Projekt IGH DataExpert

Spezifikationen Voraussetzungen

Datum : 28.01.2015
Version : 2.0.1.0

Inhaltsverzeichnis

1 Einleitung

Dieses Dokument stellt die technische Implementations-Spezifikation für das Projekt DataExpert dar.

Mit Anbieter oder Server bezeichnen wir den Händler oder Hersteller an den Bestellungen oder Katalogabfragen übermittelt werden. Mit Kunde, Installateur oder Client bezeichnen wir den Besteller respektive Anfragesteller.

2 Architektur

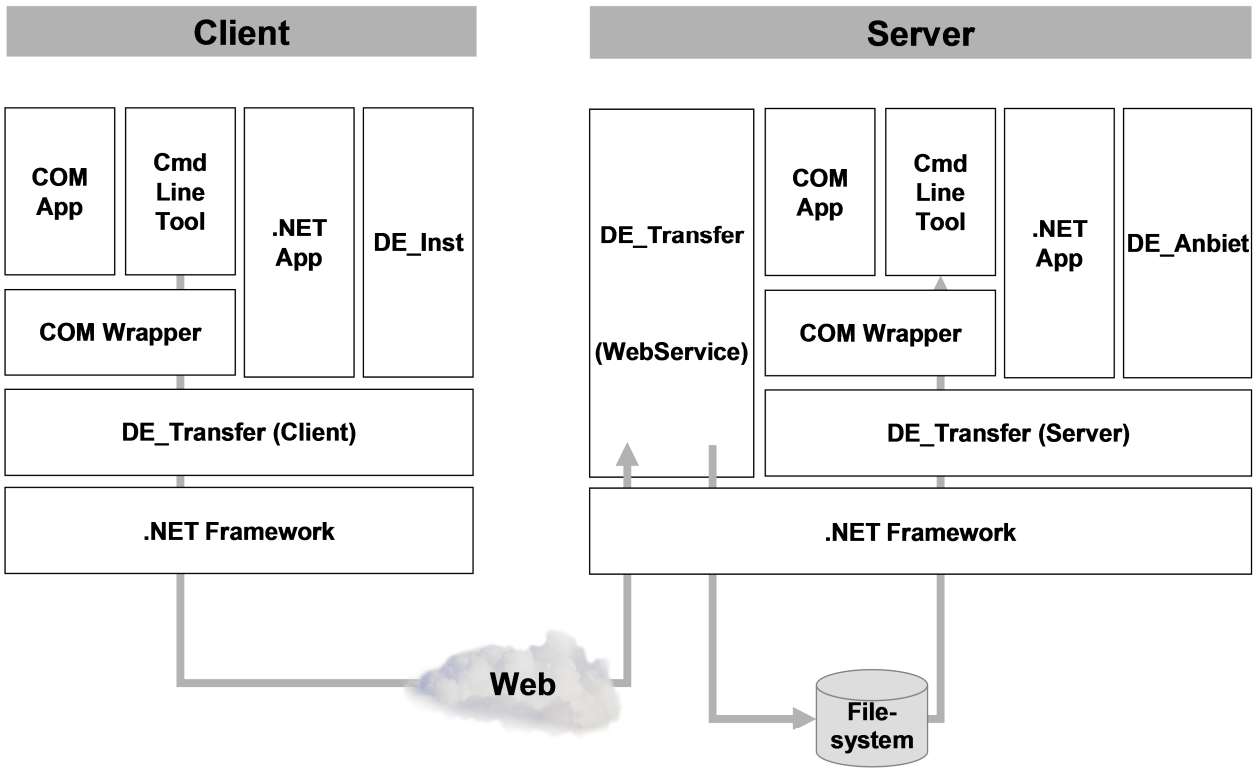

Abb: Komponentenarchitektur

Version DataExpert 1.5 (alt)

- DataExpert besteht aus den drei Hauptmodulen
  - o DE_Transfer : Kommunikation
  - o DE_Inst : Konfiguration Client
  - o DE_Anbiet : Konfiguration Server
- DE_Transfer wiederum besteht aus den 3 Modulen
  - o Client : Kommunikation zwischen Client und WebService, Haltung Konfigurationsdaten
  - o WebService : Kommunikation mit Clients
  - o Server : Zugriff auf Client-Daten, Haltung Konfigurationsdaten
- COM Applikationen greifen Clientseitig über einen COM Wrapper auf DE_Transfer zu
- .NET Applikationen können ohne COM Wrapper direkt auf DE_Transfer zugreifen

- Die Konfigurationswerkzeuge DE_Inst und DE_Anbiet greifen via DE_Transfer auf die Konfigurationsdaten zu (Beispiel für Zugriff einer .NET Applikation auf DE_Transfer)

Version DataExpert 2.0 (direkter Aufruf via WebService)

- WebService Methoden
- DE_Anbiet für Server Konfiguration

2.1 Grundsätze zur Architektur

Die Architektur von DataExpert befolgt folgende Grundsätze:

1. **Pöstler:** DataExpert ist ausschliesslich für die Übermittlung zuständig und nimmt keinen Einblick in die zu übertragenen Daten. Analog zu einem "Pöstler", der nur einen Brief überbringt und zur Erfüllung seiner Aufgabe weder den Inhalt des Briefes kennt, noch kennen soll.
2. **Kein zentraler Server:** DataExpert kennt keinen zentralen Server oder Clearing-Stelle. Kunde und Anbieter kommunizieren direkt und unabhängig von anderen an DataExpert beteiligten Parteien. Grundsätzlich darf kein Informationsmonopol entstehen (wie zum Beispiel bei einem Marktplatz).
3. Unsichtbar**:** DataExpert stellt eine B2B Kommunikations-Funktionalität für bestehende und neue Applikationen zur Verfügung. DataExpert selbst kommuniziert nicht direkt mit den Anwendern. Einzige Ausnahme sind einfachste Konfigurations- und Testwerkzeuge.
4. **Pull Prinzip:** Der Kunde ist zuständig für den Verbindungsaufbau, d.h. der Kunde meldet sich beim Anbieter an und tauscht die für ihn relevanten Informationen mit dem Anbieter aus. Der Anbieter meldet sich nie beim Kunden an.
5. **Authentisierung mit Windows Logon:** Die bereits bestehende Windows-Sicherheits-Infrastruktur wird durch DataExpert wieder verwendet. Kundenseitig erfolgen alle Authentisierungen durch das Windows Logon. Abhängig vom Windows Logon können unterschiedliche Rechte vergeben werden (Autorisierung). Anbieterseitig besitzen nur "trusted entities" direkten Zugriff auf den DataExpert Rechner oder auf die DataExpert Komponenten. Es erfolgt keine spezielle Authentisierung oder Autorisierung.
6. **Datenbank-frei:** Anbieter- wie kundenseitig kommt für DataExpert keine Datenbank zum Einsatz. Der Einsatz und die Wahl einer allfälligen Datenbank erfolgt auf Seiten der Installateure durch die Branchensoftware-Hersteller, anbieterseitig durch die entsprechende IT Abteilung.
7. **Verschlüsselung:** Zu übertragende Daten (via Internet) werden für die Übertragung verschlüsselt. Nicht zu übertragenden Daten werden ausserhalb von DataExpert durch die Betriebssystem-Infrastruktur geschützt.

3 Detailspezifikation der Komponenten

3.1 DE_Transfer Client

3.1.1 Methoden (mit WebService)

3.1.1.1 GetBizMessageList

COM Schnittstelle:

HResult GetBizMessageList([in]string partnerID, [out]string listXml, [retval,out] transactionID)

WebService Schnittstelle:

```
public string GetBizMessageList(string username, string password, out string list)
```

Funktionalität:

Erzeugt eine Liste in XML aller für diesen User zum Download bereit stehender BizMsgs.

Die Liste beinhaltet pro BizMessage:

- ID (=Filename)
- Länge (in Bytes)
- Erstellungs-/Änderungsdatum

Bemerkung:

- Server-seitig wird die XML Tabelle gegeben durch alle Files im Download Verzeichnis mit Präfix „username_“.
- Die Liste entspricht dem XML Schema ‚GetList.xsd’ im Konfigurations-Verzeichnis.

3.1.1.2 ReadBizMessage

COM Schnittstelle:

HResult ReadBizMessage([in]string partner, [in] bizMsgID, [out]string bizMsg, [out]byte[] attachmentContent, [out]string attachmentFileExt, [retval,out]string transactionID)

WebService Schnittstelle:

```
public string ReadBizMessage2(string username, string password, string
bizMsgID, out string bizMsg, out Attachment attachment)
```

Funktionalität:

- Validierung Zugriffsberechtigung auf die BizMsg: BizMsgId besitzt Präfix ‚username_’
- Liefert im Parameter ‚bizMsg’ den Inhalt des Files. Falls ein Attachment vorhanden ist, wird dieses in der Struktur Attachment zurückgeliefert. Der Typ Attachment ist eine Datenstruktur, welche die Bytes des Attachments sowie die File-Extension des Attachments enthält. Falls die Message kein Attachment enthält, ist dieser Parameter Null bzw. Nothing.

3.1.1.3 GetIkkBizMessageDate

COM Schnittstelle:

```
HResult GetIkkBizMessageDate([in]string partner, [out]string
ikkBizMsgFileModificationDate)
```

WebService Schnittstelle:

```
public string GetIkkBizMessageDate(string username, string password, out string
ikkBizMsgFileModificationDate)
```

Funktionalität:

- Liefert im Parameter , ikkBizMsgFileModificationDate’ das Änderungsdatum des ikk-Files zurück.

3.1.1.4 ReadIkkBizMessage

COM Schnittstelle:

HResult ReadIkkBizMessage([in]string partner, [out]string ikkBizMsg)

WebService Schnittstelle:

```
public string ReadIkkBizMessage(string username, string password, out string
ikkBizMsg)
```

Funktionalität:

- Liefert im Parameter ‚ikkBizMsg' den Inhalt des ikk-Files.
- Falls serverseitig kein ikk-File für den Benutzer vorhanden ist, wird ein leerer String zurückgeliefert.

3.1.1.5 RemoveBizMessage

COM Schnittstelle:

HResult RemoveBizMessage([in]string partnerID, [in]string bizMsgID, [retval,out] transactionID)

WebService Schnittstelle:

```
public string RemoveBizMessage(string username, string password, string
bizMsgID)
```

Funktionalität:

- Validierung Zugriffsberechtigung auf die BizMsg: BizMsgId besitzt Präfix ‚username_'
- Löscht das File mit Name ‚BizMsgID' aus dem Download Verzeichnis

3.1.1.6 WriteBizMessage

COM Schnittstelle:

HResult WriteBizMessage([in]string partnerID, [in]string bizMsg, [in]byte[] attachmentContent, [in]string attachmentFileExt, [retval,out] transactionID)

WebService Schnittstelle:

```
public string WriteBizMessage2(string username, string password, string bizMsg,
Attachment attachment)
```

Funktionalität:

- Validiert den Inhalt von ‚bizMsg' mit dem XML Parser.
- Schreibt den Inhalt von ‚bizMsg' in das Upload Directory.
- Die im Filenamen verwendete ID entspricht der Transaction ID.
- Falls ein Attachment versendet werden soll, kann dieses in der Struktur Attachment mitgegeben werden. Der Typ Attachment ist eine Datenstruktur, welche die Bytes des Attachments sowie die File-Extension des Attachments enthält. Falls kein Attachment versendet werden soll, kann dieser Parameter auf Null bzw. Nothing gesetzt werden.

3.1.1.7 TriggerIkkBizMessage

COM Schnittstelle:

HResult TriggerIkkBizMessage([in]string partnerID)

WebService Schnittstelle:

```
public string TriggerIkkBizMessage(string username, string password)
```

Funktionalität:

- Schreibt einen Antrag (Datei) für die Aktualisierung des ikk-Files in das Uploadikk Directory

3.1.1.8 GetCatalogList

COM Schnittstelle:

HResult GetCatalogList([in]string partnerID, [out]string listXml, [retval,out] transactionID)

WebService Schnittstelle:

```
public string GetCatalogList(string username, string password, out string listXml)
```

Funktionalität:

Erstellt eine XML Liste aller zum Download im Verzeichnis Catalog bereitstehender Dateien (Katalog).

3.1.1.9 ReadCatalog

COM Schnittstelle:

HResult ReadCatalog([in]string partner, [in] catalogID, [out]string catalogXml, [retval,out]string transactionID)

WebService Schnittstelle:

```
public string ReadCatalog (string username, string password, string catalogID,
int offset, int size, out byte[] catalogFragment, out long length)
```

Funktionalität:

- Liefert im Parameter ‚catalogFragment den Inhalt des Files

Bemerkung:

- Während dem Katalog-Download wird jedes Mal ein Event ausgelöst, wenn ein Fragment des Katalogs geladen wurde (siehe 3.1.3.1). Der Event zeigt an, wie weit der Download bereits fortgeschritten ist. Die Angabe erfolgt in %, also mit einem Wert zwischen 0 und 100.

3.1.1.10 ReadSchema

COM Schnittstelle:

HResult ReadSchema([in]string partner, [uint] branchenId, [uint] processId, [out]string schema, [retval,out]string transactionID)

WebService Schnittstelle:

```
public string ReadSchema2 (string username, string password, uint branchenId,
uint processId, out string schema)
```

Funktionalität:

- Liefert im Parameter ‚schema' den Inhalt des Files „BizMessage_'BranchenId'_'ProcessId'_'PartnerID'.xsd" aus dem Konfigurations Verzeichnis des Servers

Bemerkung:

- ‚PartnerID' entspricht der PartnerID des Anbieters.

3.1.1.11 ChangePassword

COM Schnittstelle:

HResult ChangePassword([in]string partnerID, string newPassword, [retval,out] transactionID)

WebService Schnittstelle:

```
public string ChangePassword(string username, string password, string
newPassword)
```

Funktionalität:

- Ändert im Konfigurationsfile „ClientPartner.xml“ das mit der übergebenen PartnerID assoziierte Passwort auf ‚newPassword’.
- Ändert im Konfigurationsfile „ServerUser.xml“ das Passwort von ‚username’ auf ‚newPassword’

3.1.2 Methoden (ohne WebService)

3.1.2.1 ReadPartner

COM Schnittstelle:

HResult ReadPartner([out]string partnerXml)

Funktionalität allgemein / Funktionalität Client:

- liefert in , partnerXml’ die ClientPartner Daten als XML, d.h. den Inhalt des Files ‚ClientPartner.xml’ aus dem Konfigurations-Verzeichnis.

Bemerkung:

- DE_Inst greift mit dieser Funktion auf die Partnerdaten zu

3.1.2.2 WritePartner

COM Schnittstelle:

HResult WritePartner([in]string partnerXml)

Funktionalität allgemein / Funktionalität Client:

- validiert den Inhalt von ‚partner’ mit dem ‚ClientPartner’ Schema im Konfigurations-Verzeichnis.
- verifiziert, dass alle ‚PartnerID’s eindeutig sind
- speichert den Inhalt von , partnerXml’ im File ‚ClientPartner.xml’ im Konfigurations-Verzeichnis.

Bemerkung:

- DE_Inst greift mit dieser Funktion auf die Partnerdaten zu

3.1.2.3 ReadUser

COM Schnittstelle:

HResult ReadUser([out]string userXml)

Funktionalität allgemein / Funktionalität Client:

- liefert in , userXml’ die ClientUser Daten als XML, d.h. den Inhalt des Files ‚ClientUser.xml’ aus dem Konfigurations-Verzeichnis.

Bemerkung:

- DE_Inst greift mit dieser Funktion auf die Benutzerdaten zu

3.1.2.4 WriteUser

COM Schnittstelle:

HResult WriteUser([in]string userXml)

Funktionalität allgemein / Funktionalität Client:

- validiert den Inhalt von ‚userXml' mit dem ‚ClientUser' Schema im Konfigurations-Verzeichnis.
- verifziert, dass alle ‚usernamen' eindeutig sind
- speichert den Inhalt von ‚userXml' im File ‚ClientUser.xml' im Konfigurations-Verzeichnis.

Bemerkung:

- DE_Inst greift mit dieser Funktion auf die Benutzerdaten zu

3.1.2.5 GetProperty

COM Schnittstelle:

HResult GetProperty([in]string name, [out]string value)

Funktionalität:

- analog zu Server-seitigem GetProperty

Unterstützte Properties:

- ‚LogFile' : Filename des Log-Files (inklusive absolutem Pfad)
- ‚ConfigDir' : Pfad zum Client Konfigurationsverzeichnis
- ‚BlockSize' : Katalog-Download Blocksize in Bytes

3.1.2.6 SetProperty

COM Schnittstelle:

HResult SetProperty([in]string name, [in]string value)

Funktionalität:

Setzt das spezifizierte Property.

Unterstützte Properties:

- ‚BlockSize' : Katalog-Download Blocksize in Bytes

Bemerkung:

- Die Property Werte werden persistent gehalten.

3.1.3 Events

3.1.3.1 ReadCatalogProgress

COM Schnittstelle:

ReadCatalogProgress([in]int progress)

Funktionalität:

Während dem Katalog-Download (ReadCatalog) wird dieser Event jedes Mal ausgelöst, wenn ein Fragment des Katalogs geladen wurde. Der Parameter „progess" zeigt an, wie weit der Download bereits fortgeschritten ist. Diese Angabe erfolgt in %, also mit einem Wert zwischen 0 und 100.

4 DE_Inst

DE_Inst ist die grafische Oberfläche zur Mutation der Client-seitigen XML Konfigurationsdateien "ClientUser.xml" und "ClientPartner.xml". Die Mutation erfolgt mit Hilfe eines DB Grids.

DE_Inst besitzt folgende Funktionalitäten:

- Erfassen, Ändern, Löschen, Speichern
  - aller ClientUser Daten
  - aller ClientPartner Daten
- Partner Passwort ändern
  - ändert das Passwort lokal und auf dem Server des entsprechenden Partners (Aufruf von ChangePassword)
  - vor jeder Passwortänderung werden alle vorgängig geänderten Daten gespeichert oder verworfen
- Import von ClientPartner / ClientUser Dateien
  - validieren der Daten mit entsprechendem Schema im Konfigurations-Verzeichnis
  - die importierten Daten werden zu den bestehenden hinzugefügt / überschrieben.
- Export von ClientPartner / ClientUser Datei (entspricht "safe as" Funktionalität)
- Wird DE_Inst mit einem Argument aufgerufen, so wird dieses als der Filename einer zu importierenden ClientPartner Datei interpretiert, d.h. die Daten werden importiert.
- Sind Änderungen erfolgt, so erscheint vor dem verlassen von DE_Inst ein "Speichern ja/nein" Dialog.
- Setzen der Proxy-Konfiguration.

Bemerkungen:

- DE_Inst greift auf die ClientUser und ClientPartner Daten via die DE_Transfer Client Schnittstelle zu. Damit erfolgt die Autorisierung via Windows Logon implizit durch DE_Transfer.
- Die in DE_Inst importierten Daten werden typischerweise durch einen Anbieter mit Hilfe von DE_Anbiet erzeugt.

5 DE_Anbiet

DE_Anbiet ist die grafische Oberfläche zur Mutation der Server-seitigen

- XML Konfigurationsdatei "ServerUser.xml"
- des Schemas "BizMessage_'BranchenID'_'ProcessID'_'PartnerID'.xsd".

Die Mutationen erfolgen mit Hilfe von DB Grids.

DE_Anbiet besitzt folgende Funktionalitäten:

- Erfassen, Ändern, Löschen, Speichern
  - aller ServerUser Daten
- Importieren von ServerUser Dateien
  - validieren der Daten mit "ServerUser" Schema im Konfigurations-Verzeichnis
  - die importierten Daten werden zu den bestehenden hinzugefügt / überschrieben.
- Export von ServerUser Datei (entspricht "safe as" Funktionalität)
- Für einen selektierten User-Export einer ClientPartner Datei mit den Verbindungsinformationen für diesen Server (WebService URL etc.).
- Laden

  - des Original BizMessage Schemas "BizMessage.xsd"
  - des aktuellen BizMessage Schemas "BizMessage_'PartnerID'.xsd" (Default)
- Ändern des Wertes "optional" oder "required" für die "use" Attribute im geladenen "BizMessage" Schema
- Speichern des BizMessage Schemas unter "BizMessage_'PartnerID'.xsd"

Bemerkung:

- Pro Anbieter gibt es 1 BizMessage Schema. Es werden keine "type extensions" etc. verwendet.

6 Diverses

6.1 Verzeichnis Struktur

Alle für DataExpert spezifischen Komponenten und Daten befinden sich in einem Directory Baum mit folgender Struktur

- IGH
  - DataExpert
    - **Client** : beinhaltet die ausführbaren Client Software-Komponenten
      - Log : beinhaltet Log-Files
        - DE_ClientLog.csv
      - Config : beinhaltet DataExpert spezifische Konfigurations-Files und Schemas (z.B. für ClientUser, ClientPartner)
        - ClientUser.xml
        - ClientPartner.xml
        - ClientUser.xsd
        - ClientPartner.xsd
        - Catalog.xsd (Schema zum Katalog)
        - GetList.xsd (Schema zu GetBizMessageList)
    - **Server** : beinhaltet die ausführbaren Server Software-Komponenten
      - Upload : beinhaltet von Kunden übermittelte BizMsgs
      - Download : beinhaltet an Kunden noch zu übermittelnde BizMsgs
      - Uploadikk : beinhaltet von Kunden übermittelten Aktualisierungsanforderungen
      - Downloadikk : beinhaltet die für den Kunden aktuelle ikk Datei
      - Catalog : beinhaltet Katalogdaten
      - Log : beinhaltet das Log-File
        - DE_ServerLog.csv
      - Config : beinhaltet DataExpert spezifische Konfigurations-Files und Schemas (z.B. für ServerUser, BizMessages und Kataloge)
        - ServerUser.xml
        - ServerUser.xsd
        - BizMessage.xsd (das BizMessage Schema im Original)

- BizMessage_'BranchenId'_'ProcessId'_'PartnerID'.xsd (das BizMessage Schema für diesen Anbieter)
- Catalog.xsd
- GetList.xsd (Schema zu GetBizMessageList etc.)

6.2 Autorisierungs- & Verbindungsdaten

Alle Konfigurationsdaten werden als .XML Files im Config Directory gehalten.

6.2.1 Tabelle "ClientUser"

Dient zur Autorisierung eines Aufrufs von DE_Transfer auf Client Seite.

WindowsLogon : String / PrimaryKey / Windows Logon Name des Benutzers
WriteBizMsg : Boolean / Autorisierung für die Methoden
- WriteBizMsg

ReadBizMsg : Boolean / Autorisierung für die Methoden
- GetBizMsgList
- ReadBizMsg
- RemoveBizMsg

Catalog : Boolean / Autorisierung für die Methoden
- GetCatalogList
- ReadCatalog

Administration: Boolean / Autorisierung für die Methoden
- ChangePassword
- readUser
- writeUser
- readPartner
- writePartner

Bemerkungen:

- Nicht aufgeführte Methoden wie
  - readSchema
  - getProperty

  sind für alle Benutzer mit einem Eintrag in der ClientUser Tabelle autorisiert.
- Nicht aufgeführte Benutzer besitzen keine Autorisierung für irgendeinen Methodenaufruf.
- Pro Windows Logon gibt es höchstens 1 Eintrag

6.2.2 Tabelle "ClientPartner"

Beinhaltet die Verbindungsinformationen zum Client-seitigen Aufruf eines WebServices.

PartnerID : String / Primary Key
Name : String / Name des Partners (z.B. "Sanitas Trösch")
WebServiceURL : String / URL des DataExpert WebServices
Username : String / Benutzername des Kunden bei diesem Anbieter
Password : String / Passwort des Kunden bei diesem Anbieter

Bemerkung:

- Die PartnerID ist eine Schweizweit eindeutige Identifikationsnummer des Anbieters (z.B. "1910" = Georg Fischer Rohrleitungsysteme (Schweiz) AG).

6.2.3 Tabelle "ServerUser"

Dient zur Server-seitigen Autorisierung eines Aufrufs des WebServices durch einen Client.

Username	: String / PrimaryKey / Identifizierung des Kunden
Password	: String / Passwort dieses Kunden
WriteBizMsg	: Boolean / Autorisierung für die WebService Methode „WriteBizMsg“
PID	: String / Paynet ID (optional)

Bemerkungen:

- Alle Methoden ausser "WriteBizMsg" sind für alle Benutzer mit einem Eintrag in der ClientUser Tabelle autorisiert. Nicht aufgeführte Benutzer besitzen keine Autorisierung für irgendeinen Methodenaufruf.
- Implizit ist somit der Aufruf von ReadBizMsg etc. allen Kunden erlaubt.

6.3 Filenamen

Metainformation zu BizMsgs wie Absender/Empfänger oder die ID werden in den Filenamen kodiert. Die Files in den Server-seitigen Upload- und Download-Verzeichnissen besitzen folgende Namensstruktur:

***username_datum_zeit_id*.xml**

mit:

- username : Name des Benutzers von dem / für den diese Nachricht bestimmt ist
- Datum : Erstellungsdatum in der Form JJMMTT
- Zeit : Erstellungszeit in der Form hhmmss
- id : erzeugte TransaktionsID (GUID)

ikk-Dateien:

***username_id*.xml**

- username : Name des Benutzers von dem / für den diese Nachricht bestimmt ist
- id : ikk (individuelle Kundenkonditionen)

ikk- Aktualisierungsanfrage:

***username*_refresh.xml**

- username : Name des Benutzers von dem Aktualisierungsanfrage kommt

(Inhalt der Aktualisierungsanfrage: <Item ID=“*Kunde*_refresh“ Date=“*2010-03-17*“ />)

6.4 Filename-Extensions

DataExpert verwendet folgende Filename Extensions

- .dep : zu importierende Client-seitige Partner Daten (erzeugt aus DE_Anbiet)
- .csv : Log Dateien
- .xml : alle sonstigen XML Dateien

6.5 Transportvolumen

- Katalogdateien können Grössen von > 1 MByte erreichen
- Schema, Listen und BizMsgs besitzen Grössen im ein- bis zweistelligen KByte Bereich

- Client-seitige Bandbreite >= 28 KBits/sek

6.6 Einspeisen des Kataloges

Die Kataloge werden direkt durch das Abspeichern im Katalog-Verzeichnis an DataExpert übergeben.

Bemerkungen:

- Validierung des Kataloges ist Sache der abspeichernden Applikation.
- Namenskonvention gemäss Dokument "Codierung-xml-Dateien.pdf" beachten.

6.7 Sicherheit

- Authentisierung
  - Client-seitig : durch Windows Logon
  - WebService : durch Username/Passwort gemäss Konfigurations-Tabelle "ServerUser"
  - Server-seitig : durch Windows Logon
- Autorisierung
  - Client-seitig : durch Konfigurations-Tabelle "ClientUser"
  - WebService : durch Konfigurations-Tabelle "ServerUser"
  - Server-seitig : durch Windows Administration
- Access aus Internet / Firewalls
  - gemäss firmenspezifischen Sicherheitskonzepten
  - offene HTTP / HTTPS Ports von / zum Internet
- Transport
  - durch HTTP/HTTPS
- Datenhaltung
  - Client-seitig : ausserhalb DataExpert, durch Betriebssystem Infrastruktur
  - Server-seitig : ausserhalb DataExpert, durch Betriebssystem Infrastruktur

6.8 XML Parser

DataExpert benützt für alle XML-Validierungen den .NET XML Parser

6.9 Log File

Das Log-File beinhaltet nachfolgende Informationen:

- Zeitstempel : Im Format TT.MM.JJ HH:MM:SS
- Benutzer : Windows Logon (Client), Username (Server)
- Typ : Entry, WS, Debug, Exit, Abort
- Methode : Name der aufgerufenen Methode (inkl. URL bei WS)
- Argumente : Kurzversion der übergebenen Argumente (Entry, WS)
- TransactionID : TransactionID (sofern vorhanden)
- ReturnValue : (Exit)
- Comment : Text, enthält

Bemerkungen

- Das LogFile wird als CSV (Excel) und nicht als XML File geschrieben. Dies erlaubt das schnelle hinzufügen von Log-Einträgen durch einen File-Append.
- Server und Client besitzen das gleiche Log File Format.

- Die Grösse des Log-Files ist nicht beschränkt.
- Die Log Files können mit Excel betrachtet und analysiert werden.

6.10 Installation

Ein Installationsprogramm installiert die Data-Expert Software Komponenten (Setup).

Bemerkungen:

- Das .NET Framework wird unabhängig von DataExpert installiert
- Zur Installationszeit sind folgende Elemente konfigurierbar
  - Client
    - Ort des IGH Verzeichnisbaumes
  - Server
    - Ort des IGH Verzeichnisbaumes
    - PartnerID dieses Anbieters (für Tabelle ClientPartner)
    - PartnerName dieses Anbieters (für Tabelle ClientPartner)
    - URL des WebServers (für Tabelle ClientPartner, für Deployment des WebService)
- Während der Server-seitigen Installation wird
  - Original "BizMessage_*_*.xsd" auf das Partner-spezifische BizMessage_*_*_'PartnerID'.xsd kopiert.

7 Systemvoraussetzung

7.1 Client:

WinXP, Vista, Windows 7, Windows 8
.NET Framework 4.0

7.2 Server:

- Windows Server 2008
- .Net Framework Version 4.0 (Volle Version, nicht die Client-Version!)
- IIS
- IIS 6 Management Compatibility:
- X.509

8 DataExpert-Versionen

Version	Datum	Kommentar
1.1	21.10.2002	1. Produktive Version
1.2	24.02.2005	Anpassungen am Setup-Modul
1.3	06.03.2006	Neue Methode WriteBizMessage2 implemtiert (Zusatzdatei als Attachement)
1.4	09.04.2007	Rechnungsprovider PostFinance implementiert
1.5	08.04.2010	Neue Methoden implementiert (ReadIkkBizMessage, TriggerIkkBizMessage)
2.0.0.2	06.06.2012	DataExpert 2.0 / Direktaufruf von WebService
2.0.1.0	28.01.2015	Neue Methode GetIkkBizMessageDate implementiert (Abfrage auf Publ.-Datum)