FAKULTÄT FÜR !NFORMATIK

enablerWorkflows

Ein formularorientiertes, dynamisches, web-basiertes Workflow-Management-System auf Basis der Windows Workflow Foundation

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Informatik

eingereicht von

Matthias Wohlmann
Matrikelnummer 9425261

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung:
Betreuer: Univ.Prof. Dipl.-Ing. Dr. Werner Purgathofer

Wien, 22.10.2008

(Unterschrift Verfasser) (Unterschrift Betreuer)

Vorwort

Diese Arbeit beschreibt das Workflow-Management-System "*enablerWorkflows*". Dieses dient zur Erstellung von Workflows auf Basis der Microsoft Windows Workflow Foundation, deren Endprodukte aus einzelnen Teilen zusammengesetzte Workflow-Dokumente sind. Die Idee dazu entstand im Laufe meiner Tätigkeit als freier Mitarbeiter der Firma Logic4BIZ. Hier habe ich bereits vor drei Jahren mangels passender kommerzieller Alternativen am Markt eine eigene Workflow-Runtime in das Softwareprodukt „enabler4BIZ" (eine Webapplikation basierend auf dem Microsoft .Net Framework) integriert. Hauptzweck war damals die Abbildung von Geschäftsfällen zur Dokumentenfreigabe.

Diese Workflow-Runtime war recht einfach gehalten und unterstützte nur sequenzielle Arbeitsabläufe ohne Verzweigungen und Schleifen und war außerdem schwer an neue Anforderungen anzupassen. Im Laufe der Zeit kamen immer wieder Wünsche von Kunden, deren Arbeitsabläufe mit Hilfe dieses Workflows abzubilden. Die Abbildung in der bestehenden Workflow-Runtime war jedoch meist nicht beziehungsweise nur mit sehr hohem Aufwand möglich. In Folge konnten wir (Logic4BIZ) einen dieser Kundenwünsche als kundenspezifische Implementierung, ebenfalls von mir durchgeführt, abbilden. Bei den anderen Wünschen war der geschätzte Aufwand für eine solche Implementierung für den Kunden zu hoch, ein Auftrag kam in Folge nicht zu Stande.

Als Microsoft Ende des Jahres 2006 die neue Version 3.0 des .Net Frameworks mit integrierten Workflow-Funktionalitäten (Microsoft Windows Workflow Foundation) veröffentlichte, kam bei mir die Idee auf, dieses als Basis für ein Produkt zu verwenden, mit dem es möglich sein sollte, bisher abgelehnte Kundenwünsche mit annehmbarem Aufwand abzubilden.

Da mich auf Grund meiner bisherigen Tätigkeit bei Logic4BIZ diese Materie sehr interessiert und ich außerdem schon seit Längerem ein Thema für meine Diplomarbeit gesucht habe, entschloss ich mich im Einverständnis mit der Firma Logic4BIZ dieses Produkt namens "enablerWorkflows" zu erstellen und im Rahmen einer Diplomarbeit zu beschreiben.

An dieser Stelle möchte ich mich bedanken...

... bei Johann Sauermann, Franz-Peter Walder und Christian Rehnelt von Logic4BIZ für die Ermöglichung der Bearbeitung dieses Themas im Rahmen meiner Tätigkeit in dieser Firma.

... bei meinem Arbeitskollegen Alexander Winkelmeyer für die technischen Diskussionen zur Umsetzung und als Ideenlieferant.

... bei meiner Familie und meinen Freunden für ihre Geduld im Zusammenhang mit der Entstehung dieser Arbeit.

... bei meiner besten Freundin, Lebensgefährtin und seit kurzem auch Ehepartnerin, Lilli für ihre häufigen Motivierungsversuche und für das Korrekturlesen dieser Arbeit.

Inhaltsverzeichnis

1 Einleitung

1.1 Problemdefinition

Klassische EDV-Systeme in der Geschäftswelt stellen meist nur einfache Ein-Ausgabe-Masken für die Tabellen einer dahinter liegenden Datenbank zur Verfügung. Dabei ist eine Maske meist nur genau für das Anlegen und Ansehen eines Datensatzes in einer Tabelle zuständig. Neue Information wird hier normalerweise von genau einem Benutzer zu einem singulären Zeitpunkt eingegeben und ist danach für alle anderen Benutzer im System abrufbar.

Dies entspricht jedoch nur selten den tatsächlichen Anforderungen der Informationserstellung, -verwaltung und –verteilung in einem Geschäftsbetrieb. In der Realität muss Information oft

- an verschiedenen Stellen im Betrieb gesammelt und verdichtet werden,
- von einer oder mehreren Personen geprüft und freigegeben werden,
- an mehrere Personen oder Personengruppen proaktiv verteilt werden und
- revisionssicher und wiederauffindbar abgelegt werden.

Die Datenhaltung selbst, wie sie in klassischen EDV-Systemen zur Verfügung gestellt wird, deckt hier nur den letzten Punkt der Anforderungen ab. Um den anderen oben genannten Anforderungen gerecht zu werden, müssen im System neben der Datenhaltung auch Arbeits-, Informations- und Entscheidungsfälle (Geschäftsfälle) abgebildet werden können. Diese Abbildung von Geschäftsfällen in Software wird als *Workflow* bezeichnet (Genauere Definitionen sind in Kapitel 2 zu finden).

Oft werden Workflows, die Geschäftsfälle begleitend unterstützen, in Software-Projekten je Geschäftsfall spezifisch gelöst. Allerdings ist diese Art der Abbildung in einem EDV-System eine eher komplexe, arbeitsintensive und fehleranfällige Aufgabe. Die Kosten für die Implementierung eines solchen Systems

für einen speziellen Anwendungsfall sind meist höher als der daraus gewonnene Nutzen. In vielen Fällen wird daher nach der Analyse und der Aufwandsabschätzung auf eine Implementierung auf Grund der zu hohen Kosten und des zu intensiven Zeitaufwands verzichtet.

Ziel dieser Arbeit ist es, ein Software-Tool zu erstellen, mit dessen Hilfe es möglich ist, die oben beschriebenen Anforderungen und Arbeitsabläufe in vergleichsweise kurzer Zeit in einem EDV-System abbilden zu können.

Dazu müssen gemeinsame Muster in Geschäftsfällen identifiziert, herausgearbeitet und spezifiziert, und diese als Bausteine (=*Aktivitäten*, siehe 2) für einen Workflow dem Workflow-Designer zur Verfügung gestellt werden. Die Entwicklung einer Software-Anwendung auf dieser Basis im Vergleich zur kundenspezifischen Implementierung kann dadurch einfacher, effektiver, schneller und somit kostengünstiger abgewickelt werden. Weiters soll es einem Software-Entwickler möglich sein, spezielle Teile von Kundenanforderungen, die nicht durch einen der zur Verfügung gestellten Bausteine darstellbar sind, ebenfalls abbilden zu können.

Als Lösungsvorschlag wird eine Erweiterung des web-basierten Software-Produkts „enabler4BIZ“ mit dem Namen „enablerWorkflows“ beschrieben. Diese Erweiterung baut auf der Windows Workflow Foundation auf und stellt dem Entwickler die nötigen Workflow-Bausteine und Komponenten zur Verfügung. Weiters ist eine einfache, allgemeine, web-basierte GUI enthalten, die es dem Endbenutzer erlaubt, Workflows zu starten, mit laufenden Workflows zu interagieren (Dateneingabe, Prüfungen,...) und beendete Workflows beziehungsweise deren Ergebnis, das erstellte *Workflow-Dokument*, zu betrachten.

1.2 Beispiele aus der Praxis

Im Folgenden werden vier Geschäftsfälle aus der Praxis kurz beschrieben, die als Vorlage zur Extraktion der Muster und Anforderungen gedient haben, und die in Folge mit enablerWorkflows abbildbar sein sollen. Es werden jeweils die grundlegende abzubildende Aufgabe sowie (grob) die einzelnen Schritte der jeweiligen Geschäftsfälle dargestellt. Diese Beispiele dienen nachfolgend

in dieser Arbeit immer wieder als Basis zur praktischen Veranschaulichung von theoretischen und technischen Konzepten.

Urlaubsansuchen: Hier geht es darum, Urlaubsansuchen von Mitarbeitern auf elektronischem Wege abzubilden und zu dokumentieren. Der Geschäftsfall besteht aus folgenden Schritten:

1. Stellen eines Urlaubsansuchens an den Vorgesetzten (inklusive Zeitraum und eventueller Begründung)
2. Genehmigung oder Ablehnung durch den Vorgesetzten
3. Information an den Antragsteller

Endprodukt: Ein genehmigtes oder abgelehntes Urlaubsansuchen.

Konzernweites, innerbetriebliches Vorschlagswesen: Dieses Beispiel soll den Mitarbeitern eines Großkonzerns die Möglichkeit bieten, Vorschläge im Unternehmen einzubringen, die zu einer Verbesserung in einem oder mehreren Bereichen führen (z.B. Kundenzufriedenheit, Mitarbeitermotivation, Kostenersparnis, ...). Die Vorschläge werden bewertet und gegebenenfalls umgesetzt. Bei Erfolg (Evaluierung der Umsetzung) erhält der Mitarbeiter, der den Vorschlag eingebracht hat, eine Prämie. Der Geschäftsfall besteht vereinfacht aus folgenden Schritten:

1. Einreichung eines Vorschlags durch einen Mitarbeiter
2. Vorschlag prüfen, wenn unklar oder kein Vorschlag retour zu 1
3. Zuteilung an zuständige Fachabteilung
4. Annahme durch Fachabteilung oder retour zu 3, falls Abteilung nicht zuständig
5. Gutachten erstellen

6. Wenn weiteres Gutachten nötig: Zuteilung zu weiterer Fachabteilung und weiter bei 5
7. Ansonsten: Gutachten zusammenfassen und Entscheidungsgrundlagen aufbereiten
8. Entscheidung über Umsetzung treffen
9. Falls Umsetzung abgelehnt, Information an Einreicher des Vorschlags, ansonsten
10. Festlegen der Prämie
11. Mitteilung und Auszahlung der Prämie an den Einreicher
12. Umsetzung des Vorschlags (= eigener Workflow)

Endprodukt: Dokumentation eines eingebrachten und bearbeiteten Vorschlags

Produktpass: Dieses Beispiel kommt aus dem Bereich der Nahrungsmittelproduktion. Für jedes hergestellte Produkt sollen Inhaltsstoffe und Nährwertangaben gesammelt und in einem Dokument dem Einzelhandel zur Verfügung gestellt werden. Der Workflow besteht aus folgenden Schritten:

1. Produktpass inklusive Name und Beschreibung eines neuen Produkts erstellen
2. Eintragen der Inhaltsstoffe durch die verschiedenen beteiligten Abteilungen (eventuell parallel)
3. Nährwertangaben im Produktpass ergänzen
4. Freigabe des Produktpasses
5. Weitergabe des Produktpasses an die Verkaufsabteilung beziehungsweise, in gefilterter Form, an die jeweiligen Kunden (Einzelhandel)

Endprodukt: Ein Produktpass für jedes hergestellte Produkt

Reklamationsmanagement: Dieses Beispiel wird zur Evaluierung der hier vorgestellten enablerWorkflows herangezogen und wird daher im Kapitel 6 noch detailliert beschrieben. Grob geht es um die Annahme und Abwicklung von kundenseitigen Reklamationen und deren Dokumentation im System. Endprodukt ist die vollständige Dokumentation einer kundenseitigen Beschwerde, von der Annahme über die Bearbeitung bis zur Rückmeldung an den Kunden.

1.3 Organisation der Arbeit

Im nächsten Kapitel (Kapitel 2) werden Begriffe und Konzepte im Bereich der Workflow-Management-Systeme erklärt, die in dieser Arbeit verwendet werden.

In Kapitel 3 wird die Entstehungsgeschichte von Workflow-Management-Systemen bis zum heutigen Entwicklungsstand beschrieben. Weiters werden die Basisprodukte, auf denen enablerWorkflows aufbaut, vorgestellt. Dies sind die "Microsoft Windows Workflow Foundation" und das Softwareprodukt "enabler4BIZ", inklusive der darin enthaltenen Komponente "Dynamische Formulare".

Kapitel 4 stellt das Systemkonzept vor. Basiskonzepte und die zum Bau von Workflows zur Verfügung gestellten Aktivitäten werden erklärt. Weiters wird das Laufzeitsystem inklusive Userinterface beschrieben.

Das Kapitel 5 beschreibt allgemein die Vorgehensweise bei der Implementierung eines Geschäftsprozesses mit enablerWorkflows.

In Kapitel 6 wird als Evaluierung die Prototyp-Implementierung eines Tools zur Verfolgung und Abwicklung von Kundenreklamationen mit Hilfe von enablerWorkflows vorgestellt.

Kapitel 7 fasst die Arbeit zusammen und bietet einen Ausblick auf Erweiterungsmöglichkeiten.

2 Begriffe und Grundlagen

In folgendem Kapitel sollen die in dieser Arbeit verwendeten und für die Beschreibung eines Workflow-Management-Systems notwendigen Begriffe definiert werden. Weiters sollen die die Zusammenhänge dieser Begriffe erklärt und veranschaulicht werden.

2.1 Beziehungen zwischen den Basisbegriffen

Die Grafik in Abbildung 1 zeigt die Zusammenhänge der verwendeten Begriffe auf.

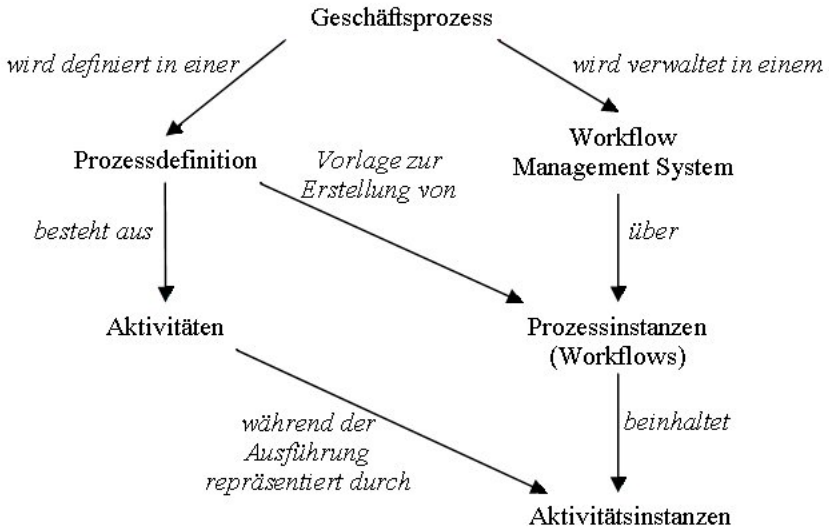

Abbildung 1: Beziehungen zwischen den Basisbegriffen [Coa99]

2.2 Geschäftsprozess

Ein Geschäftsprozess besteht aus einer Menge von verknüpften Prozeduren und Aktivitäten, die gemeinsam ein betriebswirtschaftliches Ziel realisieren.

Dies geschieht normalerweise im Kontext einer Organisationsstruktur, die Rollen und Beziehungen definiert (vgl. [Coa99, Ste96]). Verschiedenste Geschäftsprozesse sind also in jedem Unternehmen implizit vorhanden und können durch Prozessdefinitionen formal beschrieben werden.

2.3 Prozessdefinition (Workflow-Definition)

Unter einer Prozessdefinition versteht man die Repräsentation eines Geschäftsprozesses in einer Form, die automatisierte Änderungen (z.B. Modellierung) oder die Ausführung durch ein Workflow-Management-System zulässt. Die Prozessdefinition besteht aus einem Netz von Aktivitäten und Beziehungen, Kriterien, die den Start und das Ende des Prozesses definieren sowie Informationen über die einzelnen Aktivitäten, wie zum Beispiel Beteiligte, verknüpfte Applikationen und Daten (vgl. [Coa99]). Der Begriff "Workflow-Definition" kann synonym verwendet werden.

In dieser Arbeit wird der Begriff "Workflow-Definition" für Prozessdefinitionen verwendet, die mit dem grafischen Designer erstellt wurden und als Assembly verfügbar sind, die aber noch nicht im Laufzeitsystem eingebunden und dort konfiguriert wurden.

2.4 Workflow-Vorlage

Der Begriff "Workflow-Vorlage" wird in der Literatur ebenfalls als Synonym für Prozessdefinition verwendet. In dieser Arbeit soll der Begriff für Workflow-Definitionen verwendet werden, die im Laufzeitsystem eingebunden und mit Ressourcen versehen wurden und somit zur Erstellung und zum Start von Workflows den Benutzern zur Verfügung stehen.

2.5 Aktivität

Eine Aktivität ist die Beschreibung eines Arbeitsvorgangs, der einen logischen Schritt innerhalb eines Prozesses darstellt (vgl. [Coa99]). Eine Aktivität benötigt menschliche oder maschinelle Ressourcen zur Ausführung. Aktivitäten

können atomar oder zusammengesetzt sein (vgl. [DRSV03]). Zusammengesetzte Aktivitäten beinhalten selbst wieder eine oder mehrere Aktivitäten und gruppieren diese beziehungsweise definieren dadurch eine hierarchische Struktur.

2.6 Prozessinstanz (Workflow)

Eine Prozessinstanz repräsentiert eine einzelne Instanzierung eines Prozesses und beinhaltet die zugehörigen Daten. Jede Instanz repräsentiert einen separaten Ausführungs-Thread des Prozesses, der seinen eigenen internen Status und eine nach außen sichtbare Identität besitzt (vgl. [Coa99]). In dieser Arbeit werden die Begriffe "Workflow-Instanz oder "Workflow" als Synonym für eine laufende oder bereits beendete Prozessinstanz verwendet.

Ein Instanz kann nach [Hol95] folgende Zustände annehmen:

- *Initialisiert*: Eine Prozessinstanz wurde initialisiert, inklusive der dazugehörigen relevanten Workflowdaten, der Prozess wurde aber noch nicht gestartet.
- *Laufend*: Der Workflow wurde gestartet, und jede seiner Aktivitäten kann gestartet werden (sobald ihre Startbedingung erfüllt ist).
- *Aktiv*: Eine oder mehrere Aktivitäten des Workflows wurden gestartet.
- *Suspendiert*: Der Workflow ist untätig, keine Aktivitäten werden gestartet, bis der Workflow wieder fortgesetzt wird.
- *Beendet*: Der Workflow hat seine Endbedingung erfüllt, die Instanz kann gelöscht werden.
- *Abgebrochen*: Die Ausführung des Workflows wurde vor seiner normalen Beendigung gestoppt, die Instanz kann gelöscht werden.

2.7 Aktivitätsinstanz

Eine Aktivitätsinstanz repräsentiert die Instanzierung einer Aktivität innerhalb einer Prozessinstanz (vgl. [Coa99]). Eine Aktivitätsinstanz kann dabei, ähnlich einer Prozessinstanz, folgende Zustände annehmen (vgl. [Hol95]):

- *Inaktiv*: Die Aktivität wurde innerhalb der Prozessinstanz erstellt, aber noch nicht aktiviert, da die Startbedingung noch nicht erfüllt wurde.
- *Aktiv*: Die Aktivität wird ausgeführt. Gegebenenfalls wird eine Aufgabe erstellt und verarbeitet.
- *Suspendiert*: Die Aktivitätsinstanz ist inaktiv.
- *Beendet*: Die Ausführung der Aktivität ist abgeschlossen; Die Aufgabe wurde erledigt.

2.8 Aufgabe

Eine Aufgabe repräsentiert die Arbeit, die von einem Workflow-Teilnehmer im Kontext einer Aktivität erledigt werden muss (vgl [Coa99]). Aktivitäten, die menschliche Ressourcen benötigen, erstellen Aufgaben. Mehrere Aufgaben aus unterschiedlichen Workflows werden dem Benutzer in Form von *Aufgabenlisten* zur Verfügung gestellt.

2.9 Ressource

Im Kontext einer Aktivität können menschliche Ressourcen (Benutzer) oder maschinelle Ressourcen (Programme) benötigt werden. Diesen werden Aufgaben zugeteilt, wie zum Beispiel das Ausfüllen eines Formulars (Benutzer) oder das Versenden von E-Mails (Programm).

2.10 Workflow-Management-System

Ein Workflow-Management-System (WfMS) dient der aktiven Steuerung arbeitsteiliger Prozesse und ist ein Software-System, das die Modellierung, Erstellung und Ausführung von Workflows unterstützt. Es interpretiert Prozessdefinitionen, interagiert mit den Workflow-Beteiligten und ruft benötigte Programme auf (vgl. [Coa99, DRSV03, Wik08]. Ein WfMS besteht gewöhnlich aus folgenden Software Komponenten (siehe Abbildung 2):

- (Grafischer) Workflow-Designer: Zur Erstellung von Prozessdefinitionen
- Laufzeitsystem: Zum Konfigurieren, Bereitstellen und Starten von Workflows.
- Workflow-Engine: Zum Ausführen von Workflows.
- Arbeitslistenverwaltung und Userinterface: Verwaltet die Arbeitslisten und stellt sie den Benutzern zur Verfügung.
- Administrations- und Kontrollfunktionen: Zur Überwachung von Workflows

2.11 Laufzeitsystem

Das Laufzeitsystem ist die zentrale Softwareapplikation des Workflow-Management-Systems. In einem Workflow-Manager können Workflows auf Basis von Prozessdefinitionen konfiguriert und den Benutzern zum Starten zur Verfügung gestellt werden. Das Laufzeitsystem verwaltet weiters die Arbeitslisten und ruft benötigte Software-Tools auf. Zur Ausführung der Workflows selbst enthält das Laufzeitsystem eine oder mehrere Workflow-Engines. (vgl. [Hol95, SK97]).

2.12 Workflow-Engine

Als Ausführungsumgebung für Workflows kann die Workflow-Engine als State-Transition-Maschine gesehen werden, die die Prozessdefinitionen interpretiert und die Instanzierung sowie die sequenzielle Abarbeitung der Aktivitäten kontrolliert (vgl. [Hol95, Coa99]).

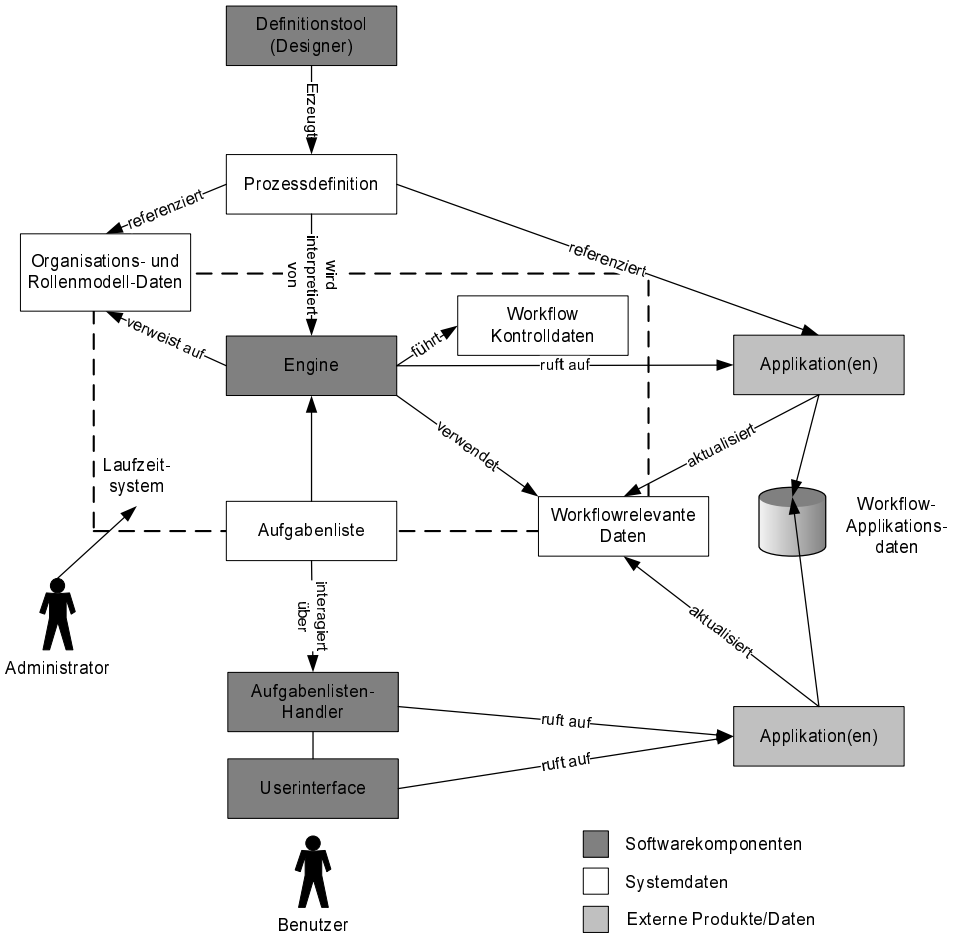

Abbildung 2: Struktur eines Workflow-Management-Systems nach [Hol95]

3 Stand der Technik

In diesem Kapitel wird zuerst die Entstehungsgeschichte von Workflow-Management-Systemen bis zum heutigen Entwicklungsstand beschrieben und danach die beiden Basisprodukte der enablerWorkflows, "Windows Workflow Foundation" und "enabler4BIZ", vorgestellt.

3.1 Workflow-Management-Systeme - Historische Entstehung und Einordnung

Um die Entstehung von Workflow-Management-Systemen zu verstehen, sollte man die Entwicklung von Informationssystemen im historischen Kontext der letzten Jahrzehnte betrachten. Heutige Informationssysteme bestehen laut [Aal04] aus bis zu vier Schichten (siehe Abbildung 3). Die Basis bildet das Betriebssystem. Allgemeine Applikationen sind Programme, die in vielen Bereichen eingesetzt werden können, wie zum Beispiel Datenbanken oder Texteditoren. Bereichsspezifische Applikationen können nur in bestimmten Geschäftsfeldern eingesetzt werden. Beispiele dafür sind Routenplaner oder Call Center Software. Die vierte Schicht stellt Programme dar, die für genau eine Organisation entwickelt wurden.

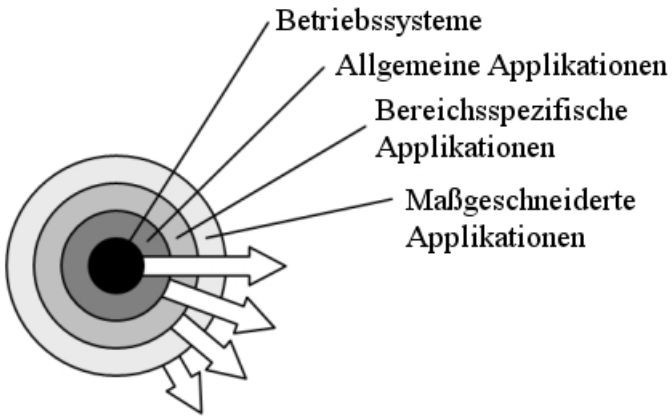

Abbildung 3: Schichtmodell von Informationssystemen aus [Aal04]

In den sechziger Jahren gab es die Schichten zwei und drei nicht, kundenspezifische Programme wurden direkt auf Basis des Betriebssystems implementiert. Anschließend haben sich die Schichten zwei und drei entwickelt. Der aktuelle Trend ist, dass sich die Schichten immer weiter verbreitern, das heißt, dass sie Funktionalitäten aus den darüberliegenden Schichten übernehmen. Zum Beispiel enthalten Betriebssysteme mittlerweile selbst Texteditoren, Datenbank-Management-Systeme haben Bereiche aus der vierten Schicht übernommen. Die Verfügbarkeit immer komplexerer Software führt zu dem Trend, Software nicht mehr neu zu schreiben sondern aus bestehenden Applikationen und Funktionen zusammen zu setzen.

Ein weiterer Trend laut [Aal04] ist die Verlagerung von datenorientierten Systemen hin zu prozessorientierten Systemen: Datenmodellierung war der Ausgangspunkt beim Design von daten-orientierter Software, bestehende Geschäftsprozesse mussten daran angepasst werden. Managementtrends haben aber in weiterer Folge dazu geführt, dass die Abbildung von real existierenden Geschäftsprozessen als Basis neuer Applikationen herangezogen wurden. Mangels Alternativen wurden Programme, die Geschäftsfälle abbilden, in der vierten Schicht als maßgeschneiderte Software implementiert, der Geschäftsfall wurde somit "hard-coded" abgebildet.

Diese Trends haben zur Entwicklung von Workflow-Management-Systemen geführt, die als allgemeine Applikationen in der zweiten Schicht des Schichtmodels nach [Aal04] anzusiedeln sind, aber durch die Möglichkeit, verschiedenste Prozesse abbilden zu können, Bereiche aus der vierten Schicht abdecken können. Workflow-Management-Systeme führen also dazu, dass Geschäftsprozesse nicht mehr in maßgeschneiderter Software implementiert werden müssen, sondern unter Verwendung bestehender Softwarekomponenen abgebildet werden können.

Die ersten Office Information Systems als Vorläufer von Workflow-Management-Systemen entstanden bereits in den siebziger Jahren (vgl [Aal04]), diese waren jedoch wenig erfolgreich. Grund dafür waren die fehlende Basistechnologie, wie langsame oder fehlende Netzwerke und ungenügende graphische Möglichkeiten.

Folgend wurden in den achziger Jahren kaum Fortschritte in diesem Bereich

gemacht, erst seit den neunziger Jahren wurde das Interesse an solchen Systemen wieder größer, und es entstand und entstehen nach und nach eine große Zahl an Workflow-Management-Systemen. Die technischen Probleme der siebziger Jahre sind gelöst, jedoch bestehen weiterhin diverse konzeptionelle Probleme, und bestehende Workflow-Management-Systeme geben unnötige Beschränkungen vor, sodass die Entwicklung in diesem Bereich noch nicht abgeschlossen ist.

Heutige Workflow-Management-Systeme können nach [Aal04] nach ihrem Fokus (datengesteuert, prozessgesteuert oder beides) und nach ihrem Strukturierungsgrad wie in Abbildung 4 dargestellt, eingeteilt werden.

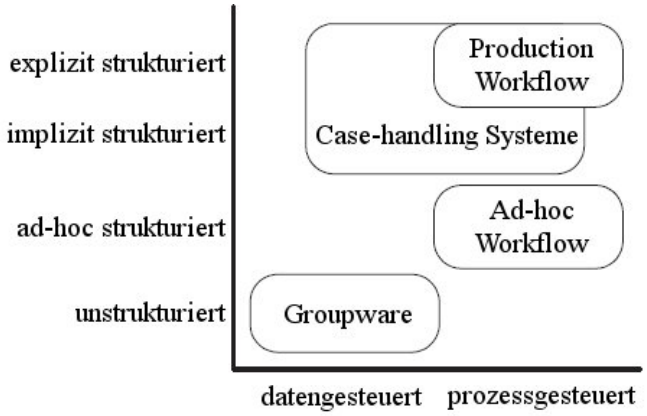

Abbildung 4: Klassifizierung von Systemen zur Unterstützung von Arbeitsprozessen nach [Aal04]

Auf der einen Seite der Einteilung befinden sich Groupware Produkte. Diese sind datengesteuert, ihr Fokus liegt also weniger auf dem zugrundeliegenden Prozess, als darauf, Daten und Informationen zu teilen. Groupware Produkte unterstützen nur unstrukturierte Prozesse.

Auf der anderen Seite stehen Systeme zur Unterstützung von Production-Workflows. Diese besitzen immer eine zugrundeliegende Prozessdefinition, in der alle möglichen Pfade bereits zum Design-Zeitpunkt definiert sind. Ist ein Pfad zum Design-Zeitpunkt nicht definiert, ist er zur Laufzeit auch nicht möglich.

Zwischen diesen beiden sind Ad-hoc-Workflows und Case-handling Systeme anzusiedeln. Ad-hoc-Workflows erlauben die Änderung des Workflow-Prozesses zur Laufzeit durch den Benutzer. Jeder Workflow besitzt somit sein eigenes Prozessmodell.

Im Gegensatz dazu wird bei Case-handling Systemen vom Benutzer nicht erwartet, das Modell zu ändern. Hier wird auf Grund der Daten entschieden, welche Aktivitäten als nächstes aktiviert werden. Außerdem sind für die Benutzer immer alle Daten sichtbar (und nicht nur ein Ausschnitt davon), und diese können jederzeit, auch nach der Ausführung der damit verbundenen Aktivitäten, geändert werden.

Das in dieser Arbeit vorgestellte Workflow-Management-System "enabler-Workflows" unterstützt laut dieser Einteilung ausschließlich Production-Workflows mit fix vordefinierten Prozessmodellen. Das Modell kann zur Laufzeit nicht mehr geändert werden.

3.2 Windows Workflow Foundation

"Microsoft Windows Workflow Foundation (WF) is a general-purpose programming framework for creating reactive programs that act in response to stimulus from external entities. The basic characteristic of reactive programs is that they pause during their execution, for unknown amounts of time, awaiting input." [SS07]

Die Windows Workflow Foundation kann also als Basis für Workflow-orientierte Anwendungen verwendet werden. Sie ist ein Teil des .NET Frameworks 3.0 und damit eine zusätzliche Klassenbibliothek zur .NET Common Language Runtime und Base-Class-Library. Im Gegensatz zu BizTalk Server oder anderen Lösungen ist WF kein Service oder ein eigener Server, sondern "nur" eine Klassenbibliothek, die aus unterschiedlichen Anwendungen ("Workflow-Hosts") verwendet werden kann [MSD08].

In der Windows Workflow Foundation sind bereits einige Basisaktivitäten enthalten, die erweitert werden können. Zusätzlich können auch neue Aktivitäten implementiert werden.

Die WF stellt im Bezug auf enablerWorkflows die „Sprache“ zur Verfügung, in der Prozessdefinitionen erstellt und abgebildet werden. Es enthält die Klassen zur Implementierung einer Workflow-Runtime und beinhaltet auch Komponenten zur Persistierung und zum Tracking von Workflows.

3.3 Management-System enabler4BIZ

Der enabler4BIZ ist ein web-basiertes Management-System, das auf dem *IBuySpy Portal*[1] von Microsoft aufbaut . Das *IBuySpy Portal* ist eine Referenzimplementierung einer Portalsoftware auf ASP.Net Basis. Der enabler4BIZ bietet unter Anderem folgende Basisfunktionalitäten:

- Benutzer- und Rollenverwaltung
- Benutzer-Authentifizierung und Assoziierung mit Rollen
- Methoden zur Überprüfung von Autorisierungen von Benutzern
- Mehrsprachenfähigkeit
- Menüführung und Menüverwaltung: Dies beinhaltet sowohl Platzierung von neuen Seiten innerhalb eines Menü-Baumes inklusive ansprechender Menüführung, als auch Prüfung der Zugriffsberechtigungen von Benutzern auf diese Seiten.
- SMTP-Connector zum Versenden E-Mails
- Datenbankzugriff: Der enabler4BIZ stellt Schnittstellen zum Absetzen einzelner SQL-Statements sowie mehrerer SQL-Statements innerhalb einer Transaktion auf Datenbanken zur Verfügung.

Weiters bietet der enabler4BIZ auch die Möglichkeit, eigene Module zu entwickeln und diese einzubinden, wodurch die Funktionalität erweitert werden kann (vgl. [Win04]). Das Workflow-Management-System “enablerWorkflows” soll auf diese Weise in den enabler4BIZ integriert werden.

[1] *http://msdn.microsoft.com/en-us/library/ms978480.aspx (letzter Zugriff Jul. 2008)*

3.3.1 Dynamische Formulare

Der Baustein "Dynamische Formulare" aus dem enabler4BIZ soll hier gesondert beschrieben werden, da es sich um einen zentralen Basisbaustein für enablerWorkflows handelt. Dabei handelt es sich um ein Framework zur Erstellung und Verwaltung von Formularen sowie von Daten, die von Benutzern in diesen Formularen eingegeben werden. Das Framework besteht aus folgenden Komponenten:

- Eine große Auswahl an Formularelementen, aus denen ein Formular zusammengesetzt werden kann. Folgende Elemente sind beispielsweise enthalten:
  - Label
  - Textbox (ein- und mehrzeilig)
  - Checkbox, Checkboxliste
  - Radiobutton, Radiobuttonliste
  - Combobox
  - Datumsauswahl-Feld
  - Datei Upload-Element
  - HTML-Element
  - User-Combobox: Listet alle enabler4BIZ Benutzer und ermöglicht die Auswahl eines Benutzers
  - FillingUser: Dies ist ein Label, das automatisch mit dem Namen des ausfüllenden Benutzers befüllt wird.
- Ein Formular-Editor: Er bietet eine grafische Oberfläche zum Erstellen eines Formulars aus Formularelementen. Einzelne Formularelemente können dabei aus einer Toolbox gewählt und in einem Grid platziert werden. Für jedes Formularelement können Eigenschaften wie beispielsweise Name, Tooltip und CSS-Klasse festgelegt werden. Vom Formular-Editor wird sowohl eine Definition des Formulars in der Datenbank

gespeichert, als auch pro Formulardefinition dynamisch eine Tabelle in der Datenbank angelegt, die die ausgefüllten Werte des Formulars aufnehmen kann. Dabei beinhaltet ein Datensatz in dieser Tabelle ein ausgefülltes Formular (siehe Abbildung 5).

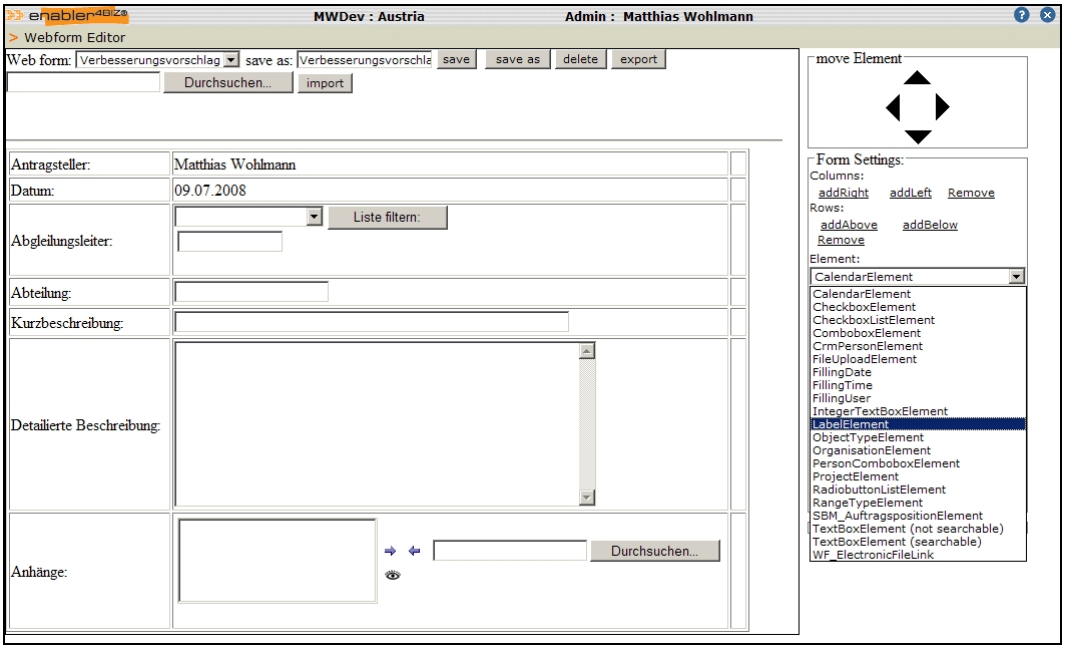

Abbildung 5: Screenshot des enabler4BIZ Formular-Editors mit aufgeklappter Toolbox

- Ein Control, das auf Webseiten platziert werden kann und einem Benutzer ein Formular in den Modi „Fill-Out“ oder „View“ anzeigt. Im „Fill-Out“-Modus kann der Benutzer neue Daten im Formular eingeben. Diese Daten werden in der für diese Formulardefinition erstellten Tabelle abgelegt.

4 Design und Implementierung

Dieses Kapitel unterteilt sich in mehrere Teile. Kapitel 4.1 befasst sich mit allgemeinen Konzepten, die sich durch das Design des gesamten Workflow-Management-Systems ziehen. Die folgenden Kapitel beschreiben die beiden großen Komponenten des Workflow-Management-Systems "enablerWorkflows": Aktivitäten, Laufzeitsystem und Userinterface: In den Kapiteln 4.2 bis 4.4 werden die Aktivitäten vorgestellt, die von enablerWorkflows, zusätzlich zu den in der Microsoft Workflow Foundation inkludierten Aktivitäten, zur Verfügung gestellt werden. Kapitel 4.5 befasst sich mit der Workflow-Engine. Kapitel 4.6 beschreibt das Userinterface des Laufzeitsystems, das in die Basis-Software "enabler4BIZ" für die Umsetzung von enablerWorkflows integriert wurde. In Kapitel 4.7 wird beschrieben, wo die Parameter der Aktivitäten gesetzt werden können und wie diese zur Laufzeit an die Aktivitäten übergeben werden. Kapitel 4.8 beschreibt das Deployment neuer Workflow-Definitionen in den enabler4BIZ. In Kapitel 4.9 wird beschrieben, welche Änderungsanforderungen auftreten können und wie im Rahmen von enablerWorkflows damit umgegangen wird. Im letzten Kapitel 4.10 wird das Berechtigungskonzept in der Laufzeitumgebung enabler4BIZ bezüglich enablerWorkflows vorgestellt.

Anmerkung: In diesem und dem folgenden Kapitel werden Teile des Datenbankschematas dargestellt und erklärt. Zum besseren Verständnis wurden folgende Konventionen in der Namensgebung von Tabellen und deren Attributen angewandt (nach [Win04]):

- Zwischentabellen für n zu m Relationen haben den Anhang "_Link" und werden Linktabellen genannt.
- Jede Tabelle hat einen vom Datenbanksystem vergebenen, numerischen Primärschlüssel (identity). Die ersten zwei Buchstaben sind eine Abkürzung für den Tabellennamen und haben den Anhang "_ID". Der Primärschlüssel von Linktabellen beginnt mit vier Buchstaben, die sich aus je zwei Buchstaben der beiden verlinkten Tabellen zusammensetzen.

- Alle anderen Tabellenattribute beginnen ebenfalls mit dieser Abkürzung.
- Fremdschlüssel haben den gleichen Namen wie der Primärschlüssel der referenzierten Tabelle.

4.1 Konzepte

4.1.1 Das Workflow-Dokument

In enablerWorkflows wird jede Benutzerinteraktion mit dem Workflow über das Ausfüllen beziehungsweise Überarbeiten von dynamischen Formularen abgebildet. Das bedeutet, das jede Aktivität, die eine Benutzerinteraktion erfordert, mit einem dynamischen Formular verknüpft ist. Die Verknüpfung erfolgt über den Parameter "FormName". Dieser referenziert den eindeutigen Namen der Formulardefinition, der beim Erstellen eines Formulars im Formulareditor des enabler4BIZ vergeben werden muss.

Die einfachste Aktivität mit Benutzerinteraktion ist "FillOutNew_User", bei der ein Benutzer ein neues, leeres Formular ausfüllen muss. Dies kann zum Beispiel im Fall eines Workflows, der Urlaubsansuchen bearbeitet, das Antragsformular für den Urlaub sein. Alle innerhalb eines Workflows ausgefüllten Formulare werden zu einem *Workflow-Dokument* zusammengefügt, wobei das jeweils zuletzt ausgefüllte Formular an letzter Stelle im Dokument eingefügt wird. Mit jedem Workflow ist somit genau ein Workflow-Dokument assoziiert.

Im weiteren Verlauf eines Workflows können durch den Einsatz der entsprechenden Aktivitäten unter anderem

- weitere Formulare ausgefüllt und an das Workflow-Dokument angehängt werden,
- bereits im Workflow-Dokument enthaltene Formulare überarbeitet werden,
- Formulare aus dem Dokument entfernt werden und

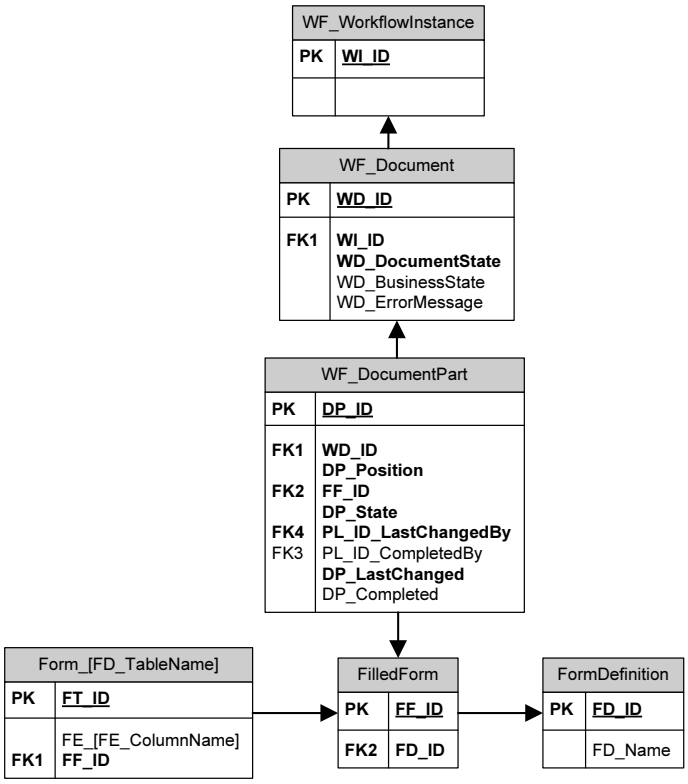

Abbildung 6: Datenbankschema für Workflow-Dokumente

- Entscheidungen (=Verzweigungen und Schleifen innerhalb des Workflows) auf Basis von ausgefüllten Formularfeldern getroffen werden.

Nach dem Ende eines Workflows ist ebenfalls die Erstellung des assoziierten Workflow-Dokuments abgeschlossen. Das Dokument als Zusammensetzung von Formularen, die von verschiedenen Benutzern ausgefüllt beziehungsweise überarbeitet wurden, ist sozusagen das Ergebnis jedes Workflows in enabler-Workflows.

Das vereinfachte Datenbankschema der Tabellen für das Workflow-Dokument ist in Abbildung 6 dargestellt.

Jeder Workflow-Instanz (Tabelle *WF_ WorkflowInstance*) ist genau ein Workflow-Dokument zugeordnet (es handelt sich um eine 1 zu 1 Beziehung). Jedes

Dokument hat einen Dokumentstatus (Attribut *WD_DocumentState*), der den Erstellungsstatus des Dokuments angibt und die Werte "*in Erstellung*", "*abgeschlossen*", "*abgebrochen*" und "*fehlerhaft*" annehmen kann, die mit dem Status des Workflows korrelieren. Sofern ein Geschäftsstatus verwendet wird (siehe 4.1.2), wird dieser im Attribut *WD_BusinessState* gespeichert. Ist im Verlauf des Workflows ein Fehler aufgetreten (Dokumentstatus: "fehlerhaft"), so wird die Fehlermeldung in *WD_ErrorMessage* abgelegt.

Die 1 bis n Teile, aus denen ein Workflow-Dokument zusammengesetzt ist, werden in Tabelle *WF_DocumentPart* gespeichert. Jeder Dokumentteil hat wiederum einen Status (Attribut *DP_State*), der folgende Werte annehmen kann: "*In Erstellung*", "*Abgeschlossen*" oder "*Gelöscht*". Im Status "In Erstellung" befindet sich ein Teil dann, wenn der Benutzer im Formular bereits Daten eingegeben, die Eingabe aber noch nicht abgeschlossen hat (Der Workflow bleibt im aktuellen Schritt, die Daten werden zwischengespeichert). Den Status "Gelöscht" nimmt ein Dokumentteil an, der durch die Aktivität "RemoveForm" (siehe 4.3.14) aus dem Workflow-Dokument gelöscht wurde. Der Teil wird in der Gesamtansicht des Dokuments nicht mehr angezeigt.

Im Attribut *DP_Position* der Tabelle WF_DocumentPart wird die Position des Teils innerhalb des Workflow-Dokuments gespeichert. Weiters wird in den Attributen *PL_ID_LastChangedBy* und *DP_LastChanged* mitprotokolliert, wer die letzte Änderung im Formular durchgeführt hat und wann diese durchgeführt wurde. *PL_ID_ComplededBy* und *DP_Completed* zeigen, wer die Eingabe abgeschlossen hat und wann sie abgeschlossen wurde. Das Attribut *PL_ID* ist der Primärschlüssel der Tabelle "*Personel*", in der die Benutzer des enabler4BIZ gespeichert sind. Der Fremdschlüssel *FF_ID* verweist auf die eigentlichen Daten des ausgefüllten Formulars, die sich in der Tabelle *FilledForm* befinden.

Die Tabellen rund um die dynamischen Formulare werden in Abbildung 6 nur grob skizziert: Die Tabelle *FilledForm* kapselt das ausgefüllte Formular. Die eigentlichen Werte der Felder des ausgefüllten Formulars stehen in einer Tabelle mit dem Namen *Form_<Name der Formulardefinition>*, die für jede Formulardefinition automatisiert angelegt wird. In dieser Tabelle ist für jedes Feld im Formular ein Attribut vorhanden. Die Tabelle *Filled-*

Form enthält weiters über den Fremdschlüssel *FD_ID* eine Referenz auf die zu Grunde liegende Formulardefinition in Tabelle *FormDefinition* mit dem Formularnamen *FD_Name*. In weiteren Tabellen rund um Tabelle *FormDefinition* werden die Eigenschaften und Positionen der Felder gespeichert, aus denen das Formular aufgebaut ist. Diese Tabellen wurden in der Abbildung nicht dargestellt.

4.1.2 Geschäftsstatus

Neben dem "technischen" Status eines Workflows/Workflow-Dokuments ("*in Erstellung*", "*abgeschlossen*", "*abgebrochen*" und "*fehlerhaft*") kann für Workflows auch ein "inhaltlicher" Status angegeben werden, also in welcher Aktivität sich der Workflow gerade befindet (z.B. Status "Datenprüfung"). Die aktuelle Aktivität kann im Userinterface zu jedem Workflow eingeblendet werden. Nach [Mue04] ist es aber oft nicht sinnvoll oder erwünscht, dem Benutzer den Status in solcher Detailschärfe anzuzeigen, speziell bei Systemen, in die auch der Kunde eingebunden ist. Als Alternative wird hier der *Geschäftsstatus* eingeführt, der jeweils einige Aktivitäten zu einem Status zusammenfasst, also zum Beispiel Aktivität "Inhaltliche Prüfung" und "Technische Prüfung" zum Status "Prüfung".

In enablerWorkflows kann für jede Workflow-Definition separat entschieden werden, ob ein Geschäftsstatus verwendet werden soll. Bei Verwendung kann der Status durch die Aktivität "SetState" (siehe 4.3.18) gesetzt beziehungsweise geändert werden: Ein Workflow hat zu einem Zeitpunkt immer genau einen Geschäftsstatus. Die erste SetState-Aktivität setzt den Status des Workflows, jede weitere SetState-Aktivität ändert ihn global für den gesamten Workflow. Der letzte gesetzte Status am Ende eines Workflows kann gleichzeitig als der Gesamtstatus des Workflows angesehen werden (z.B. "Urlaub genehmigt" bei einem Workflow, der Urlaubsansuchen bearbeitet). Der Geschäftsstatus kann im Userinterface bei jedem Workflow eingeblendet werden. Der Statustext selbst ist vom Workflow-Designer frei wählbar, die Texte werden für jede Workflow-Definition in einer eigenen Ressource-Datei abgelegt.

4.1.3 Die Aufgabenliste

Wie unter 2.8 erwähnt, werden Arbeitsaufgaben in Workflow-Management-Systemen in Form von Arbeitslisten verwaltet. In enablerWorkflows wurde diese Arbeitsliste in Form der Datenbanktabelle *WF_PendingFillFormActivity* implementiert. Aktivitäten, die eine Benutzerinteraktion erfordern (siehe 4.2.2, "*EnablerUserInteractionActivity*"), tragen ihre Aufgabe in diese Tabelle ein. Die Aufgaben werden dann vom Laufzeitsystem im enabler4BIZ dem Benutzer zur Bearbeitung angezeigt.

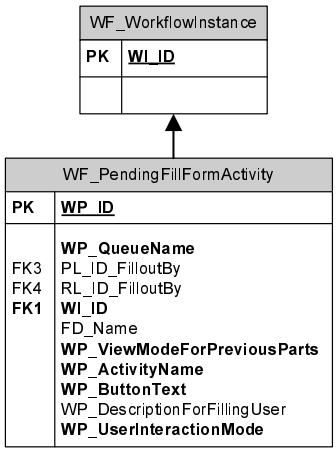

Abbildung 7: Datenbankschema WF_PendingFillFormActivity (Die Attribute der Tabelle "WF_WorkflowInstance" werden zur Vereinfachung hier nicht angeführt.)

In Abbildung 7 ist das Datenbankschema der Tabelle dargestellt. Die Tabelle enthält neben der ID folgende Attribute:

WI_ID: Referenz auf die Workflow-Instanz, von der diese Aufgabe erstellt wurde

PL_ID_FilloutBy: ID des Benutzers, dem diese Aufgabe zugeteilt ist

RL_ID_FilloutBy: Alternativ zu einem Benutzer kann auch die ID einer Rolle angegeben werden. Die Aufgabe muss dann von einem Benutzer mit dieser Rolle erledigt werden.

WP_UserInteractionMode: Kann drei Werte annehmen:

- *Neu*: Der Benutzer muss ein neues Formular ausfüllen, das am Ende des bestehenden Workflow-Dokuments angehängt wird.
- *Bearbeiten*: Der Benutzer muss ein bestehendes Formular innerhalb des Workflow-Dokuments überarbeiten.
- *Anzeigen*: Das bisher in diesem Workflow entstandene Workflow-Dokument wird dem Benutzer nur angezeigt. Er kann selbst keine neuen Daten eingeben oder ändern, muss aber bestätigen, dass er das Dokument gesehen hat.

FD_Name: gibt bei WP_UserInteractionMode = "Neu" das Formular an, das ausgefüllt werden muss beziehungsweise bei WP_UserInteractionMode = "Bearbeiten" das Formular an, das überarbeitet werden muss

WP_ButtonText: Der Benutzer muss die Erledigung der Aufgabe durch einen Button-Click bestätigen. Die Beschriftung des Buttons erfolgt individuell für jede Aktivität. Nach Eingabe eines Urlaubsantrags könnte der Text zum Beispiel "Urlaubsantrag stellen" lauten.

WP_DescriptionForFillingUser: Kann einen Kommentar für den Benutzer enthalten, der ihm beim Ausfüllen oder Ansehen des Formulars angezeigt wird.

WP_ViewModeForPreviousParts: Regelt die Sichtbarkeit der bisher ausgefüllten Formularteile für den ausfüllenden Benutzer. Folgende Varianten sind möglich:

- HideAll: Das gesamte bisher eingegebene Workflow-Dokument wird ausgeblendet.

- ShowFirstCollapsed: Nur der erste ausgefüllte Formularteil des Workflow-Dokuments wird angezeigt, allerdings zugeklappt (Der Benutzer kann die angezeigten Formularteile zur besseren Übersicht zu- beziehungsweise aufklappen.).
- ShowFirstExpanded: Der erste ausgefüllte Formularteil wird aufgeklappt angezeigt. Andere eventuell bereits ausgefüllte Formularteile sind nicht sichtbar.
- ShowLastCollapsed: Nur der letzte ausgefüllte Formularteil wird angezeigt, allerdings zugeklappt.
- ShowLastExpanded: Der letzte ausgefüllte Formularteil wird aufgeklappt angezeigt.
- ShowAllCollapsAll: Alle bereits ausgefüllten Formularteile werden angezeigt, alle zugeklappt.
- ShowAllExpandFirst: Alle bereits ausgefüllten Formularteile werden angezeigt, der erste davon aufgeklappt.
- ShowAllExpandLast: Alle bereits ausgefüllten Formularteile werden angezeigt, der letzte davon aufgeklappt.
- ShowAllExpandAll: Alle bereits ausgefüllten Formularteile werden aufgeklappt angezeigt.

Dabei können alle oder nur bestimmte Teile angezeigt oder das gesamte bisher eingegebene Workflow-Dokument ausgeblendet werden.

WP_ActivityName: ist der Name der Aktivität, von der diese Aufgabe erstellt wurde

WP_QueueName: ist der Name der Event-Queue, die verwendet wird, um den Event der Erledigung einer Aufgabe zurück an den Workflow zu schicken

4.2 Aktivitäten allgemein

Ziel war es hier, einen Grundstock an Aktivitäten zur Verfügung zu stellen, die eng mit den dynamischen Formularen des enabler4BIZ zusammenarbeiten, und die es ermöglichen, möglichst viele der in Geschäftsprozessen vorkommenden Aufgaben abzudecken. Um das zu erreichen, war es nötig, gemeinsame Muster in Geschäftsprozessen zu finden und für diese geeignete Aktivitäten zur Verfügung zu stellen.

4.2.1 Muster

[RZ96] schreibt, ein Muster ist "the abstraction from a concrete form which keeps recurring in specific nonarbitrary contexts". In [GHJV95] wurden erstmals 23 Design Patterns für objektorientierte Systeme vorgestellt. Diese zeichnen sich dadurch aus, dass sie unabhängig von der Implementierungstechnologie und den Anforderungen der konkreten Domäne beschrieben waren.

Für die Domäne der Workflows haben W.M.P. van der Aalst et al. in mehreren Arbeiten [RHEA04a, RHEA04b, AHKB03, RHAM06] Muster erkannt und diese systematisiert und katalogisiert. [AHKB03] beschreibt, dass es folgende vier Perspektiven gibt, aus denen Workflow-Spezifikationen betrachtet werden können:

- *Kontrollfluss Perspektive*: Beschreibt die Aktivitäten und deren Ausführungsreihenfolge, wie zum Beispiel sequenzielle Ausführung, parallele Ausführung oder Verzweigung. 20 Muster für diesen Bereich wurden in [AHKB03] beschrieben. Eine Überarbeitung dieser 20 Muster, sowie 23 neue Kontrollfluss Muster werden in [RHAM06] präsentiert.
- *Daten Perspektive*: Beschreibt die Auswirkungen von Workflow Applikations- und Kontrolldaten (wie zum Beispiel Formulardaten), speziell im Hinblick auf Vor- und Nachbedingungen der Aktivitätsausführung. Auch auf das Zusammenspiel von internen Workflowdaten und Daten aus externen Systemen wird eingegangen. Hier erkannte Muster wurden erstmals in [RHEA04a] beschrieben.

- *Ressourcen Perspektive*: Beschreibt den Bezug zu Organisationsstrukturen, also zum Beispiel, welche Personen oder Rollen für die Ausführung von Aktivitäten zuständig sind. Auch nicht-menschliche Ressourcen, wie zum Beispiel Anlagen und Geräte, werden berücksichtigt. Muster in diesem Bereich werden in [RHEA04b] beschrieben.

- *Operative Perspektive*: Beschreibt die elementaren Aktionen, die von Aktivitäten ausgeführt werden, wobei darunterliegende Applikationen aufgerufen und typischerweise Workflowdaten übergeben werden. Muster in diesem Bereich wurden bisher nicht beschrieben, da es sich um Anbindungen an die unterschiedlichsten Software-Systeme handelt, die spezifisch für einzelne Workflows oder Umgebungen sind. Von den im Folgenden vorgestellten Aktivitäten ist die SendMail Aktivität in diesem Bereich zu sehen, da sie Daten an den SMTP-Connector des enabler4BIZ weiterleitet.

In [RAH06] werden weitere Workflow-Patterns für den Bereich Exception Handling eingeführt. Die in dieser Arbeit beschriebene Version von enablerWorkflows stellt jedoch keine eigenen Aktivitäten für diesen Bereich zur Verfügung. Jedoch beinhalten die Basisaktivitäten aus der Windows Workflow Foundation bereits selbst einige Aktivitäten zum Thema Exception Handling. Diese können bereits jetzt ohne Einschränkung verwendet werden, und neue Aktivitäten in zukünftigen Versionen von enablerWorkflows werden auf diesen aufbauen.

In dieser Arbeit wurde versucht, einen Grundstock an Mustern auszuwählen und diesen in Form von Aktivitäten dem Workflow-Designer zur Verfügung zu stellen. Dabei ist zu beachten, dass viele Muster bereits durch die Basisaktivitäten aus der Workflow Foundation abgedeckt werden. Diese wurden allerdings zum Teil durch eigene, neue Aktivitäten ergänzt beziehungsweise ersetzt, die darauf abzielen, das Zusammenwirken mit den dynamischen Formularen des enabler4BIZ zu optimieren. Weiters soll dadurch zusätzliche Codierarbeit bei der Abbildung von Geschäftsfällen, welche bei vielen der Basisaktivitäten nötig ist, vermieden werden.

In der vorliegenden Version des enabler4BIZ werden derzeit noch relativ wenige dieser Muster in Aktivitäten abgebildet. Allerdings lassen das in ena-

blerWorkflows verwirklichte Basisframework und die in 4.2.2 beschriebenen Basisklassen es zu, neue Aktivitäten sehr schnell und mit wenig Programmieraufwand zu erstellen und hier hinzuzufügen.

Bei der Auflistung der von enablerWorkflows zur Verfügung gestellten Aktivitäten unter 4.3 wird, soweit möglich, eine Referenz zu dem jeweiligen Muster, das dadurch abgebildet wird, angeführt und das Muster kurz erklärt.

Anzumerken ist noch, dass nicht alle Muster in den oben genannten Arbeiten in Form von Aktivitäten abzubilden sind. Einige Muster müssen an anderen stellen eines Workflow-Management-Systems abgebildet werden (zum Beispiel Userinterface oder Laufzeitumgebung). Auf diese Muster wird hier nicht näher eingegangen.

4.2.2 Die Basisklassen

Um Gemeinsamkeiten der Aktivitäten zentral abbilden zu können, wurde ein System an Basisklassen erstellt, von denen alle in enablerWorkflows zur Verfügung gestellten Aktivitäten abgeleitet sind. Abbildung 8 zeigt das Klassendiagramm dieser Basisklassen. Zur Vereinfachung werden nur die wichtigsten Felder und Methoden im Diagramm dargestellt und erklärt.

EnablerActivity ist die Root-Klasse aller Aktivitäten in enablerWorkflows. Sie ist selbst von der Klasse *CompositeActivity* aus der Windows Workflow Foundation abgeleitet. CompositeActivity ist die Basisklasse für alle zusammengesetzten Aktivitäten. Das sind Aktivitäten, die selbst wiederum Unteraktivitäten beinhalten können. Es beinhalten zwar nicht alle Aktivitäten in enablerWorkflows Unteraktivitäten, aber durch die Ableitung von CompositeActivity wird die Möglichkeit dafür offen gelassen.

Die EnablerActivity enthält unter anderem eine Referenz auf die Workflow-Instanz (*WorkflowInstance*) selbst und das damit verbundene Workflow-Dokument (*Document*). Weiters wird die Methode "*Execute*" aus der Basisklasse überschrieben. Dabei handelt es sich um eine zentrale Methode, die von der Workflow-Runtime aufgerufen wird, sobald die Aktivität zur Ausführung kommt. "Execute" (siehe auch Listing 1) ruft nacheinander folgende

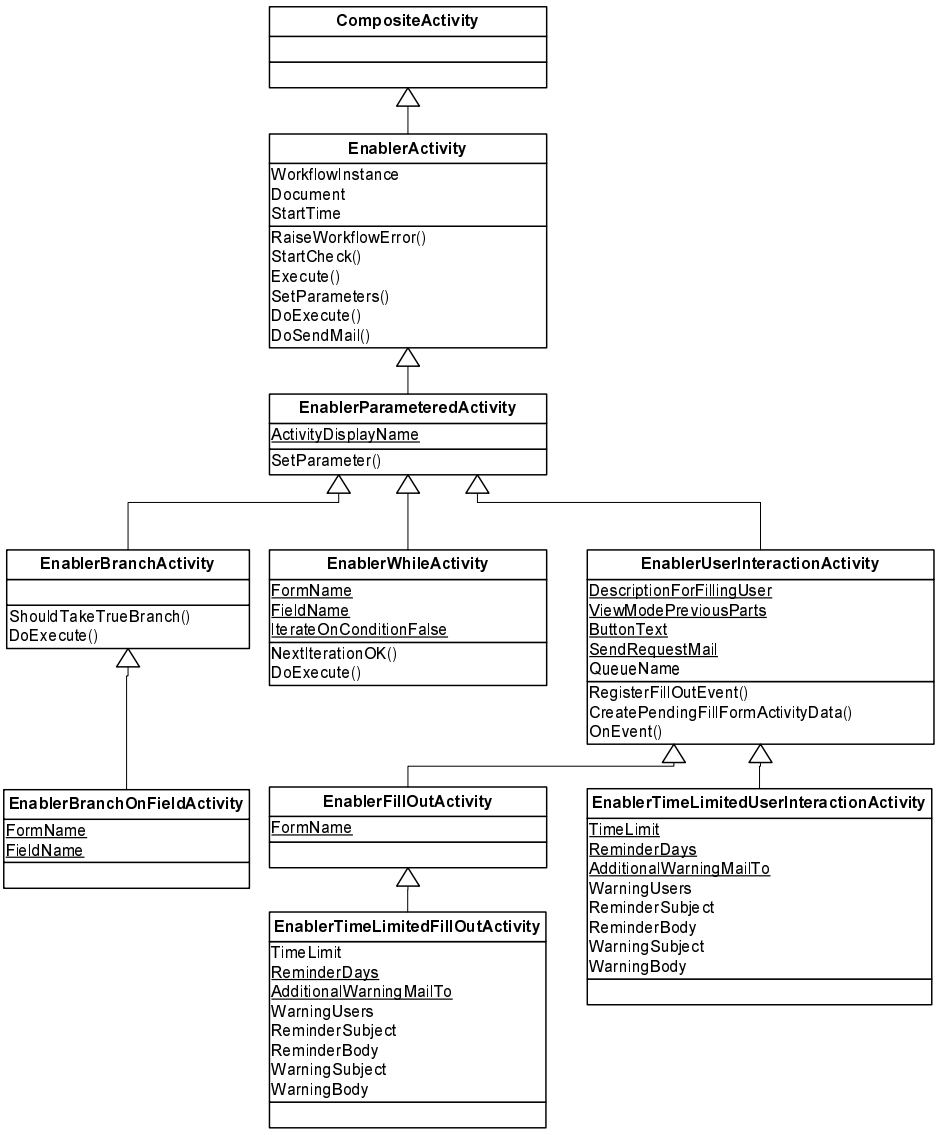

Abbildung 8: Basisklassen der Aktivitäten

Listing 1 Die Methode "Execute"

```
protected override ActivityExecutionStatus Execute(
      ActivityExecutionContext executionContext) {
  try {
    this.SetParameters();
    // Arbeit verrichten
    ActivityExecutionStatus _status = this.DoExecute(executionContext);
    // Benachrichtigungsmails verschicken
    this.DoSendMail();
    return _status;
  } catch (Exception ex) {
    try {
      // Workflow abbrechen
      this.WorkflowInstance.TerminateWorkflow();
    } catch {}
  }
}
```

Stub-Methoden auf, die in den abgeleiteten Aktivitäten überschrieben werden können:

- *SetParameters*: kann überschrieben werden um vor dem tatsächlichen Ausführen der Methode Parameter zu setzen (siehe auch 4.7.4, "Werte setzen")
- *DoExecute*: ersetzt inhaltlich die hier überschriebene Methode "Execute" und muss von den abgeleiteten Aktivitäten überschrieben werden, um die eigentliche Arbeit zu verrichten
- *DoSendMail*: kann von Aktivitäten mit Userinteraktion zur Benachrichtigung der Benutzer per E-Mail überschrieben werden

Im Fehlerfall wird die Workflow-Instanz über *TerminateWorkflow()* kontrolliert abgebrochen.

Die überschreibbare Methode "*StartCheck*" wird von enablerWorkflows aufgerufen, bevor eine neue Workflow-Instanz tatsächlich gestartet wird. Hier kann überprüft werden, ob alle Startbedingungen für die Aktivität erfüllt sind.

Schließlich ermöglicht es die Methode "*RaiseWorkflowError*" den Aktivitäten, auftretende Fehler kontrolliert an das Workflow-Management-System zur weiteren Behandlung weiterzugeben.

EnablerParameteredActivty ist die Basisklasse aller Aktivitäten, die über Parameter konfiguriert werden. Sie überschreibt dafür die Methode "SetParameters" aus der Basisklasse "EnablerActivity". Da Parameter vom Administrator oder Benutzer im enabler4BIZ gesetzt werden können, erhalten diese Aktivitäten vom Workflow-Designer über die Eigenschaft "ActivityDisplayName" einen sprechenden Namen für die Anzeige im Userinterface (für Details zur Parameterübergabe siehe 4.7.4, "Werte setzen").

EnablerBranchActivity ist die Basisklasse aller Aktivitäten, die im Workflow-Ablauf eine Verzweigung in zwei Ausführungsstränge bewirken, von denen genau ein Weg genommen wird. Die Entscheidung, welcher Zweig (True- oder False-Zweig) genommen werden soll, wird in der überschreibbaren Methode "*ShouldTakeTrueBranch*" getroffen.

Die Methode "*DoExecute*" aktiviert entsprechend des booleschen Wertes, der von "ShouldTakeTrueBranch" zurückgegeben wird, die Aktivitäten des True-Branches beziehungsweise die des False-Branches.

EnablerBranchOnFieldActivity ist die Basisklasse für Verzweigungsaktivitäten, bei denen die Entscheidung, welcher Zweig (True- oder False-Zweig) genommen werden soll, auf Basis eines ausgefüllten Formularfeldes getroffen wird. Welches Feld das ist, wird über die Parameter "*FormName*" für den Formularnamen und "*FieldName*" für das Feld selbst gesteuert. Zum Zeitpunkt der Aktivierung einer solchen Aktivität wird in allen in diesem Workflow ausgefüllten Formularen nach einem Formular mit dem angegebenen Formularnamen gesucht. Wurden im Workflow bereits mehrere Formulare mit dem gleichen Namen ausgefüllt, wird das Formularelement des zuletzt ausgefüllten Formulars für die Entscheidung herangezogen. Die tatsächliche Entscheidungsfindung passiert dann in der überschriebenen Methode "ShouldTakeTrueBranch" in der abgeleiteten Aktivität. Existiert das Formular oder das angegebene Feld gar nicht, wird automatisch der False-Zweig ausgeführt.

EnablerWhileActivity ist die Basisklasse für Aktivitäten, die eine Sequenz von Aktivitäten so lange wiederholen, solange eine bestimmte Beding-

ung erfüllt ist. Die Entscheidung, ob die Schleife erneut ausgeführt werden soll, wird, analog zur *EnablerBranchOnFieldActivity*, wieder auf Basis eines ausgefüllten Formularfeldes getroffen, das über die Parameter "*FormName*" und "*FieldName*" spezifiziert wird.

Über den Parameter "*IterateOnConditionFalse*" kann auf negative Logik umgeschalten werden, das heißt, die Schleife wird so lange ausgeführt, solange die Bedingung *nicht* erfüllt ist.

Analog zur Methode "*ShouldTakeTrueBranch*" in der *EnablerBranchActivity* wird hier in der überschreibbaren Methode "*NextIterationOK*" die Entscheidung getroffen, ob die Schleife nochmals ausgeführt werden soll.

In der Methode "*DoExecute*" werden dann auf Basis des Ergebnisses von "*NextIterationOK*" und dem Parameter "*IterateOnConditionFalse*" die innerhalb dieser Schleife enthaltenen Aktivitäten (erneut) zur Ausführung aktiviert oder die While-Aktivität beendet.

EnablerUserInteractionActivity ist die Basisklasse für Aktivitäten, die eine Benutzerinteraktion erfordern. Benutzerinteraktion wird in enablerWorkflows immer durch das Anzeigen einiger oder aller Formularteile des bis zu diesem Zeitpunkt erstellten Workflow-Dokuments realisiert. Je nach verwendeter Aktivität hat der Benutzer zusätzlich die Möglichkeit, eines der Formulare zu überarbeiten oder Daten in ein neues Formular einzugeben.

Die Information, dass ein Benutzer eine Aufgabe innerhalb eines Workflows zu erfüllen hat, muss an das Laufzeitsystem im enabler4BIZ weitergegeben werden. Dies geschieht über die Eintragung einer neuen Arbeitsaufgabe in eine Aufgabenliste (siehe 4.1.3). Die Eintragung wird durch die Methode "*CreatePendingFillFormActivityData*" vorgenommen. Die Parameter "*DescriptionForFillingUser*", "*ButtonText*" und "*ViewModePreviousParts*" für die Aufgabe werden ebenfalls unter 4.1.3 beschrieben. Der Parameter "*SendRequestMail*" gibt an, ob der Benutzer eine Aufforderung zur Bearbeitung der Aufgabe erhalten soll.

Da die Aufgabe vom Benutzer asynchron bearbeitet wird, muss sich die Aktivität nach Erstellung der Aufgabe für ein Event registrieren. Über dieses

Event meldet das Laufzeitsystem die Erledigung der Aufgabe an die Aktivität zurück. Events werden in der Windows Workflow Foundation über Queues geschickt. Über die Methode "*RegisterFillOutEvent*" wird der Event registriert, der Name der Queue ist *"QueueName*". Nach der Registrierung wird die Aktivität in den Status "*Suspendiert*" versetzt, der Workflow wird angehalten. *"OnEvent*" wird von der Workflow Foundation aufgerufen, sobald der Event ausgelöst wurde. Die Aktivität wird daraufhin beendet, der Workflow kann weiter laufen.

EnablerFillOutActivity ist die Basisklasse für Aktivitäten mit Benutzerinteraktion, bei denen das Workflow-Dokument nicht nur angezeigt wird, sondern auch ein Formular vom Benutzer neu auszufüllen oder zu bearbeiten ist. Einzig neu ist hier der Name des Formulars ("*FormName*"), das bearbeitet werden muss.

EnablerTimeLimitedFillOutActivity und EnablerTimeLimitedUserInteractionActivity sind die Basisklassen für Aktivitäten, deren Benutzerinteraktion zeitlich begrenzt sein soll. Die Parameter und die Implementierung der zeitlichen Begrenzung werden unter 4.4 beschrieben.

4.3 Die Aktivitäten im Einzelnen

Im Folgenden werden alle Aktivitäten vorgestellt, die enablerWorkflows zusätzlich zu den Basis-Aktivitäten aus der Windows Workflow Foundation anbietet. Aktivitäten, die eine zeitliche Begrenzung der Durchlaufzeit ermöglichen, sind nicht in dieser Aufstellung enthalten, sie werden im nächsten Kapitel (4.4) eingeführt.

Für jede Aktivität werden folgende Eigenschaften angegeben:

- Basisaktivität: die Basisklasse aus Kapitel 4.2.2, von der diese Aktivität abgeleitet ist

- Typ: “Zusammengesetzt” bedeutet, dass die Aktivität weitere Unteraktivitäten aufnehmen kann. Kann sie das nicht, ist die Aktivität “Einfach”. In Abbildung 9 ist beispielhaft eine zusammengesetzte Aktivität dargestellt.

- Parameter: Zusätzlich zu den aus der Basisklasse ererbten Parametern werden hier die weiteren Parameter der Aktivität aufgelistet. Das Parameterhandling selbst wird in Kapitel 4.7 erklärt. In Klammern wird markiert, ob der Parameter zum Design-Zeitpunkt der Workflow-Definition im Visual Studio bereits ausgefüllt werden kann, oder ob er eine Referenz auf Daten aus der Datenbank enthält (z.B. Benutzer oder Rollen) und daher erst zur Laufzeit im enabler4BIZ bei der Workflow-Vorlage oder beim Workflow-Start ausgefüllt werden kann. Jedenfalls können alle Parameter, die bereits zum Design-Zeitpunkt gesetzt wurden, bei Bedarf im enabler4BIZ zur Laufzeit noch geändert werden.

- Implementierte Muster: In Kapitel 4.2.1 wurden Muster für Workflow-Aktivitäten aus den Arbeiten [RHEA04a, RHEA04b, RHAM06] vorgestellt. Falls die Aktivität eines oder mehrere dieser Muster implementiert, werden diese hier kurz beschrieben.

4.3.1 FillOutNew_User

ermöglicht es, einen Benutzer einen Formularteil ausfüllen zu lassen. Der Ausgefüllte Formularteil wird unten an das Workflow-Dokument angehängt.

Basisaktivität: EnablerFillOutActivity

Typ: Einfach

Parameter:

- FillingUser (Laufzeit): Ein Benutzer aus dem enabler4BIZ, der in diesem Workflow-Schritt das angegebene Formular ausfüllen muss. Anmerkung: Der Parameter für das Formular (“FormName”) wird bereits in einer Basisaktivität beschrieben.

Implementierte Muster: Implementiert das Direct Allocation-Muster (R-DA aus [RHEA04b]): Zuteilung einer Aufgabe zu einer Ressource zum Design-Zeitpunkt. In enablerWorkflows sind als Ressourcen derzeit nur Personen möglich, nicht-menschliche Ressourcen wie Anlagen und Geräte werden nicht unterstützt.

Weiters wird das Deferred Allocation-Muster (R-FBA aus [RHEA04b]) unterstützt: Die Möglichkeit, die Aufgabe erst zur Laufzeit einer Ressource zuzuordnen. Dies ist zum Beispiel durch den FillOutModus "AutoFillFromFormField" bei Aktivitätsparametern möglich (siehe 4.7.4).

Das Case Handling-Muster (R-CH aus [RHEA04b]) wird nur bedingt unterstützt: Es beschreibt die Möglichkeit, alle Aufgaben innerhalb eines Workflows derselben Ressource zuzuordnen. Dies wird in enablerWorkflows nur für den Fall des Workflow-Erstellers selbst als Ressource unterstützt, nämlich durch den FillOutModus "AutoFillWithStartingUser" für Parameter (siehe 4.7.4).

Auch das Retain Familiar-Muster (R-RF aus [RHEA04b]) wird unterstützt: Die Möglichkeit, eine Aufgabe derselben Ressource zuzuordnen, die bereits die vorige Aufgabe erledigt hat. Dies ist möglich durch Verwendung des Formularelements "FillingUser" (siehe 3.3.1) bei der vorigen Aufgabe. Das Feld wird automatisch mit dem ausfüllenden Benutzer befüllt, der dann im nächsten Schritt über den FillOutModus "AutoFillFromFormField" (siehe 4.7.4) als zuständige Ressource übernommen werden kann.

4.3.2 FillOutNew_Role

ermöglicht es, einen Benutzer einen Formularteil ausfüllen zu lassen, der einer bestimmten Rolle angehören muss. Der ausgefüllte Formularteil wird unten an das Workflow-Dokument angehängt.

Der enabler4BIZ bietet ein rollenbasiertes Berechtigungssystem, wie es z.B. in [SCFY96] vorgestellt wird. Diese Rollen können in der Aktivität dazu genutzt werden, das Ausfüllen eines Formulars nicht an einen bestimmten Benutzer zu binden, sondern an eine Gruppe von Benutzern mit der angegebenen Rolle. Es ist egal, welcher von diesen Benutzern dann das Formular tatsächlich

ausfüllt. Sobald das Formular von einem dieser Benutzer ausgefüllt wurde, läuft der Workflow weiter. Dadurch wird verhindert, dass Workflows zum Beispiel wegen Urlaub, Krankheit oder Abgang eines einzelnen, am Workflow beteiligten Mitarbeiters "hängenbleiben" und deshalb abgebrochen und neu gestartet werden müssen, wie das bei der Aktivität FillOutNew_User der Fall sein könnte.

Basisaktivität: EnablerFillOutActivity

Typ: Einfach

Parameter:

- FillingRole (Laufzeit): Eine Rolle aus dem enabler4BIZ, der ein Benutzer angehören muss, um das Formular auszufüllen.

Implementierte Muster: Implementiert das Role-Based Allocation-Muster (R-RBA aus [RHEA04b]): Festlegung zum Design-Zeitpunkt, dass eine Aufgabe nur von einer Ressource mit einer bestimmten Rolle erledigt werden kann.

Weiters wird auch das Deferred Allocation-Muster (R-FBA aus [RHEA04b]) wie bei der FillOutNew_User-Aktivität unterstützt (siehe 4.3.1).

4.3.3 FillOutNew_MultipleUsers

ermöglicht es, einen Formularteil parallel von mehreren Benutzern ausfüllen zu lassen. Jeder Benutzer füllt dabei eine neue Kopie desselben Formulars aus. Die ausgefüllten Formularteile werden in der Reihenfolge, in der sie ausgefüllt werden, unten ans Workflow-Dokument angehängt. Jeder der angegebenen Benutzer muss das Formular bearbeiten, bevor der Workflow weiterlaufen kann. Diese Aktivität kann zum Beispiel sinnvoll sein im Fall einer Prüfung, die von mehreren Benutzern durchgeführt werden muss.

Basisaktivität: EnablerFillOutActivity

Typ: Einfach

Parameter:

- FillingUsers (Laufzeit): Mehrere Benutzer aus dem enabler4BIZ, die in diesem Workflow-Schritt das angegebene Formular ausfüllen müssen.

4.3.4 EditExisting_User

ermöglicht es, einen Benutzer einen bereits im Workflow-Dokument befindlichen Formularteil überarbeiten zu lassen, das bereits zuvor von einem anderen Benutzer hinzugefügt wurde (z.B. durch die Aktivität FillOutNew_User).

Basisaktivität: EnablerFillOutActivity

Typ: Einfach

Parameter:

- FillingUser (Laufzeit): Ein Benutzer aus dem enabler4BIZ, der in diesem Workflow-Schritt das angegebene, bereits ausgefüllte, Formular überarbeiten muss.

Implementierte Muster: Implementiert die gleichen Muster wie die FillOutNew_User-Aktivität (siehe 4.3.1).

4.3.5 EditExisting_Role

Ähnlich wie EditExisting_User (4.3.4), nur dass wie bei FillOutNew_Role (4.3.2) ein beliebiger Benutzer mit der angegebenen Rolle das Formular überarbeiten muss.

Basisaktivität: EnablerFillOutActivity

Typ: Einfach

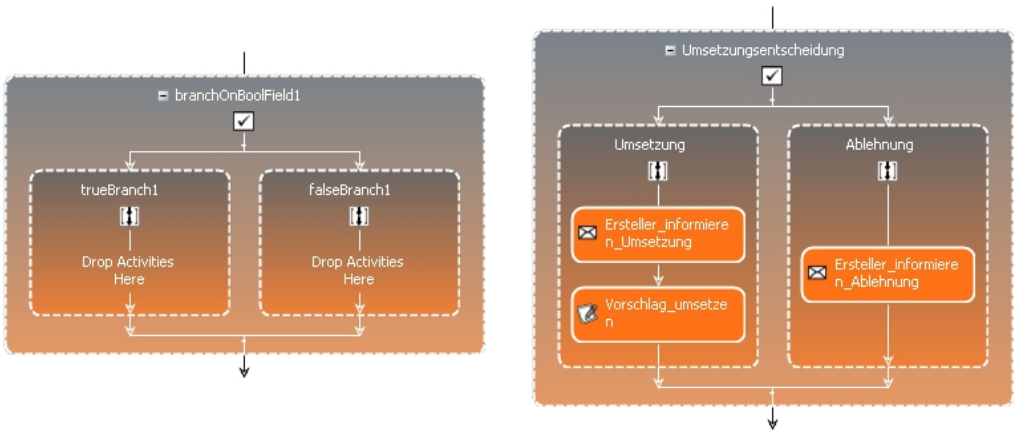

Abbildung 9: Aktivität "BranchOnBoolField". Links leer, rechts ausgefüllt mit Sub-Aktivitäten.

Parameter:

- FillingRole (Laufzeit): Eine Rolle aus dem enabler4BIZ, der ein Benutzer angehören muss, um das Formular zu überarbeiten.

Implementierte Muster: Implementiert die gleichen Muster wie die FillOutNew_Role-Aktivität (siehe 4.3.2).

4.3.6 BranchOnBoolField

ermöglicht eine Verzweigung des Workflows, abhängig vom Wert eines booleschen Feldes in einem bereits ausgefüllten Formular. Bei dem Feld kann es sich um eine Checkbox, einen Radiobutton oder eine DropDownList mit den Auswahlmöglichkeiten "true" und "false" handeln. Ist der Wert des Feldes "true", wird der True-Zweig der Aktivität und die darin enthaltenen Sub-Aktivitäten ausgeführt, ansonsten der False-Zweig. In Abbildung 9 ist stellvertretend für alle zusammengesetzten Aktivitäten diese Aktivität dargestellt: Links ist die leere Aktivität zu sehen, nachdem sie neu im Designer des Visual Studios hinzugefügt wurde. Rechts ist die Aktivität mit sinnvollen Namen versehen und mit Sub-Aktivitäten in den beiden Ausführungszweigen befüllt.

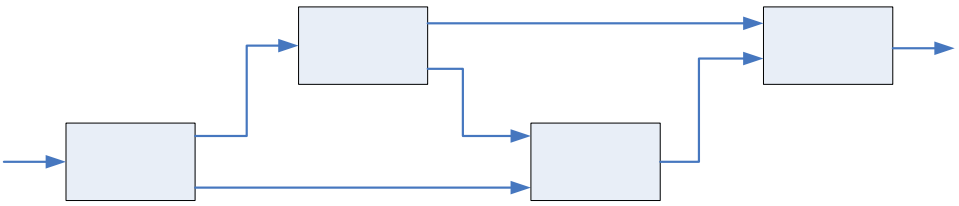

Abbildung 10: Verzweigungsfolge, die in der Windows Workflow Foundation nicht möglich ist

Basisaktivität: EnablerBranchOnFieldActivity

Typ: Zusammengesetzt

Parameter:

- Die beiden Parameter "FormName" und "FieldName", die das entsprechende Formularfeld spezifizieren, sind bereits in der Basisaktivität enthalten, siehe 4.2.2 (EnablerBranchOnFieldActivity).

Implementierte Muster: Implementiert das Exclusive Choice-Muster (WCP-4 aus [RHAM06]): Die Aufzweigung eines Pfades in zwei oder mehr Pfade, von denen genau einer aktiviert wird. Die Entscheidung, welcher Pfad aktiviert wird, wird auf Basis eines logischen Ausdrucks getroffen. In diesem Fall ist das der Wert eines booleschen Feldes im Formular.

Weiters wird das Structured Discriminator-Muster (WCP-9 aus [RHAM06]) indirekt implementiert: Die Zusammenführung von zwei oder mehr Pfaden zu einem einzelnen nachfolgenden Pfad, wobei der nachfolgende Pfad gestartet wird, sobald der erste eintreffende Pfad aktiviert wurde. Von weiteren eintreffenden Pfaden, die in weiterer Folge ebenfalls aktiviert werden, wird der nachfolgende Pfad nicht noch einmal gestartet. In enablerWorkflows (und der Windows Workflow Foundation im Allgemeinen) kann dieser Fall allerdings nicht eintreten, da jede Aktivität, die ein Muster implementiert, das zur Aufteilung auf mehrere Pfade führt, implizit auch die Zusammenführung dieser

Pfade in sich trägt (wie in Abbildung 9 zu sehen). Eine Verzweigungsfolge wie in Abbildung 10 ist daher generell nicht möglich.

Auch das Data-based Routing-Muster (Pattern 39 aus [RHEA04a]) wird implementiert: Die Möglichkeit, den Kontrollfluss des Workflows auf Basis eines Datenwertes zu ändern.

4.3.7 BranchOnIntField

ermöglicht eine Verzweigung des Workflows abhängig vom Wert einer Integer-Textbox in einem bereits ausgefüllten Formular. Der Wert der Textbox wird mit dem Operator aus dem Parameter "CompareOperator" mit dem Integer-Wert im Parameter "CompareValue" verglichen. Ist das Ergebnis dieses Vergleichs "true", wird der True-Zweig der Aktivität und die darin enthaltenen Sub-Aktivitäten ausgeführt, ansonsten der False-Zweig.

Basisaktivität: EnablerBranchOnFieldActivity

Typ: Zusammengesetzt

Parameter:

- CompareOperator (Design): Operator für den Vergleich mit dem CompareValue. Zur Auswahl stehen die Operatoren "=", "<" und ">".
- CompareValue (Design): Ein Integer Wert, mit dem der Wert aus dem Formularfeld verglichen wird.

Implementierte Muster: Implementiert die gleichen Muster wie BranchOnBoolField (siehe 4.3.6).

4.3.8 BranchOnStringField

ermöglicht eine Verzweigung des Workflows abhängig vom Inhalt einer Textbox in einem bereits ausgefüllten Formular. Der Text der Textbox wird mit

dem Operator "==" mit dem Text im Parameter "CompareValue" verglichen. Bei Gleichheit wird der True-Zweig der Aktivität und die darin enthaltenen Sub-Aktivitäten ausgeführt, ansonsten der False-Zweig. Für den Vergleich werden die beiden Texte mittels "ToLower()" in Kleinbuchstaben umgewandelt.

Basisaktivität: EnablerBranchOnFieldActivity

Typ: Zusammengesetzt

Parameter:

- CompareValue (Design): Ein Text, mit dem der Inhalt aus dem Formularfeld verglichen wird.

Implementierte Muster: Implementiert die gleichen Muster wie BranchOnBoolField (siehe 4.3.6).

4.3.9 BranchOnMultiple...Field

Zu den Aktivitäten "BranchOnBoolField", "BranchOnIntField" und "BranchOnStringField" werden die drei Varianten "BranchOnMultipleBoolField", "BranchOnMultipleIntField" und "BranchOnMultipleStringField" angeboten. Anstatt nur den Wert aus dem ersten gefundenen Formular als Basis für die Verzweigungsentscheidung heranzuziehen, werden hier alle Formulare mit dem gesuchten Namen und alle Werte in diesen Formularen berücksichtigt. Gibt der Vergleich bei allen Formularwerten "true", dann wird der True-Zweig der Aktivität ausgeführt. Ergibt mindestens ein Wert beim Vergleich "false", so wird der False-Zweig ausgeführt.

Der Einsatz dieser Aktivitäten ist vor allem für die Kombination mit der Aktivität "FillOutNew_MultipleUsers" (siehe 4.3.3) gedacht, zum Beispiel um Prüfungen durch mehrere Personen zu ermöglichen.

Basisaktivität: EnablerBranchOnFieldActivity

Typ: Zusammengesetzt

Parameter:

- Gleich wie bei der entsprechenden Aktivität für einzelne Formulare.

Implementierte Muster: Implementiert die gleichen Muster wie BranchOnBoolField (siehe 4.3.6).

4.3.10 WhileOnBoolField

ermöglicht eine Wiederholung einer Sequenz von Aktivitäten des Workflows abhängig vom Wert eines booleschen Feldes in einem bereits ausgefüllten Formular. Bei dem Feld kann es sich um eine Checkbox, einen Radiobutton oder eine DropDownList mit den Auswahlmöglichkeiten "true" und "false" handeln. Solange der Wert des Feldes "true" ist, wird die Schleife und damit die darin enthaltene Sequenz von Aktivitäten ausgeführt. Der Wert des Feldes wird auch bereits vor Beginn des ersten Schleifendurchlaufs ausgewertet.

Basisaktivität: EnablerWhileActivity

Typ: Zusammengesetzt

Parameter:

- Die beiden Parameter "FormName" und "FieldName", die das entsprechende Formularfeld spezifizieren, sind bereits in der Basisaktivität enthalten, siehe 4.2.2 (EnablerWhileActivity).

Implementierte Muster: Implementiert das Structured Loop-Muster (WCP-21 aus [RHAM06]): Wiederholtes Ausführen einer oder mehrerer Aktivitäten in einer Schleife. Es gibt eine Bedingung, auf Grund derer vor oder nach jedem Schleifendurchlauf entschieden wird, ob die Schleife weiterhin

ausgeführt wird. Die Schleife hat nur einen Eintritts- und einen Austrittspunkt. Bei der WhileOnBoolField-Aktivität findet dieses Überprüfung vor jedem Schleifendurchlauf statt.

Auch das Data-based Routing-Muster (Pattern 39 aus [RHEA04a]) wird implementiert: Die Möglichkeit, den Kontrollfluss des Workflows auf Basis eines Datenwertes zu ändern.

4.3.11 WhileOnIntField

Ähnlich wie bei "WhileOnBoolField" (4.3.10) wird hier analog zur "BranchOnIntField"-Aktivität (4.3.7) abhängig vom Wert einer Integer-Textbox die Sequenz der enthaltenen Aktivitäten wiederholt.

Parameter:

- Wie bei "BranchOnIntField" (4.3.7)

4.3.12 WhileOnStringField

Ähnlich wie bei "WhileOnBoolField" (4.3.10), wird hier analog zur "BranchOnStringField"-Aktivität (4.3.8), abhängig vom Inhalt einer Textbox, die Sequenz der enthaltenen Aktivitäten wiederholt.

Parameter:

- Wie bei "BranchOnStringField" (4.3.8)

4.3.13 AppendFixedForm/InsertFixedForm

ermöglicht es, einen Formularteil am Ende des Workflow-Dokuments beziehungsweise an der angegebenen Stelle einzufügen, ohne dieses ausfüllen zu lassen. Dadurch können zum Beispiel Formulare eingefügt werden, die keine ausfüllbaren Elemente besitzen (z.B. Anweisungen oder Erklärungen für den

Betrachter des gesamten Workflow-Dokuments). Weiters können Formulare mit zu berechnenden Feldern eingefügt werden, deren Werte danach in einer Code-Aktivität berechnet werden.

Basisaktivität: EnablerParameteredActivity

Typ: Einfach

Parameter:

- FormName (Design): Name des Formulars, das eingefügt werden soll.
- InsertPosition (Design; nur bei InsertFixedForm): Stelle, an der das Formular im Workflow-Dokument eingefügt werden soll.

4.3.14 RemoveForm

ermöglicht es, einen Formularteil aus dem Workflow-Dokument zu entfernen. Die Aktivität kann dazu verwendet werden, um Formularteile, die nur temporär für den Workflow notwendig waren, im endgültigen Workflow-Dokument nach Abschluss des Workflows aber nicht mehr aufscheinen sollen, zu entfernen.

Basisaktivität: EnablerParameteredActivity

Typ: Einfach

Parameter:

- FormName (Design): Name des Formulars, das entfernt werden soll.
- RemoveAll (Design): Gibt an, ob nur das erste gefundene Formular mit dem angegebenen Namen entfernt werden soll, oder alle Formularteile mit diesem Namen innerhalb des Workflow-Dokuments. Gesucht wird auch hier zuerst bei den zuletzt hinzugefügten Formularteilen.

4.3.15 ShowToUser

ermöglicht es, das bisher in diesem Workflow erstellte Workflow-Dokument einem Benutzer anzuzeigen. Der Workflow wird erst fortgesetzt, wenn der Benutzer das Dokument angesehen und dies bestätigt hat.

Basisaktivität: EnablerUserInteractionActivity

Typ: Einfach

Parameter:

- ViewingUser (Laufzeit): Ein Benutzer aus dem enabler4BIZ, dem in diesem Workflow-Schritt das Workflow-Dokument angezeigt werden soll.

Implementierte Muster: Implementiert die gleichen Muster wie die FillOutNew_User-Aktivität (siehe 4.3.1).

4.3.16 EnablerSequenceActivity

gruppiert eine Sequenz von Aktivitäten, ist inhaltlich gleich der in der Windows Workflow Foundation enthaltenen SequenceActivity und unterscheidet sich nur durch das den anderen Aktivitäten aus enablerWorkflows angepassten Layout. Die Aktivität ermöglicht es, eine Menge von Sub-Aktivitäten unter einem gemeinsamen Namen zu gruppieren.

Basisaktivität: Diese Aktivität ist direkt von der "SequenceActivity" aus der Windows Workflow Foundation abgeleitet.

Typ: Zusammengesetzt

Parameter: keine

Implementierte Muster: Sequence-Muster (WCP-1 aus [RHAM06]): Eine Aktivität wird nach der Beendigung einer anderen Aktivität gestartet. Dies ist das Basis-Muster sämtlicher Workflows, da dadurch erst ein sequenzieller Ablauf der Aktivitäten ermöglicht wird. Die Basisklasse aller Workflows in der Windows Workflow Foundation implementiert ebenfalls dieses Muster.

4.3.17 SendMail

ermöglicht es, eine E-Mail an einen oder mehrere Benutzer zu senden. In den Parametern „EmailSubject“ und „EmailBody“ können Platzhalter für Formularfelder in der Form „##<FormName>.<FieldName>##“ verwendet werden, die dann vor dem Senden der E-Mail durch die tatsächlichen Werte aus den ausgefüllten Formularen ersetzt werden.

Basisaktivität: EnablerParameteredActivity

Typ: Einfach

Parameter:

- EmailSubject (Design): Betreff der E-Mail
- EmailBody (Design): Inhalt der E-Mail
- Recipients (Laufzeit): Liste von enabler4BIZ-Benutzern, an die diese E-Mail gesendet werden soll.

Implementierte Muster: Implementiert das Automatic Execution-Muster (R-AE aus [RHEA04b]): Automatische Ausführung einer Aufgabe ohne die Notwendigkeit des Einsatzes einer Ressource. Die SendMail-Aktivität verschickt E-Mails sofort nach dem Start der Aktivität ohne das Zutun eines Benutzers. Dieses Muster wird im Prinzip auch von allen anderen Aktivitäten aus enablerWorkflows implementiert, die keine Benutzerinteraktion benötigen, die also als Basisklasse nicht die EnablerUserInteractionActivity haben.

4.3.18 SetState

setzt oder ändert den Geschäftsstatus (siehe 4.1.2) des Workflow-Dokuments.

Basisaktivität: EnablerParameteredActivity

Typ: Einfach

Parameter:

- StateID (Design): Nummer des Status, der gesetzt werden soll. Der entsprechende Name des Status wird aus einer eigenen Ressource-Datei für diese Workflow-Vorlage ausgelesen.

4.3.19 BranchOnState

ermöglicht eine Verzweigung des Workflows, abhängig vom Geschäftsstatus (siehe 4.1.2) des Workflows. Ist der aktuelle Status des Workflows gleich dem angegebenen Status im Parameter "StateID", wird der True-Zweig der Aktivität ausgeführt, ansonsten der False-Zweig.

Basisaktivität: EnablerBranchActivity

Typ: Zusammengesetzt

Parameter:

- StateID (Design): Nummer des Status, mit dem der aktuelle Geschäftsstatus des Workflows verglichen wird.

Implementierte Muster: Milestone-Muster (WCP-18 aus [RHAM06]): Auf Basis des Status des Workflows (üblicherweise in einer parallelen Aktivität gesetzt) wird eine Aktivität aktiviert oder nicht aktiviert, das heißt, ein Aktivitätszweig wird nur dann ausgeführt, wenn der Workflow sich in dem angegebenen Status befindet.

4.4 Aktivitäten mit zeitlicher Begrenzung

Bei der Erstellung von Workflow-Definitionen beziehungsweise beim Start von Workflows gibt es seitens des abzubildenden Geschäftsfalls oft die Anforderung, maximale Durchlaufzeiten oder Ablauftermine für einzelne Schritte des Workflows oder auch für mehrere Schritte gemeinsam oder den gesamten Workflow zu definieren.

Innerhalb von enablerWorkflows können Workflows nur in Schritten, in denen es zu einer Benutzerinteraktion kommt, „hängenbleiben". Benutzerinteraktion wird in enablerWorkflows durch folgende Aktivitätstypen verlangt:

- alle FillOutNew-Aktivitäten
- alle EditExisting-Aktivitäten
- ShowToUser-Aktivität

In diesen Aktivitäten wird normalerweise ein Benutzer oder eine Gruppe von Benutzern darauf hingewiesen, dass von ihnen eine Interaktion mit dem Workflow (z.B. Ausfüllen eines Formularteils) erwartet wird. Der Workflow läuft erst dann weiter, wenn die Interaktion seitens des Benutzers durchgeführt und bestätigt wurde. Passiert dies nicht innerhalb eines definierten Zeitraums beziehungsweise bis zu einem bestimmten Termin, ist es oft nötig, Erinnerungsmails oder Ähnliches zu verschicken.

Die Definition von fixen Ablaufterminen als Datum ist nur zum Startzeitpunkt eines Workflows sinnvoll, kann aber auch hier zu Problemen führen: Wird der Ablauftermin eines Schrittes überschritten, sind eventuell auch gleich die Ablauftermine weiterer Schritte überschritten, was dazu führen würde, dass sofort nach der Aufforderungsmail, einen Formularteil auszufüllen, die entsprechende Warnmail kommen würde, dass die Deadline überschritten wurde. In enablerWorkflows wird daher nur die Definition von Durchlaufzeiten (in Tagen) ermöglicht.

4.4.1 Zeit-begrenzte Varianten der Aktivitäten mit Benutzerinteraktion

In enablerWorkflows wird zu jeder Aktivität mit Benutzerinteraktion (zum Beispiel "FillOutNew_User") eine zweite Variante der Aktivität angeboten, die eine zeitliche Begrenzung ermöglicht (zum Beispiel "FimeLimitedFillOutNew_User"). In diesen Aktivitäten ist es möglich, sowohl Erinnerungsmails vor Ablauf einer Deadline, als auch Warnmails nach Ablauf dieser Deadline zu verschicken. Die Erinnerungsmails gehen automatisch an den ausfüllenden Benutzer dieser Aktivität beziehungsweise an alle Benutzer mit der ausfüllenden Rolle. Die Warnmail kann bei Bedarf zusätzlich noch an einen weiteren, nicht in dieser Aktivität involvierten, Benutzer geschickt werden. Die Konfiguration erfolgt über drei zusätzliche Parameter, die jede der Aktivitäts-Varianten mit zeitlicher Begrenzung anbietet:

- TimeLimit: Ist die maximale Durchlaufzeit in Tagen.
- ReminderDays: Erinnerungszeitpunkt (in ganzen Tagen vor Ablauf der Durchlaufzeit): Wird kein Wert oder 0 angegeben, wird keine E-Mail geschickt.
- AdditionalWarningMailTo: Benutzer, an den die Warnmail zusätzlich geschickt werden soll. Wird hier kein Benutzer angegeben, ergeht die Warnmail nur an die in dieser Aktivität involvierten Benutzer.

Die E-Mails beinhalten Standardtexte, die auf die in Kürze erreichte Fälligkeit eines Schrittes innerhalb eines Workflows hinweisen (Erinnerungsmail) beziehungsweise auf das Überschreiten der Fälligkeit aufmerksam machen (Warnmail). Weitere Konsequenzen, wie etwa einen automatischer Abbruch solcher Workflows, gibt es nicht.

In Abbildung 11 ist der schematische innere Aufbau einer solchen zeit-begrenzten Aktivität dargestellt, in diesem Fall die Aktivität *TimeLimitedFillOutNew_User*. Man sieht, dass parallel zu der Basis-Aktivität *FillOutNew_User* zwei Timer-Aktivitäten gestartet werden: eine für den Zeitraum bis zur Erinnerungsmail und eine für den Zeitraum bis zur Deadline, zu dem die

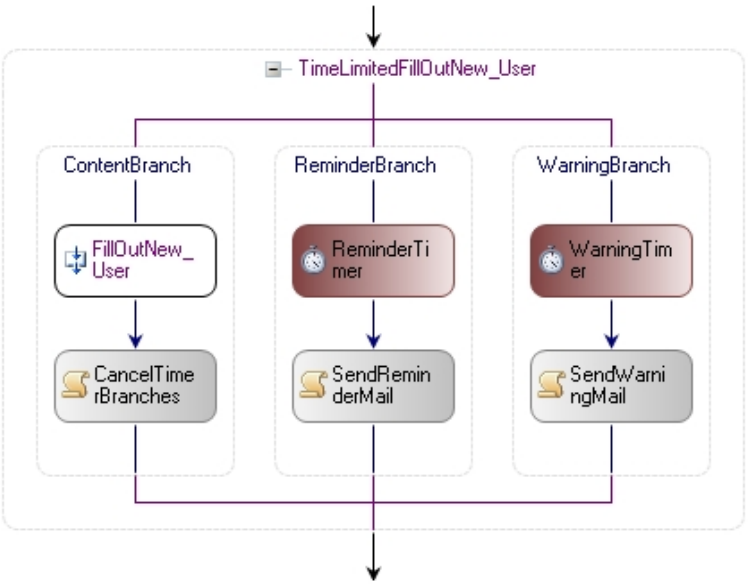

Abbildung 11: Schematische Darstellung der TimeLimitedFillOutNew_User-Aktivität

Warnmail verschickt werden soll. Die Timer-Aktivität ist Teil der Windows Workflow Foundation und blockiert den Workflow an der eingesetzten Stelle für eine angegebene Zeitspanne. Nach Ablauf einer Timer-Aktivität wird die entsprechende E-Mailerinnerung oder -warnung verschickt. Hat der Benutzer das entsprechende Formular ausgefüllt, werden alle noch laufenden Timer-Aktivitäten abgebrochen und die Aktivität *TimeLimitedFillOutNew_User* beendet.

4.4.2 TimeLimitedSequence: Aktivität für die zeitliche Begrenzung mehrerer aufeinanderfolgender Schritte im Workflow

Für den Fall, dass eine zeitliche Begrenzung nicht für einen einzelnen Schritt, sondern für mehrere aufeinanderfolgende Schritte gesetzt werden soll, stellt enablerWorkflows die Aktivität “TimeLimitedSequence” zur Verfügung. Dabei handelt es sich um eine zusammengesetzte Aktivität, die wiederum eine Sequenz von Sub-Aktivitäten in sich aufnehmen kann und diese umschließt. Diese Aktivitäten können vom Ersteller der Workflow-Definition zum Design-

Zeitpunkt in der TimeLimitedSequence-Aktivität platziert werden.

Die TimeLimitedSequence-Aktivität hat neben den selben Parameter (TimeLimit, ReminderDays, AdditionalWarningMailTo) wie die zeit-begrenzten Varianten der Einzelaktivitäten (siehe oben) noch folgenden Parameter:

- ReminderMailTo: Da die TimeLimitedSequence-Aktivität mehrere Aktivitäten mit Benutzeraktivität enthalten kann und somit nicht automatisch klar ist, welcher Benutzer der Verursacher der Verzögerung ist, muss hier vom Administrator vorab ein Benutzer ausgewählt werden, der die Erinnerungsmail erhält.

In Abbildung 12 ist links der schematische innere Aufbau der *TimeLimitedSequence*-Aktivität dargestellt. Der Aufbau ist ähnlich wie bei den anderen zeitlich begrenzten Aktivitäten wie in Abbildung 11. Statt der Basisaktivität wird hier eine "normale" Sequence-Aktivität mit dem Namen "UserActivities" verwendet. Die Sequence-Aktivität ist eine zusammengesetzte Aktivität aus der Windows Workflow Foundation und erlaubt es dem Workflow-Designer, weitere Aktivitäten in ihr zu platzieren. Für den Workflow-Designer präsentiert sich die TimeLimitedSequence-Aktivität im Visual Studio dann, wie rechts in Abbildung 12 dargestellt, als zusammengesetzte Aktivität, in die er weitere Aktivitäten hinzufügen kann.

4.5 Workflow-Engine

Hauptaufgabe einer Workflow-Engine ist die Ausführung und Koordination laufender Workflow-Instanzen. Die Windows Workflow Foundation stellt selbst eine Klasse namens "WorkflowRuntime" zur Verfügung, die in eigene Applikationen integriert werden kann und als Workflow-Engine dient. In enablerWorkflows wird diese Runtime beim ersten Aufruf initialisiert und gestartet. Um ein mehrfaches Starten zu verhindern, wurde WorkflowRuntime als Singleton abgebildet (siehe [GHJV95]). Weiters stellt die Workflow Foundation eine Reihe an Services zur Verfügung, die zur Runtime hinzugefügt werden können und jeweils spezifische Aufgaben übernehmen. In enablerWorkflows wurden daraus folgende Services ausgewählt:

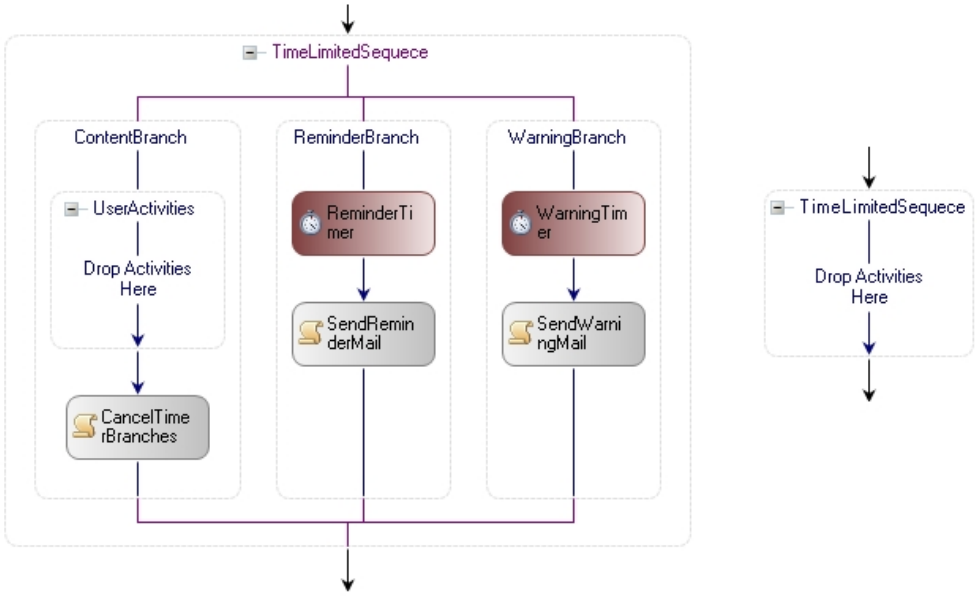

Abbildung 12: Links: Schematische Darstellung der TimeLimitedSequence-Aktivität; Rechts: Darstellung im Workflow-Designer

- SqlWorkflowPersistenceService: Legt alle Workflow-Instanzen, die sich im Status "suspendiert" befinden, also gerade keine Arbeit verrichten, in einer SQL-Datenbank ab. Spätestens beim Stoppen der Runtime, zum Beispiel, wenn die Applikation enabler4BIZ beendet wird, werden alle aktiven Workflows in die Datenbank persistiert. Dies garantiert, dass keine Workflows verloren gehen und diese beim nächsten Start der Runtime ebenfalls wieder gestartet werden.

- SQLTrackingService: Sorgt dafür, dass alle Ereignisse beim Ablauf eines Workflows (z. B.: Starten und Stoppen von Workflows und Aktivitäten) in einer SQL-Datenbank mitprotokolliert werden. Dieses Protokoll wird von enablerWorkflows den Administratoren im Userinterface zur Verfügung gestellt.

- ManualWorkflowSchedulerService: Im Gegensatz zu Client-Applikationen oder Service-Applikationen "läuft" eine Web-Applikation wie der enabler4BIZ normalerweise nur dann, wenn ein Benutzer durch den

Aufruf einer Web-Seite einen Request startet. Um zu gewährleisten, dass Workflows auch weiterlaufen, wenn gerade kein Benutzer mit der Web-Applikation interagiert, muss ein manueller Scheduler aktiviert werden, der in diesem Service integriert ist.

Die laufende Workflow-Runtime stellt unter Anderem Methoden zum Erstellen und Starten von Workflow-Instanzen zur Verfügung.

4.6 Userinterface

Die Aufgabe des Userinterface ist die Visualisierung der Benutzeroberfläche von enablerWorkflows. Dazu werden ASP.Net-Seiten verwendet, die auf einem Webserver ausgeführt werden. Als Webserver kommt der Microsoft Internet Information Server (IIS) zum Einsatz. Damit die Benutzeroberfläche in enabler4BIZ integriert werden kann, werden Module benötigt (vgl [Win04]). Ein Modul ist ein benutzerdefiniertes Websteuerelement, das eine bestimmte Aufgabe erfüllt. Es wird durch Ableiten von der Klasse EnablerModule definiert. EnablerModule ist im weitesten Sinn von System.Web.UI.UserControl abgeleitet. Damit der enabler4BIZ neue Module laden kann, werden diese als Assembly installiert. Alle darin enthaltenen Module können anschließend über die Administrationsseite des enabler4BIZ in das Portal eingebunden werden. Es muss sowohl das Assembly installiert werden, als auch die Websteuerelement-Dateien (*.ascx). Im Folgenden werden alle neuen Module von enablerWorkflows im Detail erläutert.

4.6.1 Modul "Vorlagen-Administration"

Dieses Modul ist, im Gegensatz zu allen anderen Modulen, fix ins Portal eingebunden und auch nur einmal im enabler4BIZ vorhanden. Es handelt sich um die zentrale Administration für Workflow-Vorlagen und ermöglicht es Administratoren, neue Workflow-Vorlagen zu erstellen, diese zu bearbeiten, mit Berechtigungen zu versehen (siehe 4.10.2) und den Benutzern im enabler4BIZ zur Verfügung zu stellen. Als Basis für neue Vorlagen werden dem Administrator alle aus dem Workflow-Designer importierten Workflow-Definitionen

angeboten. Es ist auch möglich, auf Basis einer Definition mehrere verschiedene Vorlagen zu erstellen, zum Beispiel für verschiedene Benutzergruppen.

Weiters können hier die Vorgabewerte der dafür vorgesehenen Workflow-Parameter gesetzt werden. Dafür werden dynamisch alle Parameter der Workflow-Definition, die vom Administrator zu setzen sind (Parameter FillOutMode = FillOutForTemplate, siehe 4.7.4), ausgelesen und in entsprechenden Controls dargestellt: Für Text-Parameter sind das Textboxen, für eine Benutzerzuordnung, wie zum Beispiel beim Parameter "FillingUser" der Aktivität "FillOutNew_User", ist dies eine Auswahlliste mit allen Benutzern aus dem enabler4BIZ, usw. .

Nicht mehr benötigte Vorlagen können archiviert werden und stehen dann den Benutzern nicht mehr zur Verfügung. Bereits laufende Workflow-Instanzen aus einer archivierten Vorlage können jedoch noch bis zum Ende weiterlaufen.

4.6.2 Modul "Workflow Starten"

Dieses Modul listet für den Benutzer Workflow-Vorlagen auf, aus denen er berechtigt ist, einen Workflow zu starten. Das Modul kann vom Portal-Administrator mehrfach im enabler4BIZ auf verschiedenen Seiten eingebunden werden. Über Settings kann ausgewählt werden, ob alle Workflow-Vorlagen angezeigt werden sollen, oder nur eine bestimmte Vorlage. Dadurch ist es zum Beispiel möglich, eine Seite mit dem Namen "Verbesserungsvorschlag einbringen" im Portal einzubinden, auf der nur die Vorlage für Verbesserungsvorschläge zu finden ist.

Der Workflow selbst wird durch Klick auf einen entsprechenden Button gestartet. Sollte es Parameter in der Workflow-Definition geben, die vom Ersteller des Workflows zu befüllen sind (Parameter FillOutMode = FillOutOnWorkflowStart, siehe 4.7.4), so werden diese vor dem eigentlichen Start des Workflows noch vom Benutzer abgefragt. Dies geschieht auf die gleiche Art wie bei der Parametereingabe bei Workflow-Vorlagen (siehe 4.6.1).

Ist der Benutzer, der den Workflow gestartet hat, selbst als Ausfüller im ersten Benutzerinteraktionsschritt des Workflows eingetragen, so öffnet sich

automatisch die Bearbeitungsansicht des Workflows, in der das erste Formular zum Ausfüllen angezeigt wird (siehe 4.6.3).

4.6.3 Modul "Aufgabenliste"

Das Modul "Aufgabenliste" listet alle laufenden Workflows auf, in denen dem aktuellen Benutzer eine Aufgabe zugeteilt wurde. Bei jedem gelisteten Workflow werden unter anderem der aktuelle Schritt und der Geschäftsstatus, in dem sich der Workflow gerade befindet, angezeigt. Weiters wird der Benutzer informiert, ob er persönlich die Aufgabe zugeteilt bekommen hat, oder diese auf Grund einer seiner Rollen im System gelistet wird (siehe 4.3.2, Aktivität "FillOutNew_Role").

Auch hier kann über Settings festgelegt werden, ob alle Workflows aufgelistet werden sollen, oder nur Workflows, die aus einer bestimmten Vorlage heraus erstellt wurden. Im zweiten Fall ist es zusätzlich möglich, die Auswahl auf einzelne Aktivitäten des Workflows einzuschränken. Auf diese Weise ist es zum Beispiel möglich, eine Seite namens "eingebrachte Verbesserungsvorschläge" im Portal einzubinden, auf der nur Workflows angezeigt werden, die aus der Vorlage "Verbesserungsvorschlag" erstellt wurden und sich im Schritt "Umsetzungprüfung" befinden.

Wählt man einen der gelisteten Workflows aus, öffnet sich die Bearbeitungsansicht des Workflows: In dieser wird das Workflow-Dokument angezeigt, also alle bereits ausgefüllten Formulare (abhängig von der gewählten Ansichtsoption, siehe 4.1.3: WP_UserInteractionMode). Muss der Benutzer ein neues Formular ausfüllen oder ein bestehendes überarbeiten, so wird dieses Formular in der Edit-Ansicht, also mit editierbaren Eingabefeldern, angezeigt. Der Benutzer schließt die Bearbeitung der Aufgabe ab, in dem er nach dem Ausfüllen des Formulars seine Eingaben über einen Button bestätigt. Der Name des Buttons ist in der Workflow-Vorlage frei definierbar und könnte zum Beispiel beim Schritt "Umsetzungprüfung" "Geprüft" heißen. Alternativ können die Eingaben zwischengespeichert und zu einem späteren Zeitpunkt vervollständigt und bestätigt werden. Nach Abschluss und Bestätigung der Eingaben wird der Workflow mit der nächsten Aktivität fortgesetzt.

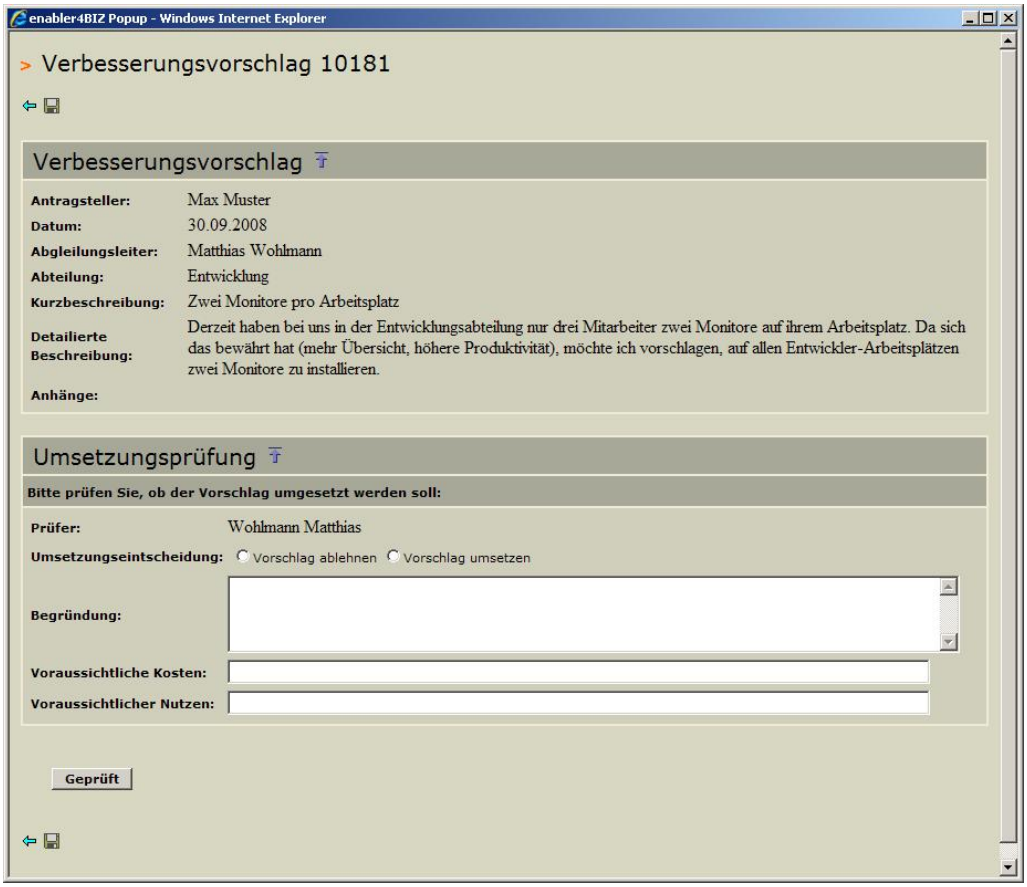

Abbildung 13: Screenshot: Verbesserungsvorschlag prüfen

Abbildung 13 zeigt einen Screenshot für die Umsetzungsprüfung eines Verbesserungsvorschlags. Man sieht das bereits ausgefüllte Formular mit dem Vorschlag und das leere Formular zur Umsetzungsprüfung, das vom Abteilungsleiter auszufüllen ist.

4.6.4 Modul "Workflow-Übersicht" beziehungsweise "Workflow--Dokument-Übersicht"

Dieses Modul dient vor allem zur Übersicht für Administratoren des Systems. Über ein Setting kann definiert werden, ob das Modul als Workflow-Übersicht oder als Übersicht über alle Workflow-Dokumente erscheinen soll. Angezeigt werden dann alle laufenden und beendeten Workflows oder die Workflow-Dokumente, die aus diesen Workflows entstanden sind. Weiters kann konfiguriert werden, welche Informationen angezeigt werden sollen. Zur Auswahl stehen hier Felder wie Workflowname, Geschäftsstatus, Ersteller, aktueller Schritt oder das Feld "Wartet auf", das bei laufenden Workflows zeigt, auf wessen Eingabe der Workflow gerade wartet. Wie auch bei anderen Modulen kann weiters eingestellt werden, ob alle Workflows/Dokumente angezeigt werden sollen oder nur diejenigen auf Basis der ausgewählten Vorlage. In letzterem Fall stehen zusätzlich alle Felder aus den involvierten Formularen als Anzeigemöglichkeit zur Auswahl. So könnte zum Beispiel bei einer Übersicht über alle Verbesserungsvorschläge das Feld "Kurzbeschreibung" des Vorschlags in der Übersicht angezeigt werden.

Für jeden gelisteten Workflow/jedes gelistete Dokument stehen mehrere mögliche Aktionen zur Verfügung: Das Workflow-Dokument kann angezeigt und, sofern man die Berechtigung dazu hat, nachträglich geändert werden. Weiters können beendete Workflows archiviert werden, sie scheinen dann nicht mehr in der Liste auf.

Wird das Modul als Workflow-Übersicht verwendet, gibt es weiters eine Detailansicht zum Workflow, die neben anderen Informationen die grafische Ansicht des Workflows aus dem Designer des Visual Studios als Image zeigt. Ist der Workflow noch nicht beendet, wird die Aktivität, in der sich der Workflow gerade befindet, markiert. Weiters wird in dieser Detailansicht die vom

SQLTrackingService (siehe 4.5) mitprotokollierte Information angezeigt, also welche Aktivität wann gestartet und beendet wurde.

In der Workflow-Übersicht kann der aktuelle Benutzer weiters, sofern er die Berechtigung dazu hat, laufende Workflows abbrechen. Dabei werden ebenfalls alle noch offenen Aufgaben, die von diesem Workflow erstellt wurden, storniert. Der Workflow wird beendet und erhält den Status "Abgebrochen".

4.7 Die Parameterübergabe

Zur Anpassung von Workflows an den abzubildenden Geschäftsprozess stellen die einzelnen unter 4.3 beschriebenen Aktivitäten Parameter zur Verfügung, über die sie konfiguriert werden können. So benötigt zum Beispiel die Aktivität "FillOutNew_User" unter anderem zwei Parameter für

- das auszufüllende Formular und
- den User, der das Formular ausfüllen muss.

Die Werte der Parameter müssen spätestens beim Start der Aktivität innerhalb der Ausführung einer Workflow-Instanz bekannt sein. In diesem Abschnitt wird beschrieben, zu welchen Zeitpunkten und auf welche Weise es möglich ist, die Werte der Parameter innerhalb von enablerWorkflows festzulegen.

Dazu werden folgende vier Bezeichnungen für Personen(gruppen) verwendet:

- Ersteller der Workflow-Definition: Workflow-Designer oder Entwickler, gegebenenfalls mit Programmierkenntnissen, der eine Workflow-Definition im Visual-Studio mit Hilfe des grafischen Designers erstellt.
- Workflow-Administrator: Benutzer im enabler4BIZ mit Administrationsrechten für die Komponente "enablerWorkflows". Er darf unter anderem Workflow-Vorlagen auf Basis einer Workflow-Definition erstellen, diese konfigurieren und sie den übrigen Benutzern zum Starten zur Verfügung stellen.

- Workflow-Ersteller: Benutzer, der eine neue Workflow-Instanz aus einer Workflow-Vorlage heraus erstellt und startet.
- Involvierter Benutzer: Benutzer, der in einem Workflow-Schritt beteiligt ist und als solcher einen Formularteil ausfüllen oder überarbeiten muss.

4.7.1 Mögliche Zeitpunkte der Parametersetzung

Im Zuge der Analyse hat sich herausgestellt, dass es sinnvoll und notwendig sein kann, die Werte von Parametern zu folgenden Zeitpunkten setzen zu können:

Während des Workflow-Design im Visual Studio: Während der Erstellung der Workflow-Definition im Designer des Visual Studios kann der Ersteller der Workflow-Definition einige Parameter der verwendeten Aktivitäten gleich an Ort und Stelle setzen. Das Problem dabei ist allerdings, dass zu diesem Zeitpunkt kein Zugriff auf die Benutzer, Rollen, und Formulardefinitionen des enabler4BIZ besteht. Somit könnten statt der IDs der entsprechenden Objekte nur die Namen der Objekte gesetzt werden. Rollen und Usernamen können aber im enabler4BIZ jederzeit von Administratoren geändert werden. Nur Formulardefinitionen haben einen fixen internen Namen, der hier verwendet werden kann. Sinnvoll ist dieser Zeitpunkt der Parametersetzung vor allem für Parameter, deren Wert sich voraussichtlich selten oder nie ändert. Bei der Aktivität „FillOutNew_User“ ist dies zum Beispiel der Parameter „FormName“.

Zeitpunkt der Konfiguration der Workflow-Vorlage im enabler4BIZ: Workflow-Definitionen werden vom Workflow-Administrator den Benutzern in Form von Workflow-Vorlagen zur Verfügung gestellt. Dies geschieht durch Integration der Workflow-Definition in den enabler4BIZ und durch Verknüpfung der daraus entstehenden Workflow-Vorlage mit Rollen (Benutzer, die einer dieser Rollen angehören, haben das Recht, aus dieser Vorlage einen neuen

Workflow zu starten (siehe 4.6.1, Vorlagenadministration).). Zu diesem Zeitpunkt können auch Parameterwerte definiert werden. Die Werte gelten dann für alle Workflows, die aus dieser Vorlage gestartet werden. Die Parameter der Vorlage können natürlich vom Administrator jederzeit geändert werden. Die Änderungen wirken sich allerdings nur auf neu gestartete Workflow aus, Parameterwerte von bereits laufenden Workflows werden dadurch nicht geändert. Zum Beispiel könnte hier bei einer "FillOutNew_User"-Aktivität eine Person ausgewählt werden, die alle Urlaubsanträge bearbeiten muss.

Beim Start eines Workflows: Hier kann der Workflow-Ersteller Werte weiterer Parameter eingeben, bevor der Workflow tatsächlich gestartet wird. Dies kann zum Beispiel sinnvoll sein, um vor dem Start eines Urlaubsansuchen-Workflows den Vorgesetzten auszuwählen, an den das Urlaubsansuchen gerichtet wird. Damit ist es möglich, verschiedene Urlaubsansuchen von verschiedenen Personen bearbeiten zu lassen.

Zur Laufzeit von Workflows: Parameterwerte einer Aktivität werden von einer Vorgänger-Aktivität zur Laufzeit des Workflows gesetzt. Zum Beispiel könnte in einem Workflow-Schritt der ausfüllende Benutzer für den nächsten Schritt bestimmt werden.

4.7.2 In der Workflow Foundation vorgesehene Varianten der Parametersetzung

Die MS Windows Workflow Foundation selbst bietet drei Varianten der Parametersetzung von Aktivitäten an:

Im Workflow-Designer: Im Workflow-Designer des Visual Studios können die Parameterwerte der Aktivitäten direkt gesetzt werden (ähnlich den Eigenschaften bei Web-Controls). Die gesetzten Werte werden beim Compilieren fix ins Assembly integriert.

Übergabe der Parameterwerte bei Erstellung einer Workflow-Instanz: Die in der Basisklasse aller Workflows enthaltene Methode "Create()", mittels der aus einer Workflow-Vorlage eine neue Workflow-Instanz erstellt wird, akzeptiert ein Dictionary als Parameter: Hier kann noch vor dem Start des Workflows eine Liste an Key-Value Pairs übergeben werden. Der Key ist in diesem Fall ein String. Existiert in der Workflow-Definition ein Property (=Parameter) mit dem gleichen Namen (Name-Matching), wird der übergebene Value automatisch diesem Parameter zugewiesen. Standardmäßig erstellt man also in der Workflow-Vorlage neue Properties mit sprechenden Namen und bindet diese an die Parameter der einzelnen Aktivitäten. Der Vorgang der Erstellung solcher Properties wird auch vom Designer gut unterstützt. Damit wird beim Setzen der Properties in der Workflow-Vorlage indirekt auch das Property der entsprechenden Aktivität gesetzt.

Innerhalb eines Workflows zur Laufzeit: Die oben beschriebenen Properties des Workflows können natürlich auch zur Laufzeit des Workflows, zum Beispiel in der Execute-Methode einer Aktivität, gesetzt oder geändert werden. Auch können hier die Properties anderer Aktivitäten geändert werden, sofern der Name der entsprechenden Aktivität bekannt ist.

4.7.3 In enablerWorkflows umgesetzte und zur Verfügung gestellte Varianten der Parametersetzung

Folgende drei Varianten der Parametersetzung werden von enablerWorkflows zur Verfügung gestellt:

Parameter zur Design-Zeit: Für Parameter, die bereits zum Design-Zeitpunkt der Workflow-Definition vom Ersteller gesetzt werden können, kann die von der Workflow Foundation unter 4.7.2 ("Im Workflow-Designer") beschriebene Variante problemlos verwendet werden. Sinnvoll ist das zum Beispiel für den Parameter „FormName", dessen Name ja schon zu Design-Zeit feststeht.

Konfiguration der Workflow-Vorlage und Start des Workflows: Für Parameter, die vom Workflow-Administrator während der Konfiguration der Vorlage oder vom Workflow-Ersteller vor Start eines Workflows im enabler4BIZ gesetzt werden sollen, ist die von der Workflow Foundation zur Verfügung gestellte Variante, die unter 4.7.2 ("Übergabe der Parameterwerte bei Erstellung einer Workflow-Instanz") beschrieben ist, mit einem großen Nachteil verbunden: Um die Parameterübergabe mittels Dictionary zu ermöglichen, müsste der Ersteller der Workflow-Definition für jeden Parameter einer Aktivität, die er dem Workflow-Administrator oder dem Workflow-Ersteller zur Verfügung stellen möchte, ein neues Property in der Workflow-Definition hinzufügen. Weiters wäre eine Unterscheidung nötig, welche Parameter vom Workflow-Administrator auszufüllen sind, und welche vom Workflow-Ersteller. Außerdem sollten Parameter einen sprechenden Namen und eine Beschreibung für diese Benutzer erhalten. Diese drei Eigenschaften müssten zusätzlich, zum Beispiel über Attribute, zu den in der Workflow-Definition erstellten Parametern angegeben werden. Beides (Erstellen der Properties und Setzen der Attribute) würde für den Ersteller der Workflow-Definition Programmierarbeit bedeuten. Um die Erstellung neuer Workflow-Vorlagen möglichst einfach zu gestalten, ist es notwendig, den Programmieraufwand für den Ersteller möglichst gering zu halten oder gänzlich zu umgehen. Daher soll enablerWorkflows hier den Großteil der Parameterübergabe übernehmen, der Ersteller der Workflow-Definition soll hier nicht mit Programmierarbeit belastet werden. Die genaue Lösung hierfür wird unter 4.7.4 beschrieben.

Zur Laufzeit: Diese Variante ermöglicht es, Parameterwerte einer Aktivität über Formularfelder zu setzen, die in einer zuvor ausgeführten Aktivität ausgefüllt wurden. Dazu werden für jeden Parameter zwei Hilfsparameter ("FillingForm" und "FillingField") eingeführt (siehe auch 4.7.4, "Parameterdefinition"). Soll ein Parameter von einem bereits ausgefüllten Formular übernommen werden, so müssen diese beiden Parameter vom Ersteller der Workflow-Definition gesetzt werden: FillingForm enthält dabei den Namen des Formulars und "FillingField" den Namen des Formularfeldes, aus dem der Parameter übernommen werden soll. Wurde dasselbe Formular mehrmals innerhalb des Workflows ausgefüllt, wird der Wert des zuletzt ausgefüllten

Formulars verwendet. Somit ist auch die Parametersetzung zur Laufzeit ohne Programmierarbeit des Erstellers der Workflow-Definition möglich.

Falls Parameter zur Laufzeit nicht direkt eins zu eins aus einem Formularfeld übernommen werden können, sondern vorher abgeändert werden müssen, besteht immer noch die Möglichkeit, dies über eine von der Windows Workflow Foundation zur Verfügung gestellte "Code"-Aktivität zu lösen: Im Code kann natürlich auf das Workflow-Dokument mit allen ausgefüllten Formularfeldern frei zugegriffen werden. Die Parameter der noch nicht gestarteten Aktivitäten können beliebig gesetzt werden. Diese Variante ist sehr flexibel, allerdings erfordert sie Programmierkenntnisse sowie detaillierte Kenntnisse der Klassen, Aktivitäten, Parameter und des Workflow-Dokuments in enablerWorkflows.

4.7.4 Parameterhandling in enablerWorkflows

Die hier vorgestellte Lösung beschreibt, wie in enablerWorkflows Aktivitätsparameter definiert und wie diese gehandhabt werden. Ziel dieser Lösung war es, den Zeitpunkt der Parametersetzung, sowie die Datenquelle, aus der der Parameter übernommen wird, möglichst flexibel zu halten und trotzdem Programmiereingriffe seitens des Workflow-Designers zu vermeiden.

Parameterdefinition: Für jeden Parameter einer Aktivität (= "Core-Parameter"), werden in der Aktivität fünf weitere Hilfsparameter definiert, die spezifizieren, zu welchem Zeitpunkt die Werte der Parameter gesetzt werden beziehungsweise aus welcher Quelle die Daten kommen. Diese Hilfsparameter sind vom Ersteller der Workflow-Definition zum Design-Zeitpunkt als Properties im Visual Studio auszufüllen. Die Hilfsparameter sind (<Parameter> wird immer durch den Namen des zugehörigen Core-Parameters ersetzt):

- FillOutMode_<Parameter>: Folgende Modi stehen zur Auswahl:
  - Fixed: Der Wert ist bereits zum Design-Zeitpunkt fixiert (im Property-Fenster des Visual Studios) und kann im enabler4BIZ nicht mehr geändert werden.

  - FillOutForTemplate_NoOverride: Vom Workflow-Administrator für die Workflow-Vorlage im enabler4BIZ auszufüllen. Der Wert gilt dann für alle Instanzen, die von dieser Vorlage gestartet werden.
  - FillOutForTemplate_OverrideOnStart: Wie FillOutForTemplate_NoOverride, Wert kann jedoch vom Workflow-Ersteller für den erstellten Workflow überschrieben werden.
  - FillOutOnWorkflowStart: Vom Workflow-Ersteller vor dem Start eines Workflows auszufüllen. Der Wert gilt nur für diese Instanz.
  - AutoFillWithStartingUser: Steht nur für den Parameter zur Verfügung, die auf Benutzer referenzieren. Bei Workflow-Start wird automatisch der Benutzer eingesetzt, der den Workflow gestartet hat.
  - AutoFillFromFormField: Der Wert des Parameters wird zum Startzeitpunkt der Aktivität aus einem bereits zuvor in diesem Workflow ausgefüllten Formularteil übernommen. Welches Formular und welches Formularfeld das ist, wird durch die Parameter "FillingForm_<Parameter>" und "FillingField_<Parameter>" spezifiziert.
  - FilledByCode: Markiert einen Parameter, der erst im Laufe des Workflows durch eine Code-Aktivität ausgefüllt wird. Die Code-Aktivität muss vom Ersteller der Workflow-Definition selbst erstellt und programmiert werden.

- DisplayName_<Parameter>: Text, der dem Benutzer als Name des Parameters im Userinterface des enabler4BIZ angezeigt wird. Der Name muss nur ausgefüllt werden, wenn einer der drei FillOut-Modi verwendet wird, bei denen ein Benutzer im enabler4BIZ den Wert des Parameters setzt.

- Description_<Parameter>: Text, der dem Benutzer als Beschreibung des Parameters angezeigt wird. Ebenfalls nur dann auszufüllen, wenn der Parameter im enabler4BIZ auszufüllen ist.

Abbildung 14: Core- und Hilfsparameter der Aktivität "FillOutNew_User"

- FillingForm_<Parameter>: Nur nötig für den FillOut-Modus "AutoFillFromFormField" und gibt in diesem Fall das Formular an, aus dem der Wert übernommen werden soll.
- FillingField_<Parameter>: Nur nötig für den FillOut-Modus "AutoFillFromFormField" und gibt in diesem Fall das Formularfeld an, aus dem der Wert übernommen werden soll.

Abbildung 14 zeigt die zwei Core-Parameter "FillingUser" und "FormName" der Aktivität "FillOutNew_User", sowie die Hilfsparameter des Core-Parameters "FillingUser" im Property-Fenster des Visual Studios. In diesem Fall wird der Workflow-Ersteller beim Workflow-Start als ausfüllender Benutzer gesetzt (FillOutMode = "AutoFillWithStartingUser"), was bedeutet, dass der Workflow-Ersteller selbst das erste Formular dieses Workflows, "Verbesserungsvorschlag", auszufüllen hat.

Parameter-Eingabe: Parameter, die mit dem FillOut-Modus "FillOutForTemplate_NoOverride" oder "FillOutForTemplate_OverrideOnStart" markiert sind, werden dem Administrator zum Zeitpunkt der Konfiguration der Workflow-Vorlage angezeigt, müssen von diesem befüllt werden und werden

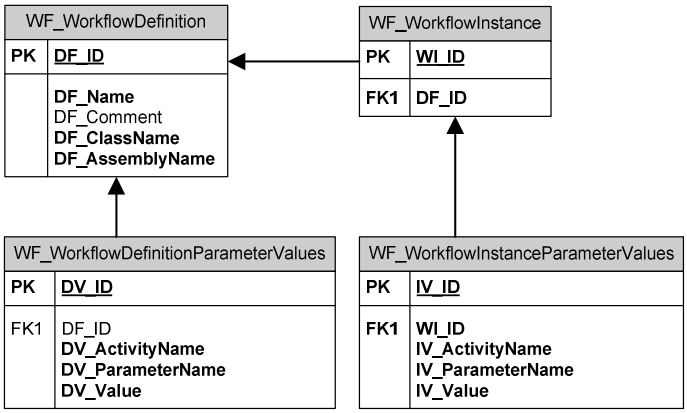

Abbildung 15: Datenbankschema für Aktivitätsparameter (Die Tabellen für Workflow-Definitionen und -Instanzen sind vereinfacht dargestellt)

in der Datenbank in der Tabelle *WF_ WorkflowDefinitionParameterValues* mit einer Referenz auf die Workflow-Vorlage (Spalte *WD_ ID*) abgelegt.

Parameter, die mit FillOut-Modus "FillOutForTemplate_OverrideOnStart" oder "FillOutOnWorkflowStart" markiert sind, werden dem Ersteller des Workflows vor dem Workflow-Start angezeigt und sind von diesem auszufüllen. Dabei werden beim Modus "FillOutForTemplate_OverrideOnStart" die vom Administrator vorgeschlagenen Werte angezeigt, können aber überschrieben werden. Die Werte werden in der Datenbank in der Tabelle *WF_ WorkflowInctanceParameterValues* unter einer gemeinsamen ID der Workflow-Instanz (Spalte *WI_ ID*) abgelegt.

In Abbildung 15 wird das Datenbankschema für die Tabellen *WF_ WorkflowDefinitionParameterValues* und *WF_ WorkflowInctanceParameterValues* dargestellt. Für Parameter von Workflow-Vorlagen (Tabelle *WF_ WorkflowDefinition*) werden die vom Administrator eingegebenen Werte der Parameter in der Spalte *DV_ Value* als Text gespeichert. *DV_ ActivityName* gibt die Aktivität an, für den der Parameter gilt. *DV_ ParameterName* gibt den Namen

des Parameters innerhalb der angegebenen Aktivität an. Analog dazu werden die Parameterwerte für Workflow-Instanzen in der Tabelle *WF_ WorkflowInctanceParameterValues* gespeichert.

Werte setzen: siehe auch 4.2.2 (EnablerParameteredActivty)

Das eigentliche Setzen der Parameterwerte der Aktivitäten passiert zum Startzeitpunkt der Aktivitäten. Die in der Klasse *EnablerParameteredActivty* überschriebene Methode *SetParameters()* (siehe auch 4.2.2, EnablerParameteredActivty), die diese Aufgabe übernimmt, ist in Listing 2 dargestellt und kommentiert. Vereinfacht wird hier, je nach FillOut-Modus, die entsprechende Datenquelle gewählt, der zu setzende Wert ausgelesen und danach der Parameterwert in der Methode *SetParameter()* (hier nicht dargestellt) mittels Reflection gesetzt.

Anmerkung zum FillOut-Modus "AutoFillFromFormField": Hier werden die bereits ausgefüllten Formulare in umgekehrter Reihenfolge durchsucht (Property *CompletedPartsReverse* im Code). Das bedeutet, dass der Wert aus dem zuletzt ausgefüllten Formular übernommen wird, falls Formulare mit demselben Namen innerhalb des Workflows bereits mehrfach ausgefüllt wurden.

4.8 Deployment neuer Workflow-Definitionen

Neue Workflow-Definitionen für enablerWorkflows werden im grafischen Designer der Entwicklungsumgebung (Microsoft Visual Studio) erstellt. Dazu wird für jede Workflow-Definition ein eigenes Softwareprojekt vom Typ "Klassenbibliothek" im Visual Studio angelegt, in dem die Workflow-Definition erstellt wird. Durch dieses Vorgehen wird dafür gesorgt, dass durch die Compilierung für jede Workflow-Definition einer eigenen Assembly (DLL) erstellt wird, die im Wesentlichen die Workflow-Definition enthält. Um eine solche Workflow-Definition in der Web-Applikation "enabler4BIZ" zu veröffentlichen, wird die entsprechende DLL in das bin-Verzeichnis der Web-Applikation kopiert.

Listing 2 Die Methode “SetParameters” (vereinfacht)

```
protected override void SetParameters() {
  //Referenz auf Workflow-Instanz holen. Die Instanz enthält zu jeder
  //Aktivität die gespeicherten Parameterwerte aus der Tabelle
  //"WF_WorkflowInctanceParameterValues"
  EnablerWorkflowInstance workflowInstance = this.WorkflowInstance_Volatile;
  //Referenz auf zugehöriges Workflow-Dokument holen
  WorkflowDocument document = this.Document_Volatile;

  //Daten zur aktuellen Aktivität finden
  WorkflowActivity enaActivity = workflowInstance.GetActivity(this.Name);
  //Aktivitätstyp auslesen
  Type activityType = this.GetType();
  //alle Parameter der Aktivität durchgehen und Werte setzen
  foreach (WorkflowParameter parameter in enaActivity.Parameters) {
    if (parameter.FillOutMode == FillOutMode.AutoFillWithStartingUser) {
      //Modus==AutoFillWithStartingUser -->
      //Workflow-Ersteller als Parameterwert setzen
      this.SetParameter(activityType, parameter.Name,
          workflowInstance.WorkflowCreator.ID);
    } else if (parameter.FillOutMode == FillOutMode.FillOutForTemplate_NoOverride
        || parameter.FillOutMode == FillOutMode.FillOutForTemplate_OverrideOnStart
        || parameter.FillOutMode == FillOutMode.FillOutOnWorkflowStart
        || parameter.FillOutMode == FillOutMode.Fixed) {
      //Parameter wurde im enabler4BIZ vor Workflow-Start gesetzt -->
      //Wert aus gespeicherter Parametertabelle lesen
      this.SetParameter(activityType, parameter.Name, parameter.GetValueObject());
    } else if (parameter.FillOutMode == FillOutMode.AutoFillFromFormField) {
      //Parameter kommt aus bereits ausgefülltem Workflow-Dokument -->
      //entsprechendes Formularfeld suchen und diesen Wert setzen

      //Alle ausgefüllten Formularteile durchsuchen
      foreach (WorkflowDocumentPart part in document.CompletedPartsReverse) {
        if (part.Form.FormDefinition.InternalName == parameter.FillingForm) {
          //Formular mit dem in FillingForm angegebenen Namen gefunden -->
          //Formularfeld mit dem Namen aus FillingField suchen
          FilledFormElement element = part.Form.GetElement(parameter.FillingField);
          //Eingegebenen Wert des Formularfeldes lesen
          object value = element.GetData();
          //und diesen Wert für den Parameter setzen
          this.SetParameter(activityType, parameter.Name, value);
          break;
        }
      }
    }
  }
}
```

Dem enabler4BIZ selbst wird die neue DLL über ein Konfigurationssetting namens „WorkflowAssemblies“ bekannt gegeben. Das Setting enthält eine Strichpunkt getrennte Liste aller Assembly-Namen, in denen sich Workflows befinden. Nach der Bekanntgabe kann im enabler4BIZ eine neue Workflow-Vorlage erstellt werden, die auf der Workflow-Definition in der DLL basiert. Beim Erstellen neuer Vorlagen werden in einer Auswahlliste alle Workflow-Definitionen zur Verfügung gestellt, die in den angegebenen Assemblies gefunden wurden (siehe 4.6.1, Vorlagen-Administration).

4.9 Änderungen/Anpassungen in Workflow-Definitionen

Während des Einsatzes einer Workflow-Definition in enablerWorkflows kann es im Lauf der Zeit zu Änderungsanforderungen kommen, zum Beispiel auf Grund von Änderungen in den zugrundeliegenden Geschäftsprozessen, in deren Folge der abbildende Workflow ebenfalls angepasst werden muss. In [Cas98, DRSV03] werden zwei Arten von Workflow-Änderungen beschrieben:

- Dynamische Workflow-Evolution (Ad-hoc Änderungen): Die Änderungen betreffen nur eine oder mehrere Workflow-Instanzen, nicht aber die Workflow-Definition selbst. Die Änderung ist das Ergebnis eines Fehlers, eines seltenen Ereignisses, oder eines speziellen Bedarfs beim Kunden und gilt individuell für eine Instanz. In dieser Version von enablerWorkflows werden Ad-hoc-Änderungen von Workflows nicht unterstützt. Allerdings lässt die zugrundeliegende Windows Workflow Foundation solche Änderungen an Workflow-Instanzen zu und stellt dafür Methoden zum Hinzufügen und Entfernen von Aktivitäten in Workflows bereit. Damit besteht die Möglichkeit, bei Bedarf die Unterstützung für Ad-hoc-Workflow-Änderungen in enablerWorkflows in einer zukünftigen Version zu integrieren.

- Statische Workflow-Evolution: Die Änderungen sind von struktureller Art. Die Änderung der Workflow-Definition gilt ab einem bestimmten Zeitpunkt für alle neuen Workflow-Instanzen. Die Änderung kann das Ergebnis einer neuen Geschäftsstrategie, die Folge von Umstrukturierungen oder Änderungen von äußeren Bedingungen (zum Beispiel

Gesetzesänderungen) sein. Einen Spezialfall stellen Änderungen dar, die zusätzlich für alle bereits laufenden Workflow-Instanzen zu berücksichtigen sind. Hier stellt sich das Problem, festzustellen, ob und wie ein laufender Workflow in die neue Workflow-Definition übergeführt werden kann. In [RRD02] wird beschrieben, welche Verträglichkeitsprüfungen hierfür durchgeführt werden müssen und, wie eine automatische Migration implementiert werden kann. In enablerWorkflows gibt es in dieser Version keine Unterstützung für die Migration laufender Workflows. Diese müssen entweder abgebrochen werden oder können auf Basis der alten Definition bis zum Ende weiterlaufen. Eine parallele Ausführung von Workflows nach der alten Definition bis zu deren Beendigung und Workflows, die auf Basis der neuen Definition erstellt wurden, ist allerdings möglich.

In enablerWorkflows werden derzeit nur statische Änderungen unterstützt. Folgende Änderungen sind dabei möglich und müssen unterschiedlich gehandhabt werden:

- Änderung der Felder in einem eingebundenen Formular:
  - Neue Felder kommen hinzu.
  - Bestehende Felder ändern sich.
  - Felder sollen gelöscht werden.
- Änderung der Parameterwerte von Aktivitäten: Unterscheidung von
  - Parametern, die im enabler4BIZ gesetzt werden können und
  - Parametern, die von der Workflow-Definition vorgegeben sind und im enabler4BIZ nicht geändert werden können
- Änderung der Struktur der Workflow-Definition:
  - Neue Aktivitäten kommen hinzu.
  - Bestehende Aktivitäten sollen gelöscht werden.

– Die Position von Aktivitäten innerhalb des Workflows soll geändert werden.

Im Folgenden werden die Änderungen im Detail erklärt und die Durchführung der Änderung im Rahmen von enablerWorkflows erläutert.

4.9.1 Änderungen im Formular

Änderungen an der Formulardefinition können jederzeit im enabler4BIZ durchgeführt werden, wirken sich aber unmittelbar auf ALLE Workflows aus, auch auf bereits beendete. Wird zum Beispiel ein Formularelement entfernt, so ist dieses Formularelement auch in der Ansicht bereits beendeter Workflows nicht mehr vorhanden. Neue Formularfelder werden auch bei beendeten Workflows eingefügt und angezeigt, allerdings mit leerem Inhalt. Sollen die bereits eingetragenen Daten von beendeten Workflows unverändert beibehalten werden, muss die Formulardefinition kopiert und unter einem neuen Namen gespeichert werden. In der Folge muss die Workflow-Definition verändert werden, damit das neue Formular verwendet wird (siehe 4.9.3).

4.9.2 Änderung der Parameterwerte von Aktivitäten

Parameter, die im enabler4BIZ gesetzt werden können: Parameter, die im enabler4BIZ für Workflow-Vorlagen gesetzt werden können, können jederzeit verändert werden. Eine solche Änderung betrifft alle danach aus dieser Vorlage gestarteten Workflows. Sie wirkt sich nicht auf bereits davor gestartete Workflows aus, auch wenn der entsprechende Schritt innerhalb des Workflows noch nicht erreicht wurde.

Parameter, die in der Workflow-Definition gesetzt wurden und im enabler4BIZ nicht geändert werden können: Solche Parameteränderungen bedürfen der Erstellung einer neuen Workflow-Definition, wie unter 4.9.3 beschrieben. In dieser neuen Workflow-Definition können die Parameter dann entsprechend geändert werden.

4.9.3 Änderung der Struktur der Workflow-Definition

Soll die Workflow-Definition selbst geändert werden, so geschieht dies durch Erstellen einer Kopie der bestehenden Workflow-Definition. Diese Vorgehensweise ist nötig, wenn

- neue Aktivitäten hinzukommen,
- Aktivitäten an eine andere Position im Workflow verschoben werden,
- Aktivitäten gelöscht werden oder
- Parameter von Aktivitäten in der Workflow-Definition geändert werden (z.B. wenn das auszufüllende Formular ausgetauscht wird).

Dazu wird der bestehende Workflow im Visual Studio kopiert und der Name des Workflows sowie der zugehörige Klassenname in den Code-Files geändert (Zum Beispiel kann ein "_v2" angehängt werden für die zweite Version dieses Workflows.). Danach werden die gewünschten Änderungen an dieser Kopie durchgeführt. Anschließend kann die DLL, die jetzt beide Versionen dieser Workflow-Definition enthält, wie unter 4.8 beschrieben, beim Kunden installiert werden. Danach wird in der Workflow-Administration im enabler4BIZ bei der entsprechenden Workflow-Vorlage die neue Workflow-Definition ausgewählt (siehe 4.6.1, Vorlagen-Administration). Dabei werden die bereits für die alte Definition gesetzten Parameter soweit möglich übernommen. Dies geschieht anhand einer Überprüfung der Parameter auf Namensgleichheit. Neue Workflow-Instanzen werden auf Basis der neuen Definition gestartet, bestehende Workflows laufen auf Basis der alten Definition ohne Beeinträchtigung weiter.

4.10 Berechtigungen

In [SCFY96] wird ein rollenbasiertes Berechtigungssystem vorgestellt, in dem Rollen als Konstrukt gesehen werden, denen auf der einen Seite Benutzer und auf der anderen Seite Berechtigungen im System zugeordnet werden.

Der enabler4BIZ hat bereits eine Benutzer- und Rollenverwaltung, wobei Rollen frei definierbar sind und jedem Benutzer beliebig vielen Rollen zugeordnet werden können. Diesen Rollen können an diversen Stellen im enabler4BIZ Berechtigungen zugeteilt werden. In diesem Kapitel werden die neuen Berechtigungen, die im Zuge der Integration von enablerWorkflows in den enabler4BIZ hinzugekommen sind, eingeführt.

4.10.1 Globale Berechtigungen

Im enabler4BIZ gibt es eine globale Berechtigungsverwaltung, in der den einzelnen Rollen diverse Berechtigungen im enabler4BIZ zugeordnet werden können. Von enablerWorkflows werden hier folgende neue Berechtigungen eingeführt:

- Workflow-Vorlagen konfigurieren: Neue Vorlagen erstellen und diese konfigurieren.
- Ablaufverfolgung Workflows sehen: In der Workflow-Übersicht (siehe 4.6.4) alle laufenden und beendeten Workflows sehen.
- Ablaufverfolgung Workflows administrieren: Zusätzlich zu "sehen" dürfen laufende Workflows auch abgebrochen werden.
- Alle Workflow-Dokumente sehen: Berechtigung, in der Workflow-Dokument-Übersicht (siehe 4.6.4) alle von Workflows erstellten Workflow-Dokumente zu sehen.
- Alle Workflow-Dokumente verwalten: Zusätzlich zu "sehen" dürfen die Dokumente auch geändert oder gelöscht werden.

4.10.2 Berechtigungen auf Workflow-Vorlagen

Für jede Workflow-Vorlage in enablerWorkflows können weitere Berechtigungen zu Rollen zugeordnet werden, die nur für diese Vorlage beziehungsweise für die aus dieser Vorlage entstandenen Workflow-Instanzen gelten:

- Start new Workflow: Erlaubt einem Benutzer das Starten eines neuen Workflows aus dieser Vorlage.
- View Workflow: Erlaubt dem Benutzer, alle Workflows, die aus dieser Vorlage erstellt wurden, zu sehen.
- Administrate Workflows: Erlaubt es, laufende Workflows abzubrechen.
- View completed Documents: Erlaubt das Ansehen aller von einem Workflow aus dieser Vorlage erstellten Workflow-Dokumente.
- View Documents in creation: Erlaubt zusätzlich das Anzeigen von noch nicht abgeschlossenen Dokumenten (Status „In Creation“).
- Allow view Metainformation: Erlaubt zusätzlich das Einblenden von Meta-Informationen beim Anzeigen der Dokumente (z.B.: Ausfüller, Ausfüllungsdatum, letzte Änderung).
- Allow view hidden and deleted Parts: Erlaubt zusätzlich das Einblenden von Formularteilen, die durch die Aktivität “RemoveForm” (siehe 4.3.14) aus dem Workflow-Dokument gelöscht wurden.
- Edit Documents: Erlaubt zusätzlich das nachträgliche Editieren der Formulare im Workflow-Dokument.

Neben diesen Berechtigungen auf Vorlagen-Ebene gibt es zwei Settings bei Workflow-Vorlagen für den Ersteller eines Workflows:

- AllowCreatorViewDocument: Legt fest, ob der Ersteller eines Workflows das erstellte Dokument während und nach der Erstellung sehen darf.
- AllowCreatorViewWorkflow: Legt fest, ob der Ersteller den laufenden Workflow verfolgen darf (Status, nächster Schritt,...).

5 Phasen der Umsetzung eines Geschäftsprozesses

In diesem Kapitel soll der allgemeine Vorgang bei der Umsetzung eines Geschäftsprozesses mit enablerWorkflows gezeigt werden. Es handelt sich dabei nicht um eine ausführliche Benutzeranleitung (diese ist nicht in der Arbeit inkludiert), sondern um eine Beschreibung der wesentlichen Punkte des Verlaufs einer Umsetzung. Die konkrete Umsetzung eines gegebenen Beispiels hingegen wird im nächsten Kapitel vorgestellt.

Alle hier aufgelisteten Phasen können von einem technisch versierten Berater durchgeführt werden, eine Einbindung eines Softwareentwicklers ist nicht nötig.

5.1 Analyse

Die erste Phase ist die Analyse des Geschäftsfalls gemeinsam mit dem Kunden. Dieser Vorgang findet außerhalb von enablerWorkflows statt und soll hier nicht näher beschrieben werden. Folgende Punkte sollten jedoch jedenfalls im Verlauf der Analyse geklärt beziehungsweise beachtet werden:

- In welche Einzelschritte kann der Workflow zerlegt werden? Bei den Einzelschritten braucht und soll es sich noch nicht um einzelne Aktivitäten handeln, sondern um logische Bausteine, wie zum Beispiel "Verantwortlichen wählen" bei der Umsetzung eines Verbesserungsvorschlags. Eventuell muss dieser Vorgang wiederholt durchgeführt werden, um komplexere Bausteine wiederum in kleinere Einzelschritte zu zerlegen.
- Welche Person oder welche Rolle ist für die Durchführung eines Schrittes verantwortlich?
- Welche Informationen sollen in den einzelnen Schritten erfasst werden (Grobentwurf der entsprechenden Formulare)?

- Welche Informationen sind für die Durchführung eines Schrittes notwendig? Diese müssen in den Schritten davor gesammelt werden.
- Welche Ereignisse gibt es, die den "normalen" Ablauf des Geschäftsprozesses beeinflussen können? Diese müssen eventuell in alternativen Workflowzweigen abgebildet werden.
- Gibt es Zeitlimits, die für einzelne Schritte oder den gesamten Geschäftsprozess einzuhalten sind? Was soll bei Überschreitung einer Deadline passieren?
- Gibt es Personen außer den direkt am Prozess beteiligten, die über den Status oder das Ergebnis des Geschäftsprozesses informiert werden sollen?
- Soll ein Geschäftsstatus (siehe 4.1.2) verwendet werden? Welche Stati gibt es und in/nach welchen Schritten werden sie gesetzt?
- Sieht jeder am Geschäftsprozess Beteiligte die gesamte im Verlauf des Prozesses gesammelte Information (also das gesamte Workflow-Dokument), oder gibt es hier Einschränkungen?
- Welche Berechtigungen (siehe 4.10.2) sollen die einzelnen Rollen in diesem Geschäftsprozess haben?

5.2 Workflow-Design

In dieser Phase wird die Workflow-Definition für den Geschäftsprozess im grafischen Designer des Visual Studios erstellt. Eine genaue Kenntnis der von enablerWorkflows zur Verfügung gestellten Aktivitäten, sowie deren Parameter und deren Einsatzmöglichkeiten ist dafür Voraussetzung. Weiters sollten auch die Basisaktivitäten aus der Windows Workflow Foundation bekannt sein.

Der Workflow-Designer wählt passende Aktivitäten, um die einzelnen in der Analysephase gefundenen Schritte abzubilden und platziert diese entsprechend im Workflow. Dabei können durchaus mehrere Aktivitäten zur Abbildung eines Schrittes zum Einsatz kommen.

In weiterer Folge werden die Core-Parameter (siehe 4.7.4) aller verwendeten Aktivitäten betrachtet: Einige Parameter können bereits zum Design-Zeitpunkt gesetzt werden, sofern die entsprechenden Werte bereits jetzt bekannt sind und diese sich voraussichtlich nicht ändern werden. Für alle weiteren Parameter muss festgelegt werden, zu welchem Zeitpunkt sie gesetzt werden sollen. Die Werte können in der Workflow-Vorlage gesetzt werden (und gelten dann für alle Workflow-Instanzen) oder zum Startzeitpunkt einer Workflow-Instanz beziehungsweise erst während des Ablaufs des Workflows ermittelt werden. Diese Wahl wird mit Hilfe der Hilfsparameter getroffen, die jeder Core-Parameter besitzt.

5.3 Formulardesign

Nach oder auch parallel zum Workflow-Design werden die Formulare im Formular-Editor des enabler4BIZ erstellt. Dies geschieht in einem separaten Testsystem oder alternativ direkt im System des Kunden. Zu jeder FillOut-Aktivität muss ein entsprechendes Formular erstellt werden. Dazu werden die passende Formularelemente gewählt, um eine Eingabe der in diesem Schritt zu erfassenden Informationen zu ermöglichen. Da es sich hierbei um die Komponente des Workflows handelt, die aus Sicht des Benutzers den Workflow repräsentiert (Den Workflow selbst bekommt der Benutzer im Normalfall nie zu Gesicht.), sollte die Umsetzung möglichst übersichtlich und grafisch ansprechend gestaltet werden. Die Formularelemente, auf Basis derer in nachfolgenden Branch- und While-Aktivitäten Entscheidungen getroffen werden, sollten speziell hervorgehoben werden. Weiters muss bestimmt werden, welche Eingabefelder Pflichtfelder sind und bei welchen Feldern eine Eingabe optional ist.

5.4 Deployment

Ist das Design der Workflow-Definition und der Formulare abgeschlossen, werden diese ins Zielsystem beim Kunden veröffentlicht. Das Deployment der Workflow-Definition erfolgt wie unter 4.8 beschrieben. Wurden die Formulare

nicht bereits im Zielsystem erstellt, können diese exportiert und beim Kunden wieder importiert werden.

5.5 Konfiguration der Workflow-Vorlage

In diesem Schritt wird im enabler4BIZ aus der Workflow-Definition eine Workflow-Vorlage erstellt, also der Workflow im Laufzeitsystem eingebunden. Dies geschieht im Modul "Vorlagen-Administration" (siehe 4.6.1). Dabei wird die Workflow-Vorlage mit einem sprechenden Namen und einer Beschreibung versehen. Weiters werden die entsprechenden Berechtigungen (siehe 4.10.2, Berechtigungen auf Workflow-Vorlagen) auf der Vorlage gesetzt und die Werte der Aktivitätsparameter eingegeben, die für alle Workflow-Instanzen gelten sollen.

5.6 Konfiguration des Userinterface

Im letzten Schritt wird aus den vorhandenen Userinterface-Modulen (siehe 4.6) ein passendes Userinterface erstellt. Dabei muss zumindest eine Seite eingebunden werden, die das Modul "Workflow-Starten" (siehe 4.6.2) enthält, auf der also ein Workflow aus dieser Vorlage gestartet werden kann.

Weiters muss es eine Möglichkeit für die Benutzer geben, ihre Aufgabenliste zu sehen und abzuarbeiten. Dies geschieht durch Verwendung des Moduls "Aufgabenliste" (siehe 4.6.3), das auf einer zentralen Seite platziert werden kann, und in diesem Fall die globale Aufgabenliste aus allen Workflows für alle Benutzer enthält. Alternativ kann das Modul auf mehreren Seiten eingebunden und so konfiguriert werden, dass es nur die Aufgaben aus einem Workflow oder auch nur aus einem speziellen Schritt des Workflows enthält. Damit lassen sich verschiedene Seiten für die verschiedenen Verantwortungsbereiche im Workflow erstellen (zum Beispiel eine Liste für den Freigeber von Urlaubsanträgen mit dem Titel "Urlaubsanträge freigeben").

Zusätzlich sollte es mindestens eine Seite geben, die das Modul "Workflow-Übersicht" (siehe 4.6.4) enthält und dem Administrator sowie sonstigen Power-Usern die Ablaufverfolgung von Workflows ermöglicht. Ist weniger der Work-

flow selbst, sondern das im Zuge des Workflows erstellte Workflow-Dokument von Bedeutung, kann eine Übersicht über diese Dokumente mit dem Modul "Workflow-Dokument-Übersicht" zur Verfügung gestellt werden. In beiden Modulen können zusätzlich zu den Standard-Feldern des Workflows, wie Name und aktueller Status, auch die wichtigsten Felder aus den Formularen zu jedem Workflow eingeblendet werden.

6 Evaluierung

Die Evaluierung von enablerWorkflows soll am Beispiel einer Umsetzung eines Reklamationsmanagements erfolgen. Dieses Projekt befindet sich derzeit in der Umsetzungsphase bei einem Kunden der Firma Logic4BIZ, der hier anonym bleiben soll. Im Folgenden werden kurz die Ausgangssituation und die Anforderungen an das System beschrieben und danach der sich daraus ergebende Workflow vorgestellt. Auf Grund der Größe und Komplexität dieses Beispiels werden die Anforderungen sowie die Lösung hier nur in Teilen beschrieben und vereinfacht dargestellt. Der gesamte Workflow ist zum besseren Überblick am Ende dieses Kapitels in Abbildung 21 dargestellt.

6.1 Ausgangssituation

Der Kunde ist in der Herstellung von mobilen sowie stationären Anlagen zur Aufbereitung von Natursteinen, Sand und Kies sowie Recyclingmaterialien tätig. Die gelieferten Anlagen ermöglichen sowohl eine Nass- als auch Trockenaufbereitung der Materialien. Bei der Konzeption und Fertigung der Anlagen wird speziell auf die Bedürfnisse und Wünsche der Kunden eingegangen, um eine höchstmögliche Produktivität sowie Wirtschaftlichkeit auf Kundenseite zu ermöglichen.

Der Bereich After Sales Service (ASS) trägt laut aktueller Organisation die Prozessverantwortung für das Reklamationsmanagement. Die Umsetzungsaufgaben für die Reklamationsabwicklung liegen in Abhängigkeit von Art und Umfang der Reklamation in der jeweiligen Fachabteilung. Das ASS stellt die schnellstmögliche Abwicklung sicher. Der Qualitätsmanager wird über alle genehmigten Reklamationen informiert (Teil der Qualitätskostenberichte).

Die einlangenden Reklamationen (Mängelrügen) werden derzeit mittels Formularen (in Papierform) aufgenommen und an die Verantwortlichen weitergeleitet. Der hohe Aufwand der Erfassung auf Papier und die fehlende Nachvollziehbarkeit in der Weiterverarbeitung führen zum Wunsch, dies mit einer Softwarelösung bestmöglich zu unterstützen.

Dies soll mittels des Workflow-Management-Systems "enablerWorkflows" umgesetzt werden.

6.2 Anforderungen

6.2.1 Allgemein

Die Softwarelösung soll folgenden Zielen dienen:

- Unterstützung und Dokumentation aller wesentlichen Schritte im Rahmen des Reklamationsmanagements
- Sicherstellung der entsprechenden Datenhaltung und der Datenanbindung an Fremdsysteme (Kundendatenbank)
- Ermöglichung passender Statistik und Auswertungen

Bei Bedarf sollen zukünftig gewünschte Anpassungen und Erweiterungen durchgeführt werden können. Dies umfasst beispielsweise zusätzliche Funktionalitäten oder auch über einfaches Customizing hinausgehende vertiefende Sonderwünsche, die eine grundlegende Erweiterung der Funktionalität bedeuten (z.B. neue Funktionsmodule, unternehmensspezifische Masken, spezifische Schnittstellen zu bestehenden Lösungen oder spezifische Auswertungen).

Die Applikation soll für ca. 50 gleichzeitige Benutzer ausgelegt sein. Es werden zwei bis drei Reklamationen pro Tag beziehungsweise ca. 500 pro Jahr angenommen. Folgende Sicherheitsanforderungen sollen gelten:

- Verschlüsselung: Möglichkeit der Verfügbarkeit über https
- Authentifizierung: User melden sich mit Username/Passwort am System an.
- Sämtliche Daten gelten als besonders sensibel und sollten auch unbedingt revisionssicher gehalten werden (Nachvollziehbarkeit).

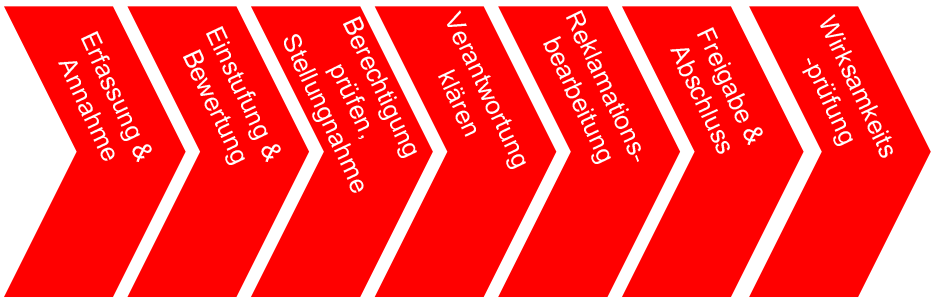

Abbildung 16: Reklamationsprozess

6.2.2 Reklamation

Eine Reklamation ist nur dann als Reklamation zu bezeichnen, wenn der Auftrag, dem die Reklamation zuzuordnen ist, vom Kunden übernommen und somit in die Gewährleistung übergeben wurde oder, wenn sich die Maschine/Anlage bereits nach der Gewährleistung befindet. So lange ein Auftrag/Projekt noch nicht abgeschlossen ist, liegen technische Störungen in der Verantwortung des zuständigen Projektleiters beziehungsweise Auftragsabwicklers und sind nicht als Reklamation zu bezeichnen. Eingegangene Beschwerden/Mängelmeldungen gelten als so genannte "Reklamationsauslöser" und werden erst nach einer Prüfung zu einer tatsächlichen Reklamation.

Der gesamte Prozess des Reklamationsmanagements ist in Abbildung 16 grob dargestellt. In jeder Phase wird ein Teilbereich des gesamten Reklamationsprozesses bearbeitet, relevante Informationen erfasst beziehungsweise Stati gesetzt. Die Anwendung sieht vor, dass es zu jeder Phase eine oder mehrere gesonderten Erfassungsmasken gibt, in welcher die berechtigten Benutzergruppen Daten erfassen und modifizieren oder ausschließlich einsehen (lesen) können.

Im Reklamationsmanagement werden folgende Rollen benötigt:

- Reklamationsmanager: Einzelner Verantwortlicher für das Reklamationsmanagement, im ASS angesiedelt

- Reklamationsverantwortlicher: Verantwortlicher für eine spezifische Reklamation
- Reklamationserfasser: Erfasser der Reklamation, wo auch immer sie eingeht

6.3 Phasen einer Reklamation

Im Folgenden werden die Phasen des Reklamationsprozesses im Detail beschrieben:

6.3.1 Reklamationsersterfassung und -annahme

In diesem Arbeitsschritt wird eine seitens des Kunden eingebrachte Beschwerde/Mängelrüge erfasst. Die Beschwerde kann vom Kunden über Telefon, Fax, E-Mail oder durch ein persönliches Gespräch eingebracht werden und wird im System durch den Reklamationserfasser (entweder Vertriebsinnendienst, Vertrieb, ASS) als Reklamationsauslöser erfasst und wenn möglich sofort einem Kunden, sowie dem Auftrag, zugeteilt. Der Kundenstamm, sowie Aufträge, sind in einem externen System (AS/400) erfasst. Der Reklamationsauslöser muss auf diese referenzieren. Weitere Informationen, welche in der AS/400 gespeichert sind, sind somit ebenfalls zugeordnet.

6.3.2 Reklamationseinstufung und Bewertung

In diesem Schritt wird vom Reklamationsmanager geprüft, ob es sich bei dem Reklamationsauslöser um eine neue Reklamation handelt oder ob die Beschwerde/Mängelrüge bereits zuvor gemeldet wurde. In diesem Fall wird der Auslöser der bestehenden Reklamation zugeordnet. Weiters wird der zuständige Projektleiter des ursprünglichen Auftrags der Reklamation zugeordnet. Dieser wird im weiteren Verlauf über den Fortschritt der Reklamation informiert. Außerdem wird ein Prüfer zugeteilt, der am Ende des Reklamationsprozesses eine Wirksamkeitsprüfung durchführt.

Folgende Varianten der Einstufung eines Reklamationsauslösers stehen zur Verfügung:

- Variante 1: Die Beschwerde/Mängelrüge ist neu und soll bearbeitet werden.
- Variante 2: Es ist bereits eine Reklamation zu diesem Thema angelegt, die Beschwerde/Mängelrüge wurde also vom Kunden doppelt gemeldet. Der Auslöser wird der bestehenden Reklamation zugeordnet und die Bearbeitung an dieser Stelle abgeschlossen.

6.3.3 Reklamationsberechtigung prüfen und Stellungnahme abgeben

In diesem Schritt wird überprüft, ob es sich um eine berechtigte Reklamation handelt. Dazu wird die gültige Verkaufsvereinbarung sowie weitere Informationen wie Abnahmedatum oder Gewährleistungsfrist seitens des Reklamationsmanagers geprüft. Weiters werden Informationen zur technischen Analyse eventuell in mündlicher Abstimmung mit der Abteilung ASS, wie beispielsweise Prüfdatum und Prüfprotokoll, kontrolliert. Anhand der zur Reklamation erfassten Informationen, sowie der Informationen zu vorangegangenen Reklamationen und zur Gewährleistung, erfolgt die Genehmigung oder Ablehnung der Reklamation inklusive einer Stellungnahme zu dieser Entscheidung.

In beiden Fällen (Genehmigung oder Ablehnung) wird der Kunde über die Entscheidung informiert. Im Falle einer Ablehnung hat der Kunde die Möglichkeit, diese Ablehnung zu beeinspruchen. In diesem Fall kann der Kunde weitere Informationen zur Reklamation einbringen. Danach wird die Berechtigungsprüfung wiederholt.

Im Fall einer endgültigen Ablehnung der Reklamation werden die bisher zur Reklamation gesammelten Informationen zur Erstellung eines Reparaturangebots an die Ersatz- und Verschleißteilabteilung weitergeleitet. Das Ergebnis dieses Angebots soll ebenfalls kurz im Reklamationsprozess dokumentiert und dieser danach beendet werden.

6.3.4 Verantwortung klären

In diesem Schritt wird vom Reklamationsmanager ein Reklamationsverantwortlicher bestimmt, der für die Bearbeitung der Reklamation zuständig sein soll. Standardmäßig muss der ausgewählte Reklamationsverantwortliche innerhalb von sieben Kalendertagen die Reklamationsbearbeitung annehmen und eine Stellungnahme abgegeben. Lehnt er die Reklamationsbearbeitung mit einer entsprechenden Begründung ab, muss vom Reklamationsmanager ein neuer Verantwortlicher ausgewählt werden. Dies geschieht so lange, bis ein gewählter Verantwortlicher die Reklamation annimmt.

6.3.5 Reklamationsbearbeitung durch den Verantwortlichen

In diesem Schritt werden vom Verantwortlichen zur Bearbeitung der Reklamation entsprechende Maßnahmen gesetzt und diese im System dokumentiert.

6.3.6 Reklamationsfreigabe und Abschluss

Nach Bearbeitung der Reklamation erfolgt zuerst die schriftliche Abnahme durch den Kunden (außerhalb des Systems) auf Basis der abgeschlossenen Maßnahmen. Das Datum des Abschlusses der Bearbeitung wird vom Reklamationsverantwortlichen gemeinsam mit einem Kommentar im System dokumentiert. Weiters erfolgt eine Dokumentation, dass die Erfassung der Zusatzaufträge zur Reklamationsposition im ERP-System erfolgt und abgeschlossen ist. Die Dokumentation der Verrechnung mit dem Schadensträger (Versicherung, Sublieferant) wird kann optional hinterlegt werden. Die Reklamation ist danach aus Sicht des Kunden abgeschlossen.

6.3.7 Wirksamkeitsprüfung

Nach einer Zeitverzögerung von sieben Tagen erfolgt eine Wirksamkeitsprüfung durch den bereits zuvor bestimmten Prüfer. Dieser kontrolliert, durch

Nachfrage beim Kunden, ob die Beschwerde/Mängelrüge aus der Reklamation nachhaltig behoben ist.

Ist dies nicht der Fall, wird die Reklamation wieder an den ursprünglichen Verantwortlichen zur Setzung weiterer Maßnahmen zur Behebung weitergeleitet, die Bearbeitung wird also in Schritt 5 fortgesetzt (siehe 6.3.5).

Nach positiver Wirksamkeitsprüfung wird die Reklamation auch intern abgeschlossen und der ursprüngliche Projektleiter informiert.

6.4 Umsetzung mit enablerWorkflows

Im Folgenden werden für jeden Schritt des Reklamationsprozesses die zur Lösung verwendeten Aktivitäten und Formulare beschrieben. Auf eine grafische Darstellung der gesamten Workflow-Definition wird auf Grund ihrer Größe hier verzichtet, es werden nur einige Ausschnitte davon in einzelnen Schritten dargestellt. Auch auf eine Abbildung der Formulare wurde aus Platzgründen zum größten Teil verzichtet.

6.4.1 Reklamationsersterfassung und -annahme

Der Reklamationserfasser startet bei Eingang einer Beschwerde/Mängelrüge einen neuen Reklamations-Workflow. Der erste Schritt des Workflows wird durch eine "FillOutNew_User"-Aktivität abgebildet, in der das Formular "Reklamationsauslöser" erfasst und in der der Workflow-Ersteller als FillingUser eingetragen wird. Dadurch öffnet sich für den Reklamationserfasser direkt nach Start des Workflows das Formular zur Erfassung des Reklamationsauslösers.

Das Formular enthält unter anderem Textfelder zur Beschreibung der Reklamation und weiters Felder zur Auswahl des Kunden und des zugehörigen Auftrags. Für diesen Zweck wurden die dynamischen Formulare des enabler4BIZ um ein Formularelement erweitert, das dynamisch Daten aus einem Fremdsystem auslesen und im Formular zur Auswahl anzeigen kann.

6.4.2 Reklamationseinstufung und Bewertung

In einer weiteren "FillOutNew_User"-Aktivität muss vom Reklamationsmanager das Formular "Reklamationsauslöser_Einstufen" ausgefüllt werden. Über eine Checkbox entscheidet er, ob es sich um eine neue Reklamation handelt oder bereits eine Reklamation für diesen Auslöser läuft. Zur Unterstützung der Entscheidung wurden die dynamischen Formulare um ein weiteres Formularelement erweitert, dass alle Reklamations-Workflows zum selben Auftrag anzeigt. Behandelt einer dieser Workflows den aktuellen Reklamationsauslöser, kann dieser selektiert werden.

Weitere Formularelemente sind eine Benutzerliste zur Auswahl des Projektleiters, sowie eine weitere Benutzerliste zur Auswahl eines Wirksamkeitsprüfers für diese Reklamation.

Nach dieser Aktivität wird eine "BranchOnBoolField"-Aktivität geschalten, die, abhängig vom Wert der Checkbox aus dem Formular "Reklamationsauslöser_Einstufen", den Workflow verzweigt.

Im False-Zweig wird mit der "SetState"-Aktivität der Geschäftsstatus des Workflows auf "Keine neue Reklamation" gesetzt und der Workflow beendet.

Im True-Zweig werden zunächst mit zwei "SendMail"-Aktivitäten der oben ausgewählte Projektleiter sowie der Wirksamkeitsprüfer darüber informiert, dass eine neue Reklamation eingegangen ist, bevor mit der Berechtigungsprüfung fortgesetzt wird.

6.4.3 Reklamationsberechtigung prüfen und Stellungnahme abgeben

Dieser Abschnitt des Workflows wird in Abbildung 17 dargestellt. Zunächst wird in der "FillOutNew_User"-Aktivität *Berechtigung_Prüfen* durch den Reklamationsmanager das Formular "Reklamationsberechtigung" ausgefüllt (siehe Abbildung 18: Ansicht für den Reklamationsmanager, wobei die bisher in diesem Workflow ausgefüllten Formulare in dieser Abbildung nicht dargestellt sind). Neben weiteren Feldern, die die Entscheidungsfindung dokumentieren, wird die tatsächliche Entscheidung über das Feld "Ergebnis" als

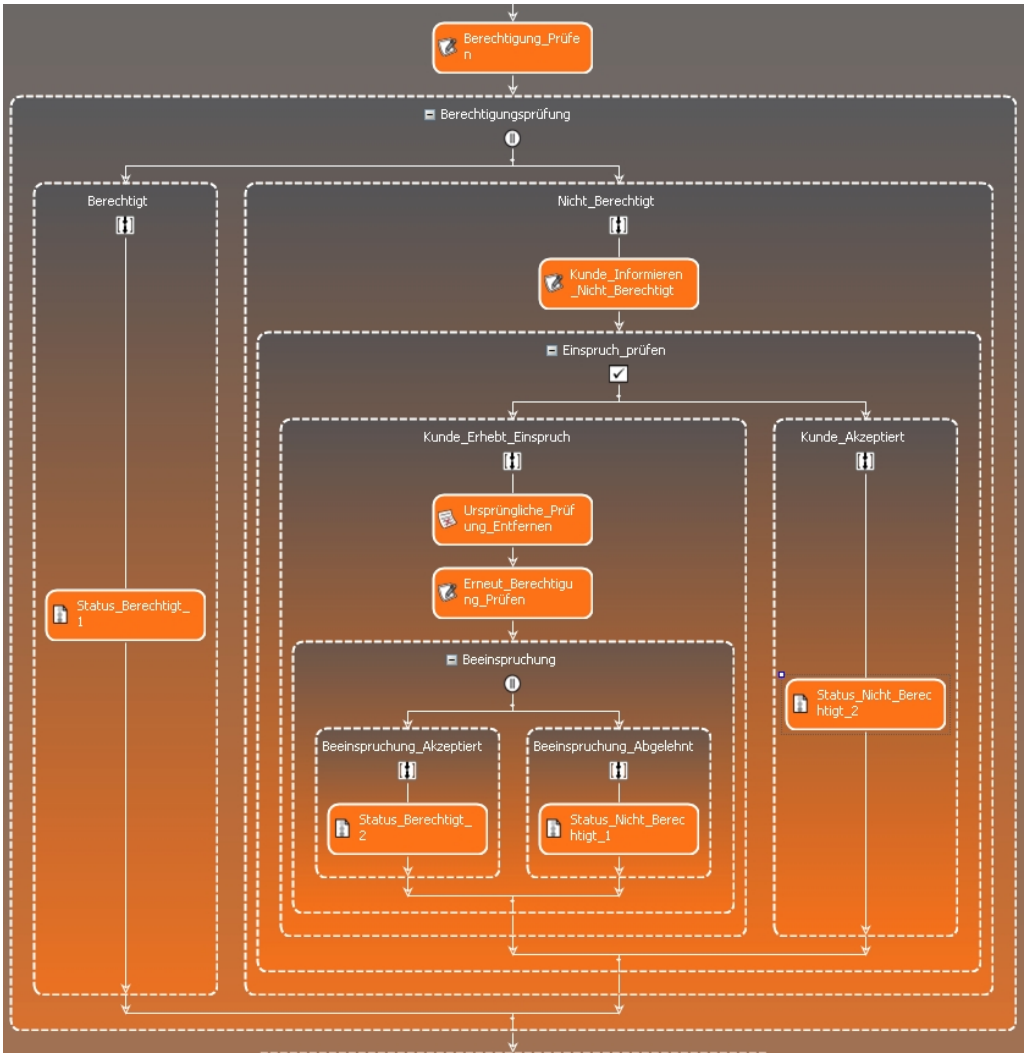

Abbildung 17: Workflow-Ausschnitt “Reklamationsberechtigung”

Radiobutton-Liste mit den zwei Auswahlmöglichkeiten “abgelehnt” und “genehmigt” dargestellt. Der Reklamationsmanager schließt seine Prüfung über den Button “Geprüft” ab.

Anschließend wird durch die “BranchOnStringField”-Aktivität *Berechtigungsprüfung*, die den Inhalt des Feldes “Ergebnis” aus dem Formular “Reklamationsberechtigung” mit dem Wert “genehmigt” vergleicht, der Workflow entsprechend verzweigt.

Im Fall einer berechtigten Reklamation wird der Geschäftsstatus des Workflows durch die “SetState”-Aktivität *Status_Berechtigt_1* auf “Berechtigt” gesetzt und der Workflow fortgesetzt.

Andernfalls muss der Reklamationsmanager den Kunden kontaktieren und ihm die Ablehnung mitteilen. Dieser Kundenkontakt wird in der Aktivität Kunde_Informieren_Nicht_Berechtigt dokumentiert. Weiters werden hier die Einspruchsgründe erfasst, sollte der Kunde einen Einspruch gegen diese Entscheidung haben.

Hat der Kunde keinen Einspruch, wird der Geschäftsstatus des Workflows durch *Status_Nicht_Berechtigt_2* auf “Nicht berechtigt” gesetzt und der Workflow fortgesetzt.

Andernfalls wird zunächst in der “RemoveForm”-Aktivität *Ursprüngliche_Prüfung_Entfernen* das Formular der ersten Prüfung aus dem Workflow-Dokument entfernt. Anschließend wird durch den Reklamationsmanager in der Aktivität Erneut_Berechtigung_Prüfen die Prüfung mittels Formular “Reklamationsberechtigung” (siehe 18) erneut durchgeführt. Je nach Ergebnis dieser zweiten Prüfung wird in der Branch-Aktivität *Beeinspruchung* der Geschäftsstatus auf “Berechtigt” oder “Nicht berechtigt” gesetzt und der Workflow fortgesetzt.

Im Anschluss an den dargestellten Block in Abbildung 17 wird auf Basis des gesetzten Geschäftsstatus in einer “BranchOnState”-Aktivität der Workflow weiter verzweigt. Ist der Status “Nicht berechtigt”, wird die Ersatz- und Verschleißteilabteilung per “SendMail”-Aktivität über die nicht berechtigte Reklamation informiert und aufgefordert, dem Kunden ein Ersatzteil- oder Reparaturangebot zukommen zu lassen. Das Ergebnis dieses Angebots wird

Abbildung 18: Formular “Reklamationsberechtigung”

noch von der Ersatz- und Verschleißteilabteilung in einem Formular dokumentiert (über eine FillOut-Aktivität), bevor der Workflow abgeschlossen wird.

Ist die Reklamation hingegen berechtigt, wird der Kunde vom Reklamationsmanager darüber informiert. Dieser Kundenkontakt wird ebenfalls in einem Formular dokumentiert. Der Workflow setzt fort mit dem Abschnitt “Verantwortung klären”.

6.4.4 Verantwortung klären

Dieser Abschnitt des Workflows wird in Abbildung 19 dargestellt. Die Reklamationsverantwortung wird in einer Schleife mittels der “WhileOnBoolField”-Aktivität *Schleife_ Annahme* geklärt. Die Schleife wird solange wiederholt, bis die Checkbox in dem Formular, das durch die Aktivität *Reklamations_ - Annahme* ausgefüllt wird, gesetzt ist, und damit ein Benutzer die Reklamati-

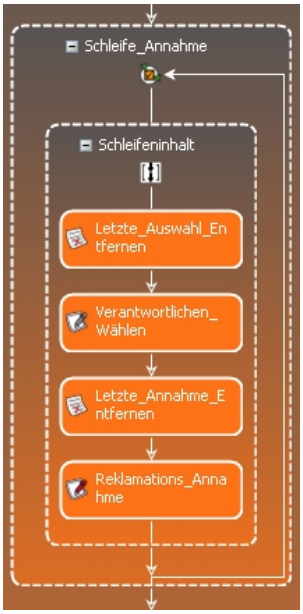

Abbildung 19: Workflow-Ausschnitt “Verantwortung klären”

onsverantwortung angenommen hat. Da das Formular beim ersten Durchlauf der Schleife noch nicht vorhanden ist, wird die Schleife auf Grund des Standardverhaltens bei nicht gefundenem Formular in jedem Fall ausgeführt.

Im ersten Schritt der Schleife wird durch die Aktivität *Letzte_ Auswahl_ Entfernen* das Auswahlformular aus dem letzten Schleifendurchlauf entfernt. Danach wählt der Reklamationsmanager im Schritt *Verantwortlichen_ Wählen* im Auswahlformular einen (neuen) Verantwortlichen für diese Reklamation aus. Vor der *Reklamations_ Annahme*-Aktivität wird noch ein eventuell vorhandenes Annahme-Formular aus dem letzten Schleifendurchlauf entfernt.

Bei der Aktivität *Reklamations_ Annahme* handelt es sich um eine “TimeLimitedFillOutNew_User”-Aktivität. Der im Schritt *Verantwortlichen_ Wählen* ausgewählte Verantwortliche wird hier automatisch als ausfüllender Benutzer übernommen. Dies geschieht über den FillOutModus “AutoFillFromFormField” (siehe 4.7.4). Der Parameter “TimeLimit” der Aktivität wird auf sieben Tage gesetzt. Bei Überschreiten dieser Deadline wird der Reklamationsmanager per E-Mail darüber benachrichtigt. Im Fall einer Ablehnung

der Verantwortung durch den ausgewählten Benutzer wird die Schleife erneut ausgeführt, und vom Reklamationsmanager kann ein neuer Verantwortlicher bestimmt werden. Andernfalls ist der Reklamationsverantwortliche bestimmt, und der Workflow wird fortgesetzt.

6.4.5 Reklamationsbearbeitung durch den Verantwortlichen

Die eigentliche Reklamationsbearbeitung erfolgt in einer "FillOutNew_User"-Aktivität, die vom Verantwortlichen auszufüllen ist. Darin wird der Verantwortliche aufgefordert, entsprechende Maßnahmen zur Bearbeitung der Reklamation zu setzen und diese im zugehörigen Formular zu dokumentieren. Nach Dokumentation der gesetzten Maßnahmen läuft der Workflow weiter.

6.4.6 Reklamationsfreigabe und Abschluss

Sobald die Maßnahmen abgeschlossen sind, wird deren Ergebnis, sowie der entsprechende Kundenkontakt, in einer weiteren "FillOutNew_User"-Aktivität vom Verantwortlichen dokumentiert. Das zugehörige Formular enthält weiters optionale Felder zur Dokumentation der Verrechnung an den Schadensträger. Der Verantwortliche schließt durch Ausfüllen dieses Formulars die Reklamation ab.

6.4.7 Wirksamkeitsprüfung

Der Workflow-Abschnitt zur Wirksamkeitsprüfung ist in Abbildung 20 dargestellt. Die Prüfung wird im Workflow wiederum durch eine Schleife umgesetzt, die ähnlich funktioniert wie die Schleife zur Klärung der Verantwortlichkeit (siehe 6.4.4).

Die beiden Aktivitäten aus 6.4.5 und 6.4.6 zu Bearbeitung und Abschluss der Reklamation liegen innerhalb dieser Schleife zu Beginn. Anschließend erfolgt eine Verzögerung des Workflows von sieben Tagen über eine Delay-Aktivität (Dabei handelt es sich um eine Basisaktivität aus der Windows Workflow Foundation.). Danach erfolgt die Wirksamkeitsprüfung durch den

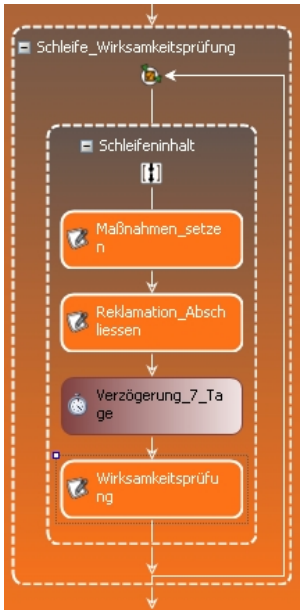

Abbildung 20: Workflow-Ausschnitt “Wirksamkeitsprüfung”

bereits zu Beginn des Workflows ausgewählten Prüfer. Er dokumentiert seine Prüfung in einem Formular. Auf Basis einer Checkbox in diesem Formular wird die Schleife bei negativer Prüfung erneut ausgeführt, und der ursprüngliche Reklamationsverantwortliche hat weitere Maßnahmen zur Bearbeitung der Reklamation zu setzen. Im Gegensatz zur Schleife in 6.4.4 werden hier die bereits ausgefüllten Formulare aus den vorherigen Schleifendurchläufen nicht entfernt. Dies führt dazu, dass alle für diese Reklamation gesetzten Maßnahmen und alle Prüfungen am Ende des Workflows im Workflow-Dokument dokumentiert sind.

Nach positiver Wirksamkeitsprüfung wird der Workflow fortgesetzt: In einer weiteren “SetState”-Aktivität wird der Geschäftsstatus auf “Abgeschlossen” gesetzt. In einer “SendMail”-Aktivität wird noch der Projektleiter vom Abschluss der Reklamation informiert und danach der Workflow beendet.

6.5 Ergebnis

Die Anforderungen des Reklamationsmanagement konnten mit enablerWorkflows problemlos erfüllt werden. Das Beispiel zeigt gut die Einsatzmöglichkeiten der verschiedenen in enablerWorkflows zur Verfügung gestellten Aktivitäten. Man sieht aber auch, dass es einiger Erfahrung im Umgang mit dem System bedarf, um etwas komplexere Anforderungen, wie zum Beispiel die Berechtigungsprüfung (siehe 6.4.3) oder die Verantwortungsklärung (siehe 6.4.4) mit enablerWorkflows umzusetzen.

Vom zeitlichen Aufwand her benötigt ein "geübter" Workflow-Designer für einen Workflow dieser Größe bei geklärten Anforderungen etwa ein bis zwei Tage um die Workflow-Definition im Visual Studio zu erstellen und weitere zwei bis drei Tage für das Design der zugehörigen Formulare im Formulareditor.

Im Zuge des Workflow-Designs für dieses Beispiel wurde die bis dahin nicht vorhandene Aktivität "BranchOnState" (siehe 4.3.19) erstellt. Die Erstellung dieser neuen Aktivität war auf Grund der bereits in den Basisklassen der Aktivitäten vorhandenen Funktionalitäten in weniger als zwei Stunden möglich. Dies zeigt, dass enablerWorkflows leicht und schnell um neue Aktivitäten erweiterbar ist.

Die Implementierung dieses Beispiels zeigt, dass es möglich sein sollte, solche und ähnliche Anforderungen leicht und schnell mit enablerWorkflows umzusetzen und dem Kunden zur Verfügung zu stellen.

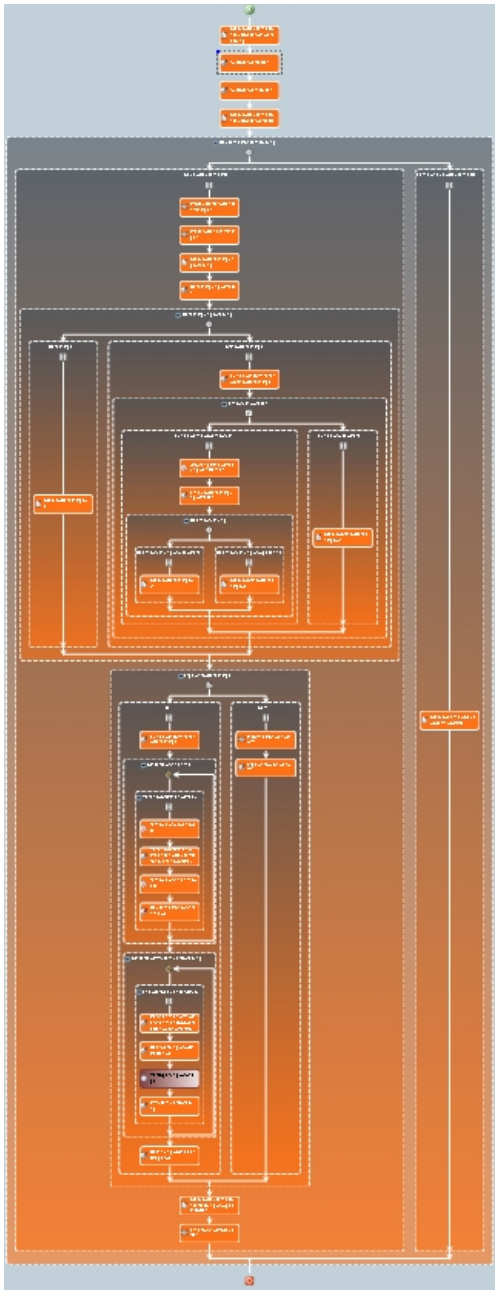

Abbildung 21: Reklamations-Workflow gesamt

7 Zusammenfassung und Ausblick

7.1 Fazit

In dieser Arbeit wurde anhand der Erweiterung "enablerWorkflows" des Softwareproduktes "enabler4BIZ" gezeigt, wie ein web-basiertes, formularorientiertes Workflow-Management-System auf Basis der neuen Windows Workflow Foundation aussehen kann.

enablerWorkflows ermöglicht durch das flexible Abbilden der einzelnen Abläufe in Unternehmen durch Workflows mit den dafür notwendigen Formularfeldern eine einfache und komfortable sowie vollständige Erfassung, Bearbeitung und Weiterleitung relevanter Informationen in Geschäftsprozessen.

Dynamische Formulare, welche im enabler4BIZ entsprechend den Anforderungen des Kunden abgebildet und umgesetzt werden, bilden die Basis für eine für den Benutzer einfache und komfortable Erfassung der Daten in den einzelnen Workflow-Schritten.

Projektleiter sowie Projektmitarbeiter können ihre Informationen und Protokolle den betreffenden Geschäftsfällen zuordnen und über vordefinierte Formulare im Workflow einreichen. Sie werden per E-Mail über aktuelle Tätigkeiten oder über Durchlaufzeitüberschreitungen informiert. Aktive Verständigung mittels E-Mail an die jeweiligen Verantwortlichen bei anstehenden Entscheidungen sorgt für termingerechte, komfortable und rasche Abwicklung der einzelnen Workflow-Schritte.

Nach vollständiger Eingabe der Basisinformationen eines Schrittes werden die Informationen entsprechend der gewählten Abbildung der Arbeitsabläufe im Workflow zu den relevanten Abteilungen und Mitarbeitern gesendet. Mittels zugrunde liegender Rollenberechtigungen erlaubt das System nur Zugriff auf die für die jeweilige Person freigegebenen Inhalte.

Die durchgängige Dokumentation erfolgt online im System über das jedem Workflow zugeordnete Workflow-Dokument, das Protokolle, Bemerkungen, Freigaben und getroffene Entscheidungen dokumentiert.

Die für enablerWorkflows in den enabler4BIZ integrierten Userinterface-Module ermöglichen einerseits die einfache Abarbeitung der persönlichen Aufga-

benlisten, andererseits wird auf den Übersichtsseiten durch anpassbare Filterfunktionen eine komfortable Lösung zur Abfrage der eingegebenen Informationen in laufenden und beendeten Workflows angeboten.

Die Windows Workflow Foundation des Herstellers Microsoft als Basis für enablerWorkflows garantiert eine zukunftsfähige Lösung durch Weiterentwicklung dieser Basis und eine Anpassung an neue Entwicklungen in der IT-Welt seitens des Herstellers. Außerdem wird dadurch sowohl die Einbindung von Aktivitäten anderer Hersteller in enablerWorkflows, als auch eine eventuell nötige Kommunikation mit anderen Workflow-Management-Systemen, die auf Basis der Workflow Foundation erstellt wurden ermöglicht.

7.2 Ausblick

enablerWorkflows in seiner vorliegenden und in dieser Arbeit beschriebenen ersten Version deckt natürlich nur die Minimalanforderungen an Workflow-Management-Systeme ab. In Kapitel 6 vorgestellte Umsetzung eines Reklamationsmanagement-Systems zeigt aber, dass sich bereits mit dieser Version einfachere Geschäftsfälle problemlos abbilden lassen. Trotzdem gibt es bereits eine Liste von möglichen Anpassungen und Erweiterungswünschen für die nächsten Versionen von enablerWorkflows. Neben diversen kleineren Features sind das folgende Themen:

7.2.1 Erweiterung der Aktivitäten-Basis

Derzeit werden in enablerWorkflows 29 verschiedene Aktivitäten (zusätzlich zu jenen aus der Workflow Foundation) angeboten, wobei es sich allerdings bei einigen Aktivitäten nur um Varianten desselben Typs von Aktivität handelt (z.B. BranchOnBoolField, BranchOnIntField, BranchOnStringField). Dieses Angebot sollte noch stark erweitert werden, um weitere Anforderungen in Geschäftsfällen abbilden zu können. Beispielhaft seien hier nur Mehrfachverzweigungen, Parallelausführung von Aktivitäten oder Exceptionhandling auf struktureller Basis erwähnt.

Als Ausgangspunkt für die Implementierung weiterer Aktivitäten können die Muster in den Arbeiten von [RHEA04a, RHEA04b, RHAM06, RAH06] herangezogen werden, die bereits die Basis für die Auswahl der bereits zur Verfügung stehenden Aktivitäten dienten.

Der Aufwand für die Implementierung weiterer Aktivitäten sollte sich in Grenzen halten. So hat beispielsweise das Hinzufügen der "BranchOnState"-Aktivität (siehe 4.3.19), deren Implementierung die Abbildung des Reklamationsmanagements-Systems notwendig machte, nur wenige Stunden in Anspruch genommen.

7.2.2 Verbessertes Handling der Aufgabenlisten

Derzeit können personenbezogene Aufgaben nur an Einzelpersonen oder Rollen gebunden werden. Neben Rollen existieren innerhalb der Organisationsstruktur von Unternehmen auch so genannte Funktionen, die einzelne Personen innehaben. Eine Abbildung von Organsisationsfunktionen in der Basissoftware enabler4BIZ und eine Anbindung an enablerWorkflows über entsprechende Aktivitäten, über die Aufgaben an diese Funktionen gebunden werden können, wäre hierfür eine wünschenswerte Erweiterung.

Außerdem ist es derzeit nicht möglich, personenbezogene Aufgaben aus der Aufgabenliste an andere Benutzer zu delegieren (außer, dies wurde im Workflow über eigene Aktivitäten speziell so vorgesehen, wie zum Beispiel bei der Klärung der Verantwortung im Reklamations-Workflow (siehe 6.4.4) . Eine solche Weiterleitung durch einen Administrator kann aber durchaus nötig sein, zum Beispiel bei Ausscheiden des betreffenden Mitarbeiters aus der Firma. In diesem Fall ist derzeit nur ein Abbruch des Workflows und ein Neustart mit neuen Verantwortlichen möglich.

7.2.3 Verbesserte Gestaltungsmöglichkeiten für den Kunden

In der derzeitigen Version von enablerWorkflows ist weder vorgesehen, dass der Kunde selbst dynamische Formulare im Formulareditor ändert oder neu erstellt, noch, dass er die Struktur des Workflows selbst anpassen kann oder

eine neue Workflow-Definition erstellt. Diese Aufgaben werden derzeit von Beratern seitens des Herstellers "Logic4BIZ" durchgeführt. In vielen Projekten wird aber die Erfahrung gemacht, dass der Kunde oft zumindest die Möglichkeit haben möchte, solche Änderungen durchzuführen.

Im Fall des Formulareditors sollte dies keine großen Probleme bereiten. Dieser ist bereits jetzt web-basiert in den enabler4BIZ eingebunden, für den Kunden auf Grund seiner sehr einfachen und technik-lastigen Gestaltung allerdings noch nicht freigeschalten. Nach einer Überarbeitung des Editors steht einer Freischaltung nichts im Wege.

Für den Workflow-Designer ist dies etwas schwieriger. Derzeit werden Workflow-Definitionen mit dem integrierten Designer des MS Visual Studios erstellt, das als Entwicklungswerkzeug für den Kunden weniger geeignet ist. Eine Integration eines bestehenden oder die Entwicklung eines eigenen, nach Möglichkeit web-basierten grafischen Designers wäre hier eine Möglichkeit. Aktuell verfügbare Lösungen, wie der *Atlas Workflow Designer*[2] oder der *Net FX Live*-Designer[3] scheinen allerdings noch nicht ausgereift zu sein. Zusätzlich müsste hier der Deployment-Vorgang (siehe 4.9.3) von neuen oder geänderten Workflow-Definitionen vereinfacht und sicherer gestaltet werden.

[2]http://www.masteringbiztalk.com/atlasworkflowdesigner (letzter Zugriff Okt. 2008)
[3]http://www.netfxlive.com (letzter Zugriff Okt. 2008)

Abbildungsverzeichnis

Literatur

[Aal04] *Kapitel* Business Process Management Demystified: A Tutorial on Models, Systems and Standards for Workflow Management. In: Aalst, Wil M. d.: *LNCS 3098*. Springer Berlin / Heidelberg, 2004 (Lecture Notes in Computer Science), S. 1–65

[AHKB03] Aalst, W.M.P van d. ; Hofstede, A.H.M. ter ; Kiepuszewski, B. ; Barros, A.P.: Workflow Patterns. In: *Distributed and Parallel Databases* 14 (2003), July, 5-51. `http://workflowpatterns.com`

[Cas98] Casati, Fabio: *Models, Semantics and Formal Methods for the design of Workflows and their Exceptions*, Politecnico di Milano, Diss., 1998

[Coa99] *Terminology and Glossary, WFMC-TC-1011.* Workflow Management Coalition. `http://wfmc.org`. Version: Feb 1999

[DRSV03] *Kapitel* Authorization and Access Control in Adaptive Workflows. In: Domingos, Dulce ; Rito-Silva, António ; Veiga, Pedro: *Computer Security, ESORICS 2003, LNCS 2808.* Springer Berlin / Heidelberg, 2003 (Lecture Notes in Computer Science), S. 23–38

[GHJV95] Gamma, Erich ; Helm, Richard ; Johnson, Ralph ; Vlissides, John: *Design patterns: elements of reusable object-oriented software*. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1995. – ISBN 0–201–63361–2

[Hol95] Hollingsworth, David: *The Workflow Reference Model.* Workflow Management Coalition, 1995 (TC00-1003). `http://wfmc.org`

[MSD08] MSDN, Microsoft Developer N.: *Windows Workflow Foundation XAML.* URL. `http://blogs.msdn.com/msdnat/archive/`

2007/07/25/windows-workflow-foundation-xaml.aspx. Version: Juli 2008

[Mue04] Muehlen, Michael Z.: *Workflow-based Process Controlling: Foundation, Design, and Application of Workflow-driven Process Information Systems*. Logos, Berlin, 2004

[RAH06] Russell, N. ; Aalst, W.M.P. van d. ; Hofstede, A.H.M. ter: *Exception Handling Patterns in Process-Aware Information Systems*, Business Process Management (BPM) Center, BPM Center Report BPM-06-04, 2006. http://BPMcenter.org/

[RHAM06] Russell, N. ; Hofstede, A.H.M. ter ; Aalst, W.M.P. van d. ; Mulyar, N.: *Workflow Control-Flow Patterns: A Revised View*, Business Process Management (BPM) Center, BPM Center Report BPM-06-22, 2006. http://BPMcenter.org/

[RHEA04a] Russell, N. ; Hofstede, A.H.M. ter ; Edmond, D. ; Aalst, W.M.P. van d.: *Workflow Data Patterns*, Queensland University of Technology, Brisbane, QUT Technical report, FIT-TR-2004-01, 2004. http://workflowpatterns.com

[RHEA04b] Russell, N. ; Hofstede, A.H.M. ter ; Edmond, D. ; Aalst, W.M.P. van d.: *Workflow Resource Patterns*, Eindhoven University of Technology, Eindhoven, BETA Working Paper Series, WP 127, 2004. http://workflowpatterns.com

[RRD02] Rinderle, Stefanie ; Reichert, Manfred ; Dadam, Peter: Effiziente Verträglichkeitsprüfung und automatische Migration von Workflow-Instanzen bei der Evolution von Workflow-Schemata. In: *Computer Science - Research and Development* Volume 17 (2002), Dezember, Nr. Nummer 4, S. 177–197

[RZ96] Riehle, Dirk ; Züllighoven, Heinz: Understanding and Using Patterns in Software Development. In: *Theory and Practice of Object Systems* 2 (1996), S. 3–13

[SCFY96] Sandhu, Ravi S. ; Coynek, Edward J. ; Feinsteink, Hal L. ; Youmank, Charles E.: Role-Based Access Control Models. In: *IEEE Computer* 29 (1996), Nr. 2, S. 38–47

[SK97] Sheth, Amit ; Kochut, Krys J.: Workflow Applications to Research Agenda: Scalable and Dynamic Work Coordination and Collaboration Systems. In: *Proc. NATO Advanced Study Institute on Workflow Management Systems and Interopability. Istanbul*, 1997, S. 12–21

[SS07] Shukla, Dharma ; Schmidt, Bobby: *Essential Windows Workflow Foundation.* Addison-Wesley Longman, Amsterdam, 2007

[Ste96] Stein, Dominik: *Definition und Klassifikation der Begriffswelt um CSCW, Workgroup Computing, Groupware, Workflow Management*, Universität Gesamthochschule Essen, Seminararbeit, Dezember 1996

[Wik08] Wikipedia: *Workflow-Management.* URL. `http://de.wikipedia.org/wiki/Workflow-Management`. Version: Juli 2008

[Win04] Winkelmeyer, Alexander: *Implementierung eines SCORM konformen Lern-Management-Systems*, FH Oberösterreich Campus Hagenberg, Diplomarbeit, 2004