

## Priority SFDC 3

Introduction .....	2
New features.....	2
SFDC Design in Priority.....	3
Forms .....	3
Columns .....	4
Triggers.....	6
The Form Definition Feed .....	9
Provisioning the device .....	9
Updating the Form Definitions on the device.....	9
User Permissions.....	9
Appendices.....	11
The Feed SQL.....	11
The XML Feed Output .....	13
The provisioning service .....	14

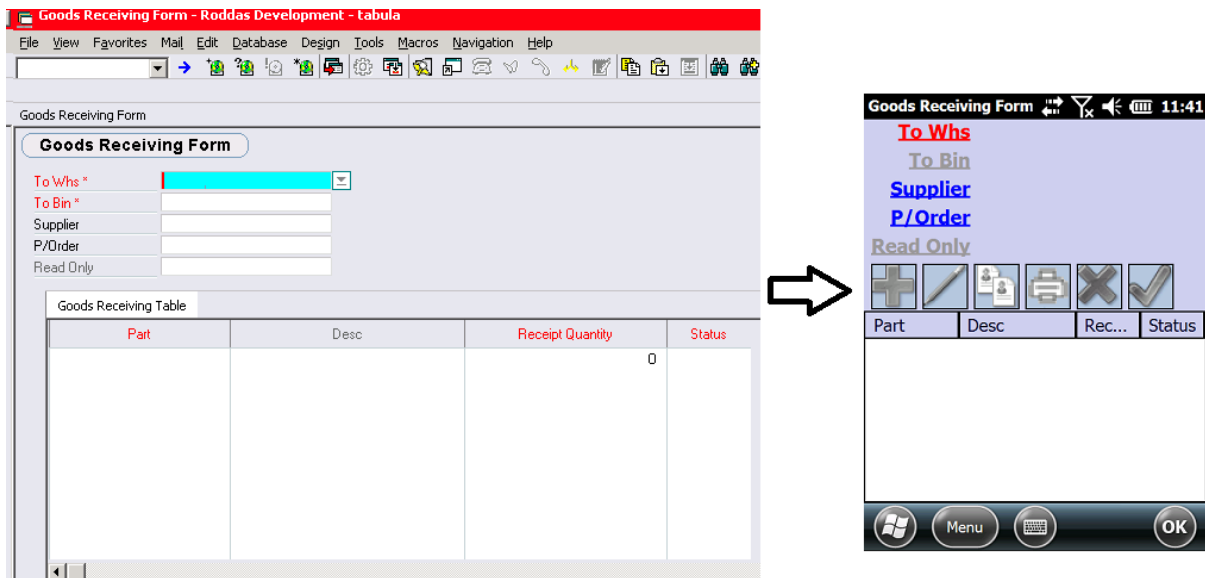
# Introduction

## New features

SFDC 3 introduces a number of new features that have been requested, both internally and by customers. Many of these features have already been successfully deployed within the PDA3 framework.

### Develop transactions in Priority

Arguably the most exciting of the new features is that version 3 implements a Form Definition Feed that exports Priority Forms to the device. This should reduce the development life-cycle of sfdc projects and mean that transactions can be designed and maintained by project managers.



### Hot Updates

As a consequence of making the form definitions an XML feed we are now able to update the transactions on a device by simply downloading the new form definition. Previously this had required a recompile and re-distribution of the entire project, so this should dramatically reduce development time.

### Provisioning

SFDC now uses the same provisioning system as the PDA project. Provisioning creates the user environment (like a windows profile) where images, form definitions and customisations are cached. The provisioning process also checks the emerge server to ensure the user has a valid licence.

### Priority Permissions

The Provisioning system provides users with a named licence, which is now being validated against the standard Form permissions in Priority.

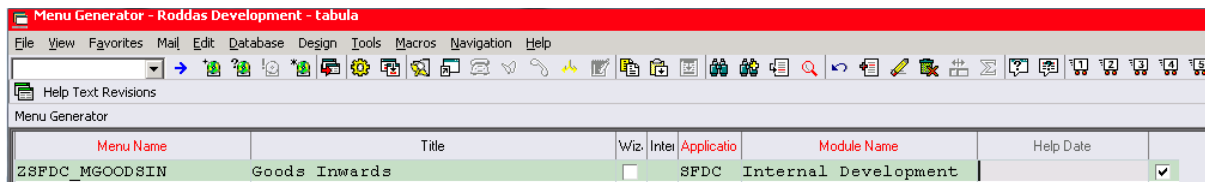
### Printing / 2d Barcode Scanning

Now supported!

# SFDC Design in Priority

## Forms

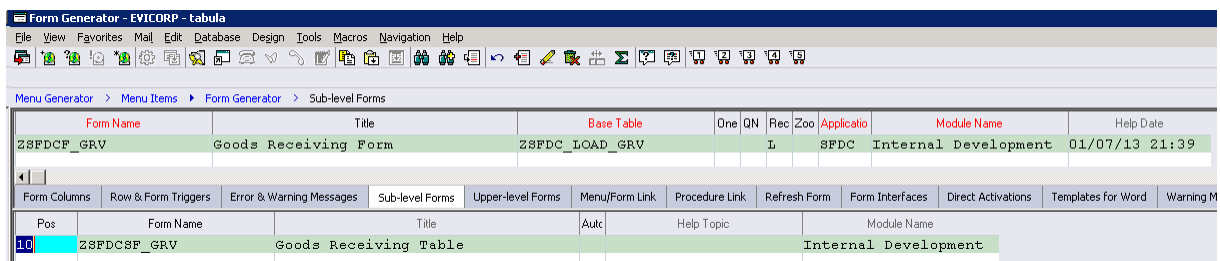
SFDC forms are identified by their inclusion on an SFDC menu (the tick box)



Menu Name	Title	WIZ	Inter	Applicatio	Module Name	Help Date
ZSFDC_MGOODSIN	Goods Inwards		<input checked="" type="checkbox"/>	SFDC	Internal Development	

Once 'Is SFDC' is ticked on the menu the menu item and all the forms it contains are included in the form definition feed.

SFDC forms must include an upper level form and \*ONE\* sub level (there is only one sub-level on the device).



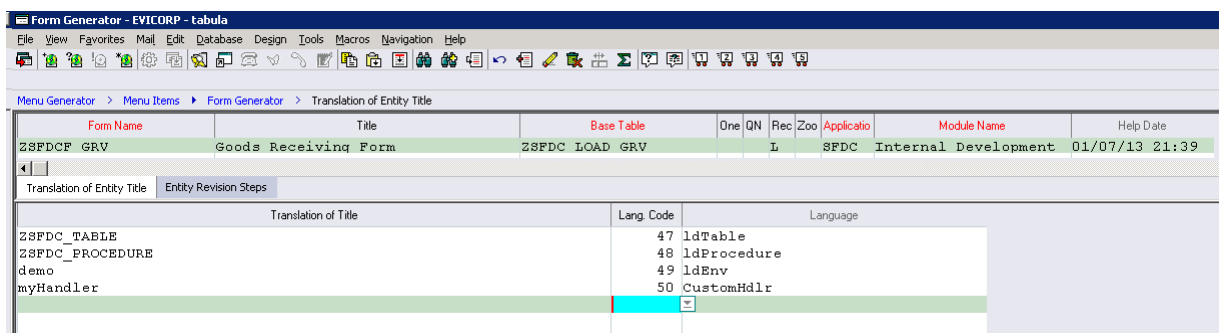
Form Name	Title	Base Table	One	QN	Rec	Zoo	Applicatio	Module Name	Help Date
ZSFDCP_GRV	Goods Receiving Form	ZSFDC_LOAD_GRV			<input checked="" type="checkbox"/>		SFDC	Internal Development	01/07/13 21:39

Form Columns	Row & Form Triggers	Error & Warning Messages	Sub-level Forms	Upper-level Forms	Menu/Form Link	Procedure Link	Refresh Form	Form Interfaces	Direct Activations	Templates for Word	Warning M
Pos	Form Name	Title	Aut	Help Topic	Module Name						
10	ZSFDCSF_GRV	Goods Receiving Table			Internal Development						

## Form Properties

The upper level form may define properties that are exported to the form definition feed. These are defined in the 'translation of entity title' form.



Translation of Title	Lang. Code	Language
ZSFDC_TABLE	47	ldTable
ZSFDC_PROCEDURE	48	ldProcedure
demo	49	ldEnv
myHandler	50	CustomHdlr

**ldTable:** The interim table to load data from the device into for this transaction

**ldProcedure:** The name of the Priority Procedure that will be executed to load data from the interim table

Note that if ldTable or ldProcedure are not specified that the transaction on the device will not be postable.

**ldEnv:** OPTIONAL: The environment in which to populate the ldTable interim table and run the ldProcedure procedure.

**CustomHdlr:** Optional: Defines the class within the cHandler.dll library that will be instantiated to override the default events provided by the framework.

Note: Custom handlers override the basic functionality of the sfdc.net framework, allowing us to write/modify customer specific forms / logic without rebuilding the framework or redistributing the base executable. These handlers are added to the cHandler.dll, and the dll is loaded to the [http://mobile.\\*](http://mobile.*) domain. SFDC device will attempt to update this dll from the server when the user environment is initialised on the device.

## Columns

### Mandatory / Read Only / Hidden Columns

Columns data is exported to the form definition feed based on settings in the Priority form designer.

**Mandatory:** A container (form or table) may not be submitted unless all of the fields marked as mandatory have been provided. Mandatory columns are designated on the device by having their label displayed in RED.

**Read Only:** Read only columns are displayed greyed out and will not accept control focus. They may however be updated programmatically with the “SELECT ... INTO :\$.COLUMN” syntax (see triggers) or by accessing the column from the framework in a customer handler. Note that read-only columns are not included in the loading data by default.

**Hidden Columns:** Hidden columns work in the same way as read-only columns, in that they can be modified by triggers or by a custom handler, but are not displayed on the device.

### Column Width

Lower level form Column Widths (i.e. in the Table) are based on the Priority column width. You may wish to override the width defined in the base table. You can do this by defining a form column extension and then specifying the column width manually.

The screenshot shows the 'Form Generator - EVICORP - tabula' application. The main window displays a table with columns: Form Name, Title, Base Table, One, QN, Rec, Zoo, Applicatio, and Module Na. The 'Form Name' column contains 'ZSFDCSF\_GRV', 'Title' contains 'Goods Receiving Table', 'Base Table' contains 'ZSFDC\_LOAD\_GRV', and 'Applicatio' contains 'SFDC Internal Deve'. Below this table, there are several tabs: 'Form Columns', 'Row & Form Triggers', 'Error & Warning Messages', 'Sub-level Forms', 'Upper-level Forms', 'Menu/Form Link', 'Procedure Link', 'Refresh Form', 'Form Interfaces', and 'Direct A'. The 'Form Columns' tab is selected, showing a table with columns: Form Column Name, Column Name, Table Name, Column, Pos, Hid, Rea, Boo, Type, Width, Dec, Full, Width ir, Sort Priori, Sort, Cov, Trigi, and Expi. The 'Form Column Name' column contains 'PARTNAME', 'Column Name' is empty, 'Table Name' is empty, 'Column' is '0', 'Pos' is '1.0', 'Hid' is 'M', 'Rea' is 'M', 'Boo' is 'M', 'Type' is 'M', 'Width' is '15' (highlighted with a red box), 'Dec' is '0', 'Full' is '0', 'Width ir' is '0', 'Sort Priori' is '0', 'Sort' is '0', 'Cov' is '0', 'Trigi' is '0', and 'Expi' is '0'. Below the table, there are tabs: 'Form Column Triggers', 'Form Column Extension', 'Help Text', 'Interfaces for Column', 'Translation-Revised FormColTitle', 'Translation-Table Column Title', and 'Form Column Revision Steps'. The 'Form Column Extension' tab is selected, showing a form with fields: Column Type (CHAR), Expression/Condition (ZSFDC\_LOAD\_GRV.PARTNAME), Target Form Name, and Target Form Title.

Context sensitive help is now available on the device by pressing ‘.’

This displays the Help message for the focused column

**Form Generator - EVICORP - tabula**

File View Favorites Mail Edit Database Design Tools Macros Navigation Help

Menu Generator > Menu Items > Form Generator > Form Columns > Help Text

Form Name	Title	Base Table
ZSFDCF GRV	Goods Receiving Form	ZSFDC LOAD GRV

Form Columns | Row & Form Triggers | Error & Warning Messages | Sub-level Forms | Upper-level Forms | Menu/Form Link

Form Column Name	Column Name	Table Name	Column	Pos	Hide	Rea	Boo	Type
TOWARHSNAME	TOWARHSNAME	ZSFDC_LOAD_GRV	0	10	<input type="checkbox"/>	M	<input type="checkbox"/>	CHAR

Form Column Triggers | Form Column Extension | Help Text | Interfaces for Column | Translation-Revised FormColTitle | Tr

Text

The Warehouse to receive the delivery.

## Column Properties

Additional (non-Priority standard) column properties are exported to the form definition feed. These are defined in the 'Translation-Revision FormColTitle' sub level form.

Form Generator - EVICORP - tabula

File View Favorites Mail Edit Database Design Tools Macros Navigation Help

Menu Generator > Menu Items > Form Generator > Sub-level Forms > Form Generator > Form Columns > Translation-Revised FormColTitle

Form Name	Title	Base Table	One QN	Rec Zoo	Application	Module Name
ZSFDCSF_GRP	Goods Receiving Table	ZSFDC_LOAD_GRP			SFDC	Internal Development

Form Columns Row & Form Triggers Error & Warning Messages Sub-level Forms Upper-level Forms Menu/Form Link Procedure Link Refresh Form Form Interfaces Direct A

Form Column Name	Column Name	Table Name	Column	Pos	Hide	Rea	Boo	Type	Width	Dec	Full	Width in	Sort Prior	Sort	Cov	Trig	Exp
PARTNAME			0	10	<input type="checkbox"/>	M	<input type="checkbox"/>		15		<input type="checkbox"/>	0	0			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Form Column Triggers Form Column Extension Help Text Interfaces for Column Translation-Revised FormColTitle Translation-Table Column Title Form Column Revision Steps

Translation of Title	Lang. Code	Language
PARTNAME	45	ZdBarcode
^[0-9]+\$	46	Regex

**2dBarcode:** OPTIONAL: Used to match a 2d name pair scan to a form Column Title. E.G. Your 2d barcode contains a field called 'WARHS', but your Form Column Name is 'TOWARHS'. By setting the 2dBarcode property to 'WARHS' the 'TOWARHS' will be populated when 'WARHS' is scanned.

**Regex:** OPTIONAL (but recommended!): Contains the regular expression with which to validate user entry for this column. Regex validation is performed *\*BEFORE\** the CHECK-FIELD, and can be used to sanity check the user entry before making a database request.

## Alternate Entry

Columns on the device may be populated by the user with the on-screen keyboard, physical keyboard, or scanning a barcode.

Additionally we provide some default alternate entry methods.

Alternate entry mode is initialised by pressing space.

**CHOOSE-FIELD:** A choose-field can be defined in Priority to select possible user entries into a drop down list (see triggers).

**Numeric Fields:** Numeric field are automatically overridden with a calculator screen entry style.

**Custom Entry Form:** A custom handler can contain user-specific forms for other types of data entry and integrations with other systems (weigh scales, signatures, whatever)

## Triggers

Note that trigger sql written in Priority is actually being run on the device (after conversion to MSSQL). Consequently not every priority sql function has currently been exported, but there is a mechanism by which additional functions can be added should they be required.

## Syntax

The following Priority syntax conventions are supported:

- **:\$COLUMN** : The value contained in the column 'COLUMN' in the current container (Form or table).
- **:\$\$.COLUMN**: The value contained in the column 'COLUMN' in the parent container (The upper level form). Note that a table container will be inactive until all the upper level references contained in its triggers are populated.
- **FROM DUMMY;** : Dummy table supported.
- **PAR1 / <P1>**: CHECK-FIELD triggers may pass parameters to an ERRMSG.
- **:\$.@**: In a column trigger ':\$.@' refers to the currently focused column.

## Form Triggers

### PRE-FORM

A PRE-FORM trigger is used to populate data into the sub level form. The trigger is executed once all the upper level columns referred to (i.e. :\$\$COLUMN) in triggers of the sublevel form have been entered.

A PRE-FORM trigger may ONLY be used on a sub-level form.

The PRE-FORM trigger is optional.

Each record inserted into the sub-level form by the PRE-FORM will trigger a PRE-INSERT event before inserting data.

### Example:

```
SELECT PARTNAME, PARTDES  
INTO :$.PARTNAME, :$.PARTDES  
FROM PART, WARHSBAL, WAREHOUSES  
WHERE PART.PART = WARHSBAL.PART  
AND WARHSBAL.WARHS = WAREHOUSES.WARHS  
AND WAREHOUSES.WARHSNAME = :$.TOWARHSNAME  
AND WAREHOUSES.LOCNAME = :$.TOLOCNAME  
AND WARHSBAL.BALANCE > 0  
;
```

### PRE-INSERT

A PRE-INSERT trigger is used to populate default values into a new record.

A PRE-INSERT may be used on either the upper or lower level form.

### Example 1:

```
SELECT 'Goods' INTO :$.STATDES FROM DUMMY;
```

### Example 2:

```
SELECT SQL.DATE8 INTO :$.CURDATE FROM DUMMY;
```

## Column Triggers

### CHOOSE-FIELD

A CHOOSE-FIELD trigger will create a drop down list style alternate entry when the alt-entry key is pressed (currently space)

### Example:

```
SELECT DISTINCT WARHSDES, LOCNAME FROM WAREHOUSES  
WHERE WARHSNAME = :$.TOWARHSNAME;
```

### CHECK-FIELD

The CHECK-FIELD is triggered after Regex validation.

If a CHECK-FIELD is successful it will fire the POST-FIELD trigger for this column (if one exists).

It is possible to have multiple checks within a CHECK-FIELD, each with their own ERRMSG number defined in Priority

Execution of the trigger stops on the first error thrown and the related message is displayed to the user.

PARAM values can be defined and used in the displayed ERRMSG.

Error & Warning Messages	
Msg Number	Message
501	Sorry <P2>, but barcode '<P1>' is invalid.
502	Part '<P1>' does not exist in warehouse <P2> / <P3>.
0	

### Example:

```
SELECT 'Christine' INTO :TEST FROM DUMMY
;
SELECT :$.@, :TEST
INTO :PAR1, :PAR2
FROM DUMMY;
ERRMSG 501 WHERE :$.@ NOT IN (
SELECT PART.BARCODE
FROM PART)
;
SELECT PARTNAME, :$$TOWARHSNAME, :$$TOLOCNAME
INTO :PAR1, :PAR2, :PAR3
FROM PART
WHERE PART.BARCODE = :$.@
;
ERRMSG 502 WHERE :$.@ NOT IN (
SELECT PART.BARCODE
FROM PART, WARHSBAL, WAREHOUSES
WHERE WARHSBAL.PART = PART.PART
AND WARHSBAL.WARHS = WAREHOUSES.WARHS
AND WAREHOUSES.WARHSNAME = :$$TOWARHSNAME
AND WAREHOUSES.LOCNAME = :$$TOLOCNAME
AND WARHSBAL.BALANCE > 0)
;
```

### **POST-FIELD**

A POST-FIELD on a column is triggered after the column is updated.

The example below shows how a barcode that has been scanned into the PARTNAME column is substituted for the associated part name.

The example also shows how the read-only field 'PARTDES' is also updated to show the description for the scanned part.

### Example:

```
SELECT PARTNAME, PARTDES
INTO :$.@, :$.PARTDES
FROM PART
WHERE BARCODE = :$.PARTNAME;
```



## The Form Definition Feed

- The form definitions feed defines the properties of Priority forms that are to be exported to the device.
- The form definition feed is an xml feed written using the standard Priority Mobile XML Feed framework.
- The feed will be located at <http://mobile.yourserver.tld:8080/sfdc.ashx>

## Provisioning the device

Provisioning is the process whereby a user environment for a given company/user is created on the device.

Note that a device may have many environments cached, for both multiple users and multiple servers.

During the provisioning of a new user environment the user scans a provisioning code. The device connects to the provisioning service at <http://soti.emerge-it.co.uk/client/provision.xml> to retrieve the Priority Username / Server for that provision code (See Appendix 3).

The company/user profile folder is used to cache the form definitions, user settings, logo images and any custom dll handlers that have been downloaded.

Provisioning must be done on-line, as the process connects to the emerge servers to confirm that the user has a valid licence. The list of valid Usernames / servers is maintained by us. It is in effect a named licence.

An already provisioned user environment may be used off-line (without connecting to the provision service).

## Updating the Form Definitions on the device

When a user environment is first created the form definition feed is downloaded to the device and cached.

When forms definitions are updated in Priority it is necessary to update the form definition on the device.

To update form definitions go to Menu > About > Update on the device. Note that all transaction windows should be closed before updating.

Custom Handlers are automatically updated at application start-up.

## User Permissions

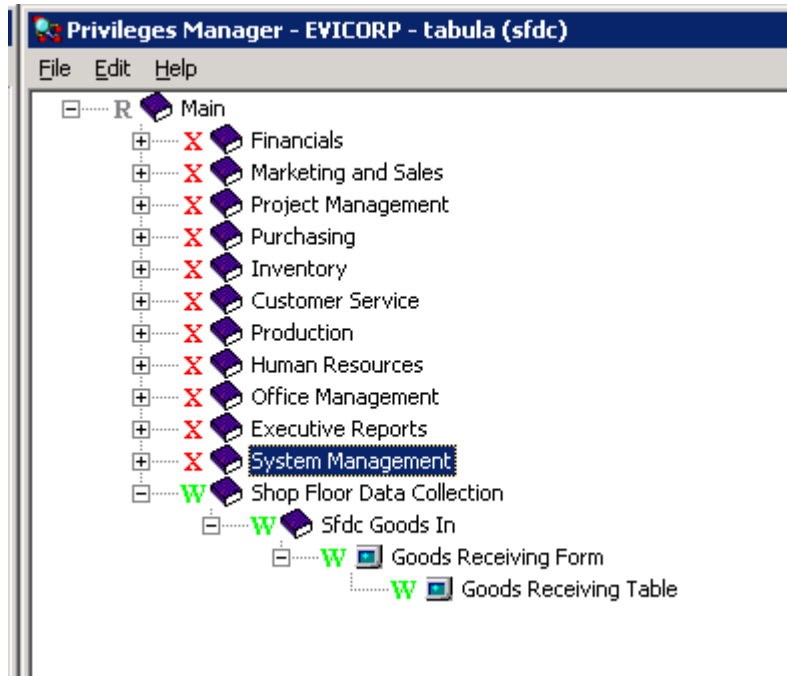
Appendix I shows the SQL query that generates the feed.

Note that the @user field is flagged as '–mandatory'

This means that the form definition feed MUST be called with the user parameter like this:

<http://mobile.yourserver.tld:8080/sfdc.ashx?user=PriorityUser>

The output from the feed (see Appendix 2) is filtered to show only forms that the user has write permission to in Priority.



This means you can create multiple Priority User groups with different level of access and assign your users to the relevant group leader using standard Priority permissions.

# Appendices

## The Feed SQL

```
USE [system]
declare @user varchar(32) --mandatory
set @user = 'service'
select '',(
    SELECT TITLE AS "@name",
    (SELECT TITLE AS "@name",
        dbo.FormProperty(menutable.FORM,'CustomHdlr') AS "@handler",
        dbo.FormProperty(menutable.FORM,'ldTable') AS "@ldTable",
        dbo.FormProperty(menutable.FORM,'ldProcedure') AS "@ldProcedure",
        dbo.FormProperty(menutable.FORM,'ldEnv') AS "@ldEnv",
        (SELECT '',
            (SELECT TriggerName AS "@trigger",
                dbo.FormTriggersSQL(menutable.FORM, TriggerName) AS "@sql"
            FROM dbo.FORMTRIGGERS(menutable.FORM)
            FOR xml path('triggers'), type),
            (SELECT CNAME AS "@name",
                CTITLE AS "@title",
                CPOS AS "@pos",
                CTYPE AS "@type",
                CWIDTH AS "@width",
                CREADONLY AS "@readonly",
                CHIDE AS "@hidden",
                MANDATORY AS "@mandatory",
                REGEX AS "@regex",
                BARCODE2D AS "@barcode2d",
                dbo.HelpText(menutable.FORM, CNAME) AS "@help",
                (SELECT TriggerName AS "@trigger",
                    dbo.TriggersSQL(menutable.FORM, CNAME, TriggerName) AS "@sql"
                FROM COLUMNTRIGGERS(menutable.FORM, CNAME)
                FOR xml path('triggers'), type)
            FROM dbo.FormColumns(menutable.FORM)
            FOR xml path('column'), type) ,
        (select '',
            (SELECT NUM as "@num",
                dbo.MessageText(menutable.FORM,NUM) as "@text"
            from TRIGMSG
            where T$EXEC = menutable.FORM
            FOR xml path('message'), type)
            FOR xml path('messages'), type)
        FOR xml path('form'), type),
        (SELECT '',
            (SELECT TriggerName AS "@trigger",
                dbo.FormTriggersSQL(dbo.FIRSTCHILDFORM(menutable.FORM), TriggerName) AS "@sql"
            FROM dbo.FORMTRIGGERS(dbo.FIRSTCHILDFORM(menutable.FORM))
            FOR xml path('triggers'), type),
            (SELECT CNAME AS "@name",
                CTITLE AS "@title",
                CPOS AS "@pos",
                CTYPE AS "@type",
                CWIDTH AS "@width",
                CREADONLY AS "@readonly",
                CHIDE AS "@hidden",
                MANDATORY AS "@mandatory",
                REGEX AS "@regex",
                BARCODE2D AS "@barcode2d",
                dbo.HelpText(dbo.FIRSTCHILDFORM(menutable.FORM), CNAME) AS "@help",
                (SELECT TriggerName AS "@trigger",
                    dbo.TriggersSQL(dbo.FIRSTCHILDFORM(menutable.FORM), CNAME, TriggerName) AS
"@sql"
                FROM COLUMNTRIGGERS(dbo.FIRSTCHILDFORM(menutable.FORM), CNAME)
                FOR xml path('triggers'), type)
            FROM dbo.FormColumns(dbo.FIRSTCHILDFORM(menutable.FORM))
            FOR xml path('column'), type) ,
        (select '',
            (SELECT NUM as "@num",
                dbo.MessageText(dbo.FIRSTCHILDFORM(menutable.FORM),NUM) as "@text"
            from TRIGMSG
            where T$EXEC = dbo.FIRSTCHILDFORM(menutable.FORM)
            FOR xml path('message'), type)
            FOR xml path('messages'), type)
        FOR xml path('table'), type)
    FROM dbo.v_ExecMenu AS menutable
    where FORM in (
        select FORM
        from v_SfdcForms
        where PARENT = exetable.T$EXEC
        and UGROUP = dbo.UserGroup(@user)
    )
)
```

```
        FOR xml path('interface'), type)
FROM dbo.v_SfdcMenu AS exetable
where exists (
    select FORM
    from dbo.v_SfdcForms
    where PARENT = exetable.T$EXEC
    and UGROUP = dbo.UserGroup(@user)
)
FOR xml path('menu'), type )
FOR xml path('sfdc'), type
```

## The XML Feed Output

```
<sfdc>
  <menu name="Sfdc Goods In">
    <interface name="Goods Receiving Form" handler="myHandler" ldtTable="ZSFDC_TABLE"
    ldtProcedure="ZSFDC_PROCEDURE">
      <form>
        <triggers trigger="PRE-INSERT" sql=" SELECT SQL.DATE8 INTO :$.CURDATE FROM DUMMY;" />
        <column name="TOWARHSNAME" title="To whs" pos="10" type="CHAR" width="4" mandatory=""
        help="The warehouse to receive the delivery.">
          <triggers trigger="CHOOSE-FIELD" sql=" SELECT DISTINCT WARHSDS, WARHSNAME FROM
        WAREHOUSES;" />
          <triggers trigger="CHECK-FIELD" sql=" ERRMSG 502 WHERE :$.@ NOT IN ( SELECT DISTINCT
        WARHSNAME FROM WAREHOUSES WHERE WARHS &lt;&gt; 0 );" />
        </column>
        <column name="TOLOCNAME" title="To Bin" pos="20" type="CHAR" width="16" mandatory="">
          <triggers trigger="CHOOSE-FIELD" sql=" SELECT DISTINCT WARHSDS, LOCNAME FROM WAREHOUSES
        WHERE WARHSNAME = :$.TOWARHSNAME;" />
          <triggers trigger="CHECK-FIELD" sql=" ERRMSG 501 WHERE :$.@ NOT IN ( SELECT LOCNAME FROM
        WAREHOUSES WHERE WARHSNAME = :$.TOWARHSNAME );" />
        </column>
        <column name="SUPNAME" title="Supplier" pos="30" type="CHAR" width="32" />
        <column name="ORDNAME" title="P/Order" pos="40" type="CHAR" width="16" />
        <column name="READONLY" title="Read Only" pos="50" type="CHAR" width="8" readonly="" />
        <column name="CURDATE" title="Transaction Date" pos="70" type="DATE" width="8" hidden="" />
        <column name="USERLOGIN" title="USERLOGIN" pos="170" type="CHAR" width="20" hidden="" />
        <messages>
          <message num="501" text="Invalid bin location. 2nd line." />
          <message num="502" text="Invalid warehouse." />
        </messages>
      </form>
      <table>
        <triggers trigger="PRE-INSERT" sql=" SELECT 'Goods' INTO :$.STATDES FROM DUMMY;" />
        <triggers trigger="PRE-FORM" sql=" SELECT PARTNAME,PARTDES INTO :$.PARTNAME, :$.PARTDES
        FROM PART, WARHSBAL, WAREHOUSES WHERE PART.PART = WARHSBAL.PART AND WARHSBAL.WARHS =
        WAREHOUSES.WARHS AND WAREHOUSES.WARHSNAME = :$$TOWARHSNAME AND WAREHOUSES.LOCNAME = :$$TOLOCNAME
        AND WARHSBAL.BALANCE &gt; 0 AND PART.PARTNAME LIKE 'H%';" />
        <column name="PARTNAME" title="Part" pos="10" type="CHAR" width="15" mandatory=""
        regex="^[0-9]+$" barcode2d="PARTNAME">
          <triggers trigger="CHECK-FIELD" sql=" SELECT 'Christine' INTO :TEST FROM DUMMY ; SELECT
        :$.@, :TEST INTO :PAR1, :PAR2 FROM DUMMY; ERRMSG 501 WHERE :$.@ NOT IN ( SELECT PART.BARCODE FROM
        PART ); SELECT PARTNAME, :$$TOWARHSNAME, :$$TOLOCNAME INTO :PAR1, :PAR2, :PAR3 FROM PART WHERE
        PART.BARCODE = :$.@ ; ERRMSG 502 WHERE :$.@ NOT IN ( SELECT PART.BARCODE FROM PART, WARHSBAL,
        WAREHOUSES WHERE WARHSBAL.PART = PART.PART AND WARHSBAL.WARHS = WAREHOUSES.WARHS AND
        WAREHOUSES.WARHSNAME = :$$TOWARHSNAME AND WAREHOUSES.LOCNAME = :$$TOLOCNAME AND WARHSBAL.BALANCE
        &gt; 0 );" />
          <triggers trigger="POST-FIELD" sql=" SELECT PARTNAME, PARTDES INTO :$.@, :$.PARTDES FROM
        PART WHERE BARCODE = :$.PARTNAME;" />
        </column>
        <column name="PARTDES" title="Desc" pos="15" type="CHAR" width="20" readonly="" />
        <column name="TQUANT" title="Receipt Quantity" pos="20" type="INT" width="17" mandatory=""
        />
        <column name="STATDES" title="Status" pos="30" type="CHAR" width="6" mandatory="">
          <triggers trigger="CHOOSE-FIELD" sql=" SELECT CUST, CUSTNAME FROM CUSTOMERS WHERE
        STATUSFLAG = 'Y';" />
        </column>
        <column name="TOLOCNAME2" title="Receiving Location" pos="999" type="CHAR" width="16"
        hidden="" />
        <column name="TOWARHSNAME2" title="Receiving Warehouse" pos="999" type="CHAR" width="4"
        hidden="" />
        <column name="ORDI" title="ORDI" pos="999" type="INT" width="13" hidden="" />
        <column name="ORDNAME1" title="PO Number for line" pos="999" type="CHAR" width="16"
        hidden="" />
        <messages>
          <message num="501" text="Sorry &lt;P2&gt;; but barcode '&lt;P1&gt;' is invalid." />
          <message num="502" text="Part '&lt;P1&gt;' does not exist in warehouse &lt;P2&gt;; /
        &lt;P3&gt;;" />
        </messages>
      </table>
    </interface>
  </menu>
</sfdc>
```

## The provisioning service

```
<?xml version="1.0" encoding="utf-8" ?>
<devices>
  <user ProvisionString="test">
    <username>test</username>
    <url>http://10.10.10.150:8080/</url>
  </user>
  <user ProvisionString="5C78CCB6">
    <username>demo</username>
    <url>http://dev.emerge-it.co.uk:8080/</url>
  </user>
  <user ProvisionString="123">
    <username>don_w</username>
    <url>http://mobile.virgintrausswater.com:8080/</url>
  </user>
  <user ProvisionString="123-456-789">
    <username>driver</username>
    <url>http://mobile.roddas.co.uk:8080/</url>
  </user>
  ...
</devices>
```