

# Schneller ROI garantiert

LiveCycle Formular-Training: Das gesamte Formularwesen vereinheitlichen, automatisieren und mit Geschäftsprozessen integrieren

Formulare auf Basis der Adobe PDF-Technologie in allen Varianten entwerfen, erstellen, adaptieren und damit die abgerufenen Daten optimal für die elektronische Weiterverarbeitung qualifizieren: ein Verfahren mit zahlreichen Vorteilen, von denen Behörden, Institutionen, Banken, Versicherungen und Industrieunternehmen nachhaltig profitieren.

## Von der Datenerfassung zur Integration in Backendsysteme

Das Formularwesen zu vereinheitlichen bedeutet auf der einen Seite die Datenerfassung durch den Einsatz von Adobe PDF zu optimieren und auf der anderen Seite die Integration der Daten und ausgefüllten Dokumente in Backendsysteme sicherzustellen.

Adobe stellt mit dem Adobe LiveCycle Designer die dazu passende und rundum professionelle Formular-Autorenumgebung bereit – als Teil von Adobe Acrobat oder der Adobe LiveCycle Enterprise Suite. Letztere zur Automatisierung von Geschäftsprozessen, der Datenintegration und Einbindung in Backendsysteme.

Entsprechend spannt das Whitepaper den Bogen von der Erstellung statischer und dynamischer Formulare zur Integration dieser Formulare in Prozesse und Systeme.

## Einführung in die Entwicklung statischer PDF-Formulare

Das Adobe PDF-Format hat sich als elektronisches Austauschformat längst etabliert und begegnet Endanwendern als elektronisches Formular immer häufiger. Ein Adobe PDF-Formular ist die computerbasierte Form eines Formulars und kann mithilfe des kostenlosen Adobe Reader® ausgefüllt, gespeichert und elektronisch weitergeleitet werden.

Somit wird das Konzipieren, Erstellen und Verteilen von PDF-Formularen zu einem profitablen Standardprozess für viele behördliche, institutionelle und geschäftliche Einsatzmöglichkeiten.

Zwei unterschiedliche Ansätze der Formulargenerierung sind möglich:

1. Die einfache Erstellung interaktiver PDF-Formulare mit Adobe Acrobat auf Basis einer vorgegebenen Formularpalette mit festem Layout.
2. Die Erstellung interaktiver, dynamischer PDF-Formulare mit dem Adobe LiveCycle Designer. Diese Variante setzt auf einer XML-Basis auf.

Damit lassen sich bestehende PDF-Dateien bearbeiten, völlig neue Formularlayouts gestalten und sogar dynamische PDF-Formulare auf Basis von XML-Technologie erzeugen.

## Ein erster Blick auf den Designprozess mit dem Adobe LiveCycle Designer

Dieses grafische Autorenwerkzeug bietet die umfassende Lösung, um Formulare zu entwerfen, die zugehörige Logik zu definieren und das Ganze sogar an gesetzliche Vorgaben anzupassen. Die Einarbeitung geht denkbar schnell. Vor dem Beginn der Formularerstellung kann ausgewählt werden, ob auf bereits bestehende Vorlagen zurückgegriffen oder ein völlig neuer Entwurf gestaltet werden soll. In beiden Fällen erzeugen die Formularwerkzeuge des Programms ein interaktives und intelligentes Formular. Dabei werden die dafür verantwortlichen Mitarbeiter von einem Assistenten im Programm durch die einzelnen Schritte geführt.

Aus der Objektbibliothek lassen sich schnell und mühelos per Drag & Drop die benötigten Formularbausteine auf die Seite ziehen. Neu angelegte Formularbestandteile können in der Objektbibliothek archiviert werden und erweitern diese. So können ganze Gruppen und Serien von weiterverwendbaren Formularteilen samt Eigenschaften und Logik entstehen: zentral abgelegt und für alle berechtigten Mitarbeiter jederzeit verwendbar.

Im Rahmen der Formularerzeugung generiert LiveCycle Designer den zugehörigen XML-Quellcode automatisch mit. Alle ins PDF eingetragenen Formulardaten können mitsamt der Datei lokal gespeichert und aus dem PDF heraus als XML-Datenstrom an weiterverarbeitende Systeme oder auch per E-Mail übergeben werden. Da LiveCycle Designer auf Basis von XML arbeitet, können die resultierenden PDF-Formulare dynamische Funktionen ausführen, die das Layout eines PDF-Formulars an den Inhalt anpassen. Änderungen und die gesamte Pflege des Formulars geschehen in einer einzigen Datei.

Komplettierte Formulare lassen sich als Vorlage speichern und können dann als Ausgangsmaterial für neue Formulare dienen. Strukturiert angelegte Objektbibliotheken und individuelle Vorlagen helfen dabei, den Prozess der fortlaufenden Formularerstellung und -adaptierung effizient und rentabel zu gestalten.

Ein wichtiger Aspekt dieser Lösung liegt aber auch auf der Validierung von Eingabedaten, damit diese hocheffizient den anschließenden Prozessschritten zugeführt werden können.

### **Validierung der Eingabedaten**

Die Datenerfassung über PDF-Formulare bietet den großen Vorteil, dass sich Eingabedaten validieren lassen. Dies ist produktiv in zweifacher Hinsicht:

- Endanwender werden bereits während der Datenerfassung auf mögliche Fehleingaben hingewiesen.
- Die Datenqualität steigt deutlich an, da nur valide Daten in die Weiterverarbeitung gehen.

### **Die Eingaben der Endanwender lassen sich präzise prüfen**

Für die einzelnen Felder sind drei Überprüfungsschritte in dieser Reihenfolge möglich:

- Feld auf Null-Inhalt testen
- Format des Feldwerts anhand eines bestimmten Feldmusters prüfen  
(Weitere Informationen zum Thema Feldmuster unter: [Einfache Muster](#))
- Überprüfungsskript aufrufen

Für die systematische Prüfung von Endanwendereingaben kann ein Überprüfungs muster definiert werden. Dieses hilft, die Eingabe für Datums-/Uhrzeitfelder, numerische Felder, Textfelder und Kennwortfelder genau zu prüfen. Standardmäßig werden Leereingaben nicht akzeptiert, wenn ein Wert im betreffenden Feld erforderlich ist. Der Abgleich von Rohwerten erfolgt direkt mit dem Überprüfungs muster. Entsprechen sie dem Muster, werden sie für die Anzeige formatiert.

Stimmen eingegebene Werte nicht mit dem Überprüfungs muster überein, erfolgt eine entsprechende Warnmeldung. Diese kann standardisiert sein oder durch eigene Überprüfungs muster-Meldungen ersetzt werden.

Zur Prüfung der Endanwendereingabe lässt sich auch ein Überprüfungs skript verwenden; zusätzlich zum Muster oder wenn ein Überprüfungs muster nicht unterstützt wird, z. B. bei Optionsfeldergruppen und Kontrollkästchen. Die Überprüfung der Eingabe per Skript stellt sicher, dass die Daten optimal weiterverarbeitet und verwertet werden können. Eigene Meldungen und Laufzeitfehler- oder Warnmeldungen werden ebenfalls unterstützt.

Mit Hilfe der Optionen auf der Registerkarte „Formularüberprüfung“ im Dialogfeld „Formulareigenschaften“ lässt sich konfigurieren, wie

- Prüfmeldungen angezeigt
- Felder mit Fehlern oder erforderliche Felder mit ungültigen Daten bzw. ohne Daten gekennzeichnet
- der Fokus auf das erste Feld mit Fehler gelegt wird

Weitere Informationen dazu unter: [Anzeigen von Prüfungsfehlern in Adobe Acrobat](#).

Um sicherzustellen, dass der jeweilige Endanwender einen gültigen Wert in das Feld eingibt, kann ein Überprüfungsmuster mit einem Wert aus einer Datenquelle dynamisch ausgefüllt werden.

#### **Anleitung zur Festlegung von Überprüfungsmuster und eigener Meldung**

1. Datums-/Uhrzeitfeld, numerisches Feld, Textfeld, Kennwortfeld, Listenfeld oder die Dropdown-Liste auswählen.
2. In der Palette „Objekt“ die Registerkarte „Wert“ anklicken.
3. Nun auf „Überprüfungsmuster“ klicken und in der Liste „Typ auswählen“ eines der vordefinierten standardisierten Überprüfungsmuster selektieren; alternativ unter „Muster“ ein eigenes Muster eingeben.
4. Im Feld „Überprüfungsmuster-Meldung“ eine Meldung eingeben, mit der die Endanwender zur Eingabe des richtigen Werts aufgefordert werden. Diese Meldung sollte das erforderliche Eingabeformat angeben. Soll die Meldung einen Zeilenumbruch enthalten, dafür die Steuerungs- und die Eingabetaste gleichzeitig drücken.
5. Soll anstelle der Warnmeldung eine Programmierfehlermeldung erscheinen, dafür die Option „Fehler“ auswählen.

#### **Eine Meldung anzeigen, sobald ein angehängtes Skript eine unzulässige Eingabe feststellt**

1. Zunächst Datums-/Uhrzeitfeld, numerisches Feld, Textfeld, Kennwortfeld, Listenfeld, Kontrollkästchen, die Dropdown-Liste oder die Optionsfeldergruppe auswählen.
2. In der Palette „Objekt“ auf die Registerkarte „Wert“ klicken und in das Feld „Überprüfungsskript-Meldung“ die gewünschte Meldung eingeben.
3. Soll anstelle der Warnmeldung eine Programmierfehlermeldung erscheinen, einfach die Option „Fehler“ auswählen.

#### **Vorgegebene Datenstrukturen (XML Schema) in neue Formulare einbinden**

Die Entwürfe neuer Formulare lassen sich so konfigurieren, dass sie über einfache Datenerfassung weit hinausgehen. Mit Designer ES2 können neu konfigurierte Formulare auch mühelos in Unternehmensdatenbanken oder Webdienste integriert werden. So kommen Datenerfassungslösungen zustande, die sich einfach entwickeln und verwalten lassen. Die über diese Formulare gewonnenen Daten der Endanwender sind von den Firmendatenquellen lesbar und können mit diesen abgeglichen bzw. diesen hinzugefügt werden. Für eine Lösung ist es lediglich erforderlich, den jeweiligen Formularentwurf mit einer oder mehreren Datenquellen zu verknüpfen und vorhandene Daten an eines oder mehrere Felder im Formularentwurf anzubinden.

#### **Die Anbindung von Formularfeldern an Datenquellen per XML-Schema**

Über das XML-Schema wird festgelegt, wie die Elemente in einem XML-Dokument formell beschrieben werden müssen. Im Prinzip stellt ein XML-Schema also die Weichen, damit die über das Formular erfassten Daten den spezifischen Anforderungen der weiterverarbeitenden Datensysteme entsprechen. Durch die Verbindung mit einem XML-Schema lassen sich die im Schema definierten Elemente und Attribute an die Felder im Formularentwurf binden. Für diese Verbindung kann auch das XML-Daten-Stammelement ausgewählt werden.

#### **Die Datenverbindung zwischen Formular und XML-Schema erstellen**

Mit dem LiveCycle Designer ist es mühelos möglich, das gewünschte XML-Schema in den Entwurf eines Formulars zu importieren. Die einzelnen Objekte im Formularentwurf lassen sich dann an die Elemente und Attribute in der XML-Schemadefinition binden.

Hinweis: Wird ein XML-Schema importiert, das ein Schemaelement des Typs <xs:include> oder <xs:import> enthält, muss das Elementattribut „schemaLocation“ auf eine Datei im lokalen Dateisystem verweisen.

1. In diesem Fall einen der folgenden Schritte ausführen:
  - „Datei“ > „Neue Datenverbindung“ wählen.
  - Ein Objekt auf der Seite auswählen. In der Palette „Objekt“ auf die Registerkarte „Bindung“ klicken und im Menü „Datenverbindung“ die Option „Neue Datenverbindung“ selektieren.
2. Dann im Feld „Name der neuen Verbindung“ einen Namen für die Verbindung eingeben.
  - Der benutzerdefinierte Name muss aus einem Wort bestehen und darf max. 127 Zeichen enthalten. Das erste Zeichen muss ein Buchstabe oder ein Unterstrich (\_) sein. Als restliche Zeichen sind Buchstaben, Ziffern, Bindestriche (-), Unterstriche (\_) oder Punkte (.) möglich.
3. Nun „XML-Schema“ auswählen und auf „Weiter“ klicken.
4. Dann einen der folgenden Schritte ausführen:
  - „Durchsuchen“ anklicken, die XML-Schemadatei suchen und auswählen.
  - URL-Verzeichnis der XML-Schemadatei in das Feld „XML-Schemadatei auswählen“ eingeben oder einfügen.
  - „Durchsuchen“ anklicken, die XML-Schemadatei suchen und auswählen.
  - „Durchsuchen“ anklicken, die XML-Schemadatei suchen und auswählen.
  - „Durchsuchen“ anklicken, die XML-Schemadatei suchen und auswählen.
5. Nun eine oder mehrere der folgenden Optionen festlegen:
  - Im Menü „XML-Daten-Stammelementname verwenden“ das Stammdatenelement auswählen, das zur Benennung des Stamnteilformulars des Formularentwurfs verwendet wird. Das Menü enthält globale Elementdeklarationen im XML-Schema.
  - (Optional) „XML-Schema einbetten“ auswählen, um das XML-Schema als Paket in der XDP-Datei einzubetten.
  - (Optional) „Eingangsdaten umformen“ aktivieren, auf „Durchsuchen“ klicken, um die XSLT-Datei zu suchen und auszuwählen, die verwendet wird, um Eingangsdaten für den Formularentwurf umzuformen.
  - (Optional) „Ausgangsdaten umformen“ aktivieren, auf „Durchsuchen“ klicken, um die XSLT-Datei zu suchen und auszuwählen, die verwendet wird, um Ausgangsdaten für den Formularentwurf umzuformen.
6. Auf „Fertig stellen“ klicken und die Daten werden in der Palette „Datenansicht“ angezeigt.
7. Hinweis:
 

Das at-Symbol (@) vor name in der Palette „Datenansicht“ zeigt, dass diese Node ein Attribut einer Options-Node ist. #data zeigt den tatsächlichen Wert der Options-Node an.

### **Organisation der Formularentwicklung**

Der Adobe LiveCycle Designer stellt wichtige und effiziente Funktionen bereit, speziell für ein umfangreiches und vielfältiges Formularwesen, in das ganze Teams eingebunden sind. Vor allem wiederkehrende Formulareile können schnell erstellt, verwaltet und wiederverwendet werden. Wertvolle Funktionalitäten für die optimierte Zusammenarbeit sind die Objektbibliothek und die Arbeit mit Formularfragmenten.

#### **Die Vorteile der Objektbibliothek**

Hier können immer wieder benötigte Formulareile – also einzelne Felder, Feldgruppen oder komplette Formularabschnitte – abgelegt und ganz nach Wunsch als Bausteine für neue Formulare eingesetzt werden. Dabei steht die Objektbibliothek als lokale Variante für den einzelnen Benutzer oder als zentrale Anwendung für mehrere verteilte Bearbeiter zur Verfügung.

#### **Künftig noch viel effizienter arbeiten mit komfortablen Formularfragmenten**

Dazu vorab einige grundsätzliche Bemerkungen: Der Begriff Fragment bezeichnet einen feststehenden, wiederverwendbaren Formulareil, also z. B. den Adressblock oder den Hinweis zum Copyright.

Die Verwaltung der Fragmente erfolgt in der Fragmentbibliothek. Dort werden sie als separate XDP-Dateien gespeichert und können bequem in Formularentwürfe eingefügt werden. Die Standalone-Version des LiveCycle Designer speichert Fragmente im Dateisystem

## **Hier ein kurzer Blick auf die Funktionalitäten:**

### **Name**

Nach der Erstellung muss das Fragment einen Namen erhalten. Wurde dafür ein bereits benanntes Teilformular- oder Skriptobjekt ausgewählt, dann dient der Name des Objekts als Standardwert. Wurde ein unbenanntes Objekt bzw. mehrere Objekte gewählt, lautet der Standardwert „Fragment <n>“, wobei <n> für eine Zahl steht.

### **Beschreibung**

Auf Wunsch kann für jedes Fragment zusätzlich eine Beschreibung eingegeben werden.

### **Neues Fragment in der Fragmentbibliothek erstellen**

In diesem Bereich wird eine Fragmentdatei in der angegebenen Bibliothek generiert.

### **Fragmentbibliothek**

Die Fragmentbibliothek, in der das neue Fragment gespeichert wird. Wird diese Bibliothek nicht in der Liste angezeigt, auf „Fragmentbibliothek öffnen“ klicken.

### **Dateiname**

Name der XDP-Datei, die das Fragment enthält. Als Standardname wird der Name des Fragments verwendet.

### **Auswahl durch Verweis auf neues Formularfragment ersetzen**

Ersetzt die Auswahl durch einen Verweis auf die neue Fragmentdatei. Bei späterer Aktualisierung der Fragmentdatei wird auch der Fragmentverweis angepasst.

### **Neues Fragment in aktuellem Dokument erstellen**

Definiert die Auswahl des Fragments in der aktuellen Datei. Es wird keine neue Fragmentdatei erstellt. Diese Option eignet sich z. B., um mehrere Fragmente in derselben XDP-Datei und Bibliothek zu erstellen.

### **Dialogfeld: ein Fragment einfügen**

Über dieses Dialogfeld (Anmerkung Texter: Hier soll doch sicher ein Screenshot abgebildet werden. Dieser ist in der Textvorlage nicht enthalten.) lassen sich Fragmente auswählen, um sie in einen Formularentwurf einzufügen. Dabei werden Skriptfragmente dem untergeordneten Element „Variablen“ des Stamnteilformulars zugeordnet. Damit kann man sie besser von den vorhandenen Skripten im Formularentwurf unterscheiden. Um das Dialogfeld aufzurufen, einfach „Einfügen“ > „Fragment“ anklicken.

und in der Fragmentbibliothek. Als Teil von Adobe LiveCycle Enterprise Suite werden Fragmente in einem gemeinsamen, zentralen Repository gespeichert.

Fragmente vereinfachen und beschleunigen das Formularwesen entscheidend. Beim Erstellen eines Formulars wird einfach ein Verweis auf das gewünschte Fragment eingefügt. Dieses wird dann im Formular angezeigt. Der Fragmentverweis enthält ein Teilformular, das auf die XDP-Datei verweist.

Alle Fragmente haben einheitliche Merkmale:

- Sie werden auf identische Weise erstellt.
- Sie können einzeln oder zu mehreren in der aktuellen Datei oder einer separaten Datei erzeugt werden. Wird ein Fragment in einer separaten Datei erstellt, generiert das Programm eine Datei für seine Speicherung im Dateisystem oder im LiveCycle ES2 Repository.
- Andere berechnete Formularverfasser können jedes Fragment nach Belieben verwenden.
- Wobei die Fragment-Quelldateien in LiveCycle Designer bearbeitet werden.

### **Zum Thema Fragmente und Teilformulare**

Um ein Fragment zu erstellen, können entweder ein vorhandenes Teilformular oder ein bis mehrere Objekte ausgewählt werden. In Teilformularen können auch Tabellen bzw. Tabellen-, Kopf- und Fußzeilen enthalten sein. Bei Objekten, die sich nicht in einem Teilformular befinden, werden diese bei der Fragmenterstellung in ein Teilformular aufgenommen.

Selbst Auswahl-Teilformularsätze mit mehreren Fragmentverweisen sind verwendbar. Dabei handelt es sich um eine Abwandlung des Teilformularsatzobjekts, mit dem man die Anzeige bestimmter Teilformulare im Satz mit Hilfe von bedingten Anweisungen anpassen kann. Bedingte Anweisungen dienen der Festzulegung, welches Teilformular aus dem Satz im bereitgestellten Formular angezeigt werden soll.

Beispiel: Jedes Teilformular eines Satzes enthält Informationen für einen bestimmten Zustand. Anhand dieses Zustands des Formulars bei seiner Einreichung lässt sich dann das anzuzeigende Teilformular bestimmen.

### **Wichtiges zum Aspekt Skriptfragmente**

Ein Skriptfragment enthält wiederverwendbare Java Script Funktionen oder Werte, wie einen Datumsparser oder einen Webdienstaufwurf. Diese Features können unabhängig von einem bestimmten Objekt gespeichert werden. Solche Fragmente enthalten ein Skriptobjekt, das in der Palette „Hierarchie“ als untergeordnetes Element von „Variablen“ aufgeführt wird. Grundsätzlich können Fragmente nicht aus Skripten erstellt werden, die Eigenschaften anderer Objekte sind, also auch nicht aus Ereignisskripten wie z. B. „validate“, „calculate“ oder „initialize“.

### **Fragmente lassen sich vielfältig und flexibel verwenden**

Immer wenn Formularinhalte oder Skripte in mehreren Formularen zum Einsatz kommen sollen, lohnt sich die Fragmenterstellung. Denn damit lassen sich schnell gemeinsame Elemente erstellen oder ändern, die mehrere Formularverfasser gemeinsam verwenden. Beispiel: Firmenlogo. Dieses kommt in vielen Formularen zum Einsatz. Und falls sich Logo-Änderungen ergeben, braucht man diese nur einmal direkt am Speicherort in der Fragmentdatei auszuführen.

Skriptfragmente sind außerdem auch hilfreich, wenn eine Funktion eine Standardmethode für die Formatierung oder für die Ausführung einiger Berechnungstypen definiert.

### **Generierte Inhalte produktiv wiederverwenden**

In Form der Fragmente lassen sich erarbeitete Formularinhalte immer wieder effizient verwenden. Denn der Fragmenteinsatz ist schneller und rationeller als den gewünschten Inhalt zu kopieren oder erneut zu generieren. Fragmente garantieren außerdem für den konsistenten Inhalt aller Formularversionen und ein einheitliches Erscheinungsbild.

**Auch hier ein kurzer Blick auf die Funktionalitäten:**

#### **Dateiname**

Bezeichnet die Datei, die das Fragment enthält.

#### **Dateityp**

Nennt das Dateiformat, wobei das XDP-Format als Standard ausgewählt ist.

#### **Fragment auswählen**

Hier kann das einzufügende Fragment aus einer Liste selektiert werden, die alle Fragmente der Datei anzeigt.

#### **Fragmentinformationen**

Hier ist die zum Fragment gehörende Beschreibung abrufbar.

#### **Fragmentvorschau**

Zeigt eine Vorschau des ausgewählten Fragments.

## **Sinnvoll im globalen Kontext für konsistente Aktualisierung**

Viele Unternehmen sind global präsent. Und in diesem Kontext macht es Sinn, Fragmente für konsistente Änderungen an mehreren oder vielen Formularen zu nutzen. Denn die Aktualisierung muss nur einmal am Speicherort ausgeführt werden. Dabei lassen sich Inhalt, Skriptobjekte, Datenbindungen, Layout und Stil verändern. Nach dem Speichern geben alle XDP-Formulare, die auf das adaptierte Fragment verweisen, die Änderungen zuverlässig wider. Um ein Fragment in einem PDF-Formular zu aktualisieren genügt es, das Formular erneut in LiveCycle Designer zu speichern.

Ein Beispiel verdeutlicht die Effizienz: Als gemeinsames Element vieler Formulare ist z. B. ein Adressblock denkbar, der ein Dropdown-Listenobjekt für das Land enthält. Will man die Werte für das Dropdown-Listenobjekt aktualisieren, sind jede Menge Formulare zu öffnen, um die Änderung überall vorzunehmen. Liegt der Adressblock aber in einem zentralen Fragment, reicht es aus, lediglich an einem Ort, nämlich in der Fragmentdatei zu ändern.

## **Quantensprung für die teambasierte Formularerstellung**

Fragmente sind ideal, um die Formularerstellung auf mehrere Mitarbeiter aufzuteilen. So können Formularentwickler mit Kenntnissen in der Skripterstellung und in den komplexeren Funktionen von LiveCycle Designer Fragmente erstellen, freigeben und die Skripten und dynamischen Eigenschaften nutzen. Gespeicherte Fragmente sind dann für alle Teammitglieder verfügbar. Und auch bei verteilter Erarbeitung weisen alle Teile eines Formulars konsistente Inhalte und einheitliche Funktionalitäten auf.

## **Zugriffsschutz für Fragmente**

In Verbindung mit der Adobe LiveCycle Enterprise Suite können Zugriffsberechtigungen für Fragmente eingerichtet werden.

## **Dialogfeld: ein Fragment erstellen**

In diesem Dialogfeld (Anmerkung Texter: Hier soll doch sicher ein Screenshot abgebildet werden. Dieser ist in der Textvorlage nicht enthalten.) kann ein Fragment aus den ausgewählten Objekten generiert werden. Um das Feld anzuzeigen, werden eines oder mehrere Objekte ausgewählt, die in das Fragment einbezogen werden sollen. Dafür einfach „Bearbeiten“ > „Fragmente“ > „Fragment erstellen“ aufrufen.

Wird das Fragment in einer Adobe LiveCycle ES Anwendung erstellt, erfolgt seine Speicherung im Repository und es wird in der Anwendungsansicht von Workbench ES2 angezeigt.

## Formularfragmente dynamisch einbinden und Formularvorlagen aus Bausteinen serverseitig erstellen

XDP-Dokumente zusammenführen

Mit dem Assembler-Dienst können mehrere XDP-Dokumente zu einem XDP-Dokument oder einem PDF-Dokument zusammengefasst werden. Dabei können bei XDP-Quelldateien mit Einfügemarke die einzufügenden Fragmente angegeben werden. Hier einige Varianten für das Zusammenführen von XDP-Dokumenten:

### Ein einzelnes XDP-Dokument assemblieren

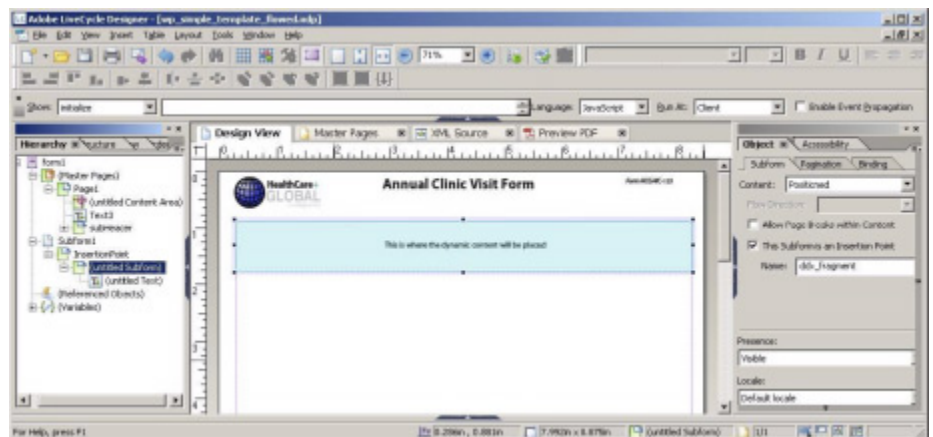
Diese Abbildung (Der Texter: In der Vorlage nicht enthalten) zeigt, wie drei XDP-Quelldokumente inklusive zugehöriger Daten zu einem XDP-Zieldokument zusammengefasst werden. Dabei bildet das erste XDP-Quelldokument das Basisdokument, von dem das Zieldokument die grundlegenden Basisattribute abrufen.

Das nachfolgende DDX-Dokument veranschaulicht das auf diesem Weg erzeugte Ergebnis.

```
<DDX xmlns="http://ns.adobe.com/DDX/1.0"> <XDP result="MyXDPRestult">
<XDP source="sourceXDP1"/> <XDP source="sourceXDP2"/> <XDP
source="sourceXDP3"/>
</XDP> </DDX>
```

### Dynamisches Einfügen von Formularfragmenten in ein XFA-Formular

Der Assembler-Dienst ist auch in der Lage, ein XFA-Formular zu erzeugen, das von einem anderen XFA-Formular mit eingefügten Fragmenten erstellt wurde. Diese Funktion ermöglicht die Verwendung von Fragmenten beim Erstellen mehrerer Formulare. Dabei wird das dynamische Einfügen von Formularfragmenten durch eine Quelle gesteuert. Lediglich eine einzige Quelle von häufig verwendeten Komponenten muss administriert werden. So kann z. B. ein Fragment für das Firmenbanner erstellt werden. Ändert sich das Banner, muss nur das Fragment angepasst werden. Formulare, die das Fragment enthalten, bleiben unverändert.



Im folgenden Beispiel sind drei Fragmente enthalten:

Die Datei wp\_simple\_contact.xdp enthält ein Fragment: „subPatient Contact.“

Die Datei wp\_simple\_patient.xdp enthält zwei Fragmente: „subPatientPhysical“ und „subPatientHealth“.

### Die „Mischenweisung“

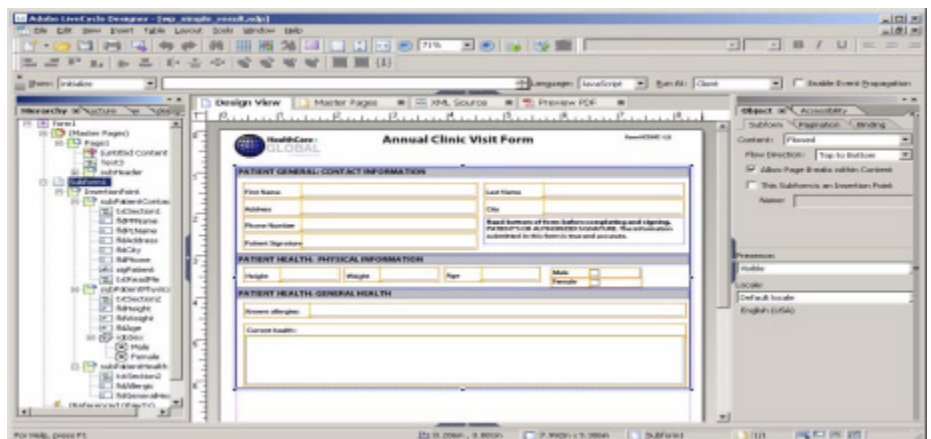
Über die sogenannte „Mischenweisung“ im Assembler-Dienst können verschiedenste Formularbausteine zusammengeführt werden, bis eine komplette Formularedatei entsteht.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- *****
*
* Simple Dynamic Assembly with a single XDP with one insertion point and
* multiple fragment flowed into that point.
*
* ***** -->
```

```

<DDX xmlns="http://ns.adobe.com/DDX/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ns.adobe.com/DDX/1.0/
http://spg.corp.adobe.com/pub/doc-o-matic/fibonacci/vrg/schemas/pdfm/ddx.xsd">
<XDP result="wp_simple_result.xdp">
<XDP source="wp_simple_template_flowed.xdp">
<XDPCContent insertionPoint="ddx_fragment" source="wp_simple_contact.xdp"
fragment="subPatientContact"
/>
<XDPCContent insertionPoint="ddx_fragment" source="wp_simple_patient.xdp"
fragment="subPatientPhysical"
/>
<XDPCContent insertionPoint="ddx_fragment" source="wp_simple_patient.xdp"
fragment="subPatientHealth" />
</XDP>
</XDP>
</DDX>

```



### PDF-Formulare komfortabel an Backendsysteme anbinden

Elektronische Formulare auf Basis der Adobe PDF-Technologie bieten unterschiedlichste Nutzungsvarianten. Zu den gebräuchlichsten Anwendungsformen gehören: Print & Fill, Fill & Print, Fill & Submit sowie Fill, Sign & Submit.

### PDF-Formulare und WebServices: eine produktive Symbiose

Was bietet ein Webservice? Das World Wide Web Consortium definiert seine Bereitstellung als Unterstützung zur Zusammenarbeit zwischen verschiedenen Anwendungsprogrammen, die auf unterschiedlichen Plattformen und/oder Frameworks betrieben werden. Ein Webservice, auch Webdienst genannt, ist im Prinzip eine Software-Anwendung, die mit einem Uniform Resource Identifier (URI) eindeutig identifizierbar ist und deren Schnittstelle als XML-Artefakt definiert, beschrieben und gefunden werden kann. Vor allem unterstützt ein Webservice die direkte Interaktion mit anderen Software-Agenten und verwendet dafür XML-basierte Nachrichten durch den Austausch über internetbasierte Protokolle.

Aus der Perspektive der Datenerfassung stellt ein Webservice folglich eine Vereinbarung darüber dar, mit welchen Daten und Methoden auf eine zentrale Anwendung zugegriffen werden kann.

Designer ES2 ist in der Lage, WebServices zu unterstützen. Die mit dieser professionellen Formular-Autorenumgebung entwickelten PDF-Formulare können diese Services aktiv nutzen. Ein weiterer entscheidender Vorteil kommt hinzu. Denn die Entwicklung von PDF-Formularen ist viel schneller und kostengünstiger als z. B. die Programmierung von Client-Anwendungen mit klassischen Sprachen wie C++ oder Java. Damit stellen sie eine hochinteressante Alternative dar, um die elektronische Datenerfassung auf eine vollkommen neue und viel wirtschaftlichere Basis zu stellen.

## Grundsätzliches zur Erstellung einer Datenverbindung mit einer WSDL-Datei

Ein Webservice stellt eine Anzahl von Operationen dar. So wird dies in einem Webservice-Beschreibungssprache-Dokument (WSDL-Dokument) definiert. WSDL kann für jede Operation eine Eingangsmeldung, eine Ausgangsmeldung oder beides definieren. Eingangsmeldungen werden an den Server gesendet, der mit einer Ausgangsmeldung antworten kann. Felder in einem Formularentwurf lassen sich so erstellen, dass sie an einen oder mehrere Webservices angebunden sind. Dabei enthält die WSDL-Datei die Datenbeschreibung, und anhand dieser wird die WSDL-Datenverbindung erstellt.

Die WSDL-Datenverbindung unterscheidet sich deutlich von anderen Datenverbindungstypen. Denn eine Webservice-Operation ähnelt einem Funktionsaufruf mit Eingangs- und/oder Ausgangsparametern. Ein bestimmtes Feld oder bestimmte Felder können die Quelle einer Eingangsmeldung und der Zielort der Ausgangsmeldung sein.

Das Zusammenspiel von Designer ES2 und einer WSDL-Datenverbindung macht die folgenden Aktionen möglich:

- Eine oder mehrere Operationen können innerhalb eines oder mehrerer Webservices gebunden werden.
- Felder, Teilformulare sowie Ausschlussgruppen können an das „click“-Ereignis einer Schaltfläche angehängt werden.
- Eine Webservice-Operation von jedem Ereignis ist über ein Skript ausführbar.
- Der Skriptzugriff auf alle zurückgegebenen Elemente eines Webdienstes wird ermöglicht; unabhängig davon, ob diese Elemente an Felder gebunden sind.
- Die Kommunikation im Stil von SOAP 1.1 (SOAP-Bindungen und ein HTTP/HTTPS-Transport) kann verwendet werden.
- Daten können mit einem Webdienst unter Verwendung des Austauschformats „doc/literal“ ausgetauscht werden.
- Client-basierte Skripten können mit Hilfe des Acrobat SOAP Java Script Objekts, das „RPC/encoded“ unterstützt, geschrieben werden.

Diese Funktionen werden nicht unterstützt:

- Einige XML-Schemafunktionen
- Protokolle wie z. B. SMTP, FTP etc. als Basis-Transportprotokoll für SOAP
- Elemente des Erweiterbarkeitstyps
- Webdienst-Suchen mit Hilfe von UDDI
- Remoteprozeduraufruf (RPC) verschlüsselter SOAP-Meldungen
- Verwendung von HTTP POST- und GET WSDL-Bindungen

Ein WSDL-Dokument macht es möglich, eine Datenverbindung zu einem sicheren Webserver herzustellen. Dieser muss allerdings eine Authentifizierung erfordern (HTTP/HTTPS zur Zugriffssteuerung oder Meldungsebene zur Anforderung eines Webdienstes oder beides).

Die HTTP/HTTPS-Authentifizierung findet auf der Transportebene statt, bei der eine Client-Überprüfung für das Zugreifen auf das WSDL-Dokument und die Verbindung mit einem sicheren Webdienst erforderlich ist. Benutzername und Kennwort, Benutzername und Kennwort-Digest sowie Client-Zertifikat werden im Rahmen der Authentifizierung akzeptiert:

- Die Authentifizierung mit Benutzername und Kennwort bietet eine einfache Zugriffsauthentifizierung. Der Benutzer muss sich über ein Dialogfeld anmelden. Die Kombination der beiden Begriffe wird als unverschlüsselter Text ohne Hashing an den Server gesendet.
- Über Benutzername und Kennwort-Digest ergibt sich eine erweiterte Authentifizierung. Dabei muss sich der Benutzer über ein Dialogfeld anmelden. Benutzername und Kennwort werden mit Hashing (= verschlüsselt) an den Server übertragen.

- Hinweis: Das Anmeldedialogfeld zeigt den Namen des Servers und den Bereich an, mit dem der Benutzer sich verbinden möchte. Da auf einem bestimmten Server mehrere Bereiche eingerichtet sein können, ist diese Information für den Benutzer hilfreich. Er kann so erkennen, welche Berechtigungen für die Anmeldung erforderlich sind.
- Als dritte Authentifizierungsvariante bietet das Client-Zertifikat die Verifizierung über digitale IDs. Dabei werden Anwender im Dialogfeld „Digitale ID wählen“ aufgefordert, in einer Liste der verfügbaren digitalen IDs die korrekte ID zu selektieren. Diese wird dann zur Authentifizierung an den Server übermittelt.
- Hinweis: Um ein Client-Zertifikat verwenden zu können, muss eine digitale ID im Windows-Zertifikatspeicher oder im digitalen ID-Dateispeicher von Designer ES2 verfügbar sein. Diese kann dann bei der Erstellung einer WSDL-Datenverbindung ausgewählt werden. Siehe dazu auch: [Importieren von digitalen IDs](#).

Die Authentifizierung auf Meldungsebene findet auf der SOAP-Meldungsebene (Simple Object Access Protocol) statt. Dabei ist für die Client-Verifizierung zum Zugreifen auf einen sicheren Webdienst ein Sicherheits-Token (Teil des SOAP-Meldungsheaders) notwendig. Die Meldungsebene akzeptiert die Authentifizierung mit Benutzername und Kennwort. Der Benutzer muss sich also über ein Dialogfeld anmelden. Acrobat 9.0 und die neueren Programmversionen unterstützen die Authentifizierung auf Meldungsebene über Benutzername und Kennwort.

Bei der Erstellung einer sicheren WSDL-Datenverbindung kann optional angegeben werden, welche Anmeldeinformationen für die Authentifizierung (HTTP/HTTPS und Meldungsebene) akzeptabel sind. Diese Anmeldeinformationen werden als Teil des HTTP/HTTPS-Protokollheaders gesendet. Anhand dieser Informationen für die Meldungsebene (SOAP) wird ein Sicherheits-Token erstellt, der dann in die SOAP-Meldungskopfzeile eingebettet wird.

Hinweis: Im Hinblick auf einen möglichen Server-Workflow, bei dem der Server Anmeldeinformationen ohne Benutzerinteraktion bereitstellen muss, ist es wichtig anzugeben, welche Art von Anmeldeinformationen akzeptabel sind.

### **Dynamische Formulare mit flexiblem Layout entwickeln und damit arbeiten**

Ein Formular mit flexiblem Layout enthält Teilformulare und andere Elemente, deren Größe sich automatisch an die vom Endanwender eingetragene Datenmenge anpasst. Als interaktive Variante kann ein solches Formular direkt vom Endanwender ausgefüllt werden. Nicht interaktive Formulare erfordern dagegen das Zusammenführen von Daten und Formular in einem serverbasierten Prozess.

Siehe dazu auch: [\\_](#)

Über Masterseiten, Inhaltsbereiche und Teilformulare wird exakt gesteuert, wie LiveCycle Designer Objekte im Formular platziert und an die Anzeige unterschiedlicher Datenmengen anpasst.

### **Unterschiedliche Darstellung im Entwurf und während der Laufzeit**

Wichtig für Formulare mit flexiblem Layout: Ihr Erscheinungsbild zur Entwurfszeit ist nicht identisch mit dem Erscheinungsbild, das der Endanwender sieht. Ein Beispiel verdeutlicht den Unterschied: Der Formularentwurf enthält eine Elementzeile für die Eingabe von Daten. Wird das Formular auf dem Client wiedergegeben, enthält es möglicherweise mehrere Elementzeilen und die Benutzer können evtl. zusätzliche Elementzeilen einfügen. Die Anzahl der zur Entwurfszeit angezeigten Zeilen hängt davon ab, ob die Objekte in der Elementzeile in ein Teilformular eingeschlossen wurden, für das die Optionen „Min-Zähler“ oder „Maximal“ festgelegt wurden. Gelten diese Optionen lässt sich steuern, wie viele Zeilen dem Endanwender zunächst zum Ausfüllen bereitstehen und wie viele Zeilen von ihm hinzugefügt werden können.

Da sich Formulare mit flexiblem Layout automatisch an die Daten anpassen, ist es nicht nötig, die genaue Größe von Objekten oder die Anzahl von Elementzeilen von Anfang an im Formular festzulegen. Über die Wahl von Optionen wie z. B. „Mehrere Zeilen zulassen“, „Seitenumbrüche im Inhalt zulassen“ und „Passend erweitern“ entstehen interaktive Formulare mit flexiblem Layout, die auch unbestimmte Datenmengen ordnungsgemäß wiedergeben.

## Formularentwürfe mit flexiblem Layout erstellen

Formulare, deren Felder sich an die Größe der eingegebenen Datenmenge flexibel anpassen, lassen sich auf zwei grundlegende Arten erstellen:

- Der Formularentwurf beginnt mit einem festen Layout.
- Der Formularentwurf wird von Anfang an mit einem flexiblen Layout erstellt.

Beide Verfahren führen zum gleichen Ergebnis, bieten aber zwei unterschiedliche Entwurfstile an. Entsprechend der persönlichen Arbeitsweise kann jeder Formularentwickler seine bevorzugte Variante wählen.

## Der Formularentwurf beginnt mit einem festen Layout

Für Formularentwickler, die sich in der Formularerstellung mit festem Layout bereits gut auskennen, ist diese Variante denkbar einfach. Das Layout des Formulars wird geplant. Die erforderlichen, für die Endanwender sichtbaren Objekte und Informationen werden angezeigt. Der Formularentwurf mit festem Layout kann mit so vielen Masterseiten und Seiten wie nötig aufgebaut werden. Sobald Objektlayout und Formatierung abgeschlossen sind, werden die Eigenschaften für jene Formularabschnitte festgelegt, die fließenden Inhalt besitzen sollen.

Dieser Ansatz stellt auch die einfachste und übersichtlichste Option dar, um ein vorhandenes Formular mit festem Layout in einen Entwurf mit flexiblem Layout zu konvertieren.

Die folgenden Schritte veranschaulichen die Vorgehensweise nach Abschluss der konzeptionellen Planungsphase:

1. Erforderliche Masterseiten erstellen.
2. Dann auf der Seite den Formularinhalt darstellen. Wird ein Formular mit einem festen Layout konvertiert, einfach die Objekte im Formular in Teilformulare aufnehmen. Steht ein leeres Formular am Anfang, ist der Ablauf wie folgt:
  - Dem Formular Objekte hinzufügen und diese in die entsprechenden Teilformulare einschließen.
  - Erforderliche Formatierung auf die Objekte anwenden.
  - Layout der Objekte in den Teilformularen und Layout der Teilformulare abschließen.
  - Nach Bedarf Skripten hinzufügen.
3. Dynamische Konzepte implementieren:
  - Aufnahme der Standard-Teilformulare auf den einzelnen Seiten aufheben. Die Teilformulare werden zu untergeordneten Elementen des Stammformulars (Formular 1), das einen fließenden Inhalt hat. Der weitere Entwurf erfolgt in flexiblem Layout. Dabei sind die Teilformulare untereinander positioniert, da die Fließrichtung von oben nach unten verläuft.
  - Erforderliche Formatierung auf die Teilformulare anwenden, z. B. Bindungstyp, Ränder etc.
  - Nun den Fluss definieren, Werte für Vorkommen, Seitenumbrüche und die Teilformulare für Kopf- und Fußbereich festlegen. Anschließend definieren, welche Teilformulare zusammengehören, Teilformulare den Masterseiten zuordnen und Ränder festlegen.
4. Nun das Formular mit Musterdaten testen.
5. Hinweis: Dieses Verfahren wird in den Musterformularen angewendet, die im Lieferumfang von Designer ES2 enthalten sind. Um die Verwendung zu optimieren, werden die Teilformulare mit dem tatsächlichen Inhalt in ein Teilformular aufgenommen, dessen Standardbindung auf „Ohne“ eingestellt ist. Werden die Teilformulare auf diese Weise konfiguriert, ergibt sich schnell ein Einblick in die Gesamtstruktur des Formulars, sobald es erstmals in der Hierarchieansicht angezeigt wird.

## Tipps für die Formularerstellung mit festem Layout

Die folgenden Punkte sind beim Entwerfen wichtig:

- Den Inhalt des Formulars nur dann auf „Textfluss“ einstellen, wenn das Layout bereits zufriedenstellend ist. Wird der Inhalt des Formulars auf „Textfluss“ gestellt, kann er auch wieder auf ein positioniertes Layout zurückgesetzt werden. Dabei können sich Probleme mit dem Formularlayout ergeben. Sollen z. B. die Teilformulare in einen einzigen positionierten Container aufgenommen werden, erfolgt die Positionierung der Teilformulare genau an der Stelle, an der sie sich in den entsprechenden Seiten befanden. Das kann zu Überlagerungen führen. Dieses Problem ist vermeidbar, indem die Teilformulare mit der Seitengruppierung umschlossen werden.
- Bei Teilformularen lautet die Standardeinstellung für den minimalen und den maximalen Zählerwert 1. Diese Werte müssen für Formulare mit flexiblem Layout angepasst werden.
- Durch das Aufnehmen von Objekten in ein Teilformular wird der gesamte über die Objekte hinausgehende Platz gelöscht. Deshalb linke und rechte Ränder hinzufügen, um das Teilformular horizontal neu auszurichten. Auch die Größe des Teilformulars ist veränderbar. Dadurch werden die Objekte aber neu positioniert, da sie in Relation zum übergeordneten Element angeordnet sind.

## Der Formularentwurf wird von Anfang an mit einem flexiblen Layout erstellt

Bei flexiblem Layout ist es möglich, den Formularentwurf mit so vielen Masterseiten und Teilformularen wie nötig aufzubauen. Das Konzept der Seiten wird dabei nicht berücksichtigt. Sie werden erst während der Verarbeitung erstellt, wenn der Formularentwurf mit den Daten zusammengeführt wird und die Menge der Daten die aktuelle Seite überschreitet. Die dynamischen Eigenschaften werden angewendet, sobald das Layout und die Formatierung abgeschlossen sind.

Die Formularerstellung mit flexiblem Layout ist häufig der optimale Ansatz, wenn ein XML-Schema am Anfang des Formularaufbaus steht. Erfahrene Anwender können diese Methode aber auch verwenden, um schon beim Formularentwurf das dynamische Verhalten besser einzuschätzen. Es dauert seine Zeit, um mit dieser Methode vertraut zu werden. Das liegt vor allem am unerwarteten Verhalten beim Entwerfen in einem fließenden Container.

Nach der Konzeptionsphase kann der Entwurf in den folgenden Schritten erfolgen:

1. Erforderliche Masterseiten erstellen.
2. Inhalt des Formulars wie folgt generieren.
  - Teilformulare und ihre Inhalte erstellen. Aus dem Standard-Teilformular der Seite wird das erste Teilformular. Die Größe dieses Teilformulars entsprechend seinem Inhalt anpassen.
  - Zusätzliche Seiten sind nicht erforderlich. Nun mit dem Erstellen aller weiteren Teilformulare fortfahren, um den Entwurf abzuschließen. Dabei werden zusätzliche Seiten angelegt, wenn das erstellte Teilformular nicht auf die aktuelle Seite passt. Dies ist nicht immer ersichtlich, da der Vorgang automatisch abläuft und die Palette „Hierarchie“ die zusätzlichen Seiten nicht anzeigt.
  - Erforderliche Formatierung auf die Objekte anwenden.
  - Das Layout der Objekte in ihren Teilformularen sowie die Anordnung der Teilformulare abschließen.
  - Nach Bedarf Skripten hinzufügen.
3. Die dynamischen Konzepte implementieren.
  - Erforderliche Formatierung auf die Teilformulare anwenden, z. B. Bindungstyp, Ränder etc.
  - Nun den Fluss definieren, indem die Werte für Vorkommen festgelegt und Seitenumbrüche zugelassen werden sowie Teilformulare für Kopf- und Fußbereiche definiert werden. Außerdem bestimmen, welche Teilformulare zusammengehören, die Teilformulare den Masterseiten zuordnen und Ränder festlegen.
4. Abschließend das Formular mit Musterdaten testen.

## Fazit

Die Adobe Formulartechnologie stellt eine leistungsfähige Formular-Autorenumgebung zur Verfügung, die sämtliche Entwicklungs- und Arbeitsprozesse rund um das Gestalten, Erstellen, Aktualisieren und Verteilen elektronischer Formulare auf einen neuen Standard hebt. Adobe macht es möglich, interaktive PDF-Formulare mit festem Layout zu erstellen. Darüber hinaus können interaktive PDF-Formulare auch dynamische Eigenschaften besitzen.

In Verbindung mit der Adobe LiveCycle Enterprise Suite erschließen sie hocheffiziente Anbindungsmöglichkeiten an bestehende Unternehmenslösungen zur direkten elektronische Nutzung der Endanwenderdaten.

Mit Adobe LiveCycle Designer erstellte Formulare können direkt verwendet werden. Die Integration mit Geschäftsprozessen kann sofort oder zu jedem beliebigen Zeitpunkt durch Einsatz der Enterprise Suite erfolgen. Somit bietet sich eine höchstmögliche Flexibilität in Hinblick auf Effizienz und Effektivität der Gesamtlösung.

Weitere Informationen zu Adobe-  
Lösungen für Unternehmen unter  
<http://www.adobe-solutions.de>



**Adobe**

Adobe Systems GmbH  
Georg-Brauchle-Ring 58  
80992 München  
Deutschland  
[www.adobe.de](http://www.adobe.de), [www.adobe.at](http://www.adobe.at), [www.adobe.ch](http://www.adobe.ch)

Adobe, das Adobe Logo, Flash, LiveCycle und Adobe Reader sind eingetragene Marken oder Marken von Adobe Systems Incorporated in den USA und/oder anderen Ländern. Alle anderen Marken sind Eigentum der jeweiligen Inhaber.

© 2010 Adobe Systems Incorporated. Alle Rechte vorbehalten.  
Bildnachweis: Adobe Systems