

Selection Criteria for the sustainable Procurement of Open-Source Software

A position paper by the Open Source Business Alliance

Federal Association for Digital Sovereignty e.V.

11 February 2025

Authors: Procurement Working Group at the Open Source Business Alliance (represented by the Spokespersons of the Working Group Birgit Becker and Claus Wickinghoff)

English translation by Richard Brock and Lily Logua of Collabora Productivity

Disclaimer

This position paper gives advice to public authorities on how they can sustainably procure open-source software and recognise and choose high-quality providers. This is merely guidance and not legal advice.

Table of Contents

Open-Source Software in Public Administration....3
How does Open-Source Software Work?....4
What are the Advantages of Open-Source Software?....4
What is the Community?....5
Procurement of Open-Source Software....5
Procurement – But Done Right!....6
Criteria for Sustainable Procurement....8
1. Relationship with the Software Manufacturer / Community....9
2. Ensuring Upstream Publication of Modifications....10
3. Ensuring High-Quality Level 3 Support....11
4. Securing the Supply Chain Through Support for Core Components....11
Certifications in the Open-Source Sector....12
Appendix: Criteria Catalogue....13
Selection and Evaluation by the Specialist Department....13
1. Criterion: Relationship with the Software Vendor/Community....13
Proposed wording as an A-Criterion....13
Proposed wording as a B-Criterion....14
2. Criterion: Ensuring Upstream Publication of Customisations....14
Proposal for A-Criterion formulation....14
Proposal for B-Criterion formulation....14
3. Criterion: Ensuring High-Quality Third-Level Support....15
Proposal for A-Criterion formulation....15
Proposal for B-Criterion formulation....15
4. Criterion: Securing the Supply Chain through Support of Core Components....16
Proposal for A-Criterion formulation....16
About the Open Source Business Alliance - Federal Association for Digital Sovereignty e.V....17
Copyright License....17

Open-Source Software in Public Administration

The public administration in Germany is striving for digital sovereignty, i.e. the ability to independently check, design and exchange its own IT systems. To achieve this goal, it is increasingly using open-source software.

Open-source software is characterised by its open and cooperative approach and the freedoms granted by the software licenses. As a result, the development and distribution models of open-source software differ from those of proprietary software. This can pose a challenge for public administration, which for decades has relied exclusively on proprietary software, particularly in its procurement and awarding processes.

The business models behind open-source solutions function fundamentally differently than those for proprietary software: With proprietary software, the manufacturer participates in every license sold, even if the license was granted by a third-party provider. With open-source software, additional services are usually offered commercially instead of the software license. This gives third-party providers the opportunity to outdo the actual software manufacturer in a bidding process by making unrealistically low offers. In this situation, no money goes to the open-source manufacturer, who is therefore unable to invest sufficiently in the further development and maintenance of the open-source solution.

One argument for using open-source software in administrative digitisation is the potential for reuse, i.e. that an authority can continue to use the software that has already been developed and paid for by another authority if the corresponding source code is published on a publicly accessible platform such as openCode.[1] However, this promise of reuse is not possible if insufficient funds are invested in the continuous development and maintenance of the software, and it is therefore not economical for the manufacturer to continue development at all. This is then detrimental particularly for customers, as less high-quality and up-to-date open-source solutions are available on the market.

This also jeopardises the IT security of corresponding open-source solutions in public administration or may lead to the failure of IT projects. These problems are then often blamed on the usage of open-source software, rather that the awarding of contracts to an unqualified vendor. This damages the general reputation of open source software.

The challenge for public administration is therefore to develop the right requirements and award criteria so that they can reliably select from service providers those who are able to offer sustainably secure and high-quality software and services.

It is already possible today to include other selection criteria in addition to the price in the evaluation of a tender. The document for tendering and evaluation of IT services (UfAB 2018)[2],

1 The German federal government has created the Centre for Digital Sovereignty (German: “Zentrum für digitale Souveränität” – short: “ZenDiS”) to push for a strategical use of open-source software in public administration. In order to achieve this goal, the centre also realises open-source projects like openCode. This is a public platform for the exchange of open-source software specifically for public authorities.

2 https://www.cio.bund.de/SharedDocs/downloads/Webs/CIO/DE/digitale-loesungen/it-beschaffung/ufab/ufab2018.html

section 5 (page 71 ff.) lists additional criteria for the procurement concept: Aspects of sustainability, Inclusion of know-how from the service-provider markets, Definition of suitability criteria, and Award criteria.

This paper provides public administration with guidance on sustainable procurement of open-source software, helping them identify and select high-quality offerings which will also strengthen the open-source ecosystem in the long run, ultimately supporting the goal of digital sovereignty.

How does Open-Source Software Work?

Open-source software is basically software like any other. For instance, there are a number of open-source solutions that are developed by professional companies and offered with professional support, warranty and liability.

The difference between open-source software and other software lies in the licensing. While non-open-source software (often referred to as proprietary or closed-source software) significantly restricts the usage of the software - usually to opening/using and creating a backup copy - the licenses of open-source software grant extensive rights of use.

In general, open-source software may be used, viewed, improved and even passed on at will. For this to work, the source code is included with the license. Many open-source licenses are provided with easy-to-fulfil license conditions, for example that a disclaimer must be included. Some of these licenses require any (modified) source code to be passed on to the recipient under the same license if derivative works are created - this is known as copy-left. To support open-source software in the public sector, the EU has developed its own license with a copyleft character, the EUPL, which is available in all EU languages.[3]

The main idea behind open-source licenses is that the user is able to adapt the software to their needs and can continue to work with the software even if their requirements change. From a security perspective, this also makes it possible to check software for vulnerabilities and backdoors. With regard to sustainability, the programme code discloses the data formats used so that existing data can be imported or exported and used in other systems.

What are the Advantages of Open-Source Software?

Overall, open-source software promotes cross-company and cross-border collaboration as well as innovation and competition. Over the last 30 years, many relevant software projects have been created in this way:

- the basic technologies for the Internet, without which digitalisation would have taken a completely different course
- the Linux kernel, which today can be found in Android smartphones and many devices such as routers
- Kubernetes, which forms the basis for modern container orchestration

3 https://eupl.eu/

- Database management systems such as MariaDB or PostgreSQL
- Open-source AI frameworks
- Web browsers such as Firefox
- Office programmes such as LibreOffice
- Open-source video conferencing systems such as Jitsi or BigBlueButton

The goal to support or create sustainable solutions with public funds can be best realised with open-source software. As the source code is open and can be used freely, public authorities can freely share developed solutions with each other. And further functionalities can be added over time as each institution customises the software to its specific needs. This is well summarised in the slogan ‘Public Money, Public Code’: publicly financed software should be published under an open-source license so that the results can be made available to the general public. In this way, the development and procurement of open-source software also contributes to the responsible and sustainable use of public funds.

What is the Community?

The term ‘community’ is often used in connection with open-source software.

Every piece of software has a user community. With open-source software however, the open license gives everyone the opportunity to help shape the software. In this way, a user community also makes contributions to the software itself, which are often actively maintained in the long term.

The members of a developer community range from interested users, who may develop software in their spare time, to programmers who work on software during their working hours, to companies whose business model is based entirely on the support and development of open-source software.

As such, an open-source community consists of various people and organisations that use the respective software and may also contribute to the maintenance and further development.

Most software products consist of various components, each of which may have its own developer community behind it. When professional software providers use such components for their software products, they often commission their own employees to participate in the development as part of the respective community.

Procurement of Open-Source Software

The open-source ecosystem consists of a diverse range of commercial and other actors (e.g., volunteers). However, software procurement can typically only take place through commercial providers, as only they are legally and organisationally capable of meeting the requirements of procurement law.

To be economically and financially viable (see § 45 of the German Federal Public Procurement

Ordinance on the awarding of public contracts), an applicant or bidder must generate revenue. In the case of open-source software, providers typically do not earn money from licensing fees. Instead, public-sector entities usually procure services related to open-source software—essentially support, operation, customisation, or further development. The quality of these services depends on the provider having a sufficient number of qualified personnel available for these tasks.

Like any software, open-source software requires continuous maintenance and updates. This applies both to solutions and customisations developed by a commercial company for a specific client and to standard software.[4] These ongoing costs must be factored into the pricing of bids.

If public-sector contracts are awarded based solely or primarily on price, this disadvantages applicants/bidders who follow a sustainable business model. These are the providers who invest in the continuous maintenance and updating of the software and secure the supply chain—following common industry practice—by signing support agreements with the manufacturers of individual software components.

Unreliable applicants/bidders can undercut prices in various ways to secure contracts:

- The provider may fail to allocate sufficient resources for in-house development or maintenance, instead relying excessively on freely available contributions from other manufacturers or volunteer developers.
- The provider may save resources by not spending time making the developments or patches they have made freely available to the general public again.
- The provider may forgo in-house developers altogether and fully rely on publicly available support channels (e.g., online forums) from the original software manufacturers for troubleshooting. By avoiding support contracts with the manufacturers of integrated components, unreliable bidders can undercut their bid price. However, this comes at the expense of the supply chain, which is essential for ensuring the quality and security of the software. If companies with such business practices are awarded contracts, insufficient investment flows back into the maintenance and further development of the software in question. This directly leads to quality issues in the services procured by the public sector and, indirectly, to the stagnation of publicly available code, which does not receive updates and improvements proportional to its usage.

This, in turn, negatively affects the open-source ecosystem as a whole and reduces the public sector's ability to procure open-source software in the future.

Procurement – But Done Right!

Concerns about unreliable providers are not unfounded, as demonstrated by examples from a survey conducted by the Open Source Business Alliance among its members in spring 2024. These examples also highlight the risks for public-sector procurement:

4 Standard software describes software programs, modules, tools et cetera that are being developed for the needs of a majority of customers and that are not being developed specially for a single customer by the provider.

Experience of an Open-Source Software Manufacturer:

> *"A federal state decided to issue a tender for our software to be used in all schools in this state. Companies X and Y won the contract, even though neither had any prior experience with our software.*
> *We also participated in the tender process. The state had very high requirements with strict criteria for performance and other factors. We carefully considered how these requirements could be met.*
> *Companies X and Y won the contract by offering unrealistically low prices and later approached us for consulting on scaling and performance issues. We declined and referred them to our standard subscription service, but there was no budget left for this in the project."*

Conclusion: The manufacturer submitted a well-calculated bid that accounted for scaling and performance. However, a competing provider won the contract with a significantly lower bid. The client might then perceive the software as inadequate, when in reality, the issue lies with the contractor, who neither had the necessary expertise nor secured it through a support contract with the software manufacturer.

With closed-source software, providers always purchase licenses from the manufacturer, ensuring financial compensation through the licensing model. In open-source software, this is not necessarily the case. In this instance, there was no contractual relationship with the open-source manufacturer, leaving them with little motivation to provide free assistance. As a result, the client felt abandoned.

Experience of another Open-Source Software Provider:

> *"A year and a half ago, the CIO of a federal state congratulated me on winning a deal for video workstations using our software, enthusiastically declaring himself a strong supporter of open source.*
> *At first, I had no idea what he was talking about. It turned out that Company Z had won a tender using our software. Ultimately, they hired a third party to fork our software and modify it to meet the state's accessibility requirements. The CIO believed this was a great win for open source—but in reality, all the expensive development work commissioned by the state is now closed-source.*
> *The absurd part? A federal ministry is now investing a large sum of money into accessibility improvements, even though Company Z had already developed these same features with taxpayer money. However, these modifications are not open-source."*

Conclusion: In this case, the provider made modifications in a fork, separating it from the original open-source code. At the same time, the provider did not release this fork under an open-source license for public use.[5]

5 There are a couple of things that have to be considered when creating a fork, a separate derivative that is. A completely new codebase is being created that has to be continously maintained. It depends on the number of users, the general reuse of the software and the emerging community if a fork can take root. LibreOffice and Nextcloud are examples for successful forks. A fork makes it possible to implement individual needs and

When commissioning modifications or enhancements to open-source software, public-sector entities must ensure that the results are published under an open-source license and ideally reintegrated into the original open-source project. The location for delivering the software should also be defined in advance, such as openCode. Otherwise, development risks heading into a technological dead end, requiring costly re-integration of updates from the original project into the fork with each update cycle.

Additionally, the administration becomes dependent on a single provider who exclusively controls the modified code. This contradicts the goal of vendor independence. If the provider ceases operations, the modified software may no longer be reliably usable. Furthermore, other institutions will struggle to reuse the adapted software.

Experience of another Open-Source Software Manufacturer:

> *"Some companies flood us with tickets and support requests, claiming that the software doesn't work. What they don't disclose is that they are fulfilling a support contract for a client, but they lack the necessary expertise and don't want to pay us for assistance.*
> *Distinguishing such cases from genuine users in need—whom we willingly help as part of our community work—is challenging."*

Conclusion: Most open-source companies provide free support within reasonable limits for individual users, typically via public channels like forums or open ticket systems. This is done voluntarily and is intended for individuals or small organisations, not commercial users. There are no guaranteed response times for such support.

Exploiting this system—by offering paid support to a client while relying on the manufacturer's public support channels—is both unethical and irresponsible. It is unethical because the manufacturer's customers ultimately fund its support staff, and unpaid extra work for these employees reduces resources available for genuine community support. It is irresponsible because standard support contracts usually include service-level agreements (SLAs), which cannot be met under such an informal arrangement.

Criteria for Sustainable Procurement

The negative examples outlined above illustrate how the open-source ecosystem can be damaged in the long term if public procurement fails to consider the specific characteristics of open-source software. This becomes a problem for public-sector clients because it reduces the availability of high-quality, up-to-date open-source software in the market and limits opportunities for its reuse.

The core idea of open-source licenses is to enable software users—in this case, the public sector—to make their own modifications, develop their own enhancements based on the software, and potentially distribute them further. In public administration, service providers are typically contracted to carry out these modifications or enhancements.

Public administrations therefore should ensure that any modifications and deliverables produced

modifications that for one reason or another were not realised in the original software project. This is only possible when using open-source software.

by contractors are made publicly available and reintegrated into the relevant open-source project. Otherwise, an unintended dependency on a single provider may arise.

For the sustainable use of open-source software, it is essential not only to consume the software but also to "invest" in the open source ecosystem. There are many ways to do this, from contributing code regularly, to creating documentation or funding developers. This is particularly important when procuring open-source software maintained by a single company—such companies should be involved in the project under tender. This ensures that funds remain available for ongoing software development and maintenance while also addressing regulatory requirements, such as those of the Cyber Resilience Act (CRA). Ultimately, this benefits the contracting authority as well.

For the procurement of open-source standard software, using price as the sole evaluation criterion is usually not suitable, as the cost of maintenance and further development is not automatically included in every offer.

The Guidelines for Tendering and Evaluating IT Services (UfAB) provide alternatives to purely price-based decisions in Section F 4.2.1. In Section B 5.4.1, sustainability is specifically highlighted as an important aspect. The guidelines state:

> *"Sustainability in execution conditions under § 128 (2) of the GWB [German Competition Act] may include contract-related execution conditions, particularly economic, innovation-related, environmental, social, or employment policy concerns."*

Providers that ensure the continuous development and maintenance of open-source software promote the above-mentioned economic and innovation-related concerns, ultimately benefiting public-sector clients (e.g., through reuse opportunities).

The following criteria should be included in tenders for open-source software to ensure sustainable procurement and identify a competent service provider. Specific text modules related to these criteria can be found in the appendix. Naturally, additional technical requirements will apply depending on the project, but those are not the focus here.

1. Relationship with the Software Manufacturer / Community

The public contracting authority benefits in various ways when the service provider can demonstrate a relationship with the software manufacturer or community.

Providers with a close relationship to the manufacturer or community play an active role in the ongoing development and maintenance of the software. They can directly contribute specific requirements and suggestions for improvement, enabling the software to be tailored to the contracting authority's specific needs. This ensures the desired functionality is met and user-friendliness improved, while also minimising the risk of incompatibilities and fostering a stable IT infrastructure.

A close collaboration between the contractor and the manufacturer or community of the utilised open-source solution ensures direct access to technical support and the latest software updates. This is crucial for the rapid resolution of issues and enhances the operational security of the

software solution. Fast response times are particularly valuable in critical applications.

The contracting authority benefits from early awareness of potential security vulnerabilities and can take preventive measures to minimise the risk of security breaches. The Cyber Resilience Act (CRA) comprehensively addresses this issue and requires supply-chain security and a breakdown of the software components via a Software Bill of Materials (SBOM). Close collaboration with the software manufacturer or community ensures the high security standards that are mandatory for public authorities.

The relationship with the manufacturer or community is also crucial in determining how long development, maintenance, and support for the utilised software will remain available. The longer an open-source software solution can be used within public administration, the more cost-effective its use of resources becomes. Long-term availability is also relevant for the reuse of the software across different administrative bodies.

Thus, incorporating this criterion ensures that the chosen open-source software is long-lasting, cost-efficient, tailored to specific needs, and secure.

2. Ensuring Upstream Publication of Modifications

A fundamental characteristic of open-source software is its adaptability to specific needs. Missing functionalities, for example, can be added through patches. However, if these changes are not submitted upstream to the central codebase, the modified version becomes a fork—a separate derivative. In the long term, this poses a maintenance challenge for the contracting authority. The central code is continuously developed by the manufacturer or community. The client's fork must also be constantly adapted, particularly with regard to new functions and particularly when security issues are fixed. Given that such codebases often consist of millions of lines of code, maintaining a fork can quickly become unaffordable and uneconomical.

It is necessary and sensible to clarify the handling of patches during the procurement process. When acquiring open-source software, the contractor should be required to submit any patches "upstream" to the central codebase. The evaluation process and the contract design should take into account that the manufacturer or community decides on the acceptance of such patches. The closer the service provider's relationship with the manufacturer or community, the higher the likelihood that these patches will actually be integrated into the central code.

By submitting modifications upstream, the entire community can review, test, and further improve these changes. This open review process enhances software quality and helps to identify and resolve security issues. Providers who actively contribute to the open-source community leverage collective expertise, resulting in more robust and secure software.

A contractor who ensures that modifications and patches are upstreamed facilitates future software updates and maintenance within the scope of the contracted project. Custom-developed features and improvements will already be integrated into the main codebase, reducing effort and costs for software updates and maintenance for the contracting authority.

Consistently submitting changes upstream supports sustainable software development, as improvements benefit all users rather than just a single project or customer. This also enables the

reuse of developed software solutions or components across the public sector under the "one-for-all principle". Additionally, this approach aligns with the "Public Money, Public Code" principle, which advocates that publicly funded software should be made available to the general public.

3. Ensuring High-Quality Level 3 Support

The quality of support has a direct impact on the satisfaction and productivity of end users. A structured support system that guarantees fast response times and technical expertise ensures that both users and administrators receive effective solutions to their issues. This not only saves valuable working time for the contracting authority but also enhances the user experience and acceptance of the software solution.

Modern software solutions are becoming increasingly complex and require in-depth technical expertise for effective support. Unlike proprietary software, open-source software is freely available, and in principle, any company can offer services and support for it. Level 1 and level 2 support are relatively straightforward. Basic user inquiries—such as resetting a forgotten password (level 1 support) or restarting a server, applying updates, or adjusting configurations (level 2 support)—require only general IT knowledge.

However, level 3 support for open-source software requires expert knowledge of the software's source code. A thorough analysis of the source code is necessary to resolve malfunctions in the software used (level 3 support).

To meet contractually agreed response times—such as those defined in Service-Level Agreements (SLAs) —the provider must either have its own qualified personnel with the necessary expertise, or secure contractual agreements for specialised support from the original software developer or community maintainers.

This criterion is of critical importance for ensuring the contractor receives the service needed for their operations. It ensures operational continuity and minimises the risk of downtime through quick and expert problem resolution.

4. Securing the Supply Chain Through Support for Core Components

The long-term availability of an up-to-date and maintained software solution is of critical importance to the contracting authority. Open-source software usually consists of various core components, which are continually integrated into new software solutions and therefore can be used in a variety of contexts. These core components, which are part of the supply chain, are often developed and maintained by volunteer contributors or organisations within the open-source community. The security of any software solution depends on ensuring that all the components used in it are kept up to date. Therefore, when awarding a contract, the contracting authority should also pay attention to the supply chain and choose providers who themselves actively engage with the open-source community to support the core components they are using.

For example, providers can support the developers of these core components in implementing the requirements and documentation obligations stipulated by the Cyber Resilience Act (CRA).

A contractor's involvement with the open-source core components signals a transparent way of

working and a willingness to share knowledge and experience with others. This strengthens trust in the provider, as it shows that they are not only benefiting from the open-source community but are also willing to contribute their own input for the common good, thereby ensuring the long-term availability of the software. A provider who proves to be an active part of the community can rely on broad support in solving problems, which ultimately also benefits the contracting authority.

This criterion ensures the long-term availability of support, updates, and further developments for all components used in open-source products, thereby guaranteeing the security of the supply chain for public administration.

Certifications in the Open-Source Sector

Similar to proprietary software, the open-source sector also offers various certifications that providers can obtain. When the public sector seeks to select a suitable service provider, it may be worthwhile to check whether there is an appropriate certification relevant to the subject of the tender that can serve as evidence of the provider's quality. If such certifications can be presented, they should be included in the evaluation of the providers.

Some open-source vendors, such as Nextcloud or Univention, certify partner companies they collaborate with. These certifications are well-suited to demonstrating an existing relationship between the provider and the software vendor (see Criterion 1).

Other open-source vendors or communities, such as LibreOffice or RedHat, certify developers who possess specialised expertise related to a particular open-source software. If a service provider employs such certified developers, this certification is a strong indicator of the provider's expertise in relation to the upstream publication of modifications and patches (see Criterion 2) as well as the provision of high-quality level 3 support (see Criterion 3).

The international "OpenChain" standard ISO/IEC 5230 focuses on software supply chains, simplified procurement, and license compliance. The OpenChain standard can be awarded by an accredited certification body or obtained through self-certification. This certification is useful for demonstrating that a provider has already engaged deeply with open-source licensing requirements and the specifics of the open-source ecosystem. Since generating a Software Bill of Materials (SBOM) is also part of this standard, the certification can further serve as evidence of a provider's commitment to securing the supply chain through support for core components (see Criterion 4).

Not all areas or open-source product groups have relevant certifications. Nevertheless, it may be beneficial for public authorities to conduct a brief preliminary search for any suitable certifications related to the subject of the tender. While certifications can serve as proof of a provider's quality, it may be counterproductive to require a certification that is excessive in scope or irrelevant to the specific procurement subject.

Appendix: Criteria Catalogue

Selection and Evaluation by the Specialist Department

This appendix is designed as a "modular toolkit" and should be adapted to the specific tendering situation. Texts from the previous chapter "Criteria for Sustainable Procurement" can also be used for supplementary explanations. These proposals do not constitute legal advice and must be reviewed for procurement compliance by the contracting authority.

The UfAB (Guideline for Tendering and Evaluation of IT Services) defines the legal framework for determining the most economically advantageous offer in Section 4.2.1. Since open-source software does not require payment for the software itself and so no automatic calculation for further development and maintenance is included, the method of a simple price evaluation is inapt to choose a sustainable offer.

In practice, the method of a simple price evaluation is not applied in many tenders anyway, instead contracting authorities use a combination of criteria to compare offers by price and by performance. This is often done by using B-criterions in the tender documents or in more complex procurement processes by requiring concepts for the operation, service and further development of the software.

In a tender, a specific scope of services is typically offered for open-source solutions. Depending on the intended objectives of the procurement, in addition to performance requirements certain sustainability requirements can be incorporated into the tender as A or B criteria.

1. Criterion: Relationship with the Software Vendor/Community

Proposed wording as an A-Criterion

- If the relevant software is developed and maintained by a commercial software manufacturer, the provider can, among other things, demonstrate a contract with the software manufacturer and specify which services are covered by this contract (e.g. a close relationship with the software manufacturer through an existing premium partnership or similar).
- If the relevant software is developed and maintained by a community, the provider can, among other things, demonstrate that they:
    - Have already made contributions to the relevant software (successful merges, etc.)
    - Employ core contributors / lead developers of the relevant software within their company
    - Have contributed to publications on the relevant software (e.g. new developments, documentation, training materials, securing the supply chain, etc.)

Proposed wording as a B-Criterion

The provider outlines their relationship with the manufacturer or community. The evaluation will focus on access to technical support, software and security updates, insider knowledge on planned developments, and influence on the further development of the software.

Rationale: For software, a close connection with the manufacturer or the maintainer community is of paramount importance to ensure high standards of IT security and support.

Maximum 4000 characters. Weighting: 50 (i.e., 0 to 250 points)

Scoring Scale:

- **0 points:** No information or no qualifying information provided by the bidder
- **1 point:** Minimal relationship with the manufacturer or community without significant influence
- **2 points:** Basic relationship, but not significantly contributing to the improvement of the software
- **3 points:** Good relationship with the manufacturer or community with demonstrable influence on the software development
- **4 points:** Productive relationship with access to insider knowledge and direct support from core developers
- **5 points:** The provider has a contractual partnership with access to insider knowledge and direct support, or the provider employs one or more core developers of the open-source project

2. Criterion: Ensuring Upstream Publication of Customisations

Proposal for A-Criterion formulation

- The provider can demonstrate commit rights in the project and/or is a Certified Contributor
- The provider employs one or more core contributors/developers of the project
- The provider has an active partnership with the software manufacturer and significant influence on the software's development (e.g. submitting feature requests, participating in prioritising bug fixes, etc.)

Proposal for B-Criterion formulation

The provider outlines how commissioned customisations or adjustments during the project can be fed back upstream into the software code.

Rationale: The sustainability of the investment and the availability of extensions and security updates should be ensured for the client, even in future versions.

Max. 4000 characters. Weighting: 50 (=> 0 to 250 evaluation points)

Scoring scale:

- 0 points: No information or no qualifying information provided by the bidder

- 1 point: Bidder can generally demonstrate in-house developer staff
- 2 points: Bidder conceptually demonstrates how developments will be integrated upstream
- 3 points: Bidder demonstrates that they have already contributed relevant code to the project
- 4 points: Bidder can demonstrate commit rights in the project and/or is a Certified Contributor
- 5 points: Bidder employs one or more core developers of the project

3. Criterion: Ensuring High-Quality Third-Level Support

Proposal for A-Criterion formulation

- The provider can contractually ensure that level 3 support is provided by the respective manufacturer
- The provider can demonstrate that they employ developers with commit rights or core developers of the respective software
- The provider can demonstrate that they have regularly implemented bug fixes on the respective software in recent years (e.g. with a link or similar to the relevant fixes)

Proposal for B-Criterion formulation

The provider outlines how they will ensure level 3 support in accordance with the required service-level agreements (SLAs), either through a contract with the manufacturer or through their own qualified personnel.

Rationale: To secure the operation and security of the software, qualified level 3 support is essential. For open-source software, this can either be secured via a contract with the manufacturer or through qualified developers of the software.

Max. 4000 characters. Weighting: 50 (=> 0 to 250 evaluation points)

Scoring scale:

- 0 points: No information or no qualifying information provided by the bidder
- 1 point: Support staff are available, but there is a lack of detail regarding their qualifications
- 2 points: In-house personnel with demonstrated qualifications for troubleshooting, but the required SLAs are not sufficiently met
- 3 points: In-house personnel with demonstrated qualifications for troubleshooting. The required SLAs are covered
- 4 points: Contractually regulated manufacturer support is in place or the bidder employs one or more core developers of the project, but the required SLAs are not sufficiently met
- 5 points: Contractually regulated manufacturer support is in place or the bidder employs one or more core developers of the project. The required SLAs are covered

4. Criterion: Securing the Supply Chain through Support of Core Components

Proposal for A-Criterion formulation

The provider supports core components or projects that are part of their software solution in accordance with the Cyber Resilience Act (CRA):

- The provider significantly assists developers of core components in meeting the Cyber Resilience Act (CRA) requirements for securing the supply chain and documenting the software contained within the core components through a Software Bill of Materials (SBOM)
- The provider has established itself as a leading force with its contributions to a core component or an external project, particularly regarding software security improvements.
- The provider employs core developers of an external project or a core component that is part of the supply chain

Proposal for B-Criterion formulation

The provider outlines how they contribute to the promotion, development, and security of the core components integrated into their software.

Rationale: Software is typically developed using existing components (libraries, databases, frameworks, etc.), which form part of the supply chain and must meet both quality and security-related requirements. By supporting these core components, the provider actively contributes to securing the supply chain.

Scoring scale:

- 0 points: No evidence of substantive contributions or active support
- 1 point: Minimal supply chain support: Some evidence of contributions exists, but without significant depth or regularity
- 2 points: Adequate supply chain support: The provider has made some substantive contributions and participated in conferences related to the involved software projects, but their engagement remains sporadic or of limited impact
- 3 points: Good supply chain support: The provider demonstrates regular and relevant participation and contributes to conferences of the involved software projects
- 4 points: The provider is recognised within the relevant open-source community for their expertise and support of specific projects within the supply chain. For example, they assist in meeting the Cyber Resilience Act (CRA) requirements for securing the supply chain and documenting the components via a Software Bill of Materials (SBOM)
- 5 points: The provider employs core developers of an external project that is part of the supply chain and has established itself as a leading force through its contributions, particularly in improving security and driving innovation

About the Open Source Business Alliance - Federal Association for Digital Sovereignty e.V.

The Open Source Business Alliance (OSBA) is the federal trade association of the German open-source industry. The OSBA represents more than 230 companies which together generate more than 126 billion euros in annual revenues. The OSBA is cooperating with scientific institutions and software user groups in order to further public awareness for the importance of open-source software and open standards for a successful digital transformation. It is also their goal to drive open-source innovations and to establish open-source software as the standard in public procurement, in research funding and in the promotion of trade and industry. The OSBA is convinced that open-source software and open standards are the crucial prerequisite for digital sovereignty, for the capacity for innovation and for security in the age of digitalisation. As such, open-source software is the answer to one of the most pressing challenges of our time.

Copyright License

This paper is licensed under a Creative Commons 4.0 International License (CC BY SA). Use of the material is permitted as long as you attribute the material to the author (BY) and share adaptations of the work under a compatible license (SA).

Publisher: © 2025 Open Source Business Alliance – Bundesverband für digitale Souveränität e.V.

Authors: Procurement Working Group at the Open Source Business Alliance

English translation by Richard Brock and Lily Logua of Collabora Productivity

Title: Selection Criteria for the Sustainable Procurement of Open Source Software

License: CC BY SA 4.0 International