

Harry M. Sneed · Ellen Wolf · Heidi Heilmann

Softwaremigration in der Praxis

**Übertragung alter Softwaresysteme in eine
moderne Umgebung**



dpunkt.verlag

Harry M. Sneed
Harry.Sneed@t-online.de
Ellen Wolf
ellen.wolf@cassini.de
Heidi Heilmann
heidi.heilmann@augustinum.net

Lektorat: Christa Preisendanz
Copy-Editing: Alexander Reischert, Redaktion ALUAN
Herstellung: Birgit Bäuerlein
Umschlaggestaltung: Helmut Kraus, www.exclam.de
Druck und Bindung: Media-Print Informationstechnologie, Paderborn

Fachliche Beratung und Herausgabe von dpunkt.büchern im Bereich Wirtschaftsinformatik:
Prof. Dr. Heidi Heilmann · heidi.heilmann@augustinum.net

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

ISBN 978-3-89864-564-5

1. Auflage 2010
Copyright © 2010 dpunkt.verlag GmbH
Ringstraße 19 B
69115 Heidelberg

Die vorliegende Publikation ist urheberrechtlich geschützt. Alle Rechte vorbehalten. Die Verwendung der Texte und Abbildungen, auch auszugsweise, ist ohne die schriftliche Zustimmung des Verlags urheberrechtswidrig und daher strafbar. Dies gilt insbesondere für die Vervielfältigung, Übersetzung oder die Verwendung in elektronischen Systemen.

Es wird darauf hingewiesen, dass die im Buch verwendeten Soft- und Hardware-Bezeichnungen sowie Markennamen und Produktbezeichnungen der jeweiligen Firmen im Allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Alle Angaben und Programme in diesem Buch wurden mit größter Sorgfalt kontrolliert. Weder Autor noch Verlag können jedoch für Schäden haftbar gemacht werden, die in Zusammenhang mit der Verwendung dieses Buches stehen.

5 4 3 2 1 0

Vorwort

Die IT-Welt ist ständig im Wandel begriffen. Die Mächtigkeit der Prozessoren verdoppelt sich nach dem Gesetz von Moore alle 18 Monate [Moor65]. Leistungsstärkere Hardware verlangt nach leistungsfähigerer Software mit zusätzlicher Funktionalität, um die höhere Hardwarekapazität auszunutzen. Die Betriebssysteme müssen demzufolge im Rhythmus von 36 Monaten für jede zweite Prozessorgeneration erneuert werden. Ihre Veränderung wirkt sich auf Programmumgebung und Datenbanksysteme aus, die mindestens alle fünf Jahre angepasst bzw. neu entwickelt werden müssen. Daraus resultiert die Halbwertszeit von fünf Jahren für Anwendungssoftware.

Zu diesem Anpassungszwang aus dem Prozessor- und Betriebssystemsektor gesellt sich der Veränderungsdruck aus dem Kommunikationsbereich. Datenübertragungskapazität und -geschwindigkeit erhöhen sich durch die Nutzung neuer Technologien wie Glasfaserkabel exponentiell. Gleichzeitig wird der Transport von Daten immer billiger, denn nach dem Gesetz von Metcalfe gilt: Je mehr Nachrichten gesendet werden, desto billiger wird das Senden einer einzelnen Nachricht [Metc73]. Dies schafft eine Überkapazität an Kommunikationseinrichtungen, die darauf ausgelegt sind, ein Maximum an Last zu tragen. Von den IT-Nutzern wird erwartet, dass sie diese zunehmende Kapazität sinnvoll in Anspruch nehmen. Dazu müssen sie aber ihre Anwendungen neu ausrichten.

Diese beiden technischen Zwänge – zunehmende Rechnerkapazität gekoppelt mit zunehmender Kommunikationskapazität – legen den IT-Anwendern nahe, ihre Anwendungen zu erneuern. Gleichzeitig kommt der Druck aus dem Markt, sich geschäftlich anders aufzustellen. Der wirtschaftliche Konkurrenzkampf verlangt nach einer Anpassung an die veränderten Wettbewerbsbedingungen. Unternehmen müssen schneller, besser und flexibler auf die globale Nachfrage reagieren. Dafür muss sich die Anwendungssoftware leichter ändern und ausbauen lassen. Dies verlangt wiederum neue Sprachen und Entwicklungsumgebungen.

Diese Kombination von technischen und betriebswirtschaftlichen Zwängen setzt IT-Anwender dem Druck aus, ihre eigene, selbstentwickelte Software regelmäßig zu erneuern. Eine Alternative dazu besteht darin, die Software komplett

neu zu entwickeln. Das ist jedoch teuer und risikoreich. Es setzt außerdem voraus, dass das Wissen über die Funktionalität der Software noch vorhanden ist: Diese Voraussetzung ist jedoch nicht immer erfüllt. Eine weitere Alternative ist die Sanierung der Software im Rahmen der bestehenden Umgebung. Wenn aber diese Umgebung selbst veraltet ist, macht das wenig Sinn, da man bald an Grenzen stößt. Eine dritte Alternative ist die Migration der Software in eine andere, vermutlich bessere Umgebung bzw. auf eine zeitgemäße Plattform, die es dem Anwender ermöglicht, die Fortschritte in der Rechner- und Kommunikationstechnologie besser auszunutzen und schneller und kostengünstiger auf neue fachliche Anforderungen zu reagieren.

Viele Anwendungssysteme sind für die Anwender ohne geschäftskritische Relevanz, nach Nicolas Carr sind sie lediglich »Commodity« [Carr03]. Es kommt weniger darauf an, wie sie funktionieren, sondern vor allem, dass sie funktionieren. Dennoch müssen auch solche Systeme gepflegt werden, um am Leben zu bleiben, und irgendwann stirbt das Pflegepersonal aus. Junge Entwickler sind kaum bereit, sich in alte Sprachen und alte Umgebungen einzuarbeiten. Somit drohen viele dieser alten »Commodities« zu verwaisen, wenn sie nicht in eine neue Sprache und eine neue Umgebung versetzt werden. Dies ist ein weiterer Grund für die Softwaremigration, der von den zuständigen IT-Managern immer häufiger genannt wird: »Wir finden kein Personal mehr, um das alte System zu pflegen.« Ergo bleibt gar nichts anderes übrig, als das alte System neu zu entwickeln oder zu migrieren.

Gerade jetzt, zu einer Zeit, in der die erste Entwicklergeneration die Bühne verlässt, ist das Thema Migration aktueller denn je. Sämtliche Eigenentwicklungen der 70er- und 80er-Jahre des letzten Jahrhunderts stehen zur Disposition. Wenn sie nicht durch Standardsoftware zu ersetzen sind und weiter gebraucht werden, können sie nur neu entwickelt oder migriert werden. Da Zeit und Mittel für eine Neuentwicklung häufig fehlen, bleibt oft nur die Migration als Lösung übrig.

In dieser Hinsicht ähnelt Software der Atomkraft: Sie kann richtig angewandt sehr nützlich sein, birgt aber auch große Gefahren in sich. Die Hauptgefahr besteht darin, sich davon nicht wieder befreien zu können. Auch wenn es gelingt, die alte Lösung in eine neue technische Umgebung zu übertragen, bleibt diese alte Lösung samt suboptimaler Algorithmen und Datenstrukturen erhalten. Besonders nachteilig ist die Tatsache, dass das Wissen, das hinter diesen Algorithmen und Datenstrukturen steckt, ein für alle Male verloren ist. Sofern es keine aktuelle Dokumentation gibt, die fast immer fehlt, ist es wegen der Benennung von Daten und Funktionen im Code und der mangelhaften Kommentierung schier unmöglich, dieses Wissen wiederzugewinnen. Man kann zwar die Lösung migrieren, aber das Problem, das sie löst, bleibt verborgen. Die Legacy-Falle ist deshalb viel gefährlicher, als es die meisten Softwareanwender wahrhaben wollen. Das Motto der ersten internationalen Konferenz zum Thema »Software Reengineering« im Jahr 1993 hieß »Given the solution, what is the problem«. Diese Frage ist bis heute unbeantwortet.

Dieses Buch ist für alle gedacht, denen eine Migration bevorsteht. Sie müssen begreifen, dass es nicht möglich ist, einen Esel in ein Rennpferd zu migrieren. Dennoch: Vor die Wahl gestellt, zu Fuß zu gehen oder auf einem Esel zu reiten, werden sie den Esel vorziehen. Eine Migration ist in der Tat nur eine Notlösung, verbunden mit vielfältigen Qualitätsabstrichen und faulen Kompromissen. Aber sie ist preiswert, zügig zu erledigen und relativ risikolos. So gesehen ist sie die geeignete Lösung für Anwender, die Zeit gewinnen wollen, bis sie sich eine neue Lösung leisten können.

Danksagung

Die drei Autoren möchten einen besonderen Dank an Herrn Professor Dr. Andreas Winter aussprechen. Er hat die Diplomarbeit von Frau Wolf an der Universität Koblenz-Landau betreut und viele wichtige Impulse zur Entstehung des Buches gegeben. Da das Buch aus dieser Diplomarbeit hervorgegangen ist, ist er als Geburtshelfer des Buches anzusehen. Frau Heilmann hat als Herausgeberin schon frühere Fassungen des Manuskripts redigiert, ehe sie als Mitautorin eingestiegen ist. Ihr danken die anderen beiden Autoren, dass aus jenem Manuskript endlich ein Buch geworden ist.

Harry M. Sneed, Ellen Wolf, Heidi Heilmann

Arget, Kobern-Gondorf, Überlingen im Februar 2010