Seminararbeit

Thema:

Continuous Delivery

Stefan Kruk
geboren am 14.08.1992
Matr.-Nr.: xxxxxxx

An der Fachhochschule Dortmund im Fachbereich Informatik erstellte
Seminararbeit
im Studiengang Softwaretechnik (Dual)

Betreuer: Dr. Kim Lauenroth

Fachbereich Informatik

Dortmund, 15. Mai 2016

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

In diesem Kapitel werden zunächst die Grundlagen erläutert, welche für das Verständnis dieser Arbeit notwendig sind. Außerdem werden in den Grundlagen alle wichtigen Begriffe erklärt, die zum Verständnis des Themas beitragen und notwendig sind. Anschließend wird auf die zugrundeliegende Problemstellung eingegangen und darauf aufbauend auf das Ziel der Arbeit.

1.1 Grundlagen

Grundsätzlich ist das in dieser Arbeit behandelnde Thema für jede Person mit einer allgemeinen Informatikausbildung ohne weiteres zu verstehen. Es kann bei dieser Personengruppe, die Kenntnisse über grundsätzlichen Prozess einer Softwareentwicklung vorausgesetzt werden. Trotzdem soll im weiteren Verlauf einige Begriffe genauer erklärt werden.

Delivery
Delivery (zu deutsch Ausliefern) beschreibt das Ausliefern, als das Verteilen, von Artefakten. Dabei kann das Artefakt eine ganze Applikation oder nur ein Service in einer Service Orientierten Architektur sein.

Deployment
Unter Deployment (zu deutsch Softwareverteilung) versteht man das installieren, eines Artefaktes. Auch hier kann ein Artefakt eine ganze Applikation oder nur ein

Service sein. Die beiden Begriffe „Delivery“ und „Deployment“ werden im Kapitel 3 Continuous Delivery noch weiter erklärt und von einander abgegrenzt.

Softwareentwicklung

Im allgemeinen wird Softwareentwicklung als ein Prozess zur Erstellung von Software verstanden, welche folgende Phase beinhaltet:

1. Planung
2. Analyse
3. Entwurf
4. Implementierung
5. Validierung und Verifikation
6. Abnahme
7. Release

Im Rahmen dieser Arbeit wird der Begriff Softwareentwicklung jedoch als Synonym für die Phasen vier bis sieben genommen. Kapitel 1.2 Problemstellung verdeutlicht noch einmal genauer, die Relevanz dieser Phasen.

Qualität

Qualität ist nach der DIN 55350 wie folgt definiert: "Qualität ist die Beschaffenheit einer Einheit bezüglich ihrer Eignung, festgelegte und vorausgesetzte Erfordernisse zu erfüllen. Dabei wird ergänzend die Einheit als materieller oder immaterieller Gegenstand der Betrachtung und die Beschaffenheit als Gesamtheit der Merkmale und Merkmalswerte definiert."

Qualitätssicherung

Die Qualitätssicherung sollte ein in die Softwareentwicklung integrierter Prozess sein, um die Einhaltung der für das Projekt festgelegten Qualitätsmerkmale zu überprüfen und so die Qualität des entstehenden Produktes zu gewährleisten. Zur Überprüfung dieser Merkmale können verschiedene Werkzeuge eingesetzt werden, die in Kapitel 3 Continuous Delivery genauer erläutert werden.

1.2 Problemstellung

Wir leben in einem Zeitalter in der fast alles über das Internet gesteuert wird. Viele Unternehmen, nennen wir sie Internetunternehmen, haben sich daher darauf spezialisiert, ihre Dienste nur im Internet anzubieten. Jedoch kann sich das Interessen der Nutzer relativ schnell ändern und durch die Möglichkeit einfach und unkompliziert zu anderen Anbietern zu wechseln entsteht ein großer Wettbewerbsdruck bei den einzelnen Unternehmen. Ist das Time-to-Market eines Unternehmens daher sehr Zeitaufwändig, kann dies zum Verlust vieler Kunden und daher im schlimmsten Fall auch zur Insolvenz des Unternehmens führen.

Eberhard Wolff beschreibt in [3, S. 2 ff.] einen Fall eines fiktiven E-Commerce Unternehmens. Das Unternehmen hatte nur eine große Software, den E-Commerce Shop. Durch neue Angebote und das dauerhaft ändernde Interesse der Kunden mussten neue Funktionen regelmäßig und in möglichst kurzen abständen dem Kunden zugänglich gemacht werden. Dies wurde jedoch durch die Tatsache behindert, dass die Software über die Jahre gewachsen ist und das erneute Ausliefern der Software für eine Funktion sich nicht lohnte. Daher wurde nur einmal im Monat neu Deployed. Der Prozess wurde außerdem dadurch behindert, dass die Qualitätssicherung zwar ein Teil der Softwareentwicklung war, jedoch Tests nur manuell ausgeführt worden sind, wodurch regelmäßig Fehler übersehen wurden.

die Software wurde schließlich mit Fehlern ausgeliefert und es stellte sich erst am nächsten Tag, oder schlimmer nach einer Woche, heraus, dass sie nicht einwandfrei funktionierte. Entwickler mussten also ihre Arbeit unterbrechen und den Fehler finden und beheben. Da jedoch ein wenig Zeit vergangen ist, seit dem die Entwickler an diesem Teil des Codes gearbeitet haben, müssen sie sich erst wieder einarbeitet, bis sie den Fehler finden und beheben können.

Das Unternehmen hat also eine große TTM-Zeit und dadurch hohe kosten. Zusätzlich entstehen immer wieder fehlerhafte Releases wodurch zusätzliche Kosten bzw. Einbußen entstehen.

1.3 Ziel der Arbeit

In dieser Arbeit soll Continuous Delivery genauer erläutert und dabei folgende Leitfragen beantwortet werden:

1. Was ist Continuous Delivery?
2. Wie sind die konkreten Phasen definiert?
3. Was sind die Vor- und Nachteile von Continuous Delivery?
4. Welche Werkzeuge werden benötigt?
5. Wie kann man Continuous Delivery in ein bestehenden Entwicklungsprozess einbinden?

Die erste Leitfrage soll den Begriff Continuous Delivery und seine Herkunft erläutert. Dabei wird kurz auf die Geschichte der Softwareentwicklungsprozesse eingegangen und erläutert wie sich der Prozess zum heutigen unterscheidet. Außerdem wird in diesem Zusammenhang noch einmal erläutert, warum sich die Prozesse verändert haben bzw. verändert werden mussten.

Die zweite Leitfrage führt die konkreten Phasen ein und beleuchtet die wesentlichen Unterschiede zueinander, sowie ihre Bedeutung und Wichtigkeit. Zudem sollen die Phasen zueinander abgegrenzt werden und mit denen bestehender Entwicklungsphasen verglichen werden.

Die Vor- und Nachteile dieses Prozesses sollen mit der dritten Leitfrage geklärt werden. Jeder Prozess hat Vor- und Nachteile. In diesem Zusammenhang wird daher beleuchtet, wann es sich für ein Unternehmen lohnt Continuous Delivery einzuführen und wann nicht. Zusätzlich soll der Auffand beschrieben werden, der dieser Prozess mit sich bringt.

Nachdem erläutert wurde, was Continuous Delivery ist, wie es strukturiert ist und welche allgemeinen Vor- und Nachteile der Prozess mit bringt, soll in der vierten Leitfrage geklärt werden, welche Werkzeuge benötigt werden, um den Prozess nutzen zu können. Dabei soll kurz auf jedes einzelne Werkzeug eingegangen und erläutert werden, wofür es gut ist und für welche Phase es wichtig ist.

Abschließend wird die in Kapitel 1.2 erläuterte Problemstellung noch einmal aufgefasst und damit für die fünfte Leitfrage ein konkreter Anwendungsfall eingeführt. Dazu wird noch einmal genauer auf den Anwendungsfall eingegangen, um diesen mit dem Prozess des Continuous Delivery Ansatzes zu lösen.

Kapitel 2

Systematische Literaturrecherche

Die Systematische Literaturrecherche stellt die Basis der Quellen und Informationen, des in Kapitel 3 Continuous Delivery vorgestellten Inhalts dar.

In diesem Kapitel wird daher aufgezeigt, wie die herangezogenen Quellen und Informationen ermittelt, welche Auswahl- und Ausschlusskriterien festgelegt wurden und was für Ergebnisse die entsprechenden Suchanfragen gebracht haben.

2.1 Auswahlkriterien und Suchbegriffe

Um die Auswahl der Literaturen zu Filtern, wird zunächst allgemeine Auswahl- und Ausschlusskriterien definiert, mit denen die im Rechercheprotokoll angegebenen Suchergebnisse begründet werden. Anschließend wird der Zugrundlegende Ansatz (P.I.C.O.C.) genauer erläutert und darauf aufbauend Suchbegriffe in Deutsch und Englisch definiert.

2.1.1 Inhaltliche Auswahlkriterien

Folgende Inhaltliche Auswahlkriterien wurden für die Recherche festgelegt: spacing

a) Dokument ist über oder hat direkten Bezug zu Continuous Delivery

b) Dokument beschreibt die Einsatzmöglichkeiten von Continuous Delivery

c) Dokument beschreibt wichtige Technologien für den Einsatz von Continuous Delivery

Mit Hilfe der Auswahlkriterien wird im Rechercheprotokoll die Relevanz der gefundenen Materialien begründet. Sie werden über die Buchstaben a) bis c) referenziert.

2.1.2 Inhaltliche Ausschlusskriterien

Folgende Inhaltliche Ausschlusskriterien wurden für die Recherche festgelegt. spacing

d) Dokument ist zu allgemein und hat nur am Rande etwas mit dem Thema zu tun (Bsp.: Enthält den Begriff nur in Referenzen)

e) Inhalt ist zu speziell (Fokus liegt auf dem Einsatz bestimmter Technologien oder Frameworks)

f) Dokument beschreibt wie ein Werkzeug und/oder Framework aufgebaut ist und funktioniert.

g) Inhaltsangabe, Einleitung, Fazit oder Abstract sind nicht Aussagekräftig bzw. lassen keine Hinweise auf den Einsatz oder der Beschreibung von Continuous Delivery zu

h) Inhalt trägt nicht zur Beantwortung der Leitfragen bei.

i) Dokument ist nicht in Deutsch oder Englisch (Material kann aufgrund der Sprachbarriere nicht verwendet werden)

j) Dokumente ist älter als 5 Jahre

Mit Hilfe der Ausschlusskriterien wird im Rechercheprotokoll die Irrelevanz der gefundenen Materialien begründet. Sie werden über die Buchstaben d) bis j) referenziert.

2.1.3 P.I.C.O.C.

Die Suchbegriffe werden mit Hilfe des PICOC-Ansatzes (siehe [1]) ermittelt. Der Begriff besteht dabei aus folgenden Aspekte: spacing

Population

Interventation

Comparison

Outcomes

Context

Population

Die Population, zu Deutsch etwa "Bevölkerung", beschreibt eine Teilmenge von relevanten Personen, wie Tester, Manager, Novizen oder Experten. Aber auch Applikationsfelder wie IT-Systeme, Command und Control Systeme oder Industrielle Gruppen wie Telekommunikations- oder kleine IT-Unternehmen.

Bezüglich der Population werden hier keine Einschränkungen gemacht, da Continuous Delivery für jede Teilmenge der gerade genannten Personen, Gebiete und Unternehmen von Bedeutung ist. Da das Thema jedoch stark mit dem Thema DevOp-Teams tangiert, wird hier zusätzlich zu diesem Thema Suchanfragen gestellt. Eine Einschränkung des Thema auf DevOps besteht jedoch nicht.

Intervention

Bei der Intervention handelt es sich um die eingesetzten Methoden, Werkzeugen, Technologien oder Prozeduren für ein bestimmtes Problem. Im Rahmen von Continuous Delivery bedeutet dies, die Werkzeuge die nötig sind um eine Continuous Delivery Pipeline aufbauen und durchführen zu können.

Comparison / Vergleich

Der Vergleich (Comparison) beschreibt die Werkzeuge einer Kontrollgruppe, welche mit denen aus der Intervention verglichen werden.

Outcomes / Auswirkung

Bei der Auswirkung soll mit Hilfe von Zahlen und Faktoren der Vergleich erläutert werden zwischen der Intervention und der Kontrollgruppe. Hier kann zum Bei-

spiel die Zeitspanne des Time-to-Market verglichen werden, was wiederum einen Einblick in die Kosten für die Implementierung/Auslieferung eines Produktes/einer Funktion gibt.

Context / Kontext

Der Kontext ist hier die Software-Entwicklung in Bezug auf Aufwand und Kosten der Produktion. Bei Continuous Delivery spielt ebenfalls der Aufbau des Teams eine Rolle. Wie bei der Population spielen hier DevOp-Teams eine zentrale Rolle.

Suchbegriffe

Im Nachfolgenden sind für jeden, der in Abschnitt 2.1.3 P.I.C.O.C genannten Aspekte Synonyme in Deutsch und Englisch festgehalten, welche die Basis für die verwendeten Suchanfragen bilden.

Kategorie	**Deutsche Begriffe**	**Englische Begriffe**
Population	DevOps	
Intervention	• Continuous Delivery • Docker • Jenkins • Deployment • pipeline	
Comparison	• Manuel (Deployment)	
Outcomes	• Kosten • Tests • Zeit	• costs • tests • time / duration
Context	• Technik • Prinzipien • Praxis • Anwendung	• techniques • principles • practice • usage

Der Begriff Deployment is sowohl in Intervention und Comparison, da dieser ein zentraler Begriff in beiden Mengen ist und unterschiedlich verwendet werden kann.

2.1.4 Zusätzliche Anmerkungen zur Recherche

Für diese Arbeit wurden zusätzlich folgenden Einschränkungen, für die Durchführung der systematischen Suche festgelegt:

Zugänglichkeit Materialien müssen entweder öffentlich oder für den Personenkreis, für die diese Arbeit angefertigt wird, ohne weitere Einschränkungen, wie ein notwendiges Login, zugänglich sein. da ansonsten der Beschaffungsaufwand zu hoch ist und die Materialien nicht für andere Studenten bzw. den Dozenten zugänglich wären.

Auswahl der Materialien Materialien werden anhand der Inhaltsübersicht, Einlei-

tung, Fazit oder eines Abstracts ausgewählt, da ein einlesen in einzelne Kapitel zu viel Zeit beanspruchen würde.

Suchergebnisse Bei auffinden von Großen Mengen bei der Suche, wird zunächst versucht durch eventuelle Filtermöglichkeiten, die Relevanz der Materialien zu ordnen und die ersten 20 Resultate begutachtet. Dabei wird die Qualität der Ordnung und die Effizienz des zugrunde liegenden Algorithmus der Suchmaschine überlassen. Sollten keine Filtermöglichkeiten vorhanden sein, wird anhand der Kurzbeschreibungen und der Titel die ersten 20 besten Treffer ausgewählt.

2.2 Quellen und Suchanfragen

2.2.1 Suchstrategie

Die Suchstrategie basiert auf der eigenen bereits gesammelten Erfahrung und vorhandenen Materialien (siehe z.B. [3]. Auf Basis dessen wurden unter anderem die in 2.1.3 Suchbegriffe genannten Suchbegriffe definiert. Zusätzlich wurden diese aus weiteren gefundenen Materialien aufgebaut.

2.2.2 Suchstring

In Abschnitt 2.1.3 Suchbegriffe wurden Suchbegriffe festgelegt, auf denen die Literaturrecherche basiert. Diese werden zu nächst mit einem “ODER"bzw. “OR„ verknüpft. Im Zweiten Schritt werden die Suchbegriffe mit einem “UND‘"bzw. ÄND„ verknüpft.

Da bestimmte Begriffe mit verschiedenen Kontexte in Verbindung gebracht werden können. Wird zunächst ein Suchstring aufgebaut, der als Erstes Element (“Continuous Delivery„ OR “Deployment„) besitzt. Nachfolgend werden nun die einzelnen Suchstrings aufgelistet, die verwendet wurden, um die systematische Literaturrecherche durchzuführen. Da Suchmaschinen einen komplexen Algorithmus aufweisen, wird ebenfalls davon ausgegangen, dass auch eine aneinander reihen von den in 2.2.2 Suchstring Ausdrücken (immer mit dem Führenden “Continuous Delivery„

Oder “Deployment‘") zum gewünschten Ergebnis führt. Dies beruht auf den Eigenschaften moderner Suchalgorithmen. Nach einer ersten suche, hat sich ergeben, dass einige Begriffe irreführend für die Suchmaschinen sind, wodurch Materialien gefunden wurde, welche absolut nichts mit dem Thema zu tun haben. Daher sind nur die folgenden Suchanfragen von Bedeutung.

$S_{Komplex}$ = ("’Continuous Delivery"‘ OR "’Deployment"‘)
AND
(“SSH,, OR “FTP,, OR “Deployment,, OR“Kosten,, OR “costs,, OR “Tests,, OR “Zeit,, OR “time,, OR “duration,, OR “Fehler,, OR “error,, OR “Qualitätssicherung,, OR “quality assurance,, OR “Technik,, OR “techniques,, OR “Prinzipien,, OR “principles,, OR “Praxis,, OR “practice,, OR “Anwendung,, OR “usage")

$S_{deployment}$ = ("’Continuous Delivery"‘ OR "’Deployment"‘)
AND
(“Deployment,,)

$S_{pipeline}$ = ("’Continuous Delivery"‘ OR "’Deployment"‘)
AND
(“Pipeline,,)

2.2.3 Quellen

In [1] wurden einige elektronische Standartquellen der Informatik genannt. In der nachfolgenden Tabelle, werden diese noch einmal aufgelistet und kurz begründet, warum jeweilige Quellen gewählt bzw. nicht gewählt wurde.

Quelle	**gewählt**	**Begründung**
ACM Digital library	✘	Nicht frei zugänglich
Citeseer library (citiseer.ist.psu.edu)	✔	Fokus auf Informatik, PDFs kostenlos verfügbar
EI Compendex (www.engineeringvillage2.org)	✘	Registrierung erforderlich
IEEExplore	✔	über die FH frei zugänglich, relevante Quelle für Informatiker
Inspec (www.iee.org/Publish/INSPEC/)	✘	Registrierung erforderlich
Google scholar (scholar.google.com)	✔	Großes Angebot, meist als PDF frei zugänglich
ScienceDirect (www.sciencedirect.com)	✘	PDFs sind nur teilweise frei zugänglich.

2.3 Ergebnisse der Recherche

In der Nachfolgenden Tabelle sind, nach Suchmaschine geordnet, die Ergebnisse der systematischen Literaturrecherche. Es wird der Suchstring, die Anzahl der gefundenen Ergebnisse, die Anzahl er betrachteten Ergebnisse, die Anzahl der relevanten Ergebnisse, die Anzahl der gewählten Ergebnisse und das Datum an dem die Suche durchgeführt worden ist.

Citeseer library					
Suchstring	gesamt	betrachtet	relevant	gewählt	Datum
$S_{komplex}$	350.724	0	0	0	13.05.2016
$S_{deployment}$	207.039	0	0	0	13.05.2016
$S_{pipeline}$	120.126	0	0	0	13.05.2016

IEEExplore					
Suchstring	gesamt	betrachtet	relevant	gewählt	Datum
$S_{komplex}$	0	0	0	0	13.05.2016
$S_{deployment}$	60	10	10	4	13.05.2016
$S_{pipeline}$	22	3	3	2	13.05.2016

Google scholar					
Suchstring	gesamt	betrachtet	relevant	gewählt	Datum
$S_{komplex}$	1.580	1	0	0	13.05.2016
$S_{deployment}$	224.00	2	1	1	13.05.2016
$S_{pipeline}$	80.800	2	1	1	13.05.2016

Der Unterschied zwischen der Anzahl der betrachteten Ergebnisse und der gewählten, kommt durch erneut gefundene, bzw. durch Materialien, welche nicht ansatzweise mit dem Thema zu tun haben zu Stande.

Kapitel 3

Continuous Delivery

Kapitel 4

Zusammenfassung und Ausblick

4.1 Zusammenfassung

4.2 Kritische Reflektion

4.3 Ausblick

Literaturverzeichnis

[1] KITCHENHAM: *Guidelines for performing Systematic Literature Reviews in Software Engineering*

[2] WIESMANN, Prof. Dr. D.: *Skript der Veranstalltung SSWT D"*. FH-Dortmund. 2014

[3] WOLFF, Eberhard: *Continuous Delivery - Der pragmatishe Einstieg*. 1. Auflage. dpunkt.verlag, 2015. – ISBN 978-3-86490-208-6

Anhang A

Anhang

A.1 Glossar

Time-to-Market (TTM) Unter dem Begriff Time-to-Market wird die Zeit von der Produktentwicklung bis zur Auslieferung auf dem Markt verstanden. In dieser Zeit müssen Kosten für die Erstellung/Entwicklung aufgebracht werden, es spielt aber keine Umsätze ein. Daher strebt jedes Unternehmen eine möglichst geringe Time-to-Market Zeit an. Insbesondere wenn es um Wettbewerb geht, muss diese Zeit kurz gehalten werden.

A.2 Rechercheprotokoll

IEEExplore		
Suchstring	Titel	Author
$S_{deployment}$	End to End Automation on Cloud with Build Pipeline: The Case for DevOps in Insurance Industry, Continuous Integration, Continuous Testing, and Continuous Delivery	Miteshi Soni
$S_{deployment}$	Continuously delivering your network	Steffen Gebert
$S_{deployment}$	Continuous Delivery: Huge Benefits, but Challenges Too	Lianping Chen
$S_{deployment}$	Towards Architecting for Continuous Delivery	Lianping Chen
$S_{pipeline}$	Continuous Delivery with Jenkins: Jenkins Solutions to Implement Continuous Delivery	Valentina Armenise
$S_{pipeline}$	Continuous Delivery: Huge Benefits, but Challenges Too	Valentina Armenise

Google Scholar		
Suchstring	Titel	Author
$S_{deployment}$	Continuous Integration	Martin Fowler
$S_{pipeline}$	Continuous Delivery with Jenkins: Jenkins Solutions to Implement Continuous Delivery	Valentina Armenise
$S_{pipeline}$	(IEEExplore) Continuous Delivery? Easy! Just Change Everything (Well, Maybe It Is Not That Easy)	Steve Neely