

Einleitung

*I have travelled the length
and breadth of this country
and talked with the best people,
and I can assure you
that data processing is a fad
that won't last out the year.*

Editor Business-Books
Prentice-Hall, 1957

1.1 Lamento

In den letzten Jahrzehnten haben sich Forschung und Lehre innerhalb der Softwareentwicklung primär mit der Erstellung von Software befasst, die Maintenance wie auch die systematische Weiterentwicklung wurden stets stark vernachlässigt. Für die frühen Phasen des Lebenszyklus wurden von großen Softwareherstellern viele Werkzeuge geschaffen, aber nur wenige für Themen wie Reengineering, Migration oder auch Maintenance. Es ist eine Schieflage entstanden, welche die tatsächliche Situation verzerrt wiedergibt.

Informatikstudiengänge sowie Softwareengineeringkurse vermitteln den Studenten und Teilnehmern Methodiken, um Software quasi „auf der grünen Wiese zu bauen“ – Konzepte wie Architektur, Effektivität, Robustheit oder auch Performanz werden mit dem Fokus auf der Synthese und selten auf Analyse vermittelt. Die Auseinandersetzung mit tatsächlich eingesetzter Software gilt als „schmutzig“ und wird sehr selten durch Lehrkörper und Seminarleiter unterstützt. Dies steht in krassem Gegensatz zu anderen Gebieten wie dem Bauingenieurwesen, dem Städtebau oder der klassischen Architektur. Diese sehen es als ihre Pflicht an, bestehende Objekte eingehend zu studieren, um aus der Vergangenheit zu lernen. Nicht so die Softwareentwicklung! Hier werden stets „neue“ Methoden entwickelt, um „neue“ Applikationen zu schaffen. Dabei werden alle bisher genutzten Methoden als überholt angesehen. Neue Vorgehensweisen werden oft mit einem Fanatismus vertreten, welcher erstaunliche Parallelen zu religiösen Zeloten aufweist.¹ Durch diese Negati-

¹ Bezeichnenderweise hat die Firma Sun „Java-Evangelisten“ und Microsoft „.NET-Evangelisten“.

on der Erfahrung wird das angesammelte Wissen über Softwareentwicklung weder tradiert noch genutzt und „alte“ Fehler werden permanent wiederholt.

Selbst in Organisationen, welche schon jahrzehntelange Erfahrungen im Umgang mit ihren eigenen Altsystemen haben, ist zu beobachten, dass man Lippenbekenntnisse für diese Systeme abgibt und beteuert, sich der Problematik bewusst zu sein, es wird jedoch nicht in einer geordneten Art und Weise konstruktiv mit diesen Altsystemen umgegangen. Ziel dieses Buches ist es, Wege für den Umgang mit diesen Altsystemen aufzuzeigen.

In der angelsächsischen Literatur werden für Altsysteme oft die Begriffe „*legacy software*“ und „*legacy systems*“ genutzt. Dieses Buch folgt diesem Sprachgebrauch und nutzt für die Software in Altsystemen den Begriff *Legacysoftware* und für das Altsystem als Ganzes den Begriff *Legacysystem*.

1.2 Legacysystem

Was ist jedoch ein Legacysystem, oder was charakterisiert es?

Ein Legacysystem ist ein soziotechnisches System, welches Legacysoftware enthält.

Folglich machen Legacysysteme nur innerhalb einer Organisation Sinn. Umgekehrt lässt sich ein solches System nicht abstrakt, quasi im Vakuum, ohne Kenntnis der dazugehörigen Organisation beurteilen. Beide, das Legacysystem wie auch die Organisation, sind parallel miteinander und aneinander gewachsen und schwer trennbar verwoben.

Legacysoftware ist eine geschäftskritische Software, welche nicht, oder nur sehr schwer, modifiziert² werden kann.

Die Lebensdauer eines Softwaresystems kann sehr unterschiedlich sein und viele Großunternehmen haben Software im Einsatz, welche über 30 Jahre alt ist. Die meisten dieser großen Softwaresysteme sind geschäftskritischer Natur, d.h. sie sind im Tagesgeschäft unabdingbar. Ein Ausfall oder Wegfall würde den Zusammenbruch des Unternehmens, oder doch zumindest herbe Verluste, bedeuten. Diese Softwaresysteme nennt man Legacysysteme und die in ihnen enthaltene Software Legacysoftware. Zwar ist die Legacysoftware immer Bestandteil eines Legacysystems, doch kann das Legacysystem als Ganzes auch neuere Software enthalten, was der Regelfall ist.

Die Legacysysteme sind naturgemäß nicht dieselben Systeme, die vor 30 Jahren in Betrieb genommen wurden. Nein, sie haben sich oft sehr stark gewandelt und wurden, in der Regel, viel komplexer als ursprünglich antizipiert.

² Salopp formuliert: Legacysoftware ist Software, bei der wir nicht wissen, was wir mit ihr tun sollen, die aber noch immer eine wichtige Aufgabe erfüllt.

Ein großer Teil der Legacysoftware ist überhaupt nicht für diese lange Lebensdauer konzipiert worden.³ Diverse äußere und innere Quellen waren für die Veränderungen an der Software verantwortlich; speziell die Veränderungen, welche durch das Geschäftsumfeld ausgelöst wurden, zwangen alle Softwaresysteme, sich anzupassen oder obsolet zu werden. Diese Form des Darwinismus führte zu einer Auslese, deren Ergebnis wir heute als Momentaufnahme sehen. Der Selektionsdruck wurde primär durch die Fähigkeit zur Anpassung ausgelöst, d.h. nur eine Legacysoftware, welche in der Lage war, sich in den letzten 30 Jahren anzupassen, ist heute noch existent. Neben den äußeren Einflüssen haben sich auch die unterschiedlichsten Softwareentwickler an der Legacysoftware versucht. Bei einer durchschnittlichen Betriebszugehörigkeit von 5-7 Jahren sind das immerhin 5 Generationen von Softwareentwicklern bei einem Systemalter von 30 Jahren. So ist es sehr ungewöhnlich, einen Menschen anzutreffen, der ein komplettes Verständnis des gesamten Legacysystems besitzt.

Die sehr lange Koexistenz von Legacysoftware und den Geschäftsprozessen innerhalb des Unternehmens hat zu einer Verknüpfung beider geführt, die jenseits der Dokumentation und der Wahrnehmung liegt. Die Geschäftsprozesse haben „gelernt“, die Stärken der Legacysysteme zu nutzen, bzw. ihre Schwächen zu umgehen. Geschäftsregeln, Abläufe, Ausnahmen und vieles andere sind in die Legacysoftware eingeflossen, ohne dass nach so langer Zeit noch ein einzelner Softwareentwickler dies vollständig weiß. Von daher birgt jeder Ersatz das Risiko, etwas vergessen zu haben.

Neben den hohen Risiken für eine Ablösung des Legacysystems selbst, stehen die stetig steigenden Kosten für die Maintenance der Legacysoftware. Diese Kosten sind nicht zu unterschätzen; so betrugen die Kosten für den Betrieb im Jahre 2002 nach Angaben von McKinsey in großen Unternehmen 65-75 % des gesamten IT-Budgets.

Die Unternehmen stehen also vor einem Dilemma.

- Auf der einen Seite ist es risikoreich das Legacysystem zu ersetzen.
- Auf der anderen Seite steigen die Unterhaltskosten stetig an!

Ein Legacysystem ist nicht nur einfach ein Stück „alte“ Software, sondern ein hochkomplexes soziotechnisches Gebilde, bestehend aus den unterschiedlichsten Teilen, s. Abb. 1.1.

Die logischen Teile eines solchen Systems sind:

- Hardware – In vielen Fällen wurde die Legacysoftware für eine Mainframehardware konzipiert und gebaut, welche heute obsolet ist⁴ oder nicht

³ Softwareentwickler haben für die lange Lebensdauer solcher Systeme das Sprichwort *Provisorien leben am längsten* geprägt.

⁴ In den letzten Jahren ist allerdings eine Renaissance der Mainframes zu beobachten. Nach vielen Jahren eines „Weg-von-der-Mainframe“-Programms wird die Mainframe als Backendsystem heute wieder gesellschaftsfähig.

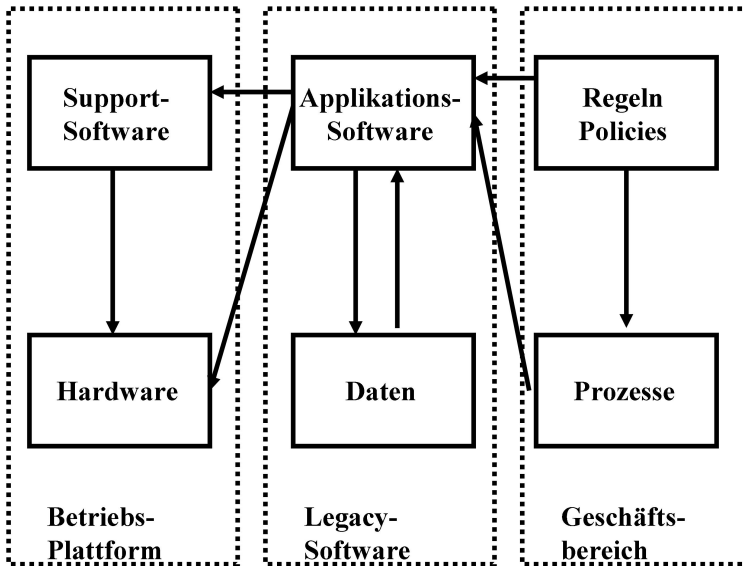


Abb. 1.1: Das Legacysystem als soziotechnisches Gebilde

mehr zur aktuellen strategischen Ausrichtung des betroffenen Unternehmens passt.

- Supportsoftware – In der Regel verlässt sich die Legacysoftware auf bestimmte Betriebssystemeigenschaften, sowie Utilities, welche vorhanden sein müssen. Dass ein Betriebssystemwechsel Probleme verursachen kann, ist nicht nur in der Mainframewelt bekannt, sondern war auch in der Windowswelt diverse Male schmerzhaft spürbar.
- Applikationssoftware – Dies ist die eigentliche Legacysoftware. Oft wird hierfür fälschlicherweise auch der Begriff Legacysystem benutzt.
- Daten – Hierbei handelt es sich um die Daten, welche von der Legacysoftware verarbeitet und genutzt werden. Diese können entweder datei- oder datenbankbasiert existieren.
- Prozesse – Die Geschäftsprozesse werden von den Unternehmen eingesetzt um ein Ergebnis zu erhalten; sie dienen der Durchführung von Tätigkeit um das gegebene Unternehmensziel zu erreichen.
- Regeln – Innerhalb der Geschäftswelt existiert ein reicher Regel- und Verhaltenscodex für Tätigkeiten innerhalb einer Domäne. So unterliegt beispielsweise die Buchhaltung strengen Wirtschaftsprüferregeln.

Das Legacysystem besteht aus allen diesen Teilen und alle verändern sich bzw. lösen Veränderungen aus, insofern ist ein Legacysystem ein hochkomplexes Gebilde. Als solches lässt es sich nicht auf einfache Lösungsschemata reduzieren.

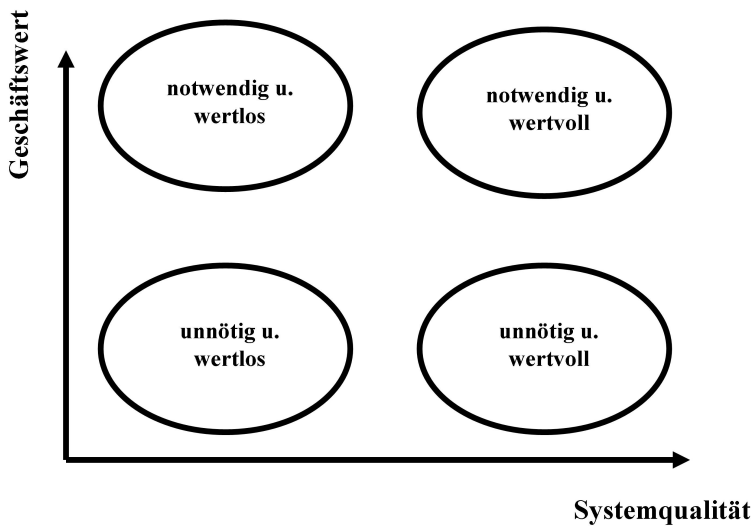


Abb. 1.2: Das Legacysystem-Assessment

1.3 Assessment

Bevor man sich überhaupt Gedanken über eine mögliche Strategie im Umgang mit der Legacysoftware macht, sollte man ein Assessment, d.h. eine Bewertung des Legacysystems, vornehmen. Eine solche Bewertung ist nicht nur für Legacysysteme unabdingbar, sondern sollte grundsätzlich für alle Systeme in Betracht gezogen bzw. durchgeführt werden. Alle im Einsatz befindlichen Softwaresysteme, nicht nur solche vom Typus Legacy, fallen in vier Kategorien, s. Abb. 1.2, wobei die Dimensionen, hier der Geschäftswert und die Qualität der Software, auf der Ordinate bzw. der Abszisse dargestellt sind. Diese Kategorien sind:

- **unnötig und wertlos** – Solche Systeme sind teuer und tragen nicht wirklich zum Geschäftserfolg bei. Im engeren Sinne handelt es sich hierbei nicht um Legacysysteme, da diese per definitionem geschäftskritisch sind, was Systeme mit niedrigem Geschäftswert nicht sein können.
- **notwendig und wertlos** – Eine ganze Kategorie von Legacysystemen fällt in diese Klassifikation, da solche Systeme ihren heutigen Zustand oft über eine sehr lange Evolutionsphase erreicht haben, s. Kap. 4. Im Sinne der unterschiedlichen Migrationsstrategien, s. Kap. 5, sind dies ideale Kandidaten für einen Ersatz.
- **unnötig und wertvoll** – Hierbei handelt es sich nicht um Legacysysteme im eigentlichen Sinne, da der notwendige geschäftskritische Anteil für eine

Legacysoftware nicht gegeben ist. Trotzdem wird man diese Systeme einer langen und sorgfältigen Maintenance, s. Kap. 7, unterziehen. Aus diesem Blickwinkel können ähnliche Mechanismen wie bei der Legacysoftware auf diese Systeme angewandt werden.

- notwendig und wertvoll – Für diese Systeme gilt dasselbe wie für die vorhergehende Klassifikation mit dem einzigen Unterschied, dass es sich hierbei tatsächlich um Legacysysteme handelt.

1.4 Dualismen

Ein Teil der Probleme, welche heute mit den Legacysystemen auftauchen, gehen auf einen fundamentalen Wandel zurück, der Ende der achtziger Jahre stattgefunden hat. Es ist der Wandel in der Betrachtung von Software. Das Verhältnis zwischen Hard- und Software kann von zwei unterschiedlichen Blickwinkeln aus gesehen werden:

- Die Software **steuert** die Hardware.
- Die Hardware **führt** die Software **aus**.

In der ersten Betrachtungsweise ist die Software ein Kontrollprogramm, welches die Hardware, sprich den Computer, kontrolliert. In der anderen Sichtweise ist es genau umgekehrt. Der Computer stellt eine Ablaufumgebung zur Verfügung und er „kontrolliert“ das Programm. In den sechziger und siebziger Jahren war die erste Denkweise weit verbreitet und kann noch heute in Realtime- bzw. *Embedded*-Systemen wiedergefunden werden.

Die Spuren der ersten Denkweise lassen sich oft in der Legacysoftware wiederfinden. Dies ist nicht besonders verwunderlich, da ein großer Teil der Softwareentwickler, die an den entsprechenden Legacysystemen beteiligt waren, in den frühen Jahren der IT ihr Handwerk erlernten. Der Versuch den Computer zu kontrollieren führt zu spezifischen Konstrukten, welche nur für einen bestimmten Compiler oder eine besondere Hardwareplattform Sinn machen. Für Softwareentwickler, welche die zweite Denkweise verinnerlicht haben, erscheinen diese Konstrukte sehr verwirrend, da in der zweiten Denkweise die Hardware nicht berücksichtigt wird oder werden muss.

Ein zweiter Dualismus ist im Zusammenhang mit Legacysoftware zu beobachten: Wir sind in der Lage, uns sehr schnell an schnelle Änderungen anzupassen, aber wir sind fast gar nicht in der Lage, uns auf langsame Veränderungen einzustellen. Ein Beleg dafür ist die Entwicklung des World Wide Web oder der mobilen Telefonie: Wir konnten uns sehr schnell an diese schnelle Veränderung anpassen, doch die Daten der meisten Großunternehmen befinden sich noch im Zustand der siebziger Jahre!

Diese unterschiedlichen Sichten auf Software und ihre Entwicklung sind nicht die einzigen Dualismen, die im Umfeld der Legacysysteme vorkommen. Ein anderes Spannungsfeld wird durch den Unterschied zwischen kommerzieller und individueller Software bzw. ihrer Nutzung und Weiterführung innerhalb der Legacysysteme aufgebaut. Die kommerzielle Software, s. Kap. 10,

besitzt zwar eine Reihe von vermeintlichen Vorzügen, hat aber auch eine Menge von realen Nachteilen. Der tatsächliche Einsatz von kommerzieller Software anstelle oder innerhalb von Legacysystemen verlangt, da es sich stets um soziotechnologische Systeme handelt, notwendigerweise auch ein gewisses Maß an organisatorischen Anpassungen.

Neben der Frage nach dem Kauf von Software steht Legacysoftware mittlerweile auch beim Out- oder Insourcing, nicht nur in Bezug auf eine Neuentwicklung, sondern auch bezüglich der Maintenance, in der Diskussion. Outsourcing und dabei speziell das Offshoring, s. Kap. 8, hat zum Teil drastische Konsequenzen, welche nicht unbedingt a priori bekannt sind.

Zusätzlich zu den bisher aufgezählten Punkten macht sich der Methodenstreit zwischen den „klassischen“ Vorgehensmodellen, s. Abschn. 11.2, und den agilen Verfahren, s. Abschn. 11.4, auch bei der Legacysoftware bemerkbar. Es ist im Grunde der Streit zwischen einer prozesszentrierten und einer produktzentrierten Sicht.

Die Summe dieser unterschiedlichen Denkweisen und Strömungen hat über die letzten 30 Jahre hinweg deutliche Spuren in den Legacysystemen hinterlassen.



<http://www.springer.com/978-3-540-25412-6>

Legacysoftware

Das lange Leben der Altsysteme

Masak, D.

2006, XII, 434 S., Hardcover

ISBN: 978-3-540-25412-6