

# Anforderungen und Aufgaben interkommunaler Zusammenarbeit in Open-Source Projekten

## Praxiswissen aus den Modellprojekten Smart Cities



# Impressum

---

Herausgeber

Koordinierungs- und Transferstelle Modellprojekte Smart Cities

c/o DLR Projektträger | [smartcities@dlr.de](mailto:smartcities@dlr.de)

**Stand:**

März 2026

**Bildnachweise:**

Titelbild: Creative Climate Cities

**Autorinnen und Autoren:**

Koordinierungs- und Transferstelle Modellprojekte Smart Cities

Fraunhofer-Institut für Experimentelles Software-Engineering (IESE) | Tizia Grether, Christoph Müller

Telefon +49 631 68002200 | [tizia.grether@iese.fraunhofer.de](mailto:tizia.grether@iese.fraunhofer.de)

Kurzzusammenfassung:

Im Rahmen der Arbeits- und Entwicklungsgemeinschaften der KTS und anhand von Interviews wurden interkommunale Entwicklungspartnerschaften mit Open-Source-Software-Projekten porträtiert und analysiert. Diese Übersicht ergänzt die Studie „Open-Source-Projekte gemeinsam steuern“ und beleuchtet die verzahnten Anforderungen, Aufgaben und Aspekte des Community- und Produkt-Managements.

[www.smart-city-dialog.de](http://www.smart-city-dialog.de)

## Einführung: Was ist Community Management?

---

Im Rahmen von Entwicklungspartnerschaften im Modellprojekt Smart Cities (MPSC) werden die gemeinschaftliche Entwicklung und Verstetigung von Open-Source-Software angestrebt. Dies erfordert, Prozesse innerhalb der Gemeinschaft zu strukturieren und die Kommunikation innerhalb und nach außen zu verantworten.

Die gemeinschaftliche Entwicklung bringt Synergien, zum Beispiel durch geteilte Kosten bei der Weiterentwicklung und Wartung der Software. Dadurch werden eine nachhaltige Verstetigung und Skalierung von Open-Source-Projekten möglich. Anwendungen städteübergreifend zusammen zu entwickeln, bringt jedoch auch Herausforderungen mit sich. Insbesondere muss sichergestellt werden, dass Wissen effektiv geteilt wird, Entscheidungen strukturiert ablaufen und der weiterentwickelte Code wieder in eine gemeinsame Hauptversion zusammengeführt wird.

Die akteursübergreifende Zusammenarbeit in Open-Source-Projekten ist in der Softwareentwicklung seit Jahrzehnten fest etabliert und durch Plattformen wie GitHub, GitLab oder openCode institutionalisiert, auf denen der Source Code der meisten Projekte gehostet wird. Entwicklerinnen und Entwickler tragen weltweit zu Projekten bei, teilen Wissen und verbessern Software durch offene Diskussionen und Review-Prozesse.

Es ist eine relativ neue Entwicklung, dass Kommunen in diesem Prozess als Akteur auftreten. Traditionell waren Städte und Gemeinden oft reine Nutzer von kommerzieller Software, die von (großen) Anbietern bezogen wurde. Doch, mit der wachsenden Bedeutung von digitaler Souveränität und Kosteneffizienz gewinnt die Kollaboration in Open-Source-Projekten weiter an Relevanz. Auch wenn sich Kommunen meist nicht direkt in die Code-Entwicklung einbringen, spielen sie eine wichtige Rolle bei der Weiterentwicklung der Open-Source-Software. Sie tragen durch ihre Ideen, Visionen und spezifischen Anforderungen maßgeblich dazu bei, dass Lösungen praxisnah und nutzerorientiert gestaltet werden. Das heißt, sie sind sowohl in der Rolle der Nutzerin als auch Input-Geberin involviert.

Die Zusammenarbeit erfordert eine entsprechende organisatorische Hülle. Kernaufgaben des Community- und Produktmanagements sind die Einberufung und Geschäftsführung eines Architekturremiums, die Betreuung der Anwender-Community sowie die Definition von Produktweiterentwicklungsprozessen und der dazugehörigen Governance im Sinne entsprechender Gremien, in denen Entscheidungen getroffen werden. Außerdem müssen die Softwarewartung und zentrale Weiterentwicklung wie Securitypatches beauftragt und orchestriert werden. Das Community- und Produktmanagement stellt somit den organisatorischen und technischen Rahmen für die kollaborative Entwicklung und Nutzung sowie den langfristigen Betrieb von OSS-Projekten dar. Relevant werden diese Rollen in der kommunalen Praxis immer dann, wenn

- (1) mehrere Kommunen die gleiche Software nutzen,
- (2) mehrere Kommunen ein Eigeninteresse daran haben, die Software gemeinsam innerhalb einer Entwicklungspartnerschaft weiterzuentwickeln und
- (3) die Kommunen die Produktweiterentwicklung beeinflussen bzw. steuern möchten.

## Doch wie lässt sich die Zusammenarbeit konkret gestalten?

Hierbei werden in der Praxis verschiedene Ansätze verfolgt, die sich je nach Größe und Struktur der beteiligten Akteure unterscheiden. Gemeinsames Ziel ist es, die Zusammenarbeit und Weiterentwicklung zu strukturieren und die Kommunikation innerhalb der Gemeinschaft und nach außen zu fördern und transparent darzustellen. Hierzu verfolgen das Community- und Produktmanagement unterschiedliche Aufgaben, die jedoch ineinandergreifen. Generell lässt sich festhalten: Das Community Management betreut die Mitglieder von OSS-Entwicklungspartnerschaften, das Produktmanagement das Produkt an sich.<sup>1</sup>

Diese Übersicht bietet eine Orientierung, welche Anforderungen und Aufgaben diesen Rollen zukommen. Dabei werden die Aspekte des Community- und Produktmanagements verzahnt betrachtet. Die Aufgaben können entweder von unterschiedlichen Akteuren oder demselben Akteur besetzt sein. Wichtig ist es, im Vorfeld vorhandene personelle Ressourcen zu prüfen und die Zusammenarbeit daraufhin auszurichten.

---

<sup>1</sup> Für ausführlichere Definitionen: Theobald, A.; Grether, T.; Zehler, T., 2025: Open-Source-Projekte gemeinsam steuern: Community- und Produktmanagement in interkommunalen Entwicklungspartnerschaften. Zugriff: <https://www.smart-city-dialog.de/wissen/publikationen/open-source-projekte-gemeinsam-steuern-community-und-produktmanagement> [abgerufen am: 30.04.2026]

# Anforderungen und Aufgaben

---

## Produktvision und -strategie

Ein übergeordnetes Management übernimmt die interne und externe Repräsentanz der Entwicklungspartnerschaft. Dazu gehört insbesondere die kontinuierliche (Weiter-)Entwicklung der Produktvision und -strategie sowie die Pflege der Community nach innen und außen. In interkommunaler Zusammenarbeit ist es wichtig, dass an einem gemeinsamen Produkt gearbeitet wird und entsprechende Weiterentwicklungen ein gemeinsames Ziel verfolgen.

Dies erfordert eine enge Zusammenarbeit mit dem Produktentwicklungsteam zur Definition und Umsetzung der Produktziele, die regelmäßig neu überprüft und gegebenenfalls angepasst werden müssen. Zum Produktentwicklungsteam gehören je nach Form der Zusammenarbeit sowohl Vertreter der kommunalen Partner als auch entsprechende Dienstleister oder weitere Akteure wie interessierte Freiwillige, die sich beispielsweise über openCode einbringen. Innerhalb der Community bedeutet dies die Organisation und Koordination von gemeinsamen (digitalen) Treffen zur Abstimmung.

## Gremien und Governance

Insbesondere erfordert die Strukturierung eines gemeinsamen Prozesses das Einberufen, Vorbereiten und Moderieren von Gremiensitzungen. Dies übersetzt sich meistens in ein strategisches (z.B. für die Gesamtstrategie) und ein technisches Gremium (z.B. für Architekturentscheidungen). Beispielsweise kann der Austausch in einem jährlichen Strategie-Workshop sowie in Workshops für Entwicklerinnen und Entwickler auf technischer Ebene umgesetzt werden. Das Community Management unterstützt das Produktmanagement bei der Entscheidungsfindung und erstellt Entscheidungsvorlagen für die Weiterentwicklung.

## Organisationsform

Der erste Schritt für eine Zusammenarbeit als Community besteht darin, eine Satzung sowie ein Kooperationsvertrag aufzusetzen bzw. weiterzuentwickeln. Je nach rechtlicher Organisationsform können weitere Aufgaben anfallen, wie zum Beispiel Buchhaltung, Kosten- und Leistungsrechnung sowie die Abwicklung und Verwaltung eventueller Mitgliedsbeiträge.

## Erschließung von Fördermöglichkeiten

Außerdem können weitere Aufgaben hinzukommen, wie die Erschließung von Fördermöglichkeiten, der Austausch mit Verfahrensherstellern sowie die Standardisierung von Schnittstellen zu benötigten Datenquellen.

## Öffentlichkeitsarbeit

Die Vertretung der Community nach außen sieht Aktivitäten zur Öffentlichkeitsarbeit und zum Wissenstransfer vor, beispielsweise die Teilnahme und Präsentation der Entwicklungspartnerschaft in einschlägigen Formaten wie Kongressen, Messen und Social-Media-Auftritten. Es sollte ein Prozess entwickelt werden, der diese Aufgaben proportional auf alle Partner verteilt.

## Support und Wissenstransfer

Die Aufgabe des Supports beinhaltet die Koordination und Durchführung von Support-Anfragen sowohl der Entwicklungspartner intern als auch der Beratungs-Anfragen von interessierten Kommunen sowie deren Onboarding. Die Ergebnisse der Beratung sind zu dokumentieren und allen Mitgliedern zur Verfügung zu stellen.

Für das Onboarding neuer Mitglieder sollten Schulungsmaterialien angefertigt und Schulungen durchgeführt werden.

Optional: Aufgabe des Supports kann außerdem die Unterstützung bei Vergaben und Ausschreibungen der Entwicklungspartner sein.

## Aufsetzen eines geeigneten Toolings

Zur Koordination und Bearbeitung von Support-Anfragen, Change- und Bug-Requests ist ein entsprechendes Ticket-System und deren Pflege notwendig. Sind diese in einem Tool zentralisiert, werden Medienbrüche verhindert.

Außerdem ist eine Infrastruktur für geeignete Kommunikations-Tools innerhalb der Community aufzusetzen.

## Partner Scouting und Management

Aufgabe ist die Identifizierung und Verwaltung von Partnerschaften, die zur Unterstützung und Weiterentwicklung des Produkts beitragen sowie die Pflege und Organisation eines Entwickler-Netzwerks.

Es können themenbezogene Informations- und Erfahrungsaustausche zwischen der Community und externen Akteuren durchgeführt werden.

Außerdem sollten ein kontinuierliches Marktscreening und eine Technologieevaluation umgesetzt werden.

## Strukturierter Produktweiterentwicklungsprozess

Für das Produktmanagement ist ein Produktentwicklungsprozess zu erarbeiten. Teil dieses Prozesses ist das Anforderungsmanagement. Dies beinhaltet die Backlog<sup>2</sup>-Pflege, das Ticketmanagement sowie die weiterführende Konzept-Entwicklung.

Bei Feature-Requests<sup>3</sup> von Mitgliedern ist jeweils ein Prozess zu entwickeln, wenn das Feature gemeinschaftlich finanziert werden soll (Prozess 1) bzw. wenn dieses nur von einer Teilgruppe der Mitglieder getragen, d.h. finanziert und gepflegt, wird (Prozess 2). Außerdem bedarf es eines Prozesses für Change-Requests (Prozess 3) sowie Bug-Requests (Prozess 4). Nachfolgende Prozessschritte bilden den prototypischen Prozess basierend auf dem Civitas-Modell ab.

- Prozess 1: Bei dem Request, ein neues Feature gemeinschaftlich zu finanzieren wird eine 75%-Zustimmungsquote für die Durchführung angesetzt. Stimmberechtigt sind alle Mitglieder der Entwicklungspartner.

---

<sup>2</sup> Ein Backlog ist eine priorisierte Liste von Aufgaben, Anforderungen oder Features, die in einem Projekt oder Entwicklungsprozess noch umgesetzt werden müssen. Im agilen Kontext (z. B. Scrum) enthält das Product Backlog alle bekannten Anforderungen an ein Produkt, die nach Wichtigkeit und Dringlichkeit sortiert sind und kontinuierlich angepasst werden (Schwaber und Sutherland 2020).

<sup>3</sup> Ein Feature-Request ist ein Vorschlag bzw. eine Anfrage von Nutzenden oder Stakeholdern für neue Funktionen oder Verbesserungen an einem Produkt (Atlassian 2025).

nerschaft. Hier muss besprochen werden, wie mit Anfragen mit hohem Entwicklungsvolumen umgegangen wird. Diese könnte begrenzt werden bzw. könnte ein jährliches Budget auf der Summe der Einlagen definiert werden, die für gemeinschaftliche Entwicklungen generell pro Jahr vorgesehen sind. Außerdem könnte hier besprochen werden, wie mit der Situation umgegangen wird, wenn eine Haushaltssperre vorliegt. Planungssicherheit könnte auch bieten, dass ein Datum festgelegt wird, bis zu dem das Budget für die gemeinsame Entwicklung ausgegeben wird; danach könnte die verbliebene Summe für andere Zwecke ausgegeben werden.

- Bei Zustimmung:
  - Es ist ein Kooperationsvertrag für jedes Mitglied aufzusetzen. Die Sicherstellung rechtlicher Konformität muss gewährleistet werden, wozu ggf. eine rechtliche Beratung zu beauftragen ist.
  - Die langfristige Qualitätssicherung wird durch das operative Management sichergestellt.
  - Die Community- und Pflegeaufwände sind neu zu bewerten.
  - Die Aufnahme von Communitykosten und Wartungskosten sind in den Kostenplan/Haushalt einzustellen.
- Bei Nicht-Zustimmung gibt es keine Möglichkeit der Wiedervorlage.
- Prozess 2: Bei dem Request, ein neues Feature von einer Teilgruppe zu finanzieren, bietet sich eine absolute Mehrheit für die Durchführung an.
  - Bei Zustimmung:
    - Es ist ein Kooperationsvertrag für jedes Mitglied aufzusetzen. Die Sicherstellung rechtlicher Konformität muss gewährleistet werden, wozu ggf. eine rechtliche Beratung zu beauftragen ist.
    - Die Eigenentwicklung wird von der Teilgruppe durchgeführt.
    - Die langfristige Qualitätssicherung wird sichergestellt.
    - Die Community- und Pflegeaufwände sind neu zu bewerten.
    - Die Aufnahme von Communitykosten und Wartungskosten sind in den Kostenplan/Haushalt einzustellen.
    - Die Herstellung der Produktreife wird durch das operative Management durchgeführt (siehe Anforderungsbaustein 3.6). Hinweis: Dies ist eine ressourcenaufwändige Aufgabe.
    - Optional: Das Feature ist im Anschluss von allen Mitgliedern als Modul/Plugin nutzbar.
  - Bei Nicht-Zustimmung ist die Entwicklung des Features außerhalb der Entwicklungspartnerschaft möglich.

- Optional: Nachträglich ist die Aufnahme in den Core<sup>4</sup> durch einen Antrag möglich. Dieser Prozess wird durch das operative Management koordiniert.
- Prozess 3: Bei dem Request, eine Komponente zu erneuern bzw. zu ändern (Change-Request) sollte eine höhere Zustimmung als bei Prozess 1 gegeben sein. Es bietet sich eine 90%-Zustimmung für die Durchführung an.
  - Das operative Management nimmt diese Requests über das ausgewählte Ticket-System (siehe Leistungsbaustein 3.5) auf und koordiniert die Umsetzung.
    - Die Community- und Pflegeaufwände sind neu zu bewerten.
    - Die Aufnahme von Communitykosten und Wartungskosten sind in den Kostenplan/Haushalt einzustellen.
- Prozess 4: Bei dem Request eines Fehlers oder einer Sicherheitslücke (Bug-Request) ist keine Zustimmung erforderlich.
  - Das operative Management nimmt diese Requests über das ausgewählte Ticket-System (siehe Leistungsbaustein 3.5) auf und koordiniert die Umsetzung.
    - Die Community- und Pflegeaufwände sind neu zu bewerten.
    - Die Aufnahme von Communitykosten und Wartungskosten sind in den Kostenplan/Haushalt einzustellen.

Das anschließende Management der Pull-Requests sowie des Release-Managements ist sicherzustellen. Ein Hauptfokus ist die Koordination und Zusammenführung von dezentral entwickelten Funktionen und Modulen in der gemeinsamen Hauptversion des sogenannten Upstreams<sup>5</sup>.

Eine weitere Aufgabe ist die Abstimmung von Schnittstellen mit dem Ziel der Vereinheitlichung, um die entstehenden Module weitestgehend übertragbar zu entwickeln.

Es sind verbindliche Regeln (Code-Konventionen) für die dezentrale Entwicklung von zusätzlichen Funktionen durch Dritte mit der Community zu erarbeiten und zu verankern.

### Herstellung der Produktreife (optional)

Ein Ziel kann die Herstellung der Produktreife sein. Umgesetzt werden kann die durch eine Hauptversion der Anwendung (White Label Lösung) mit einem vorkonfigurierten Einsatz an Funktionalitäten und Modulen.<sup>6</sup> Dies ist insbesondere für kleinere Kommunen und Kommunen mit weniger leistungsstarken, eigenen Partnern als Zielgruppe hilfreich. Die Herstellung der Produktreife kann sich jedoch auch auf gegebenenfalls einzelne Funktionalitäten oder Module beziehen.

---

<sup>4</sup> Ein Core bezeichnet den zentralen, mindestens erforderlichen Funktions- und Architekturkern einer Software, um diese zu starten. Auf diesem Funktions- und Architekturkern bauen alle weiteren Funktionen, Erweiterungen und Integrationen auf (Computer Security Resource Center (NIST)).

<sup>5</sup> Der Upstream beschreibt die Kernversion der Software, entwickelt und betreut durch die ursprünglichen Entwickler.

<sup>6</sup> Die Konzipierung als Whitelabel-Lösung ermöglicht Dritten, die Software als „leere Leinwand“ zu konfigurieren und einzusetzen. Dies kann erzielt werden, indem etwa Ortsbezüge nicht fest einprogrammiert, sondern durch Konfigurationsoptionen variabel gemacht werden.

Es empfiehlt sich, eine getrennte Umgebung für Testzwecke (Staging-Umgebung) zu betreiben, die von entwickelnden Kommunen oder deren Dienstleistern genutzt werden kann, um die Anbindung von Modulen an die App zu testen.

### **Repository Management**

Unter das Repository Management fallen die Behandlung von Change-Request und des Feedbacks aus der Community.

Außerdem sollte die openCode-Administration sichergestellt sein. Dies beinhaltet die Veröffentlichung und die Dokumentation des Codes auf [opencode.de](https://opencode.de) sowie im eigenen Repository. Idealerweise ist zur Transparenz der Verlauf des Ticketings zu veröffentlichen.

Für die Qualitätssicherung ist die Überwachung einer hochwertigen Dokumentation wichtig.

### **Wartung**

Die Koordination der Wartung der Software, d. h. von Bugfixes und Upgrades, wird am besten von einer zentralen Stelle aus gesteuert. Die Aufgaben der Wartung und der zentralen Weiterentwicklungen wie Security Patches sollten entweder selbst durchgeführt oder beauftragt und orchestriert werden.

Für die Qualitätssicherung ist bei jeder Änderung ein Review- und Feedback-Prozess einzuplanen.

### **Betrieb**

Je nach Organisation der Entwicklungspartnerschaft bietet es sich optional an, den Betrieb der Hauptversion der App in Form einer Software-as-a-Service(SaaS)-Lösung interessierten Kommunen als Full Service Leistung anzubieten.

Außerdem könnte der Betrieb einer Demo-Umgebung für Demonstrationszwecke sinnvoll sein, in der sich interessierte Kommunen die Anwendung im Echtbetrieb anschauen können.

Das operative Management unterstützt beim Third-Level Support und Back-to-Back-Verträgen über Betreiber und Dienstleister.

## Quellen

---

Die Auflistung der Anforderungen und Aufgaben wurde auf Grundlage folgender Quellen erstellt:

- Atlassian, 2025: What is a feature request and how do you write one? Zugriff: <https://www.atlassian.com/agile/product-management/feature-request>, zuletzt aktualisiert am 22.05.2025 [abgerufen am: 05.01.2026].
- Civitas Connect e.V., 2024: „CIVITAS/CORE: Ein Standard für kommunale Dateninfrastrukturen“
- Computer Security Resource Center (NIST): Glossary - Core Software. Hg. v. National Institute of Standards and Technology. U.S. Department of Commerce. Gaithersburg. Zugriff: [https://csrc.nist.gov/glossary/term/core\\_software](https://csrc.nist.gov/glossary/term/core_software) [abgerufen am: 28.01.2026].
- Open Smart City App, 2024: Vereinbarung zur Entwicklungspartnerschaft „Open Smart City App.“
- Schwaber, K.; Sutherland, J., 2020: The 2020 Scrum Guide™. Zugriff: <https://scrumguides.org/scrum-guide.html> [abgerufen am: 07.11.2025].

### Koordinierungs- und Transferstelle Modellprojekte Smart Cities

Heinrich-Konen-Straße 1 | 53227 Bonn  
Telefon: +49 30 / 67055 – 9999

E-Mail: [SmartCities@dlr.de](mailto:SmartCities@dlr.de)  
Webseite: [www.smart-city-dialog.de](http://www.smart-city-dialog.de)