Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Cancelleria federale CaF

Trasformazione digitale e governance delle TIC TDT

14.09.2026

Em002-4 Guida alle community OSS

Versione 2.0-bcf38cc

Indice

> **NOTA** Il presente documento è una bozza in continua evoluzione e fa parte degli strumenti destinati a sostenere l'Amministrazione federale nella pubblicazione di codice open source. Per maggiori informazioni si veda il README principale.

1. L'essenziale in breve

Di seguito sono elencati i punti più importanti del documento.

- La costituzione o la gestione di una community OSS **non è richiesta** dall'art. 9 LMeCA.
- Una community è rilevante quando un gruppo di utenti ha senso e devono essere create **sinergie** con altri utenti del software.
- Una community non si limita al software stesso, ma può comprendere anche i processi circostanti e la standardizzazione dei dati.
- Per costituire una community occorre sempre un **onere iniziale**. Esso prosegue di regola nel tempo. In molti casi è opportuno che l'ufficio capofila sostenga anche i costi di coordinamento.
- Le community devono perseguire uno **scopo**. Occorrono interessati e partecipanti.
- Una community può nascere e poi essere formalizzata.
- L'obiettivo della community è accrescere il **beneficio per tutti**. Principio generale: «si ottiene almeno quanto si apporta».
- La community dovrebbe essere **strutturata nel modo più semplice possibile**.
- In un primo momento la community dovrebbe essere esaminata brevemente mediante la lista di controllo Em002-4.1. Soltanto se è opportuna dovrebbe essere sviluppata ulteriormente.
- La formalizzazione avviene mediante un **concetto di community**. Occorre riprendere il più possibile da fonti esistenti. Principio generale: «non reinventare la ruota».
- Se si mira a una collaborazione, ciò va considerato già all'inizio del progetto, poiché i potenziali partner dovrebbero essere coinvolti sin dall'analisi dei requisiti. Anche sotto il profilo giuridico e del diritto degli acquisti pubblici ciò va pianificato bene sin dall'inizio.
- Le possibilità di aderire a una collaborazione esistente vanno esaminate sotto il profilo giuridico e del diritto degli acquisti pubblici.
- Per i contributi non occorre costituire alcuna community. Se tuttavia il progetto lo richiede, va firmato il corrispondente **Contributor Licence Agreement (CLA)** oppure gli sviluppatori devono essere autorizzati dal progetto a presentare il software sotto un Developer Certificate of Origin (DCO).

2. Introduzione

La presente guida si rivolge agli specialisti TIC responsabili di un'applicazione che intendono offrirla come open source o contribuire a una simile applicazione.

Ciò va organizzato in modo formale o informale.

Il presente documento fornisce informazioni importanti per l'organizzazione e la costituzione di una community[1].] nonché per lo sviluppo aperto direttamente come progetto open source.

Quale secondo elemento, per promuovere la standardizzazione, nell'ambito del presente documento è tenuto un repertorio di concetti di community esistenti. L'idea è che nuove community possano essere costituite con un onere minimo sulla base di soluzioni collaudate.

Il presente documento non contiene raccomandazioni dirette per l'organizzazione di community attorno a librerie software (parti di software che svolgono funzioni parziali, p. es. la generazione di PDF). Normalmente per esse non viene elaborato alcun concetto proprio; si utilizza invece il modello semplice definito nel modello di archiviazione del repository.

Il documento Em002-01 Guida pratica per il software open source nell'Amministrazione federale contiene, nel capitolo 4, le forme fondamentali di collaborazione, nel capitolo 8 i diversi modelli di supporto e, nell'allegato, informazioni sul comportamento di mercato dell'open source.

3. Tipo, obiettivo e scopo di una community

Una community può perseguire i seguenti scopi:

- Rilevamento di requisiti supplementari
- Diffusione di conoscenze sull'applicazione
- Migliori riscontri dagli utenti
- Promozione di traduzioni, documentazione e formazione
- Diffusione della standardizzazione
- Interazione complessivamente migliore con gli utenti
- Estensione della cooperazione ad altre parti
- Promozione di ulteriore software basato sul software

Quale tipo di community sia più adatto a un'applicazione dipende fortemente dal contesto specialistico, dai potenziali utenti e dalla strategia dell'istituzione che pubblica. Vi sono molti modi

1. La community di un progetto open source rappresenta tipicamente una piramide. + La base di questa piramide è costituita dagli utenti del software, in particolare da quelli impegnati che partecipano attivamente alla community, per esempio sotto forma di segnalazioni di errori e richieste di funzionalità o mediante contributi a mailing list. + Direttamente sopra, nella piramide, si trovano i contributori. Si tratta di parti della community che propongono propri contributi di codice. Essi tipicamente non hanno accesso in scrittura al repository. I loro contributi sono esaminati dai manutentori del progetto e integrati nel repository dopo aver raggiunto la qualità tipica del progetto. + Al vertice della piramide si trovano i manutentori o committer di un progetto. In tale ruolo uno sviluppatore assume una responsabilità maggiore. Ciò si manifesta per esempio nella possibilità di accettare contributi. Tale diritto si concretizza spesso nell'accesso in scrittura al repository. A questo livello si esercita il controllo sul software, sulla sua qualità e sulla sua gamma di funzioni. Nei progetti più complessi tale livello può essere ulteriormente suddiviso. Il kernel Linux, per esempio, conta notoriamente manutentori di sottosistemi su più livelli e, in definitiva, un solo sviluppatore, Linus Torvalds, integra le patch nel repository del progetto. Un altro esempio è dato dai progetti della Eclipse Foundation, che dispongono tipicamente di un project lead con diritti supplementari nell'ambito del processo di sviluppo della Eclipse Foundation, quali l'avvio del processo di rilascio o l'elezione formale di committer o project lead *[BITKOM2023*

diversi di costituire community. Idealmente, un progetto adeguato può aderire a una community esistente. In alternativa, una community può essere costituita con uno o più partner con pari diritti.

È importante che venga stabilita una **struttura di governance** e che i punti determinanti siano documentati in un **concetto di community** (cfr. anche *[BITKOM2023]* capitolo 4.1 e *[IZqCab2023]*).

Va rilevato che le community per il software possono anche essere combinate con quelle per i dati, la standardizzazione e i temi specialistici. La community può allora comprendere processi, standardizzazione, software e dati. Strutture esistenti quali eOperations, ADS ed eCH possono parimenti essere utilizzate ove opportuno.

Un aspetto importante di una community funzionante è che i partecipanti ricevano più di quanto investono.

Esistono le seguenti costellazioni:

- **Community attorno a un OSS della Confederazione**: la governance e il concetto di community vanno elaborati. In tale contesto occorre anche disciplinare come terzi possano apportare contributi.
- **Nuova collaborazione per risolvere un problema**: oltre alla governance e al concetto di community va costituita l'intera struttura giuridica della collaborazione.
- **Adesione a una collaborazione**: l'Amministrazione federale deve esaminare se e come può farlo.
- **Contributo a un progetto di terzi**: va verificata la politica del progetto in materia di contributi (Contributor Licence Agreement, Developer Certificate of Origin ecc.).

Le questioni di diritto degli acquisti pubblici sono trattate in Em002-7 Aspetti strategici degli acquisti e del software open source.

3.1. Caso particolare: nuova collaborazione per lo sviluppo, la manutenzione e il supporto di software open source

I costi complessivi del software open source diminuiscono quando esso è sviluppato e mantenuto congiuntamente. Ciò può avvenire senza accordi di cooperazione formali, o con nulla più di una pianificazione comune.

Se si mira a una collaborazione più formale, andrebbe svolta un'analisi di mercato per assicurarsi che la collaborazione sia la forma di cooperazione più adeguata. Le collaborazioni sono più economiche dello sviluppo individuale da parte di ciascuna parte (cfr. Em002-4.1).

Va rilevato che per le collaborazioni è opportuno avviare la collaborazione molto presto nella metodologia di progetto HERMES[2] — concretamente durante le fasi di analisi della situazione e dei requisiti.

La collaborazione è in linea di principio ammessa in Svizzera secondo l'articolo 4 LMeCA.

Oltre a tutte le altre considerazioni, le collaborazioni presentano sia una dimensione giuridica sia una dimensione di diritto degli acquisti pubblici. Attualmente ogni caso è valutato singolarmente. Con il tempo si affermeranno modelli standardizzati.

3.1.1. Dimensione giuridica

Le adesioni ad associazioni sono praticabili, anche se sono di regola valutate caso per caso: chi sta dietro l'associazione e quali obblighi vengono assunti? Una delega delle adesioni a eOperations o a

2. https://www.hermes.admin.ch/it/gestione-di-progetti/panoramica-del-metodo.html

organizzazioni analoghe sarebbe possibile.

Fintanto che non circola denaro e non devono essere assunti obblighi formali, è possibile un Memorandum of Understanding a livello di ufficio federale o di Cancelleria federale.

Una situazione che richieda la conclusione di un trattato internazionale andrebbe evitata.

Esistono inoltre due casi particolari[3] in cui una collaborazione sarebbe già coperta:

- Le commesse aggiudicate in virtù di un accordo internazionale concernente la realizzazione congiunta di un progetto da parte di Stati firmatari.
- Le commesse aggiudicate conformemente alle procedure o condizioni particolari di un'organizzazione internazionale.

3.1.2. Dimensione di diritto degli acquisti pubblici

In molti casi la collaborazione avviene tra organizzazioni del settore pubblico. In tali casi l'acquisto non pone problemi se la configurazione è «intrastatale» (cfr. Em002-7 Aspetti strategici degli acquisti e del software open source).

Contratti con imprese estere non domiciliate in Svizzera sono possibili, purché siano soddisfatte le esigenze del diritto degli acquisti pubblici.

L'acquisto delle risorse proprie di un'organizzazione avviene nel quadro di un processo d'acquisto ordinario, oppure è già concluso.

3.2. Caso particolare: adesione a una collaborazione esistente

Ciò richiede sia una base legale sia il rispetto del diritto degli acquisti pubblici.

L'approccio più semplice è la partecipazione secondo il principio «ciascuna parte sostiene i propri costi». Ciò può coprire anche le spese generali, purché siano impiegate risorse interne oppure risorse esterne correttamente messe a concorso.

Una partecipazione diretta a collaborazioni estere è più difficile. Una semplice aggregazione mediante un Memorandum of Understanding è l'opzione più facile da realizzare.

3.3. Caso particolare: contributi di codice a progetti OSS di terzi

Quando l'Amministrazione federale apporta modifiche al codice di un progetto OSS esistente, è opportuno contribuire tali modifiche direttamente a monte al progetto originario. In tal modo la manutenzione corrente non deve più essere assicurata dall'Amministrazione federale.

Secondo l'articolo 9 LMeCA tali contributi sono anzi auspicati. Sotto il profilo del diritto degli acquisti pubblici ciò non pone problemi.

È importante che sia il titolare dei diritti ad apportare il contributo. Ciò significa che l'Amministrazione federale se ne assume la responsabilità per conto dei propri collaboratori e mandatari. Concretamente va firmato l'eventuale Contributor Licence Agreement del progetto, oppure va allegato alla pull request un Developer Certificate of Origin.

Un **Developer Certificate of Origin*[4] *(DCO)** per l'Amministrazione federale potrebbe recitare:

3. https://www.eda.admin.ch/deza/it/home/partnerschaften/auftraege-beitraege/auftraege/anforderungen/rechtlich.html
4. https://developercertificate.org/

> Copyright © 2004, 2006 The Linux Foundation and its contributors.
>
> Everyone is permitted to copy and distribute verbatim copies of this licence document, but changing it is not allowed.
>
> Developer’s Certificate of Origin 1.1
>
> By making a contribution to this project, we certify that:
>
> (a) The rights to this contribution are owned by the Swiss Confederation and I am authorised to submit it under the open source licence indicated in the file; or
>
> (b) The contribution is based upon previous work that, to the best of my knowledge, is covered under an appropriate open source licence and I have the right under that licence to submit that work with modifications, whether created in whole or in part by me, under the same open source license (unless I am permitted to submit under a different licence), as indicated in the file; or
>
> (c) The contribution was provided directly to me by some other person who certified (a), (b) or (c) and I have not modified it.
>
> (d) I understand and agree that this project and the contribution are public and that a record of the contribution (including all personal information I submit with it, including my sign-off) is maintained indefinitely and may be redistributed consistent with this project or the open source licence(s) involved.
>
> *— Developer Certificate of Origin*
> *Version 1.1*

La struttura di un Contributor Licence Agreement è descritta in Em002-3 Guida alle licenze OSS, capitolo 8.9.

Il trasferimento effettivo del codice avviene conformemente alle direttive del rispettivo progetto.

4. Concetto di community

Il nucleo di una community è in definitiva la sua struttura e la sua governance. Queste sono fissate in un concetto di community.

Il concetto di community, che va elaborato dalla direzione del progetto o dell’applicazione (o commissionato a terzi), dovrebbe seguire la seguente struttura di capitoli. A seconda della natura della community concreta non tutti i sottocapitoli sono necessari. Idealmente molti capitoli rinvieranno a documenti già standardizzati o utilizzeranno processi standard dei repository corrispondenti. Il concetto di community può essere applicato anche a progetti esistenti diretti da altri, in particolare se la governance effettiva non è ancora documentata per iscritto.

Formalizzando la community si riuniscono tutti i partecipanti e si semplifica l’inserimento di nuovi partecipanti.

Bozza di struttura di un concetto di community:

1. Tabella riassuntiva (nome, repository, indirizzo di contatto, licenza, livello di supporto, esistente/nuovo)
2. Obiettivi
    a. dell’applicazione
    b. della community
3. Organizzazione
    a. Proprietario del codice
    b. Forma organizzativa / organi
    c. Diritti di voto
    d. Direzione del progetto
    e. Questioni d’acquisto (se del caso)
    f. Ulteriori aspetti di governance
4. Roadmap e processo di modifica
5. Processo di sviluppo
    a. Sviluppo aperto
    b. Diritti di committer
    c. Processo di revisione
    d. Ripartizione dei compiti e menzioni nominative
    e. Backup
6. Finanziamento della community
7. Marketing
8. Trattamento dei contributi esterni
    a. Trattamento degli errori segnalati
    b. Trattamento delle richieste
    c. Trattamento delle pull request
    d. Trattamento dei fork
    e. CLA, DCO e attribuzione di diritti / proprietà intellettuale
9. Esercizio dell’applicazione

In linea di principio ogni autorità federale è responsabile dell’elaborazione dei concetti di community. Il settore TDI può pubblicare modelli ed esempi esistenti e accoglie esempi corrispondenti in vista della pubblicazione. È inoltre opportuno che i responsabili di community si scambino informazioni tra loro.

Una possibile fonte per alcune parti è anche la documentazione della Eclipse Foundation,[5] fermo restando che qui va effettuato un adeguato adattamento nello spirito del «fare solo il necessario».

5. Decisioni di principio nella costituzione di una community

I principi della community, o la rinuncia a essa, sono stabiliti mediante Em002-4.1. Le questioni di principio determinanti figurano nella scatola morfologica seguente; ogni dimensione è illustrata nelle sezioni successive.

Dimensione	(a)	(b)	(c)	(d)	(e)
A — Gestione del prodotto	Confederazione	Organizzazione comune	Organizzazione aperta	Terzi	
B — Coinvolgimento dei fornitori	Mandatario	Partner	Indipendente	Direzione	
C — Ripartizione dei costi	Confederazione	Ciascuno per sé	Principio di causalità	Chiave di ripartizione	Contributo
D — Supporto	Pubblicazione minima	Supporto da parte della Confederazione	Supporto da parte di terzi		
E — Sviluppo	Interno	Terzi	Aperto		

Scatola morfologica community OSS.

È consigliabile, nel configurare la community, **concentrarsi in primo luogo sul prossimo futuro**; se per esempio non vi sono ancora interessati concreti all'applicazione, di regola ha poco senso definire una community con una società semplice, una propria associazione e processi complessi per l'elaborazione della roadmap.[6][7]

5. Cfr. https://www.eclipse.org/projects/handbook/#starting
6. Con un'associazione (o una fondazione) si crea una persona giuridica distinta che si occupa del software e del prodotto. Ciò si giustifica soltanto a partire da un certo livello di importanza, complessità e in presenza di più partner (con pari diritti). Una roadmap serve a mostrare al pubblico più ampio e ai potenziali ulteriori utenti del software che cosa è previsto.
7. Può anche essere possibile, in particolare per progetti che coinvolgono sin dall'inizio più attori statali, rivolgersi a fondazioni esistenti e verificare se i progetti e la governance possano essere condotti sotto la loro egida, analogamente a https://finos.org o https://osr.finos.org o https://www.eclipse.org/collaborations/ o https://iot.eclipse.org o https://outreach.eclipse.foundation/open-regulatory-compliance.

5.1. Gestione del prodotto

A seconda del tipo di applicazione, della sua importanza strategica e della sua posizione nel ciclo di vita, esistono diverse varianti per rappresentare la gestione del prodotto. Per le organizzazioni in seno alle autorità federali ciò riguarda sempre i responsabili dell'applicazione. Per il concetto di community sono particolarmente rilevanti la definizione della roadmap funzionale e tecnologica nonché i processi di rilascio.

Possibilità:

a. **Organizzazioni in seno alla Confederazione**: la Confederazione vuole mantenere il controllo; definisce da sola la roadmap.

b. **Organizzazione comune**: insieme ad altri utenti o fornitori.

c. **Organizzazione aperta**: legami lassi, nessuna roadmap attiva.

d. **Terzi**: o il prodotto esiste già, oppure il lavoro deve essere esternalizzato.

5.1.1. Organizzazioni in seno alla Confederazione:

Se l'applicazione ha grande importanza strategica o se la Confederazione dipende dalla possibilità di attuare rapidamente le proprie modifiche, può essere opportuno mantenere il controllo.

L'autorità federale decide da sola, tramite la responsabilità dei requisiti, che cosa confluisce nel software e quando.

Per i potenziali altri utenti dell'applicazione ciò significa che sono praticamente costretti a utilizzare una propria versione (fork) dell'applicazione.

In caso contrario hanno ben poche possibilità di attuare adattamenti ed estensioni che possono essere per loro importanti. Ciò non depone contro una pubblicazione come open source: gli strumenti di gestione del codice[8] offrono un ampio supporto per simili scenari, e adattamenti e correzioni di errori possono essere ripresi selettivamente in entrambe le direzioni.

Una ripartizione dei costi è tipicamente difficile con questo modello.

5.1.2. Comune (la Confederazione ha la direzione o partner con pari diritti)

Le decisioni su roadmap e priorità dovrebbero essere coordinate e prese congiuntamente con gli altri membri della community. La Confederazione partecipa quale membro della community, ma non è l'organizzazione capofila.

Affinché la decisione comune funzioni, strutture e regole devono essere stabilite preventivamente. Ciò definisce anche come i costi di attuazione degli adattamenti decisi congiuntamente sono ripartiti tra le parti.

Con questo modello si crea un'unica versione dell'applicazione, utilizzata poi da tutti i membri. I costi complessivi risultano pertanto nettamente inferiori rispetto alle altre varianti. La flessibilità dei singoli utenti è tuttavia minore ed essi devono coordinare preventivamente le proprie richieste di adattamento con tutti gli altri. La decisione è più lenta e più laboriosa.

Questo modello è particolarmente adatto ad applicazioni da sviluppare congiuntamente con altre organizzazioni chiaramente definite (p. es. i Cantoni).

8. p. es. GitHub

5.1.3. Organizzazione aperta

In questo modello i processi e le strutture sono definiti soltanto in modo lasso. La Confederazione mantiene formalmente il controllo, ma è aperta a contributi e adattamenti.

Non viene definita alcuna roadmap, o soltanto una molto sommaria; il consenso è raggiunto informalmente e in discussioni direttamente sulla piattaforma di pubblicazione. La community stessa è piuttosto dinamica; i membri possono essere molto attivi per un certo periodo e poi ritirarsi.

Questo modello è adatto anche ad applicazioni più piccole già in uso produttivo e «concluse». È inoltre di regola la forma migliore per applicazioni o componenti a orientamento tecnico che si rivolgono potenzialmente a un pubblico più ampio non definibile con precisione in anticipo.

5.1.4. Terzi

In questo modello i processi e le strutture sono definiti da terzi e l'autorità federale è membro della community senza averne la direzione. Ciò può derivare dal fatto che il progetto è stato per esempio definito da un altro Stato o Cantone. Oppure può darsi che la Confederazione non sia interessata a proseguire essa stessa il programma e trasferisca tale compito.

5.2. Coinvolgimento dei fornitori

Per il coinvolgimento dei fornitori esistono in linea di principio le seguenti possibilità:

a. **Fornitore quale mandatario**: è in linea di principio considerato sostituibile a medio termine. Il fornitore deve sviluppare l'applicazione in modo economico e con alta qualità sulla base delle commesse della gestione del prodotto (o della community), ma ha al massimo un'influenza consultiva su roadmap e priorità.

b. **Partner a lungo termine con codecisione**: devono poter codeterminare attivamente lo sviluppo. Ciò rafforza il loro ruolo ed essi sono potenzialmente disposti a investire essi stessi nell'applicazione. Tipicamente, in questo modello, i fornitori distribuiscono il software e cercano attivamente nuovi clienti.

c. **Indipendente:** i fornitori stabiliscono da sé se e come intendono lavorare al progetto. Ciò può accadere per esempio se più fornitori intendono generare attività proprie sulla base del software. Licenze liberali possono potenzialmente favorirlo. Può accadere anche se la strategia della Confederazione diverge da quella del fornitore. In tale scenario è probabile che il fornitore crei un fork.

d. **Direzione**: se il fornitore assume la gestione del prodotto, ne ha la direzione. Lo sviluppo è allora determinato da lui. D'altro canto ciò può anche significare che l'Amministrazione federale deve assumersi ben poca responsabilità. Con una pubblicazione minima è possibile che il fornitore assuma tale ruolo di propria iniziativa.

Se il fornitore assume un ruolo forte, ciò comporta vantaggi per l'Amministrazione federale: mediante una commercializzazione attiva dell'applicazione vi è la possibilità che la community cresca più rapidamente e più fortemente, riducendo così più velocemente i costi per membro.

Se si sceglie un modello con forte codecisione dei fornitori, occorre vigilare affinché non ne derivi una deroga alle prescrizioni del diritto degli acquisti pubblici. Di regola è consigliabile coinvolgere almeno due fornitori per non creare una dipendenza troppo forte da uno solo. Dovrebbe inoltre sussistere la possibilità che altri fornitori aderiscano alla community.

5.3. Ripartizione dei costi

Per ripartire i costi di manutenzione, sviluppo ulteriore e nuove funzionalità sono disponibili le seguenti opzioni:

a. **Organizzazioni in seno alla Confederazione**: in particolare nella costituzione di ecosistemi o se l’autorità federale mantiene il controllo totale, essa deve anche sostenerne i costi.

b. **Ciascuno per sé**: ognuno paga da sé le modifiche che desidera. Se necessario si decide caso per caso di finanziare congiuntamente determinate estensioni. I costi ricorrenti sono fatturati secondo una chiave di ripartizione o a richiesta.

c. **Principio di causalità**: chi ha effettivamente bisogno delle prestazioni o funzionalità le paga. Possono essere applicate tariffe orarie standardizzate.

d. **Chiave di ripartizione**: i costi sono ripartiti all’interno della community secondo una chiave definita. Soltanto le estensioni specifiche vi derogano. Queste dovrebbero essere pagate direttamente dagli utenti interessati. Il divisore dei costi può essere per esempio la dimensione del Cantone interessato o il numero di utenti. Riguardo alla chiave di ripartizione cfr. anche il capitolo «Ripartizione dei costi».

e. **Contributo**: quando l’autorità federale apporta soltanto personale a un progetto esistente.

Fondamentalmente la variante b (ripartizione dei costi) ha di regola senso soltanto se esiste anche una gestione comune del prodotto come nella variante b (comune). Soltanto chi può avere voce in capitolo sarà disposto a sostenere i costi conseguenti alle decisioni.

5.4. Supporto

Nell’ambito della ripartizione dei costi e dell’onere occorre chiarire chi fornisce quale supporto. Secondo l’art. 9 LMeCA è possibile una pura pubblicazione minima. Tale variante può non produrre l’effetto desiderato e non costituisce di per sé una community in cui l’autorità federale abbia influenza.

a. **Pubblicazione minima**: nessun supporto (ossia la pubblicazione avviene di regola senza supporto organizzato e senza partecipazione dell’autorità federale).

b. **Supporto da parte della Confederazione**: l’autorità federale (o il suo fornitore di prestazioni) fornisce un supporto definito.

c. **Supporto da parte di terzi**: l’autorità federale o la community definisce un’organizzazione di supporto.

L’effetto da conseguire, p. es. sull’ecosistema svizzero o un finanziamento iniziale di una community, può indurre la Confederazione a fornire supporto. L’entità del supporto va naturalmente definita. In linea di principio può anche essere riscosso un emolumento. Per via dei meccanismi della Confederazione, i fornitori di prestazioni o i fornitori sono più verosimilmente in grado di offrire un supporto commerciale.

Gli uffici possono offrire supporto a pagamento conformemente alla LMeCA (eventualmente anche tramite fornitori di prestazioni e terzi). A tal fine devono pubblicare i relativi emolumenti sul proprio sito web, poiché la necessaria base legale sussiste. Il collegamento al sistema di pagamento può essere discusso con il DFF.

Esempi di supporto ed emolumenti figurano nell’allegato G.

5.5. Sviluppo

Anche la metodologia fondamentale dello sviluppo può influire sul tipo di community.

a. **Interno**: l’utilizzo di repository interni significa che terzi non possono collaborare direttamente. Più rigorosa è la separazione, maggiore è l’onere di integrazione che la Confederazione deve sostenere se intende riprendere estensioni di terzi. Anche il processo di pubblicazione è più complesso.

b. **Terzi**: o un fornitore lavora per conto proprio, oppure il prodotto già avviato o diretto da terzi è sviluppato secondo regole definite.

c. **Aperto**: lo sviluppo software avviene direttamente in un repository pubblico. La pubblicazione secondo la LMeCA è così ampiamente garantita. Gli oneri si producono durante lo sviluppo. In tale scenario la collaborazione con terzi può avvenire anche prima (e più facilmente).

6. Contenuti dettagliati per il concetto di community

Il presente capitolo fornisce indicazioni per la redazione del concetto di community sulla base delle decisioni di principio prese (secondo il capitolo 5).

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

aperto / Partner / Confederazione
Contenuti o suggerimenti, se le decisioni di principio sono:
Gestione del prodotto c) Organizzazione aperta
Coinvolgimento dei fornitori: b) Fornitore quale partner
Ripartizione dei costi: b) Ciascuno per sé
Il testo di cui sopra può essere copiato nel modello e adattato.

Comune / - / Ciascuno per sé

Comune / - / Causante
Contenuti o suggerimenti, se le decisioni di principio sono:
Gestione del prodotto b) Sviluppo comune
Coinvolgimento dei fornitori: a) o b) (non rilevante)
Ripartizione dei costi: c) Principio di causalità o d) Ripartizione dei costi

Il concetto di community dovrebbe essere un documento dinamico, adattabile d’intesa con la community. Il concetto dovrebbe rispecchiare la situazione attuale e proporre opzioni di sviluppo future. Se in seguito aderiscono altri membri alla community, l’organizzazione e i processi possono essere ulteriormente elaborati e possono essere introdotti meccanismi più complessi. I sottocapitoli seguenti seguono direttamente la struttura proposta del concetto di community (cfr. capitolo 4).

Se la gestione del prodotto spetta a terzi, questi definiscono i concetti per la community.

6.1. Dati essenziali

I seguenti dati essenziali dovrebbero essere elaborati e pubblicati per ogni community, affinché i potenziali interessati possano registrarsi per il software e la community:

- Nome del software
- Repository
- Proprietario
- Unità organizzativa / ufficio responsabile
- Indirizzo di contatto
- Licenza utilizzata
- Livello di supporto
- Progetto esistente

Queste indicazioni possono essere tutte rappresentate con publiccode.yml.[9]

6.2. Obiettivi

L'obiettivo dovrebbe contenere lo scopo effettivo o la «visione» dell'applicazione. Tale visione dovrebbe figurare anche nel file **README.md** del repository (cfr. lista di controllo Em002-2.2 Analisi e preparazione OSS).

Al tempo stesso andrebbe descritto che cosa persegue la community:

- Chi deve aderire?
- Chi sono i potenziali interessati?
- Vanno ricercati attivamente nuovi membri?
- Quali sono gli obiettivi principali perseguiti con la pubblicazione come open source?

6.3. Organizzazione e governance

In linea di principio, per i nuovi sviluppi si mira a che la Confederazione sia titolare dei diritti principali sul codice sorgente.

Il tipo di conduzione del progetto dipende da quale organizzazione aveva la direzione (SAFe, HERMES) in seno alla Confederazione. Per la forma organizzativa va sempre perseguito un onere amministrativo minimo (esistente anziché nuovo; società semplice anziché associazione).

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Confederazione / - / Interno
: Poiché la Confederazione mantiene il controllo diretto, non sono necessarie strutture organizzative supplementari.
L'ente che pubblica (p. es. un ufficio federale) resta proprietario e assume tutti i compiti di coordinamento. In tal caso il capitolo «Organizzazione» può essere tenuto molto breve.

Comune / - / Interno
: Per questo scenario si raccomanda di regola l'organizzazione quale **società semplice**.
Affinché lo sviluppo comune e il coordinamento di roadmap e requisiti siano possibili, è consigliabile costituire due organi, ciascuno con un partecipante per membro della community:

  - Organo direttivo: definisce strategia e priorità.
  - Gruppo specialistico: elabora requisiti e contenuti concreti a livello tecnico. A seconda dell'ampiezza può essere istituito anche un gruppo specialistico per ogni tema.

  Proprietario del codice è l'ufficio che lo ha pubblicato.

  Se le strutture diventano più complesse o se sono coinvolti più di cinque utenti o membri supplementari, può valere la pena esaminare se la community non sia meglio organizzata come associazione. Si confrontino le spiegazioni della variante immediatamente sottostante.

Comune / - / Terzi
: Si raccomanda l'organizzazione quale **associazione**.
O viene fondata un'associazione appositamente per l'applicazione, oppure l'applicazione viene trasferita a un'associazione esistente. Un'associazione è raccomandata perché altrimenti la

9. https://yml.publiccode.tools/

ripartizione dei costi diventa complessa e i compiti di coordinamento comportano verosimilmente lavoro sufficiente per assumere un gerente a tempo parziale per l'associazione.

I diritti sul codice sono allora trasferiti all'associazione, che assume la gestione della community e le ordinazioni presso i fornitori.

Comune / Mandatario / Interno

Si raccomanda l'organizzazione quale **associazione**. O viene fondata un'associazione appositamente per l'applicazione, oppure l'applicazione viene trasferita a un'associazione esistente. Un'associazione è raccomandata perché altrimenti la ripartizione dei costi diventa complessa e i compiti di coordinamento comportano verosimilmente lavoro sufficiente per assumere un gerente a tempo parziale per l'associazione.

I diritti sul codice sono allora trasferiti all'associazione, che assume la gestione della community e le ordinazioni presso i fornitori.
Qui i fornitori non sono membri dell'associazione, bensì soltanto mandatari. Il gerente non dovrebbe essere fornito da un fornitore.

Comune / Mandatario / Terzi

Si raccomanda l'organizzazione quale **associazione**. O viene fondata un'associazione appositamente per l'applicazione, oppure l'applicazione viene trasferita a un'associazione esistente. Un'associazione è raccomandata perché altrimenti la ripartizione dei costi diventa complessa e i compiti di coordinamento comportano verosimilmente lavoro sufficiente per assumere un gerente a tempo parziale per l'associazione.
I diritti sul codice sono allora trasferiti all'associazione, che assume la gestione della community e le ordinazioni presso i fornitori.
I fornitori sono membri dell'associazione; il gerente può anche essere fornito da un fornitore.

aperto / - / Interno

Non vi è **alcuna necessità di una descrizione dettagliata di un'organizzazione**, poiché essa è deliberatamente concepita in modo aperto. È tuttavia importante stabilire chi riceve e tratta le richieste esterne (responsabilità). Va inoltre determinato se e in quale misura persone esterne possono essere coinvolte:

- Nessuna integrazione diretta: gli esterni possono soltanto presentare suggerimenti e contributi (come pull request).
- Integrazione quali contributori: gli esterni che apportano contributi frequenti e di qualità sono integrati direttamente.
- Trasferimento a esterni possibile: se, dopo la pubblicazione, gli esterni contribuiscono allo sviluppo più dell'organizzazione che pubblica, l'applicazione può anche essere loro trasferita.

L'organizzazione che pubblica resta proprietaria. Questa variante corrisponde all'organizzazione dei progetti open source «classici»; cfr. per esempio https://opensource.guide/leadership-and-governance/ per maggiori informazioni.

aperto / Mandatario / Interno

Non vi è **alcuna necessità di una descrizione dettagliata di un'organizzazione**, poiché essa è deliberatamente concepita in modo aperto. È tuttavia importante stabilire chi riceve e tratta le richieste esterne (responsabilità). Va inoltre determinato se e in quale misura persone esterne possono essere coinvolte:

- Nessuna integrazione diretta: gli esterni possono soltanto presentare suggerimenti e contributi (come pull request).
- Integrazione quali contributori: gli esterni che apportano contributi frequenti e di qualità sono integrati direttamente.

- Trasferimento a esterni possibile: se, dopo la pubblicazione, gli esterni contribuiscono allo sviluppo più dell'organizzazione che pubblica, l'applicazione può anche essere loro trasferita.

L'organizzazione che pubblica resta proprietaria.
Questa variante corrisponde all'organizzazione dei progetti open source «classici»; cfr. per esempio https://opensource.guide/leadership-and-governance/ per maggiori informazioni.
Il fornitore dovrebbe assumere soltanto compiti di coordinamento puramente tecnici (code review, valutazione).

Terzi / Partner / Interno
Non vi è **alcuna necessità di una descrizione dettagliata di un'organizzazione**, poiché essa è deliberatamente concepita in modo aperto. È tuttavia importante stabilire chi riceve e tratta le richieste esterne (responsabilità). Va inoltre determinato se e in quale misura persone esterne possono essere coinvolte:

- Nessuna integrazione diretta: gli esterni possono soltanto presentare suggerimenti e contributi (come pull request).
- Integrazione quali contributori: gli esterni che apportano contributi frequenti e di qualità sono integrati direttamente.
- Trasferimento a esterni possibile: se, dopo la pubblicazione, gli esterni contribuiscono allo sviluppo più dell'organizzazione che pubblica, l'applicazione può anche essere loro trasferita.

L'organizzazione che pubblica resta proprietaria. Questa variante corrisponde all'organizzazione dei progetti open source «classici»; cfr. per esempio https://opensource.guide/leadership-and-governance/ per maggiori informazioni.

Riguardo alla governance possono essere consultati anche *[BITKOM2023]* capitolo 4.1, *[IZqCab2023]* o https://github.com/todogroup/ospo-career-path/blob/main/OSPO-101/module7/README.md#governance-models.

Esempi di statuti associativi figurano nell'allegato E.

Nota: poiché la pubblicazione deve essere garantita conformemente all'art. 9 LMeCA, l'Amministrazione federale deve assicurare che essa non possa essere improvvisamente ritirata (cfr. le «direttive Open Source negli acquisti — acquisto di software secondo l'art. 9 LMeCA» *[BBL-WL]*, numero V.2).

6.4. Roadmap e processo di modifica

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Confederazione / Mandatario / -
L'autorità federale definisce la roadmap e decide quali modifiche vengono attuate. A tal fine sono utilizzati i processi standard dell'ufficio XY.

Confederazione / Partner / -
L'autorità federale definisce la roadmap in collaborazione con l'impresa XY. Essa decide quali modifiche vengono attuate. A tal fine sono utilizzati i processi standard dell'ufficio XY.

Comune / - / Interno
Il processo di elaborazione della roadmap e di approvazione delle modifiche va stabilito dai membri della società o dell'associazione. A seconda della struttura dei membri (numero, rapporto dimensionale ecc.) possono essere adeguati altri processi.
I processi devono tuttavia prevedere misure per la risoluzione dei conflitti e la formazione della maggioranza.

Comune / - / Terzi
Il processo di elaborazione della roadmap e di approvazione delle modifiche va stabilito dai membri della società o dell'associazione. A seconda della struttura dei membri (numero, rapporto dimensionale ecc.) possono essere adeguati altri processi.

I processi devono tuttavia prevedere misure per la risoluzione dei conflitti e la formazione della maggioranza.

Con l'aggiunta che va stabilita anche una chiave di ripartizione dei costi. Occorre inoltre riflettere su come trattare la ripartizione dei costi per adattamenti non desiderati dai membri interessati. In particolare nelle associazioni con membri di dimensioni disuguali possono verificarsi situazioni in cui un grande membro (p. es. l'autorità federale) sostiene una parte rilevante dei costi di un'estensione senza trarne alcun beneficio.

aperto / - / -
Esempio, non adatto a tutte le situazioni
Non occorre elaborare una roadmap; l'applicazione soddisfa attualmente le esigenze.

Le modifiche sono decise in una discussione aperta e si ricerca un consenso. La decisione finale spetta al proprietario del codice (Confederazione Svizzera).

6.5. Processo di sviluppo

Se entra in gioco lo sviluppo aperto, esso va definito qui. Sono importanti la ripartizione dei compiti e l'indicazione delle posizioni centrali nel concetto di community. Nonché il modo in cui è garantita la sicurezza dei contenuti del repository.

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Confederazione / - / -
I diritti di committer (accesso in scrittura al codice) spettano alle persone/agli enti/ai fornitori selezionati dalla Confederazione. Alla scadenza del contratto i rispettivi diritti sono revocati.

I fornitori di prestazioni e i fornitori sono responsabili dello svolgimento del processo tecnico di code review per i contributi esterni accettati nel merito dalla Confederazione. Essi rispondono della qualità tecnica. I fornitori sono indennizzati per tale lavoro dalla Confederazione.

Comune / Mandatario / Interno
In linea di principio tutti gli enti incaricati da un membro della community ricevono l'accesso in scrittura al codice (diritti di committer). Alla scadenza del contratto i rispettivi diritti sono revocati.

Se il codice modificato riguarda settori centrali dell'applicazione, tutte le modifiche devono essere esaminate da un ente appositamente incaricato dalla community. Tale ente risponde della qualità tecnica dell'applicazione. Esso è inoltre competente per l'esame delle modifiche accettate nel merito dalla community.

Comune / Partner / Terzi
I diritti di committer (accesso in scrittura al codice) spettano ai membri della community. Tali persone si organizzano da sé e assumono congiuntamente la responsabilità della qualità del codice.

Se un membro della community incarica un fornitore esterno di un adattamento o di un'estensione, l'adattamento è esaminato da un membro della community. Esso è indennizzato a tal fine dal committente.

Le modifiche al nucleo dell'applicazione devono essere esaminate da un secondo membro della community.

I fornitori membri della community sono inoltre responsabili dell’esame dei contributi esterni e del nuovo codice tecnico accettato nel merito dall’associazione.

aperto / - / -

Inizialmente i diritti di committer spettano agli sviluppatori o ai loro datori di lavoro. I diritti di committer sono concessi a tutti gli sviluppatori che contribuiscono attivamente al progetto per più mesi e dimostrano un’adeguata competenza sociale.

I singoli sviluppatori conservano in linea di principio i propri diritti di committer, anche se cambiano datore di lavoro o se la Confederazione sceglie un altro fornitore. I diritti di committer si estinguono soltanto se vengono ceduti volontariamente o se la maggioranza degli altri committer ne decide la revoca.

La Confederazione ha il diritto di designare quali committer singoli sviluppatori delle imprese che incarica. Non ha il diritto di revocare a qualcuno i diritti di committer senza motivo.

Le code review sono effettuate da almeno una persona che agisce quale committer. I committer si organizzano da sé.

La Confederazione si impegna a indennizzare il lavoro di revisione nella misura in cui ciò le sia finanziariamente possibile.

6.6. Ripartizione dei costi e fornitori

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Confederazione / Mandatario / Interno

I fornitori sono selezionati periodicamente mediante bandi OMC o una procedura per incarico diretto (a seconda dell’ampiezza delle prestazioni di manutenzione).

Confederazione / Partner / -

Importante: verificare preventivamente se ciò sia ammissibile sotto il profilo del diritto degli acquisti pubblici nel caso concreto.

Completeremo la documentazione su questo punto. La Confederazione sceglie il fornitore XXX quale partner strategico e mira a una cooperazione a lungo termine.

Comune / Mandatario / Interno

Ogni membro della community incarica autonomamente uno o più fornitori di software di attuare gli adattamenti e le estensioni di cui necessita. Per gli adattamenti desiderati da più membri, la ripartizione dei costi è concordata caso per caso. Il membro con la quota di costi maggiore è responsabile della scelta e dell’incarico al fornitore. (Disciplinamenti per la manutenzione del software)

Comune / Mandatario / Terzi

L’associazione seleziona periodicamente i fornitori di software in modo centralizzato mediante bandi OMC o procedure per incarico diretto (a seconda dell’ampiezza delle prestazioni di manutenzione). I costi sono attribuiti ai membri secondo una chiave: (da definire: per popolazione, utenti ecc.)

Comune / Partner / Interno

Ogni membro della community incarica autonomamente uno o più fornitori di software di attuare gli adattamenti e le estensioni di cui necessita. Per gli adattamenti desiderati da più membri, la ripartizione dei costi è concordata caso per caso. Il membro con la quota di costi maggiore è responsabile della scelta e dell’incarico al fornitore.

(Disciplinamenti per la manutenzione del software) Se necessario integrati da un contributo dei fornitori partecipanti. Esempio: *Ogni impresa di sviluppo software membro della community si*

impegna a investire annualmente almeno XY giorni lavorativi a proprie spese nello sviluppo, nell'aggiornamento tecnologico o nel marketing dell'applicazione.

Comune / Partner / Terzi
L'associazione seleziona periodicamente i fornitori di software in modo centralizzato mediante bandi OMC o procedure per incarico diretto (a seconda dell'ampiezza delle prestazioni di manutenzione). I costi sono attribuiti ai membri secondo una chiave: (da definire: per popolazione, utenti ecc.) Se necessario integrati dall'aggiunta: *Ogni impresa di sviluppo software membro della community si impegna a investire annualmente almeno XY giorni lavorativi a proprie spese nello sviluppo, nell'aggiornamento tecnologico o nel marketing dell'applicazione.*

Terzi / Mandatario / -
I fornitori sono selezionati periodicamente mediante bandi OMC o una procedura per incarico diretto (a seconda dell'ampiezza delle prestazioni di manutenzione). Con l'aggiunta: *I miglioramenti apportati dai potenziali fornitori e le loro attività nella community sono considerati quale fattore nella valutazione.*

Le modifiche desiderate da esterni non sono finanziate dalla Confederazione, bensì devono essere commissionate e pagate da essi stessi.

Terzi / Partner / -
I fornitori sono selezionati periodicamente mediante bandi OMC o una procedura per incarico diretto (a seconda dell'ampiezza delle prestazioni di manutenzione). Con l'aggiunta: *I miglioramenti apportati dai potenziali fornitori e le loro attività nella community sono considerati quale fattore nella valutazione. Le modifiche desiderate da esterni non sono finanziate dalla Confederazione, bensì devono essere commissionate e pagate da essi stessi.*

6.7. Marketing

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Confederazione / Mandatario / -
La Confederazione non commercializza attivamente l'applicazione, ma la pubblica su piattaforme di software open source. La pubblicazione dell'applicazione è annunciata negli organi specializzati. Se si manifestano interessati, esaminiamo se costituire una community comune o proseguire con la loro versione del software.

Comune / Partner / -
In aggiunta:
I fornitori possono commercializzare l'applicazione e cercare ulteriori interessati. La Confederazione può essere citata come referenza. Accettiamo consapevolmente che ne possano derivare versioni divergenti del software.

Comune / Mandatario / Interno
L'applicazione è commercializzata dai membri o dall'associazione.

Comune / Partner / Interno
L'applicazione è commercializzata dai fornitori.

Comune / - / Terzi
L'applicazione è commercializzata dall'associazione, dalla società semplice e dai loro membri.

aperto / - / -
L'applicazione può essere commercializzata dai fornitori, oppure si può rinunciare a un marketing esplicito.

Alternativa senza marketing esplicito: la Confederazione non commercializza attivamente l'applicazione, ma la pubblica su piattaforme di software open source. Negli organi specializzati segnaliamo di aver pubblicato l'applicazione e che una collaborazione è benvenuta. Mostriamo il software agli interessati e cerchiamo di integrarli nei nostri processi di elaborazione della roadmap.

6.8. Trattamento dei contributi esterni

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Confederazione / Mandatario / -

I contributi e le richieste esterni devono essere trattati anche se la Confederazione resta l'unica decisore riguardo al software (salvo in caso di pubblicazione minima, dove un fork è inevitabile). Proponiamo il seguente modo di procedere:

Trattamento degli errori segnalati
Cerchiamo di riprodurre gli errori segnalati da persone esterne e chiediamo precisazioni se necessario. Se la riproduzione riesce, la Confederazione commissiona la correzione degli errori. Se l'errore non può essere verificato o se l'onere per correggerlo è sproporzionato, ci scusiamo con la persona che ha segnalato l'errore e le chiediamo se potrebbe presentare una pull request per porvi rimedio.

Trattamento delle richieste
Rispondiamo a ogni richiesta in entrata. Se una richiesta non può essere chiarita con un onere ragionevole e un ulteriore impiego di tempo appare inefficiente, ci scusiamo richiamando le risorse limitate e offriamo di mettere in contatto la persona con un fornitore per una consulenza approfondita.

Trattamento delle pull request
Esaminiamo con cura ogni pull request. Per evitare oneri inutili al contributore, segnaliamo immediatamente se qualcosa contraddice la nostra roadmap o se è dubbio che accetteremo il contributo. La Confederazione decide autonomamente e secondo criteri tecnici quali contributi accettare. Le correzioni di errori che hanno superato il processo di revisione sono sempre accettate.

Trattamento dei fork
Sosteniamo i fork della nostra applicazione. Chi introduce l'applicazione dovrebbe creare un proprio fork. Esaminiamo i fork che ne risultano almeno una volta all'anno e riprendiamo le estensioni valide e adeguate.

Comune / - / -

Per esterni si intendono i contributi provenienti dall'esterno della community definita (membri dell'associazione). Differenze rispetto a quanto sopra:
Trattamento dei fork
Vogliamo evitare fork (duraturi) dell'applicazione. Gli interessati dovrebbero invece aderire direttamente alla community. All'interno della community cerchiamo di assicurare che tutti i membri utilizzino la versione principale.

Aperto / - / -

La costituzione di una community funzionante e di una cultura aperta è per noi importante.

Trattamento degli errori segnalati
Cerchiamo di riprodurre e correggere gli errori segnalati. Presupponiamo la collaborazione attiva delle persone che segnalano. Se possibile, esse dovrebbero creare direttamente una pull request con una correzione dell'errore.

Trattamento delle richieste
Trattiamo ogni richiesta entro al massimo una settimana. Le richieste di contributori attivi sono trattate in via prioritaria e in modo approfondito. Cerchiamo di coinvolgere altri membri nel trattamento delle richieste.

Trattamento delle pull request
Vogliamo ricevere il maggior numero possibile di pull request di elevata qualità. Coinvolgiamo tutti i

contributori attivi nelle revisioni e nella valutazione tecnica e specialistica delle pull request. Per i contributi controversi cerchiamo di giungere a una decisione mediante una votazione informale.

Trattamento dei fork
Cerchiamo di rendere superflui i fork mediante una gestione della community attiva e aperta. Se nondimeno emergono fork, li esaminiamo attivamente e cerchiamo di riprendere le estensioni e i miglioramenti ivi sviluppati. In tali casi sollecitiamo attivamente pull request.

In generale va rilevato che i contributi possono spaziare dal sostegno ad altri utenti fino all'assunzione della responsabilità di intere parti di un progetto (cfr. *[BITKOM2023]* capitolo 4.2).

6.9. Esercizio dell'applicazione

Gestione del prodotto / Fornitore quale / Sviluppo — Proposta

Comune / - / -
Un esercizio centralizzato può essere offerto in opzione dall'associazione centrale, nel senso del Software as a Service (SaaS). Va esaminato se ciò è desiderato.

- / Partner / -
Va precisato se il fornitore debba anche esercire il software.

- / Direzione / -
Va indicato che il fornitore è responsabile dell'esercizio.

I livelli di supporto e i contatti di supporto dovrebbero parimenti essere precisati in questo capitolo.

7. Informazioni importanti per i responsabili di community

7.1. Ci vuole tempo

La costituzione di community è qualcosa che va pianificato a lungo termine. Perseveranza e costanza sono due qualità importanti per una community di successo.

7.2. Ulteriori suggerimenti per la costituzione di una community

Ulteriori suggerimenti per una buona gestione della community figurano su https://opensource.guide/code-of-conduct/ e https://opensource.guide/best-practices/.

7.3. Comunicazione

La pubblicazione di software e documentazione apre all'Amministrazione federale un nuovo canale di comunicazione. Se ciò non avviene con cura e professionalità, l'immagine pubblica della Confederazione può risentirne.

Le community OSS dell'Amministrazione federale si trovano in un campo di tensione riguardo alla comunicazione: da un lato valgono le **prescrizioni generali di comunicazione**, dall'altro la community deve consentire una **collaborazione formale e informale semplice** oltre i confini organizzativi, preferibilmente senza controlli di qualità e approvazioni formali.

L'utilizzo di repository pubblici ed eventualmente di issue tracking e wiki pubblici significa che le dichiarazioni dei singoli collaboratori del progetto sono visibili al pubblico senza filtri. Ciò richiede ai collaboratori del progetto un atteggiamento professionale e il rispetto di standard qualitativi per le pubblicazioni in simili contesti.

Nei progetti OSS ciò è normalmente disciplinato mediante il **Code of Conduct**. Il codice di comportamento dell'Amministrazione federale[10] è formulato in modo molto generale, ma il principio centrale è qui applicabile: «I collaboratori adempiono i loro compiti in modo responsabile, integro e leale. Anche nella vita privata badano a non pregiudicare la reputazione, la credibilità e l'immagine della Confederazione.»

7.4. Sviluppo aperto

Se lo sviluppo software è previsto sin dall'inizio in un repository aperto, la pubblicazione avviene automaticamente. Le liste di controllo determinanti secondo le istruzioni Em002-2 vanno compilate all'inizio e i processi di sviluppo del fornitore di prestazioni o del fornitore devono consentirlo. Il vantaggio è che la community può essere costituita sin dal principio. Più organizzazioni possono inoltre collaborare allo sviluppo.

I repository chiusi costituiscono una forma intermedia, ma sono impiegati per altri partner in una collaborazione.

7.5. Gestione degli utenti

Il progetto (o l'organizzazione promotrice) stabilisce se gli sviluppatori e gli altri collaboratori del progetto lavorano a nome proprio o in forma anonima. Se le persone non lavorano in forma anonima e dispongono già di login sulle piattaforme, è più semplice utilizzare questi (siano essi professionali o privati).

La piattaforma dovrebbe in linea di principio consentire l'integrazione di terzi, poiché altrimenti nessuno sviluppatore esterno alla Confederazione potrà collaborare.

Al momento dell'uscita dal progetto i relativi diritti sul progetto vanno cancellati. Ne è responsabile la gestione del prodotto.

7.6. Fusione di community

Ove possibile il numero di community andrebbe mantenuto ridotto. Ciò significa:

- Se esiste già una community in cui è prevista una struttura analoga per problemi analoghi con le stesse persone, andrebbe utilizzata un'unica community.
- Più progetti correlati che si rivolgono allo stesso pubblico target possono essere riuniti in un'unica community.
- Non andrebbero create inutilmente strutture di community fondamentalmente diverse laddove ne esistano già.

7.7. Trattamento dei contributi di terzi

Il contributo di software ad altri progetti è trattato nel capitolo 3.3. Gli stessi principi valgono in senso inverso.

Sono importanti i punti seguenti:

- Va utilizzato un Contributor Licence Agreement adeguato per assicurare che il codice sorgente appartenga in seguito alla Confederazione (cfr. anche Em002-3 Guida alle licenze OSS, capitolo sui Contributor Licence Agreement). Esso va firmato dal proprietario del contributo.

10. https://www.epa.admin.ch/dam/epa/it/dokumente/aktuell/medienservice/120_verhaltenskodex_i.pdf.download.pdf/1

- Le conferme vanno archiviate.
- I contributi vanno esaminati prima dell’integrazione nella base di codice.
- Alle sottomissioni andrebbe risposto tempestivamente (issue, request, pull request) e trattandole nel modo più costruttivo possibile.
- Tutti i canali e i disciplinamenti di governance andrebbero indicati nel concetto di community, sul sito web e nel repository.

7.8. Segnalazioni rilevanti per la sicurezza

Oltre alle consuete segnalazioni di errori va previsto — ove la natura del progetto lo richieda — che le vulnerabilità rilevanti per la sicurezza possano essere segnalate in modo confidenziale, come avviene per esempio con un programma di bug bounty.

I documenti di trustbroker.swiss possono servire da esempio:

- https://github.com/trustbroker-swiss/trustbroker.swiss/blob/main/Security-and-Vulnerability-Disclosure.md

7.9. Evitare i fork

Se i fork sono naturali nel mondo open source, la reintegrazione di codice sorgente, funzionalità e così via provenienti da fork è tutt’altro che semplice. Può valere la pena rivolgersi alle persone o organizzazioni che stanno valutando un fork e tentare di mantenerle coinvolte nel progetto principale. Trovare un equilibrio tra interessi concorrenti è spesso necessario in tali casi.

7.10. Ripartizione dei costi

È opportuno concordare chiavi di ripartizione generali tra i partner principali. Queste possono essere adeguate ogni qualche anno o quando la situazione cambia fondamentalmente. Per un lavoro efficiente sul software i grandi partner dovrebbero piuttosto assumere quote un po' maggiori di quanto dovrebbero effettivamente (se necessario anche il coordinamento complessivo). L’organizzazione deve lavorare e non litigare sulle finanze. Se la collaborazione riguarda più progetti (per esempio le soluzioni settoriali nel settore delle ferrovie a scartamento normale), andrebbe utilizzata per tutti i progetti una chiave di ripartizione uniforme. Dal punto di vista delle autorità federali è un miglioramento se i costi non devono essere sostenuti da sole.

Possibili criteri per le chiavi di ripartizione:

- Stima della ripartizione dei benefici (p. es. tra un’autorità federale e un fornitore che spera in altri clienti)
- Numero di abitanti/utenti (p. es. tra uffici, Comuni, città)
- Capacità finanziaria (p. es. Cantoni)
- Prodotto interno lordo (p. es. tra Cantoni)
- Chi desidera maggiore controllo

Per evitare discussioni, la chiave di ripartizione non dovrebbe essere semplicemente legata al budget annuale. Una clausola nello spirito del principio di causalità può consentire adeguamenti più rapidi per singoli partner, se questi sono disposti a pagare un supplemento.

La chiave di ripartizione dovrebbe applicarsi anche alle spese generali, alla manutenzione/al supporto e a un minimo di sviluppo ulteriore. Queste parti andrebbero se possibile risolte a lungo termine.

Va tenuto presente che, in presenza di finanziamenti di terzi, tali parti vorranno tipicamente maggiore voce in capitolo. La gestione della community deve essere attrezzata per farvi fronte.

7.11. Prestazioni supplementari secondo l’art. 9 cpv. 5 e 6 LMeCA

Le autorità federali possono (esse stesse, tramite fornitori di prestazioni o fornitori) erogare prestazioni supplementari, in particolare per l’integrazione, la manutenzione, la garanzia della sicurezza delle informazioni e il supporto.

Il limite è che esse devono servire i compiti delle autorità e poter essere erogate con un onere proporzionato.

Ne consegue che l’autorità federale non può essere l’impresa di sviluppo per terzi. Essa corregge errori, può trattare aspetti rilevanti per la sicurezza, può fornire un certo supporto e aiuto all’integrazione (tutte attività che contribuiscono a costituire una community). I nuovi sviluppi e le funzionalità supplementari per terzi dovrebbero orientarsi in primo luogo allo scopo di lavoro normale dell’autorità. Può naturalmente darsi che la messa a disposizione del software faccia parte del compito; in tal caso possono essere realizzati sviluppi in qualsiasi momento. Si rileva inoltre che lo sviluppo software non è il compito principale dell’autorità federale.

Ciò significa che, se determinate funzionalità sono sviluppate essenzialmente in via esclusiva per terzi, esse vanno integrate in partenariato o tramite il fornitore.

Per evitare problemi di diritto degli acquisti pubblici, nell’acquisto di software e prestazioni occorre badare a che funzionalità possano essere messe a disposizione di terzi (eventualmente soltanto terzi statali) dai fornitori.

La fissazione degli emolumenti è già stata toccata nel capitolo «Supporto». La LMeCA costituisce già la base legale per tali emolumenti. Questi vanno comunicati sul sito web della rispettiva autorità federale. Esempi figurano nell’allegato G.

L’art. 9 cpv. 6 stabilisce che di norma non si deve fare concorrenza al settore privato. Ciò significa che gli emolumenti non dovrebbero essere fissati troppo bassi e dovrebbero quantomeno coprire i costi. In caso di dubbio è preferibile fissare tariffe orarie troppo alte anziché troppo basse.

In generale si raccomanda di lavorare con una fino a tre diverse tariffe orarie standard.

Se un dipartimento federale intende definire eccezioni al cpv. 6, ciò non deve fare concorrenza al settore privato. In determinate circostanze può essere più semplice fissare gli emolumenti e aumentarli in caso di opposizione di imprese private effettivamente in grado di offrire la prestazione.

7.12. Sovranità digitale

Fintanto che la Confederazione non dispone di un proprio repository, per garantire la sovranità digitale si raccomanda di realizzare regolarmente copie dei repository pubblici.

Va osservato che non viene copiato soltanto il codice sorgente, bensì anche altre informazioni quali issue tracking, wiki, configurazioni ecc.

Occorre inoltre risolvere la gestione degli utenti per assicurare il controllo sul codice sorgente pubblicato. L’obiettivo generale è garantire uno sviluppo indipendente dalla rispettiva piattaforma e consentire alla community di cambiare.[11]

11. Cfr. anche https://www.bfh.ch/de/aktuell/news/2024/neue-studie-digitale-souveraenitaet/

8. Allegato

8.1. Modifiche rispetto alla versione precedente

- Aggiunto il capitolo 1 «L’essenziale in breve»
- Scopo della community aggiunto al capitolo 3
- Aggiunti i capitoli 3.1–3.3 sulla collaborazione
- Nuovo capitolo 7.7 Trattamento dei contributi di terzi
- Precisato che i fornitori non dovrebbero pubblicare interamente di propria iniziativa (conformemente a *[BBL-WL]*, V.2)
- Menzionato publicode.yml
- Armonizzazione con il nuovo Em002-7
- Ulteriori modifiche e miglioramenti redazionali

8.2. Riferimenti

Cfr. *Em002 Linee guida strategiche per il software open source nell’Amministrazione federale*.

8.3. Abbreviazioni

Cfr. Em002 Linee guida strategiche per il software open source nell’Amministrazione federale ed Em002-6 FAQ sull’OSS.

8.4. Esempi di concetti di community esistenti

Nello spirito di una raccolta di best practice figurano di seguito concetti di community e documenti comparabili già elaborati. Non si tratta necessariamente soltanto di concetti di autorità federali, ma anche di altre organizzazioni.

- GERES Community (http://geres-community.ch)

Classificazione:

- Gestione del prodotto a) Sviluppo comune
- Fornitore piuttosto a) Mandatario
- Ripartizione dei costi: b) Ripartizione dei costi

Nota: il registro comunale GERES non è open source, ma molti aspetti dell’organizzazione possono essere rilevanti anche per applicazioni open source. Documenti: statuti (pubblici) e regolamento interno (su richiesta).

- iGov Portal

Il presente capitolo contiene attualmente concetti del Cantone di Berna e sarà completato non appena saranno disponibili esempi federali.

8.5. Esempi di collaborazioni

I seguenti esempi illustrano come sono state strutturate le collaborazioni.

8.6. Collaborazione: ZenDiS in Germania

ZenDiS[12] è un'organizzazione il cui obiettivo è ridurre la dipendenza nell'amministrazione pubblica (principalmente in Germania) promuovendo il software open source.

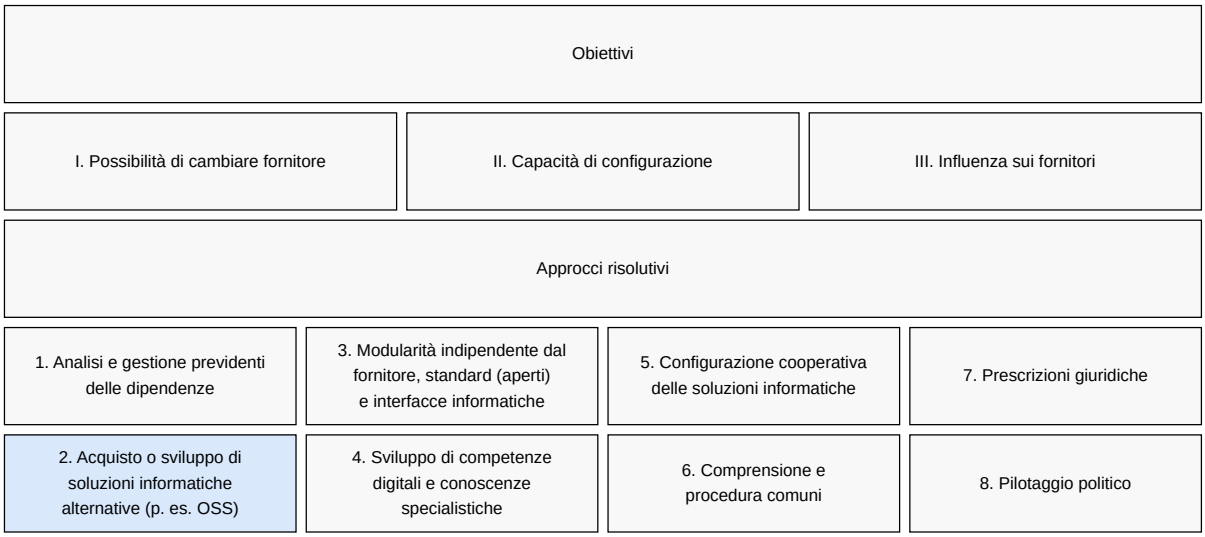

Figura 1: Obiettivi e approcci per rafforzare la sovranità digitale (ZenDiS)

A tal fine ZenDiS collabora con partner strategici.

Una cooperazione tra l'Amministrazione federale e ZenDiS non dovrebbe quindi essere possibile senza restrizioni, e un coordinamento non sarebbe vincolante. Ciò può cambiare con il tempo.

Concretamente l'Amministrazione federale dovrebbe costituire una propria organizzazione per la Svizzera e coordinarsi informalmente con ZenDiS. Ciò potrebbe eventualmente avvenire tramite eOperations, fermo restando che eOperations diventa attiva soltanto quando è annunciato un fabbisogno concreto ed è messo a disposizione un budget. Una motivazione per una simile organizzazione parallela potrebbe essere trovata nel contesto del governo elettronico o della LMeCA.

Nota: ZenDiS è soggetta al diritto degli acquisti pubblici dell'UE e della Germania. Vi sono differenze rilevanti rispetto alla Svizzera. ZenDiS non può quindi essere replicata uno a uno in Svizzera, e una cooperazione con ZenDiS non è agevole.

8.7. Collaborazione: Open Trip Planner (OTP)

Open Trip Planner[13] è un'applicazione open source per la pianificazione di viaggi. È organizzata come progetto della Software Freedom Conservancy a New York. Singoli sviluppatori sono impiegati presso imprese di trasporto e fornitori. È degno di nota che un attore del settore pubblico — concretamente ENTUR[14] — abbia assunto uno dei ruoli guida. ENTUR sviluppa in primo luogo per il proprio fabbisogno, ma ha anche un orientamento cooperativo, in particolare per i Paesi nordici. ENTUR è organizzata quale società di proprietà del Ministero norvegese dei trasporti e delle comunicazioni. In quanto maggiore attore, ENTUR assorbe anche una quota rilevante dei costi di coordinamento. Chiunque può presentare richieste di funzionalità; queste sono discusse nelle riunioni di coordinamento e sviluppate se qualcuno le finanzia. Il finanziamento avviene principalmente per singola funzionalità, da parte della o delle parti interessate. In sostanza ciascuna parte sostiene i

12. Pagina iniziale ZenDiS: Co-creating Digital Sovereignty
13. OpenTripPlanner
14. https://entur.no/

propri costi. In Svizzera le prestazioni di supporto dovrebbero inoltre essere messe a concorso separatamente.

8.8. Nuova collaborazione avviata dalla Confederazione: Swiss Trust Broker

Il Trust Broker[15] è un software della Confederazione per eIAM, disponibile come open source. Lo sviluppo è svolto dalla Confederazione. Diversi Cantoni hanno deciso di impiegarlo direttamente come SaaS o con istanze proprie. Essendo disponibile sotto licenza AGPL, le estensioni realizzate vanno a loro volta a beneficio della Confederazione. Data la sua natura critica per la sicurezza, lo sviluppo per l'Amministrazione federale è svolto esclusivamente dalla Confederazione. Altre organizzazioni possono presentare requisiti, che saranno attuati se il team di progetto li ritiene adeguati.

8.9. Nuova collaborazione avviata dai Cantoni: inosca

Tramite inosca[16] diversi Cantoni hanno costituito una comunità di sviluppo incentrata sulle autorizzazioni elettroniche, nata dal progetto eBau.[17] eBau comprende anche Caluma,[18] un framework per l'elaborazione collaborativa — un esempio di componente estratto da una soluzione più ampia e sviluppato in un framework autonomo e ampiamente applicabile. Esso può essere utilizzato anche in altri progetti, anche laddove alcune parti siano soggette a condizioni di eccezione, consentendo così al maggior numero possibile di altri enti di beneficiare del codice.

La community si riunisce almeno una volta al mese (in opzione due volte) per scambiarsi opinioni su temi tecnici e pianificare la roadmap comune. Tutti i partecipanti possono proporre temi. Se un tema è ritenuto rilevante per la community, i Cantoni interessati si annunciano e si organizzano in un gruppo di lavoro. I risultati sono restituiti continuamente alla community. inosca opera secondo un principio fondamentale: chi progetta, finanzia. Ossia, tutti i Cantoni che partecipano a un gruppo di lavoro si dividono i costi della nuova funzionalità. La funzionalità è messa a concorso e i costi sono ripartiti all'interno della community secondo una formula di ripartizione predefinita. Non appena una funzionalità è completata e introdotta, diventa immediatamente disponibile gratuitamente per tutti gli altri Cantoni.

La community ha inoltre deciso di accantonare un modesto importo annuo fisso per coprire i costi amministrativi di gestione della community. Tale budget è attribuito ogni anno alle parti che assumono un ruolo organizzativo.

8.10. SNOWPACK del WSL — contributi diretti degli sviluppatori

SNOWPACK[19] è un progetto open source sviluppato dal WSL per modellizzare la formazione del manto nevoso. Si tratta di un modello altamente specializzato e la community è stata di conseguenza integrata nella comunità specialistica esistente. Alcuni issue sono già stati individuati quali potenziali contributi. Il modello di lavoro è qui che ciascuna parte sostiene i propri costi e apporta funzionalità. Le direttive su stile di codifica e processo di rilascio sono parimenti documentate in tale contesto.

8.11. Spunti per statuti associativi

Gli statuti dovrebbero essere mantenuti il più semplici possibile.

I seguenti statuti possono servire da spunto.

- eCH: https://www.ech.ch/sites/default/files/page/STAT_d_DEF_2014-04-10_ech-Statuten.pdf

15. GitHub - trustbroker-swiss/trustbroker.swiss: Documentation of the trustbroker.swiss service
16. https://inosca.ch/
17. https://github.com/inosca/ebau
18. https://github.com/projectcaluma
19. https://snow-models.gitlab-pages.wsl.ch/snowpack-web/

- Stop Piracy: https://www.stop-piracy.ch/wp-content/uploads/2022/01/Statuten_d_10_09_2021.pdf

Ulteriori esempi di statuti saranno aggiunti non appena disponibili.

8.12. Esempi di supporto ed emolumenti

- Emolumenti per prestazioni statistiche: https://www.fedlex.admin.ch/eli/cc/2003/326/it
- Emolumenti dell'IFPDT: https://www.edoeb.admin.ch/edoeb/it/home/datenschutz/grundlagen/dsfa.html
- Prestazioni gratuite dell'IPI per corsi nel settore della proprietà intellettuale (non software, ma a titolo di esempio): https://www.ige.ch/it/prestazioni/ip-academy/informazioni-generali/prezzi

Gli esempi saranno aggiunti non appena disponibili.