# Überblick über die Softwarekomponenten

Die Software ist prinzipiell in drei Teile unterteilt:

1. Kompetenz Funktionalität (Library: CBKST)
2. Datenbank Handling (Library: storagedb)
3. Webzugang (ASP.NET project: competenceservice)

Die Komponenten A und B sind noch in der Library competenceframework (D) zusammengefasst um eine modulare Verwendung zu gewährleisten. Es befinden sich auch die Test-Projekte Test (E), für schnelles variables testen und UnitTestCompetenceservice (F), für ein automatisches regelmäßiges testen im Solution-Verzeichnis.

# A. Kompetenzfunktionalität

Es werden die folgenden zwei Methoden zur Verfügung gestellt:

* CompetenceProbabilities updateCompetenceState(DomainModel domainModel, CompetenceProbabilities competenceProbabilities, EvidenceSet evidenceSet)
* CompetenceProbabilities createInitialCompetenceProbabilities(DomainModel dm)

Der Datentyp CompetenceProbabilities kann etwa durch die statische Methode getCPFromXmlString(string xml) aus einem xml gewonnen werden (wird in der Datenbank-Tabelle competencestates gespeichert). Der Datentyp EvidenceSet kann etwa durch die statische Methode getESFromXmlString(string xml) aus einem XML erzeugt werden. Dieses XML hat die folgende Form (je nach type werden für evidence andere Tags benötigt: EvidenceType.Gamesituation benötigt nur noch den Tag gamesituation, EvidenceType.Activity benötigt nur noch den Tag activity, EvidenceType.Competence benötigt nur noch die Tags competecneid, direction und power):

<evidenceset>

<evidence>

<type>ENUM: EvidenceType in (Competence,Activity,Gamesituation)</type>

<gamesituation>string</gamesituation>

<activity>string</activity>

<competenceid>string</competenceid>

<direction>bool</direction>

<power>ENUM: EvidencePower in (Low,Medium,High)</power>

</evidence>

<evidence>

<type>ENUM: EvidenceType in (Competence,Activity,Gamesituation)</type>

<gamesituation>string</gamesituation>

<activity>string</activity>

<competenceid>string</competenceid>

<direction>bool</direction>

<power>ENUM: EvidencePower in (Low,Medium,High)</power>

</evidence>…

</evidenceset>

# B. Datenbank

Die Einstellungen der Datenbank lassen sich im file DatabaseHandler.cs, in der #region Parameters ändern, die Standardeinstellungen sind: server=localhost, database=competencedb, uid=root, password=rage. Weitere Einstellungen sind die erlaubten Zeichen der IDs (domainmodelid/trackingid) und die Länge der IDs.

Zum Speichern der Daten werden die folgenden Tabellen angelegt:

* domainmodels (id, structure)
* competencestates (id, competencestate, datetime) [speicher die xml representation vom Datentyp CompetenceProbabilities]
* trackingids (trackingid, domainmodelid, competencestateid)
* users (userid, password)
* compdev\_{tid} (id, competencestate, datetime)

Die unterstrichene Größe stellt den Primary Key dar, bei TEXT sind es die ersten 50 Zeichen. Alle Größen sind TEXT, bis auf compdev\_{tid}:id, competencestates:id und trackingids: competencestateid. Es werden die folgenden Testdaten automatisch befüllt:

* Ein Beispiel Domainmodel in die Tabelle domainmodels mit id 1
* Ein User in die Tabelle users mit ID rage und Passwort rage

# C. Webzugang

Es sind derzeit die folgenden REST-Requests implementiert:

* <URL:PORT/rest/competenceservice/storedm> (liefert eine Domainmodel Id [dmid])
* [URL:PORT/rest/competenceservice/getdm/{dmid}](URL:PORT/rest/competenceservice/getdm/%7bdmid%7d) (liefert Domainmodel DM zurück)
* [URL:PORT/rest/competenceservice/deletedm/{dmid}](URL:PORT/rest/competenceservice/deletedm/%7bdmid%7d) (löscht DM und darauf bezogene daten, wie etwa tracking ids)
* [URL:PORT/rest/competenceservice/createtrackingid/{dmid}](URL:PORT/rest/competenceservice/createtrackingid/%7bdmid%7d) (liefert eine tracking id [tid] zueiner DM Id)
* [URL:PORT/rest/competenceservice/updatecompetencestate/{tid}](URL:PORT/rest/competenceservice/updatecompetencestate/%7btid%7d) (führt einen competence update bei einer tid durch)
* [URL:PORT/rest/competenceservice/deletetrackingid/{tid}](URL:PORT/rest/competenceservice/deletetrackingid/%7btid%7d) (löscht eine tid und darauf bezogene daten, wie den Kompetenzzustand)
* [URL:PORT/rest/competenceservice/getcompetencestate/{tid}](URL:PORT/rest/competenceservice/getcompetencestate/%7btid%7d) (liefert den Kompetenzzustand zu einer tid)

Jeder Zugriff wird mittels Basic http Authentifizierung überprüft. Dazu dienen die Einträge der User Tabelle in der Datenbank. Der automatische angelegte Eintrag ‚rage,rage‘ kann für testzwecke verwendet werden.

Für Testzwecke kann der Amazon-Server verwendet werden: <http://ec2-54-149-218-203.us-west-2.compute.amazonaws.com> mit port 80.