Machbarkeitsstudie

Konsortiale Softwareentwicklung

auf der Basis von Open-Source-Software

Machbarkeitsstudie

Konsortiale Softwareentwicklung auf der Basis von Open-Source-Software

Oktober 2013

Teilnehmer der Studie

Kernteam

Name	Unternehmen	E-Mail
Heinritz, Christof	N-ERGIE Netz GmbH, Nürnberg	christof.heinritz@n-ergie-netz.de
Henke, Klaus-Dieter	CONSULECTRA Unternehmensberatung GmbH, Hamburg	k.henke@consulectra.de
Janeck, Stephan	SWM Services GmbH	janeck.stephan@swm.de
Herdt, Peter	N-ERGIE Netz GmbH, Nürnberg	peter.herdt@n-ergie-netz.de
Meyer, Christian	CONSULECTRA Unternehmensberatung GmbH, Hamburg	c.meyer@consulectra.de
Regenbogen, Gerhard	Verteilnetzbetreiber Rhein-Main-Neckar GmbH & Co. KG, Darmstadt	gerhard.regenbogen@vnb-rmn.de
Rose, Frank	Netrion GmbH, Mannheim	frank.rose@netrion.de
Roth, Michael	RheinEnergie AG, Köln	m.roth@rheinenergie.com
Tuchs, Dr., Michael	enercity, Hannover	michael.tuchs@enercity.de

Experten

Name	Unternehmen	E-Mail
Braun, Dr., Martin	WilmerHale, Frankfurt/M.	martin.braun@wilmerhale.com
Höfer, Hendrik	MicroDoc GmbH, München	hho@microdoc.com
Müller, Ralph	Zwingenberg	ralph.mueller.de@gmail.com
Riehle, Prof. Dr., Dirk	Bayave GmbH, Nürnberg	dirk.riehle@fau.de
Seibt, Richard	Open Source Business Foundation e.V., Nürnberg	richard.seibt@osbf.de

weitere Teilnehmer aus den auftraggebenden Unternehmen

Name	Unternehmen	E-Mail
Geist, Stefan	N-ERGIE Netz GmbH, Nürnberg	stefan.geist@n-ergie-netz.de
Glennemeier, Franz-Josef	enercity, Hannover	franz-josef.glennemeier @enercity.de
Müller, Björn	RheinEnergie AG, Köln	bj.mueller@rheinenergie.com
Schmidt, Kai	Verteilnetzbetreiber Rhein-Main-Neckar GmbH & Co. KG, Darmstadt	kai.schmidt@vnb-rmn.de
Sohst, Oliver	enercity, Hannover	oliver.sohst@enercity.de
Thoma, Detlef	Verteilnetzbetreiber Rhein-Main-Neckar GmbH & Co. KG, Darmstadt	detlef.thoma@vnb-rmn.de

Gesamtinhaltsverzeichnis

Kapitel 1

Einleitung

INHALTSVERZEICHNIS

1 Einleitung

1.1 Ausgangssituation in den beteiligten Unternehmen

Die Energieversorgung in Deutschland wird durch eine Vielzahl von Unternehmen unterschiedlicher Größe und Ausrichtung wahrgenommen. Viele Energieversorger schauen dabei inzwischen auf eine mehr als hundertjährige Unternehmensgeschichte zurück. Einige der heutigen Unternehmen sind dabei in der jüngeren Vergangenheit aus der Fusion mehrerer Einzelunternehmen hervorgegangen. Dieser geschichtliche Hintergrund erklärt die sehr unterschiedlichen Konzernstrukturen. Allen gemein ist die Aufteilung in unterschiedliche Geschäftsfelder, wie z. B. Erzeugung, Handel, Vertrieb, Netze, Informations- und Kommunikationstechnik (IKT) und sonstige Dienstleistungen.

Die Betrachtungen in dieser Studie fokussieren sich auf das Betreiben von Netzen, weil dort sowohl der Bedarf als auch die Möglichkeiten zu veränderten Vorgehen in der IT-Beschaffung gegeben sind. Viele der folgenden Aussagen können jedoch auch auf andere Geschäftsfelder abgebildet werden.

Die Verantwortung für das Betreiben von Netzen liegt in der Regel bei der Netzgesellschaft (Netzbetreiberrolle). Sofern es sich dabei um eine "kleine" Netzgesellschaft handelt, wird der operative Netzbetrieb häufig von einer eigenständigen Netzservicegesellschaft erbracht. Die Netzführung ist in den meisten Fällen in der Netzgesellschaft angesiedelt, kann aber in Einzelfällen auch dem Netzservice oder direkt der Konzernmutter zugeordnet sein.

Wie in allen Unternehmensbereichen hat sich auch beim Betreiben von Netzen der Einsatz von IT-Systemen als eine wesentliche Voraussetzung für eine effiziente Arbeitsorganisation einerseits, aber auch als signifikanter Kostenfaktor andererseits entwickelt.

Da innerhalb von Konzernen eine funktionale Aufgabenteilung üblich ist, erbringen die Konzerngesellschaften häufig ein Vielzahl von Dienstleistungen füreinander. Im Bereich der eingesetzten IT-Systeme (insbesondere im technischen Bereich) führt dies zu den unterschiedlichsten Konstellationen hinsichtlich Eigentum, Administration, Betrieb und Nutzung eines Systems.

Die zeitliche Entwicklung der IT-Landschaften in den einzelnen Unternehmen verlief hinsichtlich der funktionalen Ausprägung sehr inhomogen. Sie war in der Vergangenheit geprägt von knappen und teuren Hardwareressourcen, was häufig eine Beschränkung in der Datentiefe und die Spezialisierung auf besondere (Teil-)Aufgaben zur Folge hatte.

Dies gilt insbesondere für die Betriebsmitteldatenhaltung, da sich der Markt der Entwicklung von speziellen Betriebsmittelinformationssystemen (BIS) nur sehr zögerlich zugewendet hat. Aus Mangel an geeigneten IT-Systemen haben sich unternehmensspezifische Lösungen herausgebildet, die meist den Einsatz mehrerer Systeme erfordern. So verwundert es nicht, dass große Teile der technischen Sach- und Bewegungsdaten (z. B. Instandhaltungsdaten) verteilt im Geografischen Informationssystem, in spartenorientierten Datenbanklösungen und/oder auch in speziellen Modulen von ERP-Systemen gehalten werden. Für die weitere Betrachtung in dieser Studie wurden die jeweils im Einsatz befindlichen Betriebsmitteldatenhaltungen begrifflich zu einem Synomym zusammengefasst: BDH (siehe auch Abschnitte 1.4 bzw. 2.1).

Die IT-Systeme beinhalten heute also immer nur eine Teilmenge aller relevanten Netzdaten, was zwangsläufig zu Redundanzen führt und Medienbrüche in den Geschäftsprozessen verursacht. Um diesem Phänomen entgegenzuwirken, wurde in den vergangenen Jahren teilweise die direkte Kopplung der Systeme über uni- oder bidirektionale Schnittstellen vorangetrieben. Dies ist bis heute nur vereinzelt gelungen. Im technischen Bereich gilt dies besonders für die ursprünglich isoliert voneinander konzipierten, strategischen Systeme (Abbildung 1-1) in den Funktionsbereichen Netzführung, Netzdokumention (Bestandspläne) und operativer Netzbetrieb. Gemeinsam ist aber allen Unternehmen die Systemlandschaft gemäß Abbildung 1-1.

Abb. 1-1: "Strategisches Dreieck"

1.2 Abgeleiteter Handlungsbedarf

Die nach zehn bis fünfzehn Jahren anstehenden Upgrade- oder Migrationsprojekte für jedes einzelne der im vorigen Abschnitt genannten Systeme sind heute hochkomplex und kostspielig. Als Kostentreiber sind der ständig wachsende Umfang neuer Funktionalitäten, aber auch die Anpassung oder Neuentwicklung von Schnittstellen zu anderen Systemen zu nennen. Letzteres ist meistens dadurch bedingt, dass die Systeme auf unterschiedlichen Betriebssystem-Versionen basieren und proprietäre Formate und Protokolle verwendet werden. Allein schon aus diesen Gründen können bei dieser Art von Projekten Projektlaufzeiten von über zwei Jahren entstehen.

Die Netzbetreiber sehen sich dabei in dem Dilemma, dass mit jeder neuen Schnittstelle auch ein Zuwachs an Komplexität (z. B. im Systembetrieb) verbunden ist und dies die Kosten weiter in die Höhe treibt. Die prozessualen Rückwirkungen und die durch die Anpassung entstehenden Kosten sind an dieser Stelle zusätzlich einzukalkulieren.

Da sich die Aussagen dieser Studie schwerpunktmäßig aus der Anwendungssicht der Netzführung ergeben, ist in diesem Zusammenhang der Hinweis wichtig, dass die Mitarbeiter der operativen Netzführung bei ihrer Tätigkeit meistens von mehreren IT-Systemen unterstützt werden. Dabei spielt das Netzleitsystem immer eine zentrale Rolle, der Zugriff auf andere Systeme (z. B. GIS, BDH) ist jedoch die Regel.

Die Energiewende hat auf das Aufgabenspektrum der Netzführung erheblichen Einfluss. Zum Beispiel sorgt der rasante Anstieg dezentraler Einspeisungen (Wind, Photovoltaik) bei den Verteilnetz- und Übertragungsnetzbetreibern zunehmend für Probleme bei der Sicherstellung der Netzstabilität und Versorgungszuverlässigkeit. Die damit zusammenhängenden Tätigkeiten zur vorausschauenden Planung, Netzüberwachung und Durchführung sowie Dokumentation steuernder Eingriffe binden zusätzliche Ressourcen und erfordern neue Funktionen bzw. IT-Werkzeuge.

Die im Einsatz befindlichen Netzleitsysteme, die in der Vergangenheit als relativ autarke und vielfach speziell an die Bedürfnisse des einzelnen Unternehmens angepasste Systeme betrieben wurden, trifft diese Entwicklung besonders schwer, da mit dem Umbau der Netze zu Smart Grids kurz- bis mittelfristig wesentliche funktionale Erweiterungen und ein möglichst standardisierter Datenaustausch bei gleichzeitig steigendem Datenvolumen erforderlich werden.

Selbst wenn von den Netzbetreibern frühzeitig die neuen funktionalen Anforderungen in Lastenheften beschrieben werden, sind die Zeiträume für deren Realisierung häufig zu lang, sodass die notwendige Unterstützung des Betriebspersonals nicht rechtzeitig zur Verfügung steht. Außerdem kollidieren die Kosten der neuen Software (Lizenzgebühren, Wartung und Support) nicht selten mit den regulierungsbedingt knappen Budgets. Darüber hinaus sehen sich die Systemlieferanten zunehmend einem Mangel an Fachkräften ausgesetzt, der dazu führt, dass sowohl Know-how zur Pflege der teilweise überalterten Softwareteile verloren geht, als auch für aktuelle Systemerweiterungen die benötigten Ressourcen fehlen. Daher kann der geforderte Funktionsumfang durch die Systemlieferanten oft nur stufenweise oder mit Einschränkungen realisiert werden.

Die in manchen Fällen durch die Bundesnetzagentur vorgegebenen Fristen zur systemtechnischen Umsetzung der gesetzlichen Vorgaben überfordern bereits heute einzelne Softwarelieferanten und zunehmend auch die Netzbetreiber selbst, denn diese Systemertüchtigungen und -erweiterungen sind vielfach nur mit einer stärkeren Systemintegration realisierbar. Dies gilt nicht nur für technische IT-Systeme.

Die dargestellte Situation wird dadurch verschärft, dass

- es eine hohe Bindung an die einmal gewählten Systeme und deren Hersteller gibt
- die Verwendung von proprietären Formaten und Protokollen in den Systemen den Wettbewerb verhindert und
- ein Anbieterwechsel aufgrund der oben beschriebenen Komplexität und der noch höheren Kosten und Risiken oft ausscheidet.

Mit zunehmendem Handlungsdruck steigt somit die Bereitschaft, neue kostspielige Funktionen immer wieder in proprietäre Systeme zu implementieren. Vor diesem Hintergrund haben sich die Netzführer von sechs Unternehmen (siehe Abbildung 1-2) dazu entschlossen, im Rahmen einer Machbarkeitsstudie die Fragestellung zu prüfen, ob die konsortiale Softwareentwicklung (KSE) auf der Basis von Open-Source-Software (OSS) für die Netzbetreiber einen möglichen Ausweg aus dieser Situation bietet. Diese Alternative sollte es ermöglichen sowohl vorhandene Systeme flexibel weiter zu nutzen, als auch neue Funktionen flexibel modular aufzubauen.

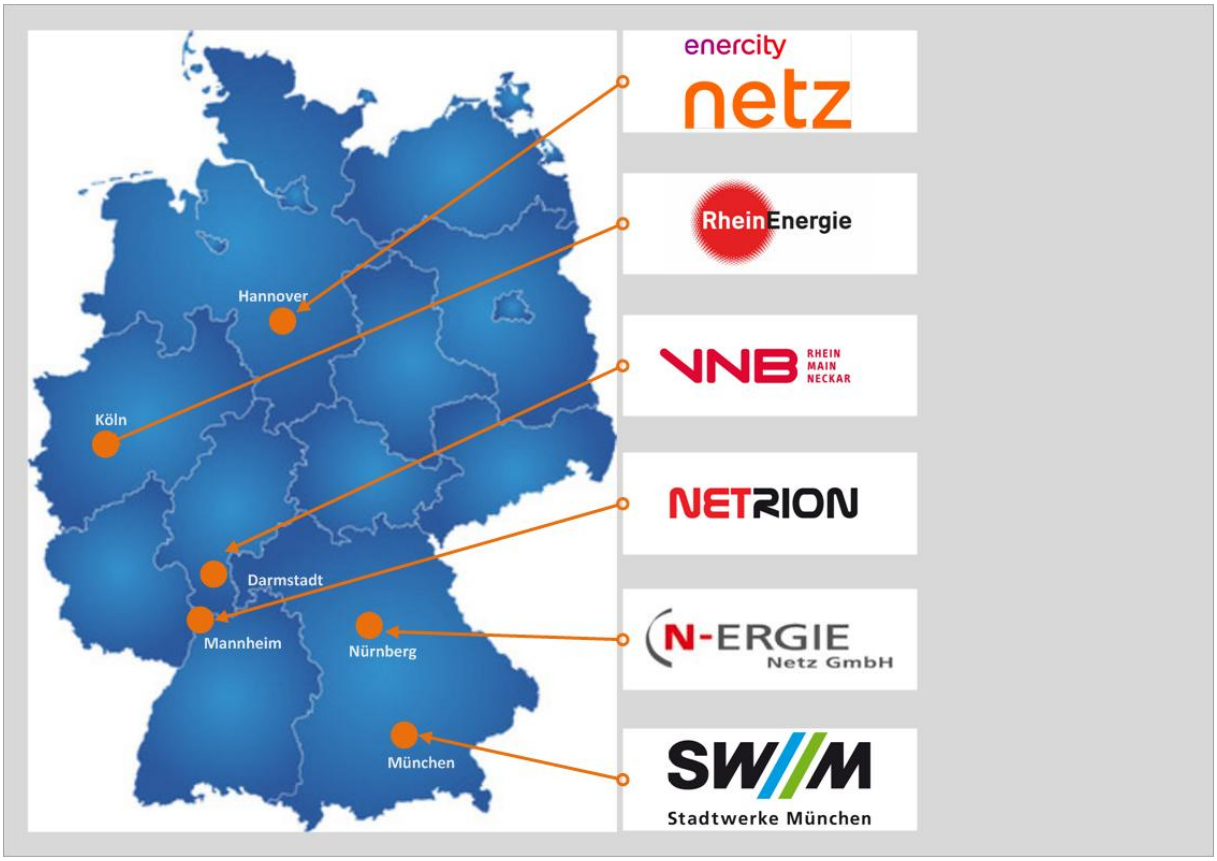


Abb. 1-2: Auftraggeber der Studie

1.3 Aufgabenstellung und Ziel der Studie

Mit der vorliegenden Machbarkeitsstudie soll die Frage beantwortet werden, ob der Einstieg in die konsortiale Softwareentwicklung auf der Basis von Open Source sinnvoll ist.

Zwar existieren in anderen Branchen bereits erfolgreiche Konsortien, neu wäre aber die gemeinsame Erstellung solcher Standards mit Open Source durch eine Gruppe von Netzbetreibern.

Bezogen auf die eingesetzte Software erstreckt sich der Untersuchungsraum auf den Bestand und den Bedarf im Bereich der vorgenannten drei großen IT-Systeme für das Betreiben von Netzen (NLS, GIS, BDH) sowie punktuell auf weitere Systeme für spezielle technische Aufgabenstellungen in deren Umfeld (siehe auch Kapitel 2).

Die Studie untersucht im Einzelnen folgende Aufgabenstellungen:

1. Analyse der Situation in den beteiligten Unternehmen, Problembeschreibung sowie vorhandene Lösungen [Kapitel 2].

2. Es wird die Frage untersucht, ob es mit KSE technische Lösungsmöglichkeiten gibt, die notwendige Qualität (Sicherheit, Funktionalität und Komplexität) der benötigten zukünftigen Software umzusetzen. Es ist zu erwarten, dass die Anforderungen an die Qualität weiter steigen werden.

   Weiterhin wird überprüft, ob die Überwindung der heutigen Systemgrenzen und die Erweiterung um neue systemverbindende Funktionalitäten technisch möglich sind [Kapitel 3].

3. Beschreibung der Ausprägungsmöglichkeiten der organisatorischen Hauptkomponenten von KSE und deren Bewertung (Rechts- und Organisationsform, Lizenzmodelle, organisatorischer Teil der Entwicklungsplattformen [Kapitel 4].

4. Aufzeigen der Bausteine zur Beurteilung der Wirtschaftlichkeit von konsortialer Softwareentwicklung im Vergleich zu konventioneller Softwareentwicklung inklusive der qualitativen und quantitativen Bewertung des Nutzens sowie einer Betrachtung der möglichen Risiken [Kapitel 5].

Anhand dieser Darstellung ist aufzuzeigen, ob die Entwicklung von Software mit diesem Modell

- in kürzerer Zeit
- mit weniger finanziellen Mitteln
- mit Sicherstellung der notwendigen Qualität

möglich ist.

5. Wenn die eingangs aufgeworfene Fragestellung zur Möglichkeit OSS zu entwickeln positiv beantwortet werden kann, liegt nach vollständiger Untersuchung der organisatorischen, technischen und wirtschaftlichen Herausforderungen mit dieser Studie ein entscheidungsfähiges Konzept für die Umsetzung eines Pilotprojekts mit Bildung eines Konsortiums einschließlich Umsetzungsroadmap vor [Kapitel 6].

Die Zielgruppen für die Ergebnisse der Studie sind:

- zunächst die beteiligten Mitarbeiter aus den Unternehmen selbst, die mit dieser Studie stichhaltige Argumente für oder gegen den Einstieg in die konsortiale Softwareentwicklung erhalten
- die Entscheider der beteiligten Unternehmen, denen die Studie eine Entscheidungsbasis zur Genehmigung der nächsten Schritte liefert
- sowie weitere interessierte Netzbetreiber
- Hochschulen und Forschungseinrichtungen
- Systemanbieter
- und potenzielle Förderer eines solchen Konsortiums.

Kapitel 2

Bestandsaufnahme, Problembeschreibung und alternatives Vorgehen

INHALTSVERZEICHNIS

2 Bestandsaufnahme, Problembeschreibung und alternatives Vorgehen

2.1 Umfang der Untersuchung

Dieser Teil der Studie beschäftigt sich mit der Aufnahme von Informationen zu den Informationssystemen im technischen Bereich der Netzbetreiber und der Identifikation des aktuellen und zukünftigen Softwarebedarfs. Der Untersuchungsraum ist mit dem Fokus auf das Betreiben von Netzen festgelegt worden, da in diesem Bereich die Notwendigkeit und die Möglichkeit zur konsortialen Entwicklung erkannt wurde.

Abb. 2-1: Untersuchungsraum IT-Anwendungen im Bereich Technik

Im Rahmen von mit Fragebogen unterstützten Interviews wurden zunächst die IT-Systeme und - soweit möglich - deren Schnittstellen zu anderen IT-Systemen evaluiert. Es wurden die folgenden Systeme betrachtet:

- Netzleitsystem (NLS)
- Geografisches Informationssystem (GIS)
- Betriebsmitteldatenhaltungssystem (BDH)
- Netzberechnungssysteme (für Planungszwecke - spartenbezogen)
- Workforcemanagementsysteme (WFM)
- Systeme zur Störungsbearbeitung und -dokumentation
- Systeme zur Haltung von Kunden- und Verbrauchsdaten, soweit es für die Aufgaben der Netzführung von Bedeutung ist (z. B. keine Abrechnungssysteme).

Die weitere Untersuchung, insbesondere in Bezug auf die Bedarfsermittlung, beschränkt sich danach im Wesentlichen auf das Netzleitsystem. Die beiden anderen Systeme mit strategischer Bedeutung (GIS und BDH) werden hinsichtlich der Integration in die leittechnischen Funktionen mitbetrachtet.

Zur Betriebsmitteldatenhaltung ist anzumerken, dass Betriebsmitteldaten (technische Sachdaten wie auch zugehörige Stamm-, Bewegungs- und Prozessdaten über den gesamten Lebenszyklus des Betriebsmittels) häufig nicht in einem einzigen System gehalten werden. Selbst beim Einsatz sogenannter Betriebsmittelinformationssysteme (BIS) sind unterschiedliche Ausprägungen anzutreffen. Teilweise werden definierte (Teil-)Datenmengen nach wie vor auch im NLS, im GIS oder weiteren Drittsystemen gepflegt. Für die Gesamtheit aller verfügbaren Daten zu Netzbetriebsmitteln ist also von einer verteilten Datenhaltung auszugehen. Bei den weiteren Ausführungen wird deshalb der Begriff Betriebsmitteldatenhaltung (abgekürzt: BDH) systemisch verwendet, um hiermit die Klammer über die Systeme zur Betriebsmitteldatenhaltung zu legen.

Für eine erste Iteration wurden die in den Unternehmen ermittelten Softwarebedarfsthemen verschiedenen Clustern zugeordnet. Diese wurden von den beteiligten Netzführungen nochmals auf ihren Stellenwert in der näheren Zukunft untersucht um herauszuarbeiten, in welchen Bereichen ein hohes Potenzial für gemeinsame Aktivitäten besteht. Die Priorisierung dieser Themencluster schließt ab mit der Auswahl eines Themas mit dem - zum Zeitpunkt der Erhebung - höchsten Nutzwert für die beteiligten Unternehmen.

Im Kontext dieser Studie wurde auch ermittelt, welche Schwachstellen es im Einsatz der betrachteten IT-Werkzeuge gibt und detailliert beleuchtet, ob und in welchem Umfang die vorhandenen Herstellerbindungen sich nachteilig auf einen wirtschaftlichen und betreibergerechten Softwareeinsatz auswirken.

Mit Blick auf den bestehenden Softwarebedarf wurde im Anschluss untersucht, unter welchen Rahmenbedingungen die konsortiale Softwareentwicklung eine grundsätzliche Alternative zur konventionellen Softwarebeschaffung darstellen kann.

2.2 IT-Systeme und Schnittstellen (IT-Systemlandschaft)

Im Ergebnis zeigt sich in den IT-Systemlandschaften hinsichtlich der eingesetzten Systeme eine große Heterogenität bei den Anbietern. Lediglich die Netzleitsysteme sind von einem Hersteller, wobei jedoch in jedem Unternehmen ein unterschiedliches Release des Systems in einer unternehmensspezifischen Ausprägung verwendet wird. Ähnliches gilt für das SAP IS-U.

Bei den Anbietern der anderen Systeme handelt es sich überwiegend um im Markt bekannte Unternehmen, die auch andere nennenswerte Referenzen zu den von ihnen angebotenen Produkten vorweisen können.

2.2.1 Systemeigner, Datenpflege, Administration

Aufgrund der gewachsenen Konzernstrukturen in den Unternehmen haben sich auch unterschiedliche Konstellationen für das Systemeigentum, die Datenpflege und die Administration der Systeme gebildet.

So kommen neben der Konzernmutter sowohl die Netzgesellschaft als auch andere Tochterunternehmen als Systemeigner in Betracht und die Administration kann je nach System direkt der technischen Fachabteilung, der IT-Abteilung, einer IT-Tochter oder sogar einem externen Dienstleister zugeordnet sein. Ein ähnliches Bild ergibt sich für die Datenpflege, wobei diese nicht selten über mehrere Abteilungen und Unternehmen bzw. Dienstleister verteilt ist. Aus Unternehmenssicht ist die vorhandene Aufteilung und Arbeitsteilung durchaus nachvollziehbar und sinnvoll.

2.2.2 Datennutzung und Datenqualität

Der überwiegende Teil der Daten, die für den technischen Betrieb der Netze erforderlich sind, werden in den drei großen Systemen (NLS, GIS, BDH) gepflegt, wobei die Datenhaltung nicht einheitlich geregelt ist. So werden z. B. bestimmte technische Betriebsmitteldaten von dem einen Unternehmen im GIS und von einem anderen wiederum im BDH gepflegt. Generelles Ziel ist in allen Unternehmen eine möglichst redundanzfreie Erfassung und Pflege der Daten um zu verhindern, dass die Datenbestände auseinanderlaufen.

Die Qualität der Daten in den führenden Systemen hat in den vergangenen Jahren stetig zugenommen. So haben die meisten Unternehmen die Ersterfassung und/oder Digitalisierung der Bestandsdaten im GIS in der vom Unternehmen vorgegebenen Genauigkeit bereits abgeschlossen und - soweit wirtschaftlich möglich und sinnvoll - auch die Bereinigung und Vervollständigung von relevanten Sach- und Fachdaten (im GIS und BDH) vorangetrieben. In den Netzleitsystemen ist inzwischen die direkte Durchführung von Schaltungen (Fernschaltung) und - in den Bereichen, in denen dies noch nicht möglich ist - die zeitnahe Nachführung der vor Ort vollzogenen Schaltungen zum Standard geworden, wodurch zu jeder Zeit der Blick auf ein aktuelles Netzzustandsbild gewährleistet ist.

Zur besseren Unterstützung von häufig wiederkehrenden betrieblichen Prozessen (Planung, Bau, Instandhaltung und Entstörung) wurden in den unterschiedlichsten Ausprägungen zyklische Datenübergaben zwischen den Systemen (meistens unidirektional) an proprietären Schnittstellen oder sogenannte Absprünge in eines der anderen Systeme realisiert. Eine weitergehende Systemintegration, wie z. B die Bereitstellung von tagesaktuellen Bestandsplänen im NLS, Schemaplänen im GIS und BDH oder der Zugriff auf Messwertarchive, ist nur vereinzelt vorzufinden und erfordert einen nicht unerheblichen zusätzlichen Aufwand (z. B. für regelmäßige Topologieabgleiche).

2.2.3 Schnittstellen zu anderen Systemen

Eine ganze Reihe von Prozessen zum Betreiben von Netzen benötigt Informationen aus mehreren Systemen, insbesondere aus den drei genannten strategischen Systemen. Der Ruf nach einer stärkeren Systemkopplung oder Systemintegration in diesem Bereich zur Verbesserung der Informationsbereitstellung, der Reduzierung von Prozessdurchlaufzeiten, der Eliminierung von Medienbrüchen und den damit erzielbaren Kosteneinsparungen ist nicht neu. Zwar sind die technologischen Voraussetzungen zur Kopplung von Systemen heutzutage vielfach gegeben, die Problematik liegt jedoch darin, dass die einzelnen Prozesse unterschiedliche Anforderungen an eine Schnittstelle haben und auf beiden Seiten der Schnittstelle proprietäre Systeme mit zum Teil völlig andersartigen Datenformaten und -strukturen verbunden werden müssen. Schon die Erst-Realisierung einer Schnittstelle scheitert deshalb häufig an den Kosten.

Eine einmal realisierte Schnittstelle kann natürlich auch über Jahre hinweg stabil laufen. Neue Prozessanforderungen sind dann zumeist der Auslöser, auch eine solche Schnittstelle zu erneuern.

Steht jedoch für eines der gekoppelten Systeme ein Upgrade oder ein Releasewechsel an, sind gleichzeitig auch die Schnittstellen zu den anderen Systemen zu ertüchtigen, wodurch der ursprüngliche Realisierungsaufwand erneut entstehen kann. Leider haben nur wenige Softwareanbieter hierfür ihre Systeme mit standardisierten Schnittstellen ausgestattet.

Der beim Releasewechsel zu erwartende Aufwand für die Anpassung (zum Teil mehrerer Schnittstellen) ist enorm hoch und reicht im Einzelfall an die Größenordnung des Systemwertes heran, sodass aus wirtschaftlichen Gesichtspunkten statt eines Releasewechsels zum Teil auch die Ablösung des Altsystems erwogen werden muss.

In einigen Unternehmen sind ungeachtet des großen Aufwandes diverse Schnittstellen zwischen den betrachteten und auch weiteren Systemen realisiert worden. Bei diesen Schnittstellen handelt es sich in der Regel um direkte, uni- oder bidirektionale Kopplungen zwischen zwei Systemen. Im Mittel verfügt jedes der betrachteten Systeme über eine bis drei Schnittstellen, wobei die Anzahl der Schnittstellen im Einzelfall auch zweistellig werden kann.

2.2.4 Ermittlung des gemeinsamen Softwarebedarfs

Die in den Interviews und den Fragebögen ca. 70 genannten Bedarfsthemen ergaben nach der ersten Iteration ein uneinheitliches Bild über den aktuellen und zukünftigen Softwarebedarf der Netzbetreiber.

Themen, die in einem Unternehmen als akuter Bedarf gekennzeichnet wurden, sind in anderen bereits (zumindest teilweise) projektspezifisch realisiert wurden. Darüber hinaus haben in einigen Netzen derzeit nur Teilaspekte eines Themas eine höhere Relevanz.

Nach der Durchführung von zwei Iterationen konnte eine grundlegende Abschätzung vorgenommen werden.

Erwartungsgemäß ist der Wunsch nach einer Verbesserung des Datenaustausches zwischen den Systemen (z. B. durch eine geeignete Middleware) bzw. der Datenbereitstellung für andere Systeme mehrfach genannt worden. Dies ist eng verknüpft mit der Realisierung von Schnittstellen und der Forderung nach einem zentralen Datenpool zur Auswertung und Weiterverwendung von Daten in anderen Systemen oder neuen Services zur Unterstützung einzelner Geschäftsprozesse.

Darüber hinaus gibt es einen gemeinsamen Bedarf an gleichartigen Berichten an externe Stellen (Verbände, BNetzA, ÜNB etc.) zu Ereignissen und Aktivitäten in den Netzen. Weiterhin wird erwartet, dass sich aus der Entwicklung der Netze zu Smart Grids gleichartige, neue funktionale IT-Anforderungen ergeben.

Ein verstärkter Bedarf ergibt sich aktuell und zukünftig auch im Umfeld der Themen Netzberechnung, Datenaustauschplattform, Last- und Einspeisemanagement. Aus den genannten Themen hat sich das Last- und Einspeisemanagement als derzeit wichtigste Anforderung herauskristallisiert. Um ein gemeinsames Verständnis über die funktionalen Inhalte zu diesem Themenblock zu entwickeln, wurde parallel zur Studienerstellung ein erster Workshop zur Ermittlung der Anforderungen durchgeführt.

Ein Ergebnis des Workshops ist, dass sich das Last- und Einspeisemanagement aus der Sicht der Beteiligten in folgende Module gliedert:

- Modul 1 Netzzustandsbild
- Modul 2 Prognose
- Modul 3 Berichtswesen
- Modul 4 State Estimation / Zustandssimulation

- Modul 5 Lastflussrechnung / Lastflusssimulation
- Modul 6 Ausfallvariantenrechnung / Ausfallvariantensimulation
- Modul 7 Schaltszenarien
- Modul 8 Schaltantragsbearbeitung.

Bei allen Modulen ist erkennbar, dass sie einen hohen Bedarf an Daten aus den anderen Systemen benötigen.

Die Module bauen zum Teil aufeinander auf. Die Entwicklung kann aber auch parallel oder bei Bedarf auch in einer anderen Reihenfolge stattfinden. Inwieweit eines oder mehrere der aufgeführten Module für eine Pilotanwendung geeignet sind, ist in einem Umsetzungsprojekt zu entscheiden (siehe Kapitel 6).

Letztlich lassen sich die genannten Anforderungen aber wieder reduzieren auf die Notwendigkeit, auch große Datenmengen aus internen und externen Systemen in geeigneter Form bereitzustellen, zu verschneiden und im Rahmen funktionaler Applikationen zu verarbeiten.

2.3 Schwachstellen im Softwareeinsatz

2.3.1 Allgemeines

Bei der Bestandsaufnahme der vorhandenen IT-Systeme und des aktuellen Softwarebedarfs sind eine Reihe von Schwachstellen bei der Beschaffung und dem Einsatz dieser Systeme und ihrer Softwarekomponenten zum Vorschein gekommen. Diese Schwachstellen treten mehrheitlich auf und lassen sich im Wesentlichen wie folgt zusammenfassen:

- Qualitätsmängel bei ausgelieferter Software (Fehlerhäufigkeit)
- fehlende bzw. mangelhafte Dokumentation
- hohe Kosten bei Erweiterungen
- hohe Kosten für Wartung
- hohe Kosten für zusätzliche Schnittstellen
- Gefahr der redundanten Datenhaltung
- Lieferzeiten zu lang oder nicht verlässlich
- mangelnde Flexibilität bei neuen funktionalen Anforderungen
- unzureichende Softwareergonomie.

Im Projektteam wurde die Einschätzung geteilt, dass es sich hierbei im Grundsatz um herstellerunabhängige Phänomene handelt.

Die meisten der genannten Schwachstellen sind für die Unternehmen nicht ohne Weiteres zu beseitigen, da dies mit einem Wechsel des Produktes bzw. des Herstellers verbunden wäre. Diese gravierende Abhängigkeit wird als "Vendor-lock-in-Effekt" bezeichnet und soll im Weiteren konkreter betrachtet werden.

2.3.2 Vendor-lock-in-Effekt

2.3.2.1 Definition

Strebt ein Kunde den Wechsel zu einem anderen Produkt an, entstehen dabei für ihn Wechselkosten. Für den Kunden ist ein Wechsel nur sinnvoll, wenn der aus dem Wechsel entstandene Nutzen größer bzw. gleich den Kosten des Wechsels ist.

In den Wirtschaftswissenschaften werden als "Lock-in-Effekt" [2-1] (von engl. to lock in: einschließen, einsperren) jene Kosten bezeichnet, die eine Änderung der aktuellen Situation aufgrund hoher Wechselkosten unwirtschaftlich machen. Die Höhe der Wechselkosten bestimmt das Ausmaß des Lock-in-Effektes. Anbieter können mit diesem Effekt Kunden an sich binden, was auch gezielt instrumentalisiert wird.

Die Kunden werden durch finanzielle Investitionen in bestimmte Technologien (Betriebssystem bzw. Laufzeitumgebung) oder durch persönliche Beziehungen an einen Anbieter bzw. an eine Anbietergruppe gebunden. Dies bewirkt eine oft als "Vendor-lock-in" bezeichnete Herstellerabhängigkeit.

2.3.2.2 Auslöser

Aus technischen Gründen galt in der Vergangenheit, dass unter anderem auch die Netzleitsysteme durch eine komplexe, monolithische, kundenspezifische Struktur gekennzeichnet waren. Dazu gab es keine Alternative; die Anwender waren überzeugt von ihrem damaligen Vorgehen. Protokolle, Dokumente und Funktionen sind deshalb bis heute proprietär und es existiert ein entsprechender Lock-in-Effekt. Diese Situation muss derzeit von den Anwendern aufgrund fehlender Alternativen toleriert werden.

Auch heute noch führen die kundenspezifischen Präferenzen in der funktionalen und technischen Ausprägung von Systemen dazu, dass produkt- oder anbieterspezifische Investitionen realisiert werden. Je mehr spezifische Investitionen sich ein Kunde leistet, desto höher steigen die Wechselkosten. Die Individualisierung von Produkten hinsichtlich der Wünsche des Kunden vertieft somit die Beziehung zwischen den Geschäftspartnern und der Wechsel zu einem Konkurrenzprodukt wird immer unwahrscheinlicher.

Weiterhin bauen Kunde und Lieferant ein produkt- oder technologiespezifisches Wissen auf. Im Falle eines Wechsels wäre dieser Lernprozess auf der Seite des Kunden zu wiederholen, wodurch die Wechselbarrieren entsprechend hoch liegen. Im Bereich der operativen Netzführung ist bei diesem Aspekt natürlich zu beachten, dass die Beherrschung des eingesetzten Werkzeuges eine zentrale Voraussetzung für einen sicheren Netzbetrieb darstellt.

Ausgehend vom spezifischen Wissen entsteht ein Gewöhnungseffekt des Kunden gegenüber dem Produkt oder dem Anbieter. Die fehlende Bereitschaft, hier Änderungen in Kauf nehmen zu wollen, führt ebenfalls zum Lock-in.

Meistens wird diese Art der Bindung als fremdbestimmt und negativ empfunden. Dieses Gefühl tritt spätestens dann auf, sobald der Kunde etwas Neues oder Anderes möchte (z. B. weil er mit dem Produkt bzw. mit dem Support für das Produkt unzufrieden ist) und sich in diesem Zusammenhang der Wechselkosten bewusst wird. Bei Softwareprodukten ist das auslösende Ereignis häufig ein erzwungenes Upgrade, z. B. durch die Abkündigung des Supports für das aktuelle Release, durch welches der Kunde sich der "Vendor-lock-in-Situation" bewusst wird.

2.3.2.3 Strategie der Anbieter

Anbieterunternehmen verfolgen - mehr oder weniger stark ausgeprägt - gezielt eine Lock-in-Strategie, um Kunden an sich zu binden. Ziel des Anbieters ist es dabei, dass der Kunde den Nutzen eines Produktes höherwertiger als die Lock-in-Kosten wahrnimmt. Deshalb versucht der Anbieter z. B. durch Personalisierungsmöglichkeiten (Customizing) und Rabattangebote den Kundennutzen in dem Maße zu steigern, dass er sich dennoch "freiwillig" in die Lock-in-Situation begibt. Sie erschweren ihren Kunden gezielt und gewollt den Wechsel zu einem anderen Produkt und verschaffen sich so gegenüber anderen Anbietern einen Wettbewerbsvorteil.

In der IT-Branche ist die "Vendor-lock-in-Strategie" sowohl auf der Hardware- als auch auf der Softwareseite weit verbreitet. Mangelnde Kompatibilität und Interoperabilität auf den Ebenen des Anwendungsprogramms, des Dateiformates oder des Betriebssystems spielen in diesem Zusammenhang eine große Rolle.

Die Lock-in-Strategie birgt für Anbieter von langlebigen Produkten aber auch Risiken. Wird beim Customizing des Produktes so stark auf die Kundenwünsche nach einer maßgeschneiderten Lösung eingegangen, dass das Produkt bis zur Auslieferung an den Kunden quasi zu einem Unikat geworden ist, steigt zwangsläufig der interne Aufwand für den Support, die Neu- und Weiterentwicklung und die Dokumentation des Produktes. Dies führt auf der Anbieterseite in der Regel zu einem Ressourcenproblem und unter Umständen zu einem Know-how-Problem. In Folge dessen führt dies auf der Kundenseite zu Unzufriedenheit und ggf. auch zu unerwarteten Kostensteigerungen.

2.3.2.4 Strategie der Kunden

Sobald sich ein Kunde der "Lock-in-Situation" bewusst ist, wird er dies bei seiner weiteren Entscheidungsfindung berücksichtigen. Zum einen kann er diese Situation in Kauf nehmen, wenn mangels Alternativen im Markt ein hoher Kundennutzen für den Einsatz des Produktes spricht. Es ist in diesem Fall jedoch schwieriger, zukünftige Kostenüberschreitungen zu verhindern. Im Gegenteil wird manchmal sogar das "Lock-in" als Erklärung für angeblich unvermeidbare Kostenüberschreitungen herangezogen.

Eine andere mögliche Strategie des Kunden könnte sein, weitere Investitionen in das vorhandene Produkt abzulehnen und mittelfristig Alternativen zu suchen. Auf Software bezogen bedeutet dies, dass der Kunde nicht zwangsläufig einen Anbieterwechsel anstreben muss, sofern er nach wie vor mit den Kernfunktionen des Produktes zufrieden ist. Dies führt aber sukzessive dazu, dass neue und auch bereits vorhandene Funktionen in anderen Systemen realisiert werden.

2.3.2.5 Situation in den Unternehmen

Das "Lock-in" gemäß der vorstehenden Definition wurde von den Netzführern mehrheitlich bestätigt. In den Unternehmen besteht Unzufriedenheit mit den Supportleistungen der jeweiligen Anbieter. Aufgrund der fehlenden Verfügbarkeit des richtigen Ansprechpartners zur Lösung eines Problems oder Fehlers tritt häufig eine verzögerte Bearbeitung auf.

Bemängelt wird auch das unzureichende Testen von Patches vor dem Einspielen beim Kunden. Das Wiederauftreten alter Fehler und neue Fehlfunktionen in anderen Bereichen, insbesondere nach Datenmodellanpassungen, sind in diesem Zusammenhang keine Seltenheit. Als Reaktion auf diese Erfahrungen lassen einige Unternehmen neue Patches z. B. mindestens vierzehn Tage auf einem eigenen Testsystem laufen, um etwaige Fehler vor der eigentlichen Verteilung zu entdecken und zu beheben.

Auch die Preisgestaltung insgesamt, insbesondere aber für neue Funktionen und Module, wird von den Endanwendern mehrheitlich kritisiert. Die Kosten sind aus der Sicht der Unternehmen nicht angemessen. Dies führt dazu, dass die Programmierung neuer, aber derzeit nicht dringend erforderlichen Applikationen aufgrund fehlender Budgets zurückgehalten wird. Vor diesem Hintergrund wird vermehrt geprüft, ob sich einzelne Funktionen auch außerhalb des Systems realisieren lassen. An dieser Stelle setzt auch die Studie an, um zu prüfen, ob die konsortiale Softwareentwicklung auf der Basis von Open Source einen geeigneten Lösungsweg zur möglichst weitgehenden Überwindung der aufgezeigten Schwachstellen und damit zur wirtschaftlichen Umsetzung der Anforderungen der Netzbetreiber bieten kann.

2.4 Bewertung und abgeleitetes alternatives Vorgehen

Die im vorherigen Abschnitt aufgeführten Schwachstellen im Softwareeinsatz können in der Einzelbetrachtung sicherlich als herstellerunabhängig auftretende Phänomene bezeichnet werden; sie beruhen jedoch im Kern auf dem "Lock-in-Effekt".

Eine Sonderstellung nimmt der in den IT-Landschaften heute noch häufig vorzufindende Ansatz der direkten Kopplung von Systemen über eine proprietäre Schnittstelle ein, da sich hier gleich mehrere "Lock-in-Effekte" vermischen. Nachteilig und kostentreibend wirken sich die zwangsläufig bei jedem Systemupgrade bzw. Releasewechsel notwendig werdenden, erneuten Anpassungen der vom System ausgehenden Schnittstellen aus. Nicht selten gehen damit auch zusätzliche Änderungen auf der Seite des gekoppelten Altsystems einher oder führen sogar zu dessen Ablösung, obwohl es die Anforderungen aus funktionaler Sicht noch voll erfüllt. Diese Auswirkung ist mit der Vernichtung von Unternehmenswerten gleichzusetzen.

Der mit der Anpassung von direkten, proprietären Schnittstellen verbundene Aufwand ist bei einer zukünftig in noch stärkerem Maße erforderlichen Systemintegration nicht mehr tragbar und erfordert einen Wandel in der Systemarchitektur. Hier verspricht mittel- bis langfristig der Einsatz einer sogenannten Middleware bzw. eines Enterprise Service Bus in einer serviceorientierten Architektur durch die Standardisierung von Adaptern bzw. Konnektoren eine erhöhte Flexibilität bei gleichzeitiger Reduzierung der Kosten für die Systemintegration (siehe Kapitel 3).

Ansonsten können die Beseitigung oder zumindest Abmilderung der Schwachstellen aus Anwendersicht bisher nur durch weitere Investitionen in das Softwareprodukt, einen ebenso kostspieligen und risikobehafteten Anbieterwechsel oder die Realisierung neuer Funktionen außerhalb der vorhandenen Systeme durch einen anderen Hersteller erreicht werden. Der "Lock-in-Effekt" lässt sich allerdings damit nicht überwinden.

Dies ist schlussendlich nur möglich, wenn das Unternehmen bei der Entwicklung von Software selbst initiativ wird.

Die Eigenentwicklung von Software bzw. neuer Softwareapplikationen wurde zwar von einzelnen Netzbetreibern schon erfolgreich betrieben, birgt aber aufgrund begrenzter Ressourcen die Gefahr, dass mittel- bis langfristig ähnliche Schwachstellen wie bei der Beschaffung über einen Hersteller entstehen.

Eine gemeinschaftliche Eigenentwicklung mehrerer Netzbetreiber könnte diese Risiken minimieren und zu einer Kostenreduzierung führen. Angesichts der großen Heterogenität der bei den Netzbetreibern eingesetzten Systeme ist das Potenzial hierfür relativ gering. Dieser Weg setzt zudem eine langfristig angelegte Kooperation der beteiligten Unternehmen voraus und erscheint nur mit dem erklärten Willen zur mittelfristigen Standardisierung der von der Eigenentwicklung unterstützten Prozesse wirtschaftlich darstellbar zu sein. Selbst wenn eigene Ressourcen für Support und Weiterentwicklung aufgebaut werden, kann dies zu Engpässen führen, da es hierfür nur einen (eigenen) Anbieter gibt.

Erst der neue Ansatz zur konsortialen Softwareentwicklung auf der Basis von Open Source bietet für eine Gruppe von Endanwendern die Chance als Lizenzgeber für die quelloffene Software, den "Vendor-lock-in-Effekt" vollständig zu überwinden, die benötigte Weiterentwicklung der Software zeitgerecht voranzutreiben und deren langfristigen Support zu sichern. Im Weiteren wird genau dieser Ansatz untersucht, um herauszufinden, ob er auf Netzbetreiber übertragbar ist.

Diese Art der Softwareentwicklung kann besonders dann eine erfolgreiche Alternative sein, wenn sich die Mitglieder des Konsortiums anfänglich auf die Bereiche mit gemeinsamem Bedarf und gleichen Prozessanforderungen konzentrieren, zu denen es noch keine Systeme oder Systemerweiterungen gibt oder in denen bereits eine weitgehende Homogenität der Systeme erreicht ist. Im vorliegenden Fall sind beide Bedingungen für das Netzleitsystem erfüllt.

Die Auftraggeber der Studie setzen alle ein System desselben Herstellers ein und haben in einem ersten Workshop aufgezeigt, dass allein im direkten Umfeld des Netzleitsystems ein übereinstimmender, dringender Bedarf an neuen Funktionen bzw. Modulen besteht, der so hoch ist, dass damit ein mehrjähriger Entwicklungsprozess initiiert werden kann. Besonders dort, wo regulatorische Anforderungen zu erfüllen sind, die per se auf einer gleichen Prozessauffassung beruhen oder gänzlich neu sind, wird die Realisierung von standardisierten Lösungen erleichtert.

Dies hat im Verlauf der Studie den Ausschlag gegeben, sich in den weiteren Betrachtungen auf das Umfeld der Netzleitsysteme zu fokussieren.

In den folgenden Abschnitten werden die Grundlagen für ein Geschäftsmodell zur konsortialen Entwicklung von Open-Source-Software als Alternative zur konventionellen Softwarebeschaffung beschrieben. Ziel ist es an dieser Stelle, einen allgemeinen Verständnisrahmen zu schaffen, der die grundlegenden Mechanismen und Merkmale dieser Alternative beschreibt. Eine weitergehende Präzisierung des Geschäftsmodells erfolgt in Kapitel 4.

2.4.1 Geschäftsmodelle für die Softwareentwicklung

Zunächst werden einige Geschäftsmodelle für die Softwareentwicklung [2-2] gegenübergestellt. Für diesen Vergleich der Geschäftsmodelle wurde angenommen, dass die Kosten für die Implementierung und Integration in allen Modellen gleich sind.

2.4.1.1 Closed-Source-Business-Model

Unter dem Closed-Source-Business-Model ist das herkömmliche Geschäftsmodell von Anbietern proprietärer Software zu verstehen, die ihren Kunden (den Endanwendern) ihr Produkt neben den oben genannten Kosten für eine Lizenzgebühr und eine Wartungs- und Servicegebühr die Software in kompilierter Form zur Nutzung überlassen. Die Kalkulation der Lizenzgebühr obliegt natürlich dem jeweiligen Anbieter und kann in seinen Kostenbestandteilen je nach Softwareprodukt variieren. Für den Vergleich der Geschäftsmodelle wurde eine Kostenaufteilung angenommen, wie sie üblicherweise an-zutreffen ist.

Die Lizenzgebühren setzen sich demnach aus folgenden Kostenblöcken zusammen (Richtwerte):

- 10 % Profit
- 16 % Forschung und Entwicklung (Research & Development, R & D)
- 64 % Werbung und Vertrieb (Marketing & Sales)
- 6 % allgemeine Verwaltungskosten (General & Administrative, G & A)
- 4 % Herstellungskosten (Cost of Goods Sold, COGS).

Für die jährlich anfallenden Wartungs- und Servicegebühren wird ein marktgängiger Wert von ca. 20 % der Lizenzkosten angesetzt.

2.4.1.2 Open-Source-Business-Model

Durch die Verwendung von Open-Source-Software ergeben sich Veränderungen in den einzelnen Kostenblöcken. Hierzu werden drei mögliche Geschäftsmodelle betrachtet:

Open-Source-Consulting-Model

Dieses Geschäftsmodell setzt voraus, dass bereits ein für den Einsatzzweck geeignetes Softwareprodukt unter Open-Source-Lizenz existiert. Ein Anbieter nutzt diese Open-Source-Software als Basis für die beim Endkunden zu implementierende Lösung. Die Kosten für die Entwicklung der Open-Source-Software wurden von der Entwicklergemeinschaft des jeweiligen Open-Source-Projektes aufgebracht. Daher fallen beim Endkunden keine Lizenzgebühren an. Die Kosten für Marketing & Sales, G & A, COGS und der Profit sind dabei vollständig in den Tagessätzen der Berater des Anbieters enthalten. Es handelt sich deshalb aus der Sicht des Anbieters um ein Consulting-Business-Model. Der Anbieter übernimmt gegenüber dem Endkunden die Lösungsverantwortung und den zugehörigen Support.

Der Vorteil für den Endkunden liegt in den geringeren Kosten durch den Wegfall der Lizenzkosten. Weiterer Nutzen kann entstehen durch die Nutzung der offenen Standards (höhere Qualität, schnellere Innovation, hohe Unabhängigkeit) und der Möglichkeit, an der Open-Source-Software kunden-individuelle Erweiterungen vorzunehmen.

Dies kann auch gleichzeitig ein Nachteil sein, da mit einer kundenindividuellen Implementierung der Aufwand für den Support steigt (siehe Abbildung 2-2) und dieser in der Regel nur vom ausgewählten Anbieter durchgeführt werden kann (siehe 2.3.2.2).

Die frei zur Verfügung stehende Open-Source-Software muss im Einzelfall vor dem Einsatz beim Kunden vom Anbieter auf Funktionsvollständigkeit und Supportverfügbarkeit geprüft werden. Gegebenenfalls sollte das nachfolgend beschriebene Commercial-Open-Source-Business-Model (s. u.) herangezogen werden.

Es besteht die Gefahr als Endanwender, durch

- den geringen eigenen Einfluss auf das Produkt der Entwickler-Community
- die vom Anbieter realisierten Modifizierungen
- die Übernahme des gesamten Projektes durch Dritte (Beispiel: MySQL-Übernahme durch Oracle)

in eine neue, wenn auch abgeschwächte Form des "Lock-in-Effektes" zu geraten.

Für einzelne Industriebereiche, wie im vorliegenden Fall speziell für Netzbetreiber, sind nutzbare Produkte in der Regel nicht existent, wodurch dieses Geschäftsmodell in der weiteren Betrachtung ausscheidet.

Commercial-Open-Source-Model

Das Commercial-Open-Source-Model ist im Grunde identisch zu dem Closed-Source-Business-Model. Dadurch, dass die Software von einer Entwicklungsgemeinschaft (überwiegend von Softwareanbietern) erstellt wird und sich der Vertriebsaufwand im Bereich für Open-Source-Software in der Regel von denen anderer professioneller Anbieter unterscheidet, lassen sich in einigen Kostenblöcken Einsparungen realisieren. Signifikante Kosteneinsparungen im Bereich der Softwareentwicklung stellen sich für die Anbieter erst ein, wenn die Entwicklungsarbeit von einer möglichst großen Entwickler-Community aus verschiedenen Unternehmen getragen wird. Grundsätzlich versprechen sich die Anbieter mit der Wahl dieses Geschäftsmodells eine Steigerung der Absatzmöglichkeiten und einen höheren Gewinn.

Für den Endnutzer der Software ergeben sich alle mit der Verwendung von Open-Source-Software verbundenen Vorteile und Kosteneinsparungen gemäß Abbildung 2-2, sofern diese vom Anbieter an den Kunden weitergegeben werden.

Aus der Sicht der Endanwender wird hier noch stärker als beim Open-Source-Consulting-Model deutlich, dass der reine Einsatz von Open-Source-Software noch keine Überwindung des "Lock-in-Effekts" bedeutet.

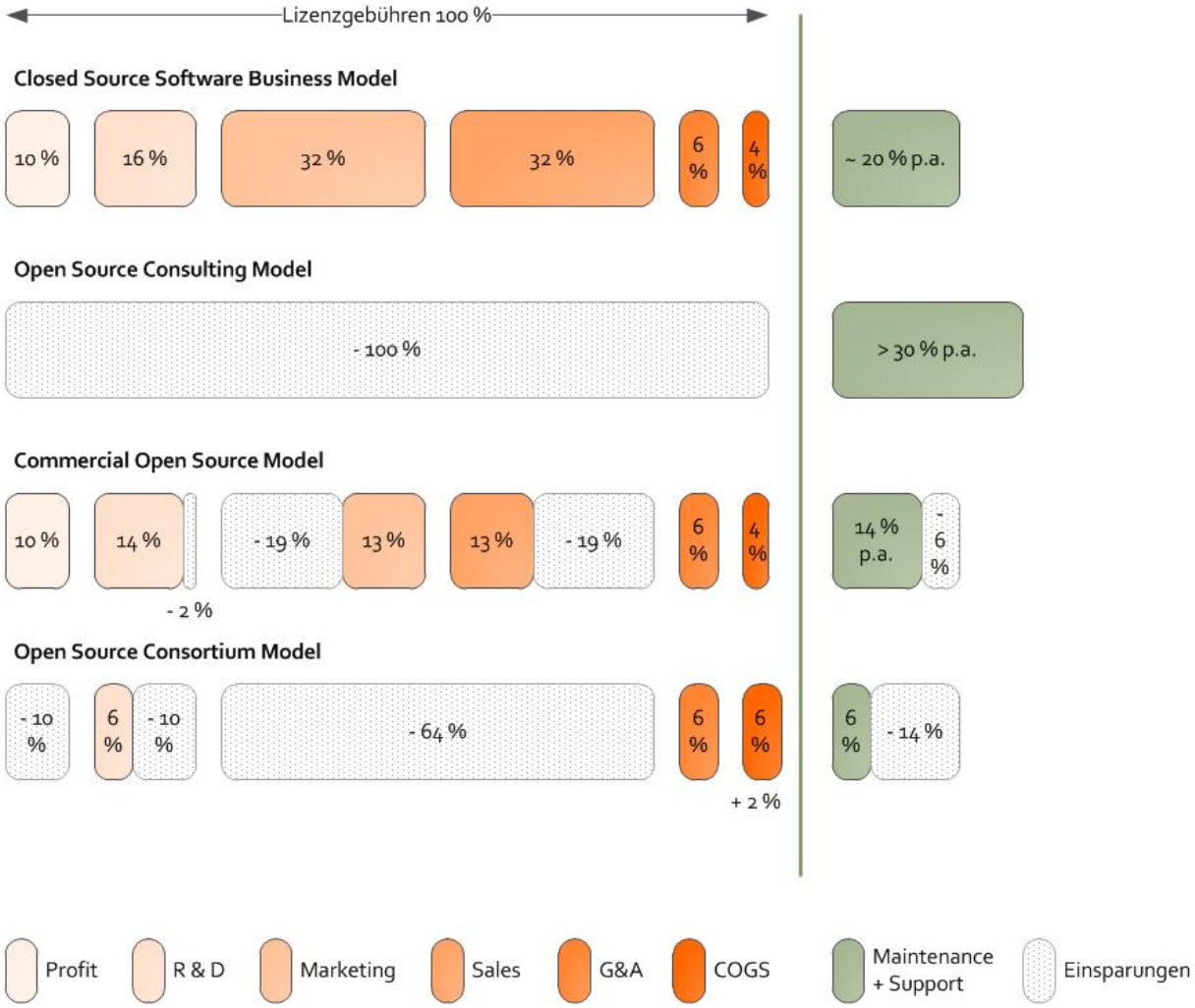


Abb. 2-2: Kostenvergleich OSS-Geschäftsmodelle

Open-Source-Consortium-Model

Im Open-Source-Consortium-Model findet sich eine Gruppe von Unternehmen - typischerweise aus einer Branche - zusammen, die auch gleichzeitig Endanwender von Software sind, um das zu entwickeln bzw. entwickeln zu lassen, was sie an Software benötigen und ggf. von der Softwareindustrie nicht bekommen.

Einzig mit diesem Geschäftsmodell wird die Herstellerabhängigkeit durchbrochen, da das Konsortium (überwiegend bestehend aus Endanwendern) nun selbst diese Rolle einnimmt (weitere Aspekte siehe Abschnitt 2.4.2).

Mit diesem Geschäftsmodell sind auch die größten Kosteneinsparungen zu realisieren, da die Software in erster Linie für die Mitglieder des Konsortiums entwickelt wird und ein weiterer direkter Vertrieb der Software nicht Bestandteil des Geschäftsmodells ist. Dem steht ein geringer Mehraufwand bei der Herstellung gegenüber, der im Wesentlichen durch die erforderlichen Abstimmungsprozesse im (in der Regel) verteilten Projektteam und unter den Konsortiumsmitgliedern bedingt ist.

Unabhängig vom Vergleich der Geschäftsmodelle stellen sich die größten Kostensenkungspotenziale jedoch ein, wenn es relativ schnell nach der Konsortiumsgründung gelingt, eine größere Anzahl von neuen Mitgliedern zu akquirieren, um die Entwicklungskosten auf viele Schultern zu verteilen (siehe Szenarien in Kapitel 5). Letztlich steht hinter diesem Potenzial der Gedanke, dass die gleiche Funktion nicht mehrfach von verschiedenen Anbietern parallel entwickelt werden muss und natürlich auch nicht mehrfach von den Anwendern zu bezahlen ist.

Darüber hinaus sinken die Kosten für die Wartung und den Service gegenüber den anderen Modellen nochmals deutlich, da alle Anwender von einer schnelleren Fehlerbehebung (Bugfixing) profitieren und die im Bereich der Wartung üblicherweise auch enthaltenen Profitanteile für das entwickelte Produkt nicht entstehen. Im Kapitel 5 zum Thema Wirtschaftlichkeit wird hierauf näher eingegangen.

2.4.2 Merkmale der konsortialen Softwareentwicklung auf Open-Source-Basis

Im Gegensatz zur konsortialen Beschaffung von Closed Source, in der das Konsortium lediglich als Einkaufsgemeinschaft von Endanwendern auftritt, wird bei der konsortialen Softwareentwicklung der Entwicklungsprozess der Software von einem Open-Source-Konsortium übernommen. Es besteht überwiegend aus Anwenderunternehmen, die die Software nicht zur Differenzierung im Wettbewerb, sondern in erster Linie zur wirtschaftlichen Optimierung einsetzen wollen.

Dieser Gedanke zielt dabei darauf ab, dass der Unternehmenszweck des Konsortiums hauptsächlich darin liegt, Dienste für die (Mitglieder-)Gemeinschaft zu erbringen. Dies ist entscheidend für den Aufbau des Konsortiums selbst und den Aufbau einer Nutzengemeinschaft. Diese Nutzengemeinschaft wird auch als Ökosystem bezeichnet, in dem neben den regulären Konsortiumsmitgliedern (Endanwendern) auch Software-Dienstleister für Entwicklung, Training und Support, akademische Mitglieder (Hochschulen und Institute), Sponsoren sowie andere Beitragende zusammenwirken. Die Anzahl der Mitglieder ist dabei ein Indikator für den Erfolg des Konsortiums.

Das Konsortium ist Eigentümer der auf Open-Source-Basis entwickelten Software, d. h. es hält die IP-Rechte (aus dem Englischen: Intellectual property - Rechte aus geistigem Eigentum), welche die Patente, Marken etc. sowie Urheberrechte, Lizenzen und Leistungsschutzrechte umfassen (weitere Informationen siehe Kapitel 4.2). Durch die Offenlegung des Codes verschwindet die Abhängigkeit der Anwender von Software-Herstellern und Dienstleistern und führt zu der Überwindung des "Vendor-lock-in". Das Konsortium sichert sich damit über den gesamten Lebenszyklus der Software den Einfluss auf alle für die Mitglieder wichtigen Aspekte.

Dies gilt zum Beispiel für den Einfluss auf die Entwicklungsgeschwindigkeit einer Software oder einzelner Softwaremodule, da in den meisten Fällen ein funktionierendes Konsortium für seine Mitglieder wesentlich mehr Entwicklungs- und Supportressourcen in kürzerer Zeit freisetzen kann als ein einzelner Software-Lieferant mit seinen begrenzten Entwicklungskapazitäten für seine proprietäre Softwarelösung. Der Zeitraum bis zur erstmaligen Verfügbarkeit der Software (time-to-market) kann somit deutlich verkürzt werden. Dabei gilt jedoch die Voraussetzung, dass die Gemeinschaft die Anforderungen an die Softwarefunktionalität zügig in einem effizienten Prozess definieren kann.

Die Konsortiumsmitglieder sind in der Position, maßgeblichen Einfluss auf die Weiterentwicklung der konsortialen Software ausüben zu können (driver seat), da sie jederzeit die Entwicklung funktionaler Erweiterungen beschließen und (unter anderem finanziell) vorantreiben können.

Bei Software mit langen Nutzungs- und Wartungszeiträumen ist der offene Sourcecode Garant für die Investitionssicherung, da die Weiterentwicklung sowie die Langzeitwartung und der Langzeitsupport auch nach vielen Jahren Einsatz noch von neuen Entwicklern und Dienstleister übernommen werden können.

2.4.3 Motivation für die konsortiale Zusammenarbeit

Die Zusammenarbeit von Unternehmen, auch wenn sie im Wettbewerb zueinander stehen, ist an sich kein neues Phänomen. Die gemeinschaftliche Erstellung von Standards für Datenformate, Prozesse und Ähnlichem ist seit langer Zeit weit verbreitet. Neu ist aber die gemeinschaftliche Erstellung solcher Standards mit Open Source durch eine Gruppe von Softwareanwendern im Rahmen eines Konsortiums [2-3].

Es gilt dabei, zwei grundsätzliche Herausforderungen zu lösen:

1. Die Einstiegshürde zur Beteiligung an gemeinschaftlicher Softwareentwicklung darf nicht zu hoch sein, d. h. es müssen organisatorische Rahmenbedingungen vorhanden sein bzw. geschaffen werden, welche eine möglichst einfache Teilnahme ermöglichen.
2. Es müssen auf technischer und organisatorischer Ebene Plattformen vorhanden sein, welche die Erweiterbarkeit und Modifizierbarkeit der erstellten Applikationen ermöglichen und dabei nur einen minimalen Wartungsaufwand benötigen.

Obgleich dieser erfolgskritischen Herausforderungen haben sich nach dem Whitepaper des COSAD-Projekts ("Collaborative Open Source Application Development") [2-3] der Open Source Business Foundation e.V. folgende Punkte als Motivation herauskristallisiert, ein Konsortium für die Entwicklung von Software zu bilden:

- die Erkenntnis, dass viele Softwareanwendungen keine wettbewerbsdifferenzierende Wirkung haben, aber mit hohen Kosten verbunden sind
- die Notwendigkeit, in einer komplexen IT-Welt die Abhängigkeit von Lieferanten und einzelnen Menschen zu verringern ("Vendor-lock-in-Effekt")
- die Notwendigkeit, Standardisierung und Flexibilisierung zu erhöhen. Durch Standardisierung lässt sich die Total Cost of Ownership reduzieren, durch Flexibilisierung können Unternehmen schneller auf Veränderungen des Marktes / Umfeldes reagieren.

Die Gewichtung dieser Punkte kann je nach Branche und Aufgabenstellung sehr unterschiedlich ausfallen. Unter Umständen ist allein ein Aspekt so bedeutend für das Geschäftsmodell und damit für das Überleben eines Unternehmens, dass sich alles andere diesem Aspekt unterordnen muss.

Bezogen auf die Anwendergruppe der an dieser Studie beteiligten Netzbetreiber sind die oben aufgeführten Punkte gegeben.

Aufgrund der vorzufindenden Rahmen- und Umfeldbedingungen gibt es hier gute Chancen für eine erfolgreiche Konsortiumsbildung:

- Der Kooperationsgedanke ist in der Branche etabliert (siehe 8KU).
- Es existieren gefestigte Netzwerke durch fachliche Zusammenarbeit in Arbeitskreisen und Usergroups.
- Alle Netzbetreiber stehen vor den gleichen drängenden Herausforderungen, die sich aus neuen Gesetzen, der Regulierung und der Energiewende ergebenden Anforderungen an die IT-Unterstützung umzusetzen, sodass von vornherein mit gleicher abgestimmter Prozessvorstellung gestartet werden kann.

Treten bereits in der Gründungsphase gewichtige Marktteilnehmer als Protagonisten auf, lassen sich Nutzengemeinschaften (Ökosysteme) etablieren. Aus einer solchen Strategie können sich für alle Teilnehmer beträchtliche Potenziale hinsichtlich Qualität, Nachhaltigkeit und Kosten ergeben.

Aus diesen Überlegungen heraus ergeben sich eine Reihe von Aspekten, die im Zusammenhang mit einer Konsortiumsgründung zu beantworten sind und in den folgenden Abschnitten näher beleuchtet werden sollen.

2.4.4 Eignung von Projekten für eine konsortiale Softwareentwicklung

Eine Übereinstimmung in einzelnen Themen rechtfertigt noch kein gemeinsames Softwareentwicklungsprojekt. Vielmehr sind vorab eine ganze Reihe von Fragestellungen zu prüfen und zu beantworten. Unter anderem sind dies:

- Gibt es unter den Netzbetreibern die gleichen funktionalen Anforderungen an das Produkt?
- Ist das Interesse an einer schnellen Umsetzung dieser Anforderungen annähernd gleich groß?
- Gibt es bereits entsprechende Produkte im Markt?
- Können die Produkte die geforderte Funktion vollständig erfüllen?
- Wenn nein, ist der Einsatz der verfügbaren Produkte trotzdem wirtschaftlicher als eine die Anforderungen abdeckende Eigenentwicklung?
- Wie hoch ist der erwartete Nutzen für die einzelnen Unternehmen?
- Kann das Produkt (oder Teile davon) auch zur Unterstützung weiterer Geschäftsprozesse genutzt werden?
- Gibt es einen Markt für die Software? Wie attraktiv ist das zu entwickelnde Produkt für andere Unternehmen (Nichtmitglieder des Konsortiums)?

Die erfolgreiche Produktentwicklung ist also von mehreren Voraussetzungen abhängig. In jedem Fall sollte für die Entwicklung des ersten Release eines ersten Produktes eine ausreichend große Gruppe von Unternehmen bereit sein, die Finanzierung zu übernehmen (zu den möglichen Formen des Beitrages siehe Kapitel 4, zu Finanzierungsmodellen siehe Kapitel 5).

Sofern aus den in der Bedarfsermittlung genannten Themen ein langfristig gemeinsamer Bedarf abzuleiten ist, könnte aus diesen Themen ein unter den Konsortiumsmitgliedern abgestimmtes Ranking für chronologisch aufeinander folgende bzw. aufeinander aufbauende Entwicklungsprojekte entstehen, mit dem eine mögliche Perspektive für das Betätigungsfeld des Konsortiums nach der Bewältigung eines ersten Projektes eröffnet wird.

2.5 Zusammenfassung

In diesem Kapitel werden aufbauend auf eine Bestandsaufnahme der IT-Systeme im technischen Umfeld der beteiligten Netzbetreiber die dabei ermittelten Schwachstellen aufgezeigt. Die konsortiale Softwareentwicklung auf Basis von Open-Source-Software wird anschließend mit ihren charakteristischen Merkmalen und als interessante Alternative zur heutigen Beschaffungspraxis von Software dargestellt.

Im Rahmen der Bestandsaufnahme wurden die vorhandenen Systeme und die dazugehörigen Eigentumsverhältnisse sowie die Zuständigkeiten für Datenpflege und Administration ermittelt und im Detail dokumentiert. Es ergab sich für alle genannten Aspekte ein relativ heterogenes Bild, lediglich die Nutzung eines Leitsystems eines Herstellers ist allen Beteiligten gemeinsam. Nicht zuletzt aus diesem Grund sind die weiteren Betrachtungen sehr stark durch eine Fokussierung auf die Netzleitsysteme geprägt.

Aus der Analyse von Funktionalitäten, die zurzeit nicht oder nur unvollständig durch die vorhandenen Systeme abgedeckt werden, ergab sich ein mehrheitlich vorrangiger Softwarebedarf auf dem Gebiet des Einspeise- und Lastmanagements. Dieser Funktionsbereich wurde in Module zerlegt, die in einer noch zu bestimmenden Reihenfolge realisiert werden könnten. Dieser Realisierung liegt die Vorstellung zugrunde, dass die Softwarebausteine nicht voll in das Leitsystem integriert werden müssen und sich damit die Möglichkeit für alternative Systemkonzepte bieten würde (siehe Kapitel 3).

Die Bestandsaufnahme ergab weiterhin, dass es eine Reihe von wirtschaftlichen, qualitativen und operativen Schwachstellen bei der Beschaffung, Implementierung, dem Einsatz der IT-Systeme und dem Datenaustausch zwischen den Systemen gibt.

Insbesondere die Anpassung von Schnittstellen ist bei den aktuellen Strukturen häufig mit hohem Aufwand verbunden. Dies kann sogar dazu führen, dass nach einem Releasewechsel in einem vorhandenen System ein hiermit gekoppeltes System ersetzt werden muss, obwohl letzteres die gestellten Anforderungen mittelfristig noch sicher erfüllen könnte. Dies ist mit einer Vernichtung von Unternehmenswerten gleichzusetzen.

Diese Schwachstellen lassen sich bei der heute praktizierten Vorgehensweise nicht grundsätzlich vermeiden, da sich durch die starke Herstellerbindung, besonders bei großen Netzleitsystemen Alternativen nur mit großem Aufwand erschließen lassen. Diese als Vendor-Lock-in-Effekt bezeichnete Abhängigkeit ist der wesentliche Auslöser, um über grundsätzliche Alternativen nachzudenken.

Vor diesem Hintergrund wurden verschiedene Geschäftsmodelle für die Softwareerstellung betrachtet. Hieraus resultiert eine Präferenz für das sogenannte Open-Source-Consortium-Model, das heißt die gemeinschaftliche Entwicklung von Open-Source-Software in einem Konsortium, da sich hier prinzipiell

- die größten Kosteneinsparungen realisieren lassen,
- nur in diesem Modell die Überwindung des Vendor-Lock-in-Effektes möglich wird und
- durch eine geänderte Softwarearchitektur eine Standardisierung von Schnittstellen erreichen lässt (siehe Kapitel 3).

Dies bedeutet, dass die Anwender Einfluss auf die Entwicklungsgeschwindigkeit und funktionale Ausrichtung einer Software bekommen, da in den meisten Fällen ein funktionierendes Konsortium für seine Mitglieder wesentlich mehr Entwicklungs- und Supportressourcen in kürzerer Zeit und zu günstigeren Konditionen freisetzen kann, als ein einzelner Lieferant für seine proprietäre Softwarelösung. Der Zeitraum bis zur erstmaligen Verfügbarkeit der Software (time-to-market) kann somit deutlich verkürzt werden.

Durch die Softwareentwicklung auf der Basis von Open Source ist außerdem zu erwarten, dass die Transparenz in den von der Software unterstützten Prozessen steigt (z. B. bei der Verwendung bestimmter Algorithmen) und somit die Vertrauensbasis zwischen Behörden (z. B. BNetzA) und Unternehmen deutlich verbessert werden kann.

Die Voraussetzungen für die Gründung eines Konsortiums aus Netzbetreibern sind günstig. Gleiche Interessen, praktizierte Kooperationen sowie steigende Anforderungen an die IT-Unterstützung durch Gesetze, Regulierung und Energiewende bilden eine gute Basis für eine erfolgreiche Geschäftstätigkeit. Mit der Ermittlung des Softwarebedarfs im Umfeld der drei strategischen Systeme NLS, GIS und BDH konnte aufgezeigt werden, dass für Netzbetreiber im Umfeld der Netzleitsysteme ein ausreichendes Potenzial für eine konsortiale Softwareentwicklung besteht.

Für ein erfolgreiches Konsortium muss allerdings vorausgesetzt werden, dass verschiedene Aufgaben und Rollen im Softwareentwicklungsprozess besetzt und effizient wahrgenommen werden. An diesem Prozess sind sowohl die Mitglieder des Konsortiums als auch verschiedene Dienstleister beteiligt. Diese Nutzengemeinschaft wird als Ökosystem bezeichnet. Es ist ein elementares Ziel, nach der Gründung eines Konsortiums starke Marktteilnehmer für die Mitwirkung in diesem Ökosystem zu gewinnen.

Wenn die Entscheidung für die Gründung eines Konsortiums fällt, besteht eine der zu bewältigenden Herausforderungen in der Auswahl eines geeigneten Piloten. Die mit dem ersten Release zu realisierenden Funktionen und Features sollten den Kernbedarf der Gruppe treffen und gleichzeitig eine ausreichend hohe Strahlkraft besitzen. Unter dieser Prämisse können mit einem hinreichend großen Portfolio weiterer potenzieller Entwicklungsprojekte zusätzliche Unternehmen als Mitglieder im Konsortium gewonnen werden (zur wirtschaftlichen Auswirkung der Mitgliederentwicklung siehe auch Szenarien in Kapitel 5).

Kapitel 3

Anforderungen und Voraussetzungen

für die konsortiale Softwareentwicklung

INHALTSVERZEICHNIS

3 Anforderungen und Voraussetzungen für die konsortiale Softwareentwicklung

3.1 Einleitung

Mit der Absicht, Software konsortial und auf Open-Source-Basis zu entwickeln, wird es notwendig, eine tragfähige Vision von den damit verbundenen Prozessen und Werkzeugen für die Entwicklung und Bereitstellung von Software zu formulieren, die es ermöglicht Systemgrenzen ohne Ablösung bestehender Systeme zu überwinden und Geschäftsprozesse mit spezifisch geringeren Kosten abzubilden. Dazu ist es erforderlich, ein Verständnis dafür zu entwickeln, welche technischen und organisatorischen Rahmenbedingungen von den Beteiligten geschaffen und gelebt werden müssen, um die gestellten Anforderungen erfüllen sowie die angestrebten Chancen und Vorteile heben zu können.

Im Folgenden wird zunächst allgemein auf die Anforderungen an die Software eingegangen. Danach wird das Werkzeugangebot vorgestellt, welches grundsätzlich geeignet erscheint, die technischen Anforderungen zu erfüllen. Danach werden verschiedene Entwicklungsmethoden unter dem Blickwinkel der Eignung für die konsortiale Entwicklung von Open-Source-Software (OSS) untersucht.

Die abschließende Bewertung enthält eine Aussage darüber, ob auf der Basis der getroffenen Annahmen die eingangs gestellten Anforderungen erfüllbar sind.

Die organisatorischen und rechtlichen Fragen als Voraussetzung für die konsortiale Softwareentwicklung werden in Kapitel 4 behandelt.

3.2 Anforderungen an die Software

Grundsätzlich gilt, dass die vom Konsortium bereitgestellte OSS mindestens das heute mit der proprietären Software vorzufindende qualitative Niveau erreichen muss. Dies betrifft im Bereich der Netzleitstellen neben der eigentlichen Funktionalität besonders Systemeigenschaften wie Reaktionszeiten und Verfügbarkeit.

Darüber hinaus müssen einige weitergehende Anforderungen erfüllt werden, um die in Kapitel 2 genannten Schwachstellen zu überwinden und eine verbesserte Wirtschaftlichkeit zu erzielen (siehe Kapitel 5). Zu diesen Anforderungen gehören im Wesentlichen:

- **Erhöhte Sicherheit**

  Besonders im Umfeld von Netzleitstellen haben sich die Anforderungen an die IT-Sicherheit in den letzten Jahren massiv verschärft. Auf der Basis der geltenden Normen und Richtlinien (ISO/IEC, BSI-Grundschutz, BDEW-Whitepaper usw.) muss eine (in welcher Form auch immer) zu integrierende OSS diese Anforderungen nicht nur erfüllen, sondern sollte aus diesem Aspekt heraus Verbesserungen bringen. Die Betrachtung der IT-Sicherheit ist auch deswegen so wichtig, weil an vielen Stellen noch angenommen wird, dass der Einsatz von OSS eher ein Sicherheitsrisiko sei.

- **Stärkere Modularität und verbesserte Wartbarkeit**

  Die vorhandenen Systeme, die im Kern häufig noch auf inzwischen veralteter Technologie basieren, werden durch die Integration zusätzlicher Anforderungen immer komplexer. Der modulare Aufbau von Softwaresystemen führt zu einer größeren Flexibilität für den Anwender, eine Anpassung an veränderte oder neue Aufgabenstellungen durchzuführen und hilft Komplexität zu reduzieren oder gänzlich zu vermeiden. Bei den meisten heutigen Netzleitsystemen findet man diese Modularität zwar hin und wieder in Marketingunterlagen, in der Realität hat man es aber mit monolithischen Strukturen zu tun (siehe Kapitel 2). Eine Ergänzung dieser Systeme oder ein Ersatz dort integrierter Funktionen durch angekoppelte Softwaremodule innerhalb einer geeigneten Softwarearchitektur wäre ein Schritt in die gewünschte Richtung.

- **Höhere Qualität der ausgelieferten Software**

  In der Praxis ist die Fehlerhäufigkeit von ausgelieferter Software trotz zugesagter und auch durchgeführter Qualitätssicherungsmaßnahmen beim Hersteller häufig auf zu hohem Niveau. Dies führt zu Verzögerungen bei der Einführung, erhöhtem Personalaufwand und Störungen in den Arbeitsprozessen.

- **Bessere Anpassbarkeit an Veränderungen in den Geschäftsprozessen**

  Gerade in der aktuellen Situation der Energieversorgung stellt sich häufig die Notwendigkeit, Geschäftsprozesse kurzfristig verändern zu müssen. Die dafür erforderlichen Anpassungen vorhandener oder die Beschaffung neuer IT-Werkzeuge können diesen zeitlichen Anforderungen oft nicht gerecht werden.

- **Verbesserte Software-Ergonomie**

  Die Software-Ergonomie charakterisiert die Gebrauchstauglichkeit von Software im Sinne von Benutzerfreundlichkeit. Kriterien sind u. a. leichte Erlernbarkeit, Bedienbarkeit und Verständlichkeit sowie weitere Qualitätskriterien (siehe hierzu DIN EN 9241, Teil 110 "Grundsätze der Dialoggestaltung"). Nicht selten klagen Anwender über eine schlechte Software-Ergonomie ihrer IT-Werkzeuge, was sich letzten Endes auch durch höhere Bearbeitungszeiten bemerkbar machen kann, den Schulungsaufwand erhöht, die Nutzung der Software auf Kernfunktionen beschränkt, Zusatzaufwände durch Fehlbedienung erzeugt und damit letztlich die Effizienz der eingesetzten Software nicht unerheblich senkt.

3.3 Werkzeuge

3.3.1 Bereitstellung einer Projekt-Hosting-Plattform

3.3.1.1 Fakten

Grundlagen des Projekt-Hosting

Da mit einer konsortialen Softwareentwicklung immer eine unternehmensübergreifende Kommunikation verbunden ist, zählt hierzu eine Infrastruktur, die dies im Besonderen für das Projekt- bzw. Entwicklerteam, aber auch für alle Mitglieder sicherstellt. Die möglichen Varianten zu deren Aufbau bzw. deren Bereitstellung durch einen Dienstleister (Service-Provider) werden nachfolgend dargestellt.

Bevor näher auf die einzelnen Varianten eingegangen wird, soll an dieser Stelle der Begriff Hosting bzw. Webhosting erläutert werden. In der Terminologie der Anbieter gliedern sich deren Dienste in drei Schichten:

1. Hardware, auch IaaS (Infrastructure-as-a-Service)
2. Betriebssystem und Datenbank, auch PaaS (Plattform-as-a-Service)
3. Anwendung, auch SaaS (Software-as-a-Service).

Mit der ersten Schicht werden Server bereitgestellt, die ständig online sind. Da häufig das erforderliche Know-how für das Betreiben solcher Server nicht vorhanden oder der Betrieb eigener Hardware nicht wirtschaftlich ist, greift man in diesen Fällen gerne auf die Angebote von sogenannten Service-Providern zurück.

Die zweite Schicht umfasst das auf dem Server installierte Betriebssystem (Windows, Linux etc.) und eine ebenfalls auswählbare Datenbank. Diese Schicht wird von den Service-Providern als "Plattform" bezeichnet.

Auf der dritten Schicht laufen die Anwendungsprogramme und Services. Installiert und betreibt man diese Software nicht selber, sondern mietet auch diese, so spricht man von Software-as-a-Service.

Grundsätzlich können alle drei Schichten selbst betrieben werden. Jedoch stellt die Miete insbesondere der ersten und zweiten Schicht für Start-up-Unternehmen und Kollaborationen wie das in dieser Studie beschriebene Konsortium eine wirtschaftliche Alternative dar, die es erlaubt, die zur Verfügung stehenden Personalressourcen gerade am Anfang stärker für den Aufbau des Unternehmens bzw. die Entwicklung der Geschäftstätigkeit zu nutzen.

Die Provider bieten für die Schichten 1 und 2 in der Regel "fertige", aber natürlich auch frei nach den Bedürfnissen des Kunden konfigurierbare Dienstleistungspakete an. Die einmal gewählte Konfiguration lässt sich üblicherweise dem Bedarf anpassen. Man zahlt also nur für die Dinge, die aktuell benötigt werden.

Bekannte Anbieter für solche Dienstleistungen sind Unternehmen wie z. B. Amazon, 1+1, Strato.

Kriterien für die Providerauswahl

Neben den Kosten der gewählten Konfiguration (Serverhardware (Prozessortyp, Arbeitsspeichergröße, Festplattentyp), Betriebssystem, Datenbank, Speicherraum, Anzahl der E-Mail-Adressen, Bandbreite etc.) und den damit verbundenen kostenrelevanten Service Level Agreements (SLA) können zur Analyse der Angebote grundsätzlich die sechs von der Open Cloud Business Initiative (OCBI) definierten Prinzipien [3-1] als gute Leitlinien für die Auswahl des "richtigen" Service-Providers herangezogen werden (Die OCBI ist eine von der OSBF (Open Source Business Foundation) initiierte Gemeinschaft von Anbietern von Cloud-Anwendungen.).

Die Prinzipien der Gemeinschaft lauten:

1. Alle User- und Metadaten eines Services werden in einem offenen Standardformat dargestellt.
2. Die Funktionalität eines Services wird über offene Standard-Schnittstellen exponiert.
3. Jeder Service-Consumer kann den Service ohne jede Diskriminierung nutzen.
4. Eigentums- und Zugriffsrechte sind für alle verarbeiteten Daten durch den Benutzer festlegbar.
5. Der Service-Provider achtet die Rechte an den Daten des Benutzers/Service-Consumers, die der ihm gewährt.
6. Open-Cloud-Services stimmen über geeignete Prozesse/Infrastruktur alle Veränderungen/ Erweiterungen des Services in einer Community ab.

Besonders den Prinzipien vier und fünf kommt eine hohe Bedeutung zu, da viele der heute vorherrschenden Plattform-Services noch eine eindeutige Abgrenzung der Eigentümer- und Verwertungsrechte an den Businessdaten vermissen lassen. Ein Blick in die Allgemeinen Geschäftsbedingungen verschafft hier Klarheit zu dem bei Vertragsabschluss anwendbaren Recht sowie zum Gerichtsstand.

3.3.1.2 Varianten

Besonders in der Startphase bietet sich das Webhosting für die Schichten 1 und 2 an, um ohne Zeitverlust und zusätzliche Investitionen mit der Arbeit (des Requirement- und/oder Entwickler-Teams) beginnen zu können. Es muss also vom Konsortium ein Provider ausgewählt werden. Nach der Wahl des Providers und der in der ersten und zweiten Schicht einzusetzenden Komponenten gilt es im Konsortium über die Ausstattung der dritten Schicht zu entscheiden. Auf dieser dritten Schicht wird die eigentliche Kommunikationsinfrastruktur eingerichtet. Diese beinhaltet vorrangig die relevanten Entwicklungswerkzeuge und Werkzeuge für die Projekt- und Teamkoordination, wie z. B.:

- Website
- Newsforum, Blogs
- Mailing list
- Quellcodeverwaltung (Code Repository)
- Issue tracking system bzw. bug tracking system (z. B. Jira oder Bugzilla)
- Downloadserver
- etc.

Die Zusammenfassung dieser Werkzeuge wird auch "Software-Forge" (zu Deutsch: Softwareschmiede) genannt. Darüber hinaus wird für diese Art von Software aufgrund des Einsatzes in Softwareentwicklungsprojekten auch der Begriff "Projekt-Hosting-Software" verwendet.

Eine vorrangig nur von den teilnehmenden Entwicklern verwendete Programmierumgebung (Integrated Development Environment, IDE) läuft üblicherweise lokal auf deren Workstations und wird für die Erstellung des eigentlichen Codes genutzt.

Die eingesetzten Werkzeuge ("Forge") und die Programmierumgebung haben technisch nichts mit der konsortial zu entwickelnden Software zu tun.

Variante 1A (ausgewählter Provider und selbstausgewählte (Einzel-)Werkzeuge)

Hier werden die einzelnen Werkzeuge vom Konsortium selbst ausgewählt, installiert und administriert. Hierdurch und durch die zusätzlich erforderliche Abstimmung der Werkzeuge untereinander entsteht ein nicht zu vernachlässigender Aufwand, der schon nach kurzer Zeit eigene Mitarbeiterressourcen auf der Seite des Konsortiums erfordern könnte.

Variante 1B (ausgewählter Provider und ausgewählte Software-Forge)

Hier wird eine Software-Forge als fertiges Produkt mit bereits aufeinander abgestimmten Werkzeugen ausgewählt. Man kann unter kommerziellen Produkten, wie Team Forge oder Cloud Forge (beide von collab.net) oder Open-Source-Produkten wie Redmine oder GForge wählen.

Variante 2 (Auswahl eines Projekt-Hosting-Plattformanbieters)

In dieser Variante werden alle drei Schichten von einem Anbieter als Paket angeboten, was als Projekt-Hosting-Plattform bezeichnet wird. Es gibt auch hier eine größere Anzahl von Anbietern. Manche Plattformen sind für Open-Source-Projekte kostenfrei nutzbar; für Plattformangebote zur Entwicklung von proprietärer Software, die es erlauben, sogenannte Private Repositorys zu erstellen, die nicht öffentlich einsehbar sind, ist eine monatliche Gebühr zu entrichten. Anbieter wie GitHub verfügen zudem über eine Enterprise-Version, die als eigene Instanz im Unternehmensnetzwerk genutzt werden kann.

Variante 3 (Projekt-Hosting unter dem Dach einer Foundation)

Die dritte Variante ist das Projekt-Hosting unter dem Dach einer Foundation, wie Apache oder Eclipse. Auch hier werden alle drei Schichten, sowie weitere Werkzeuge und Dienstleistungen angeboten. Die genannten Foundations nutzen keine der zuvor erwähnten Forges, sondern haben eigene Lösungen aufgebaut. Die Nutzung dieser Angebote ist jedoch verknüpft mit der Festlegung auf eine bestimmte Lizenz (hier: Apache Lisence 2.0 bzw. Eclipse Public Lisence 1.0), unter der der zu entwickelnde Code zu veröffentlichen ist.

3.3.1.3 Analyse und Bewertung

Überlegungen zur Variante 1

Bei der Auswahl des Providers in den Varianten 1A und 1B sind ggf. einige Restriktionen aus der Sicht des Konsortiums und seiner Mitglieder zu beachten. So sind einige Anbieter keine europäischen Unternehmen, sodass vor der Entscheidung geprüft werden muss, ob die AGBs mit den Interessen oder den rechtlichen Verpflichtungen des Konsortiums in Einklang zu bringen sind. Unternehmen wie Amazon scheiden aus Beratersicht aufgrund des sogenannten "patriot acts" als Anbieter aus, mit dem sich nach US-Gesetzgebung regierungsnahe Organisationen zum Zwecke der Terrorbekämpfung Einblicke in vertrauliche, firmeninterne Daten verschaffen dürfen.

Um eine hohe Ausfallsicherheit zu erreichen, werden von guten Anbietern bereits die modernsten technologischen Lösungen vorgehalten. Notwendige Wartungsarbeiten werden generell in last-schwachen Zeiten durchgeführt und sind häufig auf ein Zeitfenster von maximal vier Stunden begrenzt. Die garantierte Verfügbarkeit, Reaktions- und Wiederherstellungszeiten sind in gesonderten SLAs zu vereinbaren.

Grundsätzlich sollten bei der Auswahl nur Angebote berücksichtigt werden, die Programmiersprache, Datenbanksystem, Betriebssystem oder Server wählbar lassen. Die jeweiligen Angebote der Anbieter sollten außerdem darauf geprüft werden, ob sie vereinzelt proprietäre Dienste verwenden, die wiederum von anderen Anbietern nicht unterstützt werden und somit erneut zu einem Lock-in führen.

Überlegungen zur Variante 2

Die zuvor genannten Hinweise gelten ebenso für die Auswahl eines Anbieters in der Variante 2.

Darüber hinaus ist zu prüfen, inwieweit der vorgegebene Entwicklungsprozess und die angebotenen technischen Rahmenbedingungen mit den Anforderungen des Konsortiums übereinstimmen. Es sollte ein Anbieter gewählt werden, auf dessen Plattform eine größere Anzahl von aktiven Projekten gehostet wird. Es ist als vorteilhaft anzusehen, wenn der vom Anbieter auf der Plattform etablierte Entwicklungsprozess über mehrere Jahre erprobt und dieser einer größeren Gruppe von Entwicklern schon aus anderen Projekten bekannt ist.

Man kann demnach mit der Auswahl der Hosting-Plattform auch die Aufmerksamkeit einer bestimmten Entwickler-Community auf sich bzw. das eigene Projekt lenken.

Es ist noch zu erwähnen, dass das Projekt-Hosting von Open-Source-Projekten von einigen Anbietern in Abhängigkeit von der gewählten Lizenz eingeschränkt wird. Da noch nicht abzusehen ist, welche konkrete Lizenz für das erste (Pilot-)Projekt ausgewählt wird und ob für weitere Projekte vielleicht andere Lizenzen ausgewählt werden dürfen, ist möglichst ein Provider auszuwählen, der hierzu keine Einschränkungen vorsieht.

Überlegungen zur Variante 3

Die Vorteile, die das Hosting unter dem Dach einer Foundation bietet, sind insbesondere die Ersparnis von Zeit und Arbeit für die Festlegung einer eigenen Governance und die Definition von Spielregeln für den Entwicklungsprozess. Diese sollen einen verbindlichen Handlungsrahmen schaffen und eine möglichst hohe Anzahl von denkbaren Konfliktfällen zwischen den Akteuren bzw. Stakeholdergruppen vorab lösen. Dazu zählt auch das Zusammenspiel der Rollen Comitter und Contributor (siehe Kapitel 4) zu beschreiben und zu definieren, wie man den Status dieser Rolle erlangt. Die weiteren Regelungen zur Governance haben grundsätzlich die Aufgabe, durch geeignete rechtliche und faktische Arrangements die Spielräume und Motivationen der Akteure für opportunistisches Verhalten einzuschränken.

Das Hosting eines Projektes auf einer solchen Plattform bietet den Zugang zu vielen weiteren Tools aus anderen Open-Source-Projekten der Foundation, die vom eigenen Projekt mit geringen Modifizierungen und anschließender Zertifizierung wiederverwendet werden können. Hinzu kommt die sofortige Sichtbarkeit des eigenen Projektes für eine große Entwicklergemeinschaft, was dabei helfen kann, den eigenen Entwicklungsaufwand zu reduzieren.

Gleichzeitig ist strategisch zu bewerten, ob dieser "Fullservice" für die Lernkurve der Mitglieder eines Konsortiums nicht auch nachteilig wirken kann, da mit dem Beitritt zur Foundation auch die Akzeptanz der Governance und der zu verwendenden Lizenz sowie die Anpassung an die Organisationsstrukturen verbunden ist.

Bewertung

Der Entwicklungsprozess auf jeder der vorgenannten Plattformen unterstützt die Entwicklung von modularen Softwarekomponenten und damit die verbesserte Wartbarkeit sowie die bessere Anpassbarkeit. Weiterhin kann eine höhere Qualität der ausgelieferten Software sichergestellt werden.

Unabhängig von den betrachteten Varianten wird für das angedachte Konsortium und dessen Aktivitäten in jedem Fall empfohlen, ein den aktuellen Anforderungen genügendes Hostingangebot auszuwählen, um die zur Verfügung stehenden Personalressourcen für das Management der eigentlichen Softwareentwicklung und dem Aufbau des Konsortiums zu nutzen.

Wir empfehlen, die Anforderungen an ein Hostingangebot im Vorfeld bzw. parallel zu einem ersten Umsetzungsprojekt zu ermitteln und das Entwicklungsprojekt zunächst mit einem Anbieter der Variante 1 oder 2 zu beginnen. Vor dem Hintergrund des geplanten Vorhabens wird für einen schnellen Start aus Beratersicht die Variante 1B unter Verwendung eines Open-Source-Produktes favorisiert.

Entwickeln sich im Laufe der Zeit höhere oder neue Anforderungen (z. B. an die Verfügbarkeit oder die Integrität), ist zu prüfen, ob der Provider oder Projekt-Hoster sein Angebot (auch hinsichtlich verbesserter Tools und Dienste) inzwischen entsprechend den Wünschen des Konsortiums erweitern kann oder ein neuer Anbieter gefunden werden muss.

Ein Wechsel des Providers ist in Open-Source-Projekten nicht unüblich, da über die Zeit auch das Know-how im Konsortium wächst und als Reaktion darauf zu gegebenem Zeitpunkt auf ein passenderes, den Bedürfnissen entsprechendes Angebot gewechselt wird.

Nach einer gewissen Anfangsphase kann ein solcher Providerwechsel insbesondere aus strategischen Gesichtspunkten sinnvoll sein. So hat z. B. das TOPCASED-Projekt der Airbus-Gruppe erst nach mehreren Jahren den Wechsel zur Eclipse Foundation vollzogen und wird dort nun als Polarsys-Projekt weitergeführt [3-2]. Die Gründe für den Wechsel waren unter anderem die Sicherstellung der Langzeit-Wartbarkeit der Software und ein gesicherter Prozess für das IP-Management (dies gewinnt an Bedeutung, wenn die Integration von Komponenten unter lizenzrechtlichen Aspekten zulässig ist).

3.3.2 Auswahl einer Technischen Plattform

3.3.2.1 Ausgangssituation und Fakten

In den beteiligten Unternehmen ist die Erkenntnis gewachsen, dass bei der Notwendigkeit zu einer weitergehenden Systemintegration die direkte Kopplung zweier Systeme über immer wieder anzupassende, proprietäre Schnittstellen unterschiedlicher Komplexität zwangsläufig in ein Schnittstellenchaos (siehe Abbildung 3-1) mündet und dies zu enormen Kostensteigerungen führt.

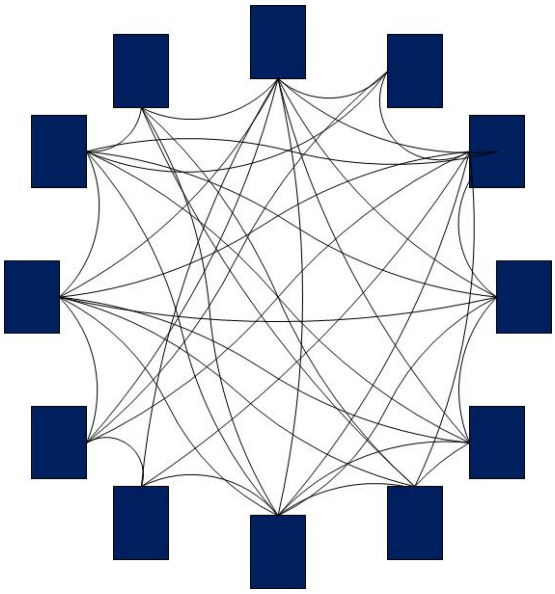


Abb. 3-1: Direkte Systemkopplung mit uni- und bidirektionalen Schnittstellen

Bereits in den Interviews wie auch in den darauf folgenden Workshops wurde deshalb darüber diskutiert, ob im Konsortium "nur" einzelne Applikationen / Module entwickelt werden sollen, für deren spätere Integration der jeweilige Endanwender selbst eine Lösung finden muss oder ob vom Konsortium gleichzeitig eine Integrationsplattform mit standardisierten Schnittstellen zu den wichtigsten bzw. zumindest zu den im ersten Release benötigten Systemen angeboten werden soll.

Letzteres erfordert aus der Sicht des Konsortiums die einmalige Auswahl einer am Softwarebedarf der nächsten Jahre ausgerichteten Plattform und der zugehörigen Komponenten. Sofern für die Plattform Open-Source-Produkte ausgewählt werden, fallen auch nur die Investitionen für den Auswahlprozess und ggf. die Kosten für die Bereitstellung auf einem Downloadserver des Konsortiums an.

Der technologische Aufbau einer solchen Plattform ist schematisch in den Abbildungen 3-2a und 3-2b dargestellt. Abbildung 3-2a zeigt die Situation nach deren Einführung, in der die Systeme zunächst über eine proprietäre Schnittstelle anzubinden sind.

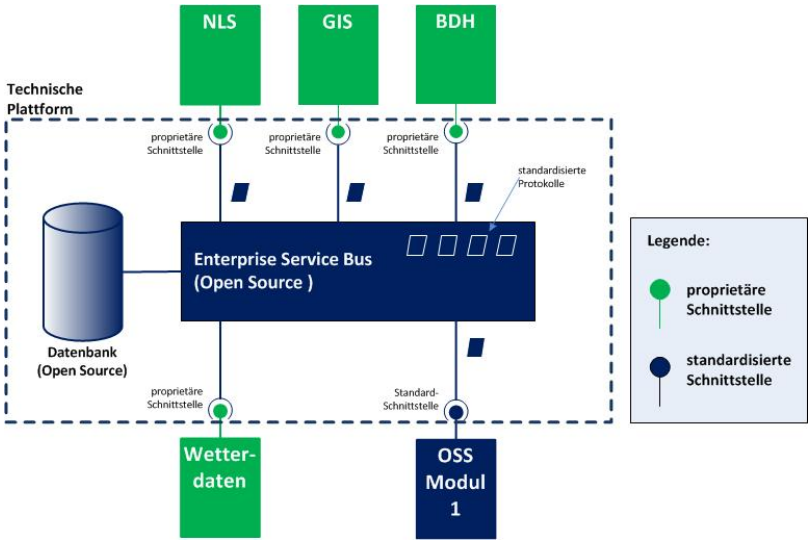


Abb. 3-2a: Struktur Technische Plattform (1. Optimierung)

Die Abbildung 3-2b hingegen zeigt den idealen Zielzustand, in dem die erforderlichen Systeme über Standardschnittstellen an die Plattform angebunden sind.

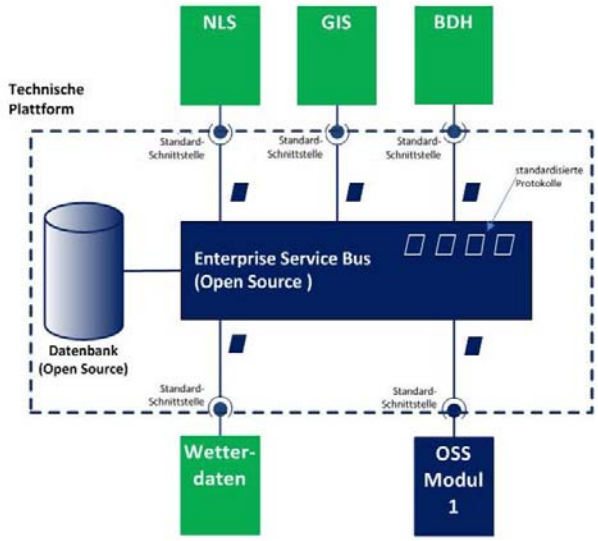


Abb. 3-2b: Struktur Technische Plattform (idealer Zielzustand)

Die Nutzung der vom Konsortium ausgewählten technischen Plattform wird den Anwendern aber keineswegs vorgeschrieben. Unternehmen, die bereits eine ähnliche Plattform einsetzen, haben die Möglichkeit entweder nur die konsortial entwickelten Anwendungen in ihre Landschaft zu integrieren (siehe Abbildung 3-3) oder die Kopplung beider Plattformen (siehe Abbildung 3-4) zu realisieren.

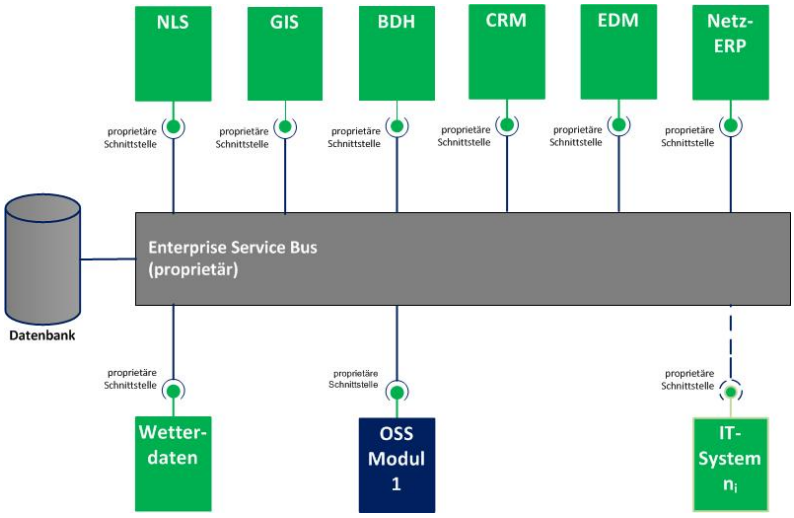


Abb. 3-3: Integrationsvariante 1 für KSE-Produkte

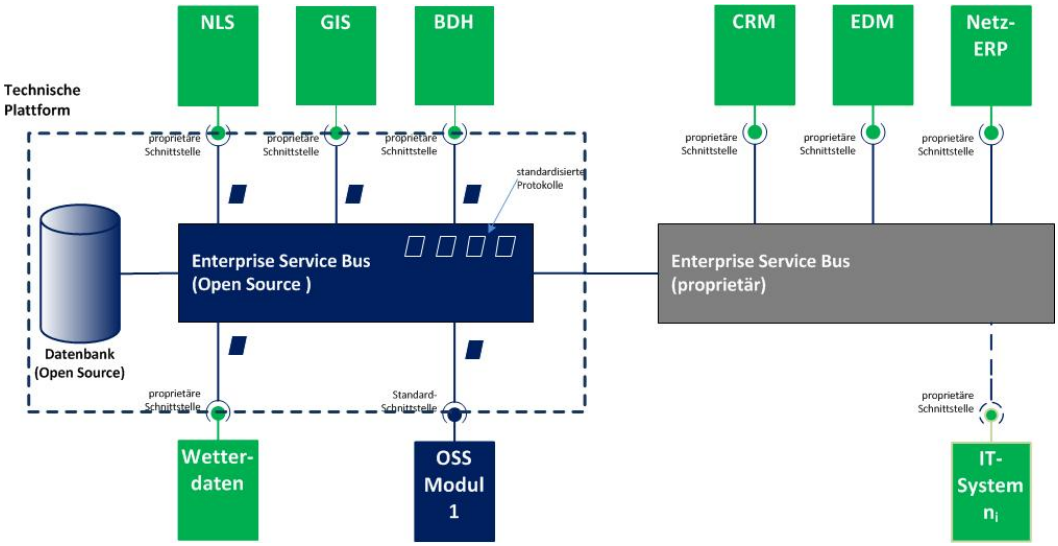


Abb. 3-4: Integrationsvariante 2 für KSE-Produkte

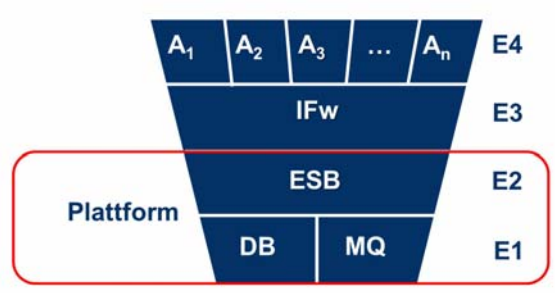
Die Bereitstellung einer Technischen Plattform bietet jedoch insbesondere für neue, kleinere Mitgliedsunternehmen die Vorteile, dass sie sich zum einen nicht mehr mit der erstmaligen Auswahl einer solchen Plattform auseinandersetzen müssen und ihnen zum anderen (bei vorhandenen Adaptern und Schnittstellen zu den von Ihnen genutzten IT-Systemen) sehr schnell und zu minimalem Customizing-Aufwand eine dokumentierte Infrastruktur bereitgestellt wird, die eine vollumfängliche Nutzung der konsortial entwickelten Software erlaubt.

Die zuvor beschriebene Plattform, folgt dem inzwischen weit verbreiteten Architekturkonzept der "Service Oriented Architecture" (SOA). Die Vorteile liegen in der Trennung von Schnittstellen und dahinterliegenden Implementierungen sowie in der Aufteilung eines komplexen Systems in verschiedene interagierende Dienste.

Häufig, wenn auch nicht zwingend anzutreffende Komponenten in SOA-Konzepten sind:

- Datenbank (DB),
- Message Queuing (MQ),
- Enterprise Service Bus (ESB) bzw. auch mit Middleware bezeichnet.

In einem vereinfachten, logischen Vier-Ebenen-Modell sind dies die Komponenten der Ebene 1 und 2 (siehe Abbildung 3-5).



Legende:
IFw=Industry Framework, A=Applikation)

Abb. 3-5: Vier-Ebenen-Modell

Die Komponenten der technischen Plattform sollten mindestens die in Abbildung 3-6 gezeigten Werkzeuge und Dienste bereitstellen.

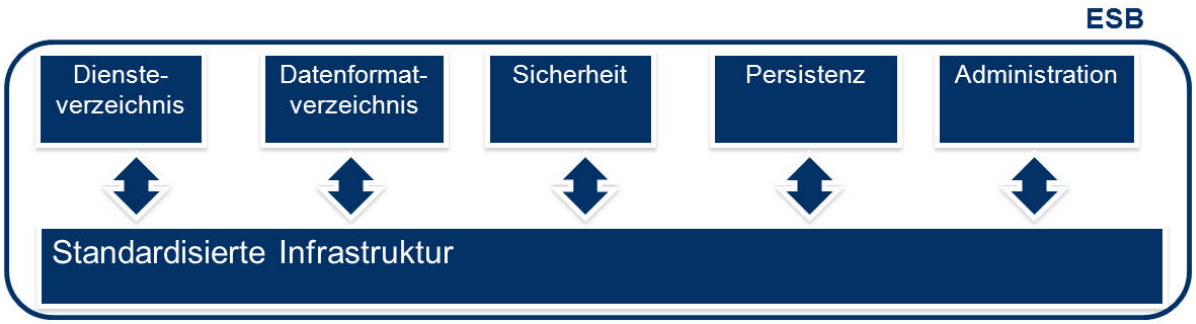


Abb. 3-6: Werkzeuge und Dienste innerhalb der Infrastruktur

3.3.2.2 Analyse und Bewertung

Vor dem Hintergrund, dass die Produkte des Konsortiums flexibel an die vorhandenen Systeme der Netzbetreiber ankoppelbar sein müssen (mit oder ohne Plattform, siehe Abbildung 3-2 bis 3-4), wird eine SOA als die geeignetste Form der Architektur für das geplante Vorhaben empfohlen. Hierdurch werden die im Abschnitt 3.2 gestellten Anforderungen an stärkere Modularität und verbesserte Wartbarkeit sowie die Anpassbarkeit an Veränderungen in den Geschäftsprozessen am besten unterstützt.

Es wird empfohlen, die Auswahl einer solchen Plattform der eigentlichen Softwareentwicklung voranzustellen. Zudem finden sich in den aktuellen Publikationen von IEC, IEEE, VDE/DKE und anderen Verbänden mit Bezug auf die zukünftigen Herausforderungen von Netzbetreibern zur Steuerung von intelligenten Netzen viele Aussagen, die allesamt einen SOA-Ansatz präferieren.

Auf dem Weg zu einer konkreten Architektur für die konsortial zu entwickelnde Software sind die folgenden drei Schritte zu vollziehen:

1. Entscheidung für SOA als allgemeines strukturierendes Architekturprinzip der zu entwickelnden Software.
2. Entscheidung für bestimmte Dienste, die als Teil einer konkreten Architektur benötigt werden.
3. Entscheidung für bestimmte Softwarekomponenten (Open-Source-Komponenten oder proprietäre Komponenten), welche die Dienste realisieren.

Nach einer Grundsatzentscheidung für eine SOA folgen die Entscheidungen zu den Punkten 2 und 3 aus den Anforderungen und der Projektplanung.

Analog zur Auswahl der Hosting-Plattform wird auch hier die Auswahl von etablierten Open-Source-Produkten (bzw. -Komponenten) für die technische Plattform favorisiert.

Die Plattform sollte über die in Abbildung 3-6 gezeigten Werkzeuge und Dienste verfügen und schon über mehrere Jahre in erfolgreichen Open-Source-Projekten eingesetzt werden. Idealerweise wird jeder Teil von einer ausreichend großen Community und nicht nur von einem einzelnen Unternehmen weiterentwickelt. Der Enterprise Service Bus (ESB) sollte eine möglichst große Anzahl von Standardschnittstellen für verschiedene Technologien bereitstellen (.net, Java, C, C++) und bereits einen umfangreichen Pool von Adaptern zu im Markt gängigen IT-Systemen vorhalten, da dies wirtschaftliche Vorteile für den Anschluss der bei den Mitgliedern vorhandenen IT-Systeme mit sich bringt.

Die Empfehlung für Open-Source-Produkte wird zum einen aus Kostengründen ausgesprochen und weil zum anderen zu erwarten ist, dass sich Dienstleister zunehmend gut mit den oben genannten Komponenten auskennen.

Es gilt dabei eine "Komplettlösung" eines großen Anbieters wie IBM oder Microsoft zu vermeiden, da diese durch kleine, aber sich addierende Unterschiede in ihren Produkten versuchen, die Kunden erneut an sich zu binden und somit wieder in eine Lock-in-Situation führen.

Es wird überdies empfohlen, auf der noch auszuwählenden technischen Plattform in einem ersten Release (ggf. als Beta-Version) einen integrierten, am Bedarf der Gründungsunternehmen ausgerichteten Prototypen zu schaffen, in dem schon große Teile eines lauffähigen Moduls enthalten sind. Das bedeutet, dass ein Teil des Systems über alle Schichten der Architektur hinweg erstellt wird; also beginnend von der Benutzerschnittstelle bis zur Datenhaltung. Ein solcher vertikaler Prototyp (auch Durchstich genannt, siehe Abbildung 3-7) dient dazu, komplexe Funktionalität zu demonstrieren und ausgewählte Aspekte in ihrer Gänze zu zeigen.

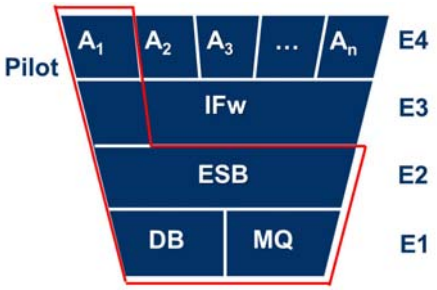


Abb. 3-7: Vertikaler Prototyp (Durchstich)

3.4 Methoden zur Softwareentwicklung

3.4.1 Fakten

In den 70er Jahren gelangten die ersten sogenannten "klassischen" Softwareentwicklungsmethoden zu einer breiteren Anwendung. Hier ist vor allen Dingen das Wasserfallmodell zu nennen. Neben den Weiterentwicklungen dieses Phasenmodells kamen in den 80er und 90er Jahren weitere schwergewichtige Entwicklungsmodelle, wie das V-Modell XT oder Rational Unified Process (RUP), auf den Markt, die bis heute (insbesondere in ihren weiterentwickelten Formen) Anwendung finden. Diese planungsgetriebenen Methoden gerieten aber in vielen Fällen in die Kritik der Auftraggeber und Endanwender, da es häufig zu Kostenüberschreitungen kam und Software ausgeliefert wurde, die nicht im gewünschten Umfang den Anforderungen entsprach.

Ende der 90er Jahre wurde erstmals das Prozessmodell Extreme Programming (XP) vorgestellt, dem dann schnell weitere leichtgewichtige Methoden, wie z. B. Crystal Methodologies, Adaptive Software Development und Scrum folgten. Im Jahr 2001 einigten sich die Vertreter der verschiedenen Prozessmodelle im Agilen Manifest (siehe Anlage 3-2) auf einen gemeinsamen Nenner, wobei es bis heute keine wirklichen Bestrebungen gibt, diese zu einem einheitlichen Prozess auszubauen. Im Gegenteil entstanden weitere Softwareentwicklungsmethoden, wie Kanban, Feature Driven Development (FDD), Test Driven Development (TDD) und andere mehr.

In den heute laufenden Open-Source-Entwicklungsprojekten werden fast durchgängig agile Methoden bzw. Mischformen aus diesen Methoden eingesetzt. Das Entwicklerteam darf sich dabei nach den Vorgaben des Auftraggebers bzw. des eigenen Unternehmens die für den Entwicklungsprozesses geeignetsten Basistechniken und Methoden auswählen und bei Bedarf auch während des Entwicklungsprozess in regelmäßigen Abständen einer Prüfung und ggf. einer Anpassung unterziehen. Diese Vorgehensweise wird mit dem Begriff "Tailoring" bezeichnet.

3.4.2 Analyse und Strukturierung

3.4.2.1 "Klassische" planungsgetriebene Methoden

Die "klassischen" planungsgetriebenen Methoden folgen alle der in Abbildung 3-8 gezeigten Struktur und zeigen folgende Merkmale:

- Jede Phase muss vollständig abgearbeitet sein, bevor die nächste begonnen werden darf.
- Am Ende jeder Phase stehen fertiggestellte Ergebnispakete, in der Regel in Form von Dokumenten.
- Der Entwicklungsablauf ist streng vorgegeben.
- Eine Benutzerbeteiligung ist nur am Anfang des Projektes während der Anforderungsanalyse vorgesehen.

Abb. 3-8: Grundsätzliche Struktur planungsgetriebener Methoden

Weitere Merkmale "klassischer" Methoden sind:

- Spezifikation wird vom AG verantwortet.
- AN ist in der Lage ein Festpreisangebot abzugeben.
- AG kann die Leistung des ANs gemäß der Spezifikation abnehmen.
- Die Realisierung ist in vielen Fällen "unsichtbar" für den AG.
- Risiko- und Gewinnaufschlag durch den AN erhöhen die Kosten.

Die Modellvarianten Wasserfallmodell, V-Modell XT und Rational Unified Process (RUP) werden in [3-3] näher beschrieben.

3.4.2.2 "Moderne" agile Methoden

Bei den agilen Methoden handelt es sich zum Teil um Weiterentwicklungen der klassischen Methoden, aber auch um komplett neuartige Ansätze. Im Gegensatz zu den planungsgetriebenen Prozessen zeichnen sich die agilen Softwareentwicklungsmethoden durch eine andere Projektkultur aus. Diese Projektkultur verschiebt die Schwerpunkte auf die vier im "Agilen Manifest" (siehe Anlage 3-2) genannten Prinzipien:

Individuen und Interaktion

Anstatt den Schwerpunkt vorrangig auf Prozesse und Werkzeuge zu legen, rücken die beteiligten Personen und ihre Interaktionen in das Zentrum der Projektarbeit. Die Entwickler können dabei eingesetzte Techniken an die speziellen Bedürfnisse des laufenden Projektes anpassen. Die Anpassungen sind danach zwar verbindlich festgelegt, werden aber trotzdem ständig auf ihre Sinnhaftigkeit hinterfragt. Weiterhin wird die direkte Zusammenarbeit der Entwickler betont. Den Schwerpunkt beim Informationstransfer bilden nicht Dokumente, sondern die verbale Kommunikation der Entwickler untereinander.

Ausführbare Software

Das Ziel bildet zu jeder Zeit ausführbare Software bereitzustellen. Lange Entwicklungszeiträume, in denen die Software an verschiedenen Stellen separat erweitert und später in einer eigenen Integrationsphase zusammengebaut wird, werden vermieden. Das angestrebte Ideal ist eine Software, die vom Anwender jederzeit angefordert werden kann und die jederzeit ausführbar ist.

Zusammenarbeit mit dem Kunden

Der Kunde kann und soll an der Entwicklung teilnehmen, denn die Qualität der Software steigt mit ihrer Akzeptanz beim Kunden. Die Interaktion ist dabei nicht auf die Anforderungsanalyse beschränkt, sondern kann immer zum Ende eines festgelegten Entwicklungszeitraumes (einer Iteration) für eine einzelne Funktion oder ein für sich lauffähiges Feature stattfinden (siehe Abbildung 3-9). Diese Vorgehensweise erlaubt es dem Kunden korrigierend in den Projektverlauf einzugreifen.

Abb. 3-9: Vorgehensweise bei agilen Methoden

Die agilen Methoden zeigen folgende Merkmale:

- Sie sind in der Regel kollaborativ angelegt und werden deshalb praktisch in allen Open-Source-Projekten eingesetzt, da diese die Anforderungen am besten erfüllen.
- Im "magischen Dreieck aus Zeit, Kosten und Funktion" werden die Kriterien "Zeit" und "Kosten" in der Regel höher bewertet als "Funktion", da eine termingerechte Fertigstellung meist wichtiger ist, als eine funktionale Optimierung, welche für das Gesamtrelease ggf. nur noch einen relativ kleinen Mehrwert bietet oder nur von wenigen Anwendern benötigt wird.

  Im Umfeld von Netzleitstellen ist allerdings davon auszugehen, dass durch entsprechende Vereinbarungen unter den Beteiligten die betriebliche Eignung und Brauchbarkeit ausgelieferter Softwaremodule sichergestellt wird.
- Der Entwicklungsprozess erhöht die Transparenz für alle Beteiligten, was wiederum eine höhere Planungssicherheit zur Folge hat.
- Das iterative Vorgehen ermöglicht eine Umplanung im laufenden Projekt.
- Sie beinhalten Maßnahmen für oder sind von vornherein flexibel gegenüber sich ändernden Anforderungen oder wechselnden Teammitgliedern.
- Kurze Release-Zyklen (time-boxed) helfen, die Integrationsprobleme einer klassischen Entwicklung zu vermeiden.
- Es gibt verbindliche Releasetermine.

- Es finden unmittelbare Abstimmungen mit den zukünftigen Nutzern statt.
- Es wird nur die Funktionalität entwickelt, die benötigt wird.
- …

Die im Juli 2012 veröffentlichte Studie "Status Quo Agile" der Hochschule Koblenz [3-4] mit über 300 teilnehmenden Unternehmen untersuchte die Verbreitung und Nutzung von agilen Methoden. Sie zeigte:

- Die Nutzung agiler Methoden hat seit 2008 einen sehr starken Aufschwung genommen (nicht beschränkt auf Open Source).
- Über 75 % der Teilnehmer nutzen agile Methoden.
- Agile Methoden werden mehrheitlich mit "klassischen" Methoden kombiniert (sowohl als auch - Nutzung oder Mischform).
- Die Nutzung agiler Methoden geht mit einer wesentlich veränderungsorientierteren Unternehmenskultur einher.
- Bereits 25 % der Anwender nutzen agile Methoden auch in Non-IT-Bereichen wie Produktentwicklung, Prozessmanagement u. Ä.
- Kanban, Extreme Programming und insbesondere Scrum sind die verbreitetsten agilen Methoden.
- Umsteiger auf agile Methoden sehen deutliche(!) Verbesserungen bei der Erfolgsquote.
- 93 % der Unternehmen bewerten die agile Methode Scrum als "gut" oder "sehr gut" (im Gegensatz zu nur 50 % bei klassischen Projektmanagement-Methoden).
- Die Erfolgsquoten mit agilen Methoden werden signifikant besser bewertet als bei klassischem Projektmanagement.

Beispielhaft seien hier die agilen Methoden XP (Extreme Programming), Test Driven Development (TDD), Feature Driven Development (FDD), Kanban, Open UP und Scrum genannt.

Die Abbildung 3-10 zeigt, wie die agile Methode Scrum im Rahmen des Konsortiums zur Anwendung kommen könnte.

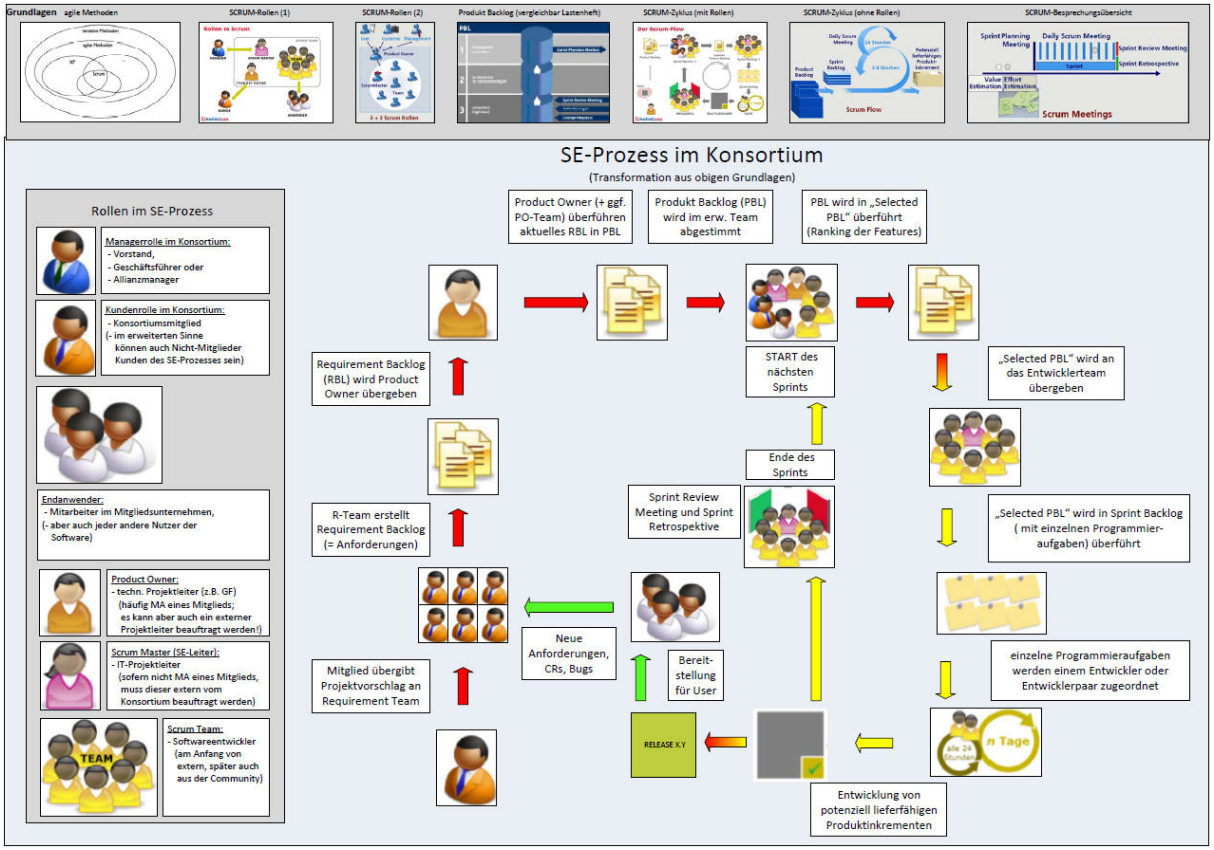


Abb. 3-10: SE-Prozess im Konsortium (hier im Beispiel mit SCRUM)

3.4.3 Bewertung der Eignung der Methoden für KSE

Für die konsortiale Softwareentwicklung auf der Basis von Open Source eignen sich grundsätzlich alle der genannten Entwicklungsmethoden. Die Auswahl der Methode oder einer Mischform (siehe auch Folie 52 in Anlage 3-3) hat keinen direkten Bezug zu den in Abschnitt 3.2 gestellten Anforderungen, sondern hängt eher davon ab, welche Vorgaben vom Konsortium an den Entwicklungsprozess hinsichtlich Transparenz und Einflussnahme gestellt werden und ob es sich bei der Softwareentwicklung um eine Auftragsarbeit oder einen Community-getragenen Entwicklungsprozess handelt.

Je nach Größe des Projektes und Größe des Entwicklerteams sollten auch die verwendeten Methoden und Basistechniken ausgewählt werden. Dabei gilt folgender Effizienz- und Schadensbegrenzungsgrundsatz: Je größer das Team, desto vorsichtiger sollte die Größe des zu realisierende Features und der dafür zur Verfügung gestellte Zeitraum bemessen sein, damit die Gefahr abnimmt, dass eine große Gruppe über einen größeren Zeitraum in eine falsche Richtung entwickelt.

Insbesondere bei der konsortialen Softwareentwicklung ist davon auszugehen, dass die Mitglieder des Entwicklungsteams überwiegend an verteilten Standorten arbeiten. Damit ergeben sich hohe Anforderungen an die Kommunikationsfähigkeit, das Zeitmanagement und die Disziplin der einzelnen Mitglieder. Generell ist bei solchen Rahmenbedingungen von etwas längeren Sprint-Zeiträumen auszugehen, da der Aufwand für Abstimmungen und das klassische Projektmanagement steigen.

Darüber hinaus sind bei größeren Projekten einige der agilen Methoden nicht wirklich skalierbar. So gehen bisherige Erfahrungen mit XP davon aus, dass XP ab einer Teamgröße von zehn Personen nicht mehr effizient durchführbar ist.

Bei Auftragsentwicklungen sollte das Entwicklerteam des Auftragnehmers im Rahmen der Vorgaben des Auftraggebers die Methode selbst wählen dürfen. Dies setzt jedoch voraus, dass ein solcher Rahmen vorab vom Auftraggeber definiert wurde. Im Konsortium fällt diese Aufgabe dem Vorstand, ggf. unter Beteiligung des technischen Beirats zu (siehe Kapitel 4). Es ist davon auszugehen, dass in dieser Variante der Softwareentwicklung das Produkt einer oder mehrerer (Teil-)Abnahmen und der Sourcecode einer IP-Kontrolle unterzogen wird.

Bei zusammengestellten Entwicklerteams mit mehreren vom Konsortium bezahlten Entwicklern sollten diese einer vom Konsortium gestellten oder ebenfalls extern ausgewählten Entwicklungsleitung unterstellt werden. Der Entwicklungsleiter wird danach in der Regel in Abstimmung mit dem Projektleiter des Konsortiums die Entwicklungsmethode auswählen bzw. vorgeben. Weitere Details müssen projektbezogen zwischen den jeweiligen Rollenträgern innerhalb der gewählten Entwicklungsmethode festgelegt werden.

Es darf in größeren Projekten auch unabhängig voneinander arbeitende Teilgruppen geben, sofern sichergestellt ist, dass die Entwicklungsarbeiten sich nicht beeinflussen oder überschneiden. Die Steuerung der Teilgruppen ist wiederum Aufgabe des Entwicklungsleiters.

Bei (Weiter-)Entwicklungen oder Modifizierungen in Form von Beiträgen (Contributions) zu einem Open-Source-Projekt ist die Entwicklungsmethode nicht mehr von Bedeutung. Hier kommt die tragende Rolle der IP-Kontrolle zu, da sichergestellt werden muss, dass keine nicht-lizenzkonformen Code-Bestandteile in das Projektrepository gelangen (siehe Kapitel 4).

3.5 Abschließende Bewertung

Zu Beginn dieses Kapitel (Abschnitt 3.2) wurden Anforderungen formuliert, die von konsortial entwickelter Software auf Basis von Open Source zu erfüllen sind, um als attraktive Alternative zur konventionellen Softwarebeschaffung infrage zu kommen.

Im darauf folgenden Abschnitt wurde nachgewiesen, dass die Werkzeuge und technischen Plattformen verfügbar sind, um OSS professionell zu entwickeln.

Die Auswahl der Methode oder einer Mischform hat keinen direkten Bezug zu den in Abschnitt 3.2 gestellten Anforderungen, sondern hängt eher davon ab, welche Vorgaben vom Konsortium an den Entwicklungsprozess gestellt werden.

Damit ist eine wesentliche Voraussetzung geschaffen, die genannten Anforderungen erfüllen zu können.

In dem Entwicklungsprozess von OSS muss sichergestellt werden, dass die bereits heute beim Einsatz von konventioneller, proprietärer Software geltenden Qualitätskriterien (Funktionalität, Zeitverhalten, Verfügbarkeit) eingehalten werden können. Dies kann vorausgesetzt werden, stellt aber hohe Anforderungen an die Qualität des Entwicklerteams.

Die genannten weitergehenden Anforderungen lassen sich ebenfalls erfüllen. Dies ist im Wesentlichen in einem iterativen, transparenten Entwicklungsprozess und der Offenlegung des Softwarecodes begründet:

- Eine *erhöhte Sicherheit* wird dadurch gewährleistet, dass durch die genannte Offenlegung des Codes und damit seine allgemeine Prüfbarkeit die Wahrscheinlichkeit für das Auffinden von Zugängen für Schadsoftware wesentlich höher ist und Fehler in der Software schneller gefunden werden. Diese Annahme wird mittlerweile auch von offizieller Seite (z. B. BSI) anerkannt. Im Übrigen unterliegt die OSS einer strikten Schreibkontrolle, mit der sichergestellt wird, dass niemand Beliebiges den Code verändern darf. Der Einsatz von OSS bietet per se keine Gewähr für ein sicheres System. Er bietet in diesem Prozess jedoch bedeutende strategische Vorteile [3-5]. Unabhängig davon muss natürlich jedes Softwaremodul so in die Systemumgebung des jeweiligen Unternehmens integriert werden, dass die Sicherheitsanforderungen eingehalten werden (Zonenkonzept, Firewalls usw.)

- Eine *stärkere Modularität* und *Verknüpfbarkeit mit vorhandenen Systemen* wird gewährleistet, wenn die im Rahmen der technischen Plattform empfohlene moderne Softwarearchitektur umgesetzt wird, in die auch die Netzleitsysteme und ggf. weitere technischen Systeme einbezogen werden müssen. Aus dieser Modularität ergibt sich automatisch eine bessere Wartbarkeit, wobei der Anwender in der Vergabe von Wartungsleistungen auch nicht an einen Dienstleister gebunden ist.

- Die *höhere Qualität* und die *schnellere Bereitstellung der ausgelieferten Software* kann durch den iterativen und transparenten Entwicklungsprozess erreicht werden, da hier die Anwender die Möglichkeit zur Einsichtnahme und damit auch zur Einflussnahme auf den Entwicklungsprozess haben. Das heißt noch vor Auslieferung eines ersten Release kann die Software auch von Anwendern begutachtet und getestet werden. Darüber hinaus müssen natürlich die vom Konsortium vorgegebenen Qualitätssicherungsmechanismen vor der Auslieferung eines Release eingehalten werden.

  Die bei OSS von Beginn an gegebene Einsicht in den Quellcode führt zudem zu einem besseren Verständnis der Software bei den Entwicklern und damit fast notgedrungen zu einer höheren Qualität bei späteren Softwareerweiterungen und -ergänzungen.

- Die *bessere Anpassbarkeit an Veränderungen in den Prozessen* beim Einsatz der konsortial entwickelten Software wird durch die Tatsache unterstützt, dass das Konsortium oder auch einzelne Unternehmen aufgrund des offenen Quellcodes jederzeit auf mehr als einen Anbieter für Entwicklung und Implementierung (sowie Customizing) zurückgreifen können, sodass Engpässe bei einem Anbieter über andere Anbieter ausgeglichen werden.

  Im Übrigen darf auch nicht verschwiegen werden, dass besonders im sensiblen Umfeld der Netzführung ein zu häufiges Einspielen von neuen Releases grundsätzlich zu einer unerwünschten Unruhe im Betriebsablauf führen kann. Die Mitgliedsunternehmen müssen also selber entscheiden, ob sie zeitnah nach den festgelegten Releaseterminen des Konsortiums (jährlich oder halbjährlich) die Installation der Software in ihrem Unternehmen anstoßen wollen.

  Ein wesentlicher Vorteil liegt auf jeden Fall darin, dass das Konsortium durch den Einfluss auf den Entwicklungsprozess eine wichtige Möglichkeit hat, den Qualitätssicherungsprozess im Sinne der Anwender zu steuern.

- Eine *verbesserte Software-Ergonomie* spielt dann eine große Rolle, wenn im Rahmen der OSS-Entwicklung auch eine entsprechende Oberfläche bereitgestellt werden soll. In diesem Fall hat das Konsortium in der Tat entsprechende Möglichkeiten, die Software-Ergonomie zu beeinflussen, z. B durch Einbeziehung entsprechender wissenschaftlicher Institute. Generell kann man wahrscheinlich sagen, dass einfach jede neue Generation an Software schlichtweg bessere Ergonomie hat, als ältere existierende, weil sie auf neuer Technologie aufsetzt. Die im Konsortium entwickelten Produkte können ihr volles Nutzenpotenzial umso mehr entfalten, je stärker der Wille der Mitglieder ist, ihre Arbeitsprozesse untereinander zu harmonisieren.

3.6 Zusammenfassung

Der Einstieg in die konsortiale Entwicklung von Open-Source-Software (OSS) als Alternative zur konventionellen Softwarebeschaffung muss gewährleisten, dass der heute erreichte Qualitätsstand hinsichtlich Funktionalität und Systemleistungen nicht unterschritten, sondern möglichst übertroffen wird. Zu solchen weitergehenden Anforderungen gehören eine erhöhte Sicherheit, eine stärkere Modularität und Offenheit gegenüber vorhandenen Systemen, eine höhere Qualität und kürzere Bereitstellungszeiten der Software sowie eine verbesserte Software-Ergonomie und Wartbarkeit.

Die Bewertungen in den vorangehenden Abschnitten haben gezeigt, dass diese Anforderungen durch geeignete Werkzeuge und Methoden erfüllt werden können.

Für die konsortiale Entwicklung von OSS sind gleich mehrere technische Voraussetzungen zu schaffen. Dazu zählt die Auswahl einer Projekt-Hosting-Plattform, auf der neben der erforderlichen Kommunikationsinfrastruktur auch die notwendigen Werkzeuge für die gemeinsame Softwareentwicklung bereitgestellt werden. Es wurden mehrere Varianten aufgezeigt, mit denen dies erreicht werden kann. Für die Startphase des Konsortiums wird die sogenannte Variante 1B empfohlen, in der ein Provider und eine Open-Source-Software-Forge auszuwählen ist.

Als mittelfristige Alternative hierzu eignen sich Projekt-Hosting-Plattformen unter dem Dach von bestehenden Foundations wie Apache oder Eclipse, wobei dies jedoch gleichzeitig an die Festlegung auf eine Lizenz und die Governance der Foundation gekoppelt ist. Es wird empfohlen, die Vor- und Nachteile dieser Variante aus strategischer Sicht zu überprüfen, wenn klarer ist, was für eine Software tatsächlich entwickelt werden soll und dann ggf. zu einem späteren Zeitpunkt dorthin zu wechseln.

Sofern nicht von der Plattform vorgegeben, sind vom Konsortium Regeln für die Nutzung und Zusammenarbeit auf dieser Plattform zu definieren.

Auf der Plattform sollte die Anwendung verschiedener Programmiersprachen möglich sein.

Unabhängig davon sind vom Konsortium organisatorische Regeln aufzustellen, die das Zusammenspiel für die Rollen der Beteiligten (Committer und Contributoren, siehe Kapitel 4) beschreiben und definieren, wie man diesen Status erhält. (siehe Kapitel 4)

Damit die vom Konsortium entwickelten Produkte möglichst schnell von allen Mitgliedern genutzt werden können, wird die Auswahl einer technischen Plattform empfohlen. Besonders für kleine Unternehmen soll mit der Bereitstellung einer integrationsfähigen Plattform sichergestellt werden, dass für die Produkte des Konsortiums eine lauffähige Komplettlösung existiert.

Diese Auswahl sollte nach einem agilen Ansatz in einem Umsetzungsprojekt vor Beginn der eigentlichen Softwareentwicklung erfolgen, damit im Entwicklungsprozess bereits auf etwaige Erfordernisse einzelner Plattformkomponenten Rücksicht genommen werden kann.

Vor dem Hintergrund, dass die Produkte des Konsortiums flexibel an die vorhandenen Systeme der Netzbetreiber ankoppelbar sein müssen (mit oder ohne Plattform, siehe Abbildung 3-2 bis 3-4), wird eine SOA als die geeignetste Form der Architektur für das geplante Vorhaben empfohlen. Als Komponenten der SOA sollten unter anderem aus Kostengründen möglichst Open-Source-Produkte gewählt werden, die bereits über mehrere Jahre in Open-Source-Projekten eingesetzt werden und sich auf eine intakte Entwickler-Community abstützen können.

Für die Entwicklung der OSS stehen mehrere Methoden zur Verfügung. Diese Entwicklungsmethode muss nicht zwingend fest vorgegeben werden. Das Entwicklungsteam sollte die Freiheit haben innerhalb des vom Konsortium vorgegebenen Rahmens, die am besten geeignetste Methode auszuwählen und bei Bedarf auch zu wechseln. Bei späteren Beiträgen zum Projekt (von Einzelpersonen oder Unternehmen) kann die Art der Entwicklungsmethode ohnehin nicht vorgeschrieben werden. Hier zählt das Ergebnis.

Mit hoher Wahrscheinlichkeit wird aber die Anwendung einer agilen Methode oder einer aus mehreren Methoden entstandene Mischform zum Einsatz kommen, da diese den Anspruch aller Beteiligten an einen transparenten Entwicklungsprozess am besten erfüllt. Softwareentwicklung auf der Basis von agilen Methoden verbreitet sich immer mehr und ist quasi Standard im Open-Source-Bereich.

Kapitel 4

Rahmenbedingungen für den Geschäftsbetrieb

INHALTSVERZEICHNIS

4 Rahmenbedingungen für den Geschäftsbetrieb

4.1 Fakten

4.1.1 Einleitung

Da es sich bei den Inhalten dieses Kapitels in weiten Teilen auch um juristische Fragestellungen handelt, wurde bei der Erstellung konsequent die juristische Beratung, einbezogen. Die dargestellten Ergebnisse und Empfehlungen sind mit ihr abgestimmt.

Nachdem sowohl der Bedarf an einer alternativen Vorgehensweise zur Softwarebereitstellung (Kapitel 2) als auch die technischen Voraussetzungen (Kapitel 3) als gegeben angenommen werden können, müssen im Weiteren die Rahmenbedingungen für den Geschäftsbetrieb eines Konsortiums definiert werden. Der Begriff "Konsortium" wird synonym für den noch näher zu beschreibenden Verbund (Rechtsform) der beteiligten Unternehmen verwendet.

4.1.2 Rollen und Aufgaben in der konsortialen Softwareentwicklung

4.1.2.1 Übersicht

Bevor im weiteren Verlauf dieser Studie die Umsetzungsmöglichkeiten analysiert und bewertet werden, sollen an dieser Stelle die dafür notwendigen Voraussetzungen und Annahmen beschrieben werden. Dazu werden zunächst das grundsätzliche Selbst- und Rollenverständnis des Konsortiums beleuchtet sowie ein grundsätzliches Rollenmodell als Basis für ein Ökosystem beschrieben. Im Anschluss wird aufbauend auf eine allgemeine Projektdefinition durch eine Darstellung der Abwicklung von Entwicklungsprojekten die Rollen- und Aufgabenverteilung in einem solchen Ökosystem weiter detailliert.

4.1.2.2 Selbst- und Rollenverständnis

Eine wesentliche Grundlage sind das Selbst- und Rollenverständnis des Konsortiums und dessen Mitglieder. Die konsortiale Softwareentwicklung hat zwei Komponenten:

- die Bündelung der Interessen und Nutzung des daraus resultierenden Markteinflusses
- die Entwicklung und Nutzung von Software auf der Basis von Open Source (OSS).

Der zuerst genannte Aspekt ist bezogen auf die Energiewirtschaft nicht neu und wird in vielen Bereichen praktiziert (Einkaufsgemeinschaften).

Die wesentliche Neuerung in dieser Branche entsteht durch die Kombination mit dem Open-Source-Gedanken. Die Open-Source-Bewegung basiert aus der Historie heraus auf der gemeinschaftlichen, internetgestützten Entwicklung von Software, die für jedermann offen zugänglich ist. Der Antrieb zur Entwicklung bzw. Weiterentwicklung kann unterschiedlich motiviert sein, es erfolgt aber keine direkte Vergütung für die Nutzung der Software. In diesen Konstrukten wurde und wird Software oft durch eine global verteilte Entwicklergemeinschaft ("Community") erstellt. Ein Beispiel dafür ist das Betriebssystem Linux.

Das ursprüngliche mit der Open-Source-Philosophie verbundene Modell der offenen Entwicklergemeinschaften ist jedoch für privatwirtschaftliche Unternehmen mit einem speziellen Anwendungshintergrund nicht geeignet, da hier belastbare Strukturen und eine funktionierende festgelegte Steuerung für die Softwareentwicklung fehlen.

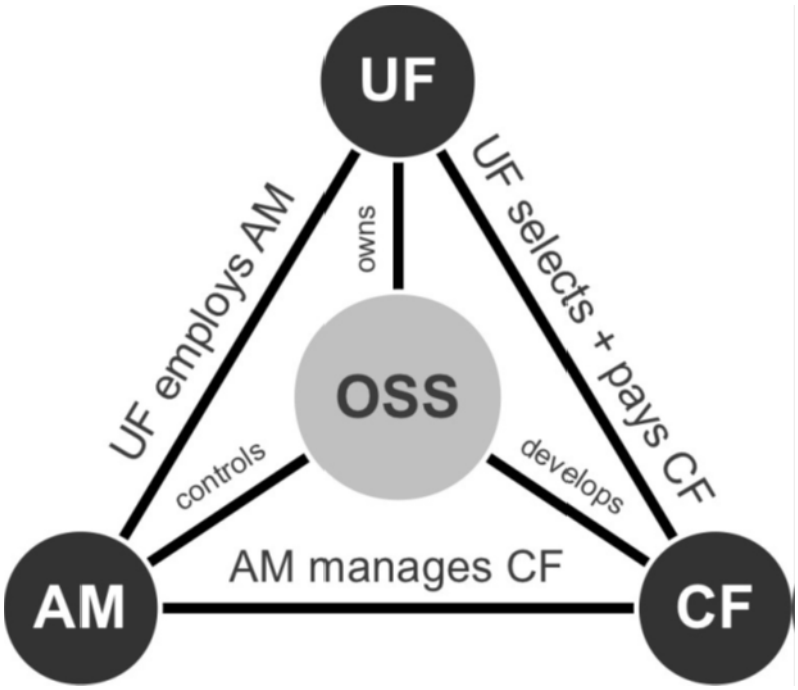
Es muss daher gelingen, eine Form der Zusammenarbeit zu finden, die die oben genannten Nachteile überwindet, aber auf der anderen Seite trotzdem die Vorteile des Open-Source-Gedankens (Flexibilität, Unabhängigkeit, Kostenreduzierung, Sicherheit) erschließt.

Das Konsortium ist als Initiator von Entwicklungsvorhaben in der Auftraggeberrolle und wird deshalb auch die Steuerung des Softwareentwicklungsprozesses für sich beanspruchen (müssen). Die klassische (1:1) Auftraggeber-Auftragnehmer- bzw. Kunden-Lieferanten-Beziehung wird dabei mittelfristig aber in den Hintergrund geraten und zwar in dem Maße, in dem sich ein funktionierendes Ökosystem etabliert (vergleiche auch Kapitel 2).

Im Vergleich zum klassischen Projektmanagement liegt die Herausforderung in der Anwendung relativ neuer Entwicklungsmethoden und -werkzeuge sowie in der Komplexität, ein größeres, sich in gewissen Teilen auch ständig veränderndes Entwicklungsteam effizient zu steuern.

4.1.2.3 Rollenmodell

Das Zusammenspiel der wichtigsten Stakeholdergruppen im konsortialen Softwareentwicklungsprozess ist in der Abbildung 4-1 dargestellt.



Legende:
UF (user foundation) = Organisation der Anwenderunternehmen (Konsortium)
CF (consulting firms) = Anbieterunternehmen (Hersteller, Berater)
AM (alliance mgmt) = Allianzmanagement (Projektkontrolle, - abnahme)

Abb. 4-1: Zusammenspiel der wichtigsten Stakeholdergruppen im konsortialen Softwareentwicklungsprozess [4-1]

Die Abbildung geht von der Situation aus, in welcher eine die Softwareanwender repräsentierende Organisation (user foundation UF - hier das Konsortium im eingangs beschriebenen Sinne) die Entwicklung von Open-Source-Software betreibt und finanziert.

Das Konsortium entwickelt die Software in diesem Fall nicht selbst, sondern gibt die Entwicklung als Auftrag an einen oder mehrere Anbieterunternehmen (Softwarehersteller oder -beratungsunternehmen (consulting firms CF)) heraus. Um sicherzustellen, dass qualitativ angemessene Software entwickelt wird, führt eine weitere Partei das sogenannte Allianzmanagement (AM), die Projektkontrolle und -abnahme durch. Das Allianzmanagement kann von den Mitgliedsunternehmen, externen Dienstleistern oder - je nach Größe des Konsortiums - durch eigenes Personal des Konsortiums durchgeführt werden.

Das Ergebnis dieser Konstellation ist Software, welche Eigentum des Konsortiums ist und den Mitgliedern sowie der Allgemeinheit als Open-Source-Software unter definierten Lizenzbedingungen zur Verfügung gestellt wird (siehe auch Open-Source-Definition der OSI [4-2]).

In dem dargestellten Modell spielt das Konsortium eher die Rolle des klassischen Auftraggebers. Dies scheint zumindest am Anfang der Geschäftstätigkeit nach Einschätzung der Teilnehmer an dieser Studie auch die momentanen realistischen Möglichkeiten widerzuspiegeln. Es ist jedoch auch möglich, dass das Konsortium bestimmte Leistungen selbst übernimmt durch

- Beistellung von Ressourcen der Anwenderunternehmen
- eigenes Personal.

Diese Leistungen können sich sowohl auf das Allianzmanagement als auch auf die Entwicklungskapazitäten beziehen.

Im Falle der Ressourcenbeistellung ist neben einer Vergütungsregelung im Rahmen der Projektfinanzierung auch die Möglichkeit denkbar, dass diese Leistungen unentgeltlich zur Verfügung gestellt werden. Diese Variante entspricht sehr stark dem ursprünglichen Open-Source-Gedanken, wo die Parteien mit dem höchsten Interesse am Fortschritt der Softwareentwicklung auch den größten Beitrag leisten. Es ist grundsätzlich nicht auszuschließen, dass Leistungen unentgeltlich erbracht werden. Dies bezieht sich vor allem auch auf die am Ökosystem beteiligten Dienstleister, die sich durch eine Beteiligung am Softwareentwicklungsprozess einen Wettbewerbsvorteil für die nachfolgende unternehmensspezifische Implementierung erhoffen können.

Die vom Konsortium entwickelte Software deckt den gemeinsamen Nenner der Bedürfnisse der Anwenderunternehmen ab. Dabei wird man in der Regel einen möglichst großen Gemeinsamkeitsgrad anstreben. Trotzdem wird eine Anpassung, beispielsweise an spezielle Bedürfnisse des jeweiligen Anwenders oder für spezielle Konnektoren zu anderen IT-Systemen, notwendig sein. Ist sie einfach oder verfügt das Anwenderunternehmen über die entsprechende Kompetenz, kann sie diese Anpassung selbst vornehmen. In der vorliegenden Situation wird ein Anwenderunternehmen aber in der Regel einen Dienstleister wie die oben erwähnten Anbieterunternehmen dafür bezahlen, die Open-Source-Software an die spezifischen Bedürfnisse des Unternehmens anzupassen.

Nach der Anpassung muss die Software betrieben werden. Dies kann von der anwendereigenen IT getätigt werden oder - je nach Komplexität und Möglichkeiten - von externen Dienstleistern, die die erforderlichen Wartungs- und Supportleistungen erbringen können.

4.1.2.4 Ökosystem

Das durch das Rollenmodell beschriebene Ökosystem ist nur lebensfähig, wenn einerseits hinreichend große und zahlreiche Softwareentwicklungsprojekte realisiert werden, andererseits auch geeignete Dienstleister zur Verfügung stehen. Sowohl für das Allianzmanagement als auch den eigentlichen Entwicklungsprozess sowie die genannten Implementierungs- und Supportleistungen ist ein hohes Maß an Prozessverständnis für die vorhandenen Anwendungen erforderlich. Vor diesem Hintergrund kommen für Entwicklungsleistungen natürlich auch die bereits heute im Bereich der EVU-Anwendungen tätigen Unternehmen (z. B. Netzleittechnikanbieter) in Frage. Es wird von entscheidender Bedeutung sein, diese Ressourcen zu erschließen, da der Einsatz branchenfremder Dienstleister zwar immer möglich sein muss, aber die Kommunikation im Projekt aufgrund des fehlenden Domänenwissens deutlich erschweren würde. In diesem Zusammenhang ist zu erwähnen, dass eine umfassende Durchsetzung des hier beschriebenen Modells zur konsortialen Softwareentwicklung eine tiefgreifende Veränderung der Marktbedingungen für die etablierten Anbieter bedeutet.

In einem funktionierenden Ökosystem wird das Konsortium die Aufträge zur Entwicklung von gemeinschaftlicher Software im Laufe der Zeit möglichst an ein breites Spektrum von Anbieterunternehmen vergeben. Gleichzeitig ist aus Anwendersicht die Durchgängigkeit der Dienstleistung (Beiträge zur gemeinschaftlichen Software, Erweiterung um anwenderspezifische Funktionalität, Unterstützung im Betrieb) wünschenswert. Es sollte hier vermieden werden, dass dies wieder alles auf einen Dienstleister zurückfällt. Wichtig ist, durch entsprechende Projektkontrolle sicherzustellen, dass sich keine Lock-in-Effekte ergeben.

Im Grundsatz wirkt natürlich der Einsatz von OSS durch deren freie Zugänglichkeit bereits dem Lock-in-Effekt entgegen. Solange von den Endanwendern ein ausreichend großer und profitabler Markt für Dienstleistungen im Umfeld der konsortialen Software aufgespannt wird, werden sich auch bei einer hohen Komplexität der Software in ausreichender Anzahl Dienstleister finden, die auch bereit sind, in eine Weiterentwicklung oder unternehmensspezifische Anpassung einzusteigen.

Eine besondere Rolle kommt im Umfeld der Entwicklung von OSS akademischen Einrichtungen zu. Die Partnerschaft mit Hochschulen und Instituten beinhaltet mehrere Aspekte. Zum einen stellt die Verwendung der Software im wissenschaftlichen Umfeld einen enormen Innovationsfaktor dar, durch den das Konsortium Zugang zu neuesten wissenschaftlichen Erkenntnissen erhält. Sofern sich nicht bereits aus der Arbeit der Studenten und Institute Beiträge (*Contributions*) zu einem Projekt-Repository ergeben, sind jedoch neue Impulse für die Weiterentwicklung der Software zu erwarten.

Zum anderen hat die Verwendung der Software im universitären Umfeld zur Folge, dass eine größere Zahl an jungen Menschen hier bereits Vorkenntnisse über die Software erwerben kann und sich damit für eine spätere Anstellung bei einem Mitgliedsunternehmen oder Dienstleister qualifiziert.

Ein dritter Aspekt ist die gemeinsame (Weiter-)Entwicklung von Software im Rahmen von geförderten Forschungsprojekten, wodurch die Entwicklungskosten für die Software weiter sinken können.

4.1.2.5 Organisation und Abwicklung von Entwicklungsprojekten

Auch für die Durchführung von konsortialen Entwicklungsvorhaben im Umfeld der KSE wird in dieser Studie der traditionelle Begriff *Projekt* verwendet. Ein Projekt ist allgemein durch eine Zielbeschreibung definiert. Diese Beschreibung enthält sicher die wesentlichen funktionalen Inhalte, aber keine konkreten Anforderungsbeschreibungen. Zur Zielbeschreibung gehört auch eine Gewichtung der projektbestimmenden Größen

- Kosten
- Termine
- funktionale Qualität.

Also muss es für die Durchführung von Projekten im Konsortium Regularien geben, in denen hierzu Aussagen getroffenen werden (Antrags- und Entscheidungsverfahren usw.). Diese Regularien werden bestimmt durch einen allgemeinen Rahmen, der im Wesentlichen durch Vereinbarungen, wie z. B. die Satzung und/oder eine Geschäftsordnung, definiert ist. Darüber hinaus müssen aber auch weitergehende projektbezogene Festlegungen getroffen werden.

Das Konsortium kann durchaus mehrere Projekte parallel führen. Jedes Projekt kann mehrere Unterprojekte haben. Jedes Projekt bzw. Unterprojekt kann sich softwaretechnisch in mehrere Komponenten gliedern.

Zum weiteren Verständnis wird nachfolgend die Abwicklung von Entwicklungsprojekten skizziert. Nach der Projektgenehmigung, in der auch das Finanzierungskonzept festgelegt wird, sind die Anforderungen zu spezifizieren. Hierfür wird die Leitungsebene ein Gremium (Requirement-Team) benennen. Dieses Team ist nicht fest installiert, sondern setzt sich fallbezogen aus wechselnden Vertretern der Mitgliedsunternehmen und bei Bedarf von Beratern zusammen.

Die Anforderungsspezifikation kann ganz oder teilweise Grundlage für eine "klassische" Ausschreibung sein, in deren Ergebnis Software zu vereinbarten Konditionen (kommerzielle Bedingungen usw.) geliefert wird. Die Kontrolle der Einhaltung dieser Konditionen und die Überprüfung der geforderten Qualität ist Aufgabe des Allianzmanagements.

Gegenüber dieser klassischen Vorgehensweise bietet die oben beschriebene gemeinschaftliche Softwareentwicklung eine Reihe von Vorteilen (Transparenz, Flexibilität usw., siehe auch Kapitel 3 und 5). Für die Umsetzung ist im Rahmen einer Projektinitialisierung ein Kernteam zu installieren, das sich aus folgenden Parteien zusammensetzen kann:

- Projektmanager aus dem Allianzmanagement
- beauftragte Softwareentwickler von entsprechenden Anbieterunternehmen
- unentgeltlich (freiwillig) tätige Softwareentwickler von entsprechenden Anbieterunternehmen
- Berater, Hochschulen und Forschungsinstitute.

Die Mitglieder des Kernteams werden im Rahmen der Projektinitialisierung (z. B. durch die Leitungsebene des Konsortiums oder einen speziell dafür vorgesehenen Entwicklungsleiter) bestimmt. Die Aufgabe des Kernteams ist es vorrangig, die Qualitätssicherung (Fehlerfreiheit, Übereinstimmung mit den Projektzielen, Erfüllung der funktionalen Anforderungen usw.) der entwickelten Software durchzuführen und den Code für die Aufnahme ins Projekt-Repository freizugeben. Die letztliche Verantwortung für die Aufnahme von Code in das Projekt-Repository kommt den in der "Open-Source-Terminologie" als *Committer* bezeichneten Personen zu, die häufig auch Mitglieder des Kernteams sind.

Eine besondere Rolle im Rahmen der Qualitätssicherung kommt dem sogenannten *Maintainer* (Instandhalter, Erhalter, Pfleger) zu. Der Maintainer ist der zuständige Moderator bzw. Ansprechpartner eines Projektes, der das Projekt betreut, ordnet, Versionen pflegt, Software-Pakete zusammenstellt und sich um die Bearbeitung von Programmfehlern kümmert. Er rekrutiert sich zumeist aus dem Kreis der Committer. Die Wahrnehmung der Rolle des Maintainers für ein oder mehrere Projekte kann aber auch quasi dienstleistend als Funktion des Konsortiums angeboten werden.

Die Zusammensetzung des Kernteams kann sich im Laufe des Projektes ändern, wobei der dominante Einfluss der Anwender einschließlich des Allianzmanagements immer gewahrt bleiben muss.

Die Entwicklung der Software selbst erfolgt durch ein Entwicklungsteam, dessen Mitglieder dafür beauftragt wurden oder bei anwachsendem Ökosystem freiwillig und unentgeltlich Beiträge zu den Projekten leisten. Auch Kernteammitglieder können diese Rolle einnehmen. Für die Realisierung der einzelnen Softwarekomponenten sind in Kapitel 3 verschiedene Methoden beschrieben und bewertet worden. Besonders im zuletzt beschriebenen Fall der gemeinschaftlichen Entwicklung haben sich iterative Verfahren (agile Methoden) etabliert, die darauf beruhen, dass aufeinander aufbauende "Releases" in festgelegten Zeitzyklen fertiggestellt und den Anwendern übergeben werden. Diese Verfahren haben u. a. den Vorteil, dass die Anwender auch während der Projektlaufzeit noch Einfluss auf die funktionalen Eigenschaften nehmen können. Unter dieser Betrachtungsweise wird das Projekt einem kontinuierlichen, iterativen Kreislauf von

- Anforderungsspezifikation
- Softwareentwicklung
- Qualitätskontrolle
- Abnahme

unterliegen (siehe auch Abbildung 3-10).

Eine solche Vorgehensweise schafft einerseits ein hohes Maß an Transparenz im Entwicklungsprozess, birgt aber auch andererseits die Gefahr, dass Projekte besonders unter den Aspekten Kosten und Termine "aus dem Ruder" laufen. Es ist daher eine der wesentlichen Aufgaben der Projekt- und Entwicklungsleiter (gemäß Abbildung 4-1 gleichzusetzen mit Allianzmanager), hier die Einhaltung der Projektziele sicherzustellen.

4.1.2.6 Wertschöpfungskette

Ein ähnlicher, wenn auch dem im vorigen Abschnitt beschriebenen projektbezogenen Entwicklungsprozess übergeordneter Kreislauf ergibt sich für die innerhalb des Konsortiums angesiedelten Wertschöpfungsstufen von einem Softwarerelease zum nächsten (siehe Abbildung 4-2).

An dieser Darstellung wird auch die im Abschnitt 4.1.2.3 beschriebene Leistungsgrenze zwischen dem Entwicklungsprozess und der Bereitstellung von Software durch das Konsortium einerseits sowie den IT-Prozessen bei den Anwendern andererseits deutlich.

Abb. 4-2: Wertschöpfungskette im Lebenszyklus von Software

4.1.3 Grundsätze für die Ausgestaltung des Konsortiums

Vor dem bis hierher dargestellten Hintergrund wurden im Rahmen dieser Studie im Projekt eine Reihe von Annahmen abgeleitet. Diese Annahmen beziehen sich auf eine mittelfristige Betrachtung, das heißt sie gelten für einen Zielzustand im Zeitraum von ca. fünf Jahren:

- Das Konsortium verfolgt primär das Ziel, für die beteiligten Versorgungsunternehmen eine Optimierung des Einsatzes von IT-Werkzeugen der Netzbetreiber hinsichtlich ihrer Wirtschaftlichkeit, Qualität und Realisierungszeiten zu erreichen. Dabei steht im Vordergrund, die in Kapitel 2 im Rahmen der Bestandsaufnahme identifizierten Nachteile zu überwinden.
- Software wird grundsätzlich zur Nutzung durch die Mitglieder und nicht zur Vermarktung bzw. zur Erzielung von Gewinnen entwickelt. Der wirtschaftliche Vorteil für die Mitgliedsunternehmen muss u. a. darin liegen, dass für sie die Kosten der Softwareerstellung und für die Zahlung von Lizenzgebühren erheblich sinken (siehe auch Kapitel 5).
- Das Konsortium reduziert sich nicht auf einen "losen Verbund" (Kooperation) von Unternehmen, sondern hat eine eigene Rechts- und Organisationsform, die der Kontrolle der Mitglieder unterliegt. Die Gründe dafür liegen im Wesentlichen darin, dass:
  - Kooperationen wegen ihrer Unverbindlichkeit häufig scheitern
  - das Konsortium in einem definierten Rahmen eigenverantwortlich handeln muss, um die Anwenderseite in einem Ökosystem zielgerecht vertreten zu können; zu diesem Handlungsrahmen gehört u. a. auch die Möglichkeit, eigenes Personal zu beschäftigen
  - das Eigentum an den Urheberrechten (siehe hierzu Ausführungen über Lizenzen) unabhängig von einzelnen Mitgliedern gehalten werden kann
  - die Außenwirkung im Rahmen geschäftlicher Aktivitäten deutlich steigt.
- Das Konsortium ermöglicht neben den Anwendern (Versorgungsunternehmen) auch die Mitgliedschaft von Dienstleistern im Umfeld des Ökosystems (wie z. B. Softwarefirmen und Berater), von akademischen Institutionen sowie ggf. Behörden und Verbänden. Der Einfluss unterschiedlicher Mitgliedsgruppen im Konsortium wird in einer rechtsformspezifischen Ausführung (z. B. Satzung) geregelt.
- Das Konsortium ist grundsätzlich auf Wachstum ausgerichtet, da hierdurch anfallende Aufwendungen stärker verteilt werden können. Eine Fokussierung auf andere Branchen neben der Energiewirtschaft ist aktuell nicht vorgesehen. Es besteht jedoch die Möglichkeit, dass die vom Konsortium entwickelte Open-Source-Software in anderen Branchen auf Interesse stößt und plötzlich auch "branchenfremde" Unternehmen die Mitgliedschaft im Konsortium beantragen.

- Die Mitgliedschaft ist nicht an eine bestimmte Nationalität gebunden. Einen Zwang zur Internationalität (z. B. durch die Rechtsform) soll es aber nicht geben.
- Es gibt hinsichtlich der Anwendungen keine Restriktionen. Das Konsortium hat aber primär nicht das Ziel, eingeführte Standard-Software (z. B. SCADA-Systeme im Sinne von leittechnischen Grundfunktionen) zu ersetzen, sondern vielmehr neue und ergänzende Funktionalitäten zu entwickeln. Hier wird durch den aktuellen Umbau der Energieversorgung mit einem entsprechenden Bedarf gerechnet. Diese Strategie kann aber durchaus zu einer Aushöhlung vorhandener Systeme im Bereich höherwertiger Funktionen führen. Eine solche Entwicklung hat natürlich eine entsprechende Wirkung auf den existierenden Markt.
- Das Konsortium entscheidet über Softwareentwicklungsaktivitäten, beauftragt, kontrolliert und bezahlt die notwendigen Dienstleistungen. Dabei gilt eine klare projektbezogene Abgrenzung zu unternehmensspezifischen Softwareanpassungen und Implementierungsaufwendungen, die außerhalb des Konsortiums abgewickelt werden.
- Sofern die beteiligten Versorgungsunternehmen (Anwender) über eigene Entwicklungskapazitäten für die Erstellung von Software verfügen, können diese natürlich ebenfalls in den Entwicklungsprozess eingebunden werden.

In den folgenden Abschnitten werden auf der Basis der oben genannten Annahmen die wesentlichen "Spielregeln" des Konsortiums analysiert und bewertet. Dabei wird zwischen

- Rechtsform des Konsortiums
- Satzung (synonym für verbindliche Vereinbarungen innerhalb der gewählten Rechtsform)
- Lizenzen

unterschieden.

4.2 Analyse und Strukturierung

4.2.1 Rechtsform

Für das Konsortium kommt prinzipiell eine Reihe von Rechtsformen infrage, von denen aber einige aufgrund von grundsätzlichen Restriktionen von vornherein ausscheiden. Hierbei handelt es sich um:

- **GbR oder vertragliches Joint Venture** ist als loser Verbund nicht geeignet für die wirtschaftliche Tätigkeit in einem Umfang, in dem eine Haftungsabschirmung für die beteiligten Unternehmen erreicht werden muss.
- **Europäische Genossenschaft (SCE), Europäische Aktiengesellschaft (SE)**, da Internationalität auf europäischer Ebene erforderlich ist (Mitglieder müssen aus verschiedenen Ländern kommen).

Folgende Rechtsformen werden betrachtet:

- **Eingetragene Genossenschaft ("eG")**
- **Eingetragener Verein ("e.V.")**
- **Eingetragener Verein ("e.V.") mit "Tochter"-GmbH**
- **Stiftung**
- **Gesellschaft mit beschränkter Haftung ("GmbH")**
- **Aktiengesellschaft ("AG").**

Kein Unterschied wurde zwischen der Unternehmensgesellschaft (UG) und der GmbH gemacht, sie unterscheiden sich nur in der Mindestkapitalausstattung (Mindestkapital EUR 1,-- statt EUR 25.000,--).

In der Anlage 4-1 sind die wesentlichen Merkmale der Rechtsformen zusammengestellt. Eine Bewertung erfolgt im Abschnitt 4.3.1.

Im Zusammenhang mit der Beurteilung der einzelnen Rechtsformen ist auch das Thema Haftung relevant. Das Konsortium kann immer in eine Haftungssituation geraten, da es im Rahmen der geschäftlichen Aktivitäten auch Verbindlichkeiten eingeht.

Fokussiert man die Haftungsfrage auf die Entwicklung und Weitergabe von OSS, so stellt sich die Situation wie folgt dar:

- Nach deutschem Recht ist die Weitergabe (mit oder ohne Entgelt) von Software grundsätzlich nie frei von Haftungsverpflichtungen. Im Fall von OSS allerdings muss nur gehaftet werden, wenn dem Urheber Vorsatz oder grobe Fahrlässigkeit nachgewiesen werden kann, §§ 521, 523, 525 BGB.
- Es ist nicht ausgeschlossen, dass das Konsortium gegenüber den Mitgliedern haftet. Allerdings ist dies eine Frage, wie die Parteien miteinander umgehen wollen. So könnten hier bestimmte Formen der Haftung per Satzung ausgeschlossen werden.
- Es ist rechtlich nicht eindeutig definiert, ob Software unter das Produkthaftungsgesetz fällt. In jedem Fall wäre hier die Haftungshöhe gedeckelt und somit versicherbar. Nach Aussage des Rechtsberaters sind sich die Juristen in ihrer Auslegung und wahrscheinlich in der aktuellen Rechtsprechung nicht einig.

Dienstleister, die OSS-Komponenten in den Unternehmen implementieren, können per vertraglicher Regelung auch zur Haftung verpflichtet werden. Dies kann sich auch auf eine ordnungsgemäße Funktion der vom Konsortium übernommenen Software beziehen, sodass der Dienstleister entsprechende Tests durchführen muss.

4.2.2 Satzung

Wie bereits erwähnt, wird der Begriff Satzung synonym für die verbindlichen Vereinbarungen in der gewählten Rechtsform verwendet. Das bedeutet natürlich, dass eine endgültige Formulierung einer Satzung erst nach Festlegung dieser Rechtsform für das Konsortium erfolgen kann.

Unabhängig von der Festlegung auf eine Rechtsform wurde davon ausgegangen, dass eine endgültige, ausformulierte Satzung nicht Bestandteil dieser Machbarkeitsstudie ist. Vielmehr ist die Ausformulierung einer Gründungsphase vorbehalten, an der auch der juristische Sachverstand aus den Gründungsunternehmen beteiligt werden muss. In diesem Zuge muss auch festgelegt werden, welche zusätzlichen Dokumente erstellt werden sollen (u. U. spezifisch für die gewählte Rechtsform), z. B. Geschäftsordnungen und Vereinbarungen der Gesellschafter/Mitglieder zu speziellen Themen. Besonders am Anfang ist es empfehlenswert, eine eher schlanke Satzung anzustreben, da so der administrative Aufwand bei Anpassungen der betreffenden Dokumente an die Anfangserfahrungen verringert werden kann. Wichtig ist jedoch, dass die Verbindlichkeit von zusätzlichen Dokumenten aus der Satzung hervorgeht.

Es ist im Rahmen dieser Studie trotzdem erforderlich, folgende wesentliche Punkte einer Satzung zu benennen und deren Inhalte weitgehend zu umreißen:

- Name und Sitz
- Geschäftsjahr
- Ziel, Zweck, Gegenstand
- Mitgliedschaft
- Organe
- Geschäftsanteile, Beiträge, Ein- und Rückzahlungen
- Auflösung
- Bekanntmachungen
- Gerichtsstand.

Nachfolgend werden die vorrangig interessierenden Punkte angesprochen und kommentiert. Im Rahmen dieser Kommentierung wird aus Gründen der besseren Verständlichkeit - soweit möglich - bereits eine Empfehlung ausgesprochen, sodass im Abschnitt 4.3 eine detaillierte Bewertung nicht mehr erfolgen muss.

In einigen Fällen werden oben aufgeführte Punkte nicht weiter kommentiert. Die Inhalte sind hier unproblematisch und können zu einem späteren Zeitpunkt festgelegt werden.

Die nachfolgenden Aussagen orientieren sich relativ stark an den Rechtsformen Verein und Genossenschaft, da sich diese in der Bewertung im Abschnitt 4.3 als Vorzugsvarianten herauskristallisieren. Sollte trotzdem eine andere Rechtsform gewählt werden, müssen die genannten Punkte in angepasster Form umgesetzt werden.

Zum besseren Verständnis werden in Abbildung 4-3 noch einmal die im Abschnitt 4.1.2 beschriebenen für das Konsortium relevanten Rollen dargestellt.

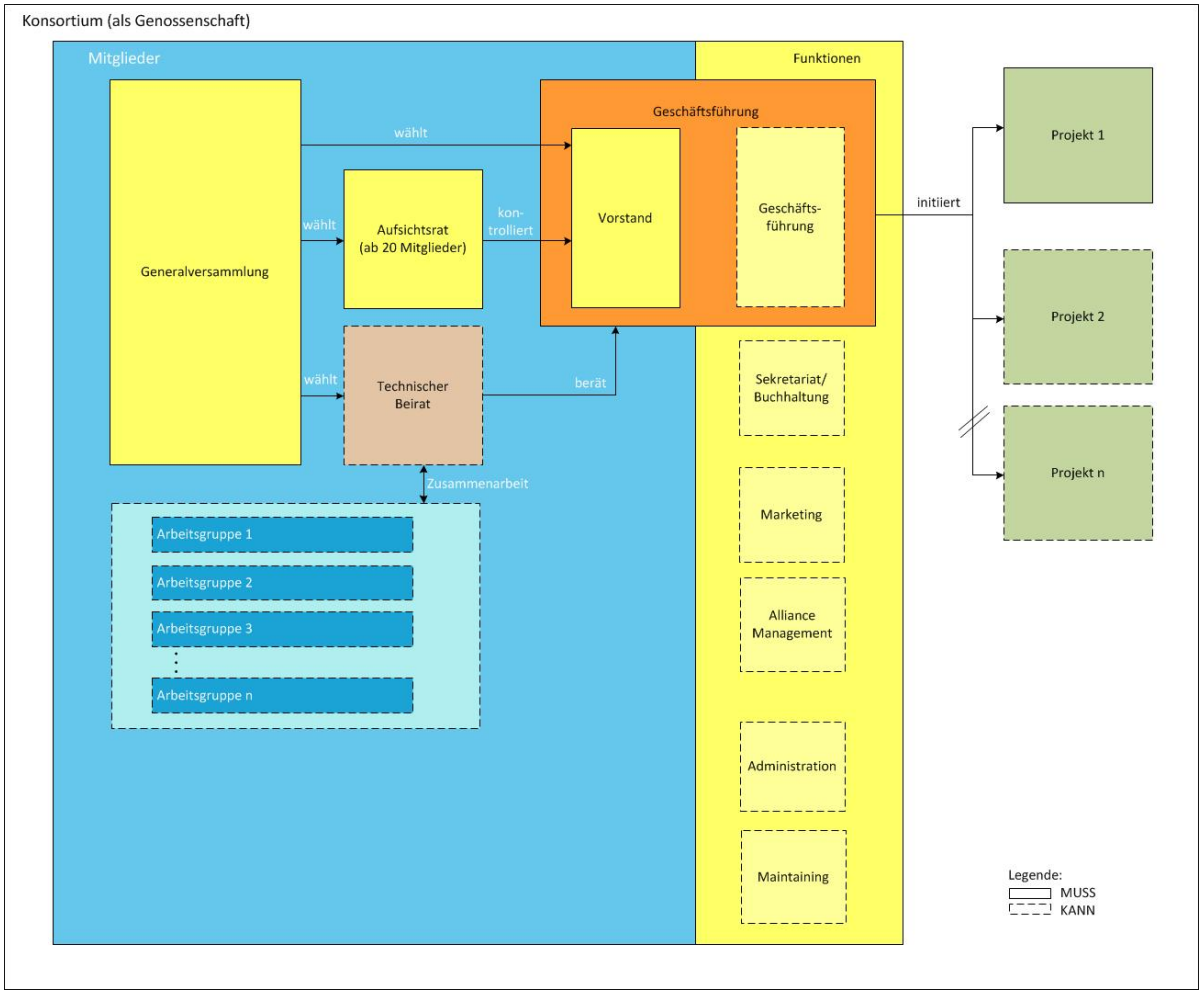


Abb. 4-3: Rollen im Konsortium

Geschäftsjahr

- Üblicherweise ist dies das Kalenderjahr (vereinfacht insbesondere die steuerrechtlichen Themen, das erste Jahr kann als Rumpfgeschäftsjahr ausgestaltet werden).

Ziel, Zweck, Gegenstand

- Zusammenführung von Unternehmen und Personen mit dem Ziel, die Entwicklung, Beschaffung und Nutzung von Software für das Betreiben von Netzen zu optimieren
- Dies wird erreicht durch:
  - die gemeinschaftliche Entwicklung der Software auf der Basis von Open Source, die unter von der Open Source Initiative (OSI) anerkannten Lizenzen allen Marktteilnehmern zur Verfügung gestellt wird
  - Aufbau und Betrieb einer geeigneten Kommunikationsinfrastruktur (Projekt-Hosting-Plattform, siehe auch Kapitel 3).
- Geschäfte mit Nichtmitgliedern sind zulässig.

Mitgliedschaft

- Natürliche, juristische Personen sowie Personengesellschaften können Mitglieder werden.
- Es ist ein Aufnahmeantrag mit festgelegten Angaben erforderlich.
- Entscheidung über Aufnahme durch das Leitungsgremium (im Weiteren beispielhaft als Vorstand bezeichnet) in einfacher Mehrheit. Idealerweise sollten Kriterien für die Entscheidung festgelegt sein, um somit kartellrechtliche Aspekte zu entschärfen. Es sollte möglichst geringe Hürden bzw. keine Hürden geben, die als gegen einzelne Unternehmen gerichtet verstanden werden können. Ablehnung mit schriftlicher Begründung, Widerspruchsrecht mit abschließender Entscheidung durch die Mitgliederversammlung.
- Beendigung der Mitgliedschaft durch:
  - Kündigung/Austritt des Mitgliedes: Eine längere Kündigungsfrist ist aus Sicht des Konsortiums empfehlenswert, z. B. ein Jahr zum Ende des Geschäftsjahres, auf der anderen Seite wird dadurch für potenzielle neue Mitglieder eine größere Schwelle aufgebaut.
  - Tod eines persönlichen Mitgliedes, Insolvenz eines juristischen Mitgliedes
  - Ausschluss durch den Vorstand per Mehrheitsbeschluss mit Begründung und entsprechendem Widerspruchsverfahren

- Ausschlussgründe liegen im Wesentlichen darin, dass ein Mitglied seinen (finanziellen) Verpflichtungen nicht nachkommt.
- Regelungen über vorhandene Guthaben: Geleistete Beiträge werden nicht zurückgezahlt.

- Stimmrechte
  - Grundsatz: pro Mitglied eine Stimme
  - Abweichungen davon können sinnvoll sein, z. B. stärkeres Gewicht der Gründungsunternehmen. Dies kann aber alternativ auch erreicht werden, wenn immer (mindestens) ein Vorstandsmitglied aus der Gruppe der Gründungsunternehmen rekrutiert werden muss. Es wird empfohlen, die Stimmrechtsverteilung zumindest am Anfang nicht zu kompliziert auszuführen (z. B. durch Vielzahl von Mitgliederkategorien).
  - Auch Dienstleister können Mitglied werden. Hierfür sollte ein Status ohne Stimmrecht in Erwägung gezogen werden. Je nach Rechtsform muss hier ggf. ein Beirat (siehe unten) oder eine Kommission vorgesehen werden. Wichtig ist es, dass die nicht stimmberechtigten Mitglieder aber andere Privilegien, wie z. B. Sprachrecht in der Mitgliederversammlung oder Antragsrecht für Projekte, erhalten.

Organe des Konsortiums

- Zwingend erforderlich sind Vorstand und Mitgliederversammlung, bei einigen Rechtsformen ist (ggf. oberhalb einer bestimmten Mitgliederzahl) auch ein Aufsichtsgremium vorzusehen.
- Vorstand:
  - wird von der Mitgliederversammlung (bzw. Aufsichtsgremium) gewählt (siehe auch Variante unter "Stimmrechte")
  - Amtsperiode: Vorschlag ⇨ 2 Jahre
  - Anzahl: Vorschlag ⇨ 3, da hier immer eine Mehrheit zustande kommt
  - Vorstände arbeiten nebenamtlich. Wenn es das Geschäftsvolumen erfordert, kann eine hauptamtliche Tätigkeit notwendig werden (ggf. nur einer).
  - Regularien für die Beschlussfassung (schriftlich, telefonisch oder elektronisch)
  - Der Vorstand sollte weitreichende Kompetenzen erhalten, um den effizienten Betrieb des Konsortiums zu gewährleisten. Dazu gehört z. B. die Entscheidung über kleinere durchzuführende Projekte, Erteilung von Aufträgen an Dienstleister ggf. im Rahmen festgelegter Wertgrenzen.

- Aufsichtsgremium (nur bei Bedarf):
  - wird von der Mitgliederversammlung gewählt.
- Mitgliederversammlung:
  - Einberufung durch Vorstand (oder Aufsichtsgremium), mindestens einmal pro Geschäftsjahr
  - wesentliche Entscheidungsbefugnisse:
    - Wahl, Entlastung und Abberufung von Vorstand (ggf. Aufsichtsgremium)
    - Wahl von Rechnungsprüfern
    - Geschäfts- und Beitragsordnungen usw.
    - Wirtschaftspläne und Jahresabschlüsse
    - Auswahl und Entscheidung über durchzuführende Projekte
    - Gründung von Tochtergesellschaften, Beteiligung an anderen Unternehmen
    - Satzungsänderungen
    - Verlegung des Sitzes
    - Auflösung
  - formale Festlegungen für die Durchführung von Mitgliederversammlungen und Wahlen (müssen an dieser Stelle nicht ausgeführt werden).
- Technischer Beirat:
  - Kein zwingendes Organ der Genossenschaft; Er berät den Vorstand, den Aufsichtsrat und bisweilen auch die Mitgliederversammlung vor allem in technischen Fragen.
  - Die Mitglieder des Beirates werden von der Mitgliederversammlung des Konsortiums gewählt, die Öffnung des Beirates auch für Externe (Dienstleister, Hochschulen usw.) kann in der Satzung geregelt werden.

Geschäftsanteile, Beiträge, Ein- und Rückzahlungen

An dieser Stelle sollen einige grundsätzliche Bemerkungen zur finanziellen Seite der Geschäftstätigkeit des Konsortiums gemacht werden. Ein wesentlicher Punkt ist hierbei, wie die Kosten für

- den laufenden Betrieb
- die Durchführung von Projekten

gedeckt werden können. Bei einigen Rechtsformen müssen die Mitglieder Geschäftsanteile oder Kapitalbeteiligungen erwerben. Aus Gründen der Haftungsbeschränkung werden diese Beträge aber eher niedrig ausfallen, sodass hier keine Basis zur Finanzierung der oben genannten Kosten vorhanden ist.

Da sich das Konsortium auf der anderen Seite als Vereinigung ohne Vermarktungsinteressen positioniert, ist es notwendig, einen Umlagemechanismus vorzusehen. Bei Vereinen ist dieser Mechanismus per se durch die Beitragsstruktur bereits vorhanden, aber auch in anderen Rechtsformen, wie z. B. die Genossenschaft, ist dies durch eine ergänzende Entgeltvereinbarung möglich. Auf diese Weise entsteht ein Budget, das zur Deckung der oben genannten Kosten herangezogen werden kann.

Es kann angenommen werden, dass das Konsortium am Anfang relativ geringe Aufwendungen für den laufenden Betrieb haben wird. Interessanter ist die Finanzierung der anstehenden Projekte. Obwohl die Möglichkeit besteht, die Projektfinanzierung jeweils individuell zu regeln, wird empfohlen, hier zumindest teilweise eine Finanzierung aus dem oben genannten Budget vorzusehen. Das hat den Vorteil, dass alle Mitglieder zur Kostenbeteiligung herangezogen werden und das Phänomen der "Trittbrettfahrerei" etwas abgeschwächt wird.

Grundsätzlich sollte die Mitgliederversammlung jährlich über die Höhe der Einzahlungen in das Budget entscheiden. Sofern die Konsortiumsmitglieder nicht einen einheitlichen Jahresmitgliedsbeitrag entrichten, kann eine beitragsabhängige Mitgliederkategorisierung vorgenommen werden. Ein hierfür geeignetes Modell sollte einfach sein (z. B. drei Klassen in Abhängigkeit von bestimmten Netzgrößen oder nach einer von Unternehmen beim Beitritt selbst zu wählende Kategoriestufe (silber, gold, platin)).

Über die Verwendung des Budgets entscheidet der Vorstand im Rahmen der Projektgenehmigung.

Zur vollständigen Übersicht über die finanzielle Abwicklung eines Geschäftsbetriebes müssen abhängig von der Rechtsform noch folgende Punkte geregelt werden:

- Geschäftsguthaben
- Rücklagen
- Rückvergütung
- Jahresüberschuss und -fehlbetrag
- Nachschusspflicht (wird in der Regel ausgeschlossen).

4.2.3 Lizenzen

Grundkonzepte der Vergabe von Lizenzen

Wenn jemand eine Software entwickelt, ist das entstehende geistige Eigentum (engl. intellectual property, IP) urheberrechtlich geschützt - wie ein literarisches Werk oder ein Kunstwerk. Das deutsche Urheberrechtsgesetz enthält einen eigenen Abschnitt zum Rechtsschutz von Software (§ 69a ff. UrhG).

Das Urheberrecht verleiht dem Eigentümer eines Werkes bestimmte Rechte an seinem Werk und sieht rechtsverbindliche Einschränkungen für die Nutzung des Werkes durch Dritte vor.

Das Urheberrecht (engl. copyright) an einem Werk bedeutet, dass der Inhaber, d. h. in der Regel der ursprüngliche Urheber oder sein Arbeitgeber (hierzu gibt es im Gesetz eine gesetzliche Überleitung der Verfügungsrechte vom Arbeitnehmer an den Arbeitgeber), darüber entscheidet, wer das Werk kopieren, anpassen und verbreiten darf. Im Regelfall ist dazu nur der Inhaber berechtigt. Die Berechtigung zum Kopieren, Verändern oder Verbreiten eines Werkes kann Dritten vom Rechteinhaber erteilt werden. In diesem Zusammenhang wird typischerweise von einer "Lizenz" gesprochen. Eine solche Lizenz ist ein Vertrag zwischen einem Lizenzgeber (d. h. dem Urheber der Software) und einem Lizenznehmer (d. h. dem Anwender der Software, der sie anschließend gemäß den Lizenzbedingungen nutzen darf) [4-3].

Mit der Lizenz einer Software regelt der Lizenzgeber also die mit der Nutzung der Software verbundenen Rechte und Pflichten für die Lizenznehmer.

Herkömmlich werden Rechte an der Software so restriktiv wie möglich eingeräumt, also z. B. kein Recht zur Weitergabe der Software an Dritte und kein Recht zum Erstellen von Vervielfältigungen. Der Nutzer (Endanwender der Software) erhält die Software in der Regel nur in kompilierter Form (Object Code, keine Überlassung von Quellcode/Sourcecode). Bei Freeware geschieht dies unentgeltlich; bei kommerzieller Software dagegen i. d. R. gegen Zahlung von Lizenzgebühren [4-4].

Damit bei einer Insolvenz des Herstellers der Endanwender der Software seine Investition nicht verliert, werden oftmals sogenannte Hinterlegungsvereinbarungen (engl. Escrow Agreement) für den Quellcode zwischen dem Softwarehersteller und dem Besteller einer Software vereinbart. Dieser Vertrag regelt die Hinterlegung des Quellcodes bei einer zwischen den Parteien vereinbarten neutralen Hinterlegungsstelle [4-5]. Mit dieser Vorgehensweise werden übrigens auch die aktuellen Anforderungen an die IT-Sicherheit erfüllt, z. B. im BDEW-Whitepaper für Leitsysteme.

Open-Source-Software verfolgt demgegenüber einen anderen Ansatz - dem Nutzer werden weitergehende Rechte eingeräumt. Typischerweise werden in diesem Zusammenhang Lizenztexte verwendet, die von der Open-Source-Initiative (OSI) anerkannt sind. Diese Anerkennung erfolgt, wenn die zehn von der OSI an Open-Source-Software gestellten Grundsätze eingehalten sind (siehe auch www.opensource.org/docs/osd). Hauptmerkmale von Open-Source-Software sind, dass der Quellcode öffentlich zugänglich ist und frei kopiert, modifiziert und verändert wie unverändert weiterverbreitet werden darf [4-6].

Die freie Nutzbarkeit wird essenziell mit Hilfe der Offenlegung des Quellcodes und der Lizenzgebührenfreiheit erreicht. Diese Voraussetzungen ermöglichen die ungehinderte Weiterentwicklung und den Vertrieb seitens der Lizenznehmer [4-4].

Die OSS-Lizenzen unterscheiden sich im Wesentlichen anhand der Vorgaben des Urhebers bei Veränderung und Weiterentwicklung seines Quellcodes. In diesem Zusammenhang trifft man häufig auf eine sogenannte Copyleft-Klausel, die festschreibt, dass Bearbeitungen des Werkes nur dann erlaubt sind, wenn alle Änderungen ausschließlich unter den identischen oder im Wesentlichen gleichen Lizenzbedingungen weitergegeben werden. Das Copyleft soll somit verhindern, dass freie Werke zum Ausgangsmaterial proprietärer Inhalte werden. Je nach dem Grad, mit dem Werke, die ein anderes Werk enthalten, als abgeleitete Werke von der Lizenz betroffen sind, wird zwischen starkem und schwachem Copyleft unterschieden - wobei der Übergang fließend ist [4-7].

Die Lizenzen lassen sich hiernach grob in Lizenzen ohne Copyleft-Effekt, Lizenzen mit strengem Copyleft-Effekt (starkes Copyleft) und Lizenzen mit beschränktem Copyleft-Effekt (schwaches Copyleft) unterteilen.

Lizenzen ohne Copyleft-Effekt

Open-Source-Lizenzen ohne Copyleft-Effekt zeichnen sich dadurch aus, dass sie dem Lizenznehmer alle Freiheiten einer Open-Source-Lizenz einräumen und für Veränderungen der Software keine Bedingungen hinsichtlich des zu verwendenden Lizenztyps enthalten. Damit kann der Lizenznehmer veränderte Versionen der Software unter beliebigen Lizenzbedingungen weiterverbreiten, also auch in proprietäre Software überführen.

Die bekanntesten Lizenzen ohne Copyleft sind die verschiedenen Versionen der BSD-Lizenz (Berkeley Software Distribution), die Apache-2.0-Lizenz und die MIT-Lizenz (Massachusetts Institute of Technology).

Lizenzen mit strengem Copyleft-Effekt

Bei Lizenzen mit einem strengen Copyleft-Effekt wird der Lizenznehmer verpflichtet, von der ursprünglichen Software abgeleitete Werke ebenfalls nur unter den Bedingungen der Ursprungslizenz weiterzuverbreiten. Dies bedeutet, dass Fortentwicklungen eines freien Ur-Programms wiederum frei sind und frei bleiben. Man spricht beim strengen Copyleft deswegen auch von einem viralen Effekt.

Die bekanntesten Lizenzen mit strengen Copyleft sind die verschiedenen Versionen der GPL-Lizenz. Die GNU General Public Licence (GPL V. 2) ist die am häufigsten angewandte Copyleft-Lizenz.

Lizenzen mit schwachem Copyleft-Effekt

Diese Lizenzen gleichen den Lizenzen mit strengem Copyleft-Effekt insoweit, als sie ebenfalls einen Copyleft-Effekt haben, der aber nur eingeschränkt und sehr viel trennschärfer ist. Die Besonderheit dieses Lizenztyps liegt in der Möglichkeit, OSS-Programmbibliotheken von beliebig lizenzierten Softwareprodukten aus zu verlinken. Bei dieser Art der Integration, bei der die verschiedenen Komponenten miteinander kommunizieren, um Funktionen bereitzustellen, werden die verschiedenen Quellcodes nicht zusammengefügt. Somit wird jede Software gemäß den Bedingungen ihrer eigenen Lizenz genutzt und verbreitet.

Grundlage für die Trennung ist eine sehr formale Betrachtung - solange ein anderer Code in einer anderen Datei liegt und (nur) Verlinkungen bestehen, kann dieser neue Code unter einer anderen Lizenz stehen. Gleichzeitig gilt für Modifikationen des unter Copyleft stehenden Code das Copyleft-Erfordernis, also Weitergabe von Bearbeitungen dieses Code nur unter den Bedingungen der ursprünglichen Lizenz. In einem solchen Fall bestehen keine Kompatibilitätsprobleme zwischen den Lizenzen für den ursprünglichen Code und den neu in separaten Dateien hinzugefügtem Code.

Als Beispiel sind hier die Eclipse Public Licence (EPL), die Mozilla Public Licence (MPL) als auch die European Union Public Licence (EUPL) zu nennen. Mit Einschränkungen gehört auch die GNU Lesser General Public Licence (LGPL) in diese Kategorie.

Manche OSS-Lizenzen gestatten es außerdem, modifizierte Versionen der Original-Software unter einer anderen Lizenz zu verbreiten, so z. B. die EUPL. Unter der EUPL-stehende Software kann dann mit unter anderen OSS-Lizenzen stehender Software kombiniert werden, indem eine entsprechende Wahl getroffen wird. Bei der Auswahl einer Lizenz ist also darauf zu achten, mit welchen anderen Lizenzen diese kompatibel ist [4-8].

Doppellizenzierung

Eine weitere Option besteht darin, Softwareprodukte einer Doppellizenzierung zu unterstellen. Bei diesem Lizenzmodell kann der Eigentümer eine Software sowohl unter einer Open-Source-Lizenz als auch unter einer proprietären Lizenz anbieten (oder auch gleichzeitig unter zwei oder mehreren Open-Source-Lizenzen). Die proprietäre Lizenz gestattet eine kommerzielle Nutzung und kann dafür benötigte Rechte wie Support- und Softwarepflegeanspruch oder Produkthaftung enthalten, wohingegen beispielsweise die GPL eine freie Nutzung ermöglicht und fördert. Sobald eine Software aber unter die GPL gestellt wurde, ist die proprietäre Lizenzierung grundsätzlich nur durch den Urheber erlaubt. Wurde die Software inzwischen von Dritten verändert, darf diese Version nicht mehr proprietär lizenziert werden, es sei denn, dass der Veränderer dem ursprünglichen Eigentümer das Recht dazu überträgt. Ein bekanntes Beispiel für eine Doppellizenzierung ist MySQL, das unter GPL und einer proprietären Lizenz verfügbar ist [4-4].

4.3 Bewertung

4.3.1 Rechtsform

Zur Entscheidungsfindung können Kriterien definiert werden, mit denen eine systematische Bewertung durchgeführt werden kann. Zu diesem Zweck wurde eine Bewertungsmatrix erstellt, die als Anlage 4-2 beigefügt ist.

Grundlage der Bewertung ist ein Punkteschema. Dabei wird jedem Kriterium ein Gewicht im Bereich von 1 bis 5 gegeben. Danach erfolgt die rechtsformbezogene Bewertung der Erfüllung der jeweiligen Kriterien. Auch hierfür ist ein Punktebereich vorgesehen, in diesem Fall von 0 bis 5. Die Bewertungspunktzahl ergibt sich aus dem Produkt beider Punktwerte.

Dieses Schema ist natürlich nicht frei von subjektiven Einschätzungen, eignet sich aber gut zur systematischen Diskussion und einer gemeinschaftlichen Bewertung.

Im Einzelnen wurden die nachfolgend genannten Kriterien bewertet. Die ergänzend genannten Punkte führen jeweils zu Abwertungen:

- einfaches Gründungsverfahren
  - Aufwand für Registereintragungen, Notar usw.
- geringe Gründungskosten
  - erforderliche Kapitalausstattung
- Möglichkeit für flexibles Stimmrechtskonzept
  - starre Stimmrechtsverteilung, z. B. durch Anteile am Kapital
- einfache Aufnahme neuer Mitglieder
  - Neuverteilung von Anteilen, Notar
- Einfluss des Konsortiums auf Mitgliedereintritt/-wechsel gegeben
  - Verkauf/Vererbung von Anteilen
- Kündigung für Mitglieder einfach möglich
  - Kündigung nicht vorgesehen oder nur durch Verkauf von Anteilen
  - Hinweis: In den Rechtsformen, in den Kündigungen vorgesehen sind (Verein, Genossenschaft), gibt es Restriktionen durch die Kündigungsfrist.

- geringe Mindestanzahl von Mitgliedern
  - Verein erfordert sieben Mitglieder
- geringes Haftungsrisiko
  - hohes Stammkapital oder Geschäftsanteile
- geringer Prüfaufwand und Veröffentlichungspflichten der Geschäftstätigkeit
  - Jahresabschluss, Bilanzen, GuV
  - Wirtschaftsprüfer, Verbandsprüfung
- Eignung für wirtschaftliche Tätigkeiten
  - Einschränkung durch Rechtsformcharakter
- Möglichkeit der Internationalität
  - keine Aufnahme ausländischer Mitglieder möglich
- starke Außenwirkung im geschäftlichen Verkehr
  - im Wesentlichen: Image (z. B. Verein) gegenüber Geschäftspartner.

Die Bewertung ergibt folgendes Bild:

Die Rechtsformen Genossenschaft und Verein (mit und ohne angehängte GmbH) schneiden annähernd gleich gut ab und bilden daher die Vorzugsvarianten. Sie setzen sich dabei deutlich von den anderen betrachteten Varianten ab.

Die Entscheidung hängt in dieser Situation letztendlich vom Selbstverständnis des Konsortiums ab. Der Verein stellt sicherlich eine unkomplizierte Basis für gemeinsame Aktivitäten dar und kommt dem ursprünglichen Open-Source-Gedanken mit der starken Ausprägung der Gemeinschaftlichkeit und (teilweise ideellen) Eigenbeteiligung in der Softwareentwicklung am nächsten.

Es ist auf der anderen Seite aber auch klar erkennbar, dass das hier angedachte Konsortium auch starke unternehmerische Züge annehmen wird. Hier bietet die Genossenschaft mit ihrem verbindlicheren, auf eigenständigem wirtschaftlichen Handeln ausgerichteten Charakter eine bessere Alternative (siehe § 1 Abs. 1 des Genossenschaftsgesetzes: "Gesellschaften von nicht geschlossener Mitgliederzahl, deren Zweck darauf gerichtet ist, den Erwerb oder die Wirtschaft ihrer Mitglieder oder deren soziale oder kulturelle Belange durch gemeinschaftlichen Geschäftsbetrieb zu fördern (Genossenschaften), erwerben die Rechte einer "eingetragenen Genossenschaft" nach Maßgabe dieses Gesetzes.").

In diesem Zusammenhang muss deutlich darauf hingewiesen werden, dass ein Verein rechtlich nicht für wirtschaftliche Zwecke konzipiert ist. In der aktuellen Rechtsauslegung gibt es in der letzten Zeit zunehmend Beispiele, wo eine Anerkennung als Verein nicht erfolgte, weil der Nachweis, dass die wirtschaftliche Tätigkeit **nicht** im Vordergrund steht, nicht geführt werden konnte.

Der Verein mit GmbH könnte möglicherweise beide Aspekte vereinen, sollte aber erst in Betracht gezogen werden, wenn die Genossenschaft grundsätzlich abgelehnt wird. Diese Aussage ist u. a. dadurch begründet, dass in der aktuellen Rechtsprechung auch hier Zweifel an der Trennung des Vereins von den wirtschaftlichen Aktivitäten der GmbH bestehen.

Die eingetragene Genossenschaft wird daher für das Konsortium als Rechtsform empfohlen.

4.3.2 Satzung

Eine zur Darstellung des Themas im Abschnitt 4.2.2 weitergehende Bewertung ist an dieser Stelle nicht erforderlich. Es wird allerdings noch einmal die Empfehlung betont, mit einer schlanken Satzung für die gewählte Rechtsform zu starten und darauf aufbauend weitergehende Dokumente, wie z. B. eine Geschäftsordnung, zu entwickeln.

Zur weiteren Erläuterung ist in der Anlage 4-3 ein beispielhafter Entwurf einer Satzung für eine Genossenschaft beigefügt.

4.3.3 Lizenzen

Die vielfältigen OSS-Lizenzen mit ihren oftmals feinen Unterschieden decken zwar fast alle denkbaren Anforderungen an OSS-Lizenzen ab, sie fördern jedoch teilweise auch die Unsicherheit bei den Anwendern. Um dem entgegenzuwirken, sind der Einsatz einer Richtlinie für OSS-Lizenzen (z. B. Beschränkung auf einige große Lizenzen sowie ein unternehmensspezifischer Prozess für Ausnahmen) sowie die Dokumentation aller in einem Unternehmen eingesetzten OSS-Lizenzen empfehlenswert [4-4].

Bestehen bereits solche Richtlinien bei den Gründungsunternehmen, so kann dadurch die Lizenzwahl für die unter dem Dach des Konsortiums zu entwickelnden Software bereits stark eingeschränkt sein.

Die Wahl einer Lizenz für die zu entwickelnde Software kann zeitlich unabhängig davon gesehen werden.

Über die Lizenzart (ohne, schwaches, strenges Copyleft) sollte jedoch eine grundsätzliche Entscheidung vorliegen.

1. Schritt: Prüfung, ob bereits Richtlinien zum Einsatz von OSS bei den Gründungsunternehmen bestehen
2. Schritt: Prüfung, ob sich aus diesen Richtlinien Einschränkungen ergeben
3. Schritt: Prüfung, unter welcher Lizenz wiederverwertbarer Code steht
4. Schritt: ggf. Beachtung weiterer Kriterien
5. Schritt: Auswahl einer geeigneten (ggf. zu vorhandenem Code kompatiblen) Lizenz.

Grundsätzlich besteht auch die Möglichkeit, eine neue, den Anforderungen des Konsortiums angepasste Lizenz zu erarbeiten. Da dies die Integration von vorhandenem Quellcode unter anderer Lizenz sowie die Verwaltung der IP-Rechte erschweren könnte (Lizenzkompatibilität), ist hiervon jedoch abzuraten. Auch die für eine neue Lizenz fehlende Entwickler-Community bzw. die zunächst erst aufzubauende Akzeptanz der Lizenz in Entwicklerkreisen ist für einen schnellen Aufbau eines Ökosystems eher hinderlich. Die Wahl einer populären, im Markt eingeführten Lizenz kann dem entgegenwirken, wobei die Festlegung auf eine konkrete Lizenz, in der Praxis auch annähernd gleichbedeutend mit der Festlegung auf die dadurch angesprochene Entwickler-Community sein kann.

Vor dem Hintergrund, dass das Konsortium zum Ziel hat, in erster Linie branchenspezifische Software bzw. Softwarekomponenten für die Mitgliedsunternehmen zu entwickeln, sollte eine Lizenz gewählt werden, die es Dritten zwar erlaubt, Modifizierungen von vorhandenen Quellcodes unter der ursprünglichen Lizenz zu verbreiten, aber es verbietet, den Code in proprietärer Software weiterzuverwenden. Damit scheiden Lizenzen ohne Copyleft (BSD-artige Lizenzen) aus. Mitglieder und Nicht-Mitglieder erhalten jedoch die Möglichkeit, auch außerhalb des Konsortiums an der Weiterentwicklung der Software zu arbeiten, die aber im Regelfall nicht am Konsortium vorbeigeht.

Jeder, der die Software modifiziert oder einen Fehler (Bug) findet und diesen behebt, muss dies nicht zwingend dem Konsortium melden. In den meisten Fällen geschieht aber genau dieses und zwar verbunden mit der Bitte an die Leitung des Projektes den modifizierten Code in das Repository aufzunehmen. Die Motivation der Beitragenden besteht darin, nicht bei jedem neuen Release den Aufwand für das Nachziehen der eigenen Modifikation betreiben zu wollen und gleichzeitig darauf zu vertrauen, dass das Konsortium der Ort ist, an dem die Software bestmöglich gepflegt und weiterentwickelt wird. Die Aufnahme dieser Beiträge ins nächste Produkt-Release sollte verbunden sein mit der Übertragung der IP-Rechte auf das Konsortium.

In diesem Zusammenhang wird generell empfohlen, dass das Konsortium selbst möglichst weitgehend die IP-Rechte an einem Softwareprodukt besitzt. Dies lässt sich mit sogenannten Contributor-Agreements erreichen, in denen sich am Projekt beteiligte Entwickler verpflichten, die Rechte an den von ihnen entwickelten Codezeilen vollständig an das Konsortium abzutreten. Werden Entwicklungsaufträge an einen Anbieter vergeben, so muss die Rechteübertragung bereits in den Vertragsbedingungen geregelt sein, quasi als KO-Kriterium für die Auftragsvergabe.

Es kommen also in erster Linie die Lizenzen mit schwachem Copyleft-Effekt in Betracht. Eine Zusammenstellung der hier möglichen Lizenzvarianten enthält Anlage 4-4.

Sollten unter dem Dach des Konsortiums mehrere voneinander unabhängige Projekte gestartet werden, so müssen diese nicht zwingend die gleiche Lizenz verwenden, obwohl dies aus Gründen der Wiederverwertbarkeit der Codebestandteile zweckmäßig ist. So kann es zum Beispiel durch die Satzung oder den Beschluss der Mitgliederversammlung erlaubt sein, dass jedes Projekt zum Projektstart die für seine Belange geeignetste Lizenz wählen darf.

Steht vor dem Projektstart fest, dass ein großer Prozentsatz der benötigten Komponenten bereits als Open-Source-Software unter einer bestimmten Lizenz verfügbar ist, kann auch dies ausschlaggebend für die Wahl der Lizenz sein. Die im Rahmen eines Projektes eingesetzten OSS-Produkte sollten eine große Community haben und nicht von einer einzelnen Firma entwickelt werden. Idealerweise sollte es eine Reihe von Unternehmen geben, die Dienstleistungen für diese Produkte anbieten.

Bei der gemeinschaftlichen Softwareentwicklung, wie auch bei der Abnahme von Auftragsentwicklungen muss der danach vorliegende Quellcode vor der Verbreitung in jedem Falle einer IP-Kontrolle unterzogen werden, um sicherzustellen, dass keine inkompatiblen Codebestandteile in dem neuen Produkt enthalten sind.

4.4 Zusammenfassung

Das vorliegende Kapitel umreißt die wesentlichen Rahmenbedingungen für den Geschäftsbetrieb des Konsortiums zur konsortialen Entwicklung von Software auf der Basis von Open Source. Dabei werden die Aspekte Rechtsform, Satzung und Lizenzen analysiert und bewertet.

Durch die Darstellung eines Rollenmodells für die konsortiale Softwareentwicklung im Rahmen eines Ökosystems werden die Unterschiede zur herkömmlichen Beschaffung und Nutzung von IT-Werkzeugen deutlich gemacht.

Darauf aufbauend werden grundsätzliche Annahmen als Randbedingungen für das Konsortium formuliert. Diese Annahmen lassen sich im Wesentlichen wie folgt zusammenfassen:

- Das Konsortium entwickelt Open-Source-Software bzw. lässt diese entwickeln und steuert dabei den Entwicklungsprozess. Dies geschieht ohne eigene Vermarktungsinteressen allein mit dem Ziel, auf diesem Wege den Einsatz von IT-Werkzeugen bei den Mitgliedern zu optimieren und den Anbietermarkt zu erweitern bzw. für kleinere Anbieter zu öffnen. Dafür ist eine Gesellschaftsform zu finden, die deutlich über eine Kooperation hinausgeht.
- Das Konsortium ist auf Wachstum ausgerichtet und auch offen für Dienstleister und akademische Institutionen, soweit es im Interesse des Konsortiums liegt.
- Es gibt keine Restriktionen hinsichtlich der Software-Applikationen. Der Schwerpunkt wird aber nicht im Ersatz vollständiger IT-Systeme im Umfeld eines Netzbetreibers, sondern in der Ergänzung neu benötigter Funktionsmodule liegen.

Aus der nachfolgenden Analyse und Bewertung ergeben sich folgende Grundaussagen:

- Als Rechtsform wird eine eingetragene Genossenschaft empfohlen, da hier die Ziele des Konsortiums in Kombination mit der daraus resultierenden wirtschaftlichen Tätigkeit am besten zu realisieren sind.
- Für die erforderliche Satzung wurden wesentliche Eckpfeiler für den zukünftigen Geschäftsbetrieb formuliert. Die konkrete Ausformulierung und ggf. Erstellung darauf aufbauender zusätzlicher Vereinbarungen bleibt der anschließenden Umsetzungsphase vorbehalten.

- Der Einsatz von Open-Source-Software erfolgt immer unter den Prämissen freie Zugänglichkeit und Nutzung für jedermann. Die Weitergabe, Veränderung und Weiterentwicklung der Software wird jedoch unter bestimmte Lizenzen gestellt, die dafür unterschiedliche Restriktionen vorsehen. Die jeweilige Lizenz wird durch den Eigentümer des Urheberrechts vergeben. Aus diesem Grund ist es notwendig, dass das Konsortium dieses Recht hat bzw. übertragen bekommt.

  Eine konkrete Festlegung der Lizenz kann projektbezogen erfolgen. Allerdings wird empfohlen, grundsätzlich sogenannte Lizenzen mit schwachem Copyleft-Effekt auszuwählen und dabei auf eine starke Marktverbreitung dieser Lizenz zu achten.

Kapitel 5

Wirtschaftlichkeit der konsortialen Softwareentwicklung

INHALTSVERZEICHNIS

5 Wirtschaftlichkeit der konsortialen Softwareentwicklung

5.1 Fakten

Die Beurteilung der Wirtschaftlichkeit im Einsatz von Informationstechnik beruht im Wesentlichen auf folgenden Kriterien:

- Kosten für Beschaffung und Betrieb
- Funktionalität
- Qualität
- Bereitstellungszeiten.

Die in den jeweiligen Systemen enthaltene Software unterstützt die betroffenen Mitarbeiter in der Bearbeitung ihrer Aufgaben. In der Funktionalität dieser Software liegt das Potenzial zur Steigerung der Prozesseffizienz in den Unternehmen, aus der letzten Endes die Rechtfertigung für die entstehenden Kosten resultiert. Wenn dieses Effizienzpotenzial gegeben ist, kommt es darauf an, dass bei veränderten Anforderungen eine entsprechende Softwareanpassung, ein neues Modul oder gar ein neues System zeitnah bereitgestellt wird, damit ein optimaler Personaleinsatz sowie ggf. die Erfüllung gesetzlicher und regulatorischer Vorgaben möglichst kontinuierlich gewährleistet sind. Dabei muss eine hohe Qualität (im Sinne von Mängelfreiheit) gegeben sein, damit der Nutzen nicht durch umfangreiche Prüfmaßnahmen bzw. häufige Arbeitsunterbrechungen zu stark eingeschränkt wird.

Aus der Analyse im Kapitel 2 ergibt sich, dass die Ausprägung der oben genannten Kriterien in den beteiligten Unternehmen nicht optimal und verbesserungsbedürftig ist. Als ein möglicher Ausweg aus dieser Situation wurde die konsortiale Softwareentwicklung auf der Basis von Open Source (KSE) identifiziert und in ihrer technischen, organisatorischen und rechtlichen Ausprägung in den Kapiteln 3 und 4 beschrieben.

In diesem Kapitel soll nun als wesentliches Entscheidungsmerkmal die Wirtschaftlichkeit der KSE untersucht und bewertet werden. Dazu ist es notwendig, folgende Einflussgrößen zu betrachten und in Beziehung zu der konventionellen Vorgehensweise zu setzen:

- Kosten für
  - die Gründung des Konsortiums
  - den laufenden Betrieb des Konsortiums
  - Entwicklung und Einsatz der Software

- Chancen
- Risiken.

Diese Einflussgrößen werden in den nachfolgenden Abschnitten analysiert und bewertet. In diesem Zusammenhang muss darauf hingewiesen werden, dass es sich bei den Kostenpositionen um teilweise recht grobe Abschätzungen handelt. Die Angaben basieren im Wesentlichen auf Einschätzungen im Rahmen von Workshops des Projektteams unter Einbeziehung externer Expertise. Dabei wurde Wert darauf gelegt, Kostenpositionen, die sich in der Wirtschaftlichkeitsbeurteilung zum Nachteil von KSE auswirken, eher pessimistisch einzuschätzen (kein "Schönrechnen" von KSE).

Die vorliegende Wirtschaftlichkeitsuntersuchung beruht im Wesentlichen auf einem Vergleich der KSE mit der konventionellen Softwarebeschaffung. Eine wesentliche Prämisse dabei war, dass trotz der genannten Unschärfe aus einer übergreifenden Gesamtbetrachtung Aussagen über die Wirtschaftlichkeit von KSE und damit für eine Empfehlung im Sinne der Machbarkeit getroffen werden können.

5.2 Analyse und Strukturierung

5.2.1 Kosten

5.2.1.1 Kosten für die Gründung des Konsortiums

Wenn die Aussagen zur Wirtschaftlichkeit zu der Entscheidung führen, in die KSE einzusteigen, wird in einer anschließenden Umsetzungsphase die Gründung in der gewählten Rechtsform (Genossenschaft gemäß Kapitel 4) vorbereitet und - weitere positive Zwischenentscheidungen vorausgesetzt - auch vollzogen. Die einzelnen Schritte dieser Phase einschließlich der Vorbereitung und Durchführung eines Pilotprojektes werden im Kapitel 6 dieser Studie beschrieben.

An dieser Stelle werden zunächst die einzelnen Kostenpositionen genannt und quantifiziert, die erforderlich sind, um durch ein Umsetzungsprojekt die Voraussetzungen für den Konsortiumsbetrieb zu schaffen, ohne dabei auf die zeitliche Verteilung einzugehen.

Tabelle 5-1 enthält eine Aufstellung der relevanten Kostenpositionen. Hierbei wurden die anfallenden Aufwände teilweise in Tagewerken abgeschätzt. Durch Multiplikation dieser Werte mit den in der Tabelle angegebenen Tagessätzen werden für die externen Kosten auch entsprechende Angaben gemacht. Bei den internen Aufwänden wurden nur Tagewerke in einer bestimmten Bandbreite abgeschätzt. Diese Bandbreite ergibt sich durch unterschiedliche Beteiligungen der Unternehmen. Bei den unteren Werten wurde von einer Beteiligung von ca. 50 % ausgegangen.

Tabelle 5-1: Einmalige Kosten

Kostenposition	Aufwand extern		Aufwand intern	Bemerkung
	Tagewerke	Betrag (EUR)	Tagewerke	externe Kostensätze pro Tagewerk: 2.500 EUR für Rechtsberatung 1.800 EUR für sonstige Beratung
Projektinitialisierung (gemäß Kap. 6)				Die internen Aufwände zur Klärung von Aufgaben und Kompetenzen in den Häusern wurde nicht berücksichtigt
Aufbau Projektorganisation	10	18.000	10 - 20	
Konsortium				
Vorbereitung				
Satzung, Geschäftsordnung und weitere Dokumente	10	25.000	50 - 100	10 Tagewerke externe Rechtberatung
Entwurf Budget- und Businessplan			20 - 40	
Beratung durch Geno-Prüfungsverband (Satzung usw.)				in der Regel kostenfrei
Gründung				Zusätzlich zu den Gründungskosten ist der gemäß Satzung bei einem Beitritt (bzw. bei Gründung) zu leistende Betrag für den minimalen Geschäftsanteil als Einmalzahlung zu entrichten. Die Höhe eines Geschäftsanteils wird in der Satzung geregelt.
Prüfung, Eintragung in das Geno-Register		2.000		
Einrichtung Internetauftritt		25.000		
Büroeinrichtung		15.000		
Sonstiges (Konto, Briefpapier usw.)		5.000		
Hostingplattform (Kommunikationsinfrastruktur)				
Auswahl	6	10.800	10 - 20	
Beschaffung				Es wird Miete angenommen, die in die laufenden Kosten eingeht.
alternativ Miete				
Technische Plattform				
Definition Anforderungen	40	72.000	40 - 80	
Auswahl				
Lizenzkosten				kostenfrei soweit Open-Source-Produkte eingesetzt werden
Installation und Kommunikation		20.000		Hardware und Implementierung (für Release 1.0 sind Schnittstellen und Adapter Bestandteil des Pilotprojektes; später ist dies abhängig von den Kosten für Schnittstellen und Adapter zu weiteren Systemen)
Pilotprojekt				
Vorbereitung				
Festlegung des Thema				
Anforderungsanalyse				
Zeitplan und Finanzierung	20	36.000	50 - 100	
Erstellung Lastenheft				
Anforderungen an die Softwareentwicklung				
Durchführung				Für die Durchführung kann an dieser Stelle noch keine Abschätzung gemacht werden, da hierfür die Informationen aus der Vorbereitungsphase erforderlich sind.
Entwicklung der Software				
Qualitätssicherung				
Summe ohne Durchführung Pilotprojekt		**228.800**	**180 -360**	

Der in der Summenzeile ausgewiesene Aufwand ist von den Gründungsmitgliedern unabhängig von Aktivitäten zur Entwicklung von Software aufzubringen. Wenn das Konsortium wieder auseinanderfällt, weil die auftauchenden Risiken (siehe Abschnitt 5.2.4) nicht beherrscht werden können, ist der größte Teil des investierten Aufwandes verloren. Aus diesem Grunde wird im Kapitel 6 auch ein gestuftes Konzept zur Umsetzung vorgeschlagen, wo das Verlustrisiko durch entsprechende Zwischenentscheidungen minimiert werden soll. An dieser Stelle wird auch deutlich gemacht, wie sich die einmaligen Kosten zeitlich verteilen.

5.2.1.2 Laufende Kosten des Konsortiums

Die laufenden Kosten des Konsortiums sind alle fixen und variablen jährlichen Kosten, die unabhängig von einzelnen Aktivitäten/Projekten zur Entwicklung von Software entstehen. Hier werden die Aufwendungen erfasst, um die erforderlichen Voraussetzungen dafür zu schaffen und das Konsortium "am Leben zu erhalten".

Die wichtigsten Kostenpositionen sind in der Tabelle 5-2 zusammengestellt und für einen Zeitraum von fünf Jahren abgeschätzt. Dabei wurde in einigen Positionen (vor allem Personalkosten) ein moderates Wachstum der Mitgliederzahlen im Konsortium angenommen.

Die Kosten wurden ohne Rücksicht darauf ermittelt, ob die dahinter stehenden Leistungen von einzelnen Mitgliedern erbracht bzw. beigestellt werden (z. B. Büroräume). Für diesen Fall wurde angenommen, dass es hier auch eine entsprechende Kostenverrechnung mit dem Konsortium gibt.

Tabelle 5-2: Laufende Kosten des Konsortiums

Kostenposition	Betrag (EUR)					Bemerkung
	Jahr 1	Jahr 2	Jahr 3	Jahr 4	Jahr 5	
Personal						
Koordination und Leitung		100.000	100.000	100.000	100.000	
						abhängig vom Wachstum des Konsortiums
Sekretariat, Sachbearbeitung Admin, CTO	120.000	150.000	150.000	200.000	200.000	
Infrastruktur	20.000	20.000	20.000	20.000	20.000	Abschätzung WS 12.12.12
Rechner						
Lizenzen						
Administration						
Lizenzverwaltung (IP-Kontrolle)						evtl. Dienstleistungsbestandteil des Anbieters der Projekt-Hosting-Plattform
Marketing (Öffentlichkeitsarbeit)						
Internetaufritt	5.000	5.000	5.000	5.000	5.000	Contentgenerierung durch die Mitglieder
(Mitglieder-)Akquisition	10.000	10.000	10.000	10.000	10.000	(Aufgabe von Vorstand
Präsentationen, Vorträge, Messen	5.000	10.000	10.000	10.000	10.000	und/oder GF); Reisekosten, Spesen
Kommunikation						
Telefon, Internet	1.000	1.000	1.000	1.000	1.000	
Büro						
Miete	10.000	20.000	20.000	20.000	20.000	
Nebenkosten	5.000	7.500	7.500	7.500	7.500	
Ausstattung		2.500	5.000	10.000	5.000	
Sonstiges						
Versicherungen, Steuern, Abgaben	10.000	10.000	10.000	10.000	10.000	
Summe	186.000	336.000	338.500	393.500	388.500	

Die Aufstellung zeigt, dass es in der Summe zu signifikanten Beträgen kommt, die z. B. durch entsprechende Beiträge abgedeckt werden müssen. Ein wirtschaftlich betriebenes Konsortium muss durch die Entwicklung und Nutzung von Software auf Open-Source-Basis auch unter Berücksichtigung der einmaligen Gründungskosten und der projektunabhängigen laufenden Kosten signifikante Kostenvorteile gegenüber der heutigen Situation erzielen.

5.2.1.3 Kosten für die Entwicklung und den Einsatz der Software

Zur Beurteilung der Kostensituation in diesem Bereich wurde ein Vergleich zwischen der konventionellen Softwarebeschaffung und der KSE auf Basis von Modellannahmen durchgeführt. Dieser Vergleich beinhaltet alle relevanten Schritte bei der Beschaffung von Software einschließlich der Wartung. Die Annahmen für die KSE entsprechen den Randbedingungen des im Kapitel 2 ausgeführten *Open-Source-Consortium-Model.*

Der Kostenvergleich wurde im Rahmen eines speziellen Workshops unter Einbindung externer Expertise durchgeführt. Die Projektgrößen unterscheiden sich durch die Höhe der Lizenzkosten, wobei hierfür 100.000 bzw. 500.000 EUR für ein konventionelles Softwareprodukt angenommen wurden.

Die Randbedingungen und Annahmen, die den Kostenschätzungen zugrunde liegen, sind jeweils in der Vergleichstabelle angegeben. Sie sind für die beiden Projektgrößen überwiegend gleich, Unterschiede wurden nur für

- das Ausschreibungsverfahren (EU-Verfahren nach SektVO bei großen Projekten)
- die Laufzeit (zwei Jahre für große Projekte)
- die laufenden Kosten für das Konsortium

gemacht.

Als Ergebnis kann festgestellt werden:

- Bei einem kleinen Projekt ergibt sich unter den getroffenen Annahmen (6 Konsorten, Lizenzkosten 100.000 EUR) ein wirtschaftlicher Vorteil für die KSE-Variante von ca. 60.000 EUR pro Konsorte. Demgegenüber stehen jedoch die projektunabhängigen laufenden Kosten (Annahme: 200.000 EUR p. a. in Summe, das entspricht ca. 33.000 EUR pro Konsorte) für den Konsortiumsbetrieb, die diesen Vorteil nahezu aufzehren, solange die projektunabhängigen laufenden Kosten nur auf dieses eine Projekt umgelegt werden können.

- Bei einem großen Projekt ergibt sich unter den getroffenen Annahmen (6 Konsorten, Lizenzkosten 500.000 EUR) ein wirtschaftlicher Vorteil für die KSE-Variante von ca. 290.000 EUR pro Konsorte. Demgegenüber stehen jedoch die projektunabhängigen laufenden Kosten für den Konsortiumsbetrieb (Annahme: 400.000 EUR p. a. in Summe, ca. 133.000 EUR pro Konsorte für 2 Jahre Projektlaufzeit). Trotz verlängerter Laufzeit und höheren laufenden Kosten für den Konsortiumsbetrieb bleibt ein signifikanter Kostenvorteil für die KSE-Variante.

In den Abbildungen 5-1, 5-2 und 5-3 sind die Abhängigkeit der spezifischen Kosten und der daraus resultierende Einspareffekt durch KSE von der Anzahl der Konsorten sowie der Projektgröße dargestellt. Die Ergebniswerte der beiden oben beschriebenen Modellannahmen sind jeweils durch eine vertikale Linie gekennzeichnet. Die durch die blaue Kurve dargestellten Einsparungen ergeben sich ohne Berücksichtigung der separat dargestellten laufenden Kosten.

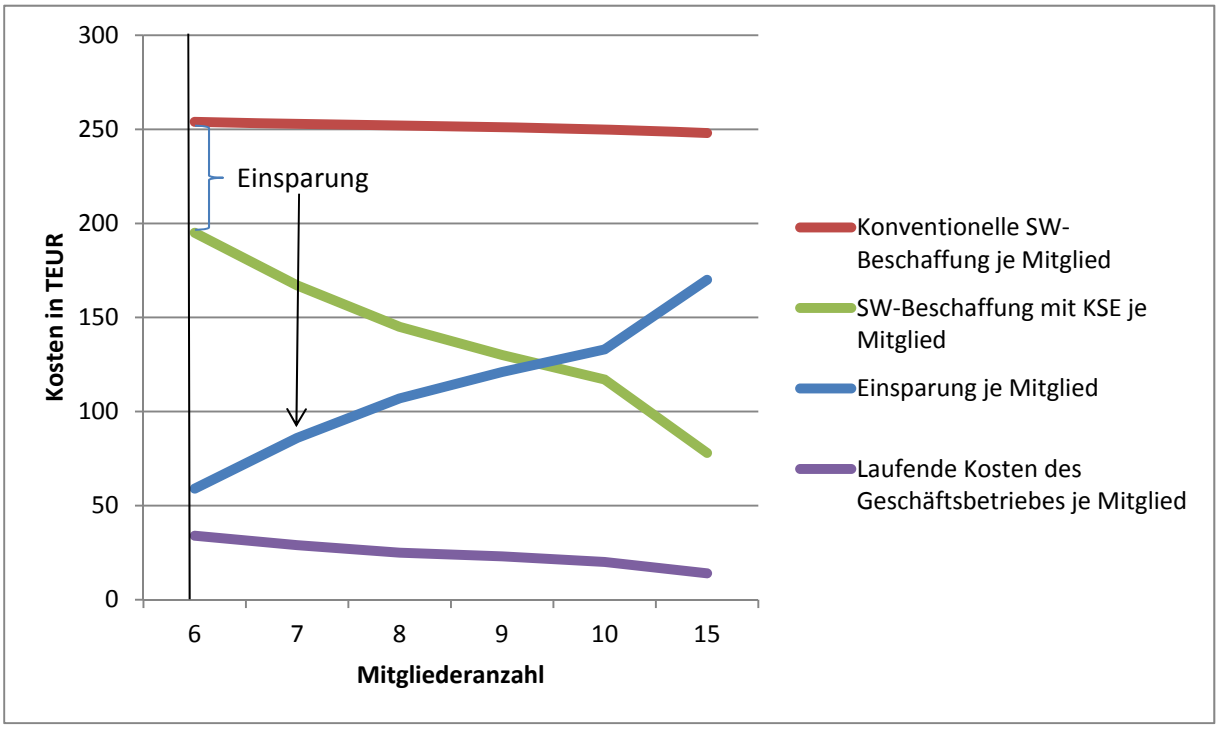


Abb. 5-1: Abhängigkeit der Kosten (in TEUR) von der Anzahl der Mitglieder im Konsortium für den Modellfall "Kleines Projekt"

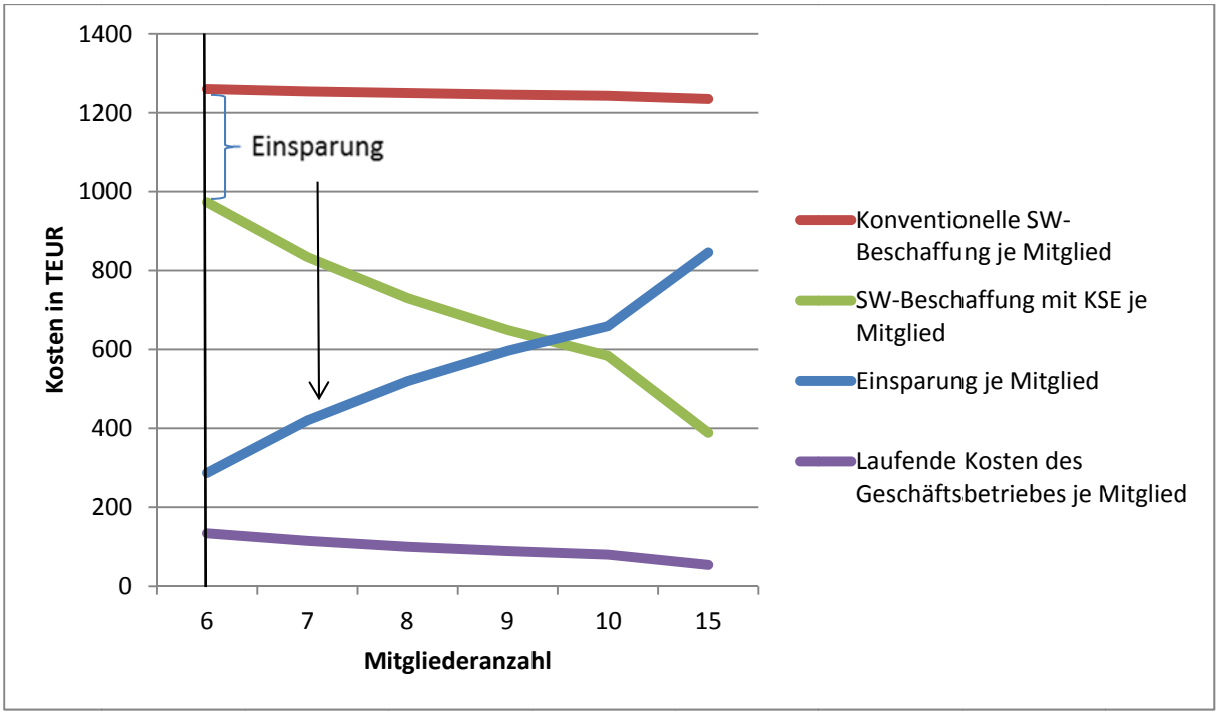


Abb. 5-2: Abhängigkeit der Kosten (in TEUR) von der Anzahl der Mitglieder im Konsortium für den Modellfall "Großes Projekt"

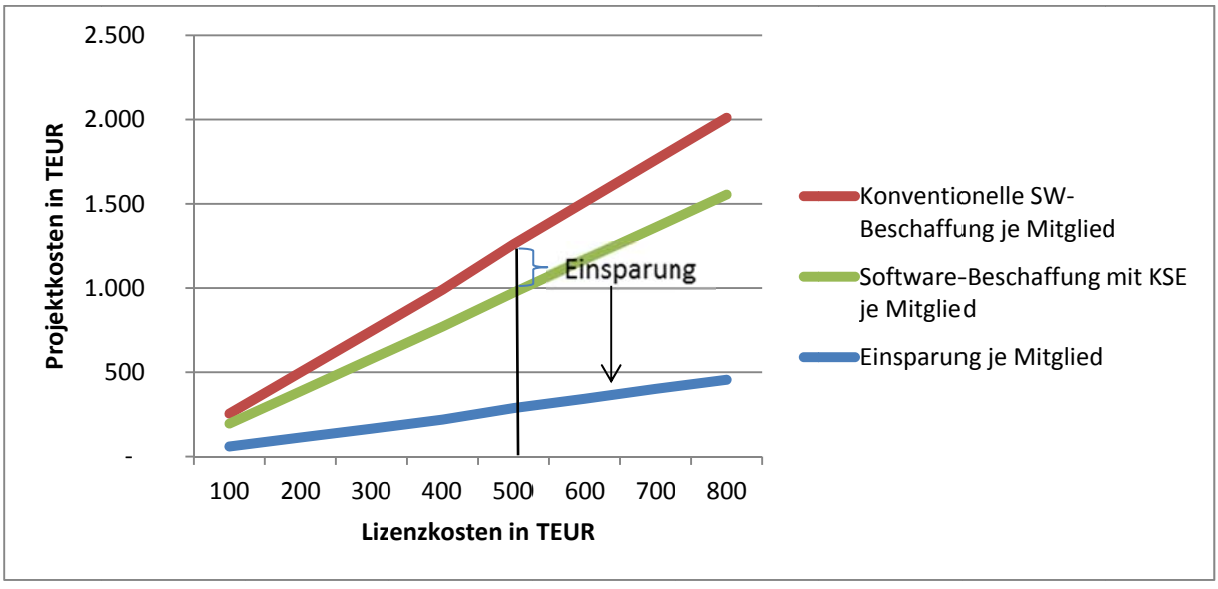


Abb. 5-3: Abhängigkeit der Kosten (in TEUR) von der Projektgröße (Lizenzkosten) bei sechs Konsorten

Im Ergebnis lässt sich festhalten, dass auch bei eher pessimistischen Ansätzen und unter Berücksichtigung der Unsicherheiten in der Kostenschätzung die KSE gute Möglichkeiten eröffnet, bereits bei der Erstellung und im Betrieb (Wartung) von Software Einsparpotenziale zu erschließen. Diese Potenziale sind umso größer, je

- mehr Mitglieder das Konsortium hat
- umfangreicher die Entwicklungsaktivitäten im Konsortium sind
- kürzer die Laufzeit in den Projekten ist, wenn dem Projekt nur die laufzeitanteiligen fixen Kosten zugerechnet werden.

5.2.2 Finanzierungsmodell

In diesem Abschnitt werden die Finanzierungsgrundsätze für den Betrieb des Konsortiums zu drei verschiedenen Zeitpunkten schematisch aufgezeigt.

5.2.2.1 Einkünfte

5.2.2.1.1 Beiträge

Aus dem jährlich zu entrichtenden Beitrag werden die laufenden Grundkosten gedeckt, das erste Projekt wird durch separate Projektbeiträge finanziert. Beide Kostenpositionen könnten durch den Einsatz von Fördermitteln oder durch Sponsoren im Sinne einer Beitragsreduzierung beeinflusst werden. Die Regularien für die mitgliederbezogene Festlegung der Projektbeiträge werden in der Satzung unter dem dort möglichen Gestaltungsspielraum definiert. Gleiches gilt für die Geschäftsanteile, wo z. B. Vorteile für Gründungsmitglieder festgelegt werden könnten.

5.2.2.1.2 Fördermittel

Die bereits angesprochene Einbeziehung von Fördermitteln kann die Wirtschaftlichkeit der konsortialen Softwareentwicklung spürbar verbessern. Dies bezieht sich sowohl auf die Gründung des Konsortiums als auch auf Projekte zur Entwicklung von Open-Source-Software.

Aufgrund des innovativen Charakters des Vorhabens im Umfeld mit den aktuellen Veränderungen in der Energieversorgungslandschaft, ist die Wahrscheinlichkeit beträchtlich, Fördermittel zu erhalten. Es wird daher empfohlen, im Zuge des Umsetzungsprojektes (siehe Kapitel 6) hier entsprechende Maßnahmen einzuleiten bzw. bereits existierende Aktivitäten zu verstärken. Dafür kann die vorliegende Studie eine wesentliche Grundlage darstellen.

Unabhängig von den genannten Fördermöglichkeiten darf eine Grundsatzentscheidung zum Einstieg in die KSE nicht davon abhängig gemacht werden.

5.2.2.1.3 Sonstige Einkünfte

Als weitere Komponente der Einkünfte des Konsortiums können Beiträge von Sponsoren veranschlagt werden, die ein Interesse an den Entwicklungsaktivitäten des Konsortiums haben. Dabei handelt es sich z. B. um Dienstleister, die den Nutzern der Open-Source-Produkte entsprechende Customizing-Leistungen anbieten wollen.

Das Konsortium selbst kann allerdings auch Dienstleistungen wie z. B. spezielle Buchveröffentlichungen oder Schulungen anbieten, aus denen sich weitere Einnahmen erzielen lassen.

5.2.2.2 Finanzströme

Abbildung 5-4 zeigt diese Finanzströme in der Anfangsphase der konsortialen Softwareentwicklung.

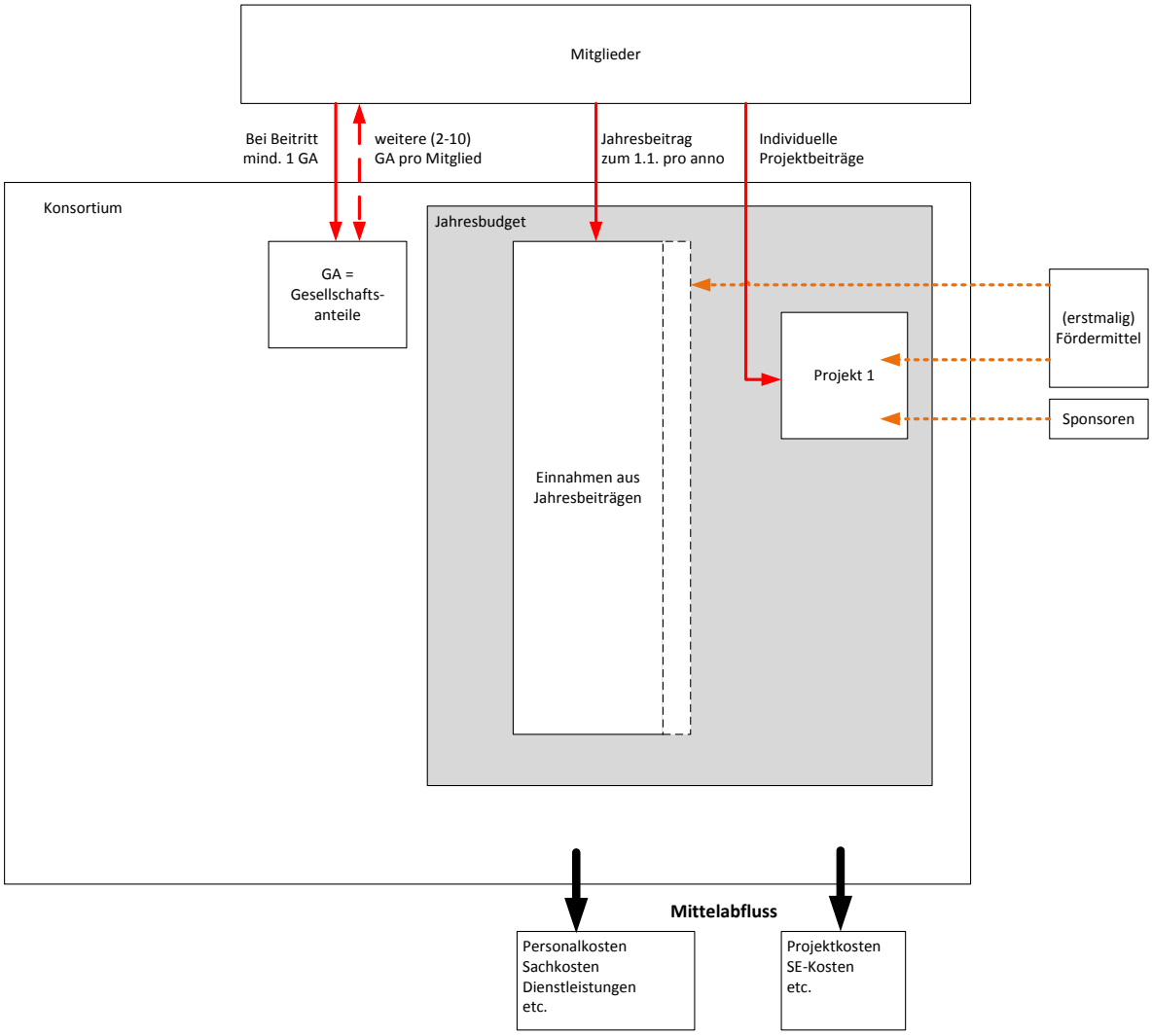


Abb. 5-4: Finanzströme im 1. Jahr

In Abbildung 5-5 werden die Finanzströme nach der Anfangsphase (z. B. im 2. Jahr nach Gründung) dargestellt. Veränderungen treten hier dadurch auf, dass

- die Anzahl der Projekte zunimmt
- die Anzahl der Mitglieder steigt
- das Konsortium durch spezielle Dienstleistungen für Mitglieder zusätzliche Einnahmen generiert.

Durch die zunehmende Mitgliederzahl besteht die Möglichkeit, einen Teil der Projektkosten über das Budget aus den Jahresbeiträgen zu finanzieren.

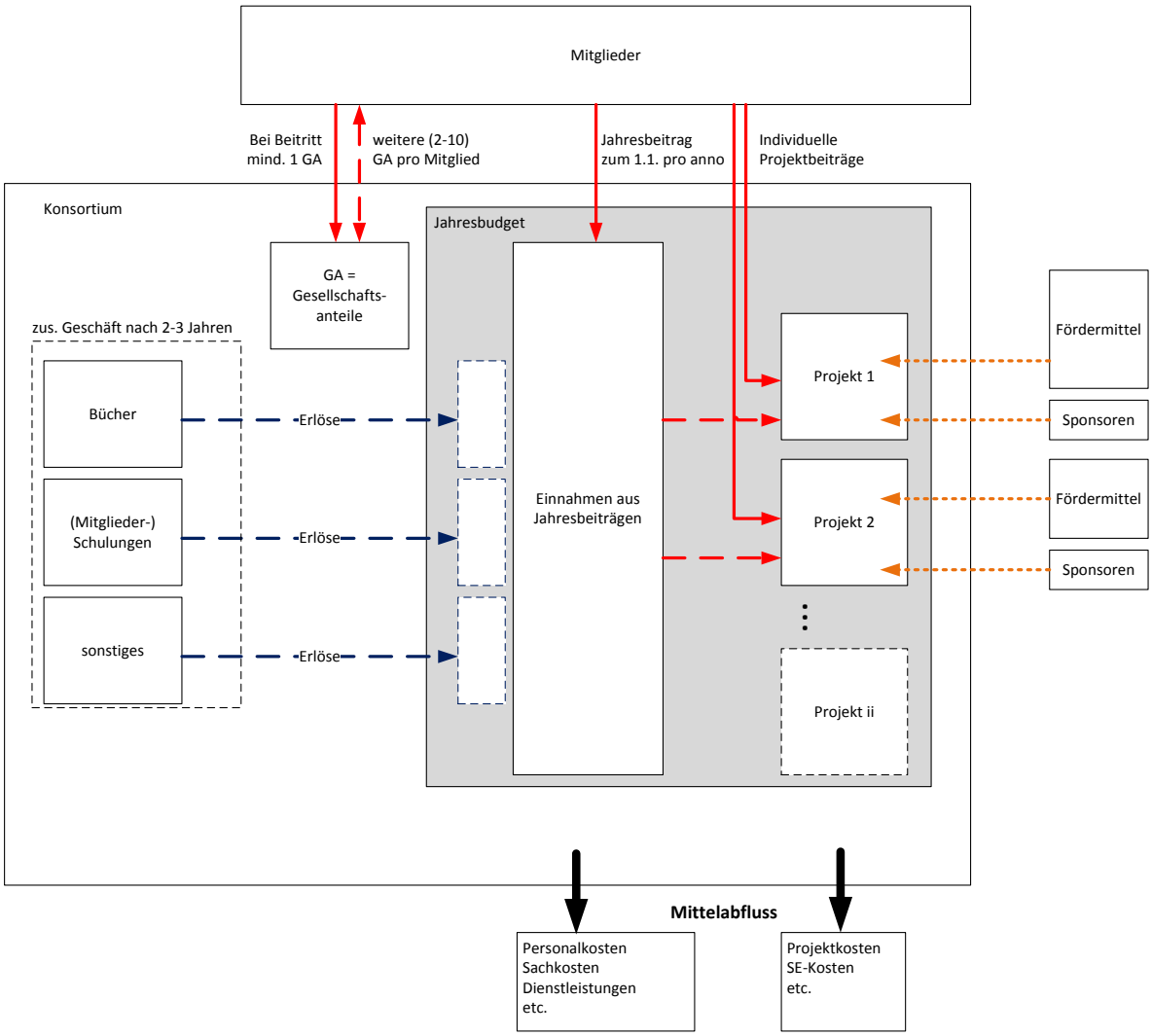


Abb. 5-5: Finanzströme ab 2. Jahr

Die Abbildung 5-6 zeigt die Situation aus mittelfristiger Sicht (z. B. nach vier Jahren). Hier wurde eine weitere Steigerung der Mitgliederzahl angenommen, sodass der Projektfinanzierungsanteil aus dem Jahresbudget sogar dominieren könnte. Individuelle Projektbeiträge einzelner Unternehmen sind natürlich hier jederzeit möglich.

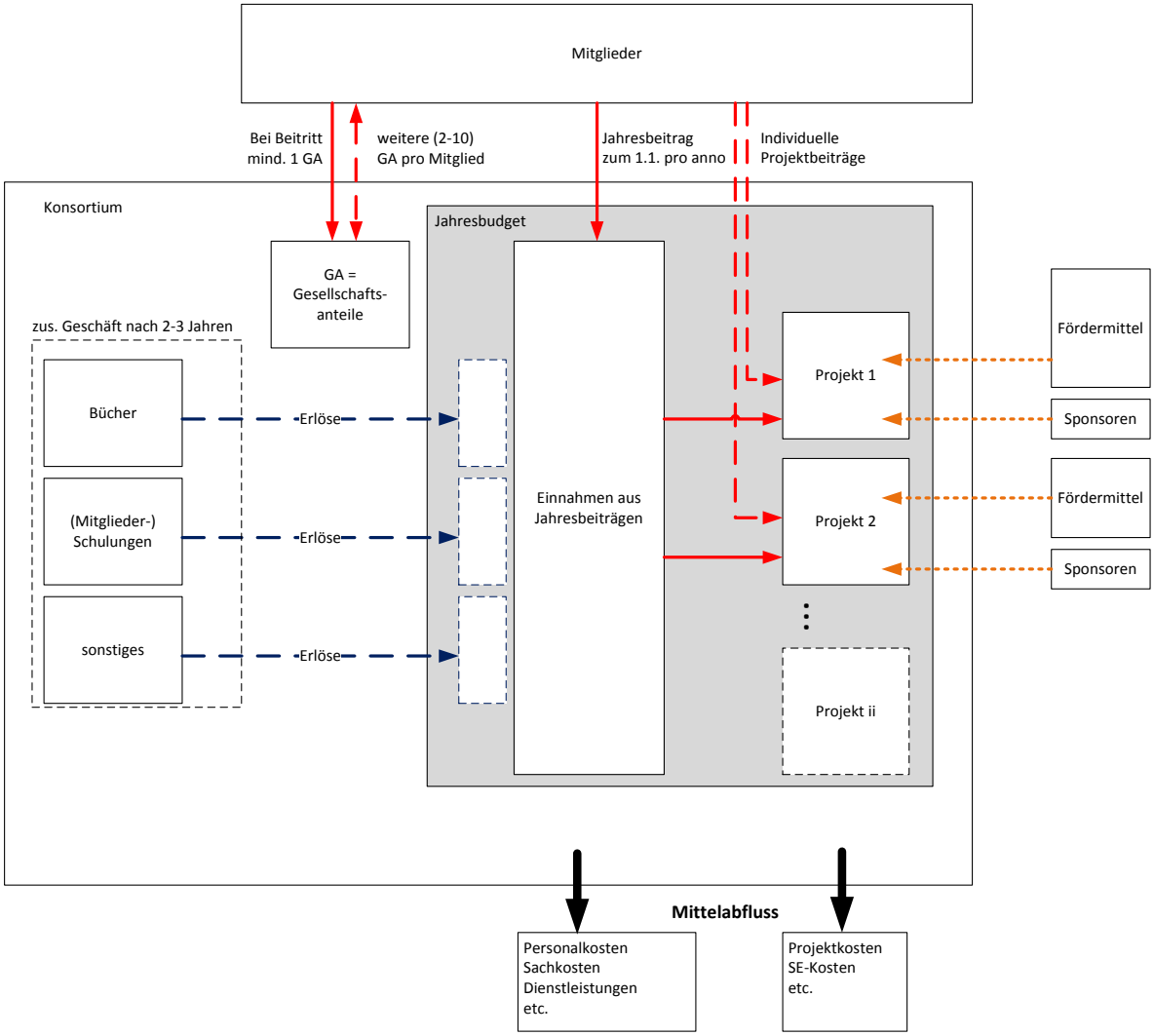


Abb. 5-6: Finanzströme ab 4. Jahr

5.2.3 Weitere Vorteile der KSE

Nachdem im letzten Abschnitt quantifizierbare Kostenunterschiede in der Beschaffung von Software untersucht wurden, sollen nachfolgend weitere Vorteile der KSE genannt werden, die teilweise einen großen Nutzen haben, aber nicht allgemeingültig für jedes einzelne der beteiligten Unternehmen quantifiziert werden können. Diese Vorteile sind in der Tabelle 5-3 zusammengestellt und gewichtet worden.

Tabelle 5-3: Weitere Vorteile der KSE

Vorteil	Gewicht (1-3)	Begründung / Erwartung
Zusätzliche Reduzierung der Softwareentwicklungskosten	**1**	Erwartung: weitere Verbesserung der Wirtschaftlichkeit bei der Softwarebeschaffung
durch Beistellung von unentgeltlichen Entwicklungskapazitäten durch Dienstleister		
Contributions von Dienstleistern und Hochschulen		
Verbesserte Prozessunterstützung beim Endanwender	**2**	Erwartung: Reduzierung von Personalkosten
durch bessere Funktionalität		stärkere Anwenderorientierung
durch frühere Verfügbarkeit (Verkürzung der Entwicklungszeiten)		
durch bessere User Interfaces (Ergonomie)		mindestens Umsetzung der SW-Ergonomierichtlinie, ermöglicht effizeinteres Arbeiten
Höhere Qualität der Software	**2**	Erwartung: Reduzierung von Personalkosten
weniger Testaufwand beim Anwender		
weniger Störungen im Arbeitsprozess		
verbessertes Bug-Fixing		
Schaffung einer modularen IT-Struktur	**3**	Erwartung: leichtere, kostengünstigere Anpassung an die Bedürnisse der Anwender, weitere Verbesserung der Wirtschaftlichkeit bei der Softwarebeschaffung
bessere Austauschbarkeit		
Erhöhte IT-Sicherheit	**2**	
Höhere Innovationsgeschwindigkeit	**2**	
Transparenz der Funktionalität gegenüber Behörden und Marktteilnehmern	**1**	Öffentlichkeitswirksamkeit
Unabhängigkeit von Lieferanten, Vermeidung Vendor-lock-in	**3**	Erwartung: Wettbewerb, weitere Verbesserung der Wirtschaftlichkeit bei der Softwarebeschaffung

5.2.4 Risiken

Die bislang dargestellten Vorteile der KSE eröffnen interessante Perspektiven für einen wirtschaftlicheren Softwareeinsatz im Vergleich zum Status Quo. Diese wirtschaftlichen Potenziale können aber nur erschlossen werden, wenn die Gründung und der Betrieb des Konsortiums mit der notwendigen Beteiligung und dem erforderlichen Bedarfshintergrund den Erwartungen entsprechend laufen. Dies ist nicht selbstverständlich, da die konsortiale Softwareentwicklung auf der Basis von Open Source zumindest in der Versorgungswirtschaft ein absolutes Novum und in der angestrebten Ausprägung einen massiven Eingriff in den vorhandenen IT-Markt für Netzbetreiber darstellt.

Es darf daher nicht verschwiegen werden, dass mit dem Vorhaben auch eine Reihe von Risiken verbunden sind, die zu einem Scheitern und damit zum Verlust des bis dahin investierten Kapitals (Gründungskosten, ggf. Pilotprojekt) führen können. Aus dieser Tatsache resultiert - wie bereits an anderer Stelle in diesem Kapitel erwähnt - eine gestufte Entscheidungsstrategie bei der Vorbereitung und dem Eintritt in die KSE, die auch Abbruchkriterien enthält (siehe Kapitel 6).

In der Tabelle 5-4 sind die identifizierten Risiken zusammengestellt, gruppiert (Gründungs- und Betriebsphase) und gewichtet worden. Bei den einzelnen Risikopositionen sind jeweils Maßnahmen zur Reduzierung der Eintrittswahrscheinlichkeit genannt.

Tabelle 5-4: Zusammenstellung der Risiken

Risiko	Gewicht (1-3) Auswirkungen	Maßnahmen
Gründungsphase		
Anzahl Mitglieder zu gering	3	Überzeugungsarbeit auf Basis der Studie
Umsetzungs- und Gründungsphase dauert zu lang	3	straffes Projektmanagement
Geeignetes Personal / Treiber kann nicht rekrutiert werden	3	attraktive Stellen anbieten (Gehälter), Neuartigkeit des Konsortiums betonen
Auswirkung eines scheiternden Pilotprojektes	3	interessante Applikation und straffes Projektmanagement
Ökosystem kommt nicht zustande	2	Interesse der Dienstleister durch großes Projektvolumen wecken
Massive Proteste der Lieferanten bei den Unternehmensleitungen	2	frühzeitige Einbindung und Information aller Beteiligten
Dominanz einiger Unternehmen im Konsortium	1	Satzung
Betriebsphase		
Mitglieder können sich aufgrund unterschiedlicher Prozesse nicht auf gemeinsame Spezifikationen einigen, Anwendungen laufen durch zu hohen Customizing-Anteil auseinander	3	klare Entscheidungswege für Antrag und Initialisierung von Projekten
zu geringes Projektvolumen	2	Ausdehnung auf weitere Applikationen im Unternehmen
Geeignetes Personal (CTO, GF) kann nicht rekrutiert werden	2	Öffentlichkeitsarbeit, attraktive Stellen anbieten (Gehälter), Neuartigkeit des Konsortiums betonen
Ökosystem kann nicht stabil gehalten werden	1	
Anzahl Mitglieder zu gering	1	Öffentlichkeitsarbeit, erfolgreiches Pilotprojekt

5.3 Bewertung

Die im vorigen Abschnitt durchgeführten Untersuchungen lassen sich wie folgt bewerten:

- Eine Entscheidung für den Einstieg in die KSE erfordert erhebliche Aufwände für die Gründung des Konsortiums sowie die Vorbereitung eines Pilotprojektes, die einmalig von den Gründungsmitgliedern aufgebracht werden müssen. Diese Aufwände wurden für externe Leistungen in Summe mit 228.800 EUR abgeschätzt. Hinzu kommen interne Leistungen der beteiligten Unternehmen, die je nach deren Beteiligung bei den einzelnen Aktivitäten mit 180 bis 360 Tagewerken veranschlagt wurden. Die Bewertung der internen Tagewerke wird jedes Unternehmen für sich selbst vornehmen. Bewertet man beispielsweise ein internes Tagewerk mit 1.000 EUR, belaufen sich die geschätzten internen Kosten in Summe auf 180.000 bis 360.000 EUR.
- In Anbetracht der potenziellen Einsparungen werden die genannten Aufwände bei einer genügend großen Zahl von Beteiligten (z. B. die sechs Unternehmen aus dieser Machbarkeitsstudie) auch unter Risikogesichtspunkten als vertretbar angesehen.
- Die laufenden, projektunabhängigen Kosten für das Konsortium werden unter normalen Betriebsbedingungen anfänglich auf 186.000 EUR p. a. geschätzt und können bei signifikantem Wachstum auf ca. das Doppelte ansteigen. Diese Kosten müssen durch die satzungskonformen Mitgliederbeiträge gedeckt werden. Für die Mitglieder wiederum ist die Zahlung dieser Beiträge wirtschaftlich nur sinnvoll, wenn sie durch Einsparungen in der Softwarebeschaffung gedeckt werden können.
- Der modellhafte Vergleich zwischen konventioneller Vorgehensweise und KSE zeigt klare Einsparpotenziale zugunsten von KSE, die sich schwerpunktmäßig bei den Lizenzkosten und den Wartungsaufwendungen ergeben. Diese Aussage ist umso tragfähiger, je
  - mehr Mitglieder das Konsortium hat
  - umfangreicher die Entwicklungsaktivitäten im Konsortium sind
  - kürzer die Laufzeit in den Projekten ist.
- Neben den modellhaft wirtschaftlichen Vorteilen bei der Softwarebeschaffung bringt der Einsatz von konsortialer Open-Source-Software eine Reihe von weiteren Vorteilen (siehe auch Kapitel 3), die zwar im Rahmen der Studie nicht quantifiziert wurden, die Wirtschaftlichkeit aber auf jeden Fall deutlich verstärken.

Dieser Effekt resultiert u. a. aus der Möglichkeit, die Arbeitsprozesse durch die schnelle Bereitstellung angepasster Funktionalitäten effizienter zu unterstützen und dadurch ein entsprechendes Einsparpotenzial bei den Personalkosten zu erschließen. Weitere Einsparpotenziale liegen im Aufbrechen von Herstellerbindungen (Überwindung Lock-in-Effekt) mit den daraus resultierenden Wettbewerbseffekten sowie der Möglichkeit, die existierende Systemwelt modular und damit kostengünstig erweitern zu können.

- Aufgrund des innovativen Charakters eines Konsortiums zur Entwicklung von Open-Source-Software und in Abhängigkeit der funktionalen Themen gibt es gute Aussichten für finanzielle Förderungen. Hier lohnt es sich, die bereits begonnenen Aktivitäten zur Erkundung von Fördermöglichkeiten zu intensivieren. Unabhängig davon sollte die Entscheidung für den Einstieg in die KSE nicht von Fördermaßnahmen abhängen.
- Der Einstieg in die KSE ist allerdings auch mit signifikanten Risiken verbunden. Um diese im Griff zu behalten, müssen geeignete Maßnahmen im Bereich der Informationsstrategie, der Öffentlichkeitsarbeit sowie des Projekt- und Risikomanagements ergriffen werden.

Die dargestellten Ergebnisse lassen den Schluss zu, dass die KSE unter den angenommenen Randbedingungen wirtschaftlich ist. Obwohl die durchgeführten Kostenschätzungen und modellhaften Vergleiche mit Unsicherheiten behaftet und dadurch natürlich auch angreifbar sind, ergibt sich in der Gesamtbetrachtung zusammen mit den nicht quantifizierten Vorteilen eine große Chance für nennenswerte Einsparungen. Unter dieser Betrachtungsweise ist die Entscheidung für den Einstieg in die KSE eine strategische Entscheidung mit guten Erfolgschancen, deren Machbarkeit u. a. durch die Beispiele erfolgreicher Open-Source-Projekte auch international untermauert wird.

5.4 Zusammenfassung

In den vorliegenden Abschnitten wurde die Wirtschaftlichkeit der konsortialen Softwareentwicklung auf Basis von Open-Source-Software (KSE) systematisch untersucht. In einem ersten Teil wurden die Kosten für die Gründung und den Betrieb eines Konsortiums analysiert. Anschließend wurde ein modellhafter Kostenvergleich durchgeführt, um eine Aussage darüber machen zu können, ob und unter welchen Bedingungen KSE Kostenvorteile gegenüber dem konventionellen Softwareeinsatz aufweist.

In einem zweiten Schritt wurden weitere wirtschaftlich relevante, aber im Rahmen der Studie nicht quantifizierte Vorteile zusammengestellt. Im Anschluss erfolgte eine Analyse der möglichen Risiken, die einen wirtschaftlichen Erfolg reduzieren oder gar verhindern können.

Die abschließende Bewertung der Untersuchungen lässt sich wie folgt zusammenfassen:

- Die mit der Konsortiumsgründung einschließlich Vorbereitung eines Pilotprojektes verbundenen einmaligen Kosten liegen in ihrer geschätzten Höhe bei 228.800 EUR für die externen Kosten. Die internen Aufwände werden in einem Bereich von 180 bis 360 Tagewerken (Werte aus Tabelle 5-1, Bandbreite ergibt sich durch unterschiedliche Beteiligung der Unternehmen) abgeschätzt, wobei hier jeweils der gültige Kostensatz der einzelnen Unternehmen zum Ansatz gebracht werden muss. Insgesamt werden die mit der Konsortiumsgründung verbundenen Kosten vor dem Hintergrund der nachfolgend aufgezeigten wirtschaftlichen Chancen als vertretbar angesehen.
- Die laufenden projektunabhängigen Kosten für das Konsortium werden unter normalen Betriebsbedingungen in Abhängigkeit von der Mitgliederzahl und dem durchgeführten Projektvolumen auf 186.000 EUR p. a. bis 393.500 EUR p. a. abgeschätzt.
- Der modellhafte Vergleich zwischen konventioneller Vorgehensweise und KSE zeigt ein klares Einsparpotenzial zugunsten von KSE, die diese laufenden Kosten deutlich übersteigen können. Dieser Effekt verstärkt sich umso mehr, je stärker das Konsortium bezüglich Mitgliederzahl und Entwicklungsaktivitäten wächst.
- Die Wirtschaftlichkeitseinschätzung zugunsten der KSE verstärkt sich deutlich durch die Identifizierung weiterer Vorteile, die sich in unterschiedlicher Form in den Arbeitsabläufen und den IT-Prozessen (Implementierung, Support usw.) der einzelnen Netzbetreiber ergeben und daher auch nicht allgemeingültig abgeschätzt werden können.
- Die Möglichkeiten zur Kostenentlastung durch Fördermaßnahmen und andere Einkünfte sollten konsequent genutzt werden, dürfen aber keine Bedingung für eine Entscheidung zum Einstieg in die KSE sein.

Die dargestellten Ergebnisse lassen den Schluss zu, dass die KSE auch unter Berücksichtigung der vorliegenden Unsicherheiten große wirtschaftliche Chancen bietet. Außerdem wird damit für Netzbetreiber eine solide Grundlage gelegt, um den zukünftigen Anforderungen an IT-Werkzeuge im regulierten Umfeld bei wachsendem Kostendruck und höherer Innovationsgeschwindigkeit gerecht werden zu können. Aus diesen Gründen ist eine strategische Entscheidung für die Gründung eines Konsortiums gerechtfertigt.

Der Einstieg in die KSE ist auch mit signifikanten Risiken verbunden. Um diese im Griff zu behalten, müssen geeignete Maßnahmen im Bereich der Informationsstrategie, der Öffentlichkeitsarbeit sowie des Projekt- und Risikomanagements ergriffen werden. Dazu gehört auch in der Anfangsphase eine abgestufte Entscheidungsstrategie unter Berücksichtigung von Abbruchkriterien.

Kapitel 6

Umsetzung

INHALTSVERZEICHNIS

6 Umsetzungs- und Finanzierungsplan

6.1 Projektstrategie

Durch die in dieser Studie zusammengetragenen und erarbeiteten Informationen wird der Nachweis erbracht, dass die konsortiale Softwareentwicklung auf der Basis von Open Source (KSE) einen technisch und wirtschaftlich nachhaltigen Lösungsansatz bietet, die heutigen Probleme bei Netzbetreibern im Einsatz von Software bzw. mit den Softwareanbietern (Stichwort "vendor-lock-in") zu reduzieren bzw. zu überwinden und neue Herausforderungen an die Unterstützung der Geschäftsprozesse schneller und besser umzusetzen.

In diesem Kapitel werden die erforderlichen Maßnahmen zum Einstieg in die KSE beschrieben und die dafür geltenden zeitlichen und finanziellen Randbedingungen aufgezeigt. Im Fokus dieser Überlegungen steht das Ziel, die Voraussetzungen für ein funktionsfähiges Konsortium zu schaffen sowie ein Pilotprojekt vorzubereiten und durchzuführen (Umsetzungsprojekt). Ausgangspunkt für diese Überlegungen ist selbstverständlich die positive Grundsatzentscheidung einer ausreichend großen Zahl von Unternehmen, der Empfehlung dieser Studie zu folgen. Die Vorgehensweise zur Herbeiführung dieser Grundsatzentscheidung ist unternehmensspezifisch und wird daher an dieser Stelle nicht näher behandelt.

Da der praktische Nachweis für die Funktionsfähigkeit des KSE-Ansatzes möglichst zeitnah erfolgen sollte, wurde festgelegt, dass das Pilotprojekt bis Ende 2014 abgeschlossen sein muss. In Abhängigkeit vom Inhalt werden grundsätzlich zwei strategische Varianten für die Projektdurchführung gesehen (siehe Abbildung 6-1):

- Variante 1: Das Pilotprojekt beinhaltet ein abgeschlossenes Funktionsmodul, das von der Größe her etwa dem sogenannten "kleinen Projekt" gemäß Modellrechnung im Kapitel 5.2.1.3 entsprechen könnte.
- Variante 2: Das Pilotprojekt beinhaltet ein erstes Modul eines umfangreichen Gesamtprojektes, das von der Größe her etwa dem sogenannten "großen Projekt" gemäß Modellrechnung im Kapitel 5.2.1.3 entsprechen und sich über mehrere Jahre erstrecken könnte (z. B. bis 2016).

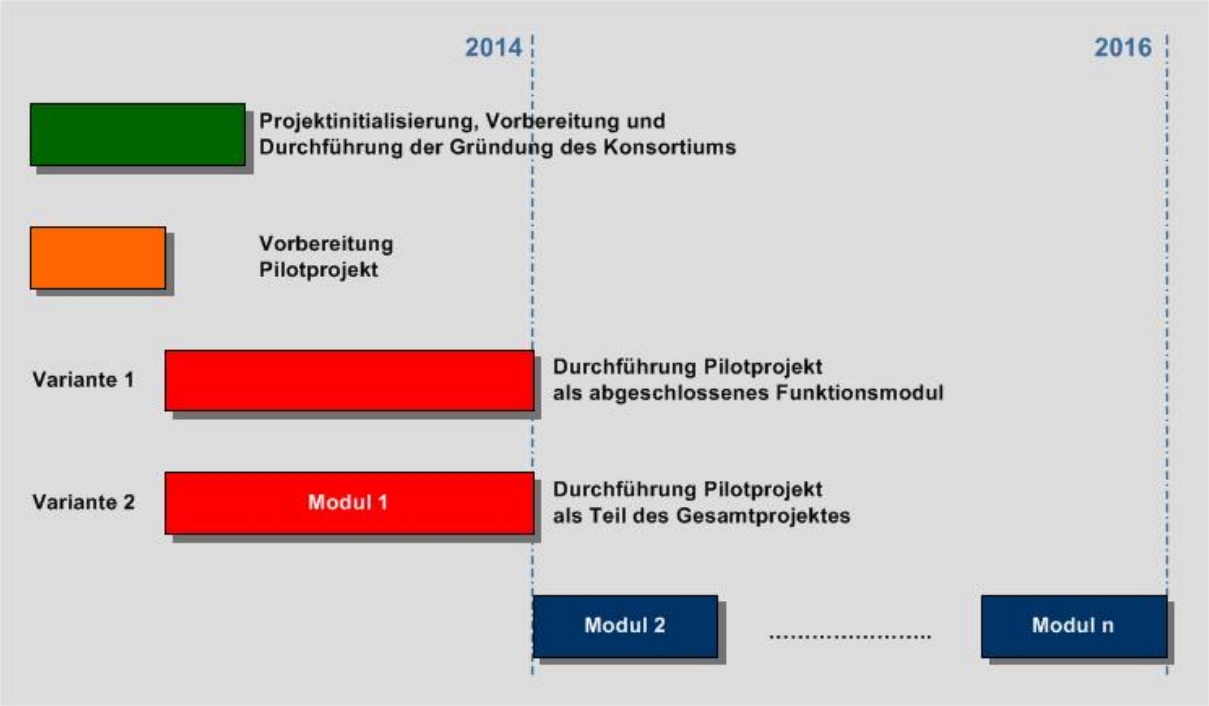


Abb. 6-1: Strategische Projektvarianten

6.2 Analyse, Strukturierung und Bewertung

6.2.1 Umsetzungsprojekt

6.2.1.1 Einleitung

Ziel des Umsetzungsprojektes ist es, alle organisatorischen, rechtlichen, finanziellen und technischen Voraussetzungen zu schaffen, um ein Konsortium zu gründen und ein geeignetes Pilotprojekt durchzuführen. Die wesentlichen Merkmale des Umsetzungsprojektes sind in den folgenden Abschnitten beschrieben.

6.2.1.2 Projektinitialisierung

6.2.1.2.1 Unternehmensinterne Festlegungen

Aufgrund der Neuartigkeit in der Beschaffung und dem Einsatz von Software macht es der Einstieg in die KSE und die damit verbundene Mitgliedschaft in einem Konsortium erforderlich, eine Reihe von betroffenen bzw. zuständigen Bereichen, Abteilungen usw. zu informieren und bei Bedarf einzubeziehen. Diese Maßnahmen sind sehr unternehmensspezifisch auszuprägen und können an dieser Stelle nicht allgemeingültig formuliert werden. Unabhängig davon sind eine Reihe von Fragen zu klären bzw. Entscheidungen herbeizuführen, die in der Durchführung des Umsetzungsprojektes von Bedeutung sind.

Typische Fragestellungen in diesem Zusammenhang sind:

a. Wie wird das Projekt unternehmensintern kommuniziert?

b. Wie sieht der unternehmensinterne Entscheidungsprozess aus?

c. Welche Unternehmensbereiche sind im Rahmen des Umsetzungsprojektes konkret einzubinden und stehen die erforderlichen Ressourcen zur Verfügung?

d. Welche Mitarbeiter/-innen werden als Unternehmensvertreter benannt und mit entsprechenden Entscheidungsbefugnissen ausgestattet, um in den unten beschriebenen Teilprojekten effektiv mitzuarbeiten?

e. Wie erfolgt die Bereitstellung des erforderlichen Budgets?

f. Wie sieht die projektspezifische Entscheidungsstruktur im Unternehmen aus?

g. Welcher Unternehmensteil wird Gründungsmitglied (Konzernmutter oder -tochter)?

6.2.1.2.2 Projektorganisation

Für das Umsetzungsprojekt ist eine entsprechende Organisation mit folgenden Merkmalen bzw. Aufgaben zu installieren:

- Projektstruktur (Teilprojekte, siehe folgender Abschnitt 6.2.1.3)
- Lenkungskreis
- Projektleitung (Gesamtprojektleiter, Teilprojektleiter)
- personelle Besetzung der Teilprojekte (Experten aus den Unternehmen)
- Projektmanagement und Controlling (Überwachung Termine, Kosten, Risiken usw.)
- Changemanagement
- externe Experten zur Unterstützung im Projektmanagement und bei der inhaltlichen Arbeit (Berater, Juristen usw.)
- Regularien für die Projektarbeit:
    - Entscheidungsmechanismen, Kompetenzabgrenzungen zwischen Projektleitung und Lenkungskreis, Verbindlichkeit von Mehrheitsentscheidungen
    - Finanzierung des Projektes, Beteiligung der Unternehmen an den Kosten, Verrechnung der Mitarbeit in den Teilprojekten usw.
    - Berichtswesen.

Im Zusammenhang mit der Projektorganisation sind auch die Befugnisse im Rahmen des Kommunikationskonzeptes festzulegen, besonders hinsichtlich der Kontakte zur Öffentlichkeit und zu Medien ("Sprecherfunktion").

6.2.1.3 Teilprojekte

Es wird vorgeschlagen, das Umsetzungsprojekt in fünf Teilprojekte (TP) zu unterteilen, die nachfolgend mit ihren wesentlichen Inhalten genannt sind, ohne dass dabei eine zeitliche Zuordnung getroffen wird.

- **TP 1 Vorbereitung und Durchführung der Gründung:**
    - Rahmenbedingungen für die Entwicklung des Ökosystems festlegen

- Satzung sowie unterlagerte Dokumente (einschließlich aller in diesem Zusammenhang zu entscheidenden Punkte) auf der Basis der im Kapitel 4 dieser Studie beschriebenen Inhalte (einschließlich Satzungsentwurf aus Anlage 4-3). In diesem Rahmen erfolgt auch eine Definition von Mitgliederstufen, Beitragsordnung, Stimmrechte usw.
- Aktivitäten zur Vorbereitung des Geschäftsbetriebes (Besetzung von Vorstand, Konzeption und Aufbau einer Website, Entwurf und Abstimmung eines Logos sowie weiterer Bestandteile einer Corporate Identity etc.)
- Kommunikationskonzept zur Publizierung des Projektes und zur Akquisition weiterer Mitglieder in Abhängigkeit von der angestrebten Mitgliederentwicklung besonders in der Anfangsphase
- Entscheidung über den Gründungszeitpunkt
- Durchführung der Gründung.

- **TP 2 Auswahl der technischen Plattform:**
  - Analyse und Definition funktionaler und nichtfunktionaler Anforderungen, Verifizierung der im Kapitel 3 dieser Studie getroffenen Annahmen und Empfehlungen
  - Auswahl der Softwarearchitektur unter Berücksichtigung der vorhandenen IT-Landschaften und zugehörigen IT-Strategien in den Gründungsunternehmen
  - Ermittlung der für den Piloten zusätzlich zu realisierenden Schnittstellen bzw. Adapter.
- **TP 3 Auswahl der Hosting-Plattform für die Softwareentwicklung:**
  - Bestandteile der Plattform auswählen, Verifizierung der im Kapitel 3 dieser Studie getroffenen Annahmen und Empfehlungen
  - ggf. Dienstleistungsangebote einholen und auswerten
  - ggf. eigene Governance und Spielregeln für den Entwicklungsprozess festlegen.
- **TP 4 Vorbereitung des Pilotprojektes:**
  - Analyse des aktuellen Bedarfes (thematisch, inhaltlich) und Festlegung des Themas
  - Anforderungsanalyse und Abschätzung / Abgrenzung der Inhalte des Pilotprojektes
  - Festlegung Zeitplan und Finanzierung
  - Erstellung eines Lastenheftes

- zusätzliche Anforderungen an die Softwareentwicklung formulieren (Methode, Transparenz, OSS-Lizenz etc.).

- **TP 5 Durchführung des Pilotprojektes:**
  - Ausschreibung und Vergabe von Dienstleistungen für Softwareentwicklung, Projektmanagement, Qualitätssicherung usw.
  - Projekt-Hosting-Plattform in Betrieb nehmen
  - Beschaffung, Implementierung und Inbetriebnahme der technischen Plattform in einer Testumgebung
  - Entwicklungsprozess starten
  - Entwicklung der notwendigen Standard-Konnektoren
  - Realisierung der ersten OS-Module
  - Abnahme und Bereitstellung der Software (Veröffentlichen auf Downloadserver)
  - Auswertung und rückblickende Analyse des Projektverlaufes.

Die Bearbeitungsinhalte sind im Rahmen der Projektinitialisierung jeweils zu detaillieren und ggf. zu ergänzen. Die Implementierung des Pilotmoduls bei den Anwendern mit den dafür erforderlichen Maßnahmen (Customizing, Schulung usw.) ist kein Bestandteil des Pilotprojektes.

Ebenso ist festzulegen, wie die Mitarbeit der Unternehmen in den Teilprojekten grundsätzlich aussehen soll. Es ist nicht zwingend notwendig, dass Vertreter aus allen Unternehmen in allen Teilprojekten mitarbeiten, jedoch müssen die Ergebnisse von allen mitgetragen werden. Hier könnte man z. B. thematisch aufteilen. Dabei wäre es optimal, wenn die Unternehmen in Summe etwa gleichmäßig belastet würden, wobei auch dies nicht zwingend erforderlich ist. Bei der Aufwandsschätzung im Kapitel 5 ist diese Unsicherheit durch eine entsprechende Bandbreite in der Beteiligung zwischen 50 % und 100 % zum Ausdruck gebracht worden. Somit hat die "Lastverteilung" auch Auswirkung auf die Finanzierung (siehe Abschnitt 6.2.2) des Umsetzungsprojektes.

Grundsätzlich ist zu diesem Punkt anzumerken, dass eine umfassende Beteiligung der Unternehmen zwar finanziell und auch zeitlich aufwendiger ist, aber andererseits auch sehr das gegenseitige Kennenlernen und die Ergebnisakzeptanz fördert. Dieser Aspekt ist bei dem hier betrachteten Vorhaben nicht zu unterschätzen.

6.2.2 Zeit- und Finanzierungsplan für das geplante Umsetzungsprojekt

Für die Durchführung der in den vorigen Abschnitten beschriebenen Maßnahmen wird folgender Zeitbedarf angesetzt:

- Projektinitialisierung 2 Monate
- TP 1: Vorbereitung und Durchführung der Gründung 5 Monate
- TP 2: Auswahl der technischen Plattform 3 Monate
- TP 3: Auswahl der Hosting-Plattform 3 Monate
- TP 4: Vorbereitung des Pilotprojektes 5 Monate
- TP 5: Durchführung des Pilotprojektes 12 Monate.

Unter der Annahme, dass die Teilprojekte zumindest teilweise parallel bearbeitet werden können, wird folgender Zeitplan vorgeschlagen:

In dem dargestellten Plan sind auch Meilensteine eingetragen, an denen Entscheidungen über die Fortführung des Projektes enthalten sind. Zu diesen Zeitpunkten besteht im Interesse einer Risikobegrenzung jeweils die Möglichkeit, dass einzelne Unternehmen aus dem Projekt aussteigen bzw. dass das Projekt abgebrochen wird. Der Finanzierungsbedarf ergibt sich im Wesentlichen aus den Betrachtungen im Kapitel 5. An dieser Stelle werden diese Kosten im Rahmen eines Zeit- und Finanzierungsplans ebenfalls in Abbildung 6-2 noch einmal zeitlich gestaffelt zusammen aufgeführt.

Um in dem Finanzierungsplan konkrete Kostenwerte angeben zu können, wurden folgende Annahmen getroffen:

- Die externen Kosten werden gemäß den Angaben im Kapitel 5 eingesetzt.
- Für die internen Kosten werden die jeweils höchsten Tagewerksangaben angenommen. Gemäß Kapitel 5.3 erfolgt hier eine Bewertung unternehmensindividuell.
- Die Kosten für die Realisierung des Piloten wurden als Anhaltspunkt auf der Basis des sogenannten "kleinen Projektes" gemäß Modellrechnung im Kapitel 5.2.1.3 für beide strategischen Projektvarianten quantifiziert.
- Es wurde angenommen, dass nach der Gründung des Konsortiums laufende Kosten anfallen, die im Kapitel 5.2.1.2 für das erste Jahr mit 186.000 EUR abgeschätzt wurden.

- Die Kosten für die Realisierung des Piloten teilen sich auf in externe Entwicklungskosten und den vorab bzw. parallel zur Entwicklung anfallenden internen Aufwand für die Spezifikation der Anforderungen (dem sogenannten Produkt Backlog) im Requirement-Team.
- Die laufenden Kosten des Konsortiums wurden vollständig als externe Kosten geführt.
- Ein zusätzlicher interner Aufwandsblock zur Begleitung des Pilotprojektes wurde angesetzt, um die in der Aufbauphase erwarteten zusätzlichen Abstimmungen im Entwicklungsprozess und Konsortiumsbetrieb (Stichwort: Lernkurve) zu bemessen. Die individuellen Unternehmensanteile können variieren.

Die einzelnen Kostenwerte wurden in der Darstellung jeweils dem Zeitpunkt zugeordnet, der in der Mitte des für die jeweilige Maßnahme angenommenen Zeitraums liegt. Darüber hinaus wurden diese Kostenwerte im Sinne einer Budgetplanung jahresbezogen zusammengeführt.

In der Summe beläuft sich das Gesamtbudget für das Umsetzungsprojekt unter den getroffenen Annahmen geschätzt auf 228.800 EUR externe Kosten plus 360 interne Tagewerke für den Aufwand zur Vorbereitung und Durchführung der Konsortiumsgründung sowie die Vorbereitung des Pilotprojektes. Dazu kommen die Kosten für die Durchführung des Piloten, die hier unter den oben beschriebenen Annahmen auf 770.000 EUR geschätzt werden, sowie die Kosten für den laufenden Betrieb des Konsortiums in einer geschätzten Höhe von 186.000 EUR. Für die Phase nach der Gründung wurden für den Aufwand der Begleitung des Pilotprojektes und der zusätzlichen Abstimmungen in der Aufbauphase des Konsortiumsbetriebes insgesamt 400 weitere interne Tagewerke angesetzt.

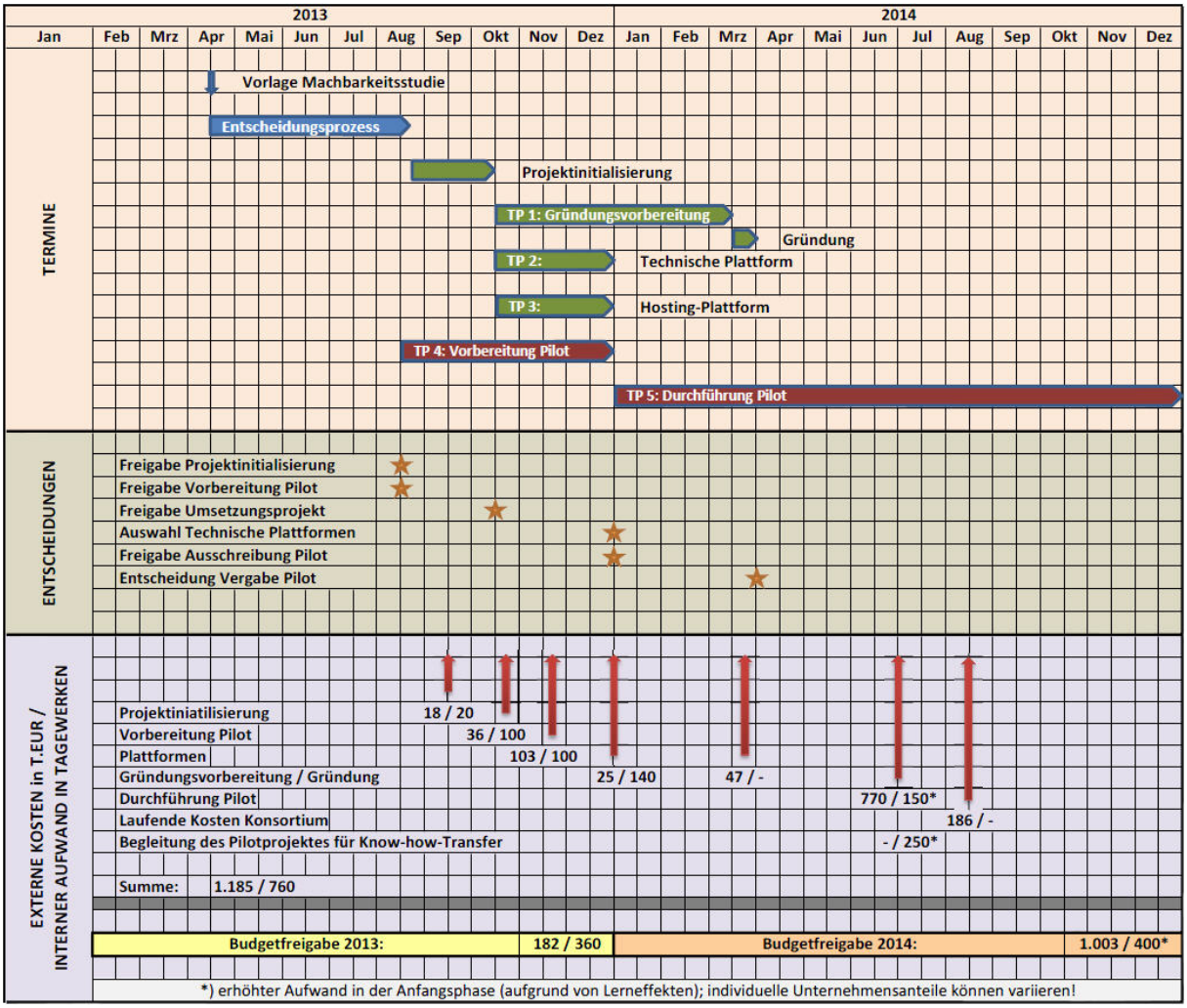


Abb. 6-2: Zeit- und Finanzierungsplan

6.3 Zusammenfassung

In diesem Kapitel werden die nächsten Schritte zur Umsetzung einer konsortialen Softwareentwicklung (KSE) nach Vorlage der Machbarkeitsstudie beschrieben.

Nach einer Phase zur Herbeiführung der Grundsatzentscheidung über das weitere Vorgehen wird ein Umsetzungsprojekt durchgeführt.

Dieses Umsetzungsprojekt gliedert sich unter den Rahmenbedingungen einer initial festzulegenden effizienten Projektorganisation in folgende fünf Teilprojekte, die den organisatorischen, finanziellen, rechtlichen und technischen Rahmen für die KSE abstecken sollen:

- Vorbereitung und Durchführung der Gründung
- Auswahl der technischen Plattform
- Auswahl der Hosting-Plattform
- Vorbereitung des Pilotprojektes
- Durchführung des Pilotprojektes.

Für die dargestellten Aktivitäten wurde ein Zeit- und Finanzierungsplan entwickelt, der die Fertigstellung des Piloten für spätestens Dezember 2014 vorsieht.

Die Gründung eines Konsortiums wird für April 2014 vorgeschlagen, damit das ausgewählte Pilotprojekt in der Verantwortung des Konsortiums realisiert werden kann.

Die Kosten für das Umsetzungsprojekt werden bei einer intensiven Beteiligung aller Unternehmen an den einzelnen Teilprojekten bis zur Gründung auf 228.800 EUR externe Kosten plus den Aufwand für 360 interne Tagewerke abgeschätzt. Diese Summen verteilen sich paritätisch auf die Teilnehmer des Umsetzungsprojektes.

Die Kostenschätzung für die Realisierung des Piloten ergibt externe Kosten in Höhe von 770.000 EUR zuzüglich 150 interne Tagewerke. Die geschätzten Kosten für den laufenden Betrieb des Konsortiums belaufen sich auf 186.000 EUR. Weitere interne Aufwände für die intensive Begleitung des Pilotprojektes und für zusätzliche Abstimmungen in der Aufbauphase des Konsortiumsbetriebes wurden mit 250 Tagewerken bemessen. Daraus ergeben sich externe Gesamtkosten von 1.184.800 EUR, die sich zusammen mit den internen Aufwendungen auf die betroffenen Unternehmen verteilen. Dieses "Risikokapital" wird vor dem Hintergrund der Aussagen in dieser Studie unter Berücksichtigung der wirtschaftlichen Chancen der KSE als durchaus tragbar angesehen.

In dem genannten Zeit- und Finanzierungsplan sind Meilensteine vorgesehen, in denen bewusst die Möglichkeit zur Überprüfung des Umsetzungsprozesses und der damit verbundenen Kostenentwicklung vorgesehen ist. An diesen Stellen besteht aus Sicht einzelner oder aller Unternehmen die Möglichkeit zur Korrektur oder sogar zum Projektausstieg. Außerdem erfolgt zum Ende des Umsetzungsprojektes eine abschließende Bewertung des gesamten Vorhabens.

Kapitel 7

Zusammenfassung und Empfehlungen

7 Zusammenfassung und Empfehlung

Der wirtschaftliche und sichere Betrieb von Versorgungsnetzen ist heute ohne die Prozessunterstützung durch leistungsfähige IT-Systeme nicht mehr denkbar. Neben den Systemen zur Führung von Bestandsplänen (Geografische Informationssysteme - GIS) und von Betriebsmitteldaten (Betriebsmitteldatenhaltung - BDH) nehmen die Netzleitsysteme eine Schlüsselstellung in der Systemlandschaft eines Unternehmens ein.

Bedingt durch die historische Entwicklung haben sich in der Beschaffung und Nutzung der Systeme verschiedene Abhängigkeiten und Schwachstellen ergeben, die einem wirtschaftlich optimalen und flexiblen Einsatz entgegenstehen. Aus diesem Grunde haben die sechs Auftraggeber dieser Machbarkeitsstudie entschieden, Alternativen zu der gegenwärtigen Praxis in der Beschaffung und im Einsatz von IT-Systemen zu suchen. Dabei wurden besondere Optimierungspotenziale darin gesehen, Software im Rahmen eines Konsortiums gemeinsam zu entwickeln und dabei konsequent den Einsatz von Open-Source-Software (OSS) zu verfolgen. Diese Studie hat das Ziel, diesen Lösungsansatz zu analysieren, zu bewerten und eine entsprechende Handlungsempfehlung auszusprechen. Dazu wurden

- in den beteiligten Unternehmen ausgewählte technische IT-Systeme und der in deren Umfeld vorhandene Softwarebedarf aufgenommen sowie die wesentlichen Schwachstellen im Einsatz der Systeme benannt
- zur Überwindung dieser Schwachstellen näher beleuchtet, ob die konsortiale Softwareentwicklung (KSE) auf Basis von OSS eine Alternative zur konventionellen Softwarebeschaffung darstellt
- wesentliche Anforderungen an die konsortiale Softwareentwicklung definiert
- untersucht, ob die technische Voraussetzungen für die Umsetzung der KSE vorhanden sind bzw. geschaffen werden können
- ein organisatorischer und rechtlicher Rahmen beschrieben
- die Wirtschaftlichkeit und Risiken bewertet
- ein Vorschlag für das weitere Vorgehen im Rahmen eines Umsetzungs- und Finanzierungsplan vorgelegt.

Zur Erarbeitung der Machbarkeitsstudie wurde ein Projektkernteam (PKT) gegründet, in dem alle Auftraggeber mit den jeweils für die Netzführung Verantwortlichen sowie die CONSULECTRA vertreten waren.

Für spezielle Fragen wurden fünf externe Experten aus den Bereichen Entwicklung von Open-Source-Software, Organisation von konsortialer Softwareentwicklung und IT-Recht punktuell in die Arbeit eingebunden.

Es wurden insgesamt elf PKT-Sitzungen sowie einige Workshops zu bestimmten Themen durchgeführt. Im Rahmen dieser Aktivitäten wurden die Studieninhalte sukzessive erarbeitet und dokumentiert.

Die vorliegende Machbarkeitsstudie stellt damit eine erfolgreiche gemeinsame Arbeit der beteiligten Unternehmen dar. Dadurch ist eine gute Basis für die Fortsetzung der Aktivitäten gelegt worden. Die Ergebnisse der Studie werden nachfolgend zusammengefasst.

Die Bestandsaufnahme ausgewählter IT-Systeme in den beteiligten Unternehmen ergab hinsichtlich der eingesetzten Produkte, der Eigentumsverhältnisse sowie der Zuständigkeiten für Datenpflege und Administration erwartungsgemäß ein relativ heterogenes Bild. Lediglich im Bereich der Netzleitsysteme zeigte sich durch die Nutzung des gleichen Produktes eine große Gemeinsamkeit. Aufgrund des hohen gemeinsamen Softwarebedarfs im Umfeld der Netzleitsysteme fokussierten sich die weiteren Betrachtungen auch auf dieses Segment.

In der Bestandsaufnahme zeigte sich weiterhin, dass es eine Reihe von wirtschaftlichen, qualitativen und operativen Schwachstellen bei der Beschaffung, Implementierung, dem Systembetrieb sowie dem Datenaustausch zwischen den Systemen gibt. Diese Defizite sind im Wesentlichen in der starken Herstellerbindung (Vendor-Lock-in-Effekt) begründet.

Die Analyse des KSE-Ansatzes ergab deutliche Potenziale die angestrebten Veränderungen

- Überwindung des Vendor-Lock-in-Effektes
- Qualitätsoptimierung (Fehlersuche, Sicherheitslücken etc.)
- Langzeitwartbarkeit der Software
- Modularität und Offenheit zukünftiger Softwarelösungen

nachhaltig zu erreichen. Der Einsatz von Open-Source-Software mit der Prämisse der freien Zugänglichkeit und Nutzung ist dabei der Schlüssel für eine Marktveränderung von einem Lieferanten- zu einem Kundenmarkt.

Zu den technischen Voraussetzungen für die Realisierung des KSE-Ansatzes gehört eine Projekt-Hosting-Plattform, auf der neben der erforderlichen Kommunikationsinfrastruktur auch die notwendigen Werkzeuge für die gemeinsame Softwareentwicklung bereitgestellt werden, die den formulierten Anforderungen entsprechen. Hierfür kann z. B. auf Dienstleister (Provider) und vorhandene Open-Source-Lösungen zurückgegriffen werden, was für die Startphase des KSE-Betriebes auch die empfohlene Variante ist.

Als mögliche Basis für den Einsatz der entwickelten Softwaremodule bei den Anwendern wurden weiterhin Empfehlungen für die Auswahl einer technischen Plattform ausgesprochen, die dem zukunftsorientierten Ansatz einer service-orientierten Architektur folgt und den Anforderungen und Zielen des Konsortiums ebenfalls entspricht.

Als weitere Randbedingung für die Umsetzung des KSE-Modells wurde beleuchtet, welche der verschiedenen Software-Entwicklungsmethoden, deren Spektrum von klassischen, plangetriebenen bis zu modernen agilen Ansätzen reicht, für das Vorhaben am geeignetsten erscheinen. Hier hat sich gezeigt, dass die Anwendung agiler Methoden für die konsortiale Softwareentwicklung aufgrund der Transparenz im Entwicklungsprozess klar zu favorisieren ist. Je nach Projektinhalt sind aber auch Kombinationen aus plangetriebenen und agilen Methoden denkbar.

Um dem Konsortium ein angemessenes Gewicht und die notwendige Verbindlichkeit zu verleihen sowie die Eigentumsrechte an Software zu sichern, ist es notwendig, eine Rechtsform zu finden, die über die lose Kooperation hinausgeht. Nach einem Variantenvergleich unter Einbeziehung von externer juristischer Expertise wurde hierfür die Gründung einer eingetragenen Genossenschaft empfohlen. Als wesentlicher Eckpfeiler für den wirtschaftlichen und operativen Geschäftsbetrieb ist im Rahmen des Gründungsprozesses eine Satzung zu formulieren. Dafür wurde dieser Studie ein beispielhaftes Muster beigefügt.

Eine wesentliche Voraussetzung für die erfolgreiche Umsetzung des KSE-Modells ist die Schaffung eines sogenannten Ökosystems. Es besteht aus Vertretern der Anwender im Konsortium und leistungsfähigen Dienstleistern (z. B. Softwarefirmen, einzelne Entwickler, Berater), in deren Zusammenspiel die Softwareerstellung und Qualitätssicherung von Softwaremodulen im Rahmen eines definierten Entwicklungsprozesses umgesetzt wird. Diese Dienstleister können neben einer hinreichend großen Anzahl von Anwendern genau wie Hochschulen auch Mitglieder im Konsortium sein. Sie können nach der Bereitstellung der konsortialen Software von den Anwendern auch mit der Implementierung, Schulung, Customizing und den Support beauftragt werden. Der KSE-Ansatz ist nur dann nachhaltig tragfähig, wenn es auf diesem Weg dauerhaft gelingt, für eine ausreichend große Zahl von Dienstleistern einen interessanten Markt aufzuspannen.

Die Weitergabe, Veränderung und Weiterentwicklung der Software wird unter bestimmte Lizenzen gestellt, die dafür unterschiedliche Restriktionen vorsehen. Die jeweilige Lizenz wird durch den Eigentümer des Urheberrechts vergeben. Aus diesem Grund ist es notwendig, dass das Konsortium möglichst vollständig Eigentümer dieser Rechte ist bzw. diese übertragen bekommt. Diese Studie enthält eine vergleichende Übersicht über die einzelnen Lizenzformen. Eine konkrete Festlegung der Lizenz kann projektbezogen erfolgen. Allerdings wird empfohlen, grundsätzlich Lizenzen mit sogenanntem schwachem Copyleft-Effekt auszuwählen und dabei auf eine starke Marktverbreitung dieser Lizenz zu achten.

Aus den bisherigen Betrachtungen wird der Schluss gezogen, dass der KSE-Einstieg als Alternative zur heutigen Praxis bei Betreibern von Versorgungsnetzen attraktiv ist und dafür die technischen, organisatorischen und rechtlichen Rahmenbedingungen geschaffen werden können. Das heißt, die Machbarkeit ist gegeben.

Um aber eine abschließende Beurteilung zu ermöglichen, wurde der Wirtschaftlichkeitsaspekt als ein weiterer Bestandteil der Studie systematisch untersucht. In einem ersten Teil wurden die Kosten für die Gründung und den Betrieb eines Konsortiums analysiert. Anschließend wurde ein modellhafter Kostenvergleich zwischen dem KSE-Ansatz und dem konventioneller Softwarebeschaffung durchgeführt.

Dieser Vergleich zeigt ein klares Potenzial für Einsparungen zugunsten der KSE, die die laufenden Kosten für den Konsortiumsbetrieb deutlich übersteigen können. Dieser Effekt ist umso größer, je stärker das Konsortium bezüglich Mitgliederzahl und Entwicklungsaktivitäten wächst. Ein signifikantes Wachstums sollte daher ein wichtiges strategisches Grundelement im Konsortium sein.

Die Möglichkeiten zur Kostenentlastung durch Fördermaßnahmen und andere Einkünfte verbessern die Wirtschaftlichkeit und sollten konsequent genutzt werden, sind aber keine Bedingung für eine Entscheidung zum Einstieg in die KSE.

Die Wirtschaftlichkeitseinschätzung zugunsten der KSE verstärkt sich deutlich durch weitere Vorteile, die sich in unterschiedlichen Formen der Effizienzsteigerung in den Arbeits- und IT-Prozessen und der Weiternutzung von Altsystemen unternehmensindividuell ergeben und daher auch nicht allgemeingültig abgeschätzt werden konnten. Diese Vorteile sind im Wesentlichen bereits im Zusammenhang mit den Eigenschaften von Open-Source-Software oben genannt worden. Sie stellen eine wesentliche Basis dar, um den zukünftigen Anforderungen an Netzbetreiber im regulierten Umfeld mit wachsendem Kostendruck und höherer Innovationsgeschwindigkeit gerecht werden zu können.

Bei der Beurteilung der Wirtschaftlichkeitsanalyse ist zu berücksichtigen, dass besonders die Abschätzung einzelner Kostenpositionen trotz der Einbeziehung externer Experten mit Unsicherheiten behaftet ist.Trotzdem lassen die dargestellten Ergebnisse den Schluss zu, dass die KSE auch unter Berücksichtigung der identifizierten Risiken große wirtschaftliche Chancen bietet. Die Entscheidung für eine Umsetzung ist daher gerechtfertigt und besonders unter einem strategischen Aspekt zu sehen.

Diese Grundsatzentscheidung vorausgesetzt wird zur weiteren Vorgehensweise vorgeschlagen, ein Umsetzungsprojekt durchzuführen, in dessen Verlauf ein Konsortium gegründet und ein Pilotprojekt durchgeführt werden. Hierdurch wird der Nachweis geführt, dass der konsortiale Ansatz auch in der Realität trägt.

Epilog

Projektaktivitäten nach Beendigung der Machbarkeitsstudie

STAND: 15.11.13

INHALTSVERZEICHNIS

8.1 Entscheidungsprozesse und Projektplanung für eine geplante Realisierung

Zu den erforderlichen Entscheidungsprozessen fand am 3. September 2013 in Frankfurt eine Sitzung mit den verantwortlichen Entscheidern aus den beauftragenden Unternehmen statt. Die dort anwesenden Geschäftsführer entschieden, dass ein Vorprojekt eine weitere Realisierung bis Mitte 2014 vorbereitet.

8.2 Bewerbung weiterer Interessenten

In der vorliegenden Machbarkeitsstudie wurde nachgewiesen, dass die wirtschaftliche und technologische Attraktivität der Idee konsortial Software auf der Basis von OpenSource zu entwickeln entscheidend von der Bewerbung weiterer Interessenten abhängt. Darüber hinaus wurden in den Kapiteln 4 und 5 Empfehlungen zu einer geeigneten Rechtsform gegeben sowie die wirtschaftlichen Konsequenzen der Idee skizziert.

Die Mitwirkungsmöglichkeiten bei dieser Idee sind vielfältig, so dass sich der Bewerberkreis sehr heterogen entwickeln kann. Die Aktivitäten der Unternehmen, die sich weiterhin der Verbreitung der OSS-Idee widmen, gehen also zielgerichtet dahin, weitere Interessenten zu gewinnen. Die ARGE sieht Potenziale in den Sparten:

- Netzbetreiber
- Netzservice
- Softwareentwickler in der Branche Energiewirtschaft
- Dienstleister in ebendieser Branche z. B. im Hosting oder Plattformanbieter
- Forschungsinstitute und Forschungsgemeinschaften innerhalb des universitären und nicht universitären Umfeldes
- Consulting-Unternehmen
- bereits bestehende OSS-Communities, die ihre Geschäftsfelder erweitern möchten
- Interessierte und fachkundige Einzelpersonen

8.3 Erschließung von Förderungsmitteln

Die EU sowie viele Ministerien auf Bundes- und Länderebene unterstützen in vielfältiger Weise innovative Ideen.

Derzeit gibt es mehrere Förderinitiativen auf europäischer und nationaler Ebene, die für das beschriebene Vorhaben als Quelle für Fördergelder in Frage kommen. Deshalb wurden innerhalb des Projektes KONSEQUENZ intensiv die Fördermöglichkeiten und die Frage der Förderungsfähigkeit der OSS-Idee in der Energiewirtschaft diskutiert und geprüft.

Die Projektgruppe entschloss sich daraufhin im Frühjahr 2013 im Rahmen der Förderungsinitiative „Zukunftsfähige Stromnetze“, eine Förderskizze beim verantwortlichen Träger einzureichen. Eine Entscheidung, ob Fördermittel bewilligt werden, steht derzeit noch aus.

Anhang A

INHALTSVERZEICHNIS

QUELLENVERZEICHNIS

ABBILDUNGS- UND TABELLENVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

NOMENKLATUR

Quellenverzeichnis

[2-1] URL: http://de.wikipedia.org/wiki/Lock-in-Effekt
[Stand 2012-11-15]

[2-2] Seibt, Richard (2010):
Foliensatz "Megatrend Open Source - Warum?" vom 29.04.2010
http://www.begeisterungsmomente.de/richard-seibt.htm
[Stand 2012-11-15]

[2-3] OSBF e.V. (2008)
COSAD-Whitepaper, deutsche Fassung
URL: http://www.osbf.eu/de/global/unsere_projekte/cosad/
[Stand 2012-10-15]

[3-1] http://www.osbf.eu/fileadmin/OSBF-Prinzipien_einer_OpenCloud_vfinal.pdf
[Stand 2013-04-02]

[3-2] http://polarsys.org/topcased-migrates-polarsys
[Stand 2013-04-02]

[3-3] http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Vorgehensmodell
Enzyklopädie der Wirtschaftsinformatik – Online-Lexikon
Hrsg.: Karl Kurbel, Jörg Becker, Norbert Gronau, Elmar Sinz, Leena Suhl
[Stand 2013-04-02]

[3-4] Hochschule Koblenz – University of Applied Scienses, BPM-Labor
Prof. Dr. Ayelt Komus (2012):
Studienbericht Status Quo Agile

[3-5] https://www.bsi.bund.de/DE/Themen/weitereThemen/FreieSoftware/index_htm.html
[Stand 2013-04-02]

[4-1] Riehle, Dirk (2012):
Ein Rahmenwerk für gemeinschaftlichen Open Source in der Energiewirtschaft
Arbeitsversion vom 2012-04-04, S. 5.

[4-2] URL: http://www.opensource.org/osd
[Stand 2012-11-15]

[4-3] Schmitz, Patrice-Emmanuel (2007 & 2009): European Union Public Licence, v.1.1, Leitlinien für Anwender und Entwickler. Seite 4 ff. Online:
http://joinup.ec.europa.eu/software/page/eupl/eupl-guidelines
[Stand 2012-10-09]

[4-4] Bundesverwaltungsamt (BVA) - Kompetenzzentrum Open Source Software
URL: http://www.oss.bund.de/node/123
[Stand 2012-10-09]

[4-5] URL: http://www.datenschutz-praxis.de/lexikon/h/hinterlegungsvereinbarung.html
[Stand 2012-10-09]

[4-6] URL: http://de.wikipedia.org/wiki/Open_Source
[Stand 2012-10-09]

[4-7] URL: http://de.wikipedia.org/wiki/Copyleft
[Stand 2012-10-09]

[4-8] ifrOSS - Institut für Rechtsfragen der Freien und Open Source Software
URL: http://www.ifross.org/lizenz-center
[Stand 2012-10-09]

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BDEW	Bundesverband der Energie- und Wasserwirtschaft e.V.
BDH	Betriebsmitteldatenhaltung
BNetzA	Bundesnetzagentur
BIS	Betriebsmittelinformationssystem
BSI	Bundesamt für Sicherheit in der Informationstechnik
COGS	Costs Of Goods Sold (Fertigungs- bzw. Umsatzkosten)
DB	Datenbank
DKE	Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE
EAM	Enterprise Architecture Management
EEG	Erneuerbaren-Energien-Gesetz
ESB	Enterprise Service Bus
EVU	Energieversorgungsunternehmen
G&A	General & Administrative (englisch für: allgemeine Verwaltungskosten)
GIS	Geografisches Informationssystem
IDE	Integrated Development Environment (Integrierte Entwicklungsumgebung)
IaaS	Infrastructure-as-a-Service
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers ("i triple e" [ai trɪpl i:] gesprochen)
IKT	Informations- und Kommunikationstechnik
IP	Intellectual Property (geistiges Eigentum)
ISO	International Organization for Standardization
IT	Informationstechnik
KSE	Konsortiale Softwareentwicklung
MQ	Message Queuing
NLS	Netzleitsystem
OCBI	Open Cloud Business Initiative

OSBF	Open Source Business Foundation e.V.
OSI	Open Source Initiative
OSS	Open-Source-Software
PaaS	Platform-as-a-Service
R&D	Research & Developement
SaaS	Software-as-a-Service
SLA	Service Level Agreement
SOA	Service Oriented Architecture
ÜNB	Übertragungsnetzbetreiber
VDE	VDE Verband der Elektrotechnik Elektronik Informationstechnik e.V.
WFM	Workforcemanagement (-system)
XP	Extreme Programming

Nomenklatur

Allianzmanagement

umfasst eine Reihe von Praktiken und Werkzeugen, die angewendet werden, um Beziehungen mit Partnern zu pflegen, zu verstärken und zu verfestigen. Allianzmanagement hat zum Ziel, aus den Beziehungen zu Partnern einen Wertbeitrag für die eigene Organisation zu generieren.

Apache Software Foundation

ist eine ehrenamtlich arbeitende Organisation zur Förderung der Apache-Softwareprojekte (http://www.apache.org/).

Betriebsmitteldatenhaltung

beschreibt im Kontext dieser Studie, die Gesamtheit der Systeme in der Stamm- und Bewegungsdaten zu Netzbetriebsmitteln gehalten werden.

Bugtracker

siehe Bug tracking system

Bug tracking system

ist ein Fallbearbeitungssystem (engl. trouble ticket system) für die Softwareentwicklung, das als Werkzeug eingesetzt wird, um Programmfehler zu erfassen und zu dokumentieren. Es wird in vielen Projekten als zentrales, öffentliches Archivierungssystem eingesetzt, um Empfang, Bestätigung, Klassifizierung und Bearbeitung von Fehlern und sonstigen Vorgängen zu handhaben. Bekannte Produkte sind Jira (kommerziell) und Bugzilla (open source).

Code Repository

ist ein schreibgeschützter Aufbewahrungsort für den aktuellen (oder auch einen älteren) Stand aller Code-Zeilen eines Projekts, von dem aus Arbeitskopienen gezogen werden können. Das Übertragen einer Version aus dem Repository in eine Arbeitskopie wird als Checkout, Auschecken oder Aktualisieren bezeichnet, während die umgekehrte Übertragung einer modifizierten Arbeitskopie oder neuen Beitrags (Contribution) als Check-in, Einchecken oder Commit definiert wird.

Committer

ist jemand, der Veränderungen an dem Code-Repository vornehmen darf. Dies kann durch eigenes modifizieren oder durch Aufnahme von Code-Beiträgen (Contributions) nach Überprüfung geschehen.

Contribution

ist ein freiwilliger Beitrag zu einem Open-Source-Projekt. Es kann sich dabei um Programmierungen (Code-Zeilen) sowie Dokumentationen zu den Software-Produkten und Übersetzungen, aber auch um Geldschenkungen handeln.

Contributor

ist jemand, der einen Beitrag zu einem Open-Source-Projekt leistet.

Copyleft

ist eine Klausel in urheberrechtlichen Nutzungslizenzen, die festschreibt, dass Bearbeitungen des Werkes nur dann erlaubt sind, wenn alle Änderungen ausschließlich unter den identischen oder im Wesentlichen gleichen Lizenzbedingungen weitergegeben werden. Sie verhindert, dass veränderte Fassungen des Werkes mit Nutzungseinschränkungen weitergegeben werden, die das Original nicht hat.

Copyleft ist ein Wortspiel, das einen Gegensatz zum englischen Begriff Copyright (deutsch Urheberrecht, wörtlich: "Kopierrecht") durch Ersetzen von "rechts" (engl. right) durch das Gegenwort "links" (engl. left) konstruiert, wobei left gleichzeitig eine grammatische Form von "(über)lassen" ist. Analog zur Vertauschung der beiden Begriffe wird auch die (urheberrechtliche) Situation umgekehrt.

Customizing

ist ein Anglizismus, der überwiegend im deutschen Sprachraum verwendet wird. Customizing (engl. to customize = anpassen, der korrekte englische Begriff ist customization) ist der Ausdruck für die Anpassung eines Serienproduktes an die Bedürfnisse eines Kunden.

Eclipse

ist eine integrierte Entwicklungsumgebung (IDE) für die Programmiersprache Java. Es existieren aber auch etliche Plug-ins für andere Sprachen.

Eclipse Foundation

ist eine gemeinnützige Gesellschaft mit der Aufgabe, die Eclipse-Open-Source-Gemeinschaft und ihre Projekte zu leiten. Die Eclipse Foundation koordiniert darüber hinaus alle komplementären Produkte und Dienstleistungen, die im internen Sprachgebrauch auch als das Ökosystem bezeichnet werden.

Einspeisemanagement (Anforderungen gemäß §11 EEG)

ist die Reduzierung der Einspeiseleistung bei Netzüberlastung. Dies umfasst sowohl thermische Überlastungen als auch Netzengpässe, die zur Nichteinhaltung der Netzspannung führen.

Extreme Programming

ist eine agile Softwarentwicklungsmethode, die das iterative Lösen einer Programmieraufgabe in den Vordergrund der Softwareentwicklung stellt und dabei einem formalisierten Vorgehen geringere Bedeutung zumisst.

GIT

ist eine freie Software zur verteilten Versionsverwaltung von Dateien, die ursprünglich für die Quellcode-Verwaltung des Linux-Kernels entwickelt wurde.

Governance

bezeichnet allgemein das Steuerungs- und Regelungssystem im Sinn von Strukturen (Aufbau- und Ablauforganisation) einer Organisation.

Intellectual Property

Als geistiges Eigentum (engl. intellectual property, kurz IP) eines Menschen wird all jenes Wissen und Kulturgut bezeichnet, das dieser sich durch geistige Anstrengungen wie Lernen, Forschen, Nachdenken, Lesen oder auch Diskutieren zu eigen gemacht hat. Der Begriff wird außerdem als Jargon für Urheberrechte und gewerbliche Schutzrechte verwendet.

Intellectual Property Rights

sind Rechte am geistigen Eigentum = Urheberrechte (z. B. an Software). Jedes IP-Recht gibt dem Inhaber exklusive Besitz-, Eigentums-, Verwendungs- und Verwertungsrechte. IP-Rechte sind insoweit monopolistische Rechte und dienen dem eigenen Produktschutz und -absatz.

Industry framework

ist ein domänenspezifisches Integrationsframework (Rahmenwerk), das speziell für einen Industriebereich (bzw. eine Branche) vorgefertigte Adapter (Konnektoren) bereitstellt, um mit geringem Aufwand IT-Anwendungen mit unterschiedlichen technischen Schnittstellen verbinden zu können,

Internetdiensteanbieter

oder Internetdienstleister (engl. Internet Service Provider, abgekürzt ISP) werden im deutschsprachigen Raum auch oft nur Provider, weniger häufig auch nur Internetanbieter oder Internetprovider genannt. Provider sind Anbieter von Diensten, Inhalten oder technischen Leistungen, die für die Nutzung oder den Betrieb von Inhalten und Diensten im Internet erforderlich sind.

Issue tracking system

siehe Bug tracking system

Hosting

(deutsch "Gastgeber sein") ist die Unterbringung von Internetprojekten, die sich in der Regel auch öffentlich durch das Internet abrufen lassen. Diese Aufgabe übernehmen Internet-Dienstleistungsanbieter (Provider oder Webhoster), die Web-Speicher, Datenbanken, E-Mail-Adressen und weitere Produkte anbieten und zum Austausch von Daten durch das Internet dienen. Diese Anbieter legen üblicherweise auf ihren Webservern die durch den Kunden, z. B. per Secure File Transfer Protocol (SFTP) oder File Transfer Protocol (FTP), hochgeladenen Websites ab. Sie bieten ggf. auch die Registrierung von Domains und deren Bekanntmachung per Domain Name Service.

Hostinganbieter

siehe Internetdiensteanbieter

Lastmanagement (gemäß EnWG)

umfasst die Steuerungen von Lastflüssen für die Gewährleistung der Sicherheit und Zuverlässigkeit zwischen Netzbetreibern und Anbietern in der jeweiligen Regelzone gemäß § 13 Abs. 4a EnWG sowie Abschaltungen bzw. Lastreduzierung im Sinne § 14a EnWG mit dem Ziel der Flexibilisierung/Optimierung der Netznutzung von Verbrauchseinrichtungen zur Vermeidung von Lastspitzen in Verteilnetzen.

Middleware

ist eine zusätzliche Schicht in einer komplexeren Software-Struktur, deren Aufgabe es ist, die Zugriffsmechanismen auf unterhalb angeordnete Schichten zu vereinfachen und die Details von deren Infrastruktur nach außen hin zu verbergen.

Migration

geht über eine einfache Aktualisierung bzw. ein Upgrade hinaus und bezeichnet vielmehr einen grundlegenden Wechsel.

Open-Source-Software

nennt man Software, deren Lizenzbestimmungen in Bezug auf die Weitergabe der Software besagen, dass der Quellcode öffentlich zugänglich ist und - je nach entsprechender Lizenz - frei kopiert, modifiziert und verändert wie unverändert weiterverbreitet werden darf.

Persistenz

ist in der Informatik der Begriff, der die Fähigkeit bezeichnet, Daten (oder Objekte) oder sogenannte logische Verbindungen über lange Zeit bereitzuhalten.

Proprietär(-e Software)

Man bezeichnet im IT-Bereich traditionell solche Dateiformate, Protokolle usw. aber auch Hardware als proprietär:

- wenn sie nicht oder nur mit Schwierigkeiten von Dritten implementierbar und deshalb nicht zu öffnen oder zu lesen sind, weil sie z. B. lizenzrechtlich, durch herstellerspezifisches Know-how oder durch Patente beschränkt sind
- wenn sie nicht allgemein anerkannten Standards entsprechen, also sozusagen "hauseigene" Entwicklungen sind.

Proprietäre Formate werden auch oft als zusätzliche Einnahmequelle von Lizenzgebühren genutzt. Proprietär wird in Bezug auf Soft- und Hardware verwendet, um diese zu freier Software und freier Hardware abzugrenzen.

Provider

siehe Internetdiensteanbieter

Quellcode

ist in der Informatik der für Menschen lesbare, in einer Programmiersprache geschriebene Text eines Computerprogrammes.

Quellcodeverwaltung

ist ein Versionsverwaltungssystem, dass typischerweise in der Softwareentwicklung eingesetzt wird, um Quelltexte zu verwalten. Alle Versionen werden in einem Archiv mit Zeitstempel und Benutzerkennung gesichert und können jederzeit wiederhergestellt werden.

Release

bezeichnet die fertige und veröffentlichte Version einer Software. Damit geht eine Veränderung der Versionsbezeichnung, meist ein Hochzählen der Versionsnummer einher.

Repository

(engl. für Lager, Depot, Quelle oder Archiv) ist ein verwaltetes Verzeichnis zur Speicherung und Beschreibung von digitalen Objekten. Bei den verwalteten Objekten kann es sich beispielsweise um Programme (Software-Repository), Publikationen (Dokumentenserver) oder Datenmodelle (Metadaten-Repository) handeln. Häufig beinhaltet ein Repository auch Funktionen zur Versionsverwaltung der verwalteten Objekte.

Requirements analysis

Die Anforderungsanalyse (engl. requirements analysis) ist in der Informatik ein Teil des Systementwicklungsprozesses (u. a. neben dem Anforderungsmanagement) sowie ein Teil der Business-Analyse. Ziel ist es, die Anforderungen des Auftraggebers an das zu entwickelnde System zu ermitteln, zu strukturieren und zu prüfen.

Requirements-Team

ist eine zumeist interdisziplinär und aus verschiedenen Stakeholdern zusammengesetzte Gruppe von Experten, deren Aufgabe es ist, die Anforderungen an das zu entwickelnde System zu dokumentieren.

Service Level Agreement

ist eine Vereinbarung zwischen einem IT-Service-Provider und einem Kunden.

Service-Provider

siehe Internetdiensteanbieter

Sourcecode

siehe Quellcode

Tailoring

(deutsch Maßschneidern) ist die (schnelle) Anpassung des Softwareentwicklungsprozesses an die Rahmenbedingungen des Projektes.

Upgrade

(engl. upgrade = hochstufen/aufrüsten) bezeichnet die Änderung eines Produktes auf eine höherwertige Konfiguration oder Version.

"Vendor-Lock-in-Effekt"

beschreibt die Abhängigkeit von einem Hersteller und seinem momentan eingesetzten Produkt, die eine Änderung der aktuellen Situation aufgrund hoher Wechselkosten unwirtschaftlich machen. Die Höhe der Wechselkosten bestimmt das Ausmaß des Lock-in-Effektes.

Web-Hosting

ist die Unterbringung (Hosting) von Webseiten auf einem Webserver eines Internetdienstanbieters. Der Webhoster genannte Provider stellt - üblicherweise gegen Bezahlung - seine Ressourcen zur Verfügung. Zu diesen Ressourcen gehören insbesondere die Bereitstellung und der Betrieb von Webservern und deren Netzwerkanbindung. Der Leistungsumfang von Web-Hosting-Angeboten variiert erheblich. Die Angebote beginnen mit einer einfachen Webpräsenz über Server mit Skriptsprachenunterstützung und Datenbank-Backend bis hin zu Paketen, die ein Web-Content-Management-System sowie Monitoring, Datensicherung, statistische Auswertungen, Lastverteilung oder Hochverfügbarkeit beinhalten.

Anhang B

INHALTSVERZEICHNIS

Anlage 3-1

Ergebnisfoliensatz WS Architektur und Plattform

(vom 23.07.2012)

CONSULECTRA
Kompetenz in Energie

Projekt KONSEQUENZ
Workshop „Architektur und Plattform“

Hamburg, 23. Juli 2012
Hendrik Höfer (microdoc)
Christian Meyer (CONSULECTRA)

CONSULECTRA

Agenda

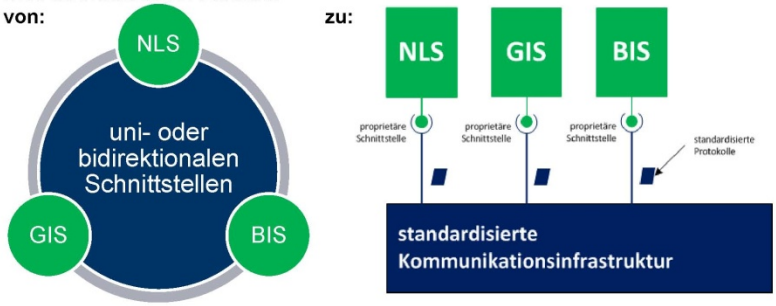
- **Grundsätzliche Überlegungen**
  - zur benötigten SW-Infrastruktur
  - zur Auswahl von einsetzbaren OSS-Produkten
  - zum SW-Entwicklungsprozess im Konsortium
- **Konkretisierung des Softwarebedarfs (Workshop Teil 1)**
- **Entwicklungsschritte für ein erstes Pilotprojekt (Workshop Teil 2)**
- **weitere Aspekte**
  - Kriterien für die Auswahl einer Hosting-Plattform
  - mögliche Dienstleistungen einer Hosting-Plattform
  - Bereitstellung der Produkte (Release1.n)
  - Implementierung
  - ggf. Customizing

 Seite 2 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur SW-Infrastruktur (1/4)

- Statt für die Umsetzung einer konkreten neuen Lösung in einer heterogenen und oft proprietären Umgebung jedes System mit jedem anderen zu verbinden (1:n Schnittstellen), sollten alle Systeme über einen gemeinsamen Kanal kommunizieren (1 Schnittstelle)
- Dies erfordert einen Wechsel von: zu:



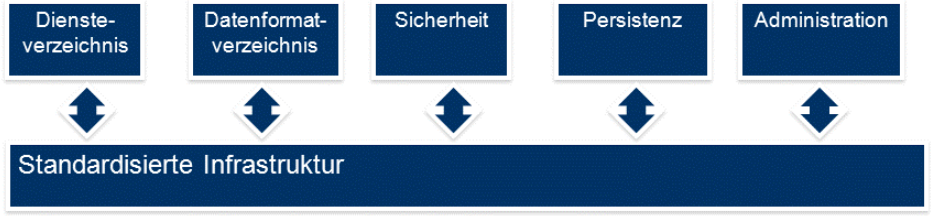
 Seite 3 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur SW-Infrastruktur (2/4)

- Die innerhalb der standardisierten Kommunikationsinfrastruktur auszuwählenden Bestandteile sind:
  - Datenbank (DB)
  - Message Queuing (MQ)
  - Enterprise Service Bus (ESB)

- Die technische Infrastruktur sollte die folgenden Komponenten für Standardfunktionen sowie Schnittstellen für unterschiedliche Technologien (z. B. C, .NET, Java, C++) beinhalten:

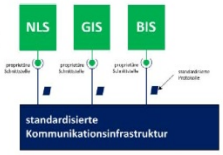


 Seite 4 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur SW-Infrastruktur (3/4)

- Die Infrastruktur zur Anbindung und zum Management der Schnittstellen muss weitestgehend standardisiert sein
- Die unter den einzelnen Systemen ausgetauschten Datenformate müssen ebenfalls weitestgehend standardisiert werden, um Skaleneffekte zu erreichen



- Der Einsatz konkreter Systeme kann, muss aber nicht, standardisiert sein
- Durch seine größere Komplexität gegenüber monolithischen Programmstrukturen erfordert die Entwicklung einer solchen serviceorientierten Architektur (SOA) einen höheren Initialaufwand und spielt ihre Einsparungen erst dann aus, wenn grundlegende Services bereits existieren und in breiteren Anwendungsgebieten eines Unternehmens genutzt werden können.

 Seite 5 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur SW-Infrastruktur (4/4)

- Die Möglichkeit für synchrone und asynchrone Kommunikation sollte vorgesehen werden
- Die Architektur sollte eine Klasse von unterschiedlichen Anforderungen abdecken und nicht spezifisch für eine einzelne Lösung sein

- Die Architektur sollte sich an den „Best Practices“ in der Energiewirtschaft und ähnlichen Industrien orientieren
- Die Architektur sollte weitgehend mit Standardprodukten implementierbar sein und Neuimplementierung von technischen Komponenten vermeiden
- Die Architektur sollte mit vorhandenen und neuen (Standard-) Technologien integrierbar sein
- Aber: Auch der Einsatz von spezialisierten Technologien für klar abgegrenzte Funktionen kann sinnvoll sein

 Seite 6 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur Auswahl von OSS-Produkten (1/2)

- Auswahlkriterien:
  - Die Produkte sollten (weitgehend) unter einer Open Source Lizenz stehen oder es sollte eine Open Source Alternative existieren (z. B. Oracle vs. MySQL)
  - Die eingesetzten Produkte müssen über viele Jahre „supported“ sein (ggf. kommerziell)
  - Die eingesetzten OSS-Produkte sollten eine „große“ Community haben und nicht von einer einzelnen Firma entwickelt werden
  - Idealerweise gibt es eine Reihe von Unternehmen, die Dienstleistungen für die Produkte anbieten

 Seite 7 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur Auswahl von OSS-Produkten (2/2)

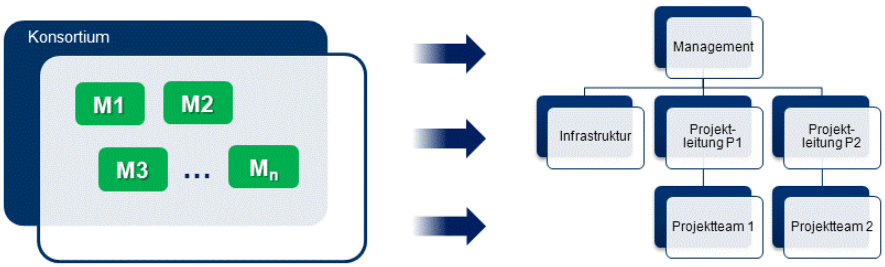
- Nichtfunktionale Anforderungen:
  - Die Architektur sollte auf Standardtechnologien basieren (z. B. JEE oder .NET im Gegensatz zu Erlang oder node.js)
  - Die Infrastruktur beinhaltet Fehlertracking, Wiki, Projektplanung, Code Repositories und Buildsysteme und wird zentral zur Verfügung gestellt
  - meist werden quelloffene Lösungen bevorzugt (Bugzilla, Mediawiki, Jira, Confluence)
  - Die Architektur/Produkte müssen:
    - in einer Standardumgebung betreib- und administrierbar sein
    - (sehr) große Datenmengen verarbeiten können
    - fehlertolerant sein und Clustering/Failover erlauben
    - echtzeitfähig sein

 Seite 8 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zur Organisation

- nur wenige Projekte ähneln in ihrer Organisation der von Standardprojekten
- bei vielen Projekten sind weitere Funktionen, wie z.B. zentrales Architektur- und Anforderungsmanagement notwendig
- die Organisation muss aus Entwicklungssicht keine eigene rechtliche Entität sein



- das Personal sollte von den Mitgliedern des Konsortiums gestellt werden

 Seite 9 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zum SE-Prozess

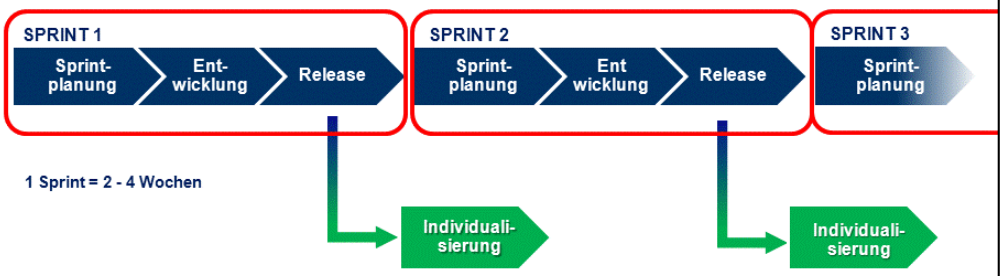
- Einige Eigenschaften eines konsortialen Entwicklungsprozesses
  - offen und transparent – der Zustand der Entwicklung ist für alle Beteiligten jederzeit einseh- und bewertbar
  - Einfluss gemäß Beitrag (z.B. Ressourcen, Geld)
  - nicht diskriminierend
  - Entwicklung und Projektsteuerung werden durch die Mitglieder des Konsortiums durchgeführt (keine Lieferanten-Kunden-Beziehung)
  - es wird eine agile, time-boxed Methodik verwendet (z.B. Scrum)

 Seite 10 23/07/2012

CONSULECTRA

Grundsätzliche Überlegungen zum SE-Prozess

- Die konsortiale Entwicklung erfolgt iterativ, time-boxed planbar und verlässlich



- Nach jedem Release kann der Stand ausgekoppelt und für eine bestimmte Umgebung individualisiert werden
  - Anpassung an die eigenen Systeme
  - Integration von proprietärem Code

 Seite 11 23/07/2012

CONSULECTRA

 Seite 12 23/07/2012

CONSULECTRA

Konkretisierung des Softwarebedarfs (Workshop Teil 1)

- Kriterien für die Auswahl von OSS-SE-Themen:
  - noch nicht oder nur in Teilen als kommerzielle Software verfügbar
  - „sexy“ und von strategisch hoher Bedeutung für
    - Konsortiumsmitglieder
    - weitere potenzielle Mitglieder des Konsortiums
  - förderungsfähig sein

 Seite 13 23/07/2012

CONSULECTRA

Einspeisemanagement

- Unter der Überschrift „Einspeisemanagement“ werden von den Netzführern folgende Themen zusammengefasst:
  - (Netz-, Kurzfrist-, EEG-) Prognosen
  - Schalt- und Wertesimulation
  - „Netzberechnung“, Netzzustandsrechnung, State Estimation
  - Auswahl und Steuerung der Einspeiser und abschaltbaren Lasten
  - Verwaltung und Dokumentation der Eingriffe

 Seite 14 23/07/2012

CONSULECTRA

Einspeisemanagement

Ermittlung der funktionalen Anforderungen

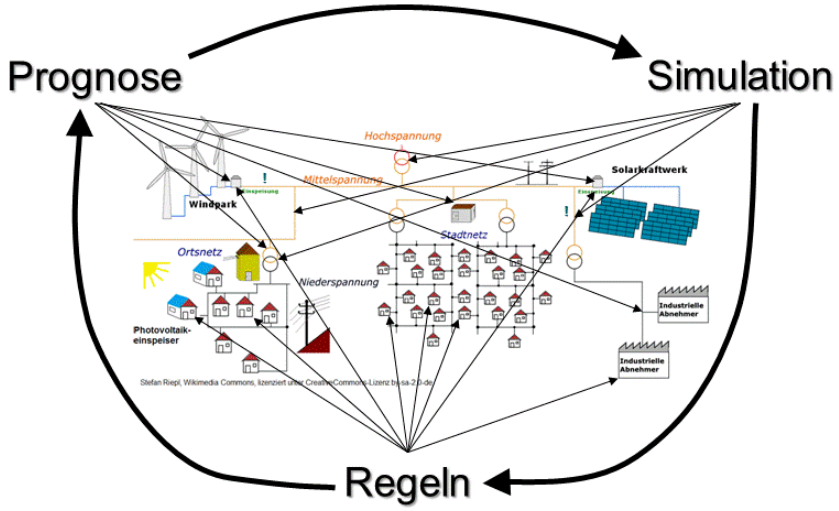
- Was wird bereits heute gemessen? In welchem Zyklus?
- Was soll zukünftig zusätzlich gemessen werden? In welchem Zyklus?
- Was wird aus den Messdaten berechnet?
- Welche weiteren Daten lassen sich mit den Messdaten verschneiden?
  - für welche Berechnungen?
  - für welche Visualisierungen?
- Auf welcher Basis soll was gesteuert werden?
- Was muss wie dokumentiert und übergeben werden?

 Seite 15 23/07/2012

CONSULECTRA

Einspeisemanagement

Was bedeutet Einspeisemanagement?

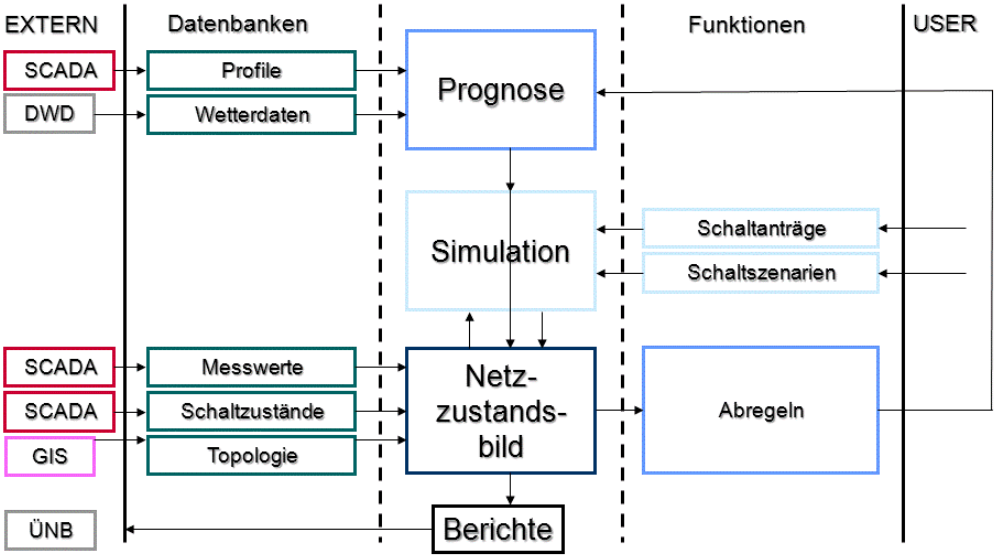


 Seite 16 23/07/2012

CONSULECTRA

Einspeisemanagement

Welche Daten werden (im NLS) benötigt?



 Seite 17 23/07/2012

CONSULECTRA

Einspeisemanagement

Welche Ausbaustufen mit welchen Funktionen sind denkbar?

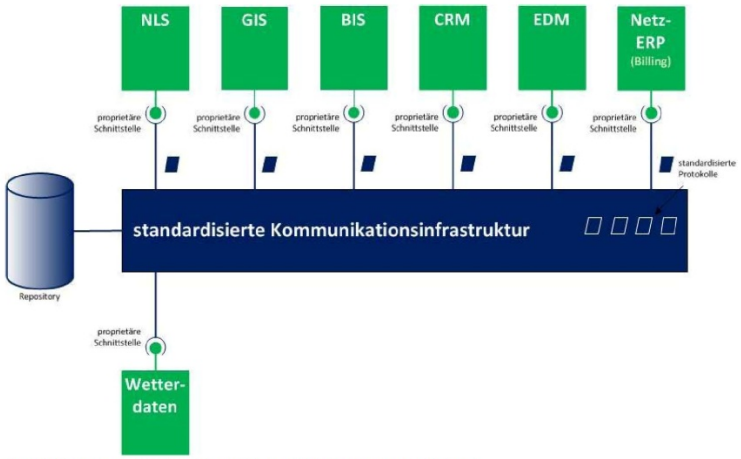
Im Workshop zu erarbeiten!

 Seite 18 23/07/2012

CONSULECTRA

Einspeisemanagement

Welche Systeme sind in welcher Ausbaustufe beteiligt?



Im Workshop sind die Ausbaustufen und Varianten zu erarbeiten!

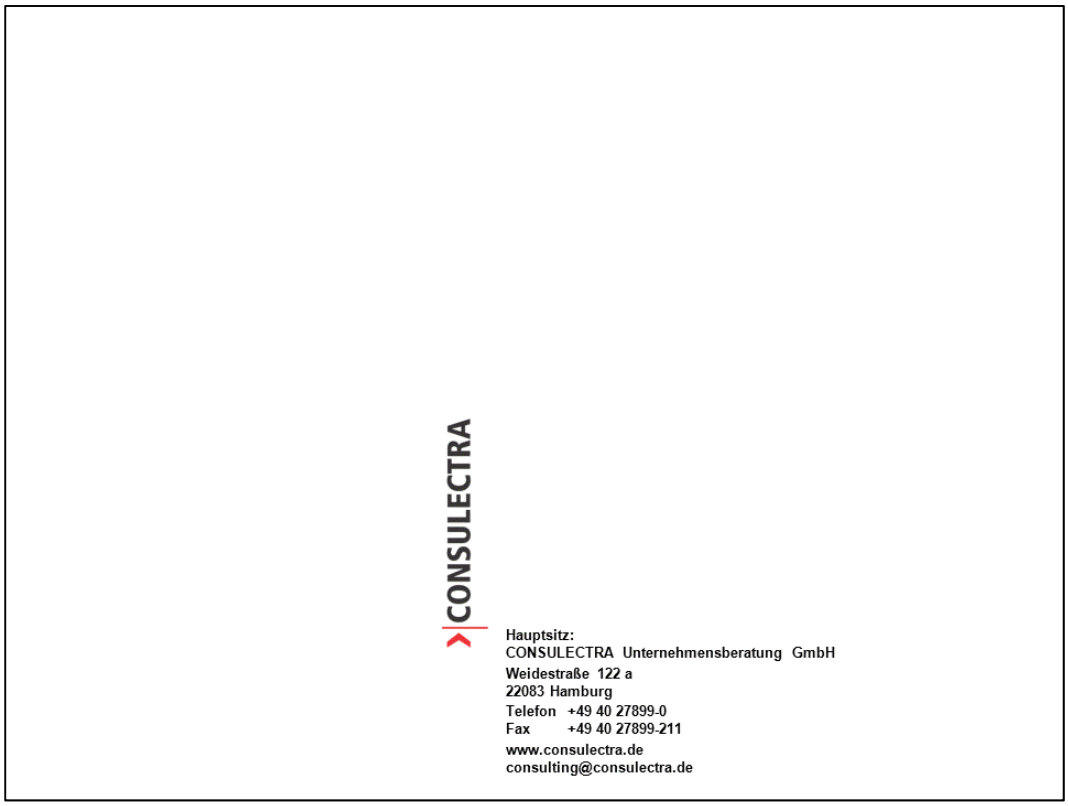
 Seite 19 23/07/2012

CONSULECTRA

Weitere Aspekte

- Kriterien für die Auswahl einer Hosting-Plattform
- mögliche Dienstleistungen einer Hosting-Plattform
  - Projektmanagement
  - Versionierung (git),
  - IP-Management
    (Überprüfen der Codebasis)
  - Testing
- Bereitstellung der Produkte (Release1.n)
- Implementierung
- ggf. Customizing

 Seite 20 23/07/2012


CONSULECTRA
Hauptsitz:
CONSULECTRA Unternehmensberatung GmbH
Weidestraße 122 a
22083 Hamburg
Telefon +49 40 27899-0
Fax +49 40 27899-211
www.consulectra.de
consulting@consulectra.de

Anlage 3-2

Agiles Manifest

Prinzipien hinter dem Agilen Manifest

Wir folgen diesen Prinzipien:

Unsere höchste Priorität ist es,
den Kunden durch frühe und kontinuierliche Auslieferung
wertvoller Software zufrieden zu stellen.

Heisse Anforderungsänderungen selbst spät
in der Entwicklung willkommen. Agile Prozesse nutzen Veränderungen
zum Wettbewerbsvorteil des Kunden.

Liefere funktionierende Software
regelmäßig innerhalb weniger Wochen oder Monate und
bevorzuge dabei die kürzere Zeitspanne.

Fachexperten und Entwickler
müssen während des Projektes
täglich zusammenarbeiten.

Errichte Projekte rund um motivierte Individuen.
Gib ihnen das Umfeld und die Unterstützung, die sie benötigen
und vertraue darauf, dass sie die Aufgabe erledigen.

Die effizienteste und effektivste Methode, Informationen
an und innerhalb eines Entwicklungsteam zu übermitteln,
ist im Gespräch von Angesicht zu Angesicht.

Funktionierende Software ist das
wichtigste Fortschrittsmaß.

Agile Prozesse fördern nachhaltige Entwicklung.
Die Auftraggeber, Entwickler und Benutzer sollten ein
gleichmäßiges Tempo auf unbegrenzte Zeit halten können.

Ständiges Augenmerk auf technische Exzellenz und
gutes Design fördert Agilität.

Einfachheit -- die Kunst, die Menge nicht
getaner Arbeit zu maximieren -- ist essenziell.

Die besten Architekturen, Anforderungen und Entwürfe
entstehen durch selbstorganisierte Teams.

In regelmäßigen Abständen reflektiert das Team,
wie es effektiver werden kann und passt sein
Verhalten entsprechend an.

Zurück zum Manifest

Anlage 3-3

Foliensatz von Prof. Riehle

OSR Group Overview

Prof. Dr. Dirk Riehle, M.B.A.

Friedrich-Alexander-University of Erlangen-Nürnberg

N-ERGIE Netz GmbH: May 19, 2010

Friedrich-Alexander-Universität Erlangen-Nürnberg

Talk Overview

TECHNISCHE FAKULTÄT

- Open Source
- Open Source Economics
- Open Source Engineering Research

PART I

Open Source

Friedrich-Alexander-Universität Erlangen-Nürnberg

What is Open Source? (Formal Definition)

TECHNISCHE FAKULTÄT

- Definition of open source software
  - Software that is provided under an OSI-approved license
  - OSI = Open Source Initiative, http://opensource.org
  - Tried (but failed) to register the “open source” trademark
- Characteristics of an OSI-approved license
  - Source code is available and accessible
  - Modifications of code are allowed (and desired)
  - Distribution of source and binary code is unrestricted
- Free software is a variant of open source software
  - Historically, free software predates open source software
  - Free software is a subset of open source software

Friedrich-Alexander-Universität Erlangen-Nürnberg

What is Open Source? (Less Formal Definition)

TECHNISCHE FAKULTÄT

"Open source is **a development method** for software **that harnesses the power of distributed peer review** and **transparency of process**. The promise of open source is **better quality**, **higher reliability**, **more flexibility**, **lower cost**, and **an end to predatory vendor lock-in**."

From http://opensource.org/docs/osd

- A process: development method, marketing method to some, ...
- Of some quality: transparent, meritocratic, self-organizing, ...
- With results: better quality, more flexibility, no vendor lock-in, ...
- Frequently with a community using and developing the project

Friedrich-Alexander-Universität Erlangen-Nürnberg

5

OSR Group Overview · Dirk Riehle · May 19th, 2010

Examples of Open Source Software

TECHNISCHE FAKULTÄT



debian



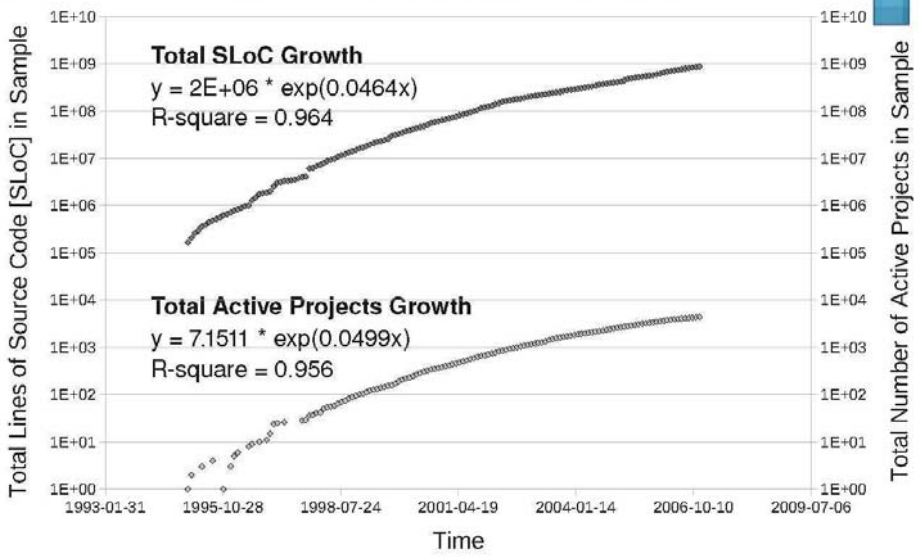
Friedrich-Alexander-Universität Erlangen-Nürnberg

6

OSR Group Overview · Dirk Riehle · May 19th, 2010

The Total Growth of Open Source [1]

TECHNISCHE FAKULTÄT



[1] Amit Deshpande, Dirk Riehle. "The Total Growth of Open Source." In Proceedings of the Fourth Conference on Open Source Systems (OSS 2008). Springer Verlag, 2008. Page 197-209.

7

Economic Impact of Open Source

TECHNISCHE FAKULTÄT

- Commercial use of open source
  - Gartner: "By 2012, more than 90 percent of enterprises will use open source [...]" [1]
  - By and large, open source has gone mainstream, is just a product like any other
  - Today, formal open source adoption strategy elusive and TCO gains unclear
  - Open source dominates software-as-a-service and in startups
- Packaged software market
  - In 2006, open source held $1.8B / $235B = 0.8% of market revenue [2] [3]
  - IDC: Open source revenue will reach $5.8B by 2011 (26% CAGR 2006-11) [2]

[1] Gartner Inc. "The State of Open Source 2008." Gartner, 2008.
[2] IDC. "Worldwide Open Source Software Business Models 2007–2011 Forecast: A Preliminary View." IDC, 2006.
[3] Software & Information Industry Association. "Packaged Software Industry Revenue and Growth, 2006."SIIA, 2006.

8

PART II

Open Source Economics

Friedrich-Alexander-Universität Erlangen-Nürnberg

Three Main Areas of Economics Research

TECHNISCHE FAKULTÄT

- Profit Motive
- Public Welfare
- Labor Market

Friedrich-Alexander-Universität Erlangen-Nürnberg

Community vs. Commercial Open Source [1] [2]

TECHNISCHE FAKULTÄT

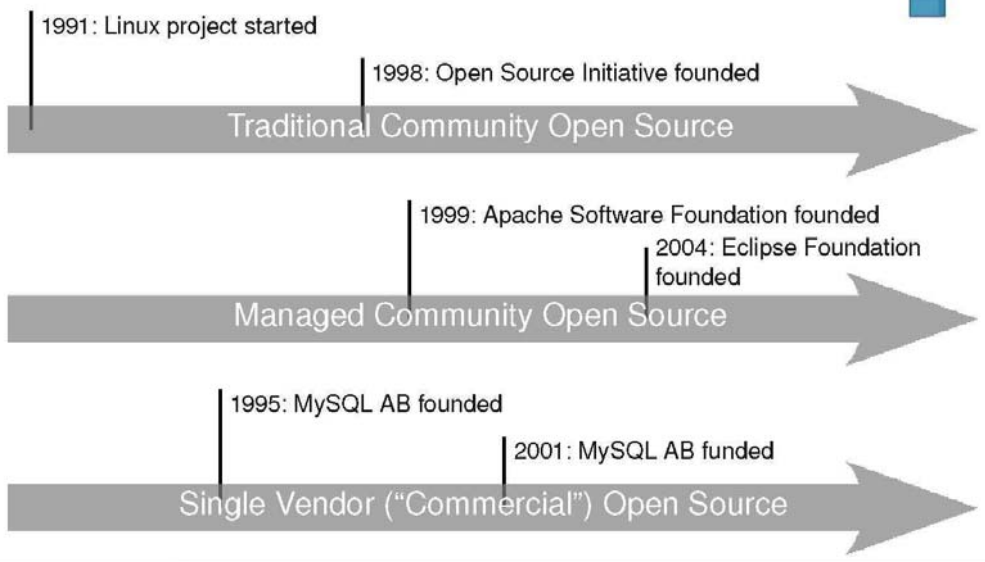
Ownership		Project Type: Single product or product line	Project Type: Multi-product assembly
	Community-owned	**Community Open Source** (e.g. Linux, Apache)	**Community Distribution** (e.g. Debian)
	Single or dominant vendor/proprietor	**Commercial Open Source** (e.g. MySQL, Alfresco)	**Commercial Distribution** (e.g. RHEL, SLES)

[1] Dirk Riehle. "The Economic Motivation of Open Source: Stakeholder Perspectives." IEEE Computer, vol. 40, no. 4 (April 2007).
[2] Dirk Riehle. "The Commercial Open Source Business Model." In Value Creation in e-Business Management (LNBIP 36). Springer, 2009.

11

Time-Line of Open Source

TECHNISCHE FAKULTÄT



Friedrich-Alexander-Universität Erlangen-Nürnberg

12

OSR Group Overview · Dirk Riehle · May 19th, 2010

Some Control Points in Open Source Projects

TECHNISCHE FAKULTÄT

IP Rights	Access Control
Copyright	Write Access
Trademarks	Governance Rules
Patents	Domain Ownership

Friedrich-Alexander-Universität Erlangen-Nürnberg

13

Open vs. Closed Development

TECHNISCHE FAKULTÄT

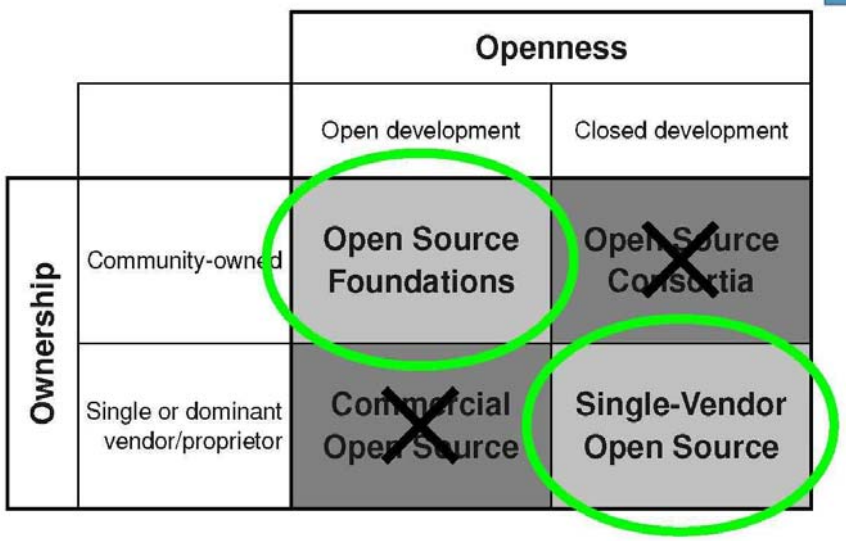
		Openness	
		Open development	Closed development
Ownership	Community-owned	**Open Source Foundations** (e.g. Apache, Gnome)	**Open Source Consortia** (e.g. OW2, GenIVI)
	Single or dominant vendor/proprietor	**Commercial Open Source** (e.g. Acquia, TWiki.net)	**Single-Vendor Open Source** (e.g. MySQL, Alfresco)

Friedrich-Alexander-Universität Erlangen-Nürnberg

14

Sweet Spots for Open Source

TECHNISCHE FAKULTÄT



Friedrich-Alexander-Universität Erlangen-Nürnberg

15

OSR Group Overview · Dirk Riehle · May 19th, 2010

The Profit Motive of Community Open Source [1]

TECHNISCHE FAKULTÄT

Higher Profits per Sale

More Sales in a Given Market

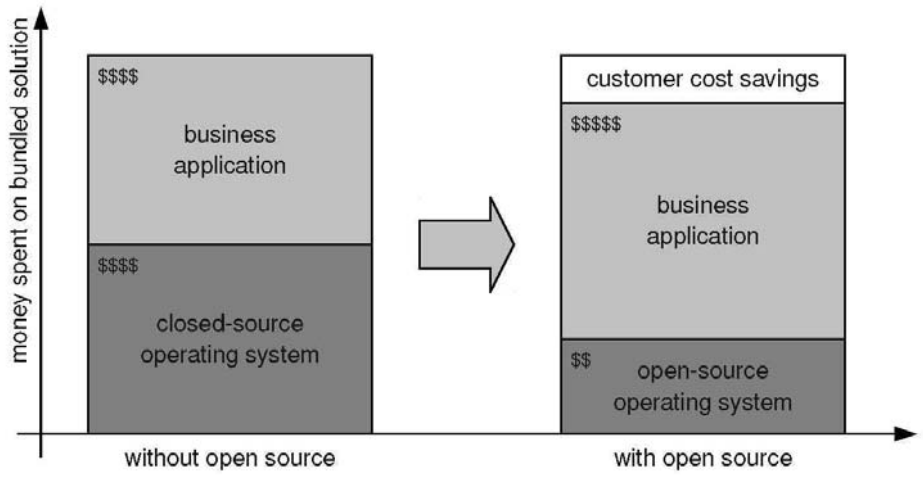
Larger Addressable Market

[1] Dirk Riehle. "The Economic Case for Open Source Foundations." IEEE Computer, vol. 43, no. 1 (January 2010). Page 86-90.

16

Higher Profits per Sale

TECHNISCHE FAKULTÄT



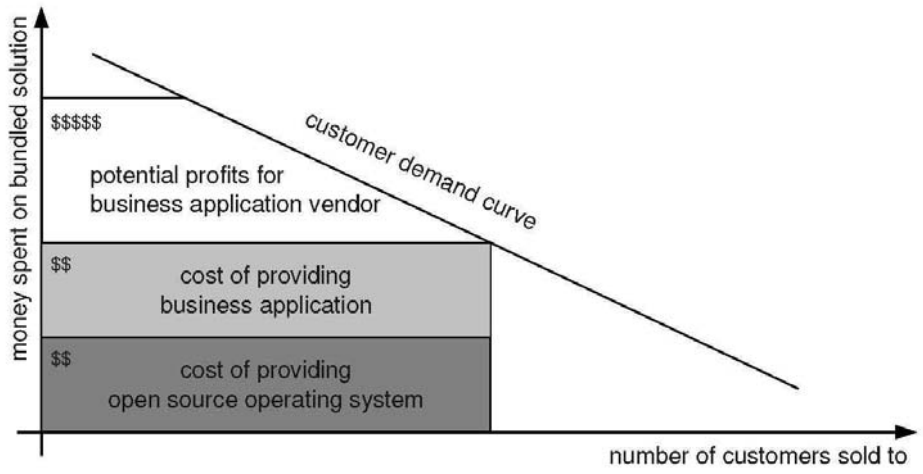
Friedrich-Alexander-Universität Erlangen-Nürnberg

17

OSR Group Overview · Dirk Riehle · May 19th, 2010

More Sales in a Given Market

TECHNISCHE FAKULTÄT



Friedrich-Alexander-Universität Erlangen-Nürnberg

18

OSR Group Overview · Dirk Riehle · May 19th, 2010

What is Single-Vendor Open Source?

TECHNISCHE FAKULTÄT

- Definition of single-vendor open source
  - Riehle: Open source owned and developed by a single firm with a profit motive [1]
  - Gartner: Open source under the patronage of a single firm [2]
- Gartner Group predicts single-vendor open source
  - "By 2012, at least 50% of direct commercial revenue attributed to open-source products or services will come from projects under a single vendor's patronage." [2]
- Further properties of single-vendor open source
  - Ownership established through copyright, trademarks, patents
  - Can and does receive venture capital funding
- Next step in the evolution of software firms harnessing user innovation

[1] Dirk Riehle. "The Economic Motivation of Open Source: Stakeholder Perspectives." IEEE Computer, vol. 40, no. 4 (April 2007).
[2] Gartner Research. "Predicts 2009: The Evolving Open-Source Software Model." Gartner, Inc. (December 2008).

19

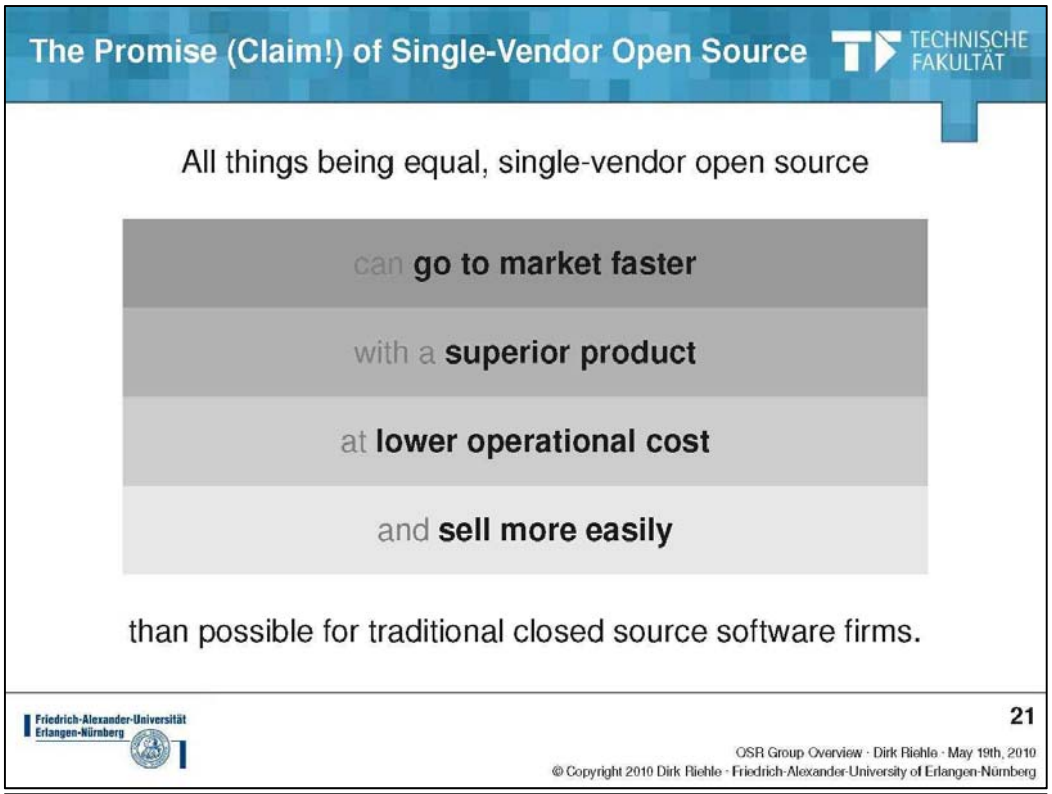
Single-Vendor vs. Community Open Source

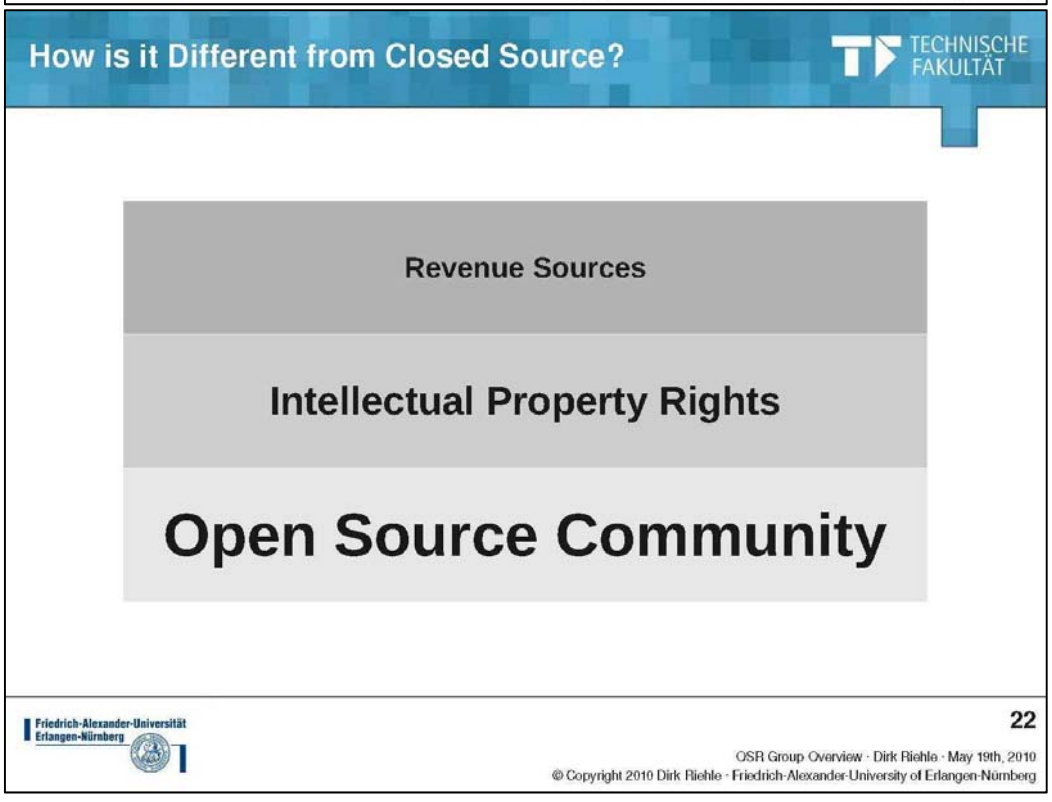
TECHNISCHE FAKULTÄT

Single-Vendor Open Source	Community Open Source
Single proprietor	Community of owners
Multiple licenses	Single license
Feature differentiated	No functionality withheld
Venture-capital backed	Cross-subsidized
Significant direct revenues	Minimal direct revenues
Asymmetric community	Community of equals

Friedrich-Alexander-Universität Erlangen-Nürnberg

20


The Promise (Claim!) of Single-Vendor Open Source
TECHNISCHE FAKULTÄT
All things being equal, single-vendor open source
can go to market faster
with a superior product
at lower operational cost
and sell more easily
than possible for traditional closed source software firms.
Friedrich-Alexander-Universität Erlangen-Nürnberg
21
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg


How is it Different from Closed Source?
TECHNISCHE FAKULTÄT
Revenue Sources
Intellectual Property Rights
Open Source Community
Friedrich-Alexander-Universität Erlangen-Nürnberg
22
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg

Revenue Sources of Single-Vendor Open Source

TECHNISCHE FAKULTÄT

Core Product

Whole Product

Operational Comfort

Support and Consulting

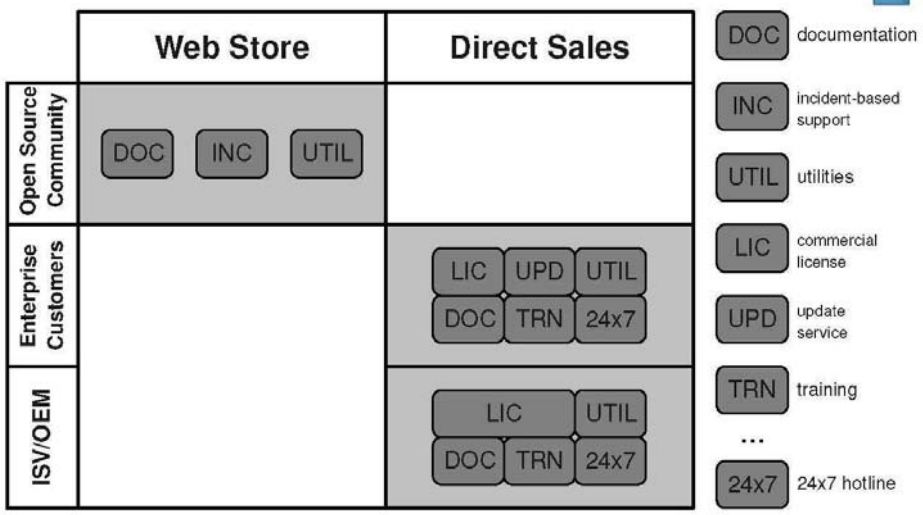
Friedrich-Alexander-Universität Erlangen-Nürnberg

23

OSR Group Overview · Dirk Riehle · May 19th, 2010

Example Product Bundling and Portfolio

TECHNISCHE FAKULTÄT



Friedrich-Alexander-Universität Erlangen-Nürnberg

24

OSR Group Overview · Dirk Riehle · May 19th, 2010

The Intellectual Property Rights Imperative

TECHNISCHE FAKULTÄT

The Intellectual Property Rights Imperative of Single-Vendor Open Source

Always act in such a way that you, and only you, possess the right to provide the open source project under a license of your choice.

Friedrich-Alexander-Universität Erlangen-Nürnberg

25

OSR Group Overview · Dirk Riehle · May 19th, 2010

IPR Best Practice: Use Reciprocal License

TECHNISCHE FAKULTÄT

Goal	Keep competitors out
Policy	Prevent competitors from using your work to compete with you
Practice	Use reciprocal open source license

Friedrich-Alexander-Universität Erlangen-Nürnberg

26

OSR Group Overview · Dirk Riehle · May 19th, 2010

IPR Best Practice: Require Relicensing Rights Grant

TECHNISCHE FAKULTÄT

Goal	Maintain full ownership of software while allowing for community contributions
Policy	Always acquire relicensing rights when accepting community contribution
Practice	Require relicensing rights grant from all community contributors at all times

Friedrich-Alexander-Universität Erlangen-Nürnberg

27

OSR Group Overview · Dirk Riehle · May 19th, 2010

The Single-Vendor Open Source Community

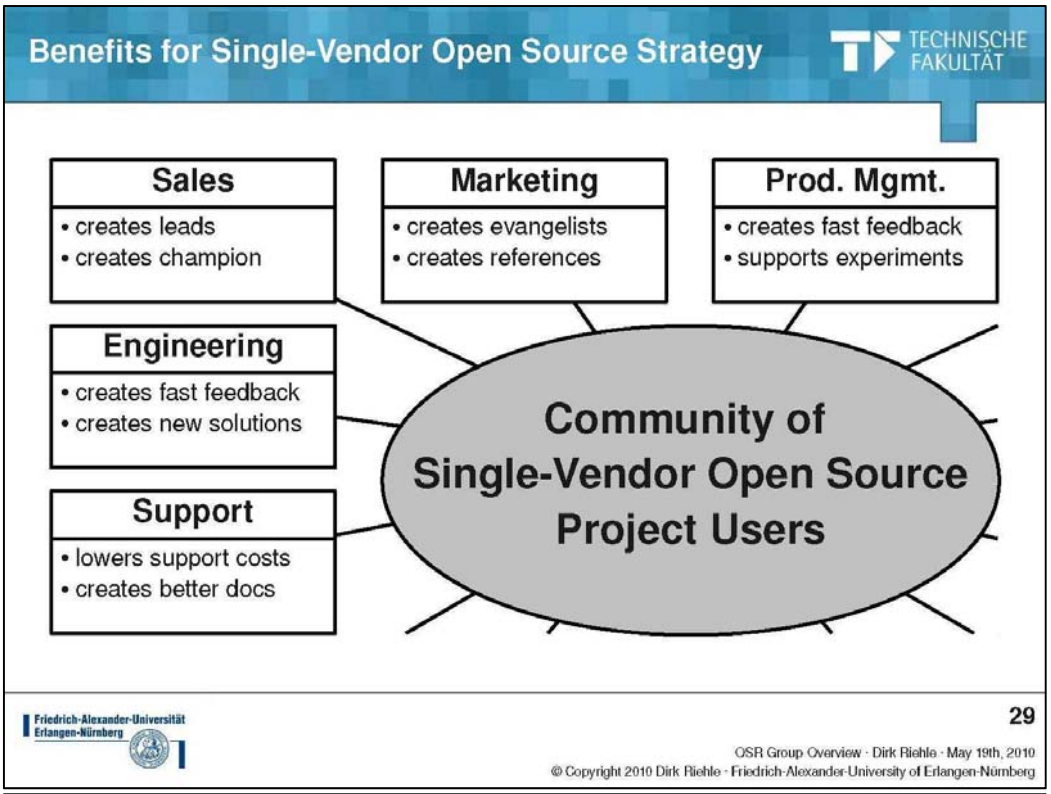
TECHNISCHE FAKULTÄT

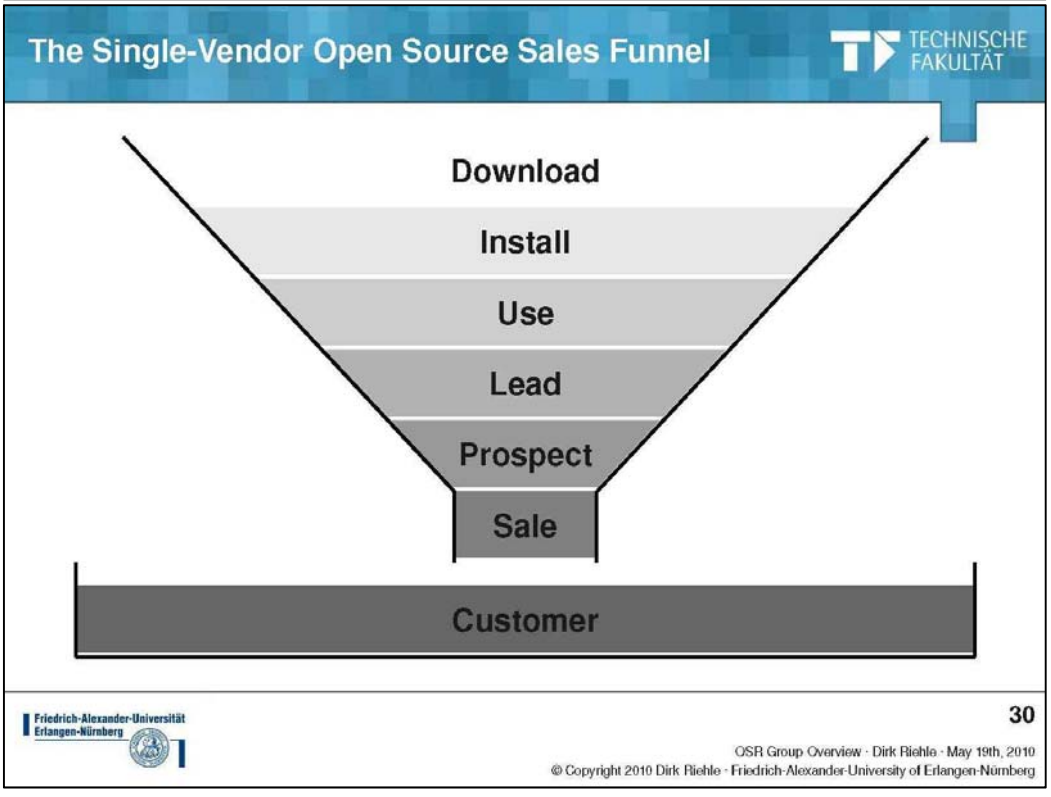
Size	Type	Revenue
1	**Enterprise Customer**	99%
10-20	**Casual Customer**	1%
100-1,000	**Non-paying User**	-

Friedrich-Alexander-Universität Erlangen-Nürnberg

28

OSR Group Overview · Dirk Riehle · May 19th, 2010


Benefits for Single-Vendor Open Source Strategy
TECHNISCHE FAKULTÄT
Sales
• creates leads
• creates champion
Marketing
• creates evangelists
• creates references
Prod. Mgmt.
• creates fast feedback
• supports experiments
Engineering
• creates fast feedback
• creates new solutions
Support
• lowers support costs
• creates better docs
Community of Single-Vendor Open Source Project Users
Friedrich-Alexander-Universität Erlangen-Nürnberg
29
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg


The Single-Vendor Open Source Sales Funnel
TECHNISCHE FAKULTÄT
Download
Install
Use
Lead
Prospect
Sale
Customer
Friedrich-Alexander-Universität Erlangen-Nürnberg
30
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg


Marketing Best Practice: Identify Leads by Behavior
TECHNISCHE FAKULTÄT
Goal
Identify leads when they are ready
Policy
Convert users to leads asap but only when they are ready
Practice
Create a place for the community, track users, and convert only when qualified
Friedrich-Alexander-Universität Erlangen-Nürnberg
31
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg


Marketing Best Practice: Network Your Users
TECHNISCHE FAKULTÄT
Goal
Prepare the grounds for sales win
Policy
Prepare the grounds for sales win without annoying users
Practice
Network users at potential customer firm
Friedrich-Alexander-Universität Erlangen-Nürnberg
32
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg

Sales Best Practice: Draw on In-House Champion

TECHNISCHE FAKULTÄT

Goal	Make the sale
Policy	Win over the decision maker by utilizing in-house champion
Practice	Bring into (competitive) sales situation in-house champion to gain credibility

Friedrich-Alexander-Universität Erlangen-Nürnberg

33

OSR Group Overview · Dirk Riehle · May 19th, 2010

Community Management

TECHNISCHE FAKULTÄT

Single-vendor open source firms need large communities, but supporting non-paying users is prohibitively expensive.

Thus, create a new business function

Community Management

with one well-defined goal:

Grow the largest possible self-sustaining community.

Friedrich-Alexander-Universität Erlangen-Nürnberg

34

OSR Group Overview · Dirk Riehle · May 19th, 2010

Labor Market and Community Open Source

TECHNISCHE FAKULTÄT

What Position Affords	Value to Employer
Validated technical abilities (developer)	Reduced hiring risk
Deeper insight, more leverage (committer)	Better product quality
Community visibility, reputation (committer)	Higher reputation, more sales
Strategic alignment with project (committer)	Lower costs, more predictability

Friedrich-Alexander-Universität Erlangen-Nürnberg

35

OSR Group Overview · Dirk Riehle · May 19th, 2010

Benefits that Accrue to Employee

TECHNISCHE FAKULTÄT

- Higher Salary
- More Job Security
- Higher Job Versatility
- Richer Job Experience

Friedrich-Alexander-Universität Erlangen-Nürnberg

36

OSR Group Overview · Dirk Riehle · May 19th, 2010

PART III

Open Source Engineering Research

Friedrich-Alexander-Universität Erlangen-Nürnberg

Principles of Open (Source) Collaboration [1]

TECHNISCHE FAKULTÄT

- Egalitarian
- Meritocratic
- Self-Organizing

[1] Dirk Riehle, John Ellenberger, Tamir Menahem, Boris Mikhailovski, Yuri Natchetoi, Barak Naveh, Thomas Odenwald. "Open Collaboration within Corporations Using Software Forges." IEEE Software, vol. 26, no. 2 (March/April 2009). Page 52-58.

38

Open Collaboration vs. Traditional Work

TECHNISCHE FAKULTÄT

Open Collaboration	Traditional Work
• Egalitarian – Open for contribution – Everyone can contribute	• Hierarchical – Closed and hidden silos – Assigned to project
• Meritocratic – Public discussion process – Decisions based on merit	• Status-oriented – Public and private discussions – Hierarchical status decides
• Self-organizing – People find their own process – People find their best project	• Assigned tasks – Prescribed process – Prescribed jobs

Friedrich-Alexander-Universität Erlangen-Nürnberg

39

OSR Group Overview · Dirk Riehle · May 19th, 2010

Open Source Communities (Relative Sizes)

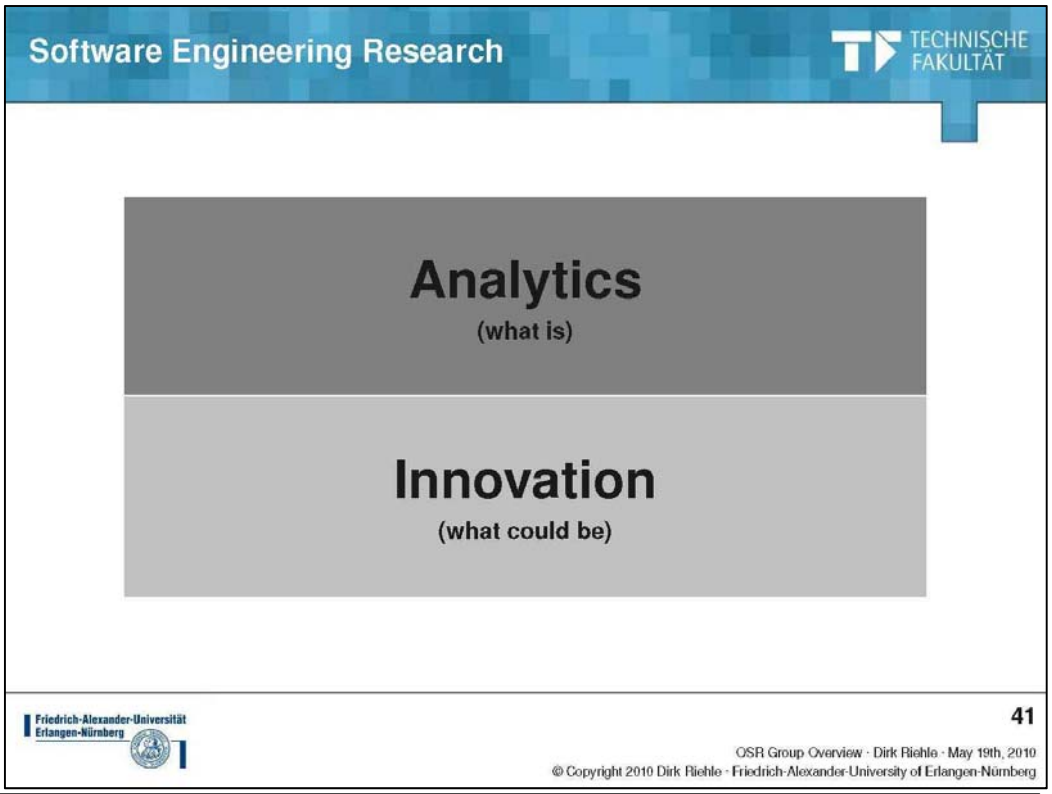
TECHNISCHE FAKULTÄT

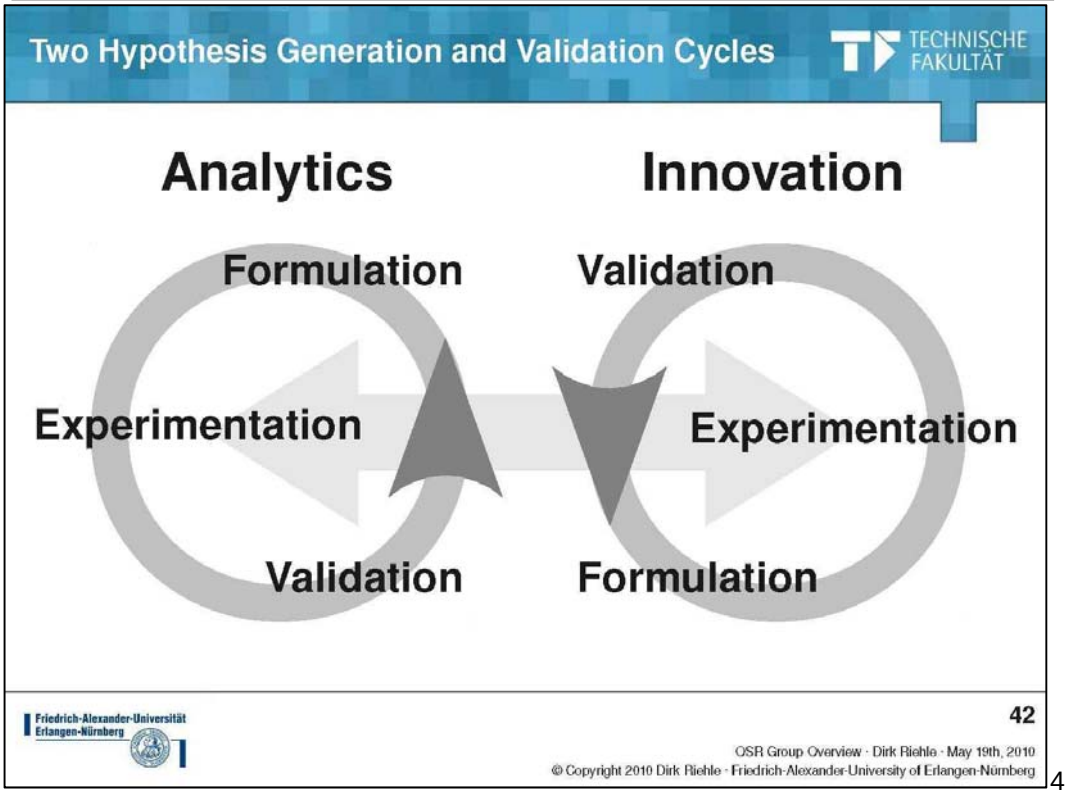
1,000-100,000	**User** (Installations)	Uses the software Helps other users
10-100	**Contributor**	Provides feedback Writes code
1	**Committer**	Assures code quality Leads project

Friedrich-Alexander-Universität Erlangen-Nürnberg

40

OSR Group Overview · Dirk Riehle · May 19th, 2010

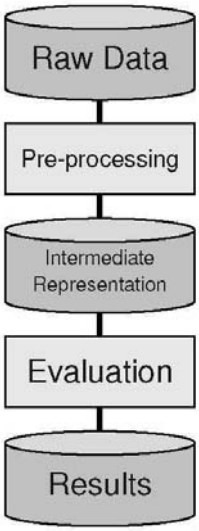

Software Engineering Research
TECHNISCHE FAKULTÄT
Analytics
(what is)
Innovation
(what could be)
Friedrich-Alexander-Universität Erlangen-Nürnberg
41
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg


Two Hypothesis Generation and Validation Cycles
TECHNISCHE FAKULTÄT
Analytics
Innovation
Formulation
Validation
Experimentation
Experimentation
Validation
Formulation
Friedrich-Alexander-Universität Erlangen-Nürnberg
42
OSR Group Overview · Dirk Riehle · May 19th, 2010
© Copyright 2010 Dirk Riehle · Friedrich-Alexander-University of Erlangen-Nürnberg

4

Analysis Approach and Tool Chain

TECHNISCHE FAKULTÄT



- Raw data source
  - Local database (ohloh.net, crawled sources)
  - Web services access (ohloh.net, sf.net, others)
- Pre-processing
  - Database queries using SQL and SQL scripts
  - Uses Java for computationally heavyweight filters
- Intermediate representation
  - Output of pre-processing stage for specific tasks
  - Aggregation speeds up analytical processing
- Analytical processing
  - Mines data for insight, hypothesis testing
  - Basic processing, Java programming, R-project
- Analysis output and results
  - Results of processing: averages, distributions, correlations
  - Presented as models, tables, graphs, charts, etc.

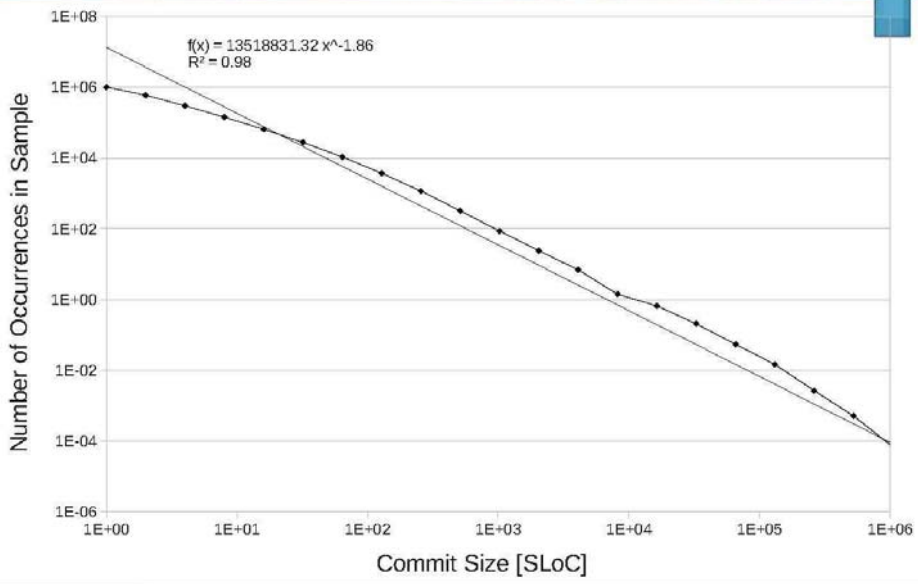
Friedrich-Alexander-Universität Erlangen-Nürnberg

43

OSR Group Overview · Dirk Riehle · May 19th, 2010

The Commit Size Distribution of Open Source [1]

TECHNISCHE FAKULTÄT

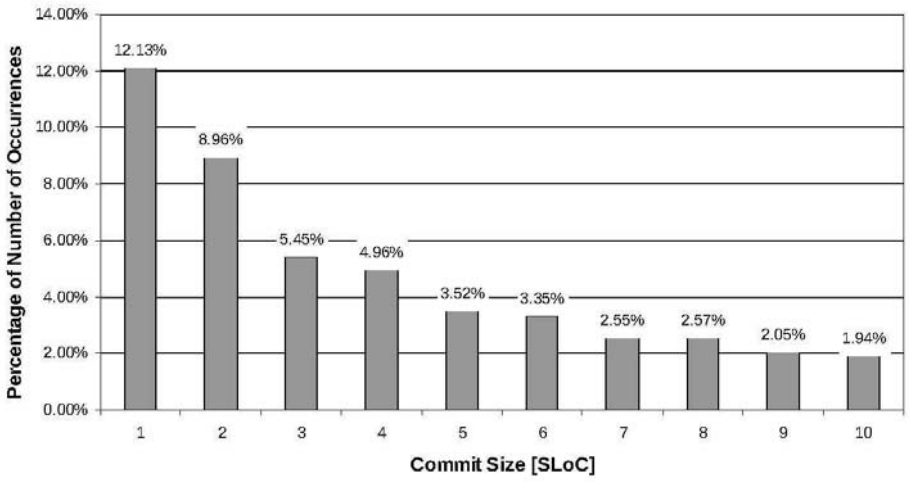


[1] Oliver Arafat, Dirk Riehle. "The Commit Size Distribution of Open Source Software." In Proceedings of the 42nd Hawaiian International Conference on System Sciences (HICSS 42). IEEE Press, 2009. Page 1-8.

44

The Dominance of Small Commits

TECHNISCHE FAKULTÄT



Friedrich-Alexander-Universität Erlangen-Nürnberg

45

OSR Group Overview · Dirk Riehle · May 19th, 2010

Example Hypotheses Based on Commit Size Analysis

TECHNISCHE FAKULTÄT

Smaller Window Sizes for Merge and Code Review Tools

Time Series View of Commits rather than Update Snapshots

Finer-grain Execution of Regression Test Suites

Friedrich-Alexander-Universität Erlangen-Nürnberg

46

OSR Group Overview · Dirk Riehle · May 19th, 2010

The Three Dominant Process Frameworks

TECHNISCHE FAKULTÄT

Plan-driven Methods

Agile Methods

Open Source

Friedrich-Alexander-Universität Erlangen-Nürnberg

47

OSR Group Overview · Dirk Riehle · May 19th, 2010

Process Frameworks in Comparison

TECHNISCHE FAKULTÄT

		Need to Change: No	Need to Change: Yes
Need to Scale	No	Plan-Driven Agile Methods Open Source	**Agile Methods** **Open Source**
Need to Scale	Yes	**Plan-Driven** **Open Source**	**Open Source**

Friedrich-Alexander-Universität Erlangen-Nürnberg

48

OSR Group Overview · Dirk Riehle · May 19th, 2010

Agile Methods vs. Open Source

TECHNISCHE FAKULTÄT

Agile Methods

- Co-location
  - Stand-up meetings
  - Short communication paths
- Code speaks for itself
  - Self-explanatory code
  - Refactor, don't comment
- Continuous integration
  - Small focussed commits
  - Frequent commits

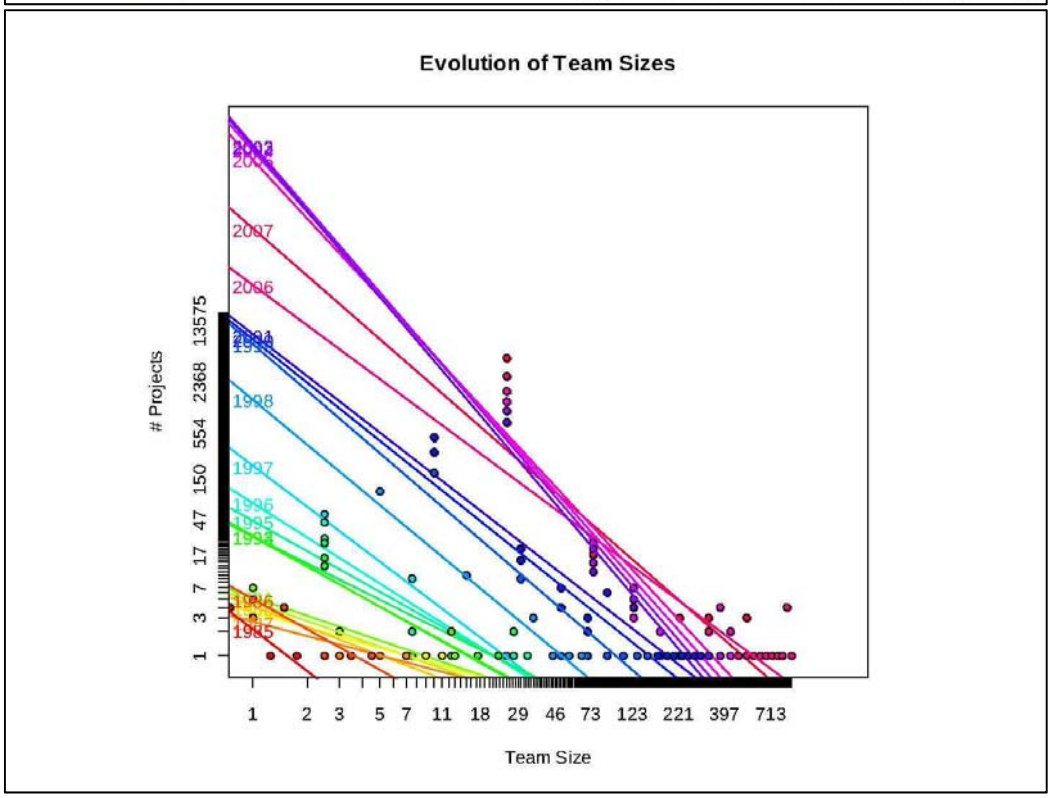
Open Source

- Distributed development
  - Lengthy email discussions
  - Across all time zones
- Frequent commenting
  - About 20% comment density
  - Frequent comment only commits
- Has not changed practices
  - Commit size constant
  - Commit frequency constant

Friedrich-Alexander-Universität Erlangen-Nürnberg

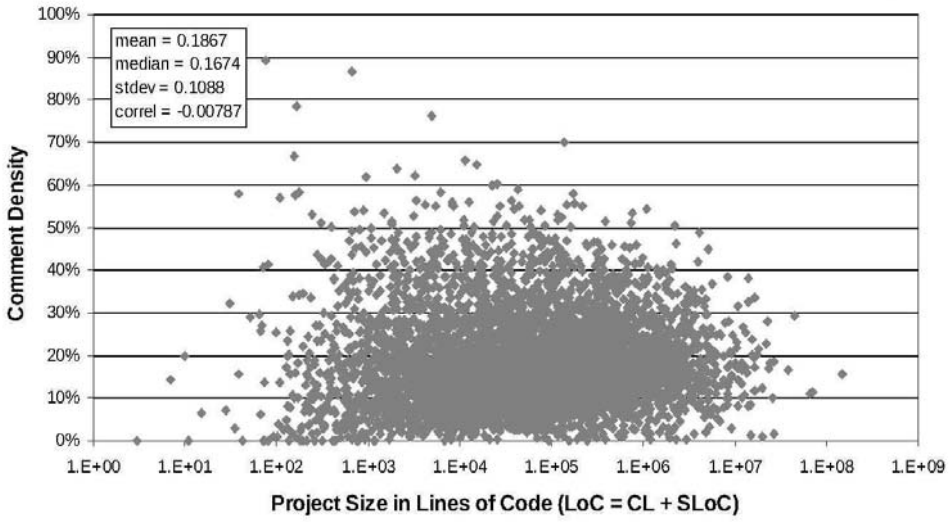
49

OSR Group Overview · Dirk Riehle · May 19th, 2010



The Comment Density of Open Source [1]

TECHNISCHE FAKULTÄT

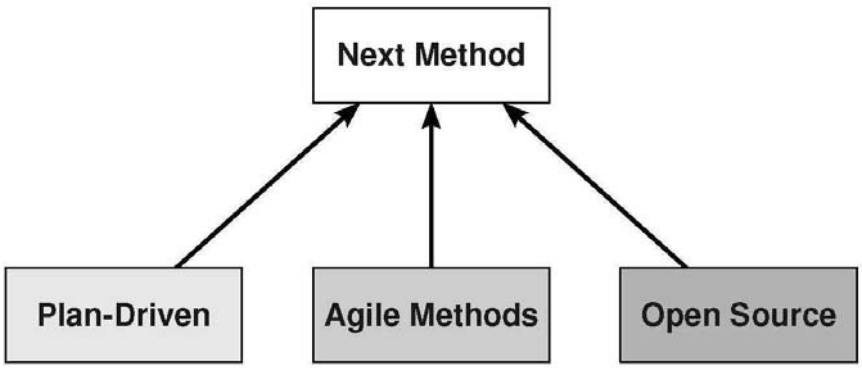


[1] Oliver Arafat, Dirk Riehle. "The Comment Density of Open Source Software Code." In Companion to Proceedings of the 31st International Conference on Software Engineering (ICSE 2009). IEEE Press, 2009. Page 195-198.

51

A New "Next" Unified Method

TECHNISCHE FAKULTÄT



Friedrich-Alexander-Universität Erlangen-Nürnberg

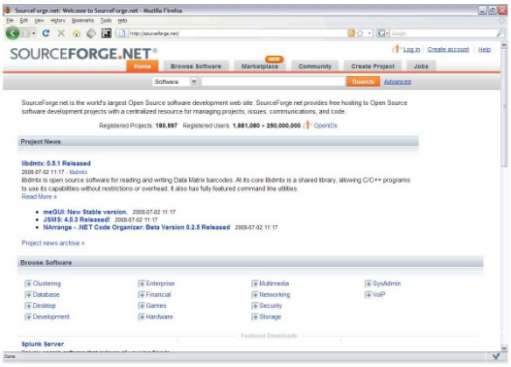
52

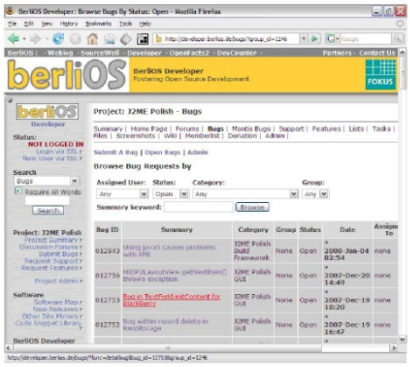
OSR Group Overview · Dirk Riehle · May 19th, 2010

Software Tool and Process Innovation

TECHNISCHE FAKULTÄT

- A software forge is an
  - extensible web-based software tools platform that
  - integrates best-of-breed tools for collaborative software development
- Best known example is SourceForge.net but there are many more

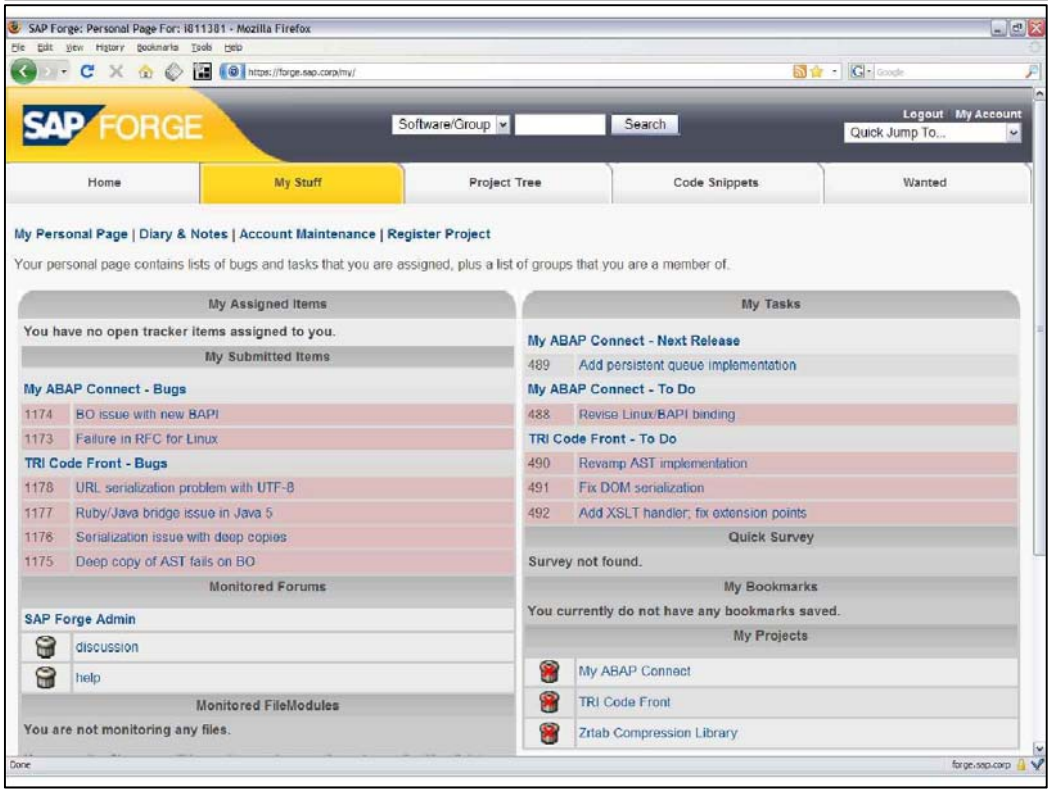


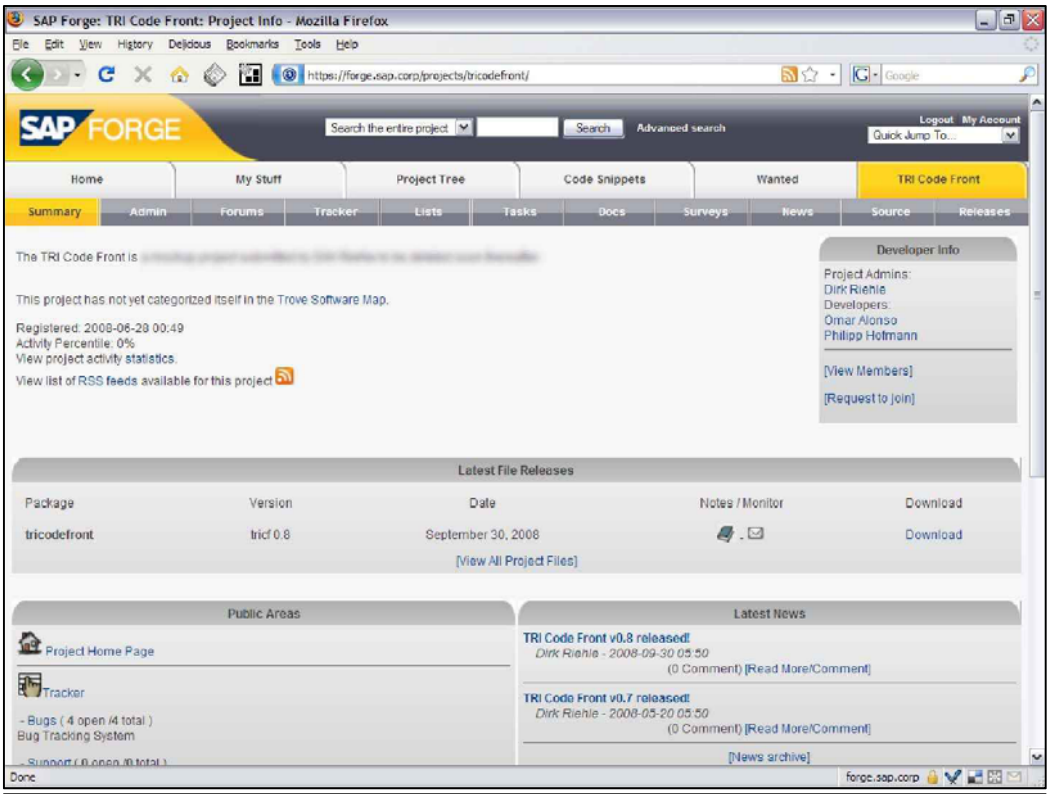


Friedrich-Alexander-Universität Erlangen-Nürnberg

53

OSR Group Overview · Dirk Riehle · May 19th, 2010





Innovation Example: The Volunteering Process

TECHNISCHE FAKULTÄT

Process Step	Forge Features
Get curious	• Project of the week • Top 10 active projects
Find interesting project	• Project search • Cross-linked projects
Understand project	• Open forums and mailing lists • Documented code and wikis
Engage with project	• One click to forum reply • Easy to install and run
Stay with project	• Email updates and conversation • Forge account/persona, reputation

Friedrich-Alexander-Universität Erlangen-Nürnberg

56

OSR Group Overview · Dirk Riehle · May 19th, 2010

PART IV

Conclusions

Friedrich-Alexander-Universität Erlangen-Nürnberg

Conclusions

TECHNISCHE FAKULTÄT

- Open Source is changing the software industry
  - Continued exponential growth is eating into closed source
  - Has penetrated all parts of software and user scenarios
- Open Source is a sustainable phenomenon
  - It is economically rational for software vendors
  - It does not depend on volunteer work alone any longer
- Open Source is great for software engineering research
  - It is public and hence analyzable software development
  - It is novel and different from approaches like plan-driven or agile

Friedrich-Alexander-Universität Erlangen-Nürnberg

58


Questions? Feedback!
http://osr.cs.fau.de - dirk.riehle@cs.fau.de
(Auf Deutsch: dirk.riehle@informatik.uni-erlangen.de)
http://dirkriehle.com - @dirkriehle
Friedrich-Alexander-Universität Erlangen-Nürnberg
© Copyright 2010 Dirk Riehle - Friedrich-Alexander-University of Erlangen-Nürnberg

Anlage 4-1

Rechtsformenvergleich

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Zweck	Förderung des Erwerbs oder der Wirtschaft der Mitglieder oder deren sozialen oder kulturellen Belange mittels gemeinschaftlichen Geschäftsbetriebs	Jeder, aber grundsätzlich kein wirtschaftlicher Geschäftsbetrieb	Stiftungszweck wird bei der Gründung von den Stiftern festgelegt	Kapitalgesellschaft zur Erreichung jedes gesetzlich zulässigen Zweckes	Kapitalgesellschaft zur Erreichung jedes gesetzlich zulässigen Zweckes	Förderung des Bedarfs ihrer Mitglieder und/oder deren wirtschaftliche und/oder deren sozialen Tätigkeiten
Gründung	Mindestens 3 Mitglieder, die eine schriftliche Satzung festlegen müssen, keine notarielle Beurkundung Gründungsprüfung Entstehung durch Eintragung in das Genossenschaftsregister	Mindestens 7 Mitglieder, die eine schriftliche Satzung festlegen müssen, keine notarielle Beurkundung Entstehung durch Eintragung in das Vereinsregister	Mindestens 1 Stifter, der das Stiftungsgeschäft schriftlich festlegen muss (die Stiftungssatzung ist hierin ein wesentlicher Bestandteil), Entstehung mit der Anerkennung durch die Stiftungsbehörde	Notarielle Beurkundung eines Gesellschaftsvertrags, der nicht notwendigerweise mehrere Gesellschafter voraussetzt Entstehung durch Eintragung in das Handelsregister	Notarielle Beurkundung eines Gesellschaftsvertrags Entstehung durch Eintragung in das Handelsregister	Mindestens 5 natürliche Personen oder 2 juristische Personen, deren Wohn-/Firmensitz in zwei Mitgliedstaaten liegen Entstehung durch Eintragung ins Genossenschaftsregister
Firma	Sach- oder Personenfirma Zusatz "eingetragene Genossenschaft" oder "eG" erforderlich	Sach- und Personenfirma Zusatz "eingetragener Verein" oder "eV" erforderlich		Sach- und Personenfirma Zusatz "mit beschränkter Haftung" erforderlich	Sach- und Personenfirma Zusatz "AG" erforderlich	Sach- und Personenfirma, Zusatz SCE
Rechtsfähigkeit	Als juristische Person rechtsfähig	Als juristische Person rechtsfähig	Als juristische Person rechtsfähig	Als juristische Person rechtsfähig	Als juristische Person rechtsfähig	Als juristische Person rechtsfähig
Gesellschafterliste	Führt die eG selbst	Führt der eV selbst	keine Mitglieder, Gesellschafter oder Eigentümer, sondern nur Mitglieder in einem Stiftungsorgan oder dem Vorstand	Unverzügliche Meldung an das Handelsregister bei jeder Veränderung		Führt die SCE selbst

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Kapital	Kein festes Kapital Jedes Mitglied hat einen Geschäftsanteil zu zeichnen, auf den Einzahlungen geleistet werden müssen Kein Mindestbetrag für den Geschäftsanteil	Kein festes Kapital Mitgliederbeiträge kraft Satzung	Kapitalausstattung von mindestens € 25.000 Keine Mindesteinlagen vorgeschrieben	Festes Stammkapital von mindestens € 25.000 Mindesteinzahlung auf jede Stammeinlage von 25 %, insgesamt jedoch mindestens € 12.500 Mindestgeschäftsanteil € 1	Festes Stammkapital von mindestens € 50.000 Mindestgeschäftsanteil € 1	Kein festes Kapital Jedes Mitglied hat mindestens einen Geschäftsanteil zu zeichnen, auf den Einzahlungen geleistet werden müssen Mindestens € 30.000
Gesellschafts-vermögen	Eigenes Vermögen der Genossenschaft als juristische Person	Eigenes Vermögen des Vereins als juristische Person	Eigenes Vermögen der Stiftung als juristische Person	Eigenes Vermögen der Gesellschaft als juristische Person	Eigenes Vermögen der Gesellschaft als juristische Person	Eigenes Vermögen der Genossenschaft als juristische Person
Haftung	Vermögen der Genossenschaft haftet den Gläubigern Für den Insolvenzfall Nachschusspflicht der Mitglieder in der Satzung regelbar	Nur das Vereinsvermögen	Vermögen der Stiftung haftet den Gläubigern	Vermögen der Gesellschaft haftet den Gläubigern Nachschusspflicht der Gesellschafter im Gesellschaftsvertrag regelbar	Vermögen der Gesellschaft haftet den Gläubigern	Vermögen der Genossenschaft selbst
Geschäfts-führung	Gesamtgeschäftsführungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtgeschäftsführungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtgeschäftsführungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtgeschäftsführungsbefugnis des Geschäftsführers, abweichende Regelungen möglich	Gesamtgeschäftsführungsbefugnis des Vorstandes, abweichende Regelungen möglich	Gesamtgeschäftsführung durch Leitungsorgan bzw. Verwaltungsorgan

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Gesetzlich vorgesehene Organe	Vorstand (mindestens 2 Personen), Aufsichtsrat (mindestens 3 Personen) und Generalversammlung (mind. 1/a), für Genossenschaften mit nicht mehr als 20 Mitgliedern: Vorstand (1 Person) Aufsichtsrat fakultativ	Vorstand, Mitgliederversammlung	Vorstand, Stiftungsrat	Geschäftsführer und Gesellschafterversammlung, Aufsichtsrat fakultativ	Vorstand, Hauptversammlung der Aktionäre, Aufsichtsrat	Generalversammlung und Aufsichtsorgan und Leitungsorgan (monistisches System)
Vertretung	Gesamtvertretungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtvertretungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtvertretungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtvertretungsbefugnis der Geschäftsführer, abweichende Regelungen möglich	Gesamtvertretungsbefugnis des Vorstands, abweichende Regelungen möglich	Gesamtvertretungsbefugnis des Leitungsorgans bzw. Verwaltungsorgans, abweichende Regelungen möglich
Kontroll- und Informationsrechte der Gesellschafter	Kontrollrechte nur über den gewählten Aufsichtsrat, Auskunftsrecht jedes Mitglieds nur in der Generalversammlung 10 % der Mitglieder können die Einberufung einer Generalversammlung verlangen (Minderheitenschutz)	Nur in der Mitgliederversammlung, Einzelheiten ggf. in der Satzung	Nur über den Stiftungsrat, Einzelheiten ggf. in der Satzung	Persönliches Auskunftsrecht jedes Gesellschafters, das jederzeit ausgeübt werden kann, entgegenstehende Vereinbarungen sind unwirksam Gesellschafter, deren Geschäftsanteile 10 % des Stammkapitals entsprechen, können die Einberufung einer Gesellschafterversammlung verlangen (Minderheitenschutz), Kontrollrechte über einen evtl. Aufsichtsrat		Unterrichtung des Aufsichtsorgans mindestens alle drei Monate. Darüber hinaus ist das Aufsichtsorgan über alle Ereignisse zu unterrichten, die sich auf die Lage der SCE spürbar auswirken

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Gesellschafterwechsel	Keine geschlossene Mitgliederzahl, Ein- und Austritt möglich Eintritt mit Zustimmung der eG Kündigung der Mitgliedschaft zum Ende eines Geschäftsjahres unter Einhaltung der in der Satzung enthaltenen Frist Beendigung der Mitgliedschaft durch Übertragung des Geschäftsguthabens, auch Teilübertragungen möglich Ausschluss aus der Genossenschaft zum Ende eines Geschäftsjahres möglich	Eintritt mit Zustimmung des eV Kündigung unter Einhaltung der in der Satzung enthaltenen Frist	Nicht vorgesehen	Keine Kündigung möglich Geschäftsanteile sind veräußerlich (notarielle Beurkundung) und vererblich	Geschäftsanteile sind veräußerlich (notarielle Beurkundung) und vererblich	Keine geschlossene Mitgliederzahl, Ein- und Austritt möglich Eintritt mit Zustimmung der SCE Kündigung der Mitgliedschaft zum Ende eines Geschäftsjahres unter Einhaltung der in der Satzung enthaltenen Frist Beendigung der Mitgliedschaft durch Übertragung des Geschäftsguthabens, auch Teilübertragung möglich
Auseinandersetzung	Anspruch des ausgeschiedenen Mitglieds auf Rückzahlung der Einlage (Geschäftsguthaben)	Kein Anspruch gegenüber dem eV	Kein Anspruch gegenüber der Stiftung	Anspruch gemäß Gesellschaftsvertrag, aber Kapitalerhaltung		Anspruch des ausgeschiedenen Mitglieds auf Rückzahlung der Einlage (Geschäftsguthabens)

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Beschluss-fassung der Gesellschafter	Jedes Mitglied hat eine Stimme, Beschlussfassung in der Generalversammlung, bei Unternehmensgenossenschaften kann 1 Mitglied bis zu 10 % der ausgewiesenen Stimmen eingeräumt werden, grundsätzlich genügt einfache Stimmenmehrheit	Jedes Mitglied hat eine Stimme, grundsätzlich genügt einfache Stimmenmehrheit, abweichende Regelungen möglich	Nur Mitglieder im Stiftungsrat	Ausübung des Stimmrechts nach Geschäftsanteilen, grundsätzlich Beschlussfassung in der Gesellschafterversammlung üblich sind Mehrheitsbeschlüsse	Ausübung des Stimmrechts nach Aktienanteilen, grundsätzlich Beschlussfassung in der Hauptversammlung der Aktionäre, üblich sind Mehrheitsbeschlüsse	Jedes Mitglied hat eine Stimme, Beschlussfassung in der Generalversammlung
Jahres-abschluss	Aufstellung durch den Vorstand innerhalb von 5 Monaten nach Schluss des Geschäftsjahres, Feststellung durch die Generalversammlung innerhalb von 6 Monaten, bestehend aus Bilanz, Gewinn- und Verlustrechnung und Anhang	Keine gesetzliche Bestimmungen	Aufstellung durch den Vorstand innerhalb von 6 - 12 Monaten (je nach Landesgesetzgebung) nach Ablauf des Geschäftsjahres	Aufstellung durch die Geschäftsführer innerhalb von 3 Monaten nach Schluss des Geschäftsjahres, Feststellung durch die Gesellschaft innerhalb von 8 Monaten (bei kleinen GmbHs 6 bzw. 11 Monate), bestehend aus Bilanz, Gewinn- und Verlustrechnung mit Anhang	bestehend aus Bilanz, Gewinn- und Verlustrechnung mit Anhang	Wie bei der eG
Rücklagen	Gesetzliche Rücklage zur Deckung von Bilanzverlusten erforderlich, sonstige Ergebnisrücklagen möglich, Satzung regelt Mindestdotierung	Möglich	Möglich	Rücklage für eigene Anteile erforderlich, hingegen keine gesetzliche Rücklage, sonstige Gewinnrücklagen möglich, Gesellschaftsvertrag regelt Mindestdotierung		Rücklage für eigene Anteile erforderlich, sonst wie bei der eG

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Steuerliche Besonderheit	Rückvergütung als Betriebsausgabe					Wie bei der eG
Gewinn- und Verlustverteilung	Gewinnverteilungsbeschluss der Generalversammlung, Verteilung an die Mitglieder nach Dotierung der Rücklagen nach dem Verhältnis ihrer auf den Geschäftsanteil geleisteten Einzahlungen	Grundsätzlich nicht vorgesehen	Grundsätzlich nicht vorgesehen	Gewinnverteilungsbeschluss der Gesellschafterversammlung, Verteilung nach Dotierung der Rücklagen entsprechend der Höhe der Geschäftsanteile, abweichende Regelungen möglich	Gewinnverteilungsbeschluss der Hauptversammlung	Wie bei der eG
Beratung und Betreuung	Durch Genossenschaftsverband insbesondere in betriebswirtschaftlichen, rechtlichen und steuerlichen Angelegenheiten	Nicht vorgesehen	Nicht vorgesehen	Nicht vorgesehen	Nicht vorgesehen	Wie bei der eG
Prüfung	Gesetzliche Prüfung durch Genossenschaftsverband im Interesse der Mitglieder, keine Prüfung des Jahresabschlusses und Einbeziehung der Buchführung und des Lageberichts bei kleinen eGs	Keine Prüfungspflicht	Keine Prüfungspflicht	Für kleine GmbHs keine Prüfungspflicht, für mittelgroße und große GmbHs Prüfungspflicht, Prüfung durch WP oder vBP	Für kleine AGs keine Prüfungspflicht, für mittelgroße und große AGs Prüfungspflicht, Prüfung durch WP oder vBP	Wie bei der eG

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Offenlegung der Publizität von Jahresabschluss und Lagebericht	Einreichung des Jahresabschlusses, des Lageberichts und des Aufsichtsratsberichts zum Genossenschaftsregister, Veröffentlichungspflicht nur für große Genossenschaften, Einreichung zum elektronischen Bundesanzeiger, Veröffentlichung im elektronischen Bundesanzeiger einsehbar für jedermann	Keine Offenlegung und Publizität	Innerhalb von sechs Monaten nach Ablauf des Geschäftsjahres sind ein Rechnungsabschluss und eine Vermögensübersicht (Jahresrechnung) zu erstellen und mit einem Bericht über die Erfüllung des Stiftungszwecks der Stiftungsaufsichtsbehörde vorzulegen	Einreichung des Jahresabschlusses, des Lageberichts und des Gewinnverwendungsvorschlags und -beschlusses zum Handelsregister, Hinweis im Bundesanzeiger auf Handelsregistereinreichung, bei großen GmbHs Veröffentlichung im Bundesanzeiger	Kleine AG: Einreichung zusammengefasster Bilanz nebst verkürztem Anhang zum Handelsregister. Mittlere AG: Einreichung zusammengefasster Bilanz, zusammengefasste Gewinn- und Verlustrechnung, einen verkürzten Anhang und Lagebericht sowie den Prüfungsvermerk und den Bericht des Aufsichtsrates zum Handelsregister. Große AG: Einreichung des gesamten Jahresabschlusses ohne Kürzungen sowie den Prüfungsvermerk und den Bericht des Aufsichtsrats zum Handelsregister. Veröffentlichung im Bundesanzeiger	Wie bei der eG

	Eingetragene Genossenschaft (eG)	Eingetragener Verein (eV)	Stiftung	Gesellschaft mit beschränkter Haftung (GmbH)	Aktiengesellschaft (AG)	Europäische Genossenschaft (SCE)
Auflösung und Beendigung	Auflösung z. B. durch Beschluss der Generalversammlung, Zeitablauf, Liquidation erfolgt in der Regel durch Vorstand aufgrund gesetzlicher Vorschriften Nach Beendigung der Liquidation Anmeldung des Erlöschens der Firma Verteilung des Reinvermögens an die Mitglieder nach Ablauf eines Sperrjahres	Im Grundsatz wie eG		Gesellschaft endet z. B. durch Zeitablauf, Gesellschafterbeschluss, gerichtliches Urteil, Insolvenzeröffnung Liquidation erfolgt durch die Geschäftsführer aufgrund gesetzlicher Vorschriften Nach Beendigung der Liquidation Anmeldung des Erlöschens der Firma	Gesellschaft endet z. B. durch Zeitablauf, Beschluss der Hauptversammlung (3/4-Mehrheit), Löschung wegen Vermögenslosigkeit, Insolvenzeröffnung Liquidation erfolgt durch den Vorstand aufgrund gesetzlicher Vorschriften Nach Beendigung der Liquidation Anmeldung des Erlöschens der Firma	Wie bei der eG

Diese Übersicht beruht im Wesentlichen auf dem unter der angegebenen URL hinterlegten Rechtsformenvergleich und entspricht der Rechtslage im Juni 2009. Spätere Änderungen sind nicht berücksichtigt. Die Angaben für die AG sind von CONSULECTRA ergänzt worden. Quelle: http://www.neuegenossenschaften.de/kooperationen_genossenschaften/rechtsformenvergleich.html.

Anlage 4-2

Bewertungsmatrix Rechtsformen

Projekt KONSEQUENZ | Bewertungsmatrix Rechtsformen | CONSULECTRA

Kriterium	Gewicht	eG		e.V.		e.V. mit GmbH		Stiftung		GmbH		AG	
		Bew.	Pkt.	Bew.	Pkt.	Bew.	Pkt.	Bew.	Pkt.	Bew.	Pkt.	Bew.	Pkt.
Einfaches Gründungsverfahren	3	3	9	4	12	1	3	2	6	1	3	1	3
Geringe Gründungskosten	3	4	12	5	15	3	9	3	9	3	9	2	6
Flexibles Stimmrechtskonzept möglich	5	4	20	5	25	5	25	3	15	1	5	1	5
Einfache Aufnahme neuer Mitglieder	3	5	15	5	15	5	15	0	0	1	3	1	3
Einfluss auf Mitgliedereintritt/-wechsel gegeben	5	4	20	4	20	4	20	0	0	1	5	1	5
Kündigung für Mitglieder einfach möglich	3	5	15	5	15	5	15	0	0	1	3	1	3
Geringe Mindestanzahl für Mitglieder	2	5	10	2	4	2	4	5	10	5	10	5	10
Geringes Haftungsrisiko	4	4	16	4	16	3	12	3	12	3	12	2	8
Prüfaufwand und Veröffentlichungspflichten der Geschäftstätigkeit gering	3	2	6	5	15	2	6	3	9	2	6	2	6
Eignung für wirtschaftliche Tätigkeiten	5	5	25	1	5	2	10	3	15	5	25	5	25
Möglichkeit der Internationalität	1	5	5	5	5	5	5	5	5	5	5	5	5
Starke Außenwirkung im geschäftlichen Verkehr	3	4	12	2	6	4	12	3	9	4	12	5	15
Summe			*165*		*153*		*136*		*90*		*98*		*94*

Anlage 4-3

Satzungsentwurf

Ein Satzungsentwurf liegt der ARGE vor. Dieser Satzungsentwurf folgt den Vorgaben des Genossenschaftsverbandes. Um aber die eigenen Spielräume für die Entwicklung einer Satzung nicht einzuengen, wurde darauf verzichtet hier eine entsprechende Vorlage abzudrucken,

Anlage 4-4

Lizenzvergleich

Kurzbezeichnung	Langbezeichnung	Beispielprojekte	ohne Copyleft (kann mit proprietärer SW gemixt werden)	mit schwachem Copyleft (kann eingeschränkt mit proprietärer SW gemixt werden)	mit starkem Copyleft (kann nicht mit proprietärer SW gemixt werden)	Patentschutz	Verbreitungsgrad [%] (geschätzt aus mehreren Quellen)	Lizenzgebühren	GPL v2.1 kompatibel
	[proprietäre Software]					ja		ja	nein
MIT	Massachusetts Institute of Technology License	PuTTY, Rails	X			nein	7	nein	ja
BSD new	Berkeley Software Distribution License	PostgreQL	X			nein	5	nein	ja
BSD 3	Berkeley Software Distribution License		X			nein	2	nein	ja
Apache 2.0	Apache License 2.0	Android	X			ja	5	nein	nein
Artistic 2.0	Artistic license 2.0	Perl	X			(k.A.)	2	nein	ja
LGPL v2.1	GNU Lesser General Public License	Linux kernel		X		nein	7	nein	ja
LGPL v2.1+	GNU Lesser General Public License	Beanshell, c3p0		X		nein	2	nein	ja
LGPL v3	GNU Lesser General Public License			X		nein	2	nein	ja
MPL 2.0	Mozilla Public License 2.0	Bugzilla, Firefox		X		ja	1	nein	nein
EPL 1.0	Eclipse Public License 1.0	Eclipse		X		ja	1	nein	nein
EUPL 1.1	European Union Public License 1.1			X		(k.A.)	"-"	nein	ja
CDDL 1.0	Common Development and Distribution License 1.0	Glassfish, Netbeans		X		(ja)	"-"	nein	nein
GPL v2.1	GNU General Public License v2.1				X	nein	ca. 60	nein	"-"
GPL v2.1+	GNU General Public License v2.1+				X	nein		nein	"-"
GPL v3	GNU General Public License v3				X	nein		nein	"-"
AGPL v3	GNU Affero General Public License v3				X	nein		nein	"-"
"sonstige							6		

= empfohlene Lizenzen (mit schwachem Copyleft)

Stand: 13.02.2013