Thema:

Modell zur Ablösung von IT-Systemen

Diplomarbeit

Arbeitsgruppe Wirtschaftsinformatik

Themensteller: Prof. Dr. rer. pol. habil. Hans-Knud Arndt

Betreuer: Prof. Dr. rer. pol. habil. Hans-Knud Arndt

Vorgelegt von: Sebastian Pohl

Abgabetermin: 01.12.07

Inhaltsverzeichnis

Verzeichnis der Abkürzungen und Akronyme

ANSI	American National Standard Institute
bspw.	beispielsweise
DBMS	Datenbankmanagementsystem
DIN	Deutsches Institut für Normungen e.V.
o. O.	ohne Ort
PO	Projektorganisation
SEP	Softwareentwicklungsprozess
SQL	Structured Query Language

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Problemstellung

In jedem großen Unternehmen findet sich eine Vielzahl von IT-Systemen, die oftmals auf die Belange bestimmter Nutzergruppen hin optimiert wurden. Vor dem Hintergrund des immensen Kostendrucks, sowie der zunehmenden Prozessorientierung lässt sich jedoch nicht selten feststellen, dass eine Ablösung oder Integration einzelner IT-Applikationen sinnvoll ist, da zwar lokale und funktionale Bedürfnisse sehr gut befriedigt werden, übergreifende Prozesse jedoch nicht berücksichtigt worden sind und Einsparpotentiale nicht realisiert werden.

Wie aber lassen sich verschiedene derartige Systeme effizient ersetzen, ohne den Erfolg einer Systemablösung zu gefährden? In diesem Zusammenhang fehlt eine umfassende Systematik, die an einer Vielzahl praktischer Situationen angewandt werden kann.

Die vorliegende Diplomarbeit orientiert sich an einer Reihe operativer Situationen aus einem Großunternehmen der Automobilindustrie, um konkrete Handlungsanweisungen in Form eines Vorgehensmodells zu entwickeln, mit deren Hilfe die oben aufgeworfenen Fragen in der Praxis gelöst werden können.

1.2 Unternehmen

Der Volkswagen Konzern, 1937 gegründet, ist der größte Automobilhersteller Europas und einer der führenden Automobilproduzenten weltweit. Zusammen mit den absatzstarken Marken VW, Audi, Seat, Škoda und den Luxusmarken Bentley, Lamborghini und Bugatti bietet der Konzern ein breites Sortiment von Produkten und Marken. Zusätzlich zum PKW-Geschäft bietet die Sparte Volkswagen Nutzfahrzeuge eine Reihe von Kleinbussen und Transporter in allen Größen bis hin zu schweren Lastkraftwagen an, die jedoch nicht alle in Europa vertrieben werden (Vgl. Volkswagen AG (2007a)).

Die Modellvielfalt und Leistungsfähigkeit des Unternehmens zeigt sich in den Zahlen der 44 Produktionsstätten rund um den Globus. Zusammen fertigen dort nahezu 325.000 Beschäftigte 129 verschiedene Modelle. Täglich laufen mehr als 24.500 Fahrzeuge vom Band, die daraufhin an Kunden in mehr als 150 Ländern ausgeliefert werden (Vgl. Volkswagen AG (2007b)).

Der Volkswagen Konzern hat seine langfristige Planung in der ‚Konzernstrategie 2015' festgeschrieben, denn nur so können kurz- und mittelfristige Entscheidungen sinnvoll getroffen werden, da verlässliche Ziele für einen weiten Zeitraum verankert sind. Im Focus stehen dabei Erfolgs- und Ertragsgenerierung. Transparente und konzernweit einheitliche Prozesse sollen zu kürzeren Entwicklungs-, Auftrags- und Durchlaufzeiten führen und eine effizientere Auslastung der Werke gewährleisten, sowie die Effektivität des Vertriebsnetzes erhöhen. Zudem soll die Kundenzufriedenheit, Qualität und Liefertreue nachhaltig gesteigert werden, um eine weltweite Führungsposition zu erlangen (Vgl. Volkswagen AG (2007c)).

1.3 Aufbau der Diplomarbeit

Für die Lösung der oben genannten Problemstellung werden zunächst einige wesentliche Begriffe im Umfeld der Systemablösung bestimmt. Sie sollen ein einheitliches Verständnis ermöglichen und komplexe Sachverhalte einfacher beschreiben lassen.

In Kapitel 3 werden Systemablösungsprojekte der Volkswagen AG beschrieben und analysiert, die zum größten Teil bereits beendet sind. Dabei werden verschiedene Einflüsse betrachtet und die Projekte auf ein einheitliches Vorgehen untersucht.

Im darauf folgenden Kapitel werden Gründe identifiziert, die zu einer Systemablösung führen. Die Auswirkung auf Systemablösungen wird untersucht und eine Klassifizierung der Ursachen vorgenommen.

Mögliche Strategien zur Ablösung von Systemen werden in Kapitel 5 dargelegt. Dabei wird geprüft, in wie weit sich die Strategien auf die Problemstellung der Systemablösung anwenden lassen.

Der zentrale Punkt dieser Diplomarbeit, die Entwicklung eines Vorgehensmodells zur Systemablösung, wird in Kapitel 6 erläutert. Dazu werden Hilfsmittel aus dem Projektmanagement auf technische und organisatorische Vorgänge einer Systemablösung angewendet, um konkrete Handlungsanweisungen in strukturierter Form festzuhalten.

2 Begriffsbestimmungen

2.1 IT-Projekt

Ein Projekt ist gekennzeichnet durch seine Einmaligkeit in der Durchführung, seiner zeitlichen Begrenztheit und seiner Komplexität. Ein durchdachtes Projektmanagement ist der Schlüssel zu einem erfolgreichen Abschluss eines Projektes (Vgl. Steinbuch (2000), S. 24).

Das Projektmanagement umfasst die Gesamtheit aller Führungsaufgaben, -organisation, -techniken und -mittel für die Abwicklung eines Projektes. Mit Hilfe des Projektstrukturplans erfolgt eine Aufgliederung des Projektes in Teilaufgaben, die wiederum in Arbeitspakete zerlegt werden können. Der Projektstrukturplan kann für die Darstellung des Projektes nach Aufbau und Ablauf verwendet werden. Der zeitliche Ablauf lässt sich dabei in Abschnitte, so genannte Phasen unterteilen, die eine sachliche Trennung gegenüber anderen Teilen des Projektablaufs bilden. Die Projektleitung ist die für die Dauer des Projektes geschaffene Organisationseinheit, die für die Planung, Steuerung und Überwachung des Projektes verantwortlich ist (Vgl. DIN 66901 (1987), S. 1 ff.).

Die Planung eines Projektes wird in folgende Kategorien unterteilt (Vgl. Steinbuch (2000), S. 31 f.):

- Aufgabenplanung
- Personalplanung
- Terminplanung
- Sachmittelplanung
- Kostenplanung
- Dokumentationsplanung
- Qualitätsplanung
- Berichtsplanung.

Die Projektsteuerung dient laut STEINBUCH dazu, die Abwicklung des Projektes gemäß dem Projektauftrag und Plan zu realisieren und zu sichern. Sie teilt sich in folgende Aufgaben (Vgl. Steinbuch (2000), S. 255):

- Sicherung der Einhaltung des Projektplanes
- Mitarbeiterführung und Mitarbeitereinsatz
- Sach- und Projektmittelbereitstellung
- Projekt- und Mitarbeiterkoordination
- Information und Kontaktpflege.

Die Projektüberwachung dient dazu, den planmäßigen Projektfortschritt laufend zu kontrollieren. In regelmäßigen Abständen werden Daten erhoben, die Rückschlüsse auf die aktuelle Projektsituation ermöglichen. Dabei sind alle quantifizierbaren Größen einzubeziehen, insbesondere Zeit-, Aufwands- und Kostengrößen. Um Abweichungen zu erkennen werden diese Ist-Werte den zu Projektbeginn definierten Plan-Werten gegenübergestellt. Werden Abweichungen festgestellt muss die Projektsteuerung entsprechende Maßnahmen einleiten, um den Projektrahmen einzuhalten (Vgl. Zimmermann et al. (2006), S. 106 f.).

IT-Projekte befassen sich mit Informations- und Kommunikationssystemen, beispielsweise im Rahmen von Softwareentwicklung oder Migrationsvorhaben und bilden eine temporäre Organisationsform innerhalb eines Unternehmens (Vgl. Wieczorrek/Mertens (2007), S. 9).

2.2 Systemablösung

Die Systemablösung ist ein IT-Projekt im Rahmen des Change Managements im Unternehmen, das den Übergang zwischen Alt- und Neusystem darstellt. Der Übergang ist dabei ein kritischer Vorgang, der in der Regel eine Datenübernahme vom Alt- zum Neusystem, die so genannte Datenmigration, enthält (Vgl. Siedersleben (2003), S. 62 ff.).

Bei der Systemablösung kann sowohl Standard- als auch Individualsoftware zum Einsatz kommen. Während Individualsoftware für einen bestimmten Anwendungsfall entwickelt wurde, ist Standardsoftware zur Lösung abstrakter Anwendungsfälle konzipiert. Bei Standardsoftware ist es häufig erforderlich, Anpassungen, so genanntes Customizing, vorzunehmen. Damit wird der Vorgang der Parametrisierung, Konfiguration und Programmierung individueller Erweiterungen bezeichnet. Individualsoftware ist selbst entwickelt oder fremdbezogen und bietet die Möglichkeit Funktionalität direkt auf den bestimmten Einsatzzweck anzupassen, wohingegen

Standardsoftware für spezielle Problemstellungen nur einen unzureichenden Funktionsumfang bietet (Vgl. Stahlknecht/Hasenkamp (2006), S. 173; Mertens et al. (2001), S. 29 ff.).

2.3 Datenbank

Im Folgenden soll der Begriff der Datenbank genauer bestimmt werden, da Begrifflichkeiten und Verständnis während der Systemablösung und der Datenmigration vorausgesetzt werden.

MERTENS definiert eine Datenbank als eine Sammlung von Daten, die logisch zusammengehören. Die Datenbank ist nach MERTENS ein Bestandteil eines Datenbanksystems, welches neben der Datenbank eine Software zur Verwaltung der Daten, dem so genannten Datenbankmanagementsystem (DBMS) beinhaltet. Das DBMS stellt für den Betrieb des Datenbanksystems eine Datendefinitions- oder -beschreibungssprache, eine Datenmanipulationssprache und eine Speicherbeschreibungssprache zur Verfügung (Vgl. Mertens et al. (2001), S. 59 ff.).

Nach der Definition von STEINER ist eine Datenbank eine selbstständige und auf Dauer ausgelegte Datenorganisation, die den Datenbestand flexibel und sicher verwalten kann. Die Datenbank besteht im Kern aus einem DBMS, das alle notwendigen Systemroutinen für Datenbankfunktionen für die gesamte Datenverwaltung wie Suchen, Lesen und Schreiben enthält. Eine Datenbanksprache bildet die Schnittstelle zwischen Benutzer und dem DBMS (Vgl. Steiner (2006), S. 5 ff.).

Ein Datenbanksystem muss Eigenschaft der Datenintegrität gewährleisten. Datenintegrität bedeutet, dass gespeicherte Daten vollständig, korrekt, aktuell, der Realität entsprechend und widerspruchsfrei, also konsistent sein müssen. Ferner ist die Sicherheit der Daten zu gewährleisten, um nach einem Störfall den korrekten Zustand der Datenbank ohne Datenverlust wiederherstellen zu können. Die Daten sind zudem vor unberechtigtem Zugriff zu schützen. Dabei kann unterschieden werden, welche Nutzer das Recht haben Daten zu lesen, zu schreiben oder zu löschen. Weitere Anforderungen an ein Datenbanksystem sind die Datenunabhängigkeit gegenüber Anwendungen, Systemplattformen, Hardware und der physischen Datenorganisation. Darüber hinaus sollte jedes Datenelement nur einmal in der Datenbank vorhanden sein, um die Redundanzfreiheit der Daten zu gewährleisten (Vgl. Stahlknecht/Hasenkamp (2006), S. 128 f.).

Zur Bearbeitung der Daten stehen die Datenbankoperationen Lesen, Schreiben, Verändern und Löschen zur Verfügung. Diese Operationen werden in Form einer Transaktion ausgeführt, die in der Regel mehrere nacheinander auszuführende Operationen enthält. Um einen konsistenten Zustand des Datenbestandes zu gewährleisten, unterliegen Transaktionen den Bedingungen des ACID-Prinzips:

- Atomarität
- Konsistenz
- Isolation
- Persistenz.

Die atomare Eigenschaft einer Transaktion ist dadurch gegeben, dass sie entweder ganz oder gar nicht ausführbar ist. Eine Speicherung der Daten darf nur bei vollständig ausgeführter Transaktion erfolgen. Bricht eine Transaktion vorher ab, müssen Änderungen rückgängig gemacht werden. Die Konsistenz besagt, dass die Datenbank nach der Ausführung einer Transaktion von einem konsistenten Zustand in einen wiederum konsistenten Zustand überführt werden muss. Transaktionen laufen zudem isoliert voneinander ab, um ein gegenseitiges Überschreiben des Datenbestandes zu vermeiden. Die Persistenz bzw. Dauerhaftigkeit besagt, dass sämtliche gespeicherten Änderungen nachfolgende Fehler und Störungen überleben (Fink et al. (2005), S. 155 f.).

Die ‚Structured Query Language' (SQL) ist eine standardisierte Datenbanksprache, die das American National Standard Institute (ANSI), dem Gegenstück zum Deutschen Institut für Normung e.V. (DIN), herausgegeben hat. SQL besteht aus vier Elementen, die den Aufgabenbereich der Datenbanksprache abdecken. Die ‚Data Definition Language' wird verwendet, um die Datenstruktur aufzubauen. Mit Hilfe der ‚Data Manipulation Language' können Daten in Form von Datensätzen geschrieben, geändert oder gelöscht werden. Eine Abfrage von Daten nach frei wählbaren Kriterien ist mit der ‚Data Retrieval Language' möglich. Aufgabe der ‚Data Security Language' ist es, die gespeicherten Daten vor einem unberechtigtem Zugriff zu schützen (Vgl. Steiner (2006), S. 6 f.).

Um den Datenbestand effizient zu verwalten, ist eine Strukturierung der Daten notwendig. Dazu wird zunächst der Aufbau des Datenbestandes betrachtet. Der Datenbestand enthält Datenobjekte, die auch Datensatz, Tupel oder Entität genannt werden. Eine Entität entspricht dabei einem zu bestimmenden Objekt materieller oder immaterieller Art. Jede Entität ist einem Entitätstyp, auch als Objekttyp bezeichnet,

zugeordnet und bildet eine konkrete Ausprägung dessen. Zu einem Entitätstyp gehören Attribute, die verschiedene Eigenschaften des Entitätstyps darstellen. Eine Entität enthält spezifische Ausprägungen der Attribute, so genannte Attributwerte. Für Attributwerte können Wertebereiche definiert werden (Vgl. Stahlknecht/Hasenkamp (2006), S. 118 ff.).

Beispielsweise sind die Modelle ‚Passat' oder ‚Golf' von Volkswagen als Entitäten zu betrachten. Sie sind vom Entitätstyp ‚Modell', dessen Attribute z. B. Maße, Leergewicht und Preis lauten. Die Entitäten ‚Passat' und ‚Golf' weisen entsprechende Ausprägungen der genannten Attribute auf.

Entitätstypen können in Beziehung zueinander stehen. An diesem Punkt haben sich seit Beginn der Datenverwaltung in Form von Datenbanksystemen mehrere so genannte Datenbankmodelle entwickelt, die sich durch die Strukturierung und die Beziehung der Entitätstypen zueinander unterscheiden:

- Hierarchisches Datenbankmodell
- Netzwerkartiges Datenbankmodell
- Relationales Datenbankmodell
- Objektorientiertes Datenbankmodell.

In einem *hierarchischen Datenbankmodell* werden Beziehungen zwischen Entitätstypen in Form einer Baumstruktur, auch als Eltern-Kind-Beziehung bekannt, dargestellt. Bis auf den obersten, der so genannten Wurzel, hat jeder Entitätstyp genau einen Vorgänger und eine nicht begrenzte Anzahl von Nachfolgern. Mit der Baumstruktur lassen sich 1:1- und 1:n-Beziehungen, nicht jedoch n:m-Beziehungen zwischen Entitätstypen darstellen. Bei 1:1-Beziehungen weist eine Entität eines Entitätstyps ein Bezug zu genau einer Entität eines anderen Entitätstyps auf. Bei 1:n-Beziehungen steht eine Entität mit beliebig vielen Entitäten eines anderen Entitätstyps in Beziehung. Entsprechend bezeichnet eine n:m-Beziehung Relationen zwischen beliebig vielen Entitäten zweier Entitätstypen (Vgl. Stahlknecht/Hasenkamp (2006), S. 119 f.).

Der oben genannte Entitätstyp ‚Modell' könnte beispielsweise die Wurzel eines hierarchischen Datenbankmodells sein. Ein Modell ist mit verschiedenen Motoren und in verschiedenen Farben lieferbar. Die Motoren wiederum können mit verschiedenen Getrieben kombiniert werden. Eine mögliche Modellierung ist in Abbildung 2.1 dargestellt.

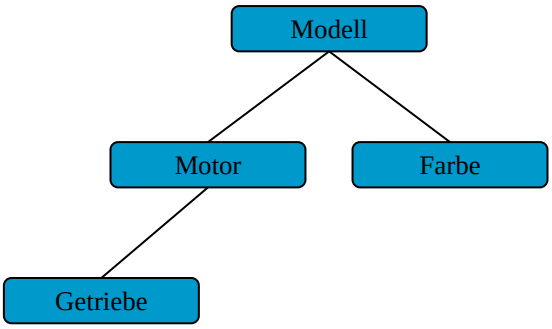


Abb. 2.1: Hierarchisches Datenbankmodell

Das *netzwerkartige Datenbankmodell* basiert auf dem hierarchischen Datenbankmodell und erweitert es dahingehend, dass jeder Entitätstyp nicht nur ein, sondern mehrere Vorgänger haben kann. Dadurch lassen sich auch n:m-Beziehungen zwischen Entitätstypen modellieren (Vgl. Stahlknecht/Hasenkamp (2006), S. 120).

Das Modell aus Abbildung 2.1 könnte dahingehend erweitert werden, dass der Entitätstyp ‚Marke' eingefügt wird, der eine Beziehung zu Modell, Motor und Getriebe aufweist, da sie alle markenabhängig sind.

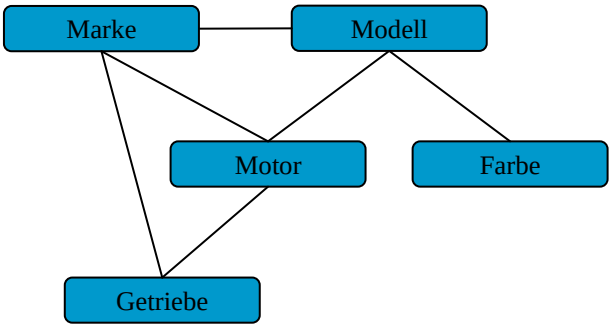


Abb. 2.2: Netzwerkartiges Datenbankmodell

Das *relationale Datenbankmodell* geht auf Edgar Frank Codd zurück, der dies bereits 1970 publizierte. Die Grundidee dahinter ist, Entitäten nicht hierarchisch, sondern mittels Relationen und Schlüssel zu verwalten. Relationen sind Tabellen, die alle Entitäten eines Entitätstyps, die so genannte Entitätsmenge, enthalten. Die Attribute werden als Spalten dargestellt, in den Zeilen finden sich die einzelnen Tupel. Jedes Tupel einer Entitätsmenge muss eindeutig identifizierbar sein. Dazu wird in jeder Relation ein Primärschlüssel definiert, der entweder aus einem geeigneten Attribut gebildet, aus einer Kombination von Attributen zusammengesetzt oder durch einen Index generiert wird. Wird der Primärschlüssel der einen Relation als Attribut in einer anderen Relation verwendet, so wird er dort als Fremdschlüssel bezeichnet. Ein Fremdschlüssel bildet eine Referenz zwischen Relationen. Mit dem relationalen

Datenbankmodell lassen sich 1:1-, 1:n- und m:n-Beziehungen darstellen (Vgl. Steiner (2006), S. 11 ff.; Fink et al. (2005), S. 156).

In Abbildung 2.3 sind Primärschlüssel farbig hinterlegt. Das Attribut ‚Marke' ist in der Relation Marke als Primärschlüssel und in der Relation Modell als Fremdschlüssel definiert. Daraus geht beispielsweise hervor, dass das Modell Golf der Marke Volkswagen zugeordnet wird. Die Relation Motorvarianten enthält einen zusammengesetzten Primärschlüssel aus Primärschlüsseln der Relationen Modell und Motor. Die Relation Motorvarianten zeigt, welche Modelle mit welchen Motoren geliefert werden können. Beispielsweise kann das Modell Golf sowohl mit einem 80 KW, als auch mit einem 180 KW starken Motor geliefert werden.

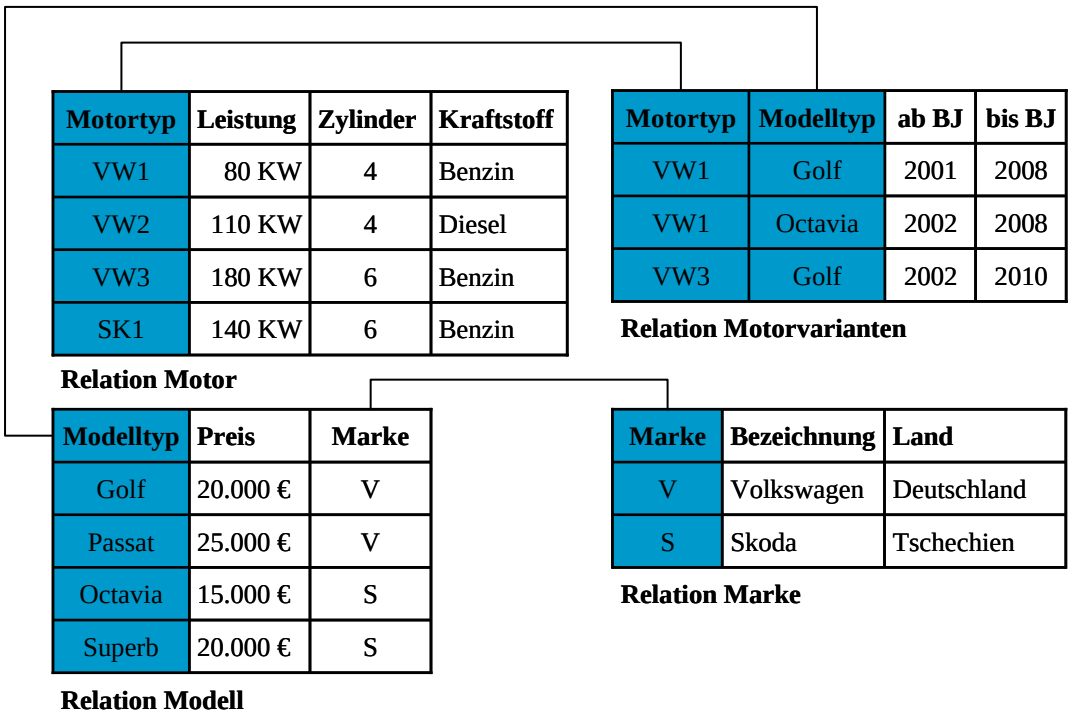


Abb. 2.3: Relationales Datenbankmodell

Aus der objektorientierten Programmierung ist das so genannte *objektorientierte Datenbankmodell* entstanden. Im Gegensatz zu den oben genannten Datenbankmodellen werden Datensätze in Form von Objekten gespeichert, die neben den Attributen Methoden zur Bearbeitung der Tupel und Objektbeziehungen enthalten. Es gilt, wie in der objektorientierten Programmierung das Vererbungsprinzip, d. h. Attribute und Methoden sind vererbbar (Vgl. Stahlknecht/Hasenkamp (2006), S. 169 f.).

Die Abbildung 2.4 zeigt, dass die Objekte ‚Benzinmotor' und ‚Dieselmotor' die Attribute des Objekts ‚Motor' erben. Damit können für einen Benzinmotor die Attribute Zylinder, Hubraum, Leistung und Zündkerzentyp verwendet werden. Das Objekt

‚Modell' enthält neben den Attributen Marke, Preis und Maße eine Objektbeziehung ‚Motorvarianten' und eine Methode ‚Preis ändern'.

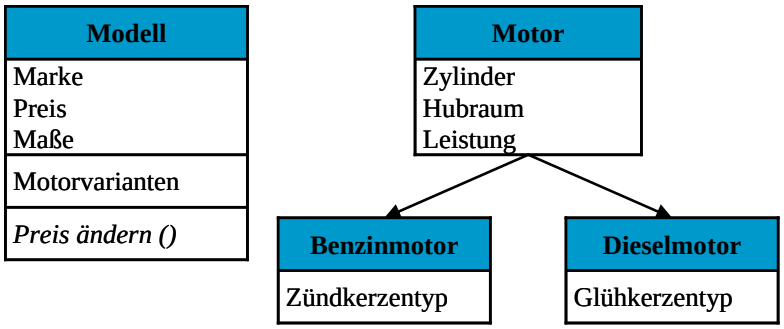


Abb. 2.4: Objektorientiertes Datenbankmodell

Neben den genannten Datenbankmodellen existiert eine Vielzahl von Mischformen, auf die an der Stelle nicht weiter eingegangen wird.

3 Vorgehensweise im Volkswagen Konzern

3.1 Erfassung

In den vergangenen Jahren wurde bereits eine Reihe von IT-Applikationen durch neue Systeme ersetzt. In diesem Kapitel werden verschiedene Projekte vorgestellt und analysiert, um die bisherige Verfahrenweise zu dokumentieren und auszuwerten. Die Informationen dazu stammen zum einen aus Interviews mit Personen, die leitende Funktionen in den Projekten wahrgenommen haben, zum anderen aus Dokumentationen oder Schulungsunterlagen der einzelnen Systeme. Die Darstellung der Projekte erfolgt in anonymisierter Form.

3.1.1 Projekt A

In der Qualitätssicherung der Volkswagen AG wurde lange Zeit das System A eingesetzt. Es bestand aus einer MS Access Applikation und einer Reihe von MS Excel Formularen. Das Einsatzgebiet beschränkte sich auf einzelne Standorte der Tochtergesellschaften. Der Konzern entschied sich, die Anbindung der Qualitätssicherung auf weitere Standorte auszuweiten, um eine bessere Datenqualität zu erreichen und größere Datenmengen auswerten zu können. Jedoch führte dieses Vorgehen in erster Linie dazu, dass das System A an seine Kapazitäts- und Belastungsgrenzen gebracht wurde. So wurden Nachlässigkeiten bei der Entwicklung des Systems aufdeckt, die eine Systemablösung unumgänglich machten.

Dazu wurde das Projekt A gestartet, das in zwei Hauptphasen unterteilt werden konnte. In Abbildung 3.1 ist zu sehen, dass zunächst das neue System entwickelt und in einem zweiten Schritt in die weltweiten Standorte des Volkswagen Konzerns eingeführt wurde.

Abb. 3.1: Projekt A im Überblick

Zur Entwicklung der Software stellt Volkswagen verschiedene Vorgehensweisen in Form eines Softwareentwicklungsprozesses (SEP) zur Verfügung. Darin sind detaillierte Konzepte enthalten, die Empfehlungen für Phasenmodelle,

Projektmanagement und Qualitätssicherung geben. Des Weiteren finden sich Vorlagen für Lasten- und Pflichtenhefte.

Mit dem Projekt A zielt der VW Konzern auf die Nutzung von Synergieeffekten durch weltweiten Datenaustausch der Qualitätssicherungen der Werke aller Marken. Dazu wurde in Phase 1 ein performantes webbasiertes System mit Hilfe des SEP entwickelt, das sich durch Stabilität, Sicherheit und Systemverfügbarkeit auszeichnet. Bereitgestellte Daten müssen weltweit verfügbar und stets aktuell sein.

Die Phase 2 des Projektes gestaltete sich durchaus schwieriger. Die Vorgehensweise war unklar und Zuständigkeiten fraglich. Gab es für Phase 1 noch eine Vorlage aus dem SEP, fehlte für die zweite Phase eine strukturierte Projektplanung. Die Abbildung 3.2 zeigt einen Überblick über den Verlauf der zweiten Phase.

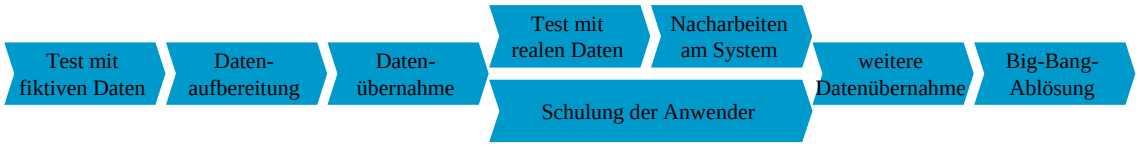


Abb. 3.2: Phase 2 im Überblick

Auf Seiten des Fachbereichs wurde im gesamten Projektverlauf nicht genügend Ressourcen bereit gestellt, da unklar war, welche Aufgaben im Verantwortungsbereich des Fachbereichs lagen. Auf Seiten der IT hingegen wurden Ressourcen nicht genutzt, da Input vom Fachbereich benötigt wurde, der auf Grund fehlender Aufgabenzuweisung keinen Input lieferte. Es ging dabei beispielsweise um die Definition von Funktionsabläufen, die für die abschließende Testphase benötigt wurden. Der Fachbereich sah die Verantwortung bei der IT und umgekehrt sah sich die IT nicht im Stande die für den Geschäftsprozess erforderliche Funktionalität zu benennen. Die fehlende klar definierte Verantwortlichkeit führte dazu, dass weder der Fachbereich noch der IT-Bereich sich mit dem besagten Thema beschäftigte. Letztendlich war dies ein Kommunikations- und Projektführungsproblem, das aber auf fehlender Definition von Strukturen und Verantwortlichkeiten beruhte. Zeitweise führte diese Situation zu einem zähen Vorankommen des Projektes und am Ende zu einer Verzögerung des Ganzen.

Für eine Datenübernahme war eine Aufbereitung der vorhandenen Datensätze unumgänglich, da teilweise verwaiste Einträge vorhanden waren, die keine weitere Bearbeitung erwarten ließen. Somit wurden diese Datensätze von der Datenübernahme ausgeschlossen. Des Weiteren erforderte das neue System aufgrund von Statusüberlegungen im Prozess Pflichtfelder im Datenmodell, die im Altsystem teilweise nicht gefüllt waren. An der Stelle wurde versucht, Einträge zu rekonstruieren

und zu vervollständigen. Konnte jedoch kein akzeptabler Wert gefunden werden, so wurde entweder ein Dummywert für diese Fälle definiert oder die betreffenden Datensätze nicht übernommen.

Bei ersten Tests des Systems A* im Fachbereich traten keine bedeutenden Fehler auf. Jedoch konnte dort nur mit fiktiven Werten getestet werden, da reale Werte nicht zur Verfügung standen. Bei weiteren Tests nach der Datenübernahme musste jedoch festgestellt werden, dass an einigen Stellen Nacharbeiten erforderlich waren. Der Projektverlauf verspätete sich dadurch, da zu diesem Zeitpunkt bereits die Systemeinführung geplant war. Da in der Zeit der Nacharbeiten mit System A weiter gearbeitet und damit der Datenbestand geändert wurde, war eine weitere Datenübernahme erforderlich.

Die Schulung der Anwender begann bereits kurz nach der Datenübernahme. Der Zeitaufwand für die Schulung war in diesem Fall relativ gering und hatte durch die parallel laufenden abschließenden Tests keine Auswirkungen auf den Projektverlauf. Zugleich war die Schulung eine Art Benutzerakzeptanztest, wenn auch die Möglichkeiten der Einflussnahme zu diesem Zeitpunkt eher als gering einzustufen waren.

Nach erfolgreicher Integration in die Systemlandschaft und abgeschlossener Testphase wurde System A* schrittweise nach der ‚Big-Bang-Methode' eingeführt, d.h. die Ablösung von System A erfolgte an den Standorten der einzelnen Tochtergesellschaften durch ein Umschalten auf das neue System A*. Einen Parallelbetrieb beider Systeme an einem Standort gab es damit nicht. Das Projekt ist bereits insofern abgeschlossen, dass ein Großteil der Tochtergesellschaften bereits mit A* arbeitet, jedoch wird A* noch an weiteren globalen Standorten eingeführt.

3.1.2 Projekt B

Im Gewährleistungsbereich kommt im Volkswagen Konzern das System B* zum Einsatz. B* löste das fast 20 Jahre alte System B ab, das auf Basis der HOST-Technologie entwickelt wurde. Auslöser für die Neuentwicklung des Systems war die Forderung nach einem einheitlichen Gewährleistungsprozess über alle Konzernmarken und Märkte weltweit.

Der Fachbereich setzte dabei auf den Einsatz neuester Technologien, weil dies größtmögliche Flexibilität und Funktionalität des neuen Systems versprach. Neben der

veralteten HOST-Technologie bot System B weder die Kapazitäten für einen globalen Einsatz noch die Möglichkeit der zentralen Pflege und Wartung des Systems.

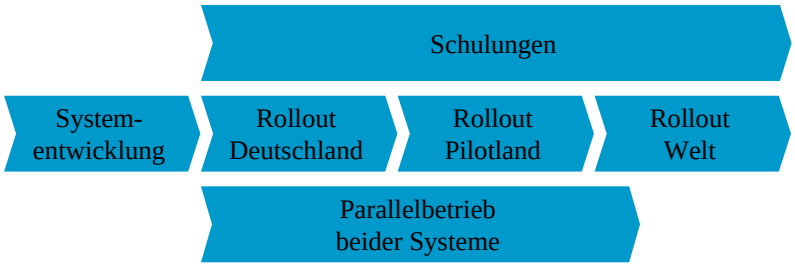


Abb. 3.3: Projekt B im Überblick

Zunächst wurde wie im Projekt A mit der Entwicklung eines Systems begonnen. Jedoch verlief dies nicht ohne Komplikationen. Ein erster Versuch wurde nach knapp einem Jahr eingestellt, da die verwendete Technologie den Anforderungen nicht gerecht wurde. Im zweiten Anlauf lag die Konzentration auf der Umsetzung der Kernfunktionalität. Zusatzfunktionen wurden zunächst durch das Altsystem B gewährleistet. Um den vollen Funktionsumfang nutzen zu können, mussten somit beide Systeme parallel genutzt werden. Das brachte erhöhte Anforderungen an die Anwender, führte zugleich zu erhöhten IT-Kosten und wirkte sich negativ auf das Budget aus. Der Parallelbetrieb konnte, wie in Abbildung 3.3 zu sehen, erst nach Beginn des weltweiten Rollouts eingestellt werden. Erst dann verfügte B* über die notwendige Funktionalität.

Die erforderliche Unterstützung im Fachbereich erwies sich als schwierig. Die Organisation des Fachbereichs, der aus Fachabteilungen aus fünf europäischen Ländern bestand, gestaltete sich aufgrund der Entfernungen als problematisch. Zum einen waren Termine schwer zu koordinieren, zum anderen konnten nicht genügend Ressourcen personeller Art bereitgestellt werden, um die Arbeit des Fachbereichs sicherzustellen. Infolgedessen mussten, so weit dies möglich war, Aufgaben des Fachbereichs durch die IT-Abteilung übernommen werden, um den Projekterfolg nicht zu verzögern oder gar zu gefährden.

Bei der Integration des Systems B* wurde deutlich, wie vorteilhaft die technische Dokumentation des Altsystems ist. Hier fanden sich neben den aus der Anwendungssoftware heraus ersichtlichen Schnittstellen jene, die nur für regelmäßige Batchabläufe im Hintergrund genutzt wurden. Dabei reichten die Abstände der Batchläufe je nach Schnittstelle von mehrmals täglich bis hin zu einmal jährlich.

Der anschließende Rollout von System B* in Deutschland brachte in erster Linie positive Resonanz der Anwender. Der nächste Schritt war ein weltweiter Rollout, der zunächst in einem Pilotland erprobt werden sollte. Dabei gab es jedoch schwerwiegende

Probleme beim Einsatz von B*, da die infrastrukturellen Voraussetzungen im Pilotland nicht den Anforderungen von System B* genügten. Die Folgen waren erhebliche Verzögerungen für den Einsatz des Systems, da ein problemloser Betrieb nicht ohne Investition in die Technik nicht sichergestellt werden konnte.

Als Konsequenz daraus wurden Mindestanforderungen an die technische Infrastruktur für einen störungsfreien Betrieb definiert. Das Rollout-Team, welches für die Planung und Durchführung der weiteren Rollouts verantwortlich war, ging mit diesen Mindestanforderungen in weitere Länder und prüfte dort zunächst die technischen Voraussetzungen. Eine anschließende Gap-Analyse signalisierte frühzeitig durch Identifizierung von Lücken Handlungsbedarfe im Rollout-Land und unterstützte einen störungsfreien Ablauf des Rollouts.

Die Einführung und Schulung von System B* wurde in jedem Land nur bei einer kleinen Anzahl von Nutzern, den so genannten Hauptanwendern begleitet. Die Schulung der übrigen Anwender übernahmen im Anschluss die Hauptanwender. So konnte sich das Rollout-Team bereits auf die Einführung von System B* im nächsten Land vorbereiten und den gesamten Rollout wesentlich forcieren.

Für etwaige Anforderungen der Rollout-Länder wurde ein Anforderungsmanagement geschaffen, welches Anforderungen erfasst, bewertet und priorisiert. Mit Hilfe dieser priorisierten Liste von Anforderungen werden jährlich ein bis zwei Releases für die Weiterentwicklung des Systems B* geplant und umgesetzt.

3.1.3 Projekt C

Das System C wurde seit Anfang der 90er Jahre konzernweit bei Volkswagen im Bereich der Beschaffung eingesetzt. Es basierte auf einer MS Access Anwendung und Oracle Forms. Eine über die Jahre gewachsene Struktur und der nicht-modulare Aufbau machten das System jedoch sehr wartungsaufwendig. Durch organisatorische Umstrukturierungen sollte ein zweiter Geschäftsprozess durch das System unterstützt werden. Die Umsetzung gelang bis zu einem bestimmten Punkt, jedoch zeigten sich schnell die Grenzen des maroden System C. Zum einen konnten durch die verwachsene Struktur und die nicht-modulare Architektur des Systems die Prozesse nicht miteinander verzahnt werden und zum anderen stieg mit dem neuen Prozess die Anzahl der Anwender. Das brachte das System nicht nur an seine Kapazitätsgrenzen, sondern erforderte zugleich einen hohen administrativen Aufwand. Die Lösung ließ sich nur in einem neuen System finden. Die Abbildung 3.4 zeigt den Projektverlauf.

Gemäß des Volkswagen SEP wurde das System C* in Zusammenarbeit mit den Fachbereichen entwickelt und mit Hilfe moderner Technologie sowohl zukunftsfähig, als auch wartungsfreundlich gestaltet. Bei der Entwicklung von C* wurden zugleich mehrere Insellösungen harmonisiert, die durch ‚Anstricken' kleiner Programme durch die jeweiligen Fachbereiche entstanden waren. Zugleich wurden Freiräume für lokale Besonderheiten der einzelnen Marken und Regionen geschaffen, um außerhalb des unterstützten Geschäftsprozesses die Nutzung der erfassten Daten zu ermöglichen. Dazu wurde eine zweite Datenbank zur Verfügung gestellt, die stets aktuelle Daten für die Kommunikation mit anderen Systemen über Schnittstellen zur Verfügung stellt.

In der Testphase des Systems wurden letzte Feinabstimmungen vorgenommen, um einen störungsfreien Übergang von alt auf neu zu gewährleisten. Um eine Recherche zu ermöglichen, wurden die Daten des Altsystems C in einem Archiv abgelegt. Dazu wurde in einem Teil der Datenbank von C* die Datenstruktur des Altsystems 1:1 nachgebildet und die Daten dort abgelegt. Diese Art der Datenbankkapselung erforderte nur einen minimalen Aufwand und ermöglichte die Abschaltung des Altsystems bei gleichzeitiger Nutzung der Altdaten. Einziger Nachteil dieser Variante war der erhöhte Bedarf an Datenbankkapazitäten.

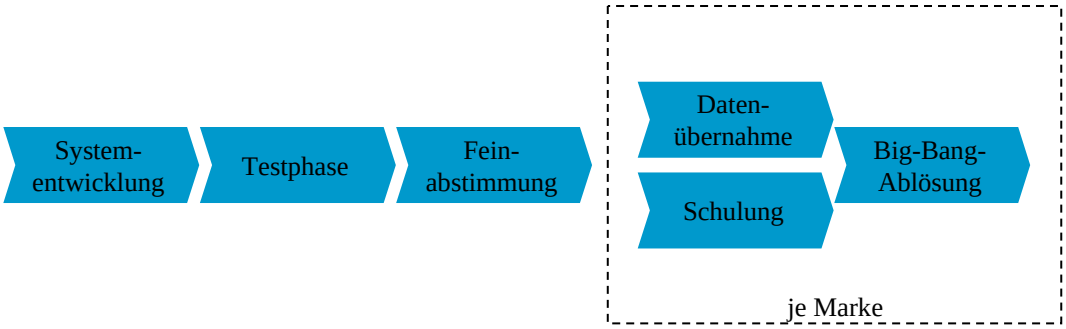


Abb. 3.4: Projekt C im Überblick

Mit der Übernahme der Daten war das System C* bereit für die Einführung. Nach erfolgter Anwenderschulung konnte das System schrittweise Marke für Marke ausgerollt werden. Dieses Vorgehen zeigte sich als sehr förderlich und garantierte eine spezifische Unterstützung der jeweiligen Fachabteilungen. Dabei wurden je Marke vor der Einführung die entsprechenden Daten übernommen. Dieses Verfahren konnte nur angewendet werden, weil die Daten nur für die jeweiligen Marken relevant waren und ein Austausch zwischen den Marken nicht von Interesse war.

3.1.4 Projekt D

Mit dem System D wurden im Volkswagen Konzern Vorgänge im Bereich Bestellung abgewickelt. Aus einer Prozessanalyse heraus ergaben sich Potenziale, die zu massiven Kosteneinsparungen führen würden. Im Ergebnis der Analyse standen zunächst zwei Alternativen. Zum einen eine SAP-Software, die an den Prozess angepasst werden müsste und zum anderen ein System X einer Tochtergesellschaft, welches jedoch an die Anforderungen dieses Geschäftsprozesses angepasst werden müsste. Eine dritte Alternative bot die Eigenentwicklung mit Hilfe des Volkswagen SEP. Kritischster Faktor des Projektes D war die Zeit. Die Vorgabe von einem Jahr für das gesamte Projekt war laut Projektleitung eine Herausforderung, die nur durch gute Zusammenarbeit aller Beteiligten zu schaffen war. In der Abbildung 3.5 ist der Verlauf des Projektes D zu sehen.

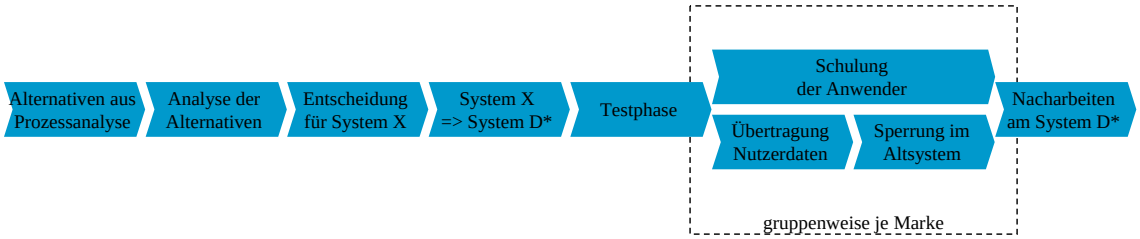


Abb. 3.5: Projekt D im Überblick

Der Aufwand für die Anpassung des SAP-Systems und die Integration bzw. Anbindung vor- und nachgelagerter Prozesse überstieg nach Abschätzung der Verantwortlichen allerdings den vorgegebenen Zeitrahmen. In Konsequenz war diese Variante ungeeignet.

Der Datenbedarf des Systems X zeigte große Schnittmengen mit dem des Systems D. Eine eingehende Analyse des Systems X zeigte, dass der Aufwand für die Erweiterung weitaus umfangreicher war, als zunächst angenommen. Hintergrund war, dass das System X ein ganz anderen Prozess und mehr noch ein ganz anderes Prozessumfeld unterstützte. Im Kern waren die Prozesse jedoch vergleichbar. Das war der ausschlaggebende Grund eine umfangreiche Überarbeitung des Systems X vorzunehmen, um daraus ein System D* zu schaffen.

Die Unterstützung durch den Fachbereich gestaltete sich auch hier durchaus schwierig, da im Hinblick auf den bestehenden und benötigten Funktionsumfang der Systeme keine Informationen bereit gestellt werden konnten. Eine Prüfung prozessrelevanter Schritte und Einflüsse war somit nur bedingt möglich.

Im Projektverlauf kam es zu einem Zeitverzug. Aufgrund dieses Zeitdrucks wurde vom neuen System D* nur eine Schnittstelle zu D geschaffen, um einen Datenaustausch zu ermöglichen. Weitere Schnittstellen sollten erst nach Produktivstart geschaffen werden, um Zeit zu gewinnen. Ein Datenaustausch mit anderen Systemen konnte somit nur über das System D geschehen, welches

System D* besaß zunächst nur eine einzige Schnittstelle zur Weitergabe von Daten – die zum System D. Das Datenmodell der beiden Systeme war sehr ähnlich und ein täglicher Datenaustausch mit geringem Aufwand zu bewältigen. Die Versorgung mit erforderlichen Daten für System D*, sowie für die angrenzenden Systeme, war so über die Datenbank und die vorhandenen Schnittstellen des Systems D gewährleistet. Dieser Umweg ermöglichte einen schnelleren Projektverlauf, da die Datenversorgung zunächst gesichert war. Standardisierte Schnittstellen wurden zu einem späteren Zeitpunkt eingeführt.

Nach erfolgreicher Testphase konnte die Einführung des Systems bei einzelnen Marken des Konzerns schrittweise realisiert werden. Nach erfolgreicher Schulung erhielten einzelne Benutzer die Zugangsberechtigung. Parallel dazu wurden während der Schulung benutzerspezifische Daten und Vorgänge aus dem Altsystem D in das System D* übertragen und in D gesperrt.

Im Rahmen des Projektes D musste der Steuerkreis stets mit Informationen versorgt werden, um vorgegebene Rahmenbedingungen nicht zu überschreiten und gegebene Ressourcen voll auszuschöpfen. Die Entscheidungen, die dort getroffen wurden, hatten zumeist strategischen Charakter. Für die operative Projektgestaltung war das Projektteam verantwortlich, das sich aus Projektleitung und Vertretern der Fachbereiche zusammensetzte. Letztere waren jedoch nicht von Anfang gewollt, ihre Verantwortung wahrzunehmen. Relevante Entscheidungen mussten dann getroffen werden, ohne die Meinung des betreffenden Fachbereichs zu hören. Dies führte zum einen zu einer sehr starken Konzentrierung auf die von Beginn an anwesenden Fachbereiche und zum anderen zu Missverständnissen und wiederholten Entscheidungen. Der anfängliche Verzug konnte letztlich durch Einbindung aller Fachbereichsvertreter in das Projekt wieder aufgeholt werden. Durch einen ausgewählten Kreis von Hauptanwendern gewann das Projektteam wertvolle Erfahrungen des täglichen Gebrauchs und nutzte damit die Möglichkeit diverse Ausnahmeregelungen zu prüfen, um den Prozess zu verschlanken.

Das Projekt D hat letztlich sein Ziel erreicht und das System D durch das System D* abgelöst. Der vorgegebene Zeitrahmen von einem Jahr wurde trotz stetigem Einsatz von Ressourcen aufgrund zu optimistischer Planungen einzelner Projektschritte nicht

eingehalten. Eine Releaseplanung für D* sichert fortan die Umsetzung der Anforderungen der Fachbereiche. Die SAP-Option aus der Prozessoptimierung bleibt weiterhin offen, da diese nur aus Gründen ausgeschlossen wurde, die den zeitlichen Rahmen einer Realisierung betreffen.

3.1.5 Projekt E

E ist ein System zum Controlling von IT-Projekten. Es basiert auf der Mainframe-Technologie und einem relationalem Datenbanksystem. Eine geplante Ablösung des Systems durch eine SAP-Lösung scheiterte auf Grund des Zusammentreffens verschiedener schwerwiegender Fehler. Durch ineffektives, nicht nachhaltiges Projektmanagement gekoppelt mit oberflächlicher Konzeption, mangelhafter Umsetzung in der Entwicklung und schlechter Kommunikation misslang das Vorhaben System E zu ersetzen. Die Abbildung 3.6 zeigt das Vorgehen im Zusammenhang.

Abb. 3.6: Projekt E im Überblick

Schon in der frühen Projektphase wurde es versäumt, eine Situationsbeschreibung vorzunehmen und die Ziele des Projektes präzise festzulegen. Eine Definition von Erfolgsfaktoren geschah ebenfalls nicht. Daneben wurde im Projektmanagement versäumt, betroffene Organisationseinheiten über geplante Maßnahmen zu informieren. Das führte nicht nur zu Unzufriedenheit und Desinteresse, sondern auch zu mangelndem Informationsfluss und fehlender Unterstützung.

In der Konzeptionsphase traten gravierende Fehler bei der Aufnahme und Dokumentation der notwendigen Daten auf. In Folge wurden nur Fragmente der komplexen Funktionsstruktur von System E identifiziert und dokumentiert. Die Notwendigkeit einer Prozessaufnahme wurde ebenso unterbewertet, wie eine anschließende Analyse der gesammelten Daten. Vorhandene Schnittstellen wurden dementsprechend nur teilweise erkannt. Die erstellten Dokumente enthielten folglich nur einen Ausschnitt der tatsächlich vorhandenen Funktionalität und keine Informationen über den gelebten Prozess, ebenso wie über vor- und nachgelagerte Systeme.

In der Entscheidungsphase wurde eine SAP-Software ausgewählt, die die beschriebenen Anforderungen nahezu erfüllen würde. Nur war dies nicht die tatsächlich notwendige Funktionalität, sondern nur ein Teil dessen. Mit dem Einsatz der Software zeigten sich

diese Lücken. Bei den Anwendern fand das System keine Akzeptanz, da zum einen die Benutzerführung eine große Umstellung bedeutete und zum anderen auf Grund fehlender Funktionalität und Schnittstellen der Anwender weiterhin zwei Systeme bedienen musste.

Das System E dominiert die SAP-Lösung jedoch nicht nur in Hinblick auf die Funktionalität. Da System E eine Eigenentwicklung ist, beschränken sich die Ausgaben auf Betriebs- und Wartungskosten. Für SAP-Systeme fallen zusätzlich Lizenzkosten in nicht unerheblichem Maße an. Diese wären unter Umständen nur zu rechtfertigen, wenn die gebotene Funktionalität höher ist als die bisher vorhandene und die Qualität bzw. die Unterstützung des Geschäftsprozesses im Rahmen der Einführung des neuen Systems entscheidend verbessert wird.

Als Konsequenz der beschriebenen Situation gilt das Projekt als nicht erfolgreich. Ein Termin für die Abschaltung der SAP-Lösung ist längst festgelegt und die Arbeit im und am System wurde eingestellt. Vorhandene Daten werden, falls relevant und nicht bereits vorhanden, in die Datenbank des Systems E übernommen.

Das Scheitern des Projektes ist letzten Endes auf vorgelagerte Prozesse der Systemablösung zurückzuführen. Durch unvollständige Informationserfassung bei der Systemanalyse konnte in der Entscheidungsphase nur ein System ausgewählt werden, dass den tatsächlichen Anforderungen nie gerecht werden konnte. Kritisch zu betrachten ist jedoch, dass dieses System letztendlich trotz alledem eingeführt wurde und System E ablösen sollte. Spätestens in einem Fachbereichs- oder Benutzerakzeptanztest hätten Mängel auffallen müssen, die eine Einführung verhindert bzw. verschoben hätten. Je nach Grad der Abweichung würden Nacharbeiten am neuen System oder gar ein Neustart des Projektes zum Erfolg führen.

3.2 Fazit

Bei der Betrachtung der einzelnen Systemablösungen wurde festgestellt, dass sie sich in ihrer Vorgehensweise stark voneinander unterscheiden. Abgesehen davon, dass zum Teil Individualsoftwarelösungen entwickelt und zum Teil Standardsoftware eingesetzt wurde, fiel eine unterschiedliche Vorbereitung und Projektplanung auf. Die Systemablösung wurde hier häufig als Teil eines Softwareentwicklungsprojektes betrachtet. Dementsprechend lag der Schwerpunkt beispielsweise beim Einsatz von Ressourcen in der Entwicklung der Software. Die Bedeutsamkeit der Ablösung selbst wurde nur unzureichend erkannt.

Ebenso sind Mängel in der Art und Form der Testphase aufgetreten. So wurden Tests teilweise nicht von der Fachabteilung durchgeführt, die das System auch einsetzen wollten, sondern von der IT-Abteilung, die durch mangelnde Projektführung in schlechter Verbindung mit der Fachabteilung stand. Folglich konnte keine wirkliche Prüfung des Systems in Hinblick auf die Anforderungen des Fachbereichs stattgefunden haben.

Die beschriebenen Projekte zeigen weiterhin, dass es ein standardisiertes Vorgehen nicht gibt. Jedes Projekt hat für den Projektverlauf seine eigene Lösung entwickelt. Im Nachhinein stellte sich oft heraus, dass ein anderer Ablauf zu einem früheren Zeitpunkt zum gleichen Projektergebnis führen würde. So wären Ressourcen effektiver und gezielt eingesetzt worden. Zugleich hätten durch einen früheren Projektabschluss Kosten eingespart werden können. Zum einen sind das Kosten, die das Projekt verursacht, zum anderen Kosten, die eventuell durch den Einsatz des neuen Systems hätten eingespart werden können.

Eine strukturierte und standardisierte Vorgehensweise würde hier den Vorteil bringen, dass die Projektplanung im Rahmen vorgegeben wäre und die Projektleitung sich vollends auf die Beschaffung von Informationen konzentrieren könnte. Vorbereitungsschritte wären vorausschauend angesetzt und würden den Projektverlauf beschleunigen.

Im folgenden Kapitel werden zunächst die Anlässe für eine Systemablösung ermittelt und kategorisiert. Sie bilden den Ausgangspunkt einer jeden Systemablösung und haben dabei unterschiedlichen Einfluss darauf.

4 Auslöser für Systemablösungen

Für die Ablösung eines Informationssystems findet sich eine Vielzahl von Gründen. In diesem Kapitel wird nach einer Differenzierung der Gründe gesucht, um sie zu gruppieren.

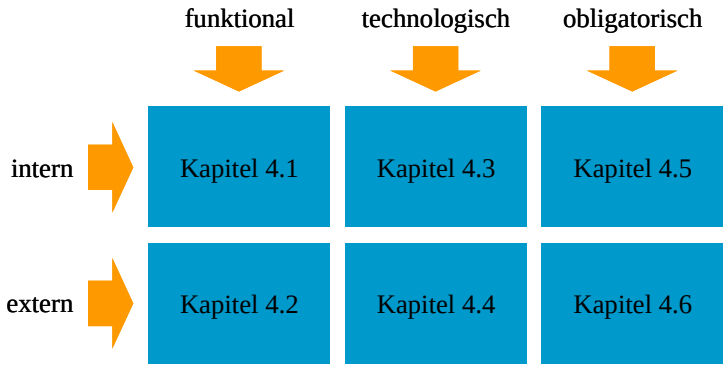
Zunächst lässt sich laut STAEHLE und SCHANZ eine Unterscheidung interner und externer Grundlagen für einen systemtechnischen Wandel vornehmen, wobei als System in erster Linie eine Organisation verstanden wird (Vgl. Staehle (1994), S.854; Schanz (1994), S.382f.).

Nach UTTERBACK kann der Wandel eines Systems durch technologische und funktionale Faktoren ausgelöst werden. Dabei wirken technologische Faktoren als treibende innovative Kraft. Funktionale Auslöser hingegen sieht UTTERBACK als Elemente, die auf Grund von Anforderungen an das System Neuerungen bedingen (Vgl. Utterback (1971), S.126 ff.).

DÖMER greift die beiden genannten Sichten auf, und fügt sie in einem Modell zusammen. Dabei klassifiziert er die Ursachen eines Migrationsprojektes anhand von zwei Dimensionen. Zunächst unterscheidet er zwischen externen und internen Faktoren. Dabei können Migrationauslöser innerhalb der Organisation sowohl in der IT, als auch in anderen Organisationseinheiten vorkommen. Ferner nimmt DÖMER eine Differenzierung in funktionale und technologische Gründe vor (Vgl. Dömer (1998), S.57f.).

Nach WIECZORREK und MERTENS gibt es jedoch noch weitere Ursachen für eine Systemablösung. Strategischer Unternehmensentscheidungen und Anforderungen durch staatliche Vorgaben können ebenfalls dazu führen, dass IT-Systeme ersetzt werden müssen (Vgl. Wieczorrek/Mertens (2007), S. 331). Eine Einordnung dieser Faktoren ist jedoch im Modell von DÖMER nicht möglich.

Erst durch Hinzufügen obligatorischer Ursachen in das Modell von DÖMER lassen sich strategische Unternehmensentscheidungen und staatlichen Vorgaben einordnen. Dabei sind die Entscheidungen der Unternehmensführung den inneren und Gesetzesvorgaben den externen obligatorischen Faktoren zugeordnet. Die Abbildung 4.1 zeigt die Kombination der genannten Dimensionen und stellt eine Erweiterung des Modells von DÖMER dar. Somit lassen sich sechs Gruppen von Auslösern einer Systemablösung unterscheiden, auf die in den folgenden Unterkapiteln im Einzelnen eingegangen wird.



Quelle: In Anlehnung an Dömer (1998), S. 58.

Abb. 4.1: Auslöser einer Systemablösung

4.1 Interne funktionale Ursachen

Häufig reicht die vorhandene Funktionalität der Altsysteme nicht, um heutigen Anforderungen gerecht zu werden. Je nach System können Funktionen hinzugefügt und erweitert werden. Ist jedoch die Differenz zwischen System und Forderung umfangreicher, gelingt die Integration fachlicher Anforderungen nur mit sehr hohem Aufwand. Nicht selten steigt bei näherer Untersuchung des Altsystems der Aufwand ins Unermessliche. Aus diesem Grunde ist bei Analyse von Aufwand und Nutzen eine Systemablösung häufig die bessere Alternative.

Solche Anforderungen sind häufig das Ergebnis größerer Neuerungen in der Prozess- oder Organisationsstruktur eines Unternehmens. Im Zuge fortwährender Optimierungen und Standardisierungen z.B. in Form des kontinuierlichen Verbesserungsprozesses unterliegen Geschäftsprozesse einem ständigen Wandel. Die Veränderungen am Geschäftsprozess können dabei unterschiedliche Auswirkungen auf ein IT-System haben. Optimierungen sind häufig mit einer Zusammenlegung von Arbeitsschritten und Funktionen, aber auch von ganzen Prozessen oder Prozessteilen verbunden. Folglich wird Funktionalität gefordert, die das System nicht erfüllen kann.

Optimierungen können jedoch auch Rationalisierung und Wegfall von Prozessteilen oder ganzen Prozessen bedeuten. Wenn das System einen oder mehrere Geschäftsprozesse unterstützt hat, die nun erheblich beschnitten wurden oder zum Teil nicht mehr existieren, kann in Folge dessen der Zweck eines Systems in Frage gestellt werden. Zudem werden gelegentlich im Zuge einer Harmonisierung von Prozessen Prozessteile wiederum in andere Geschäftsprozesse integriert. Systeme, die die Prozesse

oder Prozessteile bisher unterstützt haben, müssen dann häufig neuen Systemen weichen.

Wandlungen innerhalb der Organisationsstruktur sind zuweilen auch verantwortlich dafür, dass IT-Systeme ersetzt werden. So führt ein Zusammenlegen von Organisationsteilen typischerweise zu wachsenden Aufgaben- und Verantwortungsbereichen. Oftmals kamen in den bisherigen Geschäftsbereichen unterschiedliche IT-Systeme zum Einsatz, die nun konsolidiert und standardisiert werden müssen, sodass es zwangsläufig zur Ablösung kommt.

In Großkonzernen ist die Tendenz zur *weltweiten Bereitstellung von Daten* zu erkennen. Gleichartige Geschäftsbereiche eines Konzerns und seiner Tochterunternehmen können dadurch voneinander profitieren und Synergien nutzen. Fehler werden vermieden und gemeinsam Lösungen für Probleme gefunden. Die globale Bereitstellung der Daten ist neben Aspekten wie zentraler Wartung und Kosteneinsparungen häufig ein Grund dafür, ein einheitliches System einzuführen.

4.2 Externe funktionale Ursachen

Unternehmen, die in Wettbewerb zueinander stehen, können unmittelbar eine Systemablösung hervorrufen. Das trifft beispielsweise zu, wenn ein Unternehmen den Kunden beispielsweise eine neue Dienstleistung anbietet, die die Konkurrenz nur durch eine Systemablösung erbringen kann (Vgl. Dömer (1998), S. 57).

Ein Unternehmen muss auf den Markt reagieren, um erfolgreich zu sein. Eine solche Ausrichtung auf den Markt führt zu einer Fokussierung und zieht gewöhnlich Restrukturierungsmaßnahmen in der Organisationsstruktur nach sich (Vgl. Voigt/Linke (2005), S. 37). Diese Konzentration auf bestimmte Aufgabenbereiche bringt in der Regel neue Anforderungen mit sich, die das bisherige System nicht erfüllen kann. Eine Systemablösung ist dann häufig die Folge.

4.3 Interne technologische Ursachen

Viele Systeme sind bereits mehr als zehn Jahre in Betrieb. Sie zeichnen sich nicht selten durch schlechte Wartungsqualität auf Grund unstrukturierter Tätigkeiten mehrerer Entwicklergenerationen aus. Heute sind dann ganze Abteilungen mit der Wartung eines Systems beschäftigt. Die entstehenden Kosten für Personal- und Hardwareeinsatz

rechtfertigen in der heutigen Zeit nicht immer den Betrieb eines älteren Systems (Vgl. Dömer (1998), S. 57).

Häufig haben sich Bedingungen, die zur Zeit der Entwicklung eines Systems angenommen wurden, im Vergleich zu heute entscheidend geändert. Die Zahl der Anwender ist meistens verhängnisvoll gestiegen. Damit verbunden ist eine Steigerung des Datenaufkommens, das zusätzlich durch die Tendenz zur Erfassung von mehr und mehr Daten belastet wird. Das System gelangt dadurch häufig bis an seine *Belastungsgrenze*. Die vorhandenen Kapazitäten reichen dann nicht aus und lassen sich auf Grund der Technologiewahl nicht erweitern, um heutigen Anforderungen wie beispielsweise (bspw.) akzeptable Zugriffszeiten und Datendurchsatzraten zu gerecht zu werden.

Für den Betrieb eines Systems fallen unterschiedliche *Kosten* an. Neben Betriebs- und Wartungskosten fallen oftmals Lizenzkosten an. Werden in einem großen Unternehmen mehrere dezentrale Systeme zum gleichen Zweck eingesetzt, ist es häufig kosteneffizienter sie durch ein zentrales System zu ersetzen. Zentrale Systeme bieten den Vorteil, dass sich durch den Einsatz in mehreren Geschäftsbereichen die entstehenden Kosten für Wartung, Betrieb und Lizenzen auf die Vielzahl der anwendenden Unternehmensbereiche verteilen lassen.

Es existieren mitunter IT-Systeme, die nur kleinere Wartungsmannschaften beschäftigen. Das geht soweit, dass der verantwortliche Personenkreis für das System im Laufe der Zeit bis auf ein oder zwei Mitarbeiter reduziert wird. Reicht die Funktionalität und der Reifegrad des Systems, um die Anforderungen des Fachbereichs zu erfüllen, ist an diesem Vorgehen zunächst nichts einzuwenden. Wenn jedoch die *verantwortlichen Mitarbeiter* das Unternehmen mehr oder weniger schnell verlassen, bspw. durch Wechsel zur Konkurrenz oder Altersruhestand, und keine brauchbare Dokumentation des Systems hinterlassen, kann das soweit führen, dass der Betrieb des Systems akut gefährdet und eine Systemablösung unausweichlich ist.

4.4 Externe technologische Ursachen

Für einen gewissen Zeitraum garantieren Hersteller von Hard- und Software eine technische Unterstützung ihrer Produkte. Durch Nachfolgeprodukte verringert sich im Laufe der Zeit der Marktanteil der verwendeten Hard- oder Software. Die Hersteller bieten von nun an die technische Unterstützung nur noch gegen Gebühr an, die möglicherweise mit der Zeit ansteigt, um letzten Endes den Kunden zum Umstieg auf das neue Produkt zu bewegen. Das kann im Extremfall soweit gehen, dass die

Unterstützung für das Produkt gänzlich eingestellt wird. Das System ist dann nur noch eingeschränkt nutzbar und muss ersetzt werden (Vgl. Dömer (1998), S. 58).

Produzenten von Hard- und Software sind letztlich auch nur Unternehmen, die den Gegebenheiten des Marktes unterliegen. Zum Teil sind sie fest etabliert und behaupten sich schon über Jahre und Jahrzehnte. Es kommt jedoch gelegentlich vor, dass ein solches Unternehmen vom Markt verschwindet. Die Gründe dafür sind vielfältig. Softwarehersteller, die sich beispielsweise gegenüber dem Technologiewandel verweigern, werden nicht mehr lange auf dem Markt präsent sein. Gleiches kann passieren, wenn Hersteller plötzlich aufgekauft werden (Vgl. Versteegen et al. (2001), S.71).

DÖMER ordnet Nutzengewinne durch den unmittelbaren Einsatz neuer Technologiegenerationen ebenfalls den externen technologischen Auslösern einer Systemablösung zu. An dieser Stelle können Kostengründe ein interessanter Ansatzpunkt sein (Vgl. Dömer (1998), S. 58).

4.5 Interne obligatorische Ursachen

Systemablösungen können Folge einer strategischen Entscheidung der Unternehmensführung sein (Vgl. Wieczorrek/Mertens (2007), S. 331). Denkbar ist eine langfristige Wahl eines zukunftsfähigen Herstellers von Hard- oder Software für große Unternehmensteile, durch die sich günstige Wartungs- und Lizenzverträge abschließen lassen. Laufende Kosten sind somit langfristig bekannt und können in Budget- und Projektplanungen berücksichtigt werden. Ein weiterer Vorteil liegt darin, dass verschiedene Systeme eines Softwareherstellers in der Regel sehr gut miteinander kommunizieren. Im Gegensatz dazu ist es bei Systemen unterschiedlicher Hersteller in der Regel aufwendiger, zwei Systeme zum Austausch von Daten zu bewegen. Mitarbeiter lassen sich durch sicheren Umgang mit dem System flexibler einsetzen und Einarbeitungszeiten werden auf ein Minimum reduziert.

Durch eine derartige Entscheidung werden mittelfristig Systeme betroffen sein, die nicht unbedingt aus fachlicher Sicht, also auf Grund fehlender Funktionalität abgelöst werden, sondern vielmehr aus strategischer Sicht in Bezug auf erfolgreiche langfristige Zusammenarbeit mit dem Softwareproduzenten. Der Vorzug liegt des Weiteren darin, dass es immer den ‚richtigen' Ansprechpartner gibt, wenn es Probleme mit einem System gibt.

4.6 Externe obligatorische Ursachen

Gesetzesvorhaben sind in der Regel obligatorisch und müssen umgesetzt werden. Die einzige Wahl, die ein Unternehmen dann hat, liegt darin, das bestehende System anzupassen oder neue IT-Applikationen einzuführen, um die Gesetzesanforderung zu erfüllen. Beispielsweise musste das Kreditgewerbe wegen der Einführung des Euro seine Systeme zu einem festen Termin anpassen (Vgl. Wieczorrek/Mertens (2007), S. 331).

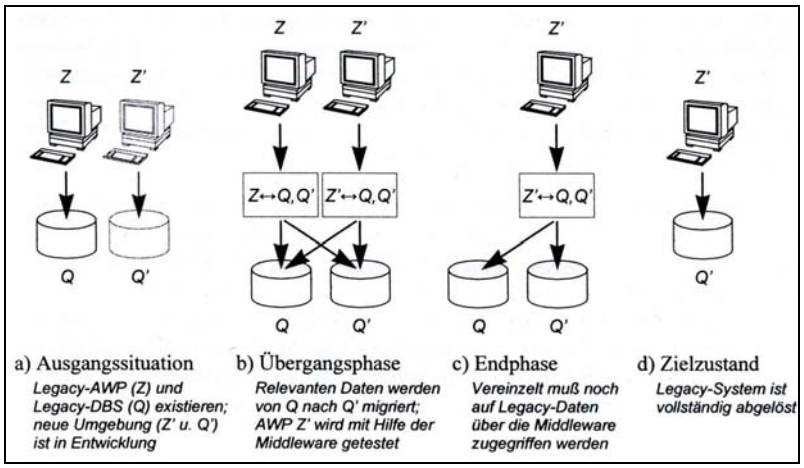
Die Ursachen einer Systemablösung konnten in sechs Kategorien eingeteilt werden. Diese Klassifizierung hat einen Einfluss auf die Planung der Systemablösung. Beispielsweise sind mit jeder Kategorie verschiedene Rahmenbedingungen und Einflussfaktoren verbunden.

Das Eintreten der oben genannten Ursachen kann zum großen Teil nicht verhindert werden. Deshalb werden im Folgenden Strategien und Vorgehensweisen gesucht, nach denen eine Systemablösung erfolgen kann, um bei Eintreten der Ursachen vorbereitet zu sein und ein standardisiertes Verfahren entwickeln zu können. Die Strategie hat dabei einen entscheidenden Einfluss auf das Vorgehen und die Planung einer Systemablösung.

5 Strategien zur Systemablösung

5.1 Gateway-Ansatz

Der Gateway-Ansatz nach SAUTER beschreibt ein evolutionäres Vorgehensmodell, in dem in einer zunächst heterogenen Systemlandschaft durch die Einführung einer Abbildungsschicht, dem so genannten Gateway, die Koexistenz von Alt- und Neusystem einen wichtigen Schritt zur Migration der Systemteile darstellt.



Quelle: Sauter (1998), S. 97.

Abb. 5.1: Gateway-Ansatz nach Sauter

Die Abbildung 5.1 zeigt das Vorgehen im Zusammenhang. Dabei ist sind die Systemteile des Altsystems durch Z für Anwendungsschicht und Q für Datenhaltungsschicht abgebildet. Analog ist das Neusystem durch Z' und Q' dargestellt. Voraussetzung dafür ist, dass sich mindestens die Anwendungsschicht klar von der Datenhaltungsschicht des zu migrierenden Anwendungssystems trennen lässt. Typisches Beispiel einer so genannten 2-Tier-Architektur sind Client/Server-Modelle. Auf dem Client befindet sich das Anwendungsprogramm, das über eine Schnittstelle direkt auf den Datenbankserver zugreift. Des Weiteren existieren 3- und 4-Tier-Architekturen, die Erweiterungen der 2-Tier-Architektur darstellen und somit ebenfalls die Voraussetzung für SAUTER's Vorgehensmodell erfüllen. Die klare Trennung der Ebenen innerhalb der Systemarchitektur ist für das weitere Vorgehen elementar wichtig, da direkt oberhalb der Datenbankschicht eine Abbildungsschicht eingefügt wird. Diese zeichnet sich durch Schnittstellen zu den Datenbanken und Anwendungsprogrammen von Alt- und Neusystems aus. Damit stellt die Abbildungsschicht eine Verbindung zwischen den beiden Systemen her und bildet so den zentralen Punkt des Vorgehensmodells.

Die Aufgaben der Abbildungsschicht sind vielfältig. Zunächst werden Abbildungs- und Datenhaltungsschicht von einander isoliert. Zugriffe werden fortan über die Abbildungsschicht gesteuert. Dabei müssen auch Anfragen des einen Systems auf die Datenbank des anderen Systems koordiniert werden und Ergebnisse zurückgeführt werden. Da die Datenmodelle der Systeme nicht übereinstimmen müssen, ist eine Übersetzung der Anfragen und Ergebnisse erforderlich und je nach Fall unterschiedlich aufwendig. Für den Zugriff von einem System auf das andere existieren ein Forward- und ein Reverse-Gateway. Mit dem Forward Gateway Z⇔Q' werden alle Anfragen und Ergebnisse der Altanwendung Z auf die Datenbank des neuen Systems Q' koordiniert und transformiert. Analog regelt das Reverse-Gateway Z'⇔Q die Zugriffe der neuen Anwendung auf die Datenbank des Altsystems.

Bevor es jedoch dazu kommen kann, werden nach Einfügen der Abbildungsschicht erst einmal die Applikationen des neuen Systems aufgebaut und eingefügt. Diese können über das Reverse-Gateway Z'⇔Q bereits auf Daten des Altsystems zugreifen. Dies ermöglicht Tests und Schulungen, aber letztlich auch den vorläufigen Betrieb der Applikationen.

Im folgenden Schritt wird nun die neue Datenquelle Q' entwickelt, anschließend implementiert und an die Abbildungsschicht angebunden. Nun können beide Systeme sowohl auf die alte als auch auf die neue Datenbank zugreifen.

Nachdem das neue System einsatzbereit ist, kann zunächst die Datenbank des Altsystems abgeschaltet werden, so dass alle Applikationen nur noch auf die Datenbank des neuen Systems zugreifen. Im letzten Schritt wird dann mit der Anwendungsschicht das Altsystem komplett stillgelegt und die Migration erfolgreich abgeschlossen (Vgl. Kölsch (2000), S. 47 ff.; Sauter (1998), S.95 ff.).

5.2 Chicken-Little

Zur Migration größerer Informationssysteme haben BRODIE und STONEBRAKER den Gateway-Ansatz weiterentwickelt und verallgemeinert. Ihr *Chicken-Little-Ansatz* basiert dabei auf eine aus elf Schritten bestehende inkrementelle Vorgehensweise zur Migration eines Informationssystems. Die eigentliche Migration wird so in kleinere Teile aufgespalten, um das Vorgehen überschaubar zu machen und das Risiko eines Misserfolgs zu minimieren.

Um die weitere Vorgehensweise zu bestimmen, werden zu migrierende Alt-Systeme zunächst anhand ihrer Systemarchitektur klassifiziert, wie es in Abbildung 5.2

dargestellt ist. Dabei steigt der Aufwand der Migration mit der Komplexität des Systems und ist abhängig von dem Grad der Zerlegbarkeit.

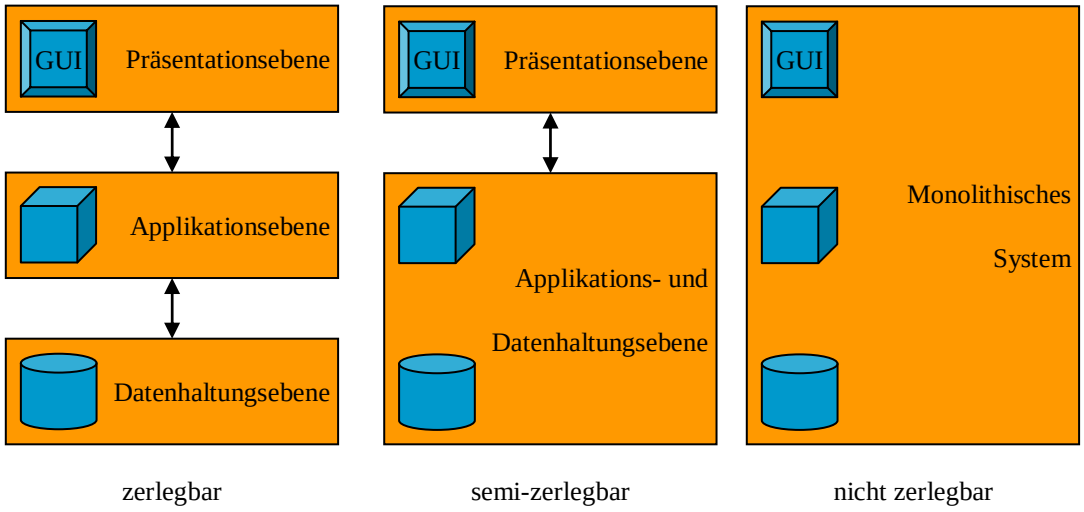


Abb. 5.2: Klassifizierung des Altsystems

Die beste Architektur für eine Migration ist die klassische 3-Tier-Architektur, die *Zerlegbarkeit* des Alt-Systems in Schnittstellen-, Applikations- und Datenhaltungsschicht.

Eine etwas schwierigere Architektur in Hinblick auf die Migration stellt das *semi-zerlegbare* System dar. Im Unterschied zur 3-Tier-Architektur lassen sich Applikations- und Datenhaltungsschicht nicht sauber voneinander trennen. In Folge dessen wird sowohl die Analyse als auch Migration komplexer und anfälliger für Fehler.

Ist der Quellcode des Alt-Systems derart verworren, dass sich weder Datenhaltung, noch Applikation oder Schnittstellen herauslösen lassen, liegt die dritte Variante, das *nicht zerlegbare* monolithische System vor.

Lässt sich ein Anwendungssystem nicht eindeutig einem der genannten Fälle zuordnen, treffen also Voraussetzungen für mehrere Varianten zu, so liegt ein hybrides Alt-System vor. Das ist häufig bei Systemen der Fall, die eine jahrzehntelange Entwicklung vorweisen können. Hier müssen entsprechend der vorgefundenen Architektur individuelle Schritte für einzelne Systemteile miteinander kombiniert werden.

Die im Folgenden benannten Migrationsschritte sollen ein strukturiertes Vorgehen ermöglichen und die das Migrationsprojekt in kleinere Teilprojekte untergliedern. Abhängig von der Beschaffenheit und der Umgebung des Alt-Systems müssen die einzelnen Schritte nicht zwangsweise sequentiell durchlaufen werden. Auch die Reihenfolge kann je nach Fall variieren (Vgl. Brodie/Stonebraker (1995), S. 31).

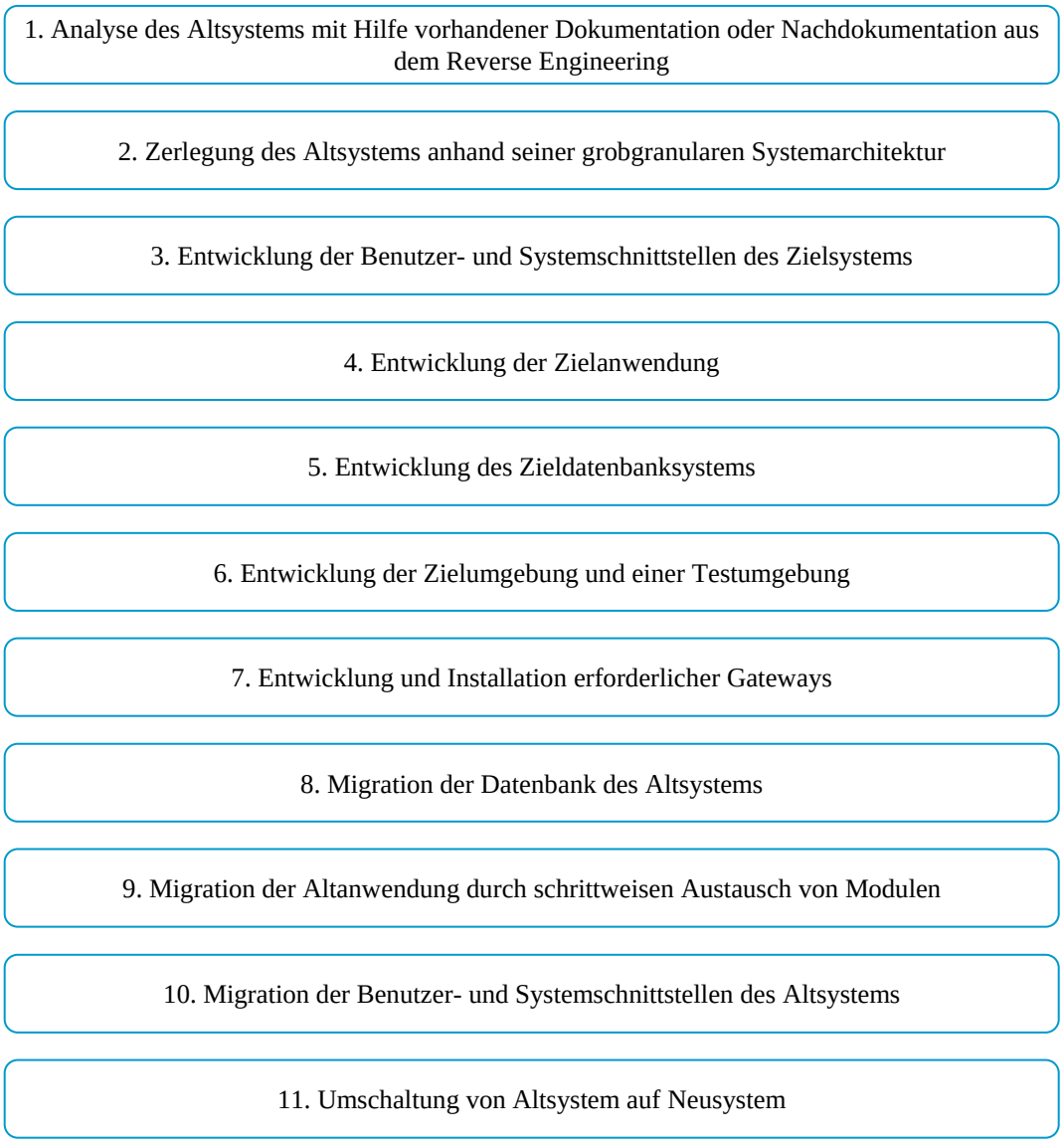


Abb. 5.3: Chicken-Little-Ansatz im Detail

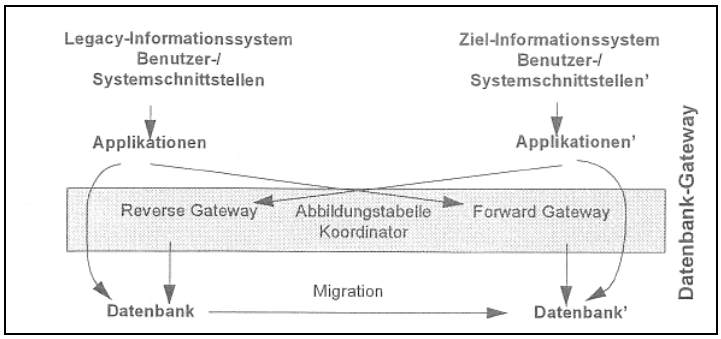
Entsprechend der vorgefundenen Systemarchitektur können Gateways nur an bestimmten Stellen eingefügt werden. Gateways haben dabei die Aufgabe, Komponenten, die unverändert bleiben sollen, von jenen zu isolieren, die einem Wandel unterliegen. Das Gateway tritt dabei als Vermittler zwischen den angeschlossenen Komponenten auf, um Anforderungen und Daten zu übersetzen. Des Weiteren übernimmt das Gateway die Koordination zwischen den verbundenen Komponenten, zum Beispiel bei Abfragen und Aktualisierungen von Daten.

Anhand der Klassifizierung des Altsystems werden die oben genannten Schritte entsprechend angewendet:

Zerlegbares Alt-System

Verfügt das Altsystem über eine 3-schichtige-Systemarchitektur, gleicht das Vorgehen weitgehend dem Gateway-Ansatz (Vgl. Abschnitt 5.1). Die Datenbank des Altsystems wird jedoch nicht im Ganzen durch ein Gateway gekapselt, sondern zunächst in einzelne Komponenten zerlegt, die dann wiederum gekapselt werden und als Dienst zur Verfügung stehen. Die Errichtung des neuen Datenbanksystems erfolgt inkrementell und komponentenbasiert, um anschließend eine schrittweise Migration der einzelnen Komponenten zu ermöglichen.

Während des Parallelbetriebs der beiden Systeme werden zwei Gateways benötigt. So können jeweils die Anforderungen der Applikation des einen Systems in das Datenbanksystem des anderen Systems übersetzt werden.



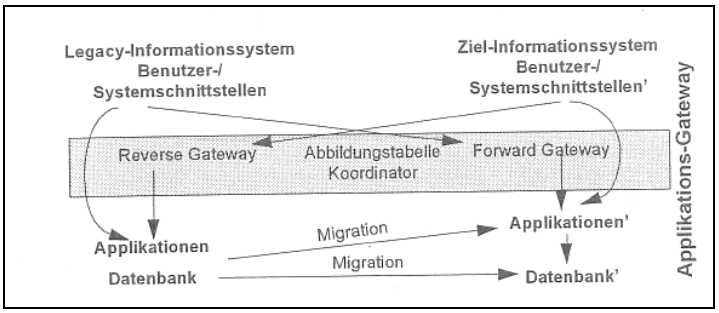
Quelle: Kölsch (2000), S.51.

Abb. 5.4: Chicken-Little Datenbank-Gateway

Der Parallelbetrieb bringt jedoch die Probleme der Redundanz und Konsistenz der Daten mit sich. Zu deren Lösung werden zwei Komponenten in die Gateway-Schicht eingeführt. Zum einen sind in einer Abbildungstabelle Informationen über die Abbildung der Entitäten der Systeme aufeinander festgehalten und zum anderen regelt ein Koordinator die Probleme des gleichzeitigen Zugriffs und der Konsistenzsicherung.

Semizerlegbares Alt-System

Bei Alt-Systemen mit semi-zerlegbaren Eigenschaften können in der Systemarchitektur lediglich die Benutzer- und Systemschnittstellen von einer vermischten Applikations- und Datenhaltungsebene getrennt werden. Somit kann die Abbildungsschicht nur oberhalb der Applikationsebene eingeführt werden.



Quelle: Kölsch (2000), S. 52.

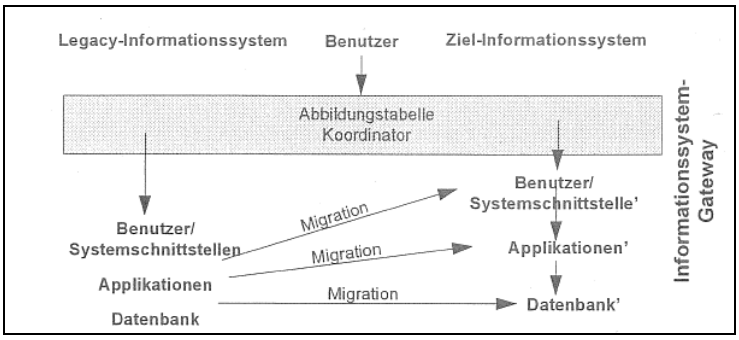
Abb. 5.5: Chicken-Little Applikations-Gateway

Aufgrund der schrittweisen Migration wird das Alt-System auf Applikationsebene zerlegt und die Funktionalität in einzelne Gateways gekapselt. Parallel dazu wird das Zielsystem komponentenartig in Form einer 3-Schicht-Archtektur aufgebaut. Das Applikations-Gateway wird um die Funktionalitäten des Zielsystems ergänzt und stellt diese zusammen mit denen des Alt-Systems für Aufrufe aus den Benutzeroberflächen zur Verfügung. Während des Parallelbetriebs werden die Aufrufe ähnlich wie bei zerlegbaren Alt-Systemen durch eine Abbildungstabelle und einen Koordinator entsprechend zugewiesen und weitergeleitet. Jedoch wird die Koordination auf Funktions- und Prozessebene durchgeführt und nicht auf Datenebene wie bei der vorhergehenden Variante. Aufrufe der Systeme werden dabei als Aufrufe des jeweils anderen Systems weitergeleitet und übersetzt. Ergebnisse dieser Aufrufe werden dann wiederum zurückübersetzt und an das entsprechende System übertragen.

Das Migrationsvorhaben gestaltet sich durch die Vermischung von Applikations- und Datenebene entsprechend schwieriger. Weil eine Trennung der beiden Ebenen nicht möglich ist, erfolgt die Migration durch zwei Aktivitäten. Zunächst wird das neue Datenbanksystem aufgebaut. Anschließend werden die Daten aus der Datenbank des Altsystems konvertiert und in das neue System transferiert. Nach erfolgreicher Migration der Daten erfolgt der Aufbau der neuen Applikationsebene. Ist die Übertragung der Systemfunktionalität aus dem Alt-System erfolgt, wird es abgeschaltet und die Migration erfolgreich abgeschlossen.

Nichtzerlegbares Alt-System

Da bei nichtzerlegbaren Alt-Systemen keine Trennung von Datenhaltungs-, Applikations- und Schnittstellenebene möglich ist, kann ein Gateway nur oberhalb der Benutzer-System-Schnittstelle eingesetzt werden. Dabei steigt der Grad der Komplexität erheblich an, da das Gateway nun jeweils ein komplettes Informationssystems kapseln muss.



Quelle: Kölsch (2000), S. 53.

Abb. 5.6: Chicken-Little Informationssystem-Gateway

Nach dem Ansatz der inkrementellen Vorgehensweise muss das nichtzerlegbare Alt-System in migrierbare Einheiten separiert werden. Zunächst werden dazu die Eigenschaften des Systems analysiert, aus denen Anforderungen entwickelt werden, die das Zielsystem erfüllen muss. Auf Basis dieses Anforderungskatalogs wird unterhalb des Informationssystem-Gateways das komplette Zielsystem implementiert. Parallel dazu werden Daten, Funktionen und Schnittstellen auf das Zielsystem übertragen und die einzelnen Einheiten migriert. Das Informationssystem-Gateway muss dabei ständig synchronisiert und aktualisiert werden. Es verfügt über eine Abbildungstabelle, die Informationen zu Daten, Funktionen und Schnittstellen der beiden Systeme enthält. Ein Koordinator regelt den Zugriff und stellt die korrekte Zuordnung mit Hilfe der Abbildungstabelle sicher.

5.3 Cold-Turkey

Der *Cold-Turkey-Ansatz* ist auch als *Big-Bang* bekannt und basiert ebenso wie der Chicken-Little-Ansatz auf einer kompletten Neuentwicklung des Informationssystems mit Hilfe moderner Entwicklungsmethoden. Während das Altsystem seinen Betrieb unbeeinflusst fortsetzt, wird parallel dazu ein neues System entwickelt und getestet. Im Unterschied zu Chicken-Litte besteht jedoch keinerlei Verbindung zwischen Altsystem und der Neuentwicklung. Es erfolgt keine schrittweise Umstellung von alt auf neu. Stattdessen wird mit der Inbetriebnahme des neuen Systems, das Altsystem im gleichen Moment abgeschaltet.

Dabei ergeben sich substanzielle Risiken für eine erfolgreiche Migration (Vgl. Brodie/Stonebraker (1995), S. 8 ff.):

- Für zukünftig geringere Wartungs- und Betriebskosten wird kein Management ein neues System finanzieren. *Zusätzlicher Nutzen* in Form erweiterter Funktionalität müssen meist zugesichert werden. Damit vergrößert sich die Komplexität des neuen Systems und das Risiko eines Fehlschlags nimmt zu.

- Die *Geschäftsbedingungen* ändern sich ständig. Die Entwicklung größerer Informationssysteme dauert häufig mehrere Jahre an. Das Altsystem unterliegt währenddessen Änderungen durch Wartung, Ad-hoc-Anforderungen des Fachbereichs und so genannten ‚midnight functions', Funktionen, die die Wartungs- bzw. Entwicklungsmannschaft des Altsystems in ihrer freien Zeit in das System einfügt. Wesentlicher als der Einfluss durch Wartung oder kleinere Ad-hoc-Änderungen sind Änderungen des Geschäftsprozesses, den das System unterstützt. Gewöhnlicherweise unterliegen diese einem ständigen Wandel und führen nicht selten dazu, dass sich die Anforderungen an das neue System durchweg verändern. Mit zunehmender Abweichung steigt die Wahrscheinlichkeit eines Scheiterns.

- Häufig ist die *Dokumentation* eines Altsystems nur der vorhandene Programmcode. Die ursprünglichen Entwickler sind oft nicht mehr verfügbar und Dokumentationen sind entweder nicht existent, veraltet oder verloren gegangen. Es bleibt nur die Möglichkeit die Funktionen mühsam aus dem Code zu rekonstruieren, um sie im neuen System abbilden zu können. Das erhöht nicht nur die Komplexität, sondern erfordert einen höheren Zeitaufwand und führt damit zu steigenden Kosten der Entwicklung.

- Oft existieren nicht dokumentierte *Abhängigkeiten* zu anderen Systemen. Für die Migration unkritische Systeme wie Berichtswesen oder andere Informationssysteme greifen dabei auf Daten des Altsystems zu, um bestimmte Berichte zu erstellen oder Kennzahlen zu bilden. Das Vorhandensein dieser Abhängigkeiten ist nur selten den Verantwortlichen des Altsystems bekannt. Bei der Neuentwicklung müssen diese Abhängigkeiten jedoch berücksichtigt werden. Die Migration wird abermals komplexer und die Wahrscheinlichkeit von Fehlern wächst.

- Viele Altsysteme müssen nahezu zu 100% der Zeit verfügbar sein. Die Zeit für eine *Übertragung der Altdaten* in das neue System dauert je nach Umfang und Zugriffsmöglichkeit auf die Daten ungeachtet dessen bis zu mehreren Tagen. Dazu müssen die Daten häufig bereinigt und aufbereitet werden bevor sie übertragen werden können. Selbst wenn das neue System voll funktionsfähig ist, kann eine solche Situation den Cold-Turkey-Ansatz scheitern lassen, wenn es

keine Möglichkeit gibt, die Altdaten in vorgegebener Zeit ohne Einfluss auf den Betrieb des Altsystems zu migrieren.

- Die Schwierigkeit des *Managements großer Projekte* wird vielfach unterschätzt. Tendenziell wächst diese mit der Anzahl der beteiligten Mitarbeiter. Die Entwicklung eines größeren Informationssystems mit mehreren hundert Beteiligten stellt hohe Ansprüche an die Koordination und Aufteilung der Arbeit. Bei zusätzlicher Anforderung von Ressourcen ist die Einarbeitungszeit neuer Mitarbeiter ebenfalls ein wichtiger Faktor, der positiv oder negativ auf das Projekt wirken kann.

- Große Projekte verspäten sich unweigerlich angesichts der oben genannten Gründe. Die *Geduld des Managements* ist schnell erschöpft und nicht selten werden jene Projekte mit einem halbfertigen System beendet. Vor dem Hintergrund finanzieller Einschränkungen und Budgetkürzungen kann es schnell dazu kommen, dass kleine Reihen von Entwicklungspannen durch massive zeitliche Verzögerungen ganze Projekte zum Scheitern bringen.

- Die Beschaffung von Informationen, die nicht direkt mit dem Kern des Projektes in Verbindung stehen, führt tendenziell dazu, dass Großprojekte in ihrem Verlauf stark anwachsen und eine *handhabbare Größe überschreiten* können. Bei großen Migrationsprojekten wird möglicherweise die Einführung neuer Methoden und Technologien geprüft. Häufig erfolgt dies durch den Einsatz zusätzlicher Fachkräfte, die nicht kritisch in Bezug auf die Migration selbst sind, aber wiederum das Budget belasten. Das Management des Projektes wird somit komplexer und anfälliger für Fehler.

- Die *Homöostase* bezeichnet in der Systemtheorie das Verlangen und die Fähigkeit, das System innerhalb gewisser Grenzen stabil zu halten. Das führt automatisch zu einer Ablehnung all dessen, was diesen Zustand stören könnte. Bei einer Cold-Turkey-Migration führt dieser Umstand dazu, dass Ängste vor der Umstellung, neuen Methoden und Technologien hervorgerufen werden. Für betroffene Mitarbeiter ist es dann einfacher den Schwierigkeiten des Migrationsprozesses aus dem Weg zu gehen. Bevor die Komplexität und das hohe Risiko des Cold-Turkey-Ansatzes ihr Gleichgewicht stören könnten, geben sie sich lieber mit ihrem ineffizienten aber funktionierenden Altsystem zufrieden, da ihnen dieses System Sicherheit und Vertrauens bietet. Dieses Desinteresse und Inakzeptanz gegenüber allem Neuen verzögert dabei nicht nur den Migrationsprozess, sondern gefährdet sogar die Aussicht auf Erfolg des Projektes.

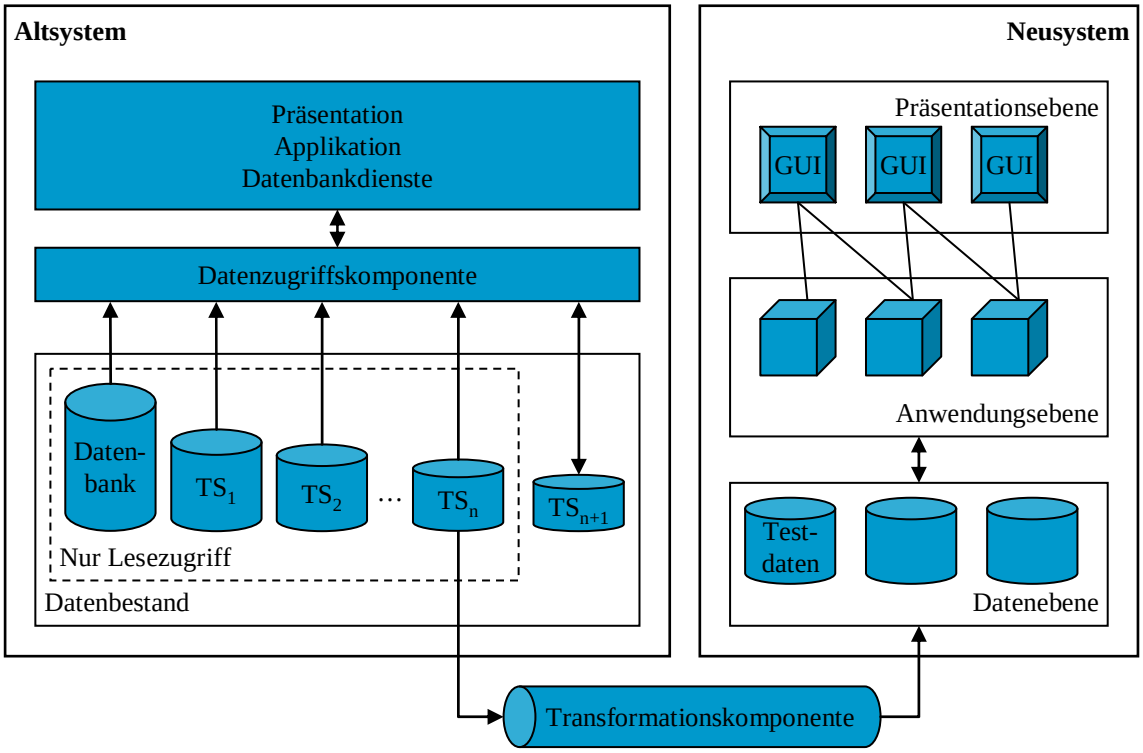
- Obwohl es oftmals angenommen wird, kann eine Cold-Turkey-Migration nicht erst dann beginnen, wenn das System und sein Umfeld in jeder Hinsicht erschlossen wurden. Große Informationssysteme sind häufig in ihrer Struktur so verwachsen, dass es unter Umständen unmöglich ist, sämtliche Finessen eines Systems zu erfassen. Daneben führt eine zu tiefe und detaillierte Erfassung zu einen extrem hohen Zeit- und Personalbedarf. Durch diese Bindung von Ressourcen kann es im schlimmsten Fall neben dem Zeitverzug zu einer *Lähmung* des Projektes kommen, da die Erfassung und Analyse bis ins kleinste Detail immer weiter fortgeführt werden kann. Wenn jedoch die Migration erst gar nicht begonnen wird, kann der Cold-Turkey-Ansatz folglich nicht erfolgreich beendet werden.

5.4 Butterfly

Der Butterfly-Ansatz beruht auf der Annahme, dass es sich um eine reine Datenmigration handelt und eine Zusammenarbeit zwischen Alt- und Neusystem nicht notwendig ist.

Während das Altsystem betriebsbereit bleibt, wird zunächst die Datenbank migriert. In einer Testumgebung wird anschließend das neue System entwickelt und erprobt, ohne dabei den alltäglichen Systembetrieb der Altanwendung zu beeinflussen oder gar zu stören. Die Butterfly-Methode benötigt jedoch im Gegensatz zum Chicken-Little-Ansatz keine Einführung von Gateways zwischen den Applikationsebenen. Mit Hilfe einer Datenzugriffskomponente und einer Transformationskomponente werden die Daten nach und nach übertragen. Während der Übertragung werden zwischenzeitliche Änderungen des Datenbestands in Temporärspeichern abgelegt, die ebenfalls in das Neusystem übernommen werden und erst mit dem finalen Abschalten des Altsystems enden. Der Zusammenhang ist in Abbildung 5.7 dargestellt.

Das Vorgehen lässt sich dabei in sechs Phasen unterteilen, wobei der Kernpunkt der Butterfly-Methode in der Phase 4 zu finden ist. Jede Phase enthält wiederum eine Menge von Aktivitäten, die unterschiedlichen Einfluss auf den Erfolg der Migration haben. Die Aktivitäten werden dabei teilweise parallel und wiederholt durchgeführt. Zum Teil ist es jedoch erforderlich, bestimmte Aktivitäten abzuschließen, um mit der nächsten Aktivität zu beginnen. Die Phasen hingegen werden sequentiell und jeweils nur einmal durchlaufen.



Quelle: In Anlehnung an Wu et al. (1999), S. 3.

Abb. 5.7: Butterfly-Ansatz

Phase 0 – Vorbereitung

Ist die Entscheidung für eine Migration erst einmal gefallen, werden im nächsten Schritt Vorbereitungen für das Projekt getroffen. Wesentliche Fragen sollen zu diesem Zeitpunkt geklärt werden. Dabei werden Benutzeranforderungen festgelegt und Erfolgsfaktoren definiert. Neben der Bereitstellung der notwendigen Hardware steht die Konzeption der Systemarchitektur im Mittelpunkt.

Phase 1 – Untersuchung des Altsystems und Entwicklung des Datenmodells

Für die Migration ist es mitunter entscheidend, das Altsystem verstanden zu haben. Dazu müssen Schnittstellen, Applikation und Datenhaltung erkannt und erfasst werden. Redundanzen werden identifiziert, um den Prozess und das System zu verschlanken. Im nächsten Schritt werden Schnittstellen festgelegt und die Funktionalität des neuen Systems beschrieben. Des Weiteren wird der Umfang der zu migrierenden Daten bestimmt. Falls Wechselwirkungen mit anderen Systemen bestehen, werden diese identifiziert und dokumentiert, um sie zu prüfen und in der Entwicklung des neuen Systems zu berücksichtigen. Neben dem Datenschema wird die Datenzugriffskomponente DAA entwickelt, die

während des Migrationsprozesses für den Datenzugriff im Altsystem verantwortlich ist und eine Koordinations- und Verwaltungsfunktion für aktuellste Datenbestände erfüllen muss. Zum Abschluss der Phase werden Regeln für das Abbilden und Übertragen der Daten, das so genannte ‚Mapping', festgelegt.

Phase 2 – Testdaten bereitstellen

Um die Entwicklung zu beschleunigen, werden keine Testdaten generiert, sondern direkt aus der Datenbank des Altsystems ausgewählt. So werden in der Entwicklung anhand reeller Daten Fehler schneller identifiziert und bereits in einer sehr frühen Projektphase behoben. Die Transformationskomponente wird anhand der Vorgaben der vorherigen Projektphasen entwickelt und mit Hilfe der Beispieldaten getestet. Diese Daten werden in einer Testdatenbank im Neusystem abgelegt und stehen nicht nur den Entwicklern zur Verfügung, sondern gestatten zu einem späteren Zeitpunkt Schulungen anhand realer aber nicht operativer Daten durchzuführen. Dieses Vorgehen vermeidet zum einen Fehler in der Entwicklung und führt zum anderen dazu, dass Fallstudien in Schulungen direkt auf existenten Datensätzen aus dem täglichen Arbeitsablauf aufbauen können, was wiederum zu einer verbesserten Anwendung der Erkenntnisse aus der Schulung in der Praxis führt.

Phase 3 – Schrittweise Migration der Systemkomponenten (bis auf Daten)

Neben den Benutzer- und Systemschnittstellen wird gleichzeitig die Applikationsebene des Neusystems schrittweise entwickelt und fortlaufend anhand der Beispieldaten aus Phase 2 wiederholt getestet. Dabei werden gegebenenfalls wieder verwendbare Komponenten des Altsystems genutzt, um das Projekt zu beschleunigen und Fehler zu vermeiden. Nach erfolgreicher Entwicklung der Komponenten werden diese in einem nächsten Schritt in die Systemarchitektur des Neusystems integriert. Anschließend werden die Komponenten nochmals getestet und gegen die Anforderungen aus den ersten Projektphasen geprüft. Alternativ kann bereits mit der Schulung der Anwender anhand bestehender Systemteile begonnen werden.

Phase 4 – Schrittweise Datenmigration und Schulung der Anwender

Die Datenzugriffskomponente aus Phase 1 wird in das Altsystem eingeführt und steuert von nun an sämtlichen Lese- und Schreibzugriff auf den Datenbestand. Neben der vorhandenen Datenbank werden mehrere Temporärspeicher (TS) in

der Datenebene des Altsystems eingerichtet. Die bestehende Datenbasis wird mit einem Schreibschutz versehen. Änderungen am Datenbestand werden fortan im Temporärspeicher TS_1 abgelegt. Unveränderte Daten werden weiterhin von der Ursprungsdatenbank abgerufen. Der Zugriff auf aktualisierte Daten ist über den Temporärspeicher TS_1 sichergestellt.

Anschließend wird der Datenbestand des Altsystems über die Transformationskomponente in das neue System übertragen. Dabei wird das Datenformat mit Hilfe von Abbildungstabellen in das neue Datenmodell überführt. Die Abbildungstabellen enthalten Regeln für die eindeutige Zuordnung von Daten basierend auf den Datenmodellen der beiden Systeme.

Ist die Migration der Datenbasis abgeschlossen, wird der Temporärspeicher TS_1 mit einem Schreibschutz versehen. Fortan werden Änderungen des aktuellen Datenbestands im Temporärspeicher TS_2 abgelegt. Nun beginnt die Überführung der Daten aus TS_1 mit Hilfe der Transformationskomponente in das neue System. Wenn der Vorgang beendet ist, wird TS_2 schreibgeschützt und migriert. Aktualisierungen des Datenbestandes werden dann in TS_3 gespeichert. Die Folge setzt sich fort, bis die Terminierungsbedingung greift.

Kerngedanke dabei ist, dass sich die Größe des benötigten Temporärspeichers kontinuierlich verringert, da der Zeitaufwand für die Migration der Daten mit der Größe der Temporärspeicher abnimmt und somit weniger Zeit für Änderungen am Datenbestand vergeht. Die Terminierungsbedingung legt einen Schwellenwert fest, der beispielsweise die Größe des Temporärspeichers beschreibt, der während der Migration der Daten gefüllt wird.

Unterschreitet nun der Temporärspeicher TS_{n+1} diesen Schwellenwert nach Migration des Temporärspeichers TS_n, wird der gesamte Datenbestand des Altsystems für Benutzerzugriffe gesperrt und die Daten aus TS_{n+1} in das Neusystem übertragen.

Parallel zu den genannten Aktivitäten werden spätestens in dieser Phase die Anwender für das System geschult und vorbereitet.

Phase 5 – Umstellung auf neues System

Die Datenbestände der beiden Systeme sind durch initiale und fortwährende Übertragung des Datenbestandes und aller Änderungen konsistent. Das Neusystem steht nun vollständig zum Einsatz bereit. Durch finales Abschalten des Altsystems und Nutzung des Neusystems wird die letzte Phase

abgeschlossen und die Butterfly-Methode erfolgreich beendet (Vgl. Wu et al. (1999), S. 2 ff.).

Die Vorteile des Butterfly-Ansatzes liegen zum einen in der durchgängigen Verfügbarkeit des Altsystems. Bis auf den Zeitraum des Umschaltens, der durch intelligente Wahl des Schwellenwertes minimiert werden kann, ist das System ständig einsatzbereit und ermöglicht so die tägliche Arbeit an den Daten. Zum anderen kann zu jedem beliebigen Zeitpunkt vor dem Umschalten auf das neue System der Migrationsprozess abgebrochen werden, da die Migration umkehrbar ist, solange die Daten über die Datenzugriffskomponente gelaufen sind. Für den Fall eines Abbruchs der Migration müssen lediglich alle Temporärspeicher nacheinander in den ursprünglichen Datenbestand des Altsystems eingebunden und die Datenzugriffskomponente entfernt oder außer Kraft gesetzt werden.

Je nach Datenaufkommen und Aktivität im Altsystem können jedoch sehr viele und große Temporärspeicher notwendig sein. Im Extremfall führt das zu einem sehr hohen Speicherbedarf und Ressourcenverbrauch. Die Entwicklung der Datenzugriffskomponente stellt ein weiteres Problem dar. Im Gegensatz zum Chicken-Little-Ansatz werden zwar keine Gateways benötigt, jedoch kann auch die Datenzugriffskomponente einen sehr hohen Entwicklungsaufwand erfordern.

Die vorgestellten Strategien zur Systemablösung stellen durchaus unterschiedliche Möglichkeiten dar, ein System durch ein anderes zu ersetzen. Es lassen sich dabei grundsätzlich zwei Ansätze unterscheiden, das stufenweise Vorgehen und der Übergang in einem einzigen Schritt.

Bei allen Strategien wird von einer gleichzeitigen Softwareentwicklung ausgegangen. Dieser Fall trifft für den Ausgangspunkt der Diplomarbeit nicht zu. Abgesehen davon, lassen sich die Strategien jedoch im Kern in einem Vorgehensmodell anwenden, da sie durch verschiedene Herangehensweisen alternative Abläufe darstellen und damit eine Reaktion auf verschiedene Ausgangssituationen ermöglichen.

6 Vorgehensmodell zur Systemablösung

Um einen reibungslosen Übergang aus einem abzulösenden Altsystem in ein neues System zu gewährleisten, ist eine Reihe von Maßnahmen erforderlich. Dieses Vorgehensmodell soll dazu dienen, erforderliche Schritte aufzuzeigen und betroffene Organisationseinheiten einzubeziehen. Durch die Einteilung in Phasen bekommt das Vorgehen eine solide strukturierte Basis, die neben der Definition von Meilensteinen den Grundstein zum Projekterfolg legt.

Das Vorgehensmodell wird in Form einer ausführlichen Prozessbeschreibung dargestellt. Es soll als Vorlage für das Managementhandbuch der Volkswagen AG dienen, um einen gültigen Standard für Systemablösungen kleinerer und mittelgroßer Systeme zu schaffen. Das Managementhandbuch besteht neben den Prozessbeschreibungen aus der Beschreibung der festgelegten Politik und Ziele, zutreffender Normen und weiteren ergänzenden Informationen. Darüber hinaus finden sich zu den Prozessbeschreibungen entsprechende Arbeitsanweisungen, die eine detaillierte Form der Tätigkeiten darstellen (Vgl. Volkwagen (2007e), S. 4).

6.1 Zweck

Diese Prozessbeschreibung beinhaltet ein Vorgehensmodell zur Systemablösung. Es stellt im Wesentlichen den Ablauf der Einführung eines neuen Systems und die damit einhergehende Ablösung eines Altsystems dar. Dabei wird die Übernahme bestehender Daten aus dem Altsystem berücksichtigt.

6.2 Anwendungsbereich

Die vorliegende Systematik ist für eine Systemablösung kleiner bis mittelgroßer Systeme anzuwenden. Voraussetzung ist die Bereitstellung eines neuen Systems, welches bereits in einem vorgelagerten Prozess ausgewählt wurde und die für den Geschäftsprozess benötigte Funktionalität bietet. Dabei kann sowohl auf Standardsoftware, als auch auf Individualsoftware zurückgegriffen werden.

Das Vorgehensmodell bietet die Möglichkeit kleinere Anpassungen an der gewählten Software vorzunehmen. Grundsätzlich muss jedoch gewährleistet sein, dass das gewählte Produkt zum gewünschten Anwendungsbereich passt. Dazu wurde bereits in einem vorgelagerten Prozess eine entsprechende Analyse des Altsystems und des Geschäftsprozesses durchgeführt. Anschließend wurde aus einer Reihe von alternativen

Produkten auf Grund bestimmter Kriterien eine Software ausgewählt. Dabei kann sowohl Standardsoftware, als auch eine final entwickelte Individualsoftware zum Einsatz kommen (Vgl. Kapitel 2.2).

Im Rahmen dieses Vorgehensmodells wird davon ausgegangen, dass die Systemablösung in Absprache mit der Geschäftsleitung durchgeführt wird. Eine Kosten-Nutzen-Analyse ist bereits im Zuge der Entscheidung für die Systemablösung zu dem Ergebnis gekommen, dass es sowohl ökonomisch als auch aus langfristiger Unternehmenssicht sinnvoll ist, das Altsystem durch das ausgewählte System zu ersetzen. Die Ursachen einer Systemablösung können dabei unterschiedlichen Einfluss auf Zeitpunkt, Umfang, sowie das Budget haben (Vgl. Kapitel 4).

6.3 Zuständigkeit

An der Systemablösung sind verschiedene Organisationseinheiten beteiligt, die im Verlauf der Planungsphase identifiziert und dokumentiert werden. Grundsätzlich ist jedoch der Fachbereich mit den Hauptanwendern für den fachlichen Input im Verlauf des Projektes verantwortlich, ebenso wie bei Entscheidungen, die grundsätzlich eine fachliche Unterstützung erfordern. Die Projektleitung trägt die Verantwortung für das Projekt bis zu dessen Abschluss und ist für die Durchführung aller Phasen, sowie für die Information beteiligter Organisationseinheiten verantwortlich.

6.4 Mitgeltende Unterlagen

Unterlagen, die zur Unterstützung des Prozesses und der Definition von Rahmenbedingungen dienen, sind für die Systemablösung noch nicht definiert worden. Sie müssen bei der Aufnahme in das Volkswagen Management Handbuch für Qualität bestimmt werden.

6.5 Beschreibung

Das Vorgehensmodell besteht aus einer Reihe von Prozessschritten, die in fünf einzelne Phasen zusammengefasst werden, die in Abbildung 6.1 dargestellt sind. Eine Zuordnung der Prozessschritte zu den Phasen lässt sich aus deren Bezeichnung ableiten. Der erste Teil charakterisiert dabei die Phase, der zweite Teil den Schritt innerhalb der

Phase. Der Prozessschritt 2.4 bezeichnet somit den vierten Prozessschritt in der zweiten Projektphase.

Die Abfolge der Phasen und Prozessschritte ist in der Regel sequentiell, da Ergebnisse am Ende einer Phase oder eines Prozessschrittes in den meisten Fällen die Eingabe für das nachfolgende Element bilden.

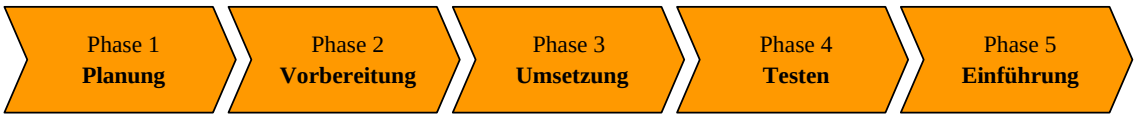


Abb. 6.1: Phasen des Vorgehensmodells

Jedes erneute Durchlaufen dieses Vorgehensmodells wird als eigenständiges IT-Projekt betrachtet. Durch das Vorgehensmodell werden eine standardisierte Ausführung und der Einsatz einheitlicher Verfahren ermöglicht (Vgl. Wieczorrek/Mertens (2007), S. 9). Jedoch bietet dieses Vorgehensmodell eine Reihe von Alternativen, um auf Basis verschiedener Ausgangspositionen und Anforderungen durchgeführt werden zu können.

6.5.1 Planung

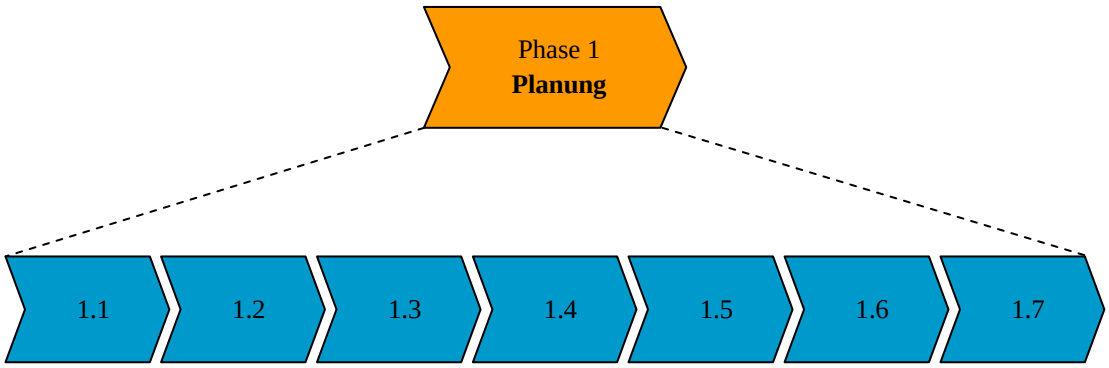


Abb. 6.2: Phase 1 – Planung

Die *Planungsphase* dient in erster Linie einer grundlegenden Orientierung und Planung, aber auch zur Abgrenzung des Projektes. Sie besteht aus sieben aufeinander folgenden Prozessschritten, die sich an der Planung eines Projektes orientieren (Vgl. Kapitel 2.1). Das Ziel der ersten Phase liegt darin, einen Konsens zwischen allen Projektbeteiligten zu erreichen, da sie zum Teil unterschiedliche Schwerpunkte sehen. Dazu werden gemeinsam Ziele und Inhalte des Projektes klar definiert, um Entscheidungen planen und treffen zu können. Eine passende Projektorganisationsform wird ausgewählt und Zuständigkeiten festgelegt. Es wird ein Projektplan erstellt, der zum einen wichtige Information über den zeitlichen Ablauf enthält und zum anderen Auskunft über

benötigte Ressourcen im gesamten Projektverlauf gibt. Zur finanziellen Kontrolle wird ein Kostenplan erstellt. Eine Risikoabschätzung verringert die Wahrscheinlichkeit ungeplanter Ereignisse (Vgl. Wieczorrek/Mertens (2007), S. 62 f.). Die Abbildung 6.2 und die Tabelle 6.1 geben einen Überblick über die Planungsphase.

Tab. 6.1: Schritte der Phase 1 – Planung

Schritt	Bezeichnung
1.1	Situationsbeschreibung
1.2	Ziel- und Aufgabendefinition
1.3	Rahmenbedingungen
1.4	Projektorganisation
1.5	Ressourcen- und Zeitplanung
1.6	Kostenplanung
1.7	Risikoanalyse

6.5.1.1 Situationsbeschreibung

Grundlage jedes Projektes ist die Initialisierung. Dazu wird mit Hilfe einer *Situationsbeschreibung* Umfang und Komplexität des IT-Systems festgehalten und Informationen über angrenzende Systeme ermittelt. Dabei werden Angaben und Abgrenzungen zu Teil- und Untersystemen gesammelt und der Ist-Zustand des Projektumfelds festgehalten. So soll ein gleiches Grundverständnis aller Projektbeteiligten gewährleistet und Missdeutungen ausgeschlossen werden. Eine klare Formulierung der aktuellen Situation und die Festschreibung der Anforderungen des Fachbereichs an das Projekt bzw. System bilden den Ausgangspunkt für die Systemablösung. Eine detaillierte Beschreibung des Migrationsanlasses kann weitere nützliche Informationen für die Projektplanung ergeben. Durch die Identifikation betroffener Organisationseinheiten und Standorte wird eine weitere Abgrenzung des Projektes vorgenommen. Gleichzeitig werden die Verantwortlichkeiten der Organisationseinheiten dokumentiert. Am Ende von Schritt 1.1 sollen einführende Informationen zum Ist-Zustand ausführlich festgehalten werden, um eine solide Basis für die weiteren Schritte zum Projekterfolg zu bilden. Dabei soll möglichst sorgfältig und umfassend dokumentiert werden, denn Fehler im Anfangsstadium eines Projektes können Auswirkungen auf die gesamte Projektlaufzeit haben (Vgl. Wieczorrek/Mertens (2007), S. 55 f.).

6.5.1.2 Ziel- und Aufgabendefinition

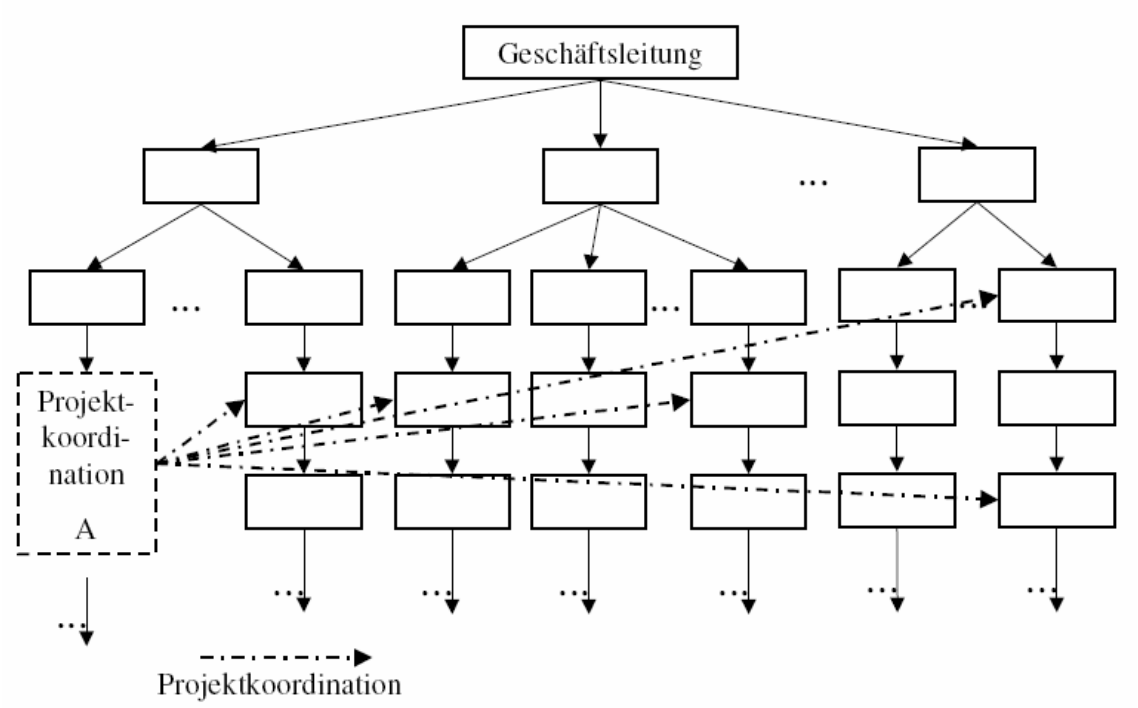
Mit Hilfe der Beschreibung des Ist-Zustandes wird nun der Soll-Zustand definiert. Zunächst werden die *Ziele* des Projektes in Form kritischer Erfolgsfaktoren definiert. Ohne diese Festlegung gibt es keine klaren Vorgaben für den Projekterfolg, d.h. es wäre nicht ersichtlich, wann ein Projekt erfolgreich ist und wann nicht. Mit der Erreichung aller kritischen Erfolgsfaktoren werden die Projektziele und damit das erfolgreiche Ende des Projektes erreicht. Ebenso kann der aktuelle Stand des Projektes daran gemessen werden. Des Weiteren sollen in diesem Schritt bereits die *Aufgaben* identifiziert werden, die zur Erreichung der Ziele notwendig sind, sofern sie nicht Bestandteil der Arbeitsanweisungen im Managementhandbuch sind. Die Identifikation von Aufgaben dient dabei in erster Linie zur Strukturierung des Projektes.

6.5.1.3 Rahmenbedingungen

In dem folgenden Schritt werden Rahmenbedingungen und Restriktionen dokumentiert. Sie werden dabei nach internen und externen Faktoren klassifiziert und grenzen so das Projektumfeld weiter ab. Externe Restriktionen bedeuten Vorgaben von außen, bspw. Vorschriften und Gesetze. Interne Restriktionen sind durch die Unternehmensstrategie und interne Vorgaben bestimmt. Die Konkurrenzverhältnisse des Marktes und die Marktkonstellation werden in den externen Rahmenbedingungen festgehalten. Die Komplexität oder die Anzahl von Iterationen innerhalb der Projektschritte sind Beispiele für interne Rahmenbedingungen, die einen Einfluss auf das Ablösungsprojekt haben können (Vgl. Wieczorrek/Mertens (2007), S. 56).

6.5.1.4 Projektorganisation

Erste Basisinformationen zum Projekt sind gesammelt. Eine Zuordnung von Aufgaben zu Organisationseinheiten, sowie Definition von Teilprojekten erfolgt mit Hilfe der *Projektorganisation (PO)*. Dabei werden Verantwortlichkeiten und strukturelle Zusammenhänge transparent dargestellt. Grundsätzlich werden in der Literatur drei verschiedene Organisationsformen für IT-Projekte unterschieden. Nachfolgend werden die Einfluss-Projektorganisation, die reine Projektorganisation und die Matrix-Projektorganisation vorgestellt.



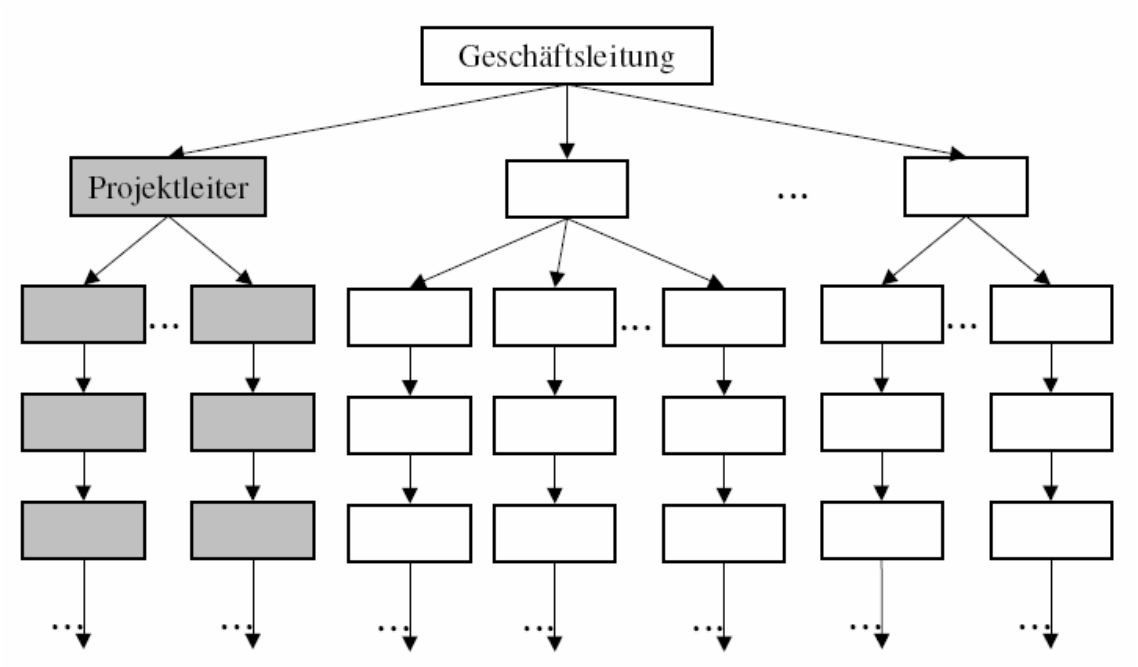
Quelle: Wieczorrek/Mertens (2007), S. 27.

Abb. 6.3: Einfluss-Projektorganisation

Die Einfluss-Projektorganisation ist auch als Stab-Linien-Organisation bekannt. Es wird keine selbstständige aufbauorganisatorische Einheit gebildet. Wie in Abbildung 6.3 dargestellt, bleiben die Mitglieder der Projektteams funktionell und personell dem jeweiligen (Linien-)Vorgesetzten unterstellt, lediglich der Projektleiter besetzt eine Stabsstelle und koordiniert das Projekt. Da keine eigene Projektorganisation ausgebildet wird, bleibt die Unternehmensstruktur unverändert. Die Steuerung der Mitarbeiter erfolgt über deren Vorgesetzten. Entscheidungen werden ebenfalls in der Linie, d. h. zwischen Vorgesetzten und Mitarbeitern getroffen. Der Projektleiter ist für den sachlichen und terminlichen Ablauf des Projektes verantwortlich. Er informiert den Auftraggeber und Entscheidungsgremien über den aktuellen Projektstand, sowie jegliche Abweichungen von der Planung und erstellt Entscheidungsvorlagen. Letztlich nimmt der Projektleiter eine koordinierende, berichtende und beratende Rolle wahr, dessen Kompetenzen und Weisungen demnach stark eingeschränkt sind. Einfluss und Durchsetzungsvermögen des Projektleiters sind die wesentlichen Faktoren für den Erfolg des Projektes (Vgl. Wieczorrek/Mertens (2007), S. 26 f.).

Der Einsatz der Einfluss-Projektorganisation bietet den Vorteil, dass nur geringe organisatorische Änderungen notwendig sind. Der Personaleinsatz gestaltet sich durchaus flexibel und Projektmitarbeiter können parallel an mehreren Projekten beteiligt sein. Jedoch ist die Identifikation der Mitarbeiter mit dem Projekt gerade bei

einer Vielzahl von Tätigkeiten eher gering. Mehrere parallele Projekte erfordern einen hohen Koordinationsaufwand der Aufgaben, um eine rechtzeitige und vollständige Bearbeitung zu gewährleisten. Verzögerungen sind durch die Verteilung auf mehrere Abteilungen und den begrenzten direkten Einfluss des Projektleiters wahrscheinlich. Dadurch entstehen häufig Abweichungen vom ursprünglichen Projektplan (Vgl. Wieczorrek/Mertens (2007), S. 27 f.). Auf Grund des hohen Koordinationsaufwands und der nicht vorhandenen eigenständigen Aufbauorganisation ist die Einfluss-Projektorganisation nur für kleinere, unkritische Projekte geeignet.



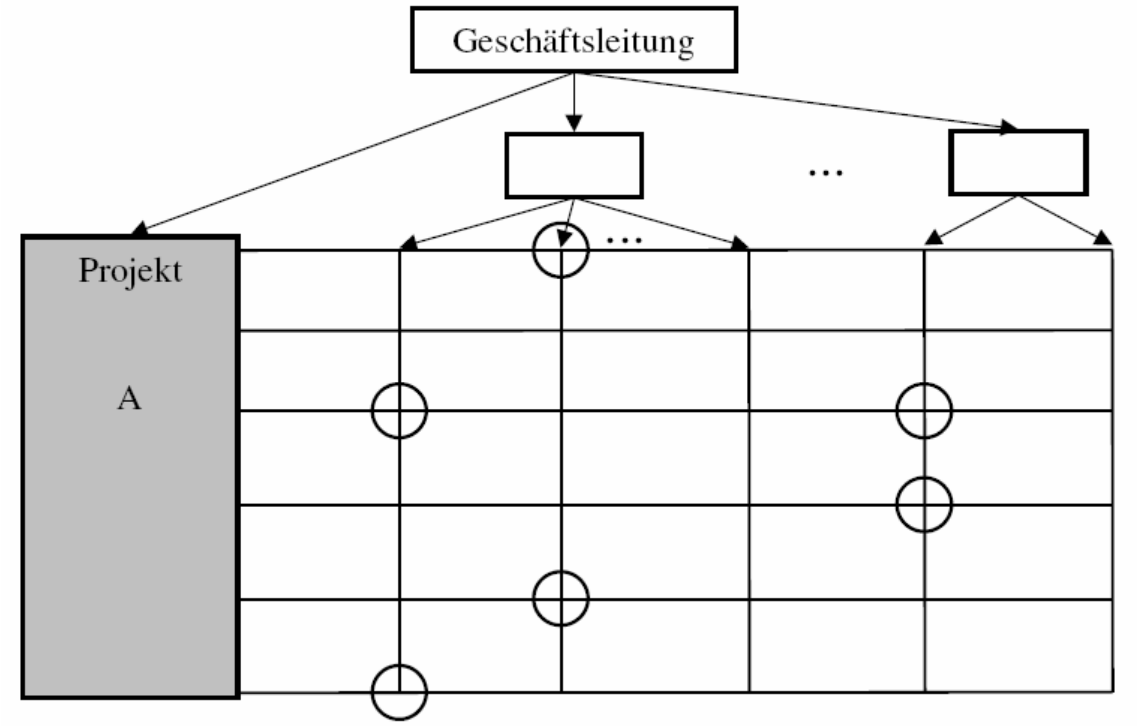
Quelle: Wieczorrek/Mertens (2007), S. 28.

Abb. 6.4: Reine Projektorganisation

Bei Verwendung der reinen Projektorganisation wird eine selbstständige Organisationsform ausgebildet und in die Organisationsstruktur des Unternehmens linienförmig, wie in Abbildung 6.4 dargestellt, eingebunden. Sie ist daher auch als Linien-Projektorganisation bekannt. Interne Projektmitarbeiter werden dabei für die Dauer des Projektes aus ihren bisherigen Positionen innerhalb der Unternehmenshierarchie herausgelöst und in die Projektorganisation eingegliedert. Sie werden von ihren gegenwärtigen Aufgaben befreit, um ihre gesamte Kraft in das Projekt einzubringen. Externe Dienstleister können direkt in die Projektorganisation eingeordnet werden. Sie unterstützen je nach Bedarf die Projektarbeit und stellen somit für jeden Zeitpunkt die benötigte Kapazität an Fachkräften zur Verfügung. Der Projektleiter führt unmittelbar das Projektteam und hat direkten Einfluss auf die Mitarbeiter. Er allein übernimmt neben der fachlichen Projektverantwortung, die

Verantwortung für die Erreichung aller Projektziele. Dazu besitzt er die Verfügungsgewalt über sämtliche Projektressourcen (Vgl. Wieczorrek/Mertens (2007), S. 28 f.).

Die Vorteile der reinen Projektorganisation liegen in der klaren Zuordnung von Zuständigkeiten. Die Projektmitarbeiter sind nur in ein Projekt eingebunden, identifizieren sich damit und fühlen sich verantwortlich. Die Arbeitsleistung wird vollständig für das Projekt ausgeschöpft. Projektbezogene Entscheidungen können auf Grund der eigenständigen Aufbauorganisation schnell getroffen werden. Nachteilig ist hingegen, dass die Projektmitarbeiter bei Projektbeginn und Projektabschluss aus der Unternehmenshierarchie aus- bzw. eingegliedert werden müssen. Daneben entstehen für den Aufbau der eigenständigen Organisationsform Aufwände und Kosten. Die Verwendung der reinen Projektorganisation empfiehlt sich besonders für Vorhaben, in die eine geringe Anzahl von Mitarbeitern über einen längeren Zeitraum eingebunden ist (Vgl. Wieczorrek/Mertens (2007), S. 29).



Quelle: Wieczorrek/Mertens (2007), S. 30.

Abb. 6.5: Matrix-Projektorganisation

Die Matrix-Projektorganisation stellt eine Kreuzung der reinen PO und der Einfluss-Projektorganisation dar. Die Vorteile beider Formen werden dadurch vereint. Neben der vorhandenen Organisationsstruktur des Unternehmens wird eine Matrix-

Projektorganisation aufgebaut. Die Projektmitarbeiter werden dabei nicht aus ihren bisherigen Positionen ausgegliedert und müssen somit nach Projektabschluss nicht wieder eingegliedert werden. Sie bleiben ihrem Vorgesetzten personell zugeordnet. Zur Durchführung von Projektarbeiten sind sie jedoch fachlich dem Projektleiter unterstellt. So können sie anteilig am Projekt arbeiten und weiterhin das Tagesgeschäft unterstützen. Der Projektleiter verfügt also für die Dauer des Projektes über projektbezogene Weisungsrechte, die in einem Mehrliniensystem, wie in Abbildung 6.5, grafisch als Matrix dargestellt werden können. Die Verantwortlichkeiten sind in der Matrix-Projektorganisation klar zugeordnet. Der Projektleiter ist dabei für den termingerechten Projektverlauf und die Einhaltung des Budgets zuständig. Die Umsetzung der fachlichen Inhalte liegt hingegen bei den Projektmitarbeitern, die gegebenenfalls durch externe Dienstleister unterstützt werden können. Da die Unternehmenshierarchie nicht geändert wird, lassen sich mehrere Projekte parallel durchführen. Schwierigkeiten können dann entstehen, wenn sich Interessen der Vorgesetzten und des Projektleiters widersprechen. In diesem Fall sind die Konflikte zu lösen oder zu eskalieren. Die Verwendung der Matrix-Projektorganisation ist besonders vorteilhaft, wenn die für das Projekt benötigten Wissensträger nur zeitlich begrenzt benötigt werden. Das Spezialwissen der Mitarbeiter fließt somit optimal in das Projekt ein und Ressourcen werden nur so lang gebunden, wie sie tatsächlich benötigt werden (Vgl. Wieczorrek/Mertens (2007), S. 29 ff.).

Die vorgestellten Projektorganisationsformen sollen als Vorlage für das Systemablösungsprojekt gelten. Eine Auswahl erfolgt je nach Gegebenheiten und kann anhand verschiedener Faktoren entschieden werden. In der Tabelle 6.1 sind dazu verschiedene Kriterien und deren Ausprägung in den einzelnen Projektorganisationen angeführt. Die Wahl der PO hat verschiedene Einflüsse auf das Projekt.

Die Steuerungsmöglichkeit des Projektleiters hängt direkt von der gewählten Projektorganisationsform ab. Im Fall einer reinen PO hat der Projektleiter den größten Handlungsspielraum, denn er ist in einer eigenständigen Aufbauorganisation unmittelbar handlungsführend. Bei einer Matrix-PO sind die Möglichkeiten weitaus eingeschränkter, da der Projektleiter und der Vorgesetzte bei der Führung der Mitarbeiter aufeinander treffen. Im Fall einer Einfluss-PO hat der Projektleiter lediglich indirekte Weisungsbefugnisse und damit die geringsten Steuerungsmöglichkeiten. Der Grad der Verantwortung des Projektleiters ist direkt von der Steuerungsmöglichkeit abhängig, denn es kann nur verantwortet werden, was auch beeinflusst werden kann. Die Verantwortung des Projektleiters ist demnach bei der reinen PO am größten und nimmt über die Matrix-PO zur Einfluss-PO deutlich ab (Vgl. Wieczorrek/Mertens (2007), S. 31).

Tab. 6.2: Formen der Projektorganisation

Kriterien	**Einfluss-PO**	**Reine PO**	**Matrix-PO**
Bedeutung für das Unternehmen	gering	sehr groß	groß
Umfang des Projektes	gering	sehr groß	groß
Unsicherheit der Zielerreichung	gering	hoch	mittel
Technologie	Standard	neu	kompliziert
Projektdauer	kurz	lang	mittel/lang
Komplexitätsgrad	gering	hoch	mittel
Bedürfnis nach zentraler Steuerung	gering	sehr groß	groß
Mitarbeitereinsatz	nebenamtlich	hauptamtlich	Teilzeit
Anforderungen an die Projektleiterpersönlichkeit	hohe Anforderungen an die Persönlichkeit	hochqualifizierter Projektleiter mit guten Methoden- und Fachkenntnissen	hochqualifizierter Projektleiter mit guten Methodenkenntnissen

Quelle: Wieczorrek/Mertens (2007), S. 32.

Ein Wechsel der PO während des Projektes ist möglich, jedoch sollte der Aufwand für die Umstellung berücksichtigt werden. Daneben sind auch Mischformen denkbar. Ein Stamm von Projektmitarbeitern kann demnach zeitweise durch unternehmensinterne Spezialisten unterstützt werden, einer Kombination aus reiner PO und Matrix-PO (Vgl. Wieczorrek/Mertens (2007), S. 31).

Für die Systemablösung wird eine Projektorganisation benötigt, die über eine solide Struktur verfügt und Zuständigkeiten transparent darstellt. Aufgaben und Kompetenzen sind dabei entsprechend der Fachbereichszugehörigkeit zuzuordnen. Je nach Bedarf sind Teilprojekte zu bilden, die einen abgegrenzten Umfang von Aufgaben in Form von Arbeitspaketen enthalten. Die Teilprojekte sollten dabei so gewählt werden, dass sie möglichst unabhängig voneinander bearbeitet werden können. Neben der Projektleitung sind das Projektcontrolling und das Qualitätsmanagement als Querschnittsfunktion über das gesamte Systemablösungsprojekt zu führen. Eine Umsetzung in Form einer Matrix-PO ist in Abbildung 6.6 dargestellt.

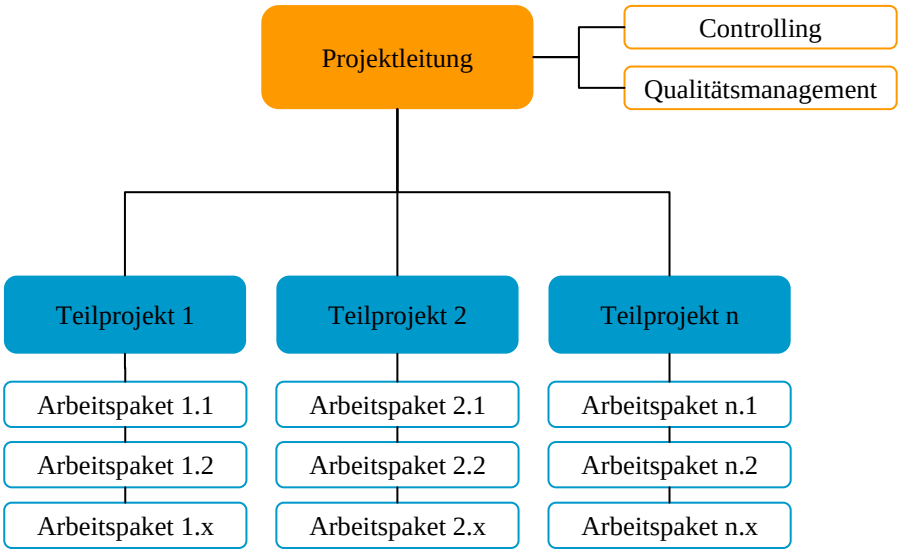


Abb. 6.6: Projektaufbauorganisation

6.5.1.5 Ressourcen- und Zeitplanung

In einem nächsten Projektschritt, der *Ressourcen- und Zeitplanung*, wird festgelegt, welche Sachmittel und Personalkapazitäten zu welchen Zeiten benötigt werden. Die Ergebnisse der Ressourcenplanung können nur selten genau ermittelt werden, da sie einen Verbrauch in der Zukunft vorhersagen müssen und sich häufig bestimmter Schätzmethoden bedienen. Eine gewisse Unsicherheit ist daher mit einzuplanen (Vgl. Zimmermann et al. (2006), S. 46).

Eine Planung der Ressourcen wird auf Basis der Arbeitspakete durchgeführt, die im vorherigen Projektschritt definiert worden sind. Dazu werden die einzelnen Bedarfe an Sachmitteln und Personalressourcen ermittelt, die zur Realisierung der Arbeitspakete notwendig sind. Dabei ist zu berücksichtigen, welche Kapazitäten der einzelnen Ressourcen zur Verfügung stehen, denn einzelne Ressourcen stehen möglicherweise nur begrenzt zur Verfügung. Mit der Erstellung von Auslastungsdiagrammen wird der Grad der Auslastung einzelner Ressourcen transparent dargestellt und eine effektive Nutzung ermöglicht. Die Kumulierung der Bedarfe der einzelnen Arbeitspakete bildet schließlich die Gesamtbedarfsplanung der Sachmittel- und Personalressourcen (Vgl. Wieczorrek/Mertens (2007), S. 100 f.).

Für die Planung der Ressourcen wird zwischen Sachmitteln und Personal unterschieden. Bei der Personalressourcenplanung wird neben internen Kapazitäten häufig auf externe Dienstleister zurückgegriffen. Sowohl intern, als auch extern stehen Spezialisten für die einzelnen Projektaufgaben nur in begrenzter Anzahl zur Verfügung. Intern sind die

Kapazitäten deshalb begrenzt, weil die Mitarbeiter häufig für Steuerungsprozesse innerhalb des Unternehmens eingesetzt sind oder Hauptaufgaben im täglichen Geschäft verrichten. Sie sind mit den betrieblichen Abläufen des Unternehmens sehr gut vertraut. Im Gegensatz dazu sind externe Dienstleister nicht mit den unternehmensspezifischen Vorgängen vertraut. Dafür bieten sie durch ihre weit reichende Erfahrung und objektive Sichtweise vielfältige Lösungsmöglichkeiten für Problemstellungen. Jedoch sind externe Dienstleister in der Regel kostenintensiv. Eine Einarbeitungszeit sollte berücksichtigt werden. Bei der Planung der Personalressourcen müssen qualitative und quantitative Anforderungen den vorhandenen Kapazitäten gegenübergestellt werden, um eine effiziente Planung zu gewährleisten. Dabei ist zu beachten, dass Personen nicht 52 Wochen zu je 5 Tagen und 8 Stunden zur Verfügung stehen. Durch Urlaub, Krankheit und Fortbildungsmaßnahmen ist mit einer Verfügbarkeit von 180 bis 220 Arbeitstagen zu rechnen. Je größer die Unabhängigkeit der einzelnen Arbeitspakete ist, desto eher lassen sich Arbeitspakete sequentiell abarbeiten und Kapazitäten verteilen. Jedoch beeinflusst dieses Vorgehen die Projektdauer (Vgl. Wieczorrek/Mertens (2007), S. 101 ff.).

Die Sachmittelplanung befasst sich mit allen nicht personenbezogenen Ressourcen. Grundsätzlich ist dabei eine Unterscheidung in verzehrbare und nicht verzehrbare Einsatzmittel vorzunehmen. Verzehrbare Sachmittel sind klassische Gebrauchsgüter, bspw. Rohstoffe und Materialien, die nur einmal verwendet und nach ihrem Gebrauch nicht mehr zur Verfügung stehen. Sie werden auch als nicht erneuerbare Ressourcen bezeichnet. Der Verbrauch finanzieller Mittel kann als verzehrbares Sachmittel eingestuft werden, da ein aufgebrauchtes Budget für weitere Vorgänge nicht mehr zur Verfügung steht. Verzehrbare Mittel sind vor allem bei der Kostenplanung von Interesse, da durch vermehrten Einsatz dieser Ressourcen einzelne Vorgänge beschleunigt werden können. Nicht verzehrbaren oder auch erneuerbare Sachmittel wie Maschinen, Büroarbeitsplätze, Besprechungs- und Schulungsräume, aber auch erforderliche Hard- und Software zeichnen sich dadurch aus, dass sie für die Dauer eines Vorgangs ganz oder teilweise belegt sind. Nach Beendigung des Vorgangs steht die Ressource für weitere Vorgänge zur Verfügung. Für alle nicht verzehrbaren Sachmittel müssen Kapazitätspläne erstellt werden, d. h. welche Ressourcen sind wann in welchem Umfang verfügbar. Knappe nicht verzehrbare Ressourcen wirken restriktiv, da zu jedem beliebigen Zeitpunkt die gesamte Inanspruchnahme einer Ressource die vorhandene Ressourcenkapazität nicht überschreiten darf (Vgl. Wieczorrek/Mertens (2007), S. 101 ff.; Zimmermann et al. (2006), S. 50 ff.).

Bereits in der Situationsbeschreibung wird mit den Anforderungen des Fachbereichs ein Zeitpunkt für die Nutzung des neuen Systems und damit der Endtermin für die

Systemablösung gesetzt. Möglicherweise wurden bereits Meilensteine festgelegt. Die Aufgabe des Projektleiters ist es, eine transparente zeitliche Planung der Phasen, Teilprojekte und Arbeitspakete vorzunehmen. Dazu werden wichtige Projekttermine – so genannte Meilensteine – und mögliche Pufferzeiten bestimmt. Pufferzeiten geben an, wie lange sich ein Vorgang verzögern darf, ohne Einfluss auf den Start nachfolgender Vorgänge zu haben. Eine effektive Zeitplanung resultiert aus der zeitlichen Planung der einzelnen Projektschritte und Arbeitspakete. Dazu werden Informationen aus vorangegangenen Schritten wie Projektorganisation und Ressourcenplanung abgeleitet. Ähnlich wie in der Ressourcenplanung werden je Arbeitspaket Anfangs- und Endtermin, sowie die geschätzte Dauer in Personenstunden bestimmt. Zur Bestimmung des zeitlichen Projektplans können verschiedene Methoden angewandt werden, die sich durch Komplexität und Übersichtlichkeit unterscheiden. Die Ergebnisse der einzelnen Verfahren sind jedoch ineinander umwandelbar. Der Einsatz von Software, wie bpsw. Microsoft Project, kann den Arbeitsaufwand dabei erheblich erleichtern (Vgl. Wieczorrek/Mertens (2007), S. 107 ff.; Versteegen et al. (2005), S. 188).

Bei der Listentechnik wird die Zeitplanung mit Hilfe von Tabellen erreicht, wobei einzelne Arbeitspakete oder –schritte jeweils eine Zeile darstellen. Neben Dauer, Anfangs- und Endtermin können weitere Informationen, wie bspw. Ressourcen und Verantwortlichkeiten je Zeile festgehalten werden (Vgl. Wieczorrek/Mertens (2007), S. 109).

Die Balkendiagrammtechnik stellt eine graphische Form der Listentechnik dar. Einzelne Vorgänge werden mit Hilfe von Balken über eine Zeitachse dargestellt. Die Länge eines jeden Balken repräsentiert dabei die Dauer des Vorgangs. Durch die Anordnung des Balkens über der Zeitachse sind Anfangs- und Endtermin bestimmt und Vorgänger, sowie Nachfolger ersichtlich. Das Balkendiagramm wird jedoch mit steigender Anzahl von Balken schnell unübersichtlich und ist somit eher für aggregierte Werte zu empfehlen (Vgl. Wieczorrek/Mertens (2007), S. 110).

Eine weitere Methode zur Zeitplanung ist die Netzplantechnik. Basierend auf Methoden der Graphentheorie werden alle Vorgänge durch zwei Ereignisse begrenzt. Ein Ereignis stellt den Beginn, das andere das Ende des Vorgangs und somit Anfangs- und Endtermin dar. Dabei kennzeichnet das Endereignis des einen Vorgangs gleichzeitig das Startereignis des darauf folgenden Vorgangs. In der Netzplantechnik können dabei Vorgangspfeilnetze, Vorgangsknotennetze oder Ereignisknotennetze verwendet werden (Vgl. Wieczorrek/Mertens (2007), S. 110).

Bei Bedarf kann die Projektterminplanung mittels Vorwärts- oder Rückwärtsplanung erfolgen. Dabei werden frühester und spätester Anfangstermin, sowie frühester und

spätester Endtermin der Vorgänge bestimmt. Für die Vorwärtsterminierung wird der Projektendtermin errechnet, indem alle Vorgänge ausgehend vom Projektstart analog des festgelegten Ablaufs kumuliert werden. Die Rückwärtsterminierung geht umgekehrt dazu vom gewünschten Projektendtermin aus und subtrahiert nacheinander alle Vorgänge laut Ablaufplan. Pufferzeiten zischen den einzelnen Vorgängen werden bei beiden Berechnungen ausgewiesen. Vorwärts- und Rückwärtsterminierung haben stets die gleiche Projektdauer zum Ergebnis. Der kritische Pfad ist der Weg von Projektstart bis Projektende, bei dem zwischen allen Vorgängen, die durchlaufen werden, keine Pufferzeit vorgesehen ist. Bezeichnend für den kritischen Pfad ist, dass Verzögerungen bei Vorgängen auf dem kritischen Pfad Auswirkungen auf den Endtermin des Projektes haben, da keine Pufferzeit zwischen den Vorgängen geplant ist (Vgl. Wieczorrek/Mertens (2007), S. 108 f.).

Durch die Aggregation der Eckdaten der Arbeitspakete auf Projektschritt und Projektphase lassen sich Termine für Meilensteine bestimmen. Grundsätzlich sollten bei der zeitlichen Planung Pufferzeiten berücksichtigt werden, um mögliche Verzögerungen abzufangen und Mehraufwände in der Projektsteuerung zu vermeiden. Der aktuelle Projektfortschritt kann bestimmt werden, indem der Umsetzungsgrad des aktuellen Vorgangs bestimmt und in den zeitlichen Ablauf eingeordnet wird.

6.5.1.6 Kostenplanung

Während des gesamten Systemablösungsprojektes fallen Kosten an. Ziel der Kostenplanung ist es, zu bestimmen, wann welche Kosten in welcher Höhe anfallen. Die Kosten sind dabei auf Arbeitspaketebene zu bestimmen und anschließend auf Prozessschritte, Phasen bis hin zum Projekt zu aggregieren. Hierbei werden Ergebnisse der vorangegangen Phasen, vor allem aus der Ressourcen- und Zeitplanung, verwendet, um die zu erwartenden Kosten zu planen. Zur Aufwandsschätzung ist die Verwendung verschiedener Methoden möglich (Vgl. Wieczorrek/Mertens (2007), S. 104 f.).

Grundsätzlich wird eine Unterteilung in Sachkosten und Personalkosten vorgenommen. Eine weitere Differenzierung der Kosten nach Kostenarten unterstützt das laufende Projektcontrolling. Dazu kann eine Unterteilung in interne bzw. externe Personalkosten, Hardware, Software, Netzwerk, Infrastruktur und Nebenkosten sinnvoll sein. Weiterhin lassen sich Kostenarten nach Kosten für die Projektabwicklung und für die Anschaffung und Inbetriebnahme des neuen Systems unterscheiden. Werden bei Systemablösungen stets gleiche Kostenartenstrukturen verwendet, lassen sich Projekte vergleichen und

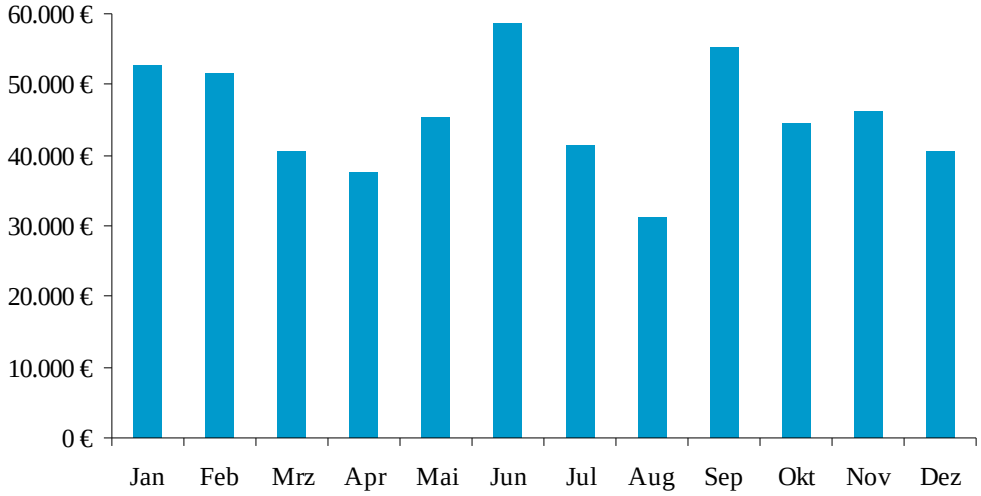
Kostenverläufe zu Analysezwecken auswerten (Vgl. Wieczorrek/Mertens (2007), S. 106 f.).

Eine Planung der Kosten wird auf Arbeitspaketebene durchgeführt. Dazu werden aus der Ressourcen- und Zeitplanung gewonnene Informationen weiterverarbeitet. In der Ressourcenplanung wurden die erforderlichen Personal- und Sachmittel zu den einzelnen Arbeitspaketen geplant. Aus der Zeitplanung lassen sich Informationen zum Zeitpunkt des Einsatzes der Ressourcen ableiten. In der Kostenplanung werden nun Kosten für die einzelnen Ressourcen hinterlegt. Dazu werden bei Personalkosten gewöhnlich Stundensätze hinterlegt und bei Sachmittel Anschaffungs- oder Herstellkosten je Einheit. Durch Multiplikation der Ressourcenkostensätze mit der benötigten Zeiteinheit aus der Ressourcenplanung lassen sich die separaten Kosten der Ressourcen je Arbeitspaket bestimmen. Anschließend werden zur Bestimmung der Kosten der Prozessschritte und Phasen die Kosten der Arbeitspakete kumuliert (Vgl. Buhl (2004), S. 80).

Der Einsatz von Softwaretools wie bspw. Microsoft Project erleichtert den Arbeitsaufwand, da benötigte Informationen aus Ressourcen- und Zeitplanung integriert zur Verfügung gestellt werden. Durch die Angaben zum Zeitpunkt des Ressourcenverbrauchs ist eine Bestimmung des Zeitpunktes der Kosten möglich. Damit kann anschließend der Kostenverlauf ermittelt und gegen das Budget aus den Rahmenbedingungen abgeglichen werden (Vgl. Versteegen et al. (2005), S. 190 f.).

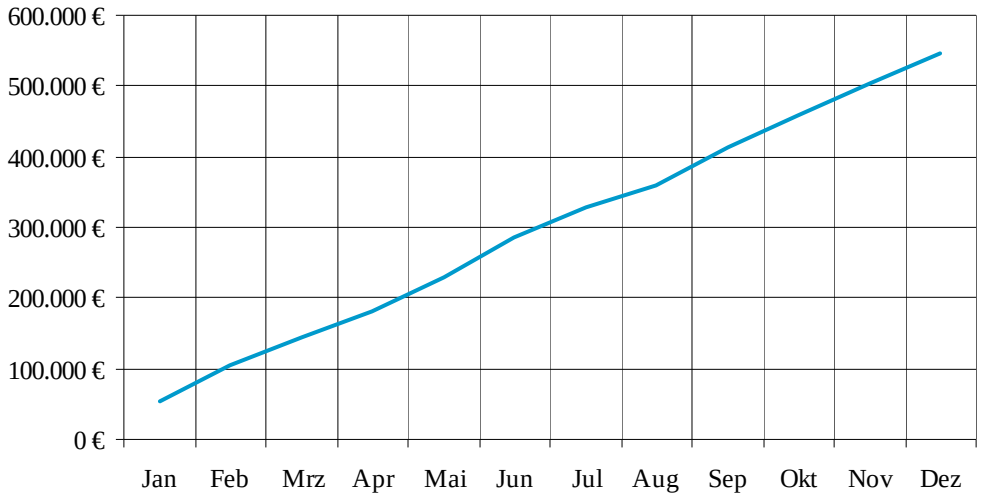
In der Ressourcenplanung wurde zwischen verzehrbaren und nicht verzehrbaren Sachmitteln unterschieden, wobei ein erhöhter Einsatz von verzehrbaren Mitteln in der Regel zu einer Beschleunigung einzelner Vorgänge führt. Analog dazu führt ein verminderter Einsatz von verzehrbaren Sachmitteln zu einer Verzögerung von Vorgängen. Da mit dem Verbrauch von Ressourcen unmittelbar Kosten zusammenhängen, lassen sich Kosten durch verminderten Verbrauch für einen gewissen Zeitraum verringern. Jedoch sollte bedacht werden, dass diese Kosten- und Vorgangsverzögerung Auswirkungen auf den Projektabschluss haben kann. Im umgekehrten Fall kann ein erhöhter Einsatz von verzehrbaren Ressourcen helfen, ein verzögertes Projekt zu beschleunigen und Terminschwierigkeiten auszugleichen. Dabei fallen stets erhöhte Kosten an, die durch das Budget abgefangen werden müssen. Ein erhöhter Personaleinsatz verursacht bspw. Kosten für Überstunden der Mitarbeiter oder Kosten für externe Dienstleister.

Zum Abschluss der Kostenplanung ist neben der tabellarischen Darstellung der Kosten eine grafische Veranschaulichung hilfreich, um einen Kostenverlauf aufzuzeigen (Vgl. Abbildung 6.7 und 6.8).



In Anlehnung an Buhl (2004), S. 82.

Abb. 6.7: Kostenplan auf Monatsebene



In Anlehnung an Buhl (2004), S. 82.

Abb. 6.8: Kostenverlauf durch kumulierte Monatswerte

6.5.1.7 Risikoanalyse

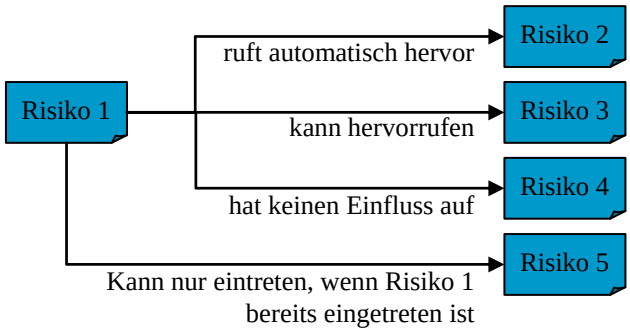
Die Systemablösung ist wie jedes IT-Projekt mit einer Reihe von Risiken behaftet, durch die ein kosten-, termin- und sachgerechter Abschluss beeinträchtigt werden kann. Aufgabe der Risikoanalyse ist es, potentielle Risiken zu identifizieren und Gegenmaßnahmen zur Vermeidung oder Minderung der möglichen Risiken einzuleiten.

Mit einem Risiko sind eine Wahrscheinlichkeit des Eintretens unerwünschter Ereignisse zu einem bestimmten Zeitpunkt und der damit einhergehende Schaden verbunden. Es werden jedoch nur projektbezogene Risiken betrachtet, allgemeine Lebensrisiken, wie z.B. Krankheit von speziellen Mitarbeitern bleiben außer Acht. Bezogen auf ein Projekt führt ein Risiko dazu, dass Projektziele nicht oder nur teilweise erreicht oder der Planung zu Grunde liegende Restriktionen verletzt werden. Projektrisiken können in allgemeine und spezielle Projektrisiken unterschieden werden. Allgemeine Risiken sind dabei Risiken die vergleichbar auch bei allen anderen Projekten auftreten, wie z.B. Unsicherheiten bei Planung und Schätzungen. Dem kann durch die Anwendung probater Methoden und Verfahren entgegengewirkt werden. Spezielle Projektrisiken ergeben sich aus der Individualität des Projektes. Die Risiken der Systemablösung steigen überproportional mit der Schwierigkeit und Komplexität des Projektes. Dabei liegen die Risiken nicht immer in der direkten Durchführung des Projektes, sondern teilweise in den Faktoren, bei denen auf keine Erfahrung zurückgegriffen werden kann, wie z.B. der Einsatz neuer Techniken und Methoden (Vgl. Wieczorrek/Mertens (2007), S. 354 f.).

Allgemein lassen sich drei Arten von Risiken unterscheiden. Sachliche Risiken drehen sich um den Gegenstand des Projektes, bei der Systemablösung bspw. das neue System. Derartige Risiken beziehen sich dabei z.B. auf Fehler bei der Entwicklung oder der Eignung des Produkts. Terminliche Risiken betreffen alle Risiken, die eine Gefahr für die Einhaltung von Projektterminen darstellen. Die Nichteinhaltung von Terminen kann unter Umständen zu finanziellen Mehrbelastungen führen, die nicht allein auf Personal- und Sachkosten zurückzuführen sind. Bei einer Systemablösung kann ein verspäteter Einsatz einer neuen Software bspw. zu einer längeren Nutzung des Altsystems führen. Dabei können für das Altsystem erneut Lizenzgebühren anfallen, die nicht geplant waren. Unter wirtschaftlichen Risiken werden jene Risiken verstanden, die zu höheren Kosten führen, als geplant wurde. Ein Projekt kann dadurch formell mit einem Verlust beendet werden (Vgl. Zimmermann et al. (2006), S. 23 f.).

Eine Risikoanalyse beginnt damit, alle potentiellen Risiken zu identifizieren. Bei der Ermittlung sachlicher Risiken können qualitative Verfahren, wie z.B. Brainstorming aller Projektmitarbeiter oder Expertenbefragung, zur Anwendung kommen. Zur Analyse terminlicher Risiken stehen spezielle Verfahren aus der Graphentheorie zur Verfügung, bspw. Kritische-Pfad-Methode. Wirtschaftliche Risiken können bestimmt werden, indem stochastische Methoden angewandt werden. Bei unklaren Zeitpunkten von Geldflüssen kann hier bspw. eine geeignete Wahrscheinlichkeitsverteilung angewendet werden. Die identifizierten Risiken sind zu dokumentieren. Dafür bieten sich bspw. Microsoft Excel Tabellen an, in denen die Liste der Risiken geführt wird. Die

strukturierte Tabellenform hat den Vorteil, dass die Pflege der Daten sehr komfortabel ist und weitere Informationen in einfacher Weise hinzugefügt werden können. In einer ergänzenden Microsoft Word Datei lassen sich bspw. nähere Angaben und Ausführungen zu den einzelnen Risiken machen (Vgl. Zimmermann et al. (2006), S. 24; Versteegen et al. (2005), S. 118).

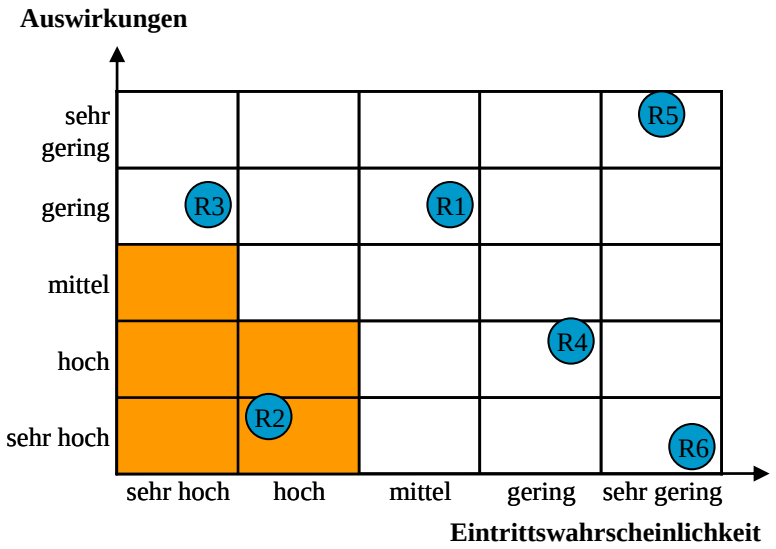


In Anlehnung an Versteegen (2005), S. 119.

Abb. 6.9: Abhängigkeiten zwischen Risiken

Die identifizierten Risiken werden nun bewertet, um das Ausmaß der Risiken zu bestimmen. Dabei sind zwei Faktoren entscheidend für das Ausmaß. Zum einen die Eintrittswahrscheinlichkeit eines Risikos und zum anderen die Auswirkungen auf den Projektverlauf. Die Bestimmung der Eintrittswahrscheinlichkeit kann mit Hilfe geeigneter stochastischer Methoden oder auf Grundlage von Erfahrungswerten erfolgen. Die Eintrittswahrscheinlichkeit hängt stark mit der Abhängigkeit zwischen den Risiken zusammen. So existieren mitunter Risiken, die bei Eintritt anderer Risiken hervorgerufen werden oder überhaupt nur dann selbst eintreten können und wiederum andere, die völlig unbeeinflusst davon sind. In Abbildung 6.9 ist der Zusammenhang grafisch dargestellt. Diese Informationen lassen sich beispielsweise in der oben genannten Risikoliste in der Tabelle ergänzen. Die Auswirkungen auf den Projektverlauf lassen sich in der Regel durch die Angabe von Geldeinheiten ausdrücken, denn sowohl terminliche Verzögerungen, als auch erhöhter Sachmittel- oder Personaleinsatz führen zu höheren Kosten bei Eintritt eines Risikos. Anstelle von Zahlenangaben kann sowohl bei der Eintrittswahrscheinlichkeit, als auch bei der Auswirkung der Risiken eine Einteilung in verschiedene Klassen vorgenommen werden. Mit Hilfe dieser beiden Dimensionen wird eine Risikomatrix erstellt, die einen Überblick über alle Risiken liefert. Durch Markierung eines oder mehrerer kritischer Bereiche lassen sich Prioritäten für die Aufmerksamkeit der Risiken ablesen. In der Abbildung 6.10 ist der kritische Bereich orange markiert. Das Risiko R2 liegt im kritischen Bereich und sollte mehr Beachtung finden als das Risiko R5, das sehr geringe

Auswirkungen im Falle seines sehr unwahrscheinlichen Eintritts auf den Projektverlauf hat (Vgl. Zimmermann et al. (2006), S. 24 f.; Versteegen et al. (2005), S. 118 ff.).



In Anlehnung an Versteegen (2005), S. 123.

Abb. 6.10: Risikomatrix

Nach Identifizierung und Klassifizierung der Risiken müssen nun Maßnahmen definiert werden, die zur Vermeidung des Eintritts oder zur Einschränkung der Auswirkungen der Risiken führen. Dabei lassen sich für einige Risiken allein durch ihre Identifikation Handlungen ableiten. Technische Mängel in Computerhardware lassen sich bspw. durch stärke Qualitätskontrollen entgegnen. Andere Risiken dagegen lassen sich nicht ausschalten, jedoch kann anderweitig Vorsorge getroffen werden. Die Gefährdung von Wirtschaftsgütern durch Katastrophen oder Brände und der damit entstandene finanzielle Schaden können bspw. durch Versicherungen abgefangen werden. Für den Fall, dass eine solche Versicherung nicht möglich ist, können Wagniskosten im Rahmen der Kostenplanung kalkuliert werden. Risiken, die nur sehr unwahrscheinlich eintreten und dabei sehr geringe Auswirkungen haben, werden häufig nicht in der Risikoanalyse berücksichtigt. Der Aufwand steht in keinem Verhältnis zum Nutzen. Deshalb werden Gegenmaßnahmen erst dann formuliert und eingeleitet, wenn es tatsächlich zum Eintritt kommt (Vgl. Zimmermann et al. (2006), S. 25).

6.5.2 Vorbereitung

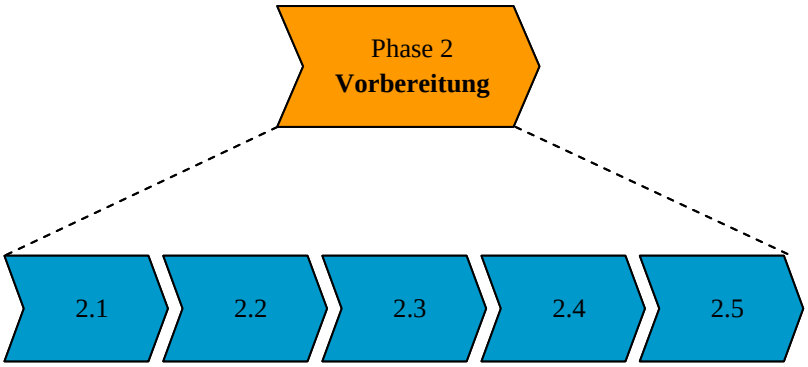


Abb. 6.11: Phase 2 – Vorbereitung

Die Phase der *Vorbereitung* kann in drei Themenbereiche unterteilt werden. Im ersten Schritt wird ein Pilotsystem installiert, das nicht nur für das ‚Look and Feel', sondern auch für die nachfolgenden Schritte eingesetzt wird. Im zweiten Schritt sollen alle Aspekte festgehalten werden, die im neuen System getestet werden sollen. Die weiteren Schritte befassen sich mit der Vorbereitung der Datenmigration. Nach der Bestimmung des benötigten Datenumfangs wird eine Zuordnungstabelle erstellt. Dabei müssen alle zu übertragenden Daten des Altsystems eine Zuordnung im neuen System aufweisen. Ist dies nicht der Fall, können die Daten später nicht übertragen werden. Im letzten Schritt wird der Datenbestand aufbereitet. Dazu werden die zugeordneten Daten auf Konsistenz, Redundanz und Anomalien analysiert, um die Datenqualität zu steigern und eine zuverlässige Datenmigration vorzubereiten. In dieser Phase ist besonders die Zusammenarbeit mit der Fachabteilung gefragt, da nur sie funktionale Gesichtspunkte des Systems beschreiben können, die das neue System während der Testszenarien erfüllen muss. Weiterhin kann nur die Fachabteilung festlegen, wie vorhandene Daten zum neuen System zugeordnet werden sollen und welchen Datenbestand sie zur täglichen Anwendung benötigt. Die Phase 2 ist im Überblick in der Abbildung 6.11 und der Tabelle 6.3 dargestellt.

Tab. 6.3: Schritte der Phase 2 – Vorbereitung

Schritt	Bezeichnung
2.1	Pilotsystem installieren
2.2	Entwicklung von Testszenarien
2.3	Bestimmung des benötigten Datenumfangs
2.4	Abbildung der Datenstruktur
2.5	Aufbereitung der Daten im Altsystem

6.5.2.1 Pilotsystem installieren

Die frühe Installation eines Pilot- oder Testsystems erlaubt einen frühen Kontakt mit dem neuen System. Fragen und Vorstellungen, aber auch Anforderungen können somit bereits sehr früh geäußert werden. Der Vorteil liegt darin, dass in einer sehr frühen Projektphase Erfahrungen mit dem System in Hinblick auf Installation, Hardwareanforderungen und Informationssysteminfrastruktur gemacht werden können. Dabei kann sich das Pilotsystem auf Hauptkomponenten des Systems beschränken. Je nach Aufwand und Zweckmäßigkeit können weitere Komponenten eingerichtet werden. Die komplette Systeminstallation erfolgt dann in Phase 3.

Aber auch bei der Entwicklung von Testszenarien kann das Pilotsystem genutzt werden, um diese zu verfeinern und veranschaulichen. Die Vorbereitungen zur Datenmigration, die Schritte 2.3 bis 2.5, können ebenfalls durch einen Zugriff auf das Pilotsystem wertvolle Informationen zu Attributen und Entitäten des Neusystems erfassen.

6.5.2.2 Entwicklung von Testszenarien

Das Testen des Systems hat bei einer Systemablösung einen hohen Stellenwert, denn genau an der Stelle wird entschieden, ob das System das erfüllt, was erwartet wird. Dazu werden Testszenarien durchlaufen, in denen bestimmte Abläufe des Geschäftsprozesses simuliert werden, der durch das System unterstützt werden soll. Wird die Testphase jedoch zu pauschal und nachlässig durchlaufen, bleiben vorhandene Fehler zunächst im System verborgen und kommen aber später beim Produktiveinsatz zum Tragen, denn das Testen kann nicht die Abwesenheit von Fehlern beweisen, sondern lediglich deren Vorkommen (Vgl. Dijkstra (1970), S. 19 ff.). Aus diesem Grund ist es wichtig, einen möglichst umfassenden Teil aller erdenklichen Situationen nachzustellen und mittels Testfällen zu prüfen, um so eventuell vorhandene Fehler zu identifizieren.

Die zentrale Anforderung an einen Test ist, dass er reproduzierbar ist. Bei einem wiederholten Vorgehen unter genau den gleichen Bedingungen, muss stets das gleiche Ergebnis erzielt werden können (Vgl. Siedersleben (2003), S. 288 f.). Grundlage dafür soll ein strukturiertes Vorgehen durch Definition von Testfällen, Anforderungen und Eintrittsbedingungen sein.

Für verschiedene Testfälle sind unterschiedliche Arten von Tests notwendig. Die Norm ISO 9126-1 enthält Qualitätsmerkmale für Produktqualität von Software, aus denen sich entsprechende Testarten ableiten lassen. Es wird zwischen funktionalen Tests, die das Qualitätsmerkmal Funktionalität prüfen, und den nichtfunktionalen Tests unterschieden, die die restlichen Qualitätsmerkmale analysieren und häufig vernachlässigt oder gar ganz außer Acht gelassen werden. Die Qualitätsmerkmale lassen sich in weitere Untermerkmale differenzieren, um eine genauere Beschreibung der Qualitätsmerkmale zu erhalten. Eine Gegenüberstellung der Qualitätsmerkmale und Testarten ist in der Tabelle 6.4 zu sehen (Vgl. Siedersleben (2003), S. 300 f.).

Tab. 6.4: Zusammenhang von Qualitätsmerkmalen und Testarten

Qualitätsmerkmal	Untermerkmale	Testart
Funktionalität	Angemessenheit, Richtigkeit, Interoperabilität mit Nachbarsystemen, Erfüllung gesetzlicher Normen, Sicherheit, Datenschutz	Funktionstest
Zuverlässigkeit	Reife, Fehlertoleranz, Wiederherstellbarkeit	Robustheitstest
Benutzbarkeit	Verständlichkeit, Erlernbarkeit, Bedienbarkeit, Attraktivität	Benutzbarkeitstest
Übertragbarkeit	Anpassbarkeit, Installierbarkeit, Koexistenz	Installationstest
Änderbarkeit	Analysierbarkeit, Modifizierbarkeit, Stabilität, Testbarkeit	-
Effizienz	Zeitverhalten, Verbrauchsverhalten bzgl. Ressourcen	Performance-Test

In Anlehnung an Siedersleben (2003), S. 301.

Ein *Funktionstest* dient zur Überprüfung der fachlichen Richtigkeit des Systems, der Kommunikation mit angrenzenden Systemen, der Erfüllung gesetzlicher Forderungen und Normen, sowie der Sicherheit des Systems vor unberechtigtem Zugriff. Der Funktionstest besteht aus mehreren Elementen. Für den Ablauf des Tests werden verschiedene Anwendungsfälle definiert, die die Bearbeitungsschritte von Anfang bis Ende beschreiben. Zustandsmodelle werden erstellt, um jeden Zustand und Zustandsübergang eines Objektes mit Hilfe von Testfällen mindestens einmal zu durchlaufen. Für Eingabefelder werden gültige und ungültige Werte festgelegt. Anforderungslisten und Anforderungsbeschreibungen halten Anforderungen fest, die mit bestimmten Eintrittsbedingungen verbunden sind. Dabei werden entsprechend Testfälle definiert, die diese Anforderungen erfüllen (Vgl. Siedersleben (2003), S. 301 f.). Durch eine Klassifizierung der Anforderungen in drei Kategorien, wie in Tabelle

6.5, wird die Bedeutung der Anforderung für den Betrieb des Systems abgebildet. Eine Anforderung der Kategorie A stellt eine kritische Funktion dar. Läuft diese bei einem Test auf einen Fehler, so muss dieser behoben werden, da sonst der Geschäftsprozess akut gefährdet ist. Ein Fehler der Kategorie B bedeutet, dass der Geschäftsprozess nur mit Einschränkungen ausgeführt werden kann. Anforderungen der Kategorie C sind unkritisch in Bezug auf den Geschäftsprozess, jedoch sollten auch sie vom System erfüllt werden. Sonderfälle des Geschäftsprozesses können damit ebenfalls gekennzeichnet und festgehalten werden, indem entsprechende Anforderungen formuliert und klassifiziert werden.

Tab. 6.5: Kategorien von Anforderungen und Fehlern

Kategorie	Beschreibung
A	Kritische Anforderung, bei Fehlern ist der Geschäftsprozess in seiner Ausführung gefährdet
B	Bei Fehlern kann der Geschäftsprozess nur mit Einschränkungen ausgeführt werden
C	Unkritische Anforderung, Geschäftsprozess ist nicht gefährdet

Die Vorbereitung für die funktionalen Tests umfasst den größten Aufwand. Sie sind nur in enger Zusammenarbeit mit der Fachabteilung zu definieren, denn nur sie kennt die Anforderungen an das System. Zur Abbildung der Testszenarien können bspw. Sequenz- und Use-Case-Diagramme oder Anforderungslisten in Microsoft Excel erstellt werden. Im Anhang A befinden sich Checklisten für den Funktionstest. Zur Erstellung der Testszenarien kann auf das Pilotsystem zugegriffen werden, um bestimmte Funktionen oder Geschäftsvorfälle nicht unberücksichtigt zu lassen. Der frühe Kontakt mit dem neuen System hinterlässt beim Fachbereich einen ersten Eindruck, durch den eine Verfeinerung der Testszenarien erreicht werden kann und eine grafische Darstellung von Testvorfällen erlaubt.

Die weiteren genannten nichtfunktionalen Tests werden im Kapitel 6.5.4.1 behandelt, da sie nicht in direkter Zusammenarbeit mit der Fachabteilung und nicht in diesem Umfang vorbereitet werden müssen.

6.5.2.3 Bestimmung des benötigten Datenumfangs

Bei einer Systemablösung wird ein vorhandenes Informationssystem ersetzt. Dieses unterscheidet sich üblicherweise erheblich in seiner Struktur und Architektur von der des Altsystems. Dies gilt sowohl für Präsentations- und Anwendungs-, als auch für die Datenebene. Nach einer Systemablösung soll im Regelfall ein Zugriff auf Daten erfolgen, die zuvor im Altsystem angelegt wurden. Dazu kann entweder der Zugriff

über eine Schnittstelle auf den Datenbestand direkt im Altsystem erfolgen oder der Datenbestand des Altsystems in das neue System überführt werden.

Ein Zugriff auf Daten im Altsystem würde bedeuten, dass das Altsystem weiter bestehen würde. Das verursacht nicht nur Kosten, sondern stellt höhere Anforderungen an Wartung und Betrieb, da in dem Fall zwei Systeme anstelle von einem betrieben werden müssen. Eine Nutzung dieser Variante ist lediglich temporär denkbar, um einen Projektverzug durch erhöhten Einsatz von Personal und Ressourcen vermeiden oder abzuwenden zu können.

Mit einer Übernahme der Daten in das neue System kann das Altsystem zum Abschluss der Systemablösung tatsächlich abgeschaltet werden, vorausgesetzt, die nicht übernommenen Daten wurden in geeigneter Form archiviert. Eine Archivierung ist nur erforderlich, wenn die Daten nicht verloren gehen sollen. Der Aufwand für eine Übernahme der Daten ist abhängig vom Umfang des zu übertragenden Datenbestandes, dessen Komplexität und der Verschiedenheit der Datenmodelle.

Die zu übertragenden Daten sind eine Teilmenge des vorhandenen Datenbestandes im Altsystem. Je kleiner die Teilmenge, desto weniger Zeit wird für die Übertragung benötigt. Der Zeitraum der Datenübernahme ist gewöhnlich kritisch, da der Zugriff sowohl auf das Altsystem, als auch auf das Neusystem in der Regel gesperrt ist, um die Konsistenz des Datenbestandes zu gewährleisten. Die Nutzung der Systeme ist also in diesem Zeitraum nicht möglich. Die Größe des zu übertragenden Datenbestandes hat also direkten Einfluss auf den Zeitraum, in der eine produktive Arbeit der Anwender im System nicht möglich ist. Folglich soll die Größe der zu übertragenden Datenmenge so gering wie möglich gewählt werden. Je kleiner jedoch die zu übertragende Datenmenge, desto kleiner ist der Datenbestand auf dem im Neusystem zugegriffen werden kann. Je nach Zweck des Systems ist ein Zugriff oder eine Recherche nach vorhandenen Daten im kleineren oder größeren Umfang erforderlich oder gar Teil der täglichen Arbeit. Dementsprechend muss der Systemzweck bei der Auswahl der zu übertragenden Daten berücksichtigt werden. Eine Wissensdatenbank oder ein System zur Verwaltung des Buchbestandes in der Bibliothek bietet mitunter weniger Spielraum zur Minimierung der Teilmenge, als ein System zur Erfassung von Kundenaufträgen. Letztendlich sollte die Fachabteilung und Hauptanwender, so genannte Keyuser, einen wesentlich Beitrag dazu leisten, zu bestimmen, welcher Datenbestand zur täglichen Anwendung benötigt wird. Vorstellbar ist eine Festlegung, dass lediglich Daten übertragen werden, auf die innerhalb eines eingeschränkten Zeitraumes, beispielsweise während der letzten 24 Monate, zugegriffen wurde oder eine Einschränkung auf bestimmte Attributwerte auf der Basis einzelner Entitätstypen. Der übrige Datenbestand wird in geeigneter Form

archiviert, um im Zweifelsfall darauf zugreifen zu können. Die vorgenommene Abgrenzung des Datenbestands ist bindend, da nachfolgende Prozessschritte lediglich diese Teilmenge des Datenbestands betrachten. Eine Ausweitung auf weitere Daten hätte später zur Folge, dass nachfolgende Prozessschritte wiederholt ausgeführt werden, um deren Ergebnisse zu überprüfen.

Die Komplexität des Datenbestandes setzt sich aus der Menge der Entitätstypen, deren Struktur und den möglichen Attributwerten zusammen. Ein Einfluss darauf kann zum einen mit Hilfe der Abbildungstabellen zur Übertragung der Daten und zum anderen mit der Aufbereitung der Daten im Altsystem erfolgen. Denkbar ist beispielsweise eine Zusammenführung gleichartiger Attributwerte, die sowohl bei der Erstellung der Abbildungstabellen oder bei der Aufbereitung der Altdaten berücksichtigt werden kann. Jedoch ist das abhängig davon, welchen Spielraum die Datenmodelle der Systeme zulassen. Dessen ungeachtet hat nicht zuletzt der Geschäftsprozess einen Einfluss darauf inwiefern die verwendeten Daten voneinander zu unterscheiden sind.

Der Einfluss auf die Datenmodelle der beiden Systeme ist bei der Verwendung von Standardsoftware nur sehr begrenzt möglich. In der Tat kann das Datenmodell entsprechend erweitert werden, jedoch wird damit eine Distanz zum Standard aufgebaut, die bei Updates und Upgrades des Systems Probleme bereiten kann, aber zumindest Aufwand bedeutet. Eine Eigenentwicklung ist an dieser Stelle häufig flexibler, allerdings trägt jede Erweiterung nicht unmittelbar zur Vereinfachung des Datenmodells bei, sondern in erster Linie zu dessen Vergrößerung. Je unterschiedlicher die Datenmodelle sind, desto größer ist der Aufwand für die Erstellung von Abbildungstabellen, die für eine Datenübernahme benötigt werden. Dabei ist es gewöhnlich von Vorteil, wenn gleichartige Datenbankmodelle verwendet werden. Eine Datenübernahme zwischen zwei relationalen Datenbanksystemen ist im Regelfall einfacher zu bewältigen als zwischen einem hierarchischen und einem relationalen Datenbanksystem. Die gebräuchlichsten Datenbankmodelle sind im Kapitel 2.3 näher erläutert.

Die beiden folgenden Unterkapitel befassen sich mit der Erstellung von Abbildungstabellen, sowie der Aufbereitung des Altdatenbestands und beschreiben somit die Möglichkeit der Einflussnahme auf die Komplexität der zu übertragenden Daten und die Überwindung der Heterogenität der Datenmodelle.

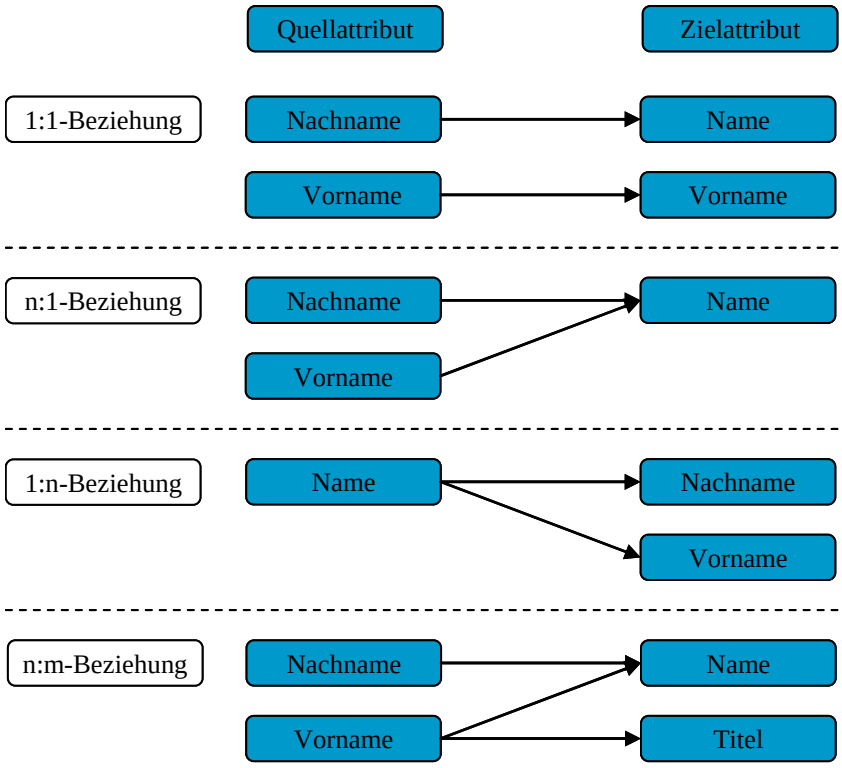
6.5.2.4 Abbildung der Datenstruktur

Der zentrale Punkt der Datenmigration liegt darin, die Lücke zwischen Alt- und Neusystem zu schließen und die ausgewählten Daten vollständig und korrekt zu übertragen. Dazu wird auf der Basis von Entitätstypen und Attributen eine so genannte Abbildungs- oder Zuordnungstabelle erstellt, die eine Zuordnung zwischen altem und neuem Datenmodell beschreibt. Daneben kann die Abbildungstabelle zusätzlich Transformationsregeln beinhalten, wenn eine Umwandlung der Daten des Altsystems zur Übernahme in das Neusystem notwendig ist. Grundlage dafür ist eine sorgfältige Analyse der Daten des Altsystems (Vgl. Siedersleben (2003), S. 62).

Das Erstellen der Abbildungstabelle, auch Mapping der Datenstrukturen genannt, ist eine Maßnahme zur Vorbereitung der Datenmigration in der letzten Phase der Systemablösung. Da mit Hilfe der Abbildungstabelle jedoch bereits in Phase 3 Daten aus dem Altsystem übertragen werden, um reale Testdaten für die Testphase bereitzustellen, muss die Abbildungstabelle zu diesem frühen Zeitpunkt erzeugt werden. Des Weiteren wird die Abbildungstabelle mit der Übertragung von realen Daten für den Test ein erstes Mal geprüft, denn Fehler in der Abbildungstabelle gefährden die spätere Datenmigration und damit den Erfolg der Systemablösung, zumindest aber bewirken etwaige Fehler einen Zeitverzug, der immense Kosten verursachen kann. Ist beispielsweise eine Zuordnung für ein Attribut oder Entitätstyp fehlerhaft oder nicht vorhanden, so kann das Datenobjekt bei der Datenmigration nicht zugeordnet werden. Eine vollständige Übertragung der Daten ist in diesem Fall nicht möglich.

Für die Übertragung selbst kann es notwendig sein, dass Programme geschrieben werden müssen, die zunächst die zu übertragenden Daten aus dem Altsystem extrahieren, um sie in einem weiteren Schritt zu transformieren und anschließend in das Neusystem zu überführen (Vgl. Leser/Naumann (2007), S. 123). Dieses Vorgehensmodell beschränkt sich an der Stelle darauf, eine Abbildungstabelle zu erstellen, die als Vorlage für solche Programme dient.

Die Zuordnung von Attributen zwischen Alt- und Neusystem kann verschiedene Formen annehmen. Das Attribut des Altsystems wird hierbei als Quellattribut, das des Neusystems entsprechend als Zielattribut bezeichnet. Analog dazu werden Entitätstypen in Quell- und Zielentitätstypen unterschieden. Je nach Anzahl der verbundenen Attribute werden bestimmte Beziehungstypen klassifiziert. Die so genannte Kardinalität gibt die Komplexität einer Beziehung an (Vgl. Bodendorf (2006), S. 16 f.). Dabei wird zwischen 1:1-, n:1-, 1:n- und n:m-Beziehungen unterschieden, die in Abbildung 6.12 beispielhaft dargestellt sind.



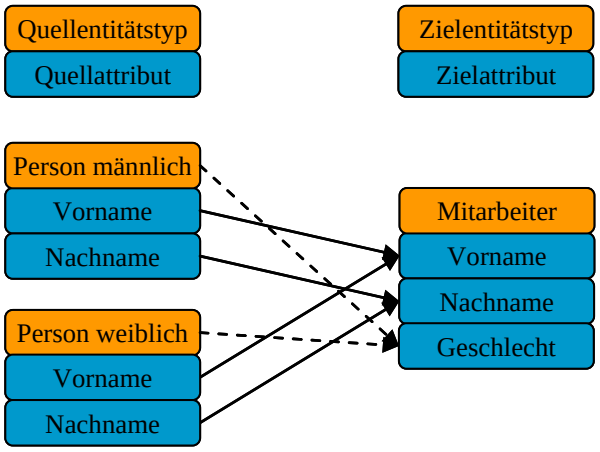
In Anlehnung an Leser/Naumann (2007), S. 128.

Abb. 6.12: Beziehungen zwischen Quell- und Zielattributen

Eine 1:1-Beziehung stellt dabei die einfachste Form der Zuordnung dar, da einem Quellattribut genau einem Zielattribut zugewiesen ist und umgekehrt. Die Bezeichnung der Attribute kann jedoch unterschiedlich sein. Eine n:1-Beziehung verknüpft mehrere Quellattribute mit genau einem Zielattribut. Dabei sind Regeln festzulegen, in welcher Form die Quellattribute in einem einzigen Zielattribut abgebildet werden. Analog dazu bezeichnet eine 1:n-Beziehung den Fall, wenn aus einem Quellattribut mehreren Zielattributen erzeugt werden. Dieser Umstand ist nicht zu verwechseln mit mehreren 1:1-Beziehungen, die das gleiche Quellattribut mit mehreren Zielattributen verknüpfen. Dabei erfolgt eine multiple Verteilung des Quellattributs mit einer Einschränkung des Wertebereichs der Attributwerte, jedoch keine Aufspaltung des Quellattributs. Bei n:m-Beziehungen sind mehrere Quellattribute mit mehreren Zielattributen verknüpft, wobei mindestes aus einem Quellattribut mehrere Zielattribute erzeugt werden, von denen wiederum eines aus mehr als einem Quellattribut gebildet wird (Vgl. Leser/Naumann (2007), S. 127 f.).

Die Beziehungen zwischen Datenobjekten erfolgen nicht nur direkt auf Attributebene, sondern mitunter auf Ebene der Entitätstypen. Strukturelle Unterschiede im Datenmodell lassen sich nicht allein durch Verknüpfungen von Attributen überbrücken. Dazu müssen Entitätstypen herangezogen werden, da sie strukturelle Elemente

darstellen, die die Attribute in das Datenmodell eingliedern. In Abbildung 6.13 sind zwei heterogene Datenmodelle dargestellt. Das Geschlecht einer Person wird in einem Modell durch die Entitätstypen abgebildet, im anderen durch einen Attributwert. Bei einer Datenübernahme soll in diesem Beispiel der Attributwert mit einem konstanten Wert in Abhängigkeit vom Quellentitätstyp gefüllt werden. In der Abbildung ist dies durch eine gestrichelte Linie dargestellt. LESER und NAUMANN nennen diesen Fall ‚höherstufige Korrespondenzen' (Vgl. Leser/Naumann (2007), S. 128 f.).



In Anlehnung an Leser/Naumann (2007), S. 128.

Abb. 6.13: Höherstufige Korrespondenz

Eine Abbildungstabelle umfasst mindestens folgende Spalten (Vgl. Siedersleben (2003), S. 62):

- Bezeichnung des Quellentitätstyps
- Bezeichnung des Quellattributs
- Transformationsregel
- Transformationsanmerkung
- Bezeichnung des Zielentitätstyps
- Bezeichnung des Zielattributs.

Transformationsregeln beschreiben eine Funktion zur Überführung des Quellattributwertes in einen gültigen Zielattributwert. Typisches Beispiel dafür sind Transformationen von Datumsformaten oder Einheiten, aber auch Umwandlungen von Datentypen z.B. ein im Textformat gespeicherter numerischer Wert, der im Neusystem durch einen Datentyp Integer abgebildet werden soll. Transformationsanmerkungen

enthalten alle Informationen, die nicht durch die anderen Spalten abgebildet sind und für eine korrekte Umwandlung der Daten erforderlich sind.

Eine weitere Spalte ‚Hinweis' enthält zusätzliche Informationen zur Datenmigration, die berücksichtigt werden müssen, um eine vollständige Übertragung der Daten zu gewährleisten, beispielsweise den Fall höherstufiger Korrespondenzen oder Wertebereiche von Attributwerten.

Eine Spalte ‚Pflichteingabe im Neusystem' gibt an, ob das Attribut mit einem Attributwert gefüllt sein muss, um die Ausführung des Systems nicht zu gefährden bzw. überhaupt mit dem Datensatz arbeiten zu können. Die Bestimmung der Pflichteingaben kann anhand der Dokumentation des Neusystems oder mit Hilfe des Prototyps erfolgen. Dabei kann bspw. bei Anwendungssystemen mit einer Statusschaltung die Pflichteingabe eines Attributes vom Systemstatus des Datensatzes abhängen.

Neben den genannten Beziehungen existieren im Datenmodell des Altsystems mitunter Attribute, die nicht in das Neusystem übertragen werden. Auch diese Attribute werden in die Abbildungstabelle aufgenommen, um die Information, dass das Attribut nicht weiter betrachtet werden muss, in nachfolgende Prozessschritte übertragen wird und berücksichtigt werden kann. Analog dazu werden die Attribute im Neusystem dokumentiert, für die keine Zuordnung aus dem Altsystem erfolgen kann, weil die Informationen bislang nicht erfasst oder gespeichert wurden.

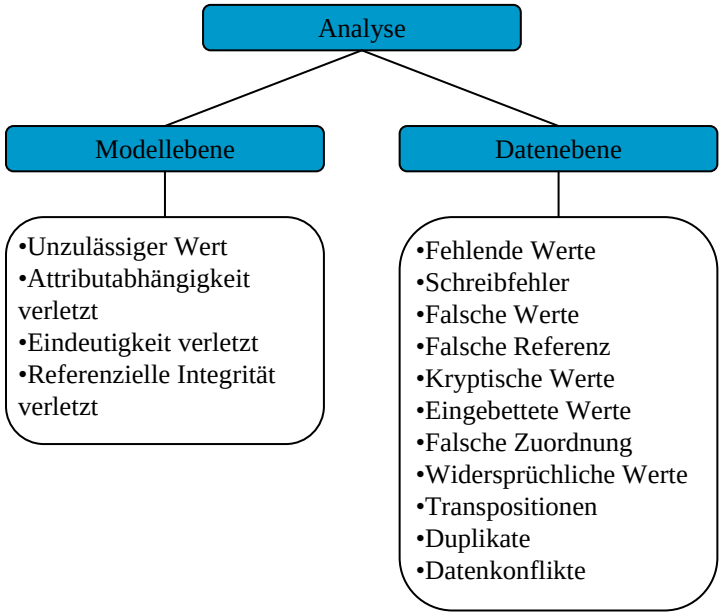
Die Abbildungstabelle bietet somit neben der reinen Zuordnung von Attributen viele Informationen, um nachfolgende Schritte zu unterstützten und zu beschleunigen.

6.5.2.5 Aufbereitung der Altdaten

In diesem Schritt wird eine detaillierte Analyse mit anschließender Aufbereitung des vorhandenen Datenbestandes vorgenommen. Im Schritt 2.3 wurde bereits der zu übertragende Datenumfang bestimmt. Eine weitere Einschränkung ist im Schritt 2.4 durch die Kennzeichnung nicht mehr benötigter Attribute erfolgt. Auf Basis dieser Abgrenzungen wird bei der Analyse der Daten nicht der gesamte Datenbestand betrachtet, sondern nur die eingeschränkte Teilmenge. Da demzufolge eine geringere Menge von Attributen untersucht und ausgewertet werden müssen, erfolgt durch dieses Vorgehen eine erhebliche Beschleunigung des Prozessschrittes. Dessen ungeachtet besteht jedoch das Risiko, dass nachträgliche Änderungen der vorangegangenen Einschränkungen eine erhebliche Nacharbeit hervorrufen können. Dies bezieht sich

sowohl auf diesen, als auch auf alle nachfolgenden Prozessschritte, die die Informationen auf Basis dieser Einschränkungen verwenden.

Mit Hilfe der Analyse des Datenbestandes werden sowohl Attributwerte, als auch Entitäten auf eine Reihe verschiedener Fehler und Konflikte hin untersucht. Die Daten können dabei in sich fehlerhaft sein oder mit Bezug auf das Datenmodell in Hinblick auf Integritätsbedingungen. In Abbildung 6.14 sind Fehler und Konflikte nach Modell- und Datenebene klassifiziert, die im Folgenden kurz beschrieben werden (Vgl. Leser/Naumann (2007), S. 317 ff.).



In Anlehnung an Leser/Naumann (2007), S. 319.

Abb. 6.14: Fehler und Konflikte in Modell- und Datenebene

Datenmodelle können mit Hilfe von Integritätsbedingungen Datenfehler vermeiden, da das DBMS nur Datensätze zulässt, die den Integritätsbedingungen entsprechen (Vgl. Kapitel 2.3). Bei der Entwicklung des Altsystems wurden möglicherweise keine Integritätsbedingungen definiert, die jedoch bei einer Datenübernahme erfüllt werden müssen, weshalb es zu folgenden Problemen kommen kann (Vgl. Leser/Naumann (2007), S. 319).

- Ein *unzulässiger Wert* liegt außerhalb des Wertebereichs des Zielattributs, beispielsweise Geburtstag = 32.01.1980.

- Eine *Abhängigkeit zwischen Attributwerten* wird verletzt, wenn zwei Attributwerte, die in Abhängigkeit zueinander stehen einen Konflikt bilden. Beispielsweise wird die Abhängigkeit ‚Alter = heute – Geburtstag' verletzt,

wenn in einem Datensatz die Attributwerte Alter = 20 und Geburtstag = 01.01.1980 zusammentreffen.

- Die *Eindeutigkeit von Attributen* wird verletzt, wenn als eindeutig gekennzeichnete Attribute Attributwerte enthalten, die mehr als ein Mal auftreten. Besonders kritisch ist das beispielsweise bei Seriennummern.

- Eine *Verletzung der referenziellen Integrität* liegt vor, wenn ein Attributwert in einem als Fremdschlüssel gekennzeichneten Attribut auf einen nicht vorhandenen Attributwert des referenzierten Attributs verweist.

Weitere Fehler und Konflikte können in der Datenebene gefunden werden. Sie lassen sich nicht durch Spezifikationen in der Modellebene verhindern (Vgl. Leser/Naumann (2007), S. 320 f.).

- *Attributwerte fehlen*, wenn die Ausprägung des Attributs ‚NULL' ist. Ein häufiger Grund dafür ist das Fehlen von Informationen zum Zeitpunkt der Eingabe. NULL-Werte können zwar durch eine Integritätsbedingung ‚not NULL' verhindert werden, jedoch führt das häufig dazu, dass bei manueller Eingabe Dummywerte wie z. B. ‚AAA' oder Leerzeichen eingetragen werden. Hinzu kommt, dass die gewählten Dummywerte oftmals nicht einheitlich sind und ein großer Aufwand zur Identifizierung betrieben werden muss.

- Gerade bei manueller Eingabe treten häufig *Schreibfehler* in verschiedenster Form auf. Mehrere verschiedene Attributwerte drücken in diesem Fall das gleiche aus, beispielsweise ‚Motorrad' und ‚Motorad'. Schreibfehler können auch bei automatischer Schrifterkennung entstehen.

- *Falsche Attributwerte* bilden einen Gegensatz zu tatsächlichen Gegebenheiten der realen Welt, beispielsweise Konzern = ‚Volkswagen' und Gründungsjahr = 1910. Das Problem liegt in der Identifizierung der falschen Attributwerte, da ein Vergleich von Attributwerten mit der realen Welt sehr aufwendig sein kann.

- *Falsche Referenzen* sind Attributwerte eines als Fremdschlüssel gekennzeichneten Attributs, die auf einen vorhandenen aber falschen Attributwert des referenzierten Attributs verweisen.

- *Kryptische Attributwerte* sind bei der Dateneingabe abgekürzte Werte. Der Bezug zur realen Welt bzw. zu dem tatsächlich bezeichneten materiellen oder immateriellen Gegenstand ist verloren gegangen, nicht dokumentiert und nicht mehr nachvollziehbar.

- Bei e*ingebetteten Attributwerten* sind Informationen verschiedener Attribute in einem Attributwert enthalten. Dieser Umstand kann durch frühere Datenübernahmen, durch verschiedene Entwicklungsstadien des Altsystems oder durch unsachgemäße Benutzereingaben auftreten, bspw. Modell = ‚VW Passat' und Marke = NULL.

- Attributwerte sind *falsch zugeordnet*, wenn ein korrekter Attributwert in einem falschen Attribut eingetragen wurde, beispielsweise Marke = ‚Passat'.

- Ähnlich wie bei einer verletzten Attributabhängigkeit bilden *widersprüchliche Attributwerte* einen Konflikt zwischen zwei abhängigen Attributen. Jedoch kann die Abhängigkeit nicht direkt im Datenmodell spezifiziert werden, beispielsweise Ort = Berlin und PLZ = 12345.

- *Transposition* bedeutet, dass Informationen innerhalb eines Attributes in verschiedenen Reihenfolgen abgelegt sind, beispielsweise Name = ‚Max Mustermann' und Name = ‚Mustermann, Max'.

- Ein *Duplikat* liegt vor, wenn zwei Entitäten den gleichen materiellen oder immateriellen Gegenstand repräsentieren. Im diesem einfachen Fall wird im Neusystem lediglich ein Datensatz erstellt, um Redundanzen zu vermeiden.

- Ein *Datenkonflikt* bezeichnet den Fall, dass Duplikate widersprüchliche Angaben enthalten, die häufig nur durch nähere Betrachtung gelöst werden können, beispielsweise die Datensätze [Konzern = ‚Volkswagen', Gründungsjahr = 1910] und [Konzern = ‚Volkswagen', Gründungsjahr = 1937].

Eine Identifikation der genannten Fehler und Konflikte ist teilweise sehr arbeitsintensiv. Automatisierungen durch Datenbankabfragen oder Programme machen die Identifikation bei großen Datenmengen überhaupt erst möglich. Jedoch können nicht alle Fehler und Konflikte allein durch Automatismen und Abfragen erkannt werden, besonders dann, wenn ein Bezug zur realen Welt besteht, ist häufig der Mensch ein notwendiger Faktor.

Nach erfolgreicher Identifikation müssen nun geeignete Strategien entwickelt werden, um die erkannten Fehler und Konflikte zu beseitigen. Damit wird nicht nur die Vollständigkeit der Daten gewährleistet, sondern auch die Datenqualität entscheidend gesteigert und Redundanzen verhindert. Die Aufbereitung der Altdaten kann entweder direkt im Altsystem bzw. im Datenbestand des Altsystems oder mit Hilfe von Transformationsregeln während der Datenübertragung erfolgen. Informationen dazu

sind in die Abbildungstabelle einzufügen, die damit eine zentrale Rolle in der Datenmigration einnimmt.

Im Neusystem können Attribute vorhanden sein, denen kein Attribut des Altsystems zugeordnet werden kann, weil die Information im Altsystem bisher nicht erfasst oder gespeichert wurde. Soll die Information jedoch zukünftig im Neusystem erfasst werden, besteht die Möglichkeit Dummyeinträge für fehlende Attribute oder Attributwerte des Altsystems zu definieren. Wenn das Attribut zusätzlich eine Pflichteingabe ist, muss ein geeigneter Attributwert bestimmt werden.

Kann eine Aufbereitung einzelner Datensätze nicht erfolgen, weil die Rekonstruktion von Inhalten nicht erfolgen kann, muss entschieden werden, ob diese Datensätze tatsächlich übernommen werden sollen. Gleiches gilt für Datensätze, die bei der Datenübernahme Fehler hervorrufen würden. Hier ist im Einzelfall zu klären, ob und wie eine fehlerfreie Datenübernahme erfolgen kann. Kann keine Lösung gefunden werden, ist der Datensatz im Zweifelsfall von der Datenübernahme auszuschließen, um Fehlern bei der Arbeit mit dem neuen System vorzubeugen.

Im Kapitel 6.5.2.4 wurde das Problem höherstufiger Korrespondenzen beschrieben. Eine mögliche Lösung dieses Problems ist die Generierung eines zusätzlichen Attributes in jeden Quellentitätstyp. Dieses Attribut ist in der Abbildung 6.15 grün dargestellt und wird im Zuge der Datenaufbereitung mit einem konstanten Attributwert oder einem Dummywert gefüllt. Als Ergebnis stehen nun anstelle einer höherstufigen Korrespondenz eine Anzahl von 1:1-Beziehungen zwischen Quell- und Zielattributen.

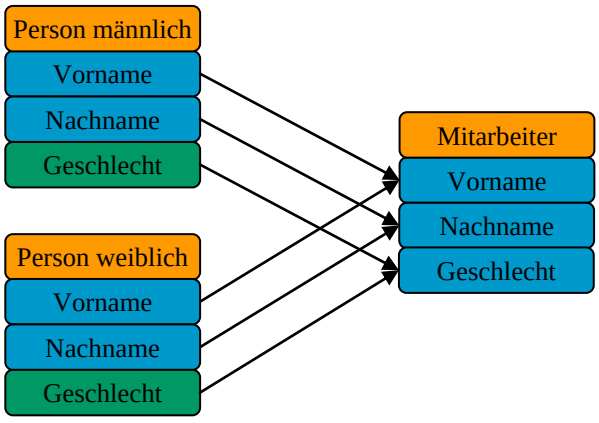


Abb. 6.15: Einführen von zusätzlichen Attributen

Mit Beendigung der Phase 2 sind entscheidende Vorbereitungsmaßnahmen für die Realisierung der Systemablösung abgeschlossen. Durch die Installation des Pilotsystems ist bereits eine erste Kontaktaufnahme mit dem neuen System, das so genannte ‚Look and Feel', möglich. Dadurch ist neben der Entwicklung der

Testszenarien eine Verfeinerung möglich, da kritische oder seltene Anwendungsfälle bereits in einer sehr frühen Phase im Pilotsystem betrachtet werden können. Die vorbereitenden Aktivitäten zur Datenmigration werden bereits in der nächsten Phase zur Erstellung von Testdaten genutzt. Hier werden ausgewählte Altdaten übertragen, um reale Testdaten zu erhalten. Gleichzeitig erfolgt damit eine erste Generalprobe für die Datenmigration.

6.5.3 Umsetzung

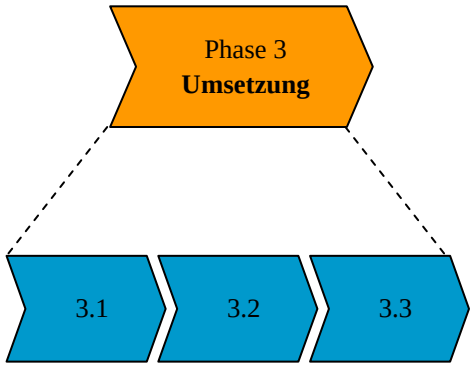


Abb. 6.16: Phase 3 – Umsetzung

Die *Umsetzungsphase* ist im Wesentlichen durch die technische Integration des neuen Systems gekennzeichnet. Darüber hinaus werden Testdaten angelegt und Maßnahmen ergriffen, um das System auf den Produktivstart vorzubereiten. Dazu werden entsprechende system- und benutzerrelevante Einstellungen auf Basis der in Phase 1 beschriebenen Anforderungen vorgenommen.

Tab. 6.6: Schritte der Phase 3 – Umsetzung

Schritt	Bezeichnung
3.1	Installation und Inbetriebnahme der Systemumgebung
3.2	Anpassen des Systems
3.3	Anlegen von Testdaten

6.5.3.1 Installation und Inbetriebnahme der Systemumgebung

Im Schritt 2.1 wurde ein Pilot- oder Testsystem eingerichtet. Bei der Installation und Inbetriebnahme konnten wertvolle Informationen zu Hardware- und Infrastrukturanforderungen gesammelt werden. Diese Erfahrungen werden nun eingesetzt um eine zügige Installation des späteren Produktivsystems zu gewährleisten.

Im ersten Schritt werden notwendige Basisdaten für den Betrieb des Systems eingerichtet. Dabei werden Benutzer und Rollen angelegt, sowie Rechte vergeben und weitere Einstellungen vorgenommen, die für eine Inbetriebnahme erforderlich sind (Vgl. Steinweg (2005), S. 181 f.).

Es werden dabei system- und benutzerrelevante Einstellungen vorgenommen. Systemrelevante Einstellungen beziehen sich hauptsächlich auf Hardware und die Systemarchitektur. Es werden beispielsweise Softwaremodule oder Teilsysteme eingerichtet und auf den gewünschten Releasestand gebracht, sowie notwendige Anpassungen durchgeführt. Das Datenbanksystem wird eingerichtet und entsprechende Einstellungen vorgenommen. Mit der Anbindung des Netzwerks können Schnittstellen zu anderen benachbarten Systemen eingerichtet werden, wie sie in Phase 1 bereits beschrieben wurden. Wichtig ist die Bereitstellung einer Schnittstelle zur Übernahme der Daten aus dem Altsystem, um Testdaten erzeugen zu können. Benutzerrelevante Einstellungen betreffen direkt oder indirekt den späteren Anwender. In erster Linie werden Benutzer und Rollen angelegt, um darauf aufbauend Berechtigungen zu vergeben (Vgl. Kohnke/Bungard (2005), S. 82 f.).

Mit Beginn der Inbetriebnahme des Systems werden eine Systemdokumentation und ein Anwenderhandbuch erstellt. Die Systemdokumentation stellt ein Referenzdokument für den Systemverwalter dar, mit dessen Hilfe sich die alle Ebenen der Systemarchitektur beschreiben lassen. Darüber hinaus sind Informationen zu Implementation, Betrieb und Wartung, Schnittstellen, Rollen, sowie Hintergrundwissen und sonstige Zusammenhänge enthalten, die für spätere Aktualisierungen bis hin zur möglichen Abschaltung und Datenübernahme von Interesse sein können. Die Systemdokumentation ist eine technische Darstellung des Systems und ein so genanntes lebendes Dokument, d. h. Änderungen von relevanten Informationen werden umgehend und ausnahmslos in die Systemdokumentation übertragen, um einen konsistenten Zustand der Dokumentation zu gewährleisten.

Das Anwenderhandbuch ist eine anschauliche Darstellung der Systemdokumentation und beschränkt sich dabei auf Informationen die speziell für den Anwender von Bedeutung sind. Es soll dem Anwender Hilfestellung geben und als Nachschlagewerk dienen, das allgemeine und spezielle Bedienungshinweise beinhaltet. Sowohl bei Verwendung von Standardsoftware, als auch vorhandener Individualsoftware sollte ein Anwenderhandbuch bereits vorliegen, das gegebenenfalls durch individuelle Erweiterungen ergänzt werden muss.

Neben der späteren Produktivumgebung wird eine Trainingsumgebung angelegt, die eine Kopie der Produktivumgebung darstellt und für Schulungs- und Testzwecke

genutzt wird. Dazu werden im letzten Schritt dieser Phase Testdaten durch die Übertragung von Daten aus dem Altsystem erzeugt.

6.5.3.2 Anpassen des Systems

Das gewählte Neusystem ist entweder eine vorhandene Standard- oder Individualsoftware. Beides sind bereits fertig entwickelte Systeme, für die ein bestimmter Einsatzzweck vorgesehen wurde und die daraufhin optimiert wurden.

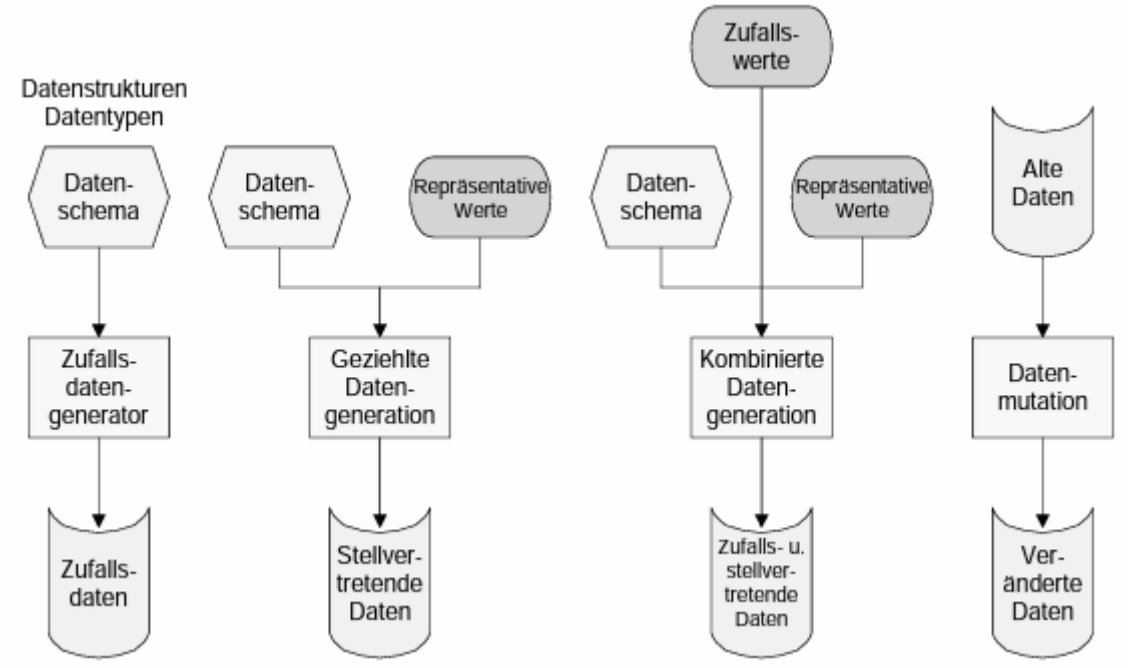
In beiden Fällen kann es notwendig sein, dass Anpassungen für den jetzigen Zweck des Systems erforderlich sind. Alle Anpassungen die nicht allein durch system- oder benutzerrelevante Einstellungen vorgenommen werden können, werden in diesem Schritt vorgenommen. Ist eine Erweiterung der gewählten Software durch Standardelemente oder Module nicht möglich, kann fehlende Funktionalität durch Individualprogrammierungen kompensiert werden.

Bei späteren Updates oder Upgrades sind diese eigenen Erweiterungen der Software mit besonderer Sorgfalt zu betrachten, da Standardsoftware im Regelfall keine Rücksicht auf Individualprogrammierungen nehmen kann. Ein späteres Release kann aber die fehlende Funktionalität enthalten und macht die eigene Erweiterung überflüssig. Um Komplikationen zu vermeiden, sollte ein sauberes Entfernen der eigenen Komponenten möglich sein. Gleichzeitig sollten Modifikationen nur so groß wie nötig sein, denn jede Modifikation bedeutet eine Entfernung von der Ursprungssoftware und steigert somit das Risiko von Problemen bei späteren Aktualisierungen oder Erweiterungen.

6.5.3.3 Anlegen von Testdaten

Für die Durchführung von Testszenarien in Phase 4 werden Testdaten benötigt. Aber bereits in Phase 3 können Testdaten verwendet werden, um system- oder benutzerrelevante Einstellungen und Erweiterungen zu prüfen.

Testdaten werden in Testszenarien unter anderem dazu verwendet, Funktionen und Regeln zu prüfen. Es gibt prinzipiell vier Ansätze zur Erstellung der Testdaten, die in Abbildung 6.17 dargestellt sind (Vgl. Sneed et al. (2007), S. 102 ff.).



Quelle: Sneed et al. (2007), S. 103.

Abb. 6.17: Ansätze zur Testdatenerstellung

Der *blinde Ansatz* zur Testdatenerstellung basiert auf der Erzeugung von Zufallsdaten. Dabei werden völlig unabhängig von der Verwendung Daten mit einem Zufallsgenerator erzeugt, die lediglich die Anforderungen des Datenmodells erfüllen. Basis der Erzeugung sind Datentyp, Länge und Bezeichnung. Für Zahlenwerte können zusätzlich Einschränkungen beispielsweise für den Wertebereich getroffen werden. Mit dem blinden Ansatz wird eine systematische Streuung von Daten mit der Hoffnung verfolgt, möglichst alle Testfälle abzudecken. Der Vorteil liegt in dem geringen Aufwand zur Erzeugung von Zufallsdaten. Dem gegenüber steht ein höherer Aufwand bei der Durchführung der Testfälle.

Der *gezielte Ansatz* betrachtet direkt die zu prüfenden Funktionen. Dabei werden die Testdaten so gewählt, dass bestimmte Funktionen ausgelöst bzw. ausgeführt werden können. Damit verbunden ist ein bestimmter Ausgang im Testfall, der direkt geprüft werden kann. Soll beispielsweise eine Funktion ‚alle Mitarbeiter mit einem Alter über 60 werden entlassen' geprüft werden, so wird ein Mitarbeiter mit dem Alter 60 und ein weiterer mit dem Alter 61 angelegt. Der Vorteil des gezielten Ansatzes liegt in der großen Testfallabdeckung mit weniger Daten, jedoch ist zur Erstellung der Daten ein erhöhter Aufwand notwendig.

Der *kombinierte Ansatz* stellt eine Vereinigung der beiden zuvor genannten Ansätze dar. Zunächst werden dabei Zufallszahlen wie beim blinden Ansatz generiert und

anschließend anhand von Funktionsspezifikationen manipuliert, um gezielt Funktionen testen zu können. Der Vorteil liegt darin, dass für kritische Funktionen gezielt Testdaten bereitgestellt werden. Der überwiegende Teil besteht jedoch aus Zufallszahlen. Der kombinierte Ansatz stellt somit einen Kompromiss zwischen blinden und gezielten Ansatz dar. Dabei werden die Vorteile beider Ansätze vereint, der Aufwand in Grenzen gehalten und eine relativ große Testabdeckung erreicht.

Der *Mutationsansatz* verfolgt die Strategie, einen bestimmten Datenbestand mutieren zu lassen. Der Testdatenbestand umfasst dabei lediglich eine minimale Menge an Daten, die notwendig ist, um vorhandene Funktionen zu testen. Mit zunehmender Funktionalität werden weitere Testdaten ergänzt, um die hinzugefügte Funktionalität zu testen. Das Problem dabei ist, dass durch das Minimum an Testdaten nur eine niedrige Testabdeckung erreicht wird.

Bei der Erstellung von Testdaten ist im Wesentlichen zu berücksichtigen, dass damit gezielt die Durchführung der entwickelten Testszenarien bzw. der Testphase gewährleistet werden muss. Generierte Zufallsdaten lassen sich allenfalls für Lasttests verwenden, für Funktionstests sind sie nur bedingt geeignet. Hier werden Testdaten benötigt, die zumindest fiktive Werte im späteren Anwendungsfall repräsentieren. Diese Testdaten können manuell erstellt werden. Da in der Regel eine größere Anzahl von Testdaten benötigt wird, ist das mit einem hohen Aufwand verbunden, der zum Teil nicht gerechtfertigt ist.

Da die Datenmigration in Phase 2 bereits vorbereitet und im Schritt 3.1 eine Schnittstelle zur Übernahme von Daten aus dem Altsystem geschaffen wurde, lassen sich repräsentative Testdaten aus ausgewählten Altdaten erzeugen. Gleichzeitig wird damit die Datenübertragung als Generalprobe für die Datenmigration getestet. Die Auswahl der für Testzwecke bestimmten Daten kann anhand der Testszenarien oder nach einem geeigneten Schema erfolgen. Aufgrund von Datenschutzbestimmungen ist zu berücksichtigen, dass mögliche sensible Daten identifiziert werden, um sie im Zuge der Erstellung von Testdaten zu verzerren. Des Weiteren werden die Testdaten in der Regel zu Schulungszwecken verwendet. Schulungen mit nicht ganz aktuellen, aber dafür realen Testdaten erlauben dem Schulungsteilnehmer mehr Nähe zur späteren Nutzung des Systems. Die erforderlichen Daten für Fallstudien der späteren Schulungen können somit bereits bei der Erstellung von Testdaten berücksichtigt werden.

Mit der Erstellung der Testdaten wird die Phase 3 abgeschlossen. Die Systemumgebung wurde hergestellt und notwendige Einstellungen für den Betrieb des Systems vorgenommen. Ebenso wurden Testdaten als Voraussetzung für die folgende Phase des Testens erzeugt.

6.5.4 Testen

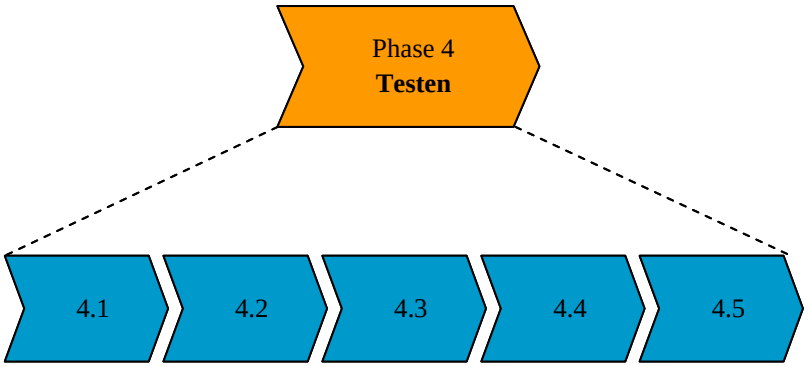


Abb. 6.18: Phase 4 – Testen

In der *Testphase* geht es hauptsächlich darum, die Qualität des Systems zu prüfen und Risiken auszuschließen, die einen störungsfreien Betrieb gefährden könnten. Dazu werden die in Phase 2 definierten Testszenarien und eine Reihe weiterer Tests durchlaufen. Bei der Auswertung der Testergebnisse werden, wenn notwendig, Maßnahmen definiert, die entsprechende Fehler und Konflikte lösen sollen. Als Vorbereitung der nächsten Phase werden Schulungen in vorbereitet und geplant. Im letzten Schritt werden bereits Hauptanwender geschult, um ihrerseits Vorbereitungen für die Phase der Einführung treffen zu können.

Tab. 6.7: Schritte der Phase 4 – Testen

Schritt	Bezeichnung
4.1	Testszenarien durchführen
4.2	Auswertung der Tests
4.3	Umsetzung neuer Anforderungen
4.4	Vorbereitung der Schulungen
4.5	Hauptanwenderschulung

6.5.4.1 Testszenarien durchführen

Im Kapitel 6.5.2.2 wurden bereits verschiedene Testarten in Zusammenhang mit Qualitätsmerkmalen genannt, die nun mit den zuvor erstellten Testdaten durchgeführt werden sollen:

- Funktionstest
- Robustheitstest
- Benutzbarkeitstest

- Installationstest
- Performancetest.

Der *Funktionstest* ist die Basis für die fachliche Verifikation des Systems, die mit Hilfe von Use-Cases oder Sequenzdiagrammen veranschaulicht werden können. Weil der Funktionstest eine umfangreiche Vorarbeit erfordert, wurde er bereits in der Phase 2 vorbereitet. Dazu wurden in Zusammenarbeit mit dem Fachbereich Testszenarien entwickelt, die nun im neuen System durchlaufen werden. Dabei wird im Wesentlichen geprüft, ob die Eingabe bestimmter Daten und das Ausführen diverser Funktionen zu Ergebnissen bzw. Ausgaben auf dem Bildschirm führen, die den Erwartungen entsprechen. Eine detaillierte Beschreibung des Funktionstest ist in Kapitel 6.5.2.2 zu finden, eine Checkliste zum Funktionstest in Anhang A.

Mit dem *Robustheitstest* wird zunächst die Reife des Systems überprüft. Dabei wird durch eine Einschränkung von Systemressourcen getestet, wie stabil sich der Zustand des Systems verhält. Darüber hinaus wird geprüft, wie robust das System auf Verletzungen von Schnittstellenspezifikationen, beispielsweise ungültige oder überlange Anfragen von benachbarten Systemen, reagiert. Zuletzt wird untersucht, wie sich das System bei Situationen verhält, die eine Wiederherstellung des Systems erfordern. Dieser Umstand kann beispielsweise dann eintreten, wenn ein Teil der Systemarchitektur durch einen Stromausfall abstürzt oder das Netzwerk ausfällt. Dann muss das System einen konsistenten Zustand wiederherstellen, um den weiteren Betrieb zu gewährleisten (Vgl. Siedersleben (2003), S. 303). Im Anhang B findet sich eine Checkliste für den Robustheitstest.

Durch den *Benutzbarkeitstest* wird die Verständlichkeit des Systems für den Anwender überprüft. Dazu werden eine Reihe potentieller Endanwender aufgefordert, repräsentative Funktionen gegebenenfalls unter Zuhilfenahme des Anwenderhandbuchs oder der Hilfefunktion auszuführen. Dadurch können nicht nur Erkenntnisse über die Bedienbarkeit und Akzeptanz des Systems an sich gewonnen werden, sondern auch Informationen, inwieweit Anwenderhandbuch und Hilfefunktion ausreichend für die korrekte Bedienung beschrieben sind (Vgl. Siedersleben (2003), S. 303 f.; Sneed et al. (2007), S. 132). Ergebnisse aus dem Benutzbarkeitstest können weiterhin dazu verwendet werden, um den Schulungsaufwand zu bestimmen, da der Zeitaufwand für die Erlernbarkeit des Systems eine Erkenntnis aus diesem Test ist. Eine Checkliste für den Benutzbarkeitstest ist im Anhang C zu finden.

Der *Installationstest* ist gerade dann von großer Bedeutung, wenn die Anwender eine Vielzahl verschiedener Rechnerumgebungen verwenden. Dazu müssen die

unterschiedlichen Umgebungen identifiziert werden und notwendige Installationen geprüft werden. Gleichzeitig wird neben der Installation an sich, die Verträglichkeit des Systems mit der Software der einzelnen Rechnerumgebungen geprüft (Vgl. Siedersleben (2003), S. 304). Eine Checkliste ist im Anhang D zu finden.

Das Ziel des *Performancetests* ist eine Überprüfung des Zeitverhaltens des Systems. Während der Simulation verschiedener Systemauslastungen von Unterlast bis Überlast wird das Verhalten der Systemressourcen, sowie die Länge von Antwort- und Wartezeiten beobachtet. Der Begriff Lasttest ist als Teil des Performancetests zu betrachten, bei dem in erster Linie das Testen der Grenzen des Systems und die Bestimmung der maximalen Last im Vordergrund steht (Vgl. Siedersleben (2003), S. 304 ff.).

Für die Ausführung der genannten Tests kann es sinnvoll sein, Werkzeuge einzusetzen. Dabei gibt es für nahezu jeden Test eine Vielzahl von Softwareprodukten, die das Testen unterstützen können. An dieser Stelle wird jedoch nicht tiefer darauf eingegangen. Der Sinn der Werkzeuge liegt darin, Testaktivitäten zu erleichtern oder zu automatisieren. Das kann beispielsweise bei Lasttests sehr hilfreich sein, um einen gleichzeitigen Zugriff mehrerer Anwender zu simulieren, ohne dies manuell tun zu müssen.

Das Ausführen der Tests kann im Trainingsystem stattfinden, da es eine Kopie der Produktivumgebung ist und dort die erforderlichen Testdaten vorhanden sind. Besonders wichtig ist eine geeignete Dokumentation der Testergebnisse, um die Voraussetzung für den nächsten Schritt, die Auswertung der Testergebnisse, zu schaffen.

6.5.4.2 Auswertung der Tests

Eine Auswertung der Tests soll dazu dienen, den Zustand des Systems zu beurteilen und Maßnahmen abzuleiten, die mögliche Schwachstellen beseitigen. Die Testergebnisse werden aufbereitet und mit den erwarteten Werten aus der Planung der Tests in Phase 2 in Form eines Soll-Ist-Vergleichs gegenübergestellt.

Durch eine Analyse der Abweichungen können in der Regel Maßnahmen definiert werden, um mögliche Fehler zu beseitigen oder fehlerhafte Einstellungen zu korrigieren. Gleiches gilt, wenn vom System Fehlercodes oder Warnungen bei der Ausführung von Tests angezeigt werden.

6.5.4.3 Umsetzung neuer Anforderungen

Dieser Schritt ähnelt im Wesentlichen den Aktivitäten, die bereits im Kapitel 6.5.3.2 beschrieben wurden. Ausgangspunkt sind jedoch die Maßnahmen und Anforderungen, die aus den Tests bzw. der Testauswertung hervorgegangen ist.

Jede Umsetzung von Maßnahmen sollte einen erneuten Test und die dazugehörige Auswertung nach sich ziehen, um den Erfolg der umgesetzten Maßnahmen zu bestätigen. Hierbei können durch die Auswertung erneut Maßnahmen definiert werden, wenn weiterhin Abweichungen oder Fehler auftreten. Dabei ist es nicht auszuschließen, dass neue Punkte erscheinen, die zuvor gar nicht auftreten konnten, weil beispielsweise eine Funktion nicht vollständig getestet werden konnte.

Die Umsetzung neuer Anforderungen ist erst beendet, wenn keine Anforderungen mehr bestehen, die den produktiven Betrieb des neuen Systems wesentlich gefährden könnten.

6.5.4.4 Vorbereitung der Schulungen

Für einen erfolgreichen Einsatz des neuen Systems ist es erforderlich, dass die Anwender mit dem Umgang des Systems vertraut gemacht werden. Dazu werden in der Einführungsphase Schulungen der Anwender in der in Phase 3 erstellten Trainingsumgebung des Systems durchgeführt. Darin sind bereits Daten enthalten, die für Testzwecke verwendet wurden. Nun ist zu prüfen, inwieweit diese Daten für Schulungszwecke geeignet sind, gegebenenfalls sind weitere Daten zu erzeugen, beispielsweise durch erneute Übertragung ausgewählter Altdaten.

Zur Durchführung der Schulungen werden zum einen Schulungsunterlagen benötigt, zum anderen muss ein Schulungsplan erstellt werden, um eine projektgerechte Koordination zu gewährleisten. Dabei ist zu bedenken, in welchen Sprachen die Schulungen, sowie die Unterlagen angeboten werden sollen.

Als Grundlage für die Erstellung der Schulungsunterlagen dienen Rollenbeschreibungen und Arbeitsabläufe der zukünftigen Anwender. Hilfreich ist es, Informationen über den Kenntnisstand der Mitarbeiter einzuholen, um entsprechend geeignete Schulungsunterlagen für möglicherweise verschiedene Schulungsumfänge auszurichten. Die Schulungsumfänge können dabei beispielsweise nach unterschiedlichen Arbeitsabläufen und Tätigkeitsfeldern der Anwendergruppen differenziert werden (Vgl. Steinweg (2005), S. 182 f.; Kletti (2007), S. 250 f.). Darüber hinaus kann auf das Anwenderhandbuch zurückgegriffen werden, um konkrete Angaben oder Bedingungen

einzelner Funktionalitäten zu berücksichtigen. Im Wesentlichen sollen in den Schulungen den Anwendern der spätere Arbeitsablauf aufgezeigt werden. Dazu werden verschiedene Fallstudien oder Szenarien entwickelt, die den täglichen Umgang mit dem System simulieren sollen. Zur Bestimmung des Schulungsaufwands können gegebenenfalls Erfahrungen aus dem Benutzbarkeitstest verwendet werden.

Die Aktualität der Schulungsunterlagen ist ein entscheidender Faktor, denn unvollständige oder veraltete Szenarien, die sich nicht wie vorgegeben durchführen lassen, führen im schlechtesten Fall dazu, dass sich der Schulungserfolg bei den Anwendern nicht einstellt. Des Weiteren kann es zu negativen Auswirkungen auf die Akzeptanz und Motivation der Anwender im Umgang mit dem neuen System kommen.

Der Schulungsplan enthält Informationen über Schulungstermine und -umfänge verschiedener Anwendergruppen. Je nach Größe des Systems lassen sich verschiedene Anwendungsgebiete differenzieren, für die eine separate Schulung sinnvoll ist (Vgl. Kletti (2007), S. 250 f.). In Tabelle 6.8 ist ein Schulungsplan beispielhaft dargestellt. Darin sind fünf Anwendergruppen definiert, die zu bestimmten Terminen entsprechende Schulungen besuchen.

Tab. 6.8: Beispiel für einen Schulungsplan

Schulungen		**System-admin.**	**Haupt-anwender**	**Schicht-leiter**	**Auftrags-bearbeiter**	**Produktion**
Systemüberblick		KW 41	KW 44	KW 47	KW 48	KW 48
Anwender-schulung	Auftragsdaten		KW 45		KW 49	
	Material- und Produktionslogistik		KW 46			KW 50
Adminschulung		KW 42				
Datenerfassung			KW 48	KW 48		

In Anlehnung an Kletti (2007), S. 251.

Der Schulungsplan ist mit der Zeitplanung aus Phase 1 abzustimmen. Wird die Einführung des Systems stufenweise für einzelne Standorte durchgeführt ist dies bei der Schulungsplanung zu berücksichtigen. Des Weiteren muss sichergestellt werden, dass die geplanten Ressourcen aus Phase 1 nun tatsächlich verfügbar sind.

Bisher wurde davon ausgegangen, dass die Schulungen intern vorbereitet und durchgeführt werden. Gerade bei Standardsoftware gibt es jedoch häufig eine Reihe externer Dienstleister, die entsprechende Schulungen anbieten. Die Entscheidung darüber ist abhängig von lokalen Gegebenheiten und muss individuell geprüft werden.

6.5.4.5 Hauptanwenderschulung

Neben normalen Anwendern eines Systems können in der Regel so genannte Hauptanwender oder ‚Key-User' identifiziert werden. Die Hauptanwender nehmen dabei eine zentrale Bedeutung, beispielsweise die fachliche Betreuung eines Anwendungsbereichs oder die lokale fachliche Betreuung eines Systems, ein und sind für lokale Anwender eine erster Anlaufpunkt für Fragen und Hilfestellungen. Ein Hauptanwender besetzt häufig leitende Funktionen und ist verantwortlich für einen Organisationsbereich. Er ist im Regelfall in Entscheidungen für den Zeitpunkt der Einführung des Systems eingebunden. Um seinerseits Vorbereitungen für die Einführung treffen zu können, ist es erforderlich umfassende Kenntnisse über das System zu besitzen, weshalb die Hauptanwender bereits zu diesem Zeitpunkt entsprechend geschult werden.

Mit Abschluss der Testphase ist das System bereit für den Produktiveinsatz. Letzte Einstellungen und Anpassungen wurden vorgenommen, um einen störungsfreien Betrieb zu ermöglichen. Die Vorbereitung und Planung der Schulungen ist eng mit der Zeitplanung verknüpft und abhängig von der Strategie der Einführung des Systems. Die Hauptanwender wurden bereits geschult, um ihrerseits organisatorische Maßnahmen einleiten zu können, die eine reibungslose Einführungsphase unterstützen.

6.5.5 Einführung

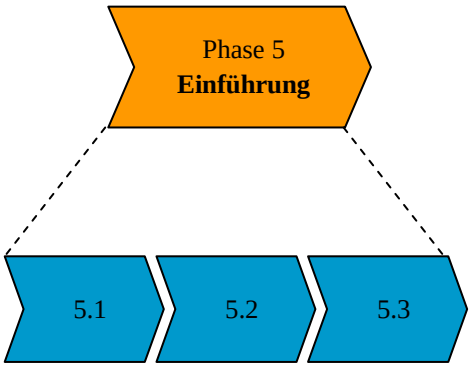


Abb. 6.19: Phase 5 – Einführung

Die Einführungsphase beschreibt die Schritte, die notwendig sind, um die Systemablösung abzuschließen. Durch die Inbetriebnahme des Systems in Phase 3 und das anschließende Testen in Phase 4 sind systemseitig alle Vorbereitungen getroffen worden. Die Hauptanwender wurden bereits geschult und treffen ihrerseits entsprechende Vorbereitungen.

Durch die Schulung der Anwender wird der korrekte Umgang mit dem System vorgestellt und sämtliche Änderungen zum Altsystem aufgezeigt. Für die Arbeit mit dem System werden jedoch in der Regel Altdaten benötigt, die durch eine Datenmigration übertragen werden. Nach Abschluss der Datenübernahme wird das neue System produktiv gesetzt und unterstützt von nun an den täglichen Aufgabenbereich der Anwender.

Der so genannte ‚Point of no return' bezeichnet den Zeitpunkt, von dem an keine Rückkehr zum Altsystem möglich ist (Vgl. Siedersleben (2003), S. 62). Das Risiko des Scheiterns ist jedoch sehr gering, da in den vorhergehenden Phasen, entsprechende Vorbereitungen immer im Hinblick auf das Projektziel getroffen wurden.

Tab. 6.9: Schritte der Phase 5 – Einführung

Schritt	Bezeichnung
5.1	Schulung
5.2	Datenmigration
5.3	Produktivstart

6.5.5.1 Schulung

In Schritt 4.4 wurde ein Schulungsplan erstellt. Darin enthalten sind Informationen welcher Anwendergruppen welche Schulungen zu welchem Zeitpunkt besuchen. Die Schulung kann dabei durch unternehmensinterne Ressourcen durchgeführt werden oder mit Hilfe externer Anbieter.

Für die Durchführung der Schulungen sind die geplanten Ressourcen bereitzustellen und organisatorische Vorbereitungen durch Unterstützung der bereits geschulten Hauptanwender vorzunehmen, die zum Teil selbst die Schulung der Anwender in ihrem Verantwortungsbereich vornehmen. Die Schulungen finden in der Trainingsumgebung statt, die bereits in Phase 3 eingerichtet wurde.

Bei stufenweisen Einführungen ist es hilfreich Feedbacks aus den Schulungen in spätere Schulungen einfließen zu lassen. Gegebenenfalls sind die Schulungsunterlagen dazu anzupassen.

6.5.5.2 Datenmigration

Bei Systemablösungen soll im Regelfall nach der Einführung eines neuen Systems auf den ‚alten' Daten weitergearbeitet werden. Dazu ist eine Datenübernahme erforderlich, die bereits in Phase 2 umfassend vorbereitet wurde. In einem ersten Schritt konnte der Datenumfang durch eine Bestimmung der Daten, mit denen tatsächlich gearbeitet wird, mitunter deutlich reduziert werden. Nach dem Erzeugen der Abbildungstabellen wurden möglicherweise weitere Entitätstypen oder Attribute von der Datenübernahme ausgeschlossen, da sie nicht im neuen System abgebildet werden oder redundant im alten System vorhanden waren. Während der Aufbereitung der Daten wurde mit dem Ausschluss einzelner Datensätze eine weitere Reduzierung erreicht.

Während der Phase 3 wurde das System eingerichtet und eine Schnittstelle zum Altsystem hergestellt. Darüber wurde bereits eine Anzahl ausgewählter Datensätze übertragen, die jedoch alle Einträge der Abbildungstabelle tangiert hat. Damit wurde sichergestellt, dass die Datenübertragung für sämtliche Attribute funktioniert und die Datenmigration hinreichend vorbereitet ist.

Die Strategie der Datenmigration ist abhängig von der Strategie der Systemeinführung. Dafür gibt es eine Reihe verschiedener Ansätze. Grundsätzlich sind direkte und stufenweise Strategien voneinander zu unterscheiden. Direkte Strategien, wie der Big-Bang-Ansatz, basieren auf einer einzigen Datenübernahme für die Produktivsetzung des neuen Systems. Dieses Vorgehen ist mit entsprechenden Risiken verbunden, da der Datenbestand zu einem Zeitpunkt übertragen werden muss und das System in einem Schritt umgeschaltet wird (Vgl. Kapitel 5.3). Bei stufenweisen Strategien können weitere Ansätze unterschieden werden, wie sie in Kapitel 5 beschrieben sind. Die Entscheidung darüber, welche Strategie gewählt wird, wurde bereits in der Planungsphase der Systemablösung getroffen.

Die Datenmigration an sich ist ein technischer Vorgang, der je nach Größe des zu übertragenden Datenbestands einen gewissen Zeitraum beansprucht. Während dieses Zeitraumes dürfen keine Änderungen am Datenbestand erfolgen, um die Konsistenz der Daten zu gewährleisten. Sowohl das Altsystem, als auch das Neusystem ist für eine Bearbeitung der Daten gesperrt und ein produktiver Einsatz während der Datenmigration nicht möglich. Das kann dazu führen, dass beispielsweise die Produktion bei industriellen Unternehmen nur eingeschränkt oder gar nicht möglich ist. Weil jedoch damit in den meisten Fällen ein finanzieller Verlust verbunden ist, muss dieser Zeitraum so kurz wie möglich gestaltet werden, weshalb bereits in Phase 2 die genannten Einschränkungen vorgenommen wurden. Die Datenmigration wird durch eine geeignete Verifikation der übertragenden Daten abgeschlossen.

Der Zeitpunkt der Datenmigration ist generell unmittelbar vor dem Produktivstart zu wählen, da nach erfolgter Datenübernahme mit dem neuen System gearbeitet werden muss. Andernfalls ist eine weitere Datenübernahme erforderlich, um die Datenkonsistenz zu gewährleisten.

6.5.5.3 Produktivstart

Die Produktivsetzung eines neuen Systems bezeichnet die Publikation des Projektabschlusses und die offizielle Systemübergabe an den Auftraggeber des Projektes (Vgl. Kletti (2007), S. 260). Dazu musste das System, das bereits vor der Systemablösung ausgewählt wurde, zunächst in Betrieb genommen werden. Die anschließende Testphase stellte sicher, dass zum einen die gestellten Anforderungen an das System erfüllt werden und zum anderen der Einsatz des Systems fehlerfrei möglich ist. Die Anwender wurden bereits geschult und sind durch die Verwendung realer Testdaten gut vorbereitet auf die Arbeit mit dem neuen System. Durch die Datenmigration wurden die benötigten Altdaten zur Verfügung gestellt. Nun kann der produktive Einsatz des Systems erfolgen. In Abhängigkeit von der gewählten Einführungsstrategie ist analog zur Datenmigration eine stufenweise Produktivsetzung möglich (Vgl. Kapitel 5).

Die Abschaltung des Altsystems erfolgt in der Regel unmittelbar nach der Datenübernahme. Das bedeutet in erster Linie, dass der Zugang zum System für Anwender gesperrt bleibt, da diese sonst Daten verändern könnten. Das Altsystem wird fortan nur noch so lange betrieben, wie es für eine Archivierung der Daten erforderlich ist.

In Phase 1 wurden die Ziele des Projektes in Form kritischer Erfolgsfaktoren festgelegt. An dieser Stelle wird die Erreichung dieser Ziele geprüft, die zugleich den Erfolg des Projektes ausmachen. Etwaige Abweichungen oder Nacharbeiten werden in einem Übergabeprotokoll festgehalten.

Der Produktivstart eines neuen Systems ist ein wichtiger Meilenstein im gesamten Systemablösungsprojekt, der zugleich dessen erfolgreiches Ende bedeutet. Mit der Unterschrift auf dem Übergabeprotokoll ist die Phase der Einführung abgeschlossen und die Systemablösung gelungen. Nachfolgende Prozesse bestimmen von nun an Betrieb und Wartung des neuen Systems bis auch dieses früher oder später durch eine erneute Systemablösung ersetzt wird.

Erfahrungen aus dem gesamten Projekt sollten entsprechend gesichert werden, um bei der Planung und Ausführung zukünftiger Systemablösungen darauf zurückgreifen zu können. Diese so genannten ‚Best Practises' sind dann ein Vorteil, um mögliche Fehler und Probleme besser zu erkennen oder gar von vorneherein auszuschließen. Des Weiteren lassen sich durch die erworbenen Erfahrungen bevorstehende Projekte genauer abschätzen, sowie Kosten sparen und Ressourcen schonen.

7 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde ein Vorgehensmodell zur Systemablösung entwickelt. Ausgangspunkt dafür war die zunehmende Prozessorientierung der Unternehmen und die Tendenz zu einer unternehmensweiten Bereitstellung von Daten, die immer häufiger dazu führen wird, dass einzelne Systeme durch neue ersetzt werden.

Anhand einiger aktueller Ablösungsprojekte der Volkswagen AG wurde festgestellt, dass kein standardisiertes Vorgehen vorhanden war. Es wurden jeweils individuelle Herangehensweisen entwickelt, wobei ein Wissenstransfer weder zu parallelen noch zu nachfolgenden Projekten stattfand. Da kein einheitliches strukturiertes Vorgehen erfolgte, erleideten die Projekte zum Teil herbe Rückschläge oder immense Zeitverzögerungen. Dies war zum Teil auf mangelndes Projektmanagement und ungenügende Tests der neuen Systeme zurückzuführen. So traten Fehler erst nach der Produktivsetzung der Systeme auf und beeinträchtigten die Arbeit damit zum Teil sehr stark. Die Beseitigung der Fehler war entsprechend aufwendig und verursachte immense Kosten. Insgesamt nahmen die bisherigen Systemablösungen zu viel Zeit und Ressourcen in Anspruch und verursachten dazu noch erhöhte Kosten. Darüber hinaus gab es noch gescheiterte Projekte, die nach einem Produktivstart durch einen erneuten Systemwechsel zurück auf das Altsystem weitere Kosten verursacht haben.

Es wurden weitere Auslöser für eine Systemablösung identifiziert und kategorisiert. So lassen sich neben internen und externen Auslösern, technologische, funktionale und obligatorische Ursachen unterscheiden. Diese geben zum Teil verschiedene Rahmenbedingungen vor und haben somit einen Einfluss auf die Systemablösung.

Auf der Suche nach geeigneten Strategien konnten alternative Ansätze gefunden werden, die Vor- und Nachteile haben. Sie stützen sich alle auf eine gleichzeitige Softwareentwicklung, lassen sich im Kern jedoch auch für Systemablösungen anwenden.

Eine Systemablösung kann nur durch eine strukturierte und konsequente Systematik zügig und zielorientiert zum Erfolg geführt werden. Dazu wurde ein Vorgehensmodell entwickelt, das auf die Erfahrungen aktueller Projekte bei der Volkswagen AG aufbaut, verschiedene Strategien bei der Systemablösung ermöglicht und unterschiedliche Ursachen berücksichtigt.

Die Idee dahinter ist die Aufteilung der Systemablösung in einzelne Phasen. Diese Phasen werden wiederum in Schritte unterteilt und bilden so übersichtliche Projekteinheiten und Arbeitspakete, für die durch eine umfassende Projektorganisation Verantwortlichkeiten festgelegt werden. Auf Basis dieser Projekteinheiten können

Ressourcen- und Zeitbedarfe ermittelt werden, die zusammen den Gesamtbedarf des Projektes bilden. Durch eine klare Definition von Zielen und Teilzielen kann der aktuelle Projektstatus jederzeit daran gemessen werden. Eine detaillierte Kostenplanung wiederum auf Basis der Projekteinheiten ermöglicht eine Kontrolle der Projektkosten.

Eine Risikoanalyse identifiziert alle Faktoren, die einen kosten-, termin- und sachgerechten Projektausgang gefährden. Nach einer Bewertung der Risiken werden Gegenmaßnahmen entwickelt, um deren Eintritt zu vermeiden oder die Auswirkungen auf das Projekt einzuschränken.

Prinzipiell wird mit dem Vorgehensmodell der Gedanke verfolgt, so detailliert wie nötig zu planen und im aktuellen Schritt bereits Vorkehrungen für den nächsten zu treffen. Dies wird in jeder einzelnen Phase deutlich und ermöglicht eine effiziente und zielorientierte Arbeitsweise. Durch klar definierte Verantwortungen werden Zuständigkeitskonflikte verhindert und der Projektverlauf weiter forciert.

Nach umfangreicher Vorbereitung und Umsetzung von Anforderungen werden ausgiebige Tests durchgeführt, denen in aktuellen Projekten der Volkswagen AG zu wenig Aufmerksamkeit gewidmet wurde. Eine effektive Schulung der Anwender wird erreicht, indem reale Produktivdaten aus dem Altsystem zu Schulungszwecken zur Verfügung gestellt werden.

Durch umfassende Information und Einbeziehung aller beteiligten Organisationseinheiten wird die Systemablösung mit der Datenmigration und der Produktivsetzung zügig abgeschlossen. Dadurch wird ein fristgemäßer Einsatz des neuen Systems ermöglicht, durch den Mehrkosten verhindert werden und planmäßige Einsparungen erfolgen können.

Das Vorgehensmodell dient als Vorlage für eine Prozessbeschreibung der Systemablösung bei der Volkswagen AG. Die Systemablösung soll im Rahmen der Prozessorientierung definiert und in das Management Handbuch für Qualität aufgenommen werden.

Mit diesem Vorgehensmodell ist die Grundlage für ein strukturiertes und effizientes Verfahren zur Systemablösung geschaffen worden. Der Erfolg in der Praxis konnte im Rahmen dieser Diplomarbeit allerdings nicht geprüft werden. Er ist durch empirische Untersuchungen anhand konkreter Aufgabenstellungen nachzuweisen.

Anhang

A Checkliste für den Funktionstest

Die nachfolgende Checkliste ist aus Siedersleben (2003), S. 302 übernommen.

Funktionsumfang

- Sind alle Anforderungen und Anwendungsfälle durch Tests abgedeckt?
- Sind in den Zustandsmodellen alle Zustände erreicht und alle Übergänge durchlaufen?

Richtigkeit

- Sind alle Eingabefelder mit gültigen und ungültigen Werten getestet?
- Sind alle Ausgaben auf Richtigkeit überprüft? Sollwerte vorhanden?
- Sind die Datumsfunktionen getestet? (ungültige Datumswerte, Schaltjahre, …)
- Sind landes- oder sprachabhängige Darstellungen getestet? (Datum, Währung, Nachkomma, Tausender-Trennung, …)
- Sind sprachabhängige Zeichensätze getestet? (Western, Eastern, Umlaute)
- Sind fachliche Fehler getestet? Werden Fehler richtig gemeldet?
- Ist das System nach jedem Fehler in einem definierten Zustand?

Sicherheit und Datenschutz

- Sind die Berechtigungsprüfungen getestet?
- Ist der Schutz gegen böswillige Hacker und interne Angriffe geprüft?
- Ist die Transaktionssicherheit bei parallelen Zugriffen gegeben?
- Funktioniert das Sperrverfahren (optimistisch oder sperrend) bei langen Transaktionen?

Interoperabilität

- Sind alle Nachbarsysteme in den Test einbezogen?
- Sind alle systemübergreifenden Anwendungsfälle getestet?

B Checkliste für den Robustheitstest

Die nachfolgende Checkliste ist aus Siedersleben (2003), S. 303 übernommen.

Reife

- Verhält sich die Software an den Grenzen von Systemressourcen gutmütig (kein Hauptspeicher mehr, kein Plattenplatz mehr, DB-Tablespace-Overflow)? Es dürfen keine Daten verloren gehen, keine inkonsistenten Einträge in der Datenbank entstehen.
- Treten unter Last bei parallelen Zugriffen Verklemmungen auf?
- Stürzen Server nach bestimmter Laufzeit ab (Speicherlecks)?
- Werden Protokolldateien, temporäre Dateien bereinigt oder läuft die Platte voll?

Fehlertoleranz, Schnittstellenrobustheit

- Reagiert die Software auf ungültige Daten (Schrottdaten) an den Schnittstellen mit sinnvollen Fehlermeldungen?
- Kann die Software durch ungültige/überlange Anfragen von außen zum Absturz gebracht werden (denial of service, ping of death)?

Wiederherstellbarkeit (Recovery)

- Liefert die Software bei Ausfall des Datenbanksystems oder einzelnen Servers oder einer einzelnen Platte (Platte voll) eine sinnvolle Fehlermeldung?
- Fährt das System nach einem Absturz von Client / Netzwerk / Server, z.B. durch Stromausfall, wieder von alleine hoch? Sind Daten dann noch konsistent?
- Können Sitzungen nach einem Netzwerkausfall weitergeführt werden?

Backup / Restore-Test

- Kann das System von einer Programm- und Datensicherung wieder in Betrieb genommen werden?
- Enthält die Sicherung einen vollständigen, konsistenten Stand?

C Checkliste für den Benutzbarkeitstest

Die nachfolgende Checkliste ist aus Siedersleben (2003), S. 304 übernommen.

Verständlichkeit

- Falls Stilvorgaben verbindlich sind: Wurden diese eingehalten?
- Sind Ausgaben, Fehlermeldungen und Online-Hilfe verständlich?
- Sind Dokumentation und Online-Hilfe verständlich, verwendbar und korrekt?

Bedienbarkeit

- Lässt das System Fehlbedienungen zu („auf die Tastatur setzen…“)?
- Wird jede Benutzeraktion vom System bestätigt (erfolgreich / Fehler)?
- Sind die Links von GUI zur Online-Hilfe korrekt?
- Ist die Oberfläche für verschiedene Benutzergruppen geeignet, z.B. für Spezialist, Gelegenheitsnutzer, Datentypist?
- Ist der Aufwand für Bedienpersonal (Operator) für den Kunden akzeptabel?

Attraktivität

- Wird das System vom dümmsten anzunehmenden Benutzer (DAB) akzeptiert?
- Wird das System von einem Experten akzeptiert?

D Checkliste für den Installationstest

Die nachfolgende Checkliste ist aus Siedersleben (2003), S. 304 übernommen.

Anpassbarkeit

- Lässt sich die Software auf verschiedene Umgebungen konfigurieren?

Übertragbarkeit

- Lässt sich die Software auf allen vorgesehenen Umgebungen installieren?
- Ist sie dort lauffähig?

Koexistenz

- Verträgt sich die Software mit anderen Produkten, die auf dem gleichen Rechner installiert sind (mit welchem nicht)?

Literaturverzeichnis

Bodendorf, F. (2006): Daten- und Wissensmanagement. 2. Aufl., Berlin u. a.

Buhl, A. (2004): Grundkurs Software-Projektmanagement. München u. a.

Brodie, M.; Stonebraker, M. (1995): Migrating legacy systems – gateways, interfaces & the incremental approach. San Francisco.

Dijkstra, E. W. (1970): EWD 249 – Notes on structured programming. Technical Report, 2. Aufl., Technological University Eindhoven.

DIN 69901 (1987): Projektwirtschaft – Projektmanagement. Berlin.

Dömer, F. (1998): Migration von Informationssystemen – Erfolgsfaktoren für das Management. Wiesbaden.

Kohnke, O.; Bungard, W. (2005): SAP-Einführung mit Change Management. Wiesbaden.

Kölsch, U. (2000): Methodische Integration und Migration von Informationssystemen in objektorientierte Umgebungen. St. Augustin.

Leser, U.; Naumann, F. (2007): Informationsintegration – Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen. Heidelberg.

Mertens, P.; Bodendorf, F.; König, W.; Picot, A.; Schumann, M. (2001): Grundzüge der Wirtschaftsinformatik. 7. Aufl., Berlin u. a.

Sauter, G. (1998): Interoperabilität von Datenbanksystemen bei struktureller Heterogenität : Architektur, Beschreibungs- und Ausführungsmodell zur Unterstützung der Integration und Migration. St. Augustin.

Schanz, G. (1994): Organisationsgestaltung – Struktur und Verhalten. 2. Aufl., München.

Siedersleben, J. (2003): Softwaretechnik – Praxiswissen für Software-Ingenieure. 2. Aufl., München u. a.

Sneed, H.; Baumgartner, M.; Seidl, R. (2007): Der Systemtest – Anforderungsbasiertes Testen von Software-Systemen. München u. a.

Staehle, W. H. (1994): Management. 7. Aufl., München.

Stahlknecht, P.; Hasenkamp, U. (2006): Arbeitsbuch Wirtschaftsinformatik. 4. Aufl. Berlin u. a.

Steinbuch, P. A. (2000): Projektorganisation und Projektmanagement. 2. Aufl. Ludwigshafen.

Steiner, R. (2006): Grundkurs Relationale Datenbanken. 6. Aufl., Wiesbaden.

Steinweg, C. (2005): Management der Software-Entwicklung. 6. Aufl., Wiesbaden.

Utterback, J. M. (1971): The Process of Innovation: A Study of the Origination and Development of Ideas for New Scientific Instruments. o. O.

Versteegen, G.; Salomon, K.; Heinhold, R. (2001): Change Management bei Software-Projekten. Berlin u. a.

Versteegen, G.; Hindel, B.; Meier, E.; Vlasan, A. (2005): Prozessübergreifendes Projektmanagement. Berlin u. a.

Voigt, B.; Linke, M. (2005): Der Erfolg eines Systemhauses – Zehn Jahre Lufthansa Systems. Heidelberg.

Volkswagen AG (2007a): Kurzportrait. http://www.volkswagenag.com/vwag/vwcorp/content/de/the_group/group_profile_and_structure.html. 22.06.2007.

Volkswagen AG (2007b): Wesentliche Zahlen. http://www.volkswagenag.com/vwag/vwcorp/content/de/the_group/key-figures.html. 27.06.2007.

Volkswagen AG (2007c): Strategie. http://www.volkswagenag.com/vwag/vwcorp/content/de/the_group/group_profile_and_structure/strategy.html. 27.06.2007.

Volkswagen AG (2007d): Qualitätssicherung. http://www.vw-personal.de/www/de/arbeiten/funktionsbereiche/qualitaetssicherung.html 28.06.2007.

Volkswagen AG (2007e): Management Handbuch für Qualität, Ausgabe 3.1. Wolfsburg.

Wieczorrek, H. W.; Mertens, P. (2007): Management von IT-Projekten. 2. Aufl., Heidelberg.

Wu, B.; Lawless, D.; Bisbal, J.; Grimson, J.; Wade, V.; O'Sullivan, D.; Richardson, R. (1999): The Butterfly Methodology – A Gateway-free Approach for Migrating Legacy Information Systems. Technical Report, University of Dublin.

Zimmermann, J.; Stark, C.; Rieck, J. (2006): Projektplanung – Modelle, Methoden, Management. Berlin u. a.

Abschließende Erklärung

Ich versichere hiermit, dass ich die vorliegende Diplomarbeit selbständig, ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Sebastian Pohl

Magdeburg, den 30. November 2007