|  |  |  |
| --- | --- | --- |
|  | | |
|  | | |
|  | **TIF Workbench**  **User and Administration Guide**  Version 1.0.9 |  |
|  | | |
|  | | |
|  | | |
| 14 February 2014 | | |

White Space

Table of Contents

[1 Getting started 5](#_Toc380161454)

[1.1 Prerequisites 6](#_Toc380161455)

[1.2 Installation from FDO 6](#_Toc380161456)

[2 Environments 8](#_Toc380161457)

[2.1 Release 1.0.9 environment 8](#_Toc380161458)

[2.2 Creating an environment 8](#_Toc380161459)

[2.3 Opening an environment 10](#_Toc380161460)

[2.4 Environment settings 10](#_Toc380161461)

[2.4.1 Integrity 10](#_Toc380161462)

[2.4.2 Database 11](#_Toc380161463)

[2.4.2.1 Permissions 12](#_Toc380161464)

[2.4.2.2 Client connections 14](#_Toc380161465)

[2.4.3 Updating an environment 15](#_Toc380161466)

[3 Environment tasks 17](#_Toc380161467)

[4 Creating a template 18](#_Toc380161468)

[5 Checking a template 18](#_Toc380161469)

[6 Deploying a template 19](#_Toc380161470)

[7 How do I? 20](#_Toc380161471)

[7.1 Use the sample TIF 20](#_Toc380161472)

[7.2 Create one or more domains 20](#_Toc380161473)

[7.2.1 From scratch (“simple domains”) 20](#_Toc380161474)

[7.2.2 By adding to an existing template 21](#_Toc380161475)

[7.3 Create traces 21](#_Toc380161476)

[7.4 Manage presentation templates 22](#_Toc380161477)

[7.5 Include other TIF files 22](#_Toc380161478)

[7.6 Use the test management types 23](#_Toc380161479)

[8 Appendix A. Environment Details 24](#_Toc380161480)

[8.1 Restoring the database 25](#_Toc380161481)

[8.2 Deploying the solution 25](#_Toc380161482)

[8.3 Presentation templates 26](#_Toc380161483)

[9 Appendix B. Download TIF from Whittle sandbox 26](#_Toc380161484)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Change History   |  |  |  |  | | --- | --- | --- | --- | | 1.0.4 | 2013/6/25 | Part of 1.0.4 release | Initial version describing setup and environments. | | 1.0.5 | 2013/6/27 | Part of 1.0.5 release | Added “How Do I?” section describing creating new domains, traces, managing presentation templates. | | 1.0.6 | 2013/7/19 | 1.0.6 Release | Added updating an existing environment. Notes on changes to 1.0.6 environment. | | 1.0.9 | 2014/2/14 | 1.0.9 Release | Expanded the Overview section to include use cases. Improved the first section to guide the user through getting started. Moved the section on Whittle server to appendix. | |

Overview

Motivation

TIF and its associated tooling address both customer and PTC field needs.

* Customer
  + Need to be able to document processes “as were” for liability
  + Configuration management of deployed configurations
  + Incremental update of processes
  + Automate documentation of process
* PTC Field
  + Rapid prototyping and development of Integrity templates
    - Avoiding any detailed knowledge of API / CLI
  + Cooperative development (via CM)
  + Creation of reusable Integrity components
  + Simplified handover between pre and post-sales
  + Documentation automatically generated

What is TIF?

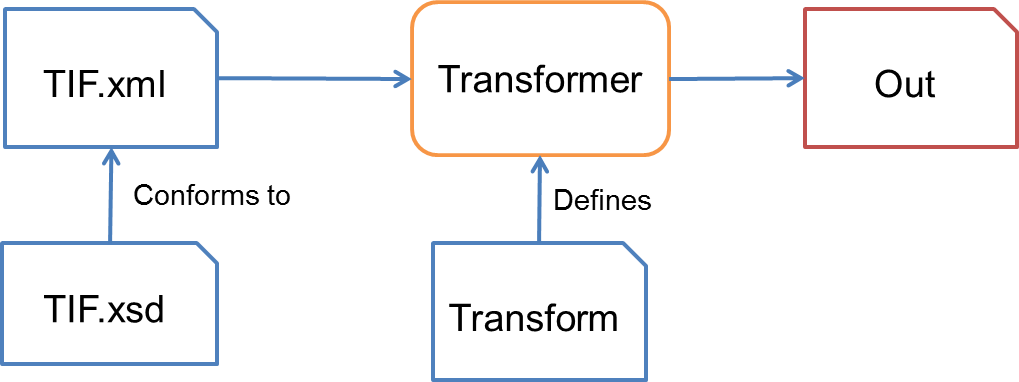
Template Interchange Format (TIF) is an XML format for the definition, exchange and comparison of administrative artifacts defined in PTC Integrity. Administrative artifacts include[[1]](#footnote-1)

* Item types with their respective workflows, etc.
* Field definitions
* Trigger definitions
* Query definitions

The goal of TIF is to capture, in a declarative form, all the artifacts that make up an Integrity template.

TIF is defined by an XML Schema: TIF.xsd.

TIF’s form (XML, declarative, schema defined) allows it to be subject to a variety of transformations for a variety of purposes: automatic generation of documentation, generation of scripts, etc. These transformations are supported by the TIF workbench allowing a template defined by TIF top be deployed into an Integrity server.



Workbench

The TIF Workbench is an application that allows users to read a TIF definition from an Integrity server, compare TIF installations and generate TIF that expresses the differences and write TIF to an Integrity server.

The workbench provides other features but these are the main ones.

Use Cases



The two actors above, Administrator and Process Developer are responsible for

1. Administrator: take TIF definitions and deploy them to one or more Integrity servers
2. Read TIF from a file: open a TIF definition in the workbench
3. Read TIF from Integrity: read the TIF definition from an Integrity server
4. Difference TIF: compare two TIFs to produce a 3rd TIF file that expresses the changes that must be made to TIF1 to make it equivalent to TIF2.[[2]](#footnote-2)
5. Developer: edit a TIF file to add, remove or edit administrative objects defined by the TIF. This is done in an editor, either as text or one that can edit XML given the TIF schema.

User Guide

Getting started

There are several step involved in getting up and running with TIF. The simplest scenario is installing the workbench, creating a TIF environment and deploying it to an Integrity server.

The following table gives the main steps with links to the sections of this guide.

|  |  |
| --- | --- |
| Install prerequisites | 1.1 Prerequisites |
| Install TIF workbench | 1.2Installation from FDO |
| Set up an Integrity server | Out of the scope of this guide. |
| Backup the Integrity database | Out of the scope of this guide. |
| Create a TIF environment | 2.2 Creating an environment |
| Set environment properties | 2.4 Environment settings |
| Populate environment. Copy the sample TIF components into your new environment. | 7.2.1 From scratch (“simple domains”) |
| Build scripts from TIF.  This is done automatically when you deploy, however it is a good check that you have set up things correctly.  Select the “scripts” task | 3 Environment tasks |
| Deploy to Integrity  Select the “deploy” task | 3 Environment tasks |

## Prerequisites

The workbench needs the following prerequisites.

|  |  |
| --- | --- |
| Integrity client | 10.4. The extraction process makes use of API features in 10.4. |
| File access to the Integrity server data directory | This is for deploying triggers, report recipes, etc. |
| Ant | Ant 1.8.2 or later |
| MKS Toolkit OR Cygwin | The deployment scripts make use of the korn shell (sh.exe), grep and other Unix tools. |
| Java runtime | JRE 1.6. The JRE bundled with the Integrity client will work. |

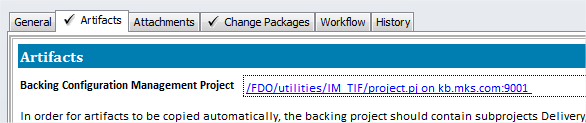
## Installation from FDO

Note: the FDO is only available to PTC personnel. If you are a customer please obtain a deliverable zip file or create a sandbox from the Whittle server (Appendix B. Download TIF from Whittle sandbox).

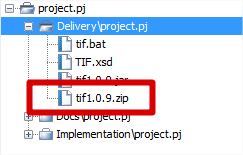
Navigate to the FDO items on the FDO server – kb.mks.com:9001.

Item 16582 contains the TIF versions.

* 16583 TIF 1.0.9



The TIF deliverables are in the project linked to the FDO version.

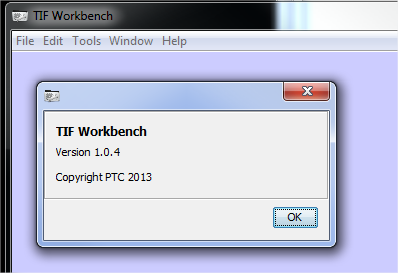


The Deliverables sub-project contains the TIF zip file.

IMPORTANT: the zip contains all the TIF resources – unzip this, don’t just copy the bat and jar file.

If your Integrity client is in the default location you can run the workbench by running tif.bat. Of you have installed your Integrity client somewhere else (the workbench needs a JRE) edit the batch file and configure IC\_HOME

SET IC\_HOME=C:\Program Files (x86)\Integrity\IntegrityClient10



Environments

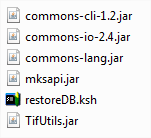
The TIF xml file defines the administrative artifacts that are in an Integrity template, however, certain other artifacts need to be available to actually deploy the TIF definition into Integrity.

* Ant build file
* The transformations that create shell scripts from TIF
* Any trigger scripts
* Any images referenced in the TIF
* Presentation templates
* Support scripts

The workbench can create a default environment for you and provide a convenient mechanism for working in it.

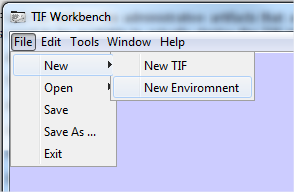
## Release 1.0.9 environment

Please not that release 1.0.6 removed dependencies on Perl and the mkslib library. The functions previously provided by three scripts are now implemented in TifUtils.jar. This only affects the externalScripts directory. For 1.0.6 this directory contains the files shown below.

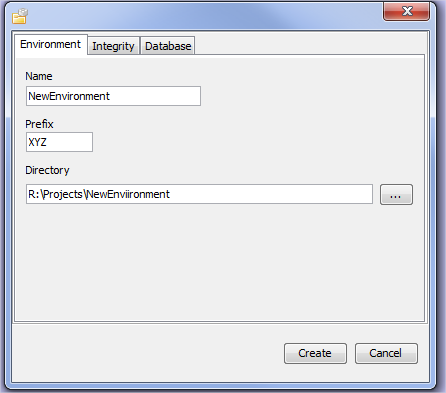


Creating an environment

To create a new TIF environment open the workbench and select File - New – Environment.

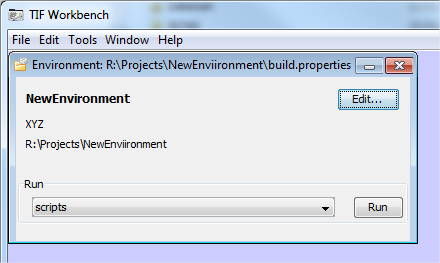


You can start by entering just the information on the first tab – other details can be entered after the environment has been created.

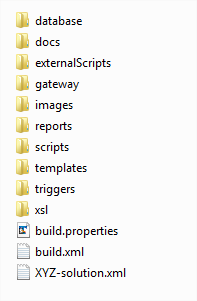


The settings shown above will create a new environment in r:/Projects/NewEnvironment. The environment specifies a prefix. This is used when naming your new TIF file and also allows the deployment process to keep the artifacts associated with your template separate from other templates: e.g. triggers, images, etc. when deployed to the Integrity server will be put in directories identified with your prefix.

Click create and you will see your new environment in the workbench.



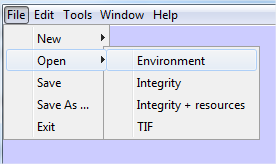
On disk you will see:

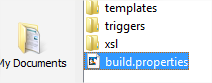


The file build.properties contains the properties you specified when creating the environment. Developing TIF from this point involves adding definitions to the TIF file: XYZ-solution.xml.

Opening an environment

To open a previously created environment, select File – Open – Environment and browse to the file build.properties in the environment.





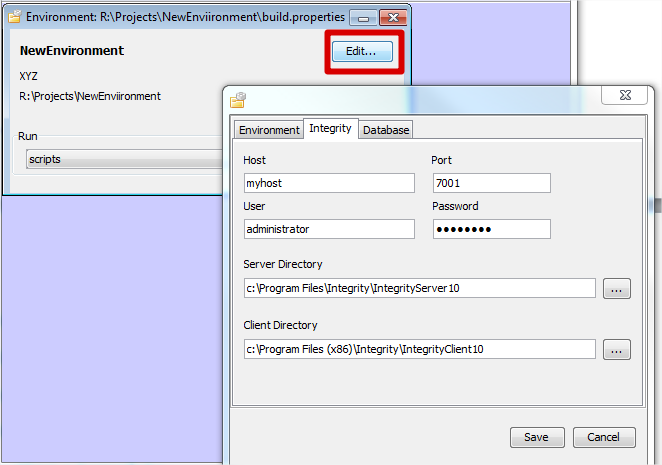
Environment settings

With the environment open in the workbench, click the Edit button to change the environment settings.

Integrity

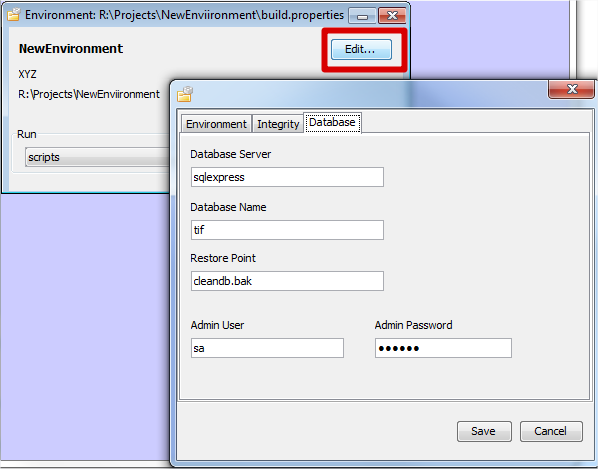
The Integrity tab allows you to set properties related to your Integrity server and client. Note that for deployment resource files will be copied to your server so you must have at least a drive mapped to the server root directory.

The Integrity system you intend to deploy to must have at least one user with administrative permissions.



Database

The database settings are present to allow you to roll-back your Integrity database as part of your development cycle[[3]](#footnote-3). To this end you should specify the server name, the database you will use and a database backup file. The backup should be the point from which you intend to deploy your TIF. This may be a “clean” Integrity database (containing, of course, at least the Integrity user you specified above), or an existing template onto which you want to create new artifacts.

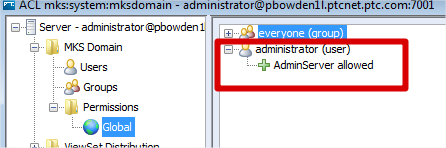


Your database should have certain setting to allow the deployment process to work. These are appropriate permissions for the *Integrity* user that will perform the deployment and an increased number of client connections.

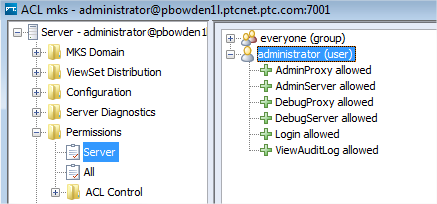
Permissions

Set the following permissions for the Integrity user you have configured in your environment.

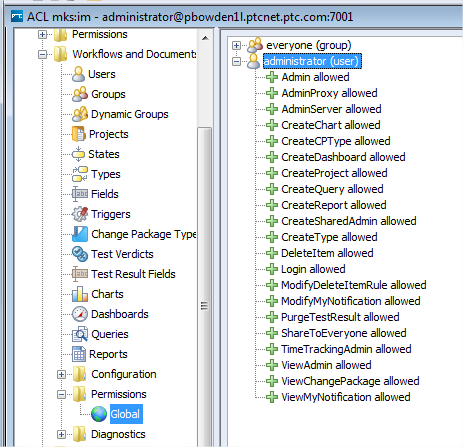
All MKS Domain permisions.



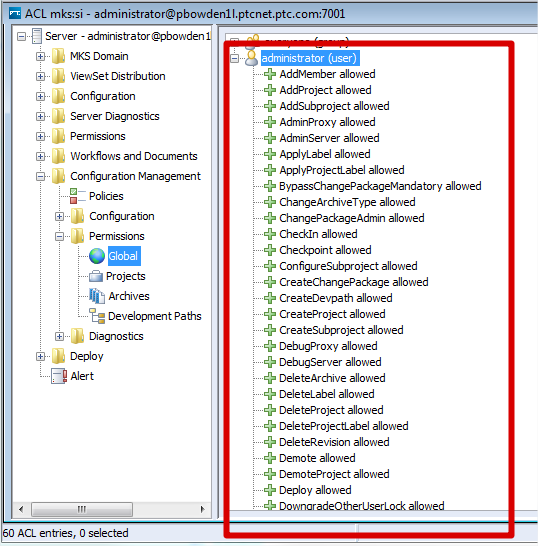
All server permissions



All workflows and documents permissions

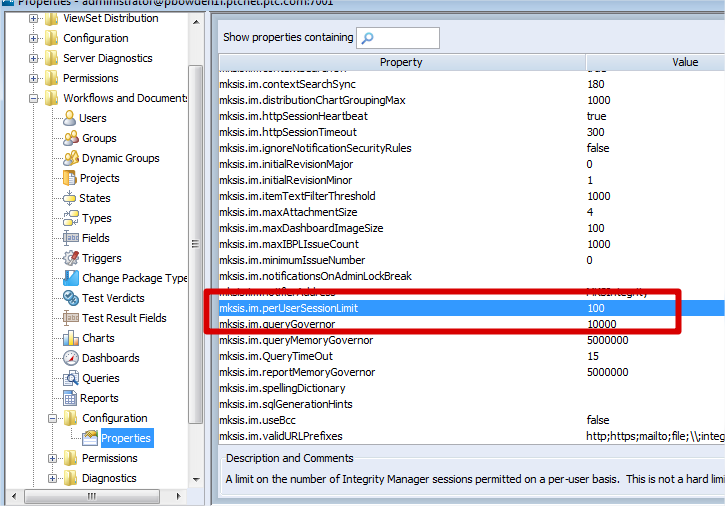


All configuration management permissions.



Client connections

Set the number of client connections (mksis.im.perUserSessionLimit) to 100.



Updating an environment

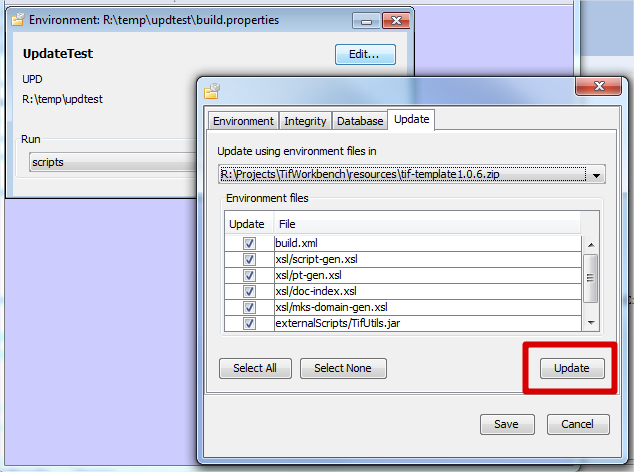
An environment contains a number of files that are not usually modified on a per-project basis but may well change with different releases of the TIF environment. The workbench provides a convenient way to update an existing environment so that it uses the transforms, scripts, etc. consistent with a particular release.

The files that are updateable are:

* build.xml
* xsl/script-gen.xsl
* xsl/pt-gen.xsl
* xsl/doc-index.xsl
* xsl/mks-domain-gen.xsl
* externalScripts/TifUtils.jar
* externalScripts/restore-db.jar

For a description of what these files do see Appendix A. Environment Details.

To update your environment, select Edit and then the Update tab.



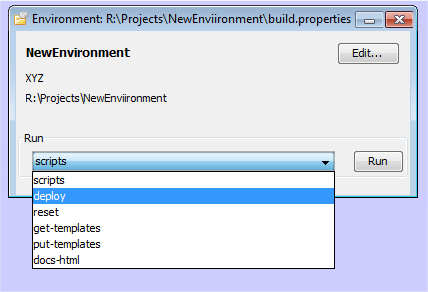
When you create a new environment the files needed are unzipped from a zip file in the resources sub-directory. Similarly, when updating the files to be used are unzipped from a particular version of the environment zip. The drop-down list will contain all the environment zips found in *resources*. This allows you to update an environment to a particular release version.

Note, no automatic backup of the target environment is taken so backup if needed.

You can select the environment files you want to update. Click Update to unzip the checked files from the selected environment zip file.

Environment tasks

There are a series of tasks you can run from the environment[[4]](#footnote-4). When you run a task a new command line window will be created to run the selected task.



|  |  |  |
| --- | --- | --- |
| Scripts | Transform the TIF to produce all the deployment scripts. | The main script is scripts/<PREFIX>.ksh. Ant will rebuild dependent files – i.e. if your template has not changed the transformation will not be applied. Touch <PREFIX>-solution.xml to force a rebuild of the scripts. |
| Deploy | Rebuild the scripts and deploy the template to Integrity. | Errors will be shown in the command window. A full log of the deployment will be in deploy.log. |
| Reset | Stop the Integrity server, restore the database specified in the environment to the restore point and restart the server. | This task will “tail” the Integrity server log so you can see when it has started |
| Get-templates | Get the presentation templates from the Integrity server and copy them into the environment. | The copy process replaces all field IDs in the template with field names and replaces any references to the server name with an anonymous version.  The Integrity server must be running to perform this task. A typical use case is the deploy your template, create presentation templates using the graphical designer in the admin client and then get-templates. |
| Put-templates | Put the presentation templates to the Integrity server. | This is done as part of the deploy task so you will rarely need to run it in isolation. |
| Docs-html | Transform the TIF into an html page using the doc-html.xsl transformation. | The resulting file will be docs/solution.html. This file references docs/main.css. |

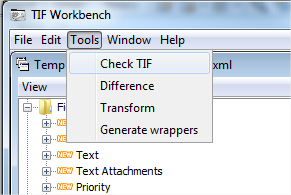
Creating a template

An empty template is created for you when you create a new environment.

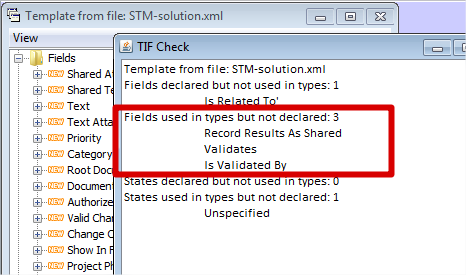
You may use an XML editor and the schema to author TIF.

Checking a template

The workbench can perform a number of useful consistency checks on your TIF. Open your TIF file (the XXX-solution.xml file) and select Tools – Check TIF.



This will scan the TIF and report on undefined and unused fields and states[[5]](#footnote-5).



The check will list

* Standard fields you have declared (this is a mistake)
* Redundant fields, i.e. declared but never used in a type.
* Undeclared fields, i.e. used in a type but not declared.
* Redundant states, i.e. declared but never used in a type.
* Undeclared stated, i.e. used in a type but not declared.

Note the TIF window does not refresh when you change the underlying file. If you want to recheck you must close and re-open the TIF file and rerun the check.

Deploying a template

As described in the table above, select the deploy task in the environment to deploy your template. The results of the deployment will be in deploy.log at the end of the process.

How do I?

Use the sample TIF

There are several sample TIF files in the resources directory. For the most part these are parts of template you can use to build up your own TIF.

Create one or more domains

When adding domains to a template the first question to ask is will you build on an existing template that already has domains defined or create your template from scratch.

Some of the things to consider are:

|  |  |
| --- | --- |
| Building on existing domains | Better for rapid prototyping and demonstrations |
|  | Much quicker to get up and running |
|  | All the metrics, reports, charts, etc. are available. |
|  | The template will contain a lot of objects that may not be needed by the customer |
| Scratch building | Better for a clean implementation. |
|  | Sample domains are minimal |
|  | Requires the metrics, reports, charts, etc. to be defined. |

If you are trying to prepare a demonstration or a prototype, building on an existing template such as ALM or GSD is often the quickest as you can just copy the TIF (via an extraction) for an existing domain and adapt it to your purposes.

Scratch building is usually more appropriate for an implementation where you may not want all the other objects that come with a standard template or you want to connect documents to custom change processes.

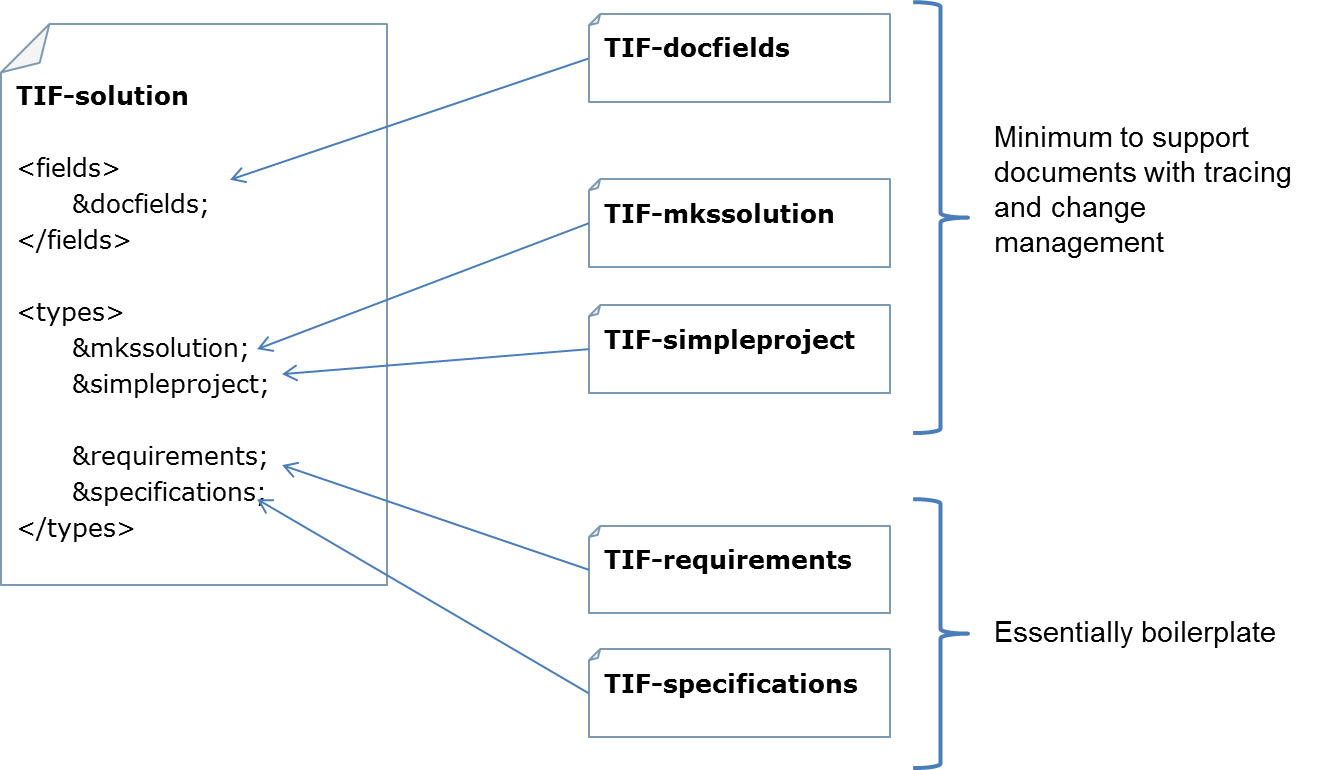
Note that sample “bare-bones” domains are available with the workbench. Eventually more elaborate TIF files will be available that contains metrics, etc.

From scratch (“simple domains”)

You can build a set of domains from scratch using the TIF in the “simple domains” sample.

This comprises a template file (TIF-solution.xml) that contains the minimum fields to support a domain (e.g. trace status, change management computed fields, etc.), a simple definition for the MKS Solution type (this is mainly a set of type properties) and two linked domains: requirements and specification.

Requirements can be traced to specifications via a “Decomposes To” trace. Requirements and specifications can have self-traces via “Is Related To”.



Unzip the simple-domains environment and work from there.

New domains can be created by simply copying an existing one and including it in the top level TIF file. Each sample domain contains the segment, node and shared item for the domain. A simple search and replace can then change the segment, node and shared item names. A property on the document type names the domain.

<property value="**Requirement**" name="**MKS.RQ.Domain**">

<description>(Configurable) The solution domain this document type is used with. Change this value only if you are copying this type to create a new domain.</description>

</property>

By adding to an existing template

When adding to an existing template you will mostly create new domains (as copies of existing ones) and update existing fields, e.g. the Shared Category field, to support your new domains. The **MDE** sample environment shows how a new “Package” domain is added to an existing ALM installation. The Package nodes have new Category values which involves updating the Shared Category field. Look for an <Edit-field> element so see how this is done. The sample environment also contains all the category images; the edit will maintain the images used in the standard templates.

This sample builds on the ALM template.

Create traces

To can create traces using the type properties and the relationships, but the TIF transform contains a convenience structure to update a relationship field (assumed to exist) and set the appropriate type properties for a trace.

In the list of fields for a node, refer to the relationship field you want to use as a trace and say how it will be used:

<field name="Decomposes To">

<traces backward="Decomposed From" description="A Requirement decomposes to a Specification.">

<trace target="Specification" />

</traces>

</field>

You may have several traces from a node via a trace.

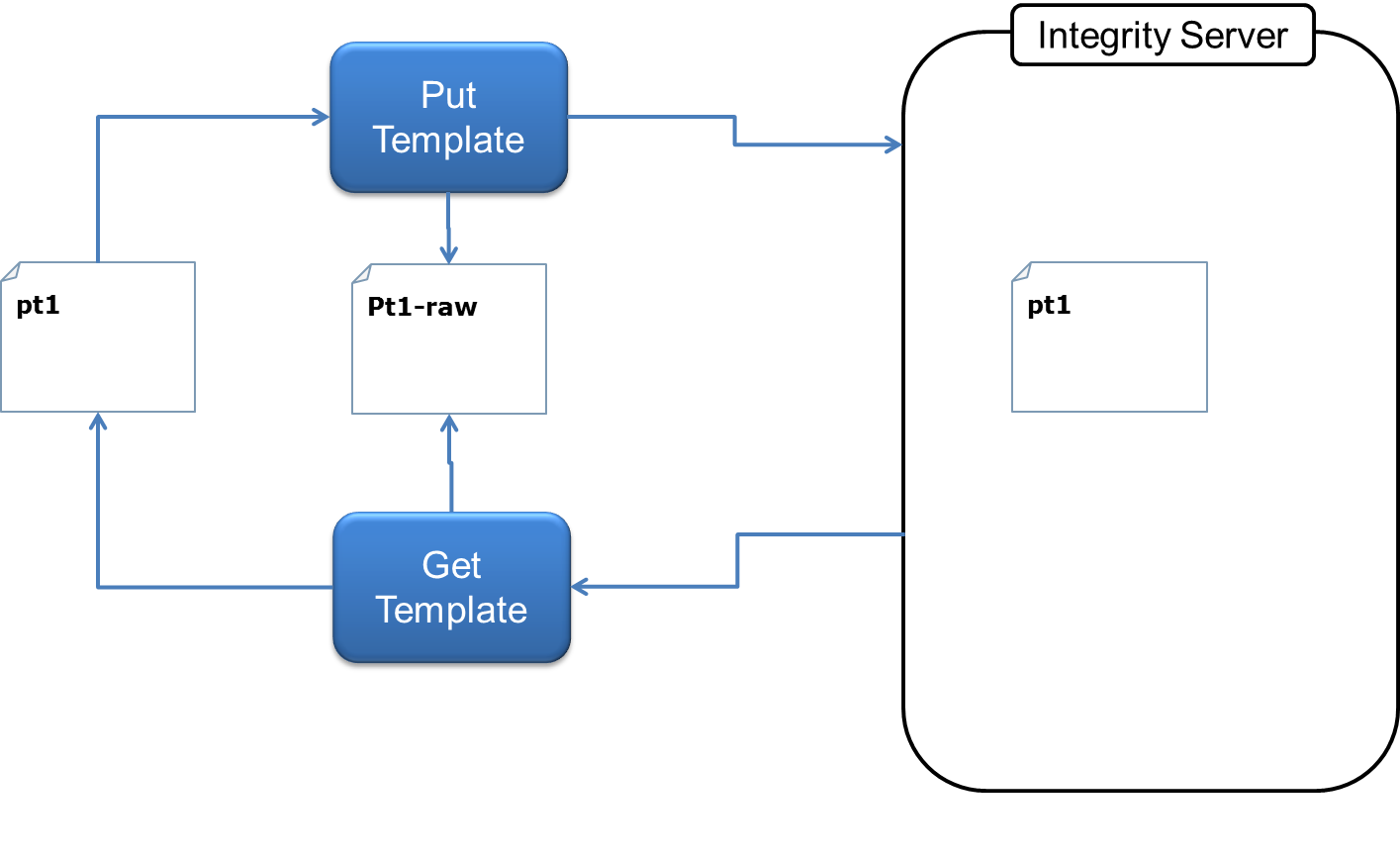
TIP: you should declare the trace in one type. Types at the other end of the trace may simply declare that the relationship field is used.

For example, a Requirement is Decomposes by a Specification. Put the Satisfies trace in the declaration of the Requirement and just specify the reverse field in the Specification.

<field name=”Decomposed From” />

Manage presentation templates

Within the workbench there are two tasks: get-templates and put-templates that allow you to manage presentation templates in your environment.



Get-templates retrieves the templates identified in the item definitions to templates/raw. It then converts all the field IDs in the “raw” templates to field names (making the template server-independent) and copies then to the templates directory. Server names are also converted to <SERVER> to avoid referring to a specific machine.

Set the presentation attribute in a type definition to bind it to a template.

<type name="Requirement" presentation="segment-red">

Put-templates works in the other direction, converting templates in your environment to “raw” form and then importing them into the Integrity database and finally setting the types to use them.

Note, currently one presentation is used for the view, edit and print templates for each type

Include other TIF files

To include a TIF file in another, declare an XML entity.

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<!DOCTYPE im-solution [

<!ENTITY requirements SYSTEM "TIF-requirements.xml">

**<!ENTITY specifications SYSTEM "TIF-specifications.xml">**

<!ENTITY mkssolution SYSTEM "TIF-mkssolution.xml">

<!ENTITY simpleproject SYSTEM "TIF-simpleproject.xml">

]>

And refer to the entity at the appropriate point in the TIF file.

</states>

</type>

&requirements;

**&specifications;**

</types>

Use the test management types

The SimpleTM sample is included to illustrate how the test management types work together. This is a minimal solution and includes for metrics and only basic traceability to a single Requirement domain.

The sample includes a Visio file of the information model. It is shown before for convenience.



Please also refer to the ALM or GSD template documentation available from PTC.

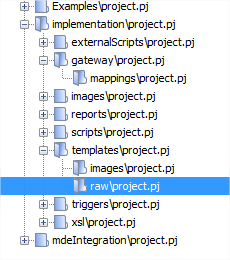
# Appendix A. Environment Details

The solution framework project gives you the basic pattern for the files and directories you will need. Several directories are standard to the framework and to encourage reuse should be configured as shared sub-projects.

|  |  |  |
| --- | --- | --- |
| **File / directory** | **Function** | **Comment** |
| build.properties | Installation-specific settings | Modify for your machine. |
| build.xml | Ant tasks | Standard. Unlikely to change |
| \*\*\*-solution.xml | TIF main file | Contains some sample XML |
| \*\*\*-triggers.xml | Trigger definitions | Contains some sample XML |
| \*\*\*-queries.xml | Query definitions | Contains some sample XML |
| \*\*\*-domains.xml | Document domains: document, node and shared item. |  |
| \*\*\*-edittypes.xml | Types that are to be edited | Contains some sample XML |
| Xsl/script-gen.xsl | Create korn shell scripts from TIF | Xsl directory should be a shared sub-project |
| Xsl/doc-gen.xsl | Create html documentation from TIF |  |
| Xsl/pt-gen.xsl | Create presentation template update scripts |  |
| ./triggers | Integrity triggers used in the solution | Solution specific |
| ./templates | Presentation templates. Note: these are processed to use field names rather than IDs to the solution is server independent. | Solution specific |
| ./images | Images used by the solution. | The template project contains a large number of samples, including all those used in the Category field. |
| ./externalScripts | Support scripts. | Standard, configure shared. |
| ./scripts | Automatically generated scripts. |  |

Basic setup comprises:

1. Create a new Source project. An “implementation” sub-project is usual for the solution.
2. Add xsl and externalScripts as shared sub-projects
3. Create subprojects
   1. Scripts
   2. Reports
   3. Images
   4. Templates
   5. Templates/raw
   6. Triggers
4. In your root directory add from the template project
   1. Build.properties
   2. Build.xml
   3. Solution sample files template-\*.xml
5. Create a backup of your “blank” Integrity database. This will be the starting point for your deployments. It may be the GSD template.
6. Modify build.properties for your environment.



Restoring the database

An ant task, “restore-db”, called by the “reset” task will restore your database based on the settings in build.properties.

Note: the SQL server command line can’t restore with arbitrary paths to the backup file. You’ll need to put it in the default location.

sql.user=sa

sql.password=mks123

sql.db.server=pbowden1l\\sqlexpress

sql.db.name=Studio2010

sql.db.restore.point=Studio2010.bak

sql.db.restore.point is the name of your backup file.

Enter

ant restore

To reset your Integrity server to a known state.

Deploying the solution

Enter

ant deploy

to run the shell scripts that will deploy your solution. The command output will be written to deploy.log so you can check any errors. The ant task will also grep deploy.log for the string “error” at the end of the deployment so you can make a quick visual check.

If you update the solution files ant will regenerate the scripts. Look in the scripts directory for these. If ant does not regenerate the script when you expect it is probably because you have updated one of the XML files that the main solution file includes (ant cannot detect this). Touch the main solution file and regenerate the scripts with

ant scripts

Presentation templates

If any of your types refer to presentation templates

<type name="Test Execution Task" attachments="yes"

icon="resources/t\_test\_task.gif"

show-workflow="true"

**presentation="testing\_task"**

cp-policy="userField=Assigned User">

the task

ant get-templates

will copy the presentation templates down from the Integrity server and to the templates directory. This process also converts the field numbers in the template to field names and convers any server references to <%SERVER%>. This means you can edit templates using the graphical editor but store them in a server independent format so others can deploy your solution.

The opposite happens with *ant put-templates*. The task *put-templates* is called as part of *deploy*.

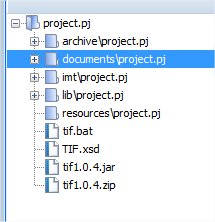
Appendix B. Download TIF from Whittle sandbox

A release of TIF is provided for external customers on an Integrity server: Whittle which is visible externally.

Navigate to Whittle.mks.com:8080 and project /TIFWorkbench/project.pj. Create a sandbox from either the head or a particular checkpoint.

This server is managed by Paul Bowden ([pbowden@ptc.com](mailto:pbowden@ptc.com)). Request an account if you want to get the TIF installer from here.

Navigate to Whittle.mks.com:8080 and project /TIFWorkbench/project.pj. Create a sandbox from either the head or a particular checkpoint.



If your Integrity client is in the default location you can run the workbench by running tif.bat. Of you have installed your Integrity client somewhere else (the workbench needs a JRE) edit the batch file and configure IC\_HOME

SET IC\_HOME=C:\Program Files (x86)\Integrity\IntegrityClient10

1. The exact set of Integrity admin objects that the workbench supports are detailed in TIF Functional Detail.xlsx [↑](#footnote-ref-1)
2. This is the default difference algorithm; others can be plugged into the TIF workbench. [↑](#footnote-ref-2)
3. NOTE: this is only implemented for Microsoft SQL Server. If you need to use Oracle you will have to modify externalScripts/restoreDB.ksh appropriately. [↑](#footnote-ref-3)
4. These tasks in the environment correspond to Ant targets you will find in build.xml. [↑](#footnote-ref-4)
5. This functionality will evolve to include more sophisticated checks. [↑](#footnote-ref-5)