DG DIGIT

Unit B.2 Interoperability and Digital Government

OSOR Handbook (*draft*)

Open Source Software in Public Administration

February 2024

D09.01 under Specific Contract 32 FWC 07929-00 BEACON Lot 2

This Handbook was produced by OpenForum Europe for the Interoperable Europe initiative of the Digital Europe Programme under the Specific Contract 32 FWC DI 07929-00 BEACON Lot 2 with the consortium DELOITTE CONSULTING & ADVISORY, WAVESTONE.

Authors

Axel Thévenet (axel@openforumeurope.com)

Ciarán O'Riordan (ciaran@openforumeurope.com)

Paula Grzegorzewska (paula@openforumeurope.com)

Contact OSOR

EU-OSOR@ec.europa.eu

https://joinup.ec.europa.eu/collection/open-source-observatory-osor

@OSOReu

Disclaimer

Open Source Software in Public Administration 1

1. Introduction 4

About OSOR 4

Who is this handbook for? 5

How this handbook was made: research, expert workshops, and webinars 5

2. Finding and using OSS 7

Where to find a solution? OSS Catalogues 7

What criteria to consider? 8

Evaluating the health of a project 9

Evaluating the service ecosystem 10

User acceptance 10

Procurement of an external solution 11

3. Developing software 11

Organisational capabilities / required skills 11

Should your solution be built on existing software? 12

Publishing an in-house solution and facilitating reuse 13

Procurement of maintenance or development services 13

Security considerations 14

4. Co-developing and participating in OSS communities 15

How collaborative projects differ from in-house projects 15

Participating in existing communities 15

What makes a good community important? 16

5. Specific aspects of open source funding 17

Budget implications 17

6. Legal framework 18

What regulation to consider when adopting OSS? 18

i. European Regulation 18

ii. National 19
iii. Local 19
Licensing 19
Identifying OSS by looking at the licence 19
Types of OSS licences 20
Licence compatibility 22
Picking the licence that works best for your needs when using and developing FOSS 22
7. Structuring the use of FOSS 23
Benefits of a formal structure 23
Legal entity 24
8. Getting help 25
Documentation 25
Forums and networking opportunities 25

1. Introduction

Implementers of open source Software[1] (OSS) in public administrations are no longer pioneers. There are now many implementers with a decade of experience, and even some with two. Information is being shared through more formally managed networks and at conferences. This handbook is the start of an ongoing project to make that information available to a wider audience and showcase the variety of resources available to those who would like to explore using open source for their organisation's digital needs.

About OSOR

The Open Source Observatory (OSOR) is a European Commission platform hosted on the Joinup collaborative portal, serving as the connector between European public administrations

[1] This category of software was given the name "free software" when the GNU Project was launched in 1983. In 1998, it began to also be known as "open source software." For the purposes of this handbook, these terms are synonyms. Another synonym is "free and open source software", or "FOSS". Software that is not open source or free software is called "proprietary software."

and diverse stakeholders involved or interested in open source. Started in 2008, it continuously engages and supports its dynamic community and promotes the use of OSS in the public sector while providing relevant expertise and information.

OSOR is a place where the public-sector OSS community can come together to publish news, find out about events, find relevant open source software solutions, and read about the use of OSS in public administrations across and beyond Europe.

The OSOR community is characterised by three main stakeholder groups: national and local public administrations, private sector, and citizens such as academics and open source software enthusiasts. Each group can contribute to the further adoption of open source and benefit from the OSOR initiative by reading up-to-date news and in-depth case studies that are posted on a regular basis.

Who is this handbook for?

This handbook is written for European public administrations and their public officials who want to develop or implement a strategy for using OSS. As OSOR has been providing resources supporting administrations in developing a comprehensive approach to OSS for the last 15 years, this handbook's aim is to be a guide that compiles key solutions and takeaways from the experience of the OSOR community. Many public administrations have used and continue using open source today, and the experience of those public administrations can help others to overcome their own challenges. This handbook compiles key challenges and potential solutions that cover various aspects of open source use in the public sector with the hope of bringing value both to those who are taking their first steps and to those with extensive background knowledge in open source.

How this handbook was made: research, expert workshops, and webinars

In 2023, OSOR organised a series of three workshops and three webinars. The workshops gathered experts from public administrations of several European Member States who have experience in developing and implementing OSS strategies within their constituency. The main theme of the series was "Open Source strategies: the public sector's needs," and each workshop had a different focus:

- Documenting the difficulties
- Brainstorming solutions
- Examining successful solutions

Each workshop was followed by a webinar presenting the workshop's conclusions to a broader audience for discussion and comment. The results were distilled to produce this handbook. This knowledge was also combined with OSOR's previous research, along with resources developed by the open source community.

How the workshops were conducted

In preparation for the first workshop, research was carried out which suggested that the challenges that are faced by public administrations when developing or implementing an OSS strategy could, as a starting point for discussions, be sorted into 8 categories:

- Releasing software
- Finding software – discovering what exists
- Selecting software – deciding which to use
- Funding
- Knowledge sharing
- Community-building
- Soft skills
- Reuse of solutions

Each category was thoroughly examined by different discussion groups. In the first workshop, each of these themes was discussed under the overarching question of the challenges encountered or observed by the attendees. As participants were selected based on being both a public administration official and an open source user, the discussions allowed for significant knowledge sharing among the participants.

While organising the results of the first workshop, it became evident that many of the challenges had long been recognised by the open source community. With this realisation, the focus of the second workshop was to address these challenges, often shared by several different organisations.

During the second workshop, participants proposed numerous potential solutions to the identified problems. Serving as an open platform for diverse ideas, this workshop provided the opportunity to test and exchange thoughts on their implementation. Participants also delved deeper into each subject, addressing technical and practical issues.

The third and final workshop provided participants with the opportunity to learn from a selected panel of successfully implemented projects. OSOR invited representatives from each of these projects to share key elements of their success within their administrations. The results were overwhelmingly positive as the audience welcomed this platform and exchanged contacts, gaining insights into each other's work.

This handbook summarises many hours of discussion. While it doesn't encompass the entirety of the exchanges during the workshops and webinars, it presents the results of the discussions by addressing major themes in each chapter and providing potential solutions. The drafting of this handbook, however, is not finished. Experts, including particularly those who participated in the workshops, but also the broader OSOR community, will be asked for review and comments. We look forward to further developing this resource.

2. Finding and using OSS

Where to find a solution? OSS Catalogues

When looking at using OSS in the public sector, one of the first challenges encountered is to find reliable and readily available solutions that can be tried and implemented. Software catalogues dedicated to open source software are a crucial resource for this.

A software catalogue is a place to gather information about software packages. The main purpose is to assist with finding OSS solutions, but they can help with a wide variety of tasks such as coordination and reduction of duplicate work by centralising information that others have put time into gathering and verifying. The benefits of this coordination work grow with the number of users who use and can contribute to the catalogue.

Examples of OSS catalogues include:

- French government's Interministerial Free Software Catalogue
- Adullact's Comptoir du libre
- Developers Italia Software Catalogue
- FSF's Free Software Directory
- Luxembourg's Open Data Portal (also includes software)
- OS Geo's "choose a project"
- F-Droid's package listing
- Open Technologies Organisation (ΕΕΛΛΑΚ) wiki of software used in public administration
- Avoinkoodi.fi
- Spanish Centro de Transferencia de Tecnología
- Opencode.de
- Developer Overheid (Netherlands)

OSOR also maintains a broader list of resources, including project code repositories, where public administrations may also find useful information.

The usefulness of a catalogue increases with the number of users who are sharing information. Every software package already has its own web page, but the value of a catalogue is in combining the information from a large number of users in a standard format. A small number of high-quality catalogues with detailed information is thus far more useful than many catalogues with only basic information. The merging of existing catalogues would be a complicated task, but coordination at some level is a likely next step, and new initiatives should be encouraged to work with the existing initiatives. Such a coordination mechanism has been discussed in one of the studies conducted by the European Commission.

The catalogues in the list above contain information such as:

- Description of the software
- List of who is using the software
- Comments and ratings from the users
- How to contact other users
- Dependencies
- Compatibility
- Licence information
- Maintenance status
- Community size & health
- Records of validation, security audits or compliance
- Links to other catalogues with data on this software
- Vendors offering commercial services

A lot of this is information that has to be gathered in a public procurement process. By gathering this information and making it available, a catalogue can avoid the same work being repeated.

Information about who is already using a solution may indicate that it would be suitable for others with a similar context or need, or could be used to allow potential users to discuss the software with existing users. There are even discussions for some catalogues to facilitate the coordination of financing when multiple users want to fix the same bug or add the same feature.

What criteria to consider?

When looking to use OSS in any system, general criteria should be considered as well as criteria that are specific to public administrations. Most or all the information required may be present in OSS catalogues. If further details are needed, the source code of the software can be consulted. Most projects make their source code available via online code hosts, often using the Git source

code management tool. When looking at a solution's characteristics, aspects you may decide to consider include:

Licensing: By determining the licence or licences used by the solution, one can already see how the solution may be reused. To learn more about types of licences and their obligations, have a look at chapter 6.

Documentation: Comprehensive and accessible documentation is vital to understanding how the software works, its features, configurations, and best practices for implementation and maintenance. Well-maintained documentation enhances usability and ease of adoption.

Maintenance: Regular updates, bug fixes, and support for older versions are crucial for ensuring the software's longevity and reliability. Evaluating the frequency and responsiveness of updates is essential to determine the software's maintenance quality.

Community support: Active and engaged communities around OSS projects often contribute to better support, quicker issue resolution, and a wealth of shared knowledge. Assessing the responsiveness and helpfulness of the community forums, mailing lists, or other support channels is important.

Governance: Understanding the project's governance structure and leadership helps in gauging its stability, decision-making process, and roadmap. A clear governance model can indicate long-term viability of the project.

Security: Assessing the software's security measures, regularity of security updates, and history of vulnerabilities is crucial. This includes evaluating how promptly security issues are addressed and the community's approach to security matters.

Compatibility and interoperability: Compatibility with existing systems, databases, formats, and standards within your infrastructure is important. Ensuring smooth integration and interoperability minimises disruptions and enhances efficiency.

Evaluating the health of a project

When selecting a solution, evaluating the health of a OSS project is vital because it provides insights into the software's long-term viability, reliability, and support. It helps gauge ongoing development, community engagement, maintenance, documentation quality, stability, governance, adoption rates, and community sentiment. Assessing these factors helps in making informed decisions about software adoption, reducing risks associated with potentially unstable or unsupported software.

A core benefit of open source software is the transparency of each project: everyone has access to see what changes are being made from one version to the next. Furthermore, if the source

code is hosted on an online platform, then there is also transparency about how many people are participating in a project and how active the development is. Leveraging the availability of this data, the open source community has come up with various solutions to help assess a project's sustainability.

Among those, the metrics developed by the CHAOSS community are great keys to compare different measurable characteristics of OSS projects. The models designed by the CHAOSS working groups aim to help anyone assess solutions they want to use. Similarly, other frameworks to assess the health and resilience of a project provide different metrics that can be useful to public administrations when looking to use open source.

Evaluating the service ecosystem

When adopting an existing open source solution, the availability of external service providers may also be a factor in selecting a solution. These providers can support your administration's needs and help fulfil security or quality requirements if needed. When looking at service providers, it is important to evaluate the presence and diversity of companies or individuals providing support, custom development, training, and consulting services for the OSS project. When assessing the number and scale of service providers, it is also good to see if they are actively participating in the projects they provide services for. This can indicate an expert level of knowledge of the project.

User acceptance

Public sector services and open source software share a common challenge: to convince new users to try their services and retain those users. As people are used to systems and tools that are sometimes drastically different, using open source should be done in dialogue with them.

Boosting user acceptance for open source software in public administrations involves adopting a user-friendly approach. This can be done by prioritising user needs, offering thorough training and support, and trying to provide seamless integration with existing systems. Successful implementation will benefit from communicating openly about the benefits and drawbacks of the planned changes. If there will be inconvenience, it is important to be able to explain what objectives this is part of achieving.

Practical ways of facilitating user acceptance can involve implementing small-scale pilot programmes for hands-on experience, showcasing success stories to build confidence, fostering collaboration, regularly evaluating user feedback, and continuously improving to meet evolving needs. Effective change management is key to a smooth transition, ensuring users feel supported throughout the process.

Procurement of an external solution

Procurement procedures are often cited as one of the main challenges but also a key to its adoption in the public sector. While both assumptions can be true, every context is different and there is no single model that can solve this problem.

From different legal frameworks to administrative practices and budgets, procurement can raise many complexities and some public administrations may find themselves trying to work within an existing procurement procedure which didn't take OSS into account. This will depend on the procedures specific to a given administration, so we can only highlight that looking at the procurement procedures right from the start can prevent surprises later on.

One key consideration is the development of standard contracts, a practice that has proven beneficial in navigating the dynamics between smaller entities and larger companies. Moreover, by collaborating with other entities and establishing standardised procedures and then collectively approaching suppliers, one can be in a stronger position to define the terms of the contract.

3. Developing software

Organisational capabilities / required skills

As the use of open source necessitates specific sets of skills, considering proper training and recruitment of trained specialists can be very helpful. Before migrating to open, collaborative development, relevant skills can be learnt by using those tools internally first.

Sometimes called "InnerSource", this can be a practical way to bridge the gap between existing internal development procedures and the planned new procedures for after the migration of development to the open model. InnerSource facilitates an open source culture within an organisation's software development, enabling in-house teams to get used to collaborative development and the associated tools as part of the preparation for migrating to an OSS model. This approach increases visibility, minimises redundancy, and empowers developers to contribute to each other's work. By breaking down silos, InnerSource promotes effective collaboration, aligning dispersed teams to a common workflow. Large organisations benefit by leveraging talent across departments, aligning to successful open source practices, and addressing common challenges in communication and discovery. Overall, InnerSource serves as a concise and effective strategy for training in-house teams to adopt open source principles and enhance their software development capabilities.

Another way to help with creating the needed skills in a team. For instance, the City of Munich in 2023 – to create an Open Source Sabbatical. The idea is to provide qualified software

developers, both internal employees and external professionals, with the opportunity to contribute to an open source project for a limited period.

Increasing the OSS skills of a team can also be addressed by making this a factor in recruitment procedures. If a human resources team is aware of the transparency and the community nature of OSS projects, they could even look into the possibilities of finding candidates with the right skills via the communication channels of OSS projects. Similarly, if IT staff of a public administration participate in OSS communities, they may find this helps to attract interest from members of those communities when recruiting new staff.

Should your solution be built on existing software?

For those accustomed to proprietary software, the traditional approach to software development involves articulating needs, creating a development team, and embarking on the development journey, or hiring an external contractor. However, the expense of software development often leads to a preference for purchasing existing solutions, prompting the creation of software only when no suitable alternatives exist.

While entities adhering to an OSS strategy can maintain this conventional approach, the OSS paradigm introduces an additional avenue worth considering. This alternative involves exploring existing OSS solutions employed by other entities, especially those within the same sector, such as public service providers.

If an existing OSS solution does not entirely meet specific requirements, the option exists to make the changes necessary, or to pay someone else to make the necessary changes. Consider a scenario where a software package meets all the technical requirements, but the interface isn’t in the right language. Alternatively, it may be in the right language but lacks a user manual – a requirement for certain administrations. Or perhaps the software lacks two-factor authentication, which could be a security measure that is required by an administration. In these cases, rather than the solution being excluded, OSS creates the option of having the necessary modifications done.

Public sector entities, within the context of OSS, should be aware of this opportunity and integrate it into their decision-making processes. Catalogues serve as valuable tools for identifying projects with functionalities which match one's requirements. Modifying existing projects or reusing specific components for a new project can present a cost-effective and efficient approach to software development within the OSS framework.

Publishing an in-house solution and facilitating reuse

Publishing software as OSS, in the simplest manner, could be summarised in choosing an appropriate licence and making the source code accessible online. While these are positive steps, there are benefits to be had by making it easy for others to reuse your software. By expanding the user base, the pool of potential contributors is also expanded, and security may be enhanced through community scrutiny.

To facilitate reuse, certain considerations should be prioritised:

- **Comprehensive documentation:** Including thorough documentation will provide clear insights into the software's functionality, making it easier for others to understand and reuse.
- **Translation support:** Recognising the importance of translation, especially for international user bases, is vital. Separating source code from textual components allows for easier translation, increasing the likelihood of third-party developer involvement.
- **Appropriate licensing:** Choosing a well-known licence that aligns with major OSS licences is essential. This ensures compatibility and encourages a wider adoption of the software.
- **Modular design:** A modular structure enhances the likelihood that others will find specific modules useful. This flexibility facilitates partial reuse, contributing to the overall success of the software.
- **Policy obligations for suppliers:** Establishing clear policies for suppliers is crucial. Key aspects include mandating an OSS licence and ensuring your administration retains control over dependencies, encompassing both software modules and services.
- **Development documentation:** Requiring suppliers to provide comprehensive development documentation ensures continuity, allowing the government to seamlessly transition to a different company if needed.
- **Encourage upstreaming:** Encouraging vendors and other users to upstream their contributions benefit the broader ecosystem. While private companies may have concerns about assisting competitors, public administrations have no conflict of interest and can focus on the collective benefit to the public sector.

Procurement of maintenance or development services

When considering support services for open source software, a public administration has the choice of relying on in-house capabilities or opting for external support through procurement. This decision involves a careful evaluation of in-house resources, capacities, associated costs, and the offerings from potential vendors.

If external support services are preferred, a procurement approach can be taken. This involves tendering and contracting with external vendors to handle maintenance, updates, and any necessary customisations. Public sector organisations should be mindful of potential security risks and dependencies on external providers. As with any software, ensuring components remain up to date can be important to for security. While proprietary software can generally only be fixed by the one company that has the source code, OSS ensures that anyone can audit, fix and distribute news code. Procedures should ensure that administrations can make use of this. You might generally get software updates from a particular company, but your procedures shouldn't block you from getting an update from another source if there is an acute need.

As procurement can be divided between the company originally developing the solutions and the one being contracted to maintain it, public organisations can stimulate competition among businesses.

When external support services are part of a tender, the accountability shifts to the chosen vendor. The procurement contract should clearly define rights related to sharing the software and its components' codes. Additionally, the contract should outline whether the vendor will incorporate software updates based on suggestions from the broader open source community. This contractual clarity ensures a smooth and transparent partnership between the public administration and the external service provider as well as long term community participation.

Security considerations

Evaluating the security of OSS differs significantly from evaluating the security of proprietary software. For management that is not yet experienced, there may be questions that can be addressed with a meeting of the relevant people. There may be questions about whether opening source code is comparable to leaving your door open, for example. For such considerations, consider training and or presentation on what OSS is. This can help dissipate common such problems.

While the question of OSS security is complex and deserves its own guidelines, certain security considerations should be taken into account when designing an OSS policy. The first involves legacy systems and the disclosure of specific software package versions. Security reviews should encompass understanding the versions of software in use, the availability of security updates, and whether upgrades are necessary. For in-house legacy software, priority should be given to publishing code that may benefit others and attract developers. This can lead to increased security rather than posing a significant security risk.

The second complex security concern pertains to trusting developers not to inadvertently publish internal data. Collaboration with other public sector Open Source Program Offices (OSPOs) can

provide valuable insights and shared experiences. Establishing collaborative documentation of best practices, including procedures to rectify any mistakes, is also helpful.

4. Co-developing and participating in OSS communities

How collaborative projects differ from in-house projects

The benefits of developing software in an open, collaborative manner extend beyond those of mere code publication. While putting an OSS licence on code makes it OSS, additional steps can yield greater advantages. This requires that public administrations build a certain level of understanding of the essence of what makes open source collaborative and how to be a valued participant in a community. One benefit of community participation is the possibility of attracting external developers. This, in turn, requires that mechanisms are in place to allow external contributions, while also thinking about the ways in which the administration wishes to keep control of the project or of the version they will use.

On the technical side, one important piece of software for managing a source code repository is Git. This is by far the most used in OSS communities, although Mercurial also exists as an alternative. Various websites exist to host software repositories, including sites specific to public administrations. The European Commission for example has created its own platform https://code.europa.eu where open source projects from the Commission and other EU institutions are shared to be reused. Similarly, the German government has created its own repository to achieve similar objectives (https://opencode.de/en).

Participating in existing communities

OSS can be used much like proprietary software. Users have the flexibility to download a specific version of software, for example, Version 7.0, use it, refer to the manual, and decide whether to upgrade when Version 8.0 is released.

However, OSS offers unique advantages due to its distinct development model:

- Incremental code changes: In OSS communities where development is organised online, code changes are introduced gradually, bit-by-bit. Changes are less likely to be lumped together with users having to accept all or none. With the source code being available, the user or another vendor has the possibility to select what is desired from an update or new version, and what isn't.
- Visible communication: When multiple entities collaborate on software development, the communication between development teams is often transparent and accessible online.

- Learning and input: Users can actively learn about the software and may even provide direct input to the developers. It's crucial for users to be aware of these possibilities.

Evaluation of involvement in OSS communities should be done on a package-by-package basis. An IT system, including dependencies, can involve hundreds or thousands of software packages and not every package requires active participation in its development. However, it can be beneficial to engage with the development of selected packages. Establishing a systematic approach for determining which software packages are worth involvement is essential.

The benefits of participating in OSS communities include:

- Awareness of development direction: By being involved, you stay informed about the direction of development, enabling better preparation and maybe the chance to have your needs taken into consideration.
- Insight into active companies: Active participation allows you to see which companies are involved in the development of specific software packages.
- Technological learning: Involvement provides an opportunity to learn about the technology, ensuring expertise for potential future needs.
- Increased acceptance of changes: Actively participating and contributing increases the likelihood of having your changes accepted upstream. This, in turn, reduces the complexity of maintenance.

What makes a good community important?

Organising OSS communities is crucial to ensuring their smooth operation. Given the hierarchical nature of the public sector, effective organisation becomes particularly vital in demonstrating a project's success and sustainability. Establishing clear governance and operational guidance in collaboration with the community is essential, allowing it to operate and grow freely.

Several examples highlight the importance of community organisation:

- W3C Accessibility Guide: The W3C has compiled a comprehensive guide on accessibility, incorporating international accessibility principles.
- UK Government Guidance: The UK Government has provided dedicated guidance explaining the value and process of initiating source code openness from the project's outset.
- Developers Italia Community: The Developers Italia community has invested resources in physical gatherings, fostering communication, efficient collaboration, and personal connections within the community.

Decision-making structure

The sustainability of an open-source community hinges on good quality leadership and transparent management. Community governance should strike a balance between openness and the organisational structure of the public administration. Key components include:

- Identifiable public sector manager(s): Responsible for enabling flexible and transparent community operation, decisions by public sector management should be clearly communicated to the community. Regular consultation with community members strengthens their motivation to contribute.
- Project manager/steering committee: This team, well-versed in OSS and public sector operations, acts as a facilitator between the community and the public administration. Inclusion of key developers in the project management team ensures specialised knowledge.

Long-term sustainability

To ensure long-term sustainability, community management should consider the following:

- Flexibility in role changes: Facilitate organic growth in community members' responsibilities and plan for potential replacements of managers to adapt to changing circumstances.
- Training future leaders: Dedicate resources to training future community leaders, recognising that a strong team is crucial for sustainability.

To look at the development of sustainable communities, check out OSOR's Guidelines for Sustainable Open Source Communities in the Public Sector to learn more.

5. Specific aspects of open source funding

Budget implications

While financing open source project use or development would need its own guidelines, some general remarks should be taken into consideration to avoid common pitfalls.

As OSS projects are not a typical procurement product, they often need alternative funding mechanisms and budgeting. Initial development may constitute a one-time capital expenditure (CapEx) while maintenance becomes an ongoing operational cost (OpEx). Smaller projects may opt for ad hoc maintenance with vendor involvement as needed. If budget structures assume software infrastructure funding as CapEx, adjustments may be necessary to accommodate OpEx for significant maintenance phases.

Being open to changes, especially from early feedback, can significantly boost a project's success. Administrations should be ready to adjust investments based on good information,

ensuring changes happen early. Grantees should start with a clear plan but be open to adapting based on user feedback, with all project info available for inspection. Overall budgeting should allow flexibility in the CapEx/OpEx divisions.

Key recommendations include checking if the software has a realistic scope and fits the budget. Having a feasible way to get users and addressing deployment needs from the start is vital. Also, plans may be required for long-term data management, security reviews, and transitioning legacy data. Ensure the software aligns with goals and supports third-party commercial support for successful outcomes.

6. Legal framework

What regulation to consider when adopting OSS?

Considering existing regulation when adopting OSS in public administrations is crucial to avoid complications down the road.

Legal compliance: Existing regulations govern various aspects of software usage and licensing. Ensuring compliance with these regulations helps organisations avoid legal issues and potential penalties. Many software regulations, at the EU level, contain special provisions or clarifications regarding OSS. OSS licences are also a part of the legal context which administrations should have some understanding of.

Data protection and privacy: Many regions have strict regulations regarding data protection and privacy. Adopting OSS that adheres to these regulations ensures the safe handling and storage of sensitive information, protecting the organisation and its users.

Interoperability: Existing regulations often specify standards and requirements for interoperability between different systems. Adopting OSS that complies with these standards facilitates smoother integration with other software and systems, reducing compatibility issues.

Security standards: Regulations may outline security standards that organisations need to follow. Choosing OSS with a strong focus on security and that complies with industry standards helps in maintaining a secure software environment.

Public sector requirements: Public sector organisations in particular must adhere to specific regulations and standards. OSS adoption should align with these requirements to ensure that public services are delivered efficiently and securely.

i. European Regulation

When looking at existing laws to follow when using OSS, the European Union has created several regulations with direct effect and directives which have been implemented nationally.

The European Interoperability Framework and the upcoming Interoperable Europe Act are two examples of design requirements created by law that should be considered when adopting OSS in public administrations. Additionally, more well-known regulations like the General Data Protection Regulation, the Open Data Regulation, or the Single Digital Gateway are also examples of laws to consider. When searching for an overview of EU digital regulations, you can find good summaries from external actors.

ii. National

As all legal systems in the EU have their disparity, it can be useful to turn to your organisation's legal expert. If that's not possible, checking the law repository of your country regarding digital regulation can be a first step. If your country is in the EU, an analysis of its legislative framework on OSS can be found in OSOR's Country Intelligence Reports. Overall, it is important to know that while your own organisation might not be used to managing OSS, most EU countries have had some form of experience with open source and legality. We therefore highly recommend contacting anyone that has been part of these experiences to learn more about the national context of your organisation.

iii. Local

While this scale of regulation might be more limited, it is also potentially the easiest one to adapt to the use of OSS. Many cities, regions and local governments create their own digital strategy. Learning about the guiding principles of these sub-national entities with a digital strategy can allow you to more effectively adapt your OSS migration project and understand how to use this system to your advantage. If the question of OSS does not seem to have been worked on previously at the local level, try to contact similar-minded individuals and organisations to exchange your knowledge and experience.

Licensing

Identifying OSS by looking at the licence

Given a software package, checking whether it is OSS does require a small amount of expertise. However, the conclusion is usually unambiguous and there are probably many others who have done this evaluation. This information is available in many of the OSS catalogues that are maintained. If a user wanted to dig a little deeper, the licence is typically mentioned on the project's website or on the repository where the source code is hosted. Most OSS projects are under a small number of common OSS licences:

- GNU General Public License (GPL)

- European Union Public Licence (EUPL)
- Mozilla Public License (MPL)
- Eclipse Public License
- Apache License
- 3-clause BSD or X11 licence (permissive or lax licence)

If a project is under any of these licences and the source code is available, then it’s OSS.

There are other licences, but generally less used. In the late 90s and early 2000s some companies and projects made their own licences, but this created complications when combining their code with code from other projects. Since then, creation of new licences has slowed and there has been some convergence towards the more common licences.

There can be OSS which is combined with proprietary software - software where some modules are OSS and others aren’t. This would mean that the combination as a whole is not OSS. This is sometimes called “open core”, where the base of the software is OSS but further development is not.

Licences were discussed at length in the OSS ecosystem in the early 2000s. While many in the private sector would consider the topic to be mostly settled, this is less often the case in the public sector.

Types of OSS licences

Probably the aspect of an OSS licence which will have the most impact is to what extent it is “copyleft” (or “reciprocal” or “share-alike”). As well as copyleft and non-copyleft licences, there are licences which are somewhat in the middle.

The oldest licences were permissive licences, which say you can do what you want with the software. Examples include the 3-clause BSD licence or the X11 licence (also called the MIT licence, but that name can be ambiguous). If the software and source code are available under such a permissive licence, it is OSS. Note that there's no requirement to publish the source code, which is why it's necessary to clarify that software under these licences is only OSS if the source code is available.

Copyleft licences were developed in the 1980s. They add the requirement that if you publish the software or a modified version, then you must make the source code of the software available under the same licence or a compatible licence (including any modifications that have been made to the software). The result is that if a software package has a copyleft licence, all future versions of that software package, by the original author or by a third party, will be OSS.

The advantage of permissive licences is that code can be reused by more people because people and companies can use the code even if they don't want to share their modifications. In the words of European Commission decision CI 495/1, permissive licences are those ”granting the broadest rights of use to the users.”

This is also the disadvantage. If a government agency produces a software package and releases it as OSS under a permissive licence, a company can copy that code and add features or fix bugs and then republish their own version as proprietary software, without making the source code available. The non-OSS version could be technically superior because it contains all the features developed by the government plus a few proprietary features. This leads to the situation where users of the OSS version will always be using a less-developed version of the software even if they developed most of the software themselves. If a government agency wants to be sure that people building on top of their work will share with the government, in the same way the government shared with them, then a copyleft licence ensures this.

The GNU GPL is the most widely used copyleft licence.

There are also licences which contain copyleft clauses plus a set of exceptions. Two other copyleft licences are the European Union Public Licence (EUPL) and the Mozilla Public License (MPL).

The MPL's copyleft clause is narrower than the GPL’s as it applies only to files, not the project as a whole. If someone downloads source code that is distributed under the MPL, and they modify those files and they also add write some code in a few new files, then the MPL’s copyleft clause requires that the changes to the project’s files be distributed under the terms of the MPL, but there are no requirements about distributing the source code in the newly added files.

The EUPL’s copyleft clause has a scope that’s similar to the GPL, however, it also has a compatibility clause which allows anyone to mix code under the EUPL with code under one of a list of licences, and distribute the combination under the other licence. The GPL and the MPL are on the list of licences. The result is that modified versions of software published initially under the EUPL may later be distributed under the GPL or the MPL.

The EUPL also deserves special attention because it is published in 23 official languages of the European Union and it is compatible with the law of all EU countries. Also, the European Commission designated it in 2021 as the default licence to be used when it releases OSS. These factors contribute to the fact that the use of the EUPL can thus be expected to grow in national and local governments in the EU.

Licence compatibility

Two licences are said to be “compatible” if it's possible to take code under one licence, add it to a project with code under another licence, and distribute this combination. The essential question is, is it possible to comply with both licences at the same time?

Some projects adopt dual licensing, allowing users to choose between multiple licences. This creates a situation similar to the compatibility clause of the EUPL, in that compatibility is made easier but the copyleft clause is weakened.

In the early 2000s it was complicated to share code between copyleft projects if they were not using the same licence. To solve this, the EUPL and the MPL added a clause saying that if code under these licences is mixed with code under the GNU GPL, the combined work can be distributed under the terms of the GNU GPL.

Maintaining clarity about applicable licences and understanding their compatibility is crucial, especially when dealing with licence stacking, where a project incorporates code under multiple licences. Maintaining clarity about the licence situation of a project is essential to ensure a smooth and legally sound collaboration between OSS projects. It is good practice in OSS projects to include a licence statement at the top of each file, stating the names of the copyright holders and the licence of the content of the file.

Also for documentation

Documentation licensing can also be important when designing a OSS project. For example, this handbook is distributed under the Creative Commons 4.0 share-alike licence. And there are many other publications that use that licence. One example is OSPO Alliance's book on building an OSPO, the Good Governance Initiative Handbook. If someone wanted to combine this book and that book, or parts of either, there would be no licensing issues.

Similarly, the Creative Commons share-alike licences contain a clause that explicitly allows their content to be distributed under the terms of the GNU GPL. If someone wanted to take some material from this handbook – or maybe from a technical book which might include some code in the text – they could add that to a software package that's under the GPL and there would be no licensing issues. Anyone distributing the combination would only have to comply with the requirements in the GPL.

Picking the licence that works best for your needs when using and developing FOSS

If you’re unsure about what licences could be the best for your project and or are looking into licensing compatibility, multiple tools have been created to help such a process:

The European Commission has its own licensing tool, the Joinup Licensing Assistant (JLA). The JLA is a helpful tool for comparing open licences. It allows users to choose a licence based on their specific needs and legal requirements. The JLA includes an analysis of the language used in each documented licence, making it easier for users to understand their legal implications. Additionally, it links to Software Package Data Exchange (SPDX) identifiers and enables users to test compatibility between two licences. This makes it simpler for developers to make informed decisions when working with code from different sources, promoting transparency and smooth collaboration in the open source community.

It also features the Compatibility Checker, a straightforward tool where users can select one inbound licence (for third-party source code intended for their project) and one outbound licence (already covering their main project source code or intended for distribution). This functionality aims to assess the extent and compatibility of licences when working with data or software components licensed under different terms. It helps determine how the project, incorporating elements from various licences, can be distributed and, if feasible, under which licences.

Other tools have also been developed to fulfil similar or equivalent objectives. For example, a recently developed tool Hermine takes a slightly different approach to provide a comprehensive toolchain for handling open source compliance. It divides licences' characteristics into three categories and provides optional information such as licence version, root of the licence name, authorisation for applying later versions, steward entity, inspiration from other licences, clauses against tivoization and status of whether it is approved as a free software licence according to the Free Software Foundation (FSF) and an open source software licence according to the Open Source Initiative (OSI). Hermine enables the definition of authorised contexts for licences restricted to certain use cases and categorises obligations within licences systematically.

7. Structuring the use of FOSS

Benefits of a formal structure

Given the widespread use of OSS in public administrations today, there's an ongoing discussion among experts on OSS-related issues. While informal discussions may already take place, there are distinct advantages to formalising this coordination.

Embracing a structured approach can significantly reduce legal workload by documenting coordination efforts, preventing the need for repetitive legal tasks. Moreover, this formality minimises legal risks as shared work undergoes thorough review, leading to early issue identification. Additionally, formal coordination encourages the sharing of knowledge and experience among stakeholders, fostering a collaborative environment. The structured approach also facilitates the leveraging of scale, enabling the pooling of resources for more efficient

outcomes. Furthermore, the formality increases visibility, attracting more input and spreading the benefits of FOSS adoption.

Legal entity

If a legal entity is to be created, the exact format and funding model will largely depend on national laws and political will, but several models have been successful. Examples include:

- ZenDis (Germany)
- OS2 (Denmark)
- Free Software Unit in DINUM (France)
- Developers Italia (Italy)

Governments can opt to hire a secretariat along with developers, or a secretariat that collects funds and disburses payments to developers, or simply create a network that organises funding for developers, promoting a collaborative financing approach.

Open Source Program Offices (OSPOs)

OSPOs have gained attention in the public sector from people hoping to address OSS challenges in the public sector via resilient structures.

Public sector OSPOs, if they are not dependent on the support of one person or party, can provide continuity in the OSS strategy of a government or administration. However, their establishment demands political will, necessitating a well-thought-out plan.

OSPO services can include procurement support, training for procurement officers, coordinated procurement of services, explaining the rationale behind OSS usage, technical support for users, security audits, assistance in policy setting, and coordination of financing among administrations.

Other organisational models worth investing time in:

These are areas which workshop participants have noted are worth investing in:

- Stewardship or maintenance of own code
- Community building:
    - Lowering the barrier for entities to become new users
    - Lowering the barrier to entities proposing new projects
    - Helping new entities become an active part of the community
- Education on releasing code

8. Getting help

Documentation

Numerous organisations and individuals have dedicated time and collaborative efforts to creating and maintaining various kinds of resources to guide others in their use of open source.

On OSOR, for example, you can find case studies, guidelines, and reports studying and exploring how to effectively deploy OSS strategies inside a public administration. There you can have a look at the Knowledge Centre, which displays resources available to OSOR users.

As mentioned earlier in this handbook, many organisations and projects have created guidelines that can be useful. Here's a non-exhaustive list of projects, guidelines and articles we think could be interesting:

- CHAOSS - Community Health Analytics in Open Source Software and their metrics
- OpenSSF Training and Guidelines on Securing FOSS
- Foundation for Public Code and their Standards for Public Code on guidelines for public sector OSS
- French free software mission guides to the use of OSS
- The OSPO alliance Good Governance Handbook
- The video series from a collaboration of public and private Dutch actors on how to use open source in public administrations
- The Italian administration guides to software reuse and digital development in general
- Todo Group's guides on various OSS and OSPO related matters
- OSPO++ articles and resources on OSPO
- OSSBIG various reports, among which there's the one on Open Source Governance

There are many more guidelines existing on this subject. You can find more information by looking at the different national organisations at the EU level here.

Forums and networking opportunities

Overall, while the public sector experience of OSS exists and has been documented, the ongoing efforts to create comprehensive guides at the EU level are still sometimes uncoordinated due to the lack of communications between administrations. We greatly encourage you to connect with other administrations working towards the same goals. There are many yearly events dedicated to the use of open source and open source in various contexts of use in the public sector. OSOR publishes monthly announcements on some of those events, and we highly encourage you to try to meet like-minded organisations or to directly contact them.