



*Der SAP-Conversion-Agent
von Itemfield*

Verarbeiten von COBOL- Daten in Conversion Agent

Conversion Agent 4

Verarbeiten von COBOL-Daten in Conversion Agent

© 2006 Itemfield Inc. Alle Rechte vorbehalten.

Itemfield hat möglicherweise Patente, angemeldete Patente, Marken, Urheberrechte oder sonstige Rechte an geistigem Eigentum inne, die Inhalte dieses Dokumentes abdecken. Sofern nichts anderes ausdrücklich in einem schriftlichen Lizenzvertrag mit Itemfield vereinbart wurde, erhalten Sie durch die Bereitstellung dieses Dokumentes keinerlei Anspruch auf diese Patente, Marken, Urheberrechte oder auf sonstiges geistiges Eigentum.

Die in diesem Dokument enthaltenen Informationen können jederzeit ohne vorherige Ankündigung geändert werden. Der Anwender trägt die Verantwortung für die Einhaltung aller anwendbaren Urheberrechte. Ohne ausdrückliche schriftliche Genehmigung durch Itemfield Inc. darf kein Teil dieses Dokumentes in irgendeiner Form oder auf irgendeine elektronische oder mechanische Weise zu irgendeinem Zweck vervielfältigt oder weitergegeben werden.

SAP AG

<http://www.sap.com>

Dokumentdaten:

Version: 4

Datum: September 2006

Inhalt

Übersicht	1
Importieren einer COBOL-Datendefinition	1
Unterstützte COBOL-Funktionen.....	2
Beispiel für einen COBOL-Import.....	2
COBOL-Projekte	5
Testen des COBOL-Parsers.....	5
Testen des COBOL-Serializers	6
Bearbeiten eines COBOL-Projekts	7
Einbinden in andere Datenumwandlungen.....	7
Bereitstellen von COBOL-Projekten als Conversion-Agent-Dienst	8

Übersicht

Conversion Agent unterstützt die Umwandlung von COBOL-Daten in eine und aus einer XML-Darstellung. Über die XML-Darstellung können Sie die COBOL-Daten in ein und aus einem beliebigen Datenformat verwandeln. Dadurch können Sie alte COBOL-Anwendungen mühelos in Anwendungen integrieren, die die Daten in irgendeinem anderen Format darstellen.

Die Unterstützung von COBOL ist auf folgende Art implementiert:

1. In Conversion Agent Studio können Sie eine COBOL-Datendefinition aus einem Copybook importieren.
2. Das dabei erzeugte Conversion-Agent-Projekt enthält:
 - Ein XSD-Schema, das eine XML-Darstellung der COBOL-Datenstruktur definiert.
 - Einen Parser, der die Eingabedaten entsprechend der COBOL-Datendefinition in eine XML-Darstellung umwandelt.
 - Einen Serializer, der die Daten aus der XML-Darstellung in die ursprüngliche COBOL-Darstellung zurückverwandelt.
3. Sie können den Parser und den Serializer als Conversion-Agent-Dienste bereitstellen, die COBOL-Daten verarbeiten, oder in andere Conversion-Agent-Projekte und -Dienste mitbeliebigem Datenformat einbinden.

Importieren einer COBOL-Datendefinition

Sie können eine COBOL-Datendefinition in ein neues Conversion-Agent-Projekt importieren. Dabei gehen Sie folgendermaßen vor:

1. Wählen Sie in Conversion Agent Studio „Datei > Neu > Projekt“ aus. Daraufhin wird der Assistent „Neues Projekt“ geöffnet.

Hinweis: Für diese Aufgabe müssen Sie „Datei > Neu > Projekt“ verwenden. Nicht möglich ist die Verwendung der Funktion „Datei > Importieren“, die für den Import vorhandener Conversion-Agent- oder Eclipse-Projekte vorgesehen ist.

2. Wählen Sie im linken Bereich Conversion Agent aus. Markieren Sie im rechten Bereich „Projekt importieren“.
3. Auf der nächsten Seite des Assistenten geben Sie einen Projektnamen und den Speicherort ein. Der Standardspeicherort ist der Workspace-Ordner von Conversion Agent Studio.
4. Auf der nächsten Seite navigieren Sie zur Datei mit der COBOL-Datendefinition (einem Copybook).
5. Nach Abschluss des Assistenten erzeugt Conversion Agent Studio ein Projekt, das ein XSD-Schema, einen Parser und einen Serializer für die COBOL-Daten enthält.
6. Testen Sie den Parser und den Serializer und setzen Sie sie in Ihren Conversion-Agent-Anwendungen ein. Zum Beispiel können Sie sie als Conversion-Agent-Dienste bereitstellen (siehe *COBOL-Projekte* weiter unten).

Unterstützte COBOL-Funktionen

Der COBOL-Import unterstützt Datendefinitionen beliebiger Komplexität. Mögliche Datentypen sind zum Beispiel gepackte Dezimalzahlen (COMP-3), Binärdaten (COMP-1, COMP-2 oder COMP-4) und logischer Dezimalpunkt (99v99). Außerdem können die Datendefinitionen Merkmale wie etwa die Bedingungen REDEFINES, OCCURS und OCCURS DEPENDING ON enthalten.

Dabei muss die Datendefinition die folgenden Anforderungen erfüllen:

- Der Code muss in Großbuchstaben geschrieben sein.
- Je Zeile sind höchstens 72 Zeichen zulässig (kein Text nach Spalte 72).
- Die erste Zeile muss ein Kommentar (* in Spalte 7) sein oder mit der Nummer einer Ebene beginnen.
- Die erste Ebenennummer muss sich in Spalte 1 oder 8 befinden.

Die folgenden COBOL-Funktionen werden derzeit nicht unterstützt:

- die speziellen Ebenennummern 66, 77 und 88
- USAGE-Bedingungen auf Gruppenebene
- die Bedingung INDEXED BY
- POINTER und PROCEDURE-POINTER

Beispiel für einen COBOL-Import

In diesem Beispiel wird die folgende COBOL-Datendefinitionsdatei importiert:

```
000100* -----
```

```

001300* INPUT SAMPLE
001600* -----
001700    01 SAMI 001.
001800*
003000    05 SAI -KEY-LNAME      PIC X(10).
003100*      ----- KEY-LNAME
003800    05 SAI -FNAME        PIC X(10).
003810*      ----- FNAME
006000    05 SAI -SALARY       PIC X(8).
006100*      ----- SALARY
006600    05 SAI -ID          PIC X(17).
006700*      ----- ID
007800    05 SAI -BDATE        PIC 9(8).
007810*      ----- BIRTHDATE (JJJJMMTT)

```

Dem Import liegt das unter *Importieren einer COBOL-Datendefinition* beschriebene Verfahren zugrunde, und das Projekt erhält den Namen COBOL_DEMO. Wie die folgende Abbildung zeigt, erzeugt das Verfahren automatisch ein XSD-Schema, einen Parser und einen Serializer:

```

*COBOL_DEMO_schema.xsd
<?xml version="1.0" encoding="windows-1252"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="SAMI001">
    <xs:complexType>
      <xs:sequence>
        <xs:element default="" name="SAI-KEY-LNAME" type="xs:string" />
        <xs:element default="" name="SAI-FNAME" type="xs:string" />
        <xs:element default="" name="SAI-SALARY" type="xs:string" />
        <xs:element default="" name="SAI-ID" type="xs:string" />
        <xs:element default="0" name="SAI-BDATE" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="SAMI001_variable_type">
    <xs:sequence>
      <xs:element name="skip" />
      <xs:element name="numericDefault" />
      <xs:element name="stringsDefault" />
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

Das vom COBOL-Importverfahren automatisch erzeugte XSD-Schema für eine XML-Darstellung der COBOL-Daten

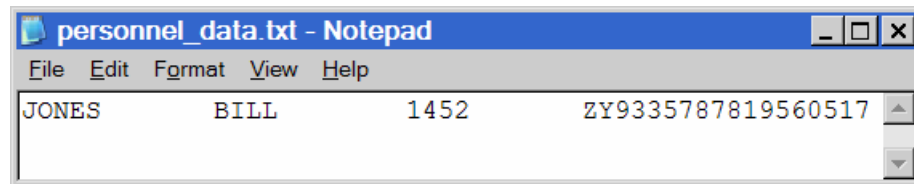
COBOL-Projekte

Dieser Abschnitt beschreibt kurz einige Verfahren zum Testen von und Arbeiten mit COBOL-Datenumwandlungen in Conversion Agent. Ausführliche Informationen zur Arbeit mit Datenumwandlungen finden Sie in *Erste Schritte mit Conversion Agent* und im *Conversion-Agent-Studio-Benutzerhandbuch*.

Testen des COBOL-Parsers

Den COBOL-Parser können Sie testen, indem Sie COBOL-Beispieldaten in XML umwandeln und die Ausgabe überprüfen. Dafür benötigen Sie eine Eingabedatei mit COBOL-Beispieldaten. Die Datenstruktur muss der importierten Datendefinition entsprechen.

In der folgenden Abbildung verwenden wir eine COBOL-Datendatei, die der oben importierten Datendefinition entspricht (siehe *Beispiel für einen COBOL-Import*).



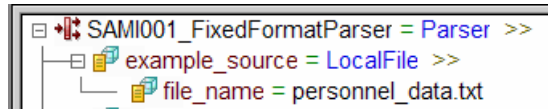
Eingabedatei mit COBOL-Beispieldaten

Dabei gehen Sie folgendermaßen vor:

1. Klicken Sie in der Ansicht Conversion Agent Explorer von Conversion Agent Studio die Skriptdatei (*. tgp) des Parsers doppelt an. Daraufhin wird der Parser im IntelliScript-Editor geöffnet.
2. Klicken Sie den Namen des Parsers (meist ist dies die erste Zeile im IntelliScript) mit der rechten Maustaste an und wählen Sie „Als Startkomponente festlegen“ aus.
3. Erweitern Sie den IntelliScript-Baum und bearbeiten Sie die Eigenschaft `example_source` des Parsers. Ändern Sie ihren Wert von `Text` auf `Local File`.

Hinweis: Der automatisch erzeugte COBOL-Parser ist so konfiguriert, dass kein Beispielquellendokument erforderlich ist. Nach Abschluss des Tests können Sie die Beispielquelle entfernen. Das Beibehalten der Beispielquelle hat zur Laufzeit keinerlei Auswirkungen auf die Datenumwandlung.

4. Weisen Sie in die Komponente `Local File` geschachtelt die Eigenschaft `file_name` zu. Navigieren Sie zur Eingabedatei, die die COBOL-Beispieldaten enthält.



Zuweisen der Eingabedatei zur Eigenschaft example_source des Parsers

5. Das Dokument wird im Beispielbereich des IntelliScript-Editors aufgeführt.
Wenn der Beispielbereich nicht sichtbar ist, wählen Sie im Menü die Option „IntelliScript > Beide“ aus. Wenn das Dokument nicht angezeigt wird, klicken Sie den Namen des Parsers im IntelliScript an und wählen „Beispielquelle öffnen“ aus.
6. Optional können Sie mit dem Befehl „IntelliScript > Beispiel markieren“ die Daten markieren lassen, die der Parser im Dokument findet.
Wenn die Daten der importierten Datendefinition entsprechen, findet der Parser alle Daten im Dokument. Der Befehl „Beispiel markieren“ sollte demnach das gesamte Dokument markieren.
Hinweis: Die Farbcodierung wird im Buch *Conversion Agent Studio in Eclipse* erklärt.
7. Aktivieren Sie den Parser mit dem Befehl „Ausführen > Ausführen“.
8. Die Ereignisansicht enthält ein Protokoll des Vorgangs. Wie Sie das Protokoll interpretieren, wird im Kapitel *Projekte ausführen und testen im Conversion-Agent-Studio-Benutzerhandbuch* erklärt.
9. Um die Parserausgabe einzusehen, klicken Sie die Datei `Results\output.xml` im Conversion Agent Explorer doppelt an.

```
<?xml version="1.0" encoding="windows-1252" ?>
- <SAMI001>
  <SAI-KEY-LNAME>JONES</SAI-KEY-LNAME>
  <SAI-FNAME>BILL</SAI-FNAME>
  <SAI-SALARY>1452</SAI-SALARY>
  <SAI-ID>ZY93357878</SAI-ID>
  <SAI-BDATE>19560517</SAI-BDATE>
</SAMI001>
```

XML-Ausgabe des Parsers

Testen des COBOL-Serializers

Nachdem Sie den COBOL-Parser getestet haben, können Sie den COBOL-Serializer auf der Parserausgabe ausführen (siehe *Testen des COBOL-Parsers*). Bei diesem Vorgang sollten wieder die ursprünglichen COBOL-Daten erzeugt werden.

1. Klicken Sie in der Ansicht Conversion Agent Explorer die Skriptdatei (*. tgp) des Serializers doppelt an. Daraufhin wird der Serializer im IntelliScript-Editor geöffnet.
2. Klicken Sie den Namen des Serializers mit der rechten Maustaste an und wählen Sie „Als Startkomponente festlegen“ aus.
3. Aktivieren Sie den Serializer mit dem Befehl „Ausführen > Ausführen“. Wenn Sie dazu aufgefordert werden, navigieren Sie zu der Datei Resul ts\output. xml , die Sie zuvor mit dem Parser erzeugt haben.
4. Die Ereignisansicht enthält ein Protokoll des Vorgangs.
5. Um die Ausgabe des Serializers einzusehen, klicken Sie die Datei Resul ts\output. txt im Conversion Agent Explorer doppelt an. Sie sollte mit der ursprünglichen Ausgabe übereinstimmen, auf der Sie den Parser ausgeführt haben.

Wenn Sie den Serializer auf der oben gezeigten Beispiel-Parserausgabe ausführen, sollte er die folgende Ausgabe erzeugen:

```
JONES      BILL      1452      ZY9335787819560517
```

Die Ausgabe des Serializers ist mit den ursprünglichen COBOL-Daten identisch.

Bearbeiten eines COBOL-Projekts

In Conversion Agent Studio können Sie ein selbst erzeugtes COBOL-Projekt bearbeiten.

Dabei sollten Sie die Bearbeitung sorgfältig dokumentieren. Diese Dokumentation kann später wichtig werden, wenn Sie die COBOL-Datendefinition ändern und in ein neues Projekt importieren und dabei Ihre Bearbeitung rekonstruieren müssen.

Einbinden in andere Datenumwandlungen

Sie können selbst erzeugte COBOL-Datenumwandlungen in andere Datenumwandlungskomponenten von Conversion Agent einbinden. Das folgende Beispiel veranschaulicht dies anhand eines häufig verwendeten Verfahrens.

Beispiel

Angenommen, Sie möchten eine alte COBOL-Anwendung in eine Datenbanksystem einbinden. Dabei können Sie folgendermaßen vorgehen:

1. So speichern Sie die COBOL-Daten in der Datenbank:

- a. Wandeln Sie die nativen COBOL-Daten mit dem automatisch erzeugten COBOL-Parser in eine XML-Darstellung um.
 - b. Wandeln Sie die XML-Darstellung mit einem Mapper in die entsprechenden Datenbankfelder um.
 - c. Der Mapper enthält die Komponente `ODBCActi on`, die das Ergebnis in der Datenbank speichert.
2. So rufen Sie die COBOL-Daten aus der Datenbank ab:
- a. Rufen Sie die Daten mit einem zweiten Mapper, der die Komponente `ODBCActi on` enthält, aus der Datenbank ab.
 - b. Der Mapper wandelt die Datenbankfelder in die XML-Darstellung der COBOL-Daten um.
 - c. Wandeln Sie das XML mit dem automatisch erzeugten COBOL-Serializer in die native COBOL-Darstellung um.

Implementierung

Sie können die automatisch erzeugten COBOL-Datenumwandlung mit anderen Datenumwandlungskomponenten in einem einzigen Conversion-Agent-Projekt verketteten. Zum Beispiel können Sie mit den Aktionen `RunParser`, `RunMapper` und `RunSeriali zer` die im obigen Beispiel beschriebenen Komponenten aktivieren. Weitere Informationen finden Sie im *Conversion-Agent-Studio-Benutzerhandbuch*.

Stattdessen können Sie die Komponenten auch in separaten Conversion-Agent-Diensten bereitstellen, die Sie der Reihe nach aktivieren. Weitere Informationen zur Ausführung der Dienste finden Sie im *Conversion-Agent-Engine-Entwicklerhandbuch*.

Bereitstellen von COBOL-Projekten als Conversion-Agent-Dienst

Nach dem Testen können Sie den Parser und den Serializer als Conversion-Agent-Dienste bereitstellen, die auf der Conversion Agent Engine ausgeführt werden.

Eine Anleitung dazu finden Sie im Abschnitt *Bereitstellen von Conversion-Agent-Diensten* im *Conversion-Agent-Studio-Benutzerhandbuch*. Achten Sie darauf, dass Sie den Parser oder Serializer als Startkomponente definieren, bevor Sie den Befehl „Projekt > Bereitstellen“ ausführen.

Weitere Informationen zur Ausführung der Dienste finden Sie im *Conversion-Agent-Engine-Entwicklerhandbuch*.