Developer Handbook

Inhaltsverzeichnis

[Einleitung 2](#_Toc474821207)

[Purpose and scope 2](#_Toc474821208)

[Changes 2](#_Toc474821209)

[Structure of document 2](#_Toc474821210)

[Related documents 2](#_Toc474821211)

[Definitions and abbreviations 2](#_Toc474821212)

[Open issues 2](#_Toc474821213)

[General Design Notes 2](#_Toc474821214)

[Design Of Framework 2](#_Toc474821215)

[Design of Calculators 2](#_Toc474821216)

[Error handling 2](#_Toc474821217)

[Initialization and Clean-up 2](#_Toc474821218)

[Test harness 2](#_Toc474821219)

[Developer Handbook Server 2](#_Toc474821220)

# Einleitung

Source Code: <https://github.com>; User: MarKogler; Repository: mDecisioMat

## Purpose and scope

Zweck der Entwicklung war es ein Programm zu entwickeln, welches Regelsätze zur Entscheidungsfindung zentral verwaltet und Clients an unterschiedlichen Stellen ermöglicht auf diese Regelsätze zuzugreifen und unter Verwendung dieser Entscheidungen zu treffen.

## Changes

Die vorliegende Version ist V1.0

## Structure of document

Dieses Dokument ist kurz gehalten und gibt kurze allgemeine Hinweise zum Design, veranschaulicht das Gerüst des Designs und geht anschließend auf Relevante Details ein um die einzelnen Komponenten genauer zu erklären.

## Related documents

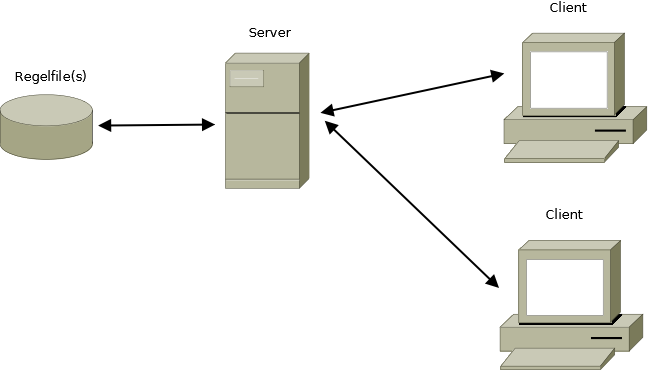
Es wird auf die Regeln für Regelfiles und auf die Installations- und Betriebsanleitung verwiesen. Außerdem wird auf das File TestListe verwiesen.

## Open issues

Der Client kann nach einem Neustart des Servers keine Verbindung zu diesem mehr herstellen und muss ebenfalls neu gestartet werden. Es wurde bei dieser Entwicklung bewusst auf einen Administrator Client verzichtet. Dieser würde den Vorteil bringen, neue Regelfiles auch ohne Neustart einlesen zu können. Weiters kann der Code noch optimiert werden (z.B.: Methoden sind voraussichtlich kleiner/granularer gestaltbar,…).

# General Design Notes

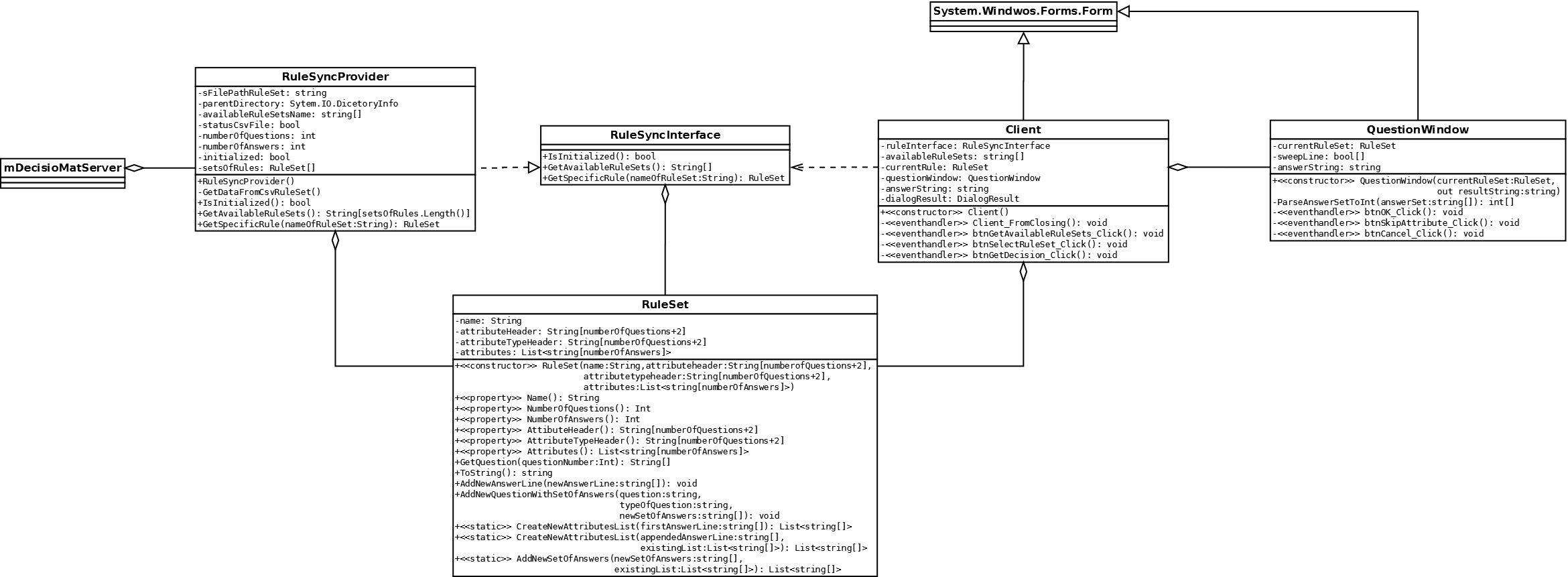
Das Design stellt von der Struktur her eine Art Data Repository dar. Der Server hat als einziger Zugriff auf die Regelfiles. Die Clients können nur fertig eingelesene Regelfiles beim Server abfragen, womit auch die Datenstruktur festgelegt ist (Klasse „RuleSet“).



# Design Of Framework

## Object Model

Der grundlegende Aufbau besteht aus den Beiden Programmteilen Client und Server. Einen Überblick über die Verwendeten Klassen bietet das nachfolgende Klassendiagramm. Client und Server verwenden die Klasse „RuleSet“ und das „RuleSyncInterface“. Der Server hostet dabei den „RuleSyncProvider“ welcher vom „RuleSyncInterface“ vererbt ist. Der Client besteht aus den beiden Fenstern „Client“ und „Question Window“.



## Description of Modules

Der Server bzw. der „RuleSyncProvider“ liest bei Initialisierung alle gefundenen .csv Dateien im Ordner „RuleSets“ ein und speichert die gültigen Regelsätze als Array von RuleSet Instanzen ab. Weiters bietet er den Clients die Möglichkeit die Namen der vorhandenen Regelsätze abzufragen, sowie einen gewünschten Regelsatz zu erhalten.

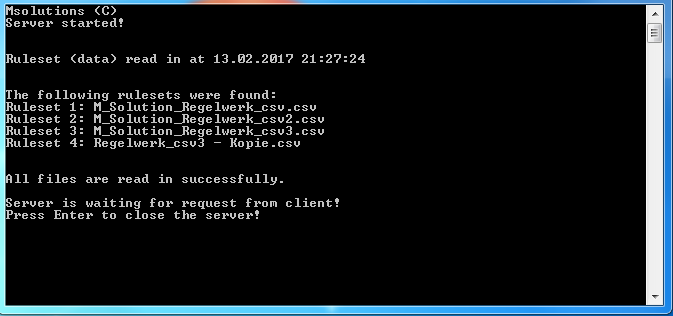
Der Client ermöglicht dem Nutzer die aktuell vorhandenen Regelsätze abzufragen und zeigt diese grafisch an. Weiters wird dem Nutzer ermöglicht einen dieser Regelsätze auszuwählen und diesen vollständig anzuzeigen, sowie eine Entscheidungsfindung anhand des ausgewählten Regelsatzes durchzuführen. Das Ergebnis wird dem Benutzer in Textform angezeigt.

# Design of Calculators

## Server

Der Server wird als Konsolenanwendung ausgeführt und besteht aus dem Hauptprogramm "Programm.cs", in welchem die Serverapplikation "ServiceHost" erstellt wird. Der Server wird erstellt und wartet auf die Anfrage eines Clients. Des Weiteren besteht der Server aus dem "RuleSyncProvider.cs". Diese cs-Datei hat folgende Aufgaben:

Im "RuleSyncProvider.cs" werden CSV-Files mit beliebigen Namen eingelesen und die in den CSV-Files abgespeicherten Regelsätze im Array "setsOfRules" vom Datentyp RuleSet abgespeichert. Damit die CSV-Files eingelesen werden können, ist es erforderlich einen Ordner mit dem Namen "RuleSets" im "Releas"-Ordner des Servers zu erstellen. Ausschließlich innerhalb dieses Ordners sind die Regelsätze abzuspeichern.   
Die Datei "RuleSyncProvider.cs" stellt die beiden Methoden "GetAvailableRuleSets" und "GetSpecificRuleSet" zur Verfügung. Mit ersteren werden alle eingelesenen und verfügbaren Regelsätze an den Client bereitgestellt. Wird am Client ein Regelsatz aus den verfügbaren ausgewählt, wird über die Methode "GetSpecificRuleSet" der Inhalt des ausgewählten Regelsatzes vom Server an den Client transferriert und für weitere Anwendungen am Client verfügbar gemacht.



## Client

Eine genaue Beschreibung der Funktion aus Nutzersicht ist der Installations- und Betriebsanleitung zu entnehmen.

Der Client speichert immer nur den aktuell ausgewählten Regelsatz auf der Variable currentRuleSet sowohl im Fenster „Client“ als auch im Fenster „QuestionWindow“. Das Fenster „Client“ legt bei Initialisierung den Channel zum Server an und enthält alle Kommunikationsschnittstellen zum Server. Das Fenster „QuestionWindow“ kommuniziert nicht mit dem Server. Dieses dient generell nur dazu dem Nutzer die Fragen zu stellen und die Eingaben entgegenzunehmen und zu überprüfen. Die Methode „AskQuestion“ übernimmt dabei das Ausgeben der aktuell gültigen Frage und das Beenden des Dialogs, wenn die Sequenz beendet ist. Das Entgegennehmen der Eingabe erfolgt im Eventhandler für das Klick Ereignis des Buttons „OK“. Der Button „skip attribute“ kann eine Frage überspringen und ignoriert dabei alle Auswahlen des Benutzers. Genauere Ausführen sind dem Code und den Kommentaren zu entnehmen

# Error handling

## Server

Der Server ignoriert Files, welche nicht in einen gültigen Regelsatz überführt werden können

## Client

Der Client hat ein GUI das entsprechend gestaltet wurde, dass der Nutzer keine Fehlereingaben machen kann. Weiters werden Verbindungsfehler zwischen Client und Server abgefangen und dem User durch eine Fehlermeldung angezeigt.

# Initialization and Clean-up

Der Code ist in der vorliegenden Form vollständig kommentiert und Kompiliert.

# Test harness

Die einzelnen Komponenten wurden mittels Unit-Tests getestet. Dafür wurde ein Testtreiber für die Klasse RuleSet angelegt. Client und Server dienten einander gegenseitig als Testumgebung. Weiters wurde auf Basis der Requirements eine Testliste erstellt, welche die Erfüllung der Anforderungen sicherstellen soll.