Unified Framework for Driving Transformations

Labnaf Language Transformer

User Guide

See also the "**Labnaf Language Transformer - Reference Guide**"

WARNING

NEVER use the language transformer on your production repository before performing all necessary tests.

ALWAYS test your language transformer commands using a repository backup.

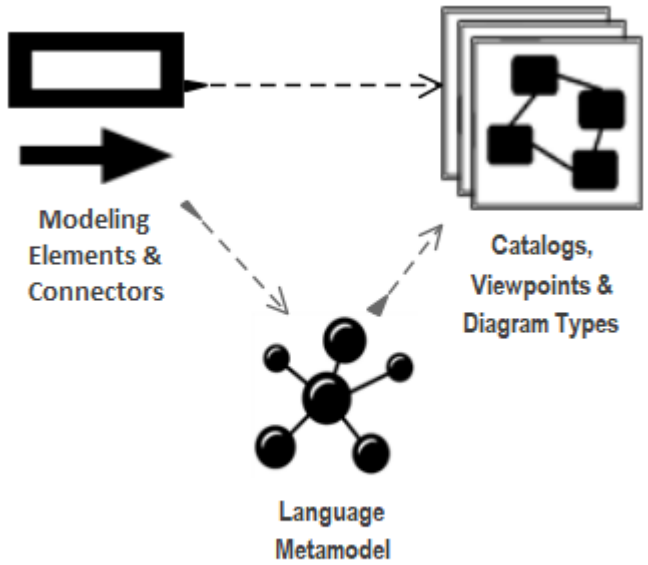
ALWAYS carefully check the resulting transformations and possible side effects. For example items could be deleted because you misspelled a type.

ALWAYS remember that type and stereotype names are case sensitive.

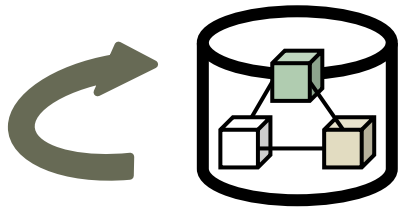
The Labnaf Language Transformer has been tested using Sparx Systems' Enterprise Architect versions 13.5 to 16.1

Labnaf Customization Steps

1. Customize the language following your organization requirements

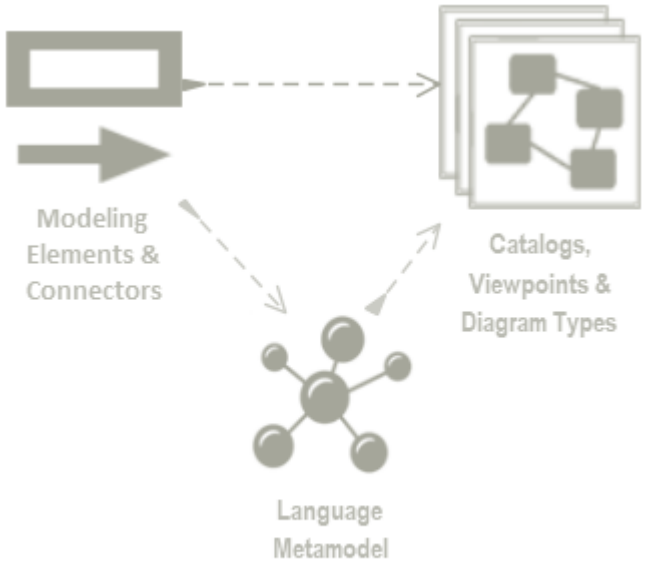


2. Adapt existing repository content

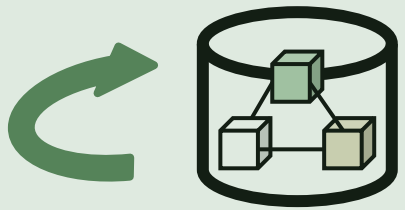


Labnaf Language Transformer

1. Customize the language



2. Adapt existing repository content



ARCHIMATE®

BPMN™

UNIFIED MODELING LANGUAGE™

Visio

Any other language

Transform any language…

We all speak the same language

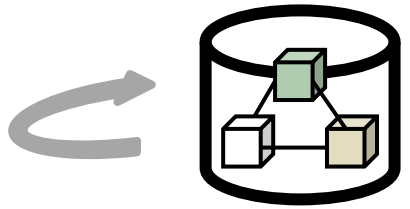
… in any direction

One **Single**
Strategy & Architecture
Modeling **Language**

You are never locked in Labnaf !

The Language Transformer adapts the language in existing repository content

- **ChangeElementType**
- **ChangeConnectorType**
- **ChangeDiagramType**
- **ChangeDiagramTypesDefinedInCsv**
- **TvRename**
- **TvDelete**



See the **Language Transformer Reference Guide** for command specific information.

Prerequisites

Install the Language Transformer using the

Installer/LabnafLanguageTransformerSetup.msi

Unzip the sample language transformations files stored in

Doc/SampleLanguageTransformations.zip

Development Lifecycle

- To transform a language in a repository, you will need a combination of transformer commands.
- We advise to address each command separately and in that order:
  - **ChangeElementType**
  - **ChangeConnectorType**
  - **ChangeDiagramType or ChangeDiagramTypesDefinedInCsv**
  - **TvRename**
  - **TvDelete**

The development lifecycle is the same for each command.

Keep backups of your different versions.

Transforming Element, Connector and Diagram Types

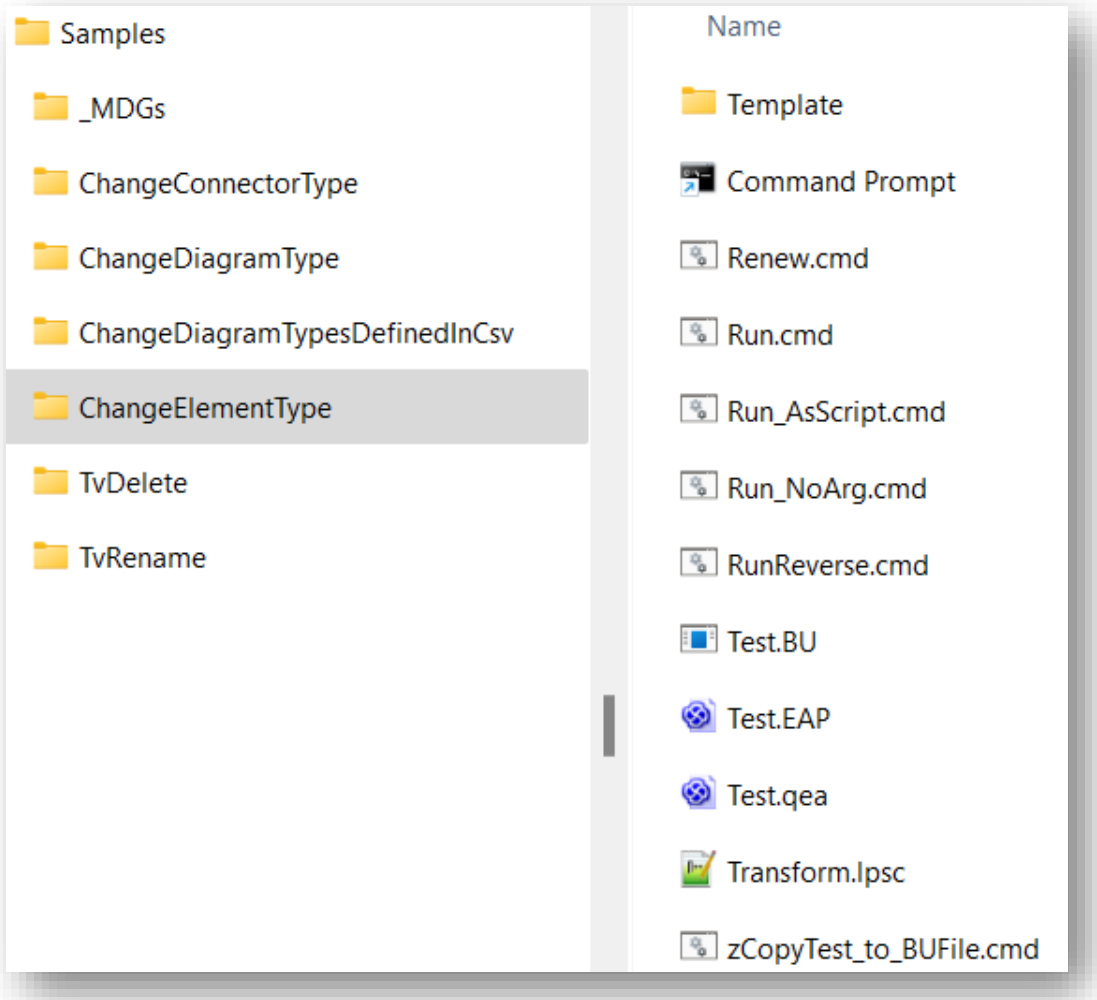
Review the content of the Sample language transformations folder that you unzipped

For each Language Transformer command, there is a sample folder with an example. Folder names correspond to language transformer commands.

The folders have a common structure. The file names are about the same.

The content of the “Run.cmd”, the “Run_AsScript.cmd” and “Test.qea” files are specific to each folder/command.

Run.cmd runs each command in a different EA session.
Run_AsScript.cmd runs all commands in a single EA session.



Create your Test Data

- Execute **Renew.cmd**
  - to copy the backup database into a fresh **Test.qea**

- Open and edit **Test.qea.**
  - Replace the content by you own test data
  - Close **Test.qea**

- Execute "**zCopyTest_to_BUFile.cmd**"
  - to create a new backup database from **Test.qea.** So you will then be able to Renew the test database with your own data before running each test.

Keep backups of your different versions.

Develop and Test Your **Type Transformation Script**

Edit and run the test script

- Edit either **Run.cmd** or **Run_AsScript.cmd** to create your own transformation commands.
- Execute the Run script

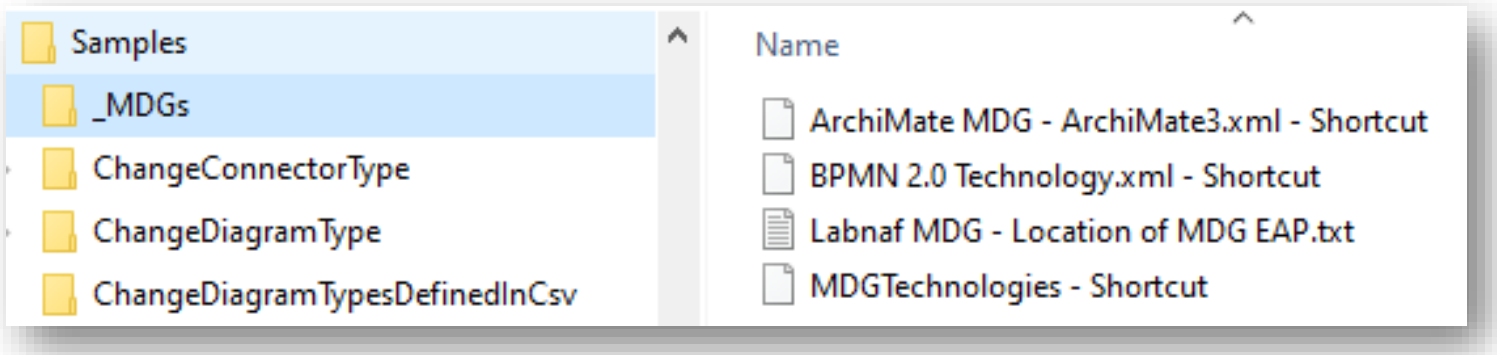
See the language transformation results

- Open **Test.qea**
- **In Sparx EA**

Keep backups of your different versions.

To find out what types and stereotypes are

- Look inside the MDG XML files



- Search for **Stereotype name="*{stereotype I am looking for}*"**
  Or **metatype="*{metatype I am looking for}*"**

```
<Stereotype name="ArchiMate_BusinessRole" metatype="BusinessRole" notes="" bgcolor="1
  <Icon type="bitmap" xmlns:dt="urn:schemas-microsoft-com:datatypes" dt:dt="bin.base6
    Qk02AwAAAAAAADYAAAAoAAAAEAAAABAAAAABABgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    wMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDA
```

- See the extended type

```
<AppliesTo>
  <Apply type="Class">
```

Test on real data

- Backup your PROD repository into an file-based repository (QEA).
- Replace “Test.qea” by that backup repository.
- Execute the transformation script “Run.cmd” and see the result

Run your transformation script on the PROD repository

- Ensure nobody is working on the repository
- Make a backup of you PROD repository
- Replace “Test.eap” by a shortcut to your PROD repository.
- Execute the transformation script “Run.cmd” and see the results

Keep backups of your different versions.

Run multiple commands from a Labnaf Power Script file (.lpsc)

lnxf {LabnafPowerScriptFileName}.lpsc {RepoPathName} [SecondsBeforeRestart] [MaxMinutesRestartAfterFirstRun]

This executes commands from a '**{LabnafPowerScriptFileName}.lpsc**' file within a **single Sparx EA session**.p

All commands refer to the **same repository {RepoPathName}** specified in the Windows command line,
so there's no need to include a repository argument inside the Labnaf script file itself.

Sample Labnaf Power Script:

```
// Script files can include comments
/* They can also include Windows environment variables in the usual form %VARIABLE% */
ChangeElementType Activity ArchiMate_BusinessProcess DerivedType LABN_Process
ChangeElementType * ArchiMate_ApplicationComponent DerivedType LABN_Application
```

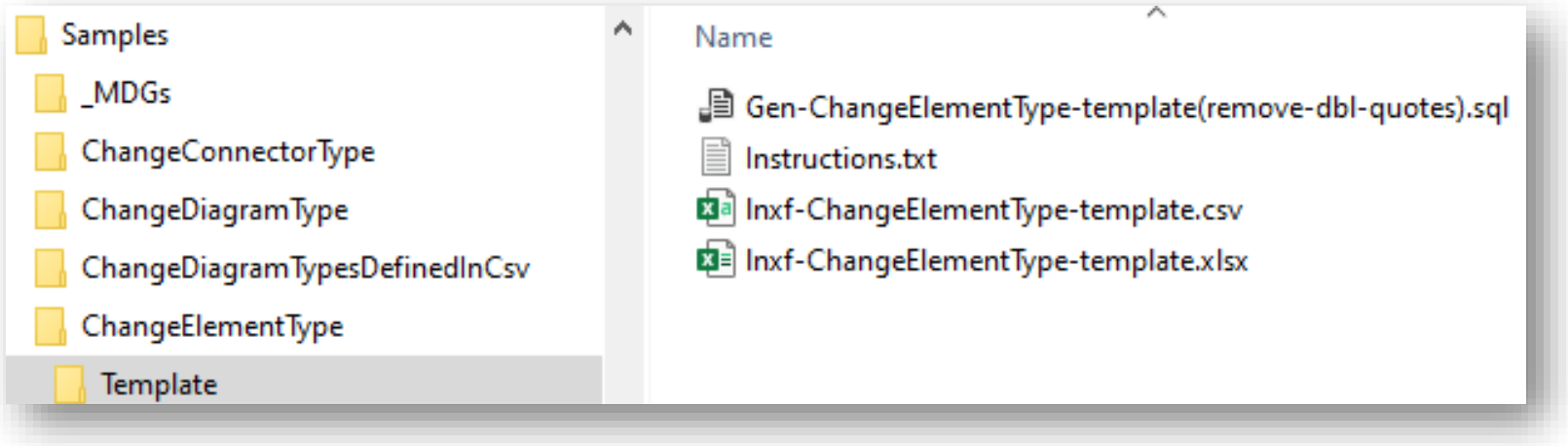
If specified as lnxf arguments, the script will run iteratively:
- Resumes after **{SecondsBeforeRestart}**
- Concludes after **{MaxMinutesRestartAfterFirstRun}**

If only {SecondsBeforeRestart} is provided, the script continues
until halted by the admin via the ESC key in the console window.

Development Lifecycle specific to the command

ChangeDiagramTypesDefinedInCsv

Use the Excel Templates to define your mappings



- Map source to target types and stereotypes in Excel

	A	B	C	D	E	F	G	H	I
1	Inxf	ChangeElementType	{RepositoryFilePath}	{FromType}	{FromStereoType}	Component	LABN_AccessPoint	{ToTagName}	{ToTagValue}
2	Inxf	ChangeElementType	{RepositoryFilePath}	{FromType}	{FromStereoType}	Activity	LABN_Activity	{ToTagName}	{ToTagValue}
3	Inxf	ChangeElementType	{RepositoryFilePath}	{FromType}	{FromStereoType}	Component	LABN_Application	{ToTagName}	{ToTagValue}
4	Inxf	ChangeElementType	{RepositoryFilePath}	{FromType}	{FromStereoType}	Collaboration	LABN_ApplicationAsAService	{ToTagName}	{ToTagValue}
5	Inxf	ChangeElementType	{RepositoryFilePath}	{FromType}	{FromStereoType}	Component	LABN_ApplicationComponent	{ToTagName}	{ToTagValue}

- Create your transformation commands based on the Excel

Development Lifecycle specific to the commands

TvRename and TvDelete

As an alternative, you can also use the Labnaf AddIn user interface