HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Luware
LEAN UNIFIED

WF Webviewer

Studienarbeit

Abteilung Informatik

Hochschule für Technik Rapperswil

Frühjahrssemester 2015

Autor(en):	Martin Eisenhammer, Josua Stähli
Betreuer:	Prof. Beat Stettler
Projektpartner:	Luware AG Zürich

1 Inhaltsverzeichnis

2 Abstract

Die Firma Luware AG in Zürich hat für eines ihrer Produkte eine WPF-Applikation entwickelt, die es ermöglicht Workflows zu betrachten und zu ändern. Weil diese WPF-Applikation auf Windows beschränkt ist, soll diese längerfristig durch eine Webapplikation abgelöst werden.

Für die Luware AG wurde im Rahmen dieser Studienarbeit eine Webapplikation entwickelt, die Workflows für die Konfiguration einer Anrufbehandlung aus einem XAML auswertet und übersichtlich als Ablaufdiagramm anzeigt. Es handelt sich um eine Client-Server Applikation, welche plattformunabhängig ist und auf gängigen Webbrowsern, wie Google Chrome, Internet Explorer und Mozilla Firefox läuft.

Serverseitig wurde die Applikation mit C# und ASP.NET MVC und clientseitig mit JavaScript und AngularJS entwickelt. Der Benutzer kann einen in XAML beschriebenen Workflow als Datei hochladen oder in eine Eingabemaske den Code per Copy and Paste eingeben. Der XAML-Code wird an den Server geschickt und ausgewertet. Der Server sendet den Workflow anschliessend mit der Konfiguration der einzelnen Elemente als JSON an den Client zurück, wo der Workflow grafisch dargestellt wird. Der Benutzer kann ein Element des Workflows auswählen und dessen Eigenschaften betrachten. Ausserdem können die Elemente per Drag & Drop verschoben werden und der Benutzer kann in den Workflow hinein- oder hinauszoomen. Der Workflow kann in zwei verschiedenen Ansichten dargestellt werden: einmal von links nach rechts oder von oben nach unten.

Als Workflow-Elemente werden die von Luware bereitgestellten Elemente unterstützt. Es können auch neue Elemente hinzugefügt werden.

Es wurde auch geprüft, ob das Ändern der Konfiguration von einer Activity möglich ist.

3 Projektbeschreibung

3.1 Vorgegebene Arbeitsbeschreibung

Diese Studienarbeit basiert auf der folgenden vorgegebenen Arbeitsbeschreibung:

Studiengang:	Informatik (I)
Semester:	FS 2015 (16.02.2015-13.09.2015)
Durchführung:	Studienarbeit
Fachrichtung:	Internet-Technologien und -Anwendungen
Institut:	INS: Institut für vernetzte Systeme
Gruppengrösse:	2 Studierende
Status:	zugewiesen
Verantwortlicher:	Stettler, Beat
Betreuer:	Stettler, Beat
Gegenleser:	tbd
Experte:	tbd.
Industriepartner:	Luware AG
Ausschreibung:	Ausgangslage: Seit .NET 3.0 bietet Microsoft mit der Windows Workflow Foundation (WF) eine Möglichkeit, komplexe Abläufe mittels Ablaufdiagramme darzustellen und deklarativ zu programmieren. Am Ende entsteht eine XML Struktur, welche in einer .NET Applikation mittels APIs bequem eingelesen und verwendet werden kann. Exkurs Windows Workflow Foundation (Quelle: http://de.wikipedia.org/wiki/Windows_Workflow_Foundation, 15.12.2014) Windows Workflow Foundation (WF, früher auch WWF und WinWF) ist eine Bibliothek innerhalb der .NET-Programmierumgebung von Microsoft. Sie dient der einfacheren Entwicklung von programmgesteuerten, komplexen Arbeitsabläufen (engl. Workflows). Diese Art von Software wird auch als Workflow-Anwendungssoftware bezeichnet. Die Windows Workflow Foundation ermöglicht eine deklarative Programmierung. Hierbei wird eine grafische Beschreibung des Ablaufes erstellt, anstatt ihn in Form von Programmcode zu modellieren. Die WF schafft damit höhere Abstraktionsebenen, die den Entwickler weg von der Codierung und näher zu den fachlichen Geschäftsprozessen rückt. Für die Einbindung des WF Designers bietet Microsoft WPF Bibliotheken an, welche eine Integration in eigene WPF Applikationen ermöglicht.

	Problemstellung: Leider bietet Microsoft auch mit der neusten WF Version des .net Frameworks 4.5 keine Integration in die von Microsoft unterstützten Webtechnologien (ASP.net / MVC) an. Zudem existieren aktuell in den Foren und Blogs keine Hinweise darauf, dass sich dies auch mit zukünftigen .NET Versionen ändern wird. Überprüft man jedoch das Internet auf diese Anforderung, stellt man fest, dass durchaus das Bedürfnis einer Webintegration des Designers vorhanden ist. Eine Firma mit diesem Bedürfnis ist die Luware AG (http://www.luware.net), welche sich auf die Entwicklung von Microsoft Lync basierten Lösungen spezialisiert hat. Das Unternehmen entstand als Startup aus einer erfolgreichen Bachelor Arbeit der HSR und beschäftigt rund 4 Jahre nach der Gründung 14 Entwickler. Ein Produkt der Luware verwendet unter anderem die WF von Microsoft und ermöglicht dadurch die einfache Konfiguration der Anrufbehandlung. Aktuell steht hierfür jedoch nur eine WPF Applikation zur Verfügung, welche in der nächsten Zeit durch eine Weblösung ersetzt werden soll. Erwartetes Resultat: Primär: Ziel dieser Arbeit soll es sein, eine .net Webapplikation (MVC etc.) zu entwickeln, welche existierenden WF XML Code auswertet und in einem Ablaufdiagramm darstellen kann. Dazu sollen die genaue Struktur inkl. der Parameter ausgelesen und dargestellt und alle möglichen Kombinationen unterstützt werden. Das Laden der Workflows darf dabei nicht mehr Zeit als mit dem WPF Designer beanspruchen. Diese Applikation soll dann als POC in die existierende Webapplikation des Produktes eingebunden werden. Je nach Zeit und Erfolg des Primären Ziels, kann dann an den sekundären Zielen gearbeitet werden. Sekundär1: Es soll die Machbarkeit der Ablösung des kompletten WF WPF Designers anhand eines POC überprüft werden. Dabei muss die Web Applikation das Auslesen, Verändern, Erstellen und Speichern von Workflows unterstützten. Die Bedienung soll dabei grösstenteils mittels Drag&Drop geschehen. Für die technologische Umsetzung kann unter Absprache des Betreuers auch auf Drittanbieter-Bibliotheken zugegriffen werden. Sekundär2: Schon heute existieren zu den einzelnen Workflows Reporting Daten, anhand dessen sich den Weg von Anrufen durch den Workflow nachverfolgen lässt. Mit diesen Reporting Daten soll ein geladener Workflow so erweitert werden, dass sich daraus ableiten lässt, wo z.B. die meisten Anrufe

	verloren gegangen sind oder welche Wege der grösste Teil der Anrufe nimmt. Vorgehen: - Technologieanalyse o Analyse von Microsoft WF o Analyse XML Struktur der Luware o Analyse von Webframeworks auf .NET Basis für eine Umsetzung - Umsetzung Primäres Ziel - Umsetzung Sekundäres Ziel
Voraussetzungen:	Voraussetzungen: - Kenntnisse in .NET Technologien (C#) - Verwendung der aktuellsten .NET Version (4.5.2) - Vorzugsweise Kenntnisse in Web Technologien (Angular JS, MVC, ASP.NET)

3.2 Vision

3.2.1 Einleitung

Das Projekt "WF Webviewer" wird im Rahmen einer Studienarbeit der HSR durchgeführt und wurde von der Luware AG in Auftrag gegeben. Das Ziel ist eine Webapplikation, welche mit der Windows Workflow Foundation (WF) erstellten XAML Code einlesen und grafisch darstellen kann.

3.2.2 Ausgangslage

Das aktuelle Produkt von Luware hat einen WPF WF Designer integriert, welcher das Erstellen und Abändern von Workflows erlaubt. Diese Workflows enthalten viele von der Luware individuell erstellte Elemente. Erstellt werden die Workflows deklarativ als XAML. Eine Unterstützung des WF Designers in ASP.net gibt es nicht und es gibt auch keine Hinweise, dass sich das bald ändern könnte.

3.2.3 Motivation

Der WPF WF Designer ist an Windows gebunden. Eine zeitgemässere Lösung wäre darum eine Webplattform, welche dieselben Funktionen wie der WPF WF Designers bietet. Diese Studienarbeit soll ein Proof of Concept für eine solche Plattform sein, indem ein Teil davon, also die Darstellung vorhandener Workflows im Web, umgesetzt wird.

3.2.4 Ziel

Das primäre Ziel der Arbeit ist eine Webapplikation, welche WPF XAML Code auswertet und in einem Ablaufdiagramm darstellt. Zudem soll geprüft werden, ob auch das Abändern von Properties in einem Workflow umsetzbar wäre.

3.2.5 Optionale Erweiterungen

- Überprüfung der Machbarkeit der kompletten Ablösung des WPF WF Designers
- Auswerten und Darstellen von Reporting Infos

3.2.6 Projektorganisation

Projektmitglieder

Die Studienarbeit wird von Martin Eisenhammer und Josua Stähli bearbeitet.

Betreuer

Der betreuende Dozent ist Prof. Beat Stettler.

Industriepartner

Auftraggeber ist die Luware AG Zürich. Ansprechpartner sind Michael Jakob und Christoph Rebsamen.

4 Management Summary

4.1 Ausgangslage

Die Luware AG aus Zürich hat für ihr aktuelles Produkt eine WPF-Applikation entwickelt, die es ermöglicht Workflows zu betrachten und zu editieren. Die Workflows enthalten von Luware erstellte individuelle Elemente und werden deklarativ als XAML erstellt.

Der eingesetzte Workflow Editor wird jedoch nicht durch ASP.NET unterstützt. Die von Luware entwickelte WPF-Applikation ist deshalb nur auf Windows lauffähig und somit plattformabhängig, eine Umsetzung als Webapplikation wäre wünschenswert.

Im Rahmen der Studienarbeit wurde eine Webapplikation entwickelt, die einen in XAML geschriebenen Workflow auswertet und als Ablaufdiagramm grafisch darstellt. Diese Webapplikation ist plattformunabhängig und kann damit auf verschiedenen Betriebssystemen und Browsern verwendet werden.

4.2 Vorgehen, Technologien

Nach einer Einarbeitung in die Windows Workflow Foundation wurden die Anforderungen analysiert. Weiterhin wurden die Windows Workflow Foundation und die Struktur der von Luware zur Verfügung gestellten Workflows analysiert. Anschliessend wurden die von Luware eingesetzten Webtechnologien (Client/Server) und Flowchart Frameworks für JavaScript studiert. Von den von Luware eingesetzten Webtechnologien wurde ASP.NET MVC für den Server und AngularJS für den Client ausgewählt.

Bei der Analyse der Flowchart Frameworks gab es eine grosse Auswahl von verschiedenen Frameworks. Es wurde das Framework Cytoscape.js gewählt, weil es sich gut zur Darstellung von Workflows eignet. Ausserdem können Elemente mit Hilfe dieses Frameworks übersichtlich angeordnet werden.

Nach der Analysephase wurde die Architektur ausgearbeitet und die Webapplikation erstellt. Auf der Clientseite wurde mit JavaScript gearbeitet und der Server wurde in C# implementiert.

Zuerst wurden die Darstellung von Activites eines Workflows und die Beziehungen zwischen Activites realisiert. Anschliessend wurde das Laden und Parsen eines XAMLs umgesetzt. Weiterhin wurde das Auslesen und Anzeigen der Konfiguration einer Activity realisiert. Es wurde darauf Wert gelegt, dass neue Activites in die Webapplikation mit angemessenem Aufwand hinzugefügt werden können. Eine Anleitung für die Benutzung der Webapplikation sowie eine Installations- und Konfigurationsanleitung wurden verfasst. Zum Schluss wurde das User Interface gemäss eines von Luware festgelegten Styleguides angepasst.

4.3 Ergebnisse

Die umgesetzte Webapplikation liest einen Workflow im XAML-Format ein und wandelt den Workflow in ein JSON um. Mit Hilfe des JSON und des Framework Cytoscape.js wird der Workflow im Browser dargestellt. Es können auch die Eigenschaften von Elementen des Workflows angezeigt werden und es können der gesamte Workflow oder einzelne Elemente verschoben werden.

Die Webapplikation ist als Client-Server-Applikation entwickelt worden. Clientseitig wird der Workflow eingelesen und mittels Cytoscape.js dargestellt. Serverseitig wird das XAML ins JSON geparst.

Das Ändern der Konfiguration von Workflow-Elementen wurde in Form eines Proof of Concept mit mehreren Ansätzen behandelt. Auf Grund von häufigen Compiler-Fehlermeldungen bei der Implementierung konnte diese Funktion nicht mehr realisiert werden.

Das Implementieren einer Reportingfunktion zur Darstellung von Wegen von Anrufen durch den Workflow konnte aus Zeitgründen nicht mehr angeschaut werden.

4.4 Ausblick

Wir sind der Meinung, dass es möglich ist, die WPF-Applikation von Luware zukünftig zumindest teilweise durch eine Webapplikation abzulösen. Das Ändern der Konfiguration einer Activity scheint mit den gefunden Ansätzen möglich. Damit könnte ein Workflow verändert, gespeichert und wieder geladen werden. Auch das Anzeigen von Wegen von Anrufen durch den Workflow könnte in Zukunft realisiert werden, das eingesetzte Framework Cytoscape.js unterstützt beispielsweise schon den Einsatz von Animationen. Das Umsetzen eines Online-Editors, der das Hinzufügen, Entfernen und Neuverbinden von Activities unterstützt, könnte jedoch noch einen grossen Aufwand bedeuten, bei dem neue Schwierigkeiten wie das Behandeln von Parallelität gehandhabt werden müssen.

5 Projektmanagement

5.1 Vorgehen

Für das Projektmanagement verwenden wir Scrum, welches einen Rahmen für das Vorgehen beim Planen und Umsetzen des Projektes gibt. Scrum basiert auf der agilen Software-Entwicklung, was bedeutet, dass wir während der Entwicklung eng mit dem Auftraggeber zusammenarbeiten.

Die User Stories wurden von der Luware bereits vorgegeben und werden im Verlaufe des Projektes auch von ihnen verwaltet und priorisiert. Die Aufwandschätzung und das Festlegen der dazugehörenden Tasks werden von uns als Entwicklerteam gemacht.

5.2 Kommunikation

Mit der Luware AG finden regelmässige Meetings statt, einerseits die im Scrum etablierten Stand-up Meetings und andererseits eine Demo am Ende eines jeden Sprints. Die Stand-up Meetings wurden zwei Mal wöchentlich jeweils per Skype abgehalten, die Demos fanden meist direkt bei Luware in Zürich am Ende jedes Sprints statt.

5.3 Projektplanung

Die User Stories und Tasks werden in einem von der Luware zur Verfügung gestelltem Team Foundation Server verwaltet.

5.3.1 Sprint 1

Dauer: 23. Februar - 8. März

User Stories:

- Analyse der Requirements
- Analyse WF Foundation
- Analyse Luware Workflows
- Analyse Webtechnologien (Server-/Clientseitig)

5.3.2 Sprint 2

Dauer: 9. März - 22. März

User Stories:

- Basis Architekturdefinition
- Darstellung einzelner WF-Element I

5.3.3 Sprint 3

Dauer: 23. März - 8. April

User Stories:

- Darstellung der Beziehungen zwischen den Elementen I
- Darstellung einzelner WF-Elemente II
- Dokumentation der Analyse Resultate
- Laden und Parsen WF-XML

5.3.4 Sprint 4

Dauer: 8. April - 19. April

User Stories:

- Detail Architekturdefinition
- Darstellung der Konfiguration/Properties der einzelnen Elemente

- Auto Alignment

5.3.5 Sprint 5

Dauer: 20. April - 3. Mai

User Stories:

- Ergänzung Darstellung Properties
- Darstellung vom Multilanguage Properties
- Darstellung der Mappings auf die Ausgänge bestimmter Elemente
- Benutzerdokumentation

5.3.6 Sprint 6

Dauer: 4. Mai - 17. Mai

User Stories:

- Support für zusätzliche/neue WF-Elemente
- Dokumentation Installation/Konfiguration
- UI polieren
- POC Changes der Properties ermöglichen

5.3.7 Offene Backlog Items

- POC Einbindung in LUCS-FE (wird von der Luware AG übernommen)
- POC Logparser -> Lemming Count

6 Anforderungen

6.1 Allgemein

Es soll eine Webapplikation entwickelt werden, welche WF Workflows der Luware einliest und im Browser grafisch darstellt.

6.2 Funktionale Anforderungen

6.2.1 Unterstützung aller WF Elemente von Luware

Die Webapplikation soll alle von der Luware in ihren Workflows verwendeten Elemente einlesen und im Workflow darstellen können. Viele dieser Elemente wurden von der Luware individuell erstellt.

6.2.2 Laden des XML Codes über Browser

Die Webapplikation soll das Laden eines Workflows per Upload der XML Datei sowie per Copy & Paste des XML Codes unterstützen.

6.2.3 Multibrowser Support

Es sollen mindestens die Browser Google Chrome 39 und Microsoft Internet Explorer 10 unterstützt werden.

6.2.4 Benutzerfreundliche Darstellung

Die Workflows sollen übersichtlich dargestellt werden was beispielsweise mit einer automatischen Ausrichtung der Elemente in eine Art Tabellenstruktur erreicht werden soll.

6.2.5 Darstellung der Parameter

Die Konfiguration der Elemente muss über einen Expander oder ein Popup ersichtlich sein.

6.2.6 Benennung der Ausgänge

Die Ausgänge der Workflow Elemente sollen gemäss ihrer Konfiguration benannt werden.

6.2.7 Einfaches hinzufügen weiterer WF Elemente

Hinzufügen eines WF Elemente muss einfach möglich sein (max. 1h Aufwand pro Element)

6.2.8 Finale Umsetzung gemäss Style Guide

Die Workflow Darstellung soll dem Style Guide entsprechen. Die dabei verwendeten Vektorgrafiken werden von einem Designer geliefert.

6.2.9 Einbindung als POC in die existierende Web Applikation

Wenn genügend Zeit vorhanden ist, soll das Ergebnis der Arbeit als Proof of Concept in die existierende Web Applikation des LUCS / TeamManager eingebunden werden.

6.2.10 Benutzerdokumentation

Beschreibung der Schritte für das Laden und Anzeigen eins WF Workflows.

6.2.11 Installations- / Konfigurationsdokumentation

Beschreibung der Installation und Konfiguration der Webapplikation.

6.3 Nichtfunktionale Anforderungen

6.3.1 Aktuellste .NET-Version (Version 4.5.2)

Die Webapplikation soll die aktuellste .NET-Version (Version 4.5.2) unterstützen.

6.3.2 Drittanbieter Software muss kostenfrei verwendet werden

Bei Verwendung von Drittanbieter Software muss darauf geachtet werden, dass sie kostenlos verwendet werden kann. Dabei muss die entsprechende Lizenz, die das ermöglicht, beachtet werden.

6.3.3 Verwendung eines aktuellen Frameworks (MVC, Angular JS)

Für die Entwicklung der Webapplikation soll ein aktuelles Framework wie MVC oder Angular-JS (JavaScript Framework) verwendet werden.

6.3.4 Verwendung von StyleCop

Um sauberen Quellcode zu schreiben soll ein Tool verwendet werden, welches den Quellcode auf Stilkonventionen überprüft und Verstösse gegen diese Richtlinien anzeigt. Damit soll sichergestellt werden, dass der Quellcode sauber geschrieben wird und die Richtlinien eingehalten werden.

6.3.5 Grafiken werden als Vektor geladen

Die von der Luware AG zu Verfügung gestellten Grafiken sind Vektorgrafiken. Für die Darstellung der Workflows auf verschiedenen Geräten wie PC, Tablet oder Smartphone eignen sich Vektorgrafiken besonders gut.

Pixelgrafiken sind ungeeignet, da beispielsweise beim Zoomen und bei der Darstellung auf kleinen Displays die Qualität verloren geht (Grafik wird unscharf). Vektorgrafiken haben beim Skalieren keine Qualitätseinbussen.

6.3.6 Wöchentliches Statusmeeting (Online oder vor Ort)

Es wird jeder Woche ein Statusmeeting online oder vor Ort durchgeführt. An diesem Meeting, welches 15 Minuten dauert, nehmen die Teammitglieder zusammen mit dem Scrum-Master und dem Product Owner teil. Die Teammitglieder erzählen, was sie am letzten Arbeitstag gemacht haben, was sie als nächstes machen werden und welche Probleme es gibt. Das Meeting findet immer zur gleichen Zeit statt.

6.3.7 Kommunikation über Skype / E-Mail

Zur Online-Kommunikation werden Skype und E-Mail genutzt. Beim wöchentlichen Statusmeeting wird Skype eingesetzt. Ausserhalb der Meetings sind die Teammitglieder, der betreuende Dozent und die Mitarbeiter der Firma Luware per E-Mail erreichbar.

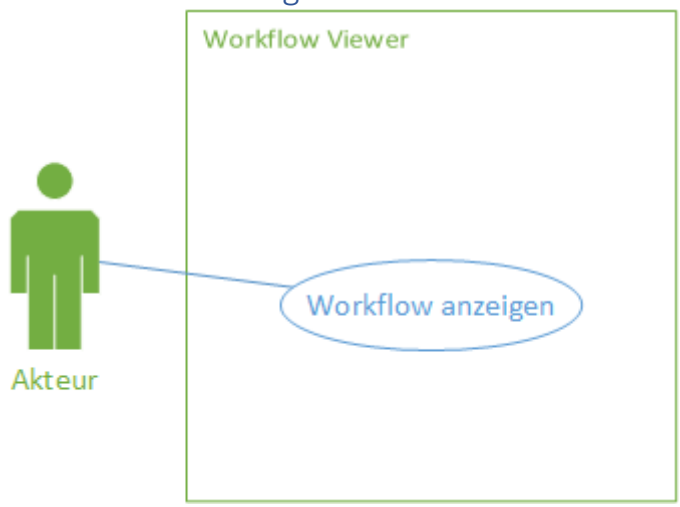
6.3.8 Prozessmodell (Scrum)

Zur Durchführung der Arbeit wird der Scrum-Prozess als Prozessmodell verwendet. Durch den Scrum-Prozess wird erreicht, dass in regelmässigen Abständen ein funktionierendes Produktinkrement abgeliefert wird.

6.4 Use Case

Mit dem Workflow-Viewer soll ein Workflow eingelesen und dargestellt werden. Dieser Use Case beschreibt, wie ein Workflow aus einer XML-Datei im Viewer geladen und dargestellt werden kann. Weil das Anzeigen von Workflows der Hauptteil von diesem Projekt ist, haben wir keine weiteren Use Cases definiert.

6.4.1 Use Case Diagramm



6.4.2 Akteure und Stakeholder

Der Akteur ist ein Benutzer des Contact-Centers von Luware und möchte einen Workflow betrachten. Er lädt einen Workflow im XML-Format hoch oder gibt ihn per Copy & Paste ein.

6.4.3 Beschreibung

UC: Workflow anzeigen

Name	Workflow anzeigen
Umfang	Workflow Viewer
Ebene	Ein Benutzer des Contact-Centers lässt einen Workflow anzeigen.
Primär-Actor	Benutzer des Contact-Centers
Stakeholder und Interessen	Benutzer des Contact-Centers: möchte einen Workflow sehen oder bearbeiten
Vorbedingungen	Es muss ein Workflow im XML-Format existieren.
Nachbedingungen	Der Workflow wird im Viewer angezeigt. Der Benutzer des Contact-Centers kann den Workflow betrachten.
Standard Szenario	Der Benutzer des Contact-Centers klickt auf “Upload Workflow”. Es erscheint ein Dialog, mit dem die XML-Datei des Workflows ausgewählt werden kann. Die ausgewählte XML-Datei eingelesen und validiert. Aus der XML-Datei werden die Elemente des Workflows generiert und dargestellt. Benutzer sieht den angezeigten Workflow.
Erweiterungen	1: Der Benutzer des Contact-Centers fügt mit Copy & Paste den XML-Code in eine Maske des Viewers ein. Die XML-Datei wird validiert. Elemente des Workflows werden generiert und dargestellt. 2: Die XML-Datei enthält Syntaxfehler. 1.Fehlermeldung, dass XML-Datei Syntaxfehler enthält.

	Vorgang wird abgebrochen. Solange die XML-Datei syntaktisch 2: Die XML-Datei entspricht nicht dem XML-Schema eines Workflows 1.Fehlermeldung, dass XML-Datei ein anderes Schema enthält als das vorgesehene Vorgang wird abgebrochen. Solange sich das Schema von den eines Workflows unterscheidet. 2: Das Laden des Workflows dauert länger als 1 Sekunde: 1.Es wird beim Einlesen und Generieren des Workflows der Fortschritt des Ladevorgangs angezeigt. Solange bis der Workflow vollständig eingelesen ist. 2.Benutzer sieht den angezeigten Workflow.
Spezielle Anforderungen	Die Elemente des Workflows werden als Vektorgrafik angezeigt.
Technik und Datenvariationen	Es werden XML-Dateien unterstützt.
Auftreten	mehrmals täglich

7 Technologieanalyse

7.1 Einleitung

In diesem Kapitel untersuchen wir einerseits, welche Technologien von der Luware AG bereits eingesetzt werden und suchen andererseits nach für unser Projekt geeigneten Frameworks und Libraries.

7.2 Server- und Client-Frameworks

7.2.1 Einleitung

Für das Projekt sollen Frameworks gewählt werden, die uns bei der Arbeit unterstützen. Ein Ziel dieser Arbeit ist es, das Projekt in die bestehende Webapplikation von Luware einzubinden. Darum haben wir die Frameworks genauer angeschaut, die sich bereits im Einsatz befinden, um zu schauen, ob sich diese auch für unser Projekt eigenen. Eine Hauptaufgabe dieser Arbeit wird das Darstellen von Workflows sein. Darum prüfen wir auch hier, ob ein entsprechendes Framework vorhanden ist, welches die Vorgaben erfüllt.

7.2.2 Aktuell bei Luware eingesetzte Frameworks

Folgende Frameworks sind im Einsatz:

- ASP.NET MVC
- SignalR
- jQuery
- KendoUI
- AngularJS

ASP.NET MVC

ASP.NET MVC ist ein serverseitiges Framework von .NET. Mit diesem Framework können Webanwendungen entwickelt werden. Es wird das Model-View-Controller Pattern (MVC) angewendet. Das hat Vorteile für komplexe Webanwendungen. Solche Webanwendungen haben dann drei Komponenten:

- Model: Komponente für die Daten (Business Logik). Dies sind normale Klassen, in denen die Business-Logik implementiert ist.
- View: Komponente für die Weboberfläche. Dies ist die HTML-Webseite. Die View bekommt die Daten aus dem Model.
- Controller: Komponente für die Manipulation der Daten und Aktualisierung der Weboberfläche mit veränderten Daten.

MVC-Request-Response-Prozess:

Der Benutzer sendet den Request in Form einer URL. Die URL wird vom UrlRoutingModule geparst. Es wird in die RouteTable geschaut, wer der Controller ist. RoutingData enthält die Daten des Request. Basierend auf den Daten wird die Model Klasse erstellt. Die Daten werden an die View geschickt. Die View erstellt den Markup basierend auf den Daten und auf die Programmlogik und das HTML wird wieder auf dem Browser angezeigt [RajatSingh].

Mit ASP.NET MVC wird auch Test Driven Development unterstützt. Das ASP.NET MVC ist gut geeignet für grosse Entwicklerteams. Die Komponenten lassen sich leicht anpassen und austauschen. ASP.NET MVC steht unter der Apache Licence 2.0. [MSDN1]

SignalR

SignalR ist ein Framework, mit den man Webanwendungen mit Echtzeit Funktionalität entwickeln kann. Dabei werden WebSockets verwendet, um eine bidirektionale Kommunikation zwischen Client (Browser) und Server zu ermöglichen.

SignalR kann zum Beispiel verwendet werden, um Polling zu implementieren. Standardmässig wird Polling mittels Ajax realisiert.

SignalR steht unter der Apache 2.0 Lizenz. [SignalR]

AngularJS

AngularJS ist ein clientseitiges Framework zum Erstellen von Single Page Apps. Das Framework wurde komplett mit JavaScript geschrieben. Mit AngularJS kann zum Beispiel Data Binding gemacht werden, das heisst die Daten aus dem Model werden ans UI gebunden. Das UI wird immer auf den aktuellen Stand gebracht, wenn sich die Daten im Model ändern.

AngularJS arbeitet mit anderen Frameworks, wie jQuery oder KendoUI zusammen.

Die Lizenz von AngularJS ist die MIT-Lizenz. [AngularJS]

KendoUI

KendoUI ist ein kommerzielles Framework für HTML5 und JavaScript und wird von der Firma Telerik angeboten. Es werden verschiedene UI-Widgets (UI-Komponenten) wie Charts, spezielle Buttons oder Texteditoren angeboten. Kendo UI arbeitet mit AngularJS zusammen.

Kendo UI kostet 699 $ pro Entwickler. [Telerik]

jQuery

jQuery ist ein JavaScript-Framework. Es vereinfacht das Umsetzen von Konzepten wie das Manipulieren des DOM-Trees oder das Event-Handling. Mit jQuery kann auf die einzelnen HTML-Elemente zugegriffen werden. [jQuery] Es können auch Animationen gemacht werden und Ajax auf einfache Weise verwendet werden [w3Schools]. jQuery steht unter der MIT-Lizenz.

7.2.3 Fazit

Um eine spätere Integration von unserem Projekt in die existierende Webapplikation von Luware zu vereinfachen, haben wir uns für Frameworks entschieden, die sich bereits im Einsatz befindenden. Konkret verwenden wir von diesen Frameworks ASP.NET MVC für die Strukturierung der Serverseite und AngularJS für die Clientseite. Auch jQuery wird an einigen Stellen benötigt. Die anderen der im vorherigen Abschnitt aufgelisteten Frameworks werden für unser Projekt nicht verwendet.

7.3 JavaScript Flowchart Frameworks

7.3.1 Einleitung

Die Hauptaufgabe der Webapplikation wird es sein, Workflows in Form von Flowcharts in einem Webbrowser darzustellen. Später könnte auch die Aufgabe dazukommen, die Workflows im Browser zu bearbeiten. Darum ist es für uns wichtig, ein flexibles und einfach zu verwendendes clientseitiges Flowchart Framework zur Verfügung zu haben. Ein solches Framework könnte von Grund auf selber erstellt werden[1]. Damit wir uns stärker auf Darstellung und Anordnung der Workflow Elemente konzentrieren können, haben wir uns entschieden, ein existierendes JavaScript Flowchart Framework zu verwenden.

7.3.2 Anforderungen

Primäre Anforderungen:

- Erlaubt das Darstellen von Workflows
- Geeignete Lizenz, die eine Verwendung in kommerziellen Produkten zulässt
- Unterstützung von SVG Grafiken

[1] Beispiel: http://www.codeproject.com/Articles/709340/Implementing-a-Flowchart-with-SVG-and-AngularJS

Sekundäre Anforderungen:

- Unterstützung von Drag & Drop (könnte zu einem späteren Zeitpunkt für das Editieren des Workflow wichtig werden)
- Gute Dokumentation und Beispiele
- Projekt wird aktiv weiterentwickelt

7.3.3 Frameworks

In einem ersten Schritt haben wir nach möglichen einsetzbaren Frameworks gesucht und einige wichtige Eigenschaften zusammengefasst. Das Ergebnis ist in folgender Tabelle ersichtlich:

Name	Url	Lizenz	Letzte Aktivität[2]	Drag & Drop	SVG
Arbor.js	http://arborjs.org/	MIT	5.2012	Ja	Ja
Cytoscape.js	http://js.cytoscape.org/	LGPL3+	2.2015	Ja	Ja
D3.js	http://d3js.org/	BSD	2.2015	Ja	Ja
Draw2D touch	http://www.draw2d.org/	GPL2 / kostenpflichtig	2.2015	Ja	Ja
flowchart.js	http://adrai.github.io/flowchart.js/	MIT	2.2015	Nein	Ja
GoJS	http://gojs.net/	(kostenpflichtig)	3.2015	Ja	Ja
Graph Dracula	http://www.graphdracula.net/	MIT	2.2015	Ja	Ja
Graphviz	http://www.graphviz.org/	EPL	4.2014	Nein	Ja
JavaScript InfoVis Toolkit	http://thejit.org/	MIT	1.2014	Ja	Ja
JointJS	http://www.jointjs.com/	MPL / kostenpflichtig	2.2015	Ja	Ja
Js-graph-it	http://js-graph-it.sourceforge.net/	Apache License V2.0	6.2014	Ja	?
jsPlumb	http://jsplumb.org/	MIT	3.2015	Ja	Ja
Kendo UI	http://www.telerik.com/kendo-ui	kostenpfl.	?	Ja	?
mxGraph	https://www.jgraph.com/	kostenpfl.	3.2015	Ja	Ja
Raphaël	http://raphaeljs.com/	MIT	2.2015	Ja	Ja
WireIt	http://neyric.github.io/wireit/	MIT	7.2014	Ja	Nein
yFiles for HTML	http://www.yworks.com/	kostenpfl.	2.2015	Ja	Ja

[2] Letzte erkennbare Aktivität (Geprüft am 6. März 2015)

7.3.4 Evaluation

Einleitung

Die Auswahl an Frameworks, die sich für das Projekt eignen würden, ist gross. Für eine genauere Evaluation haben wir deshalb versucht, die Auswahl weiter einzuschränken.

Auswahl Einschränkung

Als Mindestvoraussetzungen für die Frameworks legen wir fest, dass sicher Drag & Drop und SVG Unterstützung vorhanden sein muss, das Projekt aktiv weiterentwickelt (mindestens Update im vorherigen Monat) und eine geeignete Lizenz verwendet wird. Dadurch liess sich die Auswahl auf folgende zehn Frameworks einschränken: Cytoscape.js, D3.js ,Draw2D touch, GoJS, Graph Dracula, JointJS, jsPlumb, mxGraph, Raphaël und yFiles for HTML. (Wobei die Lizenz von Cytoscape.js noch genauer geprüft werden muss)

Diese Auswahl haben wir nochmals angeschaut und weiter eingeschränkt. Folgende Frameworks haben zusätzlich noch ausgeschlossen:

- D3.js: Zu wenig spezifisch für die Anzeige von Flowcharts ausgerichtet.
- GoJS: Kostet $2995 pro Entwickler.
- Graph Dracula: Kleine Dokumentation und wenig Beispiele
- JointJS: Kostet CHF1500 pro Entwickler. Mehr ausgerichtet auf das Erstellen von Graphen mit einem Online Editor als auf die Anzeige.
- mxGraph: Kostet $5000 pro Entwickler. Keine Möglichkeit für das Testen vor dem Kauf gefunden.
- Raphaël: Zu wenig spezifisch für die Anzeige von Flowcharts ausgerichtet.
- yFiles for HTML: Das Framework ist kostenpflichtig, den Preis erfährt man erst nach einer Offertenanfrage.

Übrig bleiben noch die Frameworks Cytoscape.js, Draw2D touch und jsPlumb, die auf den ersten Blick vielversprechend wirken. Im nächsten Schritt werden wir diese noch genauer prüfen und vergleichen.

Cytoscape.js

Allgemein:

- Bibliothek für das Analysieren und Darstellen von Graphen
- Keine Abhängigkeiten von anderen JavaScript Frameworks
- Unterstützung für mobile Geräte
- Grösse: 250 KB
- Erstellung des Graphen komplett in JavaScript anhand JSON
- Unterstützt automatische und manuelle Positionierung
- Funktionen zum Erstellen von Animationen

Dokumentation:

- Ausführliches Handbuch
- Herunterladbare Beispiele

Lizenz:

- GNU Lesser General Public License Version 3 (LGPL3)
- Nutzungsbedingungen werden noch genauer geprüft

Draw2D touch

Allgemein:

- Bibliothek für das Zeichnen von Diagrammen
- Abhängig von den Libraries jQuery, Raphael.js, Class.js, JSON und Connection routing

- Unterstützung für mobile Geräte
- Grösse: 1507 KB (+ weitere benötigte Imports: ca. 500 KB)
- Aufbau eines Diagramms durch Erstellen von JavaScript Objekten (z.B. canvas.add(new draw2d.shape.basic.Rectangle(),x,y))
- Manuelle Positionierung

Dokumentation:

- Ausführliches Handbuch und API Dokumentation
- Viele Beispiele mit Quellcode

Lizenz:

- Royalty Free
- Kosten: 699 Euro
- Kommerzielle Nutzung erlaubt
- Darf für unlimitierte Projekte von einer unlimitierten Anzahl Entwickler verwendet werden (jedoch an Firma gebunden)
- Updates während eines Jahres
- Premium Support während 3 Monaten

jsPlumb

Allgemein:

- Bibliothek für das visuelle Verbinden von Elementen auf Webseiten
- Keine Abhängigkeiten von anderen JavaScript Frameworks
- Unterstützung für mobile Geräte
- Grösse: 178 KB
- Elemente werden in HTML erstellt (Divs)
- Viele Gestaltungsmöglichkeiten (CSS)
- Manuelle Positionierung

Dokumentation:

- Ausführliches Handbuch und API Dokumentation
- Einige Beispiele mit Quellcode

Lizenz:

- MIT License und GNU General Public License Version 2 (GPLv2)
- Kommerzielle Nutzung erlaubt
- Framework muss wieder denselben Lizenztext enthalten
- Änderungen am Framework sind erlaubt

7.3.5 Ergebnis der Evaluation

Framework Auswahl

Beim Vergleichen der Frameworks sind wir zu folgenden Ergebnissen gekommen:

- Draw2D touch wirkte sehr überladen, es hat viele Abhängigkeiten zu anderen JavaScript Dateien und ist dadurch etwas 2MB gross. Weil das Framework kostenpflichtig ist und wir viele der angebotenen Funktionen für unser Projekt gar nicht benötigen, haben wir uns gegen Draw2D touch entschieden.
- jsPlumb unseren Anforderungen sehr nahe. Ein entscheidender Nachteil gegenüber Cytoscape.js ist jedoch, dass die Nodes manuell angeordnet werden müssen, während dem Cytoscape.js sogenannte Layouts zur automatischen Anordnung des Graphen bietet. Darum haben wir uns schlussendlich für Cytoscape.js als zu verwendendes Framework entschieden.

- Cytoscape.js bietet einen grossen Funktionsumfang, besonders überzeugen konnte die Möglichkeit für das Erzeugen der Graphen aus einem JSON Dokument, sowie die Funktion für das automatische Anordnen der Nodes. Zudem können die Graphen gezoomt werden, was bei grossen Workflows nützlich ist und ein scrollen überflüssig macht. Das Framework hat keine Abhängigkeiten auf andere Frameworks und ist mit 250KB relativ klein. Es ist eine ausführliche Dokumentation mit vielen Beispielen vorhanden.
  Ein Nachteil von dem Framework ist, dass es beim Styling des Graphen gewisse Einschränkungen gibt, weil nur ein Subset von CSS unterstützt wird. Diese Einschränkungen lassen sich jedoch teilweise umgehen, indem der Hintergrund der Nodes transparent gemacht wird und darüber eine Pixel- oder Vektorgrafik gelegt wird.
  Bezüglich der LGPLv3 Lizenz gibt es noch Unsicherheiten, was die Verwendung im Projekt für Konsequenzen hat. Im folgenden Abschnitt wird versucht, die genauen Bedingungen der Lizenz klären.

LGPLv3 Lizenz

Der Lizenztext von GNU Lesser General Public License v3.0 kann hier gefunden werden:

http://opensource.org/licenses/lgpl-3.0.html

Die folgenden Seiten beinhalten eine einfache Zusammenfassung der Lizenzen:

https://tldrlegal.com/license/gnu-lesser-general-public-license-v3-(lgpl-3) und
http://choosealicense.com/licenses/lgpl-3.0/

Laut diesen Zusammenfassungsseiten ist eine kommerzielle Nutzung ausdrücklich erlaubt, jedoch muss der Lizenztext beigelegt werden. Auch Änderungen am Framework sind möglich, solange diese dokumentiert werden.

Was noch für Unsicherheit sorgt ist der Punkt “Disclose Source”. Bei choosealicense.com steht dazu folgendes:

“Source code must be made available when distributing the software. In the case of LGPL, the source for the library (and not the entire program) must be made available.”

Die Frage, die sich uns hier stellt, ist, welche Teil des Quellcodes genau wieder verfügbar gemacht werden muss. Ist das nur von der verwendeten Library selber? Und ich welcher Form muss der Quellcode verfügbar gemacht werden?

Folgende Seite erwähnt einen weiteren Aspekt:

http://www.it-rechts-praxis.de/meldungen/Open-Source-Kommerzielle-Nutzung-von-LGPL-Libraries-12

Demzufolge ist eine kommerzielle Verwendung einer Library dann problemlos, wenn diese Library mit dem eigenen Projekt “dynamisch” und nicht “statisch” verlinkt wird. Dynamisch verlinkt heisst, dass es möglich sein muss, die betroffenen Library auszutauschen, ohne dass das komplette Produkt neu kompiliert werden muss. Ist diese Forderung nach dynamischer Verlinkung im Falle von JavaScript automatisch gegeben?

Folgende Antwort auf stackoverflow geht auf diese Frage ein:

http://stackoverflow.com/a/2015840

Was wir aus dieser Antwort lesen ist, dass die Library auf jeden Fall in einer separaten Skript Datei bleiben soll. Zudem muss ein Benutzer die Möglichkeit haben, auf den Source Code in unminified Form zugreifen zu können.

Auch wenn in gewissen Details der Lizenz noch Unsicherheiten bestehen, sind wir zum Schluss gekommen, dass das Framework in unserem Projekt verwendet werden kann, ohne das der komplette Source Code zugänglich gemacht werden muss. Dazu haben wir folgende Massnahmen getroffen:

- Beilegen des LGPLv3 Lizenztextes
- Beilegen des Cytoscape.js Frameworks in unminified Form
- Erwähnen der GitHub-Seite und Lizenz von Cytoscape.js im HTML Quelltext

Fazit

Wir haben uns für Cytoscape.js entschieden, weil es sich gut zur Darstellung der Workflows eignet. Die Vorgabe für die Unterstützung von SVG Grafiken ist erfüllt. Ein Cytoscape.js Graph wird im JSON Format beschrieben, dadurch können die vom Server aus dem XAML erzeugten JSON Daten ohne weitere Verarbeitung direkt angezeigt werden.

Cytoscape.js unterstützt verschiedene Layouts, die die Anordnung der Elemente steuern. Mit dieser Funktion werden die Workflows automatisch in einer übersichtlichen Art dargestellt. Es besteht auch die Möglichkeit Elemente zu verschieben. Damit kann der Benutzer die Elemente eines Workflows durch Verschieben manuell nach seinen Vorstellungen anordnen. Es kann auch in den Workflow hinein- und herausgezoomt werden, was einen guten Überblick über grosse Workflows gibt.

7.4 Weitere Frameworks und Libraries

7.4.1 Einleitung

Neben den bereits aufgezählten Frameworks verwenden wir einige weitere Erweiterungen und Bibliotheken, die sich im Verlauf der Entwicklung als nützlich herausgestellt haben. Diese sind im Folgenden nach Kategorie aufgelistet und kurz beschrieben.

7.4.2 .net Framework

Json.NET:

Die anzuzeigenden Workflows werden auf dem Server umgewandelt und als JSON auf den Client übertragen. Generiert wird das JSON Dokument durch die Serialisierung einer der Zielstruktur entsprechenden Klasse. Das .net Framework bietet mit der JavaScriptSerializer Klasse (Namespace System.Web.Script.Serialization) bereits eine Möglichkeit für die Erzeugung von JSON an. Wir haben uns dennoch für den Einsatz von Json.NET entschieden, weil die Ausführungsgeschwindigkeit um ein vielfaches höher ist und mehr Funktionen vorhanden sind. [JsonNET]

Angeboten wird dieses JSON Framework unter der MIT-Lizenz, welche sich für den Einsatz in kommerziellen Produkten eignet. [TLDRLegal1]

7.4.3 AngularJS

ng-file-upload:

Diese AngularJS Direktive vereinfacht den Datei-Upload vom Client per Ajax. Eine AngularJS Direktive ist ein benutzerdefiniertes HTML Element bzw. in diesem Fall ein HTML Attribut. Durch das Hinzufügen von dem Attribut “ngf-select” zu einem Div, wird bei einem Klick darauf ein Dateiauswahl-Dialog angezeigt. Im Gegensatz zum normalen HTML Dateiauswahl Element kann dieses Div völlig frei gestaltet werden und der Upload startet sofort nach der Dateiauswahl. Die Übertragung erfolgt im Hintergrund per Ajax.

Diese Erweiterung steht unter der MIT-Lizenz.

AngularUI Router:

AngularUI Router ist ein Routing Framework für AngularJS. Diese Erweiterung ermöglicht es, zwischen Seiten zu wechseln, ohne dass die komplette Seite neu geladen werden muss. AngularJS bietet mit ngRoute bereits eine ähnliche Funktion an, jedoch unterscheidet sich AngularUI Router in einigen Bereichen. Während ngRoute auf URLs basiert, arbeitet AngularUI Router mit Zuständen. Im Grundtemplate können mehrere sogenannte Views festgelegt werden, welche dann je nach Zustand dynamisch mit anderem Inhalt gefüllt werden. Die Hyperlinks im HTML enthalten dementsprechend keine URL, sondern den Zustand, zu dem gewechselt werden soll. Die einzelnen Seiten sind dadurch weniger stark an eine direkte URL gekoppelt.

AngularUI Router verwendet die MIT-Lizenz.

7.4.4 Cytoscape.js

cytoscape-panzoom:

Dieses Plug-in für Cytoscape.js fügt ein Bedienungselement hinzu, mit dem in einen Graph hinein- und herausgezoomt werden kann. Das Aussehen haben wir mit einem externen CSS unserem Design angepasst, sowie das Element für das Verschieben des gesamten Graphs ausgeblendet, weil das auch mit der Maus per Drag and Drop möglich ist. Auf Mobilgeräten wird das Bedienelement automatisch ausgeblendet, weil da die Bedienung mit Touch-Gesten intuitiver ist.

Das Plug-in verwendet dieselbe Lizenz LGPLv3 wie das Cytoscape.js Framework.

dagre:

Die Position der Nodes in einem Cytoscape.js Graph wird mithilfe sogenannter Layouts gesteuert, einige davon werden bereits standardmässig im Framework angeboten. dagre ist ein von einem externen Entwickler erstelltes Layout, welches sich besonders für die übersichtliche Darstellung von Bäumen und gerichteten azyklischen Graphen eignet. Über konfigurierbare Parameter können unter anderem die Abstände der Nodes und die Ausbreitungsrichtung des Graphs gesteuert werden.

dagre ist unter der MIT-Lizenz lizenziert.

7.4.5 JavaScript Frameworks

interact.js:

Mit interact.js können HTML Elemente wie divs verschiebbar und in der Grösse änderbar gemacht werden. In unserer Applikation verwenden wir das, damit der Property-Dialog in der Breite geändert werden kann.

Verwendet wird die MIT-Lizenz.

jQuery:

Mit der JavaScript Bibliothek jQuery können Elemente im DOM selektiert und manipuliert werden. Die Bibliothek stellt dabei die Kompatibilität mit älteren Webbrowsern sicher. jQuery wird für das Plug-in cytoscape-panzoom und an einigen Stellen der eigenen Skripts benötigt.

Das Framework wird unter der MIT-Lizenz verbreitet.

7.4.6 Icon Libraries

flag-icon-css:

Die Icon Bibliothek flag-icon-css enthält SVG Icons von Nationalflaggen, welche durch Einbinden von einem CSS und Zuweisen von entsprechenden Klassen an einem span-Element auf einer HTML Seite angezeigt werden können. Die Flaggen sind sowohl in quadratischem als auch 4:3 Format verfügbar. Wir verwenden diese Icons um die jeweilige Sprache von Multilanguage Strings zu visualisieren.

Die Icons stehen unter der MIT-Lizenz.

Font Awesome:

Font Awesome enthält skalierbare Vektoricons, deren Darstellung sich mithilfe CSS steuern lässt. In unserem Projekt verwenden wir unter anderem Icons für das cytoscape-panzoom Plug-in (Zoom In/ Zom Out/ Zoom Reset). Zudem ist ein animierter Waiting Spinner vorhanden, der im XAML Upload Button zum Einsatz kommt.

Für die CSS Dateien kommt die MIT-Lizenz zum Einsatz. Für den Font wird die SIL Open Font License verwendet. Diese Lizenz erlaubt die Verwendung des Fonts in einem kommerziellen Produkt. [TLDRLegal2]

7.5 Windows Workflow Foundation

Windows Workflow Foundation ist eine Bibliothek für .NET um komplexe oder einfache Arbeitsabläufe (Workflows) zu entwickeln.

Diese Bibliothek enthält Klassen zum Erstellen der einzelnen Elemente eines Workflows wie Activites und den Verbindungen dazwischen.

Ein Workflow ist eine Menge von Elementen, die Activities genannt werden und einen realen Prozess der Welt abbilden.

Jeder laufende Workflow wird von der Workflow Runtime Engine verwaltet. Vgl. [MSDN2]

7.5.1 Konzepte

Activities

Jede Aktion eines Workflows ist als Activity modelliert.

Activities werden oft als eine Collection von anderen Activities realisiert. Die Activity-Klasse ist eine mit XAML definierte Basisklasse. Activities können eine einfache Anweisung oder eine Menge von anderen Activities sein.

Activities können auch mit Hilfe der CodeActivity Klasse realisiert werden um auf Daten zuzugreifen.

Die NativeActivity Klasse stellt die ganze Laufzeit des Workflows zur Verfügung.

Activities werden mit CLR-kompatiblen Programmiersprache, wie C# erstellt.

Activity Data Model

- Variable: speichert Daten in eine Activity
- Argument: Daten werden in eine Activity verschoben.
- Expression: Activity mit einem Rückgabewert (in Argumentbindungen)

Workflow Laufzeitmodul

Workflows laufen in einer Umgebung, die Laufzeitmodul genannt wird. Der Host verwendet folgende Elemente:

- WorkflowInvoker: Der Workflow-Invoker ruft einen Workflow wie eine Methode aus. Es werden auch Eingabedaten bereitgestellt, Ausgabedaten eines Workflows abgerufen oder Erweiterungen für Activites hinzugefügt.
- WorkflowApplication: Die WorkflowApplication führt eine Single Workflow Instanz aus.
- WorkflowServiceHost: Mit einem WorkflowServiceHost können nachrichtenbasierte Interaktionen zwischen mehreren Instanzen gemacht werden.

- ActivityInstance: Der Host kann über die ActivityInstance mit dem Laufzeitmodul interagieren. Es kann zum Beispiel die Ausführung des Workflows gesteuert werden.

Workflows verwenden eine ActivityContext Klasse, um Zugriff auf die Workflow-Laufzeitumgebung zu erhalten. NativeActivity verwendet die NativeActivityContext Klasse, die von der ActivityContext Klasse abgeleitet ist. Die CodeActivityContext Klasse ist ebenfalls von ActivityContext abgeleitet und wird von der CodeActivity verwendet.

Diese Klasse werden zum Auflösen von Variablen und Argumenten, zur Planung von untergeordneten Aktivitäten und zu anderen Zwecken genutzt.

7.5.2 Archiktektur von Windows Workflows

Architektur von Activites

Die Activities werden als CLR-Typen realisiert. Diese Typen werden von Activity, CodeActivity, AsyncCodeActivitiy oder NativeActivity abgeleitet. Es gibt auch entsprechende Typen, die Werte zurückgeben können: Activity<TResult>, CodeActivity<TResult>, AsyncCodeActivity<TResult>, NativeActivity<Tresult>.

AsyncCodeActivity kann für asynchrone Aufgaben verwendet werden. Mit NativeActivity kann auf die Laufzeit zugegriffen werden.

Activities können deklarativ in XAML erstellt werden.

Lebenszyklus einer Activity

Eine Instanz beginnt im **executing** Zustand. Das heisst, dass die Activity ausgeführt wird.

Wenn die Ausführung der Activity beendet wurde und alle Arbeitsvorgänge abgeschlossen wurden dann befindet sich die Instanz im **closed** Zustand.

Wenn Arbeitsvorgänge ordnungsgemäss abgebrochen wurden und die Aktivität beendet wurde, dann befindet sich die Instanz im **canceled** Zustand.

Die Instanz befindet sich im **faulted** Zustand, wenn ein Fehler (Ausnahme) aufgetreten ist. Die Aktivität wurde beendet ohne die Arbeiten abzuschliessen. Die Ausnahme wird an den übergeordneten Activities weitergegeben.

7.6 WF XAML Struktur

7.6.1 Einleitung

WF Activities können sowohl mit XML als auch mit Programmcode definiert werden. Für das Erstellen und Bearbeiten von XML Code stellt Visual Studio einen grafischen Designer bereit, womit Workflows per Drag & Drop aufgebaut werden können. Luware setzt auf die XML Variante, weil sich dadurch Workflows durch den Endkunden ihres Produktes im Editor erstellen und bearbeiten lassen. Da die Workflows von uns eingelesen werden müssen, wird im Folgenden der Aufbau der XMLs genauer untersucht.

7.6.2 Allgemeine Struktur

XAML

Die Workflows werden in einem XAML Dokument definiert. XAML (Extensible Application Markup Language) ist eine von Microsoft entwickelte Beschreibungssprache, welche auf XML basiert. Jedes XAML Dokument ist auch ein gültiges XML Dokument, umgekehrt gilt das aber nicht.

Grundstruktur

Genau wie XML besteht ein XAML Dokument in erster Linie aus Tags (<Tag-Name>Inhalt</Tag-Name>) und Attributen (<Tag-name Attribut-Name="Wert" />). Der logische Aufbau entspricht dabei einer

Baumstruktur. Eine XML-Deklaration (<?xml ... ?>) wird vom Visual Studio Designer nicht erstellt. Eine WF Activity wird mit folgenden Tags umschlossen:

<Activity></Activity>

Namespaces

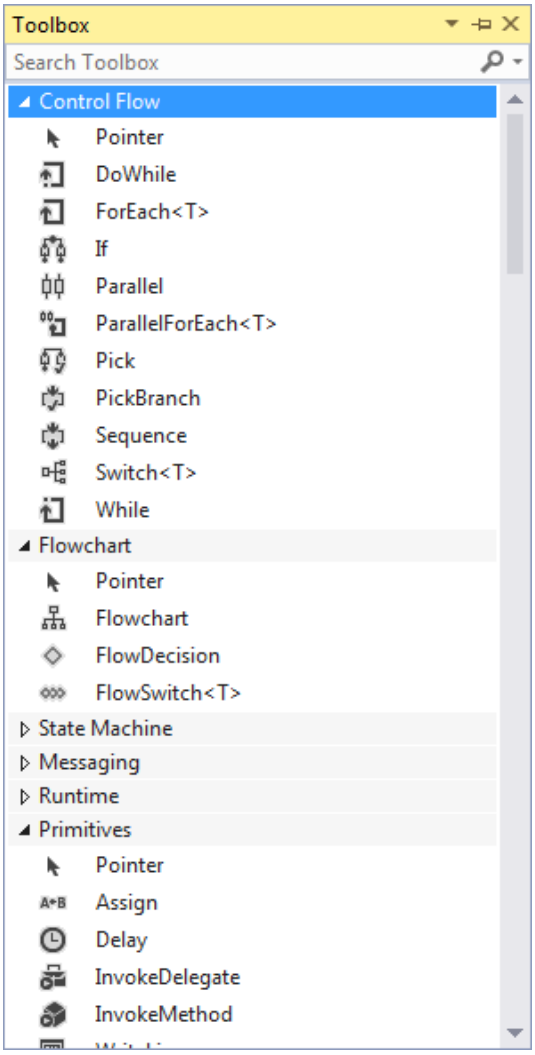
Namensräume bzw. Namespaces erweitern die Benutzbaren Elemente im XAML. Einem Namespace wird ein Präfix angehängt, wodurch Namenskonflikte verhindert werden. Es sollte höchstens ein Namespace ohne Präfix definiert werden, dieser wird dann zum Standard-Namespace. Namensräume werden im äussersten Tag als xmlns-Attribut in der folgenden Form definiert:

<Activity xmlns:Präfix="Schema-URL oder CLR-Namespace"></Activity>

In einer leeren WF Activity fügt Visual bereits einige Namespaces hinzu, beispielsweise:

- xmlns="http://schemas.microsoft.com/netfx/2009/xaml/activities"
- xmlns:sapx="clr-namespace:System.Activities.Presentation.Xaml;assembly=System.Activities.Presentation"

7.6.3 Workflow Elemente



Einleitung

Visual Studio stellt einige Activities bereit, die in den eigenen Workflows benutzt und kombiniert werden können. Gewisse dieser Activities steuern den Kontrollfluss (z.B. Schleifen/If/Switch), andere führen Aufgaben aus (z.B. Ausgabe einer Textzeile in der Konsole). Im Folgenden werden einige Activities vorgestellt.

Primitiven

- **Assign**: Weist einen Wert einer Variable oder einem Argument zu
- **Delay**: Verzögert die Ausführung eines Workflows um die angegebene Zeit
- **InvokeDelegate**: Ruft einen öffentlichen Delegate auf
- **InvokeMethod**: Ruft eine öffentliche Methode vom angegebenen Objekt oder Typ auf
- **WriteLine**: Gibt eine Textzeile in der Konsole oder einem anderen TextWriter Objekt aus

Kontrollfluss

- Schleifen: **DoWhile**, **ForEach**, **ParallelForEach**, **While**
- Entscheidung: **If**, **Switch**, **Pick**, **PickBranch**
- **Parallel**: Führt mehrere Child Activities prarallel auf
- **Sequence**: Liste von Activities, die sequentiell abgearbeitet werden

Flowchart

Mit Flowcharts können nicht-sequentielle Workflows implementiert werden. Im Grunde genommen ist ein Flowchart auch eine Activity (**FlowChart**), welche in einem vorhandenen Workflow platziert werden kann. Jeder Flowchart hat einen definierten Startpunkt. Die Ausführung ist zu Ende, wenn ein Element im Flowchart erreicht wird, welches keine fortführenden Elemente mehr hat. Folgende Elemente können in einem Flowchart platziert werden:

- **FlowDesicion**: Verzweigung basierend auf einer Boolean Bedingung
- **FlowSwitch**: Verzweigung basierend auf mehreren Vergleichskriterien

Ein weiteres Element im Flowchart ist der **FlowStep**, welches einen einzelnen Ausführungsschritt darstellt und aus einer Activity besteht. Die FlowSteps werden durch den Visual Studio Designer automatisch erstellt, wenn eine Activity platziert wird.

Grundstruktur:

```
<Flowchart>
  <Flowchart.StartNode>
    <x:Reference>...</x:Reference>
  </Flowchart.StartNode>
  <FlowStep x:Name="...">
    ...
  </FlowStep>
</Flowchart>
```

FlowDesicion:

```
<FlowDecision Condition="...">
  <FlowDecision.True>
    ...
  </FlowDecision.True>
  <FlowDecision.False>
    ...
 </FlowDecision.False>
</FlowDecision>
```

FlowSwitch:

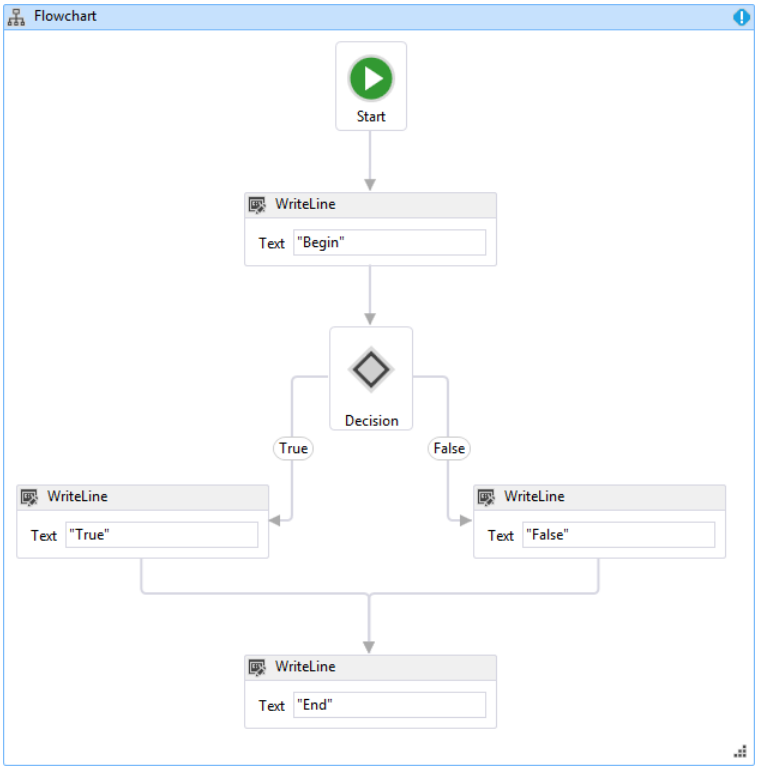
```
<FlowSwitch x:TypeArguments="x:Int32">
  <FlowSwitch.Default>
    <FlowStep>
      ...
    </FlowStep>
  </FlowSwitch.Default>
  <FlowStep x:Key="0">
```

```
    ...
  </FlowStep>
  <FlowStep x:Key="1">
    ...
  </FlowStep>
</FlowSwitch>
```

Sequenz:

```
<FlowStep>
  (Activity1...)
  <FlowStep.Next>
    <FlowStep>
      (Activity2...)
      <FlowStep.Next>
        <FlowStep>
          (Activity3...)
        </FlowStep>
      </FlowStep.Next>
    </FlowStep>
  </FlowStep.Next>
</FlowStep>
```

Beispiel Flowchart:



Attribute

Folgendes sind allgemein verwendete Attribute:

- x:Name - Gibt einer Activity einen Namen
- x:Reference - Verweist auf ein Element anhand seines Namens
- DisplayName - Anzeigename einer Activity

7.6.4 Luware spezifische Struktur

Einleitung

Luware verwendet zu einem grossen Teil selbst definierte Activities. Die äusserste Activity von einem Workflow ist jeweils ein Flowchart, welcher den eigentlichen Workflow enthält. Dieser innere Workflow beginnt oft mit der Activity AcceptCall und endet mit DisconnectCall oder VirtualTransfer.

Unsere Webapplikation soll nach Möglichkeit auch neu hinzukommende Activities unterstützen, mindestens aber die im folgenden Abschnitt aufgezählten.

Verwendete Activities

Allgmein:

- DynamicActivity
- Flowchart
- FlowStep
- FlowSwitch
- Literal
- Sequence
- Switch
- VariableReference
- VariableValue

Luware:

- AcceptCall
- AnnouncementActivity
- AskForAgentActivityLogic
- BlindTransfer
- CheckModalityActivity
- CheckServicePresenceActivityLogic
- CheckWorkingHoursActivityLogic
- ConnectActivityLogic
- ConversationRecordingActivity
- DbBaseRoutingActivityLogic
- DeclineCall
- DisconnectCall
- EmergencyActivity
- Forward
- InputCustomerActivityLogic
- IpOriginRoutingActivityLogic
- LanguageDependentTransfer
- OriginRoutingActivityLogic
- PreferredAgentRoutingActivity
- SessionHandledActivity
- StandbyDutyActivity
- TransferorPreferredAgentRoutingActivity
- VirtualTransfer

Daneben gibt es eine Reihe verschachtelter Activities, die zwar nicht direkt angezeigt werden, aber deren Properties teilweise ausgelesen werden sollen:

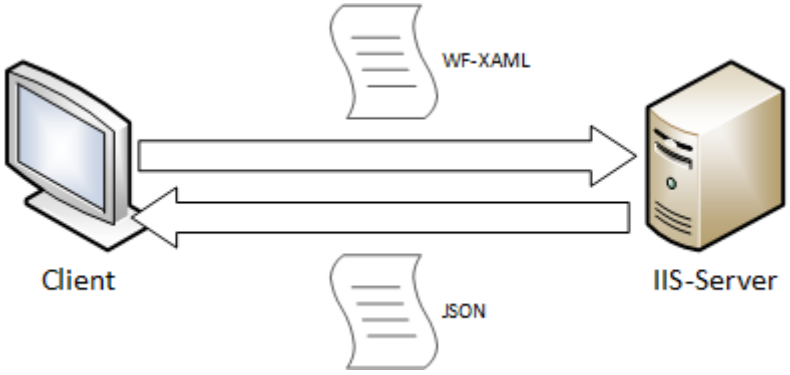
- AddTraitActivity
- InstantMessagingQuestionAnswer
- InstantMessagingStatement
- SpeechQuestionAnswer
- SpeechEmergencyStatement
- SpeechStatement

8 Architektur

8.1 Systemübersicht

8.1.1 Einleitung

Der WF Webviewer ist eine Webapplikation und ist damit in einen Server und Client Tier aufgeteilt. In einem ersten Schritt wird ein WF-XAML vom Client zum IIS-Server übertragen. Dort wird der XAML Code eingelesen, verarbeitet und in Form von JSON an den Client zurückgesendet. Die JSON Daten dienen als Grundlage für die grafische Darstellung des Workflows mithilfe eines passenden JavaScript Frameworks.



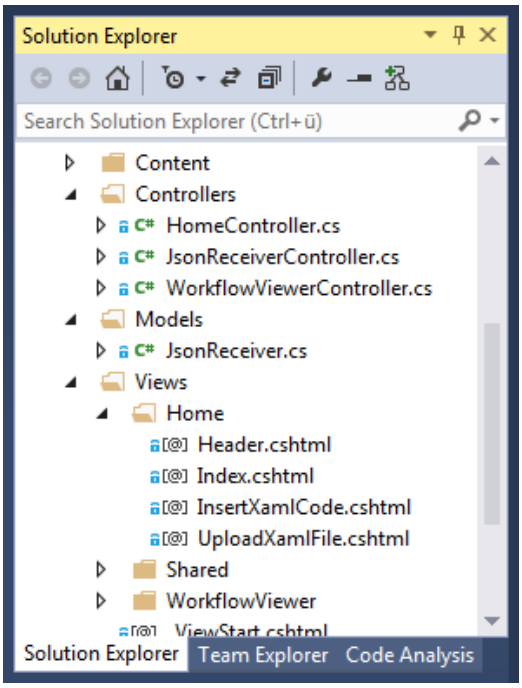
8.2 Serverarchitektur

8.2.1 Einleitung

Der Server Tier wird in ASP.NET erstellt und auf einem IIS Server ausgeführt. Das verwendete ASP.NET MVC Framework gibt, wie der Name sagt, eine Model View Controller Struktur vor. Die Ordnerstruktur in einem ASP.NET Projekt ist dementsprechend aufgebaut.

8.2.2 Model View Controller

Das MVC Muster teilt die Applikation in die Einheiten Model, View und Controller auf.



Model

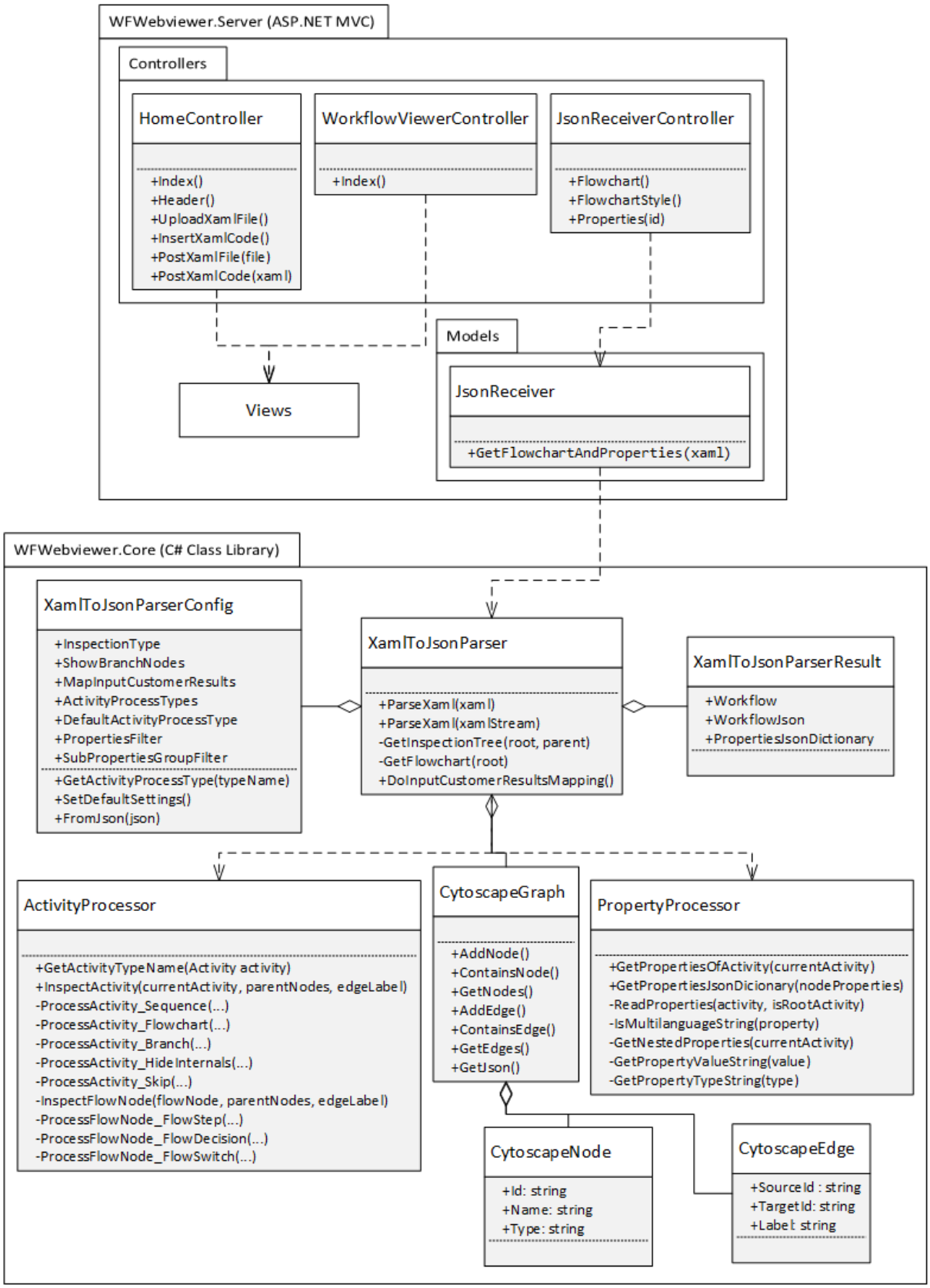
Kümmert sich um die Datenhaltung.

View

Generell stellt die View Daten aus dem Model dar und nimmt Benutzereingaben entgegen. In ASP.NET handelt es sich dabei um sogenannte Templates, die HTML sowie Elemente der Razor Templatesprache enthalten.

Controller

ASP.NET bildet URLs auf Methoden in den Controller Klassen ab. In den Methoden werden etwaige Parameter entgegengenommen und verarbeitet. Die Rückgabe kann ein String oder das Template mit dem gleichen Namen wie die Methode sein.



8.2.3 Projektaufteilung

Das ASP.NET Projekt “WFWebviewer.Server” ist die Schnittstelle zum Client. Die Hauptaufgabe des Servers ist das Einlesen von WF XAML und die anschliessende Umwandlung in JSON. Diese Funktion wird in das separate C# Klassenbibliothek Projekt “WfWebviewer.Core” ausgelagert, um die Wiederverwendbarkeit zu erleichtern. Im Projekt “WfWebviewer.Core.Testing” befinden sich Unit Tests für den XAML zu Json Konverter.

8.2.4 WFWebviewer.Server Klassen

Das ASP.NET Projekt umfasst die Controller HomeController, JsonReceiverController und WorkflowViewerController, sowie das Model JsonReceiver.

HomeController

Die HomeController Klasse liefert mit den Methoden Index und Header das Grundgerüst der HTML Seiten. Die Seiten für das Heraufladen und Einfügen von XAML Code werden als PartialView geliefert. In der Methode PostXamlFile wird das XAML aus dem File gelesen und einem Session-Objekt übergeben. Das File wird in Form eines HttpPostedFileBase Objekts als Parameter übergeben. Darauf wird der XAML Code mithilfe des XamlToJsonParsers umgewandelt. Das resultierende JSON und die Activity Properties werden auch im Session-Objekt gespeichert.

Die Methode PostXamlCode erhält den XAML Code als String, die Verarbeitung danach ist analog wie im PostXamlFile.

JsonReceiverController

Die Klasse JsonReceiverController enthält die Methoden um einen FlowChart als JSON zurückzugeben. In diesen Methoden wird das XAML aus dem Session-Objekt ausgelesen und als Content zurückgegeben. Weiterhin enthält die Klasse JsonReceiverController eine Methode FlowChartStyle(), die das CSS eines Workflows zurückgibt.

Es werden auch die Properties an Hand der id zurückgegeben.

WorkflowViewerController

Liefert die Seite mit der Workflow Anzeige als PartialView.

JsonReceiver

In JsonReceiver werden der FlowChart und die Properties aus den XAMLs gelesen und als XamlToJsonParserResult geliefert. Dies geschieht durch die Methode GetFlowchartAndProperties, die das XAML als String erwartet. Es wird auch die Konfiguration des Parsers aus dem JSON ausgelesen und dem Parser übergeben. Anschliessend wird das XAML geparst.

8.2.5 WFWebviewer.Core Klassen

Der XamlToJsonParser besteht unter anderem aus folgenden Klassen:

XamlToJsonParser

Methoden für das Umwandeln von WF XAML nach Cytoscape.js JSON und für das Auslesen der Activity Konfigurationen.

ActivityProcessor

Wandelt die durch WorkflowInspectionServices ausgegebenen Workflow Struktur in eine übersichtlichere Form um.

PropertyProcessor

Methoden für des Auslesen der Activity Konfiguration (Properties)

CytoscapeGraph

Speichert Knoten und Kanten und bietet eine Funktion, um den Graph im Cytoscape JSON Format auszugeben.

CytoscapeNode

Stellt einen Knoten dar, der dem CytoscapeGraph hinzugefügt werden kann.

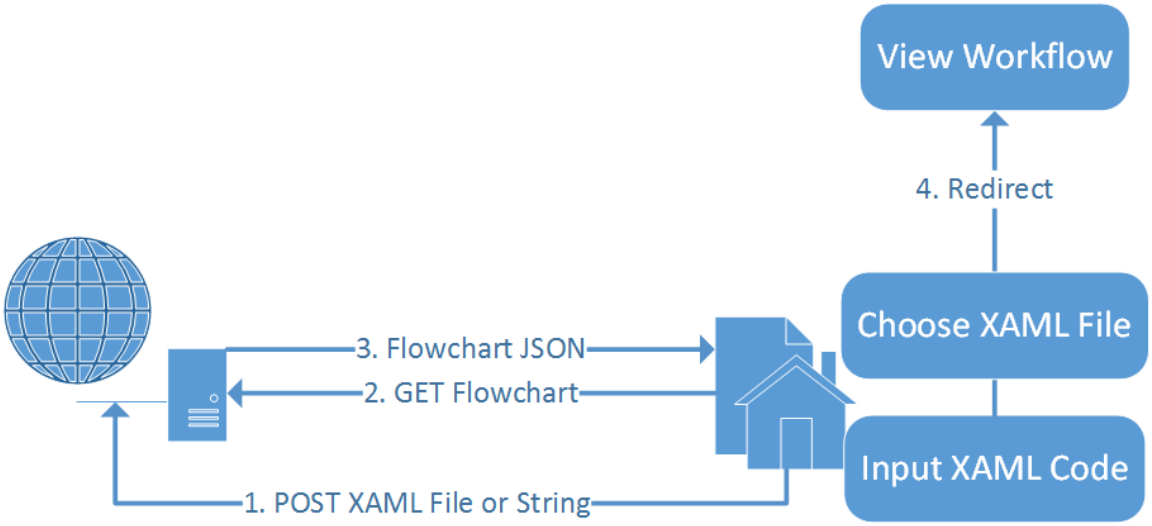
CytoscapeEdge

Stellt eine Kante dar, die dem CytoscapeGraph hinzugefügt werden kann.

8.3 Client Architektur

8.3.1 Einleitung

Der Client Tier wird mit HTML, CSS, JavaScript und AngularJS als Single-page-Webanwendung realisiert. AngularJS verwendet dabei ein Model-View-Viewmodel Pattern (MVVM). Das Model enthält die Business-Logik. In der View befinden sich die Webseiten als HTML und die CSS-Dateien. Angular JS verwendet Two-Way-Binding um Daten zwischen der View und dem ViewModel aktuell zu halten.



8.3.2 Client Aufgaben

Der Client empfängt den Workflow in Form eines JSON Dokumentes und stellt es mithilfe des Frameworks Cytoscape.js dar.

Die Logik vom Client ist in app.js im Ordner Content/wfwebviewer des Projekts WFWebviewer.Server implementiert. Dort befinden sich unter anderem die AngularJS-Controller für den Upload eines XAML-Files, Posten von XAML-Code und für die Darstellung eines Workflows mittels Cytoscape.js. Für die Darstellung wird der JsonReceiver mittels Ajax aufgerufen. Innerhalb des Aufrufs wird das Stylesheet empfangen und der Workflow mit Cytoscape.js dargestellt.

Dabei wird die Funktion cytoscape() an einer Variable $scope.cy zugewiesen. Als Parameter werden die Cytoscape-Elemente, wie Container, elements, style oder layout in einem JSON mitgegeben. Die Funktion cytoscape() instanziiert einen Graphen.

Der Style für die Workflowdarstellung befindet sich in Content/workflows/styles und wird vom JsonReceiverController geliefert und ist mit einer CSS ähnlichen Sprache aufgebaut.

8.4 Server-/Client-Kommunikation

8.4.1 Einleitung

Im ersten Schritt übermittelt der Client den XAML Code von einem Workflow an den Server. Das geschieht entweder per Dateiupload oder direkt durch einen POST von dem XAML Code. Danach kann der Client eine URL aufrufen, über die der Server die Ausgabe als JSON zurücksendet.

8.4.2 JSON Struktur

Das verwendete Framework Cytoscape.js erlaubt das Erstellen eines Graphen anhand Daten im JSON Format. Darum werden die JSON Daten vom Server bereits in einer passenden Form gesendet. Folgendes Beispiel stellt einen einfachen Graph mit drei Knoten und zwei Kanten dar:

```
{
 "elements": {
  "nodes": [
   { "data": { "id": "a" } },
   { "data": { "id": "b" } },
   { "data": { "id": "c" } }
  ],
  "edges": [
   { "data": { "id": "ab", "source": "a", "target": "b" } },
   { "data": { "id": "ac", "source": "a", "target": "c" } }
  ]
 }
}
```

Damit das Aussehen eines Knotens passend zu seinem Typ angepasst werden kann, wird der entsprechende Activity Typ auch mit übertragen. Zudem enthalten die Knoten den Namen der Activity und die Kanten deren Beschriftung.

```
{
 "nodes": [
  {
   "data": {
    "id": "0",
    "name": "Start"
   },
   "classes": "CytoscapeStart"
  },
  {
   "data": {
    "id": "1.2",
    "name": "Accept Call"
   },
   "classes": "AcceptCall"
  },
  {
   "data": {
    "id": "1.4",
    "name": "Disconnect Call"
   },
   "classes": "DisconnectCall"
  }
 ],
 "edges": [
  {
   "data": {
    "source": "0",
    "target": "1.2",
    "label": ""
   }
```

```
  },
  {
   "data": {
    "source": "1.2",
    "target": "1.4",
    "label": ""
   }
  }
 ]
}
```

Über eine weitere URL kann der Client das Stylesheet für den Cytoscape.js Graph abrufen, welches in einer CSS ähnlichen Sprache aufgebaut ist. Das Aussehen von einem Knoten kann beispielsweise folgendermassen angepasst werden:

```
node {
  shape: rectangle;
  height: 80;
  width: 100;
  content: data(name);
  color: white;
}
```

8.5 Nachteil der Architektur

Bei dem Zusammenspiel zwischen AngularJS und ASP.NET trat das Problem auf, dass bei einem Browser Refresh die Seiten nicht mehr korrekt geladen wurden. Das funktionierte nur bei der Startseite und solange ohne Refresh auf andere Seiten gewechselt wurde. Die Lösung dazu brachte ein Tutorial zu AngularJS und ASP.NET MVC: http://www.codeproject.com/Articles/806029/Getting-started-with-AngularJS-and-ASP-NET-MVC-Par

Die Refreshs funktionieren jetzt wie gewünscht, jedoch hat die vorgeschlagene Architektur einen Nachteil: Jede URL, die via ASP.NET verfügbar sein soll, muss in RouteConfig des Projekts separat registriert werden. Eine andere Möglichkeit haben wir nicht gefunden. Weil die Anzahl URLs überschaubar ist, haben wir diesen Nachteil in Kauf genommen.

9 Schlussfolgerungen und Ausblick

9.1 POC: Änderung eines Property

Als Proof of Concept soll geprüft werden, ob im Workflow Viewer die Konfiguration bzw. Properties von Activities geändert werden können.

Zum Testen haben wir in der Klasse Homecontroller eine Methode Change angelegt. Diese Methode erwartet 3 Parameter: activityId, propertyName und value (den neuen Wert).

9.1.1 Ansatz 1:

Zuerst haben wir das Ändern eines Properties wie folgt zu lösen versucht:

Wir haben das Property mittels Reflection ausgelesen.

```
public ActionResult Change(string activityId, string propertyName, string value)
{
	if(activityId != "" && propertyName != "" && value != "")
	{
		Activity workflow = (Activity)System.Web.HttpContext.Current.Ses-
	sion["workflow"];
		var activity = WorkflowInspectionServices.Resolve(workflow, activ-
	ityId);
		var property = activity.GetType().GetProperty(propertyName);
		property.SetValue(activity, new InArgument<string>((string)value));
		var wf = WorkflowInspectionServices.GetActivities(workflow).First();
		StringBuilder sb = new StringBuilder();
		System.Xml.XmlWriter writer = System.Xml.XmlWriter.Create(sb);
		WorkflowMarkupSerializer serializer = new WorkflowMarkupSerializer();
		serializer.Serialize(writer, wf);
		writer.Close();
		string xaml = sb.ToString();
	return Content(HttpUtility.HtmlEncode(xaml));
	}
	return Content("");
}
```

Wir haben diese Implementierung getestet und es kam ein ganz anderes XAML heraus, als erwartet:

<?xml version="1.0" encoding="utf-16"?><ns0:Flowchart DisplayName="Flowchart" xmlns:ns0="clr-namespace:System.Activities.Statements;Assembly=System.Activities, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35">

<ns0:Flowchart.StartNode>
<ns0:FlowStep>
<ns0:FlowStep.Next>
<ns0:FlowStep>
<ns0:FlowStep.Next>
<ns0:FlowSwitch_x0060_1>
<ns0:FlowSwitch_x0060_1.Default>
<ns0:FlowStep>
<ns0:FlowStep.Next>
<ns0:FlowSwitch_x0060_1>
<ns0:FlowSwitch_x0060_1.Expression>
<ns1:VariableValue_x0060_1 DisplayName="VariableValue<ConnectResponseType>" xmlns:ns1="clr-namespace:System.Activities.Expressions;Assembly=System.Activities, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35">
<ns1:VariableValue_x0060_1.Variable>

<ns2:Variable_x0060_1 xmlns:ns2="clr-namespace:System.Activities;Assembly=System.Activities, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" />
…

9.1.2 Ansatz 2:

Dann haben wir versucht das XAML mit Hilfe eines Workflow Objekts zu erhalten:

```
Activity workflow = (Activity)System.Web.HttpContext.Current.Session["workflow"];
var activity = WorkflowInspectionServices.Resolve(workflow, activityId);
var property = activity.GetType().GetProperty(propertyName);
property.SetValue(activity, new InArgument<string>((string)value));
//var wf = WorkflowInspectionServices.GetActivities(workflow).First();
//StringBuilder sb = new StringBuilder();
//System.Xml.XmlWriter writer = System.Xml.XmlWriter.Create(sb);
//WorkflowMarkupSerializer serializer = new WorkflowMarkupSerializer();
//serializer.Serialize(writer, wf);
//writer.Close();
//string xaml = sb.ToString();
WorkflowDesigner workflowDesigner = new WorkflowDesigner();
workflowDesigner.Flush();
string xaml = workflowDesigner.Text;
return Content(HttpUtility.HtmlEncode(xaml));
```

Es wird aber im Konstruktor WorkflowDesigner eine InvalidOperationException geworfen.

Diese besagt, dass es sich beim aufrufenden Thread um ein STA-Thread handeln muss, da dies für viele Komponenten der Benutzeroberfläche erforderlich ist.

Wir konnten nicht überprüfen, was das für ein Thread ist, weil wir nicht auf die Implementierung des Konstruktors zugreifen können.

Beim Debuggen kann nicht in den Konstruktor hineingesprungen werden. Es erscheint dieselbe Fehlermeldung.

9.1.3 Ansatz 3:

Ein weiterer Ansatz wäre es, das XAML mit Hilfe eines XamlWriters zu generieren.

```
XamlReader innerReader = new XamlXmlReader(@"..\..\Workflow1.xaml");
        XamlReader builderReader = ActivityXamlServices.CreateBuilderReader(innerReader);
        object content = XamlServices.Load(builderReader);
        object impl = ((ActivityBuilder)content).Implementation;
        var ViewStateName = new AttachableMemberIdentifier(typeof(WorkflowViewStateService), "ViewState");
        // ViewState erfolgreich geladen
        object val;
        bool ok = AttachablePropertyServices.TryGetProperty(impl, ViewStateName, out val);
        Debug.Assert(ok);
        Debug.Assert(val != null);
        XamlWriter innerWriter = new XamlXmlWriter(new StreamWriter(@"..\..\Round-Tripped.xaml"), new XamlSchemaContext());
        XamlWriter builderWriter = ActivityXamlServices.CreateBuilderWriter(innerWriter);
        XamlServices.Save(builderWriter, content);
```

Codquelle:

https://social.msdn.microsoft.com/Forums/vstudio/en-US/9cbfd297-85bc-402a-b2bb-bd2e6ade0929/problems-using-the-workflowdesigner-outwith-wpf-app

Aber das Problem ist, dass beim Speichern eine XamlObjectReaderException geworfen wird. Die Fehlermeldung lautet: "Der Typ 'System.Activities.XamlIntegration.ActivityBuilderXamlWriter' ist nicht sichtbar. Wenn der Typ lokal ist, legen Sie das LocalAssembly-Feld in XamlReaderSettings fest". Wir konnten im Internet dazu keine Lösung finden.

9.1.4 Ansatz 4:

Ich habe auch einen Ansatz gefunden, wo das Speichern in einem Thread ausgeführt wird. Dazu habe ich eine Klasse Workflow erstellt:

```
using System;
using System.Activities.Presentation;
using System.Collections.Generic;
using System.Linq;
using System.Threading;
using System.Web;
using System.Windows;
using System.Windows.Threading;
namespace WFWebviewer.Server.Controllers
{
	public class Workflow
	{
		private readonly Thread staThread;
		private WorkflowDesigner workflowDesigner;
		private SynchronizationContext synchronizationContext;
		public string FileName { get; set; }
		public Workflow()
		{
			staThread = new Thread(SetStaThreadSynchronizationContext) {
		Name = "MyStaThread" };
			staThread.IsBackground = true;
			staThread.SetApartmentState(ApartmentState.STA);
			staThread.Start();

			Thread.Sleep(5000);
		}

		public void Load(string fileName)
		{
		RunInStaThread(() =>
		{
			workflowDesigner = new WorkflowDesigner();
			workflowDesigner.Load(fileName);
		});
		FileName = fileName;
		}

		public void Save(string fileName)
		{
			RunInStaThread(() => {
			if (workflowDesigner == null)
			{
				workflowDesigner = new WorkflowDesigner();
				workflowDesigner.Load();
```

```
                        workflowDesigner.Flush();
                        workflowDesigner.Save(fileName);
                }
                staThread.Abort();
                });
        }

        private void RunInStaThread(Action action)
        {
                var thread = new Thread(state =>
                {
                        var context = state as SynchronizationContext;
                        context.Send(s => action(), null);
                });
                thread.IsBackground = true;
                thread.Start(synchronizationContext);

        }

        private void SetStaThreadSynchronizationContext()
        {

        var temp = new WorkflowDesigner();
        var context = new DispatcherSynchronizationContext(Application.Cur-
rent.Dispatcher);
        SynchronizationContext.SetSynchronizationContext(context);
        synchronizationContext = SynchronizationContext.Current;

        Application.Current.Run();
        }
    }
}
```

Codequelle:

https://social.msdn.microsoft.com/Forums/vstudio/en-US/9cbfd297-85bc-402a-b2bb-bd2e6ade0929/problems-using-the-workflowdesigner-outwith-wpf-app

Beim Aufrufen der URL mit den Parametern ist der Workflow verschwunden. Beim Neuladen (durch Drücken von F5) gibt es eine InvalidOperationException:

Der aufrufende Thread kann nicht auf dieses Objekt zugreifen, da sich das Objekt im Besitz eines anderen Threads befindet.

9.1.5 Ansatz 5:

Alle bisherigen Ansätze waren leider erfolglos. Einen weiteren Ansatz lieferte der Hinweis von Luware, dass sie die Properties der Activities eines Workflows separat in einer Datenbank abspeichern. Wenn man es hinkriegt, die im WFWebviewer an den Client übertragenen Properties mit den IDs der in der Datenbank gespeicherten Properties zu verknüpfen, könnten die Properties mit einem einfachen HTTP POST verändert werden. Wie die Verknüpfung der Properties mit den Einträgen in der Datenbank funktioniert, konnten wir mangels Zeit nicht mehr prüfen. Jedoch sehen wir diesen Ansatz als den vielversprechendsten an.

9.2 Ergebnisse

Die Webapplikation, die den XAML-Code eines Workflows einliest und als Ablaufdiagramm darstellt, konnte vollständig entwickelt werden.

Es kann ein Workflow im XAML-Format als Datei eingelesen werden oder per Copy&Paste in einer Eingabemaske. Beide Möglichkeiten zum Einlesen eines Workflows funktionieren ohne Probleme.

Der eingelesene XAML-Code wird serverseitig mittels eines Parsers ausgewertet und in ein JSON umgewandelt, damit der Workflow als Ablaufdiagramm dargestellt werden kann. Die Darstellung eines Workflow geschieht auf der Clientseite. Dazu wird das JSON zum Client gesendet. Die von Luware eingesetzten Workflow-Elemente werden korrekt dargestellt. Zur Darstellung der Workflow-Elemente wurde das Framework Cytoscape.js verwendet. Es bietet die Möglichkeit an, einzelne Elemente eines Workflows mittels Drag&Drop zu verschieben. Auch der ganze Workflow kann verschoben werden, sowie hinein- und herausgezoomt werden. Die Elemente eines Workflows werden mittels automatischer Ausrichtung übersichtlich dargestellt. Der Workflow kann auf zwei Arten dargestellt werden: einmal von oben nach unten und einmal von links nach rechts.

Die Anzeige der Eigenschaften eines Workflow-Elements wurde vollständig realisiert. Die Eigenschaften werden in einem Popup-Dialog gezeigt.

Weiterhin unterstützt die Anwendung das Hinzufügen neuer Workflow-Elemente. Dazu werden das Stylesheet des Workflows angepasst und der Parser konfiguriert. Im Stylesheet kann beispielsweise das Icon eines Workflow-Elements festgelegt werden. Bei der Konfiguration des Parsers können zum Beispiel die Darstellung der Verzweigungen festgelegt werden. Es können auch gewisse Eigenschaften ein- und ausgeblendet werden.

Die Konfiguration des Parsers wird in einem JSON-Dokument beschrieben.

Das User Interface wurde gemäss dem Styleguide der Luware realisiert.

Es wurde als POC geprüft, ob die Eigenschaften eines Workflow geändert werden können. Die Umsetzung des Änderns einer Eigenschaft bereitete jedoch Probleme. Es wurde versucht die Eigenschaften mittels Reflection auszulesen und dann zu ändern und anschliessend wieder als XAML abzuspeichern. Jedoch unterschied sich das generierte XAML zu stark von dem Original-XAML. Auch die weiteren Ansätze brachten keinen Erfolg. Schlussendlich konnte ein vielversprechender Ansatz gefunden werden, der jedoch nicht mehr getestet werden konnte.

9.3 Schlussfolgerungen

Die Anforderungen wurden vollständig erfüllt. Die entwickelte Webapplikation ist funktionstüchtig. Die Workflows können im Browser angezeigt werden. Die Applikation funktioniert ohne Probleme mit Firefox und Internet Explorer 11. Sie kann auf einem IIS-Server gehostet werden.

Die Umsetzung des User Interface mittels Style Guide der Luware ist gelungen.

Der Benutzer kann eine XAML-Datei, die den Workflow beschreibt, per Dateiupload oder Copy&Paste hochladen. Die XAML-Datei wird in ein JSON umgewandelt. Das JSON wird zur Darstellung des Workflows im Browser genutzt. Es können die Eigenschaften eines Elements des Workflows in einem Popup-Fenster angezeigt werden. Die Elemente und der ganze Workflow können mittels Drag&Drop verschoben werden. Es können neue Elemente durch das Konfigurieren des Workflow-Styles und des Parsers hinzugefügt werden.

Das Ändern von Eigenschaften der Elemente eines Workflows - insbesondere das Abspeichern der Änderungen in ein XAML - machte bei der Umsetzung Probleme und konnte nicht mehr realisiert werden.

10 Anhang

10.1 Anleitungen

10.1.1 Installationsanleitung

Installation auf IIS (Internet Information Services Manager)

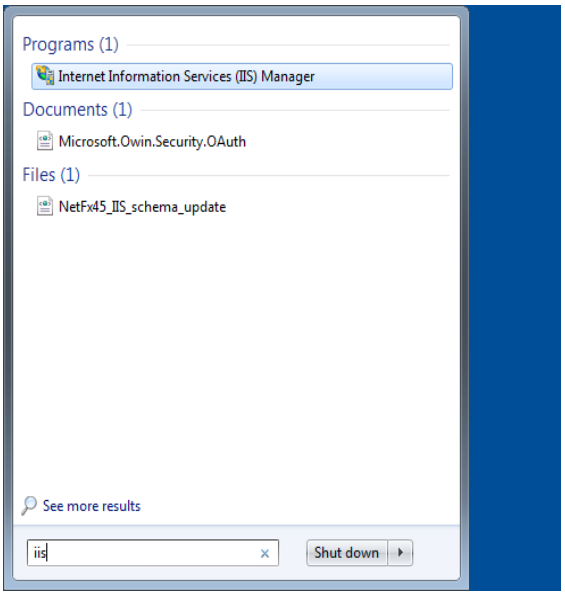
Die Installation wurde unter folgenden Systembedingungen getestet:

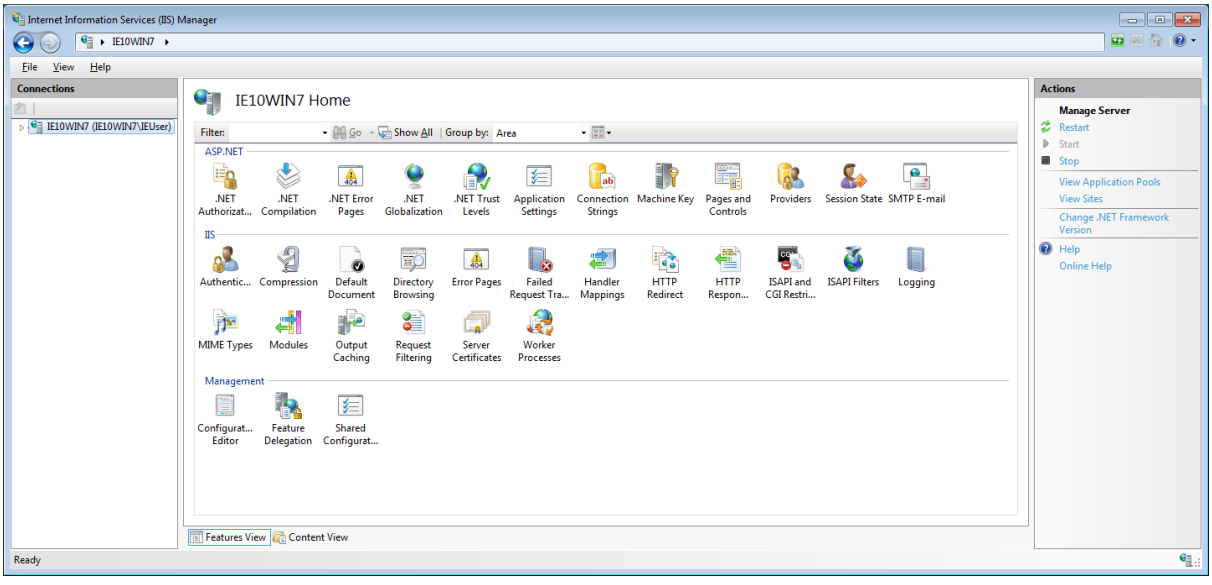
- Windows 7 (32 Bit)
- Internet Explorer 10
- Mozilla Firefox 37
- Google Chrome 42
- .NET Framework 4.5.2

Voraussetzung: .NET Framework 4.5.2 muss installiert sein. Die Applikation kann sich an einem beliebigen Speicherort befinden zum Beispiel in C:\Users\<username>\Documents.

Webapplikation auf Server einrichten

1. Internet Information Services Manager öffnen (Startmenü → nach iis suchen).

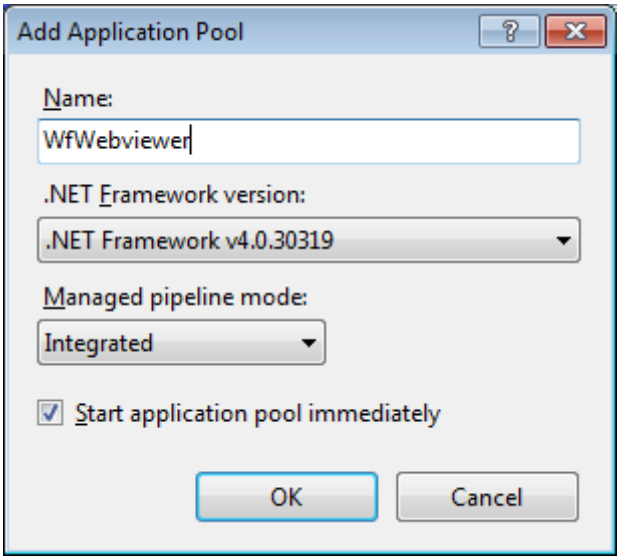




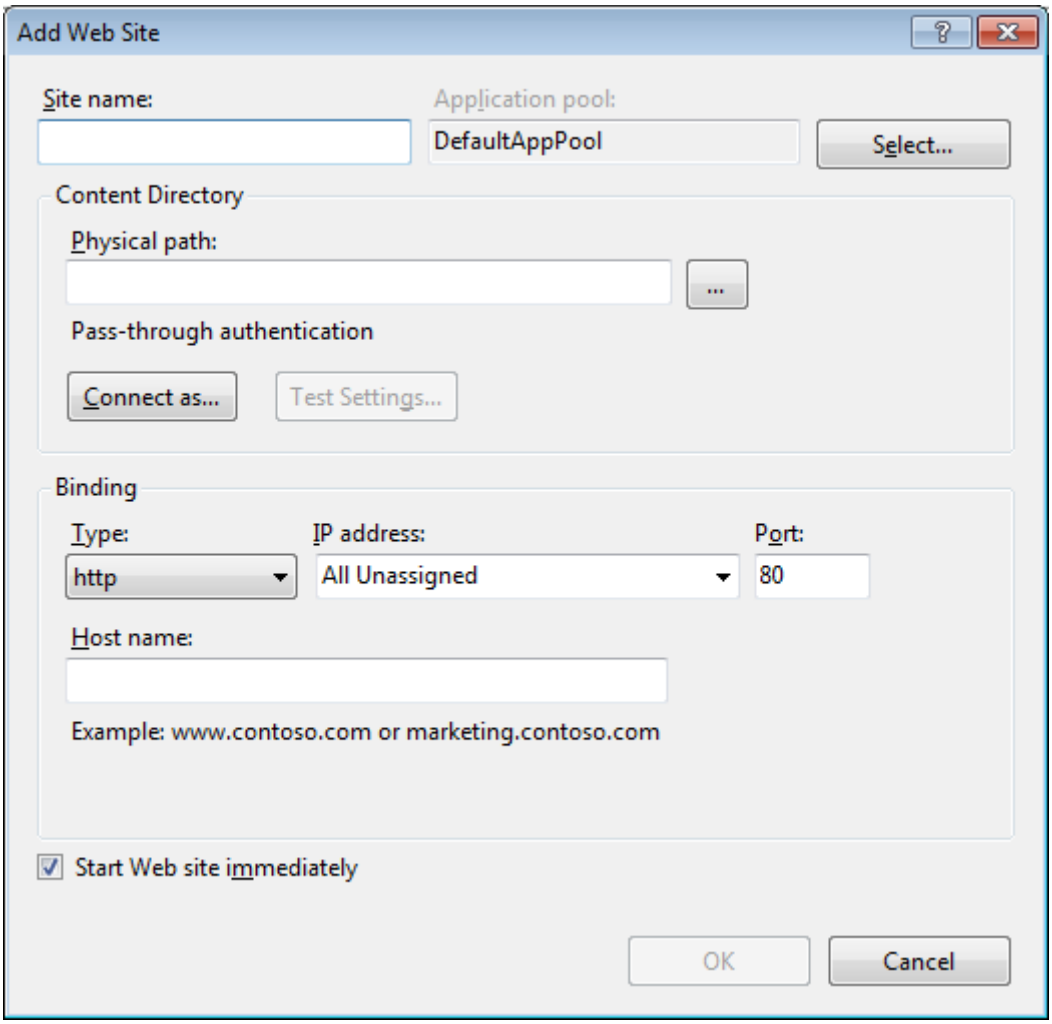
2. Unter IE10WIN7 → Application Pools → Add Application Pool einen neuen Application Pool hinzufügen.

3. Den Application Pool einen Namen geben (z.B. WfWebviewer).

4. Unter .Net Framework version die Version 4 auswählen (siehe Screenshot unten).

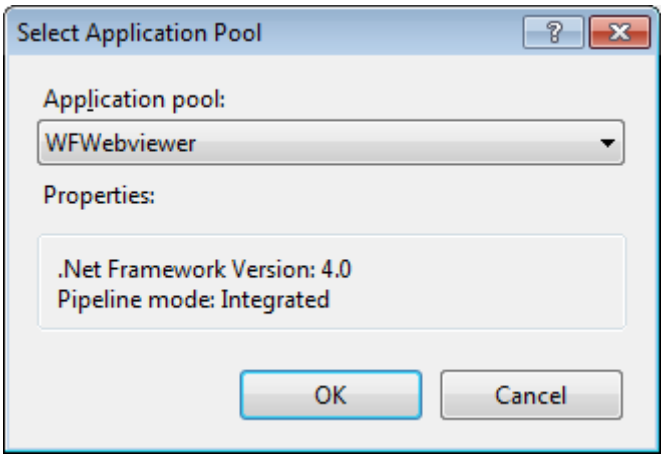


5. Unter Sites auf Add Website klicken, um eine Webseite hinzuzufügen.



6. Unter Site Name einen Namen eingeben z.B. WfWebviewer.

7. Neben Application Pool auf Select klicken, um den Application Pool auszuwählen.



8. Bei Content directory beim Physical path den Pfad zur Applikation setzen (Ordner, wo die Applikation gespeichert ist → Ordner WFWebviewer.Server auswählen).

9. Beim Binding kann eine Webadresse festgelegt werden.

10. Dialog mit OK schliessen.

Zugriffsrechte vergeben

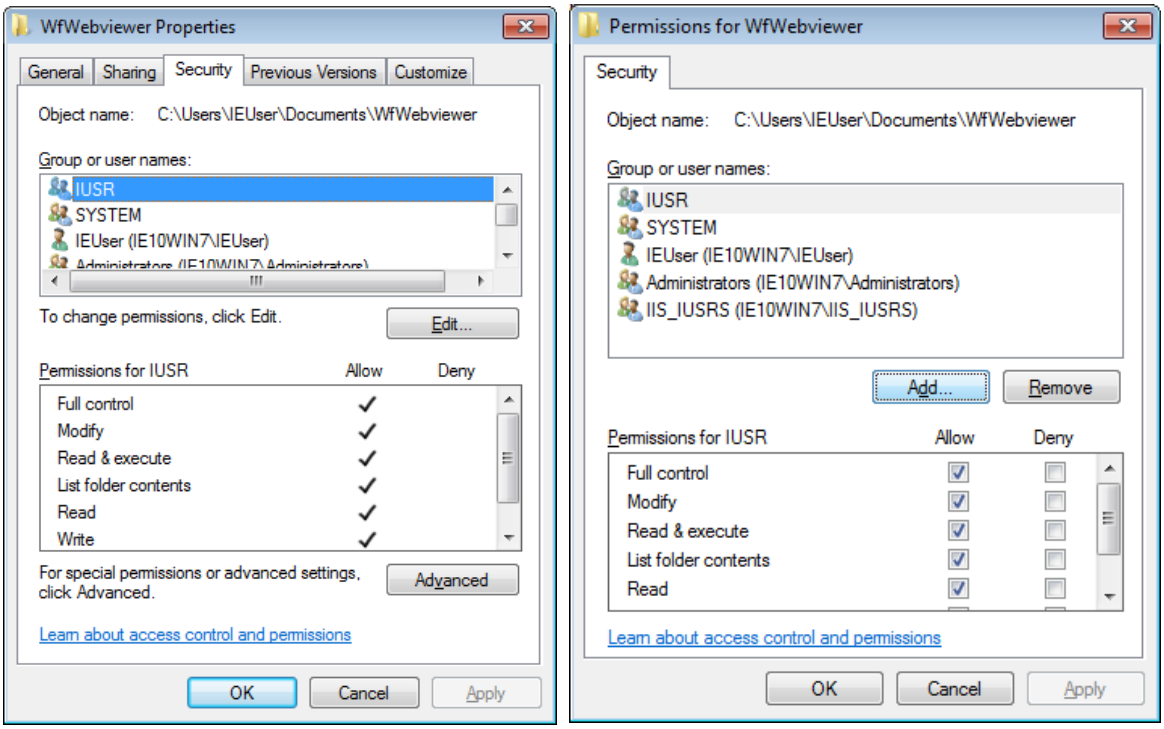
Es müssen die Zugriffsrechte für die Benutzer festgelegt werden.

1. Den WfWebviewer unter Sites auswählen.

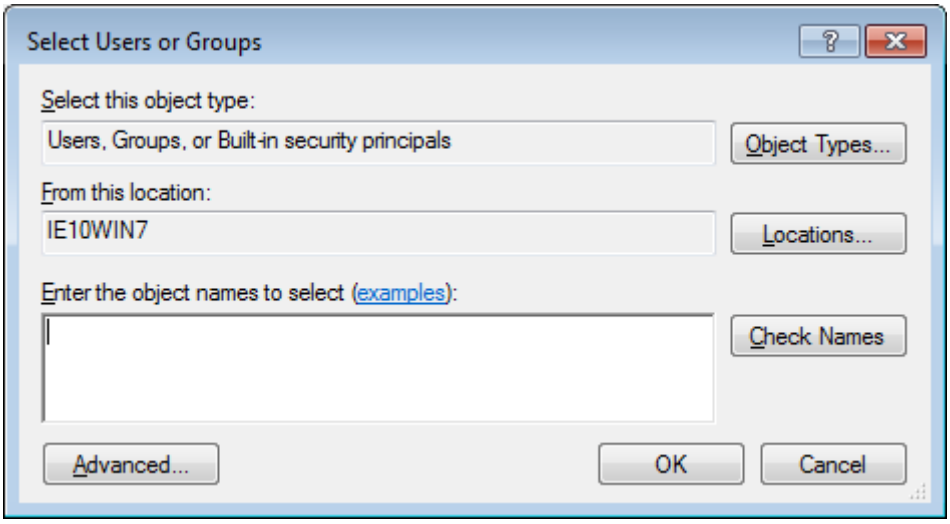
2. Mit einem Rechtsklick das Kontextmenü öffnen und auf Edit Permissions… klicken. Es öffnen sich die Eigenschaften der Applikation.

3. Den Tab Security auswählen.

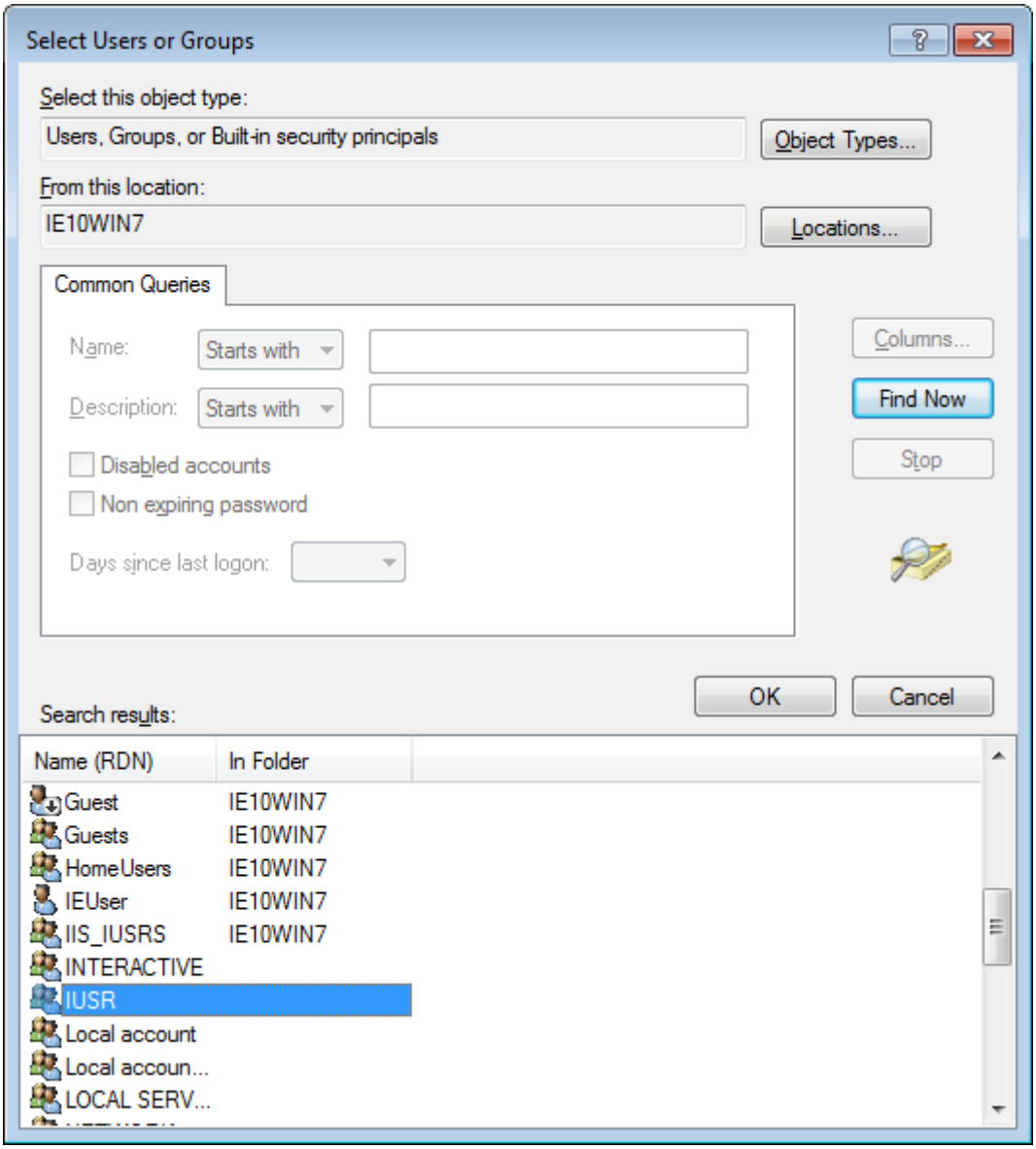
4. Auf Edit… klicken und im nächsten Dialog auf Add… klicken um einen Benutzer hinzuzufügen.



5. Im nächsten Dialog auf Advanced... klicken, um einen Benutzer zu finden.



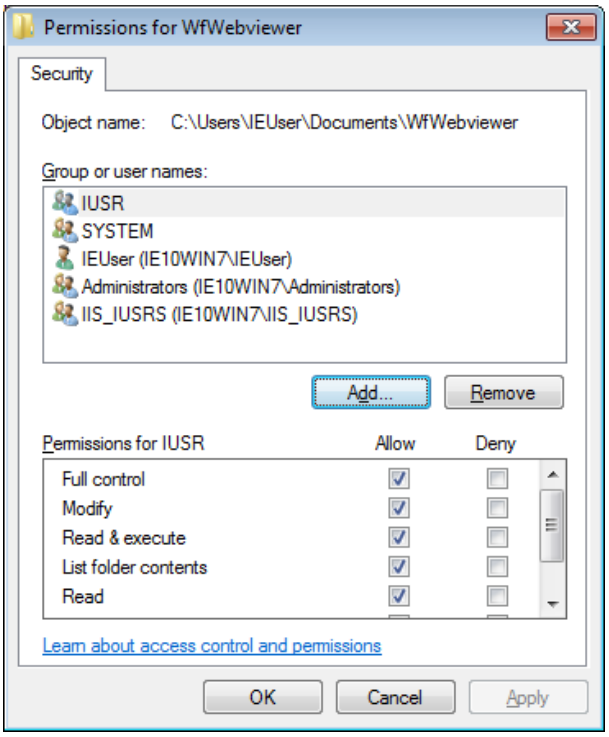
6. Im nächsten Dialog auf Find Now klicken und bei Search Results den Benutzer IUSR auswählen.



7. Die Auswahl mit OK bestätigen und den anderen Dialog mit OK schliessen.

8. Die Schritte 4-7 für die Benutzer IEUser und IIS_IUSRS wiederholen. Diese Benutzer müssen hinzugefügt werden. Anderenfalls wird im Browser eine Fehlermeldung angezeigt.

9. Nun den Benutzer IUSR auswählen und die Zugriffsrechte auf Allow setzen.



10. Den Dialog mit OK schliessen.

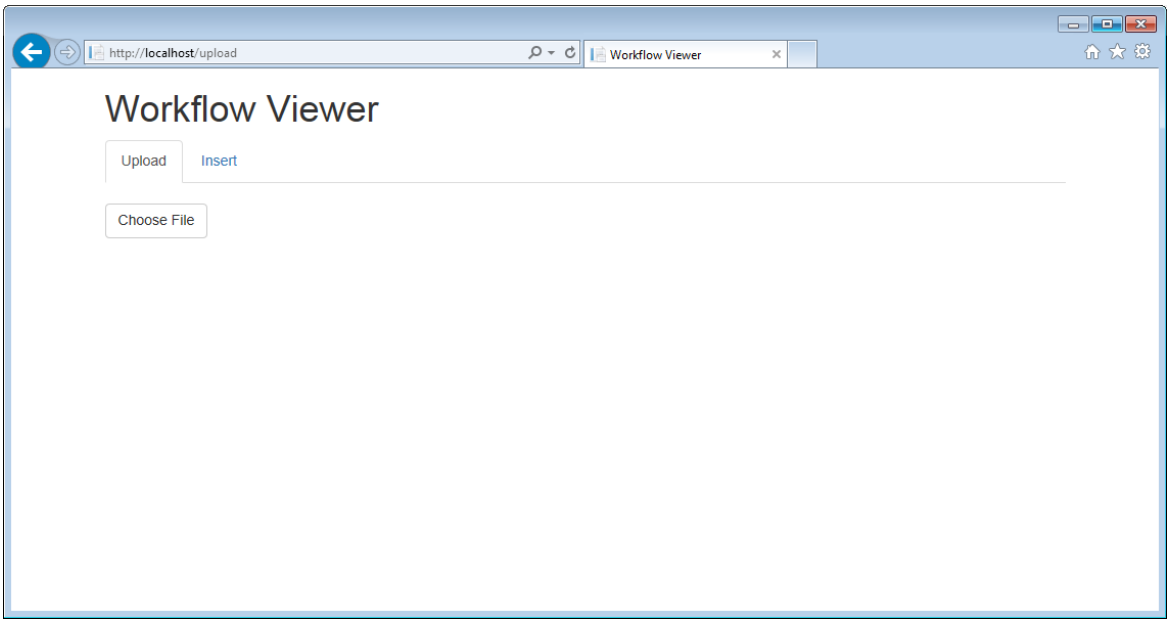
11. Im Properties Dialog auf Apply klicken und dann den Dialog mit OK schliessen.

Server starten und Webseite anzeigen

1. Rechtsklick auf WfWebviewer → Manage Website → Start

2. Internet Explorer oder Mozilla Firefox öffnen.

3. Als Adresse: http://localhost/ eingeben. Die Startseite sollte erscheinen.



10.1.2 Installation Manual

Installation on IIS (Internet Information Services Manager)

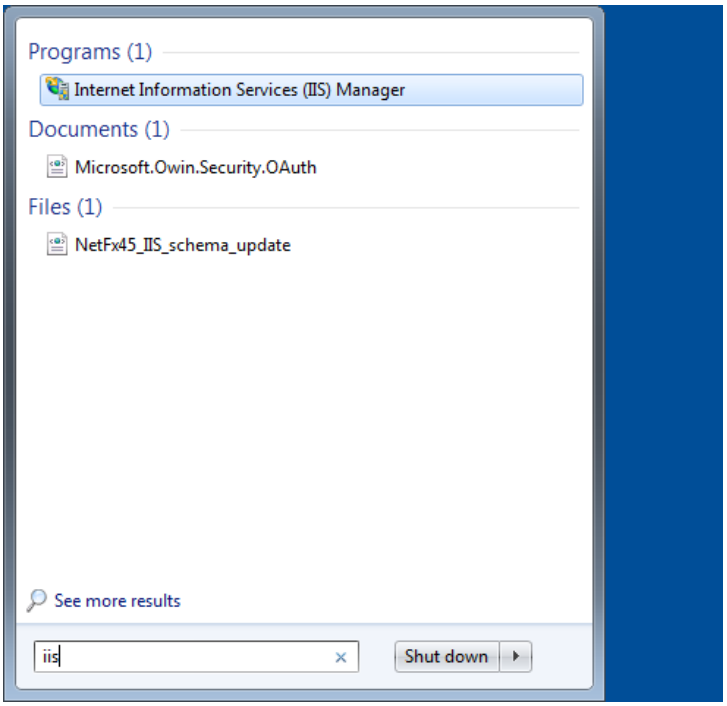
The installation was tested with the following system conditions:

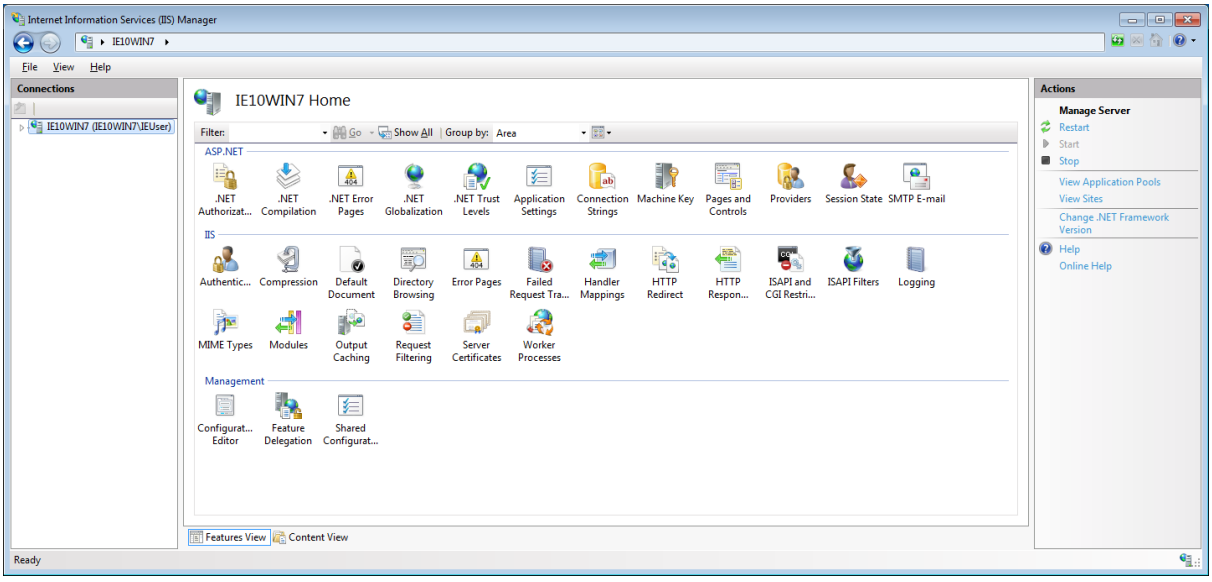
- Windows 7 (32 Bit)
- Internet Explorer 10
- Mozilla Firefox 37
- Google Chrome 42
- .NET Framework 4.5.2

Required: Make sure that .NET Framework 4.5.2 is installed. If not please install .NET Framework 4.5.2.
The application can be stored at any folder for example: C:\Users\<username>\Documents.

Install web application on IIS-server

1. Open Internet Information Services Manager. You can enter “iis” in the start menu to find it.

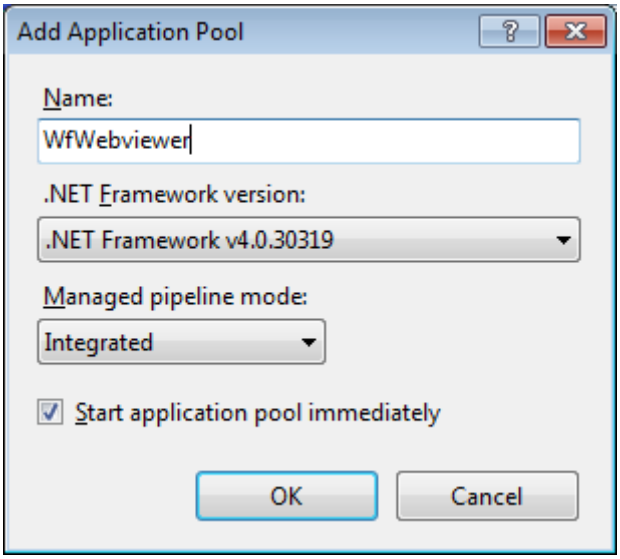




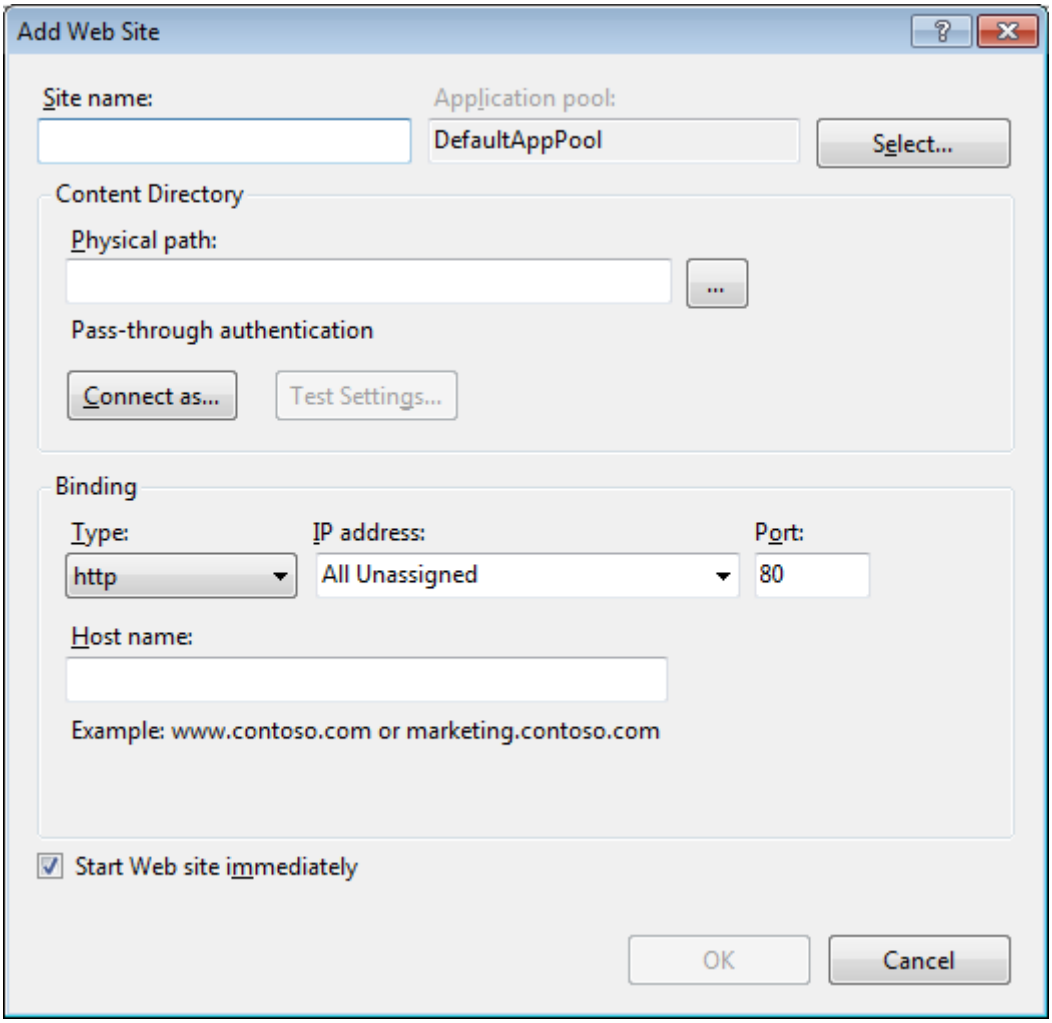
2. Click on IE10WIN7 → Application Pools → Add Application Pool to add a new application pool. The "Add Application Pool" dialog will pop up (screenshot below).

3. Give a name to the application pool (e.g. WfWebviewer).

4. Choose version 4 of .NET Framework (see screenshot below). Then click on OK.

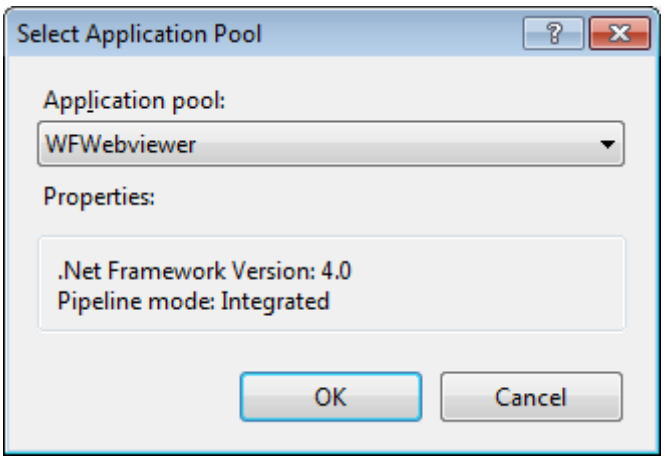


5. To add a website click on Sites → Add web site. The following dialog pops up:



6. Enter a name at "Site name".

7. Click on Select… next to “Application pool” to select an application pool. The following screen pops up:



8. Select your application pool (WFWebviewer in this example) and then click OK.

9. Click on … next to the Physical path text field to choose the location where the web application is stored (choose the directory WFWebviewer.Server).

10. You can choose a web protocol, an ip-address, port and host name (optional).

11. Click OK to close the dialog.

Assign accessing rights

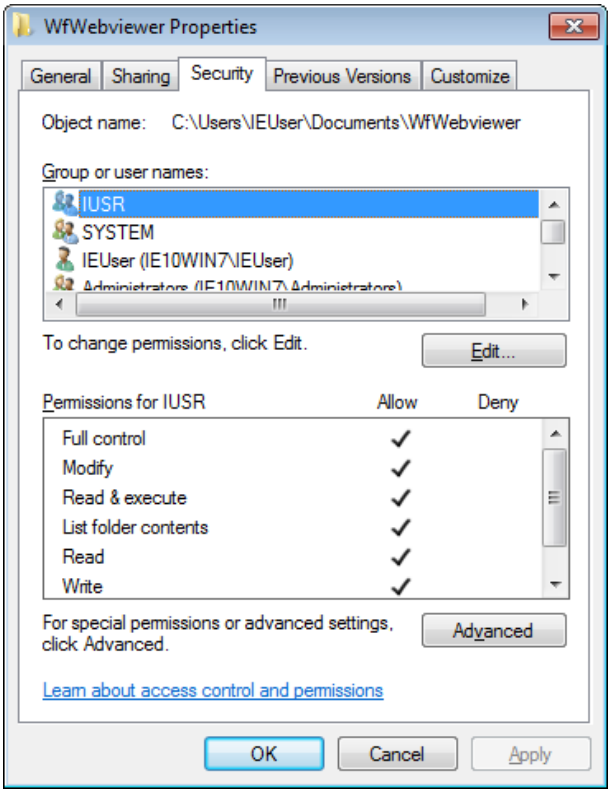
You have to assign the accessing rights to certain users in order to run the web application.

1. On sites choose your web application (WfWebviewer)

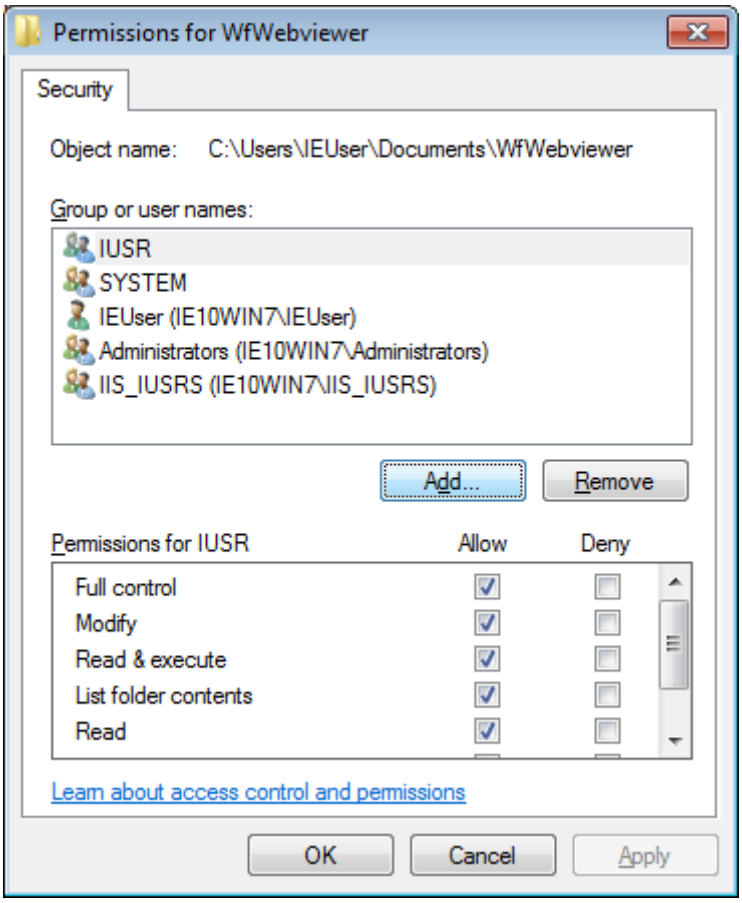
2. Click on Edit permissions… via context menu. The properties dialog will pop up.

3. Choose the tab “Security”.

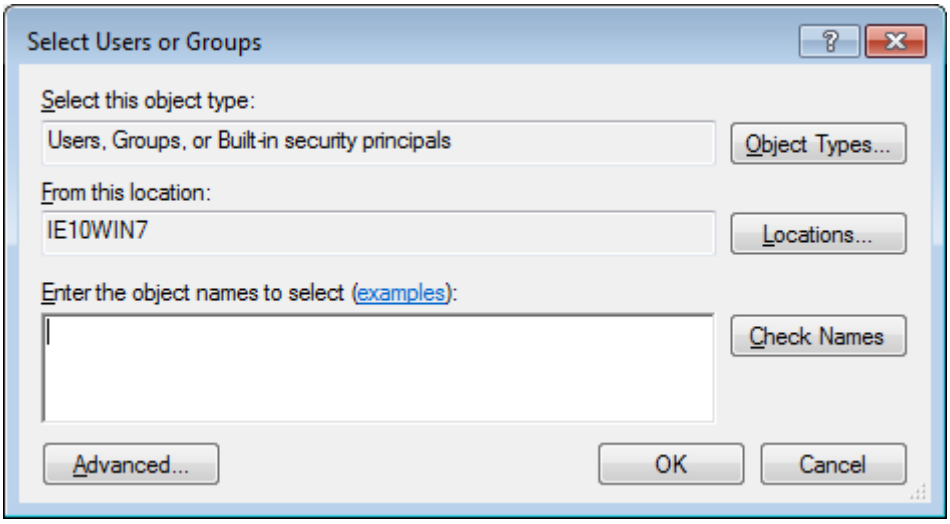
4. Click on Edit. The dialog for editing the permissions will pop up.



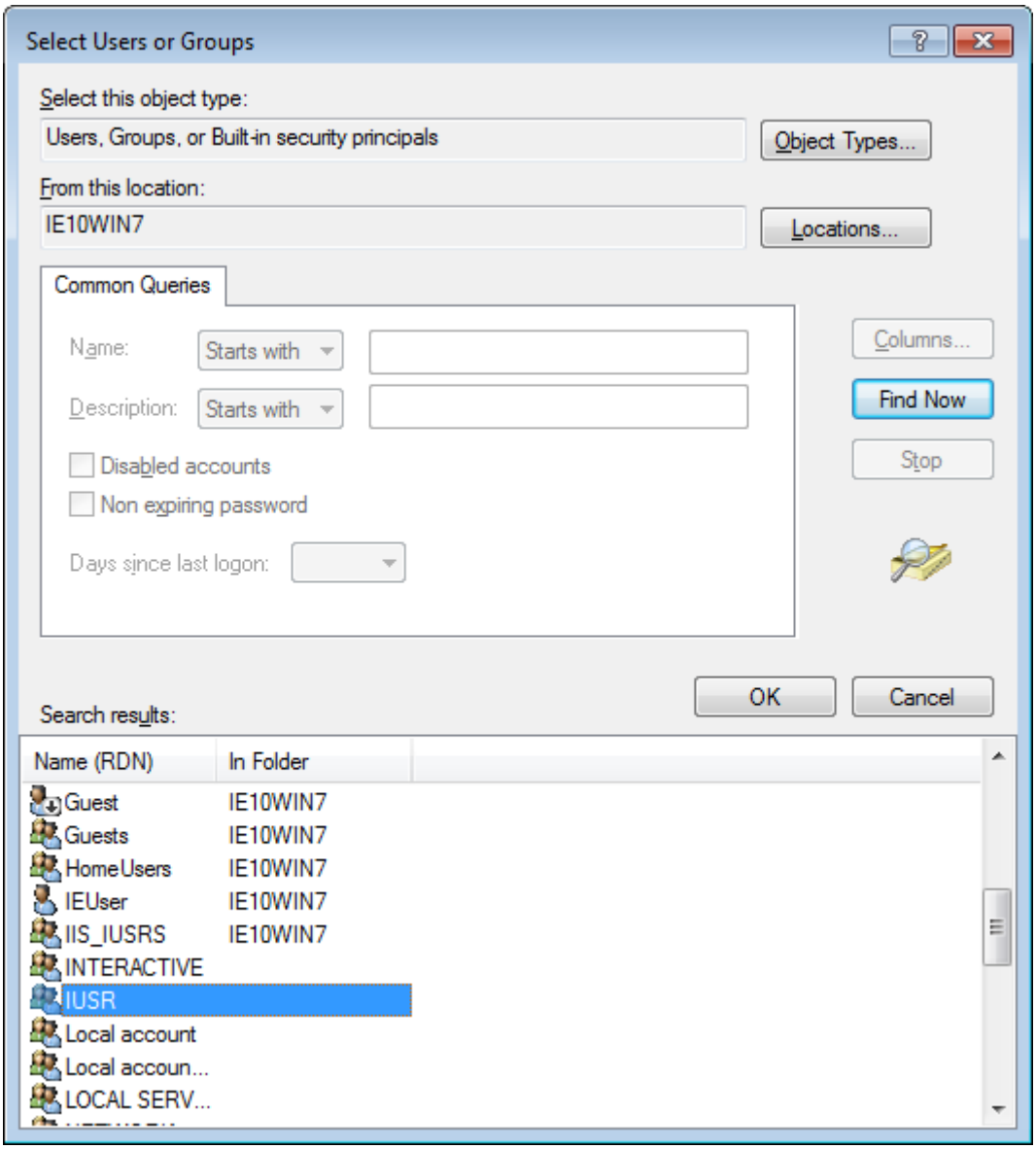
5. Click on Add. The dialog for selecting users will pop up.



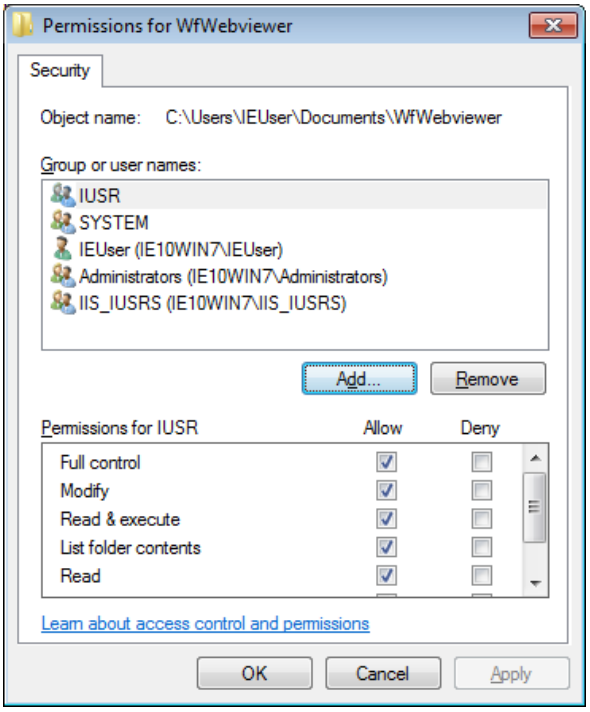
6. In the next dialog click on “Advanced…” to find a user.



6. Click on “Find Now”. A list of users will appear. Choose IUSR.



7. Click OK to close the dialog. Click OK again to close the next dialog.



8. Repeat step 4-7 for the users IEUser and IIS_IUSRS. It is required to add these users. Otherwise you will get an error message in the browser.

9. Choose IUSR and tick all the permissions. Repeat this with IEUser and IIS_IUSRS.

10. Click OK to close the dialog.

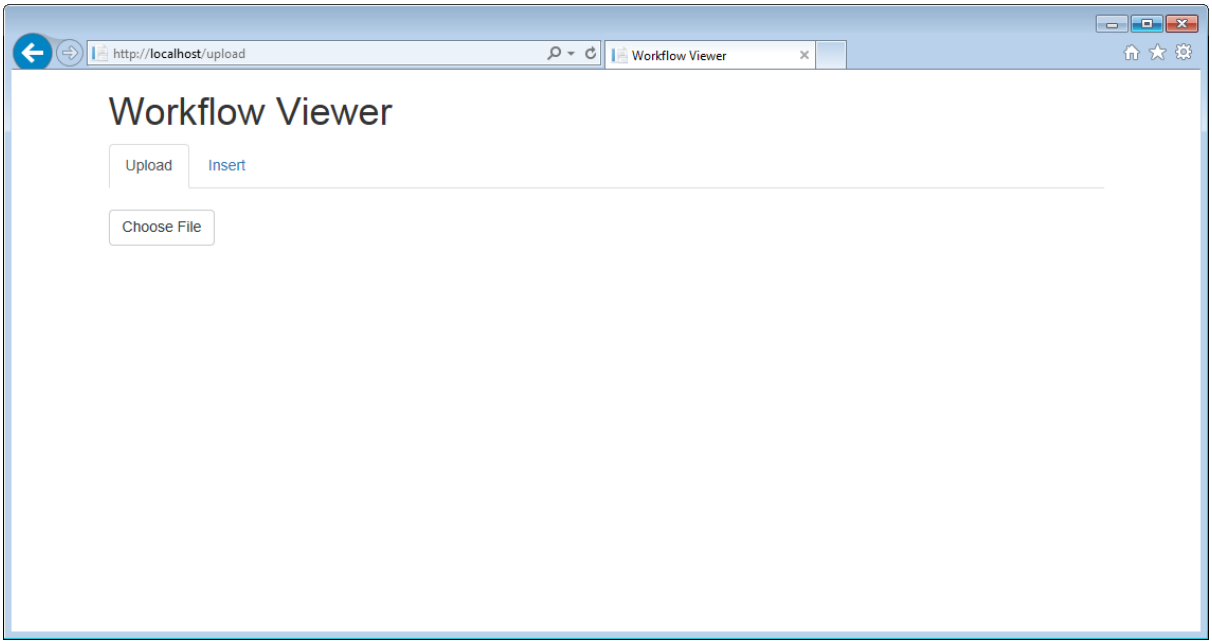
11. In the properties dialog click on Apply and then click OK to close it.

Start server and run the web application

1. Start the IIS-server by clicking on the application name → Manage website → Start.

2. Open Internet Explorer or Mozilla Firefox.

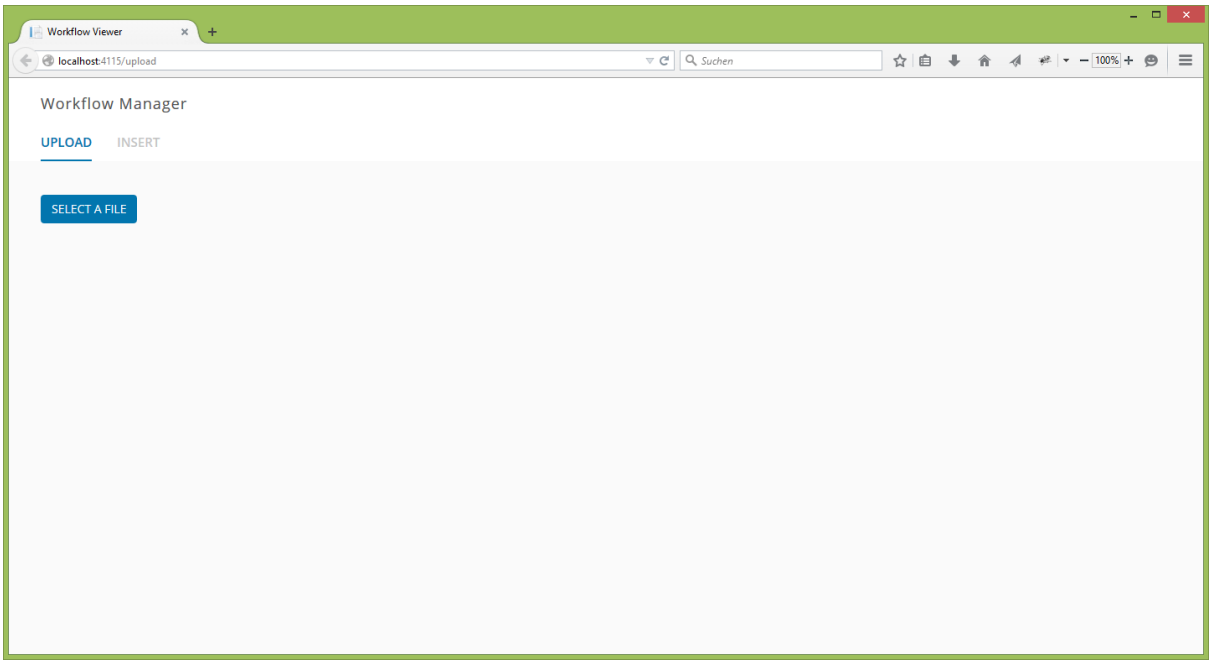
3. Enter http://localhost/ as address. The start page should appear.



10.1.3 Benutzerdokumentation

Der Workflow Viewer ist eine Webapplikation, mit der man Workflows betrachten kann.

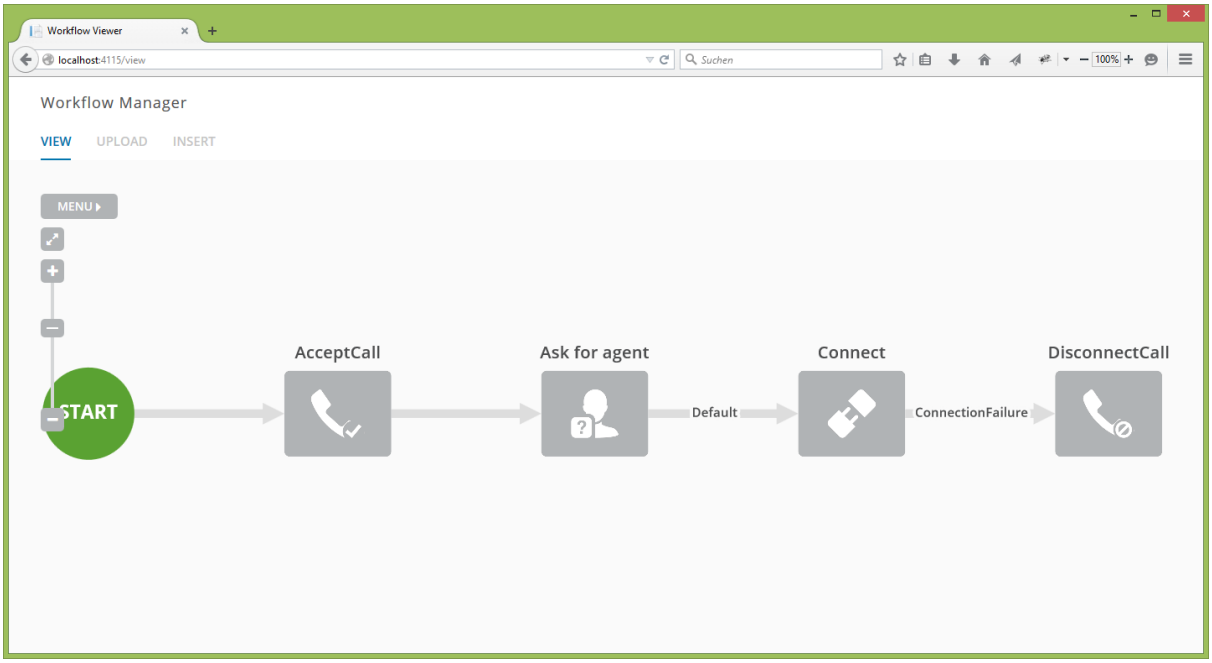
Nach dem Laden der Applikation sieht die Startseite wie folgt aus:



Man kann von der Startseite aus einen Workflow als Datei hochladen (Upload) oder den Inhalt der XAML-Datei in ein Textfeld per Copy&Paste einfügen (Insert).

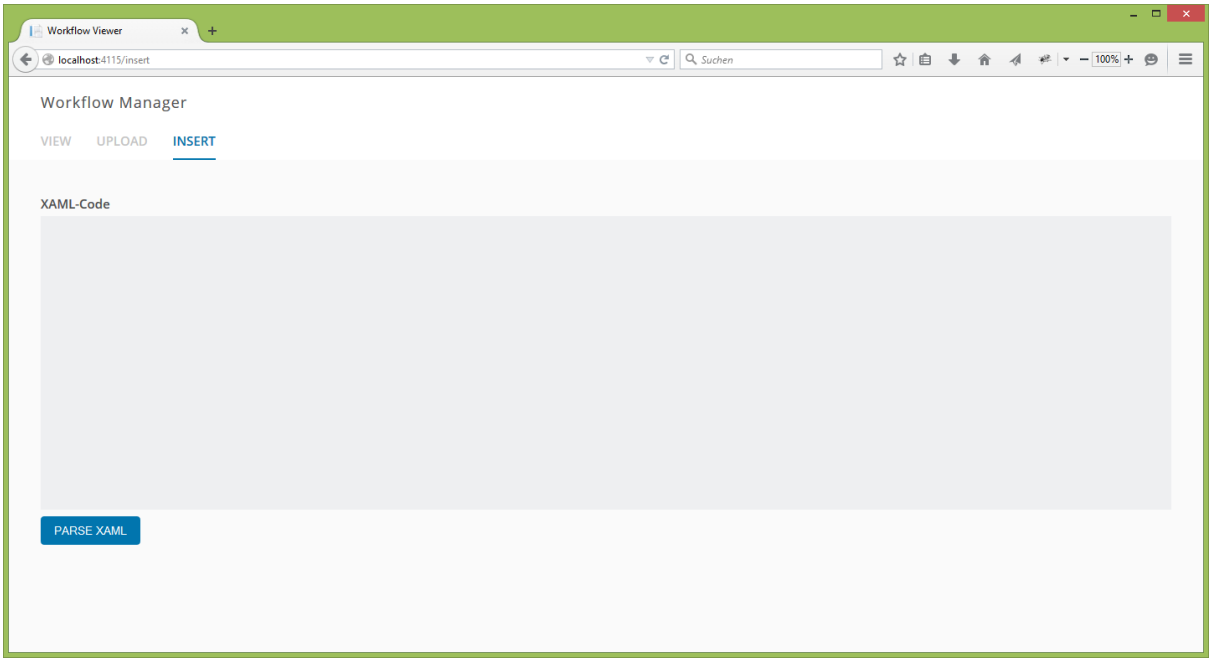
Einen existierenden Workflow aus einer Datei hochladen

1. Klicken Sie auf den Tab “Upload”, falls der Tab “Insert” aktiv ist.
2. Klicken Sie auf “Choose File”. Es öffnet sich der Dialog zur Dateiauswahl.
3. Wählen Sie die XAML-Datei die den Workflow enthält, den Sie betrachten möchten.
4. Anschliessend wird der Workflow geladen.



Einen existierenden Workflow per Copy&Paste hochladen

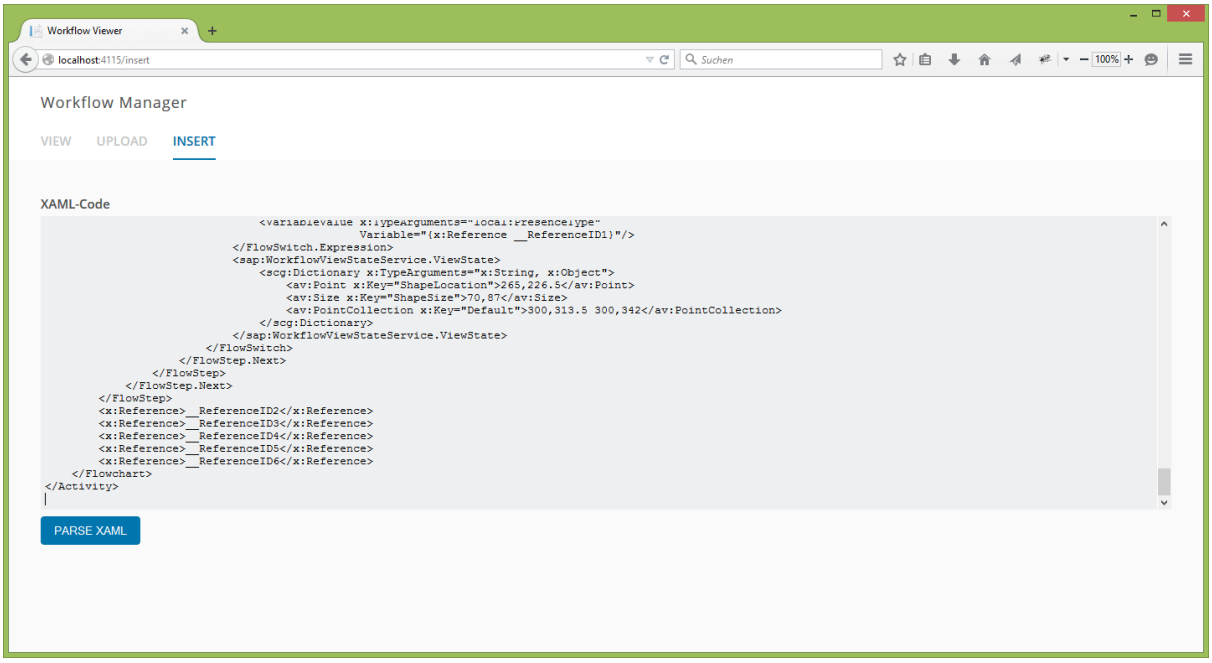
1. Klicken Sie auf den Tab “Insert”. Es erscheint der folgende Screen:



2. Wechseln Sie in eine geöffnete XAML-Datei, die den XAML-Code des Workflows enthält oder öffnen Sie eine solche Datei mit einem Texteditor.

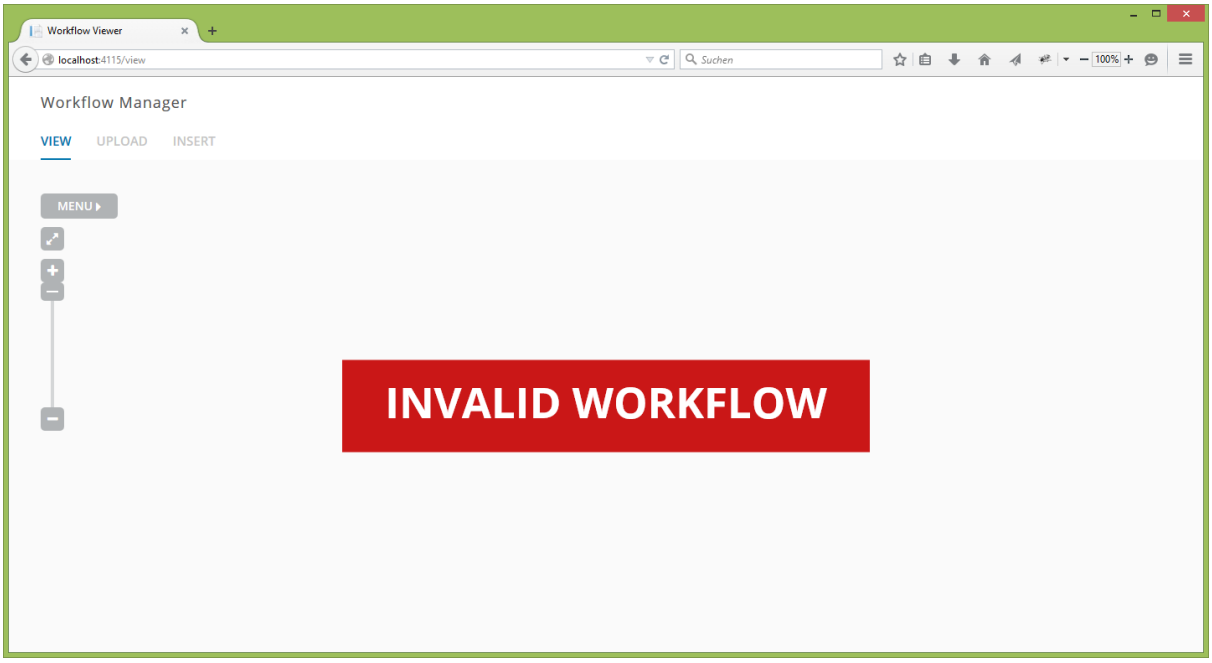
3. Markieren Sie den XAML-Code und kopieren Sie ihn mit Strg-C.

4. Klicken Sie in das grosse Textfeld und fügen Sie den XAML-Code mit Strg-V oder Rechtsklick-Einfügen ein.



5. Klicken Sie auf “Parse XAML”, um den Workflow zu laden und anzuzeigen.

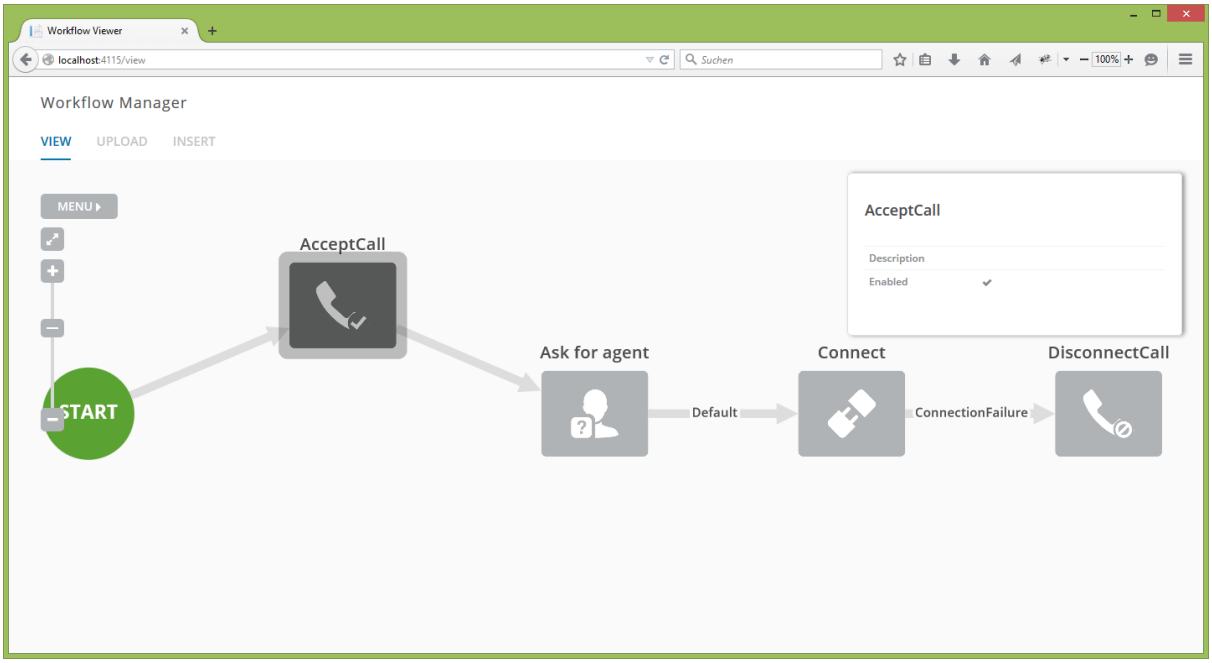
6. Wenn die XAML-Datei einen Fehler enthält, wird die Fehlermeldung “Invalid Workflow” angezeigt



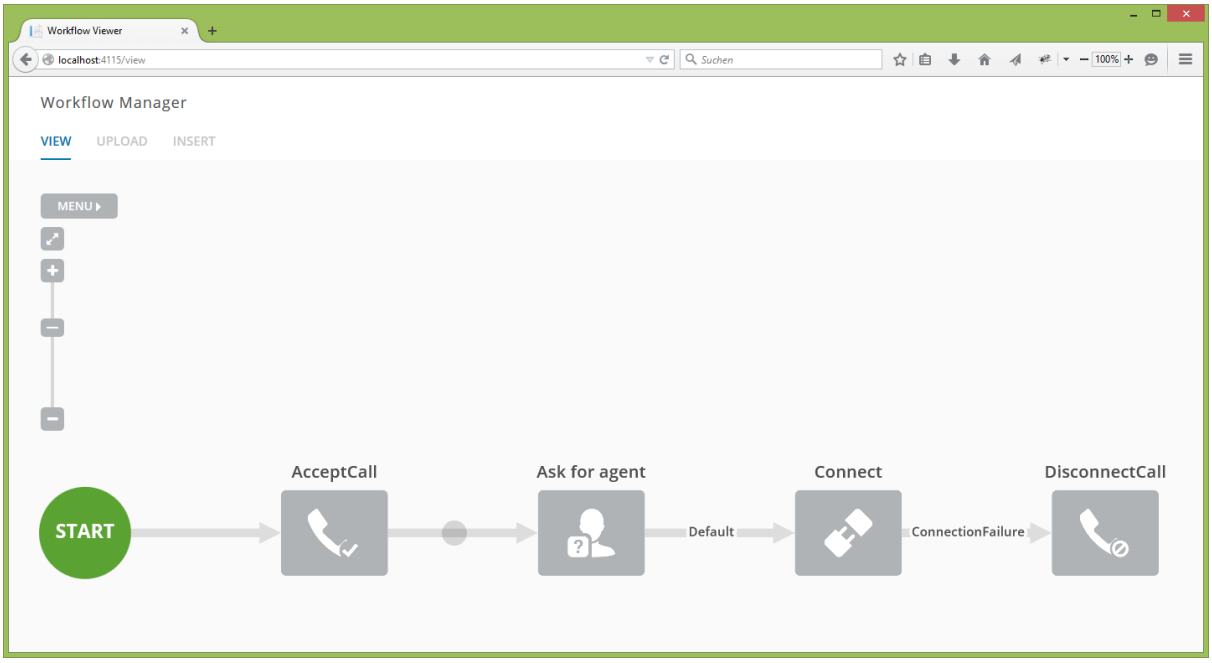
Verschieben von Elementen

Einzelne Activities können verschoben werden. Durch das Klicken auf einen anderen Bereich, lässt sich der ganze Workflow verschieben.

1. Klicken Sie auf ein Element um es auszuwählen. Es färbt sich dunkelgrau.
2. Das Element kann mit der Maus verschoben werden.
3. Klicken Sie im Menu auf Reset um die ursprüngliche Darstellung wieder zu erhalten.



Ein ausgewähltes Element kann verschoben werden, wobei die Verbindungen erhalten bleiben.

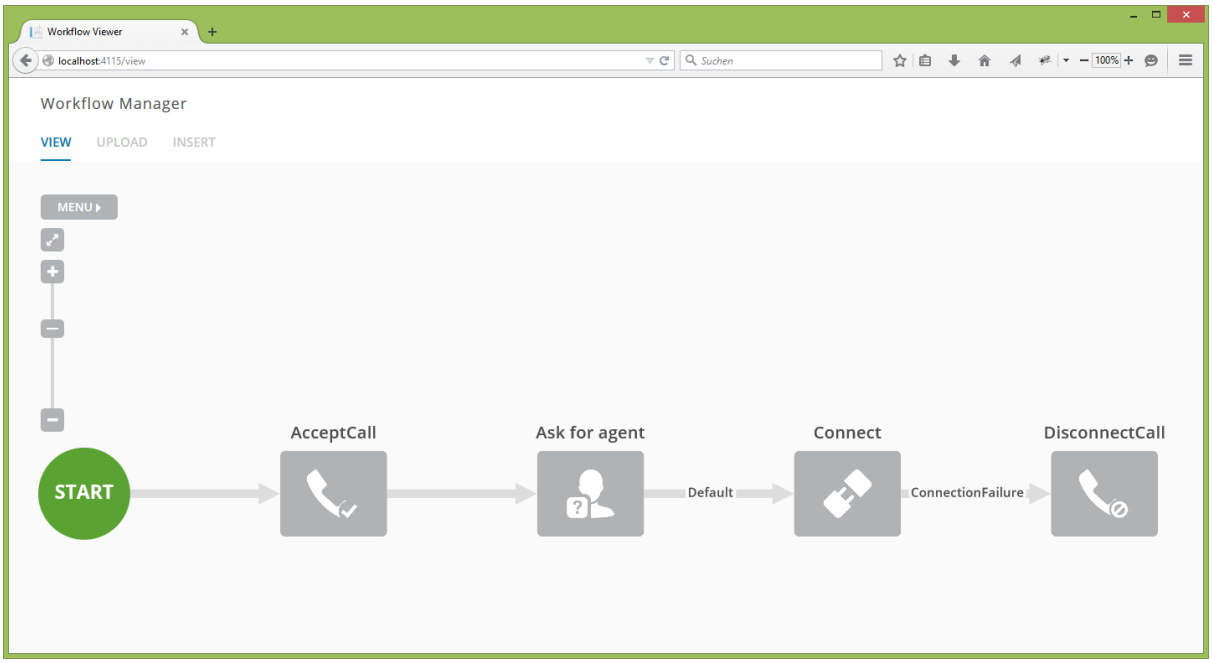


Beim Verschieben eines Workflows befindet sich an der Verbindung ein kleiner Kreis.

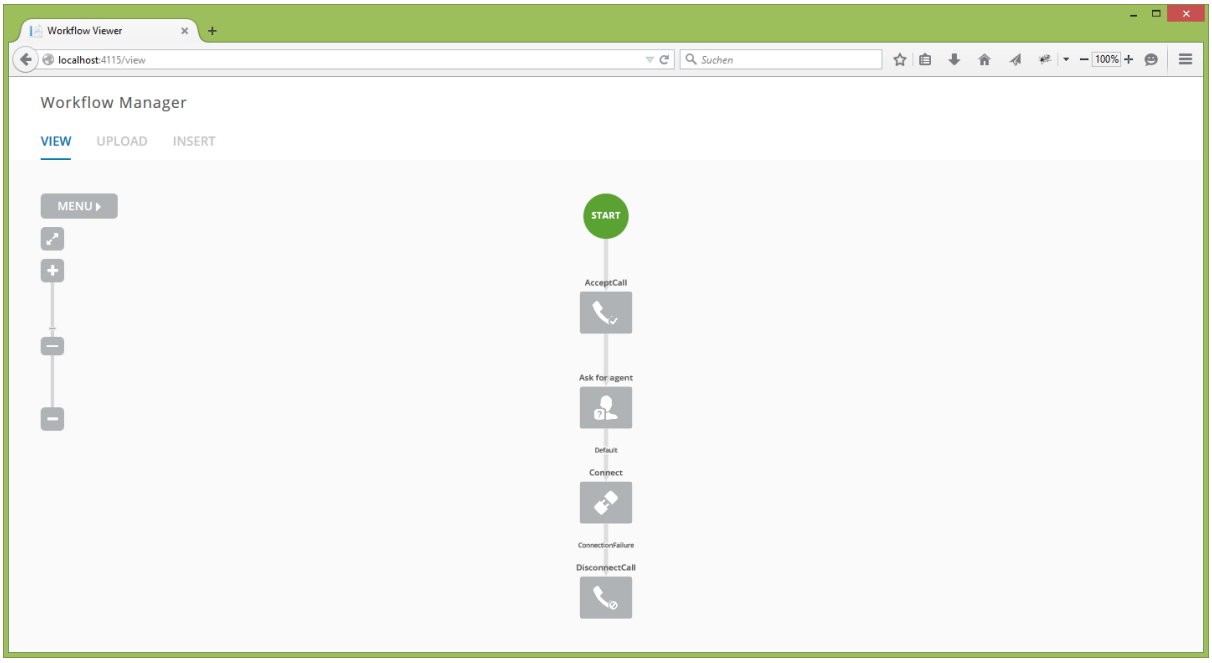
Ändern der Ansicht eines Workflows

Workflows können in 2 Ansichten dargestellt werden:

- Darstellung von links nach rechts (left - right)
- Darstellung von oben nach unten (top - bottom)



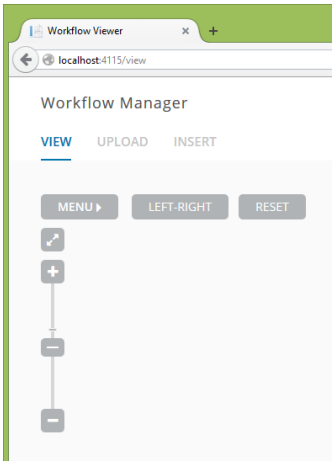
Darstellung des Workflows von links nach rechts (left - right)



Darstellung des Workflows von oben nach unten (top - bottom)

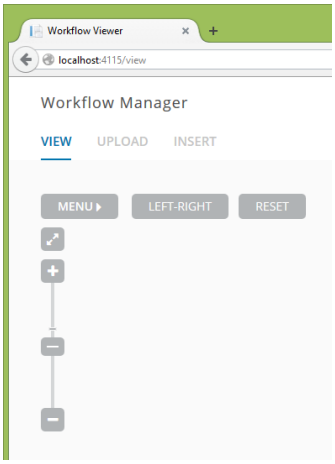
Darstellung des Workflows von oben nach unten

Der Workflow wird von links nach rechts angezeigt. Klicken Sie im Menu auf Top-Bottom um den Workflow von oben nach unten anzuzeigen.



Darstellung des Workflows von links nach rechts

Der Workflow wird von oben nach unten angezeigt. Klicken Sie im Menu auf Left-Right um den Workflow von links nach rechts anzuzeigen.

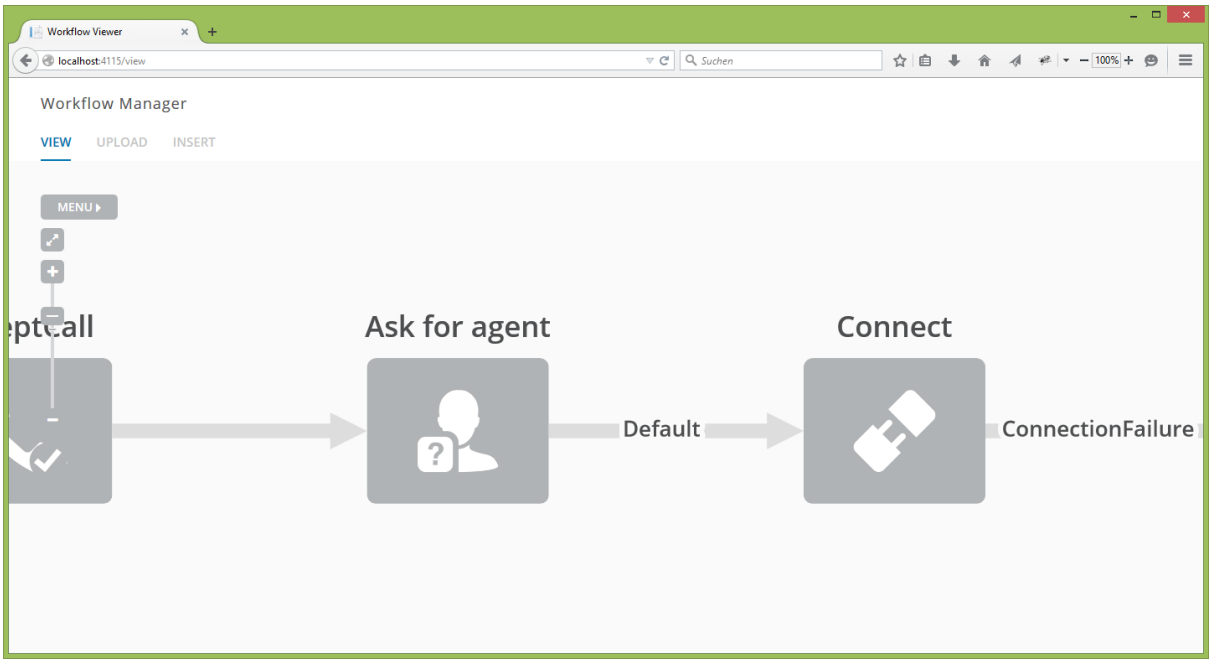


Standardmässig wird der Workflow von links nach rechts angezeigt.

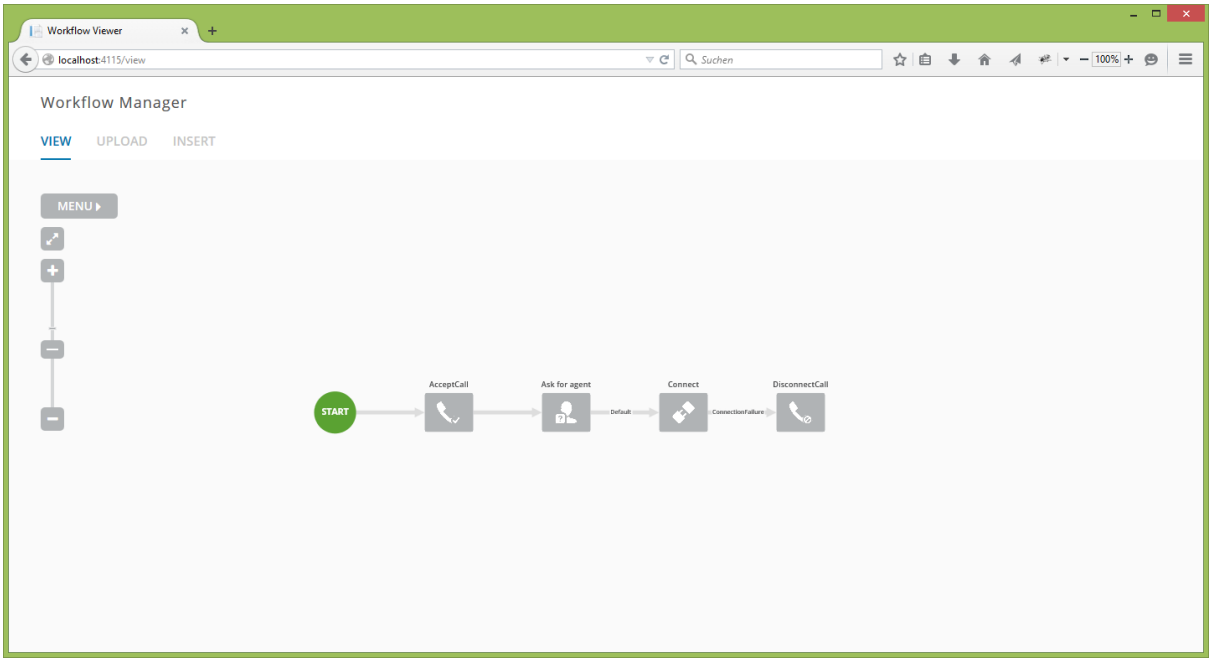
In den Workflow hinein- oder hinauszoomen

Es kann mit dem Mausrad oder mit dem Schieberegler in den Workflow hinein- oder hinausgezoomt werden.

Klicken Sie auf  an der Oberseite des Schiebereglers oder schieben Sie den Regler nach oben, um in den Workflow hineinzuzoomen.



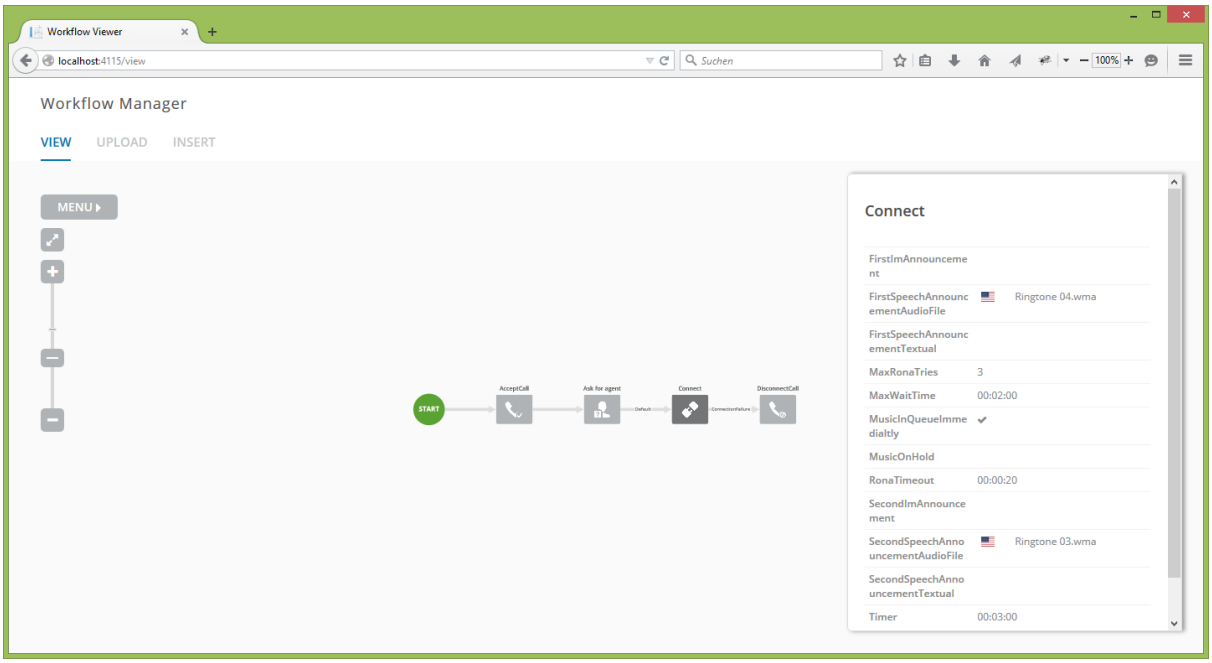
Klicken Sie auf  an der Unterseite des Schiebereglers oder schieben Sie den Regler nach unten, um aus dem Workflow herauszuzoomen.



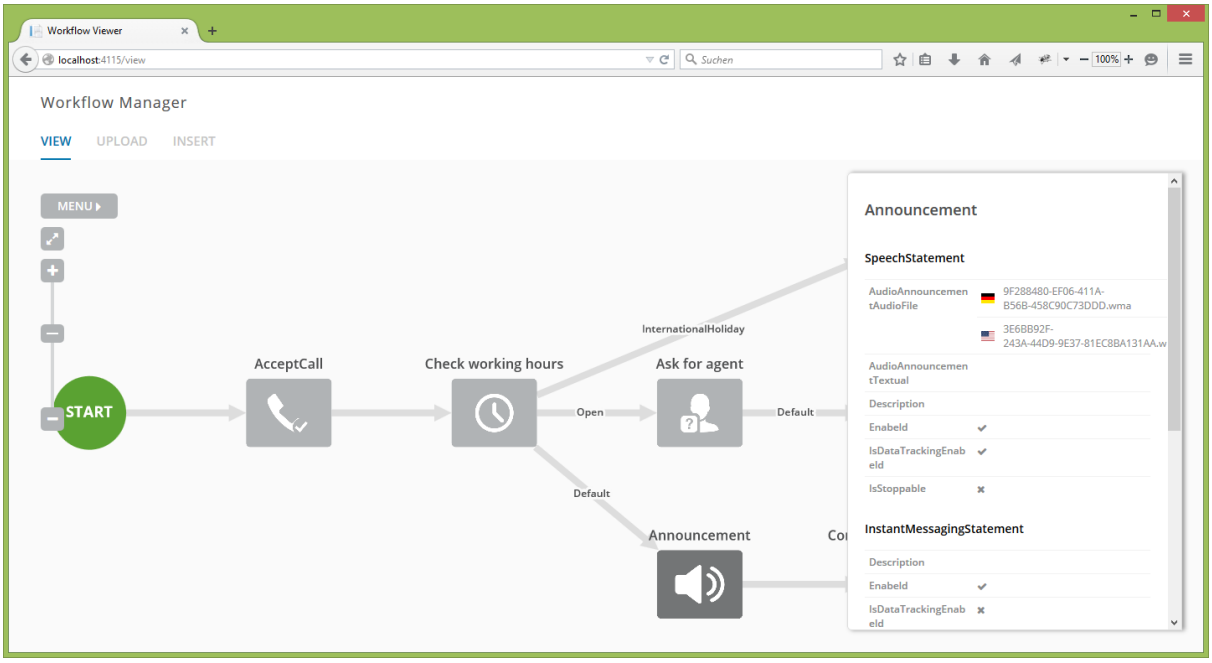
Klicken Sie auf  um den ursprünglichen Zoomlevel wiederherzustellen.

Darstellung der Eigenschaften eines Workflows

Markieren Sie ein Element des Workflows. Es erscheint rechts eine Box mit den Eigenschaften des markierten Elements.



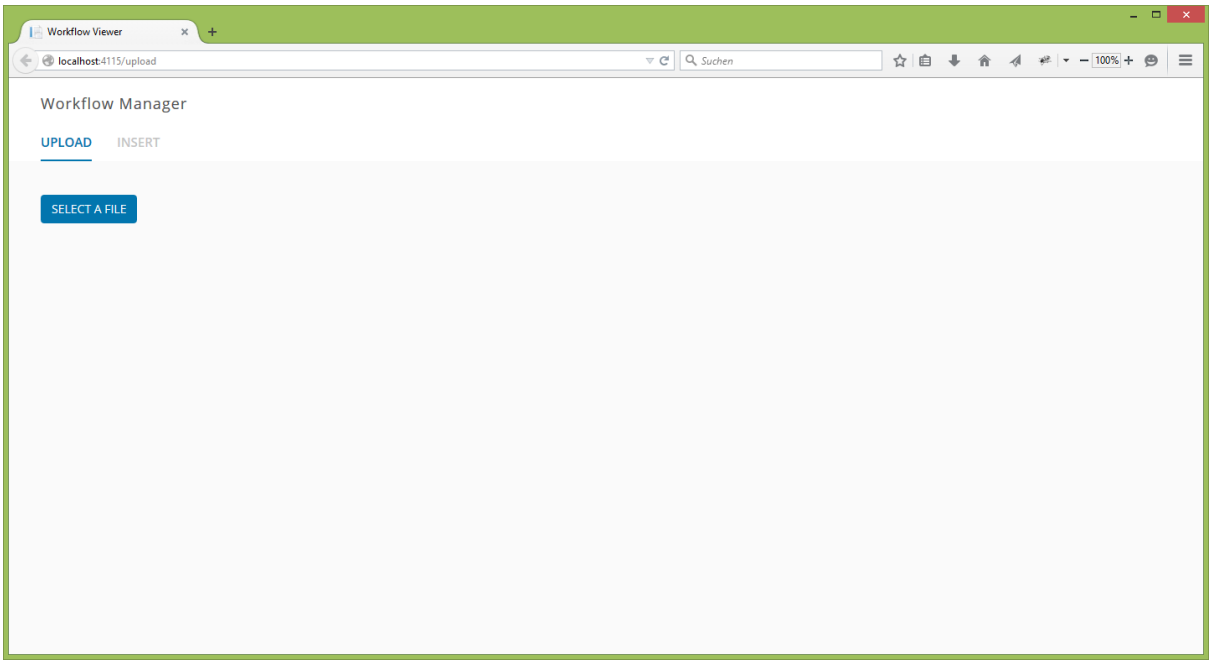
Eigenschaften eines Elements können in Kategorien zusammengefasst sein. Hier ein Beispiel:



10.1.4 User documentation

Workflow Viewer is a web application that lets you examine Workflows.

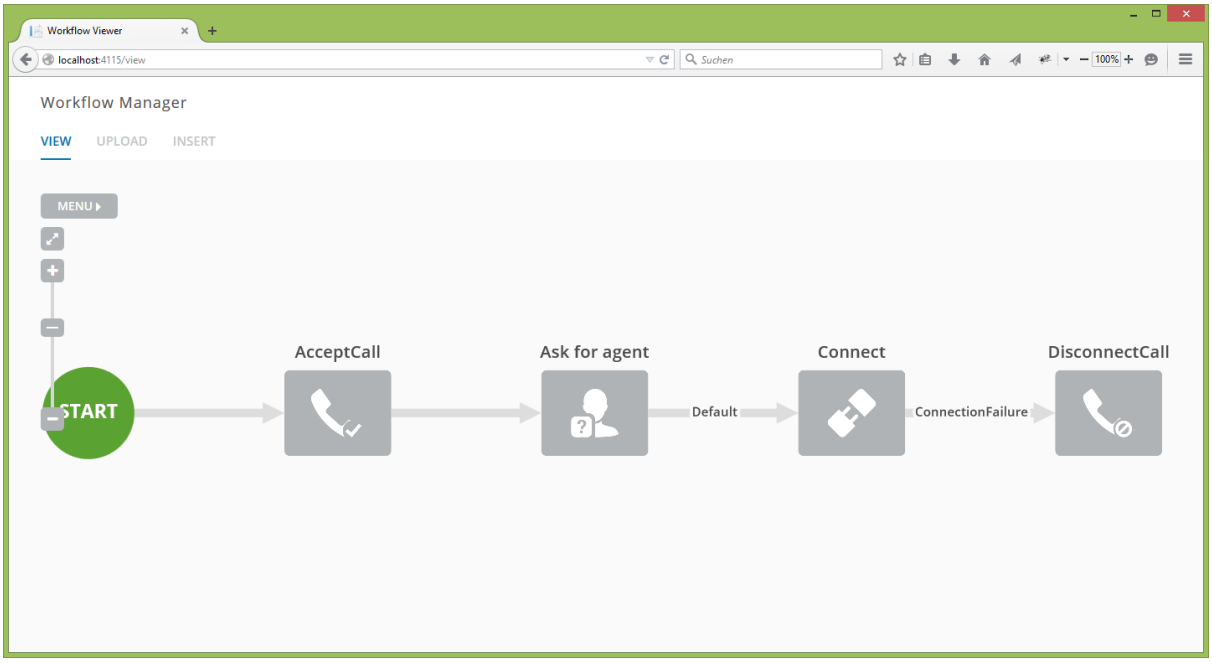
This is what the main page looks like:



On this page you can upload a workflow file or insert XAML code in the designated text field by copy and paste.

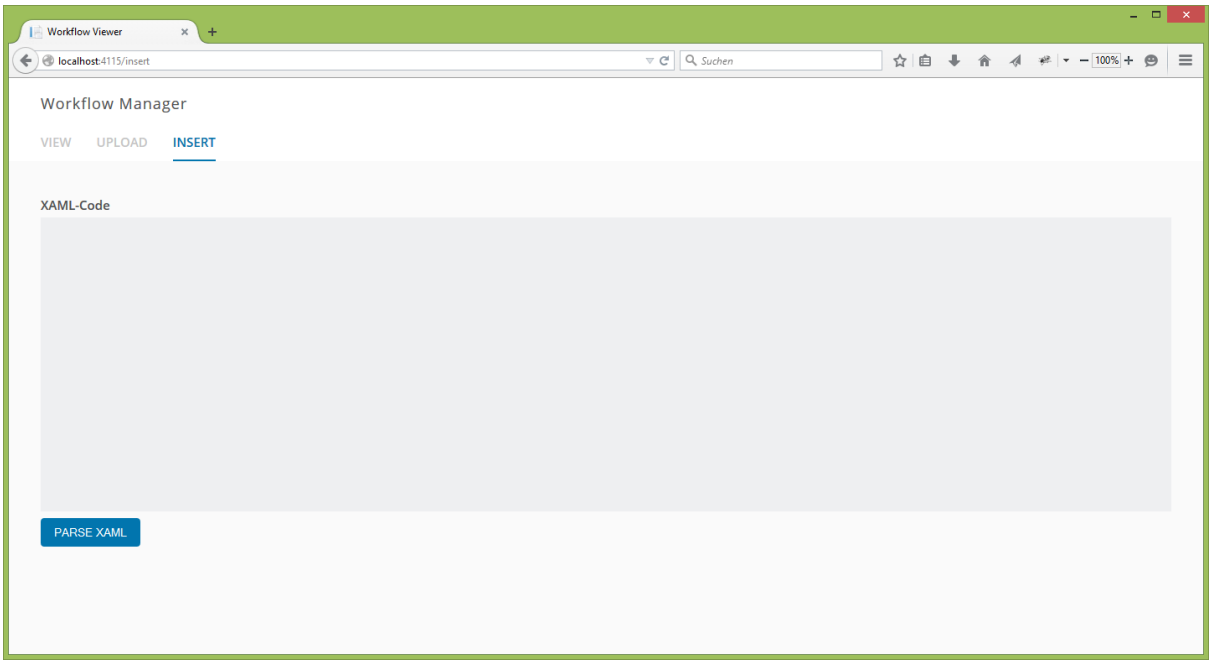
Upload a Workflow from a File

1. Click on the Upload tab (if not already active).
2. Click on “Choose File”. A file chooser dialog shows up.
3. Select a XAML file containing the Workflow you wish to examine.
4. Afterwards the Workflow is being loaded.



Upload a Workflow by Copy and Paste

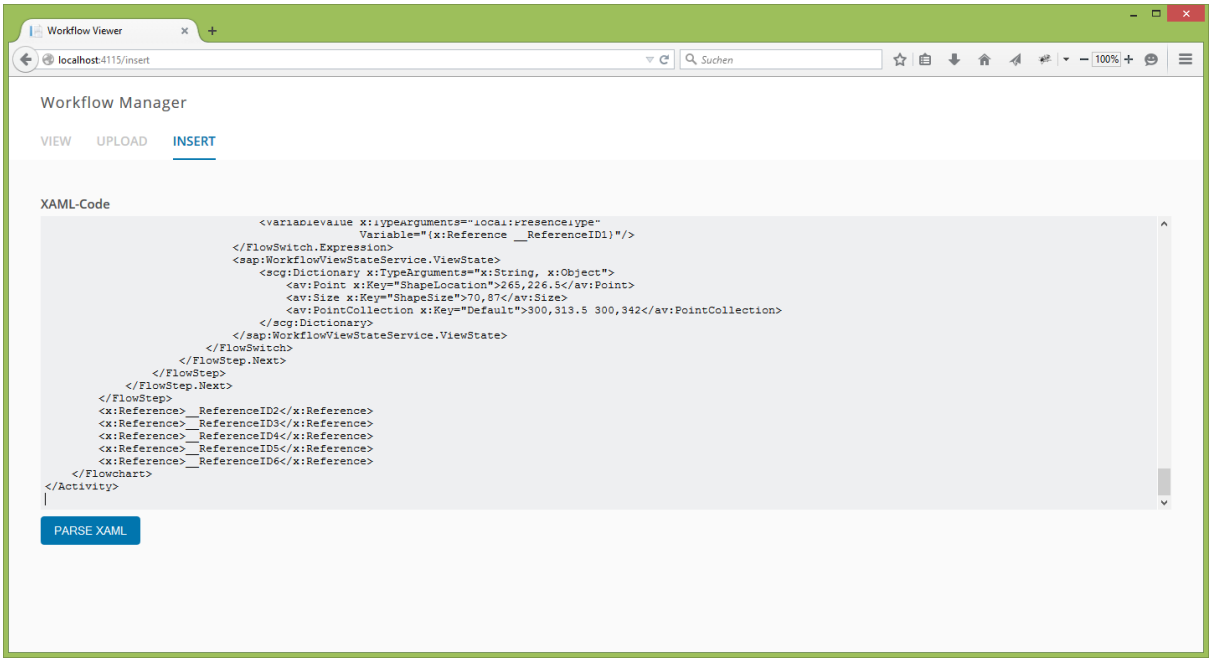
1. Click on the Insert tab (if not already active). The following page shows up:



2. Switch to an opened XAML file containing the Workflow XAML code or open it in a text editor.

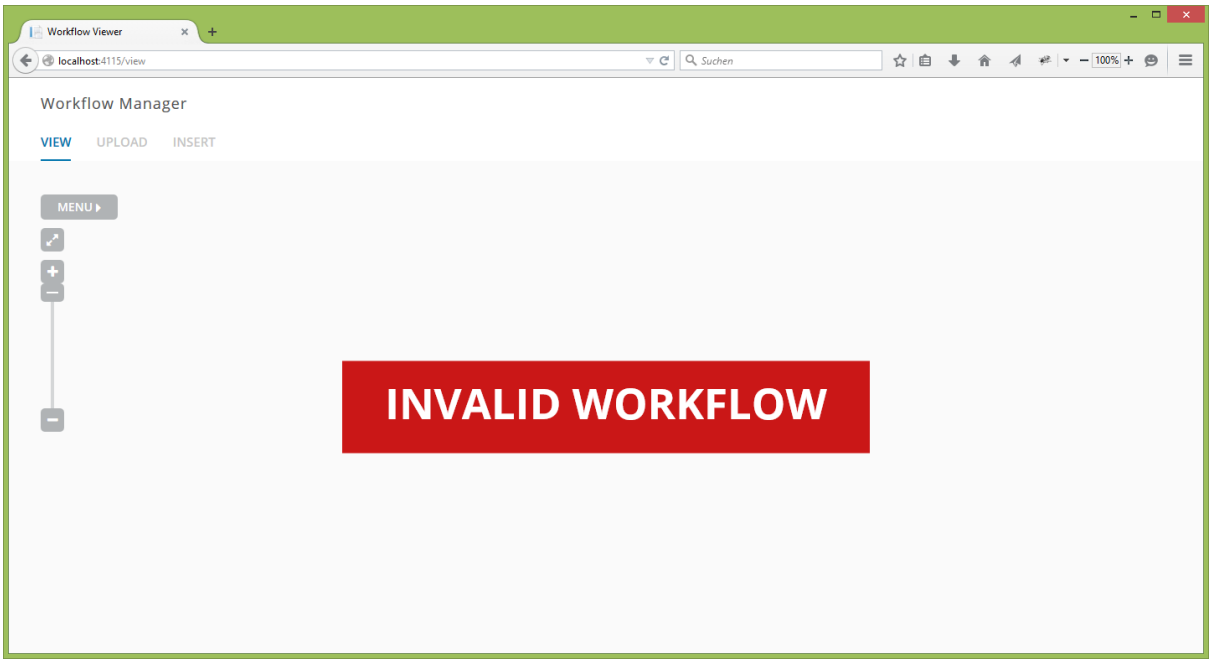
3. Select the XAML code and copy it to the clipboard.

4. Switch back to the Workflow Viewer and paste the code from the clipboard.



5. Click on “Parse XAML” in order to process and display the Workflow.

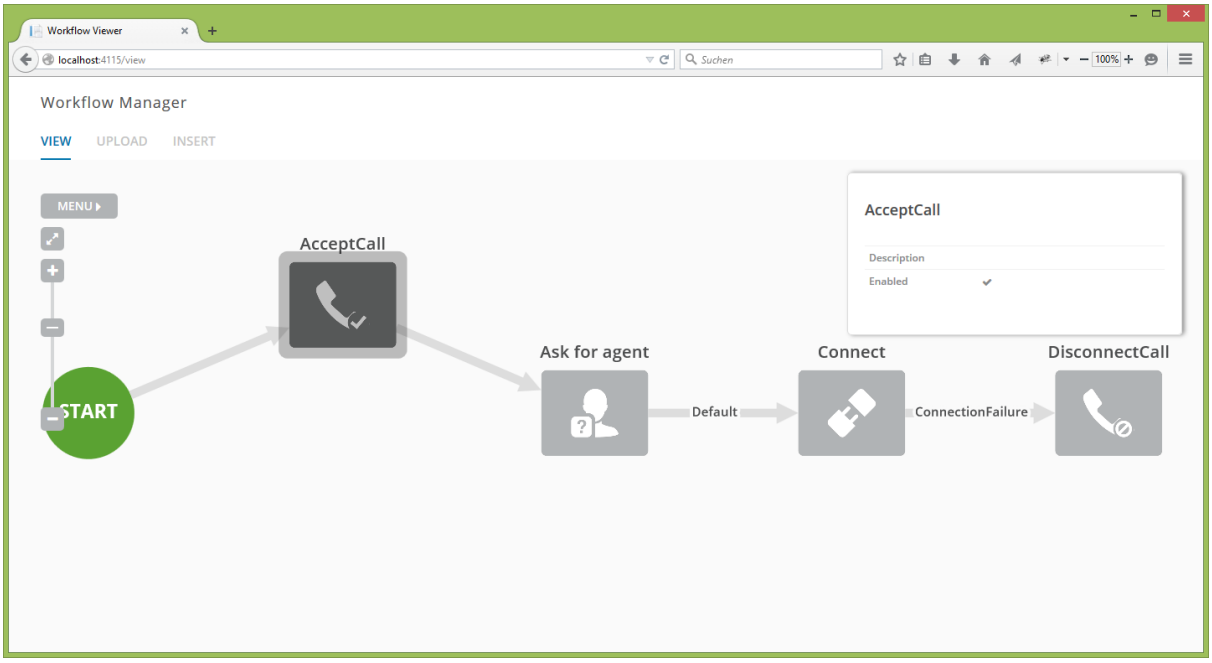
6. If the XAML code is not valid the error message “Invalid Workflow” is shown.



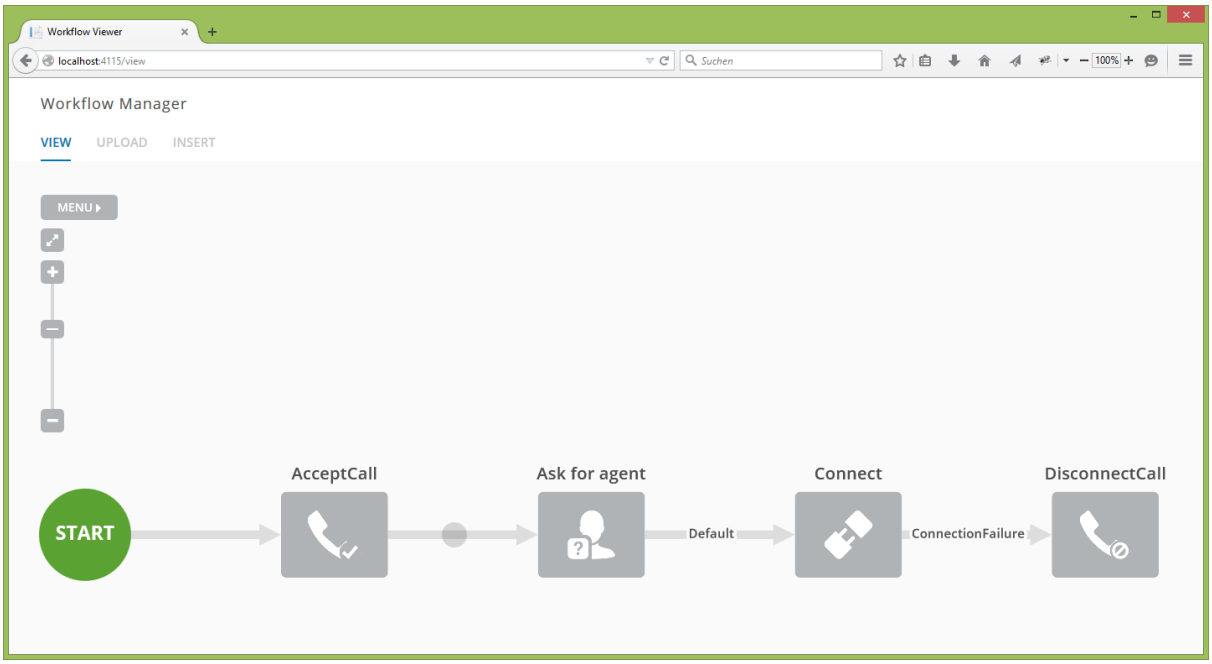
Move Elements around

It is possible to move individual Activities around. The whole Workflow can be moved by clicking and dragging on a different spot.

1. Click on an activity. Its color changes to dark grey.
2. Drag the element.
3. In the Menu click on Reset to return to the original display.



A selected Activity can be moved around. The connections remain.

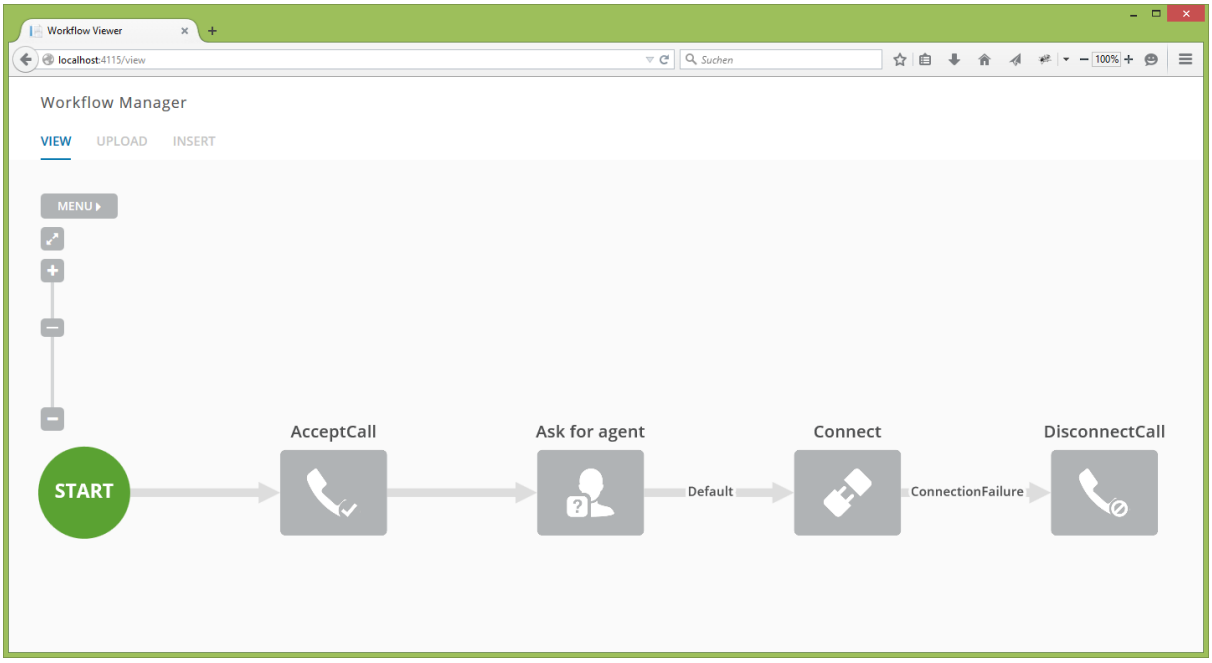


When dragging the whole Workflow a small circle shows the cursor position.

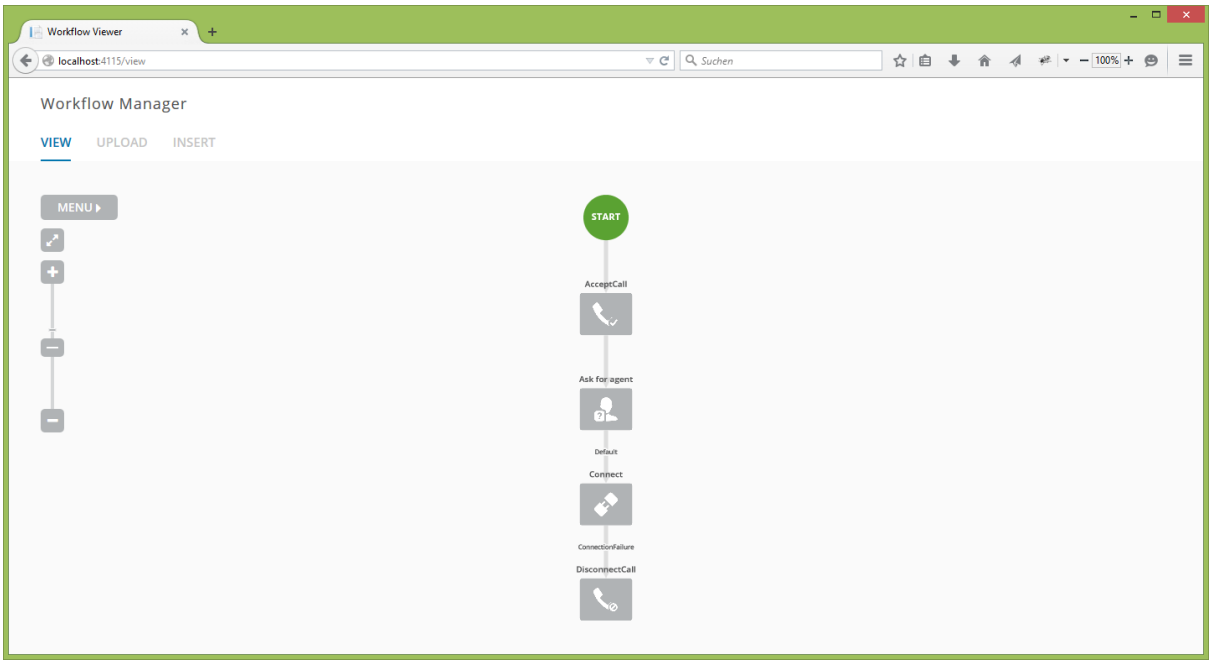
Switch the Workflow Layout

There are two layouts to choose the Workflow direction:

- Left – Right direction
- Top – Bottom direction



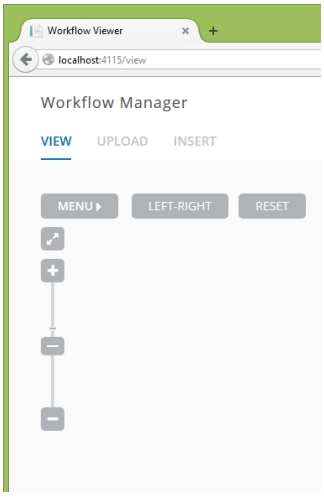
Show Workflow from left to right



Show Workflow from top to bottom

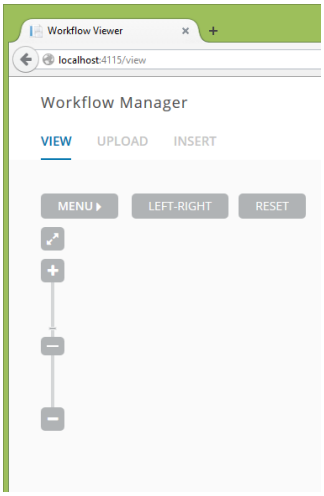
Display Workflow from Top to Bottom

In the Menu click on Top-Bottom. The Workflow Start is at the top, the following Activities are beneath.



Display Workflow from Left to Right

In the Menu click on Left-Right. The Workflow Start is on the left side, the following Activities are on the right.

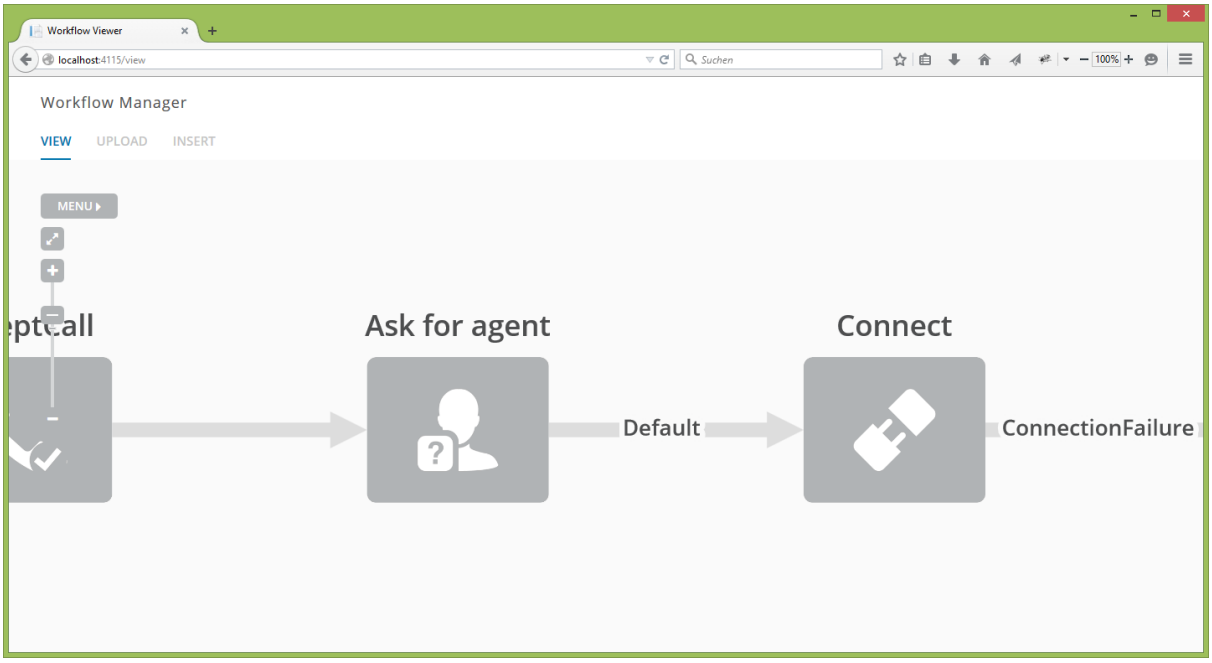


By default the Left-Right Layout is used.

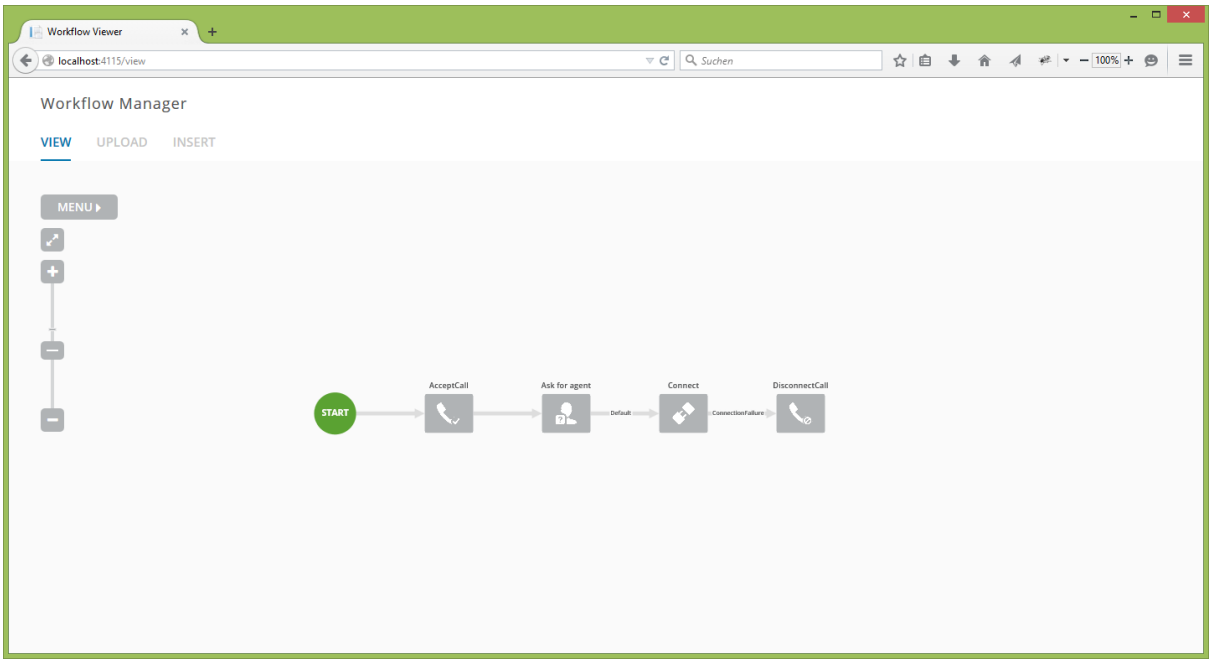
Zoom in and out on the Workflow

To zoom in and out on the workflow the mouse wheel or the slider control on the left can be used.

Click on  above the slider or move the slider up to zoom in.



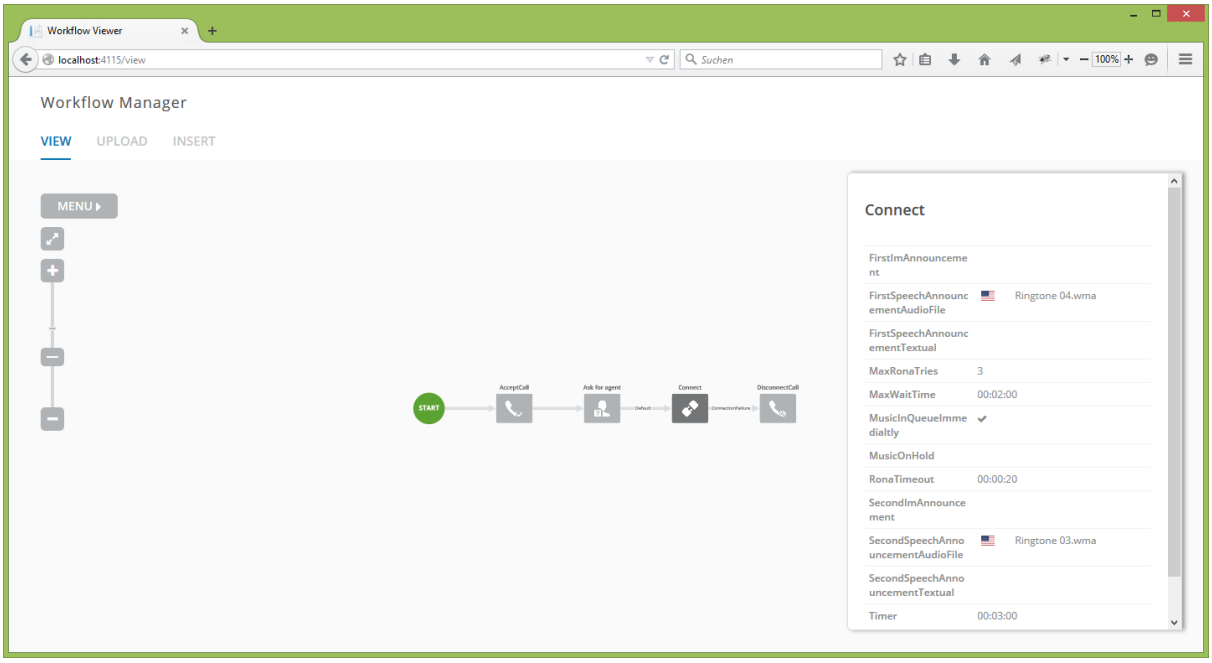
Click on  under the slider or move the slider down to zoom out.



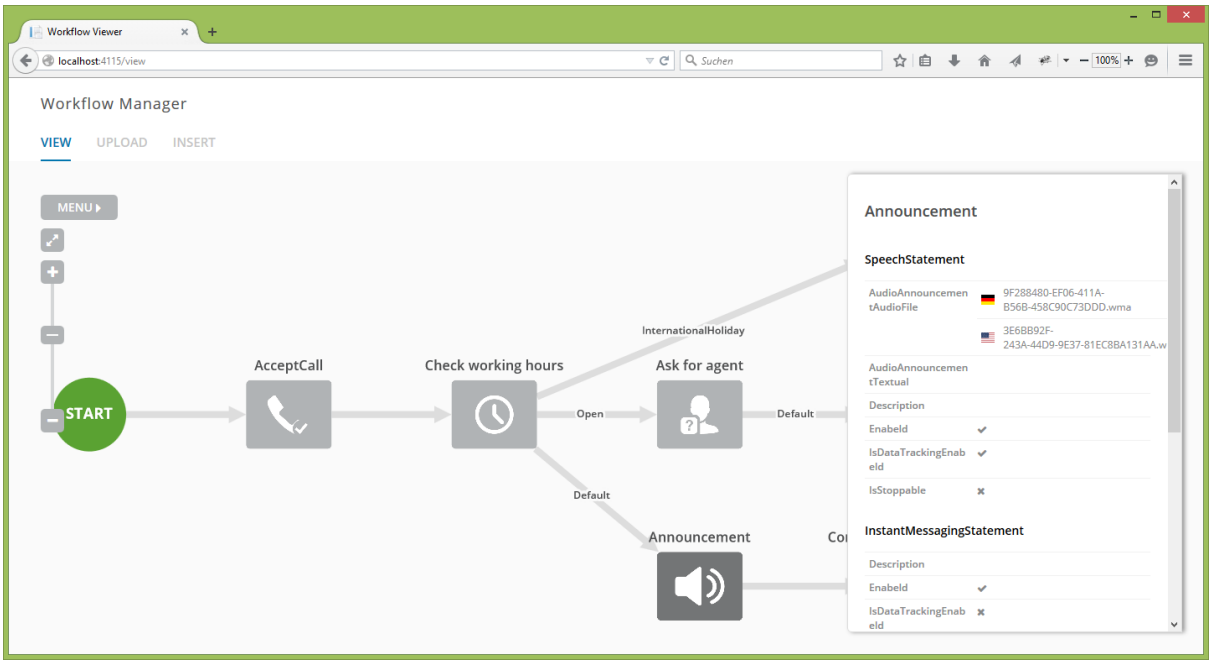
Click on  to restore the initial zoom level.

Display Activity Properties

Select an Activity in the Workflow. A pop-up appears on the right side showing the properties of that Activity.



Properties of an Activity can be grouped in categories as shown in the following example:



10.1.5 Konfigurationsmöglichkeiten

Workflow Stylesheet

Die Workflow Anzeige wird über eine CSS ähnlich Sprache festgelegt. Der Umfang dieser Sprache ist von dem Cytoscape.js Framework vorgegeben. Für den Support einer neuen Activity sollte mindestens ein Icon festgelegt werden. Das Workflow Stylesheet befindet sich in /Content/workflows/styles/flow-chart.css. Die Icons befinden sich im Ordner /Content/workflows/icons/png.

Beispiel für das festlegen eines Icons:

```
.AcceptCall {
   background-image: /Content/workflows/icons/png/acceptCall.png;
}
```

Parser Konfiguration

Einleitung

Der XamlToJsonParser kann in gewissen Bereichen konfiguriert werden, indem beim Erzeugen des Parsers ein XamlToJsonParserConfig Objekt mitgegeben wird. Dieses Config Objekt kann auch mithilfe eines passenden JSON-Dokuments (siehe unten) erzeugt werden. Im Folgenden werden die konfigurierbaren Parameter beschrieben.

Workflow Darstellung

InspectionType:

Bestimmt die Ausgabeform von dem Workflow.

- **Flowchart**: Stellt den Workflow als gerichteten Graph dar und filtert jene Activities heraus, die nicht dargestellt werden sollen.
- **InspectionTree**: Zeigt alle Activities in einem Baum dar, welcher durch die WorkflowInspectionServices (System.Activities) festgelegt wird. Wird nur für Test- und Prüfzwecke verwendet.

ShowBranchNodes:

Legt fest, ob Verzweigungen (If/Switch) als separate Activities angezeigt werden sollen, oder ob die Ausgänge direkt mit der vorangehenden Activity verbunden sind.

- **true**: Verzweigungen werden mit einem separaten Node dargestellt.
- **false**: Die Ausgänge von Verzweigungen werden mit der vorangehenden Activity verbunden.

MapInputCustomerResults:

Legt fest, ob die ExpectedInput Properties von InputCustomer auf die Ausgänge gemappt werden sollen.

- **true**: Die Ausgänge werden gemappt (z.B. DTMF: 1 / IM: 1)
- **false**: Die Ausgänge werden nicht gemappt (Anzeige: Result1, Result2, …)

Activity Verarbeitung

Das .net Framework liefert die Activities von einem Workflow in einer Baumstruktur, welche unbahängig von dem Activity Typen aufgebaut wird. Durch festlegen von ActivityProcessTypes, können die Activities in einer sinnvollen Art angeordnet werden.

DefaultActivityProcessType:

Bestimmt den Standard Verarbeitungstyp (siehe ActivityProcessTypes) für Activities, der angewendet wird, wenn kein anderer Verarbeitungstyp festgelegt ist.

ActivityProcessTypes:

Ordnet einer Activity (String) einen Verarbeitungstyp (Enum) zu.

- **Branch**: Die Activity wird angezeigt, falls ShowBranchNodes=true. Die direkten Kinder-Activites werden als Verzweigung dargestellt.
- **Sequence**: Die Activity wird angezeigt. Die direkten Kinder-Activites werden aneinandergereiht und der letzten sichtbaren Activity angehängt..
- **Flowchart**: Die Activities innerhalb des Flowcharts werden gemäss der in der Flowchart-Activity definierten Abfolge dargestellt.
- **HideInternals**: Die Activity wird angezeigt, jedoch werden alle Kinder-Activites versteckt.
- **Skip**: Die Activity wird versteckt. Die direkten Kinder-Activities werden aneinandergereiht und der letzten sichtbaren Activity angehängt.

Properties Filter

PropertiesFilter:

Eine Liste von Property-Namen, die dem Client nicht angezeigt werden sollen.

SubPropertiesGroupFilter:

Eine Liste von Property-Gruppen, die nicht angezeigt werden sollen, wenn die Properties von einer Sub-Activity stammen. Wenn beispielsweise "DisconnectCall" angegeben wird, dann werden bei Klick auf eine DisconnectCall-Activity zwar die Activities angezeigt. Wird jedoch auf eine Activity geklickt, die DisconnectCall als eine interne Activity besitzt, werden diese Properties ausgeblendet.

JSON

Das JSON-Konfigurationsdokument hat folgenden Aufbau (mit den Standardeinstellungen):

```
{
  "InspectionType": "Flowchart",
  "ShowBranchNodes": false,
  "MapInputCustomerResults": true,
  "DefaultActivityProcessType": "HideInternals",
  "ActivityProcessTypes": {
    "Flowchart": "Flowchart",
    "If": "Branch",
    "Parallel": "Branch",
    "Switch": "Branch",
    "DynamicActivity": "Skip",
    "Sequence": "Skip"
  },
  "PropertiesFilter": [
    "Activities",
    "ActivityId",
    "Body",
    "Cases",
    "DisplayName",
    "Id",
    "LocationReference",
    "Result",
    "ResultType",
    "Value",
    "Variable",
    "Variables"
  ],
  "SubPropertiesGroupFilter": [
    "DisconnectCall",
    "Switch"
  ]
}
```

Aus einem solchen JSON String kann ein Config Objekt erstellt werden durch Aufruf von XamlToJson-ParserConfig.FromJson.

Weiterführende Konfigurationsmöglichkeiten

Activity Verarbeitung

In WFWebviewer.Core.Processing.ActivityProcessor werden aktuell fünf Methoden unterschieden, wie eine Activity je nach Typ im ausgegebenen Workflow angeordnet werden soll. Diese fünf Methoden sollten für die meisten neue Activity-Typen wiederverwendbar sein, je nach Anforderungen muss hier eine neue Methode implementiert werden.

Multilanguage String Erkennung

In WFWebviewer.Core.Processing.PropertyProcessor ist die Methode IsMultilanguageString vorhanden, die anhand eines PropertyInfo Objektes entscheidet, ob es sich dabei um einen Multilanguage String handelt. Diese Methode muss gegebenenfalls angepasst werden.

10.2 Persönliche Berichte

10.2.1 Martin Eisenhammer

Als wir das Thema zugewiesen bekommen habe ich mich gefreut, dass wir ein Thema im .NET Bereich bearbeiten dürfen. Ich war motiviert so ein Thema zu bearbeiten.

Das erste Treffen mit Luware war super und ich habe neue Menschen kennen gelernt. Beat Stettler kannte ich vor dem Treffen nicht, da ich in Deutschland schon die Computernetze-Module abgeschlossen habe und diese an der HSR angerechnet wurden.

Die Vorgehensweise in der Entwicklung nach SCRUM hat mir sehr gut gefallen. Vorher hatte ich von SCRUM nur in der Software Engineering 2 Vorlesung in Deutschland gehört und etwas im Internet darüber gelesen. Ich finde die Entwicklung nach SCRUM interessant. Man arbeitet mit dem Auftraggeber zusammen und man hat zu Beginn eines Arbeitstags ein Standup-Meeting, wo man berichtet was man vorher gemacht hat, was man im Laufe des Tages machen wird und wo es Probleme gibt. So etwas Ähnliches habe ich schon im Praktikum vor dem Weiterstudium erlebt, wobei es dort für mich eher mehr ein Code-Review war.

Die User Stories waren für mich klar formuliert. Ich konnte jede meiner Aufgaben, die ich in einem zweiwöchigen Sprint hatte, durchführen und beenden, wobei ich mich in der Zeitschätzung öfters verschätzt habe. Ich kann ja nicht vorher wissen, ob ich die Aufgabe in der von mir geschätzten Zeit tatsächlich schaffe.

Die Sprint Demos und Sprint Plannings fanden aller 2 Wochen an einem Montag statt, sodass man am Wochenende etwas fertigstellen konnte.

Ich konnte in diesem Projekt gute Erfahrungen mit SCRUM machen und habe diesen Prozess in dieser Form zum ersten Mal erlebt.

Die Zusammenarbeit mit Luware hat mir gefallen. Man konnte im Standup-Meeting (per Skype) und per E-Mail Fragen stellen. Die Kontaktpersonen von Luware waren nett. Eine Antwort kam per E-Mail rechtzeitig und nicht zu spät. Ich bekam auch die E-Mails, die Josua Stähli geschrieben hat. So war ich immer auf dem aktuellen Stand.

Die Zusammenarbeit mit Josua Stähli hat mir sehr gut gefallen. Er arbeitete mit mir in dem Arbeitsraum der HSR. Am Freitag war Josua Stähli nur am Vormittag an der HSR. Am Nachmittag habe ich alleine gearbeitet. Ich konnte ihn jederzeit etwas fragen, wenn Probleme aufgetreten sind.

Ich habe schon mit Josua Stähli in den Modulen Internettechnologien und Microsoft-Technologien jeweils ein Miniprojekt bearbeitet.

Ich habe im Projekt einiges gelernt. Ich habe neue Technologien, wie Cytoscape.JS oder ASP.NET MVC kennengelernt. Dazu halfen mir die Kenntnisse aus den Modulen Internettechnologien und Microsoft-Technologien weiter.

Die Implementierung hat mir auch sehr viel Spass gemacht. Ich hatte im Laufe des Projekts wenig Probleme damit. Ich hatte schon vor dem Projekt Erfahrungen mit C# und .NET. Jedoch bei der Ausarbeitung der Architektur konnte ich mir am Anfang nicht ganz vorstellen, wie die Projektstruktur aufgebaut sein muss.

10.2.2 Josua Stähli

Das Thema “WF Webviewer” hat mich angesprochen, weil ich schon Vorwissen in C# und im Bereich der Webentwicklung hatte und mich diese Gebiete interessieren. Die Vorgabe des Themas durch eine Firma, der Luware AG in Zürich, versprach zudem, dass es sich um ein praxisorientiertes Projekt handelt. Vor Beginn der Arbeit bestanden bei mir noch einige Unklarheiten bezüglich der Aufgabenstellung. Das lag wahrscheinlich vor allem daran, weil wir vorher nur wenige Anhaltspunkte hatten, wie die Applikationen von Luware, auf denen das Projekt aufbaut, aussahen. Nach dem ersten Treffen bei Luware konnten die offenen Fragen geklärt werden und es zeigte sich mir, dass dies eine spannende und herausfordernde Arbeit werden würde.

Als Vorgehensweise bei der Entwicklung haben wir uns für Scrum entschieden, welche uns auch von Luware empfohlen wurde. Die User Stories wurden bereits von Luware vorgegeben, was mir den Einstieg in die Arbeit sehr erleichtert hat. Dadurch konnte ich mich viel stärker auf die Detailplanung und die eigentliche Umsetzung der User Stories konzentrieren.

Die Zusammenarbeit mit meinem Projektpartner Martin Eisenhammer habe ich als sehr angenehm empfunden. Wir haben in anderen Modulen bereits ein C#- und ein Web-Projekt zusammen umgesetzt, wodurch wir schon ein eingespieltes Team waren. Die anstehenden Tasks hatten wir jeweils schnell untereinander aufgeteilt, bei Problemen während der Arbeit konnten wir uns gegenseitig unterstützen.

Wir hatten zwei Mal wöchentlich ein sogenanntes Stand-up Meeting mit Luware per Skype durchgeführt, wo wir über den aktuellen Stand berichteten und bei Unklarheiten Fragen stellen konnten. Auch per E-Mail bekamen wir bei Problemen jederzeit Hilfe von den Kontaktpersonen bei Luware. Dadurch haben uns Schwierigkeiten im Verlaufe des Projektes nie lange aufgehalten. Insgesamt kann ich sagen, dass ich die enge Zusammenarbeit und die kompetente Unterstützung durch Luware sehr schätzte.

Ich sehe diese Semesterarbeit als eine grosse Bereicherung an, weil ich mich intensiv mit verbreitet eingesetzten Technologien wie ASP.NET (MVC) und AngularJS auseinandersetzen konnte, die ich vorher noch nicht kannte. Die intensive Betreuung des Projekts durch Prof. Beat Stettler und die Luware AG machten diese herausfordernde Arbeit zu einer angenehmen und lehrreichen Erfahrung.

10.3 Quellenverzeichnis

[AngularJS] AngluarJS: https://angularjs.org/ 06.03.2015

[jQuery] jQuery: http://jquery.com/ 06.03.2015

[JsonNET] Json.NET: Feature Comparison / Performance Comparison http://www.newtonsoft.com/json 24.05.2015

[MSDN1] MSDN - Übersicht über ASP.NET MVC: https://msdn.microsoft.com/de-de/library/dd381412(v=vs.108).aspx 02.03.2015

[MSDN2] MSDN: Übersicht über Windows-Workflows https://msdn.microsoft.com/de-de/library/dd489465(v=vs.110).aspx 02.03.2015

[MSDN3] MSDN: Grundlegende Konzepte über Windows-Workflows https://msdn.microsoft.com/de-de/library/dd489408(v=vs.110).aspx 02.03.2015

[MSDN4] MSDN: Architektur von Windows-Workflows https://msdn.microsoft.com/de-de/library/dd489413(v=vs.110).aspx 02.03.2015

[RajatSingh] Rajat Singh : An Absolute Beginner's Tutorial on ASP.NET MVC for Web Forms Developers http://www.codeproject.com/Articles/575397/An-Absolute-Beginners-Tutorial-on-ASP-NET-MVC-for 06.03.2015

[SignalR] ASP.NET SignalR: http://signalr.net/ 02.03.2015

[Telerik] Kendo UI von Telerik: http://www.telerik.com/kendo-ui 06.03.2015

[TLDRLegal1] TLDRLegal: MIT License https://tldrlegal.com/license/mit-license 24.05.2015

[TLDRLegal2] TLDRLegal: SIL Open Font License v1.1 https://tldrlegal.com/license/open-font-license-(ofl)-explained 24.05.2015

[w3Schools] jQuery Tutorial: http://www.w3schools.com/jquery/default.asp 06.03.2015