



Unified Framework for Driving Transformations

Labnaf PowerShell

User Guide

Labnaf PowerShell Commands

1. Overview
2. Strategy and Architecture Operations
3. Systems Integrations and Content Refactoring
4. Command Compatibility Matrix

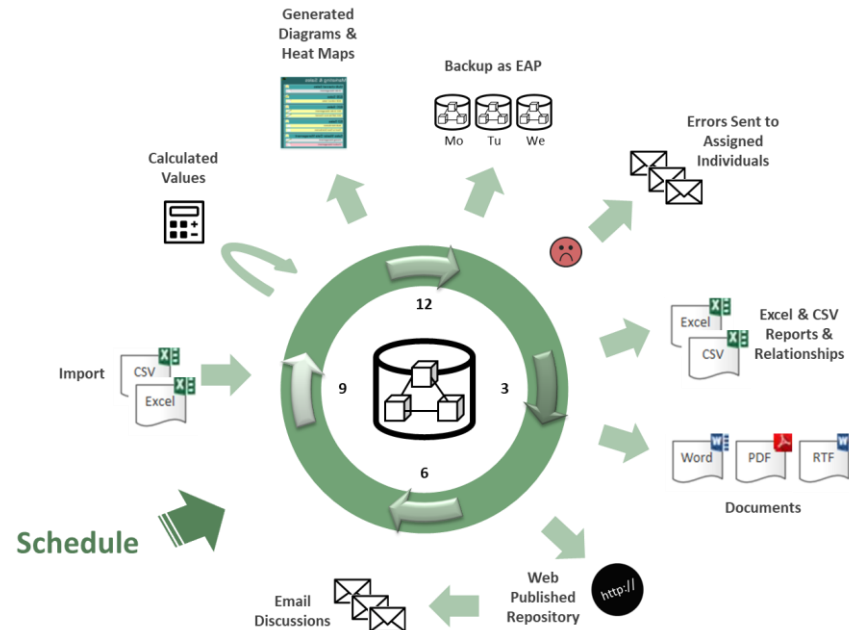
Related resources (latest versions)

- **Labnaf PowerShell Reference Guide:** https://www.labnaf.one/EndUserMaterial/Labnaf_PowerShell/Labnaf%20PowerShell%20-%20Reference%20Guide.pdf
- **Labnaf On-line Guidance:** <https://www.labnaf.one/guidance/index.html?guid=569FF62A-5210-4359-923F-4EB00EB03D61>
- **Sample data:** Provided with the Labnaf PowerShell software

The Labnaf PowerShell provides **command line** and **scheduled** access to the repository content

Commands address two main groups of uses cases

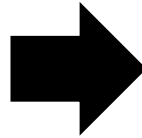
- **Strategy and Architecture Operations**



- **Systems Integrations and Content Refactoring**

Running commands on the command line

Inps



```
Usage : Inps Command [arguments]
```

```
Available Commands:
```

```
AutoConnectorsDelete
```

```
AutoConnectorsGenerate
```

```
BackupToAccessFile
```

```
CalculateTaggedValues
```

```
ClonePackage
```

```
CreatePackage
```

```
ExportToXmi
```

```
GenerateDiagrams
```

```
GenerateDoc
```

```
GenerateHTML
```

```
GenerateTabularReports
```

```
ImportConnections
```

```
ImportFromXmi
```

```
ImportTabularReport
```

```
MoveElementsToCalculatedParent
```

```
MoveElementsToPackage
```

```
MovePackagesToPackage
```

```
RenameItem
```

```
ScheduleCommand
```

```
SetDiagramProperty
```

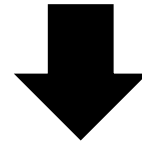
```
SqlExportToCsv
```

```
Validate
```

```
? => Show details for all commands
```

Running commands on the command line

Inps ?

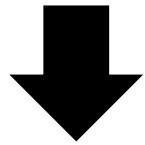


Shows a detailed description of all commands and their usage

```
C:\Program Files (x86)\Labnaf\PowerShell>lmps ?  
  
Command: AutoConnectorsDelete  
  
Description: Delete generated connectors for child elements following defined element stereotype hierarchies.  
Usage : lmps AutoConnectorsDelete [arguments]  
  
Arguments:  
    SourceRepoPathName: Path name of the source repository (EAP file).  
  
Command: AutoConnectorsGenerate  
  
Description: Generate connectors for child elements following defined element stereotype hierarchies.  
Usage : lmps AutoConnectorsGenerate [arguments]  
  
Arguments:  
    SourceRepoPathName: Path name of the source repository (EAP file).  
  
Command: BackupToAccessFile  
  
Description: Backup a DBMS or Access repository to an Access Repository.  
Usage : lmps BackupToAccessFile [arguments]  
  
Arguments:  
    SourceRepoPathName: Path name of the source repository (EAP file).  
    DestEapPathName: Path name of the destination Access repository (EAP file).  
    LogFilePath: Path name of the log file name.  
  
Command: CalculateTaggedValues  
  
Description: Calculate values for some defined tags and elements. The elements to be selected, the tags to be updated and the calculation formulas are all defined in the model repository.
```

Running commands on the command line

Inps [-]{command name}



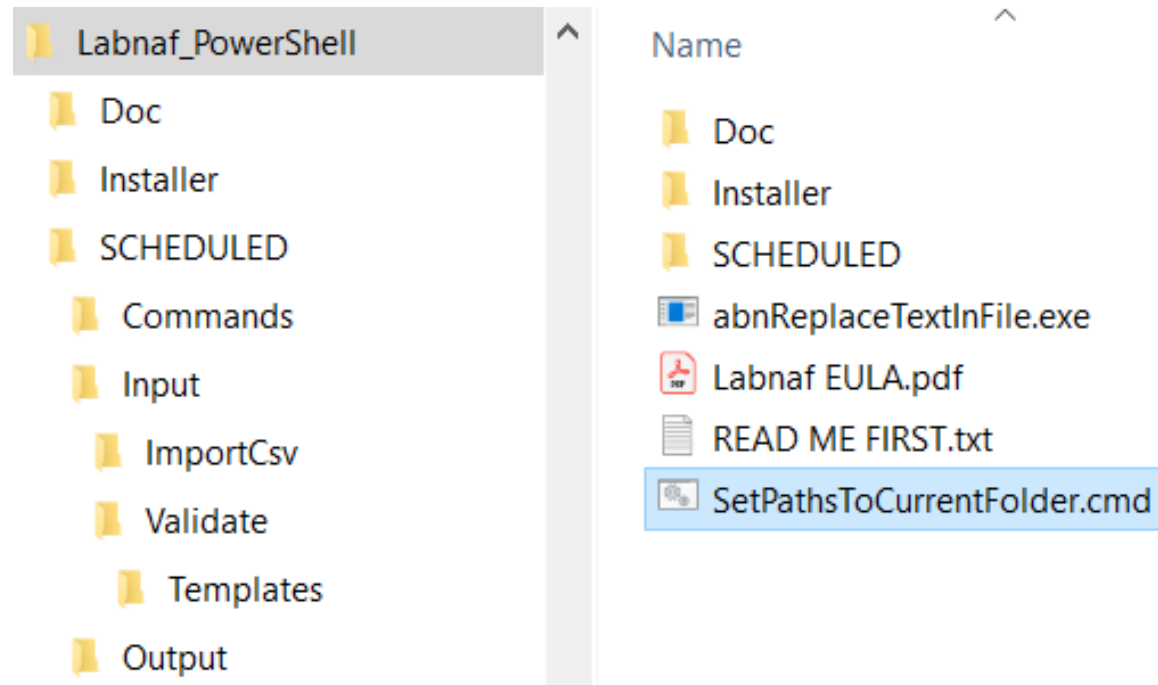
Example: if you type « Inps GenerateTabularReports » you get the following info.

```
Command: GenerateTabularReports
Description: Generate spreadsheets from a model repository based on configuration stored in that same repository.
Usage : Inps GenerateTabularReports [arguments]
Arguments:
  SourceRepoPathName: Path name of the source model repository (EAP file).
  OutputDirectoryPath: Directory path name where the spreadsheets must be generated. The name of each spreadsheet file is the name of the template report.
  [ElementPrototypeName]: The name of a specific element prototype name for which all embedded tabular report templates must be applied.
  [TabularReportTemplateName]: The name of a specific tabular report template to be applied.
```

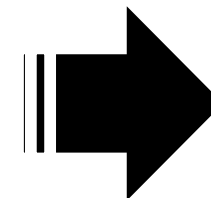
Prefix the command name with '-' to run in non verbose mode

Automatic configuration

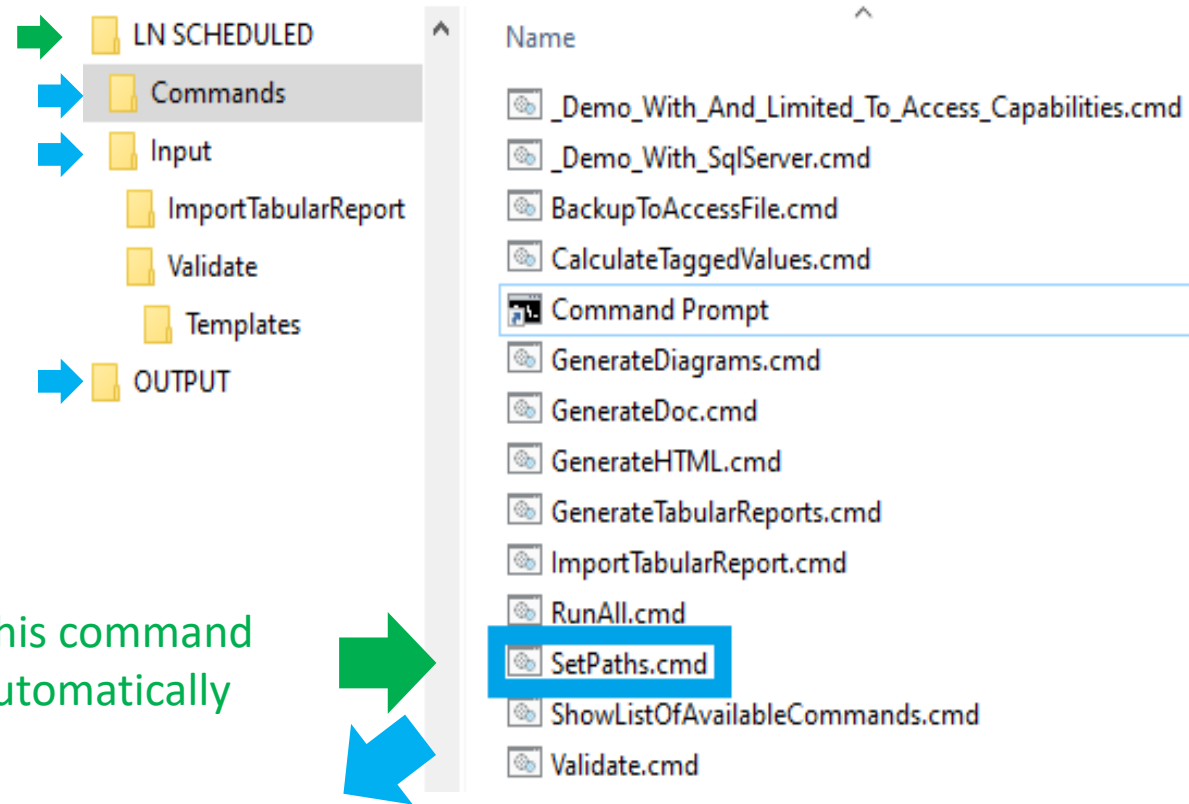
- Copy the Labnaf_PowerShell folder anywhere you want on your file system
- Double-click on “SetPathsToCurrentFolder.cmd”



This updates the Labnaf PowerShell configuration files following the “Labnaf_PowerShell” folder location.



Preconfigured
batches calling
commands with
predefined
settings

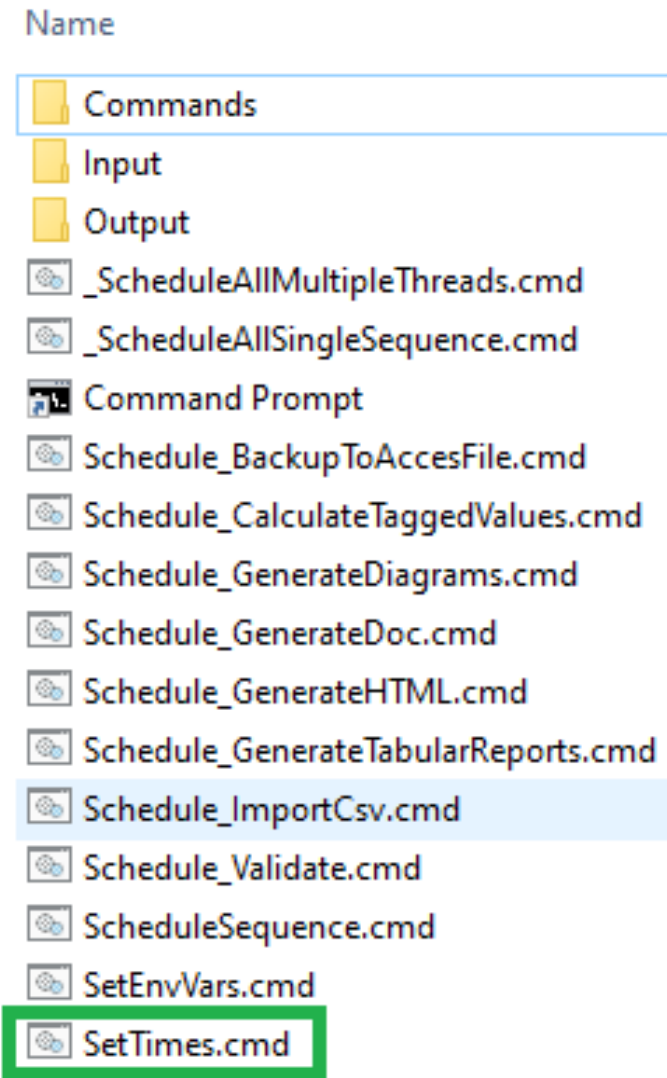
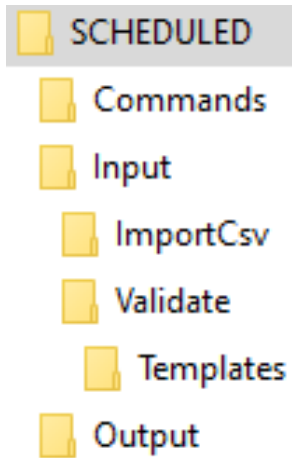


The paths in this command
were reset automatically

```
SetPaths.cmd x
```

```
1 set LABNAF_POWERSHELL=C:\Program Files (x86)\Labnaf\PowerShell\lnps.exe
2
3 set SCHEDULED_DIR=C:\Users\User\Desktop\Labnaf_PowerShell\SCHEDULED
4 set COMMANDS_DIR=%SCHEDULED_DIR%\Commands
5
6 set INPUT_DIR=%SCHEDULED_DIR%\Input
7 set OUTPUT_DIR=C:\Users\User\Desktop\Labnaf_PowerShell\SCHEDULED\Output
8
9 set REPOSITORY=%INPUT_DIR%\Repository.eap
```


Preconfigured Command Scheduling



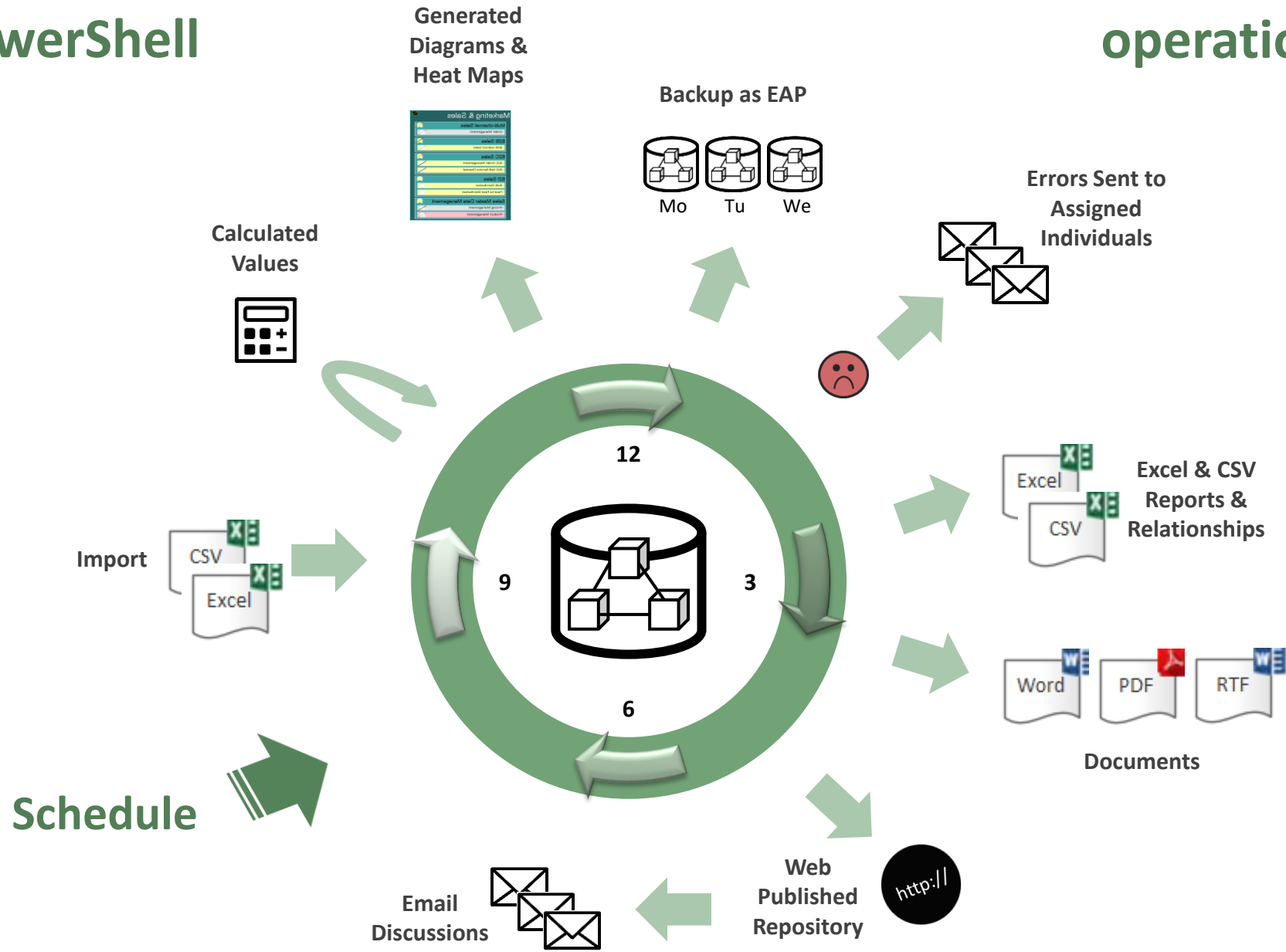
```
SetTimes.cmd
1  REM -- SINGLE START TIME --
2  Set StartTime_AllSingleSequence=00:00:00
3
4
5  REM -- SPECIFIC START TIME FOR EACH TASK --
6
7  Set StartTime_Cleanup_BackupToAccessFile=22:00:00
8  Set StartTime_Cleanup_GenerateHTML=22:00:05
9
10 Set StartTime_ImportCsv=22:30:00
11
12 Set StartTime_CalculateTaggedValues=23:00:00
13 Set StartTime_GenerateDiagrams=23:30:00
14
15 Set StartTime_BackupToAccessFile=00:00:00
16 Set StartTime_Validate=01:00:00
17
18 Set StartTime_GenerateTabularReports=02:00:00
19 Set StartTime_GenerateDoc=02:30:00
20 Set StartTime_GenerateHTML=03:00:00
21
22
23 REM -----
24
25 set SCHEDULED_MINUTES_UNTIL_RESTART=1440
```

Labnaf PowerShell Commands

1. Overview
2. Strategy and Architecture Operations
3. Systems Integrations and Content Refactoring
4. Command Compatibility Matrix

Labnaf PowerShell

Strategy and architecture operations



Labnaf PowerShell commands for Strategy and architecture operations

- Import Tabular Report (Excel, CSV)
- Calculate Values
- **Validate** and send emails to assigned individuals
- Generate Diagrams
- Generate Tabular Report (Excel, CSV), **DOC** (Word, RTF, PDF), **Html**
- Auto Connectors **Generate / Delete**
- Backup To Access File
- Schedule Command (not only Labnaf PowerShell commands)

ImportTabularReport

Sample input data for updating tagged values of existing elements

File to be imported can be .CSV or .XLSX (Excel)

	A	B	C	D	E
1	guid	application_owner	application_owner_delegates	it_responsible_service	legal_owner
2	{D303A068-2CAA-438d-9E81-287EE9777F1D}	homer.simpson@labnaf.local		Microsoft development	Labnaf
3	{305AA65E-A3F8-435b-81EC-C22EB7DF01C4}	marge.simpson@labnaf.local	lisa.simpson@labnaf.local	Enterprise Architecture	Labnaf
4	{07F7FA8B-A01C-4aed-B5C2-80C9D62BD3FF}	bart.simpson@labnaf.local		SAP development	Labnaf

OPTIONAL repository column mappings are stored in a .CSV file

	A	B
1	Input_Column_Names	Target_Column_Names
2	guid	ea_guid
3	application_owner	IT_Contact
4	application_owner_delegates	IT_Contact_Delegates
5	it_responsible_service	IT_ResponsibleService
6	legal_owner	Legal_Owner

Import Tabular Report (cont.)

To start the import:

```
C:\Program Files (x86)\Labnaf\PowerShell>lnps ImportTabularReport

Command: ImportTabularReport

Description: Import elements, properties and tagged values from a CSV or Excel file into a SQL Server database.

    If a field name mapping file (CSV) is provided, the first line must contain the following headers:
        Input_Column_Names, Target_Column_Names
    Works also with Access databases but only for updating existing elements.

    Identify existing elements using
        either the ea_guid,
        or an alternative unique key.
    To define a unique key, you simply add a '#' in front of the property or tag name
    If multiple unique keys are provided, they are searched in this order: Tagged Value, Name, Alias.

    Create the element if the element is not found and if 'EnableCreate' option is present on the command line.
    The package where new elements are added is identified by a package guid provided on the command line
    Initial value calculation applies to any imported new element.

Usage : lnps ImportTabularReport [arguments]

Arguments:

    RepoPathName: Path name of the model repository (EAP file).

    SourceFile: A CSV or Excel file containing the data that needs to be imported.

    ColumnMappingFile: An optional CSV file containing the mapping between the input and output column names or '-' if all input and output column names are the same.

    ElementStereotypeName: The stereotype of the elements that need to be updated.

    [EnableCreate]: Enable creation of new elements if they are not found.

    [TargetPackageGuidForNewElements]: The package where the new elements must be stored.
```

Beware of the different definitions of the “Notes” / “Note” property name in Sparx EA.

In the Sparx EA API, the property name is called “Notes”.

In the database, the corresponding column name is called “Note”.

For Imports

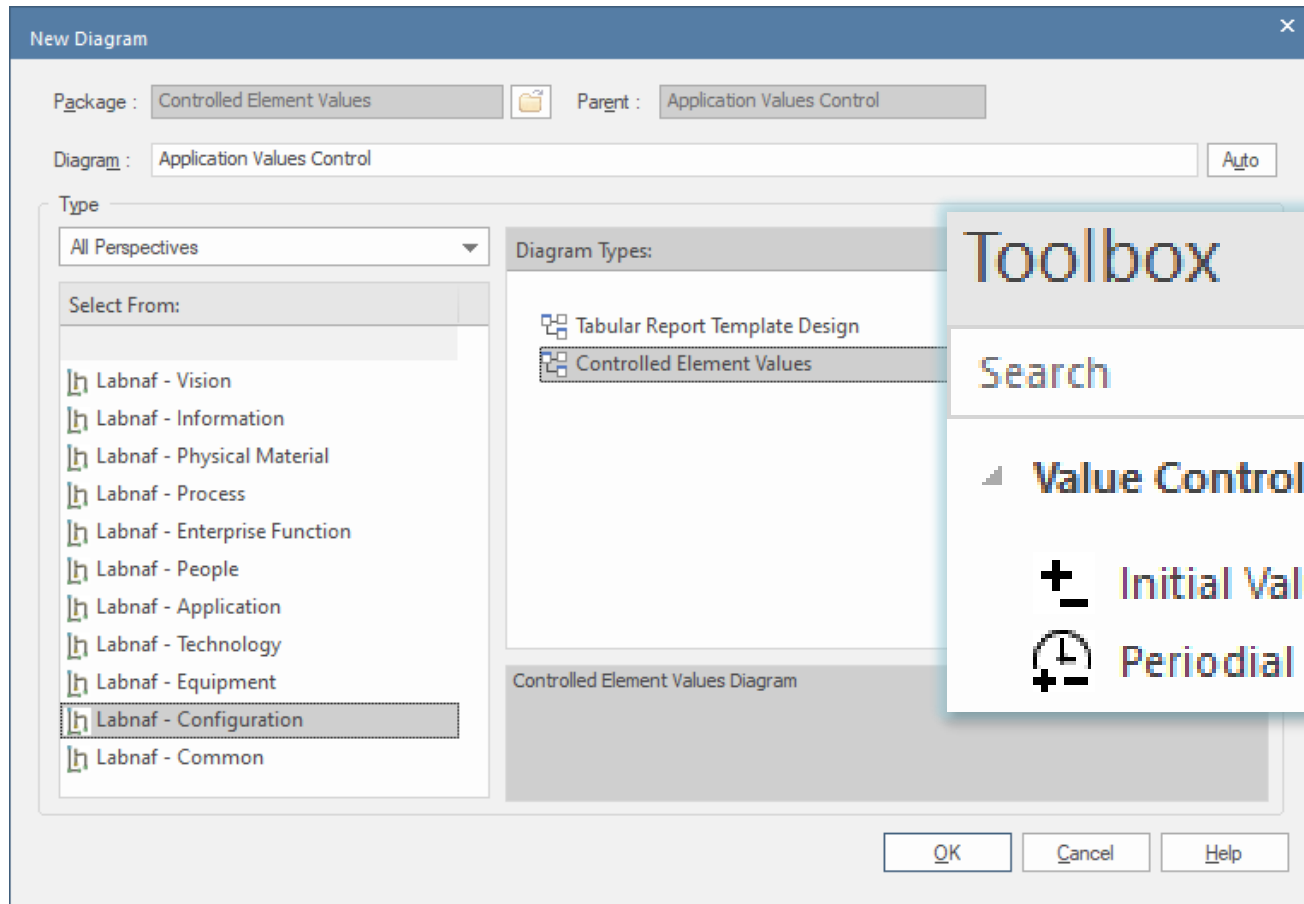
- In the import file use the word « **Notes** » (and not « Note »)

For SELECT statements

- In SQL Server, use the word « **Note** » in queries.
Example: `select Name, Note from t_Object`
- In the Sparx EA user interface, use « Note as {Something} ». Indeed « Note » is a reserved word.
Example: `select Name, Note as Element_Notes from t_Object`

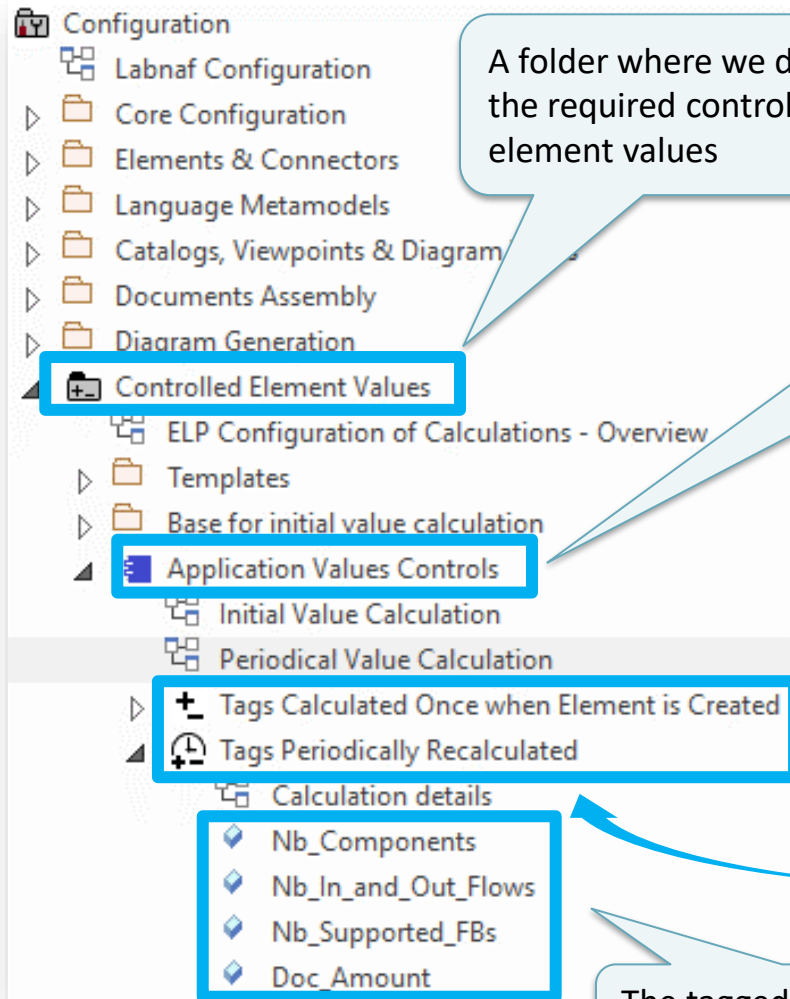
CalculateTaggedValues

Model your value calculations



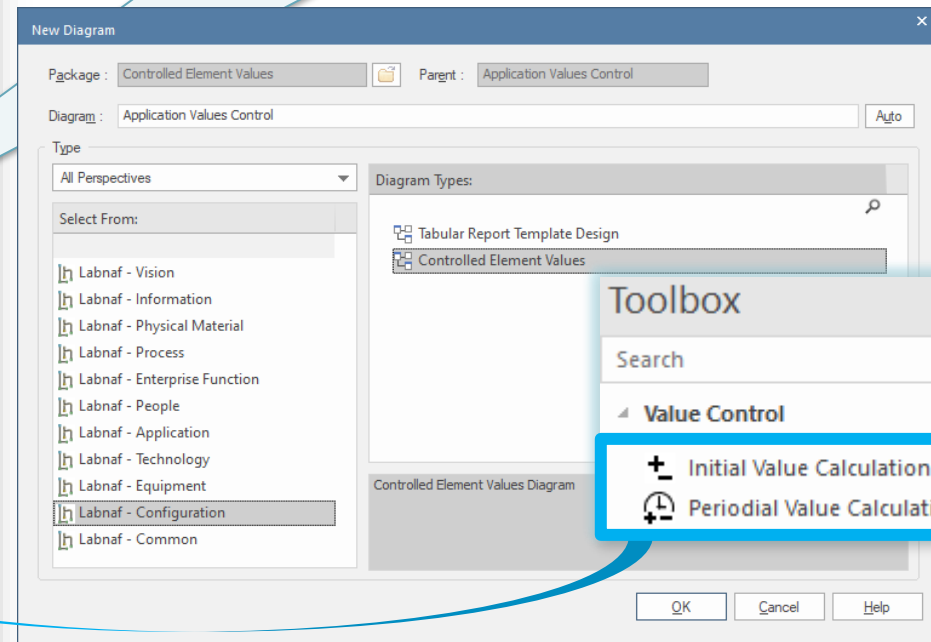
Calculate Tagged Values (cont.)

Model your value calculations



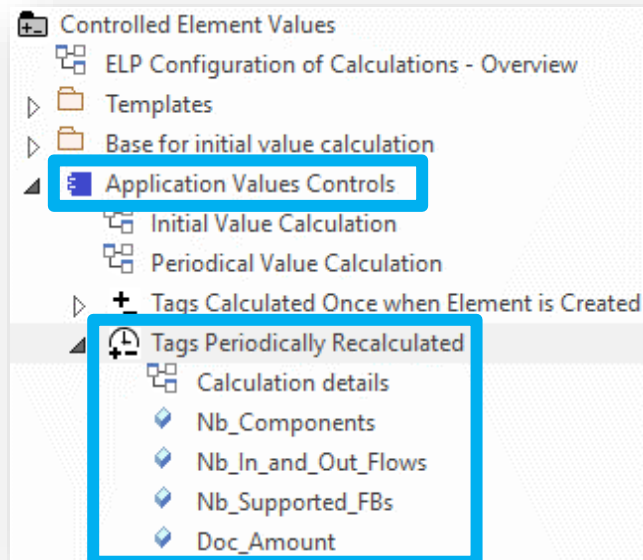
A folder where we define the required controls on element values

An element prototype for grouping all the required value controls. So we can see that the embedded value control definitions (calculations...) are for elements of that specific type and stereotype.

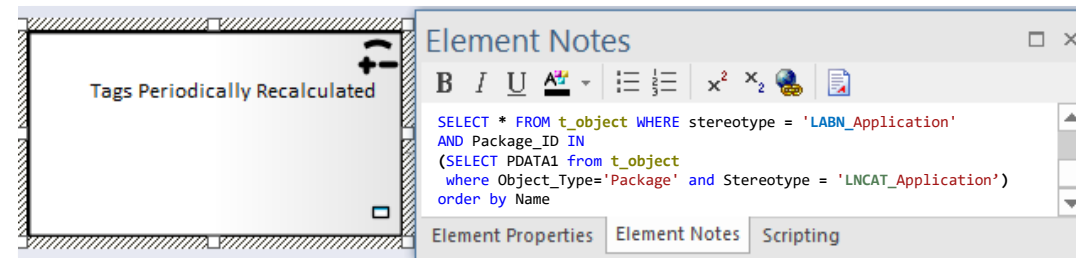


The tagged values that must be calculated

1. Structure: What tagged values need to be calculated for which stereotype?

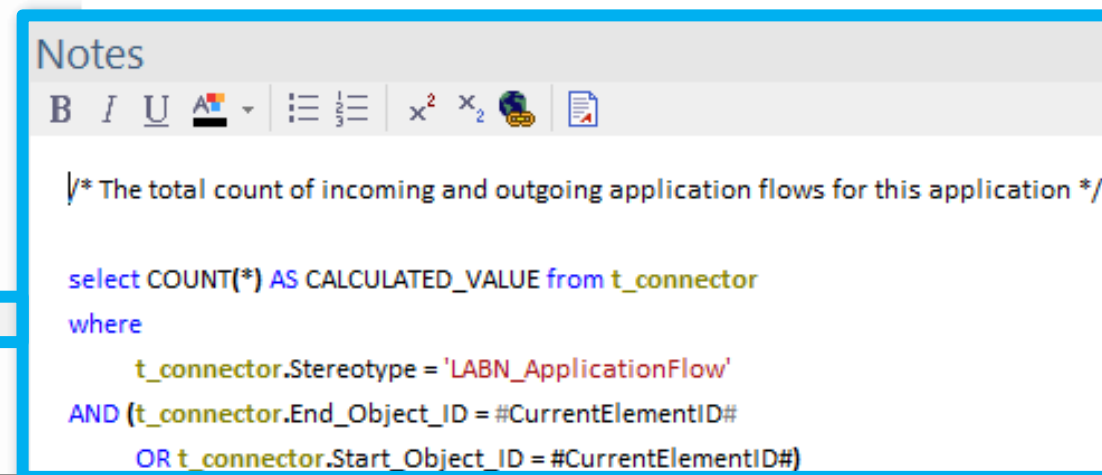
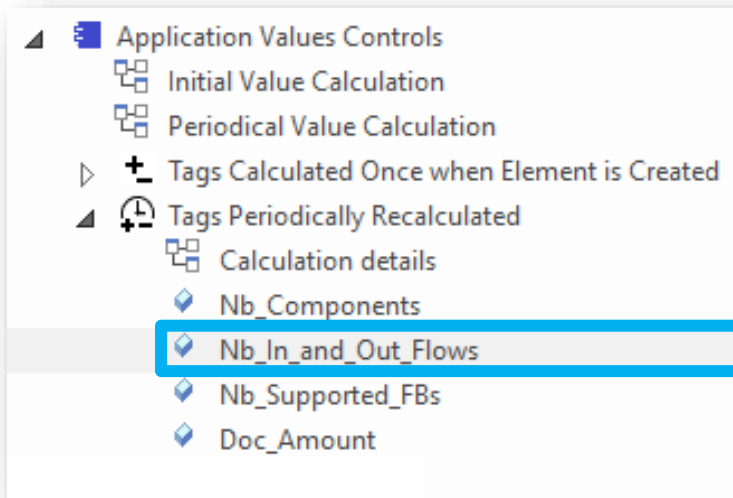


2. Scope (optional): Which elements need to be updated?



By default, all elements with the same stereotype "LABN_xxx" as the element prototype are selected from the related catalog with stereotype "LNCAT_xxx".

3. Calculation: How shall we calculate the value?



Calculate Tagged Values (cont.)

4. To start calculation:

Command: CalculateTaggedValues

Description: Calculate values for some defined tags and elements. The elements to be selected, the tags to be updated and the calculation formulas are all defined in the model repository.

Usage : Inps CalculateTaggedValues [arguments]

Arguments:

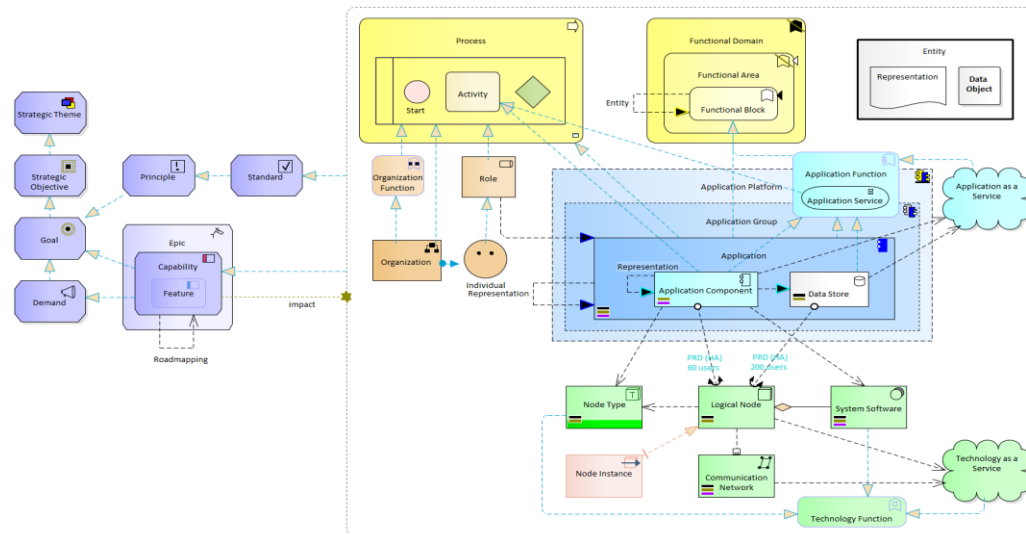
RepoPathName: Repository path name (EAP file).

[ElementPrototypeName]: A specific element stereotype for which tagged values must be calculated.

[TagName]: The name of a specific tagged value that must be calculated.

Validate

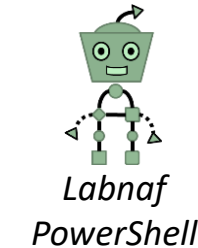
The Language Metamodel is used both for documentation & automatic model validation



While Modeling

Existing Invalid Connectors

Prevent creation of invalid connectors



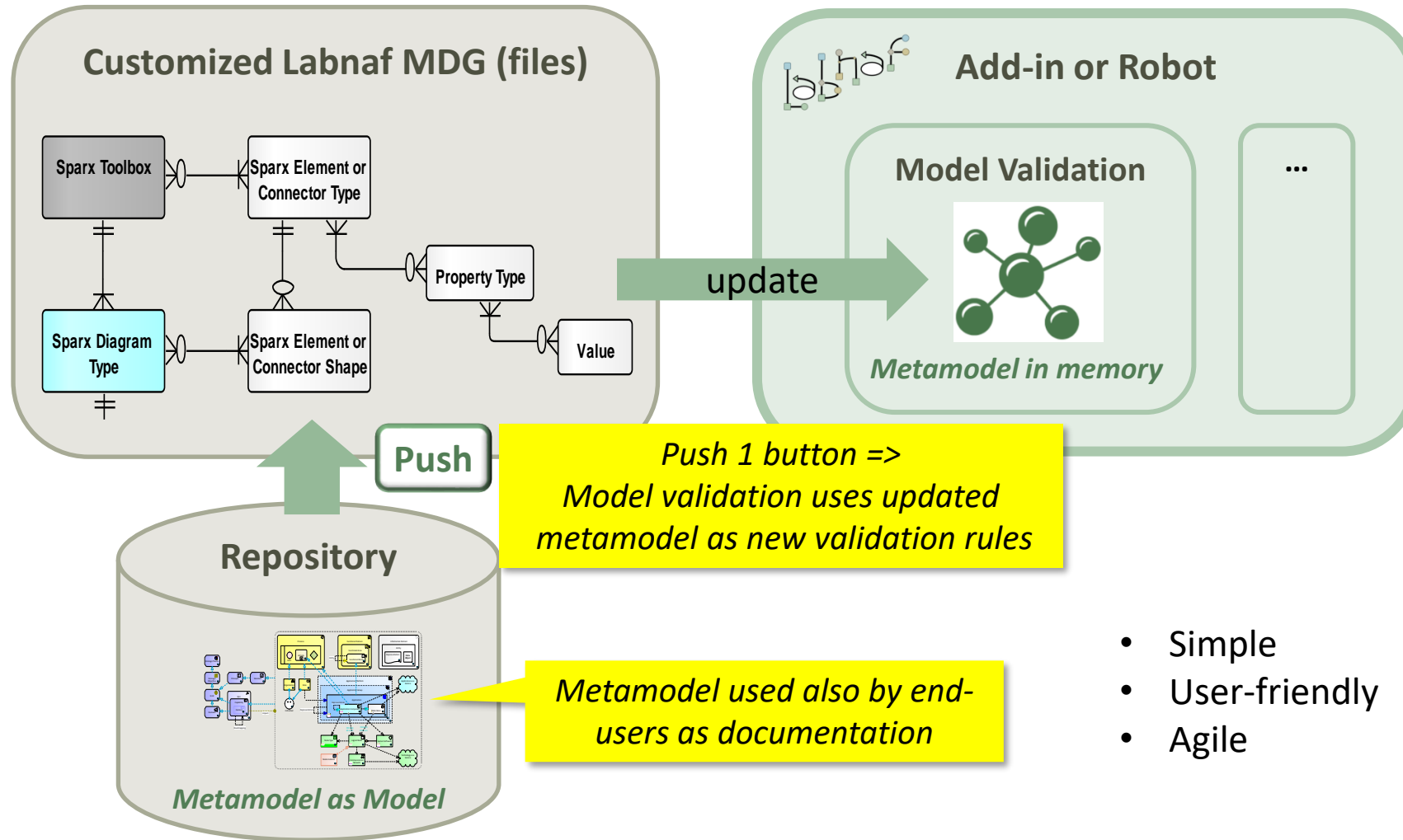
Send Error Emails to Relevant Recipients

Why do we need periodical validation?

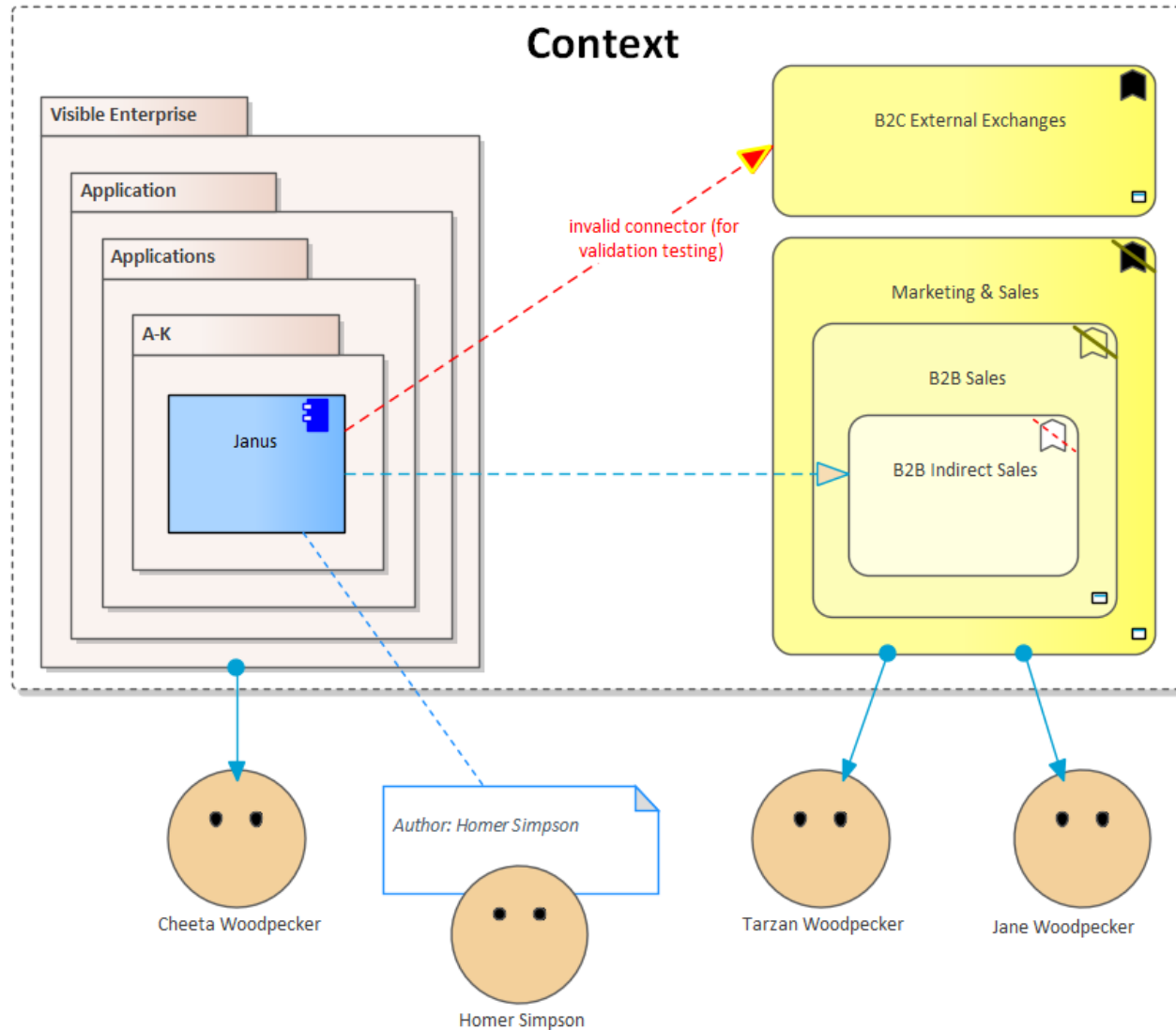
How could we have errors if we prevent users from entering errors?

- At the beginning, informal (invalid) models can be imported and their language can be transformed e.g. from ArchiMate to Labnaf.
- Then, every time you update the metamodel to adapt to your enterprise specificities, some existing model repository content becomes invalid... according to your new rules.

The default metamodel can be very easily updated:
One click on a connection in your production repository.



Error message routing is based on architecture management assignments



Error message distribution depends on

- recent authors
- individuals assigned to packages
- individuals assigned to domains
- default recipients
- and rules that you can use to combine these different combinations

The catalog of individuals includes email addresses

Sample error message sent to an assigned individual

The screenshot shows the Papercut email client interface. The top bar includes the application name 'PAPERCUT' and navigation icons for Log, Rules, Options, and Exit. The left sidebar displays a list of messages, with the selected one being 'From Model Validation Robot: Invalid objects.' dated 2019-10-16 21:37:19 (2.1KB). The main pane shows the email header and body. The header includes 'From: model.validation@labnaf.local', 'To: tarzan.woodpecker@Labnaf.local', 'Date: 2019-10-16 21:37:19 +02:00', and 'Subject: From Model Validation Robot: Invalid objects.' The body text reads: 'The [connector validation robot](#) identified some inconsistent content. Following our current knowledge, these problem(s) occur in a domain or package where you are personally involved in. Please make sure that the following [model repository](#) items get corrected either by you or by your team. Number of errors: 4' followed by four error entries: '[Labnaf Application Component](#) (LABN_ApplicationComponent) => [LABN_Triggering](#) => [Labnaf Activity](#) (LABN_Activity)', '[Labnaf Gateway](#) (LABN_Gateway) => [UML:Dependency](#) => [Labnaf Activity](#) (LABN_Activity)', '[Labnaf Application Component](#) (LABN_ApplicationComponent) => [UML:Aggregation](#) => [Labnaf Application Component](#) (LABN_ApplicationComponent)', and '[UML Component](#) (UML:Component) => [LABN_Realization](#) => [Labnaf Application Component](#) (LABN_ApplicationComponent)'.

Message Headers

From: model.validation@labnaf.local
 To: tarzan.woodpecker@Labnaf.local
 Date: 2019-10-16 21:37:19 +02:00
 Subject: From Model Validation Robot: Invalid objects.

Message Body

The [connector validation robot](#) identified some inconsistent content.
 Following our current knowledge, these problem(s) occur in a domain or package where you are personally involved in.
 Please make sure that the following [model repository](#) items get corrected either by you or by your team.

Number of errors: 4

- [Labnaf Application Component](#) (LABN_ApplicationComponent) => [LABN_Triggering](#) => [Labnaf Activity](#) (LABN_Activity)
- [Labnaf Gateway](#) (LABN_Gateway) => [UML:Dependency](#) => [Labnaf Activity](#) (LABN_Activity)
- [Labnaf Application Component](#) (LABN_ApplicationComponent) => [UML:Aggregation](#) => [Labnaf Application Component](#) (LABN_ApplicationComponent)
- [UML Component](#) (UML:Component) => [LABN_Realization](#) => [Labnaf Application Component](#) (LABN_ApplicationComponent)

Validation rules can be further customized

```

<ValidationConfiguration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd=
"http://www.w3.org/2001/XMLSchema">
  <SelectElementsInScope>
SELECT * FROM t_object WHERE stereotype like 'LABN_%' AND Package_ID IN (SELECT PDATA1
from t_object where Object_Type='Package' and Stereotype like 'LNCAT_%') ORDER BY Name
  </SelectElementsInScope>
  <Sender>model.validation@labnafdemo.com</Sender>
  <SendTo>
    <FirstAvailableAlternativeOnly>true</FirstAvailableAlternativeOnly>
    <PeopleAssignedToDomain>true</PeopleAssignedToDomain>
    <AuthorDuringMonthsAfterElementCreated>120</AuthorDuringMonthsAfterElementCreated>
    <PeopleAssignedToPackage>true</PeopleAssignedToPackage>
    <DefaultEmailAddresses>lisa.simpson@labnaf.local</DefaultEmailAddresses>
  </SendTo>
  <PublishedRepositoryWebSiteUrl>http://localhost/guidance</PublishedRepositoryWebSiteUrl>
  <DocumentationReferences>
    <GuidanceWebSiteUrl>http://www.Labnaf.one/guidance</GuidanceWebSiteUrl>
    <DiagramGuids>
      <ConnectorValidation>{269E2D0C-3B9E-4d85-915A-87905EB7271F}</ConnectorValidation>
      <ModelRepository>{EF41E336-AC6B-4407-88D9-3ECC41725132}</ModelRepository>
    </DiagramGuids>
  </DocumentationReferences>
</ValidationConfiguration>

```

If you want to be specific about the elements to be validated. By default all Labnaf elements are validated.


Error messages are sent from this email address.

Who will receive the error messages.

Error messages contains urls to invalid elements. HTML publication should be scheduled as well

Smtp Server configuration is straightforward

```
<?xml version="1.0" encoding="utf-8"?>  
<SmtpServerConfiguration xmlns:xsi="http://w  
  <Host>127.0.0.1</Host>  
  <ClientPort>25</ClientPort>  
  <EnableSSL>>false</EnableSSL>  
  <UserName>alain@labnafdemo.com</UserName>  
  <Password></Password>  
</SmtpServerConfiguration>
```



Needed to send error messages to assigned individuals

To start validation:

```
C:\A\LT\SparxDev\VSP\Projects\Alain\LabNaf\PowerShell\bin\Release\Dotfuscator>lnps validate  
Command: Validate  
Description: Validate model repository.  
Usage : lnps Validate [arguments]  
Arguments:  
  RepoPathName: Repository path name (EAP file).  
  ValidationConfigurationFile: Path name of the model validation configuration file.  
  SmtpServerConfigurationFile: Path name of the SMTP Server configuration file.
```

Enterprise Function Taxonomy & Applications Supporting Level 1

Marketing & Sales

- Multi-channel Sales
 - Order Management
- B2B Sales
 - B2B Indirect Sales
- B2C Sales
 - B2C Order Management
 - B2C Self-Service Channel
- B2I Sales
 - Bulk Distribution
 - Face-to-Face Distribution
- Sales Master Data Management
 - Pricing Management
 - Product Management

List of applications supporting the domain.

'Application Name'
Ares
Athena Cash Desk
Customer Mobile Application
Demeter
Hera
Janus
Jupiter Cash Desk
LOGIN B2B
Neptune
Venus Cash Desk
Zeus Pricing
Zeus Sales Records Management

Legend for Business Functions

Differentiator = ?	
= Y	
= N	

Externalized = Y	◀ Entirely externalized
= P	◀ Partially externalized
= N	Not externalized (default)

Nb of Applications / Business function

	0
	1
	2 or more

Other diagram(s) for this enterprise function:

[FULSales](#)

Applications managed by organizations

IT

- IT for IT
 - CMDB
 - Labnaf Powered by Enterprise Architect
- IT Marketing & Sales Apps
 - Athena Cash Desk
 - Customer Mobile Application

Application Lifecycle / Vision

- New
- Invest
- Maintain
- Phase Out
- ?

Other diagram(s) for this organization:

[Configuration](#)

Sample Diagram Templates

Enterprise Function Taxonomy & Applications Supporting Level 1

The screenshot shows a software interface with a tree view on the left, a central diagram area, and a legend on the right.

Tree View:

- ntext Diagram Element
 - Configuration
 - Labnaf Configuration
 - Core Configuration
 - Diagram Generation
 - Diagram Generation
 - Periodical Diagram Generation - Elements or Packages In Scope
 - Diagram Templates
 - Applications Supporting Functional Domain
 - FAL - Applications Supporting Functional Domain** (highlighted)
 - List of applications supporting the domain.
 - Functional Domain
 - Functional Area
 - Functional Block
 - Applications Supporting Functional Area
 - FAL - Applications Supporting Functional Area

Central Diagram Area:

- Functional Domain
 - Functional Area
 - Functional Block

Legend for Business Functions:

Differentiator = ?	
= Y	
= N	

Externalized = Y	Entirely externalized
= P	Partially externalized
= N	Not externalized (default)

Nb of Applications / Business function

- 0
- 1
- 2 or more

Other diagram(s) for this enterprise function:

- Configuration

Applications managed by organizations

The screenshot shows a software interface with a tree view on the left, a central diagram area, and a legend on the right.

Tree View:

- Applications Supporting Organization
 - OBO - Applications Owned By Organization** (highlighted)
 - Organization1
 - Organization2
 - Organization3
 - Organization4
 - Organization5
 - Application1
 - Application2
 - Application3
 - Application4
 - Application5

Central Diagram Area:

- Organization1
 - Organization2
 - Organization3
 - Organization4
 - Organization5
 - Application1
 - Application2
 - Application3
 - Application4
 - Application5

Legend for Application Lifecycle / Vision:

- New
- Invest
- Maintain
- Phase Out
- ?

Other diagram(s) for this organization:

- Configuration

The screenshot shows a software interface with a tree view on the left and a list of 'Enterprise Functions' on the right. The tree view is expanded to show 'Periodical Diagram Generation - Elements or Packages In Scope'. The 'Enterprise Functions' list includes:

- B2C External Exchanges
- B2B External Exchanges
- Information Management
- Internal Exchange
- Strategy, Legal Affairs, Risk & Compliance
- Traveler Communication
- Marketing & Sales
- Traffic Management
- Station Management
- Bus Route Planning
- Bus Maintenance
- Security
- HR & Corporate services
- Finance
- Supply Chain
- Information Technology

A yellow callout bubble contains the text: **This is optional**
By default, all elements are selected from the related catalog

To start diagram generation:

Command: `GenerateDiagrams`

Description: Generate diagrams in a model repository.

Usage : `Inps GenerateDiagrams [arguments]`

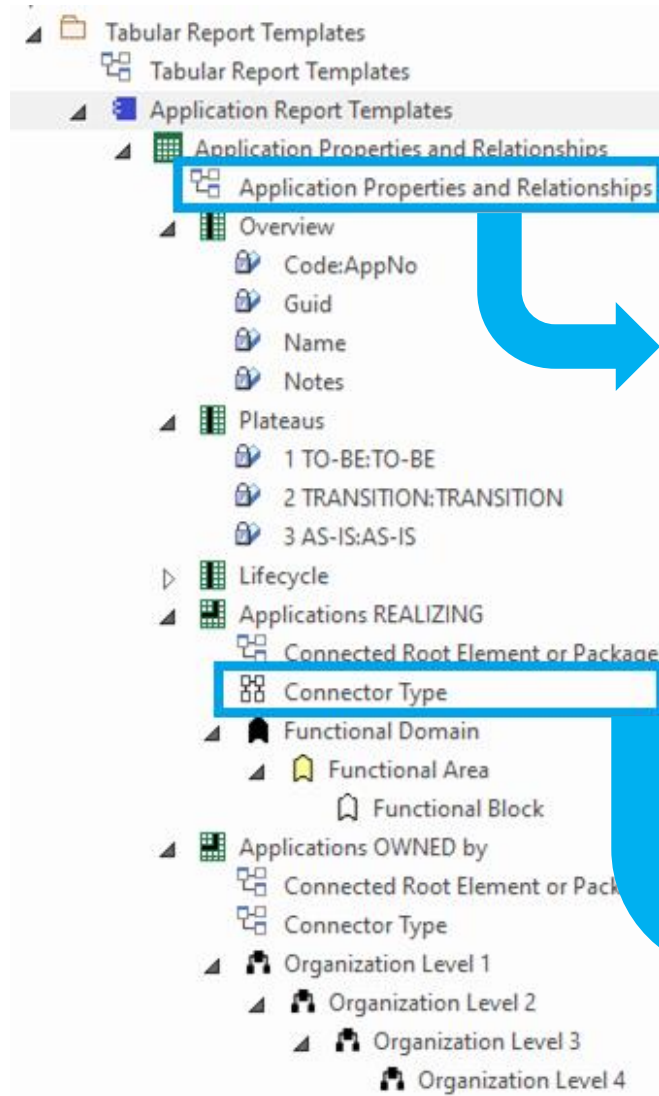
Arguments:

 RepoPathName: Path name (EAP file) of the repository where the diagrams must be generated.

 GenerationScopeDiagramGUID: A diagram containing organizations elements and/or a package of enterprise functions for which diagram generation is required.

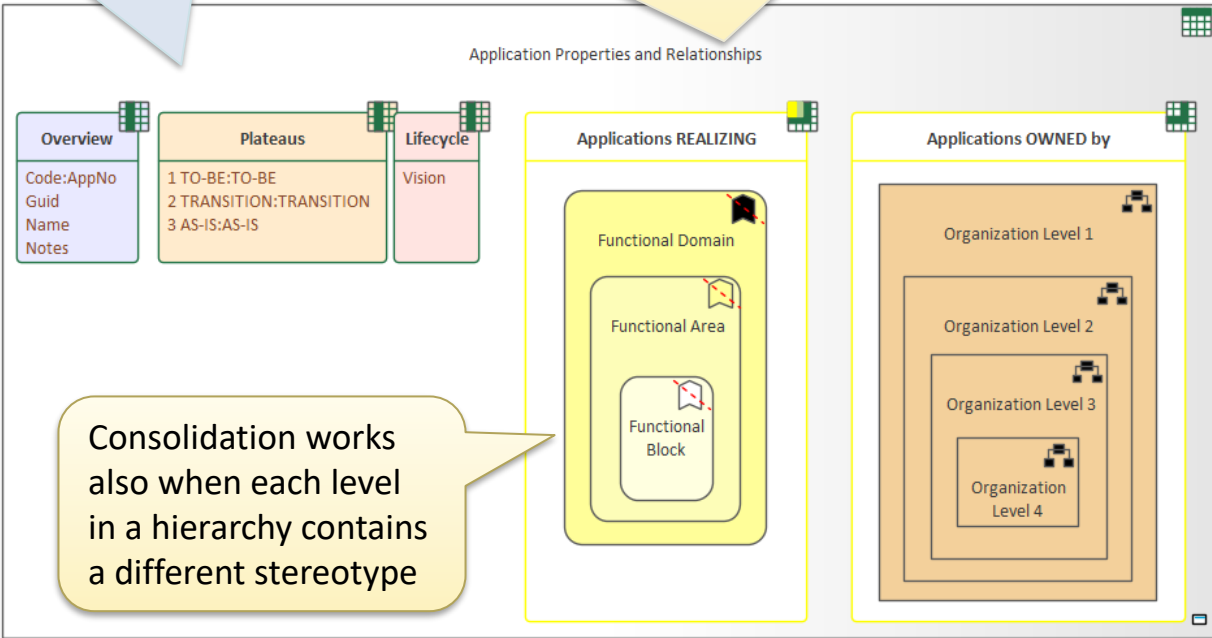
Generate Tabular Report (cont.)

Model your tabular report

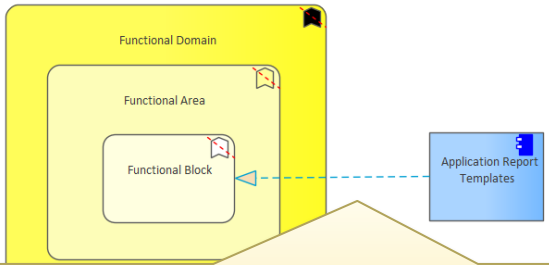
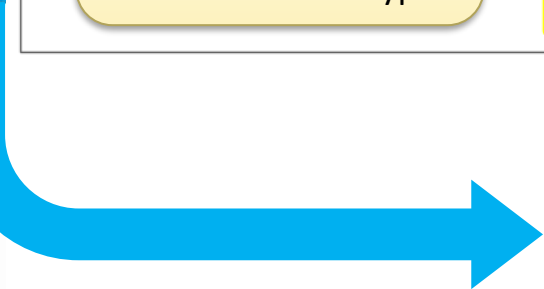


Model element properties and/or tagged values with optional renaming and colored groupings

Model specific connections in specific direction to specific types of elements. Model automatic connection **consolidation** into parent element relationships



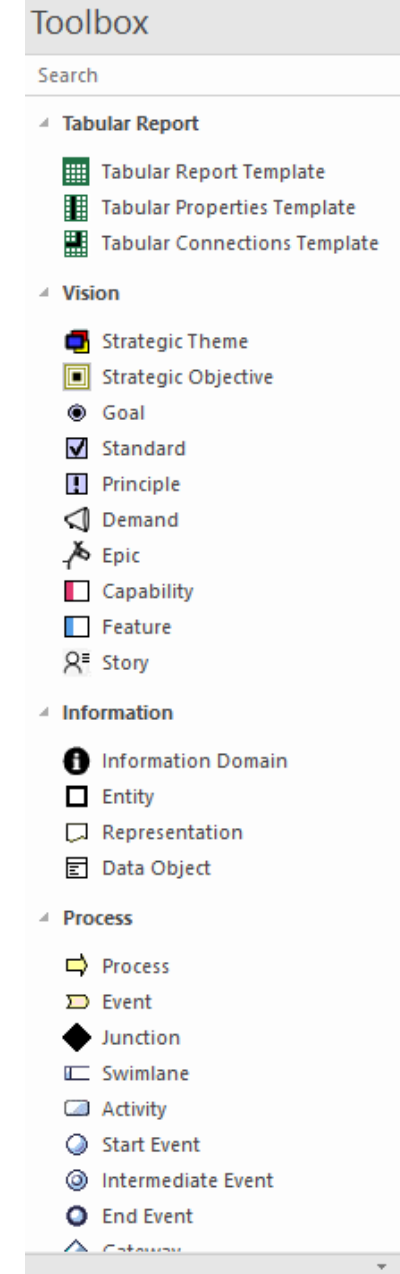
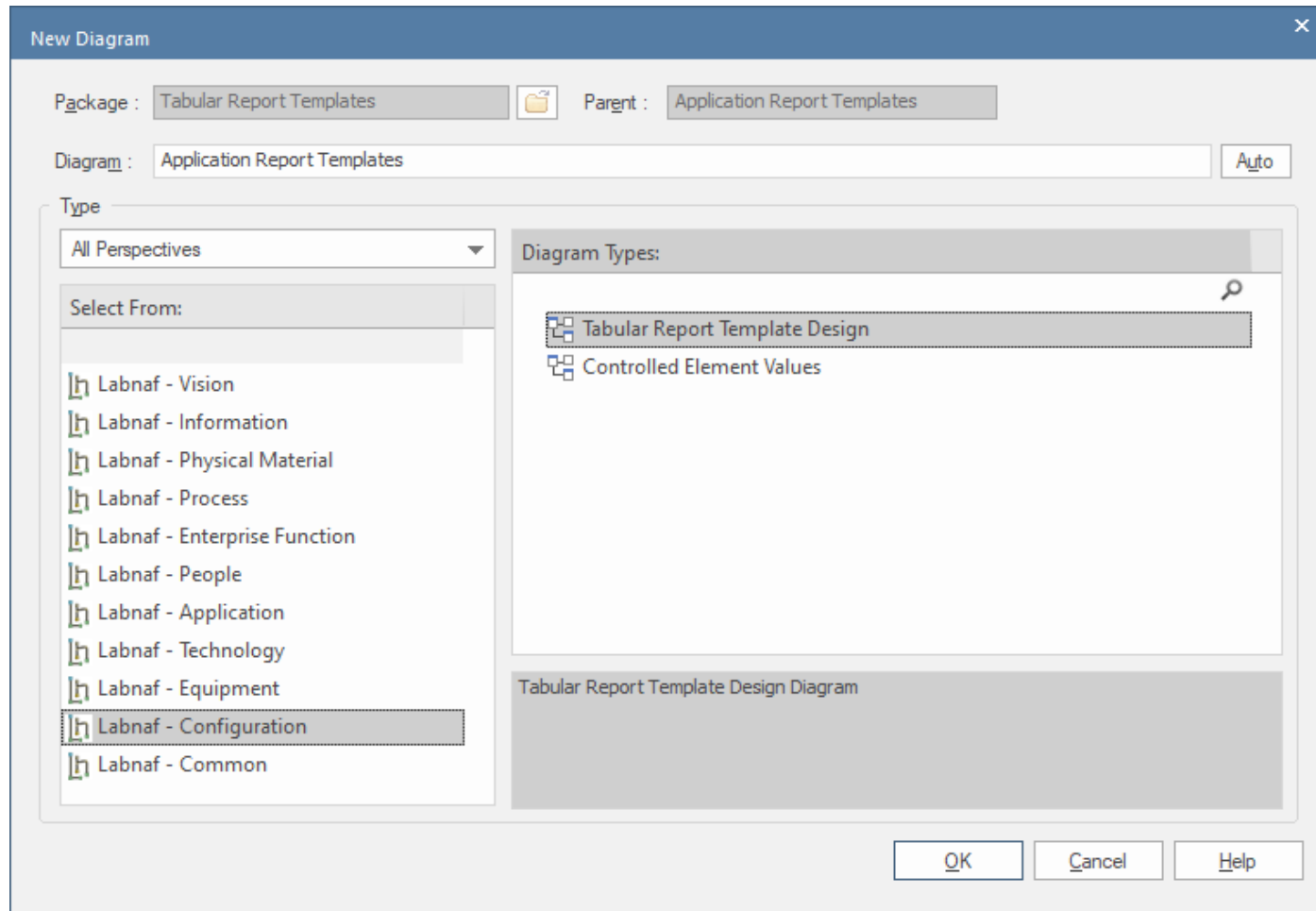
Consolidation works also when each level in a hierarchy contains a different stereotype



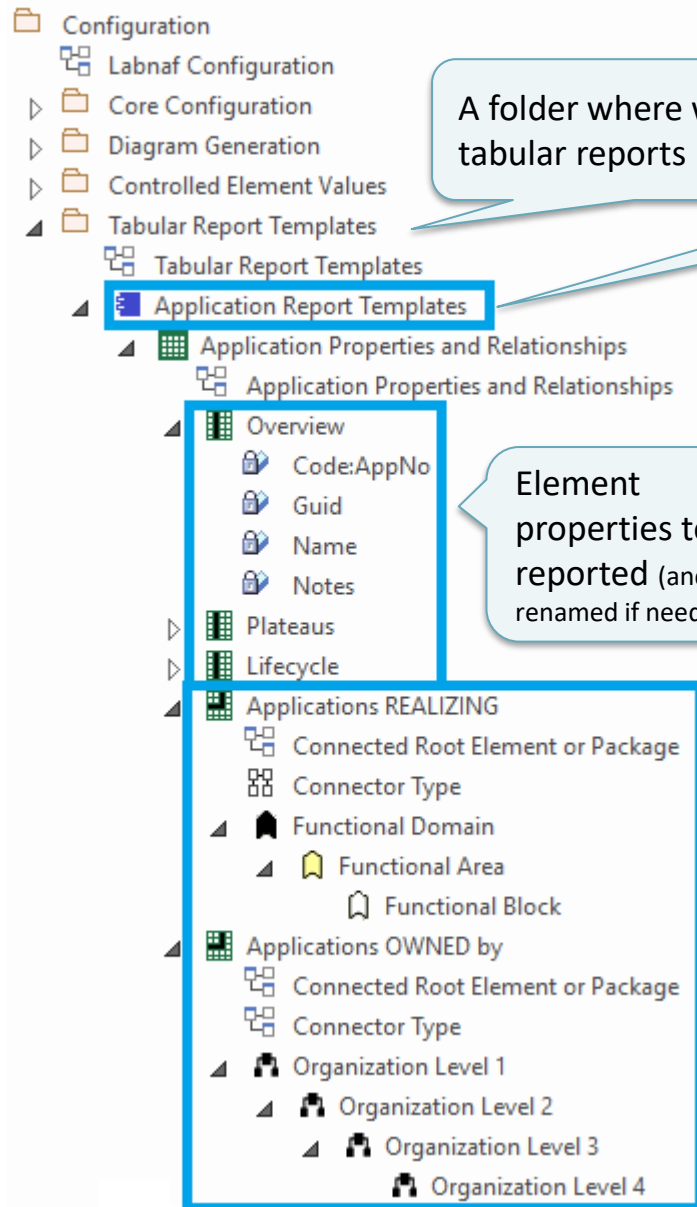
Model leaf element **connector** to be reported and consolidated

Model your tabular report

A tabular report can contain tagged values, properties and connections to any kind of element.



Model your tabular report

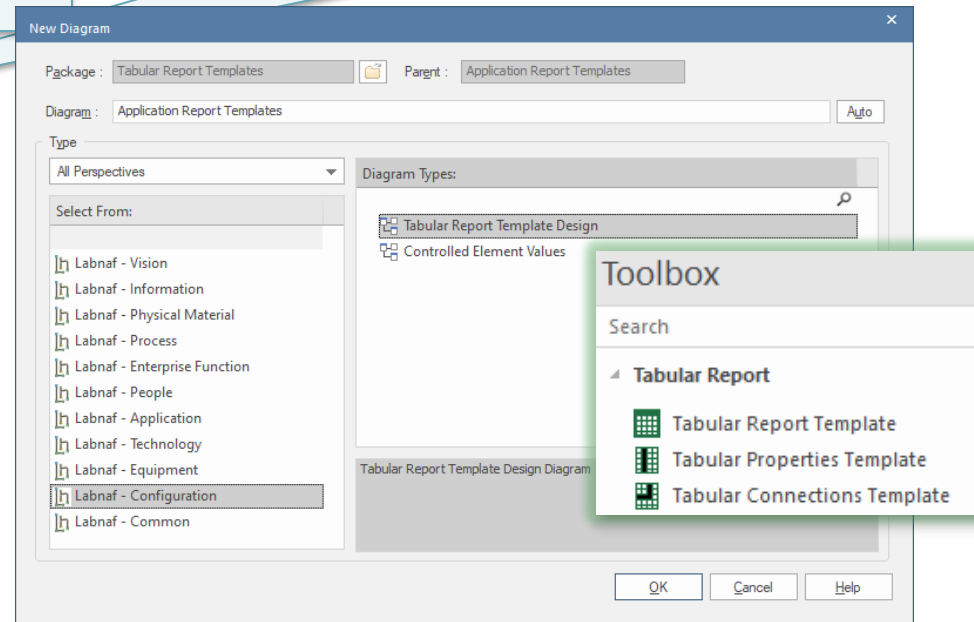


A folder where we define tabular reports

Element properties to be reported (and renamed if needed)

Element connections to be reported and consolidated at several levels of detail. Connected element types and stereotypes can be different at each level.

An element prototype for grouping tabular reports
So we can see that the embedded tabular report definitions are for elements of that specific type and stereotype.



Reported collection of elements (e.g. applications) selected following any kind of rule

Define the set of elements to be reported

By default, all elements with the same stereotype “LABN_XXX” as the element prototype are selected from the related catalog with stereotype “LNCAT_XXX”.

The screenshot shows a software interface with a tree view on the left and an 'Element Notes' window on the right. The tree view is titled 'Application Report Templates' and contains a folder 'Application Properties and Relationships'. Under this folder, there are two sub-folders: 'Overview' and 'Plateaus'. The 'Overview' folder contains four items: 'Code:AppNo', 'Guid', 'Name', and 'Notes'. The 'Plateaus' folder contains three items: '1 TO-BE:TO-BE', '2 TRANSITION:TRANSITION', and '3 AS-IS:AS-IS'. The 'Element Notes' window has a toolbar with various icons and a text area containing a SQL query. A blue callout box with white text points to the SQL query, stating 'Optional custom selection of the elements in scope'.

```
/* Selection in a 2 levels hierarchy of packages
SELECT * FROM t_object o
WHERE stereotype='LABN_Application'
AND o.Package_ID IN
(SELECT package1.package_ID FROM t_package package1
LEFT JOIN t_package package2 ON package2.package_id = package1.parent_id
WHERE package2.ea_guid = '{3E299811-1C55-43aa-841A-B1BC9869814A}')
ORDER BY Name
```

That SQL statements selects the elements that need to be included in the report.

With professional database engines, that SELECT statement can reach a level of sophistication that goes way beyond users' requirements.

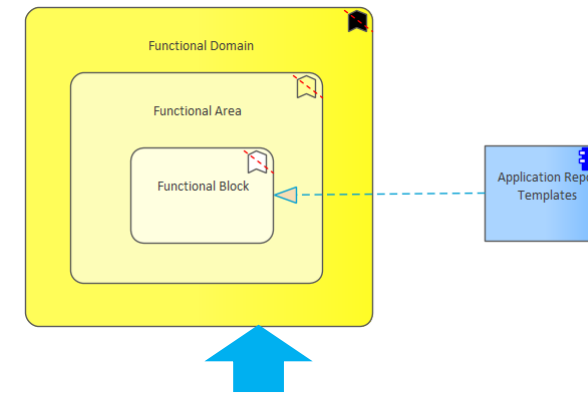
Access databases, on the other hand, have some limitations, but it is still usually sufficient to implement most use cases. Access databases are anyway not designed for running on professional database servers.

Generate Tabular Report (cont.)

If you want to limit the scope of the reported connections...

Put on a diagram the set of root elements to be selected at the other connection end.
The diagram can contain packages and elements.

Enterprise Function Levels



The screenshot shows the 'Application Report Templates' interface. On the left is a tree view with the following structure:

- Application Report Templates
 - Application Properties and Relationships
 - Application Properties and Relationships
 - Overview
 - Code:AppNo
 - Guid
 - Name
 - Notes
 - Plateaus
 - 1 TO-BE:TO-BE
 - 2 TRANSITION:TRANSITION
 - 3 AS-IS:AS-IS
 - Lifecycle
 - Applications REALIZING
 - Connected Root Element or Package
 - Connector Type
 - Functional Domain
 - Functional Area
 - Functional Block

Enterprise Functions

- B2C External Exchanges
- B2B External Exchanges
- Information Management
- Internal Exchange
- Strategy, Legal Affairs, Risk & Compliance
- Traveler Communication
- Marketing & Sales
- Traffic Management
- Station Management
- Bus Route Planning
- Bus Maintenance
- Security
- HR & Corporate services
- Finance
- Supply Chain
- Information Technology

To start tabular report generation:

```
Command: GenerateTabularReports

Description: Generate spreadsheets from a model repository based on configuration stored in that
same repository.

Usage : lnps GenerateTabularReports [arguments]

Arguments:

    SourceRepoPathName: Path name of the source model repository (EAP file).

    OutputDirectoryPath: Directory path name where the spreadsheets must be generated. The name of
each spreadsheet file is the name of the template report.

    [ElementPrototypeName]: The name of a specific element prototype name for which all embedded
tabular report templates must be applied.

    [TabularReportTemplateName]: The name of a specific tabular report template to be applied.
```

By default, all report templates will be applied.

But you can also be specific.

When a report template name ends with '.CSV' a CSV file is generated instead of Excel.

GenerateDoc

(Word, RTF, PDF)

To start document generation:

Command: `GenerateDoc`

Description: Generate a Word, RTF or PDF document from a model repository package.

Usage : `lnps GenerateDoc [arguments]`

Arguments:

`SourceRepoPathName`: Path name of the source model repository (EAP file).

`OutputPath`: Path name of the document file to be generated.

The file extension specified will determine the format of the generated document - for example, RTF, PDF

`PackageGuid`: The GUID of the package or master document to run the report on.

`TemplateName`: The document report template to use; if the PackageGUID has a stereotype of MasterDocument, the template is not required.

GenerateHtml

To start HTML generation:

Command: `GenerateHTML`

Description: Generate an HTML web site from a model repository package.

Usage : `Inps GenerateHTML [arguments]`

Arguments:

`SourceRepoPathName`: Path name of the source model repository (EAP file).

`OutputPath`: The path of the file system folder where the HTML pages must be generated.

`SourcePackageGUID`: The GUID of the repository package for which HTML must be generated.

`[WebSiteTemplateName]`: The optional name of a web style template used for HTML generation (default=Sparx EA default template).

On the web site, you can email a stable link to the current page by clicking on the little envelope.



AutoConnectorsGenerate

To start connector generation:

```
Command: AutoConnectorsGenerate
```

```
Description: Generate connectors for child elements following defined element stereotype hierarchies.
```

```
Usage : Inps AutoConnectorsGenerate [arguments]
```

```
Arguments:
```

```
SourceRepoPathName: Path name of the source repository (EAP file).
```

The generated connectors are aggregations.

Benefits of generated connectors

- Normalizes the way elements are related in a repository i.e. based on connectors
- Eases Prolaborate chart definitions: Prolaborate relies on connectors not on hierarchies

Sample Hierarchies for which aggregations are generated

- Functional Domain.Functional Area.Functional Block.Functional Category.Functional Service
- Application Platform.Application Group.Application.ApplicationComponent or Data Store
- Organization.Organization.Organization...

AutoConnectorsDelete

To start deletion of generated connectors:

```
Command: AutoConnectorsDelete
```

```
Description: Delete generated connectors for child elements following defined element stereotype hierarchies.
```

```
Usage : Inps AutoConnectorsDelete [arguments]
```

```
Arguments:
```

```
SourceRepoPathName: Path name of the source repository (EAP file).
```

The generated connectors are aggregations.

BackupToAccessFile

To start the backup to an Access file:

```
Command: BackupToAccessFile
```

```
Description: Backup a DBMS or Access repository to an Access Repository.
```

```
Usage : lnps BackupToAccessFile [arguments]
```

```
Arguments:
```

```
SourceRepoPathName: Path name of the source repository (EAP file).
```

```
DestEapPathName: Path name of the destination Access repository (EAP file).
```

```
LogFilePath: Path name of the log file name.
```

SourceRepoPathName (EAP) must point to a DBMS repository

ScheduleCommand

Alternatives to « ScheduleCommand » (If you prefer):

- **Windows Task Scheduler** (<https://docs.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-task-scheduler>)
- **Your company standard scheduler**

To schedule a **nightly** command **starting at midnight**:

- **InitialStartTime = 00:00:00** **Don't schedule 2 commands starting exactly at the same time**
- **PeriodAsMinutes = 1440** There are 1440 minutes in one day

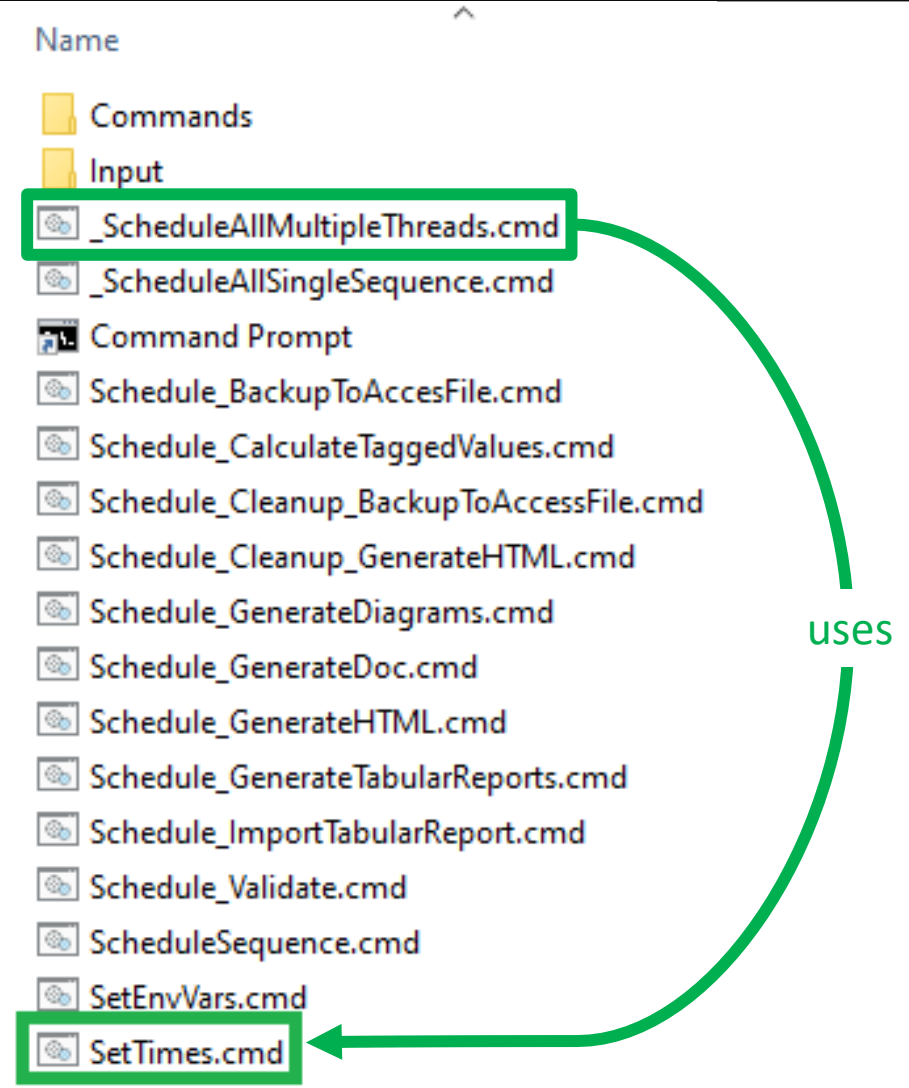
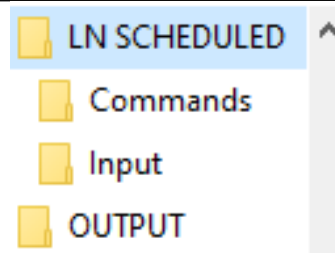
To start the scheduler:

```
Command: ScheduleCommand
Description: Schedule a task to run periodically starting at a specific time.
Usage : Inps ScheduleCommand [arguments]
Arguments:
    CommandPathName: Path name of the command that needs to be periodically started (.cmd or .bat).
    InitialStartTime: The initial start time for the task (HH:MM:SS).
    [PeriodAsMinutes]: The length of a period expressed in minutes.
```

Final Recommendation in case you use "ScheduleCommand"

- Use the preconfigured batches and settings

```
SetTimes.cmd
1 REM -- SINGLE START TIME --
2 Set StartTime_AllSingleSequence=00:00:00
3
4
5 REM -- SPECIFIC START TIME FOR EACH TASK --
6
7 Set StartTime_Cleanup_BackupToAccessFile=22:00:00
8 Set StartTime_Cleanup_GenerateHTML=22:00:05
9
10 Set StartTime_ImportTabularReport=22:30:00
11
12 Set StartTime_CalculateTaggedValues=23:00:00
13 Set StartTime_GenerateDiagrams=23:30:00
14
15 Set StartTime_BackupToAccessFile=00:00:00
16 Set StartTime_Validate=01:00:00
17
18 Set StartTime_GenerateTabularReports=02:00:00
19 Set StartTime_GenerateDoc=02:30:00
20 Set StartTime_GenerateHTML=03:00:00
21
22
23 REM -----
24
25 set SCHEDULED_MINUTES_UNTIL_RESTART=1440
```



- Take computer reboots into account

Labnaf PowerShell Commands

1. Overview
2. Strategy and Architecture Operations
3. Systems Integrations and Content Refactoring
4. Command Compatibility Matrix

Labnaf PowerShell commands for Systems integrations and content refactoring

- ClonePackage
- CreatePackage
- ExportToXmi
- ImportConnections
- ImportFromXmi
- ImportTabularReport
- MoveElementsToCalculatedParent
- MoveElementsToPackage
- MovePackagesToPackage
- RenameItem
- ScheduleCommand
- SetDiagramProperty
- SqlExportToCsv

Detailed information in the
Labnaf PowerShell Reference Guide

Latest version:

https://www.labnaf.one/EndUserMaterial/Labnaf_PowerShell/Labnaf%20PowerShell%20-%20Reference%20Guide.pdf

Labnaf PowerShell Commands

1. Overview
2. Strategy and Architecture Operations
3. Systems Integrations and Content Refactoring
4. Command Compatibility Matrix

Labnaf PowerShell

Command Compatibility Matrix

Power Shell Commands	Sql Server	Pro Cloud Server	Access
AutoConnectorsDelete	X	X	-> UI
AutoConnectorsGenerate	X		-> UI
BackupToAccessFile	X	X	
CalculateTaggedValues	X	X	
ClonePackage	X	X	X
CreatePackage	X	X	X
ExportToXmi	X	X	X
GenerateDiagrams	X	X	X
GenerateDoc	X		
GenerateHTML	X		
GenerateTabularReports	X	X	X
ImportConnections	X	X	
ImportFromXmi	X	X	X
ImportTabularReport	X		-> UI
MoveElementsToCalculatedParent	X	X	X
MoveElementsToPackage	X	X	X
MovePackagesToPackage	X	X	X
RenameItem	X	X	X
ScheduleCommand	X	X	X
SetDiagramProperty	X	X	X
SqlExportToCsv	X	X	X
Validate	X	X	X

Legend

-> UI : Use the Labnaf AddIn
i.e. the user interface