



EDIFACT TO XML—A PROGRESS® DATADIRECT XML CONVERTERS® SOLUTION

Posted on Apr. 06, 2007, 07:23 AM, by Ricardo Andres Maienza, TalentLab, in Beginner, SAP Exchange Infrastructure (XI)

Have you ever faced an EDI message exchange using SAP Exchange Infrastructure? If not, I'd say you're lucky. Because although EDI can appear trivial under certain circumstances, believe me it is not. The typical scenario could be EDI to IDoc and vice versa, for instance. Whatever format you choose on "the other side," EDI must be handled carefully, and that's not easy. I'm not an EDI expert—far from being that, actually—but I've been forced to learn some EDI basics in my last projects. The first thing that I realized was that there are a lot of EDI dialects. For instance, Odette, an EDIFACT dialect for the automotive industry, is not universally supported. Moreover, each company using EDI usually can customize EDI messages based on their needs, just as an SAP guy can create a CIM type to extend standard IDoc types functionality.



And that's one of the main reasons why we chose to use the product that we use: Progress® DataDirect XML Converters®. DataDirect differs from any other approach: it can virtually handle any EDI message, even those which are not known to the converter itself! Secondly, it can be easily be integrated in Java and .NET programs, which suits perfectly in the XI context. And, last but not least, its licensing cost is fair and affordable.

Of course, I'm aware that there are several different manners to handle this topic. Conversion Agent by Itemfield is one, but, in our case, it was simply too expensive (in terms of man-days) to handle the Odette dialect, which is not supported by the standard transformation provided. The historical method is, of course, Seeburger, but we didn't even take it into consideration, due to its high license cost.

Read on to find out how you can deal with EDI messages in SAP XI, in hours, not weeks!

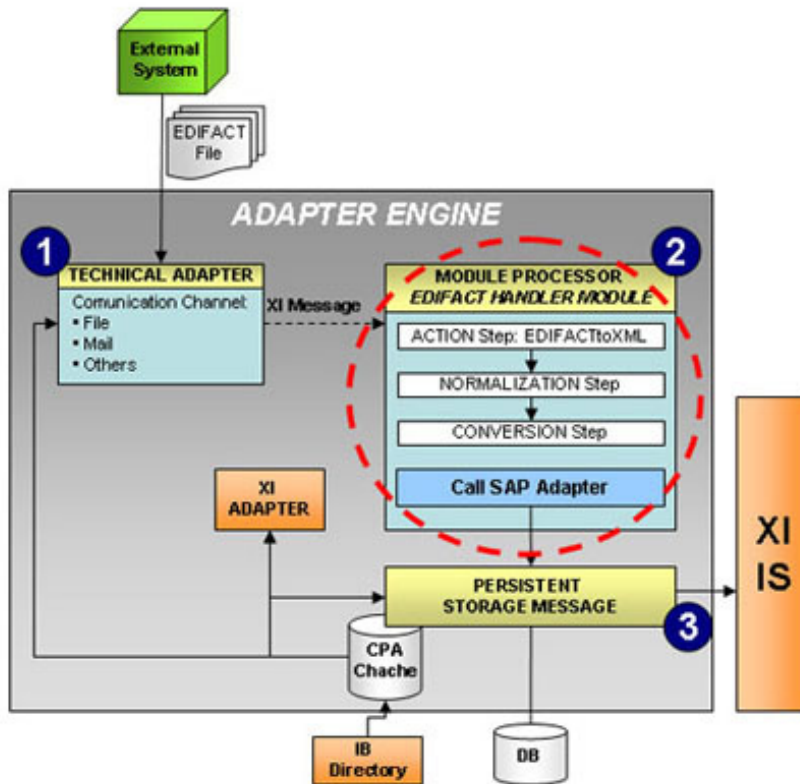
THE GOAL

In summary, the objective of this blog posting is to describe how to process an EDIFACT message inside SAP XI, using the "DataDirect XML Converters" solution.

Note: The DataDirect XML Converters library provides a huge vocabulary of EDI standards (EDIFACT, X12, EANCOM, IATA, Flat Files, etc.). For further details: www.xmlconverters.com

The Scenario:

The process flow is illustrated in the following figure:



1. The Communication Channel picks up the native Edifact message via FTP or a mail protocol.
2. The Edifact Handler Module converts the native Edifact message into XML Edifact using DataDirect XML Converters.
3. The message is sent to the XI IS Pipeline for further processing—Mapping -> Target Format (Flat File, XML, etc.).

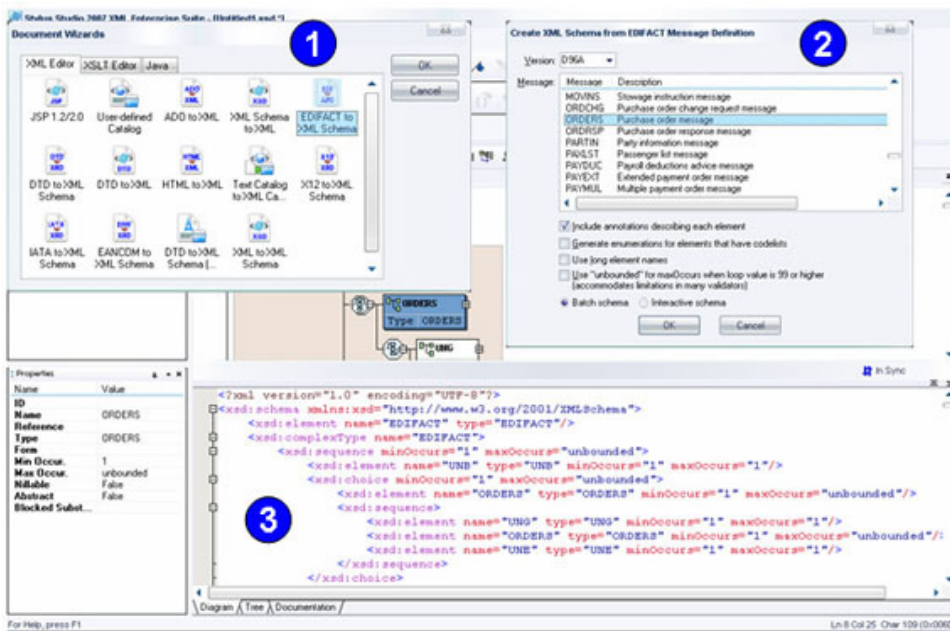
TASKS IMPLEMENTATION

- > **Stylus Studio®:** Generate the EDIFACT XML Schema. The XSD file will be imported in Repository Object like External Definition.
- > **Stylus Studio:** Generate the URL string that will be used by the XML Converter to convert the native EDIFACT to XML EDIFACT or vice versa.
- > **Nwds:** Create and deploy the module EDIFACTHandler to XI Server.
- > **XI:** Create XI repository Object (XML EDIFACT Schema importing).
- > **XI:** Create XI Directory Object (XML Converter module Configuration).

STYLUS STUDIO

EDIFACT XML Schema Generation

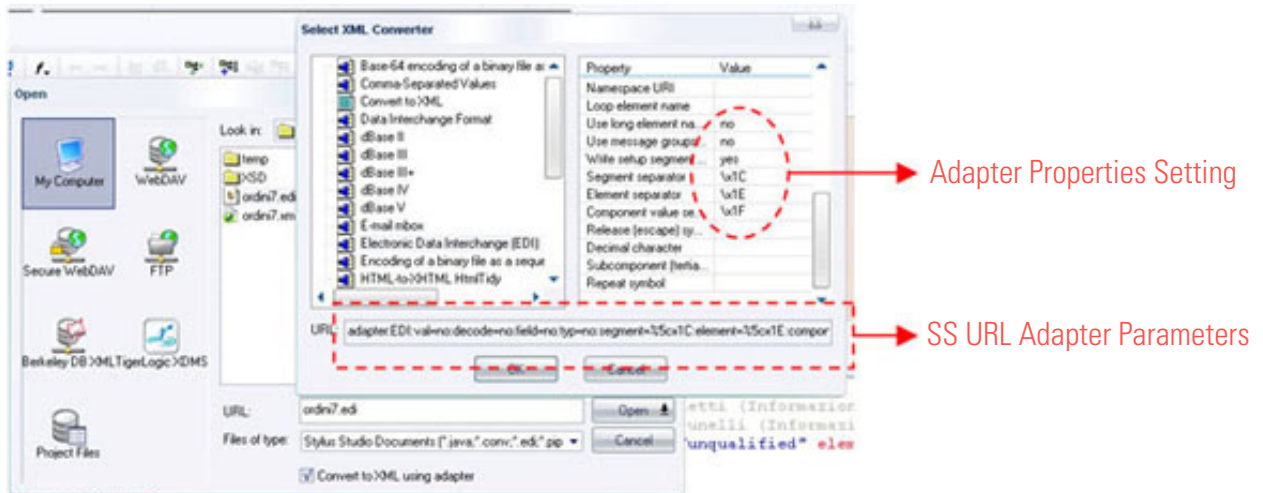
Stylus Studio suite allows you to generate XML schema for any type and any version that the EDIFACT standard supports.



URL Parameters Generation

It is necessary to create the parameter string (URL) used by the XML Converters to convert the document from one syntax to another. The URL will be used like parameters in the module tab configuration (look at the Directory—Module Configuration).

Note: Stylus Studio supports even the UNOC and UNOB syntax conversion. The special segment, element and component separator (hexadecimal characters) are used by this syntax.



NWDS

Native EDIFACT message to EDIFACT XML Conversion (AF Module):

I will not get into all of the details about how to create a module; this is described in the “How To Create Modules for the J2EE Adapter Engine” link or around in SDN.

For further details about the Module XML Converter configuration see below the AF Parameters.

THE CODE

EJB Module Code

```
package biz.talentlab.sap.nw.xi.ae.modules;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
import javax.ejb.CreateException;

import com.sap.aui.af.mp.module.*;
import com.sap.aui.af.ra.ms.api.*;
import com.sap.aui.af.service.auditlog.*;

import biz.talentlab.helpers.*;
import javax.xml.transform.stream.*;

import java.io.*;
import java.util.*;
```

```

/**
 * @ejbLocal <{com.ie.sap.nw.xi.ae.modules.EDIFACTHandlerLocal}>
 * @ejbLocalHome <{com.ie.sap.nw.xi.ae.modules.EDIFACTHandlerLocalHome}>
 * @stateless
 */
public class EDIFACTHandlerBean implements SessionBean {

    // Basic instances
    Object obj = null;
    Message msg = null;
    AuditMessageKey amk = null;
    ModuleException mEx = null;

    public ModuleData process(ModuleContext moduleContext, ModuleData inputModuleData) throws ModuleException {

        try {
            obj = inputModuleData.getPrincipalData();
            msg = (Message) obj;
            if (msg.getMessageDirection() == MessageDirection.INBOUND)
                amk = new AuditMessageKey(msg.getMessageId(), AuditDirection.INBOUND);
            else
                amk = new AuditMessageKey(msg.getMessageId(), AuditDirection.OUTBOUND);
            mc = moduleContext;
        } catch (Exception e) {
            Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while creating basic instances (obj,msg,amk,mp)");
            throw mEx = new ModuleException(auditStr + "Error while creating basic instances (obj,msg,amk,mp)");
        }

        Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Process started.");

        // Read module parameters
        String action = mpget("action");
        String ssInstallationID = mpget("ssInstallationID");
        String ssAdapterURL = mpget("ssAdapterURL");
        boolean normalize = (mpget("normalize") != null && mpget("normalize").equalsIgnoreCase("true"));
        boolean killLastChar = (mpget("kill.last.char") != null && mpget("kill.last.char").equalsIgnoreCase("true"));

        // Start process
        InputStream is = null;
        ByteArrayOutputStream baos = null;

        if (action.equalsIgnoreCase("EDIFACTtoXML")) {
            Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Required action: EDIFACTtoXML.");

            // Take the message payload
            byte[] ba = msg.getDocument().getContent();
            // Kill last char
            if (killLastChar) {
                Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Killing last character.");
                StringBuffer sb = new StringBuffer();
                for (int i = 0; i < ba.length - 1; i++) {
                    try {
                        sb.append((char) ba[i]);
                        byte[] ba2 = sb.toString().getBytes();
                        is = new ByteArrayInputStream(ba2);
                    } catch (Exception e) {
                        Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error during kill of last character. \n" + e);
                    }
                }
            } else {
                is = new ByteArrayInputStream(ba);
            }
        }

        // Normalization process
        if (normalize) {
            Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Normalization required.");
            try {
                EDIFACTNormalizer en = new EDIFACTNormalizer();
                baos = new ByteArrayOutputStream();
                en.normalize(is, baos, true);
                is = new ByteArrayInputStream(baos.toByteArray());
            } catch (Exception e) {
                Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while normalizing EDIFACT to XML \n" + e);
            }
        }
    }
}

```

```

    }
} else {
    Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Normalization not required.");
}

// Create converter instance
EDIFACTtoXML edixml = null;
try {
    if (ssInstallationID != null && ssAdapterURL != null) {
        edixml = new EDIFACTtoXML(ssInstallationID, ssAdapterURL);
        Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Converter created.");
    } else {
        edixml = new EDIFACTtoXML();
        Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Converter created (using default params).");
    }
} catch (Exception e) {
    Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while creating EDIFACTtoXML instance \n" + e);
}

// Convert
try {
    baos = new ByteArrayOutputStream();
    edixml.convert(is, baos);
    Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Conversion performed.");
} catch (Exception e) {
    Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while converting EDIFACT to XML.");
    logErrors(edixml.getErrorListener().getErrorMap());
    RecoverableException re =
        new RecoverableException(
            "Unparseable EDIFACT doc was found. See Recovered Adapter Audit Log for details. "
            + "Message sent to Error Handler procedure and original file moved to error folder.");
    throw new ModuleException(re);
}
}

else if (action.equalsIgnoreCase("XMLtoEDIFACT")) {
    Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Required action: XMLtoEDIFACT.");
    // Take the message payload
    is = new ByteArrayInputStream(msg.getDocument().getContent());

    // Create converter instance
    XMLtoEDIFACT xmledi = null;
    try {
        if (ssInstallationID != null && ssAdapterURL != null) {
            xmledi = new XMLtoEDIFACT(ssInstallationID, ssAdapterURL);
            Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Converter created.");
        } else {
            xmledi = new XMLtoEDIFACT();
            Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Converter created (using default params).");
        }
    } catch (Exception e) {
        Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while creating XMLtoEDIFACT instance \n" + e);
    }

    // Convert
    try {
        baos = new ByteArrayOutputStream();
        xmledi.convert(is, baos);
        Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, auditStr + "Conversion performed.");
    } catch (Exception e) {
        Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while converting EDIFACT to XML.");
        logErrors(xmledi.getErrorListener().getErrorMap());
        // TODO: chiamare qui EJB di error handling verso WAS
        throw new ModuleException(auditStr + "Error during XML to EDIFACT conversion. See message audit log for details.");
    }
}

else {
    throw mEx = new ModuleException(auditStr + "No 'action' parameter supplied (must be either 'XMLtoEDIFACT' or 'EDIFACTtoXML'.")
}

// Insert new payload
try {
    msg.getDocument().setContent(baos.toByteArray());
} catch (Exception e) {
    Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while setting new payload \n" + e);
}

```

```

    }
    return inputModuleData;
}

private String mpget(String pname) {
    return mc.getContextData(pname);
}

private String ex2str(Exception e) {
    StringWriter strWr = new StringWriter();
    e.printStackTrace(new PrintWriter(strWr));
    return strWr.toString();
}

private boolean moveFile() {
    String fileName = msg.getMessageProperty("http://sap.com/xi/XI/System/File", "FileName");
    String fileDir = msg.getMessageProperty("http://sap.com/xi/XI/System/File", "Directory");
    try {
        File origFile = new File(fileDir+fileName);
        File errFile = new File(fileDir+fileName+".err");
        return origFile.renameTo(errFile);
    } catch (Exception e) {
        Audit.addAuditLogEntry(amk, AuditLogStatus.ERROR, auditStr + "Error while moving file to error dir." + e);
    }
    return false;
}

private void logErrors(HashMap errors) {
    if (!errors.isEmpty()) {
        for (Iterator iter = errors.keySet().iterator(); iter.hasNext(); ) {
            String key = (String) iter.next();
            String value = (String) errors.get(key);
            Audit.addAuditLogEntry(amk, AuditLogStatus.WARNING, auditStr + key + " " + value);
        }
    }
}

public void ejbRemove() {
}

public void ejbActivate() {
}

public void ejbPassivate() {
}

public void setSessionContext(SessionContext context) {
    myContext = context;
}

public void ejbCreate() throws CreateException {
}

private ModuleContext mc;
private final String auditStr = "*** EDIFACT Handler - ";
private SessionContext myContext;
}

```

Note: To have your module resolve the XML Converters Java classes, you'll need to open `application-j2ee-engine.xml` file of your EAR project, and on "Expert Settings" put the path to the main jar (e.g. on a Linux system, this could be `/opt/XMLConverters3.0/bin/XMLConverters.jar`)

EDIFACTtoXML Code

```

/*
 * Created on 27-nov-2006
 *
 * To change the template for this generated file go to
 * Window>Preferences>Java>Code Generation>Code and Comments
 */
package biz.talentlab.helpers;
import com.ddtek.xmlconverter.*;

import javax.xml.transform.ErrorListener;
import javax.xml.transform.stream.*;
import java.io.*;

```

```

/**
 * Created on 27-nov-2006
 *
 */
public class EDIFACTtoXML {

    private String adapterURL = "adapter:EDI:encoding=iso-
8859-1:val=no:decode=no:field=no:seg=no:tbl=no:typ=no:opt=yes:count=no:eol=no:group=yes";
    // "adapter:EDI:val=no:decode=no:field=no:seg=no:tbl=no:typ=no:opt=yes:count=no";
    private Converter conv;
    private EDIFACTErrorListener errorListener;

    public EDIFACTtoXML() {
        errorListener = new EDIFACTErrorListener();
    }

    public EDIFACTtoXML(String _adapterURL) {
        adapterURL = _adapterURL;
        errorListener = new EDIFACTErrorListener();
    }

    public void convert(InputStream is, OutputStream os) throws Exception {
        // From EDI to XML
        StreamSource ss = new StreamSource((InputStream)is);
        StreamResult sr = new StreamResult((OutputStream)os);

        conv = ConverterFactory.newInstance().newConvertToXML(adapterURL);
        conv.convert(ss, sr, errorListener);
    }

    public EDIFACTErrorListener getErrorListener () {
        return errorListener;
    }
}

XMLtoEDIFACT Code
import com.ddtek.xmlconverter.*;
import javax.xml.transform.stream.*;
import java.io.*;

/**
 * Created on 27-nov-2006
 *
 * To change the template for this generated type comment go to
 * Window>Preferences>Java>Code Generation>Code and Comments
 */
public class XMLtoEDIFACT {
    private String adapterURL = "adapter:EDI:val=no:decode=no:field=no:seg=no:tbl=no:typ=no:opt=yes:count=no";
    private Converter conv;
    private EDIFACTErrorListener errorListener;

    public XMLtoEDIFACT() {
        errorListener = new EDIFACTErrorListener();
    }

    public XMLtoEDIFACT(String _adapterURL) {
        adapterURL = _adapterURL;
        errorListener = new EDIFACTErrorListener();
    }

    public void convert(InputStream is, OutputStream os) throws Exception {

        // From EDI to XML
        StreamSource ss = new StreamSource(is);
        StreamResult sr = new StreamResult();
        sr.setOutputStream(os);

        conv = ConverterFactory.newInstance().newConvertFromXML(adapterURL);
        conv.convert(ss, sr, errorListener);
    }

    public EDIFACTErrorListener getErrorListener() {
        return errorListener;
    }
}

```

EDIFACTErrorListener Code

```

/*
 * Created on 27-feb-2007
 *
 * To change the template for this generated file go to
 * Window>Preferences>Java>Code Generation>Code and Comments
 */
package biz.talentlab.helpers;

import java.util.HashMap;
import java.util.Vector;

import javax.xml.transform.ErrorListener;
import javax.xml.transform.TransformerException;

/**
 * Created on 27-nov-2006
 *
 * To change the template for this generated type comment go to
 * Window>Preferences>Java>Code Generation>Code and Comments
 */
public class EDIFACTErrorListener implements ErrorListener {

    private HashMap errorMap;
    private int errnr = 0;

    public EDIFACTErrorListener() {
        errorMap = new HashMap();
    }

    public void error(TransformerException arg0) throws TransformerException {
        errorMap.put(formatSeverity("ERROR"), formatExc(arg0));
    }

    public void fatalError(TransformerException arg0) throws TransformerException {
        errorMap.put(formatSeverity("FATAL"), formatExc(arg0));
    }

    public void warning(TransformerException arg0) throws TransformerException {
        errorMap.put(formatSeverity("WARNING"), formatExc(arg0));
    }

    public HashMap getErrorMap() {
        return errorMap;
    }

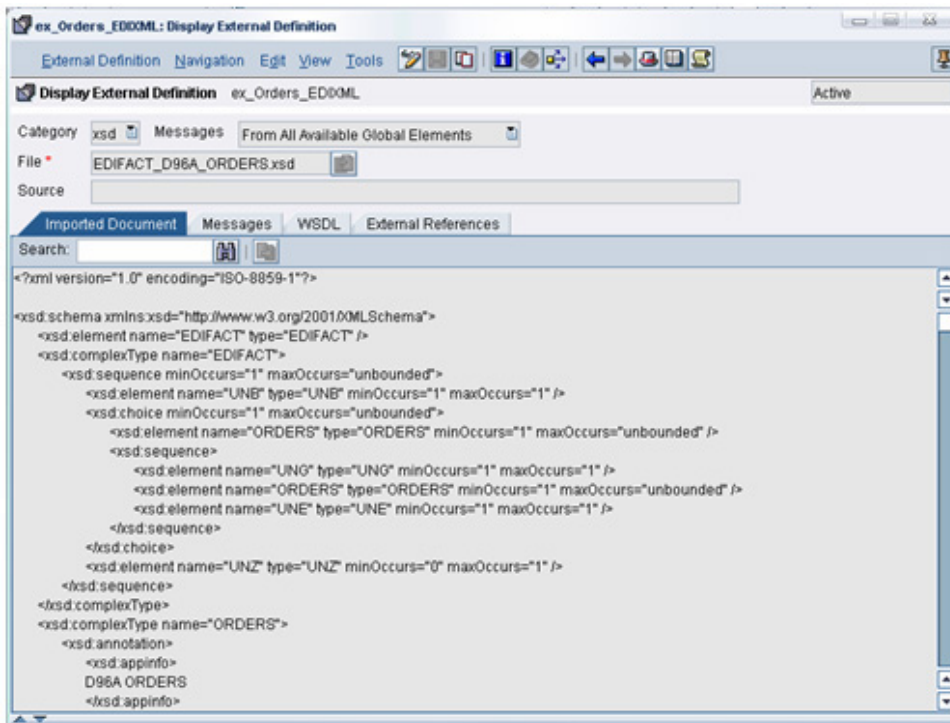
    private String formatExc(TransformerException e) {
        StringBuffer sb = new StringBuffer();
        sb.append(e.toString());
        Throwable cause = e.getCause();
        while (cause != null) {
            sb.append(" >>> " + cause.toString());
            cause = cause.getCause();
        }
        return sb.toString();
    }

    private String formatSeverity(String severity) {
        errnr++;
        return "(" + errnr + ") " + severity;
    }
}
XI

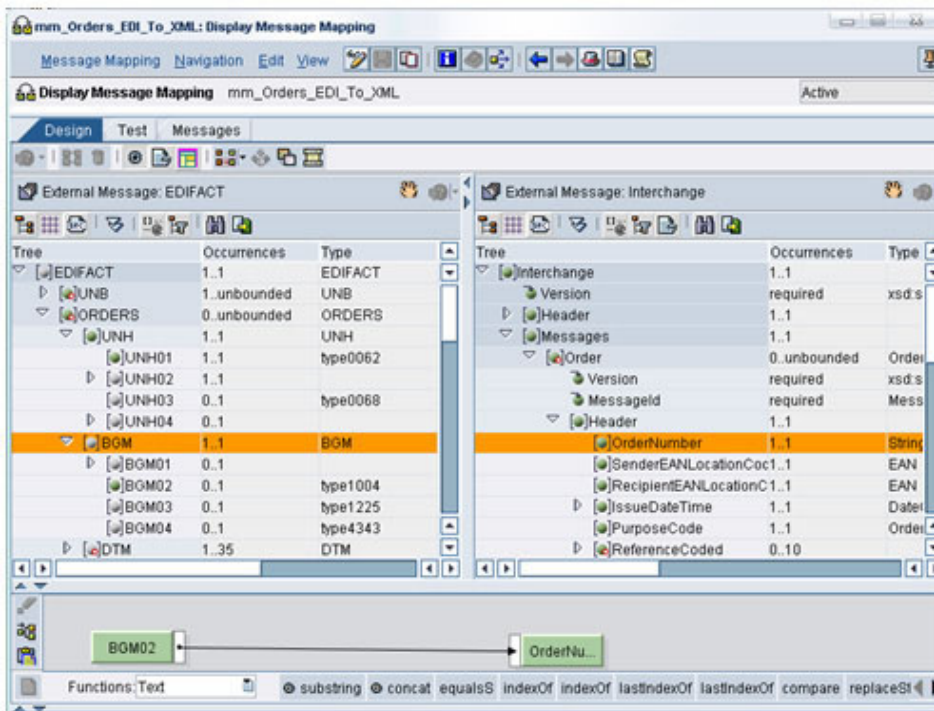
```

Design—EDIFACT XSD Importing and Mapping

Import the EDIFACT XSD generated by Stylus Studio in SAP XI (external definition)



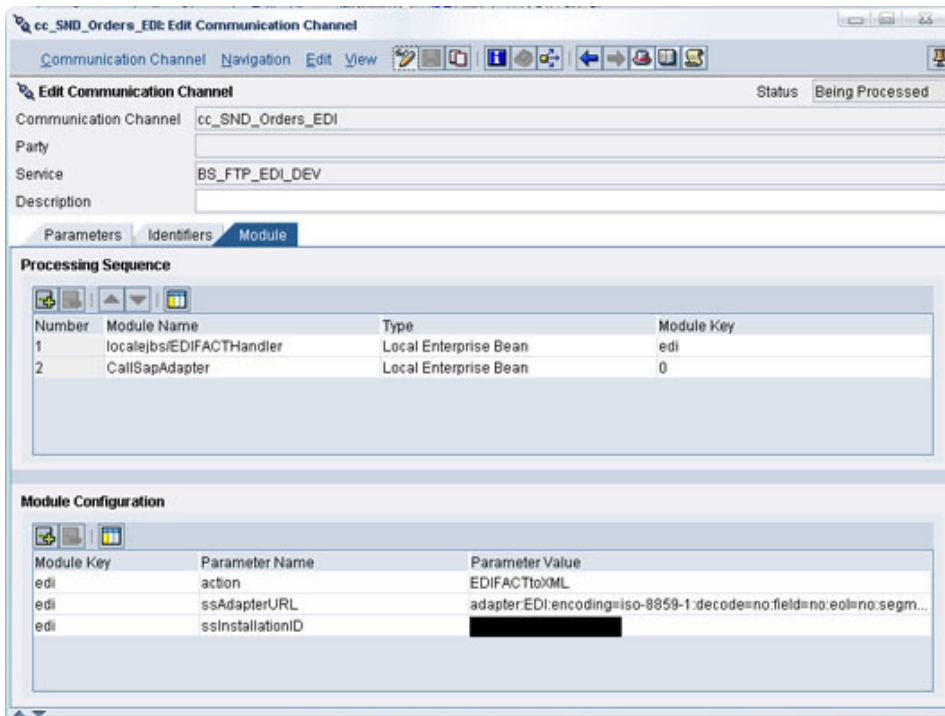
Once imported the XSD it will be used as a document for mapping



Directory—Module Configuration

After you've deployed your EDIFACT Handles module, you can call it from a module within any converter that runs on top of the SAP XI Converters Framework. You can call the service either at the sender or receiver side.

In the SAP XI Integration Directory, create a communication channel, choose a converter type (for instance FILE or MAIL), and maintain the appropriate parameters. Change to tab Module to define a local Enterprise Bean. Maintain localejbs/EDIFACTHandler as the module name, and set the parameter "edi" to the service that you deployed.



The EDIFACT Handler module parameters are:

- > action : type of conversion (EDIFACT to XML or vice versa)
- > XML Converters URL : parameter string (URL) used by DataDirect XML Converters to convert the document from one syntax to another
- > ssInstallationID : parameter used to validate the Stylus Studio installation

PROGRESS SOFTWARE

Progress Software Corporation (NASDAQ: PRGS) is a global software company that enables enterprises to be operationally responsive to changing conditions and customer interactions as they occur. Our goal is to enable our customers to capitalize on new opportunities, drive greater efficiencies, and reduce risk. Progress offers a comprehensive portfolio of best-in-class infrastructure software spanning event-driven visibility and real-time response, open integration, data access and integration, and application development and management—all supporting on-premises and SaaS/cloud deployments. Progress maximizes the benefits of operational responsiveness while minimizing IT complexity and total cost of ownership.

WORLDWIDE HEADQUARTERS

Progress Software Corporation, 14 Oak Park, Bedford, MA 01730 USA
Tel: +1 781 280-4000 Fax: +1 781 280-4095 On the Web at: www.progress.com

For regional international office locations and contact information, please refer to the Web page below:
www.progress.com/worldwide

Progress, Data Direct, DataDirect XML Converters, Stylus Studio and Business Making Progress are trademarks or registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and other countries. Any other trademarks contained herein are the property of their respective owners. Specifications subject to change without notice.

© 2007, 2010 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.
Rev. 06.10 | 6525-000000

