributions will be evaluated and accepted/rejected, and how disputes will be resolved. Open source project governance tends to encourage consensus decision-making through constructive debate. There are two predominant forms of open source project-level governance:

Benevolent Dictator (BD) Model

One person is in charge of all final decisions. The BD may choose to delegate some authority to others, but retains final approval and veto authority. This approach is most common in small projects where one team member has a much greater understanding of the project than others.

Group Decision-Making Model

All final decisions are made by the group. Decisions can be made through a variety of mechanisms including: simple majority vote, consensus, and lazy consensus (where not voting is counted as a consenting vote). [48]

[47] Key guidelines are located at: http://csrc.nist.gov/publications/PubsTC.html#Certification%20&%20Accreditation and http://csrc.nist.gov/publications/PubsTC.html#General%20IT%20Security. A forthcoming paper on Mobility privacy provides greater detail on which NIST guidelines apply at the different stages of a system's lifecycle.

[48] The review performed for this report identified that a common mechanism for voting is the Apache Software Foundation scoring mechanism where "yes" votes receive a "+1" and "no" votes receive a "-1." Some projects choose to allow any group member to veto (consensus requirement), others set requirements for the total score that must be achieved for a vote to pass using a simple majority (e.g. "+1" passes) or some modified majority (e.g. "+3" passes). Another common practice is to allow all votes to pass unless someone vetoes the proposal (lazy consensus). There is always a risk of veto abuse in group decision-making. Therefore, many projects require that vetoes be justified and encourage voting as the

The projected operation of the OSADP will permit projects using various development models to be resident in the Application Development Environment. Depending on the structure of each project, the portal may eventually host a range of project-level governance structures that include "benevolent dictator" (also referred to as directed management in section 5.3 of the ConOps) through group decision making models (referred to as meritocratic management in the ConOps).

An important distinction is the pre-release and post-release project governance; that is, policies that apply while applications are being developed versus policies in force once applications are released to the repository for use by the transportation community. Our recommendation is that pre-release applications, if being developed using an open source or V model development path, should apply a "benevolent dictator" model to speed development; if using agile development, the group decision-making model is recommended. Post-release products are more likely to use "group decision-making" models, assuming that the community surrounding them are/can be maintained in the OSADP.

The project-level governance documentation needed includes the following:

- **Overview**: Provides any potential contributor or interested party with the objectives of the development effort, links to any additional governing documents, such as contracts and licenses, who can become involved in what roles, and how an interested party can join the development effort (if more open participation is allowed) or at the least how the interested party can join a community related to the development effort.
- **Roles and responsibilities:** Describes all roles involved with the project, their levels of responsibility, the extent of their authority, and who is eligible to assume the roles. Roles can be described quite specifically or more generally depending on the needs of the project. The section clearly indicates how a contributor may join the project. This section describes who manages the project and how. In addition, a key purpose of this section is to outline the rules of engagement and rules of conduct.
- **Support:** Describes the tools, links, and user "help" mechanisms available to those joining the project. The support function will be particularly critical for post-release applications.
- **Decision making process:** Describes the processes for managing the development process and its direction. The clear delineation of the process and of how disputes are resolved within the developer community is key to the project's progressing toward a successful product. This process will prioritize issues and changes, and will determine when a stable version of the application is ready for release.
- **Contribution process**[49]: Details the process by which contributors participate in the project. The section explains project policies regarding intellectual property rights, coding

---

method of last resort, placing a greater emphasis on consensus. In Group Decision-Making, the voting group can be the pool of all project contributors, or some smaller subset of key contributors who have the power to vote contributors in or out of the voting group.

[49] Roth, Craig. *Website Governance: A How-to Guide.* http://www.craiglroth.com/Opinions%20In%20Depth%20-%20web%20governance.pdf and Behl, Pardeep. *Winning Strategies for Portal Governance*. http://www.ibm.com/developerworks/websphere/library/techarticles/0904_behl/0904_behl.html.

and other standards, and the documentation expected from the contributor. There is also a description of any review and quality control processes that are applied to contributions. One key aspect related to the contribution process is "forking," which is an open source mechanism for spawning a new project with a new governance structure. Forking is less likely to be an issue for pre-release application development when the community of developers is likely to be smaller and more in concert with each other given the procurement processes that will have originated these projects. However, post-release management of forking is likely to be an important element of governance of the open source products.

3.6 Governance Principles for OSADP

Experience with initiatives in government such as Forge.mil as well as experience with open source in general leads to the following principles to be followed in building the OSADP:

- Allow for various types of governance; OSADP should be flexible and extensible.
- Allow the Program-level governing entity the ability to review all project activity within the portal to enhance the ability to identify opportunities to build modules suitable for reuse by other projects, especially during pre-release development.
- Program-, portal-, and project level managers should be able to define a role and indicate the level of access that participants having that role should have. Not all projects will have the same requirements and project needs can be expected to change over time. OSADP policies should not lock in any one model.
- Allow system managers to define the relationships among projects, applications and application bundles. Properly defining these relationships will enable community participants to be assigned access to the correct sub-communities and regions of the application development environment.
- OSADP allows its tools to be built or added to the portal. Building a useful toolkit will be a key contributor to the success of the OSADP. Tools will automate processes such as builds, metrics, and testing. Tool building should be among the first projects to which OSADP governance structures will be applied.

Another important program-level decision is whether development of applications other than the DMA bundles will be allowed. The Web Resources Assessment Scan document envisions a wider use of the OSADP than the ConOps, which notes that the OSADP is unlikely to expand to become more generic or broad in scope. This decision is also an important basis for determining if the OSADP could ever be financially self-sustaining. While most open source portals are free, Forge.mil and a few non-Federal portals provide examples for how fees could be associated with additional services, hosting, testing, or gaining certification.[50]

Another policy consideration is language policy. The ConOps notes that the OSADP will be open to everyone on the Internet, but that the primary language will be English only. This

position is created from the perspective that translation will be costly and hosting and maintaining a portal with multi-lingual capabilities will be burdensome. However, the developers will build the portal with the potential for expansion into additional languages, should user demand for this feature be realized.

The review for this report identified automated translation tools that work with varying degrees of success. Typically, if an additional language is desired, an individual or community petitions the Portal-level governance body to decide whether the benefits support the realization of the program goals and whether the costs are in line with the financial resources.

3.7 Summary

The Critical Issues Summary lists a set of questions pertaining to portal-level (system) governance and project-level governance. This chapter added a third level: Program-level governance. Chapter 3 was written to address those questions by providing definitions and steps for establishing governance at these three levels.

While definition of the program-level governance can begin immediately, some of the portal-level and most of the project-level governance requires the completion of the Mobility application bundle ConOps and a decision on which application bundles will be funded (and thus required to use the OSADP) and which will be supported (and encouraged to use the OSADP).

Near-Term Next Steps

In the near-term, as the OSADP development efforts are launched, the US DOT policy-making group should form a Program-level Governance team. The first action is to identify and form a Portal-level Governance team and assign roles and responsibilities. The second action is to document policies that include:

1. Criteria for registering and admitting users;
2. Criteria for accepting/designating a development effort as a project;
3. Criteria for releasing an application to the repository or transitioning beyond the repository to commercialization; and
4. A strategy for managing licensing and intellectual property; and determining who will be responsible for managing the licensing process within US DOT.
5. Performance metrics for the OSADP.

The first action for the Portal-level Governance team is to confirm the use of a centralized governance approach initially, and to identify the path and the risks associated with evolving portal-level governance over time towards a “federated” structure where sub-communities have more control over their operations. A second is to document the roles, processes and implementation roles for managing the portal throughout development and operations and updates (and to associate these roles with the ConOps and System Requirements documents).

The Project-level Governance will be formulated as new projects are accepted for the OSADP. In many cases, the form of procurement and type of development path is likely to impact the breadth, scope, and type of governance needed for each project. The procurement and award

will result in the identification of who will carry the governance rights and responsibilities at the project level, and whether he/she/they are bound by any requirements that emanate from the contract or from any inbound licenses.

The textbox below summarizes the next steps in establishing governance.

Next Steps:

- ***Establish a small Program-level Governance team comprised of the Federal program managers. Have this group establish the Portal-level Governance board consisting of the portal managers, technical experts, and user representatives.***
- ***Have the Program-level board establish objectives and metrics for the Portal-level team to achieve for risk acceptability, daily operations, and decision criteria using a centralized governance approach. (An important step will be to codify this approach in the ConOps, section 5.2.2.)***
- ***Develop a transition plan and timeline for evolving Portal-level governance from centralized to federated, based on user scenarios and anticipated risks.***
- ***Have the Portal-Level group develop user rules, standard operating procedures, and project acceptance/application release criteria. Document these policies and processes and incorporate into the Portal for transparent access for users.***
- ***Once established, have the boards define roles and responsibilities for ongoing operations and licensing processes.***
- ***Establish a set of procedures for the Portal-level board to follow when accepting a new project and working with the project lead(s) to tailor governance and oversight metrics in a manner that is specific to the project, its goals, and the level of new risks it introduces (for instance, risks in security, privacy, liability, or protection of intellectual property, among others).***
- ***Develop a process for the Portal governance team to work with project leads and teams to tailor governance for specific projects. Develop a checklist of needed information to guide each project lead to describe their form of project-level governance.***

4. Protection and Use of Intellectual Property: Licensing Options and Institutional Requirements for the OSADP

4.1. Introduction

In reviewing past experiences with ITS software development, the US DOT modal partners and stakeholders envision taking a different and somewhat innovative path with the dynamic mobility applications development (in particular, those that will be Federally-funded). A free and open source software (FOSS) approach[51] has important benefits:

- The use of open-source software licenses will enable licensees to:
  - View, download, and modify the application's source code; and
  - Transfer the licenses to users of the modified code.
- By offering its products under open source licensing, the DMA Program's intention is to offer developed applications with few restrictions on use to:
  - Public transportation agencies who are deploying the applications and modifying them as needed;
  - Developers and other members of the commercial sector who are commercializing products or support services associated with the applications; and
  - Agency staff, researchers, and other members of the public who are modifying and enhancing applications to enhance the state of the practice by the user community.

The most significant risk to the success of the DMA Program is in the area of intellectual property—the protection of intellectual property and the potential risk of infringement. Licenses and contributor agreements are the key legal tools in protection and use of Intellectual Property and in mitigating the risks of infringement. However, selecting the right licenses and contributor agreements to meet DMA Program goals is a complex process, particularly given the breadth and variety of the application bundles—their initial inputs of source code and documentation as well as the resulting applications that will be used in very different ways.

Appendix A offers a longer, more detailed explanation (a primer, of sorts) regarding the types of licenses and their terms, and the processes and timing associated with license arrangements. That appendix also provides a series of considerations for choosing licenses in order to protect and use the intellectual property that "seeds" the OSADP with initial source code and documentation, and to protect the intellectual property and set the terms of use for the applications and documentation as they move into the repository.

[51] The Role of Free and Open Source Software (FOSS) and Open Data in the ITS Data Capture and Management and Dynamic Mobility Applications Program, June 2011.

The remainder of this chapter is a summary of the key considerations that are targeted at the OSADP technical development team as they begin to build the Portal. It is also offered as a means of structuring a dialogue with a legal policy team to make appropriate decisions for the DMA Program in launching and supporting an OSADP. An important distinction is that the majority of the discussion in this chapter and Appendix A applies when using the more traditional procurement processes under the FAR. In the case of using challenge awards, the differences in process and in license considerations will be identified at the end of the chapter.

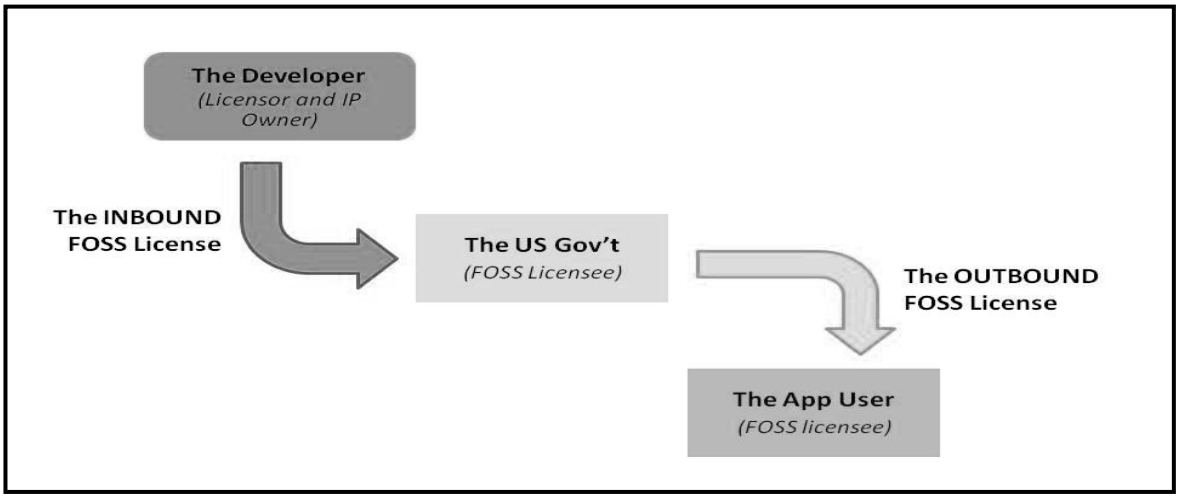
4.2 Important Basics

The need to manage intellectual property risks will come into play at three points in the application lifecycle: when the development of the application is being arranged; when the application has been accepted and is being offered to users; and when users are contributing enhancements to the application back into the repository.

The key to successfully defining the right intellectual property terms is the distinction between the licenses that the DMA Program will offer to users (the "outbound" licenses) and the licenses it will be receiving from developers and contributors (the "inbound" licenses or contributor agreements) who will provide "seed" code, develop the applications, or contribute enhancements.

The graphic below illustrates this "flow through of permissions" from the application developer, through US DOT, to the end user.

Figure 4-1. Flow of Permissions



***Inbound license restrictions cannot be removed, and flow through as a limitation on the terms of the outbound license and on any future licenses offered for downstream products. Therefore, the terms of the outbound license must be the basis for defining acceptable terms for the inbound licenses for each application*.**

In practice, there are a number of possibilities that add real-world complexities to the flow, and so present challenges to the DMA Program. The following paragraphs offer descriptions of two likely scenarios that the DMA Program may face.

Compatibility of Licenses for Merged Software

One scenario is when an application developer wants to merge the source code of existing software (also known as the secondary application) that has some of the desired functionalities into the application under development. The FAR requires that *contractors obtain permission from copyright owners (contributors) before including copyrighted works, owned by others, in technical data to be delivered to the government.*[52] The FAR defines "technical data" to include deliverables such as software.[53] Therefore, if the software deliverable incorporates code from other applications, *it is the responsibility of the developer to make the necessary intellectual property arrangements with the owner.* These arrangements take the form of a "secondary inbound license".

The terms of that secondary license will pass through into the primary inbound license that the developer gives the DMA Program, and then into the outbound license that the DMA Program offers to users. In order for the DMA Program to be able to offer the released application under the terms it wants, ***the secondary inbound license must also have terms compatible with those of outbound license*.**

Compatibility of Inbound Licenses from Multiple Developers

A second scenario is when the DMA Program contracts with multiple developers working under separate contracts to provide source code for various parts of a single application, as might be true if an application is being developed iteratively over time. Each developer owns the intellectual property for the source code it produces, so each has to offer the Program a license. *These multiple inbound licenses must each align with the terms of the outbound license, AND be mutually compatible in all other particulars.*

Other such scenarios are likely to emerge. The OSADP and the legal team that supports the OSADP will need to account for and track these complex situations.

[52] Federal Acquisition Regulations (FAR) 27.102(e).
[53] FAR 2.101.

4.3 License Categories: Levels of Openness

An open source software license describes the terms of use and establishes the level of restrictiveness or permissiveness allowed for licensing enhancements or modifications to the original product. Definitions for these two types of licenses are:

Restrictive licenses require that all software derived from the original product be also licensed as free and open. These licenses carry the "copyleft" stipulation, which allows a software program and its source code to be used without consent from its creator/owner[54]. Once software is licensed with the copyleft provision, all "daughter" versions must contain the copyleft provision. The most widely used copyleft license is the GNU General Public License (GPL) and its variants. The Free Software Foundation offers a list of GPL licenses, along with guidance on how to choose among them.[55] Restrictive licenses encourage wide participation by users and developers in product modification and improvement. However, this type of license reduces the product's attractiveness for commercialization, because the licensee is unable to charge the sort of prices that could be charged when it in effect has a monopoly on code that is a trade secret.

Permissive licenses allow modified open source and object code to be distributed under non-open licenses in addition to open source licenses. Therefore, licensees adopting and then modifying open source applications can impose restrictions on downstream end users without having to disclose source code. Among the few conditions of use in such licenses are: (1) that the original licensing terms have to be present in future licenses for derivative works, and (2) that the original copyright notice is to be included with the documentation of the derived work. The Open Source Initiative reviews licenses submitted to it; those that conform to the Open Source Definition are posted as approved on the OSI website.[56] [57]

Three of the most frequently used licenses in this category, in order of increasing permissiveness, are the Apache 2.0 license, the Berkeley Software Distribution License (BSD) 2.0, and the MIT (X11) license. The following describes these licenses and the advantages, strengths and limitations:

- **The MIT License** (MIT/X11) is the simplest of these licenses[58]. It is known as one of the common open source software (OSS) licenses [59]and is compatible with the General Public License (GPL). While the MIT/X11 License is very similar to BSD in terms of scope and permissiveness, a key similarity that it has with the Apache 2.0 license is that it contains a copyright provision as part of the license.

[54] http://www.gnu.org/philosophy/pragmatic.html.
[55] http://www.gnu.org/licenses/licenses.html.
[56] http://www.opensource.org/licenses/index.html.
[57] A general review of licenses and their uses suggests that the right license depends on the needs of the intellectual property owner. As a result, there are over two hundred variations of open source licenses. Most are slight variations on the common ones; some serve specific communities. Many of these variations are untested in the legal environment and are not considered proven or stable. There is a recent anti-proliferation movement that is supported by major software firms as a means of limiting the number of new licenses.
[58] http://www.opennetcf.com/FreeSoftware/Licensing/MITX11License/tabid/254/Default.aspx.
[59] Which means that it has been certified as open source by the Open Source Initiative (OSI) and as Free Software by the Free Software Foundation (FSF).A report by the Department of Defense offers, as a key lesson learned, the guidance to use OSS licenses which can help ensure compatibility with other licenses.

- **The Berkeley Source Distribution** (BSD-new) license is a simple, uncomplicated and widespread open source license. In comparison to the Apache 2.0 License, the BSD license is shorter and is also widely used and understood by the developer community. It is compatible with GPL licenses. An important note is to use the "new" version, as the original was incompatible with the GNU GPL.[60,61]
- **The Apache 2.0 license** is similar to the two mentioned above. It is widely used. One of its advantages is that it mentions rights under copyright law and provides the owner a license with those rights. In this respect, it is slightly more restrictive than the BSD or MIT/X11 licenses. It is longer and requires more work on the part of the intellectual property owner and the lawyers to develop; and it requires attention and work by developers who wish to modify than the other permissive licenses. It also does not easily allow for "shrink-wrap" functionality. Last, Apache is not compatible with the Creative Commons General Public License (GPL) version 2, which may create problems. (It is, however, compatible with GPL version 3.) An advantage to using the Apache 2.0 is that it mitigates concerns with infringement as it is the only commonly used permissive OSS license that contains a patent grant. It is also known to provide protections for unforeseen scenarios such as automatic property rights assignment for contributions or poison-pill–like protections against patent suits.

These three licenses are considered to be permissive. Other license categories include free/fully reciprocal, partly reciprocal and others (such as artistic). The recommendation is to adopt licenses from the permissive group because these licenses allow greater opportunities for commercialization without the burden of having to ensure that ALL enhancements or value-added modifications remain open source.

The risk with using these three licenses, however, is that the software may be modified into one or more proprietary versions which can then no longer be shared and co-developed by those who developed the original version. That risk may be mitigated by careful definition of the application within its ConOps to assure that it identifies all the features the prospective users will need for core functionality. If, however, this risk is still deemed to be too much of a concern, then the weakly restrictive licenses offer an alternative:

- **The Lessor General Public License v3.0 (LGPL v3.0)** is a compromise between the permissive and strongly restrictive licenses. It encourages co-development of the library, while allowing proprietary programs to include them. This category of license prevents the software component (often a software library) from itself becoming proprietary, yet permits it to be part of a larger proprietary program.[62] However, this license appears to be incompatible with the implementation of the free and open software policies.

An additional note about the use of two well-known open source licenses—the NASA Open Source Agreement, version 1.3 and the Mozilla Public License (MPL). The DoD report[63]

[60] http://www.opensource.org/licenses/bsd-license.php and http://www.linfo.org/bsdlicense.html.
[61] DoD, Open Technology Development: Lessons Learned and Best Practices for Military Software, May 2011, p. 63.
[62] See Appendix C.
[63] DoD, Open Technology Development: Lessons Learned and Best Practices for Military Software, May 2011.

advises against using these licenses as they are known to be incompatible with the GPL. It is for this reason that these licenses are not included in this analysis and set of recommendations.

There is one additional option and that is to place works in the public domain. To do so, the requirements are that the final application or product be free of any licenses on the original source code or other features; and that the developers and contributors agree. This agreement may be stipulated as part of a Federal contract that procures development and claims full ownership of the source code, application, and other documents. It can also be done by publishing the patentable information as '"prior art". See the textbox below for a broader definition of the public domain option.

Definition of the Public Domain Option

"Public Domain" means such works as inventions, and methods of manufacture, processing or doing business that are owned by no particular person or entity and that may be freely used by anyone. Such works belong to the public as a whole. Anyone is free to use them any way one wishes without asking anyone's permission, including commercialization. And no one can obtain copyright or patent protection for public domain material.

How Does an Intellectual Property move into the Public Domain?

(1) By Default:

- For works subject to copyright: (1) works in which the copyright was lost (e.g., all rights are lost if the owner does not take timely steps to abate infringement), (2) works in which the copyright expired and (3) works authored or owned by the federal government.
- For works which could have been patented: By publication or other public disclosure of essential patentable information, or by public demonstration of the invention, more than one year before the filing of a patent application.

(2) By Deliberate Act:

- Copyrighted materials: The owner of the IP can put it in the Public Domain by explicitly releasing ownership and authorizing the free use by anyone for any purpose. This can be accomplished by affixing a notice to the IP stating such (e.g., some freeware and shareware contain these notices)
- IP protected by a valid patent: The owner may publicly declare that no action would be taken against infringers, or may dedicate the patent to the public.

Definition from Carnegie Mellon at: http://www.cmu.edu/innovationtransfer/Home/documents/ipg6.html.

For the Federal government, the public domain option may appear to align best with the program's goal of developing and releasing free and open source software. To be an effective option, placing a work in the public domain must be acceptable to the developers who either contract with the Federal government or who form part of the development community and contribute enhancements and modifications. However, if the public domain option is institutionalized as policy against developers' interests, developers might find that working as part of the OSADP community is unattractive. Further, the public domain option release is most typically used when rights have expired or the work is somehow intangible or not eligible for rights. This standard may be difficult for the DOT to prove except in very specific circumstances.

Last, use of licenses has advantages—the use of licenses provides credits to the developers and a clear link back to the developer for questions. As noted by one expert, "…software licensing is about setting boundaries on what other people can do with your code. The complexity of licensing comes from defining and explaining those boundaries in legal, enforceable terms."[64]

This report recommends that the Mobility Program's open source policy limit developers who wish to release code or applications into the repository to do so under one of the permissive licenses (or petition the DOT for use of other licenses). If the DOT is funding the development, the contract can stipulate that the DOT intends to provide the end product publicly using one of these licenses (and thus providing the developers with attribution and copyright).

Offering a range of licenses is meant to provide flexibility for the program given the likelihood of supporting a diverse range of applications. This report also recommends the development of a petition process for new project developers who desire the use of a more restrictive license. Such a petition process is likely to involve the Program-level governance board as well as the Legal Policy team who will analyze the impact of introducing a more restrictive license option and determine if fulfilling such a request meets the objectives of the program.

Appendix A offers a more detailed primer in how to choose licenses. Table A-1 lists the most common open source licenses and provides guidance on how to analyze and choose an appropriate license. In summary, there are many considerations involved with choosing which license best applies to each application. The following lists and describes some of the key steps:

Step 1: Choosing the Outbound License.

Steps include:

1. **Identify whether the completed application is a modification or extension of existing open source software.** If it is, the current license might apply.
2. **Identify the extent to which the open source application's code will be permitted to evolve in a way that brings part or all of its code back under conventional licensing terms – that is, toward commercialization where** anyone may use the source code for any purpose, including creating divergent, incompatible, proprietary versions and proprietary modules inside larger proprietary programs.
3. **Analyze the Intended Uses and Users.** At this point in our understanding of the application bundles, it is assumed that:
    a. Applications in the M-ISIG, INFLO, R.E.S.C.U.M.E., and IDTO bundles are intended for public-sector agency use and thus the DMA program is interested in mitigating the risks of downstream costs for public agencies through license choices. The options are:

[64] Van Lindberg, *Intellectual Property and Open Source*. O'Reilly Media, Inc., 2008, page 198.

- To select licenses that effectively close off, or at least limit commercialization of the applications in them (trade-off is to forego the "success" of commercializing).
- To select permissive licenses, and find alternative avenues to limit downstream costs.

b. Applications in the FRATIS and ENABLE-ATIS bundles are likely to be extensions of applications already in commercial distribution.[65]

Assuming these assumptions are correct, the table below offers a high-level example of an analytical path that can result in license options. Once the application bundles are better understood, all of the individual applications will need to be examined at this level, at a minimum.

Table 4. 2: Considerations in Choosing an Outbound Open Source License

User	Considerations			License Options
	1.Builds on Other Open Source	2. Extent of Permitted Proprietary Use		
		2a. Permits Proprietary	2b. Permits Proprietary Library Only	
Public agency	Possibly	Yes	Probably	If new application is based only on other open source software, use similar license. If otherwise, consider one of three **permissive licenses**: • MIT/X11 • The *new* BSD license, or • Apache 2.0. If patent infringement is a concern, use ONLY Apache 2.0. Examine the benefits and limitations with providing the new enhancements or modifications in the public domain.
Individual Traveler/ Commercial Fleet	Possibly	Yes	Probably not	

[65] This assumption is based on conclusions drawn at the DMA Program Technical Team meeting on 4/20/2011.

Summary and Caveats with Outbound License Selection

Ensuring that the license that is chosen is appropriate for the situation is a complex undertaking, and a table cannot identify all the potential exceptions and impacts that might occur. Importantly, the terms of the outbound license must be defined first in a manner that is consistent with the terms of inbound licenses. Additionally, having the completed application with a full understanding of its intended uses supports deciding on which license to use.

Step 2: Aligning Inbound Licenses and Contributor Agreements with Outbound License Requirements

Once the terms and needs of an application's outbound license have been identified, compatibility with inbound license terms can be analyzed. Appendix D offers information and an illustration of how common open source licenses can be combined.

Inbound Licenses

Defining the terms of inbound license(s) must consider both US DOT and developer interests in their effects.

From a developer's perspective, an inbound license must be a good fit with a developer's business model in order for them to participate and grant the inbound license terms. Typically, this means capturing profit which also typically translates into keeping control over the source code.

However, it is within the DMA program's interest to promote open source licensing with released products in order to maximize access to the source code for further enhancements and modifications or to spur the development of derivative, innovative works. Thus, in negotiating the inbound license terms, the US DOT may find a need to accommodate trade-offs such as allowing a more restrictive license if the source code is of value to application developers.

A relatively new and growing recognition by some developers, however, is the appeal of open source as part of their business model. Dual and multiple licensing arrangements have emerged as a strategy to enable developers to both pursue a profit-oriented business as well as allow for an open source arrangements with specific types of organizations. In this situation, the licensor offers open source license terms to a segment of the market that is interested in open source while retaining proprietary license terms to the remaining market. A good example is MySQL[66] which collects profits from users who purchase the software; those proceeds help fund additional development. However, the software's source code is openly available to those who wish to improve or contribute changes. In most instances, this results in a no-frills application for open source distribution and an enhanced version for commercial sale.

[66] http://www.mysql.com/.

Contributor Agreements

A contributor agreement (a.k.a. contribution agreement) is an agreement by which an *individual contributor* to an open source project grants sufficient rights for the parties operating the project (in this case, the DMA Program) to release the contribution as part of the project (in this case, open source release on terms compatible with the outbound license). In essence, a contributor agreement is an inbound license, and its details should be given the same level of careful consideration.

The Apache Contributor Agreement v2.0[67] is generally regarded as a standard by the industry.[68]

There are three approaches to such agreements:

1. To require the contributor to assign all rights to the DMA Program for their contributions. Assignments typically grant back to the assignor (the contributor) a broad right to use the code outside of the project. This is somewhat akin in effect to the notion of dual licensing, in that the contributor can take the application proprietary.
2. To require the contributor to grant a broad license to the DMA Program.
3. To use no agreement at all. This is not recommended.

The terms of contribution agreements may include representations and warranties for the protection of the recipient (in this case, the US DOT)[69]. For example, a warranty may be required from the contributor to note that he/she actually wrote the new or enhanced contributed code; or that he/she is not employed by a company that will claim rights to it. Having such terms significantly reduces the DMA Program's exposure to charges of copyright infringement.

Another reason to have such agreements is in anticipation of the possibility that with time and experience, the DMA Program might wish to change the outbound open source license. If the DMA Program has failed to obtain contribution agreements, it will not be able to change the outbound license unless it receives permission from every contributor. This is a time-consuming, costly, and possibly infeasible scenario, and best avoided.

Summary and Caveats for Inbound License Selection

Inbound licenses will be the appropriate legal instrument for intellectual property agreements between the DMA Program and developers awarded a contract through conventional procurement. Contributor agreements are a more appropriate instrument under the following circumstances: a) if a challenge invites the participation of individuals to develop an application through an open source software development process, and b) when, after an application has been released through the repository, an individual wants to contribute enhanced code back into the user community. Whatever the terms that the DMA Program defines as acceptable for

[67] Not to be confused with the Apache 2.0 license.
[68] Meeker, *passim,* p. 147.
[69] Note: There may be a concern, however, with requiring that the contributor writes all the code as this may disallow code that utilizes other open source code as building blocks. Therefore the warranty may be that they wrote the code consists of code they wrote themselves and possible additional code that is licensed as open source.

inbound licenses and contributor agreements, these must be clearly stated in the RFP so that the intent is transparent to prospective responders.

Inbound licenses are to be used in relation to procured development; contributor agreements may be used for challenge-based open source development projects, and for post-release contributions of enhanced code. A recommendation for the staff who review the inbound terms is to be cautious about accepting inbound products with patents. This is a highly controversial practice and is currently posing challenges to the US DOT and State and local-level transportation agencies in fully embracing ITS.

4.4 Additional Considerations

The following is a set of additional considerations that may apply in most, but not all licensing situations.

OSADP Management of Post-Release Contributions of Enhanced Code

The DMA Program's need to manage intellectual property does not necessarily end at the point when users download the released open source application, so neither does the requirement that the Portal track contributions and permissions.

The OSADP ConOps anticipates that the repository will be used both "to share code and artifacts and receive contributions from the community."[70] We anticipate that this will be particularly the case for public-sector applications, which should attract significant registered user communities. In that event, community members may be interested in sharing improvements to the application with one another in the form of enhanced source code.

Thus, the Portal must be able not only to record the fact of the contribution, but also to provide the user with a contributor agreement, prepared in advance by the DMA Program, that the user must accept electronically in order to upload the enhanced code. (The upload process should also be contingent upon the user's supplying metadata on the contribution.) The record of the upload, the identity of the contributor, the fact of acceptance, and the metadata are retained by the Portal.

Open Source Licensing of Non-Software Deliverables

The DMA Program will be procuring not only the source code for the Mobility applications but associated documentation. These items are also covered by copyright, and so require a license from the developer permitting open source distribution, but are licensed separately from the source code. Creative Commons offers a range of licenses for this purpose and provides an on-line selection tool (the License Chooser).[71] Appendix A provides descriptions of the licenses for consideration and identifies their differences and advantages and limitations.

[70] *DMA OSADP Concept of Operations*, Final Draft Document, Version 3.3.3 – August 5, 2011, p. 13.
[71] http://creativecommons.org/choose/

Procuring Application Bundles through Challenge Awards

From the standpoint of intellectual property rights and their management, the use of challenges introduces two distinguishing considerations for the DMA Program. The first is an alternative legal tool for conveying IP permissions: the contributor agreement, as described above. The second is the effect that using the challenge has on when the intellectual property agreement can be finalized, and the effect that may have on the outcome of the challenge as it relates to the achievement of DMA Program goals.

If the DMA Program decides to use a challenge instead of a traditional RFP for an application, the challenge announcement will need to state that the accepted application will be released by the DMA Program under specified open source terms and license(s). Participants must agree to the contributor agreement before they may upload their contributions. The pivotal concern for the DMA program will likely be the matter of any upstream inbound licenses. A set of scenarios and decision issues are described in Appendix A.

Note on Patenting

Patenting of software is legally permissible in the US at this time; although highly controversial.[72] It is an increasing practice among universities.[73] Patent law reserves the intellectual rights to the creator, who may license them out for stipulated uses in exchange for royalty payments.

International Intellectual Property Law

International intellectual property law will be relevant if the DMA Program allows application development by international entities. International conventions[74] automatically attach copyright to every novel expression of an idea, whether it is through text, sounds, or imagery. Japan and Asia reportedly have patent laws similar to those of the US, whereas Europe has been more conservative.[75] In particular, the Europeans don't favor patenting of software.

Additionally, with the decision on whether to permit non-US entities and individuals to download the released software, the completed software will need to be compared against the U.S. Department of Commerce Bureau of Industry and Security's guidance on whether the software or any part of the application might be constrained by export controls or will require a license for this purpose.[76]

---

[72] G. Gross. *Court Patent Ruling Leaves Software Patents Intact*. PC World Business Center, June 28, 2010. At http://www.pcworld.com/businesscenter/article/199994/court_patent_ruling_leaves_software_patents_intact.html. Accessed 2/21/2011.

[73] AK Rai, JR Allison, BN Sampat, and C Crossman. *University Software Ownership and Litigation: A First Examination*. 87 North Carolina Law Review 1519-1570 (2009). Abstract at http://scholarship.law.duke.edu/faculty_scholarship/1629/. Accessed 6/1/2011.

[74] The Berne Convention for European countries and the World Trade Organization Agreement on Trade-Related Aspects of Intellectual Property Rights.

[75] *Software Patent Law: United States and Europe Compared*. iBRIEF / Patents & Technology. 2003 Duke L. & Tech. Rev. 0006, 3/21/2003.

[76] See: http://www.bis.doc.gov/licensing/exportingbasics.htm

4.5 Summary

Properly identifying, attributing, and thus protecting the intellectual property (i.e., source code, algorithms, applications, and documentation) of the developers who create it, is the critical first step in addressing intellectual property for the DMA Program. Similarly, if the new application has components that originated as intellectual property of some other person or entity, that ownership must be recognized in advance and used according to the stated terms.

The second critical step for the DMA Program is to obtain from developers the permissions needed to release and/or distribute the intellectual property as open source. Completing these two steps successfully will protect the DMA Program from claims of infringement, and protect users from the need to pay royalties and license fees to the developers. Taking these steps, and using the appropriate license arrangements recommended in this chapter, should result in the ability to enable an open source approach to application release, albeit one with potential trade-offs such as:

- Whether the DMA program seeks greater open use and availability or greater commercialization given the intended end use of each application;
- Whether the application was developed using source code and software from other secondary applications; and
- Whether contributors offered post-release enhancements or corrections to the original application.

An additional consideration is whether the DMA program seeks a user community to adopt the maintenance and upkeep of an application. If yes, in addition to the license terms, a compatible legal framework will be needed that clearly describes the appropriate permissions granted to the community.

Recommended Licensing Strategies

For the outbound licensing, we recommend the use of three permissive as a means of enabling the open source approach. They are:

- The MIT License
- The Berkeley Source Distribution License, version 2.0
- The Apache 2.0 license
- Public domain

In outbound licensing, the role of the OS Portal is to recognize and record when a released application is uploaded, and by whom, in acceptance of the open source license for that application. Procedurally, the OSADP must be able to:

- Provide a link to the application's source code and documentation,
- Identify the user attempting to download the application, and
- Recognize the user's acceptance of the open source license.

The second and third of these requirements will require further policy consideration regarding privacy implications, but this tracking is standard procedure for most open source portals.

For outbound licensing, the OS Portal will also need to provide mechanisms for license acceptance. Appendix A provides options for doing so (see sections A.4 and A.5)

Because attracting development expertise will be dependent on the attractiveness of the license terms to the developers from a business standpoint, it is recommended that the DMA Program conduct one or more outreach events (workshops and/or participatory webinars) to present the Program's intent for the applications and the desired outbound terms to get feedback before starting the procurement process.

Institutional Recommendations

An important role of the OS Portal is to display the User Agreement (perhaps providing a link on the Portal's home page). It is recommended that the registration process require the participant to agree in advance to accept the terms of the agreement before uploading contributions. Procedurally, the OS Portal must be able to:

- Provide a link to the contributor agreement,
- Recognize that an attempt is being made to upload a contribution,
- Identify the developer or contributor attempting to perform the upload, and
- Verify that the developer or contributor is operating under a license or contributor agreement, and then either :
    - Permit the upload and link it to the developer; or
    - Block the upload if the chain of linkages is incomplete, and display a message giving the reason for the blockage.

Functionally, verification can be made through automated querying of a relational database. In order to populate the database, the Portal needs to obtain data through two different channels:

- **Developer Data.** Capturing identifying information on the developer or contributor for this purpose occurs through user registration:
    - For procured projects, the OS Portal needs a process for receiving the list of approved team members and their project roles (corresponding to the role definitions on the Portal) and using the list for authentication at the time that the team member first registers. Who provides the list to the OS Portal is a role the must be defined (i.e., the DMA Program or direct transmission from the contracted developer). A process for receiving changes to the list (personnel removed from the project and those added to it) is also needed, including standards for when the revised information must be provided relative to the triggering event.[77]
    - For challenges, participants must register and provide required information (to be determined as a governance decision).
- **License/Contributor Agreement Data.** Capturing information on the relation between the developer/contributor and the license or contributor agreement depends on which of the latter is involved:
    - For procured projects, a procedure is needed for conveying license information from the DMA Program to the Portal, because the legal arrangement takes place outside of Portal transactions.

[77] For example, "contractor must inform OS Portal within one business day when a team member is removed."

- If the project is through a challenge, at least three scenarios are possible:
  - Development takes place outside the Portal, and the contestant developers wish to upload the completed application.
  - Multiple development teams compete using the ADE.
  - Participants collaborate on a single open source project.

  In all three instances, the contributor agreement, and the developer or participant's agreement, can be handled virtually by the Portal.

In relation to all projects, the role of the OSADP is to assure that each contribution is being made by a registered developer or contributor and is covered by a license or contributor agreement. Thus, the OSADP will need to establish a tracking and management system to track:

- Who is creating the code and documentation;
- What licenses or agreements are connect to that developer and that contribution;
- After release, track who downloads the applications from the repository in acceptance of the outbound license terms; and
- If contributing enhanced code back to the Portal, track who is contributing what and under what license or contributor agreement.

Recommendations When Selecting Procurement Paths

The role for the DMA Program was developed for each Mobility application and in consultation with US DOT counsel and US DOT Acquisitions. The roles and responsibilities are the following:

For a procured project, to:

- Assure that the terms of the outbound license support the intended use of the application, including potential commercialization and/or merging with the source code of other applications (especially if the other applications are also Mobility applications).
- Assure that the request for proposal fully represents the DMA Program's intent to offer the completed application as open source; lists the license(s) and terms under which it will be offered; and indicates the open source terms and license(s) that it will accept from the developer. If permissible under law, the RFP should also require that the developer agree to consultation with US DOT regarding the terms of any secondary licenses it must negotiate, and to submit the draft secondary license for US DOT's approval before it is finalized.
- Assure that the terms of the inbound license received from the contracted developer in no way conflict with or override the terms of the outbound license. By implication, the terms of any secondary licenses attached to the inbound license are equally non-conflicting and compatible with the outbound license. (See Appendix A for extended consideration of this point.)

For a competitive challenge, to:

- Decide whether full assignment of rights or a broad license is the appropriate arrangement with contributors.
- Use the appropriate legal document.

For all projects:

- To assure that the OSADP User Agreement accurately presents licensing and other terms applying to code and content posted on the Portal**.**
- To identify the person/processes for reviewing terms, negotiating the license arrangements, and signing licenses. An important decision at this point is whether this will be the US DOT legal counsel, which will require a staff commitment and probably a commitment to timing that did not exist before. If not the US DOT, a decision regarding who/whom will need to be made. This same staff will need to also commit to some oversight, conflict resolution, and other support.

Outstanding Issue

One issue that needs further exploration is the situation in the OSADP ConOps that appears to place the repository within the Registered User Environment. If this placement is correct, and if the user is an unaffiliated individual, the user registers with the OSADP and downloads the open source application; the user is the individual is the licensee, and the OSADP can link the individual's registration data to license acceptance.

From a legal standpoint, we are not clear who the licensee is, or should be, if the registered user is the employee of an entity such as a public agency or private firm. We recommend that the DMA Program seek clarification from US DOT counsel on this point. At a functional level, the OSADP will have two alternatives if the legal opinion is that the outbound licensing arrangement should be between the DMA Program and the agency or firm:

- To make "organization" a required field, and link the license acceptance to that field, or
- To enable agencies and firms to register as users while permitting individual employees to register as sub-users, in which event the sub-user's acceptance of the license terms is linked to the user field.

The textbox below describes a set of next steps.

Next Steps:

- ***Establish a comprehensive license strategy by:***
  - ***Working with US DOT legal counsel to determine whether the appropriate level of open source intellectual property expertise can be made available to the DMA Program.***
  - ***In concert with the development of program-level governance, establish a set of processes and procedures that guide how and when licensing arrangements will take place.***
  - ***Ensure that the licenses and other considerations recommended in this report are aligned with US DOT policies.***
  - ***Based on these decisions, convene a public webinar or workshop to describe the terms and receive feedback on whether such terms and processes will work for developer(s).***

5. Procurement and Development Options

The purpose of this chapter is to describe the impacts of procurement and development strategies on open source policies, and to discuss how choices impact the OS Portal's policies and structure. The chapter describes the analytical framework used to produce the recommendations. It is a two-step process for determining which alternative is best suited for an application's development:

1. **Identify and Assess Suitability of Development and Procurement Options/Level of Federal Effort:** Software development generally follows a variant of one of three process models: the V model, Agile, and open source development. With that choice, comes a determination of whether to use a traditional procurement or a challenge. This premise—the choice of process model for application development should influence the choice of procurement strategy and not the other way around is well documented in the literature.[78] The development-procurement choice is also closely correlated with the level of Federal effort required by each option.
2. **Analyze an Application's Characteristics and Risks:** Four characteristics appear particularly important when considering development within the OSADP. Results of the analysis support the analysis of suitability regarding the type of development-procurement option chose. The four characteristics and risks include:
    - **Intellectual property:** The extent to which the DMA may conflict with proprietary software.
    - **Sensitive code:** Some DMAs can affect public safety or expose sensitive commercial data in their operation if their source code is compromised.
    - **Need for Adaptability:** For some DMA development projects, performance objectives may need to adapt to uncertain user needs or changing market environments.
    - **Degree of Innovation:** The DMAs differ in where they stand on the spectrum between evolutionary and revolutionary.

The ultimate recommendation of this chapter is to analyze each application using this process to:

- Identify risks and potential trade-offs.
- Identify how the OSADP might be structured to mitigate risks or support collaboration where it is needed most.
- Reveal whether an application is suited for open source development or if a more traditional and structured development path offers greater benefits.

[78] A well-known example to the ITS community is: Marshall and Tarnoff, Guide to Contracting ITS Projects, p. 11.

5.1 Why Procurement and Development Choices Matter

The Critical Policy Issues white paper identified three procurement policy issues in relation to the Portal:

1. **Application Suitability for Portal Development:** *Are all applications suitable for development on the OS Portal?*

   This question can only be addressed after better definition is developed for the applications. Section 5.3 describes some of the key characteristics and risks to consider when choosing a development options and determining whether an application is suitable for development in the OSADP. Importantly, at this point in time, the primary objective of the OSADP is to house source code, serve as a community focal point, and host a version control system that supports sharing, modification, and updates of open source code.

2. **Federal Staff, Resources, and Costs:** *Are there differences in the demand for federal participation during development, depending on the software development methodology the developer employs? If so, what are the associated costs and benefits to the program? Does the program have access to federal personnel with the required skills?*

   This question is addressed in section 5.2 which presents three choices of for development: open source, the V, and agile; and identifies the implications for Federal participation and commitment.

3. **Procurement Strategy:** *Are the OSADP policies and structures supportive of the DMA procurement strategies?*

   This question is addressed in section 5.2 which results in a procurement strategy based on specific application attributes, and identifies adjustments to the OSADP's policies and structures needed to align the strategy and the Portal.

Two important considerations inform the analysis:

- **The choice of process model for application development should influence the choice between procurement alternatives – not the other way around.**[79] Software development can follow a wide range of process models, but three are under discussion: the V model, the agile method, and fully open source development. The Mobility Program should first decide which of these models is suitable for an application, and only then choose whether to use the procurement process or a challenge.
- **There is no one-size-fits-all option for procurement of the Mobility applications.** The applications and bundles differ on the basis of a number of distinguishable factors. In addition, the bundle teams have already made decisions with regard to how the bundles and applications are being planned for development. These factors drive the suitability of one software development approach over the others for each application, and therefore also drive the choice between the procurement alternatives.

[79] Marshall and Tarnoff, Guide to Contracting ITS Projects, p. 11.

5.2 Procurement and Development Options

Definition of Procurement Options

Table 5.1 defines the two, primary options for procurement strategies.

Table 5.1 Procurement Strategies

Procurement Method	Definition
Traditional Procurement	As defined in the FAR, "procurement" is contracting for development services from non-federal sources.[80] While procurement can take many forms procedurally, the model typically involves issuing an RFP, reviewing proposals, and awarding a contract to specific individuals or a company that is accountable for meeting specific timelines or milestones and delivering end-products with defined functionality. In a traditional procurement, only the development team selected through a process of proposal solicitation and selection is eligible to conduct the work.
Challenge	A challenge is an invitation to third parties to identify a solution to a particular problem or achieve a particular goal. Prizes and rewards, which can be monetary or non-monetary, often accompany challenges and contests.[81] (Note that a challenge is not a grant, contract, or cooperative agreement[82], and so technically is not "procurement.") In a challenge, the development objectives and the rewards for achieving them are specified, and a wide range of developers are welcome to contribute toward reaching the objectives. The "wide range" of development teams may be limited by eligibility criteria, but not so much that participants are literally preselected.

Definition of Development Options

Software development generally follows a variant of one of three process models: the V model, Agile, and open source development. There are two key characteristics that differentiate the process models—level of structure and level of adaptability.

[80] FAR 90-34, 2.101.
[81] http://challenge.gov/faq#a1
[82] PL 111-358 §24(p)(2)(B).

Structure

The V and Agile development approaches require that a particular contractor team is specified in advance of the work. For the V approach, this is necessary to ensure accountability for completion of the specified work and also to allow for the sponsor-contractor communication necessary to enable successful project execution. The same is true for the Agile approach, but to an even greater extent because of the heavy and frequent interaction between sponsor and contractor that is built into Agile. By contrast, one of the essential elements of open source development is that the development team is not selected before development work begins. Development using the V or Agile approaches requires traditional procurement.

Further, there is a distinction between *single open source projects* in which the solution and eligibility of team members "pre-selects", to some extent, the procedures and structure of the project versus *competitive open source* in which all developers have the chance to contribute. The objectives and rewards are more clearly stated for a competitive open source project than for a single-source project, but they are at least implicitly stated for both. Any open source development, whether conducted as a competition or a single source project, must be "procured" through a challenge.

Adaptability

These process models can be thought of as existing on a continuum from adaptive to predictive.[83] Open source and Agile methods lie on the adaptive side of this scale. Adaptive methods focus on adapting quickly to changing realities and to the needs of a project. Predictive methods like the V model, in contrast, focus on planning the future in detail. While a predictive team can report exactly what features and tasks are planned for the entire length of the development process, it will have difficulty changing direction if the original concept and requirements shift unexpectedly; because the plan is typically optimized for the original destination, changing direction can require starting over. For that reason, predictive teams will often institute a change control board to ensure that only the most valuable changes are considered.[84]

Table 5.2 summarizes the development options and identifies some of the key characteristics and trade-offs of each option, including the level of Federal effort involved. The last column in the table identifies the suitable procurement option. The text following the table provides greater definition of each development-procurement option.

[83] Boehm, B.; R. Turner (2004). *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston, MA: Addison-Wesley. ISBN 0-321-18612-5. Appendix A, pages 165-194.

[84] http://en.wikipedia.org/wiki/Agile_software_development

Table 5.2 DMA Development Options

Development Options	Advantages	Limitations	Level of Federal Effort		Suitable Procurement Option
			Procurement Phase	Development Phase	
V Model	• Aligned with Federal procurement processes • Many contractors qualified in the approach • Steps in the process are well-defined and easy to measure progress and identify where changes are made. • Provides means to screen project participants and reduce risk of bad actors. • Relatively low Federal labor involvement in the process	• Requires clear and stable requirements • Multiple iterations at beginning of process slow time to completion relative to Agile. • Difficult to change course if needed • All work done by single contractor or team; loss of potential benefits of broad collaboration that OS offers. • Contract holds developer responsible for final product; discourages openness.	High	Medium	Traditional
Agile	• Responsive to changing conditions and/or clarified needs • Usually faster to completion than conventional V. • Provides means to screen project participants and reduce risk of bad actors.	• Unless managed well by experienced contractor, project can fail. • Does not align well with typical Federal procurement practices, introducing risk and difficulty from a procurement perspective.	Medium	High+	Traditional
Open Source: Single Project	• Encourages broad collaboration. • Enables Program to have control over project decisions and direction, as well as review of inbound licenses.	• Incentivizing speedy, directed development is difficult	Medium	Low	Challenge
Open Source: Competitive	• Highest potential to capture innovation	• Requires detailed, stable objectives • Does not guarantee completion of project – contingent on adequate level of competent participation.	High	Medium	Challenge

The V Model[85]

The V model is the preferred method for developing ITS projects. It is distinct in its detailed structure that both *separates* the definition of what must be done from how it is done and *links* the detailed product requirements to validation and verification of product performance.

Software development using the V model is a highly structured process. The transition from one process step to the next is treated as a decision point for the purpose of risk management and project control, and involves documentation and review. When the V model is used with a DOT procured project, a technical project manager has responsibility for tracking progress and participating in those documentation reviews. While the time commitment for review may be several days for each transition point, the DOT project manager is not involved in the project's activities on a day-to-day basis.

Advantages

The V model's strength is that it correlates the concept and the functionality one hopes to achieve with the design (concept of operations, requirements, and design) prior to coding. This step-by-step development sets up a well-defined system verification and validation testing phase.[86] The protocols for validation and verification testing are defined at the time that concept of operations and requirements are being finalized.
An advantage of the V model is that its structured approach improves the odds of getting the desired product. There is reduced risk of uncertainty and test failures later in development because of the emphasis on technical planning at the front end; traceability helps mitigate scope creep and reduce uncontrolled cost increases.

Limitations

However, this level of structure can also be a disadvantage. The V model can be slow to completion if system requirements are loose at the outset, because the project is slowed by the necessary iterations among design, requirements, and the Concept of Operations steps in order to define the final product. Until those steps are aligned, the build cannot start. Similarly, the V model is not as flexible as some alternatives if external factors, such as the private market of mobile phone-based mobility applications, drive change.

Level of Federal Effort

The V model, as a "predictive" approach, entails substantial effort from the Federal contracting team during the procurement phase. High levels of activity are required to define detailed scope of work, project controls, reporting requirements, and schedule of deliverables. During the development phase, effort is moderate and intermittent, including:

- PM-level financial oversight. (If the contract type is other than firm fixed-price (e.g., cost-plus), this requires additional effort.)

[85] A "systems engineering analysis" is required for all state and local ITS projects using Highway Trust Fund monies [23 CFR §911(a)], and DOT has developed the systems engineering V for ITS and provided extensive training and guidance in its use; it is familiar to many if not all involved in the DMA Program. This approach to the planning of ITS systems, including their hardware components, is an extension of the V model for development of software.
[86] V-Model. http://en.wikipedia.org/wiki/V-Model_(software_development) .

- Periodic but infrequent multi-day documentation reviews when project moves from one major development phase to next.
- Review of project status reports and participation in teleconferences, if specified.
- Response to requests for policy clarification.

Procurement Option

Because the V model requires a contractual relationship between the sponsor and the contractor, development services using this approach should be procured using traditional procurement following the FAR.

The Agile Development Option

The Agile model is a group of methods based on iterative and incremental development, rather than a single prescription for development activity. Unlike the V model, which moves all application development activity forward sequentially toward product completion, Agile methods break tasks into small increments. Development cycles are short time frames (time boxes) that typically last from two to six weeks. Each cycle involves a team taking one piece of the overall product through a full software development cycle, from planning through requirements analysis, design, coding, unit testing, and acceptance testing. The goal is to produce a functional, customer-acceptable release—of a single piece of the project—at the end of each development cycle. At project completion and during successive development cycles, the integration of the pieces must be iteratively managed. Multiple integrative iterations may be required to release the complete product.[87]

Team composition in an Agile project is usually cross-functional and self-organizing. Agile methods emphasize face-to-face communication over written documents; more geographically dispersed efforts rely on videoconferences if available, but teleconferences at a minimum. No matter what development specializations are required on the team, each team will also contain a customer representative (the "Product Owner"). The role of this individual is to be unfailingly available to the developers to answer mid-iteration problem questions, and to assure that the emerging product addresses the needs of stakeholder users.[88]

Advantages

The reported advantages of Agile are flexibility (especially when requirements are loose at the start), speed, efficiency, and product quality. Breaking the project into multiple small steps allows, and in fact requires, the development team to focus on the highest priority aspects of the project first, and to continually reprioritize project tasks to achieve a given goal. Lower priority tasks are left to the end, and may be skipped altogether if deemed unnecessary.

[87] http://en.wikipedia.org/wiki/Agile_software_development.
[88] http://en.wikipedia.org/wiki/Agile_software_development.

Limitations

The risk of contracting for Agile is that successful execution of an Agile software development project takes training and experience on the part of the contractor, and not everyone presenting themselves as experienced in Agile development is in fact capable of it. The burgeoning popularity of Agile has spawned a proliferation of training programs in the technique. Agile has received mixed reviews on its success in part because of unevenness in the skill of team leaders.

A further risk associated with Agile is that it is not aligned with typical Federal procurement processes. Typically, these processes separate the procurement phase, in which work scope and schedule are negotiated, from the development phase, in which the work is completed to specifications agreed upon in the procurement phase. This makes accountability clear—the contractor is responsible to complete the work as negotiated. In an Agile approach, the work is reprioritized and effectively renegotiated frequently, placing new responsibilities on the COTR and likely straining the capabilities of the Federal contracting team to respond in a timely fashion.

Level of Federal Effort

Agile, a more adaptive approach than the V model, requires moderate effort in initiating the project and heavy involvement during development. The scope of work is defined at a high level, relying on the contractor to manage the details of how it is to be implemented. During development, Federal involvement includes the following:

- Each Agile team in the development requires a Product Owner (Fed) or contractor proxy for daily involvement.
- Project management financial management activities are separate and ongoing.
- At end of each iteration (every 2 to 6 weeks), the development team demonstrates modular code for an OK for release, and the Federal contracting team must evaluate and approve it.

Agile unquestionably requires a far higher level of engagement in the project on a daily basis than do the other alternatives. The benefits are faster development through quick resolution of what otherwise might be rate-limiting questions, and the ability to make on-the-spot observation of how effective the procured team actually is.

Contracting the Product Owner role out is an option, with two caveats:

1) The Program cannot assign proxy authority to the contractor to make policy decisions; this individual has to have the technical expertise and institutional acumen to grasp the implications of questions coming from the iteration team and pass them along accurately to the federal contact. While doing so frees up the federal staff member substantially, the Program incurs the additional cost of paying the contractor.
2) The Federal contact for which the proxy is standing in still has to be permitted to treat the resolution of any incoming issues as a high priority.

Procurement Option

Because the Agile approach also requires a contractual relationship between the sponsor and the contractor, development services using this approach must be procured using traditional procurement following the FAR.

Open Source Development

The open source development model engages software developers and contributors, who may be globally dispersed, in collaborative activity via the Internet. These contributors may organize into teams that compete against each other to develop an application that best meets the stated development goals. This situation can be called "competitive open source development." In other development situations, there is no competition, and development is driven instead by the cooperative and donated efforts of usually intrinsically motivated developers. In this document, this development process is called "single-project open source development." Both development processes are open source in that contributing developers are not preselected by a sponsor before work begins. Also in both cases, the contributor communities are self-organizing, although in the competitive open source case the universe of contributors is partitioned into competing teams.

Here are two illustrative scenarios:

1. **Competitive Open Source Development Model:** Registered participants can access the application's ConOps and requirements. Individuals and groups may submit entries. A completion date is stated. Participants must agree to give DOT an open source license for use of their work, should they win. The announcement describes the Application Development Environment (ADE) and its tools; the use of the ADE is optional.
2. **Single-Project Open Source Development Model:** The prize is the intrinsic value of participating in the application development experience. The development is open-ended; completion is defined by release testing. The ConOps and requirements are posted within the Application Development Environment. The challenge gives an open invitation to participate the open source development of the application using the ADE. Each participant must give DOT a contribution agreement that supports open source release of the application.

Advantages

In general, single-project open source development offers the advantages of enhanced innovation (through the participation of many different individuals), product reliability (with large number of developers, the odds of errors and bugs being detected increases[89]), and low cost (because developers contribute their work). An additional advantage is that the end-product source code and related documentation are made available at no cost to the public.[90] Competitive open source development offers the potential to harness substantial innovation by motivated developers, in a sense offering the best of the V model and single-project open source. Innovation is encouraged by leaving the methods unspecified. By contrast, projects developed under V may preclude unforeseeable approaches, by virtue of the very detailed

[89] This is known as Linus's Law, named for Linus Torvalds, originator of Linux. http://en.wikipedia.org/wiki/Linus%27_Law
[90] http://en.wikipedia.org/wiki/Open_source.

specification of performance requirements. Further, competitive open source encourages innovation by allowing all qualified developers to contribute. Both V and Agile require selection of a single contractor team before work begins, which excludes some developers and their potentially innovative ideas.

Motivation for the competing development teams occurs through the establishment of clear criteria for success and a meaningful prize. Harnessing the competitive spirit among teams of developers can encourage total investment in the development effort by all teams greatly exceed the prize purse.[91] This is in stark contrast with a single-project open source development approach, where contributions are difficult to value and compensate.

Limitations

There are two significant risks with open source development:

1. There is no guarantee that the application will ever reach a usable end stage. Some open source projects fail to attract a sufficient proportion of developers committed to the completion of the project. This is especially true of single-project open source development. Mitigating the risk of failure to reach a usable end stage requires careful design of the system for motivating developers.
2. The intent of single-project open source development is to be able to make the end product available under an open source license. Failure to manage and track the terms under which participants contribute code can fatally compromise the open source status of the product and lead to charges of infringement.

Level of Federal Effort

Single-project open source development requires fairly low investment of effort up front and very little during the development phase. At project launch, the Federal team must identify project goals and rules and tools for collaboration among contributors. During the project development, there is no financial oversight or project management activity. However, active involvement by a Federal subject matter expert may be necessary to guide and engage the developer community to secure acceptable results.

Competitive open source development involves the creation of a software development competition. At the time of project launch, the rules of the competition must be specified. These rules include terms on eligibility, cooperation, specific objectives that a development team must achieve to win the competition, and the incentives or prizes for doing so. Also at initiation, the Federal team must identify and reach out to the appropriate developer communities to ensure adequate participation. During the development phase, the Federal team may need to answer questions from competitors and continue outreach to otherwise engage developers in the competition.

Procurement Option

Whereas open source development requires that a broad range of developers be included as potential problem solvers, it cannot be encouraged through a traditional FAR-based

[91] McKinsey & Company. *And the winner is…: Capturing the promise of philanthropic prizes*. http://mckinseyonsociety.com/downloads/reports/Social-Innovation/And_the_winner_is.pdf.

procurement. It must be managed through a challenge. (See Appendix D for more information on administering challenges.)

5.3 Risks to Consider When Choosing Development Options

The development options presented in the previous section offer opportunities but carry risks, specifically in four[92] areas:

- Intellectual property
- Code sensitivity
- User need uncertainty
- Degree of innovation

Intellectual Property

Risks: Intellectual property (IP) can complicate open source development in two main ways:

1. There are a few applications[93] being considered that serve as a component of vehicle on-board control systems and that are likely proprietary to their manufacturers. In this situation, it is expected that each manufacturer would want to have control over the development of any application to enhance the performance of the existing system. It is equally likely that the manufacturers would not grant the necessary inbound license, given the risk of exposure to their proprietary source code. In this situation, these applications may be considered unsuitable for development within the OSADP.

2. Depending on the ConOps for the DMAs, some of them are likely to incorporate proprietary subroutines. (See the bundle-specific chapters in the body of this document for more detail.) In many cases, these subroutines may be able to operate in an open source environment as compiled executables with which the publicly viewable source code may interact. Depending on licensing terms, they may also be incorporated as source code.

Mitigation: Careful management of licensing is the key to managing IP issues effectively, in most software development situations. Proper licensing will be essential for managing the second risk, on the incorporation of proprietary code. In the case of the first risk, however, it may be impossible to alleviate automaker concerns about exposure of commercial secrets. If it is possible, it is likely that development will have to proceed under tightly controlled conditions.

Code Sensitivity

Risks: Exposing source code to inspection by a broad audience may introduce unacceptable risks to public safety and/or commercial operations in two primary respects.

[92] A fifth characteristic of DMAs that should influence the selection of a development option is the degree to which its development should be integrated with other DMAs. This characteristic is not analyzed in detail here, but there is a brief discussion of the issue in Appendix F.

[93] CACC, D-RIDE and F-ATIS; possibly also MAYDAY, if the R.E.S.C.U.M.E. bundle retains the application.

1. It is possible that contributors to the code could insert Trojan horses and/or backdoors in the code that would allow them to access commercial or private data or control safety-sensitive public infrastructure such as traffic signals.

2. Allowing a broad audience to inspect the source code of either safety- or commercial-critical applications *may* make it easier for malicious programmers to identify vulnerabilities that could be exploited during DMA operation.

Mitigation: To lower the probability of insertion of malicious code, mitigation options include quality control processes that a) manage access to the code development environment and b) ensure the reliability of any released code.

Regarding access management: security controls can be expected to safeguard the code (and the public) with reliability in the applications development environment. Access to the emerging source code can be controlled through governance decisions regarding permissions and the implementation of user access control and other security technologies. Additionally, if the Program wishes to reduce the risk of a malicious actor's participation in the development project, it can choose to use procurement of development that uses either the V or Agile model; in both models, the developer awarding the contract defines project membership.

Rigorous pre-release quality control measures must be implemented to ensure safe and effective operation of any Federally-sponsored application, regardless of the development approach used for a particular DMA.[94] Such quality control measures would include functional testing as well as code inspection to ensure data security and code resilience in the face of hacking attempts.

The second issue—exposing vulnerability—would affect Program decisions on releasing source code to the open source repository. If disclosure of the details of sensitive source code does, in fact, create an unacceptable commercial or safety risk, then release of the relevant source code is not permissible. This observation is independent of the software development approach used to prepare the DMA for release. Mitigating the risk, if it is real and substantial, is possible only by preventing release of the source code.

[94] See Appendix B for a discussion of pre-release quality control.

User Need Uncertainty

DMAs differ in the extent to which their performance requirements can be established before development work begins. In cases where requirements cannot be established reliably, the development process must adapt to new information discovered as it proceeds.

Risks: There are two reasons that lead to risks around user need uncertainty.

1. User needs may be difficult to discern. This may be true for DMAs that offer functionality very different from those available now. Where functionalities are similar, user communities can be identified and their needs can be estimated based on the performance of existing applications.
2. User needs may change over the course of the development process. New data may become available that enables new functionalities and engenders new needs. For DMAs that have close analogs in the private sector, those private sector applications are prone to evolve, meaning that the DMA performance requirements must also evolve to ensure the DMA's value and relevance.

Mitigation: Predictive approaches, wherein objectives are established in detail before work begins, are not well suited to conditions of changing or uncertain DMA user needs and performance requirements. In these cases, the V model and competitive open source development approaches are to be avoided, and the adaptive approaches—Agile and single-product open source—are preferred.

Degree of Innovation

Some DMAs will be relatively straightforward extensions of existing functionalities. Others will require the implementation of new functions or substantial improvement of existing algorithms. Dynamic speed harmonization or mileage-based user fee applications are examples of this latter situation.

Risks: For DMAs where substantial innovation is necessary for successful execution, using a development approach or working with a contracting team that cannot explore and/or invent a broad range of potential solutions and technical approaches is likely to result in a product that fails to meet defined needs.

Mitigation: The main approach available to the Program to mitigate this risk is to choose a development model that allows for and encourages disruptive innovation. Competitions and challenges are useful models, as are exploratory studies.

Competitions and challenges require articulation, to some degree, of a desired goal, but do not specify exactly who may participate in reaching the goal; describe what methods may be used; or as provide details that might constrain innovation in unforeseen ways. By leaving open the choice of methods, project sponsors have the opportunity to be surprised by the creativity of the developer community.

By contrast, the V development model entails the creation of detailed software requirements. The level of detail required by the contracting mechanism, such that both sponsor and contractor can be sure that the required software functionality is delivered, is prone to hamper innovation. Further, use of the V development model requires that a particular development team be selected up front. This ensures that innovation that could be brought to the project by developers not selected during the procurement will not be incorporated.

Agile development is more fluid than the V model. It begins with less detailed requirements, leading to lesser constraints on methods. It is also adaptive to new information and insights by design. As a result, it can result in substantially more innovative projects. However, it may also offer lesser opportunities for innovation than competitive open source because, again, a particular development team must be selected before development can begin.

5.4 Additional Considerations

Challenges

As noted earlier, the critical risk associated with open source development is that improper handling by the developer of intellectual property arrangements could seriously jeopardize the DMA Program's ability to offer the completed application under the open source license consistent with its intended use. Managing that risk should be the criterion the DMA Program uses to choose the form of challenge.

Inviting participants into a single open source project conducted on the OSADP would give the DMA Program direct control of the terms and tracking of contribution agreements; a competition in which participants are responsible for all upstream activity and then submit the application to be judged gives no such control. If the application to be developed is one for which there already are related applications in commercial circulation (as is true for the ENABLE ATIS applications), we recommend that the development challenge be single-product, and that the competition prize approach be avoided.

Procurements

If the V model is used for a project where collaboration is needed with other projects being performed, the RFP must speak to this expectation and require the developer to post source code to the OSADP for review.

Program-level governance decisions will assign the authority and responsibility that developers in such projects should have in relation to reviewing/commenting on related projects' source code, and considering and responding to comments on their own source code.

In stand-alone V projects, project governance (roles, responsibilities, and decision-making processes and authority) are defined in standard operating procedures within the contractor's organization.

With regard to procuring projects using agile methods, we have noted the risk to project success associated with developer training and experience in these methods. To manage this risk, we recommend that the DMA Program's review of all proposals involve not just a paper review but also a live presentation in which candidates must present evidence of a successful track record.

Use of the Agile Method

Agile unquestionably requires a far higher level of engagement in the project on a daily basis than do the other two alternatives. The benefits are faster development through quick resolution of what otherwise might be rate-limiting questions, and the ability to make on-the-spot observation of how effective the procured team actually is.

The cost is the commitment to Federal staff time, plus the opportunity cost of a Federal staff member being unavailable for other responsibilities and assignments for a prolonged period.

Contracting the Product Owner role out is an option, with two caveats:

3) The Program cannot assign proxy authority to the contractor to make policy decisions; this individual has to have the technical expertise and institutional acumen to grasp the implications of questions coming from the iteration team and pass them along accurately to the federal contact. While doing so frees up the federal staff member substantially, the Program incurs the additional cost of paying the contractor.
4) The Federal contact for which the proxy is standing in still has to be permitted to treat the resolution of any incoming issues as a high priority.

The result is that agile should be the choice only for the highest priority projects ***IF*** requirements are also loose ***AND*** speed to completion is seen as an urgent priority (as might be the case for certain applications with safety benefits.)

5.5 Summary

This chapter presented a framework for understanding the advantages and trade-offs in selecting the most effective development and procurement options for a particular DMA. Once the ConOps and requirements for the applications are delivered (approximately 2nd /3rd quarter of 2012), analysis can be applied to identify specific recommendations.

Of interest is the impact of development and procurement choices on the OSADP policies and structure. If the OSADP is to host applications development, the OSADP ConOps will need to add “Development Community” as a user case. Further, in the System Requirements that describe the portal level, flexibility in design is needed to accommodate development environments of different types:

- Some environments will need greater access controls and/or firewalls (for projects procured through traditional means or using the V model of development)
- Others will need the ability for community collaboration (challenges or open source development projects).

These requirements will affect Portal-level governance policies and systems rules of operation; and will help establish project-level governance policies.

6. Open Source Release Policy

Research suggests that there are no clearly defined policies addressing the optimal way for releasing open source software. This is most likely due to the highly collaborative and decentralized nature of open source development. Since active open source projects are constantly being improved and updated by the user community, it can be argued that they exist in a perpetual, or at least long-term, state of release.

The Portal concept facilitates the "release early, release often" concept, which was popularized by American programmer and open source advocate Eric S. Raymond. 95 Although opinions vary on how often to release, the general consensus is that the shorter the loop between release and user feedback, the more efficient the development process. Providing a virtual collaborative space will enable rapid developer/user feedback and can help capture valuable project documentation.

In the absence of an established release policy formula, the following elements are identified as having the greatest impact on the successful release of open-source applications and other products:

- Attracting developer interest and acceptance of the Portal
- Establishing a vendor community for service and support
- Developing clear criteria for acceptance and release
- Defining a business model for stakeholders

6.1 Attracting Developers to the Portal

As mentioned throughout this report, policies that are flexible and support openness and transparency as well as the retention of intellectual property create an environment that is attractive to developers. Additionally, a portal that is well-organized and has a range of tools to support an active community are important features of a successful portal.

Two additional efforts are key to attracting developers to the Portal – conducting outreach and using appropriate licenses.

Outreach to the Developer Community

Conducting outreach to the developer community is an effective strategy in trying to attract developers to the Portal. It generates awareness of the portal and increases interest.

There are several industry best practices in this area, which include:

[95] http://www.catb.org/~esr/writings/homesteading/cathedral-bazaar/ar01s04.html

- The UDSOT's Connected Vehicle Technology Challenge faced a similar challenge in creating awareness outside of the transportation industry of the opportunities and tools available for innovation around the connected vehicle concept. A number of outreach attempts succeeded in generating interest among a talented demographic that is often difficult for the government to reach through standard communication modes. A best practices report is available on the ITS Program's website to guide outreach for the DMA program[96]
- Proprietary software firms allow their developers to work on open source "side" projects. Some traditionally "closed source" firms permit developers to work on open source projects if they benefit the company or as a means of mitigating programmer "burn out" from working exclusively on company-dictated projects. A report by the open source consulting firm the Olliance Group, LLC notes that Hewlett-Packard "actively encourages its engineers to contribute and participate in the open source community," allowing them to "develop open source software on company time and property…with budgets and administrative support."[97] The report also mentions that Microsoft allows its developers to contribute to open source projects with prior senior management approval.[98] In certain cases, the global IT consulting firm, ThoughtWorks, has "hired open source committers to allow them to focus on important open source projects as part of their day-to-day work."[99] Encouraging involvement from developers at commercial firms may ensure more stable commitment to DMA projects because these individuals are professionally compensated. However, proprietary developers may be limited in the amount of time they can devote based on company work requirements or policies.

  Potential third-party mobility device manufacturers may also wish to establish a relationship with DMA projects in an arrangement that allows them to exchange open source development expertise for testing and support or assurances of compatibility.

- Last, developers are attracted to a portal that clearly defines DCM/DMA project goals, tracks developer interest (through metrics such as number of developers per project, bug reporting, uploads per user, etc.), and acts as a clearinghouse for peer-review, version tracking, and project documentation. A possible model is the web-based source code repository SourceForge, which also provides users with statistics (metrics), discussion forums, and online collaboration tools free-of-charge. [100]

Impact of License Type on Success of Open Source Projects

Appendix E presents a literature review and analysis of research regarding the impact of different types of licenses on developers' interest. On balance, the findings of the studies reviewed suggest that highly restrictive licenses present greater potential risks than benefits to the potential success of open source projects. One study (Colazo and Fang) found a consistent

[96] http://www.rita.dot.gov/press_room/press_releases/rita_001_11/html/rita_001_11.html
[97] Fan, Brian, et al. "Open Source Intellectual Property and Licensing Compliance: A Survey and Analysis of Industry Best Practices," Olliance Group, LLC (2004).
[98] Ibid.
[99] http://opensource.thoughtworks.com/
[100] http://sourceforge.net/

positive association between restrictive licenses and developer interest, developer activity, and project speed; the study also found that restrictive licenses negatively affect developer permanence on projects. This suggests that although restrictive licenses may stimulate greater initial activity, they may ultimately be detrimental to project success.

The only other indicator of a positive relationship between restrictive licenses and user interest (Subramaniam, et al.) is in a narrow context: restrictive licenses appear to increase user interest for projects aimed at non-developer users and system administrators.

There are appears to be strong empirical support for the idea that non-restrictive or less restrictive open source licenses engender greater developer participation and user interest than do projects with restrictive licenses. At the same time, however, the use of non-restrictive licenses raises the specter of open source software not remaining free and open; this possibility could in theory dissuade some developers from contributing to open source projects that lack strong copyleft licenses.

The findings on the effects of sponsorship on user interest suggest a way out of this dilemma, because they indicate that project sponsorship, and specifically sponsorship by a non-market organization like the USDOT, can help counteract potential user concerns about the likelihood of a software product that lacks a restrictive open source license remaining free and open. As the researchers of this particular study note:

> *One interpretation of this pattern of results may be that sponsorship trumps licensing in terms of its impact on users' perceptions regarding the likelihood of the software remaining free of commercial control (Stewart, et al.)*

Another attractive feature of a portal is an engaged user community that is interested in the new software and able to articulate needs in a way that establish clear requirements for the final product. It is recommended that the DMA Program consider strategies for supporting the inclusion of representatives of the user community beyond the traditional step of establishing user requirements, but keeping them engaged throughout the software development process and potentially establish them as lead adopters.

6.2 Establishing a Vendor Community for Service and Support

Establishing a vendor community to provide services, maintenance, and upgrades after product release is a critical step in open source success.

Some disadvantage of open source software that can be overcome with preparation and lifecycle planning:

- Missing commercial services such as support and service level agreements which impact the ability to run in commercial environments;
- Obstacles to commercialization;
- Missing or incomplete license attributes or missing warranty and liability clauses; or

- Non-compliance with inbound license terms.[101]

The following recommendations are steps that address some of these disadvantages and more successfully position open source applications for adoption.

Know and Support the User Community

Numerous articles identify the challenges that State and local agencies have with adopting open source software. Some agencies are leading-edge adopters and have provided bold and successful examples of incorporating open source software[102]; however, there are States that prohibit the practice given the higher installation costs and the perceived unproven nature of the software.

The Center for Strategic and International Studies has produced a survey of International, National, and State and City policies on the use of open source software. A review of these State and local laws and policies can provide a basis for determining what type of awareness, technical assistance, and training might support greater adoption with those areas that allow the use of open source software. 103

One of the recognized advantages of open source is the flexibility afforded the user in choosing a service and support vendor. While applications are offered and acquired at no cost, support vendor fees should be factored into the total cost of software ownership. In comparison, support for proprietary software is typically included by the firm that developed it and thus is paid for by the cost of the software and any required licenses. It is important to note that the growth of OSS over the past two decades has allowed for the creation of an established vendor support and systems integrator market.

While it is possible to use open source "as is" with no service or support contract, this may not be practical for complex or large scale releases such as the DMA project, though limited support may be appropriate during pilot testing and trials. The licenses that cover OSS are essentially terms-of-use and are not purchased. They also do not address intellectual property infringement, warranties, and liabilities, which are generally covered by vendor service and support contracts.104 Therefore, releasing open source applications without identifying qualified service and support vendors could leave end users vulnerable in the event of unforeseen problems with the software.

---

[101] Summarized from *Best Practices for Government: Managing Software Intellectual Property Assets*, www.Blackducksoftware.com .

[102] A well-known and frequently cited transportation example is the Tri-Met public transportation agency in Portland, Oregon.

[103] *National Open Source Policies, Center for Strategic and International Studies***.** Data Compiled by Robert Hinck, Philip Kimmey, Joshua Roberts, Dima Qassim, and Denise Zheng, March 2010. Located at: http://csis.org/files/publication/100416_Open_Source_Policies.pdf, pages 36-39.

[104] British Cabinet Office. All About Open Source: An Introduction to Open Source Software for Government IT, Version 1 (October 2011)

In establishing a support and services vendor community specifically for open source DMAs, it may be beneficial to consider government certification, such as through the Federal Supply Schedule. For example, IEC Systems, an independent systems integrator focusing on "open" non-proprietary energy management and control systems for the General Services Administration, has an established contract with the government which streamlines the purchase of its services by other federal agencies.105

The best approach would be to leverage existing vendors by educating them about the DCM/DMA project and defining a standardized service and support agreement, potentially drafted and vetted by the federal government.

Begin Transition Planning Early

Transitioning open source software into use begins with the initial planning steps and the identification of the intended use of the open source software. There are two key considerations at this stage:

- **Documentation:** The extent and depth of proper references and documentation that will be developed in conjunction with the software should be defined at the project's beginning. These include: documentation of functionality, guidance documents, configuration management, testing, and validation measures.
- **Vendor Services and Support:** Open source software is not necessarily "free" software, even if the software is provided at little or no cost to the user. Adoption of the open source software still requires installation onto hardware and incorporation into an enterprise system. Additionally, open source software requires regularly scheduled fixes and support. One key to success is to develop the interest of a commercial vendor community that will incorporate the open source software as part of its service offerings. The text box on the following page describes the commercial open source software, or COSS, industry. Although the organizations are predominantly commercial, they have done well in recent years by providing services that support the growing open source user community. The key difference is the transparency with which these vendors operate.[106]

The textbox on the next page illustrates the market's movement to develop these types of business services.

[105] GSA contract number GS-07F-0468T

[106] *Lowering the Cost of Business Intelligence With Open Source: A Comparison of Open Source and Traditional Vendor Costs*. Mark Madsen, 2010 Technology White Paper for Third Nature.

Commercial Open Source Software or COSS

A number of commercial firms have successfully formed new business services that have burgeoned into a growing industry over the last five years. Commercial open source services have evolved with recognition that companies and agencies are willing to pay for support, service, and other less tangible items like indemnification or certifying interoperability with other vendor's products.

A commercial open source vendor is similar to a traditional software vendor, but one difference is that the source code is not shrouded in secrecy. This enables more and deeper interaction between customers and developers, making the open source model more community-focused than the traditional model. These vendors provide the same services and support that traditional vendors do, frequently with more flexibility and lower cost. COSS vendors use elements of the proprietary model such as providing support contracts or selling non-open source components that can be purchased in addition to, or in place of, the free version of the software.

Summarized from: ***Lowering the Cost of Business Intelligence With Open Source: A Comparison of Open Source and Traditional Vendor Costs*** *. Mark Madsen, 2010 Technology White Paper for Third Nature, p.9.*

6.3 Identifying Business Models

Open source has been slowly gaining traction among the public sector on a global basis. A common goal of the policies reviewed is to encourage consideration of open source by removing obstacles in an organization's procurement procedures. Many public sector agencies, in particular, are biased toward proprietary options. Inclusion of open source software has the immediate effect of expanding the range of choices from which to select the most acceptable solution, and it can also generate financial interest in the open source development community and its activities.

International governments are increasingly turning to open source software as a viable alternative to traditional proprietary options for a host of reasons. The cost of proprietary licensing can represent a significant financial burden for large public organizations, thus driving the desire to move towards an open source system. The additional gains of greater customization, timely security updates, and freedom from contractual lock-in with a single developer have enticed many public entities to make the leap to open source. Policy changes that promote OSS as a viable option may be a crucial first step in setting the conditions for defining a business model.

The Icelandic government, which released its Policy on Free and Open-source Software in December 2007, recently announced a 12-month initiative to get its biggest public institutions—all the ministries, the city of Reykjavik, and the National Hospital–on open-source.[107] A major tenet of Iceland's open source policy is "to remove barriers in purchasing procedures which

[107] Brown, Mark. "*Icelandic government makes a push for open-source software*," *Wired.co.uk* (23 March 12), http://www.wired.co.uk/news/archive/2012-03/23/iceland-open-source-software (accessed April 3, 2012)

favor the buying of proprietary software on the market."[108] Similarly, the British government, which introduced its open source policy in 2004, has the goal of ensuring "a level playing field for open source and proprietary software."[109] The aim of these policies to afford some measure of equality between OSS and proprietary options through procedural change is an important move toward opening new markets for OSS support vendors and device makers.

6.4 Project Selection and Release Criteria

With attraction comes interest in how the Portal operates and processes project selection and release. First is the question of how a new DMA project can be started in the OSADP. Second is the process by which a DMA is released from the OSADP, either in the open source repository or as a compiled executable application.

Initiating a DMA Development Project

Initiating a project requires two considerations – does the project meet the criteria for selection and, if it does, what development approach should be selected?

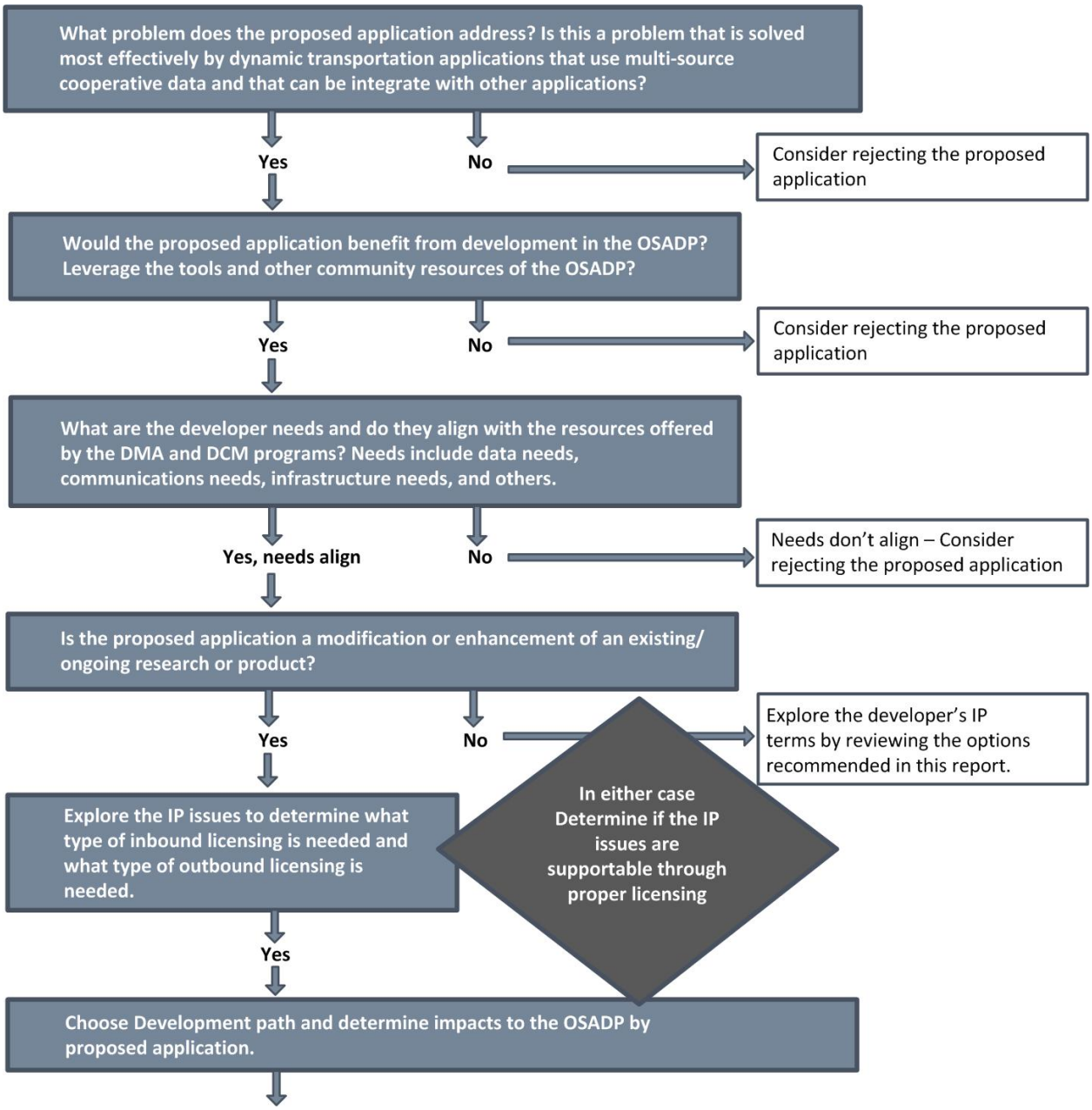
Selection Criteria

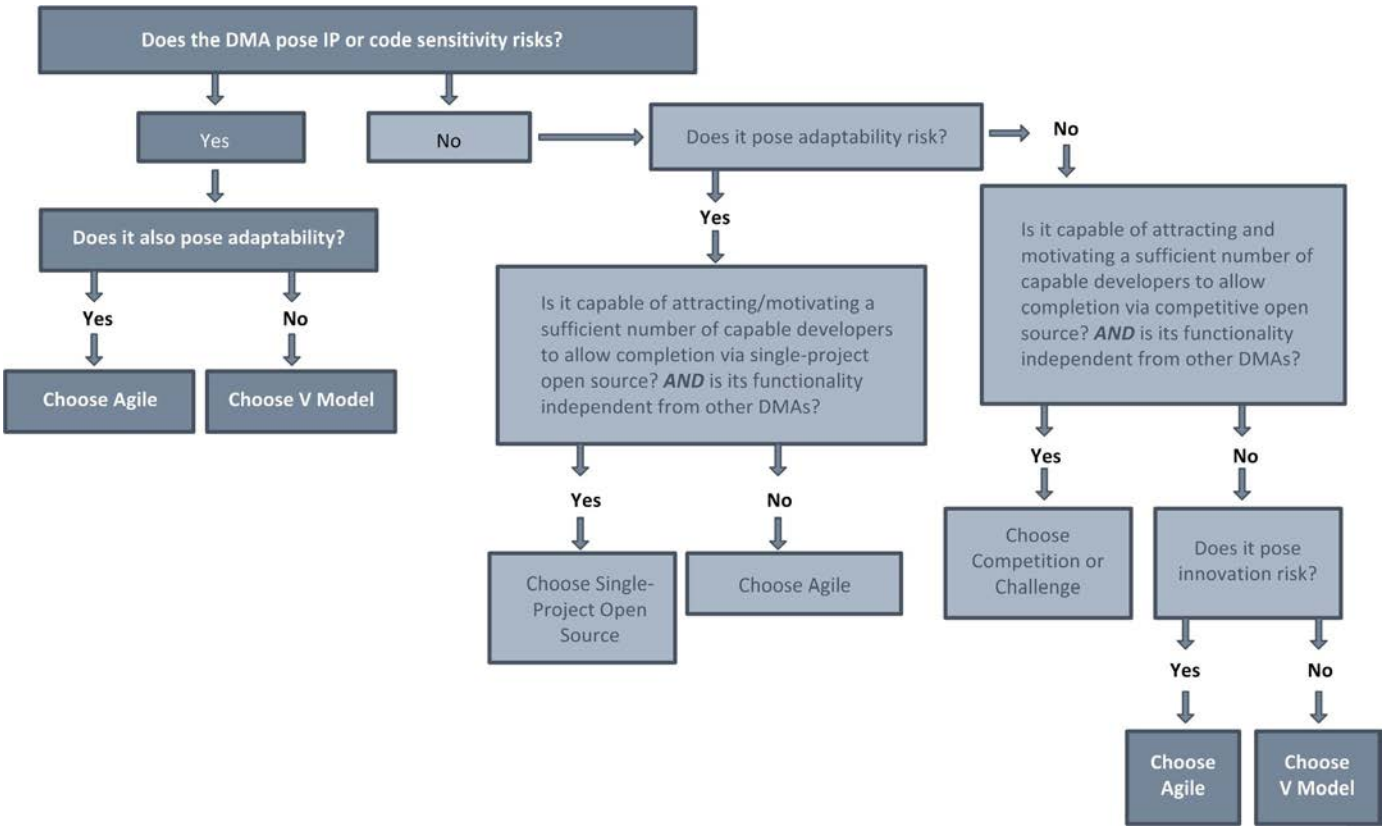
The question of how to select new DMA project is one that will need to be considered as one of the first steps by Program and Portal governance groups.

The selection criteria build from the criteria that were used in selecting these first six DMA bundles. Added to those criteria is the path for analyzing the risks and selecting the development approach (similar to Chapters 2-7). When combined, the proposed, preliminary decision tree is graphically illustrated on pages 80-81:

[108] Government Policy of Iceland. *Policy on Free and Open-source Software*. Prime Minister's Office (December 2007)
[109] Cabinet Office. *Open Source Procurement Toolkit*, http://www.cabinetoffice.gov.uk/resource-library/open-source-procurement-toolkit (accessed April 3, 2012).

Figure 6-1: DMA Development Decision Tree





Releasing a DMA Development Project

The issue of code sensitivity, which is used to distinguish among development options for the various DMAs, raises a related issue of quality control. The analysis in this document defines sensitive code as source code that would pose unacceptable risks if it were created through an open source process or if it were publicly viewable. If it were created through an open source process, malicious programmers could insert Trojan horses or backdoors that would allow access to valuable commercial data or control of public safety-critical systems during operation. It the code were publicly viewable, it is possible that malicious programmers could find vulnerabilities that they could exploit during the application's operation to access data or control public infrastructure. Together, these two possibilities raise two questions: How can the Program ensure that applications it creates can be trusted, and how can the Program decide whether to release a particular DMA's source code?

1. How can the Program ensure that its applications can be trusted?

Regardless of the development option by which the DMA is developed, and regardless of whether the DMA's source code is published, the Program must ensure that the DMA operates in a safe, reliable and accurate manner. The DMA must perform its functions without compromising user safety or privacy. Otherwise, the actions of the Program will cause harm, and the Program risks losing public trust, which is essential for it to carry out its work. Creating injurious DMAs could be programmatically fatal.

Therefore, all DMAs created by the Program must be rigorously tested and inspected. Functional testing is certainly required. Direct inspection of the code is necessary, in addition,

because malicious code such as a Trojan horse or backdoor may lie dormant until an outside action activates it.

There are industry standard practices for software quality control, and the Program must employ these practices at a minimum. It is the policy of the Mobility program that industry software practices are used as a benchmark. Further, it has been argued elsewhere that Federally sponsored software is likely to be held to a higher standard than private-sector software, as there are groups within society that are inherently distrustful of the Federal Government.

Again, this need for quality control does not differ among the development options—single-project open source, competitive open source, V model, or Agile. Although single-project open source in particular may achieve quality control through different methods, with many motivated programmers testing and inspecting the code in parallel, the need for quality does not change. Every DMA associated with the Mobility program must be trustworthy.

2. How can the Program decide whether to release a particular DMA's source code?

This is a policy decision that must be made on a DMA by DMA basis. The nub of the question is this: does exposing the source code to a wide audience pose unacceptable risk to the data that the DMA manages and/or the physical systems that the DMA controls? The issue can be decomposed as follows:

1. Does inspection of the source code allow malicious programmers to identify vulnerabilities that they could not otherwise find?
2. Who is affected by any loss of data or system control, and what is their sensitivity to that loss?
3. To what extent is the affected parties' awareness of the risks of publishing source code aligned with the realities of those risks?

At this time, further analysis is required to resolve these questions. The approach should include consideration of industry standard practices of code risk analysis, and should also consider the software procurement and management practices of state agencies that will be among the users of the DMAs.

7. Conclusions and Next Steps

The OSADP is a complex endeavor; it is more complex than previous open source portal efforts given the diversity of the envisioned applications and flexibility required to achieve a set of ideal program goals. The definition of this OSADP comes at a time, however, when the desire for and push to develop an open government is well supported.

The development of the OSADP will begin as the requirements for the DMA applications get underway. It is expected that these requirements will inform and influence the direction of the OSADP. Similarly, the policies that are established through key decisions will also inform and influence the direction of the OSADP. It is hoped that this white paper has highlighted those decisions and identified a set of next steps to provide the path to establishing OSADP policy.

A summary of the recommendations and next steps include:

Recommendation 1: Establish Governance Boards

Three levels of governance and associated roles and responsibilities are recommended, as follows:

Program-level Governance Decision-Making Board: Roles/Responsibilities

- ❑ Establishes the Portal Governance Decision-Making Group
- ❑ Works with the Portal Governance members to establish policies for a range of policies and processes (i.e., security, privacy, acceptance of new project, user access, application release, managing licensing and IP, among others) and rules of operation. Collectively, these two groups decide where/how flexibility can be tolerated.
- ❑ Responsible for financial resource commitments and conflict resolution
- ❑ Responsible for decisions regarding upgrade and maintenance

Recommendation:

- ***In the Research phase, members of this group should include the Federal DMA program managers.***

The Program-level board is constituted first and establishes the policy foundation for and focus of the Portal-level board. Together, these groups define roles and responsibilities, policies and processes, and standard operating procedures. The Program-level board remains available for critical decisions, assurance of continued funding, conflict resolution, and oversight of the timeline and progress. **Because the OSADP is being developed and operated under a Federal program, Federal policies for security, privacy, data release, and others will apply.** If the OSADP transitions to use beyond Federal research, the ultimate owners/operators will take on these roles and responsibilities.

Portal-level Governance Decision-Making Board: Roles/Responsibilities

- ❑ Establishes standard operating procedures for users
- ❑ Develops criteria for accepting new application development efforts and for releasing new applications into the repository
- ❑ Oversees/monitors operations and supports Project Managers
- ❑ Responsible for security and risk monitoring and response plans
- ❑ Active management includes review of new projects, licensing, validation and verification/testing of applications before release into repository

Recommendation:

- ***In the Research phase, members of this board are expected to include Federal application bundle managers, the OSADP contractors, and potentially some of the application developers.***

The Portal-level board implements and monitors the day-to-day operations. Their authority is derived from the Program-level board and includes the ability to decide on new projects or release of applications, based on the overall policy set by the Program. This board also plays an active role in making recommendations to the Program-level decision makers regarding portal changes, upgrades, maintenance, or other modifications.

Project-level Governance Decision Makers: Roles/Responsibilities

- ❑ Develops and proposes project-level governance to the Portal-level group. The project-level governance describes how the OSADP policies will apply to the application development community associated with each bundle/new application.
- ❑ Governance at this level is defined by two key factors: risks and the level of openness and/or control desired by the project lead and community. The level of openness is further defined by the terms of use and/or restrictions associated with the source code or datasets (described by any original or "inbound" licenses)

Project-level governance is determined through discussions with the Portal-level decision makers and is based on the policy directions established with the Program-Level group. Chapter 3 describes the different levels of governance and provides more details on the recommendations.

Recommendation:

- ***Under a contracting scenario (the Federal government funds the applications development), the decision makers are the project managers from the award-winning organization.***
- ***Under a challenge, or a non-funded development effort, these decision makers are the project leads.***

Next Steps:

- ***Establish a small Program-level Governance board comprised of the Federal program managers. Have this group establish the Portal-level Governance board consisting of the portal managers, technical experts, and user representatives.***

- ***Have the Program-level board establish objectives and metrics for the Portal-level board to achieve for risk acceptability, daily operations, and decision criteria (policy foundation).***
- ***Have the Portal-Level group develop user rules, standard operating procedures, and project acceptance/application release criteria. Document these policies and processes and incorporate into the Portal for transparent access for users.***
- ***Once established, have the boards define roles and responsibilities for ongoing operations.***

Recommendation 2: Form of Governance

It is recommended that the Portal-level governance begin as centralized (Portal-level board makes all decisions) and transition to a "federated" structure once standard policies and operating procedures are in place (project teams will assume governance/oversight efforts of monitoring for risks, establishing and implementing policies on openness and collaboration, developing licensing terms and restrictions, etc. that are specific to their projects). Depending on the structure of each project, the portal may eventually host a range of project-level governance structures that include "benevolent dictator" through group decision making models.

Next Steps:

- ***Develop a transition plan and timeline for evolving Portal-level governance from centralized to federated, based on user scenarios and anticipated risks.***
- ***Establish a set of procedures for the Portal-level board to follow when accepting a new project and working with the project lead(s) to tailor governance and oversight metrics in a manner that is specific to the project, its goals, and the level of new risks it introduces (for instance, risks in security, privacy, liability, or protection of intellectual property, among others).***

Recommendation 3: Develop a Comprehensive License Strategy

A comprehensive strategy for OSADP licensing will address processes and roles for applying "inbound" and "outbound" licenses; will review and determine the appropriate range of licenses acceptable to the DMA Program; and will establish processes for addressing exceptions.

Inbound licenses are determined by the owner of the intellectual property that is being brought into the OSADP. As part of both program-level and portal-level governance, processes will need to be established for reviewing the terms of inbound licenses and deciding whether those terms align with the DMA Program's open source approach (and thus whether the intellectual property will be allowed within the OSADP). An accompanying recommendation is for the staff that review the inbound terms to be cautious about accepting inbound products with patents.

Outbound licenses or the license terms accompany the source code and/or application to the release repository. In the repository, the source code becomes available, under both the original inbound license terms and the new the outbound license terms—assuming new intellectual property has been added—for further enhancements. Similarly, applications are released for transition with their own package of licenses that guide user terms of use and commercialization.

This report recommends three "outbound" license options:

- The MIT License (MIT/X11)
- The Berkeley Source Distribution (BSD-new)
- The Apache 2.0 license

An additional option is to attempt to place the released/completed products in the public domain, if there are no existing patents and if the developers accept this path.

This range of options reflects implementation of an open source policy that is flexible and supports both:

- **Open source development**—development of applications that are either Incentivized through challenge grants or requested by project lead(s) who seek to have collaborative development (see recommendations below on procurement and development strategies)
- **Open source release** —release of new applications as free and open software or release and availability of the source code for further modifications and enhancements). This is likely to occur with projects that are funded with Federal dollars.

While open source development and open source release are aligned well with the overall goals of the DMA Program, there is moderate probability that accommodations will be needed for protecting inbound intellectual property. Hence, a range of licenses is recommended for the OSADP. It is also recommended that a process be developed for new project developers to work with the Portal-level board to petition for use of additional licenses that are likely to be more restrictive. Such a petition process is likely to involve the Program-level governance board as well as the Legal Policy team who will analyze the impact of introducing a more restrictive license option and determine if fulfilling such a request meets the objectives of the program.

Next Steps:

- ***Establish a comprehensive license strategy by:***
  - ***Working with US DOT legal counsel to determine whether the appropriate level of open source intellectual property expertise can be made available to the DMA Program.***
  - ***In concert with the development of program-level governance, establish a set of processes and procedures that guide how and when licensing arrangements will take place.***
  - ***Ensure that the licenses and other considerations recommended in this report are aligned with US DOT policies.***
  - ***Based on these decisions, convene a public webinar or workshop to describe the terms and receive feedback on whether such terms and processes will work for developer(s).***

Recommendation 4: Analyze Risks with Applications Procurement and Development Processes to Ensure Flexibility in the OSADP Design and Policies

Criteria for accepting new projects for the OSADP must include the identification of risks. Key risks include:

- **Intellectual Property Infringement:** These risks include conflicts with intellectual property particularly when patents are unknown or not stated upfront as a project begins. If intellectual property rights are known at the beginning of a project, inbound licensing is the appropriate mitigation. If no prior rights or terms of use are described, the Project-level governance board will need to work with the Legal Policy team to determine acceptance.
- **Sensitivity of Code or Data:** These risks require that the OSADP provide greater protection for known intellectual property or sensitive data sets (those with some PII or those that can be linked with PII by associating the data with other datasets).
- **Level of Adaptability Needed in Development:** These risks include cost and schedule risks that result due to the level of (or lack of) definition of application requirements. Greater adaptability in development (and thus potentially in OSADP policies) is needed when:
    - Application requirements are unknown and flexibility is needed to incorporate new requirements as anew information or ideas arise
    - A quickly evolving market or market demand requires a faster development process.
- **Level of Innovation:** These risks result from the complexity of an application that may require more iterative and longer development processes and/or suggest a higher need for more broad-ranging collaboration, and thus may require greater accommodations of OSADP policies.

Until the actual applications are known, a comprehensive risk analysis is not possible. At a general level, though, there are a range of policy and technical options for mitigating these risks, including a thorough understanding of the impact of choosing one procurement mechanisms and development process over another with any given application.

Next Steps:

- ***Develop a checklist of information that is needed from project leads before accepting a project for procurement or into the OSADP***
- ***Analyze the potential applications to describe their risks and choose appropriate procurement and development strategies***
- ***Work with the Legal Policy team to develop guidelines for accepting source code, data sets, or other software with unknown patents***

Recommendation 5: Effective Use of the OSADP and Adoption of New Applications

Two risks that are associated with any open source portal are (1) the potential lack of interest by developers in using the portal and/or (2) the risk that applications developed within the portal will not be adopted for use.

With regard to use of the portal, there are two approaches that are key to success:

- Ensuring that the portal has transparent policies and useful tools
- Ensuring that developers are aware of the portal and its opportunities

The development of support for the applications after they are released is a critical element in adoption. While some States have laws or IT governing boards that provide disincentives against or prohibit adoption of open source applications and systems, more and more States and cities are turning to open source applications as a way of reducing initial investment costs and providing a more open and collaborative form of government.

To encourage adoption, particularly by the public sector, a strong vendor community that is capable of supporting maintenance, upgrades, and recovery (in the event of failures) is critical. Such a community is best developed simultaneous with the OSADP and requires transparency with applications development to establish the learning and training for their workforce.

In both instances, a focused outreach effort to create awareness is important. The recent ITS Connected Vehicle Technology Challenge provides an example of the difficulties and successes associated with outreach to a development community beyond the transportation community. The lessons learned are captured in a document titled, *Connected Vehicle Technology Challenge: Communications Assessment* and contains new ideas for outreach.

Next Steps for Attracting Developers and Encouraging Adoption:

- ***Ensure that OSADP policies and the portal itself supports openness and transparency to the extent possible, given intellectual property concerns.***
- ***Ensure that the OSADP is well-organized and has a range of tools to support an active community.***
- ***Engage the user community throughout the software development process and potentially establish them as lead adopters.***
- ***Understand the challenges to adoption faced by the user community including State and local laws that may prohibit the use of open source software and/or policies by State IT governance boards who view open source software as unproven and costly. In particular, work with organizations such as NASCIO and/or AASHTO to determine what States have such challenges.***
- ***Facilitate development of a vendor community by:***
    - ***Planning for and supporting development of a range of proper documentation that will guide the user.***
    - ***Planning for and engaging the vendor industry that will integrate the open source software into their service offerings, which will support the user community in installation and in receiving regularly scheduled fixes and maintenance.***

Recommendation 6: Future Transitioning of the OSADP

It is expected that if the OSADP were to transfer out from Federal funding and oversight, the owners/operators of the OSADP would inherit the roles at the Program-level and Portal-levels. To anticipate the policy support needed to transition the OSADP from Federal oversight, further research and analysis is needed.

Next Steps:

- ***Perform research to identify the value and uses of an OSADP:***
  - ***Survey a variety of types of organizations who might wish to assume ownership and operations and identify their purpose and potential uses as a means of deriving the value proposition***
  - ***Identify the factors and characteristics that are attractive to individuals and organizations other than the DOT, and identify the factors/characteristics that make the OSADP, in its current form, less attractive to potential new owners/operators***
    - ***Types of licenses***
    - ***Vendor community***
  - ***Develop an outreach plan to create greater awareness of the Portal outside of the transportation community***

APPENDIX A: Primer on Licensing Arrangements for the OSADP

As noted in Chapter 4, the biggest risks to the success of the DMA Program are in the area of intellectual property. Properly identifying, attributing, and thus protecting the intellectual property — i.e., source code and documentation—of the developers who create it is the critical first step for the DMA Program in order to be able to offer the new applications under open source terms. Similarly, if the new application has components that were already the intellectual property of some other entity, that ownership must be recognized in advance by the new applications' developers. The second critical step for the DMA Program is to obtain from all these developers the intellectual property permissions needed distribute the applications as open source.

Completing these two steps successfully will protect the DMA Program from claims of infringement, and protect users from the need to pay royalties and license fees to the developers. Licenses and contributor agreements are the key legal tools in protection and use of Intellectual Property and in mitigating the risks of infringement.

This primer is arranged in six sections:

- Section A.1 provides a brief set of definitions that set the basis for discussion in the remaining sections.
- Section A.2 describes the relation between the open source software licenses that developers and contributors will give to the DMA Program, and the licenses that the DMA program will offer the users of the Mobility applications, and the relevance these relations to the OS Portal. This section goes on to describe the terms that distinguish license types in their ability to take precedence over one another. These license conditions, and the assurance that the various inbound licenses are in alignment with one another and the outbound license, are the concern of the DMA Program's managers, working in consultation with US DOT counsel and US DOT Acquisitions. An important requirement for the OS Portal is to establish the institutional capacity for two types of licensing arrangements
    - "*Inbound" license arrangements*, which are the licenses and contributor agreements that developers give to the DMA Program in relation to a) development of the new Mobility applications and b) enhancement of the released applications once they are in the repository. These terms of these arrangements give certain permissions to the DMA Program that enable it to release the application under an open source license.
    - "*Outbound" license arrangements*, which are the licenses that the DMA Program will use when offering the completed applications on open source terms to users. The terms of the outbound license must be the basis for defining acceptable terms for the inbound licenses.
- Section A.3 addresses the choices the DMA program will need to make with regard to outbound and inbound license terms, identifies the key application characteristics that should drive the choice of outbound licenses, and makes preliminary recommendations, with the understanding that the final decisions on the license(s) for each application will be made by US DOT counsel. This section also discusses some of the considerations that will affect the receptivity to license alternatives the DMA Program might offer. The section concludes with an analysis of how the

choice of procurement approach will affect the timing of inbound license decisions and negotiations.

- Section A.4 describes the considerations for open source licensing of the non-software elements of the application.
- Section A.5 describes the OSADP functionality required to support the DMA Program's goals and objectives, first with regard to inbound licenses, and then outbound licenses. The role of the DMA Program in relation to inbound and outbound licenses is also discussed to clarify the division of responsibilities between the Program and the OS Portal.
- Section 5.7summarizes the overall guidance, describes gaps and next steps, and highlights some potential program risks for consideration.

Appendixes B, C, and D provide additional detail as background information.

A.1 Terms and Definitions

This section provides brief definitions for terms used as they are used in this white paper.

Contributor: An individual or set of individuals participating in an open source application development project; typically as volunteers.

Contributor Agreement: An agreement by which an individual contributor to an open source project grants sufficient rights for the parties operating the project to release the contribution as part of the project. A.k.a. "contribution agreement"

Copyleft: A method for making a software program (or other work) free of proprietary use and distribution restrictions, and requiring all modified and extended versions of the program to also be free of the same restrictions.

Conventional Software License: license in which licensor not only retains full ownership of the software intellectual property, but also restricts what the licensee does with the software in many critical respects, including restrictions on duplication, modification, and redistribution. (See Appendix B)

Copyright: A. legal device that gives the creator of a literary, artistic, musical, or other creative work the sole right to publish and sell that work, as well as to control the reproduction of that work, including the right to receive payment for the reproduction. A copyright is good for 70 years in many instances. 17 USC §101 extends the definition of "literary work" to include computer programs.

Developer: An entity (for example, a software development firm, nonprofit organization, or academic institution) or individual participating in procured contract development, as a volunteer contributor, or through investing on one's own.

Dual or multiple licensing. An arrangement in which the software licensor offers open source licenses to one market segment interested in that option and proprietary licenses to the remaining market sectors.

Liability: A manufacturer or seller's obligation to provide compensation for injury.

License — A grant of rights to a licensee to engage in conduct that otherwise would be a violation of the licensor's intellectual property rights.

Licensee: The recipient of permissions and restrictions on the use of intellectual property, which are granted through a license by the owner of the intellectual property.

Licensor—The owner of intellectual property, who grants permissions and imposes restrictions on the licensee's use of the intellectual property.

Patent: A grant made by a government that confers upon the creator of an invention the sole right to make, use, and sell that invention for a set period of time.

Proprietary software — Software in which the intellectual property owner retains all rights and licenses use of the software under conventional terms (see Conventional Software License, above).

Open Source—a philosophical approach to the development, modification and sharing of software that reveals the source code to users and permits them to make modifications and share those modifications with other users with permission granted in advance through licensing from the intellectual property owner.

Open Source License —contracts that allow users free and open access to source code and documentation that they do not own, enabling them to modify the software as desired. Open source licenses differ in their relative restrictiveness or permissiveness on this point; these differences are important and are discussed in the following sections.

Open Source Software— Software offered under an open source license. Open source software licensing is recognized in US and international law as an alternative to conventional licensing. The key feature of open source development is free and open access to the source code and documentation, because it enables users other than the original developer/owner to modify the software as desired.

Warranty: An assurance by the seller of property that the goods or property are as represented or will be as promised.

A.2 License Permissions and Flow

Licensing will come into play at several points in the applications development cycle and thus form a certain set of institutional requirements for the OSADP**.**

Flow of Permissions

The DMA Program will be obtaining open source licenses or contributor agreements from developers and offering the software to users under an open source license. These licenses form a "flow-through of permissions" from the application developer, through US DOT, to the end user, as shown in Figure A-1 below.

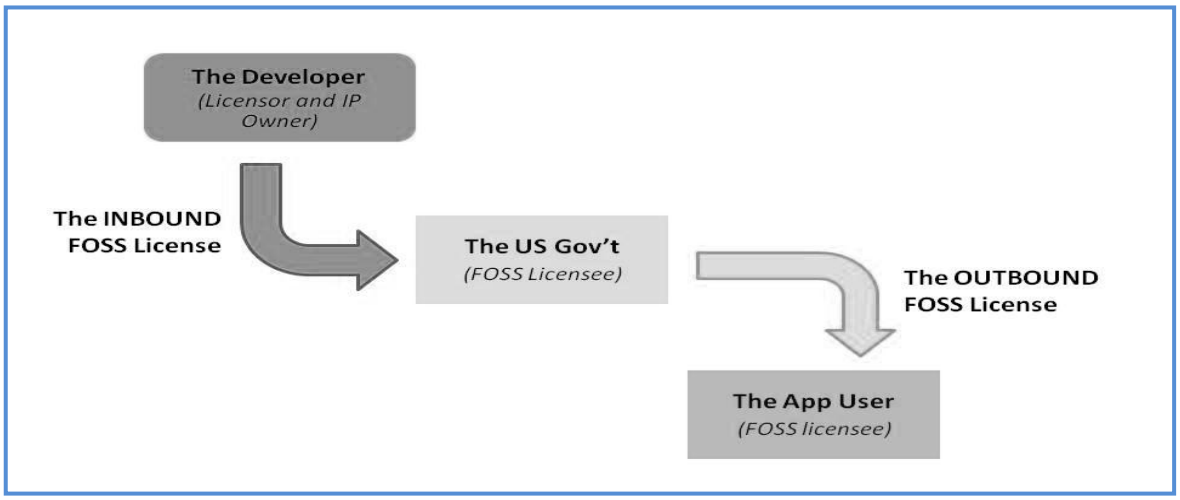


Figure A-1. Flow of Permissions

This simple figure is meant to reveal the distinction between *inbound* licenses– that is, the intellectual property rights associated with software applications being first developed and later enhanced for ultimate release by the DMA Program – and *outbound* licenses – that is, the intellectual property rights associated with the software applications when the DMA Program releases them. As described in Appendix A in the discussion of restrictive and permissive licenses, *inbound license restrictions cannot be removed, and flow through as a limitation on the terms of the outbound license and on any future licenses offered for downstream products.*

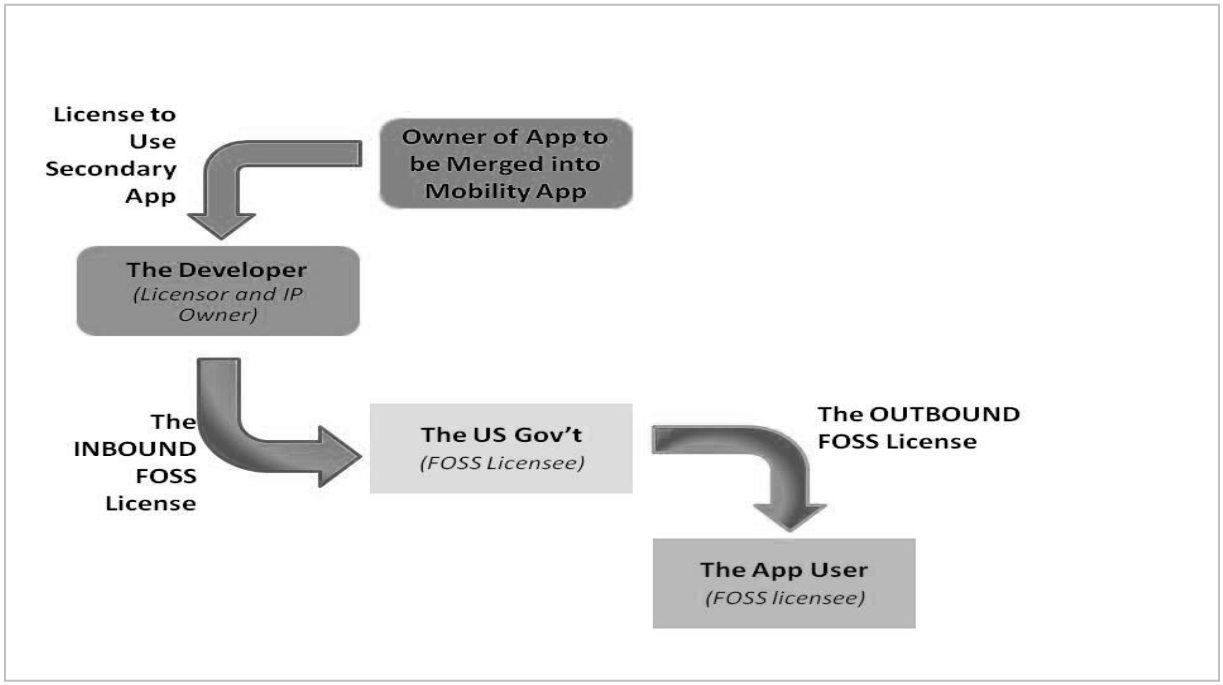
In practice, there are a number of possibilities that add real-world complexities to the flow, and so present challenges to the DMA Program. Here are two likely scenarios that the Program may face.

Compatibility of Licenses for Merged Software

One scenario is when an application developer wants to merge the source code of existing software that has some of the desired functionalities into the application under development. The FAR requires that *contractors obtain permission from copyright owners (contributors) before including copyrighted works, owned by others, in technical data to be delivered to the government.*[110] The FAR defines "technical data" to include deliverables such as software.[111] Therefore, if the software deliverable incorporates code from other applications, *it is the responsibility of the developer to make the necessary intellectual property arrangements with the owner.* **It is essential that the license that the developer gets from the secondary application's owner *also* have terms compatible with the intended terms of the Program's outbound license, because the terms of that secondary license will also pass through into the inbound license from the developer to the Program, and then into the outbound license the program offers (Figure A-2).**

[110] FAR 27.102(e).
[111] FAR 2.101.

Figure A-2. Effect of Inbound Secondary License on Outbound License



Compatibility of Inbound Licenses from Multiple Developers

A second scenario is when the DMA Program contracts with multiple developers working under separate contracts to provide source code for various parts of a single application, as might be true if an application is being developed iteratively over time. Each developer owns the intellectual property for the source code it produces, so each has offer the Program a license. **These multiple inbound licenses must each align with the terms of the outbound license, AND be mutually compatible in all other particulars (Figure A-3).**

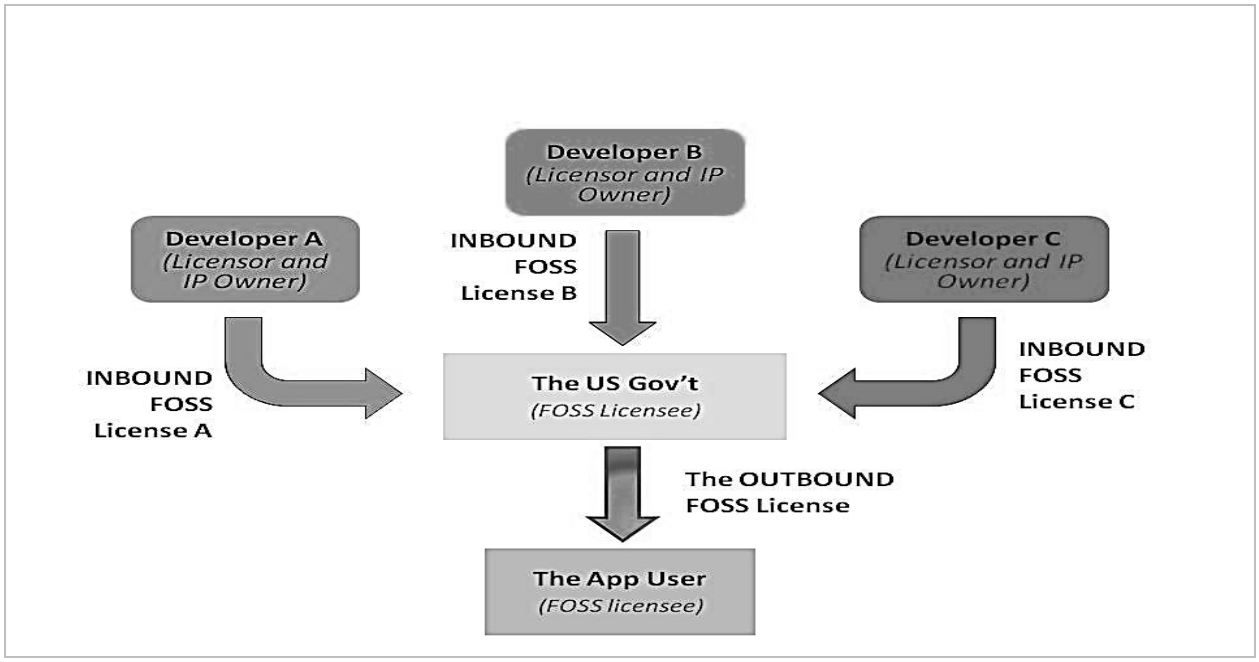


Figure A-3. Effect of Multiple Inbound Licenses on Outbound License

License Activity and the OS Portal

To understand the role of the Portal in relation to inbound and outbound license flows, it helps to think about the software development and enhancement activity at different points over the life of the Portal:

- **Time Point 1 (Figure A-4):** Portal activity is in the ADE only; no Mobility application has reached the point of release to the repository. Some projects need to communicate with each other to assure application synergy.

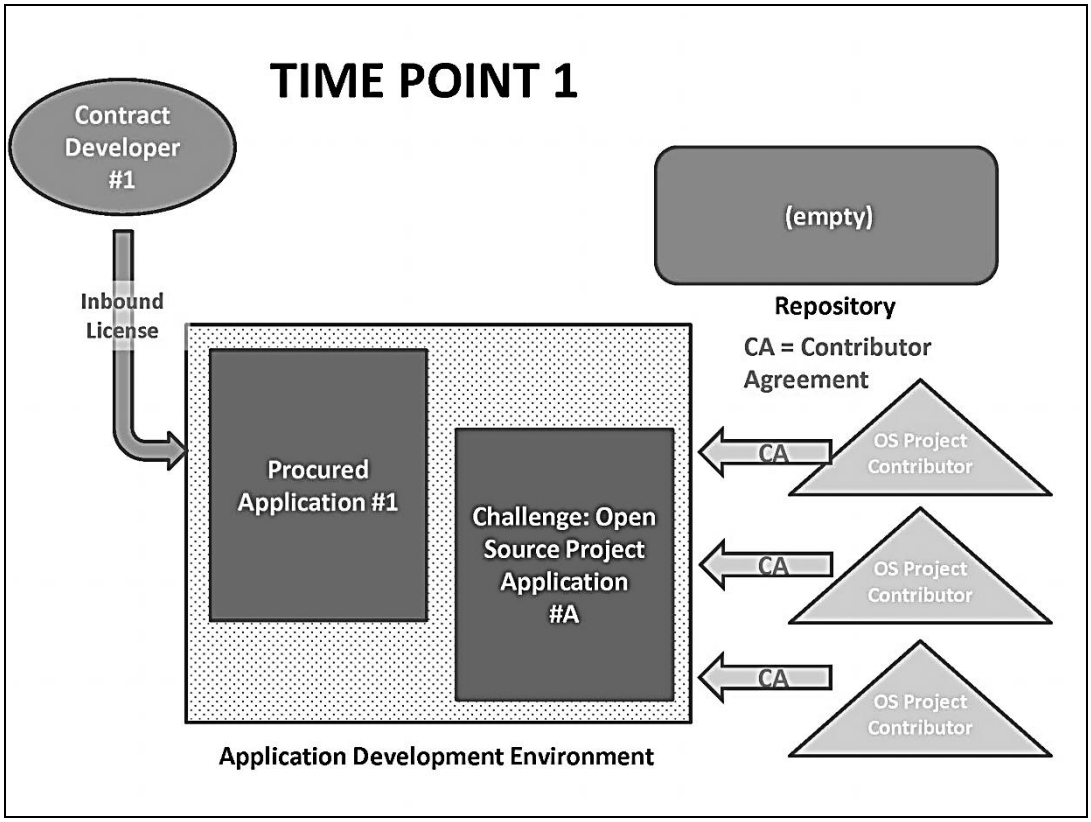


Figure A-4. Licenses in Relation to the OS Portal: Time Point 1

At Time Point 1, the procurement process will have included agreement on the license provided by the contract developer to the DMA Program; the contract development team members will register as OS Portal users and work on Application #1. Participants in the open source project created through a challenge will register with the Portal and provide the DMA Program with a contributor agreement.

- **Time Point 2 (Figure A-5):** Activity in the ADE in relation to development of the Mobility applications, AND activity in the repository: at least one app has passed acceptance testing and the DMA Program is offering it to users under an open source license.

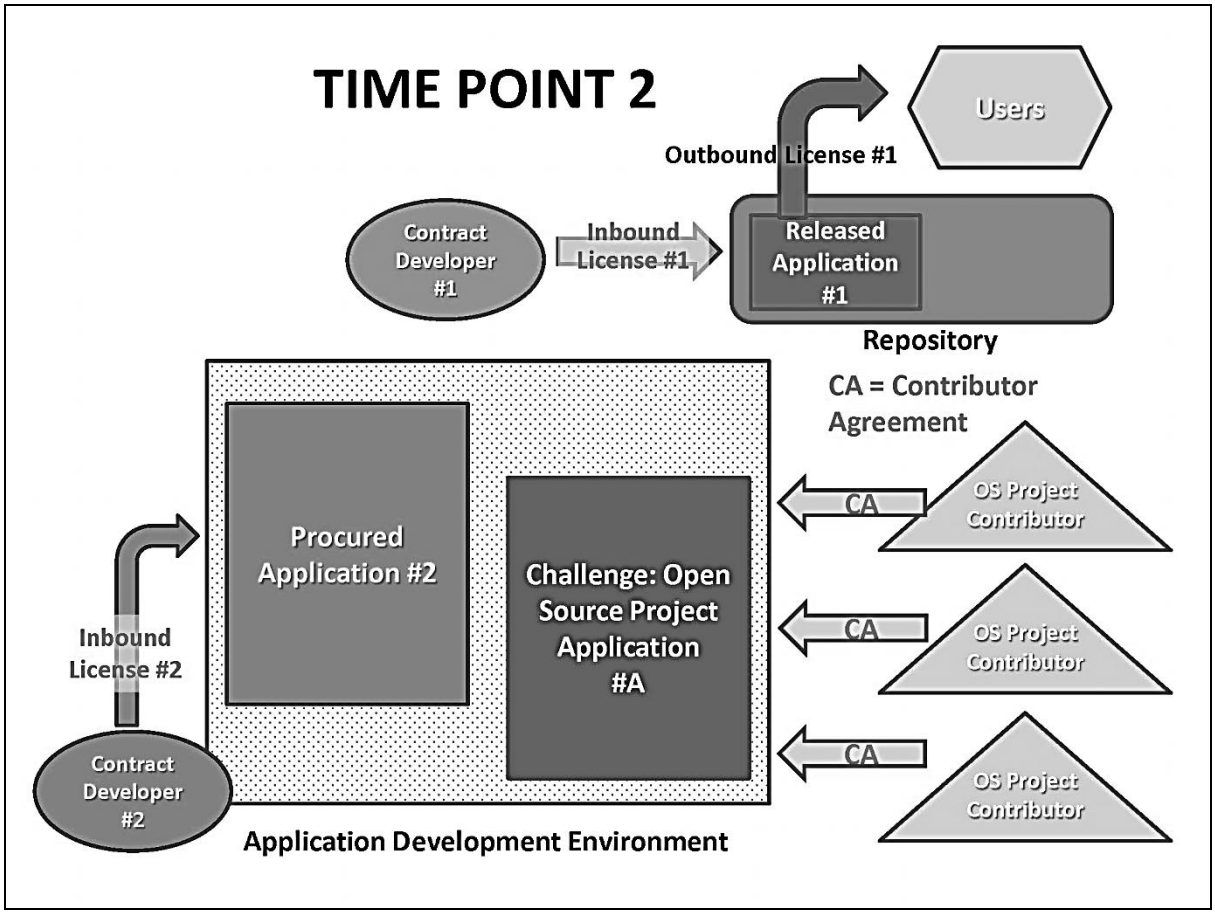


Figure A-5. Licenses in Relation to the OS Portal: Time Point 2

- **Time Point 3 (Figure A-6):** Development activity of the original Mobility applications has been completed. All accepted apps are posted to the repository, and communities of users form. Some contribute enhanced code back into the community. Contributors give Program permission to make the enhanced code open source.
- **Time Point 4** (**Figure A-6,** overlapping with Time Point 3): Some users see other possible uses for an application's source code, and **use the ADE** for a new project to incorporate that code into a new, 2nd Gen application. This new application may or may not be headed for proprietary licensing and commercialization.

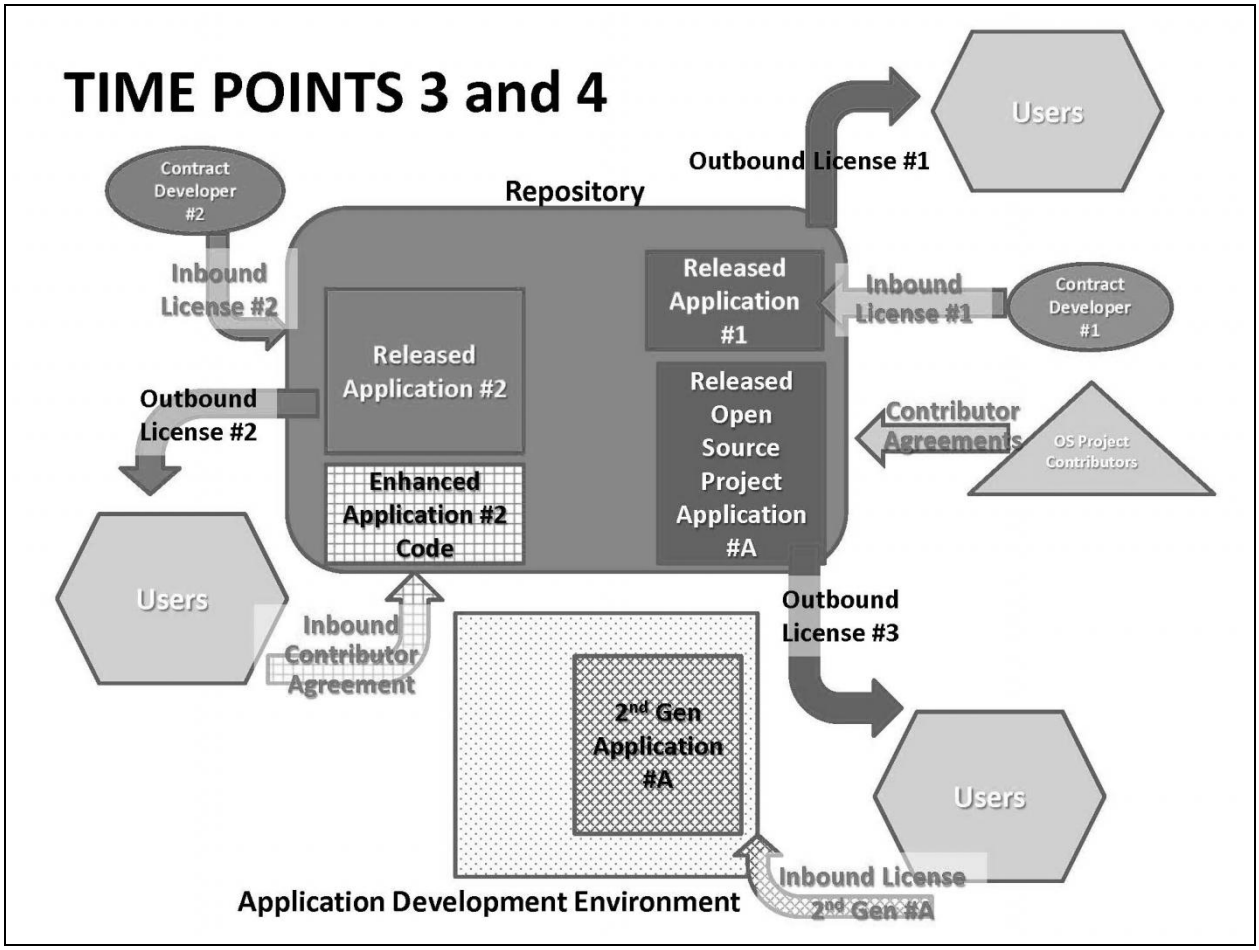


Figure A-6. Licenses in Relation to the OS Portal: Time Points 3 and 4

The significance of the flow-through of permissions for the DMA Program, and why the Portal's effective tracking of intellectual property is so essential, are due to the restrictions some categories of license impose on other categories.

License Categories

All open source software licenses comprise terms that govern how "open" the licenses for downstream applications that were derived from the original product must be. They fall on a spectrum ranging from the *restrictive*—meaning that any derivative product must also be licensed as free and open—to the *permissive*—meaning that derivative products may be licensed as free and open, proprietary, or not entirely free and open.

Restrictive licenses require that all software derived from the original product be also licensed as free and open. These licenses carry the "copyleft" stipulation, which allows a software program and

its source code to be used without consent from its creator/owner[112]. Once software is licensed with the copyleft provision, all "daughter" versions must contain the copyleft provision. The most widely used copyleft license is the GNU General Public License (GPL) and its variants. The Free Software Foundation offers a list of GPL licenses, along with guidance on how to choose among them.[113] Restrictive licenses encourage wide participation by users and developers in product modification and improvement. However, this type of license reduces the product's attractiveness for commercialization, because the licensee is unable to charge the sort of prices that could be charged when it in effect has a monopoly on code that is a trade secret.

Permissive licenses allow modified open source and object code to be distributed under non-open licenses in addition to open source licenses. Therefore, licensees adopting and then modifying open source applications can impose restrictions on downstream end users without having to disclose source code. Among the few conditions of use in such licenses are: (1) that the original licensing terms have to be present in future licenses for derivative works, and (2) that the original copyright notice is to be included with the documentation of the derived work. The Open Source Initiative reviews licenses submitted to it; those that conform to the Open Source Definition are posted as approved on the OSI website.[114] Three of the most frequently used licenses in this category, in order of increasing permissiveness, are the Apache 2.0 license, the Berkeley Software Distribution License (BSD) 2.0, and the MIT (X11) license.

The attributes of commonly used open source licenses in both categories are shown in Table A-1.

[112] http://www.gnu.org/philosophy/pragmatic.html
[113] http://www.gnu.org/licenses/licenses.html
[114] http://www.opensource.org/licenses/index.html

Table A-1. Attributes of Common Open Source Licenses

License Attribute	Restrictive		Permissive		
	GPL v3.0	LGPL v3.0 (library routines only)	Apache v2.0	Modified BSD	MIT (X11)
Copyleft?	Yes	Yes	No	No	No
Link to (aggregate with) other apps without affecting separate IP claims?	Yes	Yes	Yes	Yes	Yes
May be incorporated into proprietary code?	No	Yes	Yes	Yes	Yes
Release commercial works?	No	Yes (allows linking to proprietary code)	Yes	Yes	Yes
Create derivative works?	Yes, but all future modified code must be released under GPL	Yes, but all future modified code must be released under LGPL	Yes	Yes	Yes
Attribution (original copyright/terms)	Must be included in modified source code and distribution	Must be included in modified source code and distribution	Must be included in modified source code; licensor may also require inclusion in distribution	Must be included in modified source code and any documentation included with release	Must be included with modified source code
Contributor Agreement (who owns IP for contributions)	IP rights go to Free Software Foundation; alternatively, contributor disclaims all copyright and puts in public domain	IP rights go to Free Software Foundation; alternatively, contributor disclaims all copyright and puts in public domain	Contributor grants rights to original creator	Contributor retains	Contributor retains
Copyright indemnity	Yes	Yes	Yes	Yes	Yes
License for contributed patents	Yes	No	Yes	No	No
Disclaims all warranties	Yes	Yes	Yes	Yes	Yes
Disclaims all liabilities	Yes	Yes	Yes	Yes	Yes

A.3 Open Source Software Licensing and the Mobility Applications: Analysis and Options

The Mobility applications must be developed before they can be offered to users. Chronologically, the DMA Program will be procuring development and obtaining inbound licenses from developers and contributors before it uses an outbound open source license. However, **the lesson from the flow of permissions is that in order to obtain inbound licenses compatible with the outbound license, The DMA Program must decide in advance of procuring development what the terms of the outbound license must be.**

Choosing the Outbound License

Selection Factors

In a recently issued report[115] on lessons learned and best practices for military software, the US Department of Defense offers a number of recommended decision rules for selecting open source licenses compatible with program objectives. Many of them have applicability to the Program's situation, and are adapted below.

The DoD report lists these broad principles:

1. ***Choose a license that meets the applications expected uses*.** If the software is likely to be combined with another program, use at least one license that is compatible with that of the other program**.**
2. ***Use a proven, standard OSS license*.** Choose a license that has been certified as open source by the Open Source Initiative (OSI) *and* as Free Software by the Free Software Foundation (FSF). Non-standard licenses are often not truly open source, even when they are intended to be. As a result, code sequences from one program then often cannot be used in another program due to license incompatibility; even where they can be combined, the legal costs to the developer associated with interpreting a novel open source license can be substantial. Creating an OSS license is also very risky; it requires specialized OSS legal knowledge that contract lawyers and contracting specialists typically do *not* have, and even experts have made mistakes that were difficult to fix later.
3. ***Use a common OSS license*.** The common licenses have terms that are well understood and have a track record of acceptance.
4. ***Use a GPL-compatible license*.** Statistically speaking, most open source software is released using the GNU General Public License (GPL) version 2 or version 3 (in some part due to the Linux community's use of these licenses). This does not mean that all open source software must be released using the GPL, but rather that choosing a license *incompatible* with the GPL (both versions) is very unwise. The DoD report strongly advises *not to* use licenses known to be incompatible with the GPL, such as the NASA Open Source Agreement version 1.3 or the Mozilla Public License (MPL). Common OSS licenses that are also GPL-compatible include the MIT/X11 license, the new BSD license, the Apache 2.0 license, the Lesser GPL (LGPL), and the GPL. Analysis used for this Mobility policy report results in the

[115] DoD report, pp. 61-62.

recommendations of the MIT/X11, new BSD, and Apache 2.0 licenses, as explained in Chapter 4 (page 46) and later in this Appendix.

Yet another principle is the following:

5. ***Choose the open source license with an eye to the possibility that there may be reasons to change it in the future.*** If a licensor is unsure which license to use in a code release, it is prudent to use start with a relatively more restrictive license, which preserves the option of moving to a less restrictive one if circumstances so dictate.[116] However, this is most likely to apply only to the original licensee who owns every piece of the code. Once code licensed from others is used, the more restrictive terms will follow through and cannot be relaxed by downstream parties. The choice of license will need to factor in whether keeping restrictions in place is an appropriate goal for the code.

Analysis of Intended Users and Uses

The first principle above in choosing an open source license for an application is that it should support the application's anticipated uses. As the draft concepts of operations and requirements for the bundles start to take form in the first half of calendar 2012, more specific information with be available to the DMA Program and to US DOT legal staff to inform their choices. However, it is still possible at this time to draw some distinctions with reasonable confidence.

At this point, the intended end-users, and therefore uses, of the Mobility applications fall into two distinct categories: public agencies and individual travelers/commercial fleets. (While researchers and the interested public form a third category, the assumption here is that they would have interest in both of the other categories and so overlap with them.) Applications in the M-ISIG, INFLO, R.E.S.C.U.M.E., and IDTO bundles are intended for public-sector agency use, whereas the FRATIS and ENABLE-ATIS applications will largely be used by individual travelers and commercial fleets. FRATIS and ENABLE-ATIS applications are also more typically extensions of applications already in commercial distribution than are those in the other four bundles.[117]

From the standpoint of choosing an open source license, the issue is how these two categories map back to the DMA Program's objectives and criteria for determining the ultimate extent of its success: A basic Program objective is to "demonstrate promising applications predicted to significantly improve the capability of the transportation system to provide safe, reliable, and secure movement of goods and people." A success criterion is to facilitate the highest level of free and open competition in the commercialization of Mobility applications as well as their integration and maintenance by offering the applications under open source licenses.

Applications for Travelers/Commercial Fleets. The FRATIS and ENABLE-ATIS applications appear likely for commercialization given their intended use. The bundle technical teams anticipate that developers will likely want to modify or enhance the code in order take the improved application proprietary. Other entities may see commercial possibilities in services connected with the application's use (service support, manuals, training, and so forth).

[116] Meeker, p. 150.

[117] This statement summarizes conclusions made by the DMA Program Technical Team at our meeting with them on 4/20/2011.

The DMA Program has the option to use challenges rather than conventional procurement to foster development of some or all of the FRATIS and ENABLE-ATIS bundles/applications. If the decision for any of those applications is to use a challenge through Challenge.gov, AND the intent is to post the winning version of the application to the repository, then the DMA Program has to make a policy decision regarding how to manage the associated intellectual property.

The Terms of Participation for Challenge.gov state that all contributions are "in the public domain and may be reused, *except where governed by the individual Intellectual Property rules of individual challenges."*[118]

- If public domain is agreeable to the DMA Program, then the winning application is posted to the repository and released under those terms.
- If the DMA Program wishes to have a license from the winning contributor, then it must make this transparent via the challenge posting. Unlike the conventional procurement process, any negotiation of the license will occur only after the application has been developed.

Applications for Public-Sector Agency Use. The policy considerations for the public-sector bundles and non-proprietary applications are more complex. For public transportation agencies, one of the most challenging contractual problems associated with intelligent transportation systems has been the establishment of adequate rights for the agency with respect to a procured system's software. When an agency pays for the development of some custom transportation software, the restrictive proprietary contract prevents it from receiving the source code. As a result, the agency cannot adapt the software to its evolving needs or fix any bugs directly. The agency also may find that it must go back to the original developer/vendor for maintenance and future system upgrades. [119] The agency may then be "locked in" to the one vendor without any leverage to negotiate costs. The availability of open source licenses to software is intended to enable public agencies to avoid this situation.

The policy decision for the DMA Program applications in this group is how to best mitigate the risks of downstream cost for public agencies through license choices. The options are:

- To select licenses that effectively close off or at least limit commercialization of the applications in them—foregoing "success" in commercial terms.
- To select less restrictive or permissive licenses and find alternative avenues to achieve mitigation.

At the same time, the choice of license for some applications that may ultimately be integrated or even merged must take care not to create incompatibilities that block accomplishing this future use.

Depending on the application, an agency may want to use it in a variety of ways. Those uses and their implications for open source licensing compatibility follow.

- ***Physically linking*** *the application to existing (legacy) systems.* This is permissible under both restrictive and permissive licenses.

[118] http://challenge.gov/terms; emphasis added.
[119] http://www.fhwa.dot.gov/cadiv/segb/views/document/sections/Section8/8_3_2.htm, accessed 6/1/2011.

- ***Functionally integrating*** *the application into existing systems.* If the legacy system is proprietary, the system's license may prohibit this. The workaround is to use a plug-in.
- ***Modifying*** *the application's source code to fit agency needs*. This is permissible under both restrictive and permissive licenses.
- ***Merging the source code into existing applications*** *to create enhanced, hybrid versions of the existing applications*. The legacy applications have their own licenses, which could be proprietary, restrictive open source, or permissive open source.
  1. If the legacy application has a proprietary (conventional license), that license forbids changing the software; the two applications cannot be merged.
  2. If the legacy application is also open source:
     a. If the licenses are the same, there is no issue.
     b. If the licenses are different**, the terms of the more restrictive license will dominate.** That is, if one of the licenses has a copyleft provision, so will the license for the modified application.
- ***Sharing these modified applications*** *with other agencies.* This is permissible under both restrictive and permissive licenses; if either license has a copyleft provision, the copyleft provision then covers the hybrid. As a practical matter, the degree of restrictiveness becomes an issue if the agency wishes to distribute the hybrid application. Other agencies then need to address compatibility.

Preliminary Recommendations for the Choice of Outbound Licenses

The final determination of the best open source license(s) for each application will be made by US DOT counsel. However, the results of the above analysis can be mapped to the selection factors listed above to give a preliminary direction for these choices. The results are shown in Table A.2.

Table A-2. Preferable Options for Outbound Licenses by User Category

User	Considerations			License Options
	1.Builds on Other Open Source	2. Extent of Permitted Proprietary Use		
		2a. Permits Proprietary	2b. Permits Proprietary Library Only	
Public agency	Possibly	Yes	Probably	If new application is based only on other open source software, use similar license. If otherwise, consider one of three **permissive licenses**: • MIT/X11 • The *new* BSD license, or • Apache 2.0. If patent infringement is a concern, use ONLY Apache 2.0. Or, examine the benefits and limitations with providing the new enhancements or modifications in the public domain.
Individual Traveler/ Commercial Fleet	Possibly	Yes	Probably not	If new application is based only on other open source software, use similar license. If otherwise, consider one of three **permissive licenses**: • MIT/X11 • The *new* BSD license, or • Apache 2.0. If patent infringement is a concern, use ONLY Apache 2.0. Or, examine the benefits and limitations with providing the new enhancements or modifications in the public domain.

If we expand the options shown above for public-agency applications to take into account to the needed policy decision for each application:

- Choosing a weakly restrictive license would encourage continued open source collaboration on the application's library while at the same time permit its use for some, but not all proprietary purposes—in particular, desirable maintenance and support services. The core application could be preserved against being taken private.
- A permissive license would maximize the potential for commercial potential of the application, upgrades, and services. It carries the risk of potentially rendering the upgrades and services unacceptably expensive to agencies.

Invoking the DoD's fifth basic principle for choosing an open source license, **we recommend use of standard, permissive licenses.**

For both options, the recommendations based on the foregoing analysis are that:

1. The DMA Program must ensure that the concepts of operations and requirements for each of these applications that are presently under development reflect extensive end-user input; *and*
2. The definition of the core application accepted by the DMA Program must be extensive enough to include all essential features identified by those stakeholders. This is to ensure that public agencies will not be confronted with having some of those essential features treated as expensive proprietary add-ons.

Selecting the Terms of the Inbound License(s)

With the decision regarding the application's outbound license terms in hand, the DMA Program will be able to define compatible inbound terms, and select acceptable licenses accordingly.

Securing the inbound license(s) will be a negotiation between US DOT or another funding source and the developer, or developers. Each has interests to protect. On the developer side, the relative attractiveness of the procurement opportunity will depend on the developer's business model and interests. Below, we describe the terms that US DOT should seek to ensure compatibility between the outbound and inbound license(s), and then briefly discuss the negotiation from the developer's standpoint, and options for the DMA Program.

Acceptable Terms for Inbound Licenses: US DOT Perspective

The following conclusions are stated broadly, at the level of the license category. Assuming that US DOT counsel agrees with the above analysis, they will also determine the specific terms and compatibility among licenses within the permissive and weakly restrictive license classes.

- If the DMA Program's higher priority for an application is its commercial potential, and one or more outbound licenses with permissive terms are chosen, then the terms of the inbound license(s), as well as any secondary licenses feeding into them upstream, must also be permissive.
- If the DMA Program assigns lower priority to commercialization and favors a weakly restrictive license (an option for public-sector applications), then the terms of the inbound license(s), as well as any secondary licenses feeding into them upstream, must also be permissive or weakly restrictive. Strongly restrictive terms are unacceptable.

With regard to supported applications, the DMA Program should be alert to the issues posed by any possible non-US DOT support of application development. The issue pivots on whether the non-US DOT entity conducts the procurement of the development services and negotiates the contract and license terms, US DOT must ensure in advance that the funding organization is aligned with the DMA Program on the terms of the intended outbound license, and will negotiate the appropriate inbound license.

Acceptable Terms for Inbound Licenses: Developer Perspective

Developers have to perceive the DMA Program's requirements for the terms of an inbound license as a good fit with their business models before they will participate. In the conventional business model for software development, the object is profit. Capturing the financial yield from proprietary software is achieved by control over the code. Object code allows an end user to operate the software but does not enable the end user to make enhancements or modifications to the software or create derivative works. Access to the source code allows the end user to maintain the software, to make modifications and/or enhancements to the software, and to create derivative works.

Developers attracted to the opportunity to create the applications, or to enhance them post-release, may differ in their business reasons for involving themselves. Depending on what those reasons are, we anticipate the DMA Program will encounter varying levels of comfort and discomfort with the notion of open source licensing. If the discomfort level is not so high that the developer opts out of bidding during procurement, the DMA Program may find that it needs to trade off some of the ideal terms for the inbound license in order to secure the participation of desirable developers. The DMA Program will need to work with the US DOT legal staff on the fine details of the individual license terms to arrive at an acceptable choice.

However, over the past several years, some developers and software firms have recognized the growing appeal of open source to a segment of their market. At the same time, they have wanted to be able to retain the ability to develop and market the same applications commercially. Dual and multiple licensing have emerged as a strategy to enable them to do both. This could be an option for some applications.

In **dual or multiple licensing**, the licensor offers open source licenses (which may be either permissive or restrictive, depending on the circumstances) to the market segment interested in that option) and proprietary licenses to the remaining market. MySQL is a prominent example of this model.[120] The theory behind dual /multiple licensing is that it is a "win-win" outcome benefitting the open source community, the commercial licensor, and the commercial licensee. The open availability of source code allows the software to be improved by those who wish to contribute changes. The proceeds from commercial licensing help fund additional development and help establish the product as a commercial standard.[121]

[120] Meeker, pp. 143-144.

[121] This is something of a simplification; a company contemplating this model would also take into consideration the impact of the open source offering on its patent portfolio, use of trademarks, etc., but those details are outside the scope of this preliminary analysis.

Through this licensing strategy, the developer, who retains the intellectual property on the application, could, for example develop a basic, no frills application for open source distribution, and an enhanced version for commercial sale.

The DoD report cautions that being able to charge others for additional rights may be beneficial to the company, but it can also enable a form of lock-in depending on what the company does with the additional rights and what customers need. This approach can also weaken collaboration; some people cannot or will not contribute improvements under these asymmetrical arrangements. Whether or not this option is beneficial, and to whom, and therefore a desirable option for the DMA Program, will depend on the circumstances.[122]

Procurement Approaches and the Timing of Inbound License Decisions and Negotiations

The DMA Program is considering two procurement approaches to attracting development expertise for each application: Contracting via conventional procurement, and challenges. (See Chapter 5 for the details of these alternatives.)

Contracting

The license agreement between the developer(s) and the DMA Program will be finalized as part of the contract, prior to the start of development.

The DMA Program will need to state in the Request for Proposal that the desired application is to be offered under specified open source terms and license(s), and state explicitly the inbound open source licenses and license terms that will be acceptable to the Program. The RFP should also stipulate the license terms that are acceptable in any secondary license the developer may anticipate having to receive from the IP owner of a proprietary component.

Challenges

From the standpoint of intellectual property rights and their management, the use of challenges introduces two distinguishing considerations for the DMA Program. The first is an alternative legal tool for conveying IP permissions: the contributor agreement. The second is the effect that using the challenge has on when the intellectual property agreement can be finalized, and the effect that may have on the outcome of the challenge as it relates to the achievement of DMA Program goals.

A.4 Contributor Agreements

A contributor agreement (a.k.a. contribution agreement) is an agreement by which an ***individual contributor*** to an open source project grants **sufficient rights for the parties operating the project** (in this case, the DMA Program) **to release the contribution as part of the project** (in this case, open source release on terms compatible with the outbound license). In essence, a contributor agreement is an inbound license, and its details should be given the same level of careful consideration.

[122] DoD report, p. 64.

The Apache Contributor Agreement v2.0[123] is generally regarded as a standard by the industry.[124]

There are three approaches to such agreements:

- To require the contributor to assign all rights to the DMA Program. Assignments typically grant back to the assignor (the contributor) a broad right to use the code outside of the project. This is somewhat akin in effect to the notion of dual licensing, in that the contributor can take the application proprietary.
- To require the contributor to grant a broad license to the DMA Program.
- To use no agreement at all. This is not recommended.

The terms of contribution agreements may include representations and warranties for the protection of the recipient (for example, a warranty that the contributor wrote the code, or is not employed by a company that will claim rights to it). Having such terms significantly reduces the DMA Program's exposure to charges of copyright infringement.

Another reason to have such agreements is in anticipation of the possibility that with time and experience, the DMA Program might wish to change the outbound open source license. If the DMA Program has failed to obtain contribution agreements, it will not be able to change the outbound license unless it receives permission from every contributor. This is a time-consuming, costly, and possibly infeasible scenario, and best avoided.

Timing of the agreement

1. The challenge is a prize competition (monetary or non-monetary award) in which participants are invited to take an application's concept of operations and requirements (or bundle operational concept, if ENABLE ATIS) and develop an application. They may or may not use the ADE. A winning application will be selected: judges will select the application on the basis of stated criteria, and it must pass acceptance testing. The winning application will be posted to the repository and offered under an open source license.
2. The challenge invites all interested participants to join in the open source development of an application, the ConOps/requirements or operational concept for which are posted to the OS Portal's Application Development Environment. The completed application, if it passes acceptance testing by the DMA Program, will be posted to the repository and offered under an open source license.

In both cases, the challenge announcement should state that the accepted application will be released by the DMA Program under specified open source terms and license(s). Participants must agree to the contributor agreement before they may upload their contributions. The pivotal concern for the DMA program should be the matter of upstream inbound licenses:[125]

[123] Not to be confused with the Apache 2.0 license.
[124] Meeker, *passim,* p. 147.
[125] This issue is discussed in greater detail in Chapter 5.

- In the first scenario, the developer or contributor is essentially free to negotiate secondary licenses – and free to make errors in doing so – before submitting the contribution to be judged. An otherwise excellent application may therefore have to be rejected because upstream licenses are more restrictive than the terms of the intended outbound license. The next-best entry that does have suitable secondary licenses may be acceptable, but the marginal difference between the two may be something of a loss relative to what the DMA Program hoped to see developed. The DMA Program has the option of requiring all participants to consult with US DOT before making secondary license agreements, but it will need to weigh the cost of these consultations and how they will be paid for.
- In the second scenario, all activity for the single project is taking place on the ADA. The DMA Program will need to decide as a governance matter *in advance of the challenge announcement* what role it chooses to take in relation to the project, and assign roles and responsibilities accordingly. Having the opportunity to monitor activity, or requiring reporting from the project leadership, would mean that the Program would be aware of the possibility of needing secondary licenses for proprietary components, and the ability to guide the choice of terms to assure compatibility with the outbound license.

Treatment of Post-Release Contributions of Enhanced Code

The DMA Program's need to manage intellectual property does not necessarily end at the point when users download the released open source application, so neither does the requirement that the Portal track contributions and permissions.

The OSADP Concept of Operations anticipates that the repository will be used both "to share code and artifacts and receive contributions from the community."[126] We anticipate that this will be particularly the case for public-sector applications, which should attract significant registered user communities. In that event, community members may be interested in sharing improvements to the application with one another in the form of enhanced source code.

In that event, the Portal must be able not only to record the fact of the contribution, but also to provide the user with a contributor agreement, prepared in advance by the DMA Program, that the user must accept electronically in order to upload the enhanced code. (The upload process should also be contingent upon the user's supplying metadata on the contribution.) The record of the upload, the identity of the contributor, the fact of acceptance, and the metadata are retained by the Portal.

[126] *DMA OSADP Concept of Operations*, Final Draft Document, Version 3.3.3 – August 5, 2011, p. 13.

A.5 Open Source Licensing of Non-Software Deliverables

The DMA Program will be procuring not only the source code for the Mobility applications but associated documentation. These items are also covered by copyright, and so require a license from the developer permitting open source distribution, but are licensed separately from the source code.

Creative Commons offers a range of licenses for this purpose and provides an on-line selection tool (the License Chooser).[127] The following provides two licenses options for the DMA program to consider:[128]

1. **Attribution (CC BY) license.** This license lets others distribute, reorganize, modify, and build upon the work, even commercially, requiring only that the author for the original creation receive credit. This is the most accommodating of licenses offered. Creative Commons recommends it for maximum dissemination and use of licensed materials.
2. **Attribution Share Alike (CC BY-SA) license.** This license lets others remix, modify, and build upon the work even for commercial purposes, as long as they credit you and license their new creations under the identical terms. This license is often compared to "copyleft" free and open source software licenses. All new works based on yours will carry the same license, so any derivatives will also allow commercial use. This is the license used by Wikipedia, and is recommended for materials that would benefit from incorporating content from Wikipedia and similarly licensed projects.

Given that the terms of these licenses parallel those of permissive and restrictive open source software licenses, our tentative recommendation is that Attribution (CC BY) license should be used for non-software deliverables when the application's outbound license is to be either permissive or weakly restrictive; the Attribution Share Alike (CC BY-SA) license should only be used if the outbound license is strongly restrictive.

US DOT counsel, however, will make the final choice of these licenses.

A.6 Institutional OSADP Requirements

The OSADP functionality required to support the DMA Program's goals and objectives is the subject of this section. We discuss functionality first with regard to inbound licenses, and then outbound licenses.

Inbound Licenses: The Roles of the DMA Program and the OS Portal

For each Mobility application, the role of the DMA Program, in consultation with US DOT counsel and Acquisitions, is the following.

[127] http://creativecommons.org/choose/
[128] http://creativecommons.org/licenses/

For a procured project, to:

- Assure that the terms of the outbound license support the intended use of the application, including potential commercialization and/or merging with the source code of other applications (especially if the other applications are also Mobility applications).
- Assure that the request for proposal fully represents the DMA Program's intent to offer the completed application as open source; lists the license(s) and terms under which it will be offered; and indicates the open source terms and license(s) that it will accept from the developer. If permissible under law, the RFP should also require that the developer agree to consultation with US DOT regarding the terms of any secondary licenses it must negotiate, and to submit the draft secondary license for US DOT's approval before it is finalized.
- Assure that the terms of the inbound license received from the contracted developer in no way conflict with or override the terms of the outbound license. By implication, the terms of any secondary licenses attached to the inbound license are equally non-conflicting and compatible with the outbound license. (See Chapter 4 for extended consideration of this point.)

For a challenge, to:

- Decide whether full assignment of rights or a broad license is the appropriate arrangement with contributors.
- Use the appropriate legal document.

For all projects:

- To assure that the OS Portal User Agreement accurate presents licensing and other terms applying to code and content posted on the Portal

In contrast, and in relation to all projects, **the role of the OS Portal is to assure that each contribution is being made by a registered developer or contributor and is covered by a license or contributor agreement.**

As a passive step toward accomplishment of this role, the OS Portal will display the User Agreement (perhaps providing a link on the Portal's home page). Procedurally, the OS Portal must be able to:

- Recognize that an attempt is being made to upload a contribution,
- Identify the developer or contributor attempting to perform the upload,
- Verify that the developer or contributor is operating under a license or contributor agreement, and then either
    - Permit the upload and link it to the developer; or
    - Block the upload if the chain of linkages is incomplete, and display a message giving the reason for the blockage.

Functionally, verification can be made through automated querying of a relational database. In order to populate the database, the Portal needs to obtain data through two different channels.

Developer Data

Capturing identifying information on the developer or contributor for this purpose occurs through user registration:

- For procured projects, the OS Portal needs a process for receiving the list of approved team members and their project roles (corresponding to the role definitions on the Portal) and using

the list for authentication at the time that the team member first registers. Who provides the list to the OS Portal is a role the must be defined (i.e., the DMA Program or direct transmission from the contracted developer). A process for receiving changes to the list (personnel removed from the project and those added to it) is also needed, including standards for when the revised information must be provided relative to the triggering event.[129]

- For challenges, participants must register and provide required information (to be determined as a governance decision).

License/Contributor Agreement Data

Capturing information on the relation between the developer/contributor and the license or contributor agreement depends on which of the latter is involved.

- For procured projects, a procedure is needed for conveying license information from the DMA Program to the Portal, because the legal arrangement takes place outside of Portal transactions.
- If the project is through a challenge, at least three scenarios are possible:
  - Development takes place outside the Portal, and the contestant developers wish to upload the completed application.
  - Multiple development teams compete using the ADE.
  - Participants collaborate on a single open source project.

  In all three instances, the contributor agreement, and the developer or participant's agreement, can be handled virtually by the Portal. It is recommended that the registration process require the participant to agree in advance to accept the terms of the agreement before uploading contributions; providing a link to the contributor agreement at that time is an option.

Outbound Licenses: The Roles of the OS Portal

The role of the DMA Program in relation to outbound licenses is described above, because that activity is necessary before the Program can fulfill its role in relation to inbound licenses. The role of the OS Portal is to recognize and record when a released application is uploaded, and by whom, in acceptance of the open source license for that application.
Procedurally, the OS Portal must be able to:

- Provide a link to the application's source code and documentation,
- Identify the user attempting to download the application, and
- Recognize the user's acceptance of the open source license.

The second and third of these requirements will require further policy consideration.

[129] For example, "contractor must inform OS Portal within one business day when a team member is removed."

Mechanisms for License Acceptance

In US legal practice, any contract between two or more persons rests on two assumptions:

1. There is a mutual obligation created by the agreement: the *consideration*
2. There is mutual consent on the terms of the agreement: the *offer* and the *acceptance.*

Once an offer that involves the exchange of consideration has been made and accepted, an enforceable contract is created. Because a license is a contract, the OS Portal needs the capability to recognize this transaction. Electronic acceptance of proprietary licenses is commonplace; however, the conventions of the open source community make how the Portal should structure acceptance functionally a judgment call for the DMA Program.

Conventionally, when consumers purchase a physical piece of proprietary software, they acquire not just that physical copy (plus manual, etc.), but the right to use the software for its intended purpose. By opening the plastic wrap on the box, the so-called "shrink-wrap license"[130] binds the consumers 1) not to copy the work, 2) not to make derivative works based on the work, and 3) not to authorize anyone else to do either of these two things.

When the proprietary product and license both exist in virtual space, there are two different ways in which the offer and acceptance can take place; and small differences can be critical in determining whether a contract is formed. The "browse wrap" license involves giving the user a link to view the license terms, but doing so is not necessary in order to link to the site from which the software can be downloaded. Acceptance of the license terms is implied rather than explicit, and the enforceability of the contract is subject to dispute. Alternatively, the user must accept a "click wrap" license by taking some action, such as clicking a button that says, "I accept the terms"; this is more likely to create an enforceable contract.

In complete philosophical contrast, open source licenses do not impose affirmative obligations on licenses, but rather impose restrictions on the rights granted under the license. As a result, the continued availability of the work they want to use is contingent on their adherence to the license's terms.

The GPL usually is attached to code simply by virtue of the programmer's placing the appropriate notices in the code. The licensee does not click to accept or indicate assent by signing any document. Item 7 of the Open Source definition does not allow open source licenses to include the requirement for downstream licensees to sign a contract.[131]

We recommend that the DMA Program consult with US DOT counsel to ask whether enforceability of the open source license is important or a moot point, inasmuch as the application is being offered free of charge.

[130] Such "shrink-wrap licenses" are provided with virtually every copy of commercial software sold today. Although such licenses do no present the formalities that people usually associate with contracts, they are generally enforced as binding contracts. *Specht v. Netscape Comm. Corp.,* 00 Civ. 4871 (AKS), 2001 WL 755396 (S.D.N.Y. July 5, 2001). P. 5 ff. Cited in St. Laurent, p. 149.

[131] Meeker, p.224.

If the legal opinion is that enforceability of the outbound open source license is important, then we recommend that the OS Portal add requirements that support the click wrap option.

Regardless of the outcome on this issue, however, the OS Portal needs to have the capability to attach the outbound license to the code of the released application.

APPENDIX B: Conventional Software Licensing Terms Under U.S. Law[132]

Under a software license agreement, a licensor grants a licensee certain rights in a software product while the licensor not only retains full ownership of the software, but also can restrict what the licensee does with the software in many critical respects. These restrictions are typically designed to (1) protect the licensor's potential market for the software; and (2) protect the licensor's intellectual property rights in the software.

There are many types of software license agreements and the negotiations of license agreements can result in numerous complex issues. There are, however, a few provisions found in most license agreements that are particularly important to licensees and licensors. This section addresses five such provisions: (A) license grant, (B) use restrictions, (C) warranties, (D) indemnification provisions, and (E) provisions limiting the liability of the parties.

A. License Grant. The core of any software license agreement is the grant clause. The license grant must clearly describe the software, technology and/or intellectual property rights licensed to the licensee, as well as all of the uses of such software, technology and/or intellectual property rights that are allowed under the agreement.

A typical grant clause may contain the following wording:

> *Subject to the terms and conditions of this Agreement, Licensor hereby grants to Licensee a non-exclusive, non-transferable, worldwide, perpetual and irrevocable license to use the object code of the Software solely for Licensee's internal business purposes at the Installation Address set forth in Exhibit ____.*

Each of the terms in the above software license grant has a specific meaning that fundamentally impacts the rights of the licensor:

1. *Definition of the "Licensee."* The definition of the "Licensee" referred to in the grant clause above is important not only for legal reasons but also for financial reasons. For example, the licensor may want to restrict the definition of Licensee to protect its return on the Software. The broader the definition of Licensee, the more entities or individuals will have access to and use of the Software under the Agreement, thus reducing the amount of license fees a licensor may potentially receive.

2. *Definition of "Non-exclusive."* The term "non-exclusive" is necessary in a grant clause to indicate that the licensor reserves the right to license the same software to other licensees.

3. *Assignability/Transferability.* The grant clause above specifies that the license is non-transferable. This language is typical as licensors want to prevent licensees from transferring their rights to a third party so that the third party will be required to obtain a license from, and pay the applicable license fees to, licensor.

[132] http://www.utahbar.org/sites/midyear/html/introduction_to_software_licen.html

4. *Irrevocable License.* The grant clause above specifies that the license is irrevocable. Licensees often want the term "irrevocable" included in the license grant to ensure that after they accept the software and pay for the license, the licensor has no basis to revoke the license. The term "irrevocable" implies permanency, however, and causes concern for licensors. As discussed above, if a licensor must agree to the use of the term "irrevocable" it should ensure that the license grant is prefaced with the phrase *"Subject to the terms and conditions of this Agreement, . . ."* This wording conditions any permanency on the terms of the license, including the termination provisions, thus mitigating the licensor's concerns.

5. *Definition of "Software."* The definition of the term "Software" is also critical to both licensors and licensees. Given the importance of this definition, it is often defined in an exhibit to the license agreement to allow for an adequate description. Licensees will often want the definition of Software to include any updates or upgrades to the Software while licensors will want to make any upgrades or updates a part of their maintenance and support services to increase the revenue generated by such services.

6. *Permitted Uses.* The license agreement should clearly describe the uses of the Software permitted under the license. Many agreements simply provide that a Licensee is allowed to use the Software, while others also permit the Licensee to reproduce, copy, distribute and/or create derivative works based on the Software. Licensees may also be permitted to bundle the Software with one or more Licensee products and to market and distribute the combined product. Licensees need to be sure that all of its intended uses of the Software are clearly permitted under the terms of the license grant.

7. *Term of License.* Both the beginning and the end of the license term are important considerations for the licensee and licensor. Licensees expecting perpetual licenses should carefully review the license agreement to confirm that licensor does not have the right to unilaterally terminate the agreement.

8. Use Restrictions. A licensee's use of the licensed software will not only be governed by the terms of the license agreement between the licensor and the licensee, but also by the applicable provisions and restrictions of intellectual property law. ***Under the copyrig***ht laws, an "owner of a copy" of a computer program may:

- Make backup copies (one may be used);
- Create a new copy as part of the using process;
- Adapt the software (including by reverse engineering) to produce a new copy, for use by the owner (although only one may be used);
- Sell or give away the program, provided that the original and all backup or adaptive copies are transferred with the owned copy;
- Run the program (or authorized copy) on any machine on which it will run at any location;
- Use the program to provide services to others (e.g., time-sharing would probably be okay so long as there is only one active copy).

Because most licensors want more protection than that allowed by the intellectual property laws, licensors will include additional restrictions on a licensee's use of the software in a license agreement. Some of those restrictions are included in the grant clause while others appear throughout the license agreement.

1. Restrictions in the License Grant. In the model grant clause above, the restrictions on use are indicated with the terms "non-exclusive," "non-transferable" and "internal business purposes only."

2. Use Restrictions. Most license agreements will include a separate provision or provisions specifically describing certain additional restrictions on licensee's use of the software. One such provision is as follows:

> *Licensee agrees not to (a) modify, adapt, alter, translate, or create derivative works from the Software; (b) merge the Software with other software; (c) sublicense, lease, rent, loan, or otherwise transfer the Software to any third party; (d) reverse engineer, decompile, disassemble, or otherwise attempt to derive the Source Code for the Software; or (e) otherwise use or copy the Software except as expressly allowed under Section _____ (License Grant).*

In addition, if the software contains any confidential or proprietary information, the licensor should be sure to include certain provisions in the license agreement to protect that confidential or proprietary information.

3. Geographic Restrictions. In addition to limitations on how the software may be used, geographic restrictions on where the software may be used should also be considered when drafting software license agreements.

B. Warranties

1. Typical Express Warranties. An express warranty is one that is articulated in the license. For software licenses, the licensor may make a number of standard warranties. A licensor may warrant that the licensor has valid title to the software being licensed, that the licensor has the right to grant the license, including the license to any third-party software contained in the software, that the software will operate in accordance with the functional specifications and/or documentation, and that, except as noted in the specifications or documentation, there are no "trap doors," "time bombs," or other disabling devices.

2. Implied Warranties. Unlike express warranties, which are only created by an affirmative act of the licensor, implied warranties are created by operation of law and are automatically a part of every software license agreement unless specifically disclaimed.

C. Intellectual Property Indemnification. Indemnification refers to the extent to which the licensee will provide indemnification for infringement or violation of third party intellectual property rights. In many cases, the licensor, as the developer, will step up to responsibility for infringements. In some cases, however, the licensor will take the position that by opening up certain markets, the licensee is increasing the licensor's risk and should be responsible for indemnification.

The licensor will often require an exclusion from intellectual property indemnification for modifications or combinations for which it is not responsible. The issue, of which party should be responsible for combinations of technology, particularly when it is known that the licensed technology will be used in combination with other technology, is particularly difficult. The licensor may ask the licensee to indemnify it for such modifications and combinations.

When negotiating an indemnification provision, each party should consider the following issues:

- If the software is found to infringe the intellectual property rights of a third party, what rights and remedies does the licensee have? What options does the licensor have?

- Who pays whose legal expenses, or does each party pay its own legal expenses?
- Can a mere threat or allegation trigger the indemnification clause or does it take an actual court action?

Limitations of Liability

1. Limitations in General. Limitations of liability should never be considered "boilerplate;" their terms are critical business issues for both sides. In most cases, the parties will agree to disclaim incidental, consequential, special and punitive damages, subject to certain exclusions. Note that a liability limitation provision ordinarily should not attempt to completely exclude all direct damages.

2. Caps. In many cases, the licensor will argue that there should be a cap on its liability equal to the amount of revenue received under the contract. Alternatively, each party may request a fixed dollar cap.

Conventional Software Licensing: A Contrast to OSS Licensing

United States intellectual property law views computer software and documentation as creative works, and automatically assigns the ownership of the intellectual property to the software's creator in the form of *copyright*. Reproduction, distribution, modification, public demonstration and public display of software that is "substantially similar" to the original software are illegal without the creator's permission. A license is the formal grant of rights by the creator to engage in conduct that otherwise would be a violation of the licensor's intellectual property rights.[133]

A conventional license agreement generally includes terms protecting both the licensor and licensee in the following areas: license grant, use restrictions, warranties, indemnification provisions, and provisions limiting the liability of the parties. See Appendix A for more details.

Patenting of software is legally permissible in the US at this time, although highly controversial.[134] It is an increasing practice among universities.[135] Patent law reserves the intellectual rights to the creator, who may license them out for stipulated uses in exchange for royalty payments.

As an alternative to exercising copyright or seeking a patent award, the intellectual property owner may choose to relinquish all rights and place the work in the *public domain,* in which case the users of the work are free to do with it as they please.

Trademark provides exclusivity over the use of a name (for example, a brand). Many successful open source projects claim trademark over the name of an open source software project.

[133] http://www.utahbar.org/sites/midyear/html/introduction_to_software_licen.html

[134] G. Gross. Court *Patent Ruling Leaves Software Patents Intact*. PC World Business Center, June 28, 2010. At http://www.pcworld.com/businesscenter/article/199994/court_patent_ruling_leaves_software_patents_intact.html. Accessed 2/21/2011.

[135] AK Rai, JR Allison, BN Sampat, and C Crossman. *University Software Ownership and Litigation: A First Examination*. 87 North Carolina Law Review 1519-1570 (2009). Abstract at http://scholarship.law.duke.edu/faculty_scholarship/1629/. Accessed 6/1/2011.

International intellectual property law will be relevant if the DMA Program allows application development by international entities. International conventions[136] automatically attach copyright to every novel expression of an idea, whether it is through text, sounds, or imagery. Japan and Asia reportedly have patent laws similar to those of the US, whereas Europe has been more conservative.[137]

Under the Federal Acquisition Regulation (FAR), the usual terms of the license specified in federal procurement of development expertise reflect conventional copyright law. Although licenses granted to the Federal government are "irrevocable, royalty-free, and worldwide," the licenses may not extend to the source code and they are "non-transferable." Because of this, careful attention will need to be paid to selecting the appropriate intellectual property clauses from among the allowed alternatives. The requirements of the program should be discussed with the contracting officer and with US DOT legal counsel.

[136] The Berne Convention for European countries and the World Trade Organization Agreement on Trade-Related Aspects of Intellectual Property Rights.

[137] *Software Patent Law: United States and Europe Compared*. iBRIEF / Patents & Technology. 2003 Duke L. & Tech. Rev. 0006, 3/21/2003.

APPENDIX C: How Software is Programmed[138]

The file containing a program and instructions that the computer can read and perform is called an executable file (*.exe). Programmers do not write executable code—they write source code.

Programmers usually do not write new code entirely from scratch. Instead they use prewritten components, or "**library** routines." By using prewritten routines, programmers make their coding more efficient, and they have more assurance that the code will be free of bugs and interoperable with the platforms on which the program will run. The documentation for the routine will specify which information needs to be communicated to the routine when it is executed (for example, where the file will be written, the filename, the information to be written, and the number of bytes required).

This reuse of routines is performed formally and systematically. Programmers write their program in a text processor (or development environment). This code—**the source code**—looks like cryptic English and is the set of instructions telling the computer's processor what to do; any skilled programmer can read this code.

Since the computer cannot execute this code as written the programmers run a large, complex program called a compiler. The compiler translates the source code, including the references to the needed library routines, into **object code**; a set of binary instructions that the computer's processor can execute. A related program called a linker then links the object code to the referenced library routines, producing a program that can be executed by the computer: this is the **executable file** (which usually contains many object code files).

It is important to understand that programmers do not necessarily need access to the source code for the library routines; they only need to know which information to send to the routines and which information the routines will send back. Essentially, the routines can be "black boxes."

If a bug is found in the program, it is not the executable file or object code that must be edited, but the source code. Once corrected, the program must be recompiled. **This is why source code is so crucial: without access to source code, the user cannot correct errors and must rely on the vendor to do so.**

[138] HJ Meeker, The Open Source Alternative: Understanding Risks and Leveraging Opportunities. Hoboken, NJ: John Wiley & Sons, 2008. Pp. 7-9.

APPENDIX D: Common Restrictive and Permissive Open Source Licenses and Their Interactions

Open source software is often combined and recombined with other open source software to produce new and useful combinations. Combining software requires that developers and users obey *all* of the licenses simultaneously. Only some OSS licenses can be combined with other types of licenses while meeting the requirements of all the licenses. Figure D-1[139] summarizes how some common OSS licenses can be combined.

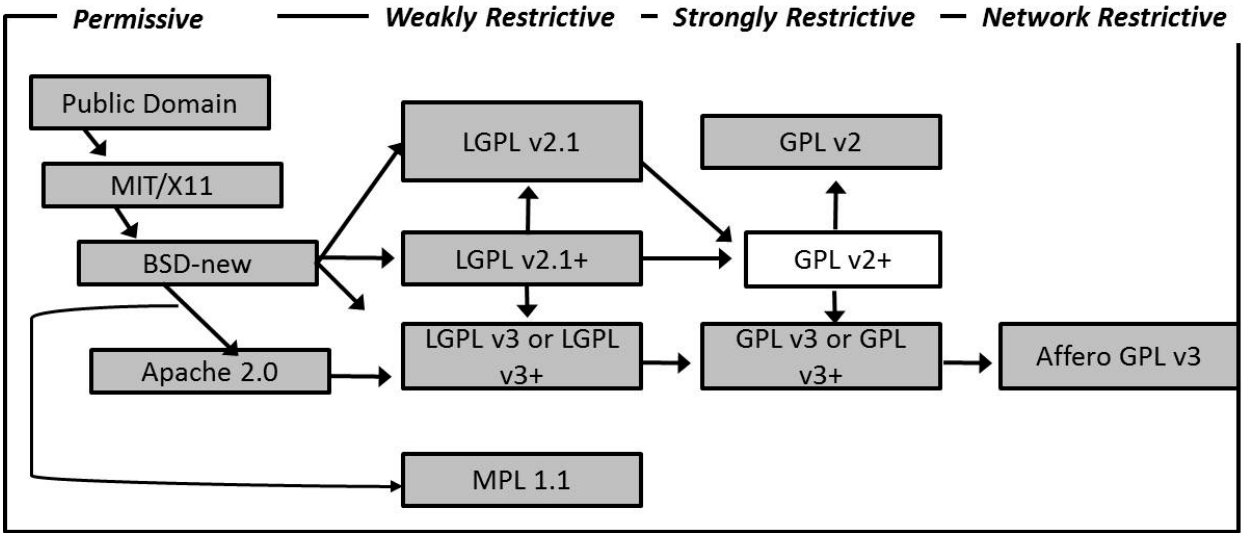


Figure D-1. Interaction among Common OSS Licenses

In Figure D-1, the shaded boxes are the names of different FOSS licenses (the "+" means "or any later version"). The figure shows the OSS licenses, organized into three groups:

1. At the left are the **Permissive** licenses, which permit the software to become proprietary. At the top left is "Public Domain" (meaning in this case "no copyright protection"), which strictly speaking is not a license but works like one: anything can be done with public domain software. The software must be explicitly released to the public domain or be created by a US Government employee.

   Next is the MIT/ X11 license, which is very permissive. Software under the MIT license is easily combined with the modern 3-clause Berkeley Software Distribution (BSD-new)

[139] Presented from *The Free-Libre / Open Source Software (FLOSS) License Slide*, by David A. Wheeler, 2007, located at: http://www.dwheeler.com/essays/floss-license-slide.html.

license, which, compared to the MIT license, adds a clause forbidding the use of the author's name to endorse or promote products without permission. Lastly, there is the Apache version 2.0 license.

2. To the right are the **Strongly Restrictive** (strong copyleft) licenses, which prevent the software from becoming proprietary. This category includes the most popular FLOSS license, the GNU General Public License (GPL). The GPL has a version 2 (GPLv2) and a version 3 (GPLv3); a "+" afterwards means "version X or later".
3. In the middle are the **Weakly Restrictive** (weak copyleft) licenses, a compromise between permissive and strongly restrictive licenses. These prevent the software component (often a software library) from becoming proprietary, yet permit it to be part of a larger proprietary program. This figure shows the rules when making other software part of the weakly protected component; there are other possibilities if the licensed component is only being used as a library.

   The GNU Lesser General Public License (LGPL) is the most popular weakly restrictive license, and has a version 2.1 (LGPLv2.1) and 3 (LGPLv3). Another such license is the Mozilla Public License 1.1 (MPL 1.1), but the MPL has the potentially serious drawback of being incompatible with the widely popular (and strongly restrictive) GPL; an MPL module cannot be used in a larger GPL'ed program."

In this figure, an arrow from box A to box B means that software with these licenses can be combined; *the combined result effectively has the license of B, possibly with additions from A.* In other words**, the more restrictive terms of the two licenses trump those of the more permissive license.**

The figure indicates compatibility with the arrows that connect the licenses. To see if software can be combined, start at their respective licenses, and find a common box by following the arrows (a/k/a "following the slide"). For example, Apache 2.0-licensed software and GPLv2+-licensed software can both reach "GPLv3 or GPLv3+", so they can be combined using GPLv3 or GPLv3+. This figure has been carefully crafted so following a path determines if two licenses are compatible. For further detail, the text of each license should be reviewed.

APPENDIX E: The Impact of License Type on the Success of Attracting Developers to Open Source Projects—A Literature Review

Summary

This appendix summarizes an initial review of empirical research on the relationship between software license type and project success in open source software development projects. The review's findings can be summarized as follows:

- On balance, restrictive (e.g., copylefted) licenses appear to have a negative effect on user interest and long-term developer involvement in open source projects.
- Non-restrictive licenses appear to stimulate user interest and developer involvement in open source projects.
- Projects with non-restrictive licenses are more likely to reach an advanced stage of development than projects with restrictive licenses.
- Project sponsorship, and, in particular, sponsorship by a non-commercial organization, stimulates user interest in OPEN SOURCE projects.
- Non-market project sponsorship may counteract potential user concerns about the likelihood of open source software developed without a restrictive license remaining free and open.

Background

The central element of the open source approach is the open source software license. Open source licenses are an alternative to the more common proprietary, commercial licenses. Open source licenses are based on several underlying principles, as described succinctly by Lawrence Rosen.[140] These include the freedom of licensees to use open source software and/or source code for any purpose; the freedom to copy and distribute the software without payment to the licensor; the freedom to create and distribute derivative works using the original software/source code without payment to the licensor; and the freedom to combine the software with other programs.

Open source software licenses exist along a spectrum of restrictiveness. On one end are highly-restrictive "copyleft" licenses, notably the GNU General Public License (GPL), that mandate not only free distribution of the software and source code, but total reciprocity of licensing: all derivate works must be licensed under the same GPL license as the original. Across the open source license spectrum are licenses that grant various amounts of latitude in terms of commercial and non-commercial distribution, use and modification of the source code, and sale of derivative software. The Berkeley Software Distribution (BSD) license is an

[140] Rosen, Lawrence, *Open Source Licensing*, Prentice Hall, 2005.

example of a type of permissive free software licenses that impose minimal requirements on use and redistribution.

Given the wide range of open source license choices available to a project initiator, a key question is: how does the restrictiveness of the license affect the project's success? Do certain types of licenses produce better results than others? This whitepaper is an initial attempt to address these questions, by examining available empirical research on the topic.

Defining Success in an open source Project

In light of the USDOT's reasons for choosing the open source model for the DCM/DMA initiative, the success of any given open source project can be defined in terms of several factors:

- *Developer interest and participation*, as evidenced by the number of developers involved, the level of software development activity, the number of bugs reported (bug reports indicate active developer scrutiny of the code being contributed), patches contributed, and other metrics.
- *Non-developer user interest* – as evidence by the number of downloads of the application (once it reaches at least the alpha stage) by potential end users who are not themselves developers.
- *Project Development* – as evidenced by the development stage the project reaches, controlling for the project's age. Software development projects typically advance through the following stages: planning, pre-alpha, alpha, beta, production-stable, and mature.

EMPIRICAL RESEARCH

This appendix does not attempt to review the full theoretical literature on open source project success. Instead, it focuses on the much smaller sub-set of articles describing empirical studies that explicitly measure the effect of license choice on various measures of project success. There appear to be relatively few peer-reviewed articles on the topic.[141] Most likely this is because a sufficiently large and diverse population of open source projects has existed only for about the last decade.

Importantly, beyond the following high-level summary, this review does not discuss the theories underlying the empirical work and the various hypotheses that the researchers tested. Instead, this review focuses on what the researchers found. Likewise, this review does not discuss the various methodologies employed (e.g., various multiple regression techniques). Instead, it interprets the findings, and reports the results of the research descriptively.

[141] For this report, a professional technical reference librarian at a major research university conducted a literature search for peer-reviewed papers.

Theoretical Framework

There are two competing paradigms regarding the impact of license type on open source project success. The first paradigm suggests that restrictive (e.g., copylefted) open source licenses should increase project success, for several reasons. First, restrictive open source licenses are in keeping with the original "hacker" culture and spirit of open source programming: that knowledge should be freely accessible to all. From this perspective, projects with restrictive licenses will be more likely to attract developers than projects with less-restrictive licenses.

In addition, restrictive licenses help prevent developers' pro-bono contributions to an open source effort from being commercialized by third parties. This removes a major concern that would deter developers who would otherwise want to be involved, for both "collective" reasons (i.e., promoting freedom of knowledge, and personal reasons (e.g., peer recognition, professional reputation, and early access to emerging software applications).

On the other hand, there are several reasons why, in theory, restrictive licenses could deter developers and/or users from being involved, and could hinder open source projects from making progress. First, restrictive licenses may affect users' perceptions of usefulness of the software, particularly among those who wish to advance a commercial interest.

Second, restrictive licenses can limit the ability of users to use the software in conjunction with other applications that are distributed with less restrictive licenses. This could be a significant deterrent for end users, who may not want to be constrained in their choices of applications.

Third, both end users and developers may find the perceived risks and burdens related to legal aspects of highly restrictive open source licenses off-putting.

Finally, the additional flexibility afforded by less restrictive licenses (e.g., the ability to use the applications concurrently with a wider variety of other software) may yield projects that are more complex, challenging, and intellectually rewarding than those developed under restrictive licenses, thereby attracting more developer interest.

STUDIES

Relatively few empirical studies have looked explicitly at the relationship between open source license type and project success. This paper describes the findings of the following five studies, which appear to represent the current state of research. As mentioned previously, this review is not comprehensive: it is an initial inquiry.

- C. Subramaniam, R. Sen and M. Nelson, "Determinants of Open Source Software Success: A Longitudinal Study", *Decision Support Systems* 46 (2009) 576-585.
- S. Comino, F. Manenti and M. Parisi, "From Planning to Mature: On the Success of Open Source Projects", *Research Policy* 36 (2007) 1575 – 1586.
- J. Lerner, J. Tirole, "The Scope of Open Source Licensing", *Journal of Law, Economics, and Organization* 21 (1) (2005) 20 – 56.
- K. Stewart, A. Ammeter, L. Maruping, "Impacts of License Choice and Organizational Sponsorship on User Interests and Development Activity in Open Source Software Projects", *Information Systems Research* 17 (2) (2006) 126 – 144.
- J. Colazo, Y. Fang, "Impact of License Choice on Open Source Software Development Activity", *Journal of the American Society for Information Science and Technology* 60 (5) (2009) 997 – 1011.

Researchers have benefitted from the existence of two online repositories of open source code and projects: SourceForge.net and Freecode.com (formerly Freshmeat). These are websites for software developers to manage and contribute to open source software projects. Both sites maintain databases of projects, including numerous qualitative and quantitative data for each project. From these and other sources, researchers have access to data on several hundred thousand open source projects (although at any given point in time many of the projects may be dormant).

Findings

Table E-1 on the following page summarizes the findings of the five studies. In the table, the arrows indicate the directionality of the association between license restrictiveness and open source project success metrics. An upward arrow indicates a positive association; a downward arrow indicates an inverse association.

Colazo and Fang found empirical support for the theory that open source projects with restrictive licenses attract developer interest and contributions. The researchers examined developer membership, coding activity, and development speed for 244 open source projects that met their specific research criteria. Regression modeling revealed that all three of these indicator variables were significantly and positively associated with restrictive (copyleft) licenses. On the other hand, the same study found that copylefted projects were associated with lower developer permanence on projects. The researches posit the explanation copylefted projects attract highly skilled developers, who gain visibility by contributing to the projects, and are therefore recruited away or move on to other projects as their careers advance.

Subramaniam, et al also found empirical support for an association between restrictive open source licenses and project success, but in a more limited context. Using SourceForge data on 8,627 open source projects, the researchers estimated regression models indicating that projects with strong copyleft licenses increased user interest (measured by the number of downloads), but only for applications targeted at non-developer users and system administrators. For open source projects targeted at developers, strong copyleft licenses were found to have a negative impact on developer interest (measured by the maximum number of developers working on a project in a given month). Across all project types, strong copyleft licenses were found to have a negative impact on both user interest and project activity.

The remaining studies (Lerner and Tirole, Comino et al., and Stewart et al) all found that restrictive open source licenses tended to dampen project activity and success (or, conversely, that non-restrictive licenses attract greater user interest/activity).

Comino et al. are notable because their study focused on the impact of license type (among other factors) and the probability that an open source project would evolve from a preliminary stage to a mature software product. Using a sample of 88,192 projects from SourceForge, the researchers developed a regression model in which the dependent variable was the development stage of each project (see the description in the bullet points on page 2 of this white paper).[142] They found that open source projects distributed under highly-restrictive licenses were less likely to reach an advanced stage of maturity than were projects with less

[142] The chronological age of the project was included as one of the model's independent variables, to control for project age.

Table E-1: Findings from Literature Review

Study	Association between license restrictiveness (or project sponsorship) and open source project success	Description
J. Lerner, J. Tirole, "The Scope of Open Source Licensing"		Open source projects with less restrictive licenses tend to attract *more* **contributors** and **software development activity**.
J. Colazo, Y. Fang, "Impact of License Choice on Open Source Software Development Activity"		**Developer membership** is *higher* in copylefted open source projects than in non-restrictive projects.
		**Developer coding activity** is *higher* in copylefted open source projects than in non-restrictive projects.
		**Project speed** is *faster* for copylefted open source projects than for non-restrictive projects.
		Copylefted open source projects are associated with *lower* **developer permanence**.
S. Comino, F. Manenti and M. Parisi, "From Planning to Mature: On the Success of Open Source Projects"		Open source projects distributed under highly restrictive licenses are *less likely* to reach **advanced stages of development.**
C. Subramaniam, R. Sen and M. Nelson, "Determinants of Open Source Software Success: A Longitudinal Study"		Overall, restrictive open source licenses *negatively impact* **user interest and project activity.**
		Restrictive open source licenses *increase* **user interest** for projects aimed at **non-developer users or and system administrators.**
		Restrictive open source licenses *decrease* **user interest** for projects aimed at **developers.**
K. Stewart, A. Ammeter, L. Maruping, "Impacts of License Choice and Organizational Sponsorship on User Interests and Development Activity in Open Source Software Projects"		Open source projects that use a non-restrictive license attract *greater* **user interest** over time than those using a restrictive license.
		Projects with sponsors attract *greater* **user interest** over time than projects without sponsors.
		Projects with non-market [non-commercial] sponsors attract *greater* **user interest** over time than projects with market sponsors.
		Projects with a nonmarket sponsor and a nonrestrictive license attract *greater* **user interest** over time than any other combination of license and sponsorship.

restrictive licenses. A possible explanation for this finding, in keeping with the theoretical framework, is that projects with restrictive licenses attract idealistic programmers who are motivated by the desire to be associated with such projects. They are more concerned with fostering this kind of free and open development activity, than with seeing specific applications come to fruition. For these programmers, the incentive to continue contributing shrinks as the project progresses. Indeed, for some programmers the incentive to participate drops off as soon as they are included in the list of contributors.

Other important relationships were noted by Stewart, et al. This study looked at the effects of project sponsorship (i.e. formal association of an open source project with a recognized organization) as well as license type on the success of open source projects. The researchers found four notable associations:

1) Projects that use non-restrictive licenses attract greater user interest over time than projects that use restrictive licenses.
2) Projects with sponsors attract greater user interest over time than projects without sponsors.
3) Projects with non-market [non-commercial] sponsors attract greater user interest over time than projects with market sponsors.
4) Projects with non-market sponsorship and non-restrictive licenses attract greater user interest over time than do projects with any other combination of license type and sponsorship.

Summary and Implications

On balance, the findings of the studies reviewed suggest that highly restrictive licenses present greater potential risks than benefits to the potential success of open source projects. The one study (Colazo and Fang) that found a consistent positive association between restrictive licenses and developer interest, developer activity, and project speed, also found that restrictive licenses negatively affect developer permanence on projects. This suggest that although restrictive licenses may stimulate greater initial activity, they may ultimately be detrimental to project success.

The only other indicator of a positive relationship between restrictive licenses and user interest (Subramaniam, et al.) is in a narrow context: restrictive licenses appear to increase user interest for projects aimed at non-developer users and system administrators.

There are appears to be strong empirical support for the idea that non-restrictive or less restrictive open source licenses engender greater developer participation and user interest than do projects with restrictive licenses. At the same time, however, the use of non-restrictive licenses raises the specter of open source software not remaining free and open; this possibility could in theory dissuade some developers from contributing to open source projects that lack strong copyleft licenses.

The findings on the effects of sponsorship on user interest suggest a way out of this dilemma, because they indicate that project sponsorship, and specifically sponsorship by a non-market organization like the USDOT, can help counteract potential user concerns about the likelihood of a software product that lacks a restrictive open source license remaining free and open. As the researchers of this particular study note:

> *One interpretation of this pattern of results may be that sponsorship trumps licensing in terms of its impact on users' perceptions regarding the likelihood of the software remaining free of commercial control (Stewart, et al.)*

APPENDIX F: Additional Considerations for Program-Level Policy Decisions

Application Testing Before Release to the Repository

The form of procurement of an application will have some bearing upon options for if/when an application passes bench-top validation but fails in field testing. At this point, criteria will be needed as a basis for selecting among the following options:

- If a the application was developed as a procured development project, whether to a) change the requirements based on lessons learned from the field test failure and send the developer back to the ADE; b) procure a new developer; or c) drop the application.
- If a challenge, whether to a) issue a new challenge with changed requirements, (b) change the structure of the challenge to give the DMA Program greater involvement in /oversight over contestant activity; or c) give up on that application.

These decisions will have a direct bearing on the language and conditions in both RFPs and challenge announcements.

Designating a Development Effort as a Project on the OSADP

Program-level governance sets the initial conditions for the OSADP to begin accepting users and projects in the context of the overall success factors of the program. When considering whether to accept the project as a development effort for the OSADP, the following considerations are useful:

- Whether the project is procured through contracts or the result of a competition or challenge.
- Whether or not the development of the application requires communication and collaboration with the developers of other projects underway because of an intended synergy among the applications.
- The availability of some developing and testing capabilities, project management resources, and channels for communication could prove attractive to non-affiliated individuals and small firms.
- Whether the risk of inbound license conflict with the intended outbound license is relatively high (See Ch. 4, Intellectual Property, for discussion on this point).

APPENDIX G: Roles and Responsibilities

Table G-1: Crosswalk with OSADP Concept of Operations , Table 4 and Figure 5

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
A. Governance and Oversight – Program Level	• Mobility Program Managers who oversee the portal and applications research projects • ITS Legal Policy team • US DOT Privacy Officer • US DOT CIO	**Program Oversight Team :** • Develops a charter for itself and for the Portal Oversight team. • Charter establishes overall policy – may begin by confirming/validating the policies in this report; further identifies new policies needed. • Defines Portal Oversight team responsibilities and boundaries for decision making. • Develops a process for decision making and conflict resolution between Portal oversight team and project-level teams. • Develops an overall timeline for efforts. • Develops and approves a baseline for security and privacy; reviews risks and establishes the level of tolerance for risks. • Develops guidelines for acceptance of new projects; develops user agreements.	• *What are the measures by which the Portal is considered successful?* • *How will conflicts be resolved? Who needs to be involved in conflict resolution? DOT legal counsel? Multiple modes?* • *What decisions are made at the Program level versus the Portal level?* • *What risks can be tolerated?* • *What are the criteria for accepting new projects?* • *What are the criteria for releasing products into the release repository? Does testing need to occur before release into the release repository? What type of testing and to what level?* • *Who will perform licensing?*	• **Governance Manager**	• Outside of the OSADP but in relation to the General Portal, Registered User Environment, Application Development Environment, and the Computing Infrastructure

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
		• Develops criteria for registering and admitting users. • Recommends procurement and development strategies. • Develops criteria for product release into the release repository. • Establishes/ensures that there is an appropriate licensing process in place and that liability is addressed. • Oversees and approves Standard Operating Procedures and other portal-based policies that are specific to development efforts (such as policies on data storage, etc.). • Develops an outreach strategy for receiving input and feedback from stakeholders and users on a regular basis.			
B. Governance and Oversight – Portal (System) Level	• Contracted system managers of the OSADP • Initial review and then periodic reviews by the FHWA CIO	**Portal Oversight Team:** • Based on charter established by the Program-level team, develops Applications Development Environment Policies, Rules of Operation (or, Standard Operating Procedures), and Rules of Conduct for the Portal. (see row D below) • Oversees and monitors operations of the Portal.	• *What are day-to-day operational needs of the Portal? Are all of them being met?* • *What rules of operation, rules of conduct, and applications environment policies will be recommended? What risks exist and what trade-offs have been*	• **Portal Manager** with input from **System Administrators**, **Portal Moderators**, and **Infrastructure Providers**	• Outside of the OSADP but in relation to the General Portal, Registered User Environment, Application Development Environment, and the Computing Infrastructure

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
		• Monitors security and other risks; implements risk mitigation technologies and policies; develops and implements response plans. • Develops and implements the process for registering and admitting users admits new users. • Implements user agreements. • Develops a checklist of information needed from project managers before accepting a new project into the OSADP. • Analyzes the new projects for risks. • Reviews and recommends new projects to the Program-level team. • Develops and monitors the needs associated with community building within the Portal (email functions, chat rooms, etc.). • Implements the outreach strategy developed by the Program-level team (and proposes changes based on the effectiveness of the results). • Connects with users to ensure ease of use and user satisfaction. • Communicates to users the rules of engagement, site policies, and compliance policies. • Develops and provides a site	*made?* • *What are the most effective actions for security and privacy? How will they be implemented? Monitored? Reported? How frequently?* • *In allowing for new projects to be posted to the portal, have the licenses been properly identified and completed?* • *What are appropriate criteria for release of products into the repository? What type of testing should be done before release? Is the testing enough to mitigate liability? Is the testing requirement a burden to developers? Are there/should there be exceptions?*		

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
		taxonomy, • Manages content. • Reviews, documents, and recommends product prioritization and release into the repository to the Program-Level team. • Manages changes, down-time; performs back-ups and patches. • Identifies and recommends maintenance and upgrades to the Program-level team; and takes actions, once approved.			
C. Governance and Oversight – Project Level	CORs are the multi-modal staff that comprise the DMA Program Management team. Project managers/ product owners are identified by: • US DOT procurement – the contract identifies the owner(s) and manager(s) of the effort • US DOT	• **<u>Contractor Officer Representatives (CORs):</u>** o Oversee development, milestones, deliverables, *if product development was procured by the US DOT.* If no US DOT procurement, no reason to assign a COR. o Ensure that other connected vehicle research efforts receive relevant results. • **<u>Project Managers/Product Owner(s):</u>** o In proposing new projects to the Portal-level team and DMA CORs, project owner(s) proposes how project development, decision-making, and collaboration will occur. o Product owner(s) defines the potential risks of the new	• *What are the development timelines? Milestones? Deliverables?* • *Who will assure quality?* • *Are their relationships or impacts to other parts of the DMA research? Connected vehicle research?* • *What are the project needs and requirements? Tools? Data?* • *Who will be the community associated with the development process? What gaps in expertise need to be*	• **Project Managers** (and potentially, **registered users**)	• Policies apply to the Application Development Environment

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
	challenge/ competition – the teams identify the owner(s)/manager(s) • Owner(s)/Manager(s) who self-identify when proposing a new project to the Program-level oversight team	projects. o Product owner(s) defines restrictions and/or licensing requirements associated with "inbound" source code and other products being brought into the portal.	*filled?*		
D. Data Environment Policies / System Rules of Operation	• **Same for Program Oversight Team and Portal Oversight Team**	(see row B above)	(see row B above)	• Includes **Unregistered** and **Registered Users** (which are likely to include project sponsors, portal managers, governance managers, portal moderator, system administrator and infrastructure provider), **Developers, Committers,**	• Policies apply to all four levels of the architecture e—General Portal, Registered User Environment, Application Development Environment, and Computing Infrastructure. Policies are developed outside of the OSADP but should include feedback from all users.

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
				Testers, Reviewers	
E. Intellectual Property Policies + Liability / Risk Mitigation Strategies + Privacy / Data Usage Policies + Data Ownership Policies	• ITS Legal Policy team	**US DOT Legal Counsel:** • Develops a comprehensive licensing strategy for the OSADP by reviewing license recommendations and approving choices. Licensing strategy will need to include the identification of decision points (how and when licensing will occur), conflict resolution, and proposal of exceptions. • Develops warranties and terms of use statements to be provided on the OSADP website. • Communicates policies to Program- and Portal-level teams and developers; communicates decisions to the procurement teams.	• *Are the recommended licenses appropriate for the OSADP? In line with US DOT policies?* • *Are warranties and terms of use statements appropriate mitigations against liability? Is more needed?* • *Will stakeholders and developers agree to these terms?* • *How will the US DOT procurement staff and CORs implement the license, warranty, and user terms as part of contracts? Are there unique issues that will need unique actions to be taken?*	• US DOT legal counsel participates in the **Governance Manager** position and also support the **project sponsors**	• Outside of the OSADP but in relation to the General Portal, Registered User Environment, Application Development Environment, and the Computing Infrastructure. To identify risks and develop appropriate policies, the portal architecture, design, and technology choices will require review by US DOT legal counsel.
F.Procurement	DMA Program Managers	**Investment Decision Team:** • Reviews applications ConOps and makes decisions regarding whether US DOT will make investments. • Develops a procurement and development strategy for each application that will receive investment, choosing among the options based on risks and opportunities.	• *Which projects are most suitable for investment by the US DOT and for the OSADP?*		

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
		• Develops the SOWs and evaluation metrics.			
	US DOT Procurement staff	**Procurement Team:** • Implements the procurement strategy and develops the procurement package (RFP, contract). • Facilitates the proposal review and ensures appropriate evaluation.	• *What type of procurement and development path is best suited for the project? What is the most suitable procurement method (contract? Broad Agency Announcement? Competition or Challenge? Cooperative Agreement? Other?)*		
G. Commercialization and Adoption	Project Managers/Product Owner(s) who proposed the project, performed development, and performed testing	**Product Owner:** • During development of the product, identifies users, facilitates user input and feedback, and identifies requirements for and develops guidance and documentation with the DMA COR. • Develops a plan for post-release with product owner that targets outreach to users, identifies market challenges and proposes solutions with the DMA COR. • Helps develop the vendor support community.	• *What are the needs of the product after release?* • *Who are the consumers/users for the product?* • *What type of vendor support is needed post release?*		
	DMA Program Managers/CORs involved in the oversight and delivery of the	**DMA COR:** • During development of the product, identifies users, facilitates user input and feedback, and identifies	• *Who will take on product support after release? What are the consequences of not having a vendor support*		

Policy	Recommended Policy Actors	Role & Responsibility Description	Decisions	Relates to ConOps User Class Profiles (Table 4)	Where are Decisions Made? (Figure 5)
	products to the release repository	requirements for and develops guidance and documentation with the product owner. • Develops a plan for post-release with product owner that targets outreach to users (particularly public sector users), identifies market challenges and proposes solutions with the product owner. • Identifies how the release of the new product (application, technology, or enhancement) impacts the ITS Architecture or standards; works with the PCB team on technology and knowledge transfer; and works with the multi-modal staff to mainstream into use. • Helps develop the vendor support community.	*community available?* • *What are market challenges to adoption (i.e., local laws that might prohibit open source applications use)? What are market risks (liability, data ownership, distribution issues, etc.)? Can they be resolved through US DOT facilitation or other Federal solutions (i.e., US DOT hosting a competition for vendors to participate?)?* • *What guidance, tools, or other reference materials are needed to support implementation and use?*		
H. Application of Other Federal Policies	• US DOT CIO • US DOT Privacy Officer • ITS Strategic Planning Group • ITS Program and Modal Administrators • DMA Program Managers	• Ensures that US DOT and other Federal policies are properly applied to the OSADP	• *Are the NIST guidelines for security and privacy properly applied?* • *What are event scenarios and what are response plans in case of an event?* • *How does the OSADP support US DOT goals? What gaps in knowledge or technologies will it fill?*	• Governance Manager	• Outside of the OSADP but in relation to all four tiers of the architecture

Table G-2: Relationship of these Roles and Responsibilities to the Concept of Operations, Section 5.5.1, Table 4

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
Unregistered User Category			
• ***Unregistered User***	Unregistered users are defined as visitors from the general public who may or may not have an interest in the OSADP. They are not registered with the portal and therefore cannot log in. An Unregistered User can view publically accessible information such as generation content about DMA, as well as other content and documents made available to the general public.	▪ Browsing OSADP public web pages ▪ Viewing and downloading public content that does not require registration ▪ Completing and submitting online registration form that will be evaluated. Completion of user registration form is a step for qualified Registered User to be considered for additional access as a Registered User.	▪ Unregistered users are bound by the policy for the OSADP. Once the policy options have been reviewed and decided upon, unregistered users will need to follow the OSADP policies or be removed from use. At this level, these policies mostly include security, privacy, and rules of conduct.
Registered User Category			
• ***Registered User***	Registered Users are users who register with and provide information to the OSADP. In addition to the privileges and access rights of Unregistered Users, Registered Users may have access to additional information and content. Specifically, they have access to resources that require registration.	▪ All privileges of Unregistered Users ▪ Bounded by user agreement terms in the registration process ▪ Having access to discussion forums, news blog, and announcements ▪ Ability to participate in online discussions ▪ Subscribing to news updates ▪ Reporting bug/error, limitation, and problems with portal content and portal software (e.g., broken links) ▪ Having access to released source code repository to view, download, test, and make changes to application open source ▪ Reviewing and updating their personal profile ▪ Submitting or proposing new and innovative ideas ▪ Reviewing and commenting	▪ Registered users are bound by the policy for the OSADP. Once the policy options have been reviewed and decided upon, registered users will need to follow the OSADP policies or be removed from use. At this level, these policies mostly include: o Providing a minimum of information through registration o Security and privacy o Rules of conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use o Participation and collaboration with projects according to rules of governance set at the project-level. A registered user may be a project manager/product owner, as described in the previous

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
		on other approved projects ▪ Discussing related project ▪ Submitting source code or data to the community to use ▪ Viewing other Registered Users' public profile	table.
Contributor Category			
• ***Project Sponsor***	Project Sponsor is a person designated by USDOT to provide oversight for funded projects. The sponsor is involved in the process of funding and giving high-level guidance to the project as it relates to the DMA program. Not expected to be involved intimately with the project at a detailed level, the Project Sponsor interfaces with the Project Manager for project related status and updates.	▪ Representing USDOT as the main contact for the project ▪ Approving funding and resources ▪ Providing guidance to Project Manager relating to the DMA program overall direction ▪ Interfacing with Project Manager for status and updates ▪ Providing final approving for staff addition and reduction proposed by Project Manager ▪ Providing advisory role in open or meritocratic management projects	▪ Project sponsors are DMA CORs, as described in the table above. They will participate in the development of policies for the OSADP and will be part of the project-level oversight. One or more of these CORs may also participate in the program-level oversight and governance. They will work with the Procurement Office staff
• ***Project Manager***	A special project member who has project leadership responsibilities including directing application development effort, working with Project Sponsor, and making decisions relating to the well-being of project including staffing and resource issues.	▪ All privileges of Registered Users ▪ Ability to vote on project decisions ▪ Access to all in-development source code repository ▪ Working with Project Sponsors to secure resource and support ▪ Providing project leadership and direct application development effort ▪ Responsible for project management including scope and schedule management ▪ Leading system engineering process ▪ Evaluating and deciding on readiness of application ▪ Collaborating with other Project Managers as necessary	▪ Project Managers are those who propose, are contracted with, or are assigned management responsibilities to participate in project planning and oversee project delivery. Project managers may also be product owners. Their participation includes setting project-level governance policies and metrics. They are required to adhere to system-level OSADP policies including: o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use o Registration

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
		▪ Access to all project source code and resources	
• ***Developer***	A Developer is a Contributor who is directly involved in developing the project applications. Developers can play multiple roles.	▪ All privileges of Registered Users ▪ Access to all in-development source code repository ▪ Participating directly in the application development effort in many different project roles, including designing system components, creating source codes, developing software, troubleshooting and fixing bugs, writing documentation, etc. ▪ Participating in online discussions ▪ Performing peer review of codes, provide suggestions, and constructive criticism ▪ Active Developer may be promoted to a Committer who has specific privileges in version control of codes ▪ Attending project meetings and discussions and collaborating with other project team members regularly	▪ A developer may also be a project manager/product owner, may be a part of the development team, or may be an external collaborator. Developers are required to adhere to system-level OSADP policies and project-level governance policies including: o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use o Registration
• ***Committer***	A Committer is an active project member who has all privileges that a Developer has with several additional access rights for configuration management, code build, and managing the Released Open Source Repository.	▪ All privileges of Registered Users and of Developer ▪ Committing code changes in configuration branches to the main trunk in code repository ▪ Initiating code build and compilation ▪ Preparing source code for release ▪ Ability to vote on certain project decisions ▪ Collaborating with other project team members regularly	▪ A committer is part of the project team and may also be a project manager/ product owner. Committers are required to adhere to system-level OSADP policies and project-level governance policies including: o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use o Registration

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
• ***Tester***	A Tester verifies functionality and features of an application or system per design document and test plan. Testing may occur at various phases of the development process.	▪ All privileges of Registered Users ▪ Access to application or target system ▪ Documenting bugs and issues and tracking them to resolution ▪ Building and compiling source code ▪ Collaborating with project team as required	▪ A tester may also be a project manager/product owner, may be a part of the development team, may be a DMA COR, or may be an external person or entity. To the extent that testers require access to the OSADP, they are required to adhere to system-level OSADP policies and project-level governance policies including: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use
• ***Reviewer***	A Reviewer reviews and provides technical opinions and critical comments on engineering products, including designs, codes and documentation, etc., as needed.	▪ All privileges of Registered Users ▪ Reviewing system engineering and other documents ▪ Reviewing source code designs, source code, and test results ▪ Collaborating with project team as required	▪ A reviewer is most likely a person who is external to the project team. It may be a DMA COR, someone contracted by the DMA COR, or a peer within the OSADP. Reviewers are required to adhere to system-level OSADP policies and project-level governance policies including: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use
Administrator Category			
• ***Portal Manager***	Portal Manager is responsible for the look-and-feel and content of the portal and the Registered User Environment, including portal news blogs, announcement bulletins, and overseeing the discussion forums	▪ All privileges of Registered Users ▪ Responsible for user experience of General Portal and the Registered User Environment including usability, navigation and search, as well as the overall look-and-feel of these environments ▪ Producing and editing blog articles ▪ Managing moderators of	▪ Portal managers (similar to the portal oversight team described in the previous table) are contracted system managers. Portal managers must adhere to OSADP policies for: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
		discussion forums and bulletins including removal of unwanted information or messages ▪ Adding, updating, and deleting data files ▪ Working with Governance Manager in adding, updating, and deleting terms of use, governance, license, policies and legal related content ▪ Portal Manager manages all content on the portal, but consults with Governance Manager and Project Managers for their respective content areas ▪ Working with Project Manager who is responsible for project specific content in the Application Development Environment	abiding by terms of license use In addition to adhering to these policies, portal managers participate in implementation and enforcement of OSADP policies, identify and mitigate risks as they arise and report them to the Program Oversight Team (or governance managers), and recommend changes in policy to the program-level oversight/ governance team.
• ***Governance Manager***	Governance Manager oversees the practice of governance policies and ensures that they are implemented properly and is also responsible for preparation and revision of license agreement, disclaimer, and other legal statements to be posted on the portal.	▪ All privileges of Registered Users ▪ Leads the practice on all governance policies, regulations, compliance and disclaimer statements, etc. ▪ Providing oversight and management of risks ▪ Performing auditing of license agreement terms ▪ Enforcing proper insertion of open source license statement in source code and monitoring open source content for compliance and compatibility ▪ Having read-only access to both Released Open Source Repository and in-development source code repository for inspection purposes	▪ Governance managers are similar to the Program Oversight Team described in the previous table. Governance managers must adhere to OSADP policies for: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use In addition to adhering to these policies, governance managers set these policies and provide oversight to the portal managers. Governance managers receive recommendations from the portal managers and modify or develop new policies as necessary.
• ***Portal Moderator***	Portal Moderator monitors discussion forums, instant chat, social networking, and other collaborating tools	▪ All privileges of Registered Users, with limited read/write access within the community communication tools	▪ Portal moderators are considered part of the portal management team (or, portal oversight team, as described in the previous table). They must adhere to OSADP

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
	in the Registered User community and has ability to remove or delete content, if deemed inappropriate based on governance and portal policies. Portal Manager may promote and demote Registered Users from the community to become Portal Moderators. Portal Moderator may be assigned to use specific communication tools or an area within the community communication forums.	▪ Reporting inappropriate activities to Portal Manager ▪ Monitoring violations of governance and policies	policies for: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use Portal moderators are active in supporting the daily communications and exchanges between and among project-level teams as well as between project teams and system administrators and portal managers. As a result, they are likely to make recommendations to the portal managers regarding changes needed in policies.
• ***System Administrator*** *If SaaS or PaaS is used, some of these services may be provided.	System Administrator is in charge of installing, supporting, and maintaining servers and other computer systems, and planning for and responding to service outages and other problems. Other duties may include scripting or light programming, project management for systems-related projects, supervising or training computer operators, and being the consultant for computer problems beyond the knowledge of technical support staff.	▪ All privileges of Registered Users ▪ Having system root access and be able to allocate system resources as needed ▪ Responsibility for system access security ▪ Adding, removing, and updating user account information, resetting passwords, etc. ▪ Assigning access rights to project content based on Project Manager's direction ▪ Ensuring network infrastructure is up and running ▪ Troubleshooting any reported technical problems ▪ Analyzing system logs and identifying potential issues ▪ Installing and maintaining software applications and tools for the Application Development Environment ▪ Auditing performance of systems and software applications	▪ System administrators are considered part of the portal management team (or, portal oversight team, as described in the previous table). They are contracted support staff. System administrators must adhere to OSADP policies for: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use System administrators are active in supporting the daily operations of the portal and make recommendations to the portal managers regarding changes needed in policies.

User Class Profiles	Role Description	Permissions and Capabilities	Relationship to the Policy Report Recommendations and Proposed Actors
		▪ Planning system capacities and disaster recovery ▪ Performing data backups and restoring system from backup after a problem or disaster occurs ▪ Applying operating system updates and patches ▪ Monitoring the sharing of data, meta-data, or other information ▪ Removing unwanted information or messages ▪ Testing and checking new data sets ▪ Adding new data sets ▪ Adding, updating, and deleting history/context information within the portal environment. ▪ Answering technical queries ▪ Responsibility for documenting the configuration of the system	
Infrastructure Provider Category			
• ***Infrastructure Provider*** *If IaaS or PaaS is used, this function may be included by the service.	Infrastructure Provider delivers computer infrastructure environment that supports advanced data acquisition, data storage, data management, data integration, data mining, data visualization, and other computing and information processing services distributed over the Internet for enabling OSADP virtual collaboration.	▪ Access to the computing resources, including processing capabilities, network resource, data security and data storage system, etc., for provisioning infrastructure services, but no access to the Application Development Environment ▪ Working with System Administrator to provide requested infrastructure resources and services for OSADP	▪ Infrastructure providers are contracted support staff and must adhere to OSADP policies for: o Registration o Security and privacy o Rules of Operation and Rules of Conduct o Policies regarding data use, data ownership, recognition of intellectual property and abiding by terms of license use

Bibliography

Atwood, Jeff. '*Coding Horror: Pick a License, Any License"*. Web article, April 3, 2007. Located at: http://www.codinghorror.com/blog/2007/04/pick-a-license-any-license.html.

Behl, Pardeep. ***Winning Strategies for Portal Governance***. Located at: http://www.ibm.com/developerworks/websphere/library/techarticles/0904_behl/0904_behl.html.

The Berne Convention for European Countries

Boehm, B. and R. Turner (2004). ***Balancing Agility and Discipline: A Guide for the Perplexed***. Boston, MA: Addison-Wesley. ISBN 0-321-18612-5.

British Cabinet Office. **All About Open Source: An Introduction to Open Source Software for Government IT, Version 1** (October 2011)

Brown, Mark. "*Icelandic government makes a push for open-source software*," Wired.co.uk (23 March 12), http://www.wired.co.uk/news/archive/2012-03/23/iceland-open-source-software

Burnett, Ed. "*Google says no to license proliferation"*. Interview with ZDNet, November 2, 2006.

Cabinet Office. ***Open Source Procurement Toolkit***. http://www.cabinetoffice.gov.uk/resource-library/open-source-procurement-toolkit

Colazo, J. and Fang, Y. *Impact of license choice on Open Source Software development activity*. Journal of the American Society for Information Science, 60: 997–1011. doi: 10.1002/asi.21039, 2009.

Coleman, Sean. ***Open-Source As An Alternative To Commercial Software***. Final Report 583 produced for the Arizona Department of Transportation, March 2009.

Comino, S., F. Manenti, and M. Parisi, "*From Planning to Mature: On the Success of Open Source Projects*". Working Paper, No. 35, 2007.

Cuddy, Matt, Alan Chachich, Aviva Brecher, Michael Razo, and Suzanne Sloan. ***Policy Analysis and Recommendations for Development of the Dynamic Mobility Applications***, Report for the US DOT, June 2012. Publication Number: FHWA-JPO-12-033

Driver, Mark. *"A CIO's Perspective on Open-Source Software"*. White Paper for Gartner, Inc. January 2011.

Fan, Brian, et al. "Open Source Intellectual Property and Licensing Compliance" A Survey and Analysis of Industry Best Practices," Olliance Group, LLC (2004).

Feldman, Robin and Kris Nelson. *"Open Source, Open Access, and Open Transfer: Market Approaches to Research Bottlenecks"*. White Paper located at: http://ssrn.com/abstract=1127571

Gonzalez de Alaiza Cardona, Dr. Jose J. ***15 Texas Intellectual Property Law Journal 157***. State Bar of Texas, Intellectual Property Law Section, 2007.

Government Policy of Iceland. ***Policy on Free and Open-source Software***. Prime Minister's Office (December 2007)

Gross, G. "*Court Patent Ruling Leaves Software Patents Intact,"* PC World Business Center, June 28, 2010. At http://www.pcworld.com/businesscenter/article/199994/court_patent_ruling_leaves_software_patents_intact.html

Guntersdorfer, Michael. *"Software Patent Law: United States and Europe Compared"*. iBRIEF / Patents & Technology. 2003 Duke L. & Tech. Rev. 0006, 3/21/2003.

Halbert, Debora. *"The Open Source Alternative: Shrink-Wrap, Open Source, and Copyright"*. Journal article for the Murdoch University Electronic Journal of Law, Volume 10 Number 4 (December 2003).

Halloran, T.J., William L. Scherlis, and Justin R. Erenkrantz. ***Beyond Code: Content Management and the Open Source Development Portal (Position Paper)***. Proceedings of the 3rd Workshop on Open Source Software (2003). Abstract located at: http://www.mendeley.com/research/beyond-code-content-management-open-source-development-portal/. Full paper located at: http://www.erenkrantz.com/Geeks/Research/Publications/ContentManagement.pdf.

Halchin, L. Elaine. ***Other Transaction (OT) Authority***, Congressional Research Service, 7/2011. Publication Number: RL34760. Located at: http://government-policy.blogspot.com/2011/07/other-transaction-ot-authority.html

Hassol, Josh, Aviva Brecher, Matt Cuddy, and Suzanne Sloan. ***State-of-the-Practice and Lessons Learned on Implementing Open Data and Open Source Policies,*** Report for the US DOT, June 2012. Publication Number: FHWA-JPO-12-030

Hinck, Robert, Philip Kimmey, Joshua Roberts, Dima Qassim, and Denise Zheng. **National Open Source Policies, Center for Strategic and International Studies.** Data Compiled in March 2010. Located at: http://csis.org/files/publication/100416_Open_Source_Policies.pdf

Le, Steven and Al Hovde, ***Task 3.3: Concept of Operations – Dynamic Mobility Applications Open Source Application Development Portal***, Final Draft Document for US DOT, Version 3.3.3 – August 5, 2011

Le, Steven and Jim Cassady, ***TASK 4.0: SyRS – Dynamic Mobility Applications Open Source Application Development Portal***, Draft Document for US DOT, Version 3.0 – October 2011

Lerner J. and J. Tirole. ***Some simple economics of open source.*** J. Indust. Economics. 50(2):197–234, 2002.

Lerner J. and J. Tirole. ***The scope of open source licensing.*** J. Law, Economics., Organ. 21:20–56, 2005.

Lindberg, Van. ***Intellectual Property and Open Source***. O'Reilly Media, Inc., 2008.

Madsen, Mark. **Lowering the Cost of Business Intelligence With Open Source: A Comparison of Open Source and Traditional Vendor Costs**, 2010 Technology White Paper for Third Nature.

Marshall, Kenneth R. and Phillip J. Tarnoff. NCHRP Report 560: Guide to Contracting ITS Projects. Transportation Research Board, Project 3-77, 2008.

Mascord, Matthew. "*How to Build an Open Source Community".* Open Source Software Advisory Service by OSS Watch. March 2011.

McGurrin, Mike and Meenakshy Vasudevan. ***The Role of Free and open Source Software (FOSS) and Open Data in the ITS Data Capture and Management and Dynamic Mobility Applications***. White Paper produced for US DOT, Version 1.0, June 2011.

McHugh, Bibiana. ***Open Data and Open Source Implementation Initiatives at the Local Level; Tri-Met's Experience with Open Source Software Implementations, Open Source Software Development, and Open Data***. Presentation at the US DOT Mobility Workshop, December 2010.

McKinsey & Company. *"And the winner is…: Capturing the promise of philanthropic prizes"*. http://mckinseyonsociety.com/downloads/reports/Social-Innovation/And_the_winner_is.pdf.

Meeker, Heather J. ***The Open Source Alternative: Understanding Risks and Leveraging Opportunities***. Hoboken, NJ: John Wiley & Sons, 2008.

Okunieff, Paul and Nancy Neuerburg. ***Transit and Open Source: Is It An Option?.*** White Paper produced for the Transportation Research Board, IDEA Program. Version 2.1, June 2005.

The Perfect Storm, White Paper sponsored by the National Association of State Chief Information Officers (NASCIO) by Red Hat. Located at: http://www.nascio.org/committees/clc/best_practices/gov-perfect-storm.pdf

Perens, Bruce. ***Open Standards, Principles, and Practice***. Blog, located at: http://Perens.com/OpenStandards/Definition.html.

Peters, Stormy. Best Practices for Creating an Open Source Policy. Blog, February 25, 2009. Located at: http://olex.openlogic.com/wazi/2009/create-open-source-policy/

Rai, A.K., J.R. Allison, B.N. Sampat, and C. Crossman. ***University Software Ownership and Litigation: A First Examination***. 87 North Carolina Law Review 1519-1570 (2009). Abstract at http://scholarship.law.duke.edu/faculty_scholarship/1629/

Rosen, Lawrence, ***Open Source Licensing***, Prentice Hall, 2005

Roth, Craig. ***Website Governance: A How-to Guide.*** Located at: http://www.craiglroth.com/Opinions%20In%20Depth%20-%20web%20governance.pdf

SAIC, ***Task 3.1: Open Source Development Web Resources Scan Assessment Report***. Produced for the US DOT, February 2011.

Scotchmer, Suzanne. "*Openness, Open Source, and the Veil of Ignorance*". White Paper, part of Papers and Proceedings of the American Economic Association, 2010.

Scott, John, David A. Wheeler, Mark Lucas, and J.C. Herz, ***Open Technology Development: Lessons Learned and Best Practices for Military Software, Report for the Department of Defense***, May 2011. Located at: http://mil-oss.org/resources/otd-lessons-learned-military-v1.pdf

Sloan, Suzanne, Ingrid Bartinique, Josh Hassol, Deirdre Herring, Amy Sheridan, and Dicky Waldron. ***Identification of Critical Policy Issues for the Mobility Program***, White Paper for the US DOT, June 2012. Publication Number: FHWA-JPO-12-035

Sloan, Suzanne, Aviva Brecher, and Josh Hassol. ***Policy Analysis and Recommendations for the DCM Research Data Exchange***, Report for the US DOT, June 2012. Publication Number: FHWA-JPO-12-036

Sloan, Suzanne, Alan Chachich, Matt Cuddy, and Michael Razo. ***Privacy and Security Analysis and Recommendations for the DCM and DMA Programs***, Report for the US DOT, June 2012. Publication Number: FHWA-JPO-12-032

St. Laurent, Andrew M. Understanding Open Source and Free Software Licensing. O'Reilly Media, Inc., 2004.

Sen, R., C. Subramaniam, and M. Nelson. "Determinants of Open Source Software License Choice." Forthcoming in the Journal of Management Information Systems. 2008.

Stewart, K., Ammeter, T., and Maruping, L."*Impacts of License Choice and Organizational Sponsorship on User Interest and Development Activity in Open Source Software Projects*," Information Systems Research (17:2), 2006, pp. 126-144.

Vasudevan, Meenakshy and Karl Wunderlich, ***Mobility Applications Program: High-Priority Applications and Development Approach***, Summary White Paper for the US DOT, 2011. Publication Number: FHWA-JPO-11-053

Webbink, Mark H., J.D. (2011) *"Introduction to Software Protection under United States Law"*. Chapter in ***International Free and Open Source Software Law Book***. Copyright © 2011 Open Source Press GmbH. This article is licensed under a Creative Commons Attribution-NoDerivs 3.0 unported license. A copy of the license is available at: http://creative.commons.org/licenses/by-nd/3.0/. Chapter is located at: http://ifosslawbook.org/united-states-of-america

Wheeler, David A. ***The Free-Libre / Open Source Software (FLOSS) License Slide,*** 2007, located at: http://www.dwheeler.com/essays/floss-license-slide.html.

Wilson, Rowan. "*Open Source Development: An Introduction to Ownership and Licensing Issues".* Open Source Software Advisory Service by OSS Watch. October 2010.

The World Trade Organization Agreement on Trade-Related Aspects of Intellectual Property Rights

U.S. Department of Transportation
ITS Joint Program Office-HOIT
1200 New Jersey Avenue, SE
Washington, DC 20590

Toll-Free "Help Line" 866-367-7487
www.its.dot.gov

FHWA-JPO-12-031

U.S. Department of Transportation
Federal Highway Administration
Research and Innovative Technology Administration