

Release-Management

Softwareentwicklung in der Wissenschaft

Sebastian Schulz

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Universität Hamburg

14. Februar 2011

- 1 Definition
- 2 Motivation
- 3 Release-Management
- 4 Beispiele aus der Freien-Software-Szene
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 Bezug zur Softwareentwicklung in der Wissenschaft
- 6 Zusammenfassung

- 1 Definition
- 2 Motivation
- 3 Release-Management
- 4 Beispiele aus der Freien-Software-Szene
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 Bezug zur Softwareentwicklung in der Wissenschaft
- 6 Zusammenfassung

Definition

Wechselwirkung zwischen Änderungs-, Konfigurations- und Release-Management

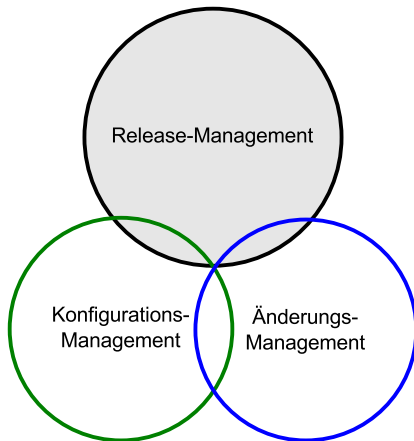


Abbildung: Zusammenhang von Release-, Änderungs- und Konfigurations-Management

Definition

Was ist Release-Management?

Definition

„Release-Management ist der Prozess, durch den Software zur Verfügung gestellt wird.“ [HW03]

Aufgaben von Release-Management

- Funktionsumfang festlegen
- Zeitplan festlegen
- Qualitätskontrolle
- Umfang und Änderungen dokumentieren
- Reproduzierbarkeit sicherstellen

Definition

Was ist Änderungs-Management?

Definition

Änderungs-Management stellt sicher, dass alle Änderungen bewertet, erprobt, implementiert und überprüft werden. [ISO05]

Aufgaben von Änderungs-Management

- Änderungen initiieren, dokumentieren und autorisieren
- Wirkung der Änderungen abschätzen/Risiko abwägen
- Implementierung koordinieren
- Abschließend Änderung überprüfen

Definition

Was ist Konfigurations-Management?

Definition

Konfigurations-Management koordiniert Maßnahmen um Konfigurationen zu lenken und zu kontrollieren.[ISO04]

Aufgaben von Konfigurations-Management

- Definition und Verfolgung von Prozessen
- Dokumentation aller Vorgänge
- Versionierung und Konfliktbehandlung
- Verwaltung von Voraussetzungen

- 1 Definition
- 2 Motivation**
- 3 Release-Management
- 4 Beispiele aus der Freien-Software-Szene
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 Bezug zur Softwareentwicklung in der Wissenschaft
- 6 Zusammenfassung

Motivation

Meine Fragestellungen am Anfang

- Was sind Anforderungen an ein Release-Management?
- Was sind Anforderung an ein wissenschaftliches Release-Management?
- Wie ist das Vorgehen bei freien Software-Projekten?

- 1 Definition
- 2 Motivation
- 3 Release-Management**
- 4 Beispiele aus der Freien-Software-Szene
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 Bezug zur Softwareentwicklung in der Wissenschaft
- 6 Zusammenfassung

Release-Manager

Was ist ein Release-Manager?

- Verantwortlich für das Release
- Letzte Entscheidungsinstanz
- Mehrere Personen für mehrere Releases denkbar
- Aufgaben delegieren (z.B.: Dokumentation, Release-Notes)

- Planen des zeitlichen Ablaufs
- Meilensteine definieren
- Fristen setzen
- Auf verschiedenen Plattformen übersetzen und testen
- Check-Liste abarbeiten
- Nach der Veröffentlichung das Feedback rückfließen lassen

- Test vor der Veröffentlichung
 - Code einfrieren
 - Kriterienkatalog erfüllen
 - Automatisierte Testabläufe
- Freigabe
 - Release-Manager gibt Software frei
 - Abnahme durch eine unabhängige Gruppe denkbar
 - Release-Notes (Abhängigkeiten, Installation)
 - Ankündigung (z.B.: Mailingliste, Website)
 - Zugriff über Webserver und Spiegelsever
 - Prüfsummen und Signierung der Archive garantiert Integrität
- Archivierung der Software
 - Leitfaden wie das Release gepackt wird
 - Für möglichst viele Benutzer verwendbar
 - Aufwand für den Benutzer gering halten (Unix, Windows)

- Umgebungen
 - Entwicklungsumgebung
 - Testumgebung
 - Vorschau
 - Produktionsumgebung
- Installation
 - Verteilungsprozess in Cluster
 - Unterstützung durch Build-Skripte
 - Handarbeit (z.B. Datenbank anpassen)

- Was ist die Idee?
 - Release identifizierbar durch Versionsnummern
 - Stabilität des Releases kennzeichnen (z.B. „0.95“)
 - Schlagwörter („alpha“, „beta“, Branching)
 - Unterstützung durch Versionsverwaltung (⇒ Git)
- Wie kann Versionsverwaltung unterstützen?
 - Meilensteine als Schlagwörter (Tags)
 - Parallele Softwareentwicklung durch Branching und Merging
- Leitfaden für Versionierung
 - Wann wird das Release als stabil bzw. nicht stabil bezeichnet
 - Welche Abstufungen gibt es
 - Kriterien für den entsprechenden Status notwendig

- Iterierbare Abfolgen
 - major.minor[.build[.revision]]
 - Nummeriert (0.9, 1.0, 1.2)
 - Alphabet (1.1a, 1.1b, 1.1c)
 - Status („alpha“, „beta“, „rc“)
 - negative Versionsnummern
- Datum („Wine 20040505“)
 - Snapshots, Nightly Build
- Jahreszahl („Windows 98“, „Ubuntu 11.04“)
- Sonstige Schemata
 - π („Tex 3.1415926“)
 - Marketing („Microsoft Office XP“)

Beispiel für Schlagwörter in Git

```
$ git tag -a "v1.1beta"  
$ git tag -l  
v1.1beta  
$ git tag -d "v1.1beta"  
Deleted tag 'v1.1beta' (was e4ee032)
```

Kleine Projekte

Was ist bei kleinen Projekten anders?

- Größe des Teams entscheidet über den Aufwand
- Wie kann das Vorgehen bei kleinen Projekten sein?
 - Aufwand gering halten
 - Versionierung verwenden
 - Dokumentation (README.txt, Changelog.txt)
- Werkzeuge
 - Makefile
 - Apache Ant
 - GNU Build System
 - Waf

Was ist „Waf“

- Auf Python basierendes Framework zum konfigurieren, compilieren und installieren von Software-Projekten
- Wichtige Funktionen
 - Erkennt selbstständig die Abhängigkeit
 - Aufgaben werden parallelisiert
 - Einfach erweiterbar
 - Gut dokumentiert („The Waf book“)
- Vordefinierte Phasen
 - `configure`: Sucht nach Bibliotheken, Compiler, etc.
 - `build`: Übersetzt das Projekt
 - `dist`: Erstellt ein Archiv mit dem gesamten Quelltext
 - `clean`: Entfernt alle mit `build` erzeugten Dateien

Listing 1: main.c

```
1 #include <stdio.h>
2 int main (void)
3 {
4     puts ("Hello World!");
5     return 0;
6 }
```

Listing 2: wscript

```
1 APPNAME = 'wafhello '
2 VERSION = '1.0 '
3 def options(opt):
4     opt.load('compiler_c ')
5 def configure(cnf):
6     cnf.load('compiler_c ')
7 def build(bld):
8     bld(features='c_program', source='main.c', \
9         target='hello')
```

Beispiel mit „Waf“ II [Tho10]

```
$ waf configure build
```

```
Setting top to           : .  
Setting out to          : ./build  
Checking for 'gcc' (c compiler) : ok  
'configure' finished successfully (0.026s)  
Waf: Entering directory './build'  
[1/2] c: main.c -> build/main.c.0.o  
[2/2] cprogram: build/main.c.0.o -> build/hello  
Waf: Leaving directory './build'  
'build' finished successfully (0.061s)
```

```
$ ./build/hello
```

```
Hello World!
```

```
$ waf dist
```

```
New archive created: wafhello-1.0.tar.bz2  
(sha='10d691febb5f87e32f3f599749be12df737158a7')  
'dist' finished successfully (0.012s)
```

- 1 Definition
- 2 Motivation
- 3 Release-Management
- 4 Beispiele aus der Freien-Software-Szene**
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 Bezug zur Softwareentwicklung in der Wissenschaft
- 6 Zusammenfassung

- Was ist der Linux Kernel?
 - Projekt 1991 durch Linus Torvalds gegründet
 - Er ist heute noch hauptverantwortlich
 - GNU GPL lizenziert
- Versionierung (2.{Major}.{Minor}[.Revision])
 - *Major*: Nur bei Änderung der Systemarchitektur
 - *Minor*: Hinzunahme von neuen Funktionen
 - *Revision*: Wird zur Fehlerbehebung verwendet
- Verantwortliche Personen
 - Hauptzweig verantwortet Linus Torvalds
 - Entwickler übernehmen später die Pflege für stabile Versionen
 - 2.4-Zweig wird nach wie vor gepflegt

- Test und Abnahme
 - Entwicklerversion („-mm Kernel“)
 - Nach zwei Wochen keine Aufnahme von neuen Features
 - Positives Feedback von der Mailingliste
- Freigabe
 - Ankündigung auf der Mailingliste
 - Verfügbar auf kernel.org und zahlreichen Spiegelserver
 - Dateien werden signiert mit GnuPG
 - Tarball, Patch
- Linux-Distributoren pflegen teilweise eigene Versionen

- Was ist SVN?
 - Versionsverwaltung
 - Als Nachfolger von CVS entwickelt
 - Entwickelt durch CollabNet
 - Seit 2009 unterstützt durch die „Apache Software Foundation“
 - Apache License v2.0 lizenziert
- Verantwortliche Organisation und Personen
 - Committers
 - Release-Manager
 - Patch Manager

- Versionierung
 - Alle Versionen sind grundsätzlich stabil
 - Versionsnummer mit „alpha“, „beta“ oder „rc“ sind nicht stabil
 - Minor Releases bleiben zueinander kompatibel
- Test und Abnahme
 - Regressionstests müssen erfolgreich sein
 - Punkte im Issue-Tracking-System müssen erledigt sein
 - Build-Farm
- Veröffentlichung
 - Quellcode wird als Archiv(.tar.bz2, .zip) zur Verfügung gestellt
 - Nur auf den Servern von collabNet verfügbar
 - Dritte Distributoren stellen binäre Pakete bereit
- Dokumentation zum Release-Prozess auf der Website verfügbar

Apache HTTP Server

Übersicht und Versionierung

- Webserver
- Apache License v2.0 lizenziert
- Projekt ist meritokratisch¹ organisiert
 - Gremium für das Projekt-Management
 - Release-Manager wird ein Freiwilliger aus dem Gremium
 - Eingaben aus der Community gewünscht
- Drei Klassifizierungen (Alpha, Beta, General Availability (GA))
- Stabile Version (2.{gerade-Zahl}.{revision})
 - Keine experimentell Erweiterungen
 - Kompatibilität zu höheren Versionen wahren
- Nicht stabile Version (2.{ungerade-Zahl}.{revision})
 - Entwicklern von Erweiterungen frühes Feedback ermöglichen
 - Projekt-Management Gremium entscheidet über den Status und die zukünftige Versionsnummer

¹Aufnahme in die Community aufgrund von Leistungen

Apache HTTP Server

Test, Abnahme, Veröffentlichung

- Test und Abnahme
 - Ankündigung auf Entwickler- und Test-Mailingliste
 - automatisierte Tests
 - stabiler Release-Kandidat läuft auf apache.org
 - Freigabe durch Wahl der Projekt-Mitglieder
- Veröffentlichung
 - Quellcode wird als Archiv bereitgestellt
 - Binäre Pakete für diverse Plattformen verfügbar
 - Weltweit verteilte Spiegelserver
 - Dateien sind mit GnuPG signiert

- 1 Definition
- 2 Motivation
- 3 Release-Management
- 4 Beispiele aus der Freien-Software-Szene
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 **Bezug zur Softwareentwicklung in der Wissenschaft**
- 6 Zusammenfassung

Bezug zur Softwareentwicklung in der Wissenschaft

- Kontext von Wissenschaftler
 - Wissenschaftler haben häufig keine oder eine geringe Ausbildung im Bereich der Software-Entwicklung
 - Software-Entwickler müssen sich in die Thematik einarbeiten
- Oft kleine Projekte
- Andere Produktionsumgebung in der Wissenschaft
 - Entwickler ist häufig auch Endbenutzer
 - Verteilungsprozess in Cluster
- Gute Dokumentation sehr wichtig
 - Ergebnisse müssen reproduzierbar sein
 - Archivierung und Verfügbarkeit müssen gewährleistet sein
 - Release-Notes (Installation der Software)
- Von freien Software-Projekten lernen
 - Seit vielen Jahren erfolgreiche Konzepte vorhanden
 - Wissen nachhaltig gesichert

- 1 Definition
- 2 Motivation
- 3 Release-Management
- 4 Beispiele aus der Freien-Software-Szene
 - Linux Kernel
 - Subversion
 - Apache HTTP Server
- 5 Bezug zur Softwareentwicklung in der Wissenschaft
- 6 Zusammenfassung

- Release-Management muss geplant und dokumentiert werden
 - Was muss ich an meinem System konfigurieren
 - Wie bekomme ich die Software zum Laufen
- Versionierung ist wichtig und notwendig
- Werkzeuge können auf dem Weg sehr hilfreich sein (Git, Waf)
- Freie Software-Projekte können als gutes Vorbild dienen

Vielen Dank!

- [CSK02] Clement-Smith, L. ; Kerkhoff, W.:
Software Release Management.
(2002)
- [EJ09] Easterbrook, S.M. ; Johns, T.C.:
Engineering the software for understanding climate change.
In: *Computing in Science & Engineering* 11 (2009), Nr. 6, S. 65–74. –
ISSN 1521–9615
- [Ere03] Erenkrantz, J.R.:
Release management within open source projects.
In: *Proc. 3rd. Workshop on Open Source Software Engineering Citeseer*,
2003
- [Goo11] Google Project Hosting:
Waf.
<http://code.google.com/p/waf/>, Februar 2011
- [HW03] Hoek, A. Van d. ; Wolf, A.L.:
Software release management for component-based software.
In: *Software: Practice and Experience* 33 (2003), Nr. 1, S. 77–98. –
ISSN 1097–024X

- [ISO04] ISO, DIN:
10007.
In: *Qualitätsmanagement– Leitfaden für Konfigurationsmanagement (ISO 10007: 2003)* (2004)
- [ISO05] ISO, DIN:
20000-1.
In: *Service Management: Specification (ISO/IEC 20000 Part 1: 2005)* (2005)
- [MHP07] Michlmayr, M. ; Hunt, F. ; Probert, D.:
Release management in free software projects: Practices and problems.
In: *Open Source Development, Adoption and Innovation* (2007), S. 295–300
- [The09] The Apache Software Foundation:
Subversion Submitted to Become a Project at The Apache Software Foundation.
http://www.apache.org/foundation/press/pr_2009_11_04.html,
4. November 2009
- [The11a] The Apache Software Foundation:
Apache HTTP Server Release Guidelines.
<http://httpd.apache.org/dev/release.html>, Februar 2011

- [The11b] The Apache Software Foundation:
Community Roles.
<http://subversion.apache.org/docs/community-guide/roles.html>,
Februar 2011
- [The11c] The Apache Software Foundation:
Getting Involved with Apache Subversion.
<http://subversion.apache.org/contributing.html>, Februar 2011
- [Tho10] Thomas Nagy:
Waf tutorial.
<http://waf.googlecode.com/svn/docs/apidocs/tutorial.html>,
2. Dezember 2010
- [Wik11a] Wikimedia:
Release Management.
[http://de.wikipedia.org/w/index.php?title=Release_](http://de.wikipedia.org/w/index.php?title=Release_Management&oldid=83836978)
[Management&oldid=83836978](http://de.wikipedia.org/w/index.php?title=Release_Management&oldid=83836978), 1. Februar 2011
- [Wik11b] Wikimedia:
Software versioning.
[http://en.wikipedia.org/w/index.php?title=Software_](http://en.wikipedia.org/w/index.php?title=Software_versioning&oldid=409736919)
[versioning&oldid=409736919](http://en.wikipedia.org/w/index.php?title=Software_versioning&oldid=409736919), 6. Februar 2011