# Funktionale Anforderungen:

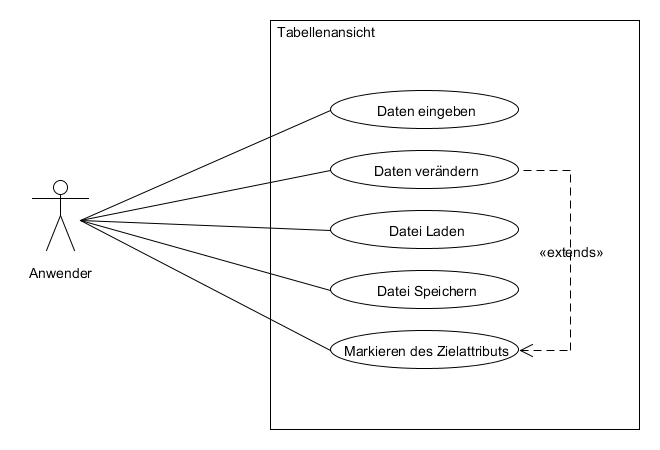
## Programmstart

|  |  |
| --- | --- |
| **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** |  |

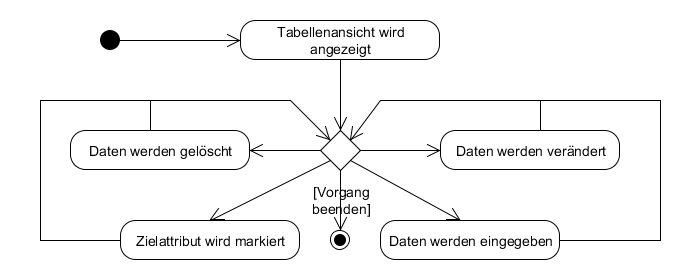
## Tabellenansicht

|  |  |
| --- | --- |
| **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 |

Anwendungsfalldiagramm für Tabellenansicht



Aktivitätsdiagramm für Tabellenansicht:



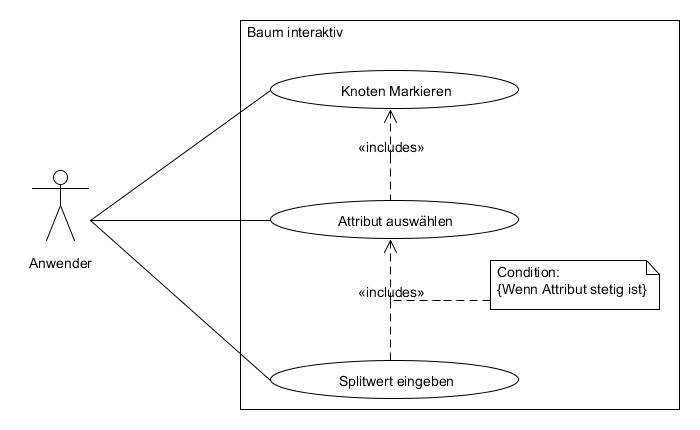
## Baum automatisch

|  |  |
| --- | --- |
| **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 |

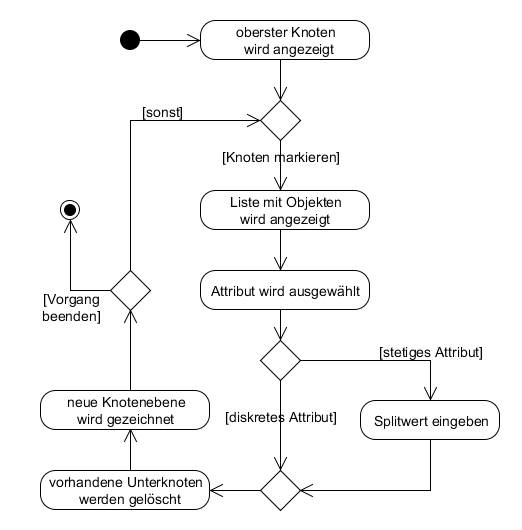
## Baum interaktiv bearbeiten

|  |  |
| --- | --- |
| **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** |  |
| **Nicht-funktionale Anforderungen** | Aussagekräftige Farben  Großer Schriftgrad |

Anwendungsfalldiagramm für interaktive Baumdarstellung:



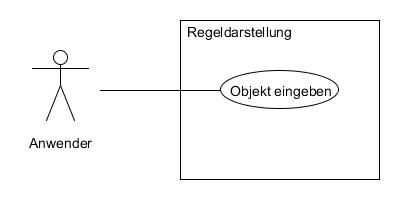
Aktivitätsdiagramm für interaktive Baumdarstellung



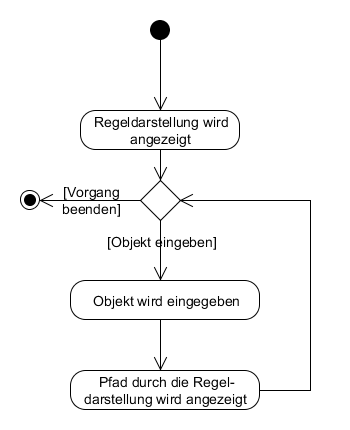
## Regeldarstellung anzeigen

|  |  |
| --- | --- |
| **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 |

Anwendungsfalldiagramm für Regeldarstellung:



Aktivitätsdiagramm für 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 ; Untersüttzung 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** |  |