Nach dem Boom: Ein Gesundheits-Check für IT-Systeme

Benedikt Stockebrand
<me@benedikt-stockebrand.de>
http://www.benedikt-stockebrand.de/

28. März 2003

Inhaltsverzeichnis

1 Einleitung

In den letzten fünf Jahren hat die IT einige schwerwiegende Veränderungen durchgemacht:

- Das unerwartete Ende eines Jahrtausends hat eine Welle hektischer Aktivität ausgelöst, Tonnen von alter Hardware und Terabytes an Software von einer Sekunde zur anderen unbrauchbar gemacht. Die IT-Projekte in diesem Zusammenhang rangierten in ihrer Komplexität vom einfachen Austausch eines Desktop-PCs bis hin zur Umstellung der gesamten IT von proprietären COBOL-Programmen auf SAP R/3 in vielen Banken.
- Gerade zwei Jahre später löste die Einführung des Euro und die damit verbundene Anforderung nach Mehrwährungsfähigkeit eine ähnliche, wenn auch weniger ausgeprägte, Projektwelle aus.
- Neben diesen spektakulären und vor allem exakt datierten Ereignissen hat die Verbreitung des Internets als neues Kommunikationsmedium die Geschäftswelt dazu gezwungen, die eigenen Prozesse grundlegend zu ändern.

Alleine diese drei Ereignisse haben der auch sonst schon stark wachsenden IT-Industrie zu einem einzigartigen Boom verholfen, der eine Unzahl von selbsternannten „Experten“ auf den Plan rief, die einen schnell verdienten Euro erhofften. Das Ergebnis waren allzu oft IT-Systeme, die den gestellten funktionalen Anforderungen nicht gerecht wurden, schlecht designt und noch schlechter implementiert waren.

Nachdem sich die allgemeine Euphorie gelegt hat, ändert sich jetzt die Situation. Die „Experten“ sind größtenteils auf der Suche nach „neuen Herausforderungen“—was kein nennenswerter Verlust ist. Manager fangen wieder an, über Kosten nachzudenken, manchmal bis hin zu dem Punkt, daß sie darüber den generierten Gewinn vergessen, von dem sie leben[1]. Aber es nützt nichts, die neue Situation zu bejammern; die entstandenen IT-Systeme müssen untersucht und dann wenn möglich kosteneffizienter gemacht — oder auch als hoffnungsloser Fall aufgegeben — werden.

Wann ist ein System „gesund“? Im Zusammenhang dieses Vortrags ist ein System gesund, wenn es die gestellten Anforderungen bei vertretbaren Kosten erfüllt. Die Anforderungen lassen sich grob, unabhängig von der spezifischen Funktionalität des Systems, in Effizienz, Zuverlässigkeit und Zukunftssicherheit — und bei neuen Systemen, rechtzeitige Inbetriebnahme — strukturieren.

In einer perfekten IT-Welt sind alle Anforderungen spezifiziert, analysiert, implementiert und erfolgreich und mit minimalen Kosten umgesetzt worden, wenn ein System in Betrieb genommen wird. In der Realität werden dagegen auch offensichtliche funktionale Anforderungen allzu oft nur ansatzweise erfüllt. Versteckte Anforderungen wie Zuverlässigkeit und Zukunftssicherheit sind dagegen weniger offensichtlich für technisch nicht ausgebildetes Management und Endbenutzer und bleiben deshalb noch öfter unerreicht.

Noch dazu wurde zu Zeiten der „New Economy“ oft genug ohne Rücksicht auf Kosten nach einer möglichst schnell verfügbaren Lösung verlangt. Das berechtigte Ziel, durch kurze „Time to Market“ eine gute Ausgangsposition im Markt aufzubauen, ist inzwischen oft weniger wichtig als Kosteneffizienz und damit Konkurrenzfähigkeit. Ganz allgemein ändern sich Anforderungen im Laufe der Zeit, manchmal so schleichend, daß sie als Änderung nicht wirklich erkannt und angemessen umgesetzt werden.

Zu guter letzt ändert sich die IT-Technologie rasant schnell. Was vor fünf Jahren noch eine erstklassige Lösung war, kann heute unvertretbar teuer, oder wenn eingesetzte Produkte nicht mehr verfügbar sind auch unmöglich, zu warten und zu betreiben sein.

In diesem Vortrag stelle ich eine Methodologie vor, um den Zustand eines IT-Systems zu analysieren. Ein traditioneller Ansatz würde die Anforderungsdefinition überarbeiten und dann mit dem existierenden System vergleichen. Dieser Ansatz hat aber zwei Nachteile: Das natürliche

[1] DeMarco [DeM93] hat dazu aus Sicht eines Software-Entwicklers einige lesenswerte Bemerkungen gemacht.

Desinteresse der IT-fremden „Business User“, in ein „funktionierendes“ System Zeit und Arbeit zu investieren und die rasante Weiterentwicklung der IT, die durch eine Anforderung nicht erfaßt wird. Deshalb schlage ich einen anderen Ansatz vor: Eine Analyse, die den Zustand des Systems erfaßt und in eine Form bringt, mit der man Management und Business User konfrontieren kann zusammen mit der Frage, ob das ihren Vorstellungen entspricht.

Worauf ich nicht eingehen will, ist die Frage, wo genau die Ursachen für die beobachteten Symptome liegen oder wie man einen Plan entwickelt, mit dem das System sich „kurieren“ läßt.

2 Die „System Health“-Matrix

Die Analyse basiert auf einem zweidimensionalen Schema, in dem die Zeilen die Systemkomponenten und die Spalten die Qualitätskriterien enthalten.

Die Zeilen sind weiter hierarchisch strukturiert. Wir unterscheiden zunächst vier Gruppen: Technische Systemkomponenten, Personal, funktionale Subsysteme und das Gesamtsystem.

Die Spalten sind ähnlich in drei Gruppen unterteilt: Die Effizienz im Regelbetrieb, das Risiko von Betriebsstörungen und die Lebenserwartung bzw. Zukunftsperspektive des Systems.

Abbildung 1 (Seite 4) zeigt eine Vorlage für eine solche Matrix.

Der Rest dieses Kapitels behandelt die Frage, wie die Spalten mit den Komponenten des Systems und die Zeilen mit den detaillierten Auswertungskriterien gefüllt werden.

2.1 Zeilen I: Technische Komponenten

Die offensichtlichsten Kandidaten für eine Analyse sind die technischen Systemkomponenten. Um keine relevanten Komponenten zu übersehen, ist es sinnvoll, sie grob in einige Klassen zu unterteilen und diese Klassen dann systematisch abzuarbeiten.

2.1.1 Infrastruktur

Auch wenn die Infrastruktur als „außerhalb“ des Systems betrachtet werden kann, lohnt es sich doch bei der Untersuchung eines einzelnen Systems (statt eines ganzen IT-Umfelds wie einem Rechenzentrum), die relevanten Aspekte des Umfelds zu untersuchen, denn Infrastruktur-Probleme können erhebliche Auswirkungen auf die enthaltenen Systeme haben.

Infrastrukturkomponenten, die einen wesentlichen Einfluß auf die „Gesundheit“ eines IT-Systems haben, sind unter anderem:

Rechenzentrumsfläche: Neben den „eigentlichen“ Kosten für Rechenzentrumsfläche gilt: Sobald ein System produktiv ist, kann ein Umzug sowohl durch den eigentlichen Transport als auch durch die möglicherweise nötige Downtime sehr teuer werden.

Stromversorgung: Stromversorgungen sind aus zwei Gründen wichtig: Wenn die externe Versorgung unzuverlässig ist, nimmt die Bedeutung von USVen und Generatoren, sowohl was die Zuverlässigkeit als auch das Sizing (insbesondere der Treibstofftanks) betrifft, erheblich zu. Außerdem brauchen Rechenzentren immer mehr Energie pro Fläche. Vor fünf Jahren war ein 19”-Schrank mit zwei Zuführungen à 16A gut ausgestattet. Spätestens mit den aufkommenden Blade-Systemen wird offensichtlich, daß die Anforderungen sich hier in den nächsten Jahren leicht verdreifachen können.

USVen und Generatoren: Ähnlich wie die reguläre Stromversorgung hat die Notfallversorgung gelegentlich entscheidenden Einfluß auf die Zuverlässigkeit eines Systems.

Klimatisierung: Unterdimensionierte oder unzuverlässige Klimatisierung hat erheblichen Einfluß auf die Zuverlässigkeit aller installierten Systeme. Im günstigsten Fall führt eine Störung zum zeitweisen Abschalten weniger wichtiger Systeme oder Subsysteme. Im schlimmsten Fall kann es zu Hardwareschäden und möglicherweise Datenverlusten kommen. Mit der erwähnten wachsenden Leistungsdichte in Rechenzentren wachsen auch dementsprechend die Anforderungen an die Klimatisierung.

	Effizienz im Regelbetrieb	**Ausfallbedingte Risiken und Szenarien**	**Langfristige Zukunftsperspektiven**
	Kosten/ Betriebsaufwand	Komponentenausfall Sicherheit Bekannte Probleme... Naheliegende Szenarien... GAU	Lebenserwartung Upgrade-Kosten Skalierbarkeit Verfügbarkeit Neue Funktionalitäten... Sonstiges...
Technische Komponenten			
Infrastruktur			
Rechenzentrumsfläche			
Stromversorgung			
USVen und Generatoren			
Klimatisierung			
Interne Netzwerke			
Internet-Anbindung			
...			
Hardware			
Server			
Storage			
Clients			
...			
Systemsoftware			
Betriebssysteme			
Volume Manager			
Cluster-Software			
...			
Middleware			
Datenbanken			
Backup Client			
Monitoring Agent			
Web Server			
...			
Applikationen			
...			
Personal			
User			
...			
Betriebspersonal			
Operators			
Systemadministratoren			
DBAs			
Externer Support			
Help Desk			
...			
Systementwickler			
Business User			
Software-Architekten			
Programmierer			
Systemarchitekten			
Funktionale Subsysteme			
...			
Gesamtsystem			

Abbildung 1: Die „System Health“-Matrix

Interne Netzwerke: Auch wenn die LAN-Technik inzwischen normalerweise sehr zuverlässig ist, lohnt es sich in manchen Fällen doch, deren Zustand noch einmal zu prüfen, weil unzuverlässige lokale Netze erheblichen Einfluß haben können.

Internet-Anbindung: Weil WAN-Verbindungen einerseits teuer und andererseits vergleichsweise störanfällig sind, spielen sie bei Internet-basierten Systemen oft eine beachtliche Rolle.

Diese Liste kann grundsätzlich weder vollständig noch allgemeingültig sein. Sie sollte aber als Checkliste beitragen, offensichtliche Komponenten nicht zu übersehen.

2.1.2 Hardware

Bei den „eigentlichen“ Systemkomponenten beginnen wir mit der eingesetzten Hardware. Dazu gehören im allgemeinen:

Server: Mit Ausnahme von einigen reinen Peer-to-Peer-Systemen gehören zu praktisch allen IT-Systemen Server.

Storage: Sobald Daten persistent gespeichert werden müssen, sollte Storage als separate Komponente betrachtet werden unabhängig davon, ob es sich um eine einzelne interne Festplatte oder ein dediziertes SAN für das System handelt. Damit vereinfachen wir uns erfahrungsgemäß später die Untersuchung der Systemzuverlässigkeit.

Clients: Auch wenn dedizierte Clients für einzelne Systeme immer mehr durch „multifunktionale“ PCs ersetzt werden, lohnt sich in vielen Fällen ein genauer Blick auf die Client-Hardware.

Wie schon bei den Infrastruktur-Komponenten kann auch diese Liste nur als Anregung dienen.

2.1.3 System-Software

System-Software, also Software, die unmittelbar auf der Hardware arbeitet und eine Hardware-unabhängige Schnittstelle zwischen Hardware und den darüberliegenden Softwareschichten bereitstellt, ist die naheliegende nächste Komponentenkategorie. Dazu gehören unter anderem:

Betriebssystem: Alleine die Tatsache, daß es einen eigenen Beruf „Systemadministrator“ gibt, sollte deutlich machen, daß Betriebssysteme ein entscheidender und komplexer Teil jedes IT-Systems sind und als solche entsprechende Beachtung verdienen.

Volume Manager: In größeren Systemen mit relevanten Mengen persistenter Daten sind Volume Manager wie VxVM/SEVM, SDS, HSM und LSM kritische Komponenten. Die Unterscheidung zwischen Server und Storage bei der Hardware findet hier eine Parallele in der Unterscheidung zwischen Betriebssystem und Volume Manager.

Cluster-Software: Mit dem Einsatz von Clustern wird zumindest theoretisch die Verfügbarkeit eines Systems erhöht, indem auf redundanter Hardware ein virtueller „Server“ aufgebaut wird. Praktisch erhöhen Cluster die Komplexität jedes Systems ganz erheblich, so daß sie auch zu einer deutlich reduzierten Verfügbarkeit führen können.

2.1.4 Middleware

Die naheliegende nächste Kategorie sind Middleware-Komponenten. „Middleware“ ist normalerweise definiert als Software, die anwendungsunabhängige High-Level-Funktionalitäten für die Anwendungen bereitstellt. Die Trennung zwischen Middleware und Applikationen ist etwas unscharf, aber für unsere Zwecke ausreichend — wenigstens solange wir uns zu diesem Thema nicht auf dogmatische Grundsatzdiskussionen einlassen.

Datenbanken: Ganz allgemein sind Datenbanken, insbesondere RDBMS', die bekanntesten Middleware-Komponenten, die einen erheblichen Einfluß auf den Zustand des gesamten Systems haben.

Backup Clients: Auch wenn sie weiter verbreitet als RDBMS' sind, bleiben sie oft unbeachtet — bis jemand dringend Daten restaurieren muß...

Monitoring Agents: Sauberes Monitoring ist wesentlich, um eine hohe Verfügbarkeit zu erreichen und deshalb sind Monitoring Agents in ihrer Bedeutung den Backup Clients eng verwandt.

Webserver: Webserver sind ein typischer Grenzfall zwischen Middleware und Anwendung. In einem webbasierten System, das ausgiebigen Gebrauch von CGIs, Servlets und ähnlichem macht, kann es durchaus sinnvoll sein, den eigentlichen Webserver als Middleware aufzufassen.

2.1.5 Anwendungen

Die meisten IT-Systeme benutzen Anwendungssoftware, die bestimmte Funktionalitäten zur Verfügung stellt. Diese Anwendungen sind naturgemäß sehr wichtig, lassen sich aber nicht mehr wie in den vorhergehenden Kategorien in allgemeiner Form strukturieren. Hier ist ein solides Verständnis des Aufbaus der Anwendungen und zusätzliche Sorgfalt nötig, um diese Komponenten möglichst vollständig und sauber strukturiert zu erfassen.

2.2 Zeilen II: Personal

Auch wenn es hochmodern ist, das Thema „Mitarbeiter“ mit einem „unsere Mitarbeiter sind unser größtes Kapital“ abzutun, betrachten wir alle Personen, die ein System benutzen, als Teil des Systems und untersuchen, wie wichtig sie für den Zustand des Systems sind.

Jeder Systemadministrator weiß, daß es zwei Arten von Menschen gibt: Sysadmins und die niederen Lebensformen. Diese Unterscheidung ist für unsere Zwecke leider unzureichend und wir unterscheiden die niederen Lebensformen weiter in User, Systementwickler und Beinahe-Sysadmins.

Die drei Gruppen, Benutzer, Betriebspersonal und Entwickler, korrelieren bis zu einem gewissen Grad mit den drei großen Evaluationskriterien Effizienz, Zuverlässigkeit und Zukunftssicherheit — wenigstens, wenn das System in einem einigermaßen gesunden Zustand ist.

2.2.1 User

Wenn ein System direkt von Usern benutzt wird, ist es lohnenswert, diese Interaktion genauer zu betrachten.

Wenn die User für das Unternehmen arbeiten, dem auch das System gehört, ist offensichtlich, daß die User als Teil des Systems „funktionieren“. Möglicherweise haben Controlling oder Personalabteilung „Performance“-Daten. Wenn nicht, kann eine Befragung der User und etwas konventioneller Dreisatz über die Anzahl der User und die Transaktionen pro Zeiteinheit weiterhelfen.

Wenn das System nur Kunden als „User“ hat, ist es etwas schwieriger, an vergleichbare Daten heranzukommen. Andererseits sind dann aber Marketing, Kundenbetreuung und oft genug auch das Top Management interessiert genug an diesen Daten, daß sie trotzdem schon routinemäßig ermittelt und beobachtet werden.

2.2.2 Betriebspersonal

Das Betriebspersonal hat eine ähnlich wichtige Aufgabe wie die User, um das „Funktionieren“ eines Systems sicherzustellen. Ihre Aufgabe ist aber umfangreicher: Sie müssen das System im Regelbetrieb betreiben, was schon recht zeit- und arbeitsaufwendig sein kann, sie müssen Betriebsstörungen schnell, zuverlässig und mit möglichst geringem Schaden beheben, sie müssen das

Wissen über das System, seine Probleme und deren Behebung bewahren und sie müssen das System zum erheblichen Teil an sich verändernde Gegebenheiten wie neue Betriebssystemversionen, Security Patches und ähnliches anpassen.

Betriebspersonal kann grob klassifiziert werden in:

Operators: Es gibt auch heute noch genug Routinearbeiten, wie das Auswechseln von Bändern in einer Tape Library oder der Einbau neuer Hardware in 19"-Schränke, um damit eine traditionelle Berufsgruppe zu beschäftigen.

Systemadministratoren: Sie kümmern sich im wesentlichen um Systemsoftware, die meisten Middleware-Komponenten mit Ausnahme von relationalen Datenbanken, und um die Netzwerkkonfiguration.

Datenbankadministratoren: Weil größere Datenbankinstallationen meist sehr schnell sehr komplex werden, gibt es Umgebungen, wo dedizierte DBAs deren Betreuung übernehmen.

Anwendungsadministratoren: Gelegentlich gibt es auch Anwendungen, die so komplex sind, daß speziell geschulte Administratoren sich ausschließlich um diese Anwendungen kümmern.

Externer Support: Sehr spezielle und manchmal auch außergewöhnlich schwierige Probleme machen gelegentlich den Einsatz von externen Spezialisten nötig. Diese Gruppe umfaßt unterschiedlichste Kompetenzen, von der Hardware-Reparatur bis hin zu sehr speziellem Produkt-Know-How.

Help Desk: Der Help Desk stellt die Verbindung zwischen Usern und Technik dar und kann oft die Probleme der User vergleichsweise strukturiert wiedergeben, was bei der Analyse des Systems sehr hilfreich ist.

2.2.3 Systementwickler

Als letzte Gruppe bleiben die Systementwickler, die das System ursprünglich entwickelt haben. In vielen Fällen, insbesondere in den letzten fünf bis sieben Jahren, waren das oft Externe „Leiharbeiter", die nur für diese Aufgabe herangezogen wurden und nach Abschluß des initialen Projekts höchstens noch in Ausnahmefällen zur Verfügung stehen.

Systementwickler lassen sich grob klassifizieren:

Business User: Auch als Nicht-Techniker haben Business User eine wesentliche Rolle in der Entwicklung von IT-Systemen, weil sie detaillierte Kenntnis der Aufgabe des zu bauenden Systems haben. Deshalb sind sie insbesondere verantwortlich für das Anforderungsmanagement.

Software-Architekten: Die Anforderungen der Business User werden von Software-Architekten analysiert und in einem Design umgesetzt.

Programmierer: Das Design wird von den Programmierern ausprogrammiert.

Systemarchitekten: Um die fertige Software benutzen zu können, müssen Infrastruktur, Hardware, Systemsoftware und Middleware geplant werden. Weil das in den letzten Jahrzehnten eine immer komplexere Aufgabe geworden ist, werden dazu gelegentlich spezialisierte Systemarchitekten herangezogen. Die eigentliche Installation, vom Hardware-Einbau bis zur Konfiguration wird aber in den meisten Fällen vom Betriebspersonal durchgeführt.

Diese Klassifikation ist erheblich vereinfacht und bezieht sich auf ein weit verbreitetes Projektverständnis, das allerdings trotz seines Verbreitung nicht unbedingt durch eine universelle Tauglichkeit glänzt.

2.3 Zeilen III: Funktionale Subsysteme

Komplexe IT-Systeme stellen im allgemeinen eine Unzahl verschiedener Funktionalitäten zur Verfügung. Einige dieser Funktionalitäten sind wichtiger oder komplexer oder auch zuverlässiger als andere. Es hilft oft, das System in „funktionale Subsysteme“ zu unterteilen, also Subsysteme, die eine bestimmte Funktionalität oder zusammenhängende Gruppe von Funktionalitäten bereitstellen.

Es ist gelegentlich nichttrivial, solche Subsysteme zu definieren. In jedem Fall erfordert es einiges Verständnis der „Fachseite“ und damit die Einbindung der Business User. Sobald die Subsysteme einmal definiert sind, ist es oft sehr einfach, aus der Analyse der relevanten Systemkomponenten und untergeordneten Subsysteme eine Einschätzung des funktionalen Subsystems abzuleiten.

Warum sind diese funktionalen Subsysteme so wichtig? Um einem technisch unterqualifizierten Management den Zustand eines Systems klarzumachen, muß man die wirtschaftliche Bedeutung von Problemen herausarbeiten. Das wird genau durch diese Subsysteme möglich; eine Aussage „mit der momentanen Backup-Lösung ist völlig unklar, ob und wenn ja wie und in welchem Zeitraum bei einem Totalverlust der Plattensysteme z.B. durch einen Wasser- oder Brandschaden noch Daten restauriert werden können, ohne daß eine vollständige Neueingabe von existierenden Belegen nötig wird“ ist für einen Geschäftsführer eher nachvollziehbar als „ich habe schon einige Datensicherungen mit tar und gzip auf ein DAT DDS-2 gesehen, aber noch keine, die funktioniert hat“.

2.4 Zeilen IV: Das Gesamtsystem

Schließlich lassen sich alle funktionalen Subsysteme in einer Aussage über das globale System, in gewisser Weise als „Management Abstract“, zusammenfassen.

Genau wie mit den einseitigen „Management Abstracts“ an anderer Stelle hat eine Zusammenfassung nur dann Sinn, wenn das System nicht zu komplex und der Zustand der Subkomponenten insgesamt recht einheitlich ist. Bei sehr komplexen Systemen, deren Zustand eine gewisse Streubreite überschreitet, kann eine Überversimplifizierung nur kontraproduktiv sein.

2.5 Die Matrix-Spalten: Untersuchungskriterien

Nachdem wir festgelegt haben, *was* wir untersuchen wollen, müssen wir die Kriterien festlegen, die uns interessieren.

Wie im Kapitel 2 schon angedeutet, gibt es drei grundlegende und in ihrer Natur sehr unterschiedliche Kriterien, die wir in den nächsten Kapiteln nacheinander betrachten.

Die drei Kriterien, Effizienz im Regelbetrieb, Zuverlässigkeit und Anpassbarkeit, entsprechen den wirtschaftlichen Anforderungen von Profitabilität, geringem Risiko und langfristiger Produktivität.

Ein viertes Kriterium, das aus wirtschaftlicher Sicht für IT-Systeme relevant ist, können wir bei bereits produktiven Systemen ignorieren: Die Inbetriebnahme ist bereits erfolgt, wir können sie nicht mehr rückwirkend vorverlegen.

2.6 Spalten I: Effizienz im Regelbetrieb

Effizienz im Regelbetrieb ist eine erste Annäherung an die kurzfristige Wirtschaftlichkeit eines IT-Systems, zumindest wenn Betriebsstörungen tatsächlich eher Ausnahme als Normalfall sind. Traditionell wird die Effizienz in „Bangs per Buck“ gemessen, wobei „Bangs“ in Abhängigkeit von der bereitgestellten Funktionalität und „Bucks“ in einer frei wählbaren, harten Währung gemessen werden.

Auf den ersten Blick scheint die Effizienz von technischen Komponenten einfach zu messen zu sein; für die Bucks fragt man das Controlling und für die Bangs wahlweise Personalabteilung,

Marketing oder die Geschäftsführung. Tatsächlich kann es aber durchaus mit Aufwand verbunden sein, die unterschiedlichen Arten von Bangs zu definieren; ein Blick auf die funktionalen Subsysteme kann hier hilfreich sein. Für jede Art von Bang, oder Transaktion, oder Geschäftsvorfall oder was auch immer, sollte eine eigene Spalte angelegt werden.

Ähnlich sieht es aus, was die Effizienz aller Personen angeht, die am Regelbetrieb beteiligt sind. Im schlimmsten Fall muß man hier messen, wie lange ein User für einen Bang braucht, den Aufwand für Routineaufgaben des Betriebspersonals ermitteln und das den Personalkosten gegenüberstellen. Systementwickler sollten an dieser Stelle keine Bedeutung haben; wenn doch, ist das ein deutliches Zeichen, daß das System massive Probleme hat.

Die funktionalen Subsysteme und das Gesamtsystem lassen sich mit etwas elementarer Schularithmetik aus den soweit ermittelten Zahlen zusammensetzen und sollten niemand mit durchschnittlichen Rechenfähigkeiten vor ein unlösbares Problem stellen.

Soweit ist die Effizienzanalyse wenig spannende Fleißarbeit, die mit etwas Glück das Controlling schon für uns erledigt hat. Aber was sagen uns diese Zahlen? Gar nichts, bis wir andere, „sinnvolle“ oder „angemessene“ Zahlen haben, gegen die wir sie vergleichen können, denn sonst wird man in den meisten Fällen vom Management etwas in Richtung „das ist zu teuer, da müssen wir die Kosten senken“ hören. Und das wiederum ist nur die psychologisch korrekte Formulierung für „ihr Techies senkt jetzt gefälligst die Kosten und dann beschweren wir uns weiter“...

Aus diesem Problem gibt es einen Ausweg, den man, völlig überraschend, aus der theoretischen Informatik abkupfern kann. Die Komplexitätstheorie beschäftigt sich damit, für algorithmische Probleme *untere Schranken* zu beweisen, die von keinem (korrekten) Algorithmus unterschritten werden können. Alle Algorithmen, die dieser unteren Schranke einigermaßen nahe kommen, gelten als „gut“.

Wir können so ähnlich vorgehen: Wir überlegen unabhängig vom existierenden System, wie effizient ein imaginäres, für unser Verständnis gut gemachtes System aussehen könnte und berechnen daraus Referenzwerte zum Vergleich. Wenn das existierende System diese Referenzwerte annähernd erreicht, ist es gesund.

Dieser Ansatz bringt einige Herausforderungen mit sich: Es ist entscheidend wichtig, sich gedanklich von dem existierenden System zu lösen und es verlangt Erfahrung, Phantasie und möglicherweise auch einige „Laborversuche“, um wirklich sinnvolle Referenzwerte zu ermitteln.

Außerdem ist „annähernd“ ein durchaus dehnbarer Begriff.

2.7 Spalten II: Zuverlässigkeit und Risiken von Systemausfällen

Bei einem perfekten System wäre die Frage nach der momentanen Effizienz des Systems schon beantwortet. In der Realität müssen wir aber die bisher gefundenen Zahlen noch um die gelegentlich auftretenden Störungen und Probleme korrigieren.

Risikoanalyse und Risikomanagement sind komplexe Themen. Für unsere Zwecke beschränken wir uns darauf, daß ein Risiko das Produkt aus der Eintrittswahrscheinlichkeit pro Zeiteinheit und dem bei Eintritt entstehenden Schaden ist. Bis auf die unmittelbaren technischen Wiederherstellungskosten muß der Schaden vom Business User ermittelt werden. Wir können aber mögliche Ausfall- und Wiederherstellungszenarien aufstellen und ihre Eintrittswahrscheinlichkeit abschätzen, so daß wir alle nötigen Informationen zu einer Risikoabschätzung zusammentragen können.

Prinzipiell kann man zwei Arten von Störungen unterscheiden: Produktionsrelevante und nicht produktionsrelevante Störungen. Während nicht produktionsrelevante Störungen, wie ein Festplattenausfall in einem RAID-Subsystem, den Betrieb aus Sicht der User nicht berühren, haben produktionsrelevante Störungen Auswirkungen auf die User und damit auf den wirtschaftlichen Nutzen des Systems. Diese Trennung ist im Grenzbereich durchaus unscharf: Ein an seiner Leistungsgrenze arbeitendes RAID5 wird durch den Performance-Einbruch bei einem Plattenausfall durchaus für die User zu Einschränkungen führen, während ein Reboot eines primären DNS-Servers, auch wenn dadurch DNS-Updates unmöglich werden, in den meisten Fällen von den Usern nicht bemerkt wird.

Damit stellt sich die Frage, ob die nicht produktionsrelevanten Störungen vielleicht zusammen mit dem Regelbetrieb untersucht werden sollten. Unabhängig von Steuerrecht und Controllingmethododologie stehe ich persönlich auf dem Standpunkt, daß auch die nicht produktionsrelevanten Störungen ungeplant auftreten, das Betriebspersonal sehr wohl stören und damit eine Untersuchung als Störung verdienen. Wer schon einmal unerwartet nachts aus dem Bett geklingelt wurde, wird diese Entscheidung vermutlich nachvollziehen können.

Die Spalten in dieser Gruppe sind deutlich differenzierter als in der Spaltengruppe zur Effizienz. Wir betrachten den Ausfall von Einzelkomponenten und Personal, Ausfälle durch Angriffe, Probleme, die in der Vergangenheit aufgetreten sind, Szenarien, die uns wahrscheinlich oder anderweitig relevant erscheinen und schließlich als „Universalfall“ den GAU[2].

2.7.1 Ausfälle von Einzelkomponenten und Personen

Vermutlich die naheliegendste Form von Ausfall ist der Ausfall von Hardwarekomponenten. Weil auch der Ausfall von Personen nach dem gleichen Schema analysiert werden kann, betrachten wir beide zusammen.

Mit allen Komponenten und Personen ist es zum erheblichen Teil Erfahrungssache, um abzuschätzen, wie wahrscheinlich ein Ausfall ist. Anders als beim Initialprojekt, das das System aufgebaut hat, können wir hier die Erfahrungen nutzen, die wir bisher im Betrieb gemacht haben.

Um eine Person oder Einzelkompoente zu beurteilen, stellen wir drei Fragen:

1. *Wie* kann die Komponente ausfallen?
2. Was hat ein solcher Ausfall für Konsequenzen?
3. Wie wahrscheinlich ist ein solcher Ausfall?

Die erste Frage ist normalerweise am schwierigsten zu beantworten, sobald wir erst die einzelnen Festplatten in einem RAID-Verbund abgehakt haben. Es setzt sowohl Erfahrung als auch Phantasie voraus, um eine schlüssige Liste von Fehlertypen zu erstellen. Bei komplexeren Komponenten müssen wir die Zusammenhänge zwischen den Komponenten verstehen und bei der Untersuchung berücksichtigen.

Die zweite Frage, nach den Konsequenzen, kann in einem komplexen System mühsam zu beantworten sein. Um methodologisch die Trennung zwischen technischer und funktional-wirtschaftlicher Ebene beizubehalten, betrachten wir zunächst nur die „unmittelbaren“ technischen Auswirkungen eines Ausfalls. Aber auch das kann ein komplexes Thema werden: Eine aus ungeklärten Gründen gecrashte Datenbank mit dazugehörigem Datenverlust läßt sich nicht alleine mit Routineprozessen wiederbeleben. Wesentlich ist in diesem Zusammenhang, nicht nur Kausalketten der Form „wenn A ausfällt, verliert B Daten und C schaltet sich selbst ab“ zu betrachten, sondern auch den Ablauf bis zur Problembehebung. Besser wäre also: „Wenn A ausfällt, erfährt das innerhalb von fünf Minuten der Sysadmin B, entweder durch entsprechendes Monitoring oder durch einen User. Er identifiziert das Problem in maximal fünfzehn Minuten und fordert den Lieferanten C mit dem Austausch innerhalb von vier Stunden gemäß Wartungsvertrag und den Backup-Admin D zum anschließenden Recovery mit Laufzeit sechs Stunden an, so daß es zu einem Ausfall insgesamt von zehn Stunden zwanzig Minuten mit Wiederherstellungskosten um € 2000,00 kommt.“ Um den Aufwand nicht beliebig zu steigern, sollten hier möglichst viele Probleme systematisch zusammengefaßt werden — weder das Monitoring noch der Wartungsvertrag mit dem Hardware-Lieferanten noch das Recovery müssen jedesmal neu aufgeschlüsselt werden. So realistische Fälle wie „keiner weiß was zu tun ist, wenn A ausfällt, weil sich eh keiner damit auskennt und sich niemand darum kümmert[3]“ sollte man auch so dokumentieren. Möglichst in rot, wenn man einen Farbdrucker zur Verfügung hat.

[2] Wer's nicht mehr weiß: Größter anzunehmender Unfall.

[3] Natürlich haben Keiner und Niemand beide letztes Jahr gekündigt...

Die letzte Frage, nach der Wahrscheinlichkeit eines Ausfalls, läßt sich meistens mit Daten aus einem Problem-Tracking-System, Logbüchern oder einfach aus der Erfahrung des Betriebspersonals ermitteln.

Auch wenn es sehr mechanistisch klingt, lassen sich Personen sehr ähnlich wie technische Komponenten untersuchen. Auch sie altern und fallen irgendwann aus, unabhängig davon ob sie in Rente gehen, frustriert kündigen oder aus gesundheitlichen Gründen zeitweise nicht zur Verfügung stehen.

Anders als technische Komponenten sind aber die Menschen, die mit einem System arbeiten, in der Lage, zu lernen. Dadurch wird das System insgesamt „lernfähig“, kann sich anpassen und selbst verbessern. Auch wenn es sich sehr mechanistisch anhört: Der Verlust jeder „menschlichen Komponente“, die alleine über bestimmtes Wissen über einen Teilaspekt des Systems verfügt, ist eine partielle Amnesie des Gesamtsystems. Unzureichende „Wissensreplikation“ ist besonders problematisch, weil ein „Desaster Recovery“, also die Suche, Einstellung und Einarbeitung eines Nachfolgers, teuer ist, Zeit kostet und letztlich auch immer nur begrenzt erfolgreich sein kann.

2.7.2 Ausfälle von technischen und funktionalen Subsystemen und dem Gesamtsystem

Die Untersuchung von Subsystemen ist ähnlich der von technischen Komponenten, auch wenn sich die Probleme etwas verschieben.

Die drei Fragen, nach Ausfallmodalitäten, Konsequenzen und Ausfallwahrscheinlichkeit, bleiben erhalten, die Antworten sehen aber etwas anders aus. Subsysteme fallen nicht selbst aus, aber aus dem Systemaufbau folgt, welche Ausfälle von Einzelkomponenten zu einem Ausfall des Subsystems führen. Die Fragen nach den Ausfallmodalitäten und der Ausfallwahrscheinlichkeit sind also mit etwas Systemkenntnis und Wahrscheinlichkeitsrechnung einfach zu beantworten.

Alleine die Frage nach den Konsequenzen eines Ausfalls läßt sich bei den funktionalen Subsystemen und dem Gesamtsystem nicht mehr auf technischer Ebene beantworten; hier sind Business User und Management gefordert. Alles, was eine technische Analyse liefern kann, sind Ausfallszenarien und -wahrscheinlichkeiten.

2.7.3 Sicherheit

Das Thema Sicherheit sollte unsere Zielsetzung höchstens am Rande interessieren, weil sicherheitskritische Systeme vor der Inbetriebnahme, nach größeren Änderungen und generell in regelmäßigen Abständen einem dedizierten Security Audit unterzogen werden sollten.

Soviel zur Theorie.

Statt nun nebenher einen vollständigen Audit durchzuführen, beschränken wir uns darauf, unsere drei Fragen noch einmal abzuwandeln und kommen so zu sechs Fragen zur Sicherheit jeder technischen Komponente, Person, jedes Subsystems und des Gesamtsystems:

1. Wie sicherheitskritisch ist die Komponente?
2. Was für sicherheitsrelevante Ereignisse bezüglich dieser Komponente sind möglich oder denkbar?
3. Wie wahrscheinlich ist jedes solche Ereignis?
4. Wie erkenne ich jedes Ereignis?
5. Welche Gegenmaßnahmen sind für jedes Ereignis geplant?
6. Was für Konsequenzen haben die Ereignisse und die dazugehörigen Gegenmaßnahmen?

Wir unterscheiden grundsätzlich zwei Arten von sicherheitsrelevanten Ereignissen: Die Entdeckung von Schwachstellen in Systemkomponenten und Angriffe auf das System. Für beide Arten interessiert uns die sicherheitstechnische Qualität der Komponente, ihre Sichtbarkeit, die Informationswege bei sicherheitsrelevanten Ereignissen, Herstellersupport und die Prozesse zur Handhabung des Ereignisses. Bei der Untersuchung von Angriffen und Angriffsmöglichkeiten erhalten wir weiter auch Einblicke in Architekturfehler, wie unzureichende Sicherheitsschichten, und die wirtschaftlichen Auswirkungen eines Angriffs.

Unbeantwortete Fragen in den sicherheitsbezogenen Spalten deuten normalerweise auf erhebliche latente Probleme hin und sollten deshalb deutlich als solche kenntlich gemacht werden.

2.7.4 Bekannte Probleme und naheliegende Szenarien

Weil wir ein existierendes und bereits in Betrieb genommenes System untersuchen, können wir auf die schon gesammelten Erfahrungen zurückgreifen und Probleme betrachten, die schon aufgetreten sind.

Dazu interviewen wir Personen, die mit dem System schon gearbeitet haben und analysieren wenn vorhanden Logbücher, Daten aus Problem-Ticket-Systemen und vergleichbaren Ressourcen. Für jedes relevante Problem legen wir eine eigene Spalte an, um darin die Erfahrungen, aufgeschlüsselt nach den betroffenen Komponenten, zu dokumentieren und bewerten.

Ähnlich verfahren wir mit „naheliegenden“ Szenarien, die zwar (noch) nicht eingetreten sind, aber trotzdem aller Erfahrung nach in der Zukunft eintreten können und deren Schaden so hoch ist, daß das angenommene Risiko diesen Aufwand rechtfertigt.

2.7.5 Der „Größte Anzunehmende Unfall“

Ein besonderes Szenario, das berücksichtigt werden sollte, ist der GAU, der als letzter Ausweg bei allen wirklich gravierenden Problemen herhalten kann, die nicht explizit berücksichtigt worden sind.

Alle Szenarien, die als Super-GAU auch noch den GAU in ihren Auswirkungen übertreffen, sind damit per Definition „Out of Scope“ einer technischen Lösung. Entweder wird dieses „Restrisiko“ bewußt in Kauf genommen oder es sollte z.B. nach einer geeigneten Versicherung gesucht werden. Wo der GAU angesetzt wird, hängt letztlich von der Bedeutung des Systems und den zur Verfügung stehenden Ressourcen ab. Eine Bank wird möglicherweise auf den Komplettausfall eines ganzen Rechenzentrums durch einen Großbrand mit der kurzfristigen Verlagerung des Betriebs in ein vorbereitetes Ausfallrechenzentrum reagieren können. Ein Diplomand wird sich dagegen nicht unbedingt einen zweiten, baugleichen PC zulegen, um auf einen Totalausfall seines Rechners durch eine strategisch umgekippte Tasse Kaffee reagieren zu können.

2.8 Spalten III: Flexibilität und langfristige Perspektive

Bis zu dieser Stelle haben wir uns nur mit dem momentanen Zustand des System beschäftigt und damit die aktuelle Wirtschaftlichkeit des Systems dokumentiert. Das ist wichtig, reicht aber zu einer Einschätzung des Systems nicht aus.

Ein System, das spezielle Hardware oder eine bestimmte Betriebssystemversion voraussetzt, ist vergleichsweise kurzlebig. Weil in praktisch alle Systeme vor der Inbetriebnahme ein erhebliches Startkapital investiert wurde, ist es normalerweise wichtig, das System so lange und kostengünstig wie möglich in Betrieb zu halten. Dummerweise wird das bei vielen Projekten übersehen.

Was noch unangenehmer ist, sind sich ständig ändernde Anforderungen. Vor zwanzig Jahren konnte eine Bank noch jede Nacht ein langes Service-Zeitfenster nutzen. Heute erwarten Kunden, daß ihr Online-Banking rund um die Uhr funktioniert. Hardware wird billiger, so daß Funktionen, die vor Jahren noch wirtschaftlich undenkbar waren, heute — bei der Konkurrenz — Standard

sind und unbedingt nachgerüstet werden müssen. Die Internet-Ableger der „Old Economy“ wachsen teilweise noch immer rasant weiter, so daß auch deren Systeme schneller wachsen als neue Hardware-Generationen mit dem Leistungsbedarf nachkommen.

Wir müssen also noch die Zukunftsaussichten eines IT-Systems berücksichtigen, wenn wir seinen Zustand richtig erfassen wollen. Neben unseren bisherigen Hauptwerkzeugen, Erfahrung und Phantasie, ist dazu auch einiges an Spekulation nötig. Wir unterscheiden grob zwei Teilaspekte: Die Lebenserwartung des Systems mit seinen gegebenen Funktionalitäten und den Umgang mit geänderten Anforderungen und neuen Funktionalitäten.

2.8.1 Lebensverlängernde Maßnahmen

In der Matrix richten wir in unserer letzten Spaltengruppe zuerst eine Matrix „Lebenserwartung“ ein, in der wir die Lebenserwartung aller Komponenten eintragen. Für Hardware ist hier entscheidend, wie lange noch Service-Verträge und Ersatzteile zur Verfügung stehen. Abhängigkeiten, wie die Voraussetzung von bestimmter Hardware für eingesetzte Software, wird hier ebenfalls dokumentiert. Bei Personen ist die Bezeichnung „Lebenserwartung“ umzuinterpretieren: Entscheidend ist hier, wie lange die Personen noch für das System zur Verfügung stehen, bevor das Rentenalter, die frustrationsbedingte Kündigung, der Fulltime-Einsatz in einem neuen Projekt oder auch das Ende der Vertragslaufzeit eintritt. Für die Subsysteme werden wie gewohnt die Daten aus denen der konstituierenden Komponenten ermittelt.

Erfahrungsgemäß kann man mit dieser Spalte Manager noch besser erschrecken als mit der Ankündigung einer außerordentlichen Wirtschaftsprüfung.

In der nächsten Spalte, „Upgrade-Kosten“, dokumentieren wir, wie weit und mit welchem Aufwand die Lebenserwartung des Systems verbessert werden kann. Dazu wird zunächst versucht, die problematischsten Komponenten zu finden, die die künstlich verlängerte Lebenserwartung nach oben beschränken. Bei allen anderen Komponenten muß dann nur nach Möglichkeiten gesucht werden, auch diese Lebenserwartung zu erreichen. Wesentlich ist, daß diese Aufstellung dem Management als Entscheidungsgrundlage helfen muß, um den Zeitpunkt der Ablösung durch ein Nachfolgesystem oder größere Renovierungsarbeiten zu planen.

Ein gesundes System sollte offensichtlich eine vernünftige Lebenserwartung haben, die sich durch vertretbare Upgrades bei Bedarf noch etwas verlängern läßt.

2.8.2 Veränderte Anforderungen

Zu guter letzt: Anforderungen ändern sich. Für jede Anforderung, von der abzusehen ist, daß sie sich voraussichtlich ändern wird, sollte eine eigene Spalte eingerichtet werden, in der für alle Komponenten dokumentiert wird, wie und zu welchen Kosten die Anforderung umgesetzt werden kann.

Die üblichen Verdächtigen in diesem Zusammenhang sind:

Skalierbarkeit: Unternehmen wachsen. Die Internet-Sparte vieler Unternehmen wächst überproportional schnell. Deshalb ist Skalierbarkeit oft der wichtigste und trotzdem gerne übersehene Kandidat für eine eigene Spalte.

Verfügbarkeit: Die Anforderungen an die Verfügbarkeit von Systemen sind im allgemeinen streng monoton steigend und konvergieren geometrisch gegen 100%. Designs, die das nicht berücksichtigen und zum Beispiel jede Nacht eine mehrstündige Offline-Phase brauchen, sollten das in einer Spalte für alle betroffenen Komponenten dokumentieren.

Neue Funktionalitäten: Wenn schon abzusehen ist, daß neue Funktionalitäten in der nächsten Zeit gefordert oder nötig sind, sollten für sie Spalten eingerichtet werden, in denen die Machbarkeit und wenigstens grob der erwartete Aufwand festgehalten werden.

Sonstige: Andere Anforderungen, wie geänderte externe Schnittstellen, rechtliche Vorgaben oder ähnliches, können und sollten genauso festgehalten werden.

3 Das Ausfüllen der Matrix

3.1 Tools

Es gibt eine ganze Reihe nützlicher Tools, die die Erstellung und Pflege der Matrix erleichtern.

Vermutlich das naheliegendste Tool ist eine Tabellenkalkulation. Viele Werte in der Matrix lassen sich weitgehend automatisch berechnen und eine Tabellenkalkulation wird auch der Größe der Matrix am ehesten gerecht. Auch wenn es nicht sinnvoll erscheint, die gesamte Analyse in eine einzige gigantische Tabelle zu stopfen, hilft eine zentrale Tabelle mit Verweisen auf separate Einzeldokumente für die umfangreicheren Zellen, um den Überblick zu behalten. Wer nicht auf gelegentliche imposante Ausdrucke verzichten will, hat vielleicht Zugriff auf einen Tintenstrahl-Plotter.

Persönlich setze ich gerne ein einfaches Time Tracking Tool, wie titrax (auch für den Palm) oder gtimer ein, um schnell und ohne großen Aufwand zu erfassen, wie viel Zeit ich für welche Aktivitäten (oder Kunden) aufbringe.

Monitoring- und Problem-Ticket-Systeme sind, wenn sie schon eingesetzt werden, oft ein essentielles Hilfsmittel, um Informationen über die Vergangenheit des Systems zu recherchieren. In kleineren Umgebungen finden sich ähnliche Informationen oft in klassischen Logbüchern, meist in der traditionellen A4-Kladde, wie sie Betriebspersonal immer wieder gerne einsetzt.

3.2 Praktische Überlegungen

Nachdem wir die Zeilen und Spalten für unser System sinnvoll definiert haben, stehen wir vor einer großen, leeren Matrix, die verdächtig nach viel Arbeit aussieht, so daß sich die Frage stellt: Ist der Aufwand, der sich abzeichnet, überhaupt ansatzweise gerechtfertigt?

In einem perfekten Projekt wären alle Punkte, die wir mit unserer Matrix untersuchen, schon berücksichtigt worden, so daß eine Dokumentation, die inhaltlich der Matrix entspricht, schon existiert. In realen Projekten sind aber viele der Punkte, die wir in der Matrix untersuchen, im allgemeinen unbeachtet geblieben, so daß die Matrix schon alleine deshalb eine Existenzberechtigung hat, weil sie liegengebliebene Arbeiten aus dem Projekt nachholt.

In einem ISO 9000-artigen Umfeld wäre es denkbar, die gesamte Matrix vollständig und systematisch zu bearbeiten. Ich persönlich habe noch kein Umfeld erlebt, wo Standards im Stil von ISO 9000 auch tatsächlich konsequent im Tagesgeschäft gelebt werden, deshalb kann ich über den Nutzen der Matrix in einem derartigen Umfeld nichts sagen.

Aber wie sieht es in den „durchschnittlichen“ IT-Umgebungen aus, die in einem Hauruckverfahren à la Yourdon [You97] oder Roberts und Roberts [RR00] ohne Rücksicht auf Qualität aus dem Boden gestampft wurden, zu einem Erfolg erklärt wurden weil niemand das sinnlos verblasene Geld verantworten wollte, anschließend über Jahre immer weiter verkommen sind, weil niemand sich mit der nötigen Kompetenz und Entscheidungsfreiheit um ihre Pflege gekümmert hat und die so vernachlässigt worden sind, daß heute niemand mehr weiß, was bei Problemen zu tun ist, weil der letzte Sysadmin, der das dazu nötige Wissen im Kopf hatte, nach einer Auseinandersetzung mit der Geschäftsführung gegangen ist?

In solchen Umgebungen kann es helfen, mit der Einführung einer Matrix das Ausmaß der Situation zu überblicken, selbst wenn die Matrix weitgehend unvollständig ist. Noch wichtiger ist, daß die regelmäßige Pflege einer Matrix hilft, Probleme in ihrer ganzen Tragweite zu erkennen. Ein Lieferant, der bankrott geht oder den Support für einige seiner Produkte einstellt, Veränderungen in der personellen Besetzung, Änderungen an der eingesetzten Hardware und Software können alle erhebliche versteckte Auswirkungen haben, die durch die Matrix aufgedeckt werden können. Das soll nicht heißen, daß die Matrix ständig auf dem aktuellen Stand gehalten werden muß. Im Zweifelsfall aktualisiert man sie vor einem Meeting mit dem Management und stellt dabei noch eine Liste wichtiger Veränderungen gegenüber dem Stand des letzten Meetings auf.

Das schöne an der Matrix ist, daß sie auch in einer sehr rudimentären Form schon hilft, die eigenen Probleme in ihrer gesamten Tragweite zu erfassen und, soweit Lösungen durch Detailver-

besserungen möglich sind, die Arbeit am System sinnvoll zu priorisieren. Und zu guter letzt: Die Matrix hilft, einem Business User oder technisch nicht vorgebildetem Management die Situation zu verdeutlichen.

3.3 Was ist mit den offenen Fragen?

Was ist mit den Zellen, zu denen es keine sinnvollen Informationen gibt?

Betrachten wir noch einmal das Beispiel Backup: Weil es nie getestet worden ist, haben wir keine Erfahrungswerte, wie lange ein Desaster Recovery wohl dauern würde, wenn ein essentielles Plattensubsystem, also bevorzugt die Bootplatte eines Servers, defekt wäre. Wir wissen nicht einmal, wie es geht und was dabei für Probleme auftreten können.

Offensichtlich sind wir da über ein anstehendes Desaster gestolpert. Aber wo findet sich das in der Matrix wieder?

Die einfache Antwort ist: In der Zelle für die Bootplatte, in der Spalte „Komponentenausfall“, notieren wir ein dickes rotes Fragezeichen. Damit sind aber auch alle funktionalen Subsysteme, die auf dem Server laufen, von diesem Problem betroffen und erben das Fragezeichen, bis hin zum Gesamtsystem, und genauso alle anderen Komponenten, die bei einem Ausfall auf ein Restore oder ein Desaster Recovery angewiesen sind. Alle Verfahren, die nicht getestet, ausreichend dokumentiert oder allgemein bekannt sind, sollten entsprechend kenntlich gemacht sein. Spätestens wenn in der GAU-Spalte Fragezeichen auftauchen, sollte allen Beteiligten bewußt werden, daß da ein latentes Desaster auf seinen Einsatz wartet.

3.4 Beispiele

Nach all der trockenen Theorie wird es höchste Zeit für einige Geschichten aus dem täglichen IT-Leben und die Frage, wo man sie in der Matrix wiederfindet.

- Als ich zuletzt bei meiner bevorzugten Internet-Buchhandlung Bücher bestellt habe, waren die sechs Seiten, auf denen in einzelnen Schritten die Bestellung abgewickelt wurde, zu einer Seite reduziert, die die aktuellen Defaults anzeigte und für jeden Abschnitt einen Link auf eine weitere Seite hatten, wo diese Defaults geändert werden konnten. Das hat mir zwei oder drei Minuten Zeit gespart und vermutlich auch die Webserver und dahinterliegende Datenbanken entlastet.

  Wo zeigt sich der Vorteil dieser Änderung in der Matrix?

  In der ersten Spalte, zur Effizienz im Regelbetrieb, sollte das sowohl bei der Performance der Hardware als auch bei den Usern sichtbar werden, selbst wenn es in der Umsatzentwicklung von anderen Ereignissen überdeckt wird.

- Ein DNS-System hat einen Primary und fünf Secondaries. Wie bei einem durchschnittlichen DNS üblich führt ein Wartungsintervall des Primary dazu, daß Updates nicht möglich sind.

  Wie erkenne ich, ob das akzeptabel ist?

  Aus der Matrix: gar nicht, denn das ist keine technische Entscheidung, sondern eine wirtschaftliche. Was wir von der Matrix wissen, ist daß wir in der Vergangenheit eine Verfügbarkeit des Primary von 99.9% hatten, also ca. acht Stunden Ausfall pro Jahr, und zusätzlich noch zwei zweistündige geplante Wartungsfenster pro Jahr. Wenn sich das in den funktionalen Subsystemen überhaupt auswirkt, könne wir mit diesen Aussagen von einem Management eine sinnvolle Antwort auf die Frage bekommen, ob das System so akzeptabel ist oder eine bessere, aber auch deutlich aufwendigere Lösung gewünscht ist.

- Eine qualitativ minderwertige Textverarbeitung wird unternehmensweit eingesetzt. Die Software ist dafür bekannt, daß sie bei Dokumenten mit mehr als 100 Seiten durchschnittlich alle halbe Stunde abstürzt und dabei in 10% aller Fälle nicht nur die bisher ungespeicherten Änderungen verliert, sondern auch die bereits gespeicherte Datei zerstört.

Wo zeigt sich das?

Wenn das Problem tatsächlich auftritt, also auch Dokumente in dieser Größenordnung mit der Software bearbeitet werden, sollte es als „bekanntes Problem“ in einer eigenen Spalte erfaßt werden, in der der Arbeitsaufwand zum Wiederherstellen des verlorenen Dokuments und die Wahrscheinlichkeit des Auftretens dieses Phänomens soweit aufbereitet werden, daß spätestens die Personalabteilung daraus den entstehenden Schaden berechnen kann.

- Wo ist der unzureichend geschulte, überarbeitete Sysadmin?

  Die mangelnde Schulung zeigt sich durch die ganze Matrix: Es wird externe Verstärkung für Routineaufgaben eingesetzt, weil der Sysadmin dazu nicht qualifiziert ist; auch bei gängigen Ausfallszenarien ist externe Unterstützung nötig.

  Der überarbeitete Sysadmin ist etwas schwieriger wiederzufinden, weil er im allgemeinen mehrere Systeme betreut, so daß seine gesamte Arbeitsbelastung nicht in einer einzelnen Matrix zu erkennen ist. Aber insbesondere bei den bekannten Problemen und der Erfahrung, wie sie in der Vergangenheit gelöst wurden, zeigt sich die unzureichende Verfügbarkeit des Sysadmins. Und zu guter letzt zeigt sich an dem Arbeitsaufwand für den Regelbetrieb oft, ob das System den Sysadmin mit unnötigen Aufgaben belastet — ein sinnvoll aufgesetztes Monitoring liest Logfiles deutlich schneller als ein Sysadmin.

  Generell gilt, daß Personalprobleme dieser Art sich nach einer Weile in der „Lebenserwartung“ des Sysadmins bezogen auf das System zeigen: Wenn schon die drei letzten Sysadmins innerhalb eines halben Jahres das Handtuch geworfen haben, sollte das Problem hier offensichtlich werden.

- Ein System wurde ursprünglich von einem externen Unternehmen aufgebaut. Leider ist das Unternehmen inzwischen pleite. Was bedeutet das?

  Die langfristige Zukunftsperspektive sollte voller Fragezeichen oder „nicht möglich“ für Upgrades oder neue Funktionen sein. Wenn das Unternehmen auch benötigten Support bereitgestellt hat, sollte sich das in den Ausfallszenarien deutlich zeigen. Falls auch noch ein Help Desk von dem Unternehmen in Anspruch genommen wurde, ist die Zeile „Help Desk“ durch alle Spalten deutlich rot unterlegt.

  Ähnlich wird auch deutlich, was passiert, wenn ein „99% vollständiges“ System ausgeliefert wurde und der externe Lieferant anschließend alle kompetenten Entwickler in andere Projekte zu anderen Kunden schickt.

- Wenn ein Security Patch installiert werden muß, dauert es drei Wochen, bis alle dazu nötigen Unterschriften vorliegen.

  Unter „Sicherheit“ sollte es in diesem Fall eine Spalte für diese Situation geben, in der die Auswirkungen für die betroffenen funktionalen Systeme, also „drei Wochen unkontrollierte Angriffsgefahr“ dokumentiert ist. Zusammen mit einer entsprechenden Abschätzung, wie wahrscheinlich damit ein erfolgreicher Angriff auf das System wird und welcher Schaden dabei entstehen kann, sollte das auch Nichttechnikern die Augen öffnen.

4 Wie geht’s weiter?

Nachdem wir die Situation unseres Systems in einer Matrix erfaßt haben, folgt die Frage, wie es an dieser Stelle weitergehen soll.

Das hängt natürlich davon ab, was die Matrix ans Licht gebracht hat. Es kann sein, daß das System in einem so hoffnungslosen Zustand ist, daß die einzig sinnvolle Reaktion ist, sich „neuen Herausforderungen zuzuwenden“. Es kann auch sein, daß eine Vielzahl kleinerer und mittlerer Probleme darauf wartet, nach und nach in Angriff genommen zu werden. Dabei hilft die Matrix als Orientierungshilfe bei der Priorisierung.

Es kann auch sein, daß ein Krisengespräch mit dem Management nötig wird. Dann ist es hilfreich, sich bei der Diskussion auf die funktionalen Subsysteme und das Gesamtsystem zu beschränken; der wirtschaftliche Schaden entsteht hier, nicht in der Technik. Im schlimmsten Fall passiert anschließend nichts, aber wenn es zur Katastrophe kommt, kann man sich einige Vorwürfe ersparen. Was natürlich am Ende auch niemand hilft.

Vielleicht übernimmt an dieser Stelle auch das Management die weitere Initiative und benutzt die Matrix, um Risikomanagement, Ressourcen- und Finanzplanung zu verbessern.

In vielen Fällen kann die Matrix helfen, die Kommunikation zwischen Technik und Management zu verbessern und damit die Möglichkeit zu schaffen, zügig und ohne größere Störungen ein System so zu verbessern, daß es wirtschaftlich konkurrenzfähig wird.

Und irgendwann, in einer fernen Zukunft, wird es vielleicht soweit kommen, daß auch ein Management seine IT-Systeme ganz selbstverständlich als hochgradig dynamische Gebilde in einem genauso dynamischen Umfeld verstehen lernt, die man nicht mit einem Projekt aus dem Boden stampft, um sie anschließend als „erledigt und abgehakt“ zu vergessen, bis die Probleme allmählich wieder so schwerwiegend werden, daß ein neues Projekt nötig wird.

Irgendwann vielleicht.

Nachwort

Dies ist das Manuskript zum Abschlußvortrag des Frühjahrsfachgesprächs 2003 der German Unix Users Group, den ich am 28. März 2003 in Bochum gehalten habe. Besonderer Dank geht an Wolfgang Sachs und Jochen Topf, die mir die Gelegenheit gaben, meine Ideen an so exponierter Stelle zu präsentieren.

Für Anregungen und Kommentare zum vorgestellten Thema bin ich immer dankbar. Am einfachsten zu erreichen bin ich per E-Mail unter der Adresse `me@benedikt-stockebrand.de`. Auf meiner Homepage, `http://www.benedikt-stockebrand.de`, habe ich noch einige andere Ressourcen zu den Themen IT-Reengineering, Systemarchitektur und Projektmanagement zusammengetragen.

Literatur

[DeM93] Tom DeMarco. Why does software cost so much? In *[DeM95]*, chapter 1. Dorset House, 1993.

[DeM95] Tom DeMarco. *Why Does Software Cost So Much?* Dorset House, 1995.

[RR00] Sharon Marsh Roberts and Ken Roberts. Do I want to take this crunch project? In *[WBK00]*, pages 25–42. Dorset House, 2000.

[WBK00] Gerald M. Weinberg, James Bach, and Naomi Karten, editors. *Amplifying Your Effectiveness*. Dorset House, 2000.

[You97] Edward Yourdon. *Death March—The Complete Software Developer's Guide to Surviving "Mission Impossible" Projects*. Prentice Hall, 1997.