Application Lifecycle = Engineering & Betrieb

Die richtige Nervennahrung verlängert das Leben

Softwareanwendungen sind ein zentraler Lebensnerv der heutigen Geschäftswelt. Ihr funktionaler Nutzen und ihre betriebliche Integration haben großen Einfluss auf den Unternehmenserfolg. Aus diesem Grund ist es enorm wichtig, nicht nur die Bereitstellung und den Betrieb, sondern auch die Wartung und zielgerichtete Weiterentwicklung von Softwareanwendungen sicherzustellen. Dazu ist eine praxisorientierte Methodik zur optimalen Unterstützung des gesamten Application Lifecycle, also der Lebensdauer einer Applikation, notwendig.

Damit Software im betrieblichen Umfeld über Jahre hinweg einsatzfähig und nützlich bleibt, muss dieser Asset wie andere wichtige Grundlagen und Instrumente der unternehmerischen Tätigkeit gepflegt und verwaltet werden – und das über den ganzen Lebenszyklus hinweg.

PLAN – BUILD – RUN – RETIRE

Bei genauerem Hinsehen ist der Application Lifecycle eines typischen betrieblichen Informationssystems jedoch anders gestaltet als der Lebenszyklus anderer Softwarearten. Im Gegensatz zu Infrastruktursoftware – wie beispielsweise Betriebssystem, Datenbank oder Application Server – sind die Lebenszyklen betrieblicher Informationssysteme weitaus weniger linear. Der Grund: Der typische Application Lifecycle eines solchen Systems ist durch seine Nähe zum Business geprägt. Dies bedeutet, dass sowohl die Gestaltung als auch die Update- und Modifikationsintervalle durch Geschäftsanforderungen gesteuert werden – und nicht durch herstellerseitige Release-Zyklen.

Software hat wie viele Produkte aus der industriellen Fertigung einen Lebenszyklus. Sie wird geplant, erstellt, geliefert, betrieben und schließlich durch ein neues Produkt ersetzt. Im Gegensatz zu anderen Produkten ist Software jedoch nach der Fertigstellung relativ einfach modifizierbar. Diese Flexibilität ermöglicht zwar Anpassungsfähigkeit an spezifische betriebliche Gegebenheiten, sie hat aber auch ihren Preis: Auch Software wird immer mehr zum Einzelstück. Das macht die Anwendung der klassischen Instrumente und Produkte für Product Lifecycle Management (PLM) – wie Programm-Management, Lifecycle Change und Configuration Management sowie Product und Process Management – praktisch unmöglich.

Lebenszyklen in der Theorie

Aus diesem Grund existieren spezielle Techniken, Methoden und Instrumente für Software Lifecycle Management (SLM), die mit ISO/IEC 12207 (Software Lifecycle Processes) sogar weitgehend standardisiert sind. Dieser Standard definiert die notwendigen Prozesse, um Anwendungen über ihren gesamten Lebenszyklus hinweg zielgerichtet verwalten zu können. Zu den wichtigsten Prozessen gehören die Beschaffung, Lieferung, Entwicklung sowie der Betrieb und die Wartung von Software. Die ISO-Norm definiert jedoch nur einen Rahmen für die verschiedenen Prozesse und hat zur Entwicklung weiterer Standards wie beispielsweise „Software Process Improvement and Capability Determination“ (SPICE) oder dem „Capability Maturity Model“ (CMM/CMMI) geführt. Trotz ihres sehr informativen und universellen Charakters sind diese Standards nur in Großunternehmen sinnvoll als Ganzes anzuwenden – für Firmen mittlerer Größe sind praktischere Modelle gefragt.

Lebenszyklen in der Praxis

Ein Application Lifecycle im betrieblichen Umfeld eines mittleren Unternehmens – also in einem Unternehmen mit ca. 100 bis 250 Mitarbeitern und einem Umsatz zwischen 20 und 100 Millionen Euro – unterscheidet sich grundsätzlich nicht von demjenigen im Kontext eines großen Unternehmens. Es ist jedoch zu erwarten, dass die Anzahl der zentral eingesetzten Softwareprodukte deutlich geringer ist. Dies gilt v. a. für den Bereich der sog. Enterprise Backoffice Systems, also der Kernanwendungen für die Unterstützung der betrieblichen Tätigkeit. Und genau das ist das Einsatzgebiet von Midrange Systemen, wie Colin Parris im IBM White Paper zur Strategie und Roadmap der IBM Power

Systems festhält (C. Parris: IBM i Strategy and Roadmap, IBM White Paper 2012).

Darüber hinaus ist zu erwarten, dass Software, die auf solchen Systemen läuft, länger als die durchschnittlichen 12 bis 15 Jahre im Einsatz bleibt. Zudem enthalten die Anwendungen zum Großteil individuell entwickelte oder speziell auf ein Geschäftsfeld angepasste Funktionen, die im Lauf der Jahre immer wieder angepasst wurden. In Bezug auf den Lebenszyklus bedeutet dies, dass die RUN-Phase die weitaus wichtigste ist und andere Prozesse, wie beispielsweise Beschaffung oder RETIRE, eine eher untergeordnete Rolle spielen. In dieser Phase sind genau zwei Dinge wichtig: das Engineering und der Betrieb der Softwarelösung.

Engineering und Betrieb

Das Engineering von Software – ob nun Anpassung und Parametrisierung von Standardsoftware oder Weiterentwicklung von Individualsoftware – erfordert den Einsatz von Instrumenten für die Umsetzung, Durchführung und Fehlerbehandlung sowie die Qualitätssicherung. Die Kontrolle der Umsetzung wird auf die plan- und messbare Entwicklung eines definierten Funktionsumfangs entlang der drei vorgegebenen Kenngrößen Kosten, Zeit und Qualität ausgerichtet.

Die Tools für die Durchführung der Fehlerbehandlung und die Mechanismen für die Qualitätssicherung sind meist direkt in den Entwicklungsinstrumenten integriert. Der Betrieb von Software erfordert jedoch auch die Kontrolle der Betriebsprozesse. Dabei wird besonderer Wert auf die plan- und kontrollierbare Nutzung von IT-Services gelegt und werden vertraglich vereinbarte Service Level Agreements (SLA) regelmäßig nach fest definierten Parametern geprüft. Dazu zählen v. a. folgende Parameter: Funktionalität, Kapazität und Verfügbarkeit. Die Kontrolle dieser Parameter endet nicht bei der Software, sondern erstreckt sich auch auf die Hardware, die Netzwerke sowie die Betriebsorganisation.

In der Praxis ist ein gutes Management des IT-Lebenszyklus nur dann möglich, wenn Aspekte des Engineerings und Aspekte des Betriebs kombiniert werden. Das Engineering hat Eigenschaften wie die Modularität, die Formbarkeit oder die Herstellbarkeit im Fokus, um eine möglichst gute Anpassung an sich verändernde betriebliche Gegebenheiten zu gewährleiten. Der Betrieb beinhaltet jedoch die Garantie von Reaktionszeiten, Sicherheit und Stabilität. Ein gutes Application Lifecycle Management kombiniert die Bedürfnisse dieser Aspekte.

Das bedeutet, dass die Anpassungen an laufender Software immer in Absprache und Abstimmung mit den Bedürfnissen eines geregelten Betriebs zu erfolgen haben. Die Kontrollinstrumente, die im Bereich der geschäftskritischen Midrange Systeme eingesetzt werden können, sind jedoch auf das Engineering ausgerichtet. Das heißt, dass die Planung und Umsetzung von Anpassungen ebenso wie Änderungen durch eine vollständige und nachvollziehbare Bearbeitung sämtlicher Fehlermeldungen von adäquaten Instrumenten abgedeckt werden müssen. Gleiches gilt auch für die Qualitätssicherung. In der Praxis hat sich der Einsatz von umfangreichen Entwicklungsumgebungen, wie beispielsweise IBM Rational, und die Kombination mit guten Ticketing-Systemen, wie IBM Tivoli oder HP Openview, als gangbarer Weg erwiesen. Außerdem ist der Einsatz von Testmethoden zu empfehlen, die mögliche Effekte von Anpassungen auf andere Teile einer Anwendung prüfen – auch wenn diese nach der Umsetzung nur wenig genutzt werden.

Gerade im langlebigen Umfeld der Midrange Systeme ist eine Vielzahl stark in die Jahre gekommener Anwendungen zu finden. Diese Anwendungen sind aus Sicht der Businessfunktionalität oftmals sehr umfangreich und wertvoll – und damit keineswegs veraltet. In vielen Fällen könnte die bestehende Software weiter verwendet werden, wenn nur die Schnittstellen moderner wären.

Modernisierung als Verlängerung des Lifecycle

Die IBM und ihr Partnernetzwerk bieten eine Vielzahl von Tools und Instrumenten zur Modernisierung bestehender Legacy-Systeme. Dabei gilt es jedoch, stets einen pragmatischen Kompromiss zwischen idealer Lösung und der bestehenden Anwendung zu finden. Wer Application-Lifecycle-Instrumente verwendet, ist hier klar im Vorteil:

Ständige Anpassungen bestehender Software an neue betriebliche Anforderungen vorzunehmen, bedeutet im Grunde nichts anderes, als den Zustand und die Funktionalität einer bestehenden Anwendung im Auge zu behalten. Wird das beachtet, bleiben dem Unternehmen kostspielige Überraschungen bei „Renovationsarbeiten“ garantiert erspart.

Der Autor Daniel Liebhart ist Dozent für Informatik an der ZHAW und Solution Manager der Trivadis AG. Er ist Autor des Buchs „SOA goes real“ (Hanser Verlag) und Co-Autor verschiedener Fachbücher. ■

www.trivadis.com