Die Zeit ist reif für Open Source Software

von Uta Kapp*, 2007*

Open Source: eine neue Herausforderung!

Jeder, der sich mit Softwareentwicklung beschäftigt, wird früher oder später mit dem Thema Open Source konfrontiert. Manche Produkte, wie Apache Webserver und Linux sind so verbreitet, dass keiner an ihnen vorbei kommt. Immer öfter stellt sich die Frage, Open Source Produkte einzusetzen, oder sogar selbst Software als Open Source Produkt bereitzustellen.

Was heißt Open Source?

Open Source Software heißt, dass der Quellcode für die Software offen gelegt wird. Viel wichtiger ist jedoch, dass sich eine große Community hinter diesem Thema verbirgt, die diese Philosophie vertritt und mitträgt. Hier müssen wir unterscheiden zwischen Open Source Software und Free (freie) Software. Frei (free) steht hier im Sinne von Meinungsfreiheit. Im Allgemeinen ist die Open Source Software, die hier angesprochen wird, auch freie Software. Die Details werden im Einzelnen in den Lizenzen geregelt.

Freie Software

Freie Software hat etwas mit Freiheit zu tun und nicht mit einem Preis. Es geht um die Freiheit des Benutzers, die Software auszuführen, kopieren, verteilen, lesen, ändern und verbessern.

Die Freiheit ein Programm für irgendeinen Zweck auszuführen.
Die Freiheit die Software zu lesen und für eigene Zwecke anzupassen. Zugang zum Quellcode ist hierfür Voraussetzung.
Die Freiheit, die Software in Original oder veränderter Form zu verteilen.
Die Freiheit, die Verbesserungen der Software zurück in die Community zu bringen, sodass alle davon profitieren können.

Die Open Source Community

Die Open Source Community kann nicht als “die” Community bezeichnet werden.

Es gibt ein großes Netz an Websites, die diese Community in jeder Hinsicht unterstützt. Dazu gehören Sites wie Sourceforge, auf denen man seine Software hosten kann, eine Vielzahl von Firmen, die die Bandbreite ihrer Server der Community zur Verfügung stellen, um Downloads durchzuführen. Ibiblio ist eine Website, auf der Bibliotheken kostenlos im Archiv gelagert werden und zum Download bereit stehen. Es gibt Foren, auf denen man sich Hilfe besorgen kann.

Java Community

Die Firma Sun Microsystems hat mit dem Projekt Java die größte Community im Open Source Bereich ins Leben gerufen. Sun wurde lange dafür kritisiert, dass sie den Java Quellcode nicht offen gelegt hatten. Dies hat sich nun geändert und es ist unter einer GPL Lizenz, mit 2 Ausnahmen, verfügbar.

Bekannteste Beispiele

Linux
Apache Webserver
Java

Warum ist die Zeit reif?

Die Zeit ist reif, auch in kommerziellen Projekten auf Open Source Produkte zu setzen. Sie ist auch reif dafür, die eigene Software als Open Source Software der Öffentlichkeit zur Verfügung zu stellen.

Für alle Bereiche sind Open Source Produkte verfügbar

Open Source Software, die auch freie Software ist, basiert notwendigerweise auf Software, die auch Open Source und frei ist. Damit es also möglich ist, sinnvoll Open Source zu erstellen, die, die Kriterien für eine Open Source Software erfüllt, muss die Basissoftware, auf die man solch ein Produkt aufsetzt, vorhanden sein. Wenn das nicht der Fall ist, dann müsste die Basissoftware zuerst erstellt und damit das Rad neu erfunden werden. In den letzten Jahren wurden für fast alle wichtigen Bereiche Bibliotheken, Tools und Basissoftware, vor allem für die Softwareentwicklung mit Java, bereitgestellt.

Dies umfasst Software für den gesamten Lebenszyklus eines Projektes:

- Modellierung
- Programmierung
- Test
- Projektverwaltung und -konfiguration
- Betriebssysteme

Netzwerke für Hosting und Community stehen zur Verfügung

Man kann auf ein bereits bestehendes Netz aus Servern für das Hosting von Software und Communities zurückgreifen. Dies ist behilflich beim Bauen einer eigenen Community für das Produkt.

Chancen

Zuerst kommt einem der Gedanke, dass es ja wunderbar ist, wenn man auf so viele „Geschenke" zurückgreifen kann. Warum soll man jedoch selbst seine Software als Open Source zur Verfügung stellen? Vor allem Firmen denken zuerst daran, dass sie ja der Konkurrenz ihr Know How offen legen und diese davon, zum eigenen Nachteil, profitiert. Wenn Trittbrettfahrer nur nehmen und nichts geben, dann legt man drauf.

Open Source Produkte nutzen statt Rad neu erfinden

Bevor man ein neues Produkt selbst entwickelt, lohnt es sich auf den einschlägigen Websites, wie z.B. Sourceforge, nach Software zu suchen, die schon ähnlich gelagert ist, wie das was man gerne erstellen möchte. Man hat dann die Möglichkeit, ein Teil der Software mit Open Source Bibliotheken zu bestücken, oder ein Produkt zu übernehmen und anzupassen.

Softwareexperten sind rar

Idealerweise möchte man von einem Open Source Produkt nicht nur am Anfang einer Softwareentwicklung profitieren, sondern möchte auch von der Weiterentwicklung der Open Source Produkte profitieren. Deshalb wird man ein Open Source Softwareprodukt idealerweise als klar abgegrenzte Komponente im eigenen System verwenden.

Auch bei Fluktuation, von Softwareexperten, bleiben diese dem Open Source Produkt erhalten

Der Einsatz von Open Source Produkten zeigt, bei genauem Hinsehen, dass langfristig die Vorteile, die es bringt mit solchen Produkten zu arbeiten, überwiegen. Dies gilt vor allem für kleine Firmen, die sich keinen großen Stab an Softwareexperten leisten können. Ein Open Source Projekt mit einer stabilen Community verfügt über mehr Experten, als ein inhouse Projekt, das zusammenbricht, wenn der Programmierexperte die Firma wechselt. Hier hat es schon viele bekannte Katastrophen gegeben. Bei einem Open Source Projekt bleibt der Entwickler erhalten, auch wenn er zur Konkurrenz wechselt.

Basisprodukt als Open Source Produkt vermarkten

Ideen sind immer gleichzeitig. Wenn gleichzeitig mehrere Firmen an der gleichen Produktart arbeiten, dann wird einer auf Dauer auf der Strecke bleiben. Wenn eine Firma seine Software zuerst als Open Source Software anbietet, dann ist es für den Konkurrenten interessanter zusammenzuarbeiten als ein neues Konkurrenzprodukt auf den Markt zu bringen.

Einsatz in kommerziellen Produkten

Wenn Open Source Produkte in kommerziellen Produkten zum Einsatz kommen, dann ist es ratsam nur Open Source Produkte einzusetzen, die mindestens in der Stabilisierungsphase ist. Spezielle Produkte, mit

begrenztem Umfang, die sehr stabil sind, können auch schon in der Startupphase eingesetzt werden. Man muss wissen, dass auf ein Open Source Team, dass nicht zur eigenen Firma gehört, keinerlei Termindruck ausgeübt werden kann. Wenn also ein Produkt einen Bug hat, der beim eigenen Produkt zu Instabilitäten führt, dann kann man es bestenfalls selbst reparieren und der Community als Bugfix zur Verfügung stellen. Wenn eine Firma nicht über dieses Know How verfügt, dann ist es besser, ein besser etabliertes Produkt auszuwählen.

Zusatzprodukte kommerziell vermarkten

Wenn man Software für ein Projekt als Open Source Software bereitstellt, dann wird die Software und damit die Firma ihren Bekanntheitsgrad steigern. Sie kann jetzt zwar nicht die eigentliche Software verkaufen, aber ihren Service allemal. Zusatzprodukte können kostenpflichtig angeboten werden.

Produktreife

Die Wahl treffen

Nach welchen Kriterien sucht man ein Open Source Produkt aus, auf das man bauen möchte? Zuerst einmal suche ich natürlich das Produkt, dass die Features hat, die meinen Zweck erfüllen. Wenn ich dann mehrere Produkte zur Auswahl habe, dann muss ich wählen. Die Wahl sollte nicht nur von der Anzahl Features abhängen, die ein Produkt zurzeit hat, sondern auch davon, wie sich die Community des Produktes darstellt.

Produktreife

Ein Produkt wird am Anfang immer von einem sehr kleinen Team, meist 1-3 Personen, gestartet. Es gibt Produkte, die schon in der Betaversion, des ersten Releases sehr zuverlässig und stabil funktionieren. Normalerweise ist ein Produkt stabil und gut eingeführt, wenn es in der 3. Release freigegeben wird. Ein Produkt kann in 3 Stadien der Reife eingeteilt werden:
Startupphase: Relativ neues Produkt
Stabilisierungsphase: Hat sich schon bewährt, wird aber noch nicht selbstverständlich in kommerziellen Bereichen verendet.
Eingeführtes Produkt, wie z.B. Apache Webserver, Linux

Produktumfang

Produkte können einen sehr kleinen Fokus haben, oder aber große voluminöse Meta-Frameworks sein, die auf viele andere Open Source Produkte zurückgreifen und mit ihnen gekoppelt sind. Produkte, die sehr komplex sind, integrieren schon viel Funktionalität, die man sonst erst selbst implementieren müsste. Sie bergen aber eine Menge Komplexität, die zur Falle werden kann.

Aktivität der Community

Wie kann man erkennen, ob ein Produkt „alive“ ist? Die Aktivität einen User Community in Form von Email und Forum lässt Schlussfolgerungen zu, ob ein Produkt ständig gepflegt und weiterentwickelt wird, oder ob es tot ist. Auch die Zeitdauer, wie lange ein Produkt aktiv auf dem Markt ist, gibt Auskunft, wie aktiv ein Produkt entwickelt wird. Die Größe der Community sagt nicht unbedingt viel aus über die Qualität eines Produktes.

Produktzuverlässigkeit

Eines der schwierigsten Aufgaben ist es, etwas über die Zuverlässigkeit eines Open Source Produktes heraus zu bekommen. Die kann man eigentlich nur selbst testen. Im Internet gibt es viele Quellen, wo in Foren oder in UserGroups über Produkte diskutiert wird. Da aber immer nur über Probleme diskutiert wird, nie über die Erfolge und das, was gut läuft, muss man sich erst selbst ein Urteil bilden.

Open Source Projekte managen

Für die Verwaltung von Open Source Projekten gibt es inzwischen eine große Anzahl an Open Source Tools, die zur Unterstützung bereitstehen. Dies sind Content Versions Management Tools, Projektkonfigurations- und Buildtools, Dokumentationstools, Reportingtools, Entwicklungsumgebung und Testtools.

Repositories

Bei einem Open Source Projekt müssen die Ressourcen des Projektes so organisiert sein, dass sich ein virtuelles Team damit organisieren kann, ohne sich gegenseitig zu behindern. Der Quellcode muss zugänglich sein. Alle erforderlichen Quellcodes müssen über öffentliche Server in öffentliche Repositories abgelegt werden.

Versionsmanagement

Das Versionsmanagement umfasst zwei Bereiche: das Versionieren der selbst bereitgestellten Software und der Umgang mit den sich ständig weiterentwickelnden Versionen der verwendeten Open Source Komponenten.

Zuverlässige Builds

Regelmäßige Builds müssen automatisch aus dem Quellcode bereitgestellt werden und dazu die richtige Konfiguration herangezogen werden. Es muss möglich sein, einen Build aus dem Quellcode so zu erstellen, dass er in jeder Hinsicht jedes Mal gleich ist, egal, in welcher Umgebung er generiert wird.

Automatisierung

Es ist sinnvoll einen Rechner bereitzustellen, auf dem täglich automatisch der neueste Stand als lauffähiger Build bereitgestellt wird. Dazu ist es notwendig, dass automatische Tests zur Kontrolle mit ausgeführt werden. Die Versionierung muss automatisch in allen Repositories registriert werden, um jeden Build nachvollziehen zu können. Reports werden automatisch generiert und auf der Projektsite abgelegt.

Konfigurationsmanagement

Eventuell ist es erforderlich, dass für jede Betriebssystemvariante eigener Binärcode und eigene Prozeduren herunterladbar sind.

Das Netzwerk nutzen

Im Internet gibt es verschieden Plattformen, auf denen es möglich ist, ein Open Source Projekt kostenlos zu hosten. So braucht sich niemand um die Serververfügbarkeit und Bandbreite für die Downloads zu kümmern. Auch Bibliotheken können kostenlos hinterlegt werden

- Sourceforge
- Ibiblio
- Codehaus
- Apache

Es gibt Server mit etablierten Newsgroups und Usergroups die eine Community nutzen kann.

Community

Kommunikation

Auch wenn technisch alle notwendigen Unterstützungen für ein Open Source Projekt zur Verfügung stehen, so bleibt das Wichtigste, die Führung des Open Source Teams, bei den Initiatoren eines Projektes. Kein Commiter wird irgendetwas zu einem Projekt beitragen, wenn er nichts dafür zurück bekommt. Auch wenn es bei einem Open Source Projekt kein Geld ist, so ist doch Anerkennung, Wertschätzung und Achtung besonders wichtig. Nur so kann ein Projekt erfolgreich funktionieren.

Programmieren in der Öffentlichkeit ist gewöhnungsbedürftig. Jeder kann weltweit die Programmierkunst und den Emailverkehr mit verfolgen und über Google suchen. Der Umgang mit dieser Situation bedarf sehr viel Sorgfalt.

Rollenverteilung

Folgende Rollen sollten idealerweise in einem Open Source Projekt verteilt werden:

Projektmanager – Initiator und Evangelist
Projektorganisator – kümmert sich um Project Health, reporting, website
Gatekeeper – kümmert sich um Kommunikation mit Neueinsteigern
PR – Vertretung in der Presse, Veröffentlichung

Foren, Usergroups, Blogs, Email, Bug-Database

Jedes Open Source Projekt braucht Kommunikationsplattformen, wie Emaillisten usw. Wenn das Projekt bei einer Plattform wie SourceForge.net gehostet ist, dann sind diese Funktionen dort schon organisiert und installiert.

Wertschätzender Umgang

Jeder, der seine Zeit in einer Open Source Community zur Verfügung stellt, tut dies freiwillig. Wenn man um Hilfe in einer Community bittet, dann muss man dies immer beachten. Forderungen oder gar Beschimpfungen gehen nur ins Leere. Dies steht manchmal im Konflikt zur Firma, wenn man ein Produkt kommerziell einsetzen möchte. Es gibt niemanden, den man zwingen könnte, ein Problem zu lösen, auch wenn der Chef dies verlangt. In dieser Beziehung hat es schon sehr unterhaltsame Emails in Foren gegeben. Wenn Forderungen an ein Produkt gestellt werden sollen, die man nicht selbst für das Open Source Produkt leisten kann, dann ist es besser, ein kommerzielles Produkt einzusetzen. In jeder Lizenz ist zu lesen, dass die Software so akzeptiert werden muss, wie sie ist.

Risiken

Kreativitätsfallen

Open Source Projekte sind im Allgemeinen für ihre Innovationskraft bekannt. Ein Produkt kann jedoch auch zu wenig Innovation und Kreativität aufweisen und nicht mit dem Technologiestandard mithalten. Schlimmer ist aber meistens, wenn zu viel Kreativität an den Tag gelegt wird. Dies führt zu Unstabilität und kann sogar zu einem Kollaps führen. Sichtbar wird dies, wenn es nicht möglich ist, eine aktuell lauffähige Version zu installieren und die letzte lauffähige Version schon sehr alt ist. Wenn ich mir die neueste Version mühsam aus dem CVS Archiv selbst zusammenbauen muss, dann läuft irgendetwas schief. Wenn es unmöglich wird, die nächste stabile Version rauszubringen, dann gibt es ein Problem.

Problematisch ist auch, wenn eine neue Version nicht abwärts kompatibel ist. Dann bedeutet es ist viel Arbeit, diese Version in das eigene System zu integrieren.

Teamfallen

Vor allem bei Produkten in der Startupphase kann es passieren, dass ein Team genau so schnell wieder auseinander strebt, wie es begonnen hat. Mit dem Team stirbt dann auch das Projekt.
Ein anderes Problem kann auftreten, wenn ein Produkt von einem freien Produkt zu einem kommerziellen Produkt umgewandelt wird. Jetzt fallen Lizenzgebühren an.

Hypefallen

Jedes Jahr gibt es neue Hypes, welches Produkt denn angeblich Standard wird in einem Bereich. Immer wieder stellt sich die Frage, ob auf das richtige Produkt gesetzt wurde, oder ob es ratsam ist, ein Produkt auszutauschen. Hier ist es das Wichtigste eine klare Entscheidung zu treffen und dafür zu sorgen, dass das Produktzusammenspiel für die eigenen Bedingungen funktioniert. Man kann auch mal einen Hype auslassen. Totgesagte leben bekanntlich länger.

Versionschaos

Im Java Programmierumfeld kreisen alle Produkte um die gleichen Basisbibliotheken. Diese sind normalerweise abwärts kompatibel. Somit ist ein Programm mit einer neueren Version einer Bibliothek lauffähig, aber nicht mit einer Älteren. Da viele Bibliotheken unter verschiedensten Namen und Versionen in

verschiedenen Produkten herumschwirren, kann es zu einer Sisyphusarbeit werden die richtigen Versionen in Einklang zu bringen. Eine neue Version kann also eine ganze Kaskade von benötigten Upgrades von anderen Produkten hinter sich her ziehen. Es ist also allgemein ratsam das eigene System mit nicht zu alten Versionen von anderen Open Source Produkten zu betreiben.

Projekt stirbt

Wenn ein Projekt stirbt auf das man gesetzt hat, dann kann man die Software in eigenen Besitz übernehmen oder die Community selbst weiterführen.

Projekt wird kommerziell

Es gibt Open Source Projekte, bei denen sich die Community plötzlich entscheidet, die Weiterentwicklung des Projektes nur noch kommerziell anzubieten. Hier gibt es die Möglichkeit, den letzten Stand des Projektes in den eigenen Besitz zu übernehmen, wenn man mit den Lizenz und Kostenbedingungen nicht zurecht kommt.

Lizenzen

Wenn man Open Source Produkte nutzen möchte, dann ist es wichtig die Lizenzbedingungen genau unter die Lupe zu nehmen und zu prüfen, ob es für den gewünschten Einsatz rechtlich geeignet ist.

Lizenzvarianten

General Public License GPL
GPL mit Ausnahme: Ein Beispiel liegt hier vor für die Programmiersprache Java. Sun hat eine Variante von Java freigegeben mit GPL Lizenz und 2 Ausnahmen.
Apache Software License: Diese Lizenz hat sehr wenige Einschränkungen für den Benutzer.

Virale Lizenzen

Es gibt Lizenzen die als „businessfreundlich“ gelten, weil sie wenig Einschränkungen definieren, was mit dem Code passieren darf. Andere Lizenzen, auch als „virale“ Lizenzen bekannt, verlangen, dass jeder auf ihnen basierende Code auch wieder offen gelegt wird. Wenn man ein Produkt mit solch einer Einschränkung in seinem Portfolio hat, dann wirkt sich dies auf den gesamten Code aus.

Ownership

Wem gehört ein Open Source Produkt?

Grundsätzlich gibt es ein Urheberrecht für Programmcode. Dieses wird normalerweise als Lizenz explizit für den Code beigefügt. Die Regelungen sind sehr vielfältig. Software der Projekte der Apache Software Foundation gehören den Mitgliedern des Vereins. Firmen können den Besitz von Software für sich beanspruchen und Beiträge von externen Commitern in ihren Besitz nehmen.

Dokumentation

Qualität

In der Startup Phase eines Open Source Produktes ist die Erfahrung die, dass die Dokumentation fast immer sehr schwach ist. Man ist oft auf unvollständige Dokumentationen, auf Fragmente von HowTo's, UserMails und Wiki's angewiesen.

Bücher

Wenn ein Produkt die Stabilisierungsphase erreicht hat, erkennt man das unter anderem daran, dass Bücher zu dem Produkt auf den Markt kommen. Der Verlag O'Reilly ist hier federführend.

Mailinglisten, Wiki, FAQ, Foren und Usergroups

Im Internet findet man in verschiedenen Wiki, FAQ, Foren und Usergroups Hinweise auf Probleme, die schon einmal von jemandem gelöst wurden.

Google

Google ist ein wahres Wunderwerk, wenn es darum geht, Lösungen für Probleme mit einem Open Source Produkt zu finden, die schon einmal jemand gelöst hat. Auch hier gilt, dass für gut eingeführte Produkte die besten Lösungen zur Verfügung stehen.