kenroads IS GmbH

Artificial Intelligence meets Real Life

Der *kenroads* Designer

Funktionsweise und Besonderheiten

(am Anwendungsfall: Entwicklung von IT-Anwendungen)

Januar 2011

kenroads Intelligent Systems GmbH
Amtsgericht Köln, HRB 70429, USt-IdNr.: DE274568191
Geschäftsführer: Jürgen Geisel, Sascha Lehner

Landmannstraße 39, 50825 Köln
Telefon: 0179 / 94 130 93
E-Mail: kenroads@kenroads.com

Die Funktionsweise des *kenroads* Designer.

Die *kenroads* Designer basiert auf der Idee, dass die inneren Zusammenhänge selbst scheinbar unstrukturierter und unzusammenhängender Eingangsdaten, die sich auf eine gemeinsame Fachdomäne beziehen, durch mehrstufige iterative Anwendung intelligenter text-basierter und semantischer Analyseverfahren (überwiegend dem Bereich der „Künstlichen Intelligenz“ zugehörig) maschinell extrahiert werden können. So enthalten etwa die Fachdokumente, die einem Geschäftsprozess als Eingabe dienen, von ihm als Ausgabe erzeugt werden oder Geschäftsregeln zu seiner Steuerung beinhalten, im Allgemeinen genügend Informationen über den Prozess selbst, um bereits die weitgehend automatische Generierung eines Anwendungs-Prototyps zu erlauben. Der praktische Nutzen dieser Vorgehensweise wird dadurch noch verstärkt, dass die erforderlichen Eingangsinformationen typischerweise entweder bereits vorliegende Datenaggregationen oder zumindest sehr einfach zu erstellende und rein fachorientierte Textdokumente (Formulare, Briefe, Urkunden, etc.), Kalkulationstabellen (Berechnungen, Tabellen, Listen, etc.) oder Ähnliches sein können.

Der *kenroads* Designer analysiert also die Daten und Dokumente, die er als Eingabe erhält, extrahiert und deduziert Wissen über den zugrunde liegenden Prozess oder die betroffene Fachdomäne, füllt bei Bedarf Wissenslücken durch Nachfragen bei einem fachversierten Ansprechpartner und generiert schließlich ein aufgabenspezifisches Ergebnis, wie z.B. eine Anwendung zur Automatisierung eines fachlichen Prozesses.

Dieses Dokument beschreibt die Funktionsweise des *kenroads* Designer und der ihm zu Grunde liegenden Verfahren in ihren wesentlichen Aspekten. Wegen der größeren Durchgängigkeit und Anschaulichkeit erfolgt diese Beschreibung an Hand eines konkreten Anwendungsbereichs und nutzt die „**Generierung von IT-Anwendungen**“ als roten Faden für die folgenden Erläuterungen, die in 7 Abschnitte aufgeteilt sind.

	1.	Überblick
Analyse	2.	Hauptmerkmale
	3.	Dokumentenanalyse
	4.	Wissenserzeugung
Generierung	5.	Aufbau einer (generierten) Anwendung
	6.	Desktop-Laufzeitumgebung („Runtime“)

1 Überblick

Idealerweise kann der *kenroads* Designer z.B. in einem Softwareentwicklungsprojekt das Anwendungsdesign und die Programmierung übernehmen, dazu beim Testen, der technischen Dokumentation und der Erzeugung der Anwenderdokumentation helfen. Auch der *kenroads* Designer benötigt dabei die Mitarbeit eines Menschen, idealerweise eines Fachexperten, den wir **Planer** nennen. Bei einfachen Anwendungen besteht die Aufgabe des Planers lediglich darin, Dokumente zu sammeln, dem

kenroads Designer zur Verfügung zu stellen und einfache Fragen zu beantworten. Bei komplexeren Anwendungen ergänzt der Planer die Dokumente um zusätzliche Informationen bzw. erstellt weitere Dokumente, die etwa zur vollständigen Beschreibung der Geschäftsregeln oder des Workflows einer gewünschten Anwendung benötigt werden.

Da der *kenroads* Designer auch komplizierte Zusammenhänge innerhalb eines Prozesses erkennen kann, braucht sich der Planer in den meisten Fällen nicht mit Fragen der Datenherkunft, der Berechnungslogik oder des Prozessablaufs zu beschäftigen. Dadurch können Anwendungen deutlich schneller und günstiger entwickelt werden.

Bei der Entwicklung komplexer Systeme oder der Erweiterung vorhandener Systeme kann der *kenroads* Designer alternativ auch als Tool zur Analyseunterstützung und Anforderungsspezifikation genutzt werden. In diesem Fall kann das Wissen über Datenmodell, Business Logik und Workflow in eine Form überführt werden, die an das jeweilige Vorgehensmodell angepasst ist (Pflichtenheft, Modelle, usw.).

Abb. 1 stellt das Verfahren im Überblick dar.

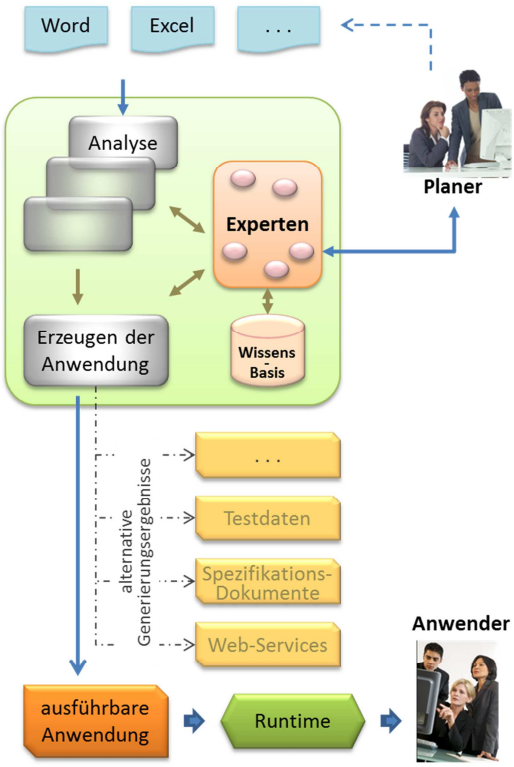

Abbildung 1: Gesamtablauf (Schema)

2 Hauptmerkmale

Die Funktionsweise des *kenroads* Designer basiert u.a. auf der Anwendung textbasierter und semantischer Analyseverfahren aus dem Bereich der Künstlichen Intelligenz (KI). Während das „Geheimnis" vieler KI-Systeme jedoch in einzelnen, dafür aber umso komplexeren Verfahren (wie z.B. neuronalen Netzen) liegt, besitzt der *kenroads* Designer eine ganze Reihe von Fähigkeiten, die ihn insgesamt in die Lage versetzen, Wissen aus Dokumenten zu extrahieren und aufgabenorientiert in ein gewünschtes Endergebnis wie etwa eine ausführbare Anwendung umzusetzen. Hierzu gehören insbesondere

a. die Fähigkeit, Sprache zu verarbeiten;
b. die Fähigkeit, Strukturen und Zusammenhänge in Dokumenten und anderen Datenmengen zu erkennen und zu analysieren;
c. die Fähigkeit, mit unsicherem bis widersprüchlichen Wissen umzugehen; sowie
d. die Fähigkeit, Wissenslücken zu erkennen und durch gezielte Fragen zu schließen.

Ein weiteres wichtiges Merkmal des *kenroads* Designer ist sein modularer Aufbau. Er besteht aus einer prinzipiell offenen Menge spezialisierter Module, den **Experten**. Jeder Experte beherrscht eine abgegrenzte Aufgabe, zu deren Erfüllung er die Funktionsweise der anderen Experten nicht kennen muss und im Normalfall auch nicht kennt.

Die Fähigkeiten, Sprache und Strukturen verarbeiten zu können, nutzt der *kenroads* Designer dazu, möglichst viel potentielles Wissen aus den Dokumenten zu extrahieren, die ihm der Planer zur Verfü-

gung stellt. Dieses Wissen wird von verschiedenen Experten bewertet, ergänzt, aussortiert und erweitert. Dabei arbeitet jeder Experte nach dem **Lokalitätsprinzip**, d. h. er sammelt so viel Wissen wie möglich und interessiert sich nicht dafür, ob und wie dieses Wissen für andere Experten nützlich ist. Da auf diese Weise auch unnötiges oder gar falsches „Wissen" erzeugt wird, versehen die Experten das von ihnen erzeugte Wissen mit Plausibilitätsannahmen und vermerken erkannte Widersprüche zwischen Wissenselementen. Außerdem kann jeder Experte Fragen an den Planer formulieren, deren Beantwortung er zur Vervollständigung seines Wissens benötigt.

3 Dokumentenanalyse

Das Herz des *kenroads* Designer ist der **Koordinator**, der die Zusammenarbeit der Experten koordiniert, die Wissensbasis verwaltet und mit dem Planer kommuniziert. Experten können untereinander nur über den Koordinator kommunizieren. Ein Experte kann dem Koordinator einen Analyseauftrag geben, den dieser an einen geeigneten anderen Experten weiterleitet. Dieser übergibt das Ergebnis wiederum dem Koordinator, der es an den Auftraggeber ausliefert. Diese strenge Kapselung wird vom *kenroads* Designer konsequent eingehalten, um die modulare Entwicklung sowie den flexiblen Ausbau des Systems und seine Anpassung an besondere Aufgabenstellung zu ermöglichen.

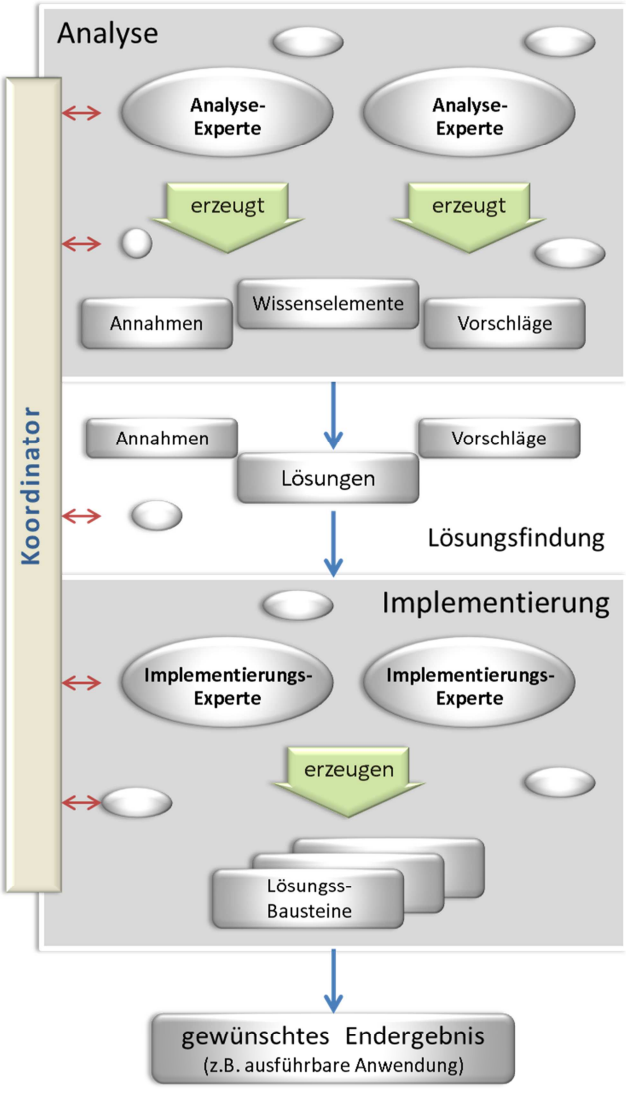

Abbildung 2: Dokumentenanalyse

Experten gehören unterschiedlichen Klassen an. Experten, die Wissen erzeugen, werden **Analyseexperten** genannt. Daneben gibt es **Implementierungsexperten**, die die gewünschten Ergebnisformate wie z.B. Anwendungsbausteine erzeugen. Diese Aufteilung ermöglicht die konsequente Trennung und Unabhängigkeit von Analyse und Ergebniserzeugung (in unserem Fall: Anwendungsgenerierung).

Die Arbeit des *kenroads* Designer kann nun in **drei Haupthasen** eingeteilt werden: Die Analysephase, die Lösungsfindung und die Implementierungsphase.

In der **Analysephase** sammelt der *kenroads*-Designer möglichst viel Wissen über die zu erzeugende Anwendung. Unsicheres oder widersprüchliches Wissen wird mit Annahmen versehen. Annahmen besitzen Plausibilitäten und können auf anderen Annahmen basieren und/oder zu anderen Annahmen in Widerspruch stehen. Auf diese Weise wird Wissen bewertet und strukturiert. Die Analysephase

beginnt mit der Analyse der Dokumente durch spezialisierte, jeweils auf bestimmte Dokumenttypen ausgerichtete, Experten. Diese erkennen Elemente, Texte und Strukturen in den Dokumenten und überführen sie in eine Typ-unabhängige Form, aus der danach von wieder anderen Experten das benötigte Wissen extrahiert wird. Die Analyse wird iterativ solange fortgeführt, wie aus dem gewonnenen Wissen weiteres Wissen abgeleitet werden kann. Am Ende der Analysephase erzeugen Experten einer bestimmen Ausprägung Vorschläge zur Implementierung, die auf dem analysierten Wissen beruhen. Diese Vorschläge werden dann später von den Implementierungsexperten umgesetzt (vgl. Abb. 2).

In der zweiten Phase, der **Lösungsfindung**, erzeugt der *kenroads* Designer aus der Menge aller Annahmen widerspruchsfreie Teilmengen, die potentielle Lösungen (zum Beispiel: mögliche Anwendungen) bilden.

In der abschließenden **Implementierungsphase** werden diese Lösungsvorschläge bewertet und aus den ausgewählten Varianten schließlich das gewünschte Endergebnis (z.B.: eine ausführbare Anwendung) erzeugt. Dazu wird für jeden betroffenen Vorschlag ein passender Implementierungsexperte aufgerufen, der einzelne Lösungsbausteine (z.B.: Anwendungsbausteine) generiert.

4 Wissenserzeugung

Das Wissen, das der *kenroads* Designer erzeugt, kann grob in drei Bereiche unterteilt werden:

1. Wissen über das Datenmodell der Anwendung;
2. Wissen über die Funktionalität der Anwendung;
3. Wissen über den Ablauf der Anwendung.

Das **Datenmodell** setzt sich im Wesentlichen aus Datenfeldern und Datenquellen zusammen. Datenfelder dienen der internen Eingabe, Ausgabe und Verarbeitung von Daten. Der *kenroads* Designer unterscheidet zwischen atomaren (typisierten) Datenfeldern und komplexen Datenfeldern wie etwa Listen oder Datenfeldgruppen. Datenquellen beschreiben die Möglichkeit, Daten außerhalb der Anwendung zu speichern bzw. zu lesen. Für den *kenroads* Designer ist eine Datenquelle dabei zunächst nichts anderes als eine Menge von Datenfeldern, für die Werte an die Datenquelle geschickt bzw. von dieser geholt werden können. Eine Datenquelle kann damit eine Datenbank, aber auch einen „Service“, repräsentieren.

Das Wissen über die **Funktionalität** der Anwendung setzt sich aus Funktionen, Formeln, Bedingungen und Vorschlägen zu Aktionen zusammen.

Das Wissen über den **Ablauf** der Anwendung wird aus Beziehungen zwischen den übrigen Wissenselementen abgeleitet und in Form von Vorschlägen zur Generierung von Anwendungsschritten und Übergangsregeln zwischen den Schritten aufgebaut.

Eine zentrale Rolle spielen Komponenten, die zum einen Datenfelder zu einer logischen Einheit zusammenfassen und damit zum Aufbau eines Datenmodells beitragen. Darüber hinaus analysiert der *kenroads* Designer aber auch eine Reihe unterschiedlicher Beziehungen zwischen Komponenten, die für die weitere Analyse von Bedeutung sind. Topologische Beziehungen bilden die Grundlage für die Bildung von Umgebungen, die bei der Analyse von komplexen Anwendungen notwendig sind, um Datenfelder und funktionale Wissenselemente einander korrekt zuzuordnen. Datenbasierte Beziehungen bilden die Grundlage für die Analyse des Anwendungsablaufs.

Die folgende Tabelle gibt einen Überblick über die wichtigsten Wissenselemente:

Wissenselement	Beschreibung	Analysierte Informationen
Datenfeld	Ein Datenfeld beschreibt eine Variable bzw. ein Objekt.	- Name - Mögliche Datentypen - Referenz auf die Herkunftskomponente - Beziehungen zu anderen Datenfeldern - Beispiele - Mögliche Werte - Beziehungen zu Datenquellen - Formeln - Default-Werte - Formatierungsvorgaben - Restriktionen - Expertenspezifische Eigenschaften
Komponente	Eine Komponente beschreibt eine Menge von zusammenhängenden Datenfeldern und die Struktur, die diese Datenfelder gemeinsam bilden.	- Name - Mögliche (Struktur-)Typen - Mögliche Verwendungen in der Anwendung - Referenz auf das Herkunftsdokument - Beziehungen zu anderen Komponenten - Enthaltene Datenfelder - Beispiele - Beziehungen zu Datenquellen - Expertenspezifische Eigenschaften - Die Umgebung dieser Komponente (andere Komponenten)
Formeln	Eine Formel kann eine beliebige (auch nicht numerische) Berechnungsvorschrift sein. Die Berechnung einer Formel kann an eine Bedingung geknüpft sein.	- Aufbau der Formel - Mögliche Operatoren bzw. Funktionen - Mögliche Operanden (Formeln, Konstante, Datenfelder) - Ausführungsbedingungen
Bedingungen	Eine Bedingung ist eine logische Formel. Bedingungen können mit Formeln, Ereignissen, Aktionen oder Datenfeldern verbunden werden.	- Aufbau der Formel - Operatoren - Mögliche Operanden (Formeln, Konstante, Datenfelder) - Verwendungszweck (Ausführungsbedingung einer Formel, Restriktion eines Datenfeldes, Übergangsbedingung im Prozess, usw.)
Datenquellen	Eine Datenquelle repräsentiert eine Datenstruktur zum Lesen und Speichern von Daten	- Felder mit Namen und Datentypen - Beziehungen zu Datenfeldern

Wissenselement	Beschreibung	Analysierte Informationen
		- Beziehungen zu Komponenten
Annahmen	Eine Annahme bezieht sich auf ein Wissenselement und weist diesem eine Plausibilität zu. Eine Annahme kann auf anderen Annahmen basieren und/oder zu anderen Annahmen in Widerspruch stehen.	- Wissenselement - Plausibilität - Basisannahmen - Widersprüchliche Annahmen
Vorschläge	Ein Vorschlag bezieht sich auf einen Anwendungsbaustein, der Teil der Ausführbare Anwendung werden könnte. Jeder Vorschlag basiert auf Annahmen.	

5 Aufbau einer (generierten) Anwendung

Der Aufbau einer vom *kenroads* Designer generierten Anwendung gestaltet sich im Beziehungsgeflecht zwischen Daten, Ablauf und Funktionalität der Anwendung (siehe Abb. 3). Die Daten (die der Anwender eingibt) steuern den Ablauf der Anwendung bzw. legen fest, welche Ablaufmöglichkeiten gegeben sind. Der Ablauf wiederum steuert die Funktionalität der Anwendung bzw. macht bestimmte Funktionalitäten für den Anwender verfügbar. Schließlich verwendet und manipuliert die Funktionalität die Daten, d. h. führt Berechnungen durch, ändert Daten, stellt Daten dar, usw. Aus diesem Dreiecksverhältnis ergeben sich auch die **drei zentralen Bausteine einer Anwendung**.

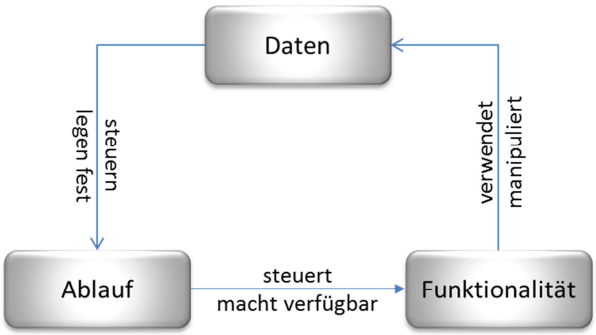

Abbildung 3

Der zentrale Anwendungsbaustein zur Haltung und Manipulation von Daten ist das **Datenfeld**, das analog zu dem gleichnamigen Wissenselement die Aufgabe hat, einen Wert bzw. eine Information zu speichern.

Der zentrale Anwendungsbaustein zur Realisierung des Ablaufs eines Arbeitsprozesses ist der **Schritt**. Durch Schritte werden zum einen alle anderen Bausteine strukturiert und zusammengefasst, zum anderen wird der Anwendungsablauf abgebildet und mit Eingabemasken und Funktionalität verbunden.

Der zentrale Anwendungsbaustein zur Realisierung der Funktionalität ist die **Aktion**, die beliebige Funktionen ausführen kann und damit neben Berechnungen und anderen Datenmanipulationen auch Ausgaben realisieren, Dokumente erzeugen sowie Daten laden und speichern kann.

Abbildung 4 zeigt alle im Folgenden näher beschriebenen Bausteine im Überblick.

Datenfeldbaustein

Datenfeldbausteine werden direkt aus den gleichnamigen Wissenselementen „Datenfeldern“ abgeleitet und sind analog aufgebaut.

Formeln, Bedingungen

Formeln und Bedingungen werden direkt aus den entsprechenden Wissenselementen abgeleitet und sind analog aufgebaut.

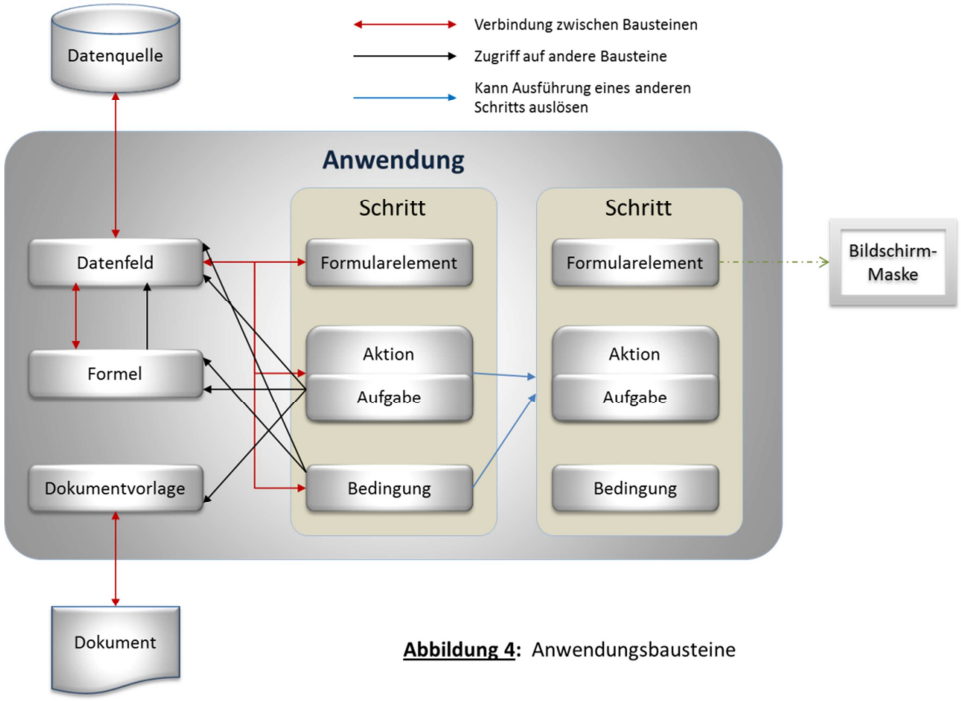

Abbildung 4: Anwendungsbausteine

Aktionen

Eine Aktion besteht aus einer Folge von **Kommandos**, die nacheinander ausgeführt werden, wobei Sprünge in Abhängigkeit von der Ausführung möglich sind. Ein Kommando ist, analog zu einer Funktion in einer Programmiersprache, die kleinste funktionale Einheit einer Anwendung. Ein Kommando ist definiert durch

- Eine eindeutige ID, die das Kommando von allen anderen Kommandos unterscheidet und implementierungsunabhängig die korrekte Ausführung zur Laufzeit der Anwendung sicherstellt.
- Die Semantik seiner Ausführung. (Was tut das Kommando?)
- Parameter, die aus der Anwendung heraus mit konstanten Werten oder Werten aus Datenfeldern belegt werden. (Womit tut es das?)
- die Resultate seiner Ausführung (Was ist das Ergebnis seines Tuns?):

Die Ausführung einer Aktion und ihrer Kommandos ist Bestandteil der Ausführung einer Anwendung und nicht Bestandteil des Verfahrens der Anwendungsgenerierung. Eine Anwendung kann allerdings nur dann fehlerfrei ausgeführt werden, wenn sichergestellt ist, dass zur Laufzeit Implementierungen aller Kommandos verfügbar sind, die dem *kenroads* Designer zur Verfügung stehen.

Aufgabe

Eine Aufgabe ist technisch gesehen eine Aktion, die allerdings gezielt vom Anwender ausgeführt wird. Aufgaben werden an einen oder mehrere Schritte gebunden, d. h. sie stehen dem Anwender nur dann zur Verfügung, wenn einer der Schritte, an die sie gebunden ist, gerade aktiv ist. Zusätzlich kann die Möglichkeit, eine Aufgabe ausführen zu können, von einer Bedingung abhängig gemacht werden.

Für jede Aufgabe kann festgelegt werden, dass sie ausgeführt werden muss, bevor der aktuelle Schritt oder die Anwendung beendet werden.

Für jede Aufgabe kann festgelegt werden, ob sie nur einmal oder mehrmals ausgeführt werden kann.

Schritte

Ein Schritt ist durch die Möglichkeiten definiert, die er dem Anwender bietet:

- Eingaben, die der Anwender machen kann oder muss.
- Ausgaben, die dem Anwender zur Verfügung gestellt werden.
- Aktionen bzw. Aufgaben, die der Anwender aufrufen kann oder muss oder die durch Eingaben des Anwenders angestoßen werden.
- Wechsel zu anderen Schritten, aktiv durch den Anwender ausgelöst oder automatisch durch die Erfüllung festgelegter Bedingungen.

Einem Schritt kann, wenn Eingaben und/oder Ausgaben vorgesehen sind, eine Bildschirmmaske zugeordnet sein.

Formularelemente

Formularelemente sind Bestandteile von Schritten. Ein Formularelement repräsentiert ein Datenfeldbaustein und dient zur Kommunikation mit dem Anwender, d. h. zur Anzeige und/oder Eingabe des Wertes dieses Datenfeldbausteins.

Datenquellen

Datenquellen realisieren den Austausch von Daten zwischen der Anwendung und ihrer Umgebung. Eine Datenquelle existiert unabhängig von der Anwendung und kann sowohl Daten an die Anwendung liefern als auch von der Anwendung erhalten. Datenquellen dienen damit sowohl der Speicherung von Daten als auch der Kommunikation mit anderen Anwendungen bzw. technischen Prozessen.

Ein Anwendungsbaustein „Datenquelle“ beschreibt lediglich eine physikalische Datenquelle. Die Bindung an eine physikalische Datenquelle (Datenbank, Service, etc.) erfolgt zur Laufzeit.

6 Desktop-Lauzeitumgebung („Runtime“)

Die Ausführung einer erzeugten Desktop-Anwendung ist Aufgabe einer separaten **Runtime**. Die Runtime interpretiert die Anwendung und baut daraus zur Laufzeit Objekte, die miteinander interagieren und so die Anwendung realisieren. Dieses Vorgehen hat zwei Vorteile: Zum einen ist dadurch die Plattformunabhängigkeit einer erzeugten Anwendung gegeben, da die Anwendung keine Plattform-spezifischen Elemente enthält. Möchte man Anwendungen auf einer neuen Plattform nutzen, so muss lediglich eine Plattform-spezifische Runtime implementiert werden.

Die Runtime stellt für jeden Typ von Anwendungsbausteinen eine Klasse zur Verfügung. Beim Start einer Anwendung wird für jeden erzeugten Anwendungsbaustein eine entsprechende Instanz dieser Klasse erzeugt. Die Anwendung wird durch die Interaktion dieser Objekte realisiert.

Der zweite Vorteil liegt darin, dass die Runtime in der Lage ist, auch in Teilen noch unvollständige oder fehlerhafte Anwendungsversionen zum Ablauf zu bringen, ohne „Systemabstürze“ befürchten zu müssen. Der Anwender (bzw. Planer) ist dadurch in der Lage, schon sehr frühzeitig mit den konkreten Ergebnissen seiner bisherigen Spezifikationen „lebensnah“ zu interagieren und dadurch aus praktischem Erleben gespeiste Erkenntnisse und Anregungen für die weitere Entwicklung zu ziehen. In diesem anhaltenden Dialog mit der werdenden Anwendung kann diese bis zur fachlichen Vollständigkeit Schritt für Schritt fertig entwickelt werden, bis dann abschließend die Generierung des gewünschten Endergebnisses erfolgt.

7 Zusammenfassung: Vorteile

Zusammenfassend bietet die Entwicklung von IT-Lösungen mit dem *kenroads* Designer eine Reihe signifikanter Vorteile gegenüber herkömmlichen Vorgehensweisen, u.a.:

- Anwender-gerechte Spezifikationsdokumente ermöglichen die gleichberechtigte Einbindung der Fachbereiche und sorgen für hohe Akzeptanz.
- Gleichzeitig vermeidet der Verzicht auf komplexe Ergebnistransformationen Informationsverluste an der Schnittstelle IT – Fachfunktionen.
- Intelligente Analyseverfahren erkennen Zusammenhänge, fördern implizites Wissen zu Tage und identifizieren Lücken oder Widersprüche in den Anforderungsspezifikationen.
- Intelligente Assistenten helfen dem Entwickler (bzw. Planer), Spezifikationslücken zu schließen und die Spezifikationen zielgerichtet weiter zu entwickeln.
- Eine eigene Laufzeitumgebung ermöglicht ablauffähige Prototypen von Beginn an und lässt Trockenübungen durch interaktives Gestalten ersetzen.

In der Summe dieser Vorteile erhöht der *kenroads* Designer die Vollständigkeit der Anforderungen sowie deren Abbildungsqualität und führt insgesamt zu Zeit- und Kostenvorteilen bei der Erstellung von IT-Lösungen.