Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

Pflichtenheft

Data Forest

Autoren: Boris Goldshteyn
Clemens Wagner
Christian Wolter

Modul: Software Engineering I (IT2101)

Studienjahrgang: IT2011

Fachrichtung: Informatik

Studienbereich: Technik

Fachbereich: FB 2, Duales Studium Wirtschaft • Technik, HWR Berlin

Stand vom: 06.11.2012

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Vorwort

1.1 Erwartete Leserschaft

Als Hauptaufgabe des Pflichtenheftes soll dieses die Anforderungen des Auftraggebers so widerspiegeln, wie diese vom Auftragnehmer verstanden wurden. Es dient damit als Basis der Verständigung quasi aller Stakeholder. Als zu erwartende Leserschaft dieses Pflichtenheft sind somit vor allem der Auftraggeber Prof. Höhne, die Nebenkundin Prof. Monet Diaz, selbige Personen als Lehrbeauftragte Professoren sowie die Entwickler des Projektes zu verstehen.

1.2 Versionsgeschichte des Dokuments

Version	Datum	Autor	Zusammenfassung der Veränderungen / Bemerkungen
1.0	3.10.2012	Boris	Texte eigefügt
1.1	18.10.2012	Christian	Anforderungen Tabellenansicht
1.2	19.10.2012	Clemens	Formatierung
1.3	19.10.2012	Clemens	Anforderungen Regeldarstellung
1.4	22.10.2012	Gemeinsam	Unvollständiges Klassendiagramm
1.5	31.10.2012	Christian	Funktionale Anforderungen vervollständigt
1.6	31.10.2012	Boris	Anforderungen Baum-interaktiv
1.7	06.11.2012	Clemens	In Vorlage eingefügt

2 Einleitung

2.1 Allgemeines

Das folgende Dokument stellt eine vollständige Beschreibung und Analyse der Entwicklung einer Lehr- und Lernsoftware im Rahmen des Kurses für Software-Engineering I im 3. Semester des Informatik Jahrgangs 2011 an der Hochschule für Wirtschaft und Recht in Berlin dar.

Diese Ausarbeitung dient als Vorlage für eine Implementierung eines funktionsfähigen Programms im fortführenden Kurs der Software-Engineering II im 4. Semester des Informatik Jahrgangs 2011.

Diese Hilfssoftware soll Dozenten der HWR bei der Ausübung ihrer Lehrtätigkeiten unterstützen, indem es auf eine leicht verständliche Art die Inhalte der Vorlesungen interaktiv den Informatikstudenten nachfolgender Jahrgänge vermitteln soll.

2.2 Abgrenzungen

2.2.1 Ausgangssituation

Im Rahmen von Lehrveranstaltungen im Studiengang Informatik der HWR zum Thema Data Mining werden unter anderem die Entscheidungsbäume behandelt.

2.2.2 Systemname

Ein Vorschlag unseres Entwicklungsteam wäre der Name: "Data Forest"

Um zu gewährleisten, dass eine Software attraktiv auf den Anwender wirkt, sollte der Name auf kreative Art den Inhalt dieses Systems aufzeigen. "Data Forest" verbindet dabei das Themengebiet um das Data Mining und die grafische Ausgabe der Entscheidungsbäume (Tree). Dabei wird ein Wald (Forest) als eine Sammlung von verschiedenen Bäumen assoziiert.

Somit erfüllt der Name zum einen die Wiedergabe der Funktionalität des Programms, zum anderen gibt es ein kreativ-attraktives Bild von der Visualisierung der Software.

2.2.3 Allgemeine Systembeschreibung

Das im Nachfolgenden beschriebene System soll als Lehr- und Lernsoftware im Rahmen der Lehrveranstaltungen dienen. Es soll Studenten ermöglichen, eigene Entscheidungsbäume auf Grundlage eines Datensatzes zu erstellen oder einen optimalen berechneten Baum als Referenz darzustellen.

Die Datenbasis kann entweder selbst eingegeben werden oder durch einlesen einer Daten erstellen werden.

2.2.4 Vorteile und Nutzen

Die Software kann das Verständnis der Studenten von Entscheidungsbäumen verbessern und den lehrenden Dozenten bei der Vermittlung des Unterrichtstoffes unterstützen.

Studenten könnten die Theorie auch außerhalb der Vorlesungen üben.

Außerdem wird auf dem digitalen Weg Papier eingespart, sodass das System auch die Umwelt schont.

2.3 Überblick über den Rest des Dokuments

<Ihr Text>

3 Fachbegriffe, Akronyme und Abkürzungen

<Ihre Tabellen mit den technischen Fachbegriffen, Akronymen und Abkürzungen, die einer eindeutigen Definition bedürfen. Das erste Mal im Text sowieso einführen und/oder Verweis zu diesem Kapitel>

3.1 Glossar

Begriff	Definition
Markieren	Mit der Maus auswählen bzw. anklicken
Identifizieren	Mit der Maus auswählen bzw. anklicken
Reiter	Siehe Tab
Tab	Ein Tab ist eine Karteikarte in einer Tabansicht. Diese erlaubt es durch Anklicken des Tabs zu diesem zu wechseln. So lässt sich ohne weiteren Platzbedarf durch die Reiter, bzw. Karteikarten blättern.
Scroll-Balken	Ein Scroll-Balken erlaubt es die dazugehörige Anzeige nach oben oder unten zu verschieben um Inhalte, welche die Größe der Anzeige überschreiten einblenden zu können.
Button	Ein Anzeigeelement welches das Anklicken durch optisches eindrücken verdeutlicht.

3.3 Abkürzungen

Abkürzung	Definition
GUI	Grafische Benutzer Oberfläche
CI	Continuos Integration, kontinuierliche Integration

4 Zielbestimmung

Bei dem System „Data Forest“ handelt es sich um eine Software für den Lehr- und Lerneinsatz. Dafür gibt es spezielle Ziele und Kriterien zu beachten, die in den nachfolgenden Absätzen näher erläutert werden..

4.1 Musskriterien

Ziel	Die Programmoberfläche muss für Demonstrationszwecke geeignet sein.
Stakeholder	Anwender (Studenten und Dozenten)
Auswirkungen auf Stakeholder	Durch eine übersichtliche und aussagekräftige Programmoberfläche lassen sich die Lehrinhalte besser an die Studenten vermitteln.
Randbedingungen	
Abhängigkeiten	Lauffähigkeit ohne Installation unter Windows
Sonstiges	

Ziel	Das Programm muss einen optimalen Entscheidungsbaum selbstständig berechnen können.
Stakeholder	Anwender (Studenten und Dozenten)
Auswirkungen auf Stakeholder	Das Programm unterstützt die Anwender, indem es schnell einen optimalen Entscheidungsbaum berechnet und als Referenzlösung darstellt.
Randbedingungen	Bestimmung des optimalen Entscheidungsbaum auf Grundlage der Entropie.
Abhängigkeiten	
Sonstiges	

Ziel	Das Programm muss es dem Anwender erlauben, einen eigenen Entscheidungsbaum interaktiv zu erstellen.
Stakeholder	Anwender (Studenten und Dozenten)
Auswirkungen auf Stakeholder	Das interaktive PC-gestützte Erstellen von Entscheidungsbäumen verringert den Arbeitsaufwand und erlaubt das Bearbeiten von großen Datensätzen.
Randbedingungen	
Abhängigkeiten	
Sonstiges	Es wird Papier eingespart.

Ziel	Das Programm muss Funktionen für das Erstellen und Bearbeiten eines Datensatzes bereitstellen.
Stakeholder	Anwender (Studenten und Dozenten)
Auswirkungen auf Stakeholder	Mit Hilfe von Computern lassen sich viel größere Datenmengen verarbeiten und ohne großen Aufwand bearbeiten.
Randbedingungen	
Abhängigkeiten	
Sonstiges	

Ziel	Das Programm muss eine Dateiverwaltung für Daten-Dateiformate bereitstellen.
Stakeholder	Anwender (Studenten und Dozenten)
Auswirkungen auf Stakeholder	Die Anwender können vorgegebene Datensätze einlesen und bearbeiten.
Randbedingungen	Aufgabenstellung: Excel-Dateien und csv-Dateien müssen unterstützt werden.
Abhängigkeiten	
Sonstiges	

4.2 Sollkriterien

Ziel	Das System soll ohne Installation unter allen gängigen Windowsbetriebssystemen lauffähig sein.
Stakeholder	Auftragsgeber
Auswirkungen auf Stakeholder	Das System ist unter allen den Vorlesungen zur Verfügung stehenden Computern lauffähig.
Randbedingungen	Java darf vorausgesetzt werden.
Abhängigkeiten	
Sonstiges	

4.4 Abgrenzungskriterien

<Ihr Text mit den Kriterien, die bewusst nicht erreicht werden sollen, falls welche>

5 Systemeinsatz

5.1 Anwendungsbereiche

Wie schon in 2.2.3 beschrieben findet diese Lehr- und Lernsoftware ihre Verwendung in Themengebiet des Data Mining. Data Mining ist eine Möglichkeit des Umgangs mit Datenmengen. Für eine Analyse dieser wird ein Verfahren benötigt, welches schnell und effizient aus der riesigen Informationsmenge Nutzen für die Verarbeitung der Daten zieht. Data Mining erkennt dabei Muster, sortiert die Datenbestände nach Kategorien und extrahiert wichtige Regeln, die Unternehmen in Zukunft wettbewerbsfähiger machen (1).

In Universitäten wird Data Mining als wichtiges Teilgebiet der Mathematik und Informatik verstanden und gelehrt. Das Ziel dieser Software soll bei der Unterstützung während der Vorlesungen und Übungen auf diesem Gebiet liegen..

5.2 Zielgruppen

Die Software wird von Studenten für Studenten geschrieben. Den Studierenden aller relevanten Fachgruppen soll der Zugang zu dieser Software ermöglicht werden. Auch die Dozenten, die auf dem Gebiet des Data Mining lehren, Seminare und Vorlesungen halten, sollen mit der Software umgehen und diese verwalten können. Des Weiteren sollten sie die teilnehmenden Studenten auf die Verwendung dieser hinweisen und auch zur Verfügung stellen.

Das Alter und Geschlecht der Anwender spielt dabei keine Rolle..

5.3 Betriebsbedingungen

Für die Verwendung der Software ist ein Computer mit einem gängigem Windows Betriebssystem notwendig. Eine Installation ist nicht nötig, das Programm zudem auch von externen Speichermedien aus gestartet werden.

Der Anwender muss kein Vorwissen im Bereich des Data Mining besitzen, der Umgang mit der Software sollte einfach, verständlich und selbsterklärend sein.

6 Technische Systemumgebung

6.1 Software

Für den Einsatz der Software wird ein Windowsbetriebssystem vorausgesetzt. Je nach gewählter Programmiersprache werden weitere Softwarekomponenten benötigt.

Als Empfehlung gilt die Programmiersprache C-Sharp (C#). Sie benötigt eine Implementierung der Common Language Infrastructure (CLI). Das .NET-Framework ist die gängige Variante für Windowsbetriebssysteme

6.2 Hardware

Die Rechnerhardware muss die Systemanforderungen des Windowsbetriebssystems und ggf. erforderlicher weiterer Softwarekomponenten erfüllen.

7 Systemübersicht und -Architektur

7.1 Übersicht über das System

Das System „Data Forest“ soll dem Benutzer die möglichen Funktionalitäten auf eine logische und leicht nachvollziehbare Art und Weise präsentieren. Jeder Schritt und jede Aktion die möglich ist soll dem Anwender somit keine Rätsel aufgeben und nur der Beschreibung entsprechend plausible Reaktionen auslösen. Um dies konsequent umzusetzen wird versucht eine Funktionalität nur dann anzubieten wenn diese auch zum gegebenen Zeitpunkt und im gegebenen Zustand Sinn macht. Aufgrund dessen, ist das System wie in **Fehler! Verweisquelle konnte nicht gefunden werden.** zu sehen aufgebaut. Es entsteht eine Art hierarchischer Abhängigkeiten die die Funktionalitäten sukzessive aufbauend dem Anwender zur Verfügung stellt.

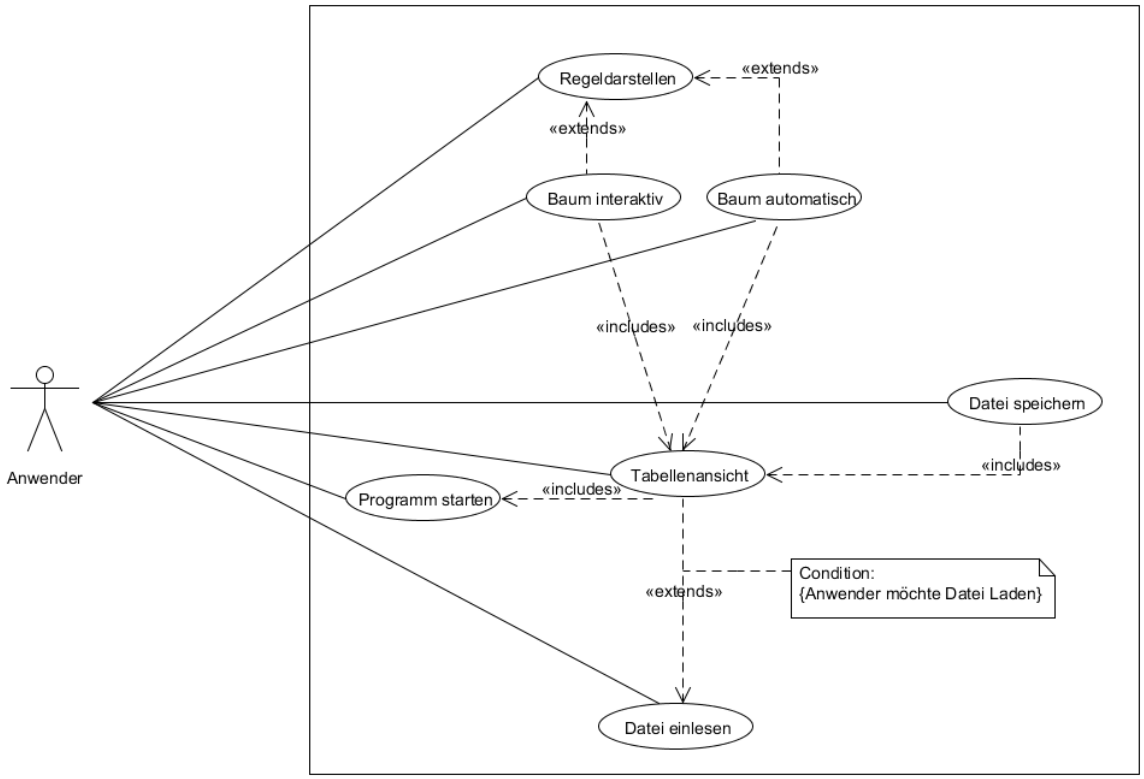

Abbildung 1: übergeordnetes Use-Case-Diagramm

Um die Transparenz groß zu halten existiert lediglich die Rolle des Anwenders, wobei es sich dabei quasi um jeden Stakeholder handeln kann, der befähigt ist, das Programm zu Bedienen. Eine Unterscheidung zwischen Student und Professor bspw. ist unnötig und existiert aufgrund dessen nicht.

So ist es dem Anwender möglich das Programm zu starten so, dass ihm die Tabellenansicht präsentiert wird. Von dort aus kann er als Erweiterung der Funktionalität der Tabellenansicht Daten einlesen lassen. Ebenfalls sind im weitere Funktionalitäten zugänglich, die auf der Tabellenansicht aufbauen. So lassen sich Baumdarstellungen erzeugen, welche sich mittels Regeldarstellung erweitern lassen. Auch das Speichern der Tabellendaten ist dem Anwender direkt möglich.

So erschließen sich dem Anwender nacheinander die Funktionalitäten was eine intuitive Bedienung hervorrufen soll.

8 Systemanforderungen

8.1 Funktionale Anforderungen

Programmstart

Name des Use-Case	Programmstart
Nummer	1
Paket	Programm
Autor	Boris Goldstein
Version	1.1
Kurzbeschreibung	Das vollständige Programm soll fehlerfrei ausgeführt werden
Beteiligte Aktoren	Endanwender (Studenten/Dozenten), Kunde (Höhne), Entwickler
Fachverantwortlicher	
Referenzen	Soll auf jedem gängigem Windows Betriebssystem laufen Keine Installation notwendig
Vorbedingungen	Kompilierte und ausführbare Datei
Nachbedingungen	Alle Programmkomponenten müssen funktionsfähig sein
Typischer Ablauf	Programmstart durch Doppelklick auf die ausführbare exe/jar - Datei
Alternative Abläufe	Dateien über „Öffnen mit…“ Menü mit dem Programm starten
Kritikalität	Essentiell
Verknüpfungen	
Funktionale Anforderungen	Wenn der Anwender die ausführbare Programmdatei ausführt, muss das System starten. Wenn das System erfolgreich gestartet ist, muss das System die Tabellenansicht darstellen.
Nicht-funktionale Anforderungen	

Tabellenansicht

Name des Use-Case	Tabellenansicht
Nummer	2
Paket	Oberfläche
Autor	Boris Goldstein
Version	1.1
Kurzbeschreibung	Die Tabellenansicht erlaubt das Anzeigen und Verändern der vorliegenden Daten Eingabe eigener Datensätze
Beteiligte Aktoren	Endanwender (Studenten/Dozenten), Kunde (Höhne), Entwickler
Fachverantwortlicher	
Referenzen	Max. 16 Attribute (Spalten) und 1000 Objekte (Zeilen) Automatische Werterkennung (stetig/diskret) Dezimaltrenner: Punkt Markierung des Zielattributes
Vorbedingungen	Das Programm muss richtig laufen
Nachbedingungen	Anzeigen einer Tabelle mit konsistenten Daten
Typischer Ablauf	Programm wird gestartet → Tabelle wird angezeigt Daten einfügen/einlesen
Alternative Abläufe	Von anderen Ansichten über Tableiste zur Tabellenansicht wechseln
Kritikalität	Essentiell
Verknüpfungen	Programmstart
Funktionale Anforderungen	Nachdem das System gestartet wurde, muss die Tabellenansicht angezeigt werden. Sobald die Tabellenansicht angezeigt wird, muss das System auf eine Aktivität des Anwenders warten. Wenn der Anwender Daten auf der Tabellenoberfläche löscht, muss das System die verknüpften Daten im Speicher löschen und anschließend wieder auf eine Benutzeraktivität warten. Wenn der Anwender Daten auf der Tabellenoberfläche verändert, muss das System die verknüpften Daten im Speicher verändern und anschließend wieder auf eine Benutzeraktivität warten. Wenn der Anwender Daten auf der Tabellenoberfläche eingibt, muss das System die Daten im Speicher mitspeichern und anschließend wieder auf eine Benutzeraktivität warten.

	Wenn der Benutzer Attribut in der Oberflächentabelle als Zielattribut markiert, muss das System auch im Speicher das markierte Attribut als Zielattribut kennzeichnen und anschließend auf eine Benutzeraktivität warten. Wenn der Anwender den Vorgang Tabellenansicht beendet, muss das System beendet werden.
Nicht-funktionale Anforderungen	Große, dem Inhalt angepasste Spalten Gute Lesbarkeit der Werte Farbliche Markierung von bestimmten Werten/Attributen

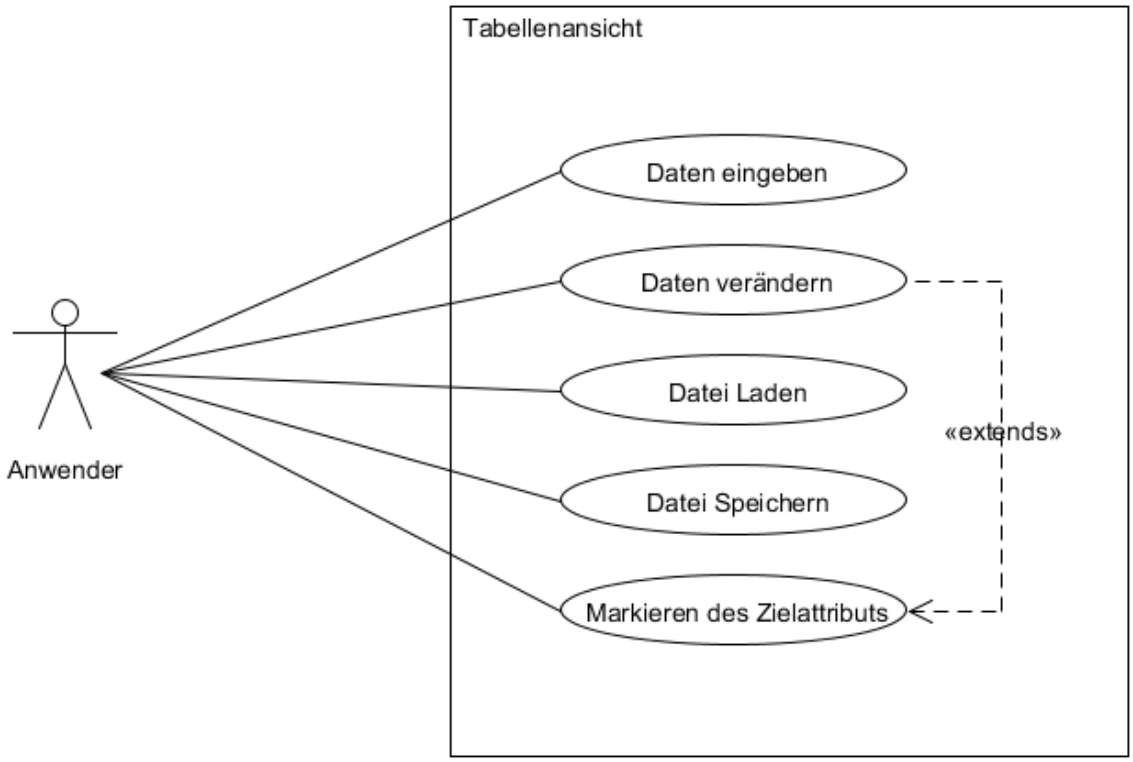

Abbildung 2: Use-Case-Tabellenansicht

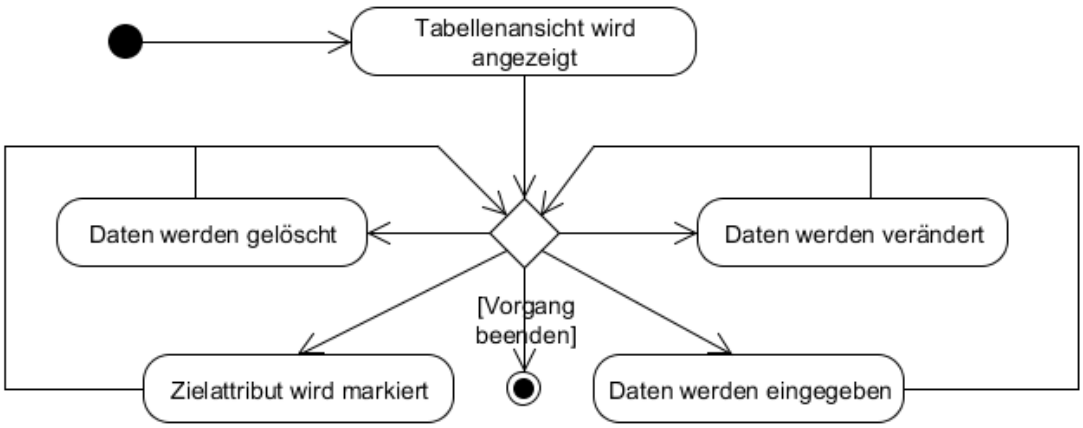

Abbildung 3: Aktivitätsdiagramm Tabellenansicht

Baum automatisch

Name des Use-Case	Baum automatisch
Nummer	3
Paket	Oberfläche
Autor	Christian Wolter
Version	1.1
Kurzbeschreibung	Der optimale Baum (nach Algorithmus) wird berechnet und ausgegeben.
Beteiligte Aktoren	Endanwender (Studenten/Dozenten)
Fachverantwortlicher	
Referenzen	-Algorithmus für optimalen Entscheidungsbaum -Vorgaben nach Aufgabenstellung
Vorbedingungen	Tabellenansicht mit Daten wird angezeigt
Nachbedingungen	Optimaler Baum wird angezeigt
Typischer Ablauf	1 Tabellenansicht anzeigen 2 Baum automatisch anklicken
Alternative Abläufe	1 Baum interaktiv wird angezeigt 2 Baum automatisch anklicken
Kritikalität	Muss / sehr wichtig
Verknüpfungen	<<includes>> Tabellenansicht
Funktionale Anforderungen	Wenn der Anwender die automatische Baumdarstellung aufruft, muss das System den nach Algorithmus optimalen Entscheidungsbaum berechnen und auf der Programmoberfläche ausgeben.
Nicht-funktionale Anforderungen	Aussagekräftige Farben Großer Schriftgrad

Baum interaktiv bearbeiten

Name des Use-Case	Baum interaktiv bearbeiten
Nummer	4
Paket	Oberfläche
Autor	Clemens Wagner
Version	1.1
Kurzbeschreibung	Die Anzeige „Baum interaktiv“ erlaubt es einem einen Entscheidungsbaum anhand selbst ausgewählter Kriterien bzw. Attribute aufzustellen.
Beteiligte Aktoren	Endanwender (Studenten/Dozenten)
Fachverantwortlicher	
Referenzen	Aufgabenstellung: „Baum interaktiv“
Vorbedingungen	Programm muss gestartet sein, valide Daten sind geladen (Tabellenansicht zeigt korrekte Daten an)
Nachbedingungen	Der Interaktive Baum wird angezeigt und ist editierbar
Typischer Ablauf	Es wird ein Knoten angezeigt, welcher alle Daten der Tabelle repräsentiert. Dieser Knoten wird angeklickt und eine Liste mit allen Attributen die dieser Knoten repräsentiert angezeigt. Das Attribut, nach dem der Baum aufgespannt werden soll wird ausgewählt und der Baum wird entsprechend weiter gezeichnet. → Es entstehen neue Knoten, für die erneut ein weiteres Aufspannungsattribut gewählt werden kann. So entsteht interaktiv ein Entscheidungsbaum.
Alternative Abläufe	Wird ein vorhandener Knoten mit Unterknoten ausgewählt und ein neues Attribut für diesen gewählt → Unterknoten verschwinden. Bei einem Attribut, mit kontinuierlichem Wertebereich muss zusätzlich ein Split-Wert angegeben werden.
Kritikalität	Muss / sehr wichtig
Verknüpfungen	Programmstart, Datei einlesen
Funktionale Anforderungen	Nach dem Start muss das System fähig sein den obersten Knoten anzeigen zu können. Das System soll dem Anwender eine Möglichkeit bieten eine Aktivität auszuführen. Das System muss fähig sein einen Knoten markieren zu können. Wenn ein Knoten markiert ist, soll das System fähig sein eine Liste mit Objekten anzeigen zu können.

	Das System soll dem Anwender eine Möglichkeit bieten ein Attribut auswählen zu können. Wenn ein Attribut ausgewählt ist, soll das System dem Anwender die Möglichkeit geben über die Art des Attributes entscheiden zu können. Wenn ein Attribut stetig ist, muss das System dem Anwender die Möglichkeit geben ein Splitwert eingeben zu können. Wenn ein Splitwert eingegeben ist, wird das System fähig sein die Änderungen zu übernehmen. Wenn das diskrete Attribut ausgewählt worden ist, wird das System fähig sein die Art des Attributes zu übernehmen. Wenn die Art des neuen Attributes ausgewählt ist, muss das System fähig sein alle Unterknoten zu löschen. Wenn die alten Unterknoten gelöscht sind, soll das System fähig sein eine neue Knotenebene zeichnen zu können. Wenn eine neue Knotenebene gezeichnet worden ist, soll das System dem Anwender die Möglichkeit bieten eine weitere Aktivität auszuführen. Wenn der Anwender alle Knoten gezeichnet hat, muss das System die Möglichkeit geben den Vorgang beenden zu können. Das System wird fähig sein die Baumdarstellung zu schließen. Wenn der Anwender den Vorgang nicht abschließen will, muss das System die Möglichkeit bieten zur Knotenauswahl zurückkehren zu können und weitere Knoten zu markieren.
Nicht-funktionale Anforderungen	Aussagekräftige Farben Großer Schriftgrad

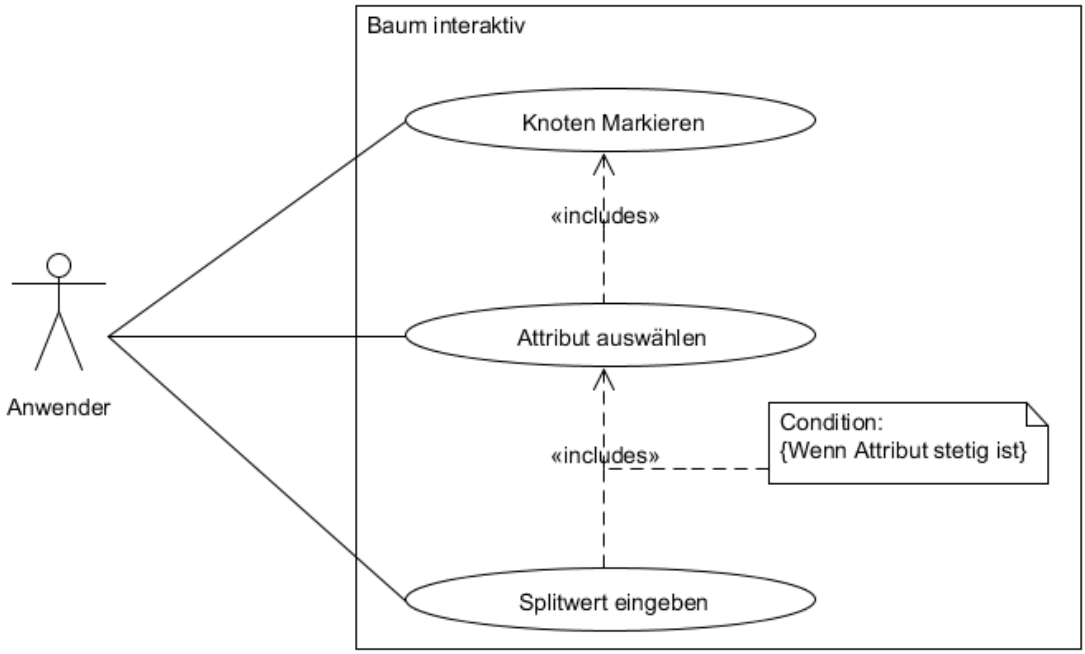

Abbildung 4: Use-Case Baum interaktiv

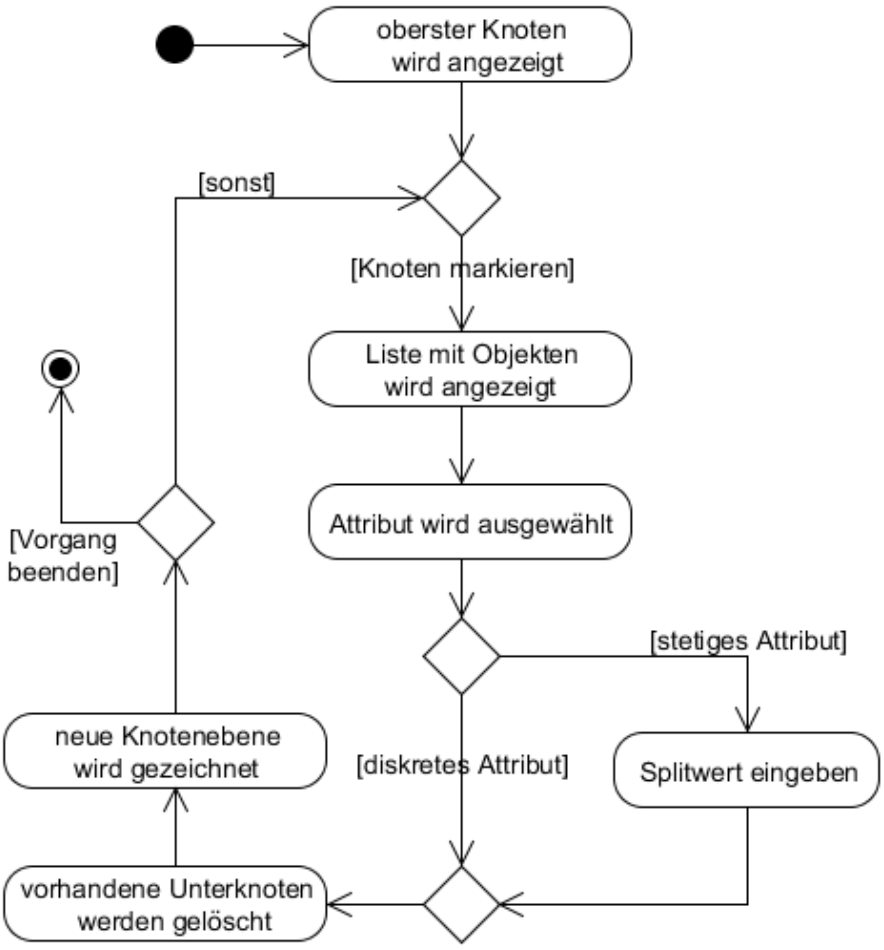

Abbildung 5: Aktivitätsdiagramm der Baumdarstellung

Regeldarstellung anzeigen

Name des Use-Case	Regeldarstellung anzeigen
Nummer	5
Paket	Oberfläche
Autor	Christian Wolter
Version	1.1
Kurzbeschreibung	Die Regeldarstellung soll die Funktionsweise des vorher angezeigten Baums verdeutlichen
Beteiligte Aktoren	Endanwender (Studenten/Dozenten)
Fachverantwortlicher	
Referenzen	-Algorithmus für optimalen Baum -selbst erstellte Regeln für interaktiven Baum
Vorbedingungen	Baum wird angezeigt (automatisch oder interaktiv)
Nachbedingungen	Regeldarstellung wird angezeigt
Typischer Ablauf	1 Baum anzeigen lassen (automatisch oder interaktiv) 2 Regeldarstellung aufrufen
Alternative Abläufe	
Kritikalität	Muss / sehr wichtig
Verknüpfungen	<<includes>> Baum automatisch Baum interaktiv
Funktionale Anforderungen	Das System muss die Möglichkeit bieten aus einer Baumdarstellen (Baum interaktiv oder Baum automatisch) zur Regeldarstellung zu wechseln Das System muss die Regeldarstellung zur zuvor angezeigten Baumdarstellung erstellen. Das System muss die korrekte Regeldarstellung zur zuvor angezeigten Baumdarstellung anzeigen Das System muss zur weiteren Bedienung bereit sein. Das System muss die Möglichkeit bieten weitere Aktionen durzuführen Das System muss dem Anwender die Möglichkeit bieten ein Objekt einzugeben Das System muss eine Eingabemaske für das Objekt anzeigen. Das System muss für das eingegebene Objekt einen Pfad durch den Entscheidungsbaum berechnen Das System muss den berechneten Pfad klar und deutliche dem Anwender in der Regeldarstellung anzeigen

	Das System muss für weitere Aktionen bereit sein. Das System muss dem Anwender die Möglichkeit bieten die Regeldarstellung zu verlassen.
Nicht-funktionale Anforderungen	Aussagekräftige Farben Großer Schriftgrad

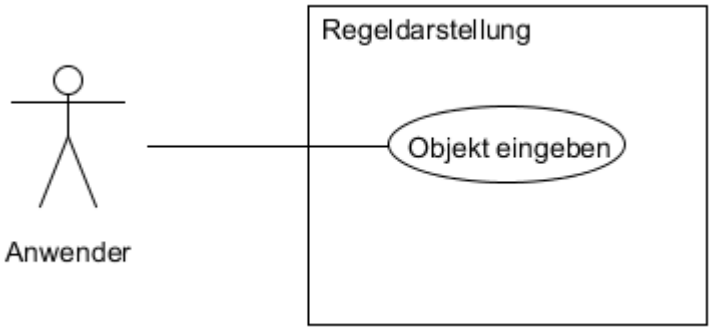

Abbildung 6: Use-Case Regeldarstellung

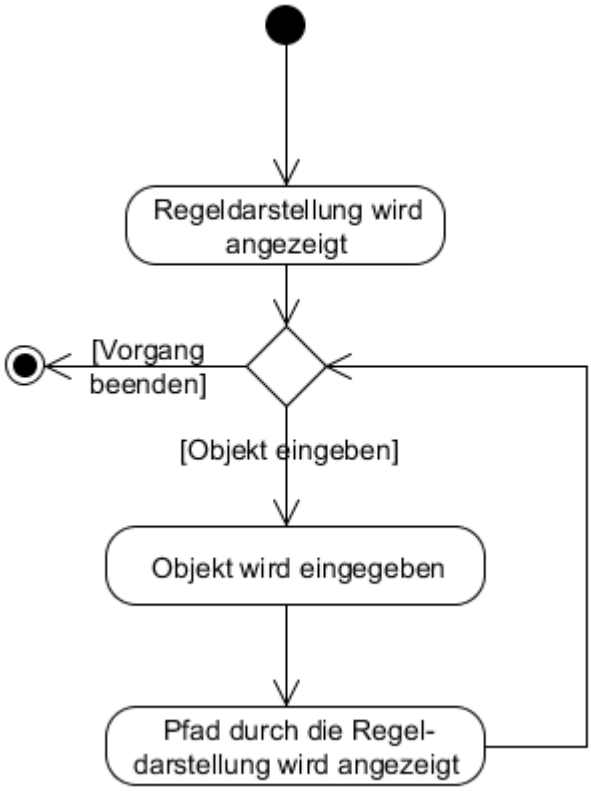

Abbildung 7: Aktivitätsdiagramm Regeldarstellung

Datei Speichern

Name des Use-Case	Datei speichern
Nummer	6
Paket	Dateiverwaltung
Autor	Boris Goldstein
Version	1.1
Kurzbeschreibung	Nutzer kann Werte, Tabellen und Entscheidungsbäume in einer Datei abspeichern
Beteiligte Aktoren	Endanwender (Studenten & Dozenten), Kunde (Höhne), Entwickler
Fachverantwortlicher	
Referenzen	Export/Import von Excel-Dateien ; Unterstützung des csv-Formats ; Trennzeichen laut Aufgabenstellung beachten
Vorbedingungen	Programm läuft, Dateien müssen vollständig und richtig eingetragen sein, Programm sollte Ergebnisse geliefert haben
Nachbedingungen	Abgespeicherte Daten sollten auch wieder eingelesen und weitergenutzt werden können
Typischer Ablauf	In richtiger Ansicht Menüpunkt „Datei speichern“ wählen → Ort und Format im Explorer-Fenster aussuchen → „Speichern“ klicken
Alternative Abläufe	-
Kritikalität	mittel
Verknüpfungen	Programmstart; * Regeldarstellung, Baum zeichnen
Funktionale Anforderungen	Wenn der Anwender eine Datei speichern möchte, muss das System vom Anwender auf der Programmoberfläche einen Speicherort für die Daten der Tabelle und ein Dateiformat mit Hilfe eines Dateimanagers anfordern. Wenn der Anwender einen Speicherort angegeben hat, muss das System die Daten der Tabelle unter dem angegeben Speicherort und Dateiformat abspeichern.
Nicht-funktionale Anforderungen	

Datei einlesen

Name des Use-Case	Datei einlesen
Nummer	7
Paket	Dateiverwaltung
Autor	Christian Wolter
Version	1.0
Kurzbeschreibung	Nutzer kann Dateien verschiedener Formate mit Daten einlesen
Beteiligte Aktoren	Endanwender (Dozenten & Studenten), Kunde (Höhne), Entwiickler
Fachverantwortlicher	
Referenzen	Trennzeichen laut Aufgabenstellung beachten!!
Vorbedingungen	Programm läuft, Datei enthält verwertbare Daten
Nachbedingungen	Tabellenansicht mit korrekten Daten angezeigt
Typischer Ablauf	Menüleiste mit „Datei-Öffnen“ → im Explorer Daten auswählen und öffnen
Alternative Abläufe	Rechtsklick auf die zu öffnende Datei → „Öffnen mit…“
Kritikalität	Sehr hoch
Verknüpfungen	Programmstart
Funktionale Anforderungen	Wenn der Anwender eine Datei einlesen möchte, muss das System den Anwender über einen Dateimanager eine kompatible Datei auswählen lassen. Wenn der Anwender eine kompatible Datei ausgewählt hat, muss das System diese Datei laden und die enthaltenen Daten auf der Oberflächentabelle anzeigen.
Nicht-funktionale Anforderungen	

8.2 Qualitätsanforderungen

An die Qualität des Systems werden hohe qualitative Anforderungen gestellt. Insbesondere sind folgende qualitätsmerkmale in Jedem Fall zu beachten:

- So ist es nicht tolerierbar, wenn das System zur Laufzeit abstürzt oder einfriert. Es muss durch entsprechende Fehlermeldungen ein Absturz des Programms verhindert werden, sodass ein Datenverlust soweit wie möglich ausgeschlossen ist
- Die Anwendung sollte alle Aktionen nachvollziehbar schnell innerhalb weniger Sekunden erledigen und bei längeren Aktionen im Rahmen von mehr als fünf Sekunden dem Benutzer darüber informieren, dass das Programm derzeit mit der Berechnung beschäftigt ist. Bei Aktionen oberhalb der 20 Sekunden ist ein Fortschrittsbalken angemessen.
- Die Oberfläche muss flüssig und gefühlt unmittelbar reagieren.

8.3 Benutzungsoberfläche

Konkrete Anforderungen an die Oberfläche

Die Graphische Oberfläche (GUI) des Programms muss diverse sehr konkrete Anforderungen erfüllen. So muss diese über eine Tabellenansicht verfügen, welche ebenfalls als Schnittstelle zum Bearbeiten der Daten dienen sollte. Des Weiteren ist in den Anforderungen direkt von einer Baumdarstellung die Rede welche ebenfalls über die GUI erreich bar ist. Die GUI enthält neben diesen Prinzipiellen Anzeigen noch weitere Menüs die beim Markieren eines Knotens oder Bearbeiten der Daten erscheinen.

Entwurf der Graphischen Oberfläche

Die Graphische Benutzeroberfläche besitzt am Oberen Fensterrand eine Menüleiste. Der Rest des Fensters wird vollständig von einer Ansicht mit verschiedenen Reitern ausgefüllt. Dort ist Standartmäßig nur der Reiter Tabellenansicht geöffnet und ausgewählt.

Tabellenansicht

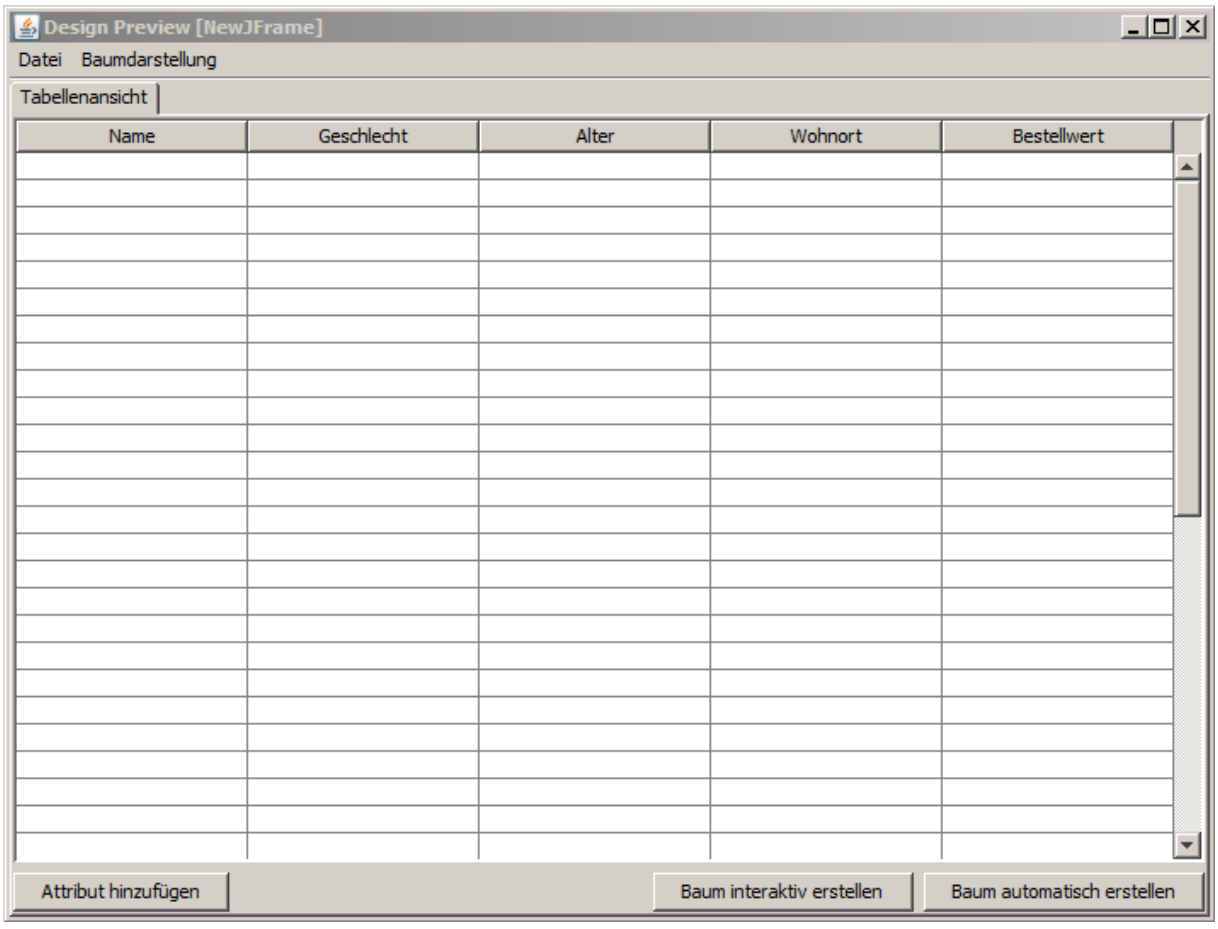

Abbildung 8: Entwurf der Tabellenansicht

Ein Entwurf des Fensters mit Tabellenansicht ist in Abbildung 8 zu sehen. Der Reiter selbst wird fast vollständig von einer Tabelle in Beschlag genommen. Diese besitzt einen Scroll-Balken, falls der Inhalt der Tabelle die größte des Reiters übersteigt. Die Tabelle enthält als letztes stets eine Leere Zeile, wodurch das Eingeben von neuen Daten möglich wird. Werden in diese Zeile Daten eingegeben, wird automatisch eine neue leere Zeile erzeugt. Ein Rechtsklick auf den Spaltenkopf

erlaubt das Umbenenne sowie Löschen eines Attributes. Neben der Tabelle an sich verfügt der Reiter Tabellenansicht noch über insgesamt drei Buttons unterhalb der Tabelle am unteren Rand des Reiters. Linksbündig ist der Button „Attribut hinzufügen“ angeordnet, welcher es ermöglicht ein Attribut sprich eine neue Spalte der Tabelle hinzuzufügen. Rechtsbündig befinden sich die Buttons zum Erzeugen der Baumdarstellung. Der Button „Baum automatisch erstellen“ veranlasst das Programm den optimalen Baum nach dem Konzept der Entropie zu berechnen. Entsprechend dient der Button „Baum interaktiv erstellen“ dazu, die Erstellung des Interaktiven Baums zu starten. Die erzeugen Baumdarstellungen werden dann in einem neuen Reiter rechts neben dem bereits vorhandenen Reiter für die Tabellenansicht dargestellt. Diese Reiter lassen sich im Gegensatz zum Reiter Tabellenansicht durch ein Kreuz rechts neben dem Titel des Reiters schließen.

Baumdarstellung

Ein Beispiel einer Baumdarstellung sieht man in Abbildung 9. Der Reiter „Baumdarstellung“ wird fast vollständig von einer Zeichenfläche eingenommen, in die der Baum gezeichnet wird. Am unteren Fensterrand besitzt der Reiter (rechtsbündig) noch den Button zum Einschalten der Regeldarstellung. Die Regeldarstellung wird direkt in den Baum hinein gezeichnet verändert somit das Aussehen des Baums. Der Button kann als Toggle-Button umgesetzt werden, welcher das Wechseln zwischen den Ansichten erlaubt. Die Darstellung des Baums entspricht dabei den konkret gegebenen Anforderungen des Auftraggebers.

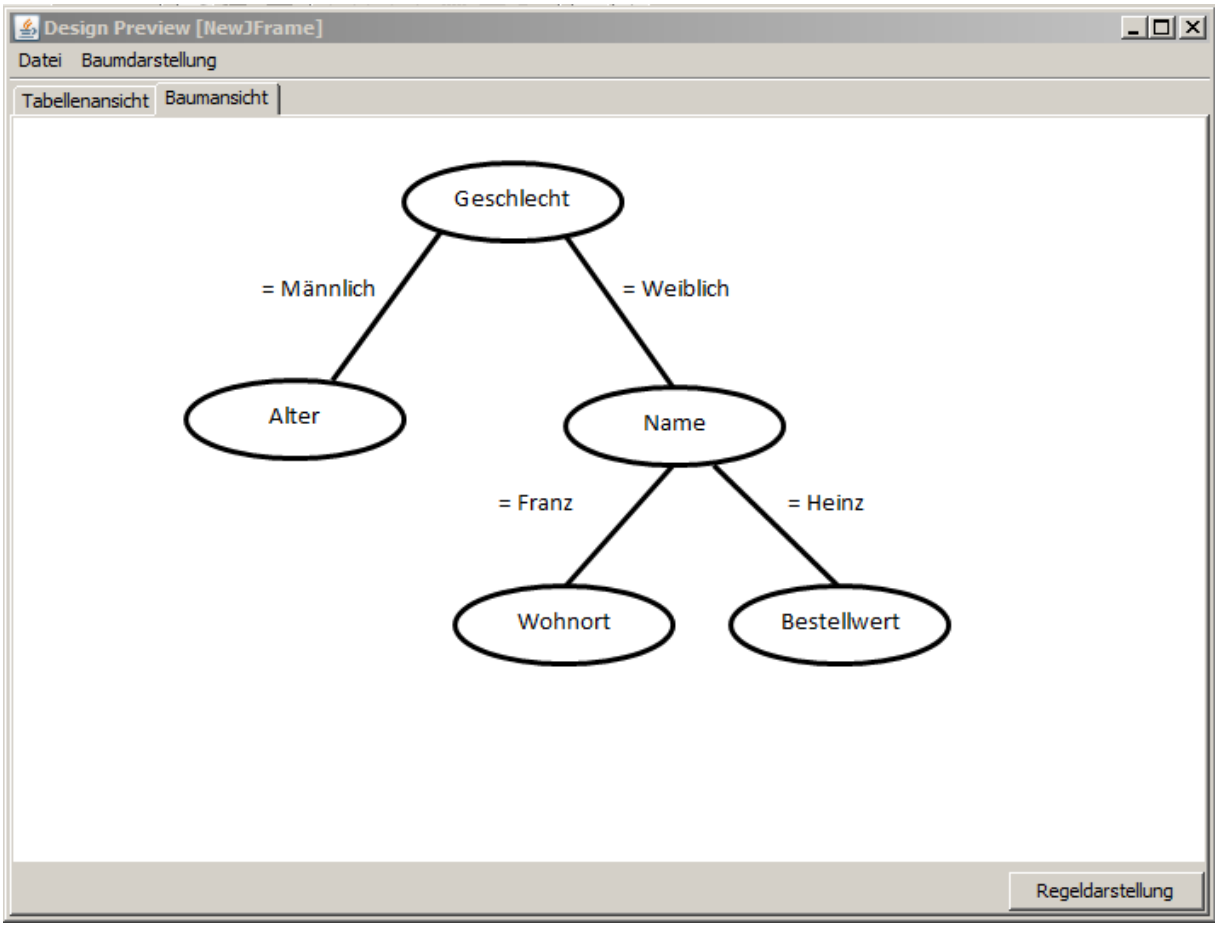

Abbildung 9: Entwurf der Baumdarstellung

Bei der interaktiven und der automatischen Baumdarstellung handelt es sich um dieselbe Darstellung (Reiter). Lediglich der Baum wird entweder bereits vollständig oder nur das Startattribut dargestellt. Daten wie die Anzahl der enthaltenen Elemente sowie die Entropie werden in einem Tooltipp beim Überfahren eines Knotens angezeigt. Zusätzlich können diese in einem Infobereich am unteren Fensterrand angezeigt werden. Wird ein Knoten ausgewählt so erscheint als Popup-Menü eine Tabelle der von diesem Knoten repräsentierten Objekte. In der angezeigten Tabelle lässt sich dann ein Attribut auswählen, nach dem weiter unterschieden werden kann. Stetige Attribute verlangen über einen Dialog die Eingabe des Split wertes.

Menüleiste

Die Menüleiste enthält die Menüs Datei, Baumdarstellung sowie eventuell ein Hilfe und About Menü.

Das Datei Menü enthält die Unterpunkte „Neu“ um eine neue leere Tabelle zu erzeugen, „Öffnen“ zum Öffnen von Dateien, „Speichern“ zum Speichern der aktuellen Tabelle, „Importieren“ zum Importieren von Excel Dateien, „Exportieren“ zum Speichern der Daten als Excel-Datei sowie „Beenden“ welcher das Programm beendet.

Das Menü Baumdarstellung enthält die Punkte „automatischen Baum erstellen“ sowie „interaktiven Baum erstellen““

8.4 Andere Nicht-funktionale Anforderungen

Insbesondere sollte bei der Entwicklung des Systems auf folgende nicht-funktionale Anforderungen geachtet werden:

- Das System muss sich übersichtlich und gut strukturiert dem Benutzer präsentieren
- Die Oberfläche sollte durch ihre Farbgebung den Benutzer auf Funktionalitäten Hinweisen und diese untermalen und keinesfalls von diesen ablenken. Die Farben sollten sich am Windows Standard Design orientieren und von diesen nur in Ausnahmefällen abweichen.
- Es sind grelle Farben in jedem Fall zu vermeiden. Vor allem auf großen Flächen sollten nur dem Auge angenehme Farben angewandt werden. Besonders zu bevorzugen sind unbunte sowie Pastellfarben zu bevorzugen.
- Schriften sollten immer durch einen Starken Kontrast vom Hintergrund abgehoben sein.
- Um das Lesen zu vereinfachen sollte stets dunkle Schrift auf hellem Hintergrund verwendet werden
- Eine Schriftgröße von 9Pt in der Graphische Benutzer Oberfläche und 10Pt in der Baumdarstellung sollte keinesfalls unterschritten werden.

9 Systemmodelle

9.1 Datenmodelle

Der folgende Abschnitt beschreibt das Datenmodell hinter der graphischen Darstellung des Entscheidungsbaums. Auf das Datenmodell der GUI wird hierbei ganz bewusst verzichtet, da dieses stark von der gewählten Programmiersprache und des entsprechenden Frameworks abhängt. Das in Abbildung 10 gezeigte Datenmodell beschreibt somit die Klassen der Kernfunktionalität.

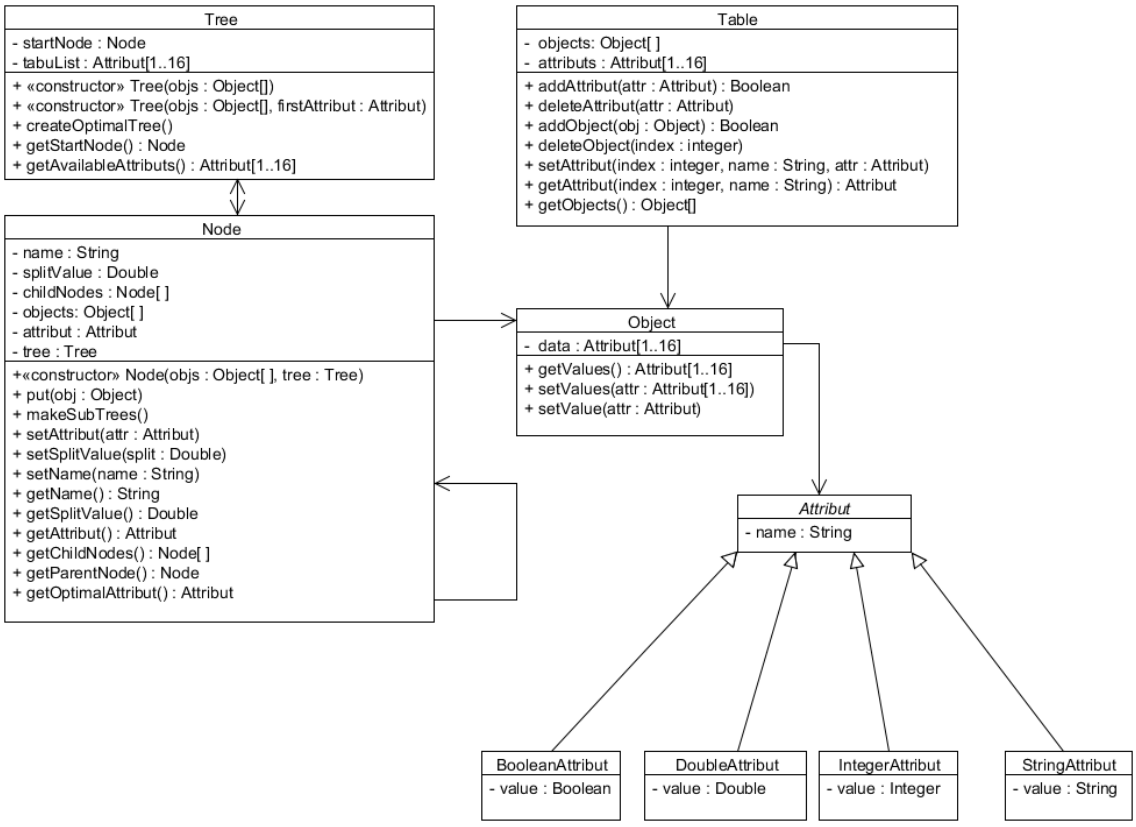

Abbildung 10: Klassenmodell

Als oberstes in der Datenhierarchie steht einerseits der Baum (Tree) sowie die Tabelle (Table) sie repräsentieren die Darstellung der Daten als Baum bzw. als Tabelle. Der Kern beider Darstellungen sind die Objekte (Object) selbst. So verfügt die Tabelle direkt über eine Liste sämtlicher Objekte. Ein Objekt entspricht dabei einem Datensatz und somit einer Zeile in der Tabelle. Jedes Objekt besitzt bis zu 16 Attribute. Die Abstrakte Klasse Abttribut fast die Klassen BooleanAttribut, DOubleAttribut, IntegerAttribut sowie StringAttribut zu einem gemeinsamen Nenner zusammen. Jeder der Unterklasse implementiert diese abstrakte Klasse. Jeder der Attribut-Klassen verfügt über den Member value. Dieser Wert entspricht bei jedem der Attribut-Klassen einem anderen Datentyp.

Die Tabelle verfügt neben der Liste von Objekten auch noch über eine Liste die alle Attribute. Anhand dieser Attributliste können neue Objekte auf Validität überprüft werden und erstellt werden. Das Hinzufügen von Objekten übernimmt ebenfalls die Tabelle selbst. Sie bietet daneben noch Funktionalitäten zum Ändern und Löschen von Attributen und Werten.

Der Baum besteht prinzipiell aus einem Verweis auf den ersten Knoten (Node) des Baums. Der Baum insgesamt besteht dann aus einzelnen Knoten (Nodes) welche über beliebig viele Kind-Knoten (ChildNodes) verfügen können und alle in der Liste objects vorhandenen Objekte auf die Kindknoten verteilen ggf. anhand des Split-Wertes (SplitValue). Der Node besitzt neben den obligatorischen Getter und Setter Methoden auch über die Methoden put() und makeSubTree(). makeSubTree entält den Algorithmus, der den optimalen Entscheidungsbaum anhand der Entropie berechnet. Put hingegen fügt ein übergebenes Objekt den Knoten hinzu. Dieser entscheidet anschließend in welchen Kind-Knoten dieses Objekt eingefügt werden soll und fügt es diesem hinzu. So wandert ein neu eingegebenes Objekt den Baum bis zu den Blättern hinab.

10 Anforderungen an die Entwicklungsumgebung

10.1 Software

Wie schon in 2.2.3 beschreiben, wird die Programmiersprache C# für die Entwicklung empfohlen, somit auch als ihre Umgebung das Visual Studio Express von Microsoft.

Da die wichtigste Anforderung an die entwickelte Software eine Windowsumgebung ist, wurde sich für eine von Microsoft selbst entwickelte Programmiersprache C-Sharp (C#) (1) und die dazu passende, von Microsoft kostenlos zur Verfügung gestellte Entwicklungsumgebung Visual Studio Express (2) entschieden, was den Vorteil hat, dass alle Zielrechner das .NET Framework von vornherein besitzen und keine externe Laufzeitumgebung vom Anwender installiert werden muss.

Zusätzliche Software für die Entwicklung ist ein Tool zur Versionskontrolle und eine Anwendung für die Continuous Integration(siehe 10.3).

Aufgrund guter Erfahrungen mit der Versionskontrollsoftware Git, wird diese für die Benutzung bei der Entwicklung empfohlen. Durch Erweiterungen kann Git auch in die Entwicklungsumgebung Visual Studio integriert werden (2)..

10.2 Hardware

Die Rechnerhardware muss die Systemanforderungen der Entwicklungsumgebung und des Betriebssystems erfüllen.

Empfohlen wird ein Prozessor der neueren Generation und ausreichend Arbeitsspeicher, um alle Funktionen der Software ohne Performanceverlust testen und ausführen zu können.

Vorausgesetzt wird ein Monitor zur Ausgabe, Maus und Tastatur zur Eingabe der Software.

10.3 Orgware

Für den Betrieb der Continuous Integration wird ein CI-Server benötigt. Dazu gibt es eine Vielzahl von CI-Serverangebote mit einem Web-Interface, Empfehlenswert sind die Anwendungen Jenkins (2) und Teamcity (3). Es gibt auch die Möglichkeit einen eigenen CI-Server aufzusetzen.

Jenkins hat den Vorteil der Unterstützung von Git (10.1), benötigt aber zusätzliche Plug-Ins für die Programmierung unter dem .NET Framework. Teamcity dagegen ist von vornherein mit Visual Studio verträglich. Beide CI-Server geben Möglichkeiten zum Verwalten und grafischer Darstellung von Commits, Tests und Änderungen der Entwicklung..

11 Ergänzungen

<Ihr Text mit den Anforderungen an das System, die in den obigen Kapiteln keinen Platz gefunden haben, z.B. spezielle Installationsbedingungen, Datenbankbeschreibungen, zu berücksichtigende Normen, Vorschriften, Patente und Lizenzen, etc.>

Literaturverzeichnis

1. Universität Jena Folien zu C#. [Online] [Zitat vom: 01. 11 2012.] http://psc.informatik.uni-jena.de/publ/CSharp-it0402.pdf.

2. Visual Studio Express. [Online] [Zitat vom: 01. 11 2012.] http://www.microsoft.com/visualstudio/deu/downloads#d-2012-express.

3. Visual Studio Gallery. [Online] [Zitat vom: 01. 11 2012.] http://visualstudiogallery.msdn.microsoft.com/63a7e40d-4d71-4fbb-a23b-d262124b8f4c.

4. Jenkins Homepage. [Online] [Zitat vom: 01. 11 2012.] http://jenkins-ci.org/.

5. TeamCity Homepage. [Online] [Zitat vom: 01. 11 2012.] www.jetbrains.com/teamcity/.

6. **Queckbörner, Sabine.** Folien zu Data Mining der Universität Kaiserslautern. [Online] [Zitat vom: 31. 10 2012.] http://wwwlgis.informatik.uni-kl.de/archiv/wwwdvs.informatik.uni-kl.de/courses/seminar/WS0304/ausarbeitung1.pdf.

Anhang A: <Titel>

<Und so viele Weitere, falls nötig>