Karol Frühauf, Gerald Linhofer

Konfigurationsmanagement – das Bindeglied der IT-Prozesse

Konfigurationsmanagement als Bindeglied der IT-Prozesse unterstützt im Kontext des IT-Betriebs sowohl das Bereitstellen der Dienste als auch das Betreiben der Lösungen. Der Beitrag beschreibt dies mit Hilfe der IT-Infrastruktur und macht die zentrale Rolle des Konfigurationsmanagements für die Prozesse, die sich mit dem Verändern der produktiven Konfiguration der Lösung(en) beschäftigen, deutlich. Das in diesem Beitrag vorgestellte Prozessverständnis wird mit dem BS15000/ITIL und dem MITO-Reifegradmodell verglichen. Außerdem wird auf den Unterschied zwischen Konfigurationsmanagement für Hardware und Software sowie zwischen Entwicklung und Betrieb eingegangen und die zentralen Konzepte des Konfigurationsmanagements werden kurz behandelt.

Inhaltsübersicht

1 Einleitung

Die Kunst des Fortschritts ist es,
inmitten der Veränderung Ordnung
zu bewahren und
inmitten der Ordnung Veränderungen
zu ermöglichen.

Alfred North Whitehead

Viele IT-Mitarbeiter haben kaum Assoziationen, wenn sie nach Konfigurationsmanagement in ihrer Umgebung gefragt werden. Erzählt man ihnen Geschichten aus dem Leben, die mit ein bisschen mehr Konfigurationsmanagement anders geendet hätten, dann hellt sich ihr Gesicht sofort auf: »Das ist Konfigurationsmanagement? Das machen wir ja, nur nicht immer konsequent genug.« Der Begriff ist nicht geläufig, aber die dahinter steckenden Aufgaben werden praktiziert.

Der einleitende Spruch des Mathematikers Whitehead (1861-1947) formuliert die Mission und das Dilemma des Konfigurationsmanagements treffend. Das Ändern der Systemkonfiguration darf nicht willkürlich sein, es muss nachvollziehbar bleiben, andererseits darf die Reglementierung das Ändern nicht schwerfällig bis unmöglich machen. Nachvollziehbar heißt, dass man herausfinden kann, welche Konfiguration man hat, sollte etwas nicht funktionieren. Und es soll auch leicht zu erfahren sein, was man gehabt hat, als es noch funktionierte. Weiß man auch noch, was nun anders ist als zuvor, dann ist man dem Übel sehr viel näher. Der Grund des Übels ist zwar noch immer nicht be-

kannt, aber man kann mit Fug und Recht annehmen, dass er sich in den geänderten Teilen befindet. Wenn die Konfiguration aus Hunderten von Teilen – Dateien und/oder Geräten – besteht, dann ist dies bereits eine große Hilfe. Im Wesentlichen leistet Konfigurationsmanagement genau dies.

Konfigurationsmanagement beschäftigt sich mit dem Verwalten von Teilen und ihrer Beziehungen untereinander. Ein System wird durch Änderungen von einer Konfiguration in eine andere überführt. Anstöße für diese Änderungen sind geänderte Anforderungen an bereits angebotene Dienste (Services), Anforderungen an neue Dienste, Probleme mit dem Erbringen der zugesagten Dienstleistungen, d.h. mit der bestehenden Lösung (Solution) hinsichtlich ihrer Funktionalität, Zuverlässigkeit oder Leistung. Ein weiterer Grund sind präventive Maßnahmen, die Absicht, das Funktionierende zu ändern, bevor es nicht mehr funktioniert und deshalb geändert werden muss. Das Konfigurationsmanagement hat eine zentrale Rolle beim Meistern dieser Aufgaben. Es schafft die notwendigen Voraussetzungen für den sinnvollen Umgang mit den Änderungen.

Zwei Bereiche sind von besonderer Bedeutung, wenn es um den IT-Betrieb geht: Das Verfolgen der Änderungen, die direkt am produktiven System selbst vorgenommen werden, sowie das Übernehmen der Änderungen in die Produktion, nachdem sie an einem Testsystem erprobt wurden. Im ersten Fall liegt die Herausforderung darin, alle durchgeführten Änderungen, seien sie noch so geringfügig, aufzuzeichnen, damit man sich im Falle von Problemen bis zur Konfiguration, an der das entscheidende Missgeschick passiert ist, zurückkämpfen kann.

Im zweiten Fall stellen sich Fragen folgender Art: Kann man die Softwarekonfiguration vom Testsystem unverändert übernehmen? Gibt es keine Parameter, die für das produktive System anders gesetzt werden müssen? Sind die Hardwaregeräte, ohne Änderung der Parametrisierung, im produktiven System einsetzbar? Gibt es nirgendwo Abhängigkeiten zwischen Adressen? Auf derartige Fragen sollte man mit Hilfe des Konfigurationsmanagements rasch und zuverlässig die Antworten finden können.

2 Konfigurationsmanagement im Kontext

2.1 Dienstleistung (Service) und Lösung (Solution)

Ein IT-Betrieb bietet Dienstleistungen an, wie z.B. das Betreiben eines ERP-Systems oder eines Portals. Hierzu wird eine IT-Infrastruktur, bestehend aus Hardware (Rechner, Peripheriegeräte, Netze) und Software (Betriebssoftware, Applikationen, Werkzeuge), benötigt. Aus diesen Mitteln werden die Lösungen zusammengestellt, die das Erbringen der einzelnen Dienstleistungen ermöglichen. Abbildung 1 zeigt die zwei Ebenen – die Bereitstellung der Dienstleistung (Service) und der dazugehörigen Lösung (Solution) – mit Konfigurationsmanagement (KM) als Drehscheibe für alle greifbaren Ergebnisse.

Aus den Anforderungen des Marktes, z.B. Abschluss eines neuen Service Level Agreements (SLA), und ihrer Änderungen, z.B. neue Geschäftsvorfälle, wird vom Service Design (1) eine Spezifikation der Dienstleistung erstellt. Aus dieser werden die Anforderungen an die IT-Infrastruktur abgeleitet, für die das Solution Engineering (2) eine Lösung spezifiziert und implementiert. Beim Konzipieren der Lösung werden Technologietrends berücksichtigt und bekannte Mängel analysiert. Die Ergebnisse werden vom KM verwaltet.

Die Lösung wird vom Solution Rollout (3) für die Nutzung bereitgestellt und, synchronisiert mit dem Service Deployment (4), in Betrieb genommen. Das Service Deployment sorgt dafür, dass die Verfügbarkeit der Dienstleistung bekannt wird, die zum Betreiben benötigten Prozesse, wie z.B. Hotline, installiert sind und die Benutzer motiviert werden, die Dienstleistung anzunehmen.

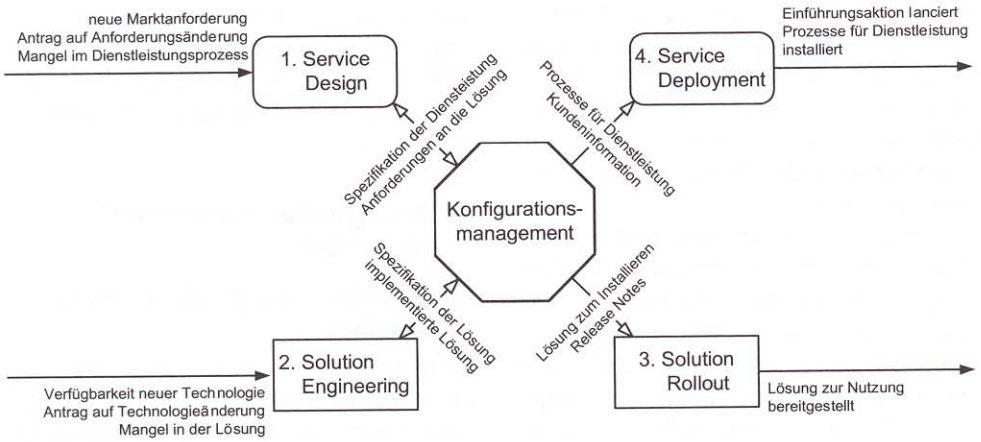

Abb. 1: Bereitstellung von Dienstleistungen und Lösungen

Beispiel: Portal
Das Produktmanagement eines Portalanbieters möchte am Markt einen zusätzlichen Service lancieren. Am bestehenden Portal kann mittels Mail eine Frage an den Support geschickt werden. Neu soll zusätzlich die Möglichkeit bestehen, mit Mail eine Chat-Session zu eröffnen, um mit dem Support einen Dialog führen zu können.

Diese Anforderung wird im Service Design (1) zunächst in einen Prozess umgesetzt, mit dem die Zuweisung der Aufgaben und die Arbeitsweise der Supportmitarbeiter neu gestaltet wird. Zudem werden die Anforderungen an die Unterstützung durch IT formuliert.

Das Solution Engineering (2) setzt diese Anforderungen in eine Lösung um, die innerhalb des Kostenrahmens den Anforderungen gerecht wird. Die implementierte Lösung – geänderte und neu erstellte Softwareeinheiten – steht im KM-Repository zur Verfügung.

Das Solution Rollout (3) führt die Applikationsserver nach, auf denen das Portal instanziiert ist. Damit der Markt Notiz nimmt von diesem einmaligen Angebot, setzt das Service Deployment (4) die Marketingmaßnahmen um, die vom Service Design für die Einführung am Markt definiert wurden, z.B. werden alle bestehenden Kunden mit einer Mail auf die Neuerung aufmerksam gemacht. Ebenfalls Aufgabe des Service Deployments ist es, den Support auf die neue Aufgabe vorzubereiten und den Chat-Prozess in der Supportorganisation zu installieren.

2.2 Konfiguration ändern

Als Ausgangsbasis für unsere Betrachtungen nehmen wir an, dass ein System bereits in Betrieb ist und seine Konfiguration verändert werden soll. Wir wollen untersuchen, welche Information und welche Prozesse nötig sind, um diese Veränderung geordnet, effizient und nachvollziehbar vorzunehmen. Die einzelnen Prozesse sind, bis auf Konfigurationsmanagement, englisch benannt.

In Abbildung 2 ist der Steuerfluss (weiße Pfeile) und Materialfluss (graue Pfeile) zwischen den Prozessen dargestellt, wobei in diesem Kontext neben Hardware auch Software als Material zu verstehen ist. Die Prozesse zum Bearbeiten dieser Materialien sind mit dem Platzhalter »Änderung bewirken« dargestellt, es ist jedoch angedeutet, dass die drei Prozesse Service Design, Solution Engineering und Build darunter zu verstehen sind. Solution Rollout und Service Deployment machen die Änderung für den Betrieb und die Benutzer verfügbar.

Das Issue Management ist die Anlaufstelle für Anwender. Dieser Prozess nimmt das Anlie-

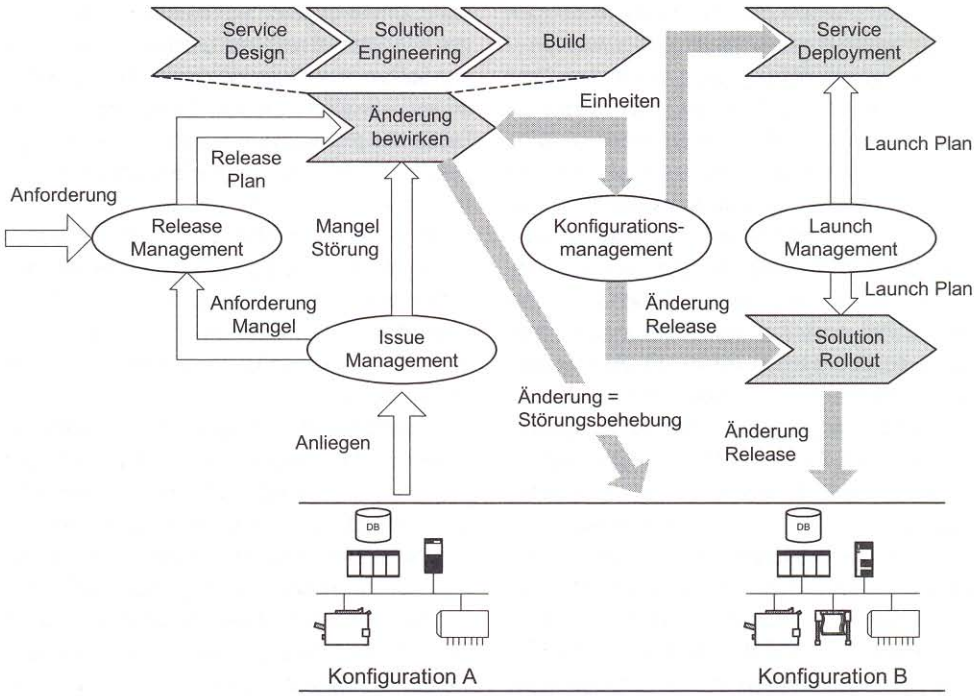

Abb. 2: Konfigurationsmanagement im Kontext

gen des Anwenders entgegen, entscheidet über die Natur des Anliegens, leitet die angemessene Bearbeitung ein und verfolgt diese bis zu ihrem Abschluss. Gegenstand der Bearbeitung dieses Prozesses ist also jeweils ein einzelnes Anliegen.

Das Release Management sorgt dafür, dass die Änderungen geplant und koordiniert bereitgestellt werden, es ist »das Projektmanagement für das Bereitstellen aller Änderungen, die für ein Release bestimmt sind«. Durch Mängel oder neue Anforderungen bedingte Änderungen der Systemkonfiguration werden einzelnen Releases zugewiesen und ihre Implementierung bis zur Inbetriebnahme des Release verfolgt. Gegenstand der Bearbeitung sind also alle Änderungen, die gemeinsam in Produktion genommen werden sollen (Release). Release Management kann auf den Ebenen Dienstleistung, Lösung und über Lösungen hinaus für die gesamte IT-Infrastruktur betrieben werden. Die Prozesse Service Design, Solution Engineering und Build erledigen die fachliche Arbeit am Release, d.h., sie spezifizieren, implementieren und testen die Änderungen.

Unter Build verstehen wir hier die Montage von Hardwareeinheiten bzw. das Erzeugen des lauffähigen Codes aus Quellcode (Applikation aus Eigenentwicklung). Gegenstand der Bearbeitung sind Konfigurationen, die aus Einheiten gebildet sind, die vom KM verwaltet werden.

Könnte man jede Änderung ohne Zeitverbrauch erledigen, wäre kein Release Management nötig. Selbst wenn die Änderungen Zeit beanspruchten, jedoch voneinander unabhängig wären, bräuchte man kein Release Management. Seine koordinierende Funktion wird dort benötigt, wo ausgehend von derselben Konfiguration mehrere Änderungen parallel vorgenommen werden und diese technisch oder zeitlich voneinander abhängig sind, d.h., es ist nicht egal,

welche Änderungen zusammen und in welcher Reihenfolge in Produktion genommen werden.

Das Launch Management plant den Zeitraum, in dem alle Änderungen, die das Release ausmachen, produktiv gesetzt werden. Dies schließt den Hardware-Umbau genauso ein, wie die Software-Updates und die eventuell nötige Migration von produktiven Daten. Der Plan sieht den günstigen Fall vor, in dem alles wunschgemäß funktioniert, sowie Fallback-Szenarien für den Fall, dass das neue Release nicht produktiv gesetzt werden kann. Launch Management setzt den Plan auch um, steuert die Ausführung der fachlichen Arbeiten in den Prozessen Solution Rollout und Service Deployment. Es ist »das Projektmanagement für die Durchführung der Umkonfigurierung des Systems«.

Zu den Teilprozessen von Solution Rollout gehören das Verteilen der Softwarelieferung, Installation der (Hardware- und/oder Software-)Änderung und die Inbetriebnahme der geänderten Konfiguration. Die Softwarelieferung verteilen bedeutet, diese den Stellen zukommen zu lassen, an denen sie benötigt werden. Die Softwarelieferung ist die Untermenge des Release einer Applikation, die zum Zielsystem transportiert wird. Die Hardwareänderung installieren heißt, die Hardwareeinheiten am Zielsystem anschließen und ihre Funktionstüchtigkeit prüfen. Bei Softwareeinheiten wird ihre neue Version dem Betriebssystem bekannt gegeben und die Applikation parametrisiert. Die durch die Änderung entstandene neue Konfiguration wird in Betrieb genommen, indem alle Vorkehrungen getroffen werden, damit die geänderte Konfiguration genutzt werden kann, z.B. die Daten werden migriert oder das Job Scheduling wird abgestimmt.

Ist nur Software in Produktion zu nehmen und sind keine Datenbestände von der Änderung betroffen, dann kann die Aufgabe des Launch Managements auf einen Knopfdruck zusammenschrumpfen. Dies gilt, wenn die Software automatisch verteilt und installiert wird. Bei einem Release mit Umbauten an Hardware, Umadressieren von Systemen, Änderungen von Datenstrukturen und neuen Versionen von Software wird die Inbetriebnahme zu einer anspruchsvollen Aufgabe. Da in den meisten Fällen der produktive Betrieb nicht allzu lange gestört werden darf, muss die Inbetriebnahme minutiös geplant und diszipliniert durchgeführt werden. Fallback-Szenarien, Vorwärtsstrategien für mögliche Missgeschicke nach dem Point-of-no-Return müssen im Vorfeld überlegt sein, ansonsten würde man im Ernstfall während der Inbetriebnahme wertvolle Zeit verschenken.

In Abbildung 2 ist das Konfigurationsmanagement im Materialfluss als Zwischenlager für die Einheiten dargestellt. Dies trifft für Softwareeinheiten und Dokumente zu; alle ihre Änderungsstände werden im KM-Repository abgelegt. Wie über diese wird jedoch auch über Hardwareeinheiten in der KM-Datenbank Buch geführt. KM verwaltet Informationen über alle Einheiten und über die Beziehungen zwischen ihnen bzw. zwischen ihren Versionen. Gegenstand der Bearbeitung sind also Einheiten und Konfigurationen aus diesen Einheiten.

Beispiel: Portal

Das Service Design erstellt das Dokument »Spezifikation der Dienstleistung Chat mit Support« und modifiziert das bereits existierende Dokument »Anforderungsspezifikation Portal«, in dem die Anforderungen an Chat mit Support aufgenommen werden. Beide Dokumente sind im KM-Repository abgelegt und ihre Versionen und deren Status sind von KM überwacht. Dies ist auch der Fall für die »Spezifikation der Lösung für Portal«, die durch den Lösungsansatz für Chat mit Support ergänzt wird. Alle anderen betroffenen Dokumente, wie Designbeschreibungen, Testspezifikationen usw. werden in einer neuen Version die Auswirkungen dieser neuen Anforderung enthalten.

Neu erstellt wird ein Programm, das die Benutzerschnittstelle für Chat mit Support sowie die Abwicklung des Chats selbst implementiert.

Zudem wird die bisherige Schnittstelle für Mail an Support geändert, um dem Benutzer zu ermöglichen, die Chat-Session zu initialisieren. Auch die Versionen all dieser Softwareeinheiten werden im KM-Repository abgelegt und ihr Status vom KM überwacht.

2. Vom Anliegen zu seiner Erledigung

Mit Anliegen bezeichnen wir die Nachricht, mit der sich ein Benutzer an den Help Desk wendet: Er kann seine Arbeit nicht auf die Art erledigen, wie er es möchte, und sucht Hilfe. Wir wählten den Begriff Anliegen, weil im Umfeld von IT-Servicemanagement der Begriff Problem von ITIL anders besetzt ist als sonst im Konfigurationsmanagement üblich (vgl. Kasten 1).

Die Nachricht des Benutzers löst den Prozess Anliegen-Management aus. Im ersten Schritt wird das Anliegen analysiert und entschieden, welcher Art es ist; daraus ergibt sich seine weitere Behandlung. Gemäß Abbildung 3 kann das Anliegen vom Typ Störung, Mangel, Anforderung, Irrtum, Frage oder Auftrag sein. Bei Störungen und Mängeln muss zuerst der sie verursachende Fehler lokalisiert werden, bevor eine Abhilfe möglich ist. Die Beseitigung des Fehlers führt zur Änderung der Konfiguration. Eine modifizierte oder neue Anforderung bedingt Engineering-Arbeit und führt ebenfalls zur Änderung der Konfiguration. Eine gezielte Frage und ein Irrtum bei der Bedienung des Systems werden durch Beratung erledigt. Einem Auftrag wird mit dem Erbringen der gewünschten Dienstleistung entsprochen.

Der Unterschied zwischen Störung und Mangel ist wesentlich. Störung liegt vor, wenn wichtige Dienste nicht verfügbar sind. Am Beispiel unseres Portals wäre eine Störung, wenn der Benutzer die Ware auswählen, aber die Bestellung nicht abschicken kann. Bei einer Störung ist das höchste Gebot, die Dienste wieder so schnell wie möglich verfügbar zu machen. An unserem Portal müsste man schnell dafür sorgen, dass die Bestellungsabwicklung wieder funktionstüchtig ist. Eine Störung wird entweder direkt behoben oder eine Änderung wird bewirkt und außerhalb des geplanten Releasezyklus in Produktion genommen. Die hierzu durchgeführte Änderung kann eine provisorische Lösung zur Symptombekämpfung sein (z.B. Restart des Servers mit der Bestellungsabwicklung) und, wie in Abbildung 2 angedeutet, darf an KM vorbei realisiert werden.

Hält sich der Schaden in Grenzen, dann wird das Anliegen als Mangel taxiert und versucht, seine Ursache zu beseitigen. In unserem Beispiel könnte ein Mangel z.B. ein deutscher Text in einem ansonsten französischen Bildschirmformular sein, wenn als Portalsprache Französisch gewählt wurde. Der Mangel wird behoben, indem

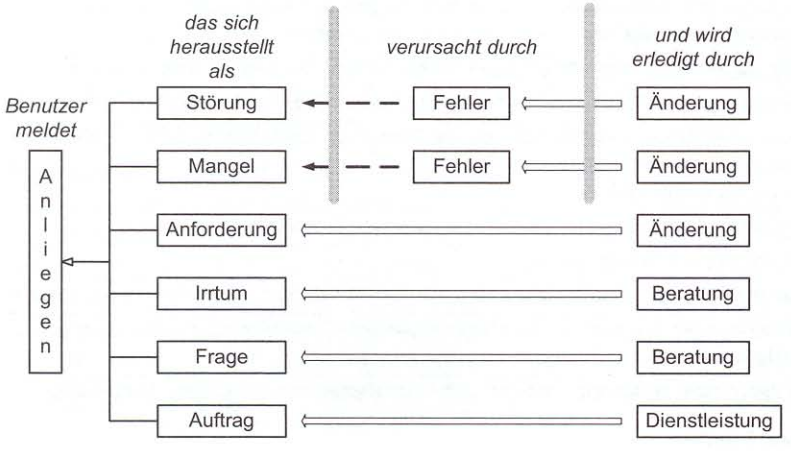

Abb. 3: Vom Anliegen zu seiner Erledigung

man den Fehler (oder mehrere miteinander) lokalisiert und seine Beseitigung mit einer Änderung für ein künftiges Release plant und durchführt.

Erkennt man im Anliegen des Benutzers eine nicht implementierte Anforderung, dann wird der Benutzer zunächst vertröstet und sein Anliegen an das Release Management weitergereicht. Im Rahmen der Releaseplanung wird dann entschieden, ob überhaupt und wenn ja in welchem Release die Anforderung implementiert wird. Eine kundenorientierte Organisation wird die Spur zum Urheber der Anforderung sichern und ihn informieren, wann die Anforderung voraussichtlich implementiert sein wird (auch wenn die Antwort niemals ist). Ein Benutzer könnte bei unserem Portal den Wunsch äußern, mehr als eine Verrechnungsmöglichkeit gespeichert zu haben und jeweils beim Bestellen eine auswählen zu können. Falls diese Möglichkeit noch nicht spezifiziert ist, dann teilt der Help Desk ihm dies mit, bedankt sich für die Anregung und stellt weitere Information in Aussicht. Sollte die Implementierung der Anforderung für eine der folgenden Releases eingeplant werden, bekommt der Benutzer die versprochene Information mit dem voraussichtlichen Zeitpunkt, ab dem die gewünschte Wahl der Verrechnung am Portal möglich sein wird.

Im Unterschied zur Anforderung muss beim Auftrag keine Änderung an der Konfiguration des Systems vorgenommen werden. Es geht um das Anfordern einer Dienstleistung, die nur vom Betreiber erbracht werden kann, z.B. eine Datei vom Backup zurückholen oder einen neuen Benutzer einrichten. In einem Informationscenter kann ein Auftrag auch eine Änderung zur Folge haben, beispielsweise wenn ein PC-Benutzer ein zusätzliches Softwarepaket installiert haben möchte. Dieser Fall ist in Abbildung 3 nicht dargestellt.

Bis jetzt haben wir die Behandlung eines einzelnen Anliegens diskutiert. Insbesondere gilt dies für Störungen und Mängel, die jeweils als Einzelfall angesehen wurden. Für die Prävention muss man die Daten über alle Anliegen analysieren, um systematische Fehler erkennen zu können. Wenn man z.B. feststellt, dass ein gewisser Gerätetyp nach einer bestimmten Zeit fehleranfällig wird, kann man nach einer bestimmten Frist die Geräte austauschen, bevor sie ausfallen. Erkennt man, dass die Tätigkeitsart »Netzwerk konfigurieren« am häufigsten zu Fehlern führt, kann man gezielte Schulungsmaßnahmen ergreifen. Ähnliche Erkenntnisse sind nur mit Hilfe einer Fehlerdatenbank möglich, in der alle Fehler erfasst und nach vordefinierten Kriterien charakterisiert sind. Das Beheben eines systematischen Fehlers ist eine vorbeugende Maßnahme mit dem Ziel, die Benutzer vor Unbill zu bewahren. Betrifft die prä-

Anliegen	Ein Bedürfnis nach Hilfestellung durch eine Serviceorganisation (engl. issue). Im Konfigurationsmanagement sonst häufig als Problem bezeichnet.
Störung	Starke Beeinträchtigung oder gar Unmöglichkeit der Nutzung des Systems (engl. incident).
Mangel	Das System erbringt nicht die vereinbarte Leistung, kann aber genutzt werden (engl. defect).
Fehler	Die Ursache der Störung oder des Mangels; der Sachverhalt, der das unerwünschte Verhalten des Systems verursacht (engl. fault). In ITIL als Problem bezeichnet.
Irrtum	Eine Handlung durch Menschen, die zum unerwünschten Ergebnis führt (engl. error).
Frage	Geäußerter Wunsch nach Beratung.
Anforderung	Formulierung eines gewünschten Verhaltens oder Eigenschaft des Systems (engl. requirement).
Auftrag	Die Anmeldung des Anspruchs, eine vertraglich zugesagte Dienstleistung zu beziehen (engl. service request).
Änderung	Überführen einer Konfiguration von einem definierten Zustand in einen anderen (engl. change).

Kasten 1: Definition von Begriffen

ventive Maßnahme die aktuelle Konfiguration im Betrieb, so wird sie im Rahmen der geplanten Releasezyklen umgesetzt.

3 Das Wesen des Konfigurationsmanagements

3.1 Zielsetzung und Definition

Mit Konfigurationsmanagement werden folgende Zielsetzungen verfolgt:

- Identifizierbarkeit der Objekte
- Reproduzierbarkeit der Objekte
- Nachvollziehbarkeit von Änderungen an den Objekten

Daraus ergibt sich direkt die Definition von Konfigurationsmanagement: Verfahren technischer und organisatorischer Natur, mit denen

- der Änderungsstand von (Hardware-, Software-, Dokumentations-)Einheiten eindeutig gekennzeichnet und diese Einheiten (Software, Dokumente) so aufbewahrt werden, dass sie nicht verändert werden können,
- die Identifikation von Konfigurationen zu bestimmten Zeitpunkten festgestellt wird und die Konfiguration jederzeit wieder herstellbar ist,
- die Änderung der Konfiguration und ihrer Bestandteile ein kontrollierter und, wegen erzeugten Aufzeichnungen, nachvollziehbarer Vorgang ist, mit dem die Konsistenz der Konfiguration und Rückverfolgbarkeit der Änderungen gewahrt wird.

3.2 Objekte unter Kontrolle von Konfigurationsmanagement

Was in der Zielsetzung abstrakt als Objekt bezeichnet wurde, ist in der Definition zu Einheiten und Konfigurationen konkretisiert. Die Einheiten können von ihrem Wesen her Hardwaregeräte, Programme und Daten sowie Dokumente sein. Je nach Art wird unterschiedliche Information zu ihrer Verwaltung benötigt.

Die Hardwareeinheiten müssen minimal mit folgenden Angaben beschrieben sein:

- eindeutige Kennung, z.B. Inventarnummer
- Gerätetyp, z.B. Bildschirm XY
- Seriennummer
- Status

Zur Verwaltung von Software- und Dokumentationseinheiten benötigt man zumindest folgende Angaben:

- eindeutige Kennung der Einheit (z.B. Name der Programmdatei oder Nummer des Dokuments)
- eindeutige Kennung der Version (z.B. 1.4)
- Name (z.B. Name des Programms oder Titel des Dokuments)
- Status

Eine Konfiguration ist eine Gesamtheit zusammenpassender Teile. Unter Hardwarekonfiguration versteht man eine Gesamtheit von Hardwareeinheiten und Dokumenten, unter Softwarekonfiguration analog eine von Softwareeinheiten und Dokumenten. Eine Systemkonfiguration besteht aus Hardware- und Softwareeinheiten samt zugehörigen Dokumenten.

Beispiel: KM-Datenbank

In der KM-Datenbank eines Informationscenters werden PCs, Peripheriegeräte, Programme und Dokumente verwaltet. Tabelle 1 zeigt ein Beispiel für das Hardwareinventar. In diesem Fall ist direkt angegeben, in welcher Hardwarekonfiguration die Einheit eingebaut ist.

In der Zusammenstellung der Softwareeinheiten (vgl. Tab. 2) ist dies nicht der Fall, weil die gleiche Version der Software an mehreren Orten installiert sein kann.

Um sich das Leben leichter zu machen, wird man eine Standard-Softwarekonfiguration definieren, so dass für den jeweiligen Arbeitsplatz nur die Besonderheiten zusätzlich geführt werden müssen (vgl. Tab. 3). Das Gleiche kann man auch mit der Hardware tun. Eine Standard-Hardwarekonfiguration würde man auf der Ebene der Gerätetypen definieren, während bei Software dies auf der Ebene von Versionen geschieht.

Kennung	Gerätetyp	Seriennummer	Status	Konfiguration
IN22858	...			
IN26859	PC Dell 1820	653.366963	eingebaut	HWAP 223
IN26860	Drucker HP Laser 2400	1029.23.6547	in Reparatur	HP
IN26860	Drucker HP Laser 2400	1029.23.6675	eingebaut	HWAP 223
IN26861	Logische Maus 5	342.237.54334	eingebaut	HWAP 223
IN26862	Sherry Tastatur 32	31234.23461	eingebaut	HWAP 223
IN26863	...			

Tab. 1: KM-Datenbank mit Hardwareeinheiten

Kennung	Version	Name	Status
...			
D-1023	2.1	Arbeitsplatz Installationsanleitung	freigegeben
bs.exe	3.9	Betriebssystem	freigegeben
remington.exe	1.3	Textverarbeitungsanwendung	freigegeben
datel.exe	2.1	Buchhaltungsanwendung	freigegeben
photoladen.exe	4.7	Grafikeditor	freigegeben
photoladen.exe	5.4	Grafikeditor	in Prüfung
machtlos.exe	3.2	Präsentationsanwendung	freigegeben
D-1031	1.7	Benutzerhandbuch Drucker HP Laser 2400	freigegeben
...			

Tab. 2: KM-Datenbank mit Softwareeinheiten

Kennung	Version
D-1023	2.1
bs.exe	3.9
remington.exe	1.3
datel.exe	2.1
photoladen.exe	4.7

Tab. 3: Definition der Softwarekonfiguration »Standard-Arbeitsplatz-Software«

Die Systemkonfiguration, die vollständige Konfiguration für einen konkreten Arbeitsplatz (AP 223), ist in unserem Beispiel (vgl. Tab. 4) definiert durch die Standard-Softwarekonfiguration für alle Arbeitsplätze (SW AP), die konkrete Hardwarekonfiguration des Arbeitsplatzes (HWAP 223) und die zusätzliche Softwareeinheit auf diesem Arbeitsplatz (die Präsentationsanwendung).

Kennung	Version	Name
HWAP 223		Konfiguration Arbeitsplatz-Hardware
SW AP		Standard-Arbeitsplatz-Software
machtlos.exe	3.2	Präsentationsanwendung

Tab. 4: Systemkonfiguration vom Arbeitsplatz AP 223

3.3 Vom Konfigurationsmanagement verwaltete Beziehungen

Die Definition einer Konfiguration ist eine »besteht aus«-Beziehung. Um dem Anspruch der Reproduzierbarkeit und der Rückverfolgbarkeit

gerecht zu werden, ist das Verwalten der Information über weitere Beziehungen nützlich:

- *»ermöglicht«-Beziehung* zwischen der Dienstleistung und den Bestandteilen der Lösung, die zu ihrer Unterstützung benötigt werden

  Mit Hilfe dieser Information kann man alle Einheiten der Lösung identifizieren, die von einer Änderung der Dienstleistung betroffen sein könnten, oder erkennen, ob bei der Aufgabe der Dienstleitung die Bestandteile der Lösung entfernt werden können, da sie auch von keiner anderen Dienstleistung beansprucht werden.
- *»implementiert«-Beziehung* zwischen einer Anforderung an die Lösung und deren Bestandteile, die zur Implementierung der Anforderung beitragen

  Die Auswirkungen einer beantragten Änderung können leichter ermittelt werden, wenn diese Information verfügbar ist.
- *Änderungskonfiguration* mit allen Einheiten, die eine Änderung implementieren, welche durch eine Mängelmeldung oder einen Änderungsantrag ausgelöst wurde

  Wenn ein Release durch die erledigten Änderungsanträge und Mängelmeldungen definiert ist, dann kann man anhand dieser Information alle Einheiten finden, die für das Release produktiv gesetzt werden müssen.
- *»entstanden aus«-Beziehung* zwischen Dokumenten, anhand der man für jedes Dokument erkennt, welches Dokument als Vorgabe gedient hat

  Dies ist bei der Änderung von Dokumenten nützlich zu wissen, da man die Dokumente leicht identifizieren kann, die von der durchgeführten Änderung potenziell betroffen sind.
- *Kompatibilitätsbeziehung* zwischen Softwareversionen und Hardwareeinheiten und zwischen Versionen von Softwareeinheiten

  Damit kann man unsinnige Konfigurationen vermeiden.

3.4 Aufgaben des Konfigurationsmanagements

Aus den obigen Ausführungen kann man die folgenden essenziellen Aufgaben für Konfigurationsmanagement ableiten.

Kennzeichnung

Alle Einheiten, welcher Art auch immer, müssen eindeutig gekennzeichnet werden, damit sie referenziert werden können. Die Konvention für die Kennzeichnung muss sicherstellen, dass die vergebene Kennung eindeutig ist. In Tabelle 1 ist ein Beispiel mit der Vergabe einer Nummer für Hardwareeinheiten als Kennung zu sehen, Tabelle 2 zeigt ein Beispiel mit der Vergabe eines aussagekräftigen Namens für Softwareeinheiten und einer Nummer für Dokumente.

Inventarisierung

Übersicht über alle Einheiten jeglicher Art. In Tabelle 1 und Tabelle 2 sehen wir auszugsweise Angaben zur Inventarisierung.

Ablage

Die Softwareeinheiten und Dokumente werden samt ihrer Versionsgeschichte im KM-Repository abgelegt. Das KM-Repository hat eine Baumstruktur; sie orientiert sich an der Struktur der Lösungen und dementsprechend heißen dann die Ebenen des Baumes (von unten nach oben) beispielsweise Komponente, Teilsystem, System.

Beispiel: Portal

Das Portal, das um Chat-Service erweitert werden soll, besteht z.B. aus den Teilsystemen Präsentation, Benutzerverwaltung und Support, Benutzerdienste, Angebote, Bestellen, Datenbank, Systeminfrastruktur. Die Teilsysteme Datenbank und Systeminfrastruktur bestehen jeweils aus einer Hardware- und einer Softwarekomponente, die anderen nur aus Software. Der neue Chat-Service wird in einer neuen Komponente im Teilsystem Benutzerverwaltung und Support implementiert und benötigt die Änderung einer bestehenden Komponente im Teilsystem Präsentation.

Statusüberwachung

Jede Art einer Einheit hat ihr eigenes Statusmodell. Eine Spezifikation (z.B. in Arbeit, in Prüfung, freigegeben) hat andere Zustände als eine Mängelmeldung (z.B. offen, eingeplant, behandelt, geprüft, ausgeliefert, geschlossen), obwohl beide als Dokumente gelten. Für verschiedene Zwecke kann dieselbe Einheit mehrere Statusangaben besitzen. Beispielweise kann ein Gerät einen Status bezüglich seines Einsatzes haben (z.B. im Lager, eingebaut, in Reparatur, verschrottet) und einen bezüglich einer vorgesehenen Änderung seiner Konfiguration für ein Release (z.B. keine Änderung geplant, Änderung geplant, Änderung geprüft).

Aufzeichnungen

Über jede Änderung an jeder Einheit sollte man wissen, wer was wann warum und wie gemacht hat. Werden Softwareeinheiten und Dokumente in einem Werkzeug für Versionsverwaltung abgelegt, dann wird von diesen aufgezeichnet, wer was wann geändert hat. Zudem wird das Werkzeug bei jeder Transaktion nach einem Kommentar fragen. Mit einer Konvention kann man eine Auswertung der Kommentare ermöglichen.

Beispiel: Versionsverwaltung

Die Werkzeuge für die Versionsverwaltung vergeben eine fortlaufende Nummer (1, 2, 3, ...) für Versionen. Die Kennung m.n ist eine verbreitete Form für Versionen von Dokumenten. Der Kommentar beim Ablegen der neuen Version des Dokuments kann dazu genutzt werden, die Zuordnung der internen Version z.B. 1.3 zu der Versionsnummer, z.B. 8, mit der es im Werkzeug abgelegt wird, sichtbar zu machen.

Eine andere nützliche Konvention besteht darin, beim Speichern der neuen Version einer Quelldatei im Kommentar die Kennung der Mängelmeldung, z.B. M323 anzugeben, die der Auslöser für die gerade durchgeführte Änderung war. Auf diese Weise kann man auch bei einfachen Werkzeugen zur Versionsverwaltung eine Änderungskonfiguration definieren; das wären in unserem Beispiel alle Einheiten, in deren Kommentar M323 vorkommt.

Rückverfolgbarkeit

Die bekannte Beziehung zwischen den Anforderungen und den Einheiten, die sie implementieren, hilft schneller und umfassender die Auswirkungen einer Änderung der Anforderung zu beurteilen. Wird die Information über die von einer Änderung betroffenen Einheiten, die z.B. auf einen bestimmten Änderungsauftrag zurückzuführen ist, aufbewahrt, dann kann man im Falle des »Versagens« der Änderung schneller die fehlerhafte Einheit finden.

Beziehungen

Die Bedeutung der Verwaltung von Beziehungen haben wir bereits gesehen. In der Systementwicklung ist die wichtigste Beziehung die Reihenfolge für Kompilation und Linken; sie wird in einer Form verwaltet, die eine automatische Ausführung erlaubt. Weitere Beispiele sind die Reihenfolge für Installation und Inbetriebnahme, Abhängigkeiten bzgl. Daten in der Datenbank und positive oder negative Beeinflussung zwischen Systemparametern. Auf diesem Gebiet sind Eigenmittel gefragt, da die meisten am Markt erhältlichen Konfigurationsmanagement-Werkzeuge geringe Unterstützung bieten.

3.5 Konfigurationsmanagement von Hardware versus Software

Jede Kopie einer Softwareeinheit weist identische Eigenschaften auf. Benötigt wird die Information, an welchen Orten die gleiche Version der Softwareeinheit eingebaut ist; wenn sich ein Fehler an einem Ort bemerkbar macht, dann ist er in allen Kopien enthalten (auch wenn er sich nicht in allen sofort manifestiert).

Die »Kopien« des gleichen Typs einer Hardwareeinheit können sich unterschiedlich verhalten (Fabrikationsfehler, unterschiedliches Abnützungsverhalten). Deshalb werden Hardwareeinheiten mit Typ und Seriennummer identifiziert und es wird Buch darüber geführt,

wo welcher Typ mit welcher Seriennummer eingebaut ist (siehe Tab. 1). Die identische Hardwareeinheit kann zu jeder Zeit nur an einem einzigen Ort eingebaut sein.

Der andere Unterschied besteht darin, dass eine defekte Version der Softwareeinheit durchaus weiter eingesetzt werden kann, wenn man nur weiß, wie die Fehlerumstände zu vermeiden sind. Mit einer defekten Hardwareeinheit hat man in der Regel viel weniger Freude und Geduld. Sie wird ersetzt und als defekt markiert, damit sie nicht versehentlich anderswo eingebaut wird. Die Version der Softwareeinheit wird in der Regel nicht als defekt markiert (aber mit der Zeit auch ersetzt).

3.6 Konfigurationsmanagement im Betrieb versus Entwicklung

Das Versionieren – Kennzeichnen des Änderungsstandes einer Einheit und die Aufbewahrung aller ihrer Änderungsstände, damit auf sie zugegriffen werden kann – ist in der Systementwicklung, insbesondere für Programme und Dokumente, unabdingbar. Die geprüften Versionen müssen entweder für Verbesserung wieder verfügbar sein oder die geprüften und für gut befundenen Versionen weiterer Verwendung zugeführt werden (und nicht eine andere Version, über deren Eigenschaften wir keine Aussage machen können).

Im Betrieb wird das Versionieren in einem weit geringeren Maße angewendet. Die Hardwareeinheiten sind im obigen Sinne nicht versioniert; es gibt verschiedene Einheiten vom selben Typ mit jeweils eigener Kennung. Sie haben keine Änderungsstände, sie sind entweder eingebaut ins produktive System oder warten in einem »Ersatzteillager« auf ihren Einsatz. Ähnlich könnte man auch die verschiedenen Versionen der angelieferten Softwareeinheiten (Applikationen, Systemsoftware, Softwareprodukte, Softwarewerkzeuge) auffassen: Die Version ist entweder ins produktive System eingebaut, wartet auf den Einbau oder sie wurde aus der Produktion zurückgezogen und archiviert.

In der KM-Datenbank wird Information über die Einheiten geführt. In der Systementwicklung werden die Softwareeinheiten selbst mit der ganzen Geschichte ihrer Änderungsstände im KM-Repository abgelegt.

Im Betrieb werden die Systemsoftware- und Applikationsparameter sowie Skripte versioniert. Der Grund hierfür ist, dass sie geändert werden (müssen), sie werden quasi am produktiven System »weiterentwickelt«. Um ihre Änderung zurückverfolgen und gegebenenfalls einen früheren Änderungsstand wieder herstellen zu können, müssen sie versioniert werden. Dies ist nicht immer einfach. Zum Teil sind Parameter nur online änderbar und die Änderung »verschwindet« in der Systemsoftware oder Applikation. Im günstigeren Fall werden alle durchgeführten Änderungen automatisch aufgezeichnet, im schlimmsten muss dies manuell und mit eiserner Disziplin bewerkstelligt werden.

4 Vergleich von Modellen

Wie verhält sich das hier Erläuterte zu den anderen Modellen wie BS15000, ITIL oder MITO? (Hinweis: MITO ist ein Modell für das Bewerten eines IT-Betriebs, ein Assessment-Verfahren, das von einem Arbeitsteam der Fachgruppe Informatik der Swiss Association for Quality, SAQ, erarbeitet wurde). Im Kasten 2 sind die mit Konfigurationsmanagement verwandten Prozesse aus den einzelnen Werken mit ihrer Bezeichnung einander zugeordnet. Wir wollen auf die Unterschiede näher eingehen.

Konfigurationsmanagement ist ziemlich einheitlich, nur bei MITO wird der Prozess in einem Atemzug mit dem Management der Lösungsänderungen genannt. Auch was den Inhalt des Prozesses betrifft, sind die Modelle vergleichbar.

In BS15000 und ITIL umfasst Incident Management das Beheben der Störung und Mängel, wobei die schnelle erneute Verfügbarkeit im Vordergrund steht. Die Ursachenforschung, sowohl für den Einzel- als auch für systemati-

sche Fehler, obliegt dem Problem Management, das dann die Änderung der Konfiguration beim Change Management beantragt. Das Release Management (in früheren ITIL-Publikationen hieß es Software Control & Distribution) sorgt dafür, dass die Änderung produktiv gesetzt wird. Im BS15000 findet man zusätzlich für die Übernahme der Änderungen in die Produktion auf der Serviceebene den Prozess Implementation of New or Changed Service. Weil dies der einzige Unterschied ist, wurde in Kasten 2 nur ITIL aufgenommen.

MITO kennt Issue Management als den Prozess für die Triage und das Verfolgen der Bearbeitung des Anliegens. Für die Bearbeitung selbst werden die zwei Ebenen Service und Solution unterschieden. Sowohl die Managementaufgaben – Service Change Management bzw. Solution Change and Configuration Management – als auch die Ausführung – Service Design and Update bzw. Solution Implementation and Maintenance – werden von separaten Prozessen bearbeitet. Bei der Ausführung kommt zusätzlich die Bearbeitung einer Störung in Incident Processing hinzu. Für die Übernahme der Änderung(en) in die Produktion auf beiden Ebenen sorgt der Prozess Solution Rollout and Service Deployment.

Die Prävention ist bei BS15000 und ITIL dem Problem Management zugeteilt. In MITO gehören die statistische Auswertung der Mängel und Störungen und das Veranlassen einer Änderung zu den Aufgaben des Service Change Managements. In unserem Modell fällt diese Aufgabe allen Prozessen zu, die am Erstellen oder Ändern der Lösung beteiligt sind, primär Service Design und Solution Engineering: Es ist ihre ureigenste Aufgabe, das Feedback auf ihr Tun in Form der Anliegen zu analysieren, um einerseits eine Änderung der Lösung im Sinne der Prävention zu veranlassen und andererseits in der Engineering- und Entwicklungsarbeit gegen die bereits bekannte Art von Mängeln gefeit zu sein.

Im Gegensatz zu ITIL unterscheidet MITO konsequent die zwei Ebenen Service und Solution sowohl für die Management- als auch für die Ausführungsaufgaben. Wir haben uns hier vor allem auf die Trennung der Management- und Ausführungsaufgaben konzentriert. Das Issue Management wirkt auf beiden Ebenen, die anderen Managementprozesse werden für beide Ebenen (evtl. mehrfach) instanziiert. Die ausführenden Prozesse sind immer die gleichen, unabhängig davon, auf welche Art sie von den Managementprozessen angetrieben werden; dies wird bei einer Störung etwas hek-

Hier	ITIL	MITO
Konfigurationsmanagement	Configuration Management	Solution Change and Configuration Management
Issue Management	*der Managementteil von* Incident Management, Problem Management, Change Management	Issue Management
Release Management	*der Managementteil von* Change Management	Service Change Management, Solution Change and Configuration Management
Launch Management	Release Management	Solution Rollout and Service Deployment
Service Design, Solution Engineering, Build, Solution Rollout, Service Deployment	*der Ausführungsteil von* Incident Management, Problem Management, Change Management, Availability Management,	Incident Processing, Service Design and Update, Solution Implementation and Maintenance
Service Design, Solution Engineering	Problem Management	Service Change Management

Kasten 2: Zuordnung zu Bezeichnungen in ITIL und MITO

tischer sein als bei einer Neuentwicklung an ihrem Anfang.

5 Schlussfolgerungen

Ein Grund für die Schwierigkeiten beim Etablieren vom Konfigurationsmanagement ist, dass die Nutznießer immer nur die anderen sind. Meint man. »Ich weiß ja, was ich geändert habe«, hört man da allenthalben, »und wenn es nicht funktioniert, dann korrigiere ich das halt wieder.« Dabei vergisst man, dass man nicht alle Beziehungen überschaut und die Änderung z.B. eines einzigen Parameters zu unerwünschten Effekten in einem sonst nicht berührten Systemteil führen kann. Bei jeder Unbill müsste also der Feuerwehrmann zuerst alle Kollegen fragen, ob sie etwas in den letzten Stunden geändert haben und was es gewesen ist. Dies ist um Mitternacht herum eine sehr unpopuläre Maßnahme. In vielen Umgebungen jedoch der schnellste Weg, jeden merken zu lassen, dass das Festhalten der Information über Änderungen nicht nur den Kollegen zugute kommt, sondern auch dem eigenen Schlaf.

Die verwandten Prozesse Issue Management, Release Management und Launch Management bauen auf die Unterstützung durch das Konfigurationsmanagement. Die Mängelmeldung ohne eindeutige Kennung der vom Mangel betroffenen Einheiten ist ein Muster mit beschränktem Wert. Die Planung eines Release ohne eindeutige Bezeichnung der Änderungsanträge und Mängelmeldungen ist schwer kommunizierbar. Ohne Kenntnis der Beziehungen zwischen den Einheiten, die von den Änderungen betroffen sind, ist es unmöglich, die günstigste Reihenfolge der Bearbeitung für den Plan zu finden. Und wie soll man das Release mit allen durchgeführten und in der Prüfung für gut befundenen Änderungen in die Produktion überführen, wenn man nicht weiß, was man geprüft hat oder gar was alles geändert wurde? Konfigurationsmanagement ist die Drehscheibe zwischen den Prozessen, ist ein Bindeglied.

Konfigurationsmanagement produziert nicht, spezifiziert keine Systeme, entwirft nicht, plant nicht, implementiert nicht, prüft oder betreibt nicht. Hygiene in einem Spital trägt zur Gesundung auch nicht bei, die Hygienemaßnahmen verlangsamen nur den Diagnose- und Therapieprozess. Sie sind jedoch Voraussetzung dafür, dass der Patient während der Behandlung nicht kranker wird, als er beim Spitaleintritt war. Ähnlich sorgt Konfigurationsmanagement dafür, dass man im Entwickeln und Betreiben mit »sauberen« Einheiten und Konfigurationen operiert, d.h. jeweils an den richtigen (Versionen von) Einheiten arbeitet, die richtigen Konfigurationen manipuliert. Es hilft, Änderungen ungestört und gelenkt durchführen zu können und inmitten der Veränderung Übersicht zu bewahren.

6 Literatur

[BS15000] BS15000:2000 Specification for IT Service Management.

[Frühauf et al. 2000] *Frühauf, K.; Ludewig, J.; Sandmayr, H.:* Software-Projektmanagement und -Qualitätssicherung. 4. Auflage, vdf, Zürich, 2000, ISBN 3-7281-2822-8.

[ITIL 2000] *ITIL:* Service Support, Office of Government Commerce, UK, 2000.

[ITIL 2001] *ITIL:* Service Delivery, Office of Government Commerce, UK, 2001.

[Scheuing 2003] *Scheuing, A. (Hrsg.):* MITO – Ein Reifegradmodell für den Informatik-Betrieb (MITO – Maturity model for IT-Operation). SAQ – Swiss Association for Quality, Fachgruppe Informatik, 2003, ISBN 3-85849-020-2.

Karol Frühauf
Gerald Linhofer
INFOGEM AG
Rütistr. 9
CH-5401 Baden
karol.fruehauf@infogem.ch
gerald@linhofer.com
www.infogem.ch