12-12-2018

Marktplatz zur Koordinierung und Finanzierung von Open Source Software

Georg J.P. Link
University of Nebraska at Omaha, glink@unomaha.edu

Malvika Rao
Incentive Research

Don Marti
Mozilla

Andy Leak
Mountain View Smart Contracts

Rich Bodo
Mountain View Smart Contracts

Recommended Citation
Link, G.J.P., Rao, M., Marti, D. et al. HMD (2018). https://doi.org/10.1365/s40702-018-00474-6

HMD
https://doi.org/10.1365/s40702-018-00474-6

SCHWERPUNKT

Marktplatz zur Koordinierung und Finanzierung von Open Source Software

Georg J. P. Link · Malvika Rao · Don Marti · Andy Leak · Rich Bodo

Eingegangen: 12. September 2018 / Angenommen: 4. Dezember 2018

Zusammenfassung Open Source ist ein zunehmend beliebter Kollaborationsmechanismus für die Entwicklung von Software, auch in Unternehmen. Unsere Arbeit schafft die fehlende Verbindung zwischen Open Source Projekten, Unternehmen und Märkten. Ohne diese Verbindung wurden Koordinations- und Finanzierungsprobleme sichtbar, die zu schwerwiegenden Sicherheitslücken führen. In diesem Paper entwickeln wir acht Design Features, die ein Marktplatz für Open Source haben sollte, um diese Probleme zu beseitigen. Wir begründen jedes Design Feature mit den bestehenden Praktiken von Open Source und stellen einen Prototypen vor. Abschließend diskutieren wir, welche Auswirkungen die Einführung eines solchen Marktplatzes haben könnte.

Schlüsselwörter Open Source Software · Marktplatz · Kollaboration · Finanzierung

G. J. P. Link (✉)
University of Nebraska at Omaha, Omaha, NE, USA
E-Mail: glink@unomaha.edu

M. Rao
Incentive Research, Ottawa, ON, Kanada
E-Mail: malvikar@gmail.com

D. Marti
Mozilla, Mountain View, CA, USA
E-Mail: dmarti@mozilla.com

A. Leak · R. Bodo
Mountain View Smart Contracts, Mountain View, CA, USA

A. Leak
E-Mail: andy@r210.com

R. Bodo
E-Mail: richbodo@gmail.com

Published online: 12 December 2018

Marketplace to Coordinate and Finance Open Source Software

Abstract The popularity of open source as a collaboration mechanism for developing software is increasing. Organizations increase their engagement. In our work, we draw the missing connection between open source projects, organizations, and markets. Without this connection, we have seen severe software vulnerability result from coordination and financing breakdowns. In this paper, we develop eight design features that a market place for open source should have to address these breakdowns. We develop the design features based on literature about the practices of open source. We present a prototype and discuss what implications would result from implementing such a market place.

Keywords Open Source Software · Markets · Collaboration · Financing

1 Einleitung und Motivation

Open Source ist zu einem populären und bewährten Kollaborationsmechanismus in der Software Entwicklung erwachsen (Germonprez et al. 2018). Auch deutsche Unternehmen verwenden und entwickeln Open Source Software, wie zum Beispiel Continental, Daimler, Deutsche Bank, Deutsche Börse, Deutsche Telekom, SAP und Suse – alle Mitglieder der Linux Foundation. Dies ist kein Zufall, da Open Source Software die digitale Infrastruktur unserer modernen Gesellschaft bildet (Eghbal 2016). Insbesondere für Infrastruktur und nicht differenzierende Software bildet Open Source Software die Grundlage auf der auch proprietäre Software entwickelt wird (Lerner und Schankerman 2010). Jeder, der das Internet nutzt, greift auf Router, Webserver und Browser zu, die oft mit Open Source Software laufen.

Die weit verbreitete Nutzung von Open Source Software birgt jedoch auch Gefahren für Privatpersonen, Unternehmen und die Gesellschaft. Die 2014 gefundene Sicherheitslücke Heartbleed brachte diesen Umstand in das allgemeine Bewusstsein (Eghbal 2016). Heartbleed existierte in der kryptographischen Open Source Library OpenSSL, die von einer Mehrheit an Webservern verwendet wird, um Internetverkehr abzusichern (Durumeric et al. 2014). Die Sicherheitslücke erlaubt es, ohne Zugangsbeschränkung, Arbeitsspeicherinhalte von Servern abzurufen, unter anderem schützenswerte Informationen. Es wurde bekannt, dass OpenSSL von wenigen Entwicklern unterhalten wird, die aber kaum Spendengelder erhielten und darüber hinaus nicht für ihre Arbeit an OpenSSL bezahlt wurden.

Als Antwort auf Heartbleed haben die Industrie und die Linux Foundation die Core Infrastructure Initiative gegründet, die mit finanziellen Mitteln Entwickler bezahlt, um wichtige Software wie OpenSSL zu unterhalten. Das Problem ist, Open Source Software zu identifizieren, die weit verbreitet aber nicht ausreichend finanziert ist. TideLift adressiert dieses Problem, indem es die Benutzung von Open Source Software ihrer Mitglieder analysiert und quantifiziert, um entsprechend Gelder an die Open Source Projekte auszubezahlen. Andere Finanzierungsmöglichkeiten können von Open Source Projekten selbst initiiert werden. Sie können durch Crowdfunding auf Open Collective finanzielle Unterstützung einfordern. Dies erfordert

gezieltes Marketing und Fähigkeiten, die nicht typischerweise von einem Softwareentwickler erwartet werden. Wenn es darum geht, gezielt neue Funktionen oder das Fixen von Bugs zu bezahlen, können Open Source Bounties ausgeschrieben werden, die an einen Entwickler ausbezahlt werden, der eine Lösung entwickelt und einreicht. Bei Entwicklungswettbewerben können mehrere Lösungen eingereicht werden. Der Entwickler mit dem beste Ergebnis wird belohnt, während andere Entwickler leer ausgehen. Entwickler können auch direkt bezahlt werden. Arbeitsverhältnisse oder Vertragsarbeiten sind jedoch nicht öffentlich.

Insgesamt gibt es mehrere Möglichkeiten, die Entwicklung von Open Source Software zu finanzieren. Diese schaffen zum Teil Anreize, die gegen die kollaborativen Grundprinzipien von Open Source gehen. Beispielsweise ist ein Entwickler angehalten für ein Open Source Bounty nachzuweisen, dass er die Lösung hat. Wenn er aber die Lösung offen in der Open Source Community programmiert und frühzeitig Feedback einholt, kann auch jemand anderes die Lösung fertigstellen und die Belohnung erhalten. Somit wird ein Entwickler davon abgehalten, die Entwicklung in der Open Source Community bekannt zu geben, bevor er nicht das Open Source Bounty erhalten hat.

Was fehlt ist ein Marktmechanismus für Preissignale. Ein Markt würde die Zahlungsbereitschaft von Anwendern bündelt, um Entwicklern zu zeigen, welche neuen Features und Bugfixes am meisten gewünscht sind und dementsprechend auch entlohnt werden (Germonprez et al. 2018). Wir haben unsere Idee eines Marktplatzes in Bezug auf Sicherheit in Open Source Software vorgestellt (Rao et al. 2018). Gegenstand dieses Beitrags ist die konzeptionelle Beschreibung des Marktplatzes, der die Kollaborationsprinzipien von Open Source bewahrt und verstärkt, während es ebenfalls die Finanzierung von Open Source verbessert.

2 Grundlagen: Koordination in Open Source

Open Source Projekte bestehen aus Software und einer Community. Open Source Software hat eine Softwarelizenz, die von der Open Source Initiative offiziell als Open Source Lizenz anerkannt wird.[1] Die Community sind Projektmitglieder, die sich an der Entwicklung der Software beteiligen.

Mitglieder in Open Source Projekten einigen sich intern auf Strukturen und Regeln, die das Kollaborationsverhalten bestimmen (O'Mahony und Ferraro 2007; Kelty 2008). Selbst wenn Open Source Projekte dieselben Kollaborationswerkzeuge nutzen, entwickeln sie eigene Verhaltensweisen, die im Kommunikationsverhalten offensichtlich werden (Crowston und Howison 2005). Dennoch gibt es Gemeinsamkeiten, die von Open Source Projektmitgliedern anerkannt werden (Elliott und Scacchi 2008).

Softwareentwicklung in Open Source Projekten ist oft informell und selbstbestimmt (Mockus et al. 2002; Scacchi 2004). Entwickler arbeiten oft allein. Aufgaben, die zu umfangreich für einen Entwickler sind, bleiben unbearbeitet, bis zugrundeliegende Teilaufgaben gelöst sind (Howison und Crowston 2014). Entwickler loten im

[1] Open Source Definition und Anerkannte Lizenzen: https://opensource.org/licenses.

Vorfeld einer größeren Änderung aus, wie die Meinungen anderer Entwickler sind und entwickeln eine Lösung die das Ergebnis dieser Gespräche ist. Kommunikation ist häufig asynchron in E-Mail-Listen, Foren, und Issue-Trackern. Issue-Tracker nehmen dabei eine besondere Rolle ein. Ähnlich eines Forums erlauben Issue-Tracker fokussierte Unterhaltungen, die nach Issues strukturiert sind. Ein Issue enthält eine Beschreibung eines Bugs oder Features und erlaubt Entwicklern sich gezielt darüber zu unterhalten. In Issues werden Ideen, Probleme, Workarounds, Lösungsstrategien, und Alternativlösungen ausgetauscht und darüber abgestimmt. Die Diskussion in einem Issue wird offiziell als beendet erklärt, indem der Status des Issues von offen auf geschlossen geändert wird.

Projektmitglieder beteiligen sich aus eigenem Interesse, weil sie lernen wollen, das Ideal von Free Software unterstützen wollen oder die Gemeinschaft in Open Source Communities genießen wollen (von Krogh et al. 2012). Sie beteiligen sich auch, weil sie von ihrem Arbeitgeber dafür bezahlt werden oder weil die Mitarbeit in Open Source Projekten die Karrierechancen verbessert (von Krogh et al. 2012). Einige Open Source Projekte werden von Entwicklern dominiert, die sich im Rahmen ihrer Erwerbstätigkeit beteiligen (Kelty 2013; Riehle et al. 2014). Dies ist nicht verwunderlich, weil Unternehmen Open Source Software in ihren Wertschöpfungsketten verwenden (Dahlander 2005).

Unternehmen können Open Source Software ohne Einschränkung und ohne Lizenzkosten verwenden. Sie kann günstiger sein als proprietäre Software, ist aber nicht kostenlos nach den Total Cost of Ownership (Wheeler 2015). Unternehmen stellen Entwickler ein, die Open Source Software für sie anpassen, integrieren, konfigurieren und updaten. Unternehmen haben dabei eine Wahl. Entweder sie nutzen nur die Software und geben nichts an die Community zurück oder sie erlauben ihren Entwicklern, sich an den Projekten zu beteiligen (Dahlander und Magnusson 2005). Letzterer Ansatz bringt mehr Vorteile (Bonaccorsi et al. 2006). Entwickler, die sich an Projekten beteiligen, lernen mehr über die Software und können somit ihre Aufgaben im Unternehmen besser erfüllen (Nagle 2018). Außerdem können Entwickler Einfluss auf die Softwareentwicklung nehmen und Bedürfnisse des Unternehmens einfließen lassen (Dahlander und Wallin 2006).

Die zunehmende Beteiligung von Unternehmen beeinflusst die Zusammenarbeit in Open Source Projekten. Freiwillige Entwickler betrachten den Einfluss von Unternehmen skeptisch (Spaeth et al. 2015). Außerdem brauchen Unternehmen Planungs- und Rechtssicherheit. Diese Faktoren haben zur Gründung von Open Source Foundations geführt (Riehle 2010). Foundations stellen Rechtssicherheit her, indem sie die Verwaltung von Rechten und Trademarks an Open Source Software übernehmen. Sie liefern zuverlässig technische und verwaltende Unterstützung für Open Source Projekte. Sie schaffen ein Kollaborationsumfeld, das Projektmitglieder und Unternehmen zu Gleichberechtigten macht. Unternehmen, Foundations und Open Source Projekte haben Open Source Praktiken weiterentwickelt, um besser miteinander Software zu entwickeln (Germonprez et al. 2017). Open Source Projekte sind strategischer geworden, planen langfristiger und bieten längere Supportzeiträume (Fitzgerald 2006).

Trotz zunehmender Investitionen durch Unternehmen und Unterstützung durch Foundations hat Open Source ein Koordinierungs- und Finanzierungsproblem. Das liegt daran, dass Open Source Software häufig eine Zusammensetzung von mehreren Open Source Libraries[2] ist. Einmal programmierte Funktionen stehen allen zur Verfügung, was zu Abhängigkeiten zwischen Open Source Projekten führt. Zum Beispiel führt das Löschen einer häufig genutzten Open Source Library zu Fehlern in vielen anderen Projekten (Collins 2016). Sicherheitslücken in einer Open Source Library machen auch alle Open Source Software (und proprietäre Software), die diese Library verwenden, unsicher (Durumeric et al. 2014). Diese Abhängigkeiten erfordern Koordination zwischen Open Source Projekten (Eck 2018) und eine stabile Finanzierung der Entwicklungsarbeit (Eghbal 2016). Diese Probleme wurden auch in der Politik erkannt. Beispielsweise hat der U. S.-Kongress bei der Linux Foundation angefragt, wie diese Probleme angegangen werden (Walden und Harper 2018).

Das Bewusstsein ist in der öffentlichen Debatte erst durch Heartbleed aufgekommen (Eghbal 2016). Als Antwort wurde die Core Infrastructure Initiative gegründet, die Entwickler bezahlt damit sie sich um die Weiterentwicklung von kritischer Software kümmern, die jedoch keine koordinierende Rolle übernimmt. Auch bei TideLift wird nur die Finanzierung sichergestellt, es findet aber keine Koordinierung zwischen Open Source Projekten statt. Wo hingegen Geld und Arbeit direkt verknüpft sind, ist bei Bug Bounties oder Open Source Bounties. Bug Bounties loben Geld für das Finden von Bugs aus. Der Finder muss nachweisen, dass ein Bug existiert und ein Angriffsszenario demonstrieren. Das verhindert Koordinierung mit anderen Entwicklern, bis die Belohnung ausbezahlt ist. Open Source Bounties werden auf die Erledigung von festgelegten Arbeiten ausgelobt, wie die Entwicklung von neuen Features. GitCoin (https://gitcoin.co/) ist eine Implementierung eines Open Source Bounty Systems und beschreibt die Arbeitsschritte, die zusätzlich zur normalen Open Source Entwicklung notwendig sind. Auch hier findet keine Koordinierung zwischen Projekten statt. Zusammenfassend lässt sich feststellen, dass es keine Finanzierungsmöglichkeit gibt, die gleichzeitig eine Koordinierungsaufgabe zwischen Open Source Projekten übernimmt.

Finanzierung und Koordination können aus ökonomischer Sicht betrachtet werden. Benkler (2002) ist Vordenker auf diesem Gebiet und hat Open Source[3] als neuen Koordinierungsmechanismus neben Märkten und Unternehmen positioniert. Märkte standardisieren die Zusammenarbeit von Marktteilnehmern und vermitteln Angebot und Nachfrage durch Preissignale. Unternehmen erkennen Marktpotentiale und koordinieren Angestellte durch Weisungen von Vorgesetzten. Open Source koordiniert die Entwicklung von Open Source Software, indem Entwickler sich selbst Aufgaben aussuchen und bearbeiten. Open Source weiß die Kreativität von Entwicklern effektiver einzusetzen, als Märkte oder Unternehmen (Benkler 2002).

[2] Eine Open Source Library ist Open Source Software die von anderer Software eingebunden wird. Beispielsweise ist Kryptographie ein komplexes Problem und Entwickler sparen Zeit wenn sie dafür auf eine Library, wie OpenSSL, zurückgreifen.

[3] Benkler schreibt über Peer Produktion und nutzt Open Source als Beispiel.

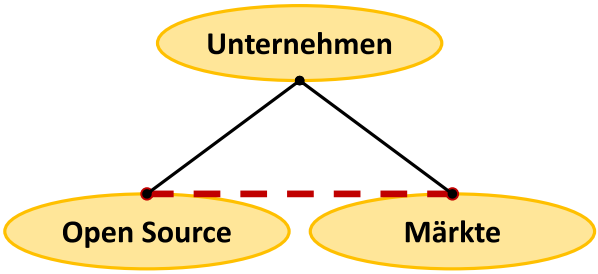

Abb. 1 Die drei Kollaborationsmechanismen sind miteinander verbunden; Open Source und Märkte aber nur indirekt durch Unternehmen

Als Idealtypen existieren Märkte, Unternehmen und Open Source nicht als ausschließende Alternativen, sondern sie interagieren miteinander (siehe Abb. 1). Unternehmen wägen ab, ob es günstiger ist, selbst zu produzieren oder am Markt einzukaufen (Coase 1937). Ein Manager, der diese Entscheidung fällt, verbindet somit ein Unternehmen mit Märkten und wägt ab, welche Vorgehensweise zu wählen ist. Open Source erfährt zunehmend Beteiligung von Unternehmen (Fitzgerald 2006; Germonprez et al. 2017). Unternehmen verwenden und entwickeln Open Source Software, weil es Entwicklungskosten reduziert, Entwicklungszeiten kürzt, und industrieübergreifende Standards etabliert. Unternehmen konkurrieren und differenzieren ihre Produkte und Dienstleistungen indem sie proprietäre Software-Anpassungen oder Module entwickeln (Lerner und Schankerman 2010). Märkte sind mit Unternehmen und Unternehmen sind mit Open Source direkt verbunden. Unternehmen haben durch diese Verbindungen eine vermittelnde Rolle. Es fehlen jedoch effektive Marktmechanismen für Open Source. Ziel dieser Arbeit ist die dritte Verbindung in Form eines Marktplatzes für Open Source zu beschreiben.

3 Vorgehensweise

Wir beteiligen uns als „Engaged Researchers“ (Van de Ven 2007) aktiv in Open Source Projekten. Seit 20 Jahren nehmen wir verschiedene Rollen ein, einschließlich Mitglied, Entwickler und Maintainer. Der Erstautor erforscht seit mehr als drei Jahren die Kooperation zwischen Unternehmen und Open Source Projekten, hat über 100 Seiten Fieldnotes gesammelt und 40 Interviews durchgeführt. Er hat sein ökonomisches Verständnis während einer Ausbildung zum Bankkaufmann und im Studium der Betriebswirtschaftslehre ausgebildet. Die zweite Autorin hat ihren Doktor im Bereich der Software Economy erlangt. Der dritte Autor war Technical Editor für das Linux Journal und Editor in Chief für Linux World. Er hat für Zeitschriften und Blogs geschrieben, inklusive LinuxWorld, Linux Today, Linux Journal und opensource.com. Insgesamt haben wir tiefe Einblicke in die Kollaborationsmechanismen um Open Source gewonnen und selbst daran teilgenommen.

Dieses tiefe Verständnis zu Open Source und Marktmechanismen bildet die Grundlage für unsere Forschung. Die Zweitautorin und der Drittautor haben gemeinsam die Idee für einen Marktplatz für Open Source Software Issues entwickelt. Das Projekt wurde von Mozilla finanziert, um einen Prototyp zu entwickeln. Der Prototyp wurde als Lösung für ein praktisches Problem und unter theoretischer

Berücksichtigung von Kollaborationsmechanismen in Open Source Projekten entwickelt. Der Prototyp ist unter der Mozilla Public License Version 2.0 lizensiert und auf GitHub verfügbar.[4] Der vierte Autor ist der Hauptentwickler des Prototypen. Alle Autoren waren an der konzeptionellen Debatte beteiligt. Das Ziel des Prototypen ist, die Idee des Marktplatzes auszugestalten, zu implementieren und zu testen (Peffers et al. 2007). Der Prototyp ist die Verkörperung unserer Ideen und fokussierte als Artefakt die Diskussionen des Autorenteams und deckte unterschiedliche Ansichten auf. Wir hielten wöchentlich Videokonferenzen, um gemeinsam über Entwicklungsfortschritte und Hindernisse zu reflektieren, neue Funktionen zu diskutieren und Szenarien durchzuspielen.

Wir haben 17 Szenarien entwickelt, um die Funktionsweise des Marktplatzes und die Auswirkung auf Open Source Projekte durchzuspielen.[5] In Gesprächen mit Mitgliedern von Open Source Projekten auf mehr als zehn Konferenzen haben wir die Szenarien immer wieder verwendet, um die Idee des Marktplatzes zu erklären und um Feedback zu erhalten. Das Feedback war überwiegend positiv. Viele sehen einen Nutzen in unserem Marktplatz. Negatives Feedback warnte vor eventuellen negativen Einflüsse auf bestehenden Open Source Kollaborationen. Das Benutzerinterface Design haben wir einer Nutzbarkeitsstudie mit 20 Versuchsteilnehmern im Rahmen des Mozilla Festivals 2017 unterzogen. Anhand von Rückmeldungen haben wir eine neue und vereinfachte Benutzeroberfläche entwickelt. Aktuell sprechen wir mit Open Source Projekten, die Interesse haben unseren Prototypen einzusetzen. Die in diesem Paper entwickelten Designprinzipien sind somit das Ergebnis von theoretischen und praktischen Überlegungen, sowie von Validierung durch potentielle Benutzer (Sein et al. 2011).

4 Modellierung eines Marktplatzes für Open Source Software Issues

Das Ziel unserer Arbeit ist es, Marktmechanismen zu beschreiben, die die Kollaborationsprinzipien von Open Source bewahren und dabei die Finanzierung verbessern. Unsere Kernidee ist ein Kontrakt, der Ähnlichkeiten mit einem Future aus dem Finanzwesen hat. Ein Future ist ein Vertrag zwischen einem Käufer und Verkäufer über die zukünftige Lieferung einer Ware (z. B. Getreide) zu einem vorab festgelegten Preis (Carlton 1984). Als standardisierte Verträge werden Futures an Warenterminbörsen gehandelt, wobei Qualität, Menge, sowie Lieferort (e. g. Getreidesilostandort) für alle gleich festgelegt sind. Futures sind dafür bekannt, Preiserwartungen der Marktteilnehmer abzubilden und neue Informationen frühzeitig in die Preisbildung einzubeziehen (z. B. Getreidepreise steigen, wenn ein Unwetter die Ernte verhagelt) (Carlton 1984).

Wir wandeln das Konzept des Futures dahingehend ab, dass keine Ware geliefert wird, sondern der Status eines Issues (offen oder geschlossen) den Wert und die Auszahlung eines Kontrakts bestimmt. Kontrakte werden, genauso wie Futures, als

[4] Link zum Prototypen des Marktplatzes: https://github.com/bugmark/.

[5] Die Szenarien sind öffentlich dokumentiert unter: https://github.com/bugmark/exchange/wiki/Design#user-stories.

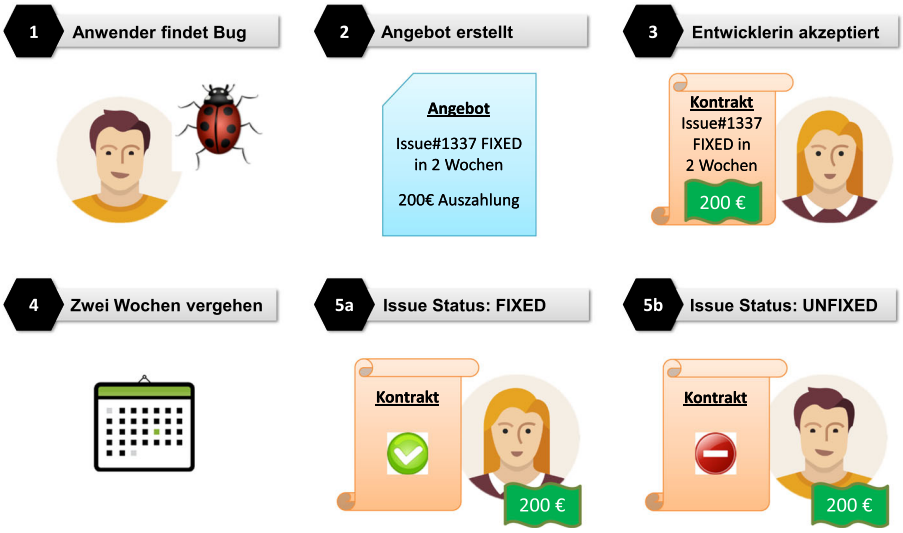

Abb. 2 Bildliche Darstellung von Beispiel 1

standardisierte Verträge frei am Markt gehandelt. Wir erweitern durch Kontrakte die Bedeutung von Issues, indem wir zur Koordinierungsfunktion von Issues auch die Koordinierungsfunktion und Finanzierungsfunktion eines Marktplatzes hinzufügen. Die Funktionsweise des Kontrakts wird in den folgenden Ausführungen näher erläutert. Wir beginnen unsere Beschreibung mit dem Beispiel 1 (siehe Abb. 2 für die Ablauffolge und Abb. 3 für eine visuelle Repräsentation des Kontrakts aus dem Beispiel):

Beispiel 1: (Basisfall Bugfix) Schritt 1: Anwender Adam findet einen Software Bug. Adam beschreibt den Bug im Issue-Tracker (Issue #1337) und ruft einen Marktplatz auf. Schritt 2: Adam erstellt ein Angebot für die Auszahlung von 200 € mit einem Fälligkeitsdatum in zwei Wochen. Adam kauft 200 Einheiten – je 1 € mögliche Auszahlung – zum Einheitspreis von 0,80 € und hinterlegt 160 € auf einem Treuhandkonto. Schritt 3: Entwicklerin Beth sieht das Angebot und hat Zeit den Bug zu fixen. Beth kauft die 200 Einheiten zum Einheitspreis von 0,20 € und hinterlegt 40 € auf dem Treuhandkonto. Der Kontrakt ist zustande gekommen: Adam besitzt die UNFIXED Position und Beth besitzt die FIXED Position. Schritt 4: Zwei Wochen vergehen, während Beth an einem Fix arbeitet und Adam darauf wartet. Schritt 5: Am Fälligkeitsdatum gibt es zwei mögliche Ergebnisse: der Bug ist gefixt oder bleibt ungefixt. Falls der Bug gefixt wurde, wurde auch das dazugehörige Issue #1337 geschlossen. In diesem Fall erhält Beth die Belohnung von 160 € und ihre hinterlegten 40 €. Andernfalls verliert Beth ihre hinterlegten 40 € und Adam erhält die vollen 200 €.

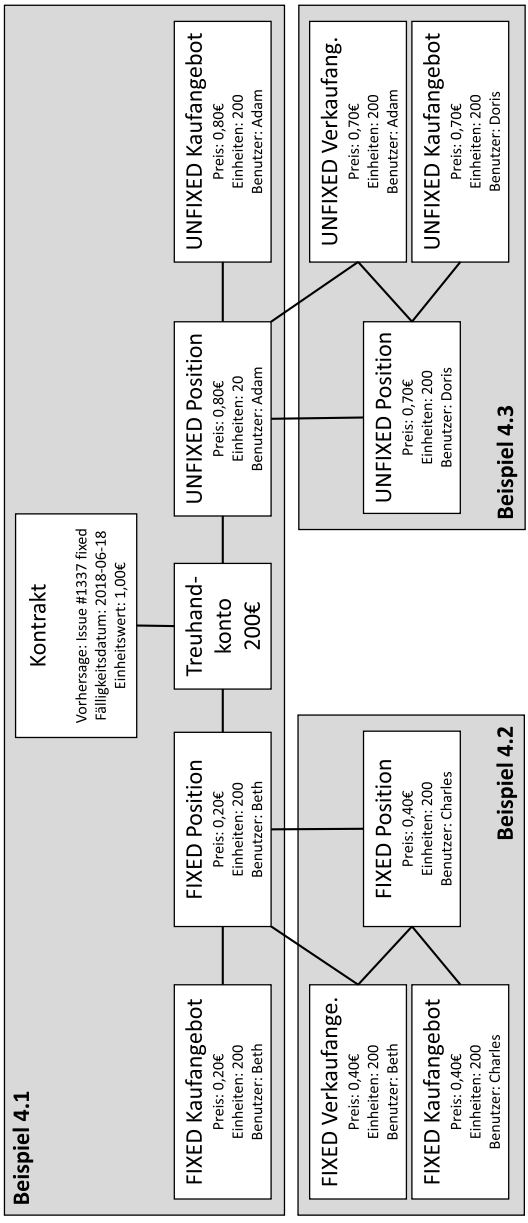

Abb. 3 Graphische Darstellung eines Kontrakts, die zeigt, wie die Bestandteile zusammenhängen und mit der Zeit entstehen (siehe Beispiele). Ein Kontrakt hat eine Vorhersage und ein Fälligkeitsdatum. Darunter werden Treuhandkonten mit Positionen angehängt, die festhalten, welche Benutzer ein Anrecht auf die Auszahlung haben. Die Besitzer der Positionen können diese weiter veräußern (Beispiele 2 und 3), ohne die andere Seite zu beeinflussen oder den Wert des Treuhandkontos zu verändern

4.1 Teillösungen

Viele Probleme erfordern zur Lösung unterschiedliche Kenntnisse. Zum Beispiel kann ein Bug Kenntnisse sowohl in Datenbanken als auch in Kryptographie erfordern. In solchen Fällen lassen sich Probleme am besten lösen, wenn Entwickler mit entsprechenden Kenntnissen zusammenarbeiten und Informationen austauschen. Ohne Marktplatz sind Entwickler darauf angewiesen, eigene Beziehungen zu aktivieren oder zu hoffen, dass andere Entwickler zufällig auf das Issue aufmerksam werden. Forschungsergebnisse haben gezeigt, dass in Open Source schwierige Probleme erst gelöst werden, wenn kleine Teilprobleme gelöst sind (Howison und Crowston 2014). Mit einem Marktplatz könnten Entwickler zusätzlich durch Preissignale Aufmerksamkeit auf Issues lenken mit denen sie Hilfe benötigen. Marktmechanismen könnten somit die Koordinierung von Teillösungen in Open Source verbessern und schwierigere Probleme lösbar machen.

Proposition 1 Ein Marktplatz für Open Source sollte die Zusammenarbeit von Entwicklern unterstützen, die jeweils Teilprobleme lösen (Beispiel 2).

Beispiel 2: (Teillösung) Wie zuvor (Beispiel 1) sind Adam (160 € für die UNFIXED Position) und Beth (40 € für die FIXED Position) einen Kontrakt eingegangen, der eine Auszahlung von 200 € in zwei Wochen vorsieht in Abhängigkeit vom Status des Issues #1337. Nach einer Woche bemerkt Beth, dass ihr wichtige Fähigkeiten fehlen, die zur Lösung des Bugs notwendig sind. Sie veröffentlicht ihre Teillösung und verkauft ihre FIXED Position. Charles kauft Beth die 200 Einheiten zum Einheitspreis von 0,40 € ab. Beth hatte ursprünglich 40 € hinterlegt und erhält 80 €, verdient somit 40 € für ihre Arbeit. Die zweite Woche vergeht, während Charles an einem Fix arbeitet. Am Fälligkeitsdatum gibt es zwei mögliche Ergebnisse: der Bug ist gefixt oder bleibt ungefixt. Falls der Bug gefixt wurde, erhält Charles als Belohnung die Auszahlung von 200 € und verdient somit 120 €. Andernfalls verliert Charles seine 80 €. Adam erhält die gesamte Auszahlung von 200 € und verdient 40 €. In jedem Fall hat Beth 40 € verdient.

4.2 Flexibilität

Das vorherige Beispiel 2 zeigt, dass die Entwicklerin Beth die Flexibilität hat, über den Marktplatz ihre Position zu veräußern, ohne die Kontraktbedingungen für Adam zu verändern. Flexibilität ist eine grundlegende Eigenschaft der Kollaborationsmechanismen in Open Source, was von Moment zu Moment eine andere Entscheidung bedeuten kann (Benkler 2002). Ein fairer und gleichberechtigter Marktplatz sollte es Marktteilnehmern auf beiden Seiten erlauben, ihre Position zu veräußern und somit aus dem Kontrakt auszusteigen. Das folgende Beispiel 3 beschreibt, dass auch Anwender, wie Adam, Flexibilität bewahren.

Proposition 2 Ein Marktplatz für Open Source sollte Marktteilnehmern die Flexibilität geben, jederzeit von einem Kontrakt zurückzutreten (Beispiel 3).

Beispiel 3: (Flexibilität) Wie zuvor (Beispiel 1) sind Adam (160 € für die UNFIXED Position) und Beth (40 € für die FIXED Position) einen Kontrakt eingegangen, der eine Auszahlung von 200 € in zwei Wochen vorsieht in Abhängigkeit vom Status des Issues #1337. Nach zwei Tagen beschließt Adam, dass er sein Geld zurück möchte, weil er auf eine andere Software wechselt. Er verkauft seine UNFIXED Position an Doris. Doris kauft die 200 Einheiten der UNFIXED Position für einen Einheitspreis von 0,70 € und bezahlt somit Adam 140 €. Adam hat somit 20 € für den Bugfix beigesteuert während Doris 140 € für die verbleibende Zeit bezahlt. Beth bleibt von der Transaktion unberührt und entwickelt weiter.

4.3 Software Abhängigkeiten

Ein Software-Feature A hängt manchmal von einem anderen Software-Feature B ab. Solche Abhängigkeiten werden in Issue-Trackern abgebildet und verhindern, dass Feature A vor Feature B fertig gestellt wird. Die Koordinierung wird erschwert, wenn die Features A und B in unterschiedlichen Open Source Projekten (z. B. in einer Library) entwickelt werden. In unserem projektübergreifenden Marktplatz kann ein Entwickler, der die FIXED Position eines Kontrakts für Feature A besitzt (mit der Absicht, die Arbeit selbst zu erledigen), ein neues Angebot für die Fertigstellung von Feature B erstellen. Weil der Entwickler von der Fertigstellung des Features B abhängt, wäre ein früheres Fälligkeitsdatum vor dem eigenen Fälligkeitsdatum für Feature A zu wählen. Die Abhängigkeit von Features A und B sind nur im Issue-Tracker vermerkt, nicht jedoch in den Kontrakten, was die Standardisierung und Handelbarkeit von Kontrakten bewahrt.

Proposition 3 Ein Marktplatz für Open Source sollte Entwickler darin unterstützen, Hilfe von anderen Entwicklern anzufordern.

4.4 Mehrere Kontrakte

Ein Projekt hat häufig mehrere Issues für die gleiche Software. Mehrere neue Features bekommen jeweils ein eigenes Issue und erlauben dadurch getrennte Kontrakte. Die Kontrakte können das gleiche Fälligkeitsdatum haben, um sicher zu stellen, dass die Features im nächsten Release berücksichtigt werden können.

Es ist auch möglich, mehrere Kontrakte für das selbe Issue abzuschließen (z. B. mehrere Kontrakte, die ausgezahlt werden, wenn Issue #1337 gefixt wurde). Insbesondere durch das Sammeln von mehreren Kontrakten auf einem Issue könnte ein Marktplatz Geld für die Belohnung von Innovationen anhäufen. Zum einen können solche Kontrakte für das gleiche Fälligkeitsdatum zwischen verschiedenen Marktteilnehmern abgeschlossen werden. Dies ermöglicht es, größere Summen einzusammeln, beispielsweise wenn weitere Anwender Bereitschaft zeigen wollen, für ein Issue zu bezahlen. Es ermöglicht auch, mehrere Entwickler für ihre Teilarbeit zu entlohnen. Zum anderen können solche Kontrakte verschiedene Fälligkeitsdaten haben. Dies ist hilfreich, um über den Marktplatz festzustellen, zu welchem Zeitpunkt Marktteilnehmer es am wahrscheinlichsten halten, dass das Issue gefixt werden kann (aus Entwicklersicht) oder sein sollte (aus Benutzersicht). Über den

Marktplatz bestimmen Entwickler und Benutzer somit den Preis und eine Deadline für ein Feature.

Proposition 4 Ein Marktplatz für Open Source sollte mehrere Kontrakte erlauben.

4.5 Kaution

Bei Gemeingütern besteht häufig das Trittbrettfahrerproblem. Dabei kann jeder frei von einem Gemeingut profitieren, hat aber selbst keinen Anreiz zum Gemeingut beizutragen. Eine Lösung bietet der Dominant Assurance Vertrag (Tabarrok 1998), wobei ein Projektmanager Agenten bezahlt, die eine Kaution hinterlegen müssen, die sie nur zurückerhalten, wenn sie tatsächlich zum öffentlichen Gut beigetragen haben. Spieltheoretische Analysen zeigen, dass ein Dominant Assurance Vertrag die Auszahlungen so anpasst, dass die beste Strategie für alle Marktteilnehmer ist, zu Gemeingütern beizutragen (Tabarrok 2017). Wir übertragen die Idee auf Open Source, wo ein Entwickler eine Kaution hinterlegen sollte, um für einen Bugfix bezahlt zu werden.

In Beispiel 1 haben der Anwender Adam und die Entwicklerin Beth jeweils eine Kaution von 160€ und 40€ hinterlegt. Wird der Bugfix nicht rechtzeitig fertig, bekommt der Anwender die 40€ als Entschädigung und Gewinn zusätzlich zur eigenen Kaution ausgezahlt. Die Entwicklerin zeigt mit der Kaution ihre Gewissheit, den Bugfix zum Fälligkeitsdatum fertigstellen zu können.

Aus Sicht eines Entwicklers beseitigt die Kaution des Kontraktpartners das Ausfallrisiko. Außerdem stellt der Dominant Assurance Vertrag für Entwickler eine einfache Entscheidung dar. Der Entwickler kann zum einen seine Zeit investieren, sich an dem Bugfix beteiligen, und sich die Kaution des Anwenders verdienen. Zum anderen kann er seine Kaution riskieren und sich nicht an dem Bugfix beteiligen. In diesem Fall wäre es ratsam, die eigene Position auf dem Marktplatz an einen anderen Entwickler zu veräußern, um das Verlustrisiko zu minimieren. Gleichzeitig erhält ein anderer Entwickler den Anreiz, sich an dem Bugfix zu beteiligen.

Aus Sicht eines Anwenders bindet die Kaution Kapital. Erst am Fälligkeitsdatum entscheidet der Kontrakt, ob der Anwender mit seiner Kaution den Entwickler bezahlt. Andernfalls erhält der Anwender auch die Kaution des Entwicklers, als Gewinn und Entschädigung für die Nichtlieferung des Bugfixes.

Ein interessanter Sonderfall sind Spekulanten, die selbst nur an Gewinn nicht aber einem Issue interessiert sind. Angenommen ein Spekulant glaubt daran, dass ein Bugfix nicht vor Fälligkeitsdatum fertig gestellt werden kann. Mit der Absicht, seine Kaution zurück zu bekommen, könnte er eine geringere Kaution von einem Entwickler verlangen. Beispielsweise könnte er 195€ hinterlegen und vom Entwickler nur 5€ Kaution verlangen. Liegt der Spekulant richtig, gewinnt er 5€. Andernfalls hat er vielleicht eine seltene Lösung für ein schwieriges Problem finanziert. In einem Marktplatz mit Dominant Assurance Verträgen werden der Glaube, dass ein Bugfix möglich ist und dass ein Bugfix nicht möglich ist, gleich behandelt und schaffen beide den Anreiz, einen Bugfix zu entwickeln.

Proposition 5 Ein Marktplatz für Open Source sollte eine Kaution von Marktteilnehmern verlangen.

4.6 Entkopplung von Arbeit und Auszahlung

Die Auszahlung eines Kontrakts ist allein vom Status eines Issues abhängig. Zum Fälligkeitsdatum erhält der Besitzer der entsprechenden Position die Auszahlung, unabhängig davon, wer an der Lösung beteiligt war. Diese Entkopplung von Arbeit und Auszahlung ermöglicht interessante Szenarien.

Beispielsweise hat jedes Open Source Projektmitglied einen Informationsvorteil und kann damit auf dem Marktplatz Geld verdienen. Projektmitglieder, die sich um die Verwaltung, Kategorisierung, und Dokumentation von Issues kümmern, sind wichtiger Bestandteil von Open Source Projekten (Ohira und Yoshiyuki 2013). Ihre Tätigkeiten werden kaum wahrgenommen oder belohnt. Mit dem Marktplatz könnten sie FIXED Positionen kaufen, geeignete Entwickler finden und die Positionen mit Gewinn weiter veräußern. Dadurch wird ihre Koordinierungstätigkeit unterstützt und finanziert.

Proposition 6 Ein Marktplatz für Open Source sollte Arbeit und Auszahlung entkoppeln.

4.7 Anonymität

In Open Source Projekten kennen sich die Mitglieder häufig untereinander, treffen sich auf Konferenzen und etablieren ein Arbeitsverhältnis (Crowston et al. 2007). Projektmitglieder bauen sich einen Ruf auf, der bestimmt, welche Rollen sie in einem Open Source Projekt einnehmen können. Etablierte und respektierte Mitglieder sind prädestiniert dazu, Maintainer zu werden (Faraj et al. 2015). Als solche entscheiden sie, welche neuen Software-Features akzeptiert und welche Issues dadurch geschlossen werden.

Die Identität von Marktteilnehmern könnte zu Preisunterschieden führen. Beispielsweise könnte ein Maintainer wegen seinem guten Ruf eine geringere Kaution hinterlegen müssen. Neue und unbekannte Marktteilnehmer würden dadurch benachteiligt. Ein neutraler Marktwert für Issues kann so nicht ermittelt werden. Wir wollen nicht beeinflussen, wie Maintainer und Entwickler in Projekten interagieren und schlagen deshalb vor, die Marktteilnehmer nicht zu identifizieren. Der Preis sollte das einzige Koordinierungsmerkmal sein. Außerdem braucht es keine Identität, da die Kaution das Ausfallrisiko beseitigt.

Proposition 7 Ein Marktplatz für Open Source sollte die Anonymität von Marktteilnehmern gewährleisten.

4.8 Integration mit Bestehenden Open Source Praktiken

Open Source Projekte haben ihre eigenen Praktiken entwickelt (Elliott und Scacchi 2008; Kelty 2008). Dies umfasst geteilte Wertvorstellungen, etablierte Kollaborationsplattformen und erprobte Workflows. Jedes Open Source Projekt etabliert angepasste Varianten dieser Praktiken (Markus 2007). Wenn die etablierten Praktiken gestört werden und Projektmitglieder mit den neuen Praktiken nicht einverstanden sind, dann können sie das Projekt forken und ein neues Projekt gründen, in dem sie die Praktiken nach ihren eigenen Vorstellungen gestalten können (Gamalielsson und Lundell 2014). Es ist daher wichtig, bestehende und abgestimmte Open Source Praktiken zu unterstützen und nicht zu stark zu verändern. Konkret sollten Open Source Projekt Maintainer weiterhin unabhängig Entscheidungen über die Softwareentwicklung treffen können und bestehende Kollaborationsplattformen sollten wie gewohnt weiterverwendet werden können. Unser Marktplatz gewährleistet diese Anforderung, indem wir lediglich den Status von Issues auslesen und keine Veränderung an den Werkzeugen der Open Source Projekte erfordern.

Proposition 8 Ein Marktplatz für Open Source sollte in bestehende Open Source Praktiken integriert werden.

5 Proof-of-Concept Implementierung

Wir haben einen Prototypen eines Marktplatzes entwickelt. Mit dieser Proof-of-Concept Implementierung demonstrieren wir die Umsetzbarkeit unserer Design Features (Hudson und Mankoff 2014). Der Softwareprototyp nutzt das Ruby on Rails Framework, speichert Daten in einer PostgreSQL Datenbank und bietet eine webbasierte Benutzeroberfläche (HTML, CSS und JavaScript). Die event-driven Architektur ist inspiriert von der Blockchain Technologie, um einen dezentralisierten Marktplatz zu ermöglichen. Dazu können Events in einer Blockchain unveränderbar und verteilt gespeichert werden, sodass Manipulationen von Plattformbetreibern aufgedeckt werden können. Weiterhin ist unser Prototyp mit bestehenden Open Source Kollaborationsplattformen, wie zum Beispiel GitHub, integriert. Statusänderungen von GitHub Issues werden in den Markt eingelesen. Angebote im Marktplatz werden in GitHub Issues als SVG-Bilder in Kommentaren angezeigt, die dynamisch mit aktuellen Daten vom Marktplatz geladen werden.

Der wichtigste Teil der Implementierung ist der Kontrakt. Ein Kontrakt regelt die Rechte und Pflichten der Marktteilnehmer. Die Design Features müssen in Kontrakten berücksichtigt werden, da diese das Verhalten von Open Source Projektmitgliedern beeinflussen sollen. Wir beschreiben das Kontrakt Design, indem wir das Leben eines Kontrakts von Erstellung bis Fälligkeit durchlaufen (angelehnt an Beispiel 1) aber komplexere Anwendungsfälle sind möglich. Abb. 3 zeigt die Zusammenhänge der Bestandteile eines Kontrakts.

Ein neuer Kontrakt wird erstellt, wenn zwei passende Kaufangebote zusammentreffen. Die Kaufangebote müssen für unterschiedliche Seiten sein – für FIXED und UNFIXED Positionen – müssen übereinstimmende Preise, Anzahl der Ein-

heiten, Vorhersagen, und Fälligkeitsdaten haben. Einheitspreise der Kaufangebote müssen in der Summe den Einheitswert von 1 € ergeben. Zum Beispiel könnte der Einheitspreis für eine FIXED Position 0,80 € und für eine UNFIXED Position 0,20 € betragen. Beide Kaufangebote hinterlegen gemeinsam den gesamten Auszahlungsbetrag (jeder seinen Anteil). Der Einheitspreis bestimmt, welchen Anteil die Marktteilnehmer als Kaution auf einem Treuhandkonto hinterlegen müssen. Das Treuhandkonto beinhaltet den gesamten Auszahlungsbetrag. Die Anzahl der Einheiten geben an, wieviel ein Marktteilnehmer kaufen möchte. Gekaufte Einheiten werden als Positionen abgebildet.

Ein Kontrakt wird durch die Vorhersage und das Fälligkeitsdatum definiert. Für jeden Kontrakt gibt es eine beliebige Anzahl an Treuhandkonten, Positionen und Angebote. Dieses Design erlaubt es Marktteilnehmern, jede beliebige Anzahl an Kontrakteinheiten (auch Teilpositionen) zu kaufen und zu verkaufen, sofern sie zum selben Kontrakt gehören. Kleinere Angebote von Anwendern können zu größeren Kontrakten für Entwickler gebündelt werden.

Eine Position verkörpert das Recht, am Fälligkeitsdatum eine Auszahlung zu bekommen, falls die Vorhersage entsprechend eingetreten ist. Die Vorhersage „Issue #1337 ist am Fälligkeitsdatum FIXED“ ist ein typisches Beispiel. Der Besitzer der FIXED Position bekommt die Auszahlung, falls Issue #1337 geschlossen wurde, andernfalls erhält der Besitzer der UNFIXED Position die Auszahlung. Die Auszahlung wird vom Treuhandkonto vorgenommen. Ob ein Issue gefixed ist, entscheidet der Projekt Maintainer nach etablierten Praktiken im Open Source Projekt und auf Grundlage technischer Abwägungen.

6 Diskussion und Implikationen

Kontrakte, die auf den Status von Issues handeln, haben einige Vorteile, können aber auch ungewünschte Nebeneffekte haben. Grundannahme ist, dass Open Source Projektmitglieder den Preissignalen folgen und Positionen kaufen, sofern sie davon profitieren. Projektmitglieder haben einen Informationsvorteil, da sie den zukünftigen Status eines Issues beeinflussen können. Während Insiderhandel auf Kapitalmärkten illegal ist, nutzen wir dies als Anreizinstrument.

6.1 Preissignale als Koordinierungsmechanismus

Benkler (2002) hat Open Source als dritten Koordinierungsmechanismus neben Märkten und Unternehmen identifiziert. Unser Marktplatz schafft die fehlende Verbindung zwischen diesen drei. Unternehmen und Open Source Projekte können unseren Marktplatz nutzen, um Arbeit zu koordinieren. Die maßgebliche Eigenschaft des Marktes ist die Schaffung von Preissignalen, die Aufmerksamkeit lenkt. Issues können große Handelsvolumen erzeugen, weil sie für viele andere Projekte wichtig sind, weil viele Unternehmen von dem Fix eines Issues profitieren würden oder weil ein Issue eine fatale Sicherheitslücke ist und kritische Geschäftsdaten bedroht. Entwickler können den Preissignalen folgen, um möglichst viel Geld im Marktplatz zu verdienen und bearbeiten dabei die am höchsten bewerteten Issues zuerst.

Der Martkplatz kann Koordination zwischen Open Source Projekten effektiver machen. Bisher war eine Anfrage bei einem anderen Open Source Projekt nicht auffälliger als andere Anfragen. Angenommen, ein Open Source Projekt wird durch Preissignale auf ein viel gewünschtes Feature aufmerksam, das aber als Voraussetzung die Anpassung einer unabhängig entwickelten Open Source Library hat. Ein Projektmitglied kann kostenneutral (nicht risikofrei) die Preissignale weiterleiten indem es FIXED Positionen im eigenen Projekt und UNFIXED Positionen für die Library kauft.

Ein weiterer Vorteil des Marktplatzes ist, dass Spekulanten (Personen, die von Preisfluktuationen profitieren möchten) Kontrakte kaufen und verkaufen können, ohne selbst an der Softwareentwicklung beteiligt zu sein. Preisschwankungen und -unterschiede können ausgenutzt werden. Die Aufmerksamkeit durch Spekulanten und der Zufluss von Geld in den Marktplatz hat positive Auswirkungen auf die Open Source Projekte. Spekulanten, die darauf wetten, dass Issues nicht gefixt werden kaufen die UNFIXED Position und schaffen dadurch Anreize für Entwickler, diese Issues zu fixen.

Die Entkopplung von Arbeit und Auszahlung ermöglicht neue Beschäftigungsverhältnisse. Ein Unternehmer kann FIXED Positionen eines Open Source Projektes kaufen und gleichzeitig Entwickler einstellen, die die Issues bearbeiten. Er handelt auf dem Marktplatz, folgt Preissignalen, trägt das Risiko und weist die angestellten Entwickler an. Er bezahlt die Entwickler aus den Kontraktauszahlungen und verbucht Überschüsse als Gewinn. Die Angestellten genießen die Vorzüge einer Festanstellung. Das Open Source Projekt wird weiterhin von den gleichen Maintainern wie zuvor verwaltet und gelenkt, die auch weiterhin entscheiden welche Lösungen angenommen und welche Issues dadurch geschlossen werden.

6.2 Auswirkung auf Open Source Projekte

Als Vermittler zwischen zahlungswilligen Anwendern und Open Source Projekten erfüllen Unternehmen eine wichtige Rolle in der Identifikation von Bedürfnissen und dem Ausloten der Zahlungsbereitschaft. Dafür haben Unternehmen einen Einfluss auf Open Source Projekte. Die Einführung eines Marktplatzes hat das Potential, Anwender mit Open Source Software Entwicklern direkt zu verbinden, was den Einfluss von Unternehmen verringern könnte.

Der Marktplatz könnte es weiteren Bevölkerungsgruppen ermöglichen, sich an Open Source zu beteiligten. Durch unseren Marktplatz könnten Personen einen Lebensunterhalt damit verdienen, an Open Source zu arbeiten, ohne von einem Unternehmen angestellt zu sein. Das könnte die Diversität in Open Source Projekten erhöhen. Der Marktplatz hat das Potential, den mit Open Source erzeugten Wohlstand anders als bisher zu verteilen.

Die Einführung von Marktmechanismen und die Bezahlung von Entwicklern wird von einigen kategorisch abgelehnt (Hansson 2013). Eine Befürchtung ist, dass der Gemeinschaftsgeist verloren geht, wenn die selbst-bestimmte und intrinsisch motivierte Beteiligung der bezahlten Arbeit weicht.

Wo bisher kein Preisschild an Open Source hing, würde ein Marktplatz die Illusion schaffen, dass Open Source Projekte bewertet würden. Das könnte zu Unmut von Projektmitgliedern führen, die ihre Projekte vergleichen. Entwickler könnten kleinere Projekte verlassen, um sich an prestigereicheren und größeren Projekten zu beteiligen. Diese Trends bestehen jetzt schon, auch ohne Marktplatz (Beecher et al. 2009). Die meisten Open Source Projekte werden nicht aktiv entwickelt, weil sie das Interesse der Entwickler verloren haben. Unser Marktplatz könnte hingegen von Benutzern verwendet werden, Preissignale für Open Source Projekte zu senden, woran andernfalls kein Entwickler Interesse hätte.

Unser Marktplatz ist darauf ausgelegt, bestehende Kollaborationspraktiken zu unterstützen. Unser Marktplatzentwurf belässt die Autonomie über die Entwicklung von Open Source Software den Projektmitgliedern, die den Marktplatz ignorieren oder gezielt ihr Handeln darauf ausrichten können. Beispielsweise könnten Kontrakte mit unterschiedlichen Fälligkeitsdaten auf demselben Issue Trittbrettfahrer begünstigen, weil die Auszahlung der Kontrakte lediglich von dem Status eines Issues abhängt. Wenn ein Entwickler beispielsweise, ein früheres Fälligkeitsdatum hat und das Issue fixt, um die Auszahlung zu erhalten, dann zahlen Kontrakte mit späteren Fälligkeitsdaten automatisch auch an andere Halter der FIXED Position aus. Diese Halter haben keinen Anreiz, sich am Bugfix zu beteiligen. Projektmitglieder könnten durch nichtbearbeiten von Issues das Risiko für Trittbrettfahrer gezielt erhöhen und entsprechend gegensteuern. Inwiefern Trittbrettfahrer ein Problem darstellen, hängt von weiteren Faktoren ab, beispielsweise die Beweggründe der Marktteilnehmer und deren Bereitschaft, sich an der Entwicklung zu beteiligen. Weitere Versuche sind notwendig, um das Verhalten der Marktteilnehmer zu verstehen.

7 Fazit und Ausblick

Open Source entwickelt sich ständig weiter. Wir stellen in diesem Paper einen Marktplatz vor, der dafür geeignet ist, die Zusammenarbeit in Open Source auch für neue Herausforderungen anzupassen. Wir haben unseren Marktplatz danach entworfen wie Open Source derzeit funktioniert und wo wir Verfehlungen in Koordinierung und Finanzierung beheben können. Als Konzeptstudie haben wir noch keine Einblicke in die tatsächlichen Auswirkungen, die ein solcher Marktplatz haben würde. Im Moment verhandeln wir mit Interessenten darüber, den Marktplatz für ein Open Source Projekt aufzusetzen. Dazu eignet sich ein Projekt, das von einem Unternehmen betreut wird, das bereit ist, Geld über den Marktplatz an Entwickler zu bezahlen. Wir werden das Experiment wissenschaftlich begleiten und beobachten, welches Marktverhalten sich entwickelt. Gleichzeitig arbeiten wir an einem kontrollierten Laborexperiment, in dem wir mit Studenten testen werden, wie sich die Gegenwart von verschiedenen Marktsignalen auf die Zusammenarbeit auswirken.

Förderung Diese Arbeit wurde von Mozilla unterstützt.
Diese Arbeit wurde von der Alfred P. Sloan Foundation Digital Technology Forschungsbeihilfe „Open Source Health and Sustainability“, Num: 8434 (https://sloan.org/grant-detail/8434) unterstützt.

Literatur

Beecher K, Capiluppi A, Boldyreff C (2009) Identifying exogenous drivers and evolutionary stages in FLOSS projects. J Syst Softw 82:739–750. https://doi.org/10.1016/j.jss.2008.10.026

Benkler Y (2002) Coase's penguin, or, Linux and „the nature of the firm". Yale Law J 112:369–446

Bonaccorsi A, Giannangeli S, Rossi C (2006) Entry strategies under competing standards: hybrid business models in the open source software industry. Manage Sci 52:1085–1098. https://doi.org/10.1287/mnsc.1060.0547

Carlton DW (1984) Futures markets: their purpose, their history, their growth, their successes and failures. J Futur Mark Pre 4(3):237

Coase RH (1937) The Nature of the Firm. Economica 4:386–405

Collins K (2016) How one programmer broke the internet by deleting a tiny piece of code. In: Quartz. https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code/. Zugegriffen: 30. Nov. 2018

Crowston K, Howison J (2005) The social structure of free and open source software development. First Monday. https://doi.org/10.5210/fm.v10i2.1207

Crowston K, Howison J, Masango C, Eseryel UY (2007) The role of face-to-face meetings in technology-supported self-organizing distributed teams. Prof Commun IEEE Trans 50:185–203. https://doi.org/10.1109/TPC.2007.902654

Dahlander L (2005) Appropriation and appropriability in open source software. Int J Innov Manag 9:259–285

Dahlander L, Magnusson MG (2005) Relationships between open source software companies and communities: observations from Nordic firms. Res Policy 34:481–493. https://doi.org/10.1016/j.respol.2005.02.003

Dahlander L, Wallin MW (2006) A man on the inside: unlocking communities as complementary assets. Res Policy 35:1243–1259. https://doi.org/10.1016/j.respol.2006.09.011

Durumeric Z, Kasten J, Adrian D et al (2014) The matter of Heartbleed. In: Proceedings of the 2014 Conference on Internet Measurement Conference. ACM, Vancouver, S 475–488

Eck A (2018) Coordination across open source software communities: Findings from the rails ecosystem. In: Tagungsband Multikonferenz Wirtschaftsinformatik (MKWI) 2018 Leuphana Universität Lüneburg, S 109–120

Eghbal N (2016) Roads and bridges: the unseen labor behind our digital infrastructure. Ford foundation. http://www.fordfoundation.org/library/reports-and-studies/roads-and-bridges-the-unseen-labor-behind-our-digital-infrastructure/. Zugegriffen: 30. Nov. 2018

Elliott MS, Scacchi W (2008) Mobilization of software developers: the free software movement. Inf Technol People 21:4–33

Faraj S, Kudaravalli S, Wasko M (2015) Leading collaboration in online communities. Mis Q 39:393–412

Fitzgerald B (2006) The transformation of open source software. Mis Q 30:587–598. https://doi.org/10.2307/25148740

Gamalielsson J, Lundell B (2014) Sustainability of open source software communities beyond a fork: how and why has the LibreOffice project evolved? J Syst Softw 89:128–145. https://doi.org/10.1016/j.jss.2013.11.1077

Germonprez M, Kendall JE, Kendall KE et al (2017) A theory of responsive design: a field study of corporate engagement with open source communities. Inf Syst Res 28:64–83. https://doi.org/10.1287/isre.2016.0662

Germonprez M, Link GJP, Lumbard K, Goggins S (2018) Eight observations and 24 research questions about open source projects: illuminating new realities. Proc ACM Hum Comput Interact 2:57. https://doi.org/10.1145/3274326

Hansson DH (2013) The perils of mixing open source and money (DHH). http://david.heinemeierhansson.com/2013/the-perils-of-mixing-open-source-and-money.html. Zugegriffen: 30. Nov. 2018

Howison J, Crowston K (2014) Collaboration through open superposition: a theory of the open source way. Mis Q 38:29–50

Hudson SE, Mankoff J (2014) Concepts, values, and methods for technical human–computer interaction research. In: Olson JS, Kellogg WA (Hrsg) Ways of knowing in HCI. Springer, New York, S 69–93
Kelty CM (2008) Two bits: the cultural significance of free software. Duke University Press, Durham
Kelty CM (2013) There is no free software. J Peer Prod 1:3
von Krogh G, Haefliger S, Spaeth S, Wallin MW (2012) Carrots and rainbows: motivation and social practice in open source software development. Mis Q 36:649–676
Lerner J, Schankerman M (2010) The comingled code: open source and economic development. MIT Press, Cambridge
Markus M (2007) The governance of free/open source software projects: monolithic, multidimensional, or configurational? J Manag Gov 11:151–163. https://doi.org/10.1007/s10997-007-9021-x
Mockus A, Fielding RT, Herbsleb JD (2002) Two case studies of open source software development: Apache and Mozilla. ACM Trans Softw Eng Methodol 11:309–346. https://doi.org/10.1145/567793.567795
Nagle F (2018) Learning by contributing: gaining competitive advantage through contribution to crowdsourced public goods. Organ Sci 29:569–587. https://doi.org/10.1287/orsc.2018.1202
Ohira M, Yoshiyuki H (2013) A new perspective on the socialness in bug triaging: a case study of the eclipse platform project. In: Proceedings of the 2013 International Workshop on Social Software Engineering. ACM, New York, S 29–32
O'Mahony S, Ferraro F (2007) The emergence of governance in an open source community. Acad Manage J 50:1079–1106. https://doi.org/10.5465/AMJ.2007.27169153
Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S (2007) A design science research methodology for information systems research. J Manag Inf Syst 24:45–77. https://doi.org/10.2753/MIS0742-1222240302
Rao M, Link GJP, Marti D et al (2018) A trading market to incentivize secure software. In: 17th Annual Workshop on the Economics of Information Security (WEIS) Proceedings Innsbruck. https://weis2018.econinfosec.org/wp-content/uploads/sites/5/2016/09/WEIS_2018_paper_27.pdf Zugegriffen: 30. Nov. 2018
Riehle D (2010) The economic case for open source foundations. Computer (Long Beach Calif) 43:86–90. https://doi.org/10.1109/MC.2010.24
Riehle D, Riemer P, Kolassa C, Schmidt M (2014) Paid vs. volunteer work in open source. In: System Sciences (HICSS), 2014 47th Hawaii International Conference on, S 3286–3295
Scacchi W (2004) Free and open source development practices in the game community. IEEE Softw 21:59–66. https://doi.org/10.1109/MS.2004.1259221
Sein MK, Henfridsson O, Purao S et al (2011) Action design research. Mis Q 35:37–56
Spaeth S, von Krogh G, He F (2015) Research note—perceived firm attributes and intrinsic motivation in sponsored open source software projects. Inf Syst Res 26:224–237. https://doi.org/10.1287/isre.2014.0539
Tabarrok A (1998) The private provision of public goods via dominant assurance contracts. Public Choice 96:345–362
Tabarrok A (2017) Making markets work better: dominant assurance contracts and some other helpful ideas. In: cato unbound. https://www.cato-unbound.org/2017/06/07/alex-tabarrok/making-markets-work-better-dominant-assurance-contracts-some-other-helpful. Zugegriffen: 30. Nov. 2018
Van de Ven AH (2007) Engaged scholarship: a guide for organizational and social research. Oxford University Press, Oxford
Walden G, Harper G (2018) Letter to mr. Zemlin from the one hundred fifteenth congress of the United States committee on energy and commerce. https://energycommerce.house.gov/wp-content/uploads/2018/04/040218-Linux-Evaluation-of-OSS-Ecosystem.pdf. Zugegriffen: 30. Nov. 2018
Wheeler DA (2015) Why open source software / free software (OSS/FS, FOSS, or FLOSS)? Look at the numbers! https://www.dwheeler.com/oss_fs_why.html. Zugegriffen: 30. Nov. 2018