

5 Entwicklung multimedialer Anwendungen

5.1 Einführung

Bei der Entwicklung von multimedialen Anwendungen ist eine systematische Vorgehensweise unabdingbar. Von einem bestimmten Ausgangspunkt aus versucht man, ein definiertes Ziel zu erreichen. Dafür sind in der Regel verschiedene Wege möglich.

Eine genaue Analyse des Ist-Zustands und die Feststellung der Anforderungen an eine Anwendung führen zu einer realistischen Zieldefinition und zur Auswahl des Realisierungsweges.

- Voraussetzung für die erfolgreiche Entwicklung ist, wie in Kapitel 4 diskutiert, eine gute Konzeption. Folgende Punkte sollen eine Zusammenfassung geben:
- Realistische Bedarfsermittlung
Was ist das Problem? Was wollen wir erreichen? Ist eine MM-Darstellung sinnvoll?
- Sorgfältige Planung und systematische Vorgehensweise
Regiebuch, Pflichtenheft, Werkzeuge, Projektplanung
- Regelmäßige Projektbesprechungen, um den Entwicklungsstand und den Zeitrahmen ständig unter Kontrolle zu halten
- Revision
- Erfahrung und gute Ausbildung der Mitarbeiter
- Methodik, Fachwissen, Psychologie und ggf. Didaktik
Beispiel: MM-Lernprogramm bedeutet keinen programmierten Unterricht!

5.2 Entwicklungsprozess

5.2.1 Vorgehensmodelle

Dieses Kapitel ist ein Exkurs zu einigen Vorgehensmodellen aus dem Gebiet des Software Engineering.

Phasenmodell

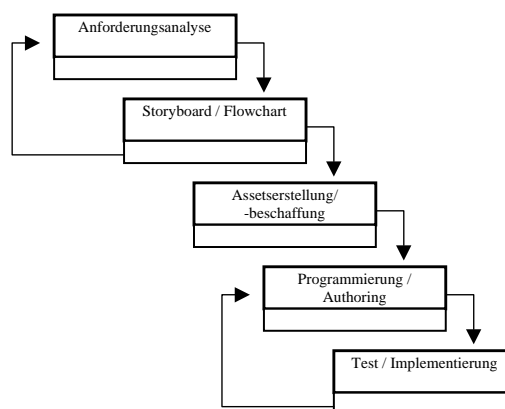
Phasen Das Projekt wird in mehrere Phasen aufgeteilt. Diese Phasen sind in einer zeitlichen und/oder logischen Reihenfolge angeordnet und müssen auch so durchlaufen werden. Die Ergebnisse einer Phase bilden die Basis für die nachfolgende. Der Vorteil ist, dass dadurch komplexe Aufgaben transparenter werden. Die einzelnen Schritte können unterschiedlich tief gegliedert sein. Beispiele für Phasen sind: Vorphase, Analyse, Design, Realisierung und Einführung.

Diese Vorgehensweise ist bei multimedialen Anwendungen sehr einengend und nicht praktikabel. Ein mögliches Vorgehensmodell für manche Aufgaben in einem multimedialen Projekt könnte aber auf das Phasenmodell aufbauen. Die einzelnen Schritten müssen jedoch feiner als im Phasenmodell unterteilt sein. Sie können parallel zueinander verlaufen wie z.B. bei der Erstellung von Assets.

Wasserfallmodell

Das Modell besteht aus Phasen, mit der Möglichkeit zur Rückkopplung nur zur unmittelbar vorangegangenen Phase. Eine Iteration, die z.B. aufgrund entdeckter Schwachstellen notwendig wird, ist nur zwischen jeweils zwei aufeinander folgenden Phasen erlaubt. Um das Ziel nicht zu verfehlen und um auf dem jeweils definierten Stand aufbauen zu können, sollte jede Phase durch einen Test abgeschlossen und dokumentiert werden.

Abb. 5-1
Wasserfallmodell für
MM-Entwicklung



Prototyping

Multimediale Anwendungen zeichnen sich durch ihre Komplexität aus, die meist schwer in Phasen erfassbar ist. Das Ziel des Prototyping besteht darin, möglichst schnell ein erstes Ergebnis zu präsentieren. Das Verfahren hat den Vorteil, dass bereits in der Planungsphase Etappen definiert, Zwischenergebnisse präsentiert und Änderungen dynamisch vorgenommen werden können (s. Abb. 5-2). Werden die Anforderungen erfüllt, wird das System eingeführt, werden Sie nicht erfüllt, wird entweder das Konzept und die Realisierung oder nur die Realisierung überarbeitet.

Das Prototyping ist ein sinnvolles Verfahren bei der Multimedia-Entwicklung, das für folgende Tätigkeiten eingesetzt wird:

- Überprüfung der Machbarkeit
- Demonstration für den Kunden
- Erläuterung von Designaspekten

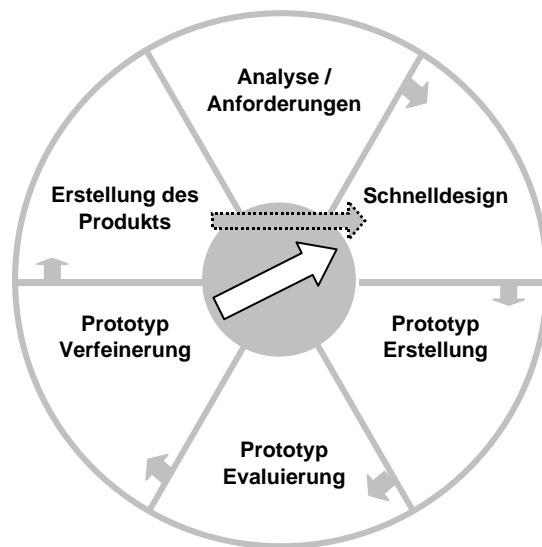


Abb. 5-2
Das Prototyping

Spiralmodell

Das Spiralmodell ist durch ein phasenweises Vorgehen, eine feste Folge von wiederkehrenden Aktivitäten in jeder Phase und einer Miteinbindung des Prototyping-Modells und der Alternativenbewertung wie z.B. der Risikoanalyse in den jeweiligen Prototypen gekennzeichnet.

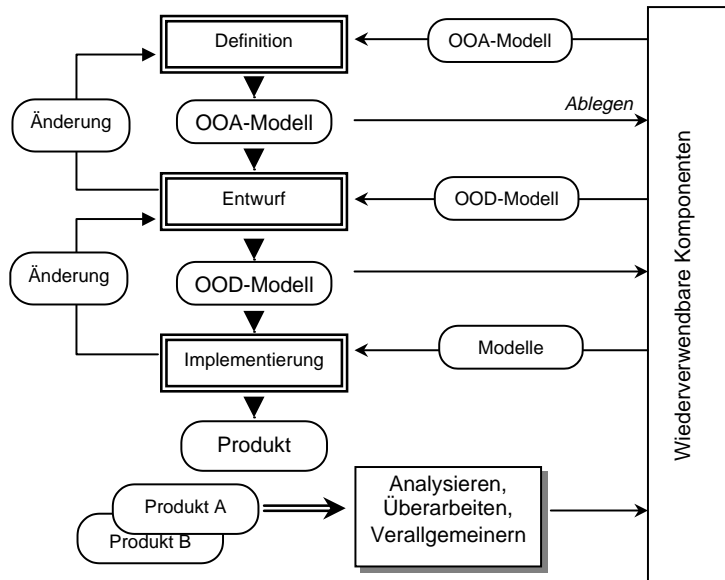


Abb. 5-4
Das objektorientierte
Modell

5.2.2 Autorensystem

Programmiersprachen sind universell einsetzbar zum Lösen verschiedener Probleme. Sie sind nicht speziell für die Entwicklung multimedialer Anwendungen konzipiert. Entscheidend bei der Entwicklung eines multimedialen Systems ist das Kosten-Nutzen-Verhältnis. In den meisten Fällen sind die Entwicklungskosten und die Folgekosten beim Einsatz von Programmiersprachen viel höher als beim Einsatz von Spezialwerkzeugen.

Auf dem Markt gibt es viele hervorragende Programmiersprachen, die auch für die Entwicklung von multimedialen Anwendungen geeignet wären, die aber aus Kosten-Nutzen-Gründen oft ausscheiden. Beispiele hierfür sind die visuellen Programmiersprachen sowie die Programmiersprache Java.

Autorensysteme sind Werkzeuge zur Erstellung multimedialer Anwendungen. Sie haben eigene Philosophien, die für das jeweils entsprechende System charakteristisch sind. Entscheidend bei der Wahl eines Systems ist die Plattform, auf der die Anwendung entwickelt bzw. angewendet werden kann. Tabelle 5-1 zeigt einige Beispiele für verschiedene Kategorien von Autorensystemen. Auf den Markt kommen ständig neue Produkte.

Programmiersprachen

Werkzeuge

	Kategorie/Philosophie			
	Kartenorientiert	Buchorientiert	Zeitorientiert	Strukturorientiert
Grundprinzip	Karteikasten	Metapher eines Buches	Theater bzw. Filmproduktion	Ablaufplan (Flussdiagramme)
Darstellung	Bildschirm entspricht einer Karteikarte, die Objekte enthält. Die Navigation ermöglicht den Sprung in andere Karten.	Bildschirm entspricht der Seite eines Buches (s. Karteikasten). Das System ist ereignisorientiert und enthält eine mächtige Programmiersprache. Objekte haben eine feste Hierarchie.	Bildschirm repräsentiert eine Bühne, auf der Darsteller (Objekte) mit entsprechenden Rollen agieren. Das System ist regieorientiert (Regieplan, in dem »fast« alle Vorgänge zeitorientiert gesteuert werden).	Die Entwicklungsarbeit und Steuerung der Aktivitäten auf dem Bildschirm erfolgt mit Flussdiagrammen. Für den Entwickler besonders große Übersichtlichkeit
Beispiele	HyperCard und SuperCard	ToolBook von Asymetrix	Macromedia Director	AuthorWare
Plattform	Mac	Windows, Internet	Windows, Mac, Internet	Windows, Mac

Tab. 5-1
Kategorien von
Autorensystemen

5.2.3 Wirtschaftlichkeit und Einsatzgebiete

Bei der Entwicklung von multimedialen Anwendungen gelten die gleichen Ansätze zum wirtschaftlichen Arbeiten wie für jedes IT-Projekt. Da es sich bei Multimediaprojekten meist um viele Medien handelt, die evtl. von Dritten zu beziehen sind, müssen zusätzlich die rechtlichen Aspekte und evtl. anfallende Lizenzgebühren berücksichtigt werden.

Die Wirtschaftlichkeit trägt zu Gewinn bei, ohne dies zwangsläufig zu erzwingen. Ein Gewinn kann erzielt werden, wenn der Preis über den Kosten liegt, die hoffentlich wirtschaftlich gestaltet werden. Um den Gewinn zu errechnen, müssen alle Kosten systematisch festgehalten werden (s. Tab. 5-2). Der Deckungsbeitrag ist ein Teilgewinn, der zwischen variablen Kosten (mengenabhängig) und Fixkosten unterscheidet. Die variablen Kosten und Aufwendungen werden dann vom erzielten Umsatz abgezogen, um die Höhe des Deckungsbeitrags zu ermitteln. Ausgehend von einem positiven zu erzielenden Deckungsbeitrag kann der Verkaufspreis kalkuliert werden. Stückzahl und Preis spielen eine wichtige Rolle, um Umsatz zu machen. Große Stückzahlen können z.B. dazu führen, dass der Preis niedrig gehalten werden kann bzw. um mehr Gewinn zu erwirtschaften. Da evtl. nicht alle Exemplare verkauft werden können, sollen die zu erwartenden Erlöse um die Restexemplare (Abgabe zu Schleuderpreisen) gemindert werden.

Sowohl der Aspekt der Wiederverwendung (s. Kapitel 4.5.6) als auch die Wahl des für die Anwendung geeigneten Werkzeugs tragen zum wirtschaftlichen Entwicklungsprozess einer Anwendung bei.

Allgemein	- Beratung - Vorbereitung - Projektabwicklung - Fixkosten - juristische Beratung
Entwicklung / Erstellung	- Programmierung - Authoring - Testen u.Ä. - Produktion * - Verpackung *
Assets und Lizenzen	- Erstellungskosten - Recherche - Lizenzgebühren * - etc.
Marketing	- Werbeaufwand - Rabatte * - Schenkungen *
* in der Regel abhängig von der Stückzahl (erstellte bzw. verkaufte Exemplare)	

Tab. 5-2
Kosten eines Projektes

Der Einsatz von Werkzeugen zur Erstellung multimedialer Anwendungen ist bei Weitem wirtschaftlicher (jedoch nicht unbedingt effizienter) als die Verwendung visueller Programmiersprachen. Dies trifft zumindest auf folgende Aspekte zu:

- Kürzere Einarbeitungszeit für neue Mitarbeiter zum Erlernen von professionellen Techniken (auch beim Einsatz von multimedialen Komponenten). Professionalität bei der Entwicklung von professionellen multimedialen Anwendungen.
- Kürzere Entwicklungszeiten durch das Vorhandensein fertiger Komponenten gegenüber der Verwendung herkömmlicher Programmiersprachen.
- Einfache und schnelle Änderung, Wartung und Updates

Es stellt sich nicht die Frage, welches Werkzeug oder Entwicklungssystem das beste ist. Diese Frage wäre laienhaft!

Die Entscheidung für die Entwicklungshilfen hängt von vielen Faktoren ab. Unter anderem sind folgende zu berücksichtigen:

- Wofür braucht man die Anwendung?
- Welche Rahmenbedingungen gibt es?
- Welchen Nutzen hat das System?

- Wie ist die Wirtschaftlichkeit?
- Welche Fachleute werden benötigt?

Eine Multimedia-Anwendung ist nicht nur eine Frage des Gewinns, sondern auch des Nutzens. Eine Kosten-Nutzen-Analyse ist jedoch nicht immer einfach. Viele Aspekte des Nutzens, wie z.B. die Imageförderung, sind nicht direkt messbar. Kosten sind leichter zu ermitteln: Herstellkosten, Marketing, Kosten für Service und nicht bezahlte Dienstleistungen, Fixkosten u.Ä.

Nutzen können PR, Image, Motivationssteigerung, Verkaufsförderung, Spareffekte etc. sein. Bei Lernsystemen können u.a. Kosten für Kurse, der Wegfall von Fahrtzeiten, Übernachtungskosten, Mitarbeiterabwesenheit (Kursbesuch) erheblich zur Kostenreduzierung beitragen. Eine orts- und zeitunabhängige Ausbildung kann so angeboten werden. Gute Ausbildung von Mitarbeitern führt z.B. zu weniger Ausschussware, schnellerem Arbeiten und einer Verbesserung der Qualität.

5.2.4 Zeitschema

Der Entwicklungsprozess kann wie bei vielen Softwareprojekten nach dem Phasenmodell oder anderen Konzepten erfolgen. Dennoch muss man feststellen, dass die Entwicklung multimedialer Anwendungen einige Besonderheiten aufweist, die beachtet werden müssen.

Die Erstellung eines multimedialen Systems kann mehrere Dimensionen aufweisen, die das Einsetzen von verschiedenen Vorgehensmodellen möglich bzw. notwendig machen. Es kann beispielsweise zwischen der informationstechnologischen und inhaltlichen Bearbeitung getrennt werden. So kann ein Vorgehensmodell mit verschiedenen Pfaden entstehen, die zum fertigen System führen. Die Entwicklung multimedialer Anwendungen ist ein iterativer Prozess, der dem Prototyping (Spiralmodell) am nächsten kommt. Eine wichtige Etappe bei Multimediaprojekten ist die Evaluierung der Prototypen bei iterativen Verfahren. Diese muss/soll mit den Benutzern des künftigen Systems erfolgen.

*Orientierungsphase und
Grobkonzept*

Eine Orientierungsphase ist für jedes Multimediaprojekt wichtig. Diese Phase kann bis zum Erstellen des ersten Prototyps dauern. Folgende Schritte können den Weg für die Entwicklung ebnen:

- Erstellung des Pflichtenhefts
- Erstellen von Flowcharts für die grobe Struktur der Anwendung
- Skizzieren von Handlungen, Layouts und Screen-Designs

- ❑ Überlegung zum Einsatz von Medien für den ersten Prototyp sowie für das Endprodukt
- ❑ Die Erstellung des Prototyps, der als ein lebendiges Konzept dienen kann. Er soll einige Schlüsselszenen enthalten. Der Prototyp soll funktionell und gestalterisch einwandfrei sein. Lieber wenige Inhalte und Details gut realisieren als viele und schlecht.

Die Vorstellung des Prototyps beim Kunden und die Abstimmung kann nun erfolgen. Der Prototyp kostet den Entwickler zwar Zeit und Aufwand, bringt ihm aber auf der anderen Seite mehr Sicherheit. Der Prototyp ist wie ein lebendiges Konzept, das bereits am Anfang der Entwicklung dem Entwickler und dem Kunden eine gemeinsame Vorstellung vom Endprodukt vermitteln kann.

Abstimmung

Erst danach soll mit dem Feinkonzept begonnen werden. Jetzt kann das Storyboard detailliert, Assetslisten erstellt und der genaue Bedarf ermittelt werden. Im Feinkonzept ist u.a. Folgendes enthalten:

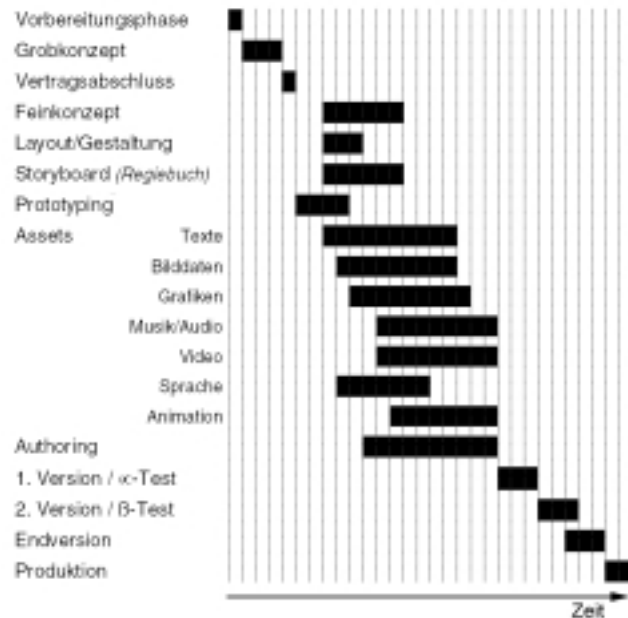
Feinkonzept

- ❑ Navigationsstruktur und Ablauf
- ❑ Detaillierung des Storyboard
- ❑ Beschaffung der Medien und Erstellung von Assetslisten:
Mit Hilfe der oft erwähnten Mediendatenbank kann festgestellt werden, welche wiederverwendbaren Komponenten in dem neuen Projekt eingesetzt werden können.

Abbildung 5–5 zeigt ein Zeitschema für die Entwicklung multimediale Anwendungen in den Phasen vom Feinkonzept bis zur Produktfertigung (Produktion). Folgende Schritte sollen den Entwicklungsprozess skizzieren:

- ❑ Analyse und Konzeption
- ❑ 1. Prototyp
- ❑ Storyboard
- ❑ Assetproduktion
- ❑ Medienintegration / Authoring
- ❑ Optimierung, Test
- ❑ Mastering
- ❑ Produktion, Replikation und Vertrieb

Abb. 5-5
Zeitschema zur
Erstellung von
multimedialen
Anwendungen



5.2.5 Mastering

Mastering ist die Vorstufe der Vervielfältigung (CD-Pressen) und des Vertriebs. Die Erstellung einer Master-CD (Premaster) erfolgt nach dem Abschluss der Entwicklungsarbeit und soll vom Entwickler vorgenommen werden. Das Mastering ist von große Bedeutung, um Aussagen über Installationserfolg, Lauffähigkeit, Performance und Cross-Plattform-Tauglichkeit zu machen.

Premaster

Eine CD-ROM hat eine Kapazität von ca. 660 MB. Für das Premastering kommen verschiedene Dateisysteme in Frage. Im Folgenden werden einige erwähnt, die für die Multimedia-Produktion von Bedeutung sind:

ISO 9660

Dieses Dateiformat ist geeignet für MS-DOS, Win3x, Win95/98/NT, MacOS und Atari.

Es gelten strikte Benennungsregeln: Ein Dateiname darf maximal 8 alphanumerische Zeichen (Großschreibung) und eine Erweiterung von 3 Zeichen haben. Das einzige erlaubte Sonderzeichen ist der Unterstrichstrich »_«. Alle Dateinamen (Filme, Castlib, Verzeichnisse, externe Medien etc.) müssen entsprechend benannt sein. Deshalb soll

eine Entscheidung zur Verwendung dieses Formats frühzeitig getroffen werden, um Dateien von Anfang an entsprechend zu benennen.

ISO 9660 Level 2

Dieses Format kann für Win95/98 und MacOS verwendet werden. Für die Benennung von Dateien und Verzeichnissen kann man bis zu 30 Zeichen benutzen. Das Format wird von Win3x nicht unterstützt.

Mac HFS (Hierarchical File System)

Dieses Format ist optimal für die Verwendung mit MacOS. HFS ist das Dateisystem, das Apple-Macintosh-Computer verwenden.

Das Mac-ISO-Hybridformat hilft bei CD-ROMs, die sowohl auf Windows als auch unter MacOS laufen sollen. Es stellt sicher, dass unter dem jeweiligen Betriebssystem nur der ISO 9660- bzw. HFS-Teil sichtbar ist.

Hybridformat

Sinnvoll und wirtschaftlich ist es allemal, Checklisten für das Pre-mastering zu erstellen und durchzugehen, bevor die Presswerke die Pre-master-CD bekommen. Folgende Punkte müssen beachtet werden:

- Durchführen des Funktionstests auf allen Plattformen. Dies beinhaltet das Testen von externen Aufrufen, Autostartfunktionen, Vorhandensein aller Dateien und Verzeichnisse etc. Der Test soll nie auf dem Entwicklungsrechner durchgeführt werden. Einige Komponenten könnten nämlich aus den auf der Festplatte gespeicherten Daten und nicht von CD-ROM eingelesen werden.
- Plattformspezifische Tests definieren und durchführen.
- Virentest der fertigen CD unter Windows und MacOS mit aktuellen Programmen durchführen.
- CD-R beschriften und mit Produktionsnamen, Version, Datum und Name des Erstellers versehen.

5.3 Verwendung von Datenbanken

Die Medien einer multimedialen Anwendung werden in der Regel in externen Dateien gespeichert. Der Zugriff auf diese Daten innerhalb einer Anwendung erfolgt dann über Dateiverzeichnisse und Dateien, die durch das Dateisystem eines Betriebssystems verwaltet werden.

Die Speicherung und Verwaltung solcher Objekte ist meist plattformspezifisch, was Probleme mit sich bringen kann. Dazu kommt,

dass die Komponenten in unterschiedlichen Formaten vorliegen, die ihrerseits teilweise plattformabhängig sind. Die Entwicklung und das Einsetzen von Multimedia-Datenbanksystemen kann solche Probleme beseitigen. Ob und wie schnell sich solche standardisierte Datenbanksysteme durchsetzen, soll nicht Thema dieses Buchs sein.

Bereits der Einsatz von Datenbanken für die Speicherung von Textinhalten kann den Entwicklungsprozess beschleunigen und flexibilisieren. Beispiele hierfür sind:

- Speicherung von Texten verschiedener Sprachen
- Speicherung von Daten zur Parametrisierung der Anwendung, von Benutzerprofilen und anderen wichtigen Informationen wie z.B. Steuersätzen
- Zugriff auf aktuelle Daten für Präsentationen u.Ä.

Tabellen

Eine (relationale) Datenbank setzt sich aus einer oder mehreren Tabellen (*Table*) zusammen, die in Beziehung zueinander stehen. Eine Tabelle ist aus Datensätzen (*Records*) aufgebaut, die ihrerseits aus Feldern (*Fields*) bestehen. In diesen Feldern werden die Daten gespeichert. Für das schnelle Wiederfinden der Daten können bestimmte Felder als Schlüsselfelder definiert werden. Ein Primärschlüssel enthält einen eindeutigen Eintrag, um den Datensatz einer Person oder Firma zu identifizieren. Eine Adress-Datenbank kann aus Hunderttausenden von Datensätzen bestehen. Ein Datensatz hat Datenfelder wie Name, Straße, PLZ, Ort etc. Die Adresse von X wird in einem Datensatz (Zeile) gespeichert. Für das Auffinden einer bestimmten Adresse (z.B. Firma X) kann das künstliche Merkmal Firmennummer als Primärschlüssel verwendet werden. Mit Hilfe von Indizes (Index) können wichtige Felder indiziert werden, so dass die Suche nach Datensätzen, die in diesen Feldern ein bestimmtes Merkmal enthalten, schnell erfolgen kann.

Verknüpfung

Die Verknüpfung einer Tabelle mit anderen Tabellen kann z.B. über den Primärschlüssel erfolgen. So können alle Bestelldaten mit der Kundennummer in einer extra Tabelle gespeichert werden. Ein Kunde hat viele Bestellpositionen, die über seine Kundennummer auffindbar sind. Er hat eine Adresse, die auf die gleiche Art und Weise aus einer anderen Tabelle geholt werden kann.

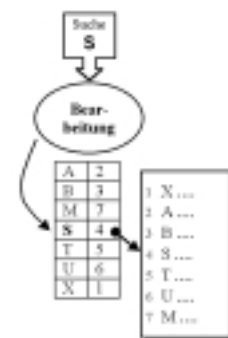
Das Verwenden mehrerer Tabellen hilft, eine sonst unvermeidbare Redundanz zu umgehen. So ist die Speicherung der Adresse bei jeder Bestellposition nicht erforderlich, da jederzeit über die dort gespeicherte Kundennummer die Adresse zu erhalten ist.

Die wichtigsten Operationen für die Bearbeitung von Daten in einer Datenbank sind: Hinzufügen, Löschen, Suchen und Ändern. Für multimediale Anwendungen ist es wichtig festzustellen, welche Datenbankfunktionen für welchen Benutzer (Anwender, Administrator etc.) zur Verfügung zu stehen. Beispiele für unterstützte Datenbankfunktionen sind:

- Nur lesen: Die Informationen werden aus einer Datenbank eingelesen und präsentiert. Der Nutzer kann/darf die Daten selbst nicht verändern. Die Informationen können in diesem Fall auch auf einer CD-ROM o.Ä. gespeichert sein.
- Lesen und Schreiben: Der Nutzer kann Informationen aus einer Datenbank (nur lesend) abrufen. Gleichzeitig kann er bzw. das System weitere Informationen speichern, wie Informationen über Erfolgskontrolle, eigene Notizen u.Ä. Extra Tabellen müssen für das Schreiben auf der Festplatte angelegt werden, um die Daten speichern zu können. Die zu lesenden Daten können auf der Festplatte oder auf einer CD-ROM abgelegt sein.
- Update: Der Nutzer (bzw. Administrator) kann die gespeicherten Informationen verändern und zurückspeichern. Die Datenbanktabellen müssen auf einem überschreibbaren Medium (z.B. Festplatte) gespeichert sein. Auch eine Kombination aus CD-ROM und Festplatte ist denkbar, was allerdings nur mit größerem Aufwand zu realisieren ist. Die Originaldaten können auf der CD-ROM gespeichert sein, wobei die geänderten Daten auf die Festplatte gespeichert werden. Das System muss erst die Tabellen der Festplatte, danach die der CD-ROM durchsuchen, um evtl. die aktuellsten Daten zu präsentieren.

In den meisten Fällen werden Erweiterungen von Drittanbietern benötigt, um aus einem Autorensystem über normierte Schnittstellen auf Datenbanken zuzugreifen.

Für kleine Anwendungen wird oft mit Hilfe von Dateien und Datenfeldern (Array) eine vereinfachte, mit mehr Programmieraufwand verbundene Datenverwaltung realisiert. Die Abbildung (links) zeigt eine Skizze zur Verdeutlichung solcher Lösungen. Eine Datendatei enthält eine beliebige Anzahl von Zeilen. Die Zeilen sind systemintern nummeriert (z.B. relativ zum Dateianfang), so dass ein Sprung zu einem Datensatz über die Zeilennummer möglich ist. Die Schlüsselfelder (z.B. Kundennummer) werden redundant in Datenfeldern (Liste) gespeichert. Der Inhalt des Datenfeldes »Kundennummer« ist immer in sortierter Form vorhanden. Das Datenfeld enthält zusätzlich zur



Kundennummer die Satznummer, die die Position des Datensatzes in der Datei angibt. Die Datensätze sind unabhängig von ihrer Position in der Datei (welche Zeile) lokalisierbar. Die Suche in der sortierten Liste ist auch bei zigtausend Einträgen relativ schnell im Vergleich zu einer sequentiellen Suche in der Datei selbst (langwierig). Bei jedem neuen Datensatz muss die sortierte Liste aktualisiert werden (Einfügen und neu sortieren). Die meisten Autorensysteme stellen Funktionen zur Bearbeitung, zum Suchen und Sortieren von Listen zur Verfügung. Um bei großen Listen effizienter zu suchen, kann man Suchalgorithmen selbst implementieren. Bei einer einfachen Binärsuche kann ein Eintrag bei 20.000 Sätzen nach 13 Zugriffen auf die Liste gefunden werden bzw. als nicht vorhanden festgestellt werden. Auch die Liste (Indizes) selbst kann in einer Datei gespeichert werden. So muss die Liste nicht immer bei Programmstart neu erstellt werden. Dadurch wird das Starten der Anwendung beschleunigt.

Selbstverständlich muss überlegt werden, ob es nicht günstiger ist, eine Erweiterung zu kaufen, die die Benutzung von Standard-Datenbanken ermöglicht.

Was man auf keinem Fall tun sollte ist, die Daten in der Anwendung selbst (Textfelder o.Ä.) zu speichern. Dies gilt besonders, wenn es sich um Texte handelt, die sich ändern können. Das Ändern und Aktualisieren kann die Folgekosten in die Höhe treiben.

5.4 Multimedia-Entwicklung mit Macromedia Director

5.4.1 Director-Grundkonzept

Dieses Kapitel vermittelt eine Übersicht über das Autorensystem Director. Das Kapitel ist kein Handbuch zur schrittweisen Einführung und zum Erlernen des Werkzeugs, sondern eine schnelle Einführung in Grundkonzept, Philosophie, Möglichkeiten und die Arbeitsweise von Director. Dies soll unabhängig von der aktuellen Version geschehen, solange es sich nicht um Funktionalitäten handelt, die in älteren Versionen fehlen.

Der Film

Director basiert auf dem Filmprinzip. Ein Film beschreibt, welcher Darsteller in welcher Rolle (als Sprite) an welcher Stelle wann und mit welchen Eigenschaften auf der Bühne erscheint.

Director stellt Werkzeuge und Fenster für die Verwaltung und Bearbeitung von Darstellern (Komponenten und Objekte) sowie für das Zusammenwirken dieser Komponenten in einem Zeitraster zur Verfügung.

Director-Filme sind interaktive Multimedia-Anwendungen, die Animationen, Sound, Text, Digitalvideo und andere Medientypen enthalten können. Ein Film kann so klein und einfach wie ein animiertes Logo oder so komplex wie ein Online-Chatroom oder ein Spiel sein. Ein Director-Film ist eine Datei, die alle Medien steuert, die während des Abspielens des Films erscheinen. Ein Film kann mit externen Media verknüpft sein oder zu einer Reihe von Filmen gehören, die aufeinander verweisen.

Dank der Interaktivität von Director kann der Film auf Ereignisse reagieren und sich auf vorgegebene Weise ändern. Director dividiert den zeitlichen Ablauf in eine Reihe von Bildern, die den Einzelbildern eines Zelluloidfilms ähneln.

**Abb. 5-6**

Übersicht Director

Die Bühne

Die Bühne (*stage*) ist der sichtbare Teil eines Films (Bildschirmbereich), auf dem die Anwendung abläuft. Jeder Film hat mehrere charakteristische Eigenschaften (Eigenschaften der Bühne) wie Größe, Position und Farbe der Bühne. Die wichtigsten Eigenschaften können im Dialogfeld »Filmeigenschaften« angegeben werden.

Eigenschaften der Bühne

Die Filmeigenschaften (*movie properties*) können

- über Menü (Modifizieren -> Film -> Eigenschaften) bzw.
- rechte Maustaste eingestellt werden.

Mit der Bühne legt man fest, wo Mediaelemente erscheinen.

Abb. 5-7
Die Bühne



Kobold = Sprite

Mit Hilfe von Darstellern (Objekte wie Grafiken, Bilder u.Ä.) können die Rollen (Aktivitäten) auf der Bühne übernommen werden. Darsteller befinden sich nie selbst auf der Bühne, sondern nur ihre »Kobolde« (*Sprites*). Ein Darsteller kann somit mehrere Auftritte auf der Bühne gleichzeitig übernehmen (mehrere Kobolde).

Die Bühne in Abbildung 5-7 zeigt 5 Objekte (*Sprites*): Weinglas, Weinflasche, Käse, Brot und Teller (mit Schatten). Das Bild beschreibt einen Zeitpunkt auf der Bühne, in dem diese Objekte sichtbar sind.

Die Besetzung

Director organisiert Darsteller in Bibliotheken, die Besetzungen genannt werden. Eine Besetzung (*cast*) dient der Speicherung aller Elemente (Darsteller = *cast members*) eines Films.

Darsteller

Darsteller können Bitmap-Grafiken, Vektorformen, Sounds, Flash-Filme, Digitalvideos u.Ä. sein.

Abbildung 5-8 zeigt u.a. die Darsteller, deren Sprites wir bereits auf der Bühne begegnet sind. Darsteller können Namen haben wie z.B. das Textfeld »zeit«. Jeder Darsteller ist eindeutig durch seine Referenznummer identifizierbar (z.B. Darsteller 1: Wein(-Glas); Darsteller 7: Übergangseffekt). Bedingt durch das Entwicklungskonzept war es sinnvoll, sowohl dem Glas als auch der Weinflasche denselben Namen zu geben (Wein). Die Sprites beider Darsteller sollen als Navigationselemente zum Verzweigen in das Kapitel »Wein« dienen. Dabei soll ein Lingo-Skript für alle Navigationselemente verwendet werden. In der Regel empfehlen wir, Darstellern eindeutige Namen zu geben.

**Abb. 5-8**

Das Besetzungsfenster

Man kann zwischen zwei Kategorien von Besetzungen unterscheiden: der internen (*internal*) und der externen (*external*). Interne Besetzungen werden in der Anwendung gespeichert. Externe Besetzungen werden in Dateien außerhalb der Anwendung gespeichert. Die Anwendung enthält Verweise auf diese Daten. Darsteller können auf verschiedene Art und Weise entstehen wie:

- ❑ Durch Drag and Drop aus einer anderen Anwendung.
- ❑ Alle im Director erstellten Objekte wie Grafiken, Vektorformen, Texte, Felder, Skripte etc. werden automatisch in die Besetzung aufgenommen.
- ❑ Durch Importieren vorhandener Medien (Dateien und Ressourcen).

Ein Darsteller dient als Vorlage für einen oder mehrere Koblode (*Sprites*). Sprites sind Platzhalter auf der Bühne für einen Darsteller. Sie sind Objekte, die steuern, wann, wo und wie ein Medium in einem Film erscheint. Sie agieren zu einer bestimmten Zeit (wird bestimmt über den Koboldkanal im Drehbuch) und an einem bestimmten Ort (Position) auf der Bühne. Eigenschaften von Kobolden können verändert werden (Farbe, Größe, Ausrichtung, Form etc.). Das Ändern der Eigenschaften (Erscheinungsbild der Sprites) erfolgt ohne Auswirkungen auf die Darsteller. So kann ein Sprite gedreht, geneigt und gekippt werden. Die Sprite-Eigenschaften können mit dem Sprite-Inspektor, im Drehbuch oder über Lingo-Befehle geändert werden.

Sprite

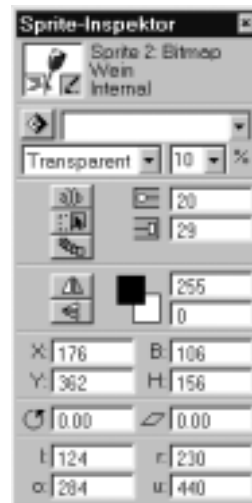
Der Sprite-Inspektor

Mit dem Sprite-Inspektor können Sprite-Eigenschaften angezeigt und bearbeitet werden. Darüber hinaus zeigt das Sprite-Overlay die gängigsten Eigenschaften für ausgewählte Sprites direkt auf der Bühne

Sprite-Inspektor

an. Einige wichtige Eigenschaften werden mit Hilfe des Sprites »Wein« (Glas) nachstehend beschrieben (s. Abb. 5–9):

Abb. 5–9
Sprite-Inspektor



- Verhalten* Unter Verhalten versteht man bereits geschriebene Skripte in Lingo-Anweisungen. Durch Anbringen an Sprites kann Lingo-Interaktivität hinzugefügt werden, ohne Lingo-Skripte schreiben zu müssen.
Der Sprite »Wein« hat kein zugewiesenes Verhalten.

- Farbefecktmodus* Farbefecktmodus zeigt an, welche Art Farbefeckt auf das Sprite angewendet wurde.

Im Folgenden werden einige Farbefeckte beschrieben:

- Kopieren** zeigt alle Originalfarben in einem Sprite an. Alle Farben einschließlich Weiß sind undurchsichtig, wenn die Grafik keinen Alphakanal-Effekt (Transparenz) enthält. »Kopieren« eignet sich als Hintergrund oder für Sprites, die nicht vor anderen Grafiken erscheinen. Sprites mit dem Farbefeckt »Kopieren« werden schneller animiert als Sprites mit anderen Farbefeckten.
- Matt** entfernt das weiße Begrenzungsrechteck aus dem Sprite. Alle innerhalb der Begrenzung liegenden Grafiken sind undurchsichtig. Der Farbefeckt »Matt« verbraucht genau wie die Maskenfunktion mehr Arbeitsspeicher als andere Farbefeckte, weshalb mit »Matt« bearbeitete Sprites langsamer animiert werden als andere Sprites.

- ❑ **Hintergrund transparent** macht alle Pixel in der Hintergrundfarbe des ausgewählten Sprites durchsichtig, so dass der Hintergrund sichtbar wird.
- ❑ **Transparent** macht alle hellen Farben durchsichtig, so dass hellere Objekte, die unter dem Sprite liegen, sichtbar werden.
- ❑ **Stanzen** kehrt genau wie der Farbeffekt »Umkehren« überlappende Farben ins Gegenteil um, nichtüberlappende Farben sind jedoch transparent. Das Sprite ist nur sichtbar, wenn es ein anderes Sprite überschneidet.
- ❑ **Abdunkelung/Aufhellung** vergleicht die RGB-Farben der Vordergrund- und Hintergrundpixel und verwendet die dunkelste/hellste Pixelfarbe.

Das Sprite »Wein« hat den Farbeffekt »transparent«.

- ❑ Um ein Sprite ein- oder auszublenden, kann man die Mischungseinstellung (0 bis 100) im Sprite-Inspektor eingeben. Dabei bewirkt 0 das Einblenden und 100 das Ausblenden eines Sprites. *Mischung*
- ❑ Falls aktiviert, kann das ausgewählte Text-Sprite während des Abspielens auf der Bühne weiterhin bearbeitet werden. Beim Sprite »Wein« nicht aktiviert. *Bearbeitbar*
- ❑ Falls aktiviert, kann das ausgewählte Sprite während des Abspielens auf der Bühne gezogen werden. Beim Sprite »Wein« nicht aktiviert. *Bewegbar*
- ❑ Falls aktiviert, bleibt das ausgewählte Sprite auf dem Bildschirm und hinterlässt eine Spur aus Grafiken auf seinem Pfad, wenn der Film abgespielt wird. Wenn »Spuren« nicht aktiviert ist, wird das ausgewählte Sprite aus den vorherigen Bildern gelöscht, wenn der Film abgespielt wird. Beim Sprite »Wein« nicht aktiviert. *Spuren*
- ❑ Start und Ende zeigen die Nummern des Anfangs- und Endbilds des Sprites an. Das Sprite »Wein« erscheint auf der Bühne während die Frames 20 bis 29 gezeigt werden. *Start- & Endframe*
- ❑ Mit horizontal und vertikal Kippen wird das Sprite horizontal oder vertikal gespiegelt, um eine umgekehrte Grafik zu erzeugen. *Kippen*
- ❑ Die Farbfelder »Vordergrundfarbe« und »Hintergrundfarbe« legen die Farben des ausgewählten Sprites fest. *Farben*

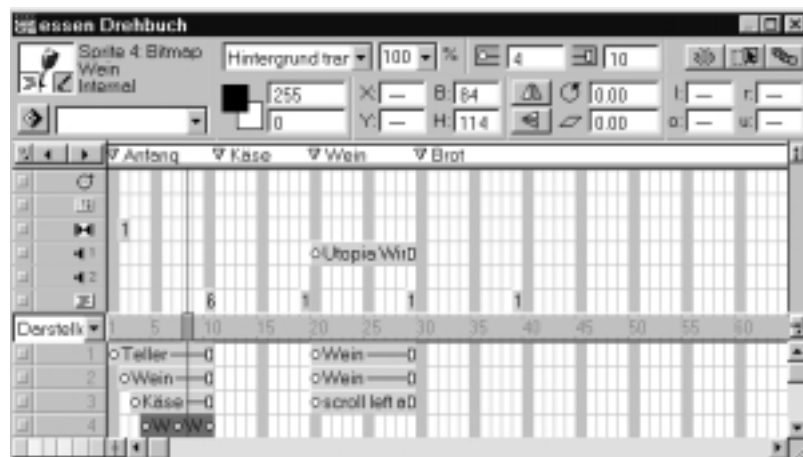
- Position* Horizontaler (X) und vertikaler (Y) Registrierungspunkt zeigen die Position des Registrierungskreuzes in Pixel von der oberen linken Ecke der Bühne aus gemessen an.
Beim Sprite »Wein« 176, 362.
- Breite & Höhe* Breite und Höhe zeigen die Größe des Begrenzungsrechtecks des Sprites in Pixel an.
Beim Sprite »Wein« 106 (B) und 156 (H).
- Drehungswinkel* Der Drehungswinkel dreht das Sprite um die eingegebene Gradzahl.
- Neigungswinkel* Der Neigungswinkel neigt das Sprite um die eingegebene Gradzahl.
- Koordinaten* Die Sprite-Koordinaten geben die exakte Position eines ausgewählten Sprites an.

Das Drehbuch

Das Drehbuch (*Score*) koordiniert die Medien des Films und legt fest, wann Objekte (z.B. Grafiken, Texte) erscheinen und Sounds abgespielt werden. Spezielle Kanäle steuern das Tempo, den Sound und die Farbpaletten des Films. Das Drehbuch ordnet außerdem Skripte (Lingo-Anweisungen) zu, die angeben, wie der Film auf bestimmte Ereignisse oder Eingaben reagiert.

Das Drehbuch ist ein wichtiges Fenster, das es ermöglicht, durch die Zeit zu navigieren und den zeitlichen Ablauf des Films festzulegen. Im Drehbuch wird die Position jedes Elements verfolgt.

Abb. 5-10
Das Drehbuchfenster



Die zeitlichen Abläufe aller Filmsequenzen werden somit gesteuert. Im Folgenden sind einige wichtige Komponenten des Drehbuchfensters aufgeführt:

- Raster aus Zellen (*Cells*). Zellen sind die kleinsten Einheiten im Drehbuch. Sie enthalten Informationen über Darsteller.
- Eine Spalte von Zellen wird als Bild (*Frame*) bezeichnet. Es ist eine Momentaufnahme des Films.
- Eine Zeile von Zellen bezeichnet man als Kanal (*Channel*). Kanäle *Kanäle* nehmen bestimmte Typen von Informationen auf:
 - Drehbuchmarkierung (*Marker Channel*)
 Markierungen können entlang der Zeitachse des Drehbuchfensters festgelegt werden. Eine gesetzte Marke kann mit einem Text versehen werden (Benennung). Markierungen dienen der Orientierung bei der Konstruktion eines Films sowie als Ansprungsstelle von einer anderen Stelle aus.
 - Effektkanäle:
 - Tempo
 Einige Steuerungsmöglichkeiten können eingestellt werden, wie z.B. Anzahl Frames pro Sekunde (tempo in *fps*), Wartezeit in Sekunden (*Wait*), Warten bis eine Taste (Tastatur/Maus) gedrückt wird, Warten bis ein Video oder eine Audiosequenz (*Queue Point*) beendet ist. Dies kann sich auch auf einen Soundkanal (*Sound Channel*) beziehen. Eine Animation kann somit ohne Programmierung mit einer Sprachausgabe/Musik synchronisiert werden. Die Einstellung der fps kann helfen, in einem bestimmten Bilderbereich die Geschwindigkeit einer Animation zu regeln (zwischen zwei definierten Tempo-Kanal-Einträgen). *Synchronisation*
 - Palettenkanal
 In diesem Kanal wird die Farbpalette, die für das Bild (Spalte von Zeilen) aktiv ist, platziert. Falls keine Farbpalette definiert ist, wird in der Regel die Systempalette verwendet.
 - Übergang (*Transition*)
 Verschiedene Kategorien von Übergangseffekten können definiert werden. Der Übergangseffekt kann sich auf die gesamte Bühne oder nur auf die veränderten Objekte beziehen. Auch Dauer und Art (fein/grob) können gesteuert werden. Ein Übergang findet immer zwischen dem Ende des

aktuellen Bildes und dem Anfang des Bildes statt, für das der Übergang eingestellt wurde.

- Soundkanäle

Es gibt zwei Soundkanäle für die Ausgabe von Musik, Sprache u.Ä. Außerdem sind noch weitere »virtuelle« Soundkanäle vorhanden (Soundkanäle Nr. 3 bis 8).

- Lingo-Kanal

In diesem Kanal können Bildskripte (Frame Scripts) platziert werden.

Sprite-Hierarchie

- Sprite-Kanäle

Ein Film kann bis zu 1000 Sprite-Kanäle enthalten. Im Dialogfeld »Filmeigenschaften« kann die Anzahl der Kanäle festgelegt werden. Sprites werden auf der Bühne in Ebenen angezeigt, und zwar in der Reihenfolge des Kanals, in dem sie im Drehbuch erscheinen. Das heißt, die Sprite-Hierarchie wird über den Kanal geregelt. Sprites in Kanälen mit höherer Nummerierung erscheinen über Sprites in Kanälen mit niedrigerer Nummerierung.

□ Eigenschaften und Sonstiges

Die wichtigen Sprite-Eigenschaften kann man in den Sprite-Beschriftungen anzeigen und verändern. Eine detaillierte Beschreibung der Eigenschaften sind im Sprite-Inspektor (Kapitel Der Sprite-Inspektor) zu finden.

Mehrere Drehbücher

Mit Hilfe von zusätzlichen Drehbuchfenstern ist es möglich, verschiedene Teile eines Films gleichzeitig anzuzeigen und in ihnen zu arbeiten. So kann man z.B. ein zweites Drehbuchfenster öffnen, um an einer anderen Stelle im Film zu arbeiten, ohne dafür den Bildlauf verwenden zu müssen. Außerdem können Sprites zwischen verschiedenen Drehbuchfenstern hin und her geschoben werden.

Weitere Fenster

In Director sind einige sinnvolle Fenster zur Erzeugung und Steuerung von Filmelementen wie Malfenster, Textfenster, Feldfenster, Skriptfenster etc. enthalten.

Einige dieser Fenster, die allgemein für die Entwicklung benötigt werden, seien hier kurz beschrieben:

Text und Felder

□ Text und Felder

Es gibt zwei Arten von Feldern für Texte. Diese können über das

»Textfenster« bzw. »Feldfenster« definiert werden. Textfelder können im Gegensatz zu »Feldfenster« mit Anti-Aliasing versehen werden. In Feldfenster können im Gegensatz zu Textfeldern Eingaben durch den Benutzer erfolgen.

- **Kommandofenster (*Message*)**
Sinnvolles Instrument zum Testen von Anweisungen und deren Auswirkung sowie für die Ausgabe von Information (Entwicklungsphase).

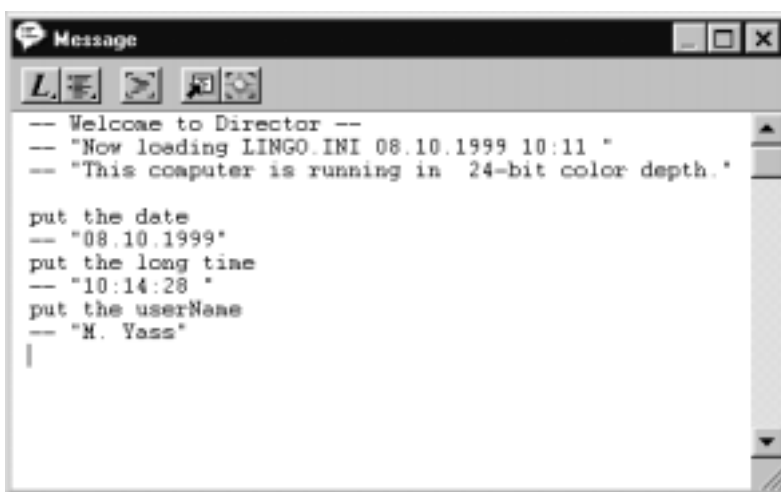


Abb. 5-11

Das Kommandofenster

- **Verhalten (*Behavior*)**
Verhalten sind Lingo-Skripte, die einzelnen Sprites zugewiesen werden können. Das System stellt einen Baukasten mit zahlreichen solcher kleiner Skripts zur Verfügung. Die Zuweisung erfolgt nach einem der im Folgenden beschriebenen Wege:
Menü: XTRAS -> Behavior Library
Menü: Windows -> Inspectors -> Behavior

Verhalten können aus der Verhaltensbibliothek auf einen Sprite mittels »Drag & Drop« gezogen werden (auf der Bühne bzw. im Drehbuch). Ein Frame kann nur ein Verhalten (Skript) haben. Sprites können mehrere Verhalten haben.

- **Rastergrafik und Vektorformen**
Bitmaps können im Malfenster erstellt oder aus einem der gängigen Grafikeditoren importiert werden (z.B. in GIF- oder JPEG-Format). In Director können Bitmaps auch mit Alphakanal-Daten (transparent) und animierten GIFs importiert werden. Das Malfen-

Rastergrafik

ster enthält eine Reihe von Werkzeugen, mit denen man die Bitmaps bearbeiten und Effekte auf die Bitmaps anwenden kann.

Director verwendet bei einer Bitmap das Anti-Aliasing. Das bedeutet, dass die Bitmap-Farben an den Rändern in die Hintergrundfarben übergehen und weichgezeichnet erscheinen.

Vektorformen

Vektorformen können mit Hilfe der Zeichenwerkzeuge im Vektorformfenster erstellt werden. Sie werden durch Definieren der Punkte gebildet, durch welche die Linie läuft. Die Vektorform kann aus einer Linie, Kurve oder unregelmäßigen Form bestehen und mit einer Farbe oder einem Verlauf gefüllt sein.

5.4.2 Animation und Effekte

□ Pfadanimation

Das letzte Bild des Sprites wird im Drehbuch markiert und das Sprite mit der Maus zur neuen Position bewegt. Die Größe kann durch Ändern der Größe des Objektes an seiner Endposition manipuliert werden.

- Schlüsselbilder werden mit <alt> Maus im »Score« definiert.
- Bearbeiten: Modifizieren -> Sprite-Füllen (*Tweening*) -> Pfad, Größe, Farbe, Beschleunigung usw.

□ Echtzeitaufnahme

Director merkt sich den Weg des Sprites (Aufzeichnen), auf dem es mit der Maus bewegt wird. Die Aufnahme beginnt erst bei dem markierten Bild.

- Im »Score« ab Bild ... markieren.
Menü: Steuerung (*control*) -> Echtzeit-Aufnahme (Real time recording). Sprite mit der Maus bewegen (Sprite rot umrahmt).
Loslassen, wenn fertig.

□ Zellanimationen

Auto-Verzerrung

Mit der Funktion »Auto-Verzerrung« kann man Animationen erstellen, deren Bitmap-Darsteller sich allmählich von einem Bild zum anderen verändern. Auto-Verzerrung erzeugt Zwischendarsteller für jeden Darsteller, der frei gedreht, in Perspektive gebracht, schräggestellt, verzerrt oder geneigt wird. Die folgende Schritte beschreiben die Erstellung der Zwischendarsteller:

- Bitmap-Darstellers auswählen (im Malfenster).
- Schaltfläche »Frei drehen«, »Perspektive«, »Neigen«, »Verzerren« oder »Strecken« zum Ändern der Grafik verwenden.

- Während die veränderte Grafik weiterhin ausgewählt ist, muss im Menü »Xtras« der Punkt »Auto-Verzerrung« gewählt werden. Die Anzahl der Darsteller, die erstellt werden sollen, kann nun eingegeben werden. Director erzeugt neue Darsteller und wendet auf jeden ein Zwischenmaß der Änderung an. Die neuen Darsteller erscheinen in den ersten verfügbaren Besetzungspositionen.

Die Animation kann wie folgt erstellt werden:

- Bilder markieren (ersten Darsteller im Cast anklicken; dann letztes Bild mit Shift-Klick markieren), mit der Maus ziehen und ins Drehbuch (*Score*) platzieren. Dabei soll Frame-Ende gleich Frame-Anfang gesetzt werden.

Solange alle Bilder im Score noch markiert sind:

- Modifizieren -> Bild in Kanal (*space in time*); die Trennung soll den Wert 1 haben.

5.4.3 Verwaltungs- und Sicherheitsaspekte

In diesem Kapitel werden einige Aspekte zur Modularisierung und zur Sicherheit der Anwendung erläutert.

Zerlegung und Organisation

Eine Anwendung (Film) kann in mehrere Teilfilme aufgeteilt werden. Dies hat viele Vorteile:

- Vermeidung von sehr großen Filmdateien. Da die Ladezeit eines Films von der Größe abhängt, kann man durch Zerlegen eines Films gute Effekte erzielen. Die Speicherverwaltung und die Übertragung (auch im Internet) ist effizienter und schneller.
- Durch die Modularisierung des Programms können Programmteile getrennt entwickelt werden.
- Teamarbeit: Mehrere Personen können gemeinsam an einem Projekt arbeiten. Voraussetzung für den Erfolg ist ein gutes Regiebuch, das die Programmteile sowie die festgelegten Projektstandards genau beschreibt.
- Bei hybriden CD-ROMs ist die Aufteilung empfehlenswert. Der Hauptfilm kann plattformabhängig (Mac, Windows etc.) erstellt

werden. Die weiteren Teile sind plattformunabhängig und können vom plattformabhängigen Hauptteil aufgerufen werden.

Die einzelnen Filmteile können mit Hilfe des Lingo-Befehls »play« aufgerufen werden.

Beispiel: Eine Anwendung soll in drei Teilfilme (Kapitel Käse, Wein, Brot) sowie einen Hauptfilm (Essen) aufgeteilt werden. Die Anwendung wird durch Starten von essen.exe ausgeführt. Die Navigation von und zu bzw. zwischen den einzelnen Kapiteln erfolgt über Grafikelemente (Symbole). Folgende Navigationsanweisungen werden benötigt:

- Sprung zum Kapitel: Dies wird mit Hilfe des Befehls
play movie »dateiname« ausgeführt, wobei »dateiname« keine exe-Datei ist.
- Die Rückkehr zur vorherigen Position erfolgt mit Hilfe des Befehls:
play done

Ein weiterer Aspekt der Zerlegung ist die Verwendung von externen Besetzungen. Externe Besetzungen sind völlig unabhängige Dateien (Bibliotheken für Darsteller). Die Verwendung von externen Besetzungen ist dann zu empfehlen, wenn in mindestens zwei Teilfilmen/Filmen dieselben Darsteller benötigt werden. In diesem Fall werden solche Darsteller in externe Besetzungen (Cast) abgelegt. Bei der Verwendung solcher Komponenten sollen folgende Punkte beachtet werden:

- Darstellerdateien müssen genau geplant und dokumentiert werden (Inhalte).
- Eine Verschiebung von Darstellern innerhalb einer Besetzung (Position) kann zu großen Konflikten führen.
- Änderungen an einem Darsteller (z.B. Eigenschaften) haben Auswirkungen auf andere Filme/Teilfilme, die auf die gleiche Darstellerdatei zugreifen.

Organisieren der Filmdaten

Die beste Methode zum Organisieren einer großen Produktion besteht darin, eine kleine Projektor-Datei zu erstellen, die den Film startet und dann in Shockwave- oder geschützte Filme verzweigt. Auf diese Weise braucht man den Projektor nicht jedesmal neu zu erstellen, wenn ein Teil des Films geändert wird.

Alle verknüpften Medien (Bitmaps, Sounds, Digitalvideos usw.) müssen sich auf derselben relativen Position befinden, auf der sie sich auch beim Erstellen des Films befanden. Damit keine verknüpften Medien vergessen werden, wenn man einen Film auf einem Datenträger

ger ausliefert, sollte man stets alle verknüpften Dateien im selben Ordner wie den Projektor oder im Projektor-Ordner ablegen.

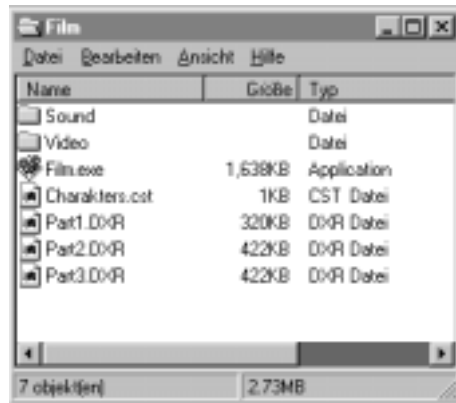


Abb. 5-12

Organisation der Daten

Erstellung von ausführbaren Programmen

Nach der Fertigstellung eines Films bestehen mehrere Möglichkeiten festzulegen, in welcher Form er an Benutzer ausgeliefert werden kann. Filme können als Projektoren, Shockwave-Filme (DCRs), geschützte Filme (DXRs) oder Java-Applets vertrieben werden.

Vertrieben von Filmen

Ein Film kann hinsichtlich Lingo-Quellcode und Besetzung geschützt werden. Dies ist immer vor der Auslieferung eines Prototyps bzw. der Anwendung bei einem Kunden auszuführen. Director entfernt die o.g. Komponenten aus dem Film. Die ursprüngliche dir-Datei wird bei der Konvertierung mit einer dxr-Datei überschrieben. Dies ist unwiderruflich, deshalb müssen zuvor unbedingt Kopien erstellt werden.

Geschützt speichern

In der Regel werden Filme entweder als Shockwave-Film, der auf einer Web-Seite abgespielt wird, oder als Projektor, der auf den Computer des Benutzers geladen oder auf einem Datenträger vertrieben wird, vermarktet.

Projektoren, Shockwave-Filme, geschützte Filme und Java-Applets können nicht in Director bearbeitet werden. Sie müssen die Quelldatei bearbeiten und anschließend einen neuen Film in einem der Vertriebsformate erstellen. Speichern Sie stets Ihre Quelldateien.

Quellfilme (DIRs) sind in der Regel nicht zu vertreiben, es sei denn, man möchte Benutzer in die Lage versetzen, den Film in der Director-Autorenumgebung zu bearbeiten.

Ein Projektor ist eine eigenständige (Stand-alone-) Version eines Films, der nicht in einem Web-Browser abgespielt werden soll. Projektoren können die gesamte Software (einschließlich Xtras) enthalten, die zum Abspielen eines Director-Films erforderlich ist. Durch Aktivie-

EXE-Filme (Projektor)

ren der Option »Shockwave-Player« im Dialogfeld »Projektoroptionen« kann die Größe von Projektoren erheblich verringert werden. Ist diese Option ausgewählt, entfernt Director die Player-Software aus dem Projektor und fügt lediglich die Dateien ein, die den Shockwave-Player im aktuellen System finden. Diese Art kleinerer Projektoren kann wie eine normale Anwendung auf dem Computer ausgeführt werden, allerdings muss dazu der Shockwave-Player im System installiert sein. Ein kleiner Projektor eignet sich hervorragend zum Herunterladen von Filmen, die nicht in einem Web-Browser abgespielt werden sollen. Sie können mehrere Filme, externe Besetzungen, Xtras und verknüpfte Medien in eine einzige Projektor-Datei packen. Weiterhin können Sie die Filmdaten mit der Shockwave-Komprimierung in einem Projektor komprimieren.

Mit der Option »Vollbild« erscheint der Bereich des Bildschirms, der von der Bühne nicht verdeckt wird, in der Farbe der Bühne.

Folgende Hinweise zu Projektoren sind zu beachten:

- ❑ Projektoren sind systemspezifisch: Versionen für Mac, Windows müssen beachtet/erstellt werden.
- ❑ Projektoren können keine anderen Projektoren aufrufen. Sie können sowohl geschützte als auch ungeschützte Director-Dateien abspielen.
- ❑ Falls einzelne Darsteller die Option »Mit Datei verknüpfen« haben, müssen diese externen Dateien mit dem Projektor ausgeliefert werden, da sie erst zur Laufzeit eingelesen werden.

DXR-Filme

Geschützte Filme (DXRs) sind unkomprimierte Filme, die vom Benutzer nicht bearbeitet werden können. Dieses Format ist zu empfehlen, wenn unkomprimierte Filme auf Datenträgern vertrieben werden. Geschützte Filme können schneller von einem Datenträger abgespielt werden als Shockwave-Filme, da sie nicht erst dekomprimiert werden müssen. Diesen Filmen ist der Vorzug zu geben, wenn genügend Speicherplatz zur Verfügung steht. Geschützte Filme enthalten - genau wie Shockwave-Filme - weder Informationen, die zum Bearbeiten des Films erforderlich sind, noch die Software zum Abspielen des Films. Sie können deshalb nur in einem Projektor, als Film in einem Fenster oder in einem Shockwave-Player abgespielt werden.

DCR-Filme (Shockwave)

Ein Shockwave-Film (auch DCR genannt) ist eine komprimierte Version der Daten eines Films. Shockwave-Filme werden hauptsächlich zum Vertrieb im Internet und zum Abspielen in einem Web-Browser erstellt. Man kann damit auch komprimierte Filme, die nicht in einem Projektor enthalten sind, auf einem Datenträger vertreiben.

Wird ein Film als Shockwave-Film gespeichert, werden die Daten komprimiert, alle benötigten Informationen zum Bearbeiten des Films entfernt und keine Software zum Abspielen des Films mitgeliefert. Ein Shockwave-Film kann in einem Web-Browser abgespielt werden, wenn der geeignete Shockwave-Player (bzw. Plug-in) im System installiert ist. Außerdem kann er als Film in einem Fenster oder in einem Projektor abgespielt werden.

Bei Filmen, die im Internet in einem Web-Browser abgespielt werden sollen, müssen alle verknüpften Medien beim Abspielen des Films auf der angegebenen URL vorhanden sein.

Ein von Director erstelltes Java-Applet ist ein Film, der in Java konvertiert wurde. Java-Applets erfordern keinen Shockwave-Player und bieten eine Alternative zum Abspielen einfacher Filme auf Websites, auf denen keine Plug-ins erlaubt sind. Da nicht alle Director-Funktionen für einen als Java gespeicherten Film verfügbar sind, müssen eine Reihe von Fragen bei der Erstellung berücksichtigt werden. Java-Applets können nicht in einen Projektor integriert oder als Film in einem Fenster abgespielt werden.

Java-Applet

Die Erstellung von ausführbaren Programmen erfolgt mit Hilfe des Menüpunkts Datei -> Projektor erstellen (*Create Projector*). Director wandelt eine dir-Datei in ein völlig eigenständiges Programm (Projektor) um. Die ursprüngliche Datei bleibt erhalten. Das Programm ist ohne Director abspielbar und darf lizenzfrei weitergegeben werden. Beim Starten des Programms werden die benötigten Runtime-Komponenten in einem temporären Verzeichnis entpackt.

EXE-Film (Projektor)

Erstellen von Shockwave-Filmen

Ein geöffneter Film (.dir) kann wie folgt in einen Shockwave-Film gespeichert werden:

- Aus dem Menü »Datei« den Punkt »als Shockwave-Film speichern« wählen
- Den Namen und den Pfad für die neue Datei eingeben. Damit Probleme mit verknüpften Medien vermieden werden, sollte man neue Shockwave-Filme stets im gleichen Ordner wie die DIR-Originaldatei speichern. Director fügt die Dateierweiterung .dcr automatisch hinzu.
- Ein HTML-Dokument mit dem notwendigen Code zum Abspielen des aktuellen Films in einem Browser kann erstellt werden (»HTML erstellen« wählen). Das HTML-Dokument wird mit

denselben Dateinamen und der Erweiterung .htm in Windows bzw. der Erweiterung .html auf dem Macintosh erstellt.

Wenn ein Shockwave-Film ins Internet gestellt werden soll und Xtras erfordert, muss man dafür sorgen, dass die Xtras im Dialogfeld »Film-Xtras« aufgeführt sind und die Option »Bei Bedarf herunterladen« für jedes erforderliche Xtra ausgewählt ist.

5.4.4 Weitere Hilfsmittel

Film ausdrucken

Ein Film kann auf verschiedenste Weise aus der Erstellungsumgebung ausgedruckt werden. So kann eine Abbildung der Bühne, des Drehbuchs, die Darstellernummer und der Inhalt von Textdarstellern im Besetzungsfenster, alle Skripte oder ein bestimmter Skriptbereich (Film-, Besetzungs-, Drehbuch- und Sprite-Skripte), die Kommentare im Markierungsfenster, die Grafiken des Besetzungsfensters oder das gesamte Besetzungsfenster ausgedruckt werden.

Das Ausdrucken des Filminhalts kann aus folgenden Gründen nützlich sein:

- Der Film kann auf Papier ausgegeben werden, um Änderungen zu markieren.
- Weiterleitung des Films an andere Teammitglieder.
- Anfertigung von Infozetteln für eine Präsentation.

Film ausdrucken

Komprimierung im Director bedeutet Speicherung eines Films mit dem geringstmöglichen Speicherbedarf. Dies führt zur Optimierung der Abspielgeschwindigkeit. Die Komprimierung soll durchgeführt werden, bevor ein Director-Film auf eine CD-ROM überspielt wird. Dies ist über »kompakt sichern« (*Save and Compact*) im Menüpunkt Datei (*File*) zu erreichen.

Speicherverwendung

Der Speicher-Inspektor zeigt an, welche Speichergröße Director für den Film zur Verfügung stellt, wie viel Speicher von verschiedenen Teilen des aktuellen Films verwendet wird und wie viel Speicher der Film insgesamt in Anspruch nimmt. Der Inspektor kann darüber hinaus alle entfernbaren Elemente aus dem Arbeitsspeicher löschen, bevor ein speicherintensiver Vorgang ausgeführt wird (Fenster -> Inspektoren -> Speicher).

Xtras

Xtras sind Softwarekomponenten, mit denen die Funktionalität von Director erweitert wird. Sie bieten wichtige Fähigkeiten, mit denen z.B. eine Verbindung mit dem Internet hergestellt werden kann. Xtras

geben Drittentwicklern die Möglichkeit, Director mit spezialisierten Funktionen zu erweitern.

Jedes Xtra, das in einem Film benötigt wird, muss zusammen mit dem Film vertrieben werden. Xtras können in Projektoren integriert sein oder von Benutzern aus dem Internet heruntergeladen werden.

5.5 Multimedia-Entwicklung mit ToolBook

Die ToolBook-II-Familie wurde für die Erstellung von Desktop-Anwendungen entwickelt, die auch online- und in hybrider Umgebung laufen sollen. Ein Schwerpunkt lag dabei im Bereich der interaktiven Lernsysteme. Folgende Hauptprodukte gehören zu dieser Familie:

- ToolBook II Assistant
Der Fach-Autor kann den ToolBook-II Assistenten einsetzen, um damit schnell und sicher Prototypen bis hin zu fertigen Anwendungen zu entwickeln. Er kann sich dabei voll auf die inhaltlichen Aspekte konzentrieren. Hierfür können Templates und vordefinierte Konstruktionselemente aus Bibliotheken (*Catalog*) verwendet werden. Assistenten (*Coach*) geben dem Autor ausführliche Objektinformationen sowie Vorschläge für die nächsten Arbeitsschritte.
- ToolBook II Instructor
Mit dem Instructor können Anwendungen funktional erweitert werden. Er ermöglicht u.a. Datenbankzugriff, komplexere Navigation oder berechnete Objektanimation. Auch eigene Templates und Katalogobjekte können erstellt werden. Natürlich kann die Entwicklung auch ausschließlich mit dem Instructor erfolgen. Assistent und Instructor sind voll kompatibel miteinander.
- Librarian
Ein System für die Verwaltung von Unterrichtsmaterialien auf Web-Basis. Die Bearbeitung und das Management der Unterrichtsinhalte bezieht sich auf die mit Assistant, Instructor sowie anderen Applikationen erstellten Inhalte. Die Schnittstelle zu Assistent- und Instructor-Lerneinheiten ist optimal. Fragestellungen und Bewertungselemente liefern Informationen zur Erfassung von Prüfungsergebnissen und anderen Maßnahmen an Librarian.

ToolBook-II Instructor ist in erster Linie ein Werkzeug, um interaktive Schulungsanwendungen und Trainingskurse im Internet oder Intranet zu realisieren. Das Paket enthält integrierte Konzepte und fördert den Einstieg auf unterschiedlichen Entwicklerebenen im Projektteam.

Der Instructor umfasst einige Katalogobjekte, die die Möglichkeiten des Autors erweitern. Beispiele hierfür sind der »Video Sequencer« (synchronisiert die Anzeige eines Textfeldes Frame-genau mit einem Videoclip), das »Call Out« (ein komfortabler Beschriftungspfeil), »Open Dokument Button« und Platzhalter für PowerPoint und andere Medien.

HTML Der HTML-Export unterstützt Features wie »cascading style sheets«. Damit können auch überlappende ToolBook-Objekte für das Web exportiert werden. Zur schnellen Datenübertragung via Internet lassen sich ToolBook-Anwendungen mit »Impulse for Neuron« um bis zu 80% komprimieren.

DHTML Die visuelle Programmierung mit dem »Actions Editor« erlaubt Entwicklern, Aktionssequenzen zu definieren und Funktionalitäten hinzuzufügen, die nach DHTML exportiert werden können. Damit ist der »Actions Editor« eine anwenderfreundliche und flexible Alternative zu OpenScript oder JavaScript.

Alle Katalogobjekte, die Pfadanimation und jedes Verhalten, die mit dem neuen »Event messaging«-Modell erstellt sind, können vollständig in das Internetformat konvertiert werden. Autoren können nun ihre Arbeit nach DHTML konvertieren. Das »Ereignis/Aktions«-System gestattet das Hinzufügen von Aktionen ohne Programmierung.

Der Instructor unterstützt eine Vielzahl von Objekttypen. Auch die große Anzahl von Fragetypen mit automatischer Bewertung, die in der DHTML/JavaScript Runtime erzeugt werden, können die Entwicklung von Lernprogrammen vereinfachen. Beispiel für Fragetypen sind:

- Lückentext
- Hotword-Lückentext
- Multiple-Choice-Schalter und -Felder
- Multiple-Choice-Schätzung
- Wahr/Falsch-Tests
- Multiple Choice (definierbar)
- Arrange Objects (anpaßbar)
- Drag- /Drop-Objekte zur Drag&Drop-Unterstützung
- Match-Item Felder
- Objekt für Zuordnungen (anpassbar)
- Schieberegler

Der »Universal Media Player« unterstützt fast alle modernen Medientypen. Das Einbinden der von der neuen Windows Medienwiedergabe unterstützten Formate ist einfach (AVI, WAV, MIDI, MPEG, QTW, MP3 etc.). Außerdem können die Formate von RealNetworks und Macromedia Flash verwendet werden.

Universal Media Player

Der Zugriff auf Datenbanken mit der ADO-Technologie (*Active Data Objects*) zum Arbeiten mit Microsoft Access und anderen Datenbankformaten wird vom Instructor unterstützt.

5.5.1 ToolBook - Aufbau und Konzept

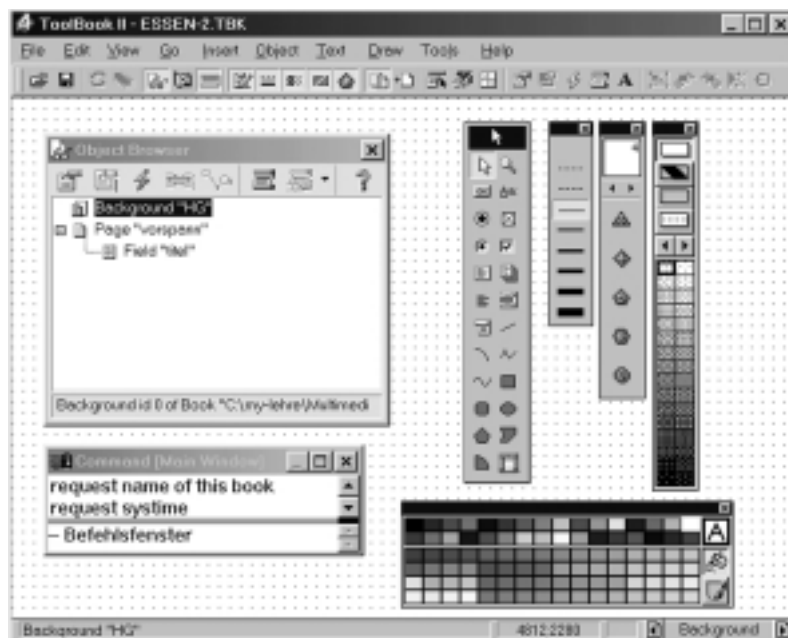
MM-ToolBook ist ein objektorientiertes Werkzeug zur Erstellung von MM-Anwendungen. Bei den Objekten handelt es sich um Objekte der realen Welt. Dies muss nicht zwangsläufig den Definitionen der OO-Techniken entsprechen. Eine Anwendung ist in Form eines oder mehrerer Bücher organisiert. Ein Buch besteht aus Seiten mit Objekten und Skripten (*Scripts*) und kann mehrere Kapitel umfassen.

Abbildung 5-13 zeigt eine Übersicht der Arbeitsfläche sowie Fenster und Werkzeugpaletten im Autorenmodus (Entwickler). Der »Object Browser« gibt eine Übersicht über den verwendeten Hintergrund, seine Seiten sowie die Objekte einer Seite. Das Befehlsfenster (*Command*) ermöglicht die Eingabe von Anweisungen z.B. zu Testzwecken. Die Werkzeugpalette enthält Werkzeuge (*Tools*) um Objekte wie Schaltflächen, Text- und Eingabefelder, Zeichenobjekte u.a. zu erzeugen.

Befehlsfenster

Weitere Paletten für Linienarten, geometrische Formen, Schattierungen sowie Farbmanipulation werden zur Verfügung gestellt.

Abb. 5-13
Übersicht ToolBook



Objektprinzip

Objekte (*Objects*) haben Eigenschaften und besitzen Methoden zur Behandlung von Ereignissen (*Events*). Im Folgenden sind einige Beispiele für Objekte aufgeführt:

- Buch (*Book*)

tbk Die Anwendung wird in Form eines bzw. mehrerer Bücher organisiert und gespeichert. Neben den normalen Büchern existieren Systembücher und eigendefinierte Systembücher. Sie enthalten globale Behandlungsroutinen für die Bearbeitung von bestimmten Ereignissen. Systembücher werden in einer Liste mit dem Namen »sysBooks« innerhalb einer Anwendung verwaltet. Sie haben die Dateierweiterung ».sbk«.

sbk
- Hintergrund (*Background*)

Der Hintergrund ist ein einheitliches Designelement, das von mehreren Seiten eines Buches gemeinsam verwendet werden kann. Jedes Buch hat mindestens einen Hintergrund. Objekte auf einem Hintergrund erscheinen auf jeder Seite, die diesen Hintergrund verwendet. So können Navigationselemente, gestalterische Komponenten u.Ä. auf dem Hintergrund platziert werden. Die Objekte des Hintergrunds werden für alle seine Seiten verwendet.

□ Seiten (*Pages*)

Ein Seitenobjekt stellt die Basiseinheit eines Buches dar. Objekte, die auf einer Seite angelegt werden, erscheinen vor Objekten auf dem Hintergrund und können diese überlagern.

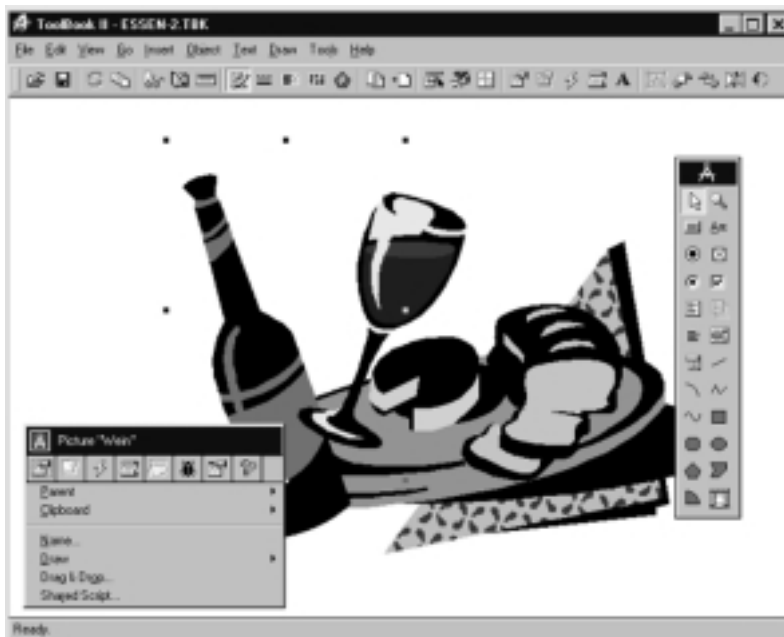


Abb. 5-14

ToolBook-
Seitenübersicht

Abbildung 5-14 zeigt den Entwurf einer Menüseite. Mit Hilfe der Werkzeugpalette können Schaltflächen, Zeichenobjekte, Felder u.a. erzeugt werden. Das Bild enthält Objekte, die mit Hilfe des Malwerkzeugs »irregularPolygon« genau umrahmt werden. Die Eigenschaften »Linienart« und »Füllfarbe« des neuen Objektes können nun auf »transparent« gesetzt werden. Die Navigation kann mit Hilfe der unsichtbaren Objekte erfolgen. Die Eigenschaften eines Objektes können mit Hilfe der rechten Maustaste verarbeitet werden (s. Fenster Abb. 5-14).

□ Schaltflächen (*Buttons*)

□ Grafikobjekte (*Graphic Objects*)

□ Felder (*Fields*)

□ Ansichtobjekt (*Viewer*)

Ein Fenster, das jede Seite eines beliebigen Buches zeigt. Eine Anwendung kann mehrere Viewer haben.

- Clip-Rahmen (*Stage*)
In einem Clip-Rahmen können Videosequenzen abgespielt oder Rastergrafiken gezeigt werden.

Hierarchie Die Objekte sind hierarchisch angeordnet. Die Hierarchie der Objekte (*Object Hierarchy*) ist wie folgt festgelegt:

ToolBook

System books (enthalten gemeinsame Skripte (global))

books (Anwendung)

backgrounds (Hintergrund)

pages (Seiten)

groups (Gruppen)

fields, buttons, graphic
objects

Ereignisse

Ereignisse (*Events*) werden durch den Anwender oder vom System ausgelöst. Wenn ein Ereignis eintritt, sendet das System eine Botschaft an das betroffene Objekt. Objekte können dann mit Behandlungsroutinen eine Aktion durchführen. Beispiele für Ereignisse sind:

- Tastatureingaben (*Keyboard event*)
- Mausereignisse (*Mouse events*)
- sonstige Ereignisse (*Enter Page, Start Applikation etc.*)

Handler

OpenScript-Anweisungen dienen zur Behandlung eines Ereignisses (Behandlungsroutinen). Ein Objekt kann mehrere »Handler« haben. Wenn das Objekt keine Behandlungsroutine zur Bearbeitung des Ereignisses besitzt, wird die Botschaft in der Hierarchie weitergeleitet. Die Skriptsprache »OpenScript« wird für die Erstellung von Behandlungsroutinen verwendet. Skripte können auch mit Hilfe eines Assistenten (Autoskript) erstellt werden.

Arbeitsebenen in ToolBook

ToolBook verfügt über zwei Ebenen. Die erste ist die Entwicklungsebene für Autoren (*Author Levels*) und die zweite ist die Anwender-

ebene (*Reader Levels*). Der Entwickler kann jederzeit seine Arbeit testen, indem er in die Anwenderebene wechselt (z.B. Taste F3). Die folgende Tabelle zeigt einen groben Vergleich, was auf welcher Ebene möglich ist.

Author	Reader
Entwicklung einer Anwendung	
Erzeugung von Objekten	
Def. der Objekteigenschaften	
Schreiben v. Skripten	
Textfelder bearbeiten	Textfelder bearbeiten
Ausdrucken	Ausdrucken
Starten von Applikationen	Starten von Applikationen

5.5.2 Terminologievergleich Director - ToolBook

Director und ToolBook liegen unterschiedliche Philosophien zugrunde. Die folgende Tabelle zeigt einen Vergleich der verwendeten Terminologie.

Einige der Begriffe sind selbstverständlich nicht identisch, geben jedoch einen Hinweis für die Programmierung.

Director	ToolBook
Darsteller (<i>Internal Cast</i>)	Ressourcen
Darsteller (<i>External Cast</i>)	Clip
Film (<i>Movie</i>)	Buch (<i>Book</i>)
Bild (<i>Frame</i>)	Seite (<i>Page</i>)
Window	Ansichtobjekt (<i>Viewer</i>)
Lingo (Skriptsprache)	OpenScript (Skriptsprache)