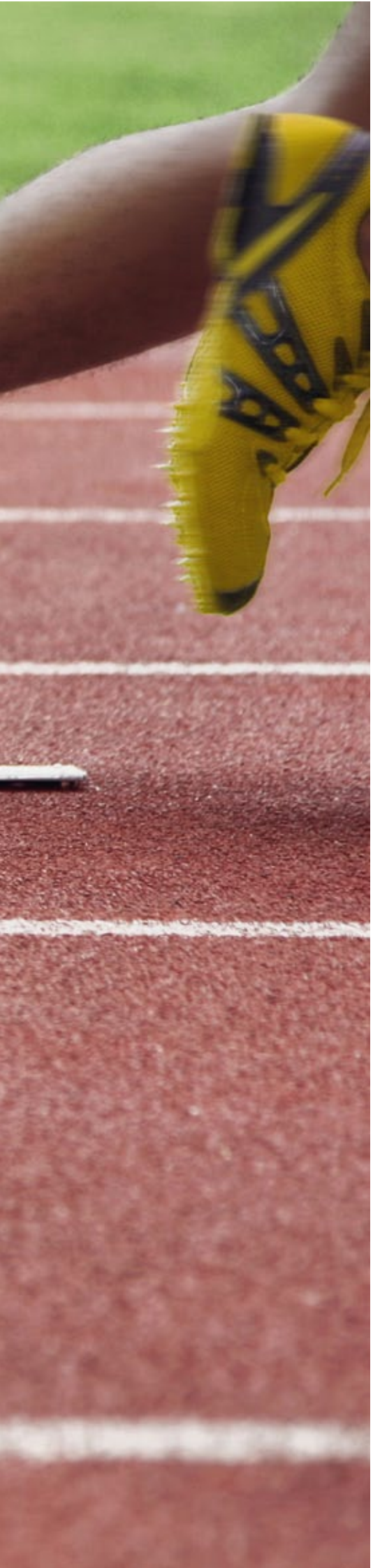
Open Source Software
im geschäftskritischen Einsatz

ERNST & YOUNG
Quality In Everything We Do



Editorial

Wir gehen davon aus, dass bis in den nächsten Jahren nahezu alle der weltweit grössten Unternehmen Open Source Technologien in missionskritischem Umfeld einsetzen werden. Die Frage ist somit nicht ob, sondern wie Unternehmen und öffentliche Institutionen künftig mit Open Source Software umgehen.

Wie bei anderen Grundsatzentscheiden zu Softwareplattformen oder Geräten sind Unternehmen auch bei der Frage des Einsatzes von Open Source Software gut beraten, die Auswirkungen betreffend Chance und Risiko zu analysieren und zu beurteilen. Zum Thema Open Source gibt es heute zu beiden Sichtweisen praxisnahe Grundlagen, welche den kontrollierten Einsatz auch in geschäftskritischen Anwendungen unterstützen und zu einer prüfenswerten Option machen.

Ernst & Young hat diese Publikation erarbeitet, um Vorteile, Risiken und Good Practices rund um Open Source Software vorzustellen, die Varianten des professionellen Einsatzes darzulegen, einen Einblick in die Lizenzthematik zu geben und sich generell einen Überblick zu Open Source zu verschaffen. Wir möchten damit die aktive Auseinandersetzung mit dieser Thematik fördern und einen Beitrag zu einer praxisgerechten Beurteilung leisten.

Jürg Brun
Partner
Advisory Services

Ferdinand Kobelt
Partner
Advisory Services

Reto Aeberhardt
Senior Manager
Advisory Services

Dr. Matthias Stürmer
Senior
Advisory Services

Inhaltsverzeichnis

1. Executive Summary

Open Source ist keine Glaubensfrage

Auch wenn das Thema Open Source in Fachkreisen oft emotional beladen ist und immer noch Unklarheiten herrschen, gibt es eine klare Definition und objektive Vor- und Nachteile betreffend Open Source Software. So wird Software dann als Open Source bezeichnet, wenn sie unter einer von der Open Source Initiative anerkannten Open Source Lizenz veröffentlicht wurde. Solche Lizenzen erlauben, die Software kostenlos zu kopieren und den Quellcode beliebig zu verbessern und zu verbreiten – Freiheiten, die mit proprietärer Software nicht möglich sind (weitere Informationen auf Seite 20ff). Dieselbe Software kann somit von unterschiedlichen Firmen integriert und gewartet werden, wodurch die Abhängigkeit zum Software-Hersteller, der so genannte Vendor Lock-in, sinkt. Gleichzeitig müssen technisches Knowhow erarbeitet (siehe Abschnitt «3. Professioneller Einsatz von Open Source», Seite 12) und die mit der Open Source Lizenz verbundenen Pflichten genau eingehalten werden (siehe Abschnitt «4. Rechtliche Aspekte von Open Source», Seite 15).

Open Source Software muss gesteuert werden

Obwohl selten Marketing für Open Source Software betrieben wird, findet sich heute in jedem Betriebssystem und in fast jeder Fachapplikation Software unter Open Source Lizenzen. Zudem werden immer noch in vielen Entwicklungsabteilungen Open Source Komponenten und Software-Lösungen in geschäftskritischen Anwendungen eingesetzt, ohne dass dies den Verantwortlichen bewusst ist. Um den Umgang mit Open Source Software gezielt zu steuern und die IT Governance zu verbessern, führen Verantwortliche heute oft spezifische Open Source Strategien, Policies oder Architekturrichtlinien ein (siehe Abschnitt «2. Vorteile, Risiken und Good Practices», Seite 6).

Volles Potential mittels Open Source Strategie nutzen

Wird Open Source Software unkontrolliert eingesetzt, kann das volle Potential selten ganz ausgeschöpft werden. Eine ganzheitliche, auf die jeweilige Organisation angepasste Open Source Strategie ist daher notwendig, um unerkannte Chancen zu nutzen, Risiken frühzeitig zu erkennen und entsprechende Massnahmen festzulegen. Sei es beim Einsatz von vollständigen Open Source Lösungen, bei der Verwendung von Open Source Komponenten in der Software-Entwicklung oder gar bei der Lancierung einer eigenen Open Source Community – ein vorausschauendes Vorgehen ist in jedem Fall von Vorteil. Lizenzrechtliche Aspekte spielen dabei genauso eine zentrale Rolle wie auch Software-Beschaffung, Gesamtkostenbetrachtung, Evaluation und nachhaltiger Knowhow Aufbau (siehe Abschnitt «Good Practice: Open Source Strategie», Seite 11).

Wissen aus erster Hand ist gefragt

In einem derart dynamischen Umfeld wie bei Open Source Communities entwickeln sich je nach Ausgangslage neue Software-Projekte rasant zu disruptiven Innovationen. Aber auch etablierte Open Source Lösungen können durch starke Veränderungen in deren Communities rasch Neupositionierungen im Markt erlangen. Laufend werden neue Geschäftsmodelle und Lizenzierungssysteme sowie Analyse- und Evaluationsmethodiken entwickelt. Wie in dieser Publikation aufgezeigt, existieren verschiedene Good Practices. Deren Anwendbarkeit und Umsetzung muss dennoch stets individuell betrachtet werden. Deshalb ist Wissen aus erster Hand von Personen gefragt, die direkten Zugang zu den wichtigen Akteuren im Open Source Markt besitzen.

2. Vorteile, Risiken und Good Practices

Kosteneinsparungen sind oftmals ein guter Grund, sich bewusst mit dem Einsatz von Open Source Software zu beschäftigen. Aber auch die Reduktion der Abhängigkeit von Software-Herstellern, Marketing und Employer-Branding sowie die rasche Verbreitung von Open Source Software sind wichtige strategische Vorteile.

Dem gegenüber stehen Risiken, die beim Einsatz von Software generell und insbesondere beim Umgang mit Open Source zum Tragen kommen. Die vielfach komplexen Lizenzbestimmungen müssen eingehalten, die Verwendung von Open Source Software kontrolliert und der Support gesichert werden. Und wie bei der Einführung von neuer proprietärer Software besteht bei der Migration auf Open Source Software ein inhärentes Migrationsrisiko.

Von den identifizierten Vorteilen und möglichen Risiken lassen sich bewährte Massnahmen ableiten, durch welche die Vorteile realisiert und die Risiken minimiert werden können. Eine bewährte Good Practice ist dabei die Entwicklung einer in die Gesamt-IT-Strategie integrierten Open Source Strategie. Dieser Ansatz ermöglicht die ganzheitliche Behandlung der Thematik und stimmt die verschiedenen Umsetzungsmassnahmen aufeinander ab.

Die untenstehende Grafik und der folgende Text veranschaulichen Vorteile und Risiken im Umgang mit Open Source Software und zeigen daran angelehnte Good Practices auf.

Vorteile von Open Source Software

Kosteneinsparungen

Good Practices:

- Software Portfolio Assessment
- Total Cost of Ownership Analysis
- Shared Maintenance

Kontrolle über Software und Daten

Good Practice:

- Vendor Risk Assessment

Reputationsgewinn

Good Practice:

- Open Source Marketing

Rasche Verbreitung

Good Practice:

- Open Source Project Management

Risiken von Open Source Software

Verletzung der Open Source Lizenzbestimmungen

Good Practices:

- Software Licensing Compliance
- Software Development Guidelines

Unkontrollierter Einsatz von Open Source Software

Good Practices:

- Quality Assessment Framework
- Enterprise Architecture Governance

Support nicht gesichert

Good Practice:

- Support Assessment

Migration auf Open Source misslingt

Good Practice:

- Integration Strategy

Ausnutzen der Vorteile: Good practices

Kosteneinsparungen

Der Einsatz von Open Source Software kann bei richtigem Vorgehen kurz- und langfristig IT-Kosten einsparen.

Einerseits entfallen durch die Ablösung proprietärer Software durch Open Source Alternativen sämtliche Lizenzkosten, was zu unmittelbaren Einsparungen führt. So ist beispielsweise eine Open Source Lösung innerhalb einer Cloud-Umgebung meist ohne Mehraufwand flexibel skalierbar. Andererseits bestehen bei Open Source Lösungen keine Verpflichtungen, was die Wahl des Anbieters bezüglich Pflege und Ausbau der Plattform betrifft. Mit dieser gewonnen Unabhängigkeit wächst die Wahlfreiheit und damit die Verhandlungsstärke, was längerfristig zu Kosteneinsparungen bei Wartung und Weiterentwicklung führt. Auch stehen typischerweise Software-Verbesserungen allen Anwendern zur Verfügung, wodurch wiederkehrende Kosten für Pflege und Ausbau der Software geteilt werden.

Wird allerdings unkoordiniert vorgegangen, kann der Integrationsaufwand auf die Open Source Plattform rasch die eingesparten Lizenzkosten aufwiegen oder es können gar Mehrkosten entstehen. Auch können bei ungenügender Planung Folgeaufwände wie Umschulungen und Datenmigrationen sowie unerwartete Inkompatibilitäten oder fehlende Benutzerinakzeptanz eintreten. Deshalb sind systematische Vorabklärungen und unabhängige Schätzungen der vorgesehenen Investitionen und Kosteneinsparungen wichtig.

Software Portfolio Assessment:
Um relevante Kosteneinsparungsmöglichkeiten zu identifizieren, empfehlen wir für proprietäre Software-Lösungen zu analysieren, ob dafür geeignete Open Source Alternativen am Markt vorhanden sind.

Total Cost of Ownership Analysis:
Wir empfehlen, eine umfassende Total Cost of Ownership Analyse durch erfahrene und unabhängige Experten erstellen zu lassen, um die unterschiedlichen Vorgehensszenarien ganzheitlich vergleichen zu können.

Shared Maintenance:
Wir empfehlen zu prüfen, ob intern entwickelte Software (beziehungsweise gewisse Komponenten) als Open Source Projekt veröffentlicht werden können, um die Wartungs- und Weiterentwicklungskosten mit anderen Nutzern zu teilen.

Kontrolle über Software und Daten

Die Kontrolle über eingesetzte Software und interne Daten stellt einen strategisch wichtigen Aspekt der IT Governance dar, die durch den Einsatz von Open Source Software verbessert werden kann.

Der Einsatz von Open Source Software stellt eine Möglichkeit dar, die heute oft durch externe Anbieter kontrollierte Informatik wieder unabhängiger und flexibler zu gestalten. Durch Offenlegung des Quellcodes ist die Funktionsweise einer Software-Lösung öffentlich einsehbar und durch unabhängige Dritte auf Sicherheit und Datenschutz überprüfbar. Auch verwendet Open Source Software typischerweise offene Standards und Formate, sodass Daten über klare Schnittstellen frei zugänglich sind und digital nachhaltig gespeichert werden. Und soll bei einer eingesetzten Open Source Lösung eine Anpassung vorgenommen werden, kann dies direkt innerhalb der Community umgesetzt oder durch einen externen Anbieter realisiert werden.

Dennoch kann bei ungünstiger Auswahl einer Open Source Lösung eine ungewollte Knowhow-Abhängigkeit zu einem Open Source Anbieter geschaffen werden. Letzten Endes liegt es in der Natur der Sache, dass der Einsatz von Software Abhängigkeiten schafft. Entscheidend ist, dass die Abhängigkeiten zu einzelnen Lieferanten gering gehalten beziehungsweise bewusst gesteuert werden.

Vendor Risk Assessment:
Wir empfehlen, eine ganzheitliche Überprüfung der Abhängigkeiten zu Herstellern der eingesetzten oder einzuführenden Software vorzunehmen. Dies gilt sowohl für proprietäre Software als auch für Open Source Lösungen.

Reputationsgewinn

Der professionelle Umgang mit Open Source Software kann sich positiv auf das Image des Unternehmens auswirken.

In Fachkreisen, zunehmend aber auch im breiten Geschäftsumfeld, sind die Vorteile von Open Source Software bekannt. So erhalten Technologieunternehmen wie auch öffentliche Institutionen positive Medienpräsenz, wenn sie vormals eigene Software unter einer Open Source Lizenz veröffentlichen. Ihnen wird vorbildliches «Digital Citizenship» attestiert, das öffentlichkeitswirksam kommuniziert werden kann. Bei öffentlichen Institutionen wird oftmals durch die Politik ein vermehrter Einsatz von Open Source Software erwartet, weil dadurch lokale Innovations- und Wirtschaftsförderung betrieben und die Chancengleichheit verbessert wird.

Des Weiteren hat sich gezeigt, dass das Vertrauen von motivierten IT-Mitarbeitenden und leistungsbereiten Studierenden bezüglich Open Source Software hoch ist. Arbeitgeber gewinnen an Attraktivität, da der professionelle Umgang mit Open Source Software anspruchsvolle Positionen für qualifizierte Fachkräfte schafft.

Rasche Verbreitung

Software lässt sich unter einer Open Source Lizenz rascher verbreiten als unter einer proprietären Lizenz.

Die rasche und ungehinderte Verbreitung von Open Source Software verursacht positive Netzwerkeffekte für die initialisierende Organisation. Ist eine Software-Lösung als de facto Standard etabliert, können beispielsweise kostenpflichtige Erweiterungen oder professionelle Wartungsleistungen angeboten werden. Das Unternehmen oder die öffentliche Institution, welche die Open Source Plattform lanciert hat und massgeblich weiterentwickelt, besitzt das Knowhow und den Ruf, hochqualifizierte Dienstleistungen rund um die Software zu erbringen.

Durch geschickte Beeinflussung eines Open Source Projekts wächst dessen Verbreitung rasch und gewinnt an Marktdurchdringung. So kann sich ein Unternehmen durch die Initialisierung und Koordination eines Open Source Projekts auch in einem gesättigten Markt etablieren. Durch erfolgreiches Communitiy-Management können neue Anwender und Entwickler der Open Source Software gewonnen werden, die Zusammenarbeit wird verstärkt und die Verbreitung und der Nutzen der Open Source Plattform steigen (siehe Abschnitt «Erfolgreiches Community Management», Seite 23).

Open Source Marketing:
Sowohl für Unternehmen als auch für öffentliche Institutionen empfehlen wir, Open Source Aktivitäten professionell und breitenwirksam zu kommunizieren, um den beabsichtigten Reputationsgewinn zu erzielen.

Open Source Project Management:
Wir empfehlen Firmen und Institutionen, eigens gestartete Open Source Projekte gezielt zu steuern und intensives Community Building zu betreiben. Wichtig dabei ist die Beibehaltung der Balance zwischen Kontrolle und Offenheit mittels geeigneter Community Governance.

Risikominimierung: Good Practices

Verletzung der Open Source Lizenzbestimmungen

Der Umgang mit Open Source Lizenzen ist komplex, kann aber durch geeignete Richtlinien für die Software-Entwickler kontrolliert werden.

Durch die Vielzahl existierender Lizenzbestimmungen, die sich in wesentlichen Bereichen oder auch nur in Nuancen unterscheiden, ist die rechtliche Einhaltung aller Lizenzregelungen der eingesetzten Open Source Software eine Herausforderung. Wird Open Source Quellcode unter einer bestimmten Lizenz unkontrolliert in eine eigene Software integriert und anschliessend ausserhalb der Organisation verteilt, kann dies dazu führen, dass der gesamte Programmcode der internen Software veröffentlicht werden muss. Andernfalls riskiert die Organisation eine Klage wegen Nichteinhaltung der Lizenzbestimmungen.

Deshalb ist es sowohl bei der Software-Beschaffung als auch bei der Entwicklung und Software-Verteilung wichtig zu wissen, welches die relevanten Detailbestimmungen und jeweiligen Konsequenzen bei der Wahl einer bestimmten Open Source Lizenz sind.

Unkontrollierter Einsatz von Open Source Software

Obwohl Open Source Produkte lizenzkostenfrei sind, kann deren unkontrollierter Einsatz unerwartete Kosten generieren.

Weil Open Source Software gratis aus dem Internet geladen werden kann, wird oftmals die Einkaufsabteilung umgangen und Open Source Technologien ohne Prüfung durch Beschaffungsgremien und gegenüber Architekturvorgaben eingesetzt. Dadurch kann neben dem rechtlichen Risiko auch ein Wildwuchs an eingesetzten Produkten und Komponenten entstehen, der längerfristig schwierig wartbar ist und zu Mehrkosten führt. Wichtig ist deshalb der kontrollierte Einsatz von Open Source Software, damit IT-Governance Richtlinien sowie Vorgaben bezüglich Unternehmensarchitektur eingehalten werden.

Pro Anforderungsgebiet bestehen meist verschiedene Open Source Alternativen, die mittels systematischer Evaluation miteinander verglichen werden sollten. Dabei spielen einerseits Kriterien aus der herkömmlichen Software-Beschaffung wie Funktionalität, Benutzerfreundlichkeit, Stabilität oder Sicherheit eine wichtige Rolle. Andererseits sind auch Open Source spezifische Charakteristiken wie Open Source Lizenz, Community-Grösse, Entwicklerheterogenität, Support-Verfügbarkeit, Releasemanagement oder rechtliche Trägerschaft von hoher Relevanz, um die Qualität einer Open Source Lösung zu bestimmen.

Software Licensing Compliance:
Bei Einsatz, Integration und Freigabe von Open Source Software empfehlen wir durch erfahrene Juristen prüfen zu lassen, welche Risiken durch die betroffenen Lizenzen bestehen und wie deren Bedingungen eingehalten werden müssen.

Quality Assessment Framework:
Wir empfehlen ein einheitliches, auf die Eigenschaften von Open Source sowie die Anforderungen der Organisation abgestimmtes Evaluationsvorgehen für die Auswahl geeigneter Open Source Lösungen.

Software Development Guidelines:
Damit Software-Entwickler auf die unterschiedlichen Herausforderungen sensibilisiert werden, empfehlen wir die Erarbeitung von Richtlinien betreffend Entwicklung und Integration von Open Source Komponenten.

Enterprise Architecture Governance:
Um eine langfristig wartbare Unternehmensarchitektur zu gewährleisten, empfehlen wir die Erarbeitung von Richtlinien sowie einer Zielarchitektur bezüglich Open Source Technologien, unter anderem mit einer Whitelist geprüfter Open Source Lösungen.

Support nicht gesichert

Wartung und Support eines Open Source Produkts sind nicht immer gesichert.

Obwohl es auch viele Mischformen gibt, kann grundsätzlich zwischen zwei unterschiedlichen Arten von Open Source Projekten unterschieden werden: 1) den von Organisationen geleiteten und 2) den durch dezentralisierte Entwickler (Communities) koordinierten Projekten.

1) Bei institutionellen Open Source Projekten hat ein Unternehmen oder eine öffentliche Stelle eine Software unter einer Open Source Lizenz freigegeben und veröffentlicht regelmässig neue Versionen. Die Mitarbeitenden der Organisation besitzen als Urheber der Lösung das nötige Knowhow zur Fehlerbehebung und Weiterentwicklung der Software. Erfährt die Software eine hohe Verbreitung, bieten mit der Zeit auch andere Unternehmen Dienstleistungen für die Open Source Software an. Die Erbringung von Support-Services stellt dabei ein typisches Geschäftsmodell mit Open Source Software dar. Mittels entsprechender Service Level Agreements oder Abonnements (Subscriptions) werden wie bei proprietärer Software Wartungsleistungen mit bestimmten Reaktionszeiten angeboten. Mehr dazu im Kapitel «3. Professioneller Einsatz von Open Source» ab Seite 12.

2) Anders verhält es sich bei Community getriebenen Projekten. Diese werden von einer Vielzahl Freiwilliger entwickelt und koordiniert. Fehlerkorrekturen, Releasemanagement, Dokumentation und andere Tätigkeiten werden oft dezentral von unterschiedlichen Akteuren ausgeführt. Mittelfristig wird typischerweise eine Stiftung oder ein Verein (Foundation oder Association) als rechtliche Mantelorganisation gegründet. Der Support durch solche Communities ist kosteneffizient und meist zeitnah. Allerdings werden ohne Leistungsvereinbarungen keine Garantien abgegeben. Es bedarf deshalb der eingehenden Kenntnis der jeweiligen Community, um die Sicherstellung des Supports einschätzen zu können. Eine Möglichkeit zur Support-Sicherung ist die Anstellung von Software-Entwicklern aus der jeweiligen Open Source Community.

Support Assessment:
Wir empfehlen, Support und langfristige Weiterentwicklung einer Open Source Lösung eingehend zu prüfen und gegebenenfalls mit der Beschaffung von Service Level Agreements oder Anstellung von Kernentwicklern sicherzustellen.

Migration auf Open Source misslingt

Migrationen auf Open Source Lösungen können an mangelnder Anwenderakzeptanz oder auch an technischen Abhängigkeiten scheitern.

Wie beim Wechsel auf neue proprietäre Plattformen ist auch die Migration auf Open Source Lösungen eine Herausforderung, die nicht unterschätzt werden darf. Allerdings treten bei der Einführung von Open Source Software neben den üblichen Migrationshürden noch zusätzliche Risiken auf.

Einerseits sind die heutigen proprietären Plattformen oft stark in andere, geschlossene Systeme integriert. Solche technischen Abhängigkeiten, Inkompatibilitäten, unzugänglichen Daten oder nicht vorhandenen beziehungsweise inkompatiblen Schnittstellen können verursachen, dass Open Source Lösungen schlecht zu integrieren sind. So werden Migrationen auf Open Source Software verzögert oder müssen gar vollständig abgebrochen werden. Ein wichtiger Lösungsansatz ist somit das hybride Vorgehen, welches die Kombination von Open Source und proprietärer Software vorsieht.

Andererseits stellen Unsicherheit und Unwissen der Endanwender zusätzliche Risiken dar. Die Benutzer sind oftmals bekannte Software-Marken und proprietäre Software-Produkte gewohnt und kennen deshalb allfällige Open Source Alternativen nicht. Funktional ebenbürtige Open Source Lösungen sind meist nicht stark beworben und verbreitet. Dies kann dazu führen, dass die Anwenderakzeptanz ohne gezielte Gegenmassnahmen bei der Einführung von Open Source Software gering ist. Somit sind eine intensive Informationspolitik und Sensibilisierung bezüglich Open Source sowie Spezialausbildungen für Power User und andere Schlüsselpersonen wichtig.

Integration Strategy:
Wir empfehlen die Entwicklung einer pragmatischen und sanften Integrationsstrategie, die unter anderem gezielte Pilotversuche, hybride Vorgehensweisen sowie intensive Benutzerschulungen und Sensibilisierungsmassnahmen vorsieht.

Good Practice: Open Source Strategie

Eine integrierte Open Source Strategie unterstützt nachhaltigen Erfolg in der IT.

Die beschriebenen Vorteile, Risiken und Leading Practices zeigen auf, dass Einführung und Wartung von Open Source Software viele Bereiche in der Informatik tangiert. Vorgehen in eine Richtung können Konsequenzen in einem anderen Gebiet auslösen. Auch verursachen Migrations- und Lösungsentscheidungen oftmals langfristige Auswirkungen, die gut geprüft und geplant werden müssen, um erfolgreich zu sein.

Damit alle Ziele, Regelungen und Massnahmen betreffend Open Source Software aufeinander abgestimmt und innerhalb der Gesamt-IT-Strategie eingebettet sind, entwickeln immer mehr Unternehmen und öffentliche Institutionen eine explizite Open Source Strategie. Diese beinhaltet alle relevanten Bereiche, die im Zusammenhang mit Einführung, Nutzung und Freigabe von Open Source Software stehen.

Typische Themengebiete einer Open Source Strategie sind in nebenstehender Checkbox aufgelistet.

Mögliche Elemente einer Open Source Strategie

- ☐ Vorteile und Risiken von Open Source Software bezogen auf die Organisation
- ☐ Generelle Richtlinien und Ziele zu Open Source Software
- ☐ Vorgaben zur Reduktion von Abhängigkeiten zu proprietärer Software
- ☐ Berücksichtigung von Open Source Lösungen bei Software-Beschaffungen
- ☐ Vorgaben bezüglich Open Source Lizenzen, beispielsweise Pure Open Source oder auch Open Core
- ☐ Kriterien zur Evaluation von Open Source Software
- ☐ Regelung betreffend Freigabe von Open Source Software
- ☐ Aus- und Weiterbildung bezüglich Open Source Technologien, Organisation, Lizenzen etc.
- ☐ Umsetzungsmassnahmen der Strategieziele wie Studie, Pilotprojekte, Kompetenzstelle etc.

3. Professioneller Einsatz von Open Source

Der Einsatz von bereits auf dem Markt vorhandener Open Source Software kann grundsätzlich auf drei Arten erfolgen: Entweder wird Open Source Software kostenlos aus dem Internet heruntergeladen, installiert und so wie sie ist eingesetzt (Szenario 1). Oder ein Unternehmen beziehungsweise eine öffentliche Institution baut intensiv Knowhow und Ressourcen zu bestimmten Open Source Produkten auf um diese in geschäftskritischen Bereichen einzusetzen (Szenario 2). Oder ein externer Open Source Anbieter wird beigezogen, der die Einführung und Wartung der Open Source Software professionell begleitet (Szenario 3). Welches dieser Szenarien im individuellen Fall zielführend ist, muss situativ entschieden werden. Eine Orientierungshilfe bietet untenstehende Tabelle.

	1. Einsatz ohne professionellen Support	2. Einsatz mit internem Support	3. Einsatz durch externen Anbieter
Szenario	Open Source Software gratis aus dem Internet laden und so wie sie ist einsetzen	Interner Aufbau von Knowhow und Ressourcen zu bestimmten Open Source Lösungen, um diese intensiv und langfristig einzusetzen	Unterstützung durch einen externen Open Source Anbieter, um Open Source Software rasch zu integrieren und anzupassen
Einsatzbereich	Nicht geschäftskritische Bereiche	Geschäftskritische Bereiche und wettbewerbsdifferenzierende Technologien	Geschäftskritische Bereiche, in denen unmittelbar vertieftes Knowhow der Software verfügbar sein muss
Zielgruppe	Private, kleine und mittlere Unternehmen, kleine Schulen, Non-Profit Organisationen	Grossunternehmen, öffentliche Verwaltungen, grosse Institutionen	Grossunternehmen, öffentliche Verwaltungen, grosse Institutionen
Vorteile	▸ Niedrige Kosten ▸ Rasche Umsetzung	▸ Hohe Flexibilität dank internem Knowhow ▸ Keine Anbieterabhängigkeiten	▸ Direkter Zugang zum Knowhow der Open Source Entwickler ▸ Korrekturen und Weiterentwicklung auf Auftragsbasis ▸ Auswahl verschiedener Open Source Anbieter ▸ Weitere Vorteile gemäss Service Level Agreement (siehe Abschnitt «Service Level Agreements für Open Source Lösungen», Seite 13)
Nachteile	▸ Kein garantierter Support ▸ Keine Haftungsansprüche	▸ Hohe Investitionen und grosser Zeitaufwand durch Knowhow-Aufbau ▸ Höhere interne Fixkosten durch mehr Mitarbeitende ▸ Keine Zertifizierungen für Hardware und Software	▸ Externe Kosten durch Open Source Anbieter ▸ Knowhow-Abhängigkeit zum Open Source Anbieter
Risiko und Absicherung	Hohes Risiko: Es bestehen keinerlei Support-Verträge oder Garantien	Mittleres Risiko: Der Support hängt von Knowhow und Verfügbarkeit der internen IT ab	Niedriges Risiko: Gewährleistung geschieht gemäss Auftragsbeschreibung oder Service Level Agreement

Über Open Source Anbieter

Als Open Source Anbieter werden Firmen bezeichnet, die selber eine Open Source Lösung lanciert haben oder zumindest massgeblich an deren Weiterenwicklung beteiligt sind. Beurteilt werden kann dies anhand der Abklärung, ob Software-Entwickler des Unternehmens einen so genannten «Committer»-Status haben. Das heisst, diese Programmierer können selbständig Änderungen am zentral verwalteten Quellcode vornehmen. Dies deutet an, dass das Unternehmen über das notwendige Knowhow verfügt, die Lösungen mit regelmässigen Sicherheitsaktualisierungen und Upgrades zu warten und auf neue Bedürfnisse anzupassen. Open Source Firmen bieten kompetente Beratung für ihre Produkte an, realisieren Software-Einführungen und Migrationen, entwickeln Verbesserungen und Erweiterungen und führen Mitarbeiterschulungen durch. Insbesondere garantieren Open Source Anbieter mittels Support-Verträgen während einer verbindlich deklarierten Zeitspanne die Unterstützung für eine bestimmte Software-Version. Damit kann die Open Source Lösung stabil betrieben werden und ist langfristig wartbar. Solche und weitere Dienstleistungen ermöglichen diesen Unternehmen wiederum die grundlegende Weiterentwicklung der Open Source Software und sichern damit das nachhaltige Bestehen eines Open Source Projekts.

Service Level Agreements für Open Source Lösungen

Bedeutend ist die Absicherungsfunktion, die Open Source Anbieter erbringen. Mittels klar definierter Service Level Agreements und Subscriptions erlangen Open Source Lösungen die für den geschäftskritischen Einsatz geforderten Garantieleistungen und Risikoabsicherungen. Mögliche Leistungen der Service Level Agreements von Open Source Anbietern sind in folgender Checkbox aufgeführt.

Mögliche Leistungen von Open Source Anbietern

- ☐ Direkter Zugang zum Knowhow der Kernentwickler der Open Source Software
- ☐ Definierte Antwortzeiten für Support-Anfragen
- ☐ Support über verschiedene Kanäle (Web, VPN, Email, Chat, Telefon, Remote-Desktop, vor Ort)
- ☐ Angebot von professionellen Dokumentationen, Schulungen und Zertifizierungskursen
- ☐ Zeitnahe, pro-aktive und kundenfreundliche Auslieferung von Security Patches
- ☐ Mindestdauer für Wartung und Support von bestimmten Software-Versionen
- ☐ Zugesicherte, regelmässige Software Releases und Updates
- ☐ Garantie für Kompatibilität mit anderen Software-Lösungen
- ☐ Zertifizierungen für bestimmte Hardware und proprietäre Software-Systeme
- ☐ Integration von Korrekturen und Erweiterungen in die Hauptentwicklungsversion
- ☐ Absicherung gegen Rechtsansprüche am geistigen Eigentum (Copyright, Patente)
- ☐ Haftung bei Unterbrüchen und Fehlfunktionen
- ☐ Bereitstellung zusätzlicher proprietärer Erweiterungen und Hilfswerkzeuge

Open Source versus Open Core

In den letzten Jahren haben sich mehrere Hybrid-Geschäftsmodelle etabliert, bei denen Aspekte von Open Source und proprietärer Lizensierung auf unterschiedliche Art und Weise miteinander verbunden werden. Ein solches hybrides Modell ist Open Core, das sich von klassischen proprietären Lizenzmodellen hauptsächlich durch den Grad der «Beimischung» und Freigabe von Open Source Software unterscheidet.

In einem Open Source Modell ist der Quellcode der gesamten Software-Lösung unter einer offiziellen Open Source Lizenz veröffentlicht oder zumindest für die Kunden verfügbar. Damit kann der Kunde auch nach Ablauf eines Service Level Agreements die Software-Lösung unbegrenzt und zu jedem Zweck weiternutzen. Als Geschäftsmodell hat der Open Source Anbieter dabei verschiedene Dienstleistungsoptionen, die dem Kunden Mehrwert verschaffen. So erlauben Beratung, Integration, Schulung, Weiterentwicklung, Unterhalt, Garantien, Betrieb und langfristige Betreuung eine grosse Zahl an Dienstleistungsmöglichkeiten rund um eine Open Source Lösung. Gleichzeitig wird die Anbieterabhängigkeit, der so genannte Vendor Lock-in, minimiert, denn der Kunde kann den Open Source Anbieter aus rechtlicher Sicht beliebig wechseln. Bei weit verbreiteter Open Source Software ermöglicht das Open Source Modell die Umstellung auf einen anderen, kompetenten Anbieter zu niedrigen Kosten. Voraussetzung für einen erfolgreichen Open Source Anbieterwechsel ist, dass der neue Anbieter die Software-Lösung gut kennt und möglichst eigene Entwickler angestellt hat, die Software-Änderungen in die Open Source Community zurückspeisen können (siehe vorherigen Abschnitt «Über Open Source Anbieter», Seite 13). Aber auch bei selten eingesetzten Open Source Produkten oder Open Source Individuallösungen ist die Einarbeitung eines neuen Open Source Anbieters meist deutlich kosteneffizienter als eine vollständige Neuanschaffung, Migration und Umschulung der Mitarbeiter.

Beim Open Core Modell veröffentlicht der Hersteller nur einen Teil der Software unter einer Open Source Lizenz, stellt jedoch die übrigen Bereiche der angebotenen Software unter eine proprietäre Lizenz und verkauft entsprechende Lizenzverträge. Die proprietären Bestandteile umfassen üblicherweise für den Betrieb im professionellen Umfang wichtige Funktionen. Die Open Source Version wird dabei typischerweise «Community Edition» oder ähnlich genannt, die Version mit proprietärer Software häufig «Enterprise Edition». Das Geschäftsmodell ist dabei im wesentlichen identisch mit anderen proprietären Modellen und gehört nach keiner verbreiteten Definition zum Bereich der Open Source Software. Damit verschliesst Open Core den Kunden die wesentlichen Vorteile von Open Source wie Unabhängigkeit und Offenheit. So ist der Wechsel des Dienstleistungsanbieters normalerweise nicht ohne eine Migration auf eine andere Software-Lösung möglich.

	Open Source Modell	**Open Core Modell**
Community Version der Software-Lösung	Der vollständige Software-Quellcode ist unter einer Open Source Lizenz veröffentlicht und externe Beiträge können ungehindert integriert werden.	Der Kern der Software ist unter einer offiziellen Open Source Lizenz veröffentlicht, es fehlen jedoch kritische Funktionen und Erweiterungen.
Enterprise Version der Software-Lösung	Für diese kommerzielle Open Source Software wird ein kostenpflichtiges Service Level Agreement mit einem Hersteller abgeschlossen, aber der vollständige Software-Quellcode ist unter einer Open Source Lizenz erhältlich.	Die Software wird kostenpflichtig unter einer proprietären Lizenz ohne Zugang zum Quellcode und den mit Open Source Software verbundenen Rechten erworben.

4. Rechtliche Aspekte von Open Source

Wesentliche Fragen im Zusammenhang mit Open Source

In der Praxis stellen sich im Zusammenhang mit Open Source oft diverse Fragen mit rechtlichem Bezug: Wodurch unterscheiden sich die Open Source Lizenzen? Welche rechtlichen Bedingungen und Konsequenzen hat der Einsatz von Open Source Software? Wie beeinflusst Open Source Software das Lizenz-Management? Welche Lizenz eignet sich für die Veröffentlichung eigener Software?

Die Beantwortung der Fragen ist zentral für den professionellen Einsatz und die nachhaltige Entwicklung von Open Source Software. Deshalb führt der folgende Abschnitt den Einfluss des Urheberrechts auf Open Source Software aus, charakterisiert die wichtigsten Open Source Lizenzen, erläutert detaillierte Abgrenzungen und Unterscheidungen, beschreibt den Begriff «Dual Licensing» und zeigt die Verbreitung verschiedener Open Source Lizenzen auf.

Open Source und Urheberrecht

Der Begriff «Open Source» wird im Gesetz nicht erwähnt und besagt lediglich, dass der Quellcode einer Software offen zugänglich ist. Open Source Software unterliegt jedoch uneingeschränkt allen gesetzlichen Bestimmungen. Insbesondere gilt dies auch für das Urheberrecht. Computer-Software ist in aller Regel urheberrechtlich geschützt. Die weit verbreitete Annahme, der Hersteller von Open Source Software gäbe sein Urheberrecht auf, ist falsch.

Das Urheberrecht räumt seinem Inhaber das ausschliessliche Nutzungsrecht an der Software ein und damit auch das Recht, Dritten die Nutzung zu verbieten oder zu erlauben. Die vertragliche Erlaubnis zur Nutzung einer urheberrechtlich geschützten Software wird als «Lizenz» bezeichnet. Mit der Erteilung einer Lizenz sind in der Regel Rechte und Pflichten verbunden. Bei proprietärer Software besteht meist die Pflicht zur Bezahlung einer Lizenzgebühr. Dem Urheberrechtsinhaber steht es frei, die Erteilung einer Lizenz auch von anderen Auflagen und Bedingungen abhängig zu machen. Dies ist bei Open Source Lizenzen typischerweise der Fall, denn sie enthalten meist Auflagen oder Bedingungen zu einem der folgenden Themenkreise:

- Veränderungen am Quellcode
- Einbettung des Quellcodes in fremde Software
- Pflicht zur Offenlegung des veränderten Quellcodes
- Weiterverbreitung des veränderten Quellcodes

Überblick über die wichtigsten Open Source Lizenzen

Open Source Lizenzen existieren in den verschiedensten Ausführungen und lassen sich nach den Lizenzbedingungen unterscheiden. Für den Nutzer von Open Source Software ist es entsprechend wichtig zu wissen, unter welcher Lizenz die Software veröffentlicht wurde und welche spezifischen Auflagen folglich zu beachten sind. Sämtliche hier beschriebene und eine Reihe weiterer Lizenzen finden sich im Originalwortlaut auf der Website der Open Source Initiative.

GNU General Public License (GPL)

Die GNU General Public License (GPL) stellt die verbreitetste Open Source Lizenz dar. Sie wurde erstmals im Jahr 1989 von der Free Software Foundation herausgegeben und 1991 beziehungsweise 2007 in den Versionen 2 beziehungsweise 3 erneuert. Die GPL gewährt vier Freiheiten: Erlaubt sind explizit die uneingeschränkte Nutzung der Software und deren kostenlose Vervielfältigung, der freie Zugang zum Quellcode und dessen veränderte Weiterverbreitung. Einzigartig an der GPL ist ihr sogenannter viraler Effekt. Er bewirkt, dass veränderte GPL-Komponenten nur dann weitergegeben werden dürfen, wenn ihr Quellcode verfügbar gemacht und ebenfalls der GPL unterstellt wird. Diese erzwungene Veröffentlichung stellt einerseits einen so genannten «starken Schutz» der Freiheit dar, der Weiterentwicklungen von einmal veröffentlichter GPL-Software für immer frei verfügbar hält. Andererseits stellt dieser auch als «Copyleft» bezeichnete Mechanismus ein rechtliches Risiko für Unternehmen dar, da mit GPL-Code kombinierte Eigenentwicklungen mitsamt Quellcode unter der GPL veröffentlicht werden müssen. In dieser Hinsicht unterscheidet sich die GPL von andern Lizenzen dadurch, dass die gesamte Software und nicht nur einzelne Komponenten in Betracht gezogen werden. Die Betrachtung bezieht sowohl funktionale wie auch einige wirtschaftliche Aspekte mit ein. Dies macht die GPL «strenger» und den durch die GPL bewirkten Schutz der Freiheiten «stärker».

GNU Library or Lesser General Public License (LGPL)

Die GNU Library oder Lesser General Public License (LGPL) wurde ebenfalls 1991 von der Free Software Foundation parallel zur GPL als Version 2 herausgegeben. 1999 wurde sie in der Version 2.1 leicht überarbeitet und anschliessend 2007 gleichzeitig zur GPL als Version 3 veröffentlicht. Die LGPL wurde im Hinblick auf die in der Software-Entwicklung stark verbreitete Nutzung von Programmbibliotheken entworfen. Solche Software-Komponenten, so genannte Libraries, sind explizit für die Wiederverwendung in Software-Applikationen konzipiert. Um dabei nicht wie die GPL eine Veröffentlichung des gesamten Quellcodes der «Mantel»-Applikation zu erzwingen, dürfen nun unter der LGPL lizenzierte Komponenten auch weiterhin von proprietärer Software angewendet werden. Allerdings müssen Veränderungen an der LGPL-Bibliothek selber, identisch zur GPL, wiederum unter der LGPL veröffentlicht werden. So wird bei der LGPL von einem «schwachen Schutz» der Freiheit gesprochen.

GNU Affero General Public License (AGPL)

Die GNU Affero General Public License (AGPL) wurde zuletzt 2007 von der Free Software Foundation in Zusammenarbeit mit der Firma Affero in der Version 3 veröffentlicht. Die Lizenz basiert auf der GPL Version 3 und hat zum Ziel, eine Art «Schlupfloch der Freiheit» bei Application Service Providern (ASP) zu schliessen. Denn wird GPL lizenzierte Software auf einem Webserver betrieben, erhalten die Anwender Zugang zur Applikationsausgabe, nicht aber zu den ausführbaren Dateien. Deshalb ist der ASP nicht verpflichtet, seine Weiterentwicklungen an der GPL-Software zu veröffentlichen, denn sie wird physisch nicht an die Kunden verteilt. Anders ist dies bei AGPL lizenzierter Software, die auch bei Nutzung über Netzwerkverbindungen als Quellcode freigegeben werden muss. Mit Blick auf diese Eigenschaft könnte die AGPL in Zukunft für Konzepte wie Cloud Computing und Software as a Service immer wichtiger werden.

MIT-, BSD- und Apache Licenses

Sogenannte liberale Open Source Lizenzen wie die MIT License, die BSD License und die Apache License stammen zwar aus verschiedenen Open Source Communities, unterscheiden sich inhaltlich aber nur im Detail. Im Gegensatz zu den genannten Lizenzen der Free Software Foundation gewähren die liberalen Lizenzen die Möglichkeit, den Quellcode direkt in proprietäre Software einzubinden. Einzige Bedingung ist oftmals die Angabe des Copyrights oder die Mitlieferung des Lizenztextes, sodass die Open Source Programmbibliotheken als eigenständige Komponenten in der proprietären Software namentlich aufgeführt werden müssen.

Mozilla Public License (MPL)

Die Mozilla Public License (MPL) stellt ähnlich zur LGPL eine Lizenz mit schwachem Schutz der Freiheiten dar. Die MPL ist am konsequentesten als File-basierte Lizenz ausgestaltet. Allerdings ist sie als eine der wenigen liberaleren Lizenzen nicht kompatibel mit der GPL. Das heisst, dass MPL lizenzierter Code nicht in GPL-Software integriert werden darf.

Abgrenzung von weiteren Software-Kategorien

Im Zusammenhang mit Open Source tauchen oft weitere Begriffe wie Public Domain Software, Freeware und Shareware auf, was zu Missverständnissen führen kann. Eine Abgrenzung ist nicht immer einfach, da für die genannten Begriffe keine allgemein verbindlichen Definitionen bestehen. Die folgenden Umschreibungen halten einige Begriffsmerkmale fest.

Public Domain Software

Bei Public Domain Software bestehen keinerlei Auflagen oder Bedingungen im Zusammenhang mit der Nutzung. Darüber, ob der Quellcode der Software offen zugänglich ist, sagt der Begriff nichts aus. Dies kann, muss aber nicht der Fall sein.

Freeware

Freeware darf ohne oder gegen freiwillige Bezahlung genutzt werden. Die Nutzung kann aber mit anderen Auflagen und Bedingungen verknüpft sein. Darüber, ob der Quellcode der Software offen zugänglich ist, sagt der Begriff nichts aus. Dies kann, muss aber nicht der Fall sein.

Shareware

Shareware bezeichnet meist proprietäre Software, welche gratis und mit eingeschränkter Funktionalität zu Testzwecken genutzt werden darf. Der Quellcode von Shareware ist üblicherweise nicht offen zugänglich.

Unterscheidung der wichtigsten Open Source Lizenzen

Untenstehende Tabelle stellt die Charakteristika der genannten Lizenzen in einer Übersicht dar.

Die Restriktionen sind tendenziell von unten nach oben beziehungsweise von links nach rechts abnehmend.

	Starker Schutz der Freiheiten		**Schwacher Schutz der Freiheiten**	**sog. Liberale Open Source Lizenzen**	
	AGPLv3	*GPL (v2 und v3)*	*LGPLv3*	*Apache License 2.0*	MIT License und BSD License
Freier Zugang zum Quellcode	**ja**	**ja**	**ja**	**ja**	**ja**
Der Quellcode darf innerhalb der rechtlichen Einheit verändert und mit beliebiger anderer Software kombiniert werden.	**ja**	**ja**	**ja**	**ja**	**ja**
Der Quellcode darf auf Webservern verschlossen bleiben: Wird die Software nicht physisch an Kunden oder Partner verteilt, sondern wird sie ausschliesslich beispielsweise auf einem Webserver den Benutzern zur Verfügung gestellt, muss der Quellcode nicht veröffentlicht werden.	**nein**	**ja**	**ja**	**ja**	**ja**
Der Quellcode darf mit proprietärer Software verteilt werden: Solange LGPL lizenzierte Software ausschliesslich extern beispielsweise als Programmbibliothek aufgerufen wird, darf sie zusammen mit proprietärer Software verteilt werden.	**nein**	**nein**	**ja**	**ja**	**ja**
Veränderungen dürfen verschlossen bleiben: Als wesentlicher Unterschied zu den Lizenzen der Free Software Foundation (AGPL, GPL und LGPL) erlauben liberale Lizenzen, dass der Quellcode in proprietäre Software eng integriert werden darf. Verbesserungen und Erweiterungen des Quellcodes müssen somit nicht mehr frei gegeben werden, sondern dürfen verschlossen bleiben.	**nein**	**nein**	**nein**	**ja**	**ja**
Einzige Pflicht ist das Einfügen eines vorgegeben Copyright Vermerks und einer Haftungsausschlussklausel im Quellcode.	**nein**	**nein**	**nein**	**nein**	**ja**

Lizenzkompatibilität (siehe dazu nächster Abschnitt)

Open Source Lizenzen im Zusammenspiel: Kompatibilität und Dual Licensing

Die Tabelle auf Seite 17 lässt Rückschlüsse auf die Kompatibilität von Open Source Lizenzen untereinander zu. Beispielsweise darf Software unter einer Apache Lizenz in solche unter einer GPL integriert werden, nicht aber umgekehrt, da die GPL restriktiver ist als die Apache Lizenz.

Beim Dual Licensing veröffentlicht der Hersteller dieselbe Software unter zwei unterschiedlichen Lizenzen. Typischerweise wird einerseits eine restriktive Lizenz (insbesondere die GPL) und andererseits eine proprietäre Lizenz verwendet. Dadurch werden dem Hersteller die Vorteile beider Welten zuteil. Einerseits verbreitet sich die Software rascher dank der kostenlosen Open Source Version. Andererseits kann mittels Verkauf der Software unter der proprietären Lizenz Umsatz generiert werden. Kunden wiederum haben Interesse an der proprietären Lizenz, wenn sie die Software innerhalb eigener proprietärer Software weiterverwenden wollen.

Erfolgreiche Beispiele solcher Dual Licensing Modelle finden sich oft im Datenbankbereich. Der Datenbankhersteller bietet seine Software unter den zwei erwähnten Lizenzen an. Dadurch erhält die Plattform eine hohe Anwenderzahl, was sich positiv auf das Testen und die Fehlersuche und damit letztlich auf deren Stabilität auswirkt. Wenn nun ein Unternehmen aus dem Business-Applikationen-Umfeld die Datenbank in sein eigenes Software-Produkt einbauen will, kann es vom Datenbankhersteller eine proprietäre Lizenz erwerben. Dadurch erhält der Hersteller der Applikation die legale Möglichkeit, die Restriktionen der GPL-Lizenz zu umgehen, die Datenbank in die eigene Lösung zu integrieren und diese unter neuer, proprietärer Lizenz zu vertreiben.

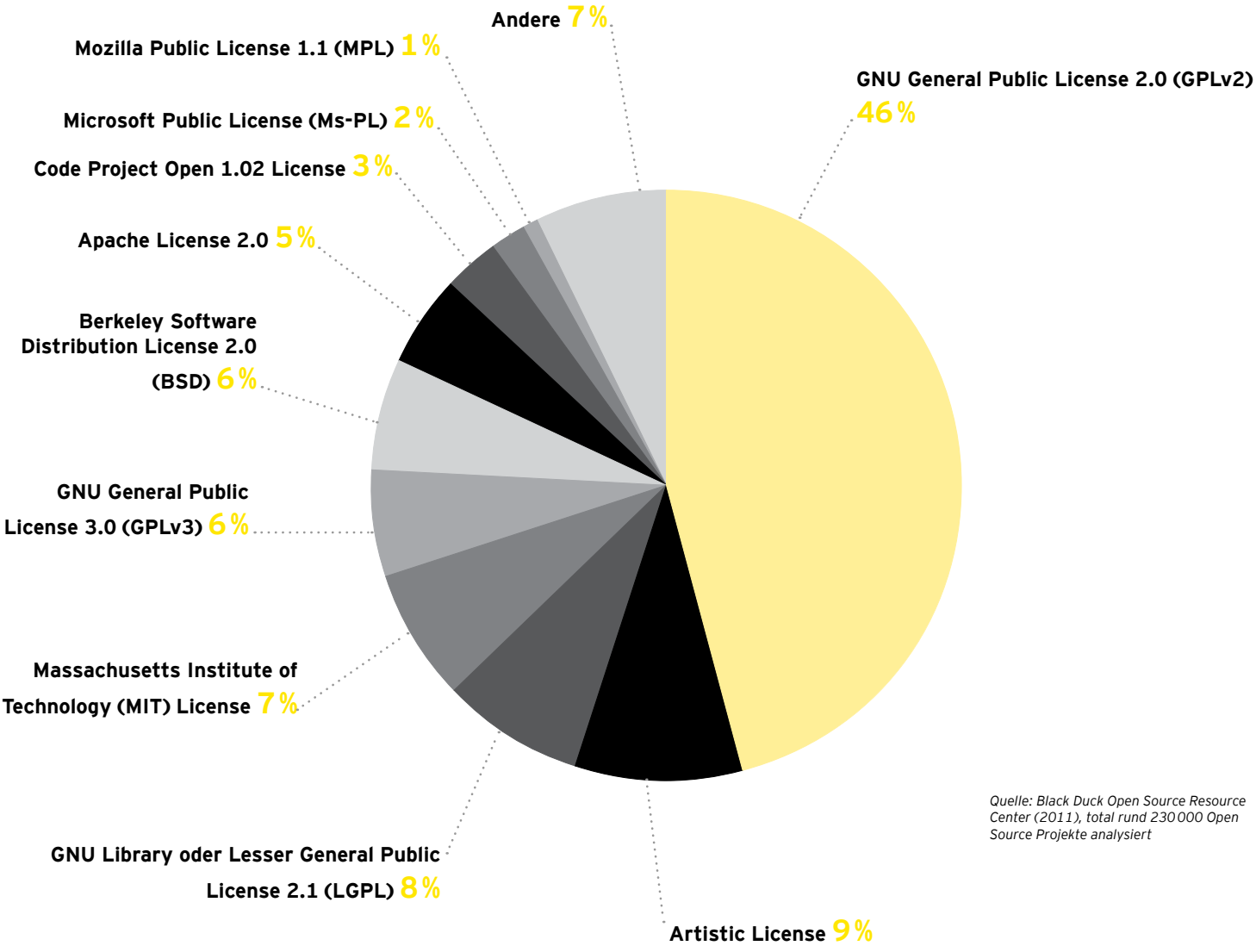
Checkbox License Compliance bei Open Source Software

- ☐ Wir wissen, ob in unserer Organisation Open Source Software eingesetzt wird
- ☐ Wir wissen, unter welcher Lizenz die betreffende Open Source Software steht
- ☐ Wir haben festgelegt, ob und wann bei uns GPL-Software in der Entwicklung verwendet werden darf
- ☐ Unsere Programmierer sind sich bewusst, welche Folgen die Integrierung von GPL-lizenzierten Komponenten in unsere Software hat
- ☐ Wenn wir eigene Software freigeben, überlegen wir uns genau, welche Open Source Lizenz wir wählen
- ☐ Regelmässige Kontrollen stellen sicher, dass unsere internen Bestimmungen über den Umgang mit Open Source Software eingehalten werden

Verbreitung von Open Source Lizenzen

Laut Black Duck Software, Inc., einem auf Open Source Compliance Management spezialisierten Unternehmen, verwendet heute gut die Hälfte der rund 230 000 Open Source Projekte die GNU General Public License Version 2 (GPLv2). Rund ein Zehntel der Projekte sind je unter der für Perl-Software typischen Artistic License und der für Software-Komponenten konzipierten LGPL veröffentlicht. Mit etwa 7 % liegt die MIT License auf Platz vier und die Version 3 der GPL auf Platz fünf mit steigender Tendenz. Weitere bekannte Open Source Lizenzen sind die BSD License, die Apache License und die Mozilla Public License.

Obwohl sie bezüglich Anzahl lizenzierter Software-Projekte eher im kleineren Prozentbereich liegen, sind viele sehr häufig eingesetzte Open Source Lösungen wie etwa der Apache Webserver oder der Webbrowser Mozilla Firefox unter diesen so genannten liberalen Lizenzen veröffentlicht. In den letzten Jahren ist die Anzahl Open Source Projekte basierend auf der Microsoft Public License gewachsen, da unterdessen auch dieser für proprietäre Software bekannte Hersteller die Entwicklung von Open Source Software unterstützt.



Quelle: Black Duck Open Source Resource Center (2011), total rund 230 000 Open Source Projekte analysiert

5. Hintergrundwissen zu Open Source Software

Einblick in die Software-Entwicklung

Um zu verstehen, welche Eigenschaften Open Source Software mit sich bringt und welche nicht, bedarf es einer kurzen Einführung in die Software-Entwicklung. Üblicherweise wird Software in Form eines von Menschen lesbaren Quelltextes entwickelt. In den meisten Programmiersprachen wird dieser Quellcode anschliessend mittels eines so genannten Compilers in einen hochperformanten, aber nur durch Prozessoren lesbaren Binär-Code umgewandelt. Aus solchen ausführbaren Dateien setzt sich das auf dem Desktop-Computer oder dem Server ausgeführte Betriebssystem sowie die eingesetzten Standardprogramme und Fachapplikationen zusammen.

Abhängigkeit von Herstellern proprietärer Software

In der Software-Branche werden üblicherweise Nutzungslizenzen für Software verkauft. Dabei wird den Kunden ausschliesslich der Binär-Code mit vielen Einschränkungen zur Nutzung zur Verfügung gestellt. Die eigentliche Software-«Bausubstanz», der Quellcode, bleibt weiterhin verschlossenes Eigentum des Software-Unternehmens. Der Software-Anwender ist somit abhängig von einer Firma und ihrer Strategie, ob und wie das Software-Produkt künftig weiterentwickelt wird. Anders ist dies bei Open Source Software, die sowohl als Binär-Code als auch mit dem zugehörigen Quellcode ausgeliefert wird und explizit das Vervielfältigen und Verändern der Software erlaubt. Zu berücksichtigen ist, dass dennoch das Urheberrecht des Software-Herstellers gilt, er eine Open Source Software also auch unter einer herkömmlichen Lizenz verkaufen kann. Dies wird wie zuvor beschrieben Dual Licensing genannt.

Definition von Open Source Software

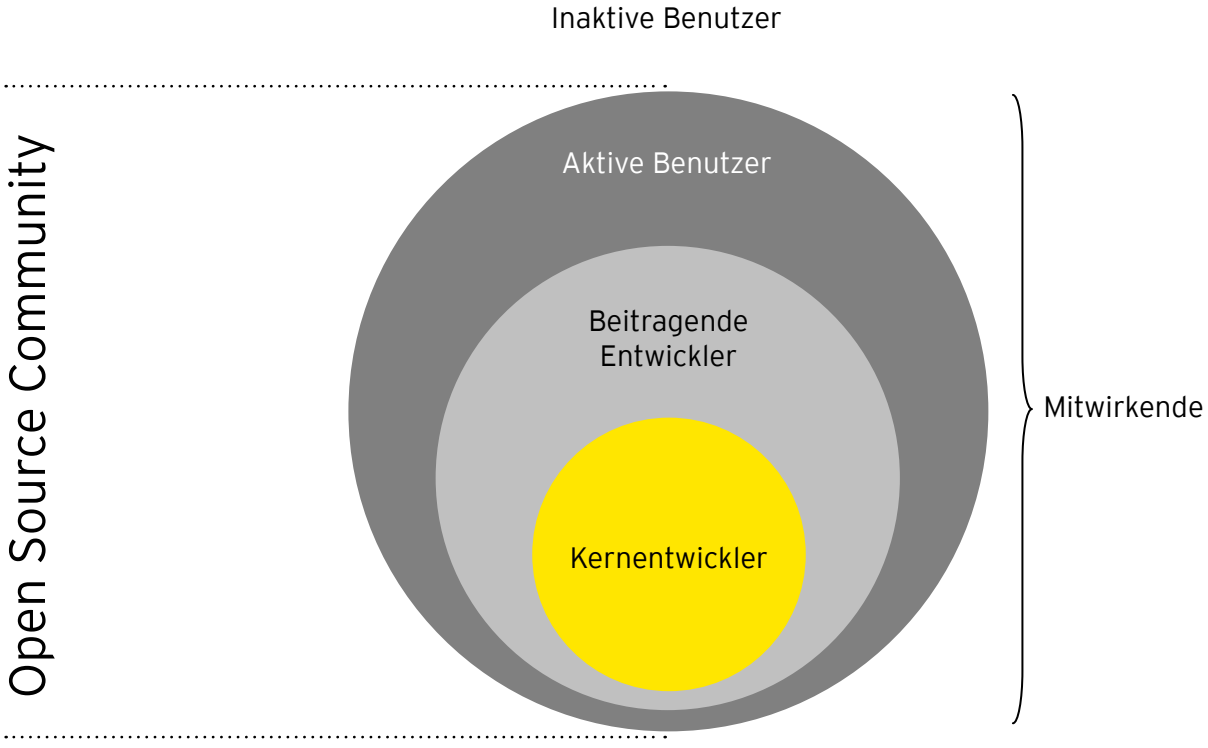
Ein Software-Produkt wird als Open Source Software bezeichnet, wenn es unter einer der rund 70 von der Open Source Initiative (OSI, www.opensource.org) abgesegneten Lizenzen veröffentlicht ist (siehe Abschnitt «4. Rechtliche Aspekte von Open Source», Seite 15). Somit stellt Open Source Software nicht in erster Linie eine Technologie oder ein Geschäftsmodell dar, sondern definiert die zentralen Eigenschaften der jeweiligen Software-Lizenz. Alle offiziellen Open Source Lizenzen geben der darunter veröffentlichten Software folgende Merkmale vor:

1. Der Quelltext der Software liegt in einer für Menschen verständlichen Form vor.
2. Die Software darf beliebig kopiert, verbreitet und genutzt werden.
3. Die Software darf verändert und in der veränderten Form weitergegeben werden.

Open Source Projekte und Communities

Wird nun eine Software-Lösung unter einer Open Source Lizenz veröffentlicht, so wird oft von einem so genannten Open Source «Projekt» gesprochen. Dieser Begriff deutet an, dass beim Open Source Entwicklungsmodell nicht nur die Software wichtig ist, sondern dass sich auch eine lebhafte Community bildet, die nach eigenen Regeln und Strukturen den Quellcode weiterentwickelt. Es ist wichtig zu berücksichtigen, dass ein Open Source Projekt – im Gegensatz zu der Definition «Projekt» im deutschen Sprachgebrauch – keinen fixen Start- und Endtermin besitzt, sondern so lange als aktiv bezeichnet wird, wie die Software weiterentwickelt wird.

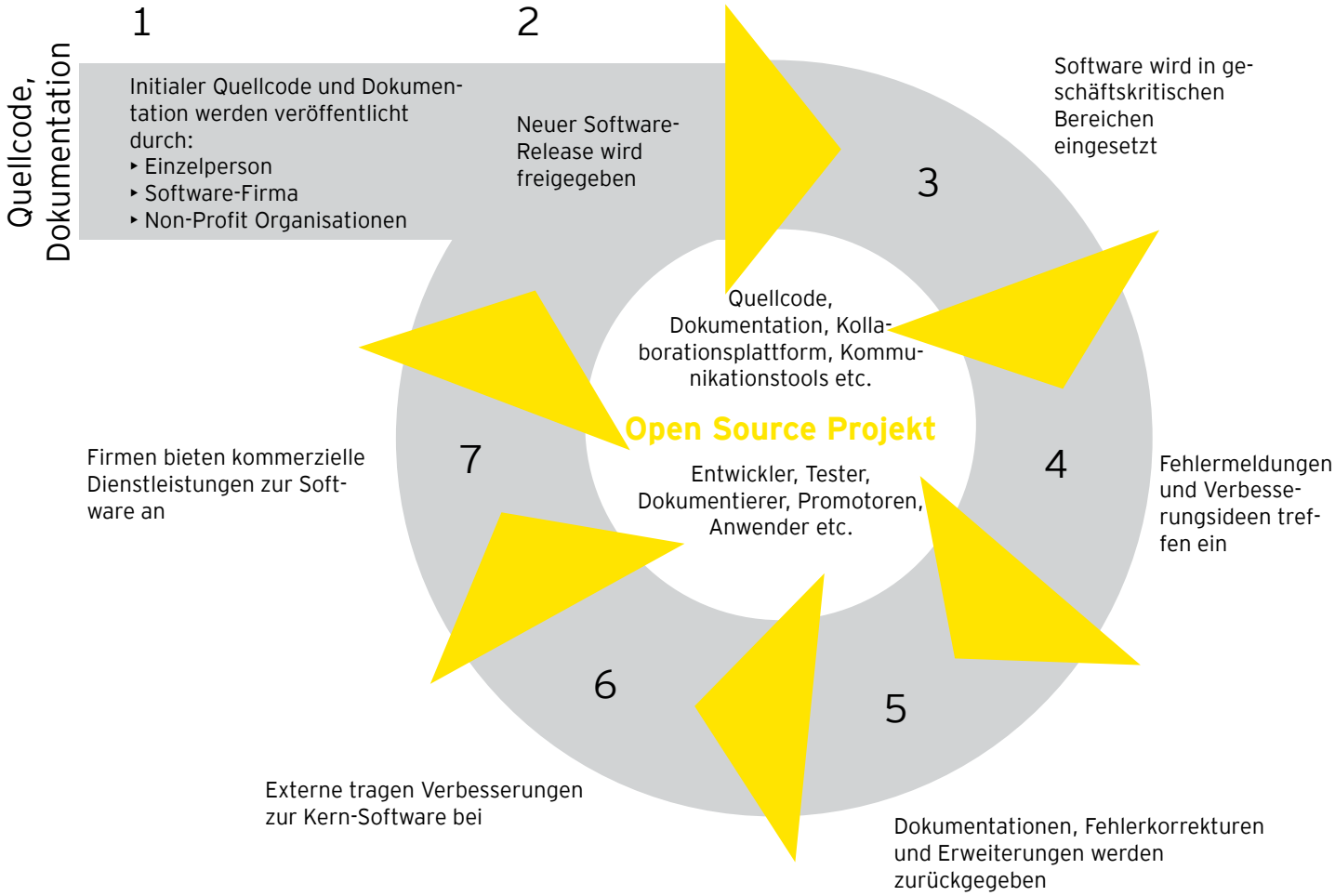
Typischerweise sind Open Source Communities schalenförmig zusammengesetzt: Zu innerst sind die Kernentwickler, eine oftmals kleine Gruppe von beispielsweise 10 bis 20 Software-Entwicklern, die den Grossteil der Software programmiert haben und die grundlegende Weiterentwicklung vorantreiben. Sie besitzen Committer-Rechte beim Open Source Projekt, können also jederzeit alle Bereiche des Quellcodes beliebig anpassen. Etwas weiter aussen sind die beitragenden Software-Entwickler. Diese sind für eine bestimmte periphäre Software-Komponente verantwortlich, programmieren neue Erweiterungen oder korrigieren Fehler und Sicherheitslücken im Quellcode. Diese Personen können typischerweise punktuell den zentralen Quellcode verändern. Noch weiter aussen sind die aktiven Benutzer, welche die Open Source Lösung selber einsetzen und gleichzeitig aktiv in der Community mitwirken, beispielsweise durch Dokumentationsbeiträge, Übersetzungen, Fehlerberichterstattung oder Beantwortung von Support-Anfragen. Diese Anwender haben normalerweise keinen Schreibzugang zum zentralen Repository und dem darin befindlichen Quellcode. Ausserhalb der eigentlichen Open Source Community befinden sich schliesslich die inaktiven Anwender, welche die Software lediglich benutzen, aber keinerlei Beiträge oder Rückmeldungen zurückliefern. Auch diese Personen können den Quellcode im Repository nicht verändern, sondern lediglich die Software runterladen.



Community Building Prozess

Der folgende Zyklus zeigt die typischen, wiederkehrenden Etappen des Community Building Prozesses eines Open Source Projekts:

Jedes Open Source Projekt beginnt mit der Veröffentlichung des initialen Quellcodes durch eine Einzelperson, eine Software-Firma, eine öffentliche Institution oder eine andere, nicht gewinnorientierte Organisation. Aktive Open Source Projekte geben neue Software-Releases innerhalb kurzer Zyklen frei, welche die Stabilität verbessern und den Funktionsumfang erhöhen. Wird die Open Source Lösung breit eingesetzt und kommt sie in geschäftskritischen Bereichen zur Anwendung, schicken die Benutzer oftmals Verbesserungswünsche und Fehlermeldungen an die Open Source Projektverantwortlichen. Wächst die Community, helfen auch Anwender mit, die Dokumentation oder gar die Qualität und Funktionalitäten der Open Source Software mittels Fehlerkorrekturen und Applikationserweiterungen zu verbessern. Wenn sich externe Entwickler gut in den Quellcode eingearbeitet haben, können sie auch an der Kern-Software Beiträge leisten. Nimmt die Nachfrage für Support, Wartung und Weiterentwicklung zu, bilden sich typischerweise Software-Unternehmen, die kommerzielle Dienstleistungen für das Open Source Produkt anbieten. Mit diesen zusätzlichen Entwicklern und Unternehmen wächst die Open Source Community, die den nachhaltigen Fortbestand der Open Source Software gewährleistet. Je nach Entwicklung kann aber auch eine Aufteilung der Community stattfinden, wie der nachfolgende Abschnitt «Erfolgreiches Community Management» auf Seite 23 beschreibt.



Erfolgreiches Community Management

Open Source Projekte werden stets durch eine Einzelperson, ein Software-Unternehmen oder eine Non-Profit Organisation wie beispielsweise eine öffentliche Institution oder eine Forschungsabteilung initialisiert. Ist der Community Building Prozess erfolgreich, wachsen die Benutzer- und Entwicklerzahlen, die Software wird kontinuierlich weiterentwickelt. Der Initiator muss sich nun entscheiden, ob er weiterhin die volle Kontrolle über das Open Source Projekt behält oder ob er andere Personen und Organisationen an den Entscheidungsprozessen teilhaben lassen will.

Gewährt der Initiator anderen mehr Einfluss, kann er mit noch höherer Verbreitung der Software sowie mehr externen Verbesserungen und Erweiterungen des Quellcodes rechnen. Allerdings verliert er dabei einen Teil der Kontrolle, denn die anderen Stakeholder können künftig bei technischen oder organisatorischen Fragen mitbestimmen.

Behält der Initiator weitgehend die Kontrolle, kann er alleine über die Weiterentwicklung der Software bestimmen und so ungestört seine Geschäftsstrategie umsetzen. Allerdings sinkt dadurch die Akzeptanz bei Anwendern und Entwicklern, wodurch das Risiko steigt, dass die Investitionen und Beiträge von Externen nie sehr gross sein werden. Wenn die Software sehr oft eingesetzt wird und der Initiator sich aus Sicht der übrigen Community gravierend wiederholt falsch verhält, kann sich gar eine Abspaltung bilden, ein so genannter «Fork». Unzufriedene Personen und Organisationen können dank der Open Source Lizenz den Quellcode kopieren und auf einer neuen Kollaborationsplattform unter neuem Namen eigenständig weiterentwickeln. Ob der Fork langfristig erfolgreich ist, ob das ursprüngliche Open Source Projekt überlebt oder gar eine Parallelentwicklung entsteht, hängt von der jeweiligen Situation ab.

Checkbox Erfolgreiches Community Management

- ☐ Auf Transparenz und nachvollziehbare Entscheidungsprozesse wird geachtet
- ☐ Wichtige Beitragende aus der Community werden in Entscheidungsfindungen einbezogen
- ☐ Es wurde evaluiert, eine unabhängige Stiftung oder einen Verein (Foundation oder Association) zu schaffen, um alle wichtigen Stakeholder optimal ins Community Management einzubeziehen
- ☐ Auch kritische Stimmen werden in der Community respektvoll behandelt

Open Source versus Free Software (Freie Software)

Obwohl im IT-Alltag die beiden Bezeichnungen meist als Synonym verwendet werden und aus technischer Sicht tatsächlich identisch sind, ist in Fachkreisen die Unterscheidung zwischen Open Source Software und Freier Software wichtig und wird oft heftig diskutiert. Dies ist einerseits historisch bedingt, denn der Begriff «Freie Software» ist bereits 1983 eingeführt worden, während die Definition von Open Source erst 1998 entstanden ist. Andererseits vertreten die beiden Ansätze einen etwas unterschiedlichen thematischen Schwerpunkt, der im Folgenden erläutert ist. Hinweis: In dieser Publikation wird ausschliesslich die Bezeichnung «Open Source» verwendet. Dies geschieht mit dem Wissen, dass Freie Software die Wurzeln und das grundlegende Gedankengut der gesamten Entwicklung darstellt.

Freie Software als Begriff für freies Wissen

Freie Software betont die langfristig gesicherte Verfügbarkeit des Quellcodes, um das Wissen frei zugänglich zu halten und so den Nutzen für die Gesellschaft zu erhöhen. Dazu wurde unter anderem das Wortspiel «Copyleft» (to leave = überlassen) eingeführt. Dieser Begriff besagt, dass veränderte und verbreitete Versionen des Quellcodes wiederum frei verfügbar sein müssen. Die weit verbreitete GNU General Public License (GPL) beinhaltet den Mechanismus, dass einmal unter der GPL veröffentlichte Software stets unter der GPL verfügbar sein muss. Deshalb wird die GPL auch oft als «virale» Lizenz bezeichnet, weil sie andere Software-Teile damit «anstecken» kann. Aus Sicht der Free Software Foundation, die den Begriff Freie Software geprägt und die GPL veröffentlicht hat, ist die Freiheit der Software-Benutzer ausschliesslich durch diesen viralen Effekt geschützt.

Open Source als Begriff für Offenheit

Beim Denkansatz von Open Source Software steht der offene Quellcode und dessen vielseitiger Einsatz im Mittelpunkt. Diese Ausrichtung wurde Ende der neunziger Jahre bewusst als pragmatischer Gegenpol gewählt, da der Begriff «Freie Software» von der Bedeutung her keine geeignete Ausgangslage für den Business-Einsatz darstellte. Die damals gegründete Open Source Initiative stellte zehn Kriterien zusammen, die sich inhaltlich eng an die Definition von Freier Software anlehnen. Allerdings begrüsst der Begriff «Open Source» explizit auch Lizenzen, die eine Weiterverwendung von Open Source Software in proprietärer Software gutheissen.

Kurzer historischer Überblick

Die folgenden Ereignisse, gruppiert in drei Epochen, zeigen punktuell die Entwicklung und Verbreitung von Open Source Software.

Die Pionier Epoche

1985 Richard Stallman gründet die Free Software Foundation (FSF) und prägt den Begriff «Free Software»
1989 Die FSF veröffentlicht Version 1 beziehungsweise 1991 Version 2 der GNU General Public License (GPL)
1991 Linux Torvalds lanciert die Entwicklung des Linux-Kernels unter der GPL Version 2
1993 Die Linux-Distribution Debian wird gegründet

Die Business Epoche

1998 Eric Raymond, Bruce Perens und Tim O'Reilly gründen die Open Source Initiative (OSI) und prägen den Begriff «Open Source»
1999 Netscape veröffentlicht als erstes grosses Unternehmen bisher proprietären Quellcode, den Netscape Navigator, unter einer Open Source Lizenz
1999 Der Linux-Dienstleister Red Hat geht an die Börse NASDAQ
2000 IBM kündigt an, in den kommenden Jahren eine Milliarde US Dollar in die Linux-Entwicklung zu investieren
2001 IBM veröffentlicht die Software-Entwicklungsplattform Eclipse im geschätzten Wert von USD 40 Millionen unter einer Open Source Lizenz

Die Mainstream Epoche

2004 Die Firma Canonical lanciert Ubuntu, ein Linux-Desktop mit Endbenutzern als Zielpublikum, das heute von rund 12 Millionen Benutzern eingesetzt wird
2007 Die Free Software Foundation veröffentlicht Version 3 der GNU General Public License (GPL)
2007 Sun Microsystems veröffentlicht das Java Development Kit unter der GPL und nennt die Software-Plattform OpenJDK
2008 Google lanciert das Linux-basierte Betriebssystem Android für Smartphones, das bereits im Jahr 2011 auf rund 20 Millionen Geräten eingesetzt wird
2008 Die Französische Gendarmerie migriert 70 000 Desktops von Microsoft auf die Linux-Distribution Ubuntu
2009 In Brasilianischen Schulen wird Linux an über 300 000 virtuellen Arbeitsplätzen installiert
2011 In der Spanischen Region Andalusien ist auf rund 500 000 Schulcomputern die eigene Linux-Distribution Guadalinux installiert
2011 Die Versicherungsgesellschaft LVM migriert 10 000 Arbeitsplätze auf die Linux-Distribution Ubuntu

Häufige Vorurteile

«Für Open Source Software gibt es keinen Support»

Stimmt nicht. Wie im historischen Überblick dargestellt, hat sich seit Ende der neunziger Jahre eine rege, innovative Business-Landschaft von Anbietern rund um Open Source Software entwickelt, die zum Teil grosse Open Source Einführungen verantwortet haben. Für viele Open Source Lösungen existieren heute Software-Unternehmen, die gleich wie bei proprietärer Software Service Level Agreements anbieten (siehe Abschnitt «3. Professioneller Einsatz von Open Source», Seite 12).

«Die rechtliche Situation rund um Open Source Software ist unklar»

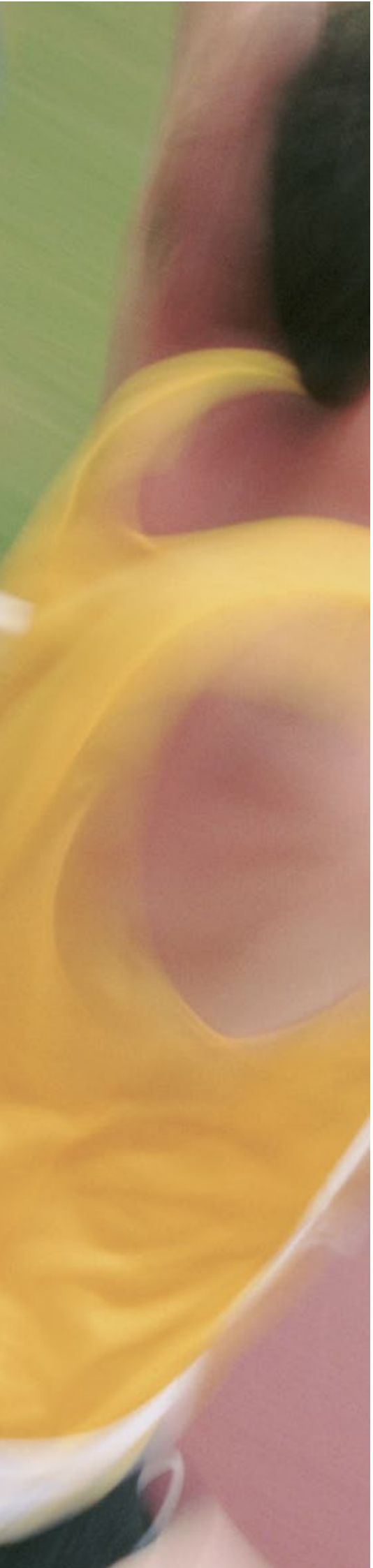
Stimmt nicht. Open Source Software wird immer unter einer klar definierten, von der Open Source Initiative (OSI) zertifizierten Lizenz veröffentlicht, welche festlegt, wie die Software verwendet werden darf und wie nicht. Bei spezifischen lizenzrechtlichen Fragestellungen sollte ein individuelles Gutachten von Experten eingeholt werden, um Lizenzkompatibilitäten und Einhaltung der Bestimmungen sicherzustellen (siehe Abschnitt «4. Rechtliche Aspekte von Open Source», Seite 15).

«Open Source Software ist gratis»

Sofern die Software heruntergeladen wird, kostet Open Source Software tatsächlich nichts. Für den professionellen Einsatz sollten jedoch Service Level Agreements oder Subscriptions beschafft werden, welche die zuverlässige Weiterentwicklung, die Wartung und den Support sicherstellen (siehe Abschnitt «3. Professioneller Einsatz von Open Source», Seite 12). Entscheidend ist, dass der Quellcode von Open Source Software frei verfügbar ist und verändert sowie vervielfältigt werden darf. Dadurch verringert sich die Abhängigkeit vom Software-Hersteller, das spart langfristig Kosten.

«Open Source Software ist wenig verbreitet»

Stimmt nicht. Der Einsatz von Open Source Software ist vielfältig, er findet heute millionenfach auf dem Desktop, auf Servern, auf Mobile Devices und auch in Embedded Systems statt. Selbst Grossunternehmen und Banken setzen im geschäftskritischen Umfeld Open Source ein. Da es aber oft keine firmeneigenen Produkte sind, wird nicht intensiv über die Anwendung von Open Source Software informiert. Deshalb bleibt der tatsächliche Einsatz von Open Source Software oft unterschätzt. Auch wären viele der heute bekannten Internetunternehmen ohne massiven Einsatz von Open Source Software nie erfolgreich geworden.



6. Fazit

Open Source Software ist ein wichtiger Bestandteil der Informatik geworden. Richtig genutzt, erlauben Open Source Lösungen wesentliche kurzfristige und langfristige Kosteneinsparungen. Open Source hilft auch, strategische Vorteile wie Unabhängigkeit, Sicherheit und Transparenz zu schaffen. Die Investitionssicherheit steigt, die digitale Nachhaltigkeit wächst.

Allerdings setzt der Umgang mit Open Source Software Knowhow und Erfahrung voraus, um die vorhandenen Vorteile zu nutzen und Risiken erfolgreich zu meistern. Mit gezielten Massnahmen lassen sich die Herausforderungen beim Einsatz und der Verbreitung von Open Source Software bewältigen. Die in dieser Broschüre erläuterten Good Practices helfen dabei, vom bewährten Vorgehen anderer Unternehmen und öffentlicher Institutionen zu lernen und durch den kontrollierten Umgang mit Open Source Technologien das volle Potential zu entfalten.

Zur Unterstützung bei der Umsetzung von Good Practices, bei Lizenzfragen und weiteren Themen zu Open Source Software im geschäftskritischen Einsatz steht Ernst & Young mit einem kompetenten Open Source Advisory Team gerne zur Verfügung.

Ernst & Young

Assurance | Tax | Legal |
Transactions | Advisory

Über Ernst & Young
Ernst & Young ist ein weltweit führendes Unternehmen in den Bereichen Wirtschaftsprüfung, Steuern, Transaktionen und Beratung. Unsere 141 000 Mitarbeitenden auf der ganzen Welt verbinden unsere gemeinsamen Werte sowie ein konsequentes Bekenntnis zur Qualität. Wir differenzieren uns, indem wir unseren Mitarbeitenden, unseren Kunden und unseren Anspruchsgruppen dabei helfen, ihr Potenzial auszuschöpfen.

Ernst & Young bezieht sich auf die globale Organisation der Mitgliedsfirmen von Ernst & Young Global Limited (EYG), von denen jede eine eigene Rechtseinheit bildet. EYG, eine Gesellschaft mit beschränkter Haftung nach britischem Recht, erbringt keine Dienstleistungen für Kunden. In der Schweiz ist die Ernst & Young AG ein führendes Wirtschaftsprüfungs- und Beratungsunternehmen mit rund 2 000 Mitarbeitenden an 10 Standorten und bietet auch Dienstleistungen in den Bereichen Steuern und Recht sowie Transaktionen und Rechnungslegung an.

Kontakt

Advisory Services		
Jürg Brun (Partner)	+41 58 289 32 03	juerg.brun@ch.ey.com
Ferdinand Kobelt (Partner)	+41 58 289 69 31	ferdinand.kobelt@ch.ey.com
Reto Aeberhardt (Senior Manager)	+41 58 289 67 40	reto.aeberhardt@ch.ey.com
Dr. Matthias Stürmer (Senior)	+41 58 289 61 97	matthias.stuermer@ch.ey.com