

syngenio in der Fachpresse

Medium: Javamagazin
**Thema: Qualität in Java-Projekten, Teil 5:
Die Kundensicht**
Autor: Detlef Ahlers
Ausgabe : 11.2005

Glücksgefühle einer Jungfernfahrt

„Ich will, dass dieses Problem bis Sonnenuntergang gelöst ist. Und schafft mir den Entwickler herbei, der dafür verantwortlich ist.“ Was Tag für Tag Hunderten von IT-Verantwortlichen durch den Kopf geht – und gelegentlich auch ausgesprochen wird – beschreibt im Kern die unterschiedlichen Glücksgefühle bei einer Jungfernfahrt mit einem Neuwagen und der Inbetriebnahme einer neuen Software.



Kontakt & weitere Informationen:

syngenio AG
Ivonne Machnacz
Andreas-Hermes-Straße 3
D-53175 Bonn
Fon +49 (0)2 28-6 20 95-100
Fax +49 (0)2 28-6 20 95-150
ivonne.machnacz@syngenio.de
www.syngenio.de

Qualität in Java-Projekten, Teil 5: die Kundensicht

Glücksgefühle einer Jungfernfahrt

■ VON DETLEF AHLERS

„Ich will, dass dieses Problem bis Sonnenuntergang gelöst ist. Und schafft mir den Entwickler herbei, der dafür verantwortlich ist.“ Was Tag für Tag Hunderten von IT-Verantwortlichen durch den Kopf geht – und gelegentlich auch ausgesprochen wird – beschreibt im Kern die unterschiedlichen Glücksgefühle bei einer Jungfernfahrt mit einem Neuwagen und der Inbetriebnahme einer neuen Software.



Was war passiert? Eigentlich hätte mit dem neuen Release alles besser werden sollen. Eine Reihe von Fehlern war beseitigt worden, die Eingabefelder waren für die Anwender komfortabler gestaltet worden und einige kleinere Verbesserungen hatte man auch mit eingebaut. Der Betrieb hatte lange darauf gewartet. Und dann der Super-GAU: Nach der Umstellung in den frühen Morgenstunden zeigte

die Software bereits am Vormittag unter Last gravierende Fehler, die dazu führten, dass sehr schnell mehrere Fachabteilungen nicht mehr arbeiten konnten. Die Anwender waren wütend, dem IT-Leiter standen die Schweißperlen auf der Stirn.

Dieses Szenario ist denkbar und wahrscheinlich auch möglich. Häufig sind es aber kleinere Probleme, die die Freude an einem neuen Release trüben.

Implementierung nicht berücksichtigt wurde.

- Es kommt zu Pseudoproblemen wegen unzureichender Schulung der Anwender oder des IT-Mitarbeiters.

Ihr Feedback

Wir freuen uns auf Ihr Feedback zu unserer „Qualität in Java-Projekten“-Reihe. Bitte schreiben Sie uns: qualitaet@javamagazin.de.

- Neue Funktionen verhalten sich anders als erwartet.
- Die im Operating etablierten Verfahren und Abläufe funktionieren nicht mehr.
- Das neue Release stellt erhöhte Anforderungen an die Hardware, was bei der

Qualität in Java-Projekten

Das passierte in den vorigen Ausgaben

- Teil 1: Mit agilem Qualitätsmanagement zum Projekterfolg
- Teil 2: Der anwenderbezogene Qualitätsaspekt
- Teil 3: Last und Performance
- Teil 4: das Vier-Augen-Prinzip
- **Teil 5: die Kundensicht**

- Mangels Testvorgaben konnte der Erfolg der Implementierung nicht abschließend verifiziert werden

Der Schuldige

Nachdem das akute Problem gelöst ist und man wieder Zeit zum Nachdenken hat, begibt man sich mit einer gewissen Leidenschaft auf die Suche nach der Ursache, die einer mittelalterlichen Tradition folgend gern mit einem Namen versehen wird. Und weil der Entwickler, der durch sein persönliches Engagement maßgeblich zur Lösung beigetragen hat, noch greifbar ist, erklärt man ihn der Einfachheit halber zum Alleinschuldigen.

Wenn der Entwickler die Ratschläge aus den vorangegangenen Artikeln dieser Serie umgesetzt hat, ist er ein schlechter Kandidat für eine solche Schuldzuweisung. Gleichwohl ist auch er nicht davor sicher, in das Kreuzfeuer der Kritik zu geraten.

Probleme der beschriebenen Art sind zu zahlreich und zu häufig, als dass man ihre Ursachen pauschal in der fehlenden Kompetenz einzelner Personen suchen

darf. Sehr viel wahrscheinlicher sind unzureichende Schnittstellen zwischen der Entwicklung und dem Betrieb, die häufig wie folgt begründet sind:

1. Die Entwicklung versteht sich als Lieferant von Software und agiert ähnlich wie externe Softwarelieferanten. Es gibt wenige sehr formelle Schnittstellen zum Betrieb. Die Fachabteilungen werden als Auftraggeber angesehen, der Betrieb ist ein notwendiges Übel, das man nur deshalb braucht, weil die Applikationen ja schließlich auf irgendeiner Hardware laufen müssen.
2. Der Betrieb begründet sein Primat innerhalb der IT mit der Unterstellung, dass Software grundsätzlich von verschiedenen Quellen bezogen werden kann. Die geringe Wertschätzung gegenüber der Entwicklung im eigenen Hause kommt in einer unnötigen Abgrenzung zum Ausdruck.
3. Entwicklung und Betrieb haben gleichermaßen das Verständnis einer engen und partnerschaftlichen Zusammenarbeit. Allerdings leben beide in ihren Kul-

turen, haben eigene auf unterschiedlichen Methoden basierende Prozesse, die dem jeweils anderen nicht hinreichend bekannt sind, und scheuen sich noch davor, ihre Prozesslandschaften zu konsolidieren.

Während die ersten beiden Problemfelder sicherlich in einer besonderen Weise und damit jenseits der Möglichkeiten dieses Artikels gewürdigt und gelöst werden müssen, bietet das dritte beschriebene Missverhältnis zumindest die Chance einer Lösung. Beginnen wir einfach damit, uns gegenseitig zu verstehen.

Der Betrieb

Vergessen wir das Extreme Programming und den Rational Unified Process – zumindest für die Dauer der Lektüre dieses Artikels. Der Betrieb ist eine andere Welt mit anderen Regeln.

ITIL (Information Technology Infrastructure Library) ist das am weitesten akzeptierte Framework für ein IT-Prozess-Management. In den späten 80ern vom OGC (Office of Government Com-

merce) entwickelt, beschreibt ITIL in einer Sammlung von derzeit 45 Büchern im Rahmen eines Best-Practices-Ansatz ein IT-Prozess-Management. ITIL beschreibt die Prozesse, die notwendig sind, um eine IT-Infrastruktur effektiv und effizient zu managen, sodass die zwischen der IT-Organisation und deren „Kunden“ vereinbarten Services erbracht und die dazugehörigen Service-Level garantiert werden können. Im Kern enthält ITIL zehn Prozesse und die Funktion Service Desk, die jeweils klare und messbare Ziele haben und vielfältige Schnittstellen zueinander besitzen (Tabelle 1).

Der Servicegedanke

Es ist noch nicht sehr lange her, als Server, Mainframes, Netze und Applikationen im Mittelpunkt der Betrachtung der IT-Verantwortlichen standen. Kunden existierten allenfalls dort, wo es sich um zahlende Kunden handelte – und auch hier waren sie keinesfalls immer die Könige.

Einer der wesentlichen Ansätze von ITIL ist nun der Servicegedanke. Ausgehend von der Überlegung, dass IT in einem Unternehmen keineswegs unantastbar ist, muss sie sich genau wie jede andere Organisationseinheit der Frage stellen, welchen Mehrwert sie für das Unternehmen erbringt. Da ein Finanzbuchhalter keinen unmittelbaren Nutzen aus einem

Server oder einem Router ziehen kann, ist zu prüfen, welche Leistungen der IT – also welche IT-Services – es sind, durch die seine Arbeit und der zugrunde liegende Geschäftsprozess unterstützt werden.

IT-Services werden vom IT-Service-Manager in Kenntnis der Anforderungen, die die Geschäftsprozesse des Unternehmens an die IT stellen, definiert, vereinbart und in einem Servicekatalog festgeschrieben. Service-Level (SL) spezifizieren IT-Services hinsichtlich Leistung, Verfügbarkeit, Servicezeiten etc. Beispiele für IT-Services sind z.B.:

- Es muss möglich sein, von einem PC-Arbeitsplatz Mails mit Attachments intern und extern zu versenden, zu empfangen und bis zu einem maximalen Volumen von 1 GB zu speichern.
- Es muss möglich sein, von einem PC-Arbeitsplatz nach Maßgabe der in den Sicherheitsrichtlinien festgeschriebenen Einschränkungen Zugriff auf das Internet zu erhalten.
- SAP R/3 wird mit den Modulen FI, CO und MM für zwei Buchungskreise bereitgestellt.

An dieser Stelle wird Folgendes deutlich: Der Grad der Detaillierung eines IT-Services kann sehr unterschiedlich sein. In den ersten beiden Beispielen werden kei-

ne Aussagen getroffen – und damit auch keine Zusagen gemacht – über das verwendete Mailsystem und die beteiligte Hardware. Es bleibt sogar offen, ob diese Leistungen von der IT selbst oder einem externen Dienstleister erbracht werden. Das dritte Beispiel ist, wenn man SAP R/3 an sich als Spezifikation ansieht, funktional in jeder Hinsicht sehr detailliert.

IT-Services werden unter Verwendung von Hardware, Betriebssoftware, Applikationen, Parametrisierungen, Benutzerprofilen u.a. von der IT erbracht. Diese Elemente bezeichnet ITIL als Configuration Items (CIs). Ein IT-Service wird in der Regel durch mehrere CIs realisiert. Ein CI kann Grundlage für mehrere IT-Services sein. CIs sind die kleinsten in der Configuration Management Database (CMDB) erfassten Elemente.

Die Mutter aller Probleme

ITIL schreibt nicht vor, wie detailliert ein IT-Service zu spezifizieren ist. Das ist einerseits gut für den Betrieb, der damit alle Freiheitsgrade hat, ein dem Unternehmen angemessenes Service-Management aufzubauen, andererseits für diejenigen – intern oder extern –, die Software erstellen, ein Risikofaktor. Wenn IT-Services das Maß aller Dinge sind und eine Software nichts anderes darstellt als ein CI, das dazu beiträgt, einen Service zu erbringen, dann ist die Erwartungshaltung des Betriebs an eine Software damit eigentlich vollständig und umfassend beschrieben: Leistet die Software hinsichtlich Funktionalität, Verfügbarkeit und Leistung mindestens das, was ich brauche, um mit ihr diesen einen in meinem Servicekatalog beschriebenen IT-Service zu erbringen? Das Problem an dieser Stelle ist nur, dass diese betriebliche Erwartungshaltung stets unausgesprochen bleibt und der Servicekatalog häufig für die Entwicklung – insbesondere die externe – nicht zugänglich ist.

Stattdessen erfolgt die Kommunikation zwischen Betrieb und Entwicklung im Rahmen der Beauftragung mittels eines Pflichtenheftes, das den Software-Lifecycle-Prozess in der Entwicklung anstößt. Das Problem ist offensichtlich. Die fachliche Qualität des Pflichtenheftes mit Blick auf das „Höchste aller Güter“ ist von fundamentaler Bedeutung.

Prozess	Ziel
Incident Management	Schnellstmögliche Beseitigung von Einschränkungen bei den IT-Services (Störungen)
Problem Management	Ermittlung der Ursachen, die den Serviceeinschränkungen zugrunde liegen, und deren Beseitigung; dadurch Reduzierung der Anzahl der Incidents und Stabilisierung der IT-Infrastruktur
Configuration Management	Bereitstellung detaillierter und aktueller Informationen über die IT-Infrastruktur für andere IT-Service-Prozesse
Release Management	Sicherstellen, dass nur korrekte, getestete und autorisierte Hard- und Software eingesetzt wird
Change Management	Minimierung der Auswirkungen auf den produktiven Betrieb bei der Durchführung von Änderungen (Changes)
Service Level Management	Erfüllung der Kundenanforderungen mit preiswerten, quantifizierbaren IT-Services in definierter Qualität
Availability Management	Optimierung der Verfügbarkeit der IT-Services mit dem Ziel, die Geschäftsprozesse bestmöglich zu unterstützen
IT Service Continuity Management	Planung der erforderlichen Maßnahmen zur kontrollierten Wiederherstellung der IT-Services nach einem Eventualfall
Capacity Management	Sicherstellung der kostengünstigen und rechtzeitigen Bereitstellung aller Kapazitäten für die IT-Infrastruktur
IT Financial Management	Identifizierung, Überwachung und Weiterverrechnung der Kosten für das IT-Service-Management

Tabelle 1: die zehn ITIL-Prozesse

Die Lösung ist ebenfalls klar: „Liebe Betriebsleute, schreibt präzisere Pflichtenhefte!“, aber auch: „Liebe Entwickler, ganz gleich, ob man euch zur Kritik eingeladen hat oder nicht, prüft eure Vorgaben! Lest das Pflichtenheft auch mit den Augen des Betriebs und denkt dabei an die IT-Services. Hinterfragt, wenn euch etwas unvollständig oder suspekt erscheint!“

Appelle dieser Art stellen natürlich keine grundsätzliche Lösung des Problems dar. Es kommt vorrangig darauf an, das bereits bekannte „Vier-Augen-Prinzip“ (siehe *Java Magazin* 10.2005) mit dem Ziel der Qualitätssicherung auch in diesen Bereich hineinzutragen und durch geeignete Review- und Freigabeelemente in einem bereichsübergreifenden Prozess zu verankern.

Die Problemkinder

Viele Leser werden bereits mit Erstaunen zur Kenntnis genommen haben, dass Release Management und Change Management hiernach keine Domänen der Entwicklung sind. Allerdings auch nicht des Betriebes. ITIL beschreibt Prozesse in IT-Organisationen, nicht in IT-Betriebseinheiten!

Auch wenn die Hürde genommen ist, ein Pflichtenheft mit der notwendigen Qualität zu erstellen, so bleiben doch im weiteren Verlauf eines Softwareprojekts und in der anschließenden Betriebsphase eine Reihe von Stolpersteinen. Um dies zu verdeutlichen, soll zunächst dargestellt werden, was eine Betriebsorganisation, die nach ITIL arbeitet, unter Change Management und Release Management versteht.

Change Management nach ITIL

Ein Change ist eine Änderung eines oder mehrerer Configuration Items (CIs). Im Sinne von ITIL kann ein Change sowohl die Zuschaltung einer Leitung in einem Netzwerk, das Einrichten eines Benutzers in einer Active Directory sowie auch der Wechsel eines Software-Release sein.

Die Aufgabe des Prozesses Change Management ist es, alle Änderungen zu erfassen, zu genehmigen, zu planen und deren Umsetzung zu kontrollieren. Der Change-Management-Prozess wird von einem Change Manager geleitet. Die Anlässe bzw. Auslöser für einen Change können vielfältiger Art sein, z.B.:

- Problemlösungen, Störungsbeseitigung
- Einführung neuer Funktionen auf Anforderung der Fachabteilungen bzw. Kunden
- Einrichten neuer Benutzer, Zuteilung von Lizenzen
- Erweiterung der Ressourcen zur Vermeidung von Engpässen
- Updates der Betriebssoftware

Bereits in einer IT-Umgebung mittlerer Größe und Komplexität wird die Anzahl der Changes, die zu jedem Zeitpunkt anstehen, eine Größe erreichen, die nur durch wirkungsvolle Abläufe und wohl definierte Zuständigkeiten beherrschbar ist. Es gilt der Grundsatz: kein Change ohne einen Request for Change (RfC). Der Ablauf bei einem Change ist unabhängig von seiner Art und Größe stets identisch:

1. RfC erfassen
2. RfC akzeptieren
3. Change klassifizieren
4. Change autorisieren
5. Change planen
6. Realisierung und Test überwachen
7. Change freigeben
8. Change Review und Abschluss

Das Change Management ist nicht verantwortlich für die Durchführung von Projekten wie Entwicklungsprojekten für Software. Die Verantwortung hierfür wird vom jeweiligen Projektmanager wahrgenommen, sie liegt außerhalb der ITIL-Prozesse. Die Genehmigung eines solchen Projektes und die Produktivsetzung unterliegen allerdings dem Change Management.

Aus Sicht des Betriebes ist diejenige Organisation, die eine Software für den Betrieb zur Verfügung stellt, zunächst nichts anderes als ein Lieferant, mit dem eine Kunden-Lieferanten-Beziehung besteht. Dies widerspricht nicht dem eingangs geforderten partnerschaftlichen Verhältnis zwischen Betrieb und Entwicklung und stellt auch keine Herabstufung der Entwicklung dar, sondern ermöglicht die strukturierte Definition von Schnittstellen bezogen auf diesen einen Prozess.

Im Falle eines externen Softwarelieferanten werden diese Beziehungen in Underpinning Contracts (UCs) geregelt, die

Anzeige

Beziehung zur Entwicklungsabteilung im eigenen Hause wird in Operational Level Agreements (OLAs) vereinbart. Diese sprachliche Unterscheidung trägt dem Umstand Rechnung, dass in beiden Fällen die Zusammenarbeit sehr unterschiedlich gestaltet werden kann und auch muss. Die Beziehung zu einer internen Entwicklungsabteilung eines Unternehmens könnte wie ein externer über UCs geregelt werden. Allerdings würde man sich damit der Möglichkeit einer engen und intensiven Zusammenarbeit zwischen Betrieb und Entwicklung berauben.

Das Change Management ist der Prozess, der in klassischer Weise eine Qualitätssicherung im operationalen Betrieb bewirkt. Jede Änderung in der produktiven IT-Infrastruktur hat ein definiertes Freigabe- und Genehmigungsverfahren zu durchlaufen.

Release Management nach ITIL

Die folgende Definition wird auf viele Leser befremdlich wirken: „Ein Release ist eine Reihe neuer oder geänderter Configuration Items (CIs), die zusammenhängend getestet und in die Produktionsumgebung überführt werden.“ Wenn wir uns daran erinnern, das ITIL unter einem Configuration Item nicht notwendigerweise ein Software CI versteht, dann ist der Betrieb mit dieser Definition meilenweit von dem traditionellen Verständnis der Entwicklung über Releases entfernt.

Da, wie oben gezeigt, nicht die Software oder die Hardware im Mittelpunkt des betrieblichen Denkens steht, sondern die zu erbringenden IT-Services und die dazu erforderlichen Configuration Items, macht es Sinn und ist zwingend notwendig,

1. auch Hardware und ggf. sogar Parametersätze im Release-Kontext zu definieren
2. Software, Hardware sowie Parameter nach den gleichen Verfahren zu behandeln
3. durch Bildung von Release Packages eine ganzheitliche Sicht zu erhalten

Die Aktivitäten im Rahmen des Release Management sind:

- Versionsgrundsätze und Planung zu bestimmen
- Release-Planung durchzuführen
- Entwurf und Entwicklung bzw. Bestellung und Ankauf zu veranlassen
- Erstellung und Zusammensetzung der Version
- Tests zu veranlassen
- Release-Freigabe durchzuführen
- Planung des Roll-out
- Vorbereitung und Schulung
- Verteilung und Installation durchzuführen

Ein Release-Manager steuert das Release-Management.

Da der gesamte Softwareentwicklungsprozess (alternativ der Ankauf) hier nach eine in den Release-Management-Prozess integrierte Phase ist, kommt es besonders darauf an, bei einem Entwicklungsprojekt mit eigenem Change und Release Management, die Abgrenzung und die Übergabepunkte mit diesem IT-Service-Prozess klar abzustimmen

Varianten

Release-Management wird häufig als Teil des Change Management oder des Configuration Management implementiert. Neben der Tatsache unterschiedlicher Prozesswelten, was allein bereits problematisch sein kann, kommt hier hinzu, dass gleiche Begriffe, ähnliche Zwecke und identische Ansprüche in verschiedenen Organisationen nebeneinander existieren, was zwangsläufig zu Problemen in beliebiger Vielfalt führen muss. Um nur einige der angekündigten „Problemkinder“ zu nennen, zwei Beispiele: Im ersten geht es um die Fehlerklassifizierung im Betrieb. In dem Artikel „Der Blick über meine Schulter“ [4] im Rahmen dieser Serie wurde eine Fehlerklassifizierung von Prioritätsklasse 4 (die Funktionalität der Anwendung ist nicht beeinträchtigt) bis Prioritätsklasse 1 (der Betrieb der Anwendung ist nachhaltig gefährdet) aufgestellt. Während der Entwicklungsphase einer Software ist eine solche Klassifizierung sinnvoll und notwendig.

Allerdings haben die Betriebsverantwortlichen ihre eigenen Vorstellungen über die Klassifizierung von Fehlern. Die Theorie ist einfach: Nach ITIL interessieren Softwarefehler niemanden, solange sie keine Auswirkungen auf die IT-Services und die unterstützten Geschäftsprozesse haben oder potenziell haben können. Die Bestimmung der Auswirkungen und der Dringlichkeit von Fehlern ist im Sinne einer serviceorientierten und wirtschaftlichen Betriebsführung nur mit genauer Kenntnis der Geschäftsprozesse möglich.

Das zweite Beispiel berücksichtigt den Wandel der Infrastruktur. Die Planung und Durchführung von Tests bei der Erstellung des ersten Release einer neuen Software sind zwar nicht unproblematisch, weisen aber in der Regel keine Prozessschwächen auf, da sie im Rahmen

ITIL – kurz zusammengefasst

Eine wesentlich Neuerung von ITIL – und sicherlich vielerorts auch ein Hemmnis für die Akzeptanz – ist die Abkehr von der alten Denkweise in Strukturen und Aufbauorganisationen. Das Gruppen- und Abteilungdenken, die Abgrenzung gegeneinander, die traditionellen Kämpfe um Vormachtstellung von Help Desk vs. Support oder Betrieb vs. Entwicklung haben in einer IT-Organisation, die sich für ITIL entschieden hat, keinen Raum. ITIL stellt den Vorteil für das Unternehmen in den Mittelpunkt und definiert dazu Prozesse, Rollen und Verantwortungen, die, wenn sie in bestehenden Organisationen etabliert werden, nur gruppen- und abteilungsübergreifend ihre Wirkung entfalten können. Für IT-Organisationen mit einer eigenen Entwicklung bietet ITIL die Chance, durch be-

reichsübergreifende IT-Service-Prozesse, die mit den etablierten Entwicklungsprozessen harmonisch abgestimmt sind, den Vorgang der Softwareentwicklung von der Anforderung bis zur Betriebsphase ganzheitlich unter Berücksichtigung aller für das Unternehmen relevanten Aspekte zu gestalten und zu leben. Aber auch Unternehmen, die im Auftrag ihrer Kunden Individualsoftware entwickeln, profitieren von der Kenntnis der IT-Service-Prozesse nach ITIL. Wenn sie zudem die Fähigkeit und die Bereitschaft besitzen, sich der jeweiligen Prozesslandschaft ihres Auftraggebers anzupassen, ist damit – neben der inneren Qualität der Software – eine weitere notwendige Voraussetzung für die Nachhaltigkeit der erbrachten Leistung erfüllt.

eines Projekts mit der erforderlichen Methodik abgewickelt werden. Kritischer ist dagegen der Umgang mit einem Release-Update während der kontinuierlichen Weiterentwicklung und Verbesserung. Die vom ITIL Change Management veranlassten und vom ITIL Release Management durchgeführten Tests berücksichtigen stets Release Packages, in denen der jeweilige infrastrukturelle Kontext in Form aller beteiligten CIs zusammengefasst sind. Diese Release Packages sind aufgrund häufiger Changes insbesondere der beteiligten Hardware und der interagierenden Software keineswegs statisch. Die initial erarbeiteten Testfälle und -methoden in der Entwicklung sind daher stets anzupassen. Ein möglicher Fehlerfall ist, dass z.B. nach dem Release-Wechsel einer Applikation mit einem Web-Interface Fehler auftreten, weil zwischenzeitlich als Reaktion auf Sicherheitslücken der Standard-Webbrowser auf allen Arbeitsplätzen ausgetauscht und das Betriebssystem restriktiver eingestellt wurden.

Die Integration

Die Brücke zwischen der Softwareentwicklung und dem IT-Service-Management nach ITIL wird in dem ITIL Volume „Application Management“ geschlagen, das allerdings weniger bekannt und verbreitet ist als die zehn ITIL-Kernprozesse. Während das traditionelle Software Lifecycle Management die Phasen Anforderung, Design, Build und Deploy umfasst (Abliefern und Rechnung stellen!), berücksichtigt das ITIL Application Management natürlich die Betriebsphase und definiert die notwendigen Interaktionen mit allen „klassischen“ ITIL-Prozessen (Abb. 1).

Die Berücksichtigung der betrieblichen Erfordernisse erfolgt durch formalisierte Schnittstellen der zehn ITIL-Kernprozesse mit jeder der Phasen des Application-Lifecycle-Prozesses. Der generische Ablauf eines Application Management Lifecycle nach ITIL beschreibt einen Zyklus der Phasen Requirements, Design, Build, Deploy, Operate und Optimize. Der in Abbildung 1 skizzierte Lifecycle ist eine mögliche Variante der Implementierung eines solchen Prozesses. Das Beispiel weicht geringfügig von der „reinen Lehre“ des Application Management

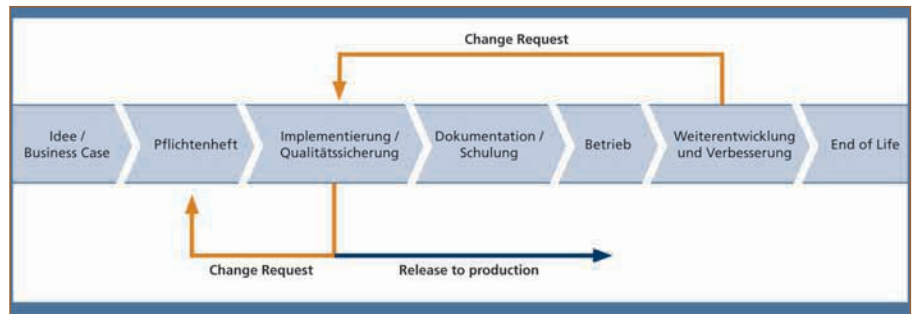


Abb. 1: Beispiel eines Software-Lifecycle-Prozesses

nach ITIL ab, veranschaulicht aber die Möglichkeiten der frühzeitigen Berücksichtigung betrieblicher Aspekte durch die Entwicklung:

- **Idee/Business Case:** Business-Ziele, Service-Level und Finanzaspekte werden im Vorfeld ganzheitlich im Rahmen des Service-Level-Managements und des Financial Management berücksichtigt.
- **Pflichtenheft:** Die nicht funktionalen Anforderungen, mit denen „der Betrieb fest im Blick“ behalten wird, fließen gesteuert durch das Change und Release Management in das Pflichtenheft ein.
- **Implementierung/Qualitätssicherung:** Die Entwicklung erfolgt zielgerichtet mit Blick auf die Service-Level-Anforderungen. Tests fokussieren die IT-Services unter Berücksichtigung des infrastrukturellen Kontextes. Priorisierungen werden den Erwartungen der Bedarfsträger gerecht. Etwaige Eskalationen werden zur Ausnahme.
- **Dokumentation/Schulung:** Die Betriebs-einführung erfolgt „weich“.
- **Betrieb:** Funktion und Leistung der Applikation werden gegen die geforderten und allen Beteiligten bekannten Service-Level gemessen.
- **Weiterentwicklung und Verbesserung:** Problembedingte und funktionale Änderungen werden über standardisierte und bewährte Prozesse an die Entwicklung zurückgeführt.
- **End of Life:** Durch Einträge in der Configuration Management Database wird sichergestellt, dass die Software nicht unbeabsichtigt zum Einsatz kommt.

Schlussfolgerung

Es besteht kein Widerspruch zwischen der klar geregelten, ziel- und ergebnisori-

entiert gesteuerten ITIL-Prozesswelt und den sich größtenteils selbst steuernden Abläufen des agilen Projektmanagements in der Softwareentwicklung. ITIL hat sich als De-facto-Standard für das Management von IT-Services durchgesetzt. Die positiven Effekte einer solchen Standardisierung hinsichtlich sprachlicher Normung und Einheitlichkeit der Prozesse wirken bislang nur in Betriebsorganisationen. Eine Softwareentwicklung nach ITIL gibt es nicht. Obwohl ITIL den Anspruch erhebt, ganzheitlich IT-Prozesse zu definieren, liefert es keine Vorgaben für den Prozess der Softwareentwicklung und für das Projektmanagement in Softwareprojekten. Gleichwohl besteht die Notwendigkeit, beide Welten miteinander zu verbinden, wofür ITIL vielfältige Schnittstellen zur Verfügung stellt.

Detlef Ahlers ist Senior Consultant bei syngenio und berät Kunden bei der Gestaltung und Implementierung von IT-Serviceprozessen nach ITIL.

Links & Literatur

- [1] Elmar Borgmeier: Intelligenz statt Konformität. Qualität in Java-Projekten, Teil 1: Mit agilem Qualitätsmanagement zum Projekterfolg, in *Java Magazin* 7.2005
- [2] Lars Bachert, Stefan Reisner: Das können wir nicht gebrauchen! Qualität in Java-Projekten, Teil 2: Der anwenderbezogene Qualitätsaspekt, in *Java Magazin* 8.2005
- [3] Jörn Eyrych, David Gärtner: Fuß von der Bremse. Qualität in Java-Projekten, Teil 3: Last und Performance, in *Java Magazin* 9.2005
- [4] Dirk Mögenburg: Der Blick über meine Schulter. Qualität in Java-Projekten, Teil 4: das Vier-Augen-Prinzip, in *Java Magazin* 10.2005
- [5] The IT Service Management Forum: www.itsmf.com
- [6] Jan van Bon: IT Service Management – an Introduction, Independent Publishers, 2002
- [7] The ITIL Application Management Book