DESERT Workflow Tutorial

This document gives a detailed description for how to use the DESERT extension package. To use this tutorial, the OTIF core and OTIF DESERT extension package needs to be installed and the instructions in the *"Getting Started"* and *"Getting Started with the DESERT tool chain"* shortcut files should be followed to run the framework.

We start from the point when the OTIF framework has just started.

Look at the DESERT workflow to understand how it works

This part gives a detailed overview about the DESERT workflow model and how it works. It describes the different elements of the DESERT tool chain.

This section is useful for those who wants to get to know the DESERT tool chain and for those who wants to modify the DESERT tool chain.

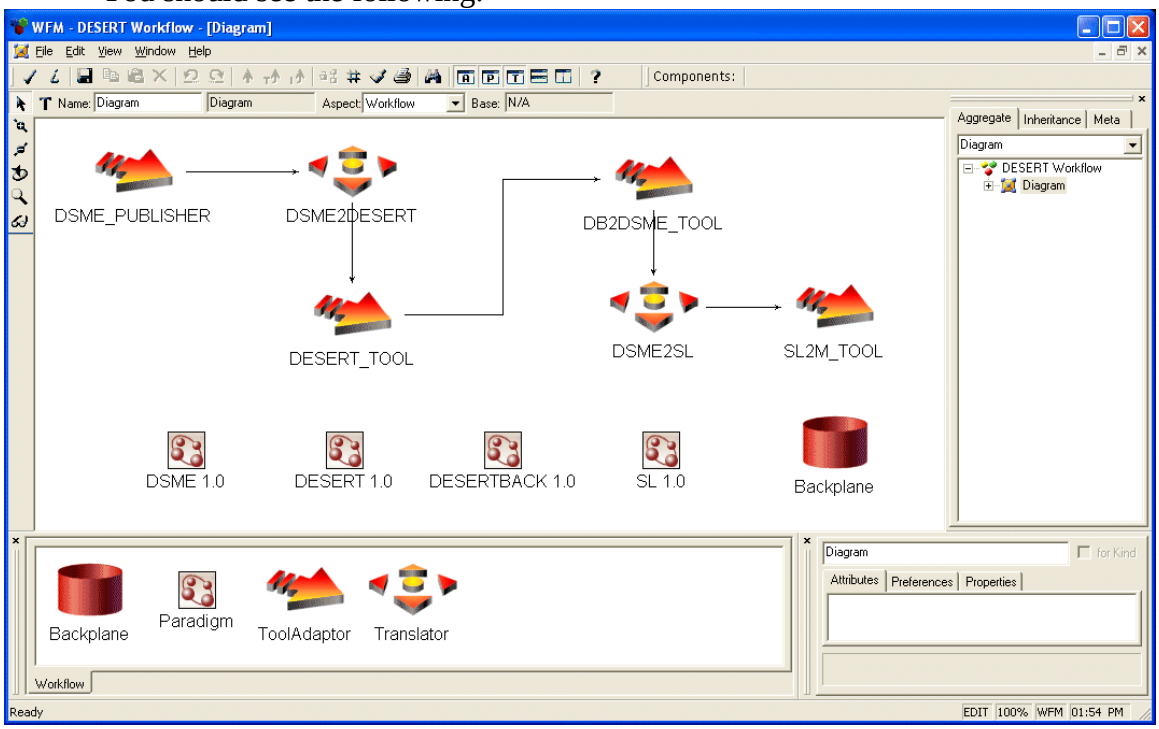
1. Open the DESERT workflow model file in GME.

To start to work with the DESERT workflow model please import the *DESERT_Workflow.xml* file from the Documents folder of the OTIF framework into the GME application or open the *DESERT Workflow* file under *Start Menu/Programs/OTIF/Workflows* folder. GME is the workflow model editor.

Note: if you have problem with importing the XML file please check that the *WFM.xmp* workflow paradigm registered correctly from the Documents folder of the OTIF framework in GME.

Click on the *DESERT Workflow* element in the GME Object Inspector window and open the *Diagram* element.

You should see the following:



DESERT workflow diagram in GME

The model shows the ESML workflow that contains two translators and four types of tool adaptor.

The tool chain has four tool adaptors:
(1) *DSME_PUBLISHER*,
(2) *DESERT_TOOL*,
(3) *DB2DSME_TOOL and*
(4) *SL2M_TOOL*.

Note: every tool adaptor is unique program.

The arrows on the model represent document flows. Each document flow has a type, which is assigned from the set of available document types enumerated at the bottom as “Paradigm” atoms (e.g. *DSME 1.0*, etc.). In GME, these are sets, containing the document flow edges as elements (and this containment relationship represents the typing information). Using the set-visualization mode of GME, one can select a paradigm, and the window will show the shows belonging to that paradigm.

The *DSME_PUBLISHER* can publish *DSME* documents, which will be translated by the *DSME2DESERT* translator into *DESERT* documents. This translator translates the abstract representation of the design space captured in the fetch document into a representation for the design space exploration tool (*DESERT_TOOL*).

Thus, the translated documents can be fetched by the *DESERT_TOOL* tool adaptor. This tool shows the constraints that have been modeled and the size of the design space. The tool lets you apply the constraints selectively using the select box and *Apply* button on the toolbar. Application of constraints results in a pruning of the design space.

Upon exiting the tool publishes the pruned design space (*DESERTBACK)* document, which can be fetched by the *DB2DSME_TOOL* tool adaptor.

The *DB2DSME_TOOL* tool adaptor can fetch *DESERTBACK* documents and starts the design selector/configuration generation tool. This tool lets you browse the design configuration post pruning. You can choose any configuration and press *Generate* button to generate and send a particular configuration as a *DSME* document, which will be translated by the *DSME2SL* translator into *SL* document.

The *DSME2SL* translator generates a concrete representation of the build configuration that is suitable for instantiation in Simulink. The translator publishes *SL* documents, which may be fetched by the *SL2M_TOOL* tool adaptor.

The *SL2M_TOOL* tool adaptor can fetch *SL* documents and generates a Matlab M-script file from them. This script file when executed in Matlab, creates the final assembly Simulink model.

How to use this workflow configuration

The above description explained how the DESERT workflow works. This section describes how to use the OTIF framework elements to execute this workflow.

This section shows how to “interpret” this workflow model and send it to the server.

This model defines a specific configuration of OTIF. To make the Backplane work with this configuration we have to interpret it like any other models in GME.

2. To interpret this workflow model to register it in the Backplane.

To interpret this model, the *Interpret (i)* button must be activated in GME. After the interpretation, a popup window will appear and ask the password of the Backplane server. In the default installation you may leave it empty. But if you changed the password of the server you have to specify the correct password here.

If you are done you may press the OK button to send this workflow configuration to the Backplane server.

After this interpretation and registration process we are ready to use our new workflow model.

We are ready to play with the DESERT tool chain

This section describes how we can use the previously registered DESERT tool chain.

3. Let's start to use the DESERT tool chain.

After installing the DESERT extension package we can find a number of shortcut files in the *Start Menu/Programs/OTIF/Tooladaptors/DESERT Tool Chain* folder. Please look for the "*DSME_PUBLISHER*" shortcut.

Note: this type is defined in the workflow model.

Let's publish a document into the DESERT tool chain. First of all start the "*DSME_PUBLISHER"* shortcut file, which can publish a *DSME* document as we can see in the *DESERT* workflow model.

Specify the name of the document. You may change other parameters of the document but it is not necessary for now.

Note: please attach the corresponding .m files to the document.

Click on *Publish* button and select the *etc_compiler.mga* file from the *Samples* directory of the OTIF framework. This is a *DSME* type of MGA file, which will be translated by the *DSME2DESERT* translator as the DESERT workflow model says.

You may close the application now.

Note: the *etc_compiler.mga* file should be created first by importing the *etc_compiler.xml* file into GME and save it.

4. Fetch the translated *DESERT* document by the DESERT tool

Let us fetch the translated document to see the result. Start the "*DESERT_TOOL"* shortcut file, which can fetch *DESERT* documents.

It will automatically start the design space exploration tool, where you can apply different constraints.

When you finished you can close the tool and choose to send the output to the Backplane server as a *DESERTBACK* document.

You may close the application now.

5. Fetch the *DESERTBACK* document and choose the right configuration for you

Start the *"DB2DSME"* shortcut file, which can fetch *DESERTBACK* documents.

It will automatically start the design selector/configuration generation tool. You can choose any configuration and press *Generate* button to generate and send a particular configuration as a *DSME* document.

You may close the application now.

6. Fetch the translated *SL* document by the SL2M tool.

Start the *"SL2M"* shortcut file, which can fetch *SL* documents. It will automatically generate a Matlab M-script file and you may save it for later usage.

You may close the application now.

Note: if you change the names of the tool adaptor types or use different kind of DESERT workflow model you may not be able to use these shortcuts.