Rem

RemRes Architektur

ResRes Architekturdokument

Christoph Hauer

Version 0.1

FH Wiener Neustadt

BIT11

# Inhalt

[1 Inhalt 1](#_Toc378792001)

[2 Kernanforderungen 2](#_Toc378792002)

[3 Architektur 2](#_Toc378792003)

[4 RemRes API / Protokoll 2](#_Toc378792004)

[4.1 Basis Typen 2](#_Toc378792005)

[4.2 RemRes Message / Nachrichten 3](#_Toc378792006)

[5 RemRes Service 3](#_Toc378792007)

[5.1 Dynamisches ausgewertetes Protokoll 3](#_Toc378792008)

[5.2 Schnittstellen 4](#_Toc378792009)

[6 RemRes Client 4](#_Toc378792010)

[7 Logical View 4](#_Toc378792011)

[7.1 Protokoll 4](#_Toc378792012)

[7.2 Datenspeicherung 4](#_Toc378792013)

[7.3 RemRes Service 5](#_Toc378792014)

[7.4 RemRes Client (Library) 7](#_Toc378792015)

[8 Physical View 8](#_Toc378792016)

[9 Ressourcen 8](#_Toc378792017)

# Kernanforderungen

* Mit RemRes muss es möglich sein Eigenschaften eines Computer remote über ein simples aber erweiterbares Protokoll zu überwachen.
* RemRes soll möglichst einfach alle verschiedenste Szenarien angepasst werden.
* Das Protokoll soll von der Implementierung des Services weitestgehend getrennt werden.
* Das Protokoll soll mittels TCP/IP und XML realisiert werden und leicht anpassbar sein.

# Architektur

Das System RemRes wird logisch in zwei Subsysteme untergliedert: RemRes Service und RemRes Client. Diese Subsysteme werden über das RemRes XML Protokoll miteinander verbunden. Dieses Protokoll ist in einer RemRes API in Objekten gekapselt.

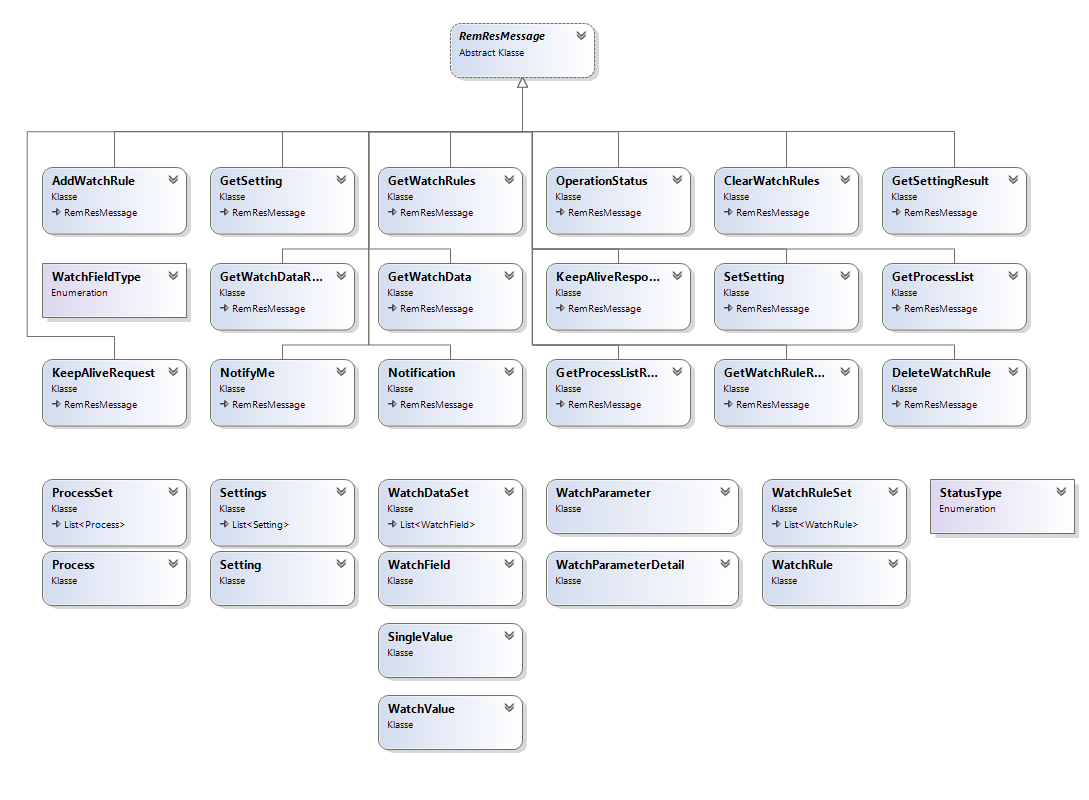


# RemRes API / Protokoll

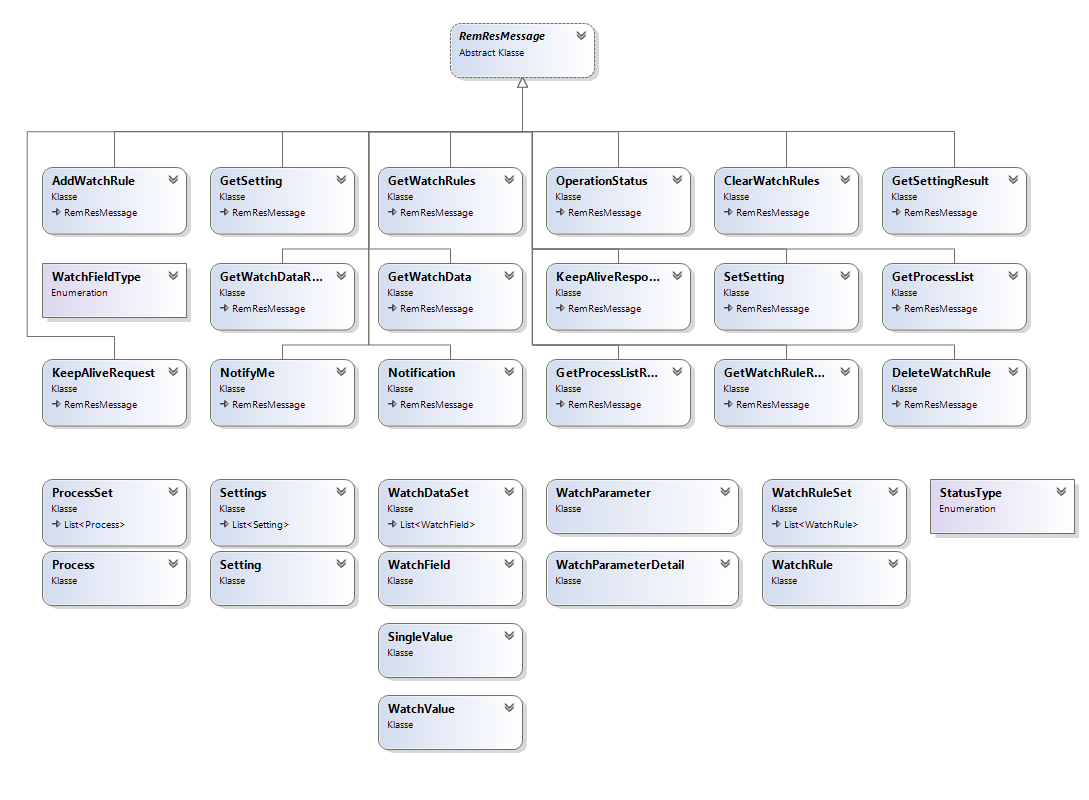
Das RemRes Protokoll wird mittels XML realisiert. Diese XML Nachrichten werden mittels Serialisierung aus .NET Objekten erzeugt und können daher einfach in einer Library bzw. API gekapselt werden.

Dieses API setzt sich aus zwei Teilen zusammen. Zum einen den Messages und zum anderen den Basis Typen. Die Messages oder Nachrichten leiten alle von einer abstrakten Basisklasse RemRes Message ab und sind je nach Aufgabe implementiert. Die Basis Typen werden in allen Nachrichten immer wieder verwendet und sind nicht von RemRes Message abgeleitet. Die Basis Typen können nicht als Message über das Netzwerk verschickt werden.

## Basis Typen



## RemRes Message / Nachrichten



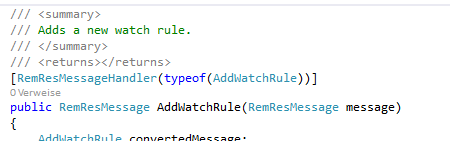
Für eine Detaillierte Erklärung der XML Messages verweise ich hier auf das Dokument **RemRes Protokoll.docx**.

# RemRes Service

Der RemRes Service ist grundlegend in 3 große Teile geteilt die über definierte Schnittstellen miteinander interagieren.

* Network Connect System
  + Zugriff auf das Netzwerk.
  + Behandlung von eingehenden Nachrichten.
  + Verwaltung der offenen Verbindungen.
  + Umwandlung von Text (XML) Messages in Objekte und zur Verfügung Stellung für andere Teilsysteme.
* Execution System
  + Finden der Ausführungspläne bzw. Methoden für die jeweilige Nachricht im dafür vorgesehenen System. (Schnittstelle zum WatchSystem)
* Watch System
  + Überwachung der Computereigenschaften basierend auf den über das Protokoll mitgeteilten Regeln zur Überwachung.
  + Generierung von Notifikation Nachrichten.
  + Bereitstellung der Ausführungspläne bzw. Methoden für die jeweiligen Nachrichten.

## Dynamisches ausgewertetes Protokoll

Die Protokoll Messages werden beim empfangen im Service durch die im Network System konfigurierte Methode zu Objekten umgewandelt. Diese Nachrichten Objekte werden an das Execution System übergeben. Das Execution System sucht anhand des Types der Message mittels Reflection im Watch System nach einer geeigneten Methode die mittels Attribut „RemResMessageHandler“ auf den Nachrichten Typ gebunden ist.

## Schnittstellen



# RemRes Client

In diesem Dokument wird mittels RemRes Client der RemRes Test Client gleichgesetzt. Der RemRes Test Client bietet im Vergleich zu einem vollständigen Client weniger Funktionen.

Der RemRes TestClient soll die möglichkeit bieten jede Message (bis auf spezielle) an den Service übermitteln zu können und so seine Funktionsfähigkeit zu überprüfen.

# Logical View

## Protokoll

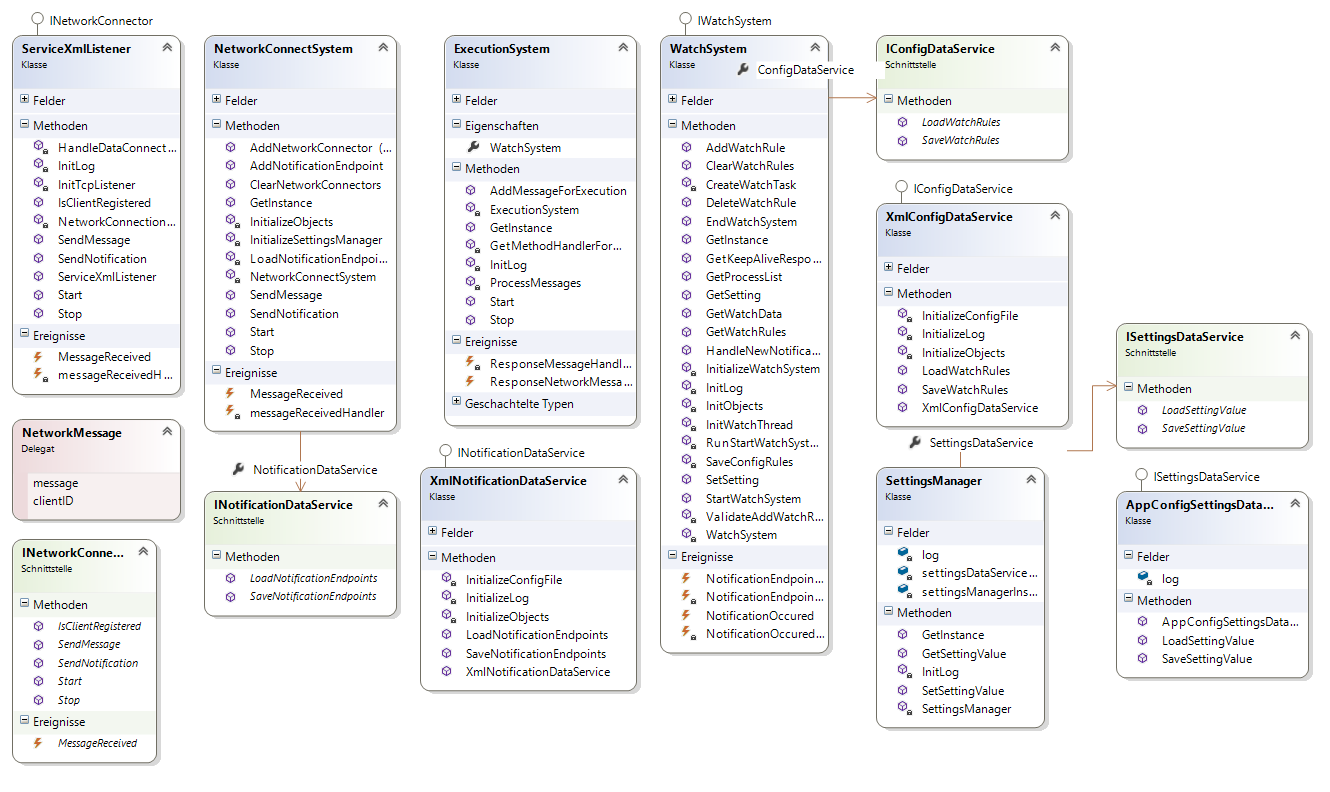
Für eine Detaillierte Erklärung der XML Messages verweise ich hier auf das Dokument **RemRes Protokoll.docx**.

## Datenspeicherung

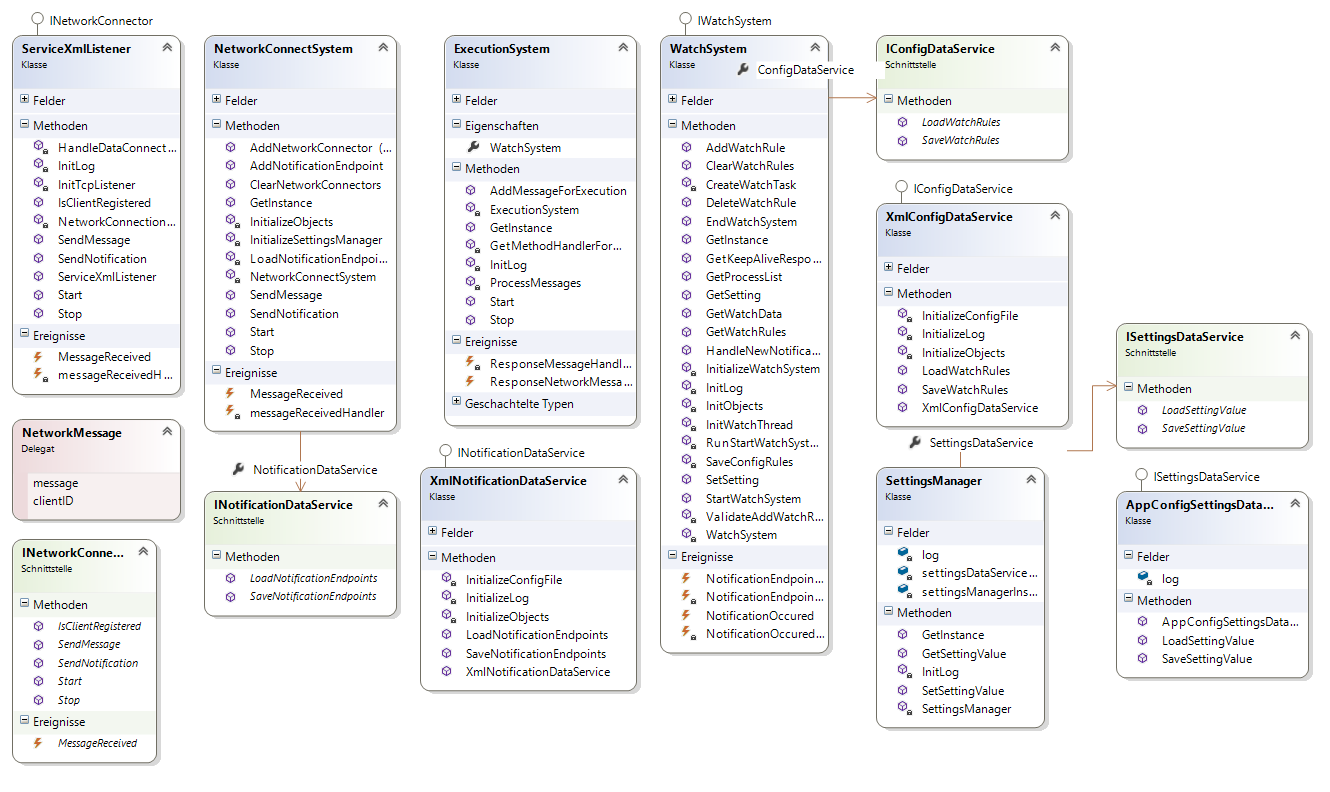
Die Datenspeicherung erfolgt über das ganze Projekt mittels XML Files. Diese Datenzugriffsklassen sind mittels Dependency Injection implementiert und können dadurch durch beliebige andere Datenspeicherung (auf zum Beispiel eine Datenbank ausgetauscht werden).

## RemRes Service

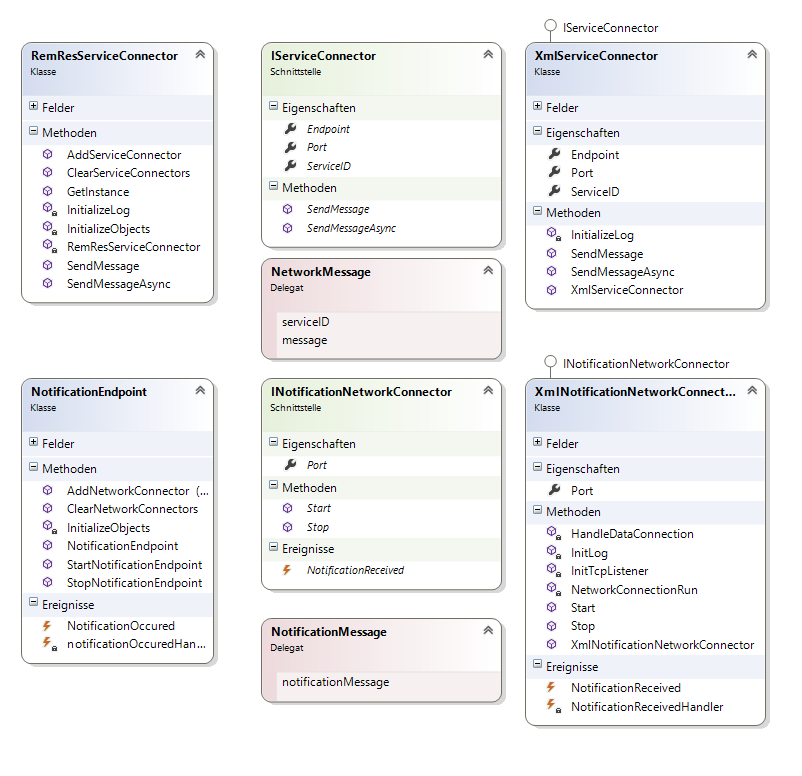
**Teil 1**



**Teil 2**



## RemRes Client (Library)



# Physical View



# Ressourcen

Hardware

* PC
* Notebook
* Windows Server

Betriebssysteme:

* Windows 7
* Windows 8
* Windows 8.1

Codierung

* Visual Studio 2013

Dokumentation

* Microsoft Office 2013