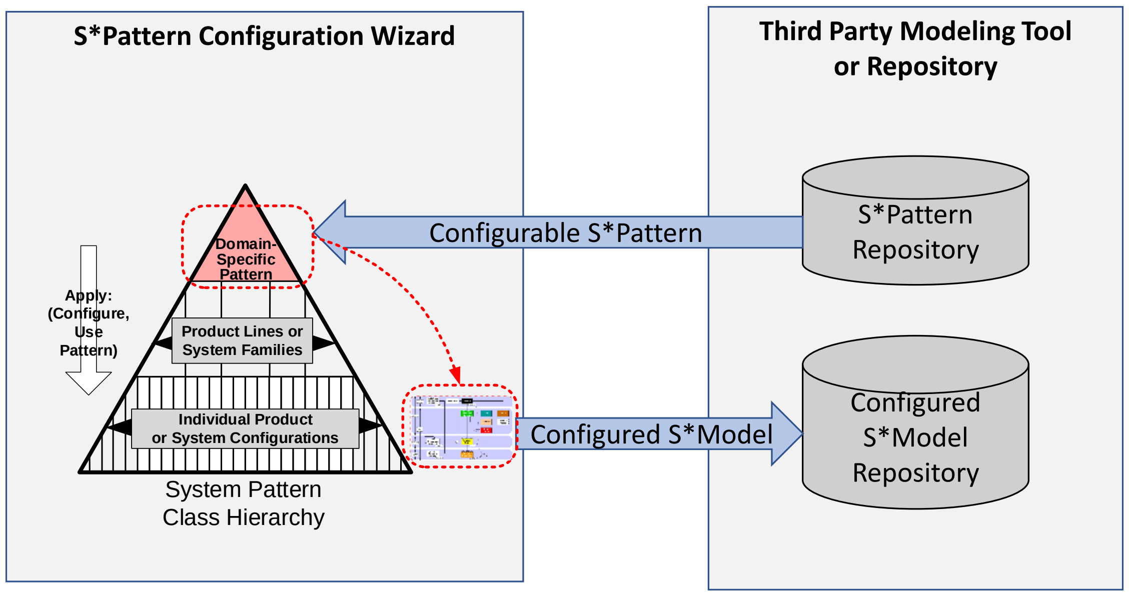
Guide to the S*Pattern Configuration Wizard


S*Pattern Configuration Wizard
Third Party Modeling Tool or Repository
Domain-Specific Pattern
Apply: (Configure, Use Pattern)
Product Lines or System Families
Individual Product or System Configurations
System Pattern Class Hierarchy
Configurable S*Pattern
S*Pattern Repository
Configured S*Model
Configured S*Model Repository

10/27/2022

Contents

1 Introduction 5

1.1 Document Purpose and Scope 5

1.2 Intended Readership and Prerequisite Knowledge 5

1.3 Introduction to the Pattern Configuration Wizard 5

1.4 Use with Third Party Modeling Tools and Languages 6

1.5 Compatible Software 6

1.6 Public Access 6

1.7 History of Configuration Wizard Technology Generations 6

1.8 Document History 7

1.9 References 7

2 Using the Configuration Wizard 8

2.1 Setting Up the Environment 8

2.2 The Wizard User Control Panel 8

2.2.1 The Control Panel Buttons 8

2.2.2 The Control Panel Check Boxes 12

2.2.3 The Control Panel Navigation Links 14

2.2.4 The Control Panel Progress and Error Logs 16

2.3 Performing a Preliminary Test 17

2.4 How to Configure a Pattern Using the Wizard 18

2.5 Saving and Retrieving Configuration Parameter Input Data 20

2.6 The Configuration Process Algorithm; Configuration Predecessors and Successors 22

2.7 Typical Performance Times for Configured Model Generation 24

2.8 Purging the Configuration Wizard of Confidential Data 25

3 External Interface Specification and Wizard Internals 26

3.1 The Wizard External Interfaces Specification 26

3.2 The Internal Data Flow, Queries Architecture, and JOIN Technology 26

3.3 The Wizard Internal Control Program and Source Code 26

3.4 The Wizard Queries and their Source Code 27

4 Appendices 28

4.1 Appendix A: Wizard External Interfaces Specification—Sample Extract 28

4.2 Appendix B: Control Program Source Code—Sample Extract 30

4.3 Appendix C: Detail Query Source Code—Sample Extract 33
4.4 Appendix D: Example Pattern and Configured Model for Preliminary Testing 35
4.5 Appendix E: Pattern File Names and Paths Table 36
4.6 Appendix F: Data Flow and Queries Architecture Diagrams 39

1 Introduction

1.1 Document Purpose and Scope

This document describes the S*Pattern Configuration Wizard, including its use and internal design documentation.

1.2 Intended Readership and Prerequisite Knowledge

Section 2 is intended for users of the Configuration Wizard. It assumes general familiarity with S*Models and S*Patterns, from the perspective of a person using the Configuration Wizard to generate configured S*Models from reusable S*Patterns. Consult the References for background on S*Models and S*Patterns.

Section 3 is intended for methodologists and tool integrators and maintainers. It provides both high level and detailed documentation of the design of the Configuration Wizard and its intended integration with modeling tools and repositories. Consult the References for detail documentation on the S*Metamodel and its mapping to third party tools and languages.

1.3 Introduction to the Pattern Configuration Wizard

The Pattern Configuration Wizard is a packaging of the automated algorithms for generating configured S*Models from (and conforming to) reusable S*Patterns. It is intended to be integrated with commercial and public domain modeling tools and repositories, as in Figure 1. It is provided with open-source code and documentation for ease of integration, support, customization, or as an example for creation of similar or modified capabilities.

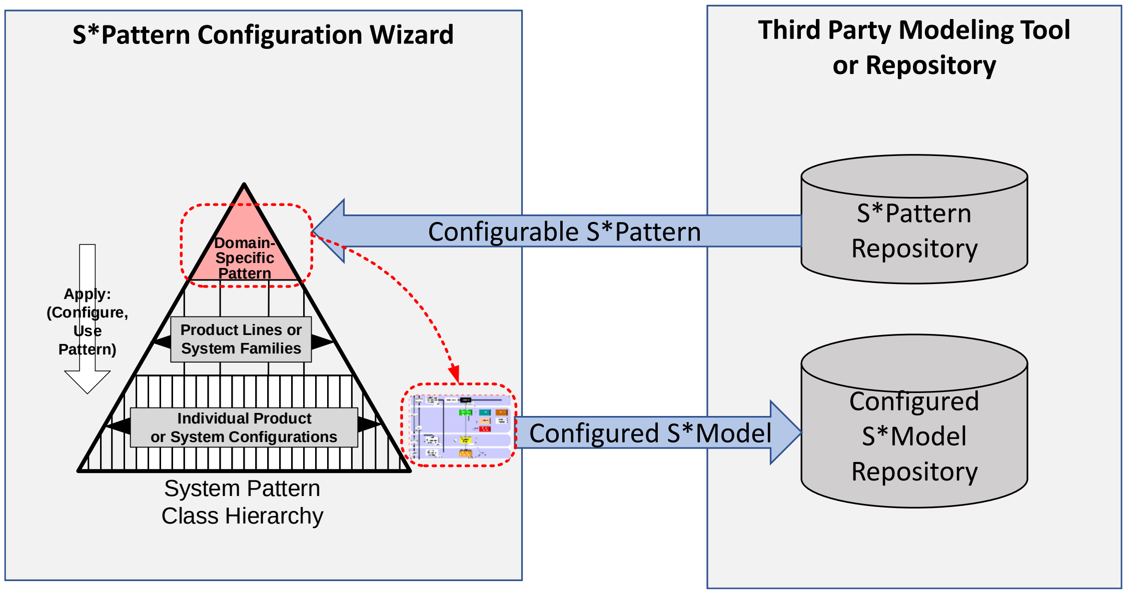

S*Pattern Configuration Wizard
Third Party Modeling Tool or Repository
Domain-Specific Pattern
Apply: (Configure, Use Pattern)
Product Lines or System Families
Individual Product or System Configurations
System Pattern Class Hierarchy
Configurable S*Pattern
S*Pattern Repository
Configured S*Model
Configured S*Model Repository

*Figure 1: Integrated with a Third-Party Modeling Tool or Repository, the Pattern Configuration Wizard Generates Configured S*Models from Reusable S*Patterns*

1.4 Use with Third Party Modeling Tools and Languages

The Pattern Configuration Wizard is in effect a specialized algorithm add-on to an existing model authoring tool or model repository. It is intended to be relatively portable across such tools and repositories, by utilizing published neutral interfaces to them, and applying semantic mappings based on the neutral reference S*Metamodel. It assumes that (1) the modeling tool with which it is integrated has been formally mapped to the S*Metamodel using a documented mapping, (2) that the S*Patterns it will receive from that tool conform to the S*Metamodel mapped to the local tool and language, and (3) that the configured S*Models it sends back to the modeling tool can be received by that tooling. Refer to Figure 1. The mappings include both semantics and package folder structure.

Accordingly, the Pattern Configuration Wizard assumes that the modeling system with which it is integrated has been equipped with the relevant export and import capabilities to generate and receive the exchanged S*Patterns and configured S*Models shown in Figure 1, as specified by the Wizard Interface Specification discussed in Section 3.1.

1.5 Compatible Software

The S*Configuration Wizard version described by this document has been implemented in using the database query engine capabilities of Microsoft Power Query, within the Microsoft Excel platform. It assumes a run time environment consisting of Microsoft Windows® 10 and Microsoft Excel® 2016, or equivalents. The current version of Configuration Wizard at the time of this document is V1.12.25.

1.6 Public Access

Along with related publications of the S*Metamodel, S*Metamodel mapping to the local tooling, and related S*Patterns literature, the S*Configuration Wizard is provided under Creative Commons public licensing with its source code and descriptive documentation, to advance the practice of pattern-based methods in Model-Based Systems Engineering (MBSE). It is accordingly being used in related projects of the INCOSE MBSE Patterns Working Group, and available to the public, without warranty as to performance or fitness. Consult the References.

1.7 History of Configuration Wizard Technology Generations

The version of the S*Patterns Wizard described by this document is the third in a series of generations of automated configuration agents for S*Patterns. Those generations have refined the practice and related automation of pattern-based methods, in this history:

Table 1: Generations of Pattern Configuration Agent Technologies

Generation and Time of Origin	**Configuration Agent Generation**	**Compatible Modeling Tools**	**Scalability**
First generation, 2005	SE Pattern Workbook: Based in desktop spreadsheet automation for self-contained S*Patterns and S*Models.	Self-contained	Small and medium-scale models and patterns.
Second generation, 2013	SE Pattern Agent: Desktop agent software integrated as add-on to commercial third party MBSE tooling across multiple COTS tools.	Models and patterns in DOORS®, Requisite Pro®, Enterprise Architect SysML ®, Magic Draw / Cameo Systems Modeler SysML®	Small and medium-scale models and patterns.
Third generation, 2021	Configuration Wizard: Based in desktop or server bulk JOIN software integrated as add-on to commercial third party MBSE tooling across multiple toolsets.	Generic OMG SysML®, Dassault Cameo Systems Modeler® SysML, Sparx Enterprise Architect® SysML.	Medium and large-scale models and patterns.

1.8 Document History

Version	Date	Content
1.2.7	08.31.2022	Consolidate material from earlier documentation, for new edition.
1.2.8	10.27.2022	Correct header formatting. Add confidential data purging section.

1.9 References

1. "Systematica® Metamodel, Version 8.0", System Sciences, LLC, 2022.
2. "Using OMG SysML™ With Systematica Methodology Release 4.0: Mapping Guide Configured for Sparx Systems Enterprise Architect™ V15", System Sciences, LLC, 2022.
3. "S*Metamodel Mapping for MagicDraw/Cameo Systems Modeler Version 19", System Sciences, LLC, 2019. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:systematica_mapping_for_magicdraw_csm_v1.9.1a.pdf
4. Schindel, W., and Peterson, T., "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques", tutorial in Proc of INCOSE 2016 Great Lakes Conference on Systems Engineering. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:pbse_tutorial_glrc_2016_v1.7.4.pdf
5. Schindel, W., "Realizing the Promise of Digital Engineering: Planning, Implementing, and Evolving the Ecosystem", Proc. of INCOSE 2022 International Symposium, Detroit, MI. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:realizing_the_vision_of_digital_engineering_is2022_v1.3.4.pdf
6. "MBSE Patterns Working Group", Working Group Meeting June, 2022. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:mbse_patterns_wg_mtg_is2022_06.26.2022_v1.3.5_.pdf
7. Schindel, W., "What Is the Smallest Model of a System?", *Proc. of the INCOSE 2011 International Symposium*, International Council on Systems Engineering, 2011. http://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:what_is_the_smallest_model_of_a_system_v1.4.4.pdf
8. Schindel, W., Lewis, S., Sherey, J., Sanyal, S., "Accelerating MBSE Impacts Across the Enterprise: Model-Based S*Patterns", Proc. of INCOSE 2015 International Symposium, July, 2015. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:accelerating_mbse_impacts_across_the_enterprise_using_model-based_s-patterns_v2.1.1.pdf
9. Patterns WG. "Methodology Summary: Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models", INCOSE Patterns Working Group, 2019. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:pbse_extension_of_mbse--methodology_summary_v1.6.1.pdf
10. INCOSE MBSE Patterns Working Group Web Site. https://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns
11. "Semantic Technologies for Systems Engineering (ST4SE): Project Report", INCOSE MBSE Patterns Working Group, 2022.

2 Using the Configuration Wizard

2.1 Setting Up the Environment

The initial environment that must be established consists of:

1. The modeling tool or model repository, in which configurable S*Patterns will be found for use by the Configuration Wizard, and to which configured S*Models can be returned after they are created by the Configuration Wizard.
    a. This modeling tool must include a semantic profile for its models that has been constructed from a formal mapping of the S*Metamodel into the semantic space of the tool. This formalizes the semantics of S*Patterns and S*Models in the modeling tool. Such profiles and mappings are available for a number of popular contemporary modeling tools and repositories.
    b. The modeling tool should have been equipped with pattern export and model import capabilities for exchange of patterns and models with the Configuration Wizard, per the Interface Specification of Appendix A.
    c. The environment in which the modeling tool operates should have file exchange access with the environment in which the Configuration Wizard operates—whether in the same computer or across a network.
2. The desktop environment in which the Configuration Wizard will be run, to generate configured models from configurable patterns.
    a. The Configuration Wizard (an .xlsm file) should be installed in this environment.
    b. The installed Configuration Wizard provides a Control Panel, described in the next section, which can be used to identify additional file system location environmental details described later below. The environment in which the Configuration Wizard will run should have file exchange access with the environment in which the Modeling Tool operates—whether in the same computer or across a network.

2.2 The Wizard User Control Panel

Start the Configuration Wizard, and view its Control tab to see the Wizard Control Panel, shown in Figure 2 below. Note that the Control Panel includes several Buttons, used to control the Configuration Wizard; a set of Logs, used to display progress or errors during use of the Configuration Wizard; a set of Check Boxes, used to manage Configuration Wizard options; and a set of Navigation Links for the user.

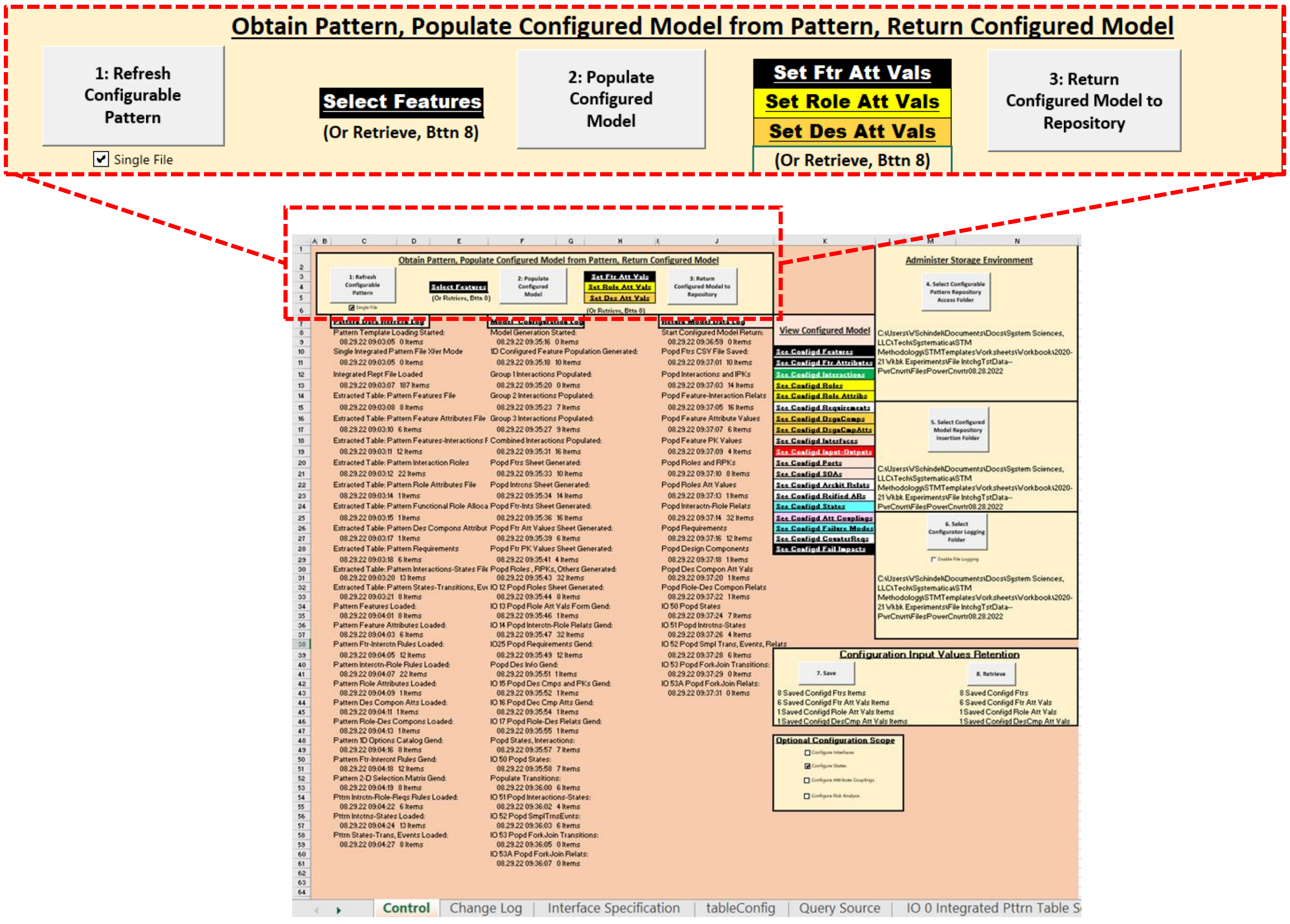
2.2.1 The Control Panel Buttons

The control buttons on the panel are as follows:

Buttons 1-3 are used during the basic cycle of generating models from patterns, as follows:

- Button 1: Load Pattern from Repository
    - This button causes the Configuration Wizard to fetch and load a Pattern that has been output by the Model Repository. A progress log is generated during that process.
- Button 2: Generate Configured Model from Pattern
    - This button causes the Configuration Wizard to begin to configure a Model from a Pattern. This will require preceding Button 2 with additional user input of configuration data, described further in Section 2.4. A progress log is generated during that process.

- Button 3: Return Configured Model to Repository
    - This button causes the configured model produced by the Configuration Wizard to be transmitted to the Model Repository.


Obtain Pattern, Populate Configured Model from Pattern, Return Configured Model
1: Refresh Configurable Pattern
Single File
Select Features
(Or Retrieve, Bttn 8)
2: Populate Configured Model
Set Ftr Att Vals
Set Role Att Vals
Set Des Att Vals
(Or Retrieve, Bttn 8)
3: Return Configured Model to Repository
View Configured Model
See Configd Features
See Configd Ftr Attributes
See Configd Interactions
See Configd Roles
See Configd Role Attribs
See Configd Requirements
See Configd DsgnComps
See Configd DsgnCmpAtts
See Configd Interfaces
See Configd Input-Outputs
See Configd Ports
See Configd SOAs
See Configd Archit Relats
See Configd Reified ABs
See Configd States
See Configd Att Couplings
See Configd Failure Modes
See Configd CounterReqs
See Configd Fail Impacts
Administer Storage Environment
4. Select Configurable Pattern Repository Access Folder
5. Select Configured Model Repository Insertion Folder
6. Select Configurator Logging Folder
Enable File Logging
Configuration Input Values Retention
7. Save
8. Retrieve
Optional Configuration Scope
Configure Interfaces
Configure States
Configure Attribute Couplings
Configure Risk Analysis
Control
Change Log
Interface Specification
tableConfig
Query Source
IO 0 Integrated Pttrn Table S

Pattern Data Refresh Log

```
Pattern Template Loading Started:
  08.29.22 09:03:05  0 Items
Single Integrated Pattern File Xfer Mode
  08.29.22 09:03:05  0 Items
Integrated Rept File Loaded
  08.29.22 09:03:07  187 Items
Extracted Table: Pattern Features File
  08.29.22 09:03:08  8 Items
Extracted Table: Pattern Feature Attributes File
  08.29.22 09:03:10  6 Items
Extracted Table: Pattern Features-Interactions F
  08.29.22 09:03:11  12 Items
Extracted Table: Pattern Interaction Roles
  08.29.22 09:03:12  22 Items
Extracted Table: Pattern Role Attributes File
  08.29.22 09:03:14  1 Items
Extracted Table: Pattern Functional Role Alloca
  08.29.22 09:03:15  1 Items
Extracted Table: Pattern Des Compons Attribut
  08.29.22 09:03:17  1 Items
Extracted Table: Pattern Requirements
  08.29.22 09:03:18  6 Items
Extracted Table: Pattern Interactions-States File
  08.29.22 09:03:20  13 Items
Extracted Table: Pattern States-Transitions, Ev
  08.29.22 09:03:21  8 Items
Pattern Features Loaded:
  08.29.22 09:04:01  8 Items
Pattern Feature Attributes Loaded:
  08.29.22 09:04:03  6 Items
Pattern Ftr-Interctn Rules Loaded:
  08.29.22 09:04:05  12 Items
Pattern Interctn-Role Rules Loaded:
  08.29.22 09:04:07  22 Items
Pattern Role Attributes Loaded:
  08.29.22 09:04:09  1 Items
Pattern Des Compon Atts Loaded:
  08.29.22 09:04:11  1 Items
Pattern Role-Des Compons Loaded:
  08.29.22 09:04:13  1 Items
Pattern 1D Options Catalog Gend:
  08.29.22 09:04:16  8 Items
Pattern Ftr-Intercnt Rules Gend:
  08.29.22 09:04:18  12 Items
Pattern 2-D Selection Matrix Gend:
  08.29.22 09:04:19  8 Items
Pttrn Intrctn-Role-Reqs Rules Loaded:
  08.29.22 09:04:22  6 Items
Pttrn Intctns-States Loaded:
  08.29.22 09:04:24  13 Items
Pttrn States-Trans, Events Loaded:
  08.29.22 09:04:27  8 Items
```

Model Configuration Log

```
Model Generation Started:
  08.29.22 09:35:16  0 Items
1D Configured Feature Population Generated:
  08.29.22 09:35:18  10 Items
Group 1 Interactions Populated:
  08.29.22 09:35:20  0 Items
Group 2 Interactions Populated:
  08.29.22 09:35:23  7 Items
Group 3 Interactions Populated:
  08.29.22 09:35:27  9 Items
Combined Interactions Populated:
  08.29.22 09:35:31  16 Items
Popd Ftrs Sheet Generated:
  08.29.22 09:35:33  10 Items
Popd Intrcns Sheet Generated:
  08.29.22 09:35:34  14 Items
Popd Ftr-Ints Sheet Generated:
  08.29.22 09:35:36  16 Items
Popd Ftr Att Values Sheet Generated:
  08.29.22 09:35:39  6 Items
Popd Ftr PK Values Sheet Generated:
  08.29.22 09:35:41  4 Items
Popd Roles , RPKs, Others Generated:
  08.29.22 09:35:43  32 Items
IO 12 Popd Roles Sheet Generated:
  08.29.22 09:35:44  8 Items
IO 13 Popd Role Att Vals Form Gend:
  08.29.22 09:35:46  1 Items
IO 14 Popd Interctn-Role Relats Gend:
  08.29.22 09:35:47  32 Items
IO25 Popd Requirements Gend:
  08.29.22 09:35:49  12 Items
Popd Des Info Gend:
  08.29.22 09:35:51  1 Items
IO 15 Popd Des Cmps and PKs Gend:
  08.29.22 09:35:52  1 Items
IO 16 Popd Dec Cmp Atts Gend:
  08.29.22 09:35:54  1 Items
IO 17 Popd Role-Des Relats Gend:
  08.29.22 09:35:55  1 Items
Popd States, Interactions:
  08.29.22 09:35:57  7 Items
IO 50 Popd States:
  08.29.22 09:35:58  7 Items
Populate Transitions:
  08.29.22 09:36:00  6 Items
IO 51 Popd Interactions-States:
  08.29.22 09:36:02  4 Items
IO 52 Popd SmplTrnsEvnts:
  08.29.22 09:36:03  6 Items
IO 53 Popd ForkJoin Transitions:
  08.29.22 09:36:05  0 Items
IO 53A Popd ForkJoin Relats:
  08.29.22 09:36:07  0 Items
```

Return Model Data Log

```
Start Configured Model Return:
  08.29.22 09:36:59  0 Items
Popd Ftrs CSV File Saved:
  08.29.22 09:37:01  10 Items
Popd Interactions and IPKs
  08.29.22 09:37:03  14 Items
Popd Feature-Interaction Relats
  08.29.22 09:37:05  16 Items
Popd Feature Attribute Values
  08.29.22 09:37:07  6 Items
Popd Feature PK Values
  08.29.22 09:37:09  4 Items
Popd Roles and RPKs
  08.29.22 09:37:10  8 Items
Popd Roles Att Values
  08.29.22 09:37:13  1 Items
Popd Interactn-Role Relats
  08.29.22 09:37:14  32 Items
Popd Requirements
  08.29.22 09:37:16  12 Items
Popd Design Components
  08.29.22 09:37:18  1 Items
Popd Des Compon Att Vals
  08.29.22 09:37:20  1 Items
Popd Role-Des Compon Relats
  08.29.22 09:37:22  1 Items
IO 50 Popd States
  08.29.22 09:37:24  7 Items
IO 51 Popd Intrctns-States
  08.29.22 09:37:26  4 Items
IO 52 Popd Smpl Trans, Events, Relats
  08.29.22 09:37:28  6 Items
IO 53 Popd ForkJoin Transitions:
  08.29.22 09:37:29  0 Items
IO 53A Popd ForkJoin Relats:
  08.29.22 09:37:31  0 Items
```

```
C:\Users\WSchindel\Documents\Docs\System Sciences,
LLC\Tech\Systematica\STM
Methodology\STMTemplatesWorksheets\Workbook\2020-
21 Wkbk Experiments\File IntchgTstData--
PwrCnvrt\FilesPowerCnvrtr08.28.2022

C:\Users\WSchindel\Documents\Docs\System Sciences,
LLC\Tech\Systematica\STM
Methodology\STMTemplatesWorksheets\Workbook\2020-
21 Wkbk Experiments\File IntchgTstData--
PwrCnvrt\FilesPowerCnvrtr08.28.2022

C:\Users\WSchindel\Documents\Docs\System Sciences,
LLC\Tech\Systematica\STM
Methodology\STMTemplatesWorksheets\Workbook\2020-
21 Wkbk Experiments\File IntchgTstData--
PwrCnvrt\FilesPowerCnvrtr08.28.2022
```

```
8 Saved Configd Ftrs Items                8 Saved Configd Ftrs
6 Saved Configd Ftr Att Vals Items        6 Saved Configd Ftr Att Vals
1 Saved Configd Role Att Vals Items       1 Saved Configd Role Att Vals
1 Saved Configd DesCmp Att Vals Items     1 Saved Configd DesCmp Att Vals
```

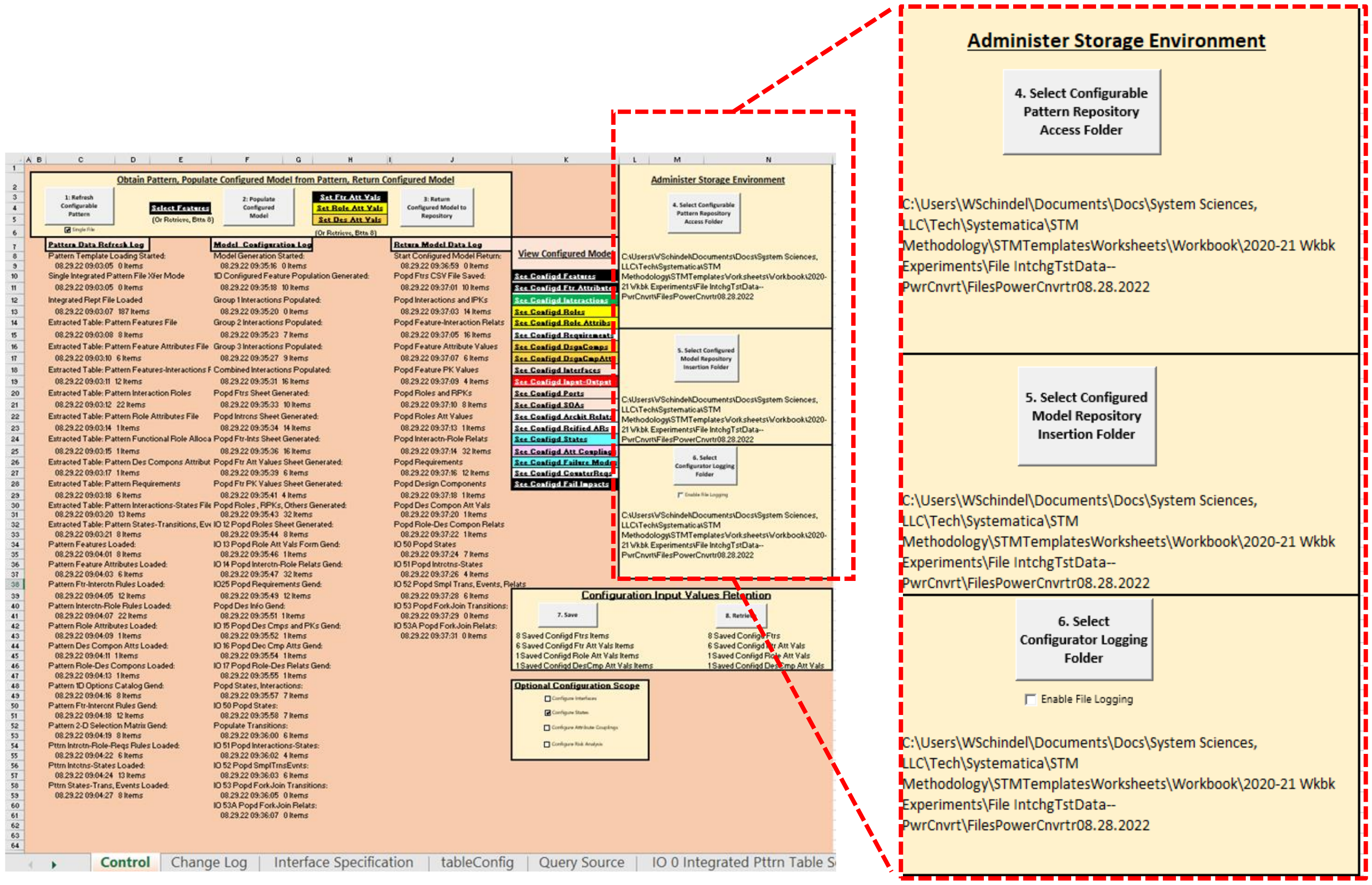
Configure Interfaces [ ] · Configure States [✓] · Configure Attribute Couplings [ ] · Configure Risk Analysis [ ]

Figure 2: The Control Panel of the Configuration Wizard, Including Control Buttons 1-3

Buttons 4-6 are used to identify the file directories in which various data will be found or stored.

- Button 4: Identify Source of Patterns
    - This button allows the user to select a file directory folder in which Patterns will be inserted by the Pattern Repository / Modeling Tool, as exported data from that repository.

- Button 5: Identify Destination for Configured Models
    - This button allows the user to select a file directory folder in which Configured Models will be inserted by the Configuration Wizard.
- Button 6: Identify Log File Destination
    - This button allows the user to select a file directory folder in which Log Files will be stored, if the file logging option is selected.



Obtain Pattern, Populate Configured Model from Pattern, Return Configured Model
1: Refresh Configurable Pattern
Select Features
2: Populate Configured Model
Set Ftr Att Vals
Set Role Att Vals
Set Des Att Vals
3: Return Configured Model to Repository
Pattern Data Refresh Log
Model Configuration Log
Return Model Data Log
View Configured Model
Configuration Input Values Retention
7. Save
8. Retrieve
Optional Configuration Scope
Control
Change Log
Interface Specification
tableConfig
Query Source
IO 0 Integrated Pttrn Table S
Administer Storage Environment
4. Select Configurable Pattern Repository Access Folder
5. Select Configured Model Repository Insertion Folder
6. Select Configurator Logging Folder
Enable File Logging

Administer Storage Environment

4. Select Configurable Pattern Repository Access Folder

C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022

5. Select Configured Model Repository Insertion Folder

C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022

6. Select Configurator Logging Folder

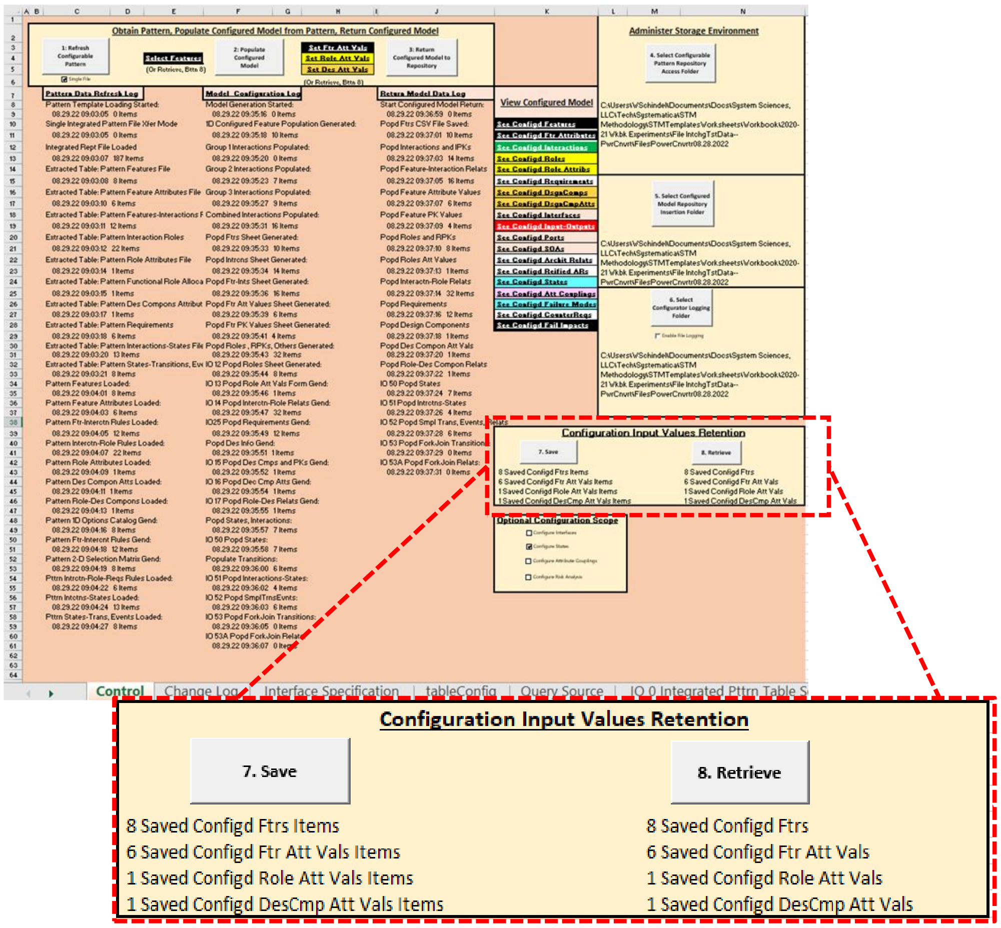
☐ Enable File Logging

C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022

Figure 3: Buttons 4-6, Selectable Locations for Files

Buttons 7-8 are used to save or retrieve user input information that is part of the configuration process.

- Button 7: Save Configuration Data
  - Because the pattern configuration process involves user insertion of significant amounts of configuration parameter values, this button allows that information to be saved (en masse in a single user-specified file) after it has been entered, so that it can later be retrieved and re-used without re-entering it. The saved configuration information consists of selected features and feature primary key values, populated feature attribute values other than primary key attributes, populated role attribute values, and populated design component attribute values. Multiple differing configuration sets may be saved in different files that can later be quickly invoked.
- Button 8: Retrieve Configuration Data
  - The opposite of Button 7, this button causes previously saved configuration parameter information to be retrieved for re-use, from a user specified separate file (see Button 7 above). The retrieved configuration information consists of selected features and feature primary key values, populated feature attribute values other than primary key attributes, populated role attribute values, and populated design component attribute values.


Obtain Pattern, Populate Configured Model from Pattern, Return Configured Model
1: Refresh Configurable Pattern
Single File
Select Features
(Or Retrieve, Bttn 8)
2: Populate Configured Model
Set Ftr Att Vals
Set Role Att Vals
Set Des Att Vals
(Or Retrieve, Bttn 8)
3: Return Configured Model to Repository
View Configured Model
See Configd Features
See Configd Ftr Attributes
See Configd Interactions
See Configd Roles
See Configd Role Attribs
See Configd Requirements
See Configd DsgnComps
See Configd DsgnCmpAtts
See Configd Interfaces
See Configd Input-Outputs
See Configd Ports
See Configd SOAs
See Configd Archit Relats
See Configd Reified ARs
See Configd States
See Configd Att Couplings
See Configd Failure Modes
See Configd CounterReqs
See Configd Fail Impacts
Administer Storage Environment
4. Select Configurable Pattern Repository Access Folder
5. Select Configured Model Repository Insertion Folder
6. Select Configurator Logging Folder
Enable File Logging
Optional Configuration Scope
Control
Change Log
Interface Specification
tableConfig
Query Source
IO 0 Integrated Pttrn Table S

```
Pattern Data Refresh Log
Pattern Template Loading Started:
   08.29.22 09:03:05  0 Items
Single Integrated Pattern File Xfer Mode
   08.29.22 09:03:05  0 Items
Integrated Rept File Loaded
   08.29.22 09:03:07  187 Items
Extracted Table: Pattern Features File
   08.29.22 09:03:08  8 Items
Extracted Table: Pattern Feature Attributes File
   08.29.22 09:03:10  6 Items
Extracted Table: Pattern Features-Interactions F
   08.29.22 09:03:11  12 Items
Extracted Table: Pattern Interaction Roles
   08.29.22 09:03:12  22 Items
Extracted Table: Pattern Role Attributes File
   08.29.22 09:03:14  1 Items
Extracted Table: Pattern Functional Role Alloca
   08.29.22 09:03:15  1 Items
Extracted Table: Pattern Des Compons Attribut
   08.29.22 09:03:17  1 Items
Extracted Table: Pattern Requirements
   08.29.22 09:03:18  6 Items
Extracted Table: Pattern Interactions-States File
   08.29.22 09:03:20  13 Items
Extracted Table: Pattern States-Transitions, Ev
   08.29.22 09:03:21  8 Items
Pattern Features Loaded:
   08.29.22 09:04:01  8 Items
Pattern Feature Attributes Loaded:
   08.29.22 09:04:03  6 Items
Pattern Ftr-Interctn Rules Loaded:
   08.29.22 09:04:05  12 Items
Pattern Interctn-Role Rules Loaded:
   08.29.22 09:04:07  22 Items
Pattern Role Attributes Loaded:
   08.29.22 09:04:09  1 Items
Pattern Des Compon Atts Loaded:
   08.29.22 09:04:11  1 Items
Pattern Role-Des Compons Loaded:
   08.29.22 09:04:13  1 Items
Pattern 1D Options Catalog Gend:
   08.29.22 09:04:16  8 Items
Pattern Ftr-Interent Rules Gend:
   08.29.22 09:04:18  12 Items
Pattern 2-D Selection Matrix Gend:
   08.29.22 09:04:19  8 Items
Pttrn Intrctn-Role-Reqs Rules Loaded:
   08.29.22 09:04:22  6 Items
Pttrn Intctns-States Loaded:
   08.29.22 09:04:24  13 Items
Pttrn States-Trans, Events Loaded:
   08.29.22 09:04:27  8 Items

Model Configuration Log
Model Generation Started:
   08.29.22 09:35:16  0 Items
1D Configured Feature Population Generated:
   08.29.22 09:35:18  10 Items
Group 1 Interactions Populated:
   08.29.22 09:35:20  0 Items
Group 2 Interactions Populated:
   08.29.22 09:35:23  7 Items
Group 3 Interactions Populated:
   08.29.22 09:35:27  9 Items
Combined Interactions Populated:
   08.29.22 09:35:31  16 Items
Popd Ftrs Sheet Generated:
   08.29.22 09:35:33  10 Items
Popd Intrcns Sheet Generated:
   08.29.22 09:35:34  14 Items
Popd Ftr-Ints Sheet Generated:
   08.29.22 09:35:36  16 Items
Popd Ftr Att Values Sheet Generated:
   08.29.22 09:35:39  6 Items
Popd Ftr PK Values Sheet Generated:
   08.29.22 09:35:41  4 Items
Popd Roles , RPKs, Others Generated:
   08.29.22 09:35:43  32 Items
IO 12 Popd Roles Sheet Generated:
   08.29.22 09:35:44  8 Items
IO 13 Popd Role Att Vals Form Gend:
   08.29.22 09:35:46  1 Items
IO 14 Popd Interctn-Role Relats Gend:
   08.29.22 09:35:47  32 Items
IO25 Popd Requirements Gend:
   08.29.22 09:35:49  12 Items
Popd Des Info Gend:
   08.29.22 09:35:51  1 Items
IO 15 Popd Des Cmps and PKs Gend:
   08.29.22 09:35:52  1 Items
IO 16 Popd Dec Cmp Atts Gend:
   08.29.22 09:35:54  1 Items
IO 17 Popd Role-Des Relats Gend:
   08.29.22 09:35:55  1 Items
Popd States, Interactions:
   08.29.22 09:35:57  7 Items
IO 50 Popd States:
   08.29.22 09:35:58  7 Items
Populate Transitions:
   08.29.22 09:36:00  6 Items
IO 51 Popd Interactions-States:
   08.29.22 09:36:02  4 Items
IO 52 Popd SmplTrnsEvnts:
   08.29.22 09:36:03  6 Items
IO 53 Popd ForkJoin Transitions:
   08.29.22 09:36:05  0 Items
IO 53A Popd ForkJoin Relats:
   08.29.22 09:36:07  0 Items

Return Model Data Log
Start Configured Model Return:
   08.29.22 09:36:59  0 Items
Popd Ftrs CSV File Saved:
   08.29.22 09:37:01  10 Items
Popd Interactions and IPKs
   08.29.22 09:37:03  14 Items
Popd Feature-Interaction Relats
   08.29.22 09:37:05  16 Items
Popd Feature Attribute Values
   08.29.22 09:37:07  6 Items
Popd Feature PK Values
   08.29.22 09:37:09  4 Items
Popd Roles and RPKs
   08.29.22 09:37:10  8 Items
Popd Roles Att Values
   08.29.22 09:37:13  1 Items
Popd Interactn-Role Relats
   08.29.22 09:37:14  32 Items
Popd Requirements
   08.29.22 09:37:16  12 Items
Popd Design Components
   08.29.22 09:37:18  1 Items
Popd Des Compon Att Vals
   08.29.22 09:37:20  1 Items
Popd Role-Des Compon Relats
   08.29.22 09:37:22  1 Items
IO 50 Popd States
   08.29.22 09:37:24  7 Items
IO 51 Popd Intrctns-States
   08.29.22 09:37:26  4 Items
IO 52 Popd Smpl Trans, Events, Relats
   08.29.22 09:37:28  6 Items
IO 53 Popd ForkJoin Transitions
   08.29.22 09:37:29  0 Items
IO 53A Popd ForkJoin Relats:
   08.29.22 09:37:31  0 Items

C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplates\Worksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022
C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplates\Worksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022
C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplates\Worksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022
```

Configuration Input Values Retention

7. Save	8. Retrieve
8 Saved Configd Ftrs Items	8 Saved Configd Ftrs
6 Saved Configd Ftr Att Vals Items	6 Saved Configd Ftr Att Vals
1 Saved Configd Role Att Vals Items	1 Saved Configd Role Att Vals
1 Saved Configd DesCmp Att Vals Items	1 Saved Configd DesCmp Att Vals

Figure 4: Buttons 7-8 Save and Retrieve User-Entered Configuration Information

2.2.2 The Control Panel Check Boxes

The Control Panel also provides several "check boxes", to control optional behaviors:

- **<u>The Integrated Import File checkbox</u>**: This option is about the Wizard importation of a configurable pattern from the modeling tool pattern repository. Depending on how that tool has been optioned and operated, this pattern import may involve multiple model exchange files or a single integrated model exchange file. The check box indicates which of these two options is to be used during the (Button 1) pattern import process. As a related aid, in the case of import of the single integrated file option, the Wizard will extract a set of multiple files from that integrated file and place them in the import area, similar to what would have been the case for the multiple file import case.
- **<u>The File Logging checkbox</u>**: This option is a diagnostic utility that will record logged files of what is being exchanged between the Wizard and the Pattern and Model Repository.

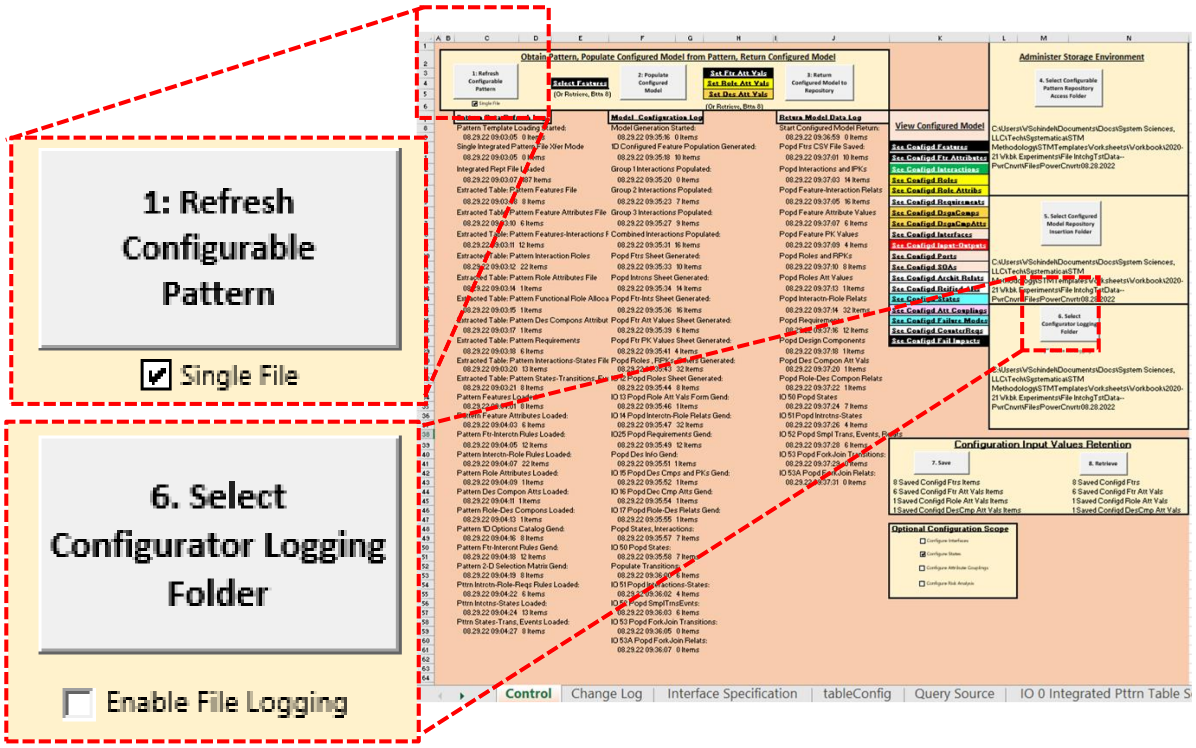
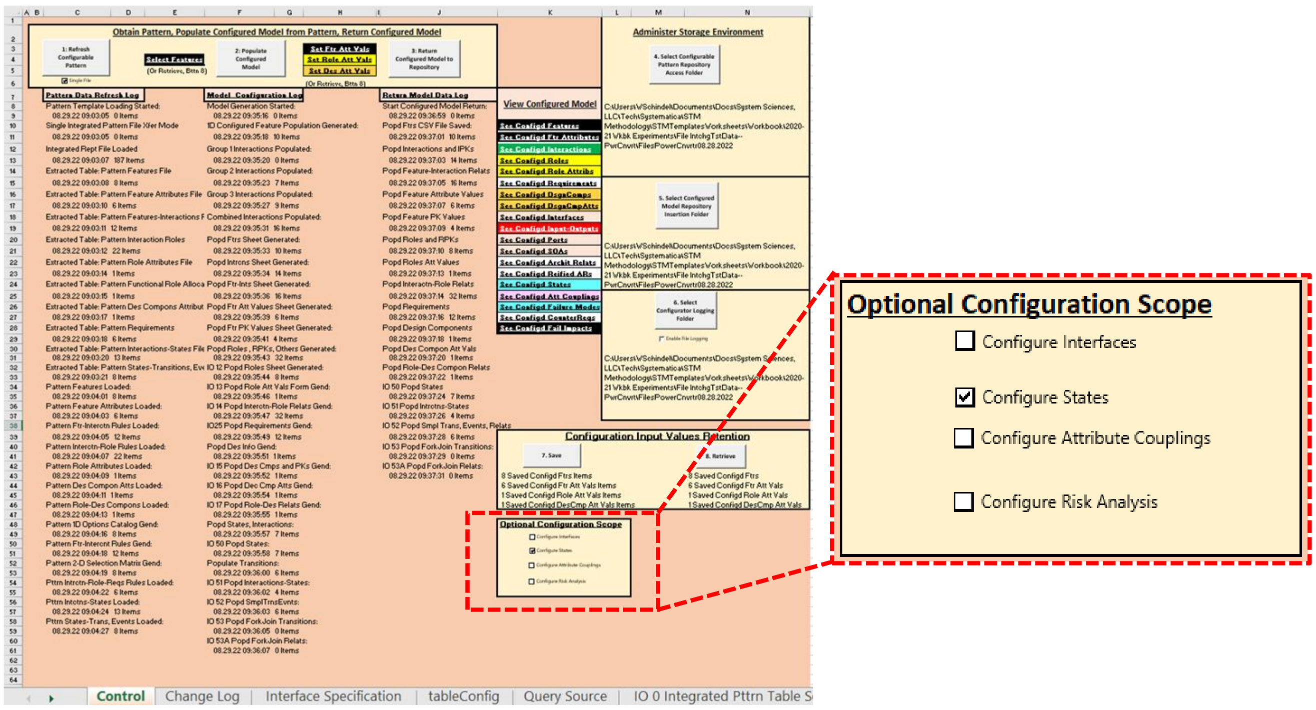

1: Refresh Configurable Pattern
Single File
6. Select Configurator Logging Folder
Enable File Logging
Obtain Pattern, Populate Configured Model from Pattern, Return Configured Model
Administer Storage Environment
View Configured Model
Configuration Input Values Retention
Optional Configuration Scope
Control
Change Log
Interface Specification
tableConfig
Query Source
IO 0 Integrated Pttrn Table S

Figure 5: Check Box Options for Pattern Import (Single Integrated File) and Logging

Four check boxes control the “scope of pattern configuration”. The Configuration Wizard can be used to populate and configure smaller or larger subsets of the full scope of the S*Metamodel, based on those additional checkboxes. The Configuration Wizard always populates and configures at least model Features, Interactions, Logical Systems, Requirements, and Design Components. Beyond that, the four optional checkboxes can be marked to populate and configure additional subsets of the S*Metamodel:

- **Interface Context Model Checkbox**: Additionally configures Interfaces, Input-Outputs, Ports, and Architectural Relationships. This expands the scope of the pattern import (Button 1), model configuration (Button 2), and configured model export (Button 3) processes.
- **State Model Checkbox**: Additionally configures States, Transitions, and Events. This expands the scope of the pattern import (Button 1), model configuration (Button 2), and configured model export (Button 3) processes.
- **Attribute Coupling Model Checkbox**: Additionally configures Attribute Couplings. This expands the scope of the pattern import (Button 1), model configuration (Button 2), and configured model export (Button 3) processes.
- **Risk (FMEA) Model Checkbox**: Additionally configures Failure Modes, Failure Impacts, and Counter-Requirements. This expands the scope of the pattern import (Button 1), model configuration (Button 2), and configured model export (Button 3) processes.


Obtain Pattern, Populate Configured Model from Pattern, Return Configured Model
1: Refresh Configurable Pattern
Select Features
2: Populate Configured Model
Set Ftr Att Vals
Set Role Att Vals
Set Des Att Vals
3: Return Configured Model to Repository
Administer Storage Environment
4. Select Configurable Pattern Repository Access Folder
5. Select Configured Model Repository Insertion Folder
6. Select Configurator Logging Folder
Pattern Data Refresh Log
Model Configuration Log
Return Model Data Log
View Configured Model
Configuration Input Values Retention
7. Save
8. Retrieve
Control
Change Log
Interface Specification
tableConfig
Query Source
IO 0 Integrated Pttrn Table S

Optional Configuration Scope

- ☐ Configure Interfaces
- ☑ Configure States
- ☐ Configure Attribute Couplings
- ☐ Configure Risk Analysis

Figure 6: Check Box Options for Expanding or Reducing Scope of Model Configuration

2.2.3 The Control Panel Navigation Links

The Control Panel tab also includes several navigation links, for travel to other tabs. See Figure 7. One group of navigation links is used in connection with the entry of pattern configuration parameter values:

- **<u>Select Features link</u>**: This navigates to the Stakeholder Feature and Feature Primary Key Value selection page (see Figure 12). It is that page into which the Wizard user will enter pattern configuration information *<u>before using Button 2,</u>* to select which Features of the pattern will be populated in the configured model (or not), and for Features with Primary Key attributes, what values of those Feature Primary Keys will be used to creates specialized instances of those Features in the configured model. This entry is normally carried out after a pattern has been loaded into the Wizard via Button 1, and before Button 2 is clicked to generate a configuration. Refer to Figure 11. It is the main user control over the configuration of the pattern. An alternate means of quickly selecting those Features and their Feature Primary Key values is the use of Button 8, retrieving en masse a previously saved set of such selections. See Section 2.5.
- **<u>Set Feature Attribute Values link</u>**: This navigates to the Feature Attribute Values data entry page. It is on that page the Wizard user will insert configuration information *<u>after using Button 2</u>*, to insert values into the newly populated (non-PK) Feature Attributes that were populated as part of the configuration process.
- **<u>Set Role Attribute Values link</u>**: This navigates to the Role Attribute Values data entry page. It is on that page the Wizard user will insert configuration information after using Button 2, to insert values into the newly populated (non-PK) Role Attributes that were populated as part of the configuration process.
- **<u>Set Design Component Attribute Values link</u>**: This navigates to the Design Component Attribute Values data entry page. It is on that page the Wizard user will insert configuration information after using Button 2, to insert values into the newly populated (non-PK) Design Component Attributes that were populated as part of the configuration process.

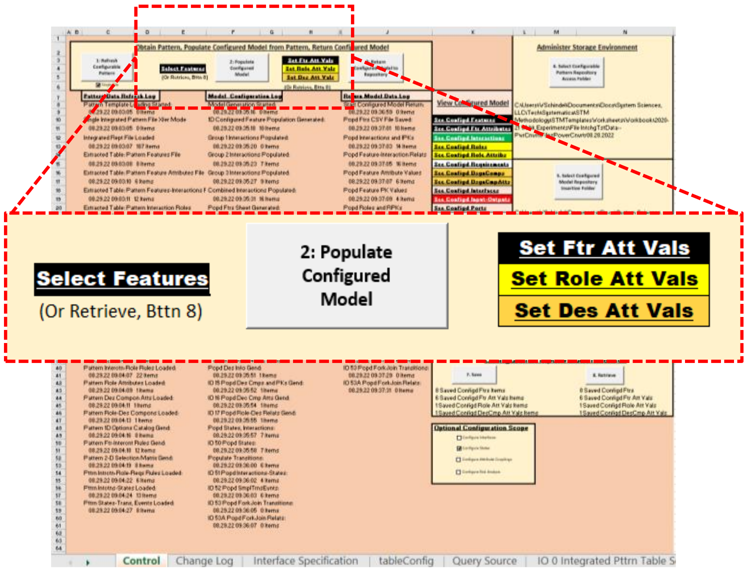
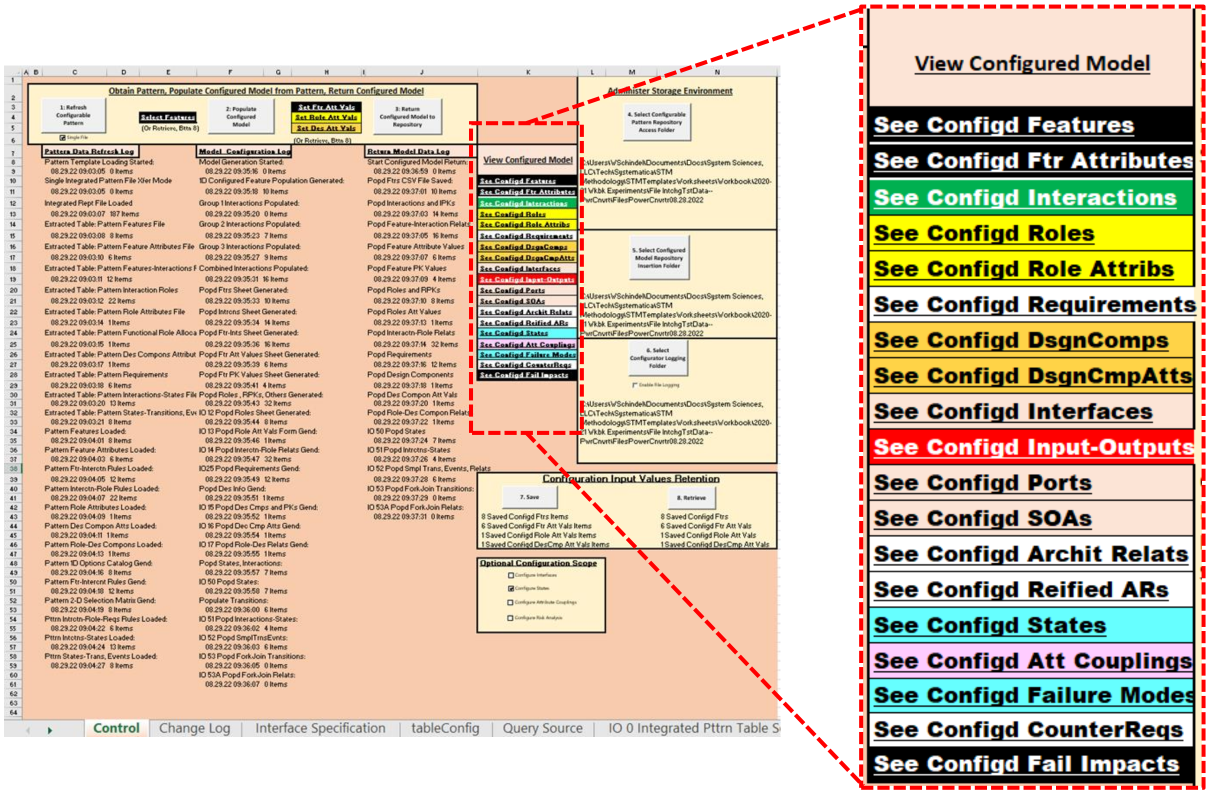


Figure 7: Navigation Links to Other Pages for Configuration Input Values

Another group of navigation links (see Figure 8) is used for the utility ability to navigate to and inspect the internal representation of portions of the configured model (after it is generated using Button 2) before it is transferred back to the model repository and rendered there in its modeling language specific form:

- Configured Model Features
- Configured Model Interactions
- Configured Model Roles
- Configured Model Requirement Statements
- Configured Model Design Components
- Configured Model Interface Context
- Configured Model States
- Configured Model Attribute Couplings
- Configured Model Risk Analysis / FMEA


Obtain Pattern, Populate Configured Model from Pattern, Return Configured Model

1: Refresh Configurable Pattern | Select Features (Or Retrieve, Bttn 8) | 2: Populate Configured Model | Set Ftr Att Vals | Set Role Att Vals | Set Des Att Vals (Or Retrieve, Bttn 8) | 3: Return Configured Model to Repository

Pattern Data Refresh Log | Model Configuration Log | Return Model Data Log

View Configured Model

See Configd Features
See Configd Ftr Attributes
See Configd Interactions
See Configd Roles
See Configd Role Attribs
See Configd Requirements
See Configd DsgnComps
See Configd DsgnCmpAtts
See Configd Interfaces
See Configd Input-Outputs
See Configd Ports
See Configd SOAs
See Configd Archit Relats
See Configd Reified ARs
See Configd States
See Configd Att Couplings
See Configd Failure Modes
See Configd CounterReqs
See Configd Fail Impacts

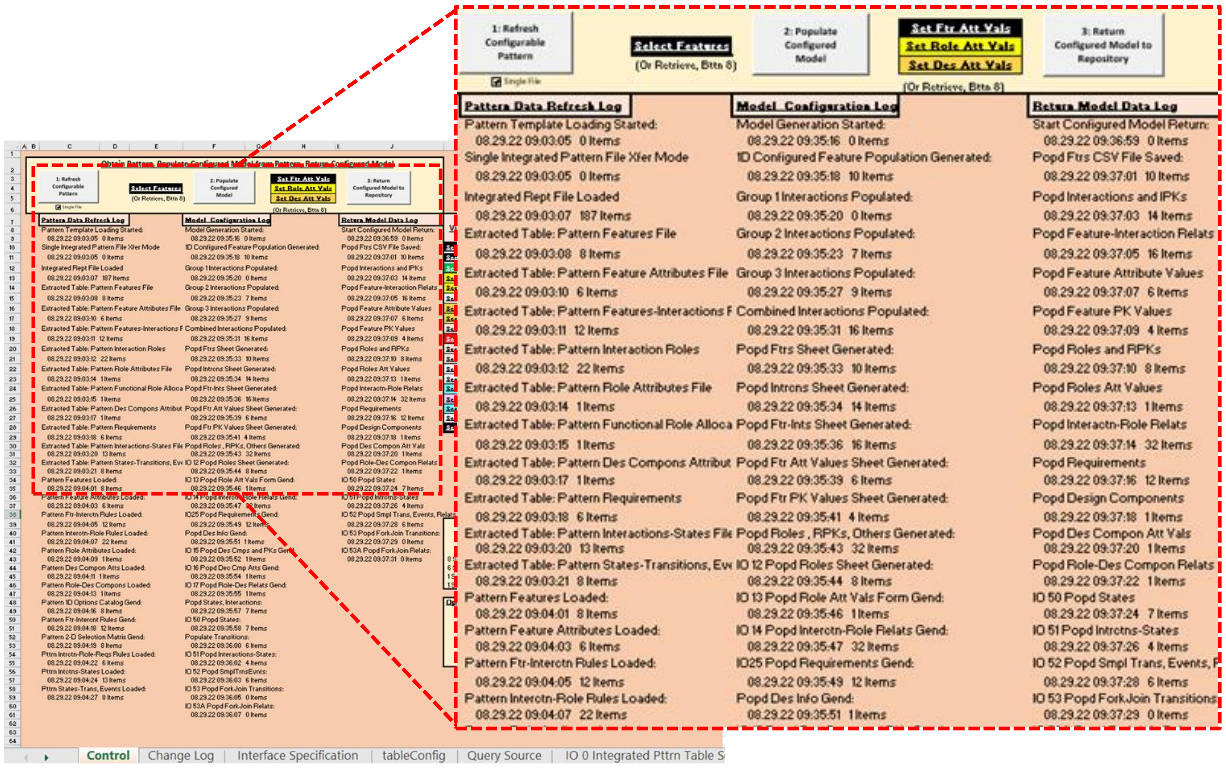
Administer Storage Environment
Configuration Input Values Retention
Optional Configuration Scope
Control
Change Log
Interface Specification
tableConfig
Query Source
IO 0 Integrated Pttrn Table

Figure 8: Navigation Links to View Configured Model Elements

2.2.4 The Control Panel Progress and Error Logs

The progress and error logs on the Control Panel (see Figure 9) are as follows:

- **<u>Pattern Data Refresh Log</u>**: Located below Button 1, this log shows progress through the process started by that button—loading (importing) the configurable pattern data provided by the pattern repository. Each entry shows a time stamp and indication of the number of items of various types that were received. Errors may be reported at this level of granularity.
- **<u>Model Configuration Log</u>**: Located below Button 2, this log shows progress through the population of a configured model. Each entry shows a time stamp and indication of the number of items of various types that were populated. Errors maybe reported at this level of granularity.
- **<u>Return Model Data Log</u>**: Located below button 3, this log shows progress through the return (export) of configured model data intended for the model repository.



Pattern Data Refresh Log

```
Pattern Template Loading Started:
  08.29.22 09:03:05  0 Items
Single Integrated Pattern File Xfer Mode
  08.29.22 09:03:05  0 Items
Integrated Rept File Loaded
  08.29.22 09:03:07  187 Items
Extracted Table: Pattern Features File
  08.29.22 09:03:08  8 Items
Extracted Table: Pattern Feature Attributes File
  08.29.22 09:03:10  6 Items
Extracted Table: Pattern Features-Interactions F
  08.29.22 09:03:11  12 Items
Extracted Table: Pattern Interaction Roles
  08.29.22 09:03:12  22 Items
Extracted Table: Pattern Role Attributes File
  08.29.22 09:03:14  1 Items
Extracted Table: Pattern Functional Role Alloca
  08.29.22 09:03:15  1 Items
Extracted Table: Pattern Des Compons Attribut
  08.29.22 09:03:17  1 Items
Extracted Table: Pattern Requirements
  08.29.22 09:03:18  6 Items
Extracted Table: Pattern Interactions-States File
  08.29.22 09:03:20  13 Items
Extracted Table: Pattern States-Transitions, Ev
  08.29.22 09:03:21  8 Items
Pattern Features Loaded:
  08.29.22 09:04:01  8 Items
Pattern Feature Attributes Loaded:
  08.29.22 09:04:03  6 Items
Pattern Ftr-Interctn Rules Loaded:
  08.29.22 09:04:05  12 Items
Pattern Interctn-Role Rules Loaded:
  08.29.22 09:04:07  22 Items
```

Model Configuration Log

```
Model Generation Started:
  08.29.22 09:35:16  0 Items
1D Configured Feature Population Generated:
  08.29.22 09:35:18  10 Items
Group 1 Interactions Populated:
  08.29.22 09:35:20  0 Items
Group 2 Interactions Populated:
  08.29.22 09:35:23  7 Items
Group 3 Interactions Populated:
  08.29.22 09:35:27  9 Items
Combined Interactions Populated:
  08.29.22 09:35:31  16 Items
Popd Ftrs Sheet Generated:
  08.29.22 09:35:33  10 Items
Popd Intrcns Sheet Generated:
  08.29.22 09:35:34  14 Items
Popd Ftr-Ints Sheet Generated:
  08.29.22 09:35:36  16 Items
Popd Ftr Att Values Sheet Generated:
  08.29.22 09:35:39  6 Items
Popd Ftr PK Values Sheet Generated:
  08.29.22 09:35:41  4 Items
Popd Roles , RPKs, Others Generated:
  08.29.22 09:35:43  32 Items
IO 12 Popd Roles Sheet Generated:
  08.29.22 09:35:44  8 Items
IO 13 Popd Role Att Vals Form Gend:
  08.29.22 09:35:46  1 Items
IO 14 Popd Interctn-Role Relats Gend:
  08.29.22 09:35:47  32 Items
IO25 Popd Requirements Gend:
  08.29.22 09:35:49  12 Items
Popd Des Info Gend:
  08.29.22 09:35:51  1 Items
```

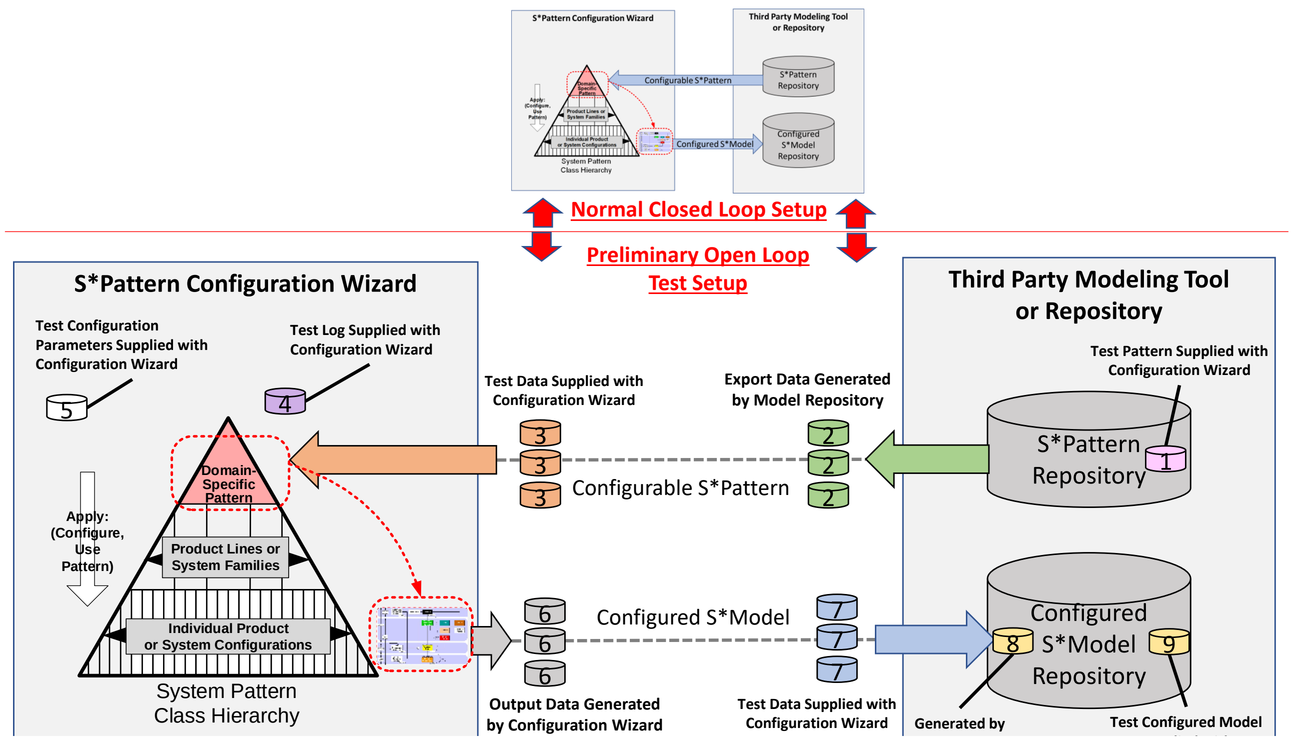
Return Model Data Log

```
Start Configured Model Return:
  08.29.22 09:36:59  0 Items
Popd Ftrs CSV File Saved:
  08.29.22 09:37:01  10 Items
Popd Interactions and IPKs
  08.29.22 09:37:03  14 Items
Popd Feature-Interaction Relats
  08.29.22 09:37:05  16 Items
Popd Feature Attribute Values
  08.29.22 09:37:07  6 Items
Popd Feature PK Values
  08.29.22 09:37:09  4 Items
Popd Roles and RPKs
  08.29.22 09:37:10  8 Items
Popd Roles Att Values
  08.29.22 09:37:13  1 Items
Popd Interactn-Role Relats
  08.29.22 09:37:14  32 Items
Popd Requirements
  08.29.22 09:37:16  12 Items
Popd Design Components
  08.29.22 09:37:18  1 Items
Popd Des Compon Att Vals
  08.29.22 09:37:20  1 Items
Popd Role-Des Compon Relats
  08.29.22 09:37:22  1 Items
IO 50 Popd States
  08.29.22 09:37:24  7 Items
IO 51 Popd Intrctns-States
  08.29.22 09:37:26  4 Items
IO 52 Popd Smpl Trans, Events, F
  08.29.22 09:37:28  6 Items
IO 53 Popd ForkJoin Transitions
  08.29.22 09:37:29  0 Items
```

Figure 9: Progress and Error Logs

2.3 Performing a Preliminary Test

The intended use of the Configuration Wizard involves end-to-end interaction with a Model Authoring Tool or Model Repository. Before attempting that integrated activity for the first time, a simpler preliminary pre-integration test is recommended, as summarized by Figure 10:



S*Pattern Configuration Wizard
Third Party Modeling Tool or Repository
Configurable S*Pattern
Configured S*Model
S*Pattern Repository
Configured S*Model Repository
Normal Closed Loop Setup
Preliminary Open Loop Test Setup
S*Pattern Configuration Wizard
Test Configuration Parameters Supplied with Configuration Wizard
Test Log Supplied with Configuration Wizard
5
4
Domain-Specific Pattern
Apply: (Configure, Use Pattern)
Product Lines or System Families
Individual Product or System Configurations
System Pattern Class Hierarchy
Test Data Supplied with Configuration Wizard
Export Data Generated by Model Repository
3
2
Configurable S*Pattern
Third Party Modeling Tool or Repository
Test Pattern Supplied with Configuration Wizard
S*Pattern Repository
1
6
7
Configured S*Model
Output Data Generated by Configuration Wizard
Test Data Supplied with Configuration Wizard
Configured S*Model Repository
8
9
Generated by
Test Configured Model

Figure 10: A Preliminary "Open Loop" Test Before Integration

This test uses the example test data provided with the Configuration Wizard (Refer to Appendix D), by "breaking" the closed loop normal configuration into an "open loop" test configuration—see Figure 10. The preliminary test simplifies the process of integration of the Configuration Wizard, Third Party Modeling Tool or Repository, and their IT environment. The recommended test steps correspond to the diagram in Figure 10:

1. Install the Test Pattern (1), supplied with the Configuration Wizard, in the Pattern Repository.
2. Operate the Pattern Repository export process to generate Export Data (2), which should conform to the Interface Specification for pattern files, in Appendix A. A simple check on that conformance is comparison to (3), the test data supplied with the Configuration Wizard.
3. Install the Configuration Wizard and use Button 1 to import the pattern Test Data (3) provided with the Configuration Wizard.
4. Check the Import Log (under Button 1) for agreement with (4), the Test Log supplied with the Configuration Wizard. The imported pattern items counts should agree with that reference.
5. Install the test Configuration Input Parameters file (5) in the Configuration Wizard accessible file environment. Use the Retrieve (Button 8) operation to load that configuration data file (5).

6. Use Button 2 to generate a Configured Model (6), and compare the log generated under Button 2 to the test data log (4) supplied with the Configuration Wizard. The logged items counts should be the same. Compare the Configured Model (6) files generated to the Test Data (7) files provided with the Configuration Wizard. They should be the same.
7. Import the test data configured model files (7), provided with the Configuration Wizard, into the Model Repository, using the repository import process, which should conform to the Interface Specification of Appendix A.
8. A simple check on the conformance of (8) is to compare it to the test data configured model (9) supplied with the Configuration Wizard.

When all the above "open loop" tests pass, then the user should be able to revert to the "normal set up" integration seen in Figure 1 and Figure 10. The same results should be obtainable in closed loop form without using the extra test files supplied with the Configuration Wizard.

2.4 How to Configure a Pattern Using the Wizard

Figure 11 summarizes the overall flow to generate a configured model from a configurable pattern:

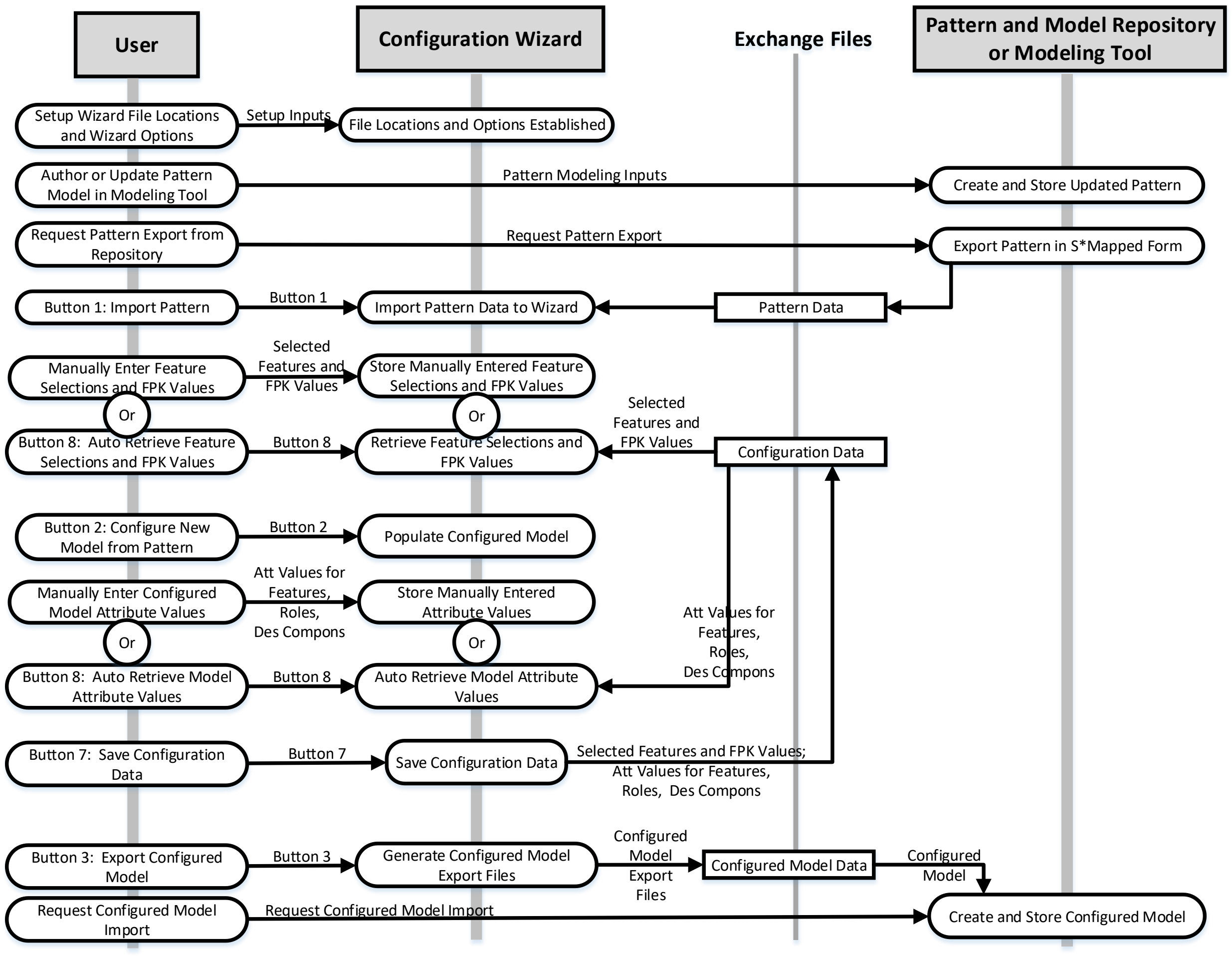
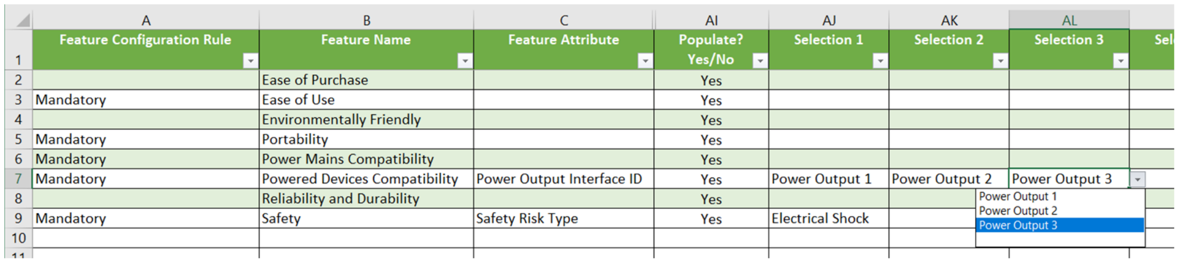

User
Configuration Wizard
Exchange Files
Pattern and Model Repository or Modeling Tool
Setup Wizard File Locations and Wizard Options
Setup Inputs
File Locations and Options Established
Author or Update Pattern Model in Modeling Tool
Pattern Modeling Inputs
Create and Store Updated Pattern
Request Pattern Export from Repository
Request Pattern Export
Export Pattern in S*Mapped Form
Button 1: Import Pattern
Button 1
Import Pattern Data to Wizard
Pattern Data
Manually Enter Feature Selections and FPK Values
Selected Features and FPK Values
Store Manually Entered Feature Selections and FPK Values
Or
Or
Selected Features and FPK Values
Button 8: Auto Retrieve Feature Selections and FPK Values
Button 8
Retrieve Feature Selections and FPK Values
Configuration Data
Button 2: Configure New Model from Pattern
Button 2
Populate Configured Model
Manually Enter Configured Model Attribute Values
Att Values for Features, Roles, Des Compons
Store Manually Entered Attribute Values
Or
Or
Att Values for Features, Roles, Des Compons
Button 8: Auto Retrieve Model Attribute Values
Button 8
Auto Retrieve Model Attribute Values
Button 7: Save Configuration Data
Button 7
Save Configuration Data
Selected Features and FPK Values; Att Values for Features, Roles, Des Compons
Button 3: Export Configured Model
Button 3
Generate Configured Model Export Files
Configured Model Export Files
Configured Model Data
Configured Model
Request Configured Model Import
Request Configured Model Import
Create and Store Configured Model

Figure 11: Overview of the Pattern Configuration Process

As indicated by Figure 11, after clicking on Button 1 to load a pattern into the Configuration Wizard, and before clicking on Button 2, the user should enter a selection of Features from the Configurable Pattern that are to be populated in the Configured Model. This is done through the Select Features link shown in Figure 7, bringing up the Features Selection Matrix shown in Figure 12.



	A	B	C	AI	AJ	AK	AL	
1	Feature Configuration Rule	Feature Name	Feature Attribute	Populate? Yes/No	Selection 1	Selection 2	Selection 3	Sel
2		Ease of Purchase		Yes				
3	Mandatory	Ease of Use		Yes				
4		Environmentally Friendly		Yes				
5	Mandatory	Portability		Yes				
6	Mandatory	Power Mains Compatibility		Yes				
7	Mandatory	Powered Devices Compatibility	Power Output Interface ID	Yes	Power Output 1	Power Output 2	Power Output 3	
8		Reliability and Durability		Yes				
9	Mandatory	Safety	Safety Risk Type	Yes	Electrical Shock			
10								

Figure 12: Features Selection Matrix

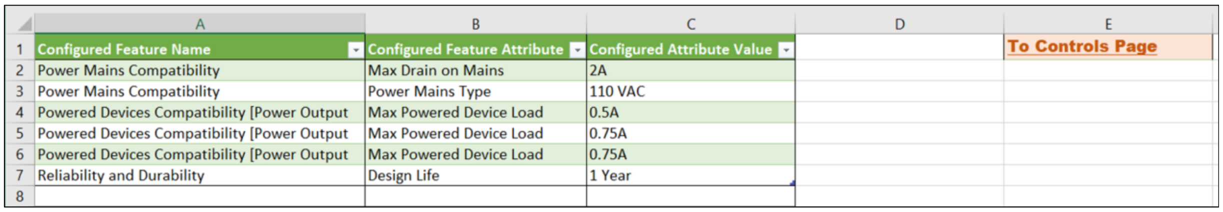
The Features Selection Matrix displays all the Features of the loaded Pattern. For each such Feature, the user may enter Yes or No to indicate whether it is to be populated (included) in the Configured Model to be generated by the Wizard. In addition, the matrix shows which pattern Features, if any, include Feature Primary Key attributes, naming them in the matrix in the Feature Attribute column. The subset of Features that have Primary Key attributes may be “multiply instantiated” in the Configured Model, with each such instance differentiated by a unique value to be populated in its Feature Primary Key Attribute. The pattern may contain an enumerated list of possible Feature Primary Key values to populate in such instance, available for selection using pull-down menus as shown in Figure 12.

After entering Feature selection and Feature Primary Key value information into the Feature Selection Matrix shown in Figure 12, the user may click Button 2 to generate a configured model based on those selections. A configuration progress and errors log will be generated below Button 2, as shown in Figure 9.

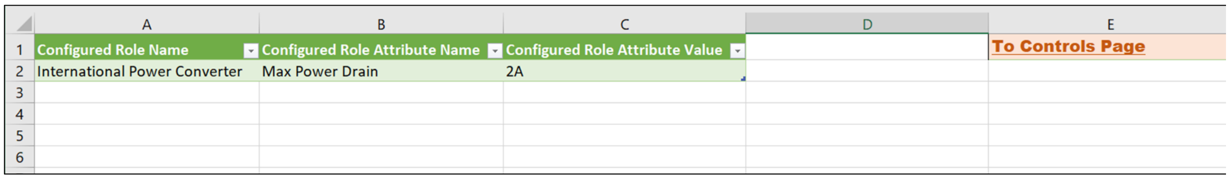
After that population has occurred, the resulting configured model data structure is ready to allow the user to also enter values for populated Feature Attributes (not the PK type, just ordinary attributes that can have values), as well as attributes of populated Roles, and attributes of populated Design Components. These are entered by navigating to the three respective screens using the navigation links shown in Figure 7. Each of those three screens displays a list of populated classes and their attributes (Feature Attributes, Role Attributes, Design Component Attributes), and invites entry of values for those attributes. Refer to Figure 13.

Each of those three screens provides a return navigation link back to the Control Panel screen.

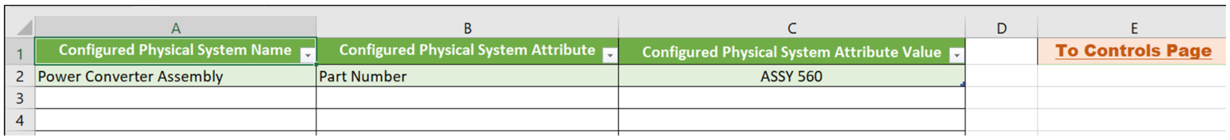
After entry of these attribute values, the user has the option of saving those entries to an external record file, in case they may be needed again in a future situation with this or another configuration. This is described in the next section.



Configured Feature Name	Configured Feature Attribute	Configured Attribute Value		To Controls Page
Power Mains Compatibility	Max Drain on Mains	2A		
Power Mains Compatibility	Power Mains Type	110 VAC		
Powered Devices Compatibility [Power Output	Max Powered Device Load	0.5A		
Powered Devices Compatibility [Power Output	Max Powered Device Load	0.75A		
Powered Devices Compatibility [Power Output	Max Powered Device Load	0.75A		
Reliability and Durability	Design Life	1 Year		



Configured Role Name	Configured Role Attribute Name	Configured Role Attribute Value		To Controls Page
International Power Converter	Max Power Drain	2A		



Configured Physical System Name	Configured Physical System Attribute	Configured Physical System Attribute Value		To Controls Page
Power Converter Assembly	Part Number	ASSY 560		

Figure 13: Configured Model Attribute Value Entry Screens, for Features, Roles, and Design Components

2.5 Saving and Retrieving Configuration Parameter Input Data

Buttons 7 and 8 can be used to save and retrieve manually-entered configuration attribute data, so that it does not need to be manually re-entered. This information is saved in a user-designated file whose name and location are identified by the user. So, many such configuration parameter value files may be saved, for different configurations. The files used to save this information are of file type .xlsx.

It is not necessary for the Wizard user to view the interior content of these files, which are created by and used by the Configuration Wizard itself. Such a saved configuration values file has four internal tabs, containing the following information semantic structures and values information:

- Configured Features Selected, and their Feature Primary Key Values, for the currently loaded pattern's library of potential features;
- Configured Feature Attribute Values for the current configured model's populated Features;
- Configured Role Attribute Values for the current configured model's populated Roles;
- Configured Design Component Attribute values for the current configured model's populated Design Components;

Establishing values for these items may usefully occur at two different times. Refer to Figure 11:

1. **<u>After loading of a Pattern, via Button 1, and prior to configuring that pattern, using Button 2</u>**: At this time, the selection of Features and their Feature Primary Key values should occur. (This does not involve the other Feature Attributes, Role Attribute, or Design Component Attributes.) This may be done by either manually entering that information on the Feature Selection tab, or else by automated setting of those values using the Retrieve (Button 8) capability. As long as Button 1 has been used once to load a pattern of interest, Button 8 may be used multiple times to generate

different configurations of that pattern, each using Button 2 to generate a different configured model, without repeating Button 1.

2. **<u>After populating a configured model, using Button 2</u>**: At this time, configured model class instances have been populated by the Configuration Wizard, including the population of attribute "slots" for instances of Features, Roles, and Design Components. So, it is at this point that values for those attributes may be inserted. This may be done by either manually entering that information on the three tabs for those attributes, or else by automatically setting of those values using the Retrieve (Button 8) capability. As long as Button 2 has been used once to generate the populated classes carrying these attributes, Button 8 may be used multiple times to generate different configurations of that pattern.

As a further aid to Wizard ease of use and robustness, during the "Retrieve" (Button 8) operation, the Configuration Wizard performs a "match-up" process between:

A. the <u>semantic structure</u> of the currently loaded <u>pattern</u> (obtained via Button 1) or currently configured <u>model</u> (obtained via Button 2)

in comparison to (matched up with)

B. the <u>semantic structure</u> of the saved file of previously saved (via Button 7) configuration data

This "match-up" process is performed because the currently loaded pattern or configured model may be different (a little or a lot) than the pattern or configured model that was in use when the configuration information was saved (via Button 7). A simple example of this would be the addition of a new Feature, Feature Attribute, Role, or Role Attribute to the pattern, months after an earlier saving of a configuration values file. The "match-up" process aligns the subset of saved values information with the current pattern or configured model information, saving a lot of manual inspection or data entry effort. The Wizard user is still free to also further edit the configured values data after the Retrieve (Button 8) operation, to make up any subsequent changes, and the result may also be saved (Button 7) as a new saved configuration file.

The above match-up process occurs automatically, transparent to the Wizard User, but visibility of the results is posted in the log entries below Button 7 and Button 8, where the number of retrieved or saved items is visible. See Figure 4.

In order to avoid accidental mis-matches of configuration information and the pattern or configured model to which that information applies, performing a pattern load (Button 1) will automatically clear out (erase) any currently entered configuration information for selected Features and Feature Primary Keys, as well as any currently entered values for Feature Attributes, Role Attributes, and Design Component Attributes. These may be quickly restored by a Retrieve (Button 8) operation.

As a consequence of the above automatic match-ups, there can be situations in which the Retrieve (Button 8) operation may be required at two different times in the following sequence:

1. After loading a pattern (Button 1), the use of Button 8 will automatically load previously saved Feature and Feature Primary Key selection data that will have impact on a subsequent pattern configuration operation.

2. After subsequently configuring the pattern (using Button 2), a configured model will have been created with a semantic structure determined by a combination of the pattern and the above configuration information. At that point, a second Retrieve (Button 8) operation can be used to re-establish attribute values for the configured model's populated Features, Roles, and Design Components. This is particularly important before exporting the resulting configured model back to the model repository (Button 3).

Based on the above, user warning messages appear whenever Button 1 or 2 are pressed:

- The warning message upon pressing Button 1 is a reminder that continuing that operation will wipe out any currently entered configuration values. This provides the user opportunity to stop short of this process, to save that information first if desired (via Button 7).
- The warning message upon pressing Button 2 is a reminder that continuing that operation assumes that the user has already entered feature configuration information, either manually or via Retrieve (Button 8). (For example, a pushing of Button 1 to load a pattern could have wiped out that information, which needs to be refreshed by Button 8.)

2.6 The Configuration Process Algorithm; Configuration Predecessors and Successors

The pattern configuration process, which is automated by the Configuration Wizard, generates a configured model from a configurable pattern combined with configuration input information. An informal illustrative overview of this is provided by

Figure 14 for a subset of the S*Metamodel classes and relationships populated in the Configured Model by this process.

This is a propagation process, in which "upstream" configured model elements are combined with the pattern (which includes configuration rules) to generate "downstream" configured model elements. The order of that propagation is a consequence of it being based on the generic structure of the neutral S*Metamodel (mapped to the tools and languages in use). This simplifies the nature of configuration analysis and rules, so that the order of precedence and succession becomes central to understanding pattern configuration.

Based on that, Figure 15 formalizes a more complete (compared to

Figure 14) description of the predecessor-successor order of configured model population. In Figure 15, population of model classes that are listed across the top (defining columns) will trigger population of model classes that are listed down the left side (defining rows), but only when a grey cell appears at the intersection of the related column and row.

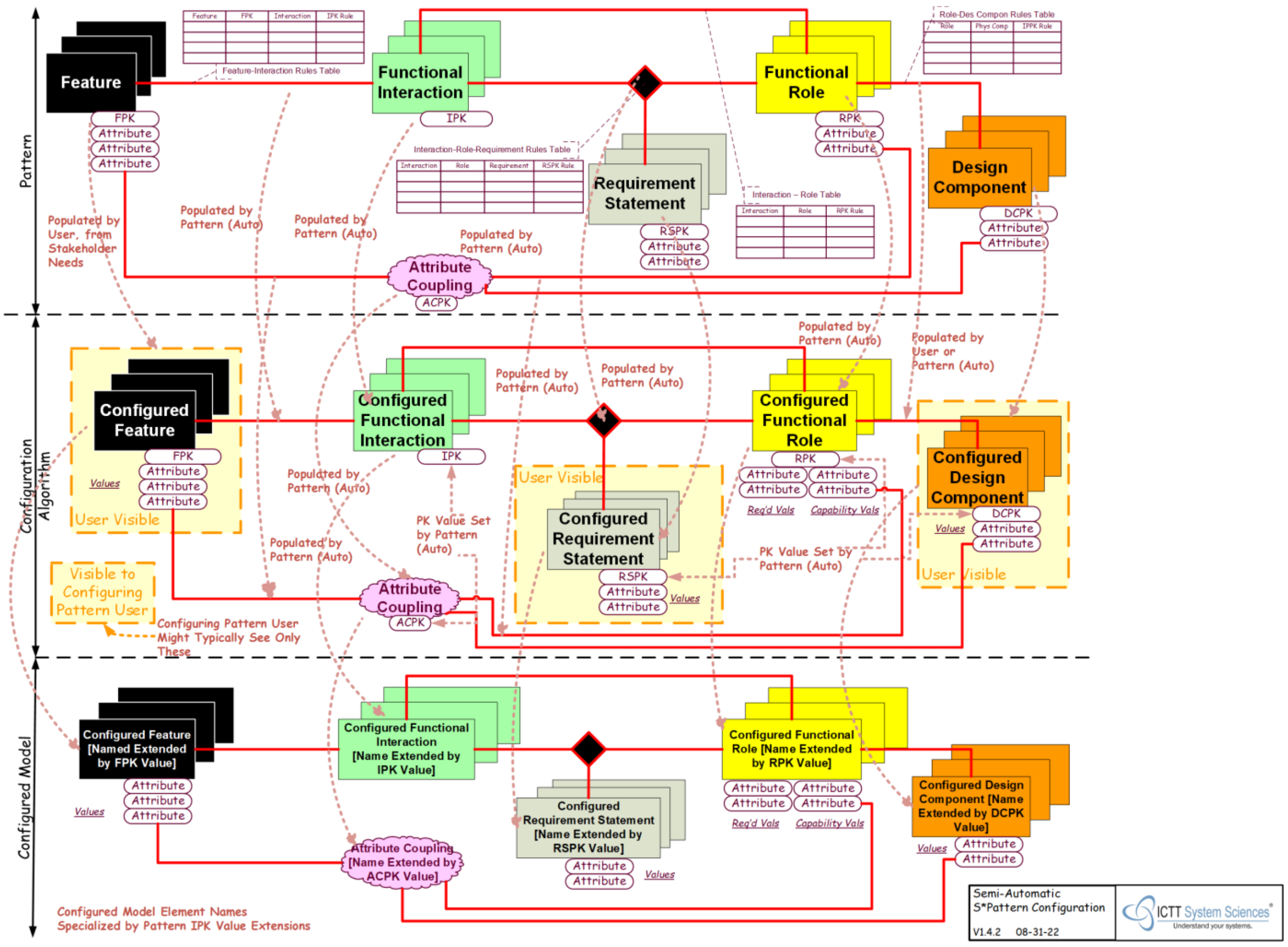
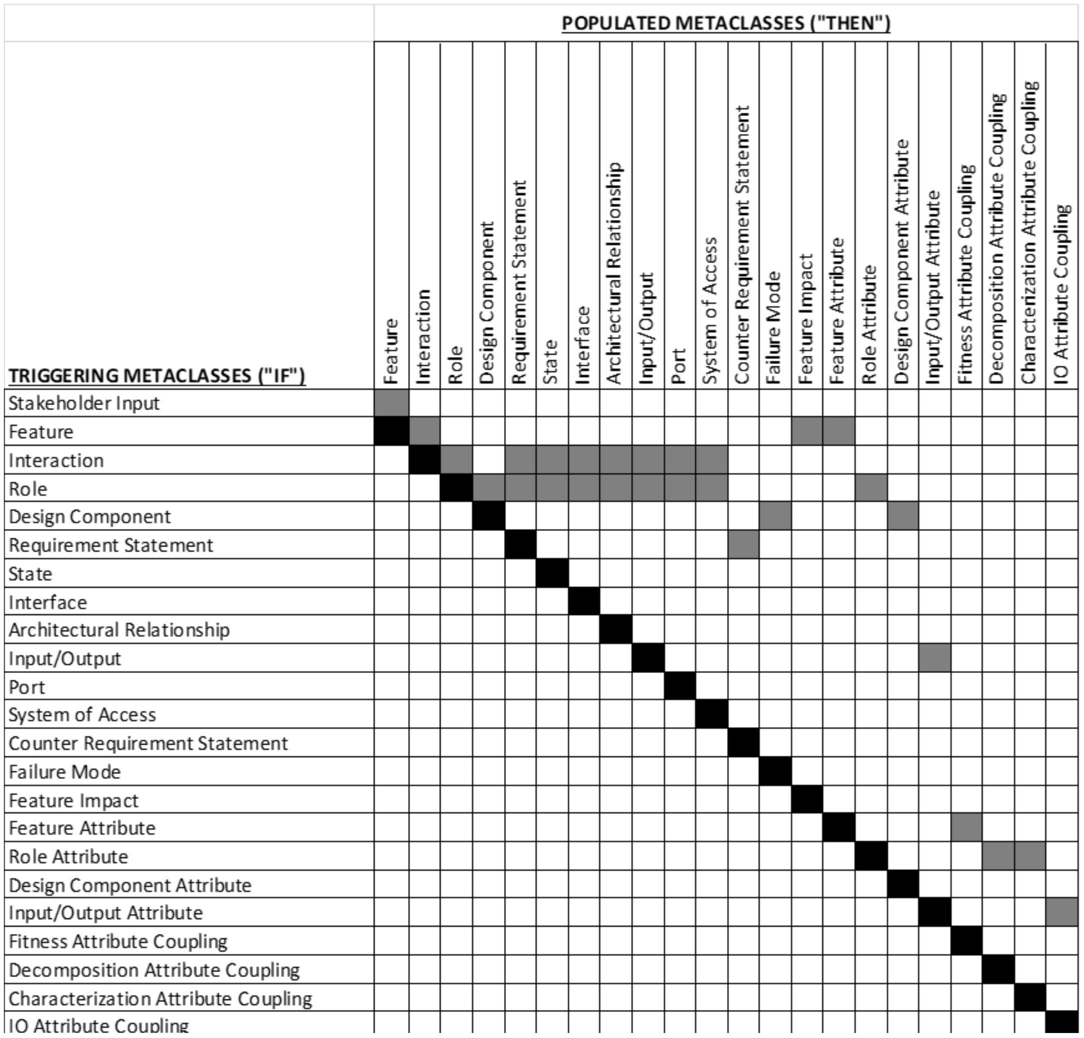

Pattern
Configuration Algorithm
Configured Model
Feature
FPK
Attribute
Feature-Interaction Rules Table
Functional Interaction
IPK
Functional Role
RPK
Role-Des Compon Rules Table
Design Component
DCPK
Requirement Statement
RSPK
Interaction-Role-Requirement Rules Table
Interaction – Role Table
Attribute Coupling
ACPK
Populated by User, from Stakeholder Needs
Populated by Pattern (Auto)
Populated by User or Pattern (Auto)
Configured Feature
Configured Functional Interaction
Configured Functional Role
Configured Design Component
Configured Requirement Statement
User Visible
Values
Req'd Vals
Capability Vals
PK Value Set by Pattern (Auto)
Visible to Configuring Pattern User
Configuring Pattern User Might Typically See Only These
Configured Feature [Named Extended by FPK Value]
Configured Functional Interaction [Name Extended by IPK Value]
Configured Functional Role [Name Extended by RPK Value]
Configured Design Component [Name Extended by DCPK Value]
Configured Requirement Statement [Name Extended by RSPK Value]
Attribute Coupling [Name Extended by ACPK Value]
Configured Model Element Names Specialized by Pattern IPK Value Extensions
Semi-Automatic S*Pattern Configuration
V1.4.2 08-31-22
ICTT System Sciences
Understand your systems.

Figure 14: Informal Illustration of the Pattern Configuration Process



	POPULATED METACLASSES ("THEN")																					
TRIGGERING METACLASSES ("IF")	Feature	Interaction	Role	Design Component	Requirement Statement	State	Interface	Architectural Relationship	Input/Output	Port	System of Access	Counter Requirement Statement	Failure Mode	Feature Impact	Feature Attribute	Role Attribute	Design Component Attribute	Input/Output Attribute	Fitness Attribute Coupling	Decomposition Attribute Coupling	Characterization Attribute Coupling	IO Attribute Coupling
Stakeholder Input	▨																					
Feature	■	▨												▨	▨							
Interaction		■	▨		▨	▨	▨	▨	▨	▨	▨											
Role			■	▨	▨	▨	▨	▨	▨	▨	▨					▨						
Design Component				■									▨				▨					
Requirement Statement					■							▨										
State						■																
Interface							■															
Architectural Relationship								■														
Input/Output									■									▨				
Port										■												
System of Access											■											
Counter Requirement Statement												■										
Failure Mode													■									
Feature Impact														■								
Feature Attribute															■				▨			
Role Attribute																■				▨	▨	
Design Component Attribute																	■					
Input/Output Attribute																		■				▨
Fitness Attribute Coupling																			■			
Decomposition Attribute Coupling																				■		
Characterization Attribute Coupling																					■	
IO Attribute Coupling																						■

*Figure 15: Formal List of Configuration Predecessors and Successors, S*Metamodel*

2.7 Typical Performance Times for Configured Model Generation

Owing to the compression power of S*Patterns, a comparatively small S*Pattern can be used to generate relatively large configured S*Models. The time required for this mostly automated process is of interest, and has been an emphasis in the progression of technologies utilized in the three generations of Configuration Wizards to date, described by Section 1.7.

For the basic model generation process described by Section 2.4, the following stages contribute to the overall performance time:

Table 2: Contributions to Automated Run Time

Automated Step of Configuration Process	Performing Component
1. Pattern export time from Pattern Repository	Modeling Tool or Repository, Export Process
2. Pattern Import time for Configuration Wizard	Configuration Wizard, Button 1
3. Configured Model population time	Configuration Wizard, Button 2
4. Configured Model export time	Configuration Wizard, Button 3
5. Configured Model import time	Modeling Tool or Repository, Import Process

The time required for these steps naturally depends heavily upon model sizes and computing engine configurations. However, based on the progressive optimization across several generations of Configuration Wizard technologies, the dominant time in the above stages, for large configured models, is typically seen to be Step 5, the importing of a new configured model into the modeling tool or repository. This is in keeping with years of experience with the insertion of data into databases. A few minutes for the entire series of steps is typical.

It is also expected that this model configuration process is used to originate a new configured model, not for its subsequent model life cycle editing. Accordingly, the process described here is expected to be relatively infrequent across the life cycle of configured models.

2.8 Purging the Configuration Wizard of Confidential Data

During use, the Configuration Wizard temporarily holds Configurable Pattern and Configured Model data in its internal data tables. This allows for multiple configuration iterations (repeat cycles of Figure 11) to be carried out before a configured model is returned to the external Model Repository.

In case the Pattern or Model data used with the Configuration Wizard is confidential and you wish to provide the Configuration Wizard to someone else for other use, you can purge (remove) the internal Pattern and Model data from the Wizard, using the following automated steps:

1. Using Button 1, load an “empty” or non-confidential “demonstration” pattern into the Wizard.
2. Using Button 2, generate an “empty” or non-confidential “demonstration” configured model.
3. Using Button 3, generate a set of “empty” or non-confidential “demonstration” output files.

3 External Interface Specification and Wizard Internals

3.1 The Wizard External Interfaces Specification

Appendix A provides a sample of the External Interfaces Specification for the Configuration Wizard. Note that this interface specification is actually a built-in tab that is part of the Configuration Wizard itself.

As implied by Figure 11, there are three external actors that interact with the Configuration Wizard, implying External Interfaces on the Wizard for its interactions with all three:

1. **Interfaces to the Modeling Tool or Repository**: These are implemented as file exchanges, using Comma Separated Variable (.csv) files. The Interface Specification of Appendix A indicates the file names, column headings, and semantic contents of each column. (While this form is not as sophisticated as XML exchanges or API services, it provides a relatively simple to observe and debug form of interface in the current generation of technology.)
2. **Interfaces to the Human User**: As illustrated by the user sections of this document, these interfaces include tabular text data entry and display, control buttons, check boxes, and navigational links. These are detailed in both the Interface Specification and in Sections 1 and 2 of this User Guide.
3. **Interfaces to Saved Configuration Parameter Files**: As shown in Figure 11, and discussed in Section 2.5, the Configuration Wizard can save and load stored configuration parameter value files. These files are of type (.xlsx), contain multiple tabs, and their tabs and columns are documented by the Interface Specification of Appendix A.

3.2 The Internal Data Flow, Queries Architecture, and JOIN Technology

The overall data flow internal to the Configuration Wizard is dominated by the flow through a network of relational queries. This technology was chosen for the third generation Configuration Wizard based on the insight that the core conceptual S*Pattern Configuration Algorithm is in effect based upon projections of "upstream" already-configured model data onto "downstream" pattern data and its rules, and that this is equivalent to a series of relational INNER JOINs. To achieve the third generation Configuration Wizard goal of scalability to very large models, commercially proven JOIN technologies have been used, where industry optimization of JOIN algorithms over decades of platforms have created a sound base. The current JOIN platform used for the Configuration Wizard is based on the Microsoft Power Query® M-Code capability, available in forms including native Microsoft Excel 2016.

The flow of these queries is controlled by the Configuration Wizard internal Control Program, discussed in the next section. The queries themselves are specified by M-Code, which is similar in its JOIN aspects of interest here to ANSI SQL.

Appendix F provides detailed data flow diagrams of the queries across the entire process summarized by Figure 11. For each of the queries there, details of the individual query are specified by the query source code. A sample of that query source code is provided by Appendix C.

3.3 The Wizard Internal Control Program and Source Code

The overall flow of Configuration Wizard queries, as well as other Configuration Wizard interaction with the external actors, is controlled by the Configuration Wizard Control Program. Among other things, this control program manages the Control Panel user interactions, invokes queries, and reads and writes various external exchange files.

The Control Program is written in Visual Basic, and its (commented) source code can be directly accessed through the Excel Developer Menu for the Configuration Wizard. A sample of the Control Program source code is provided by Appendix B.

The purpose of the Control Program is overall control of the Configuration Wizard, but not detailed query semantic data processing of pattern and model data, which is provided by the Queries described in the next section.

3.4 The Wizard Queries and their Source Code

The detailed semantic information processing of the patterns and models by the Configuration Wizard is provided by the encoded Queries. These are encoded in (commented) M-Code, which resembles relational SQL queries for the core JOIN functions used by the Configuration Wizard.

The source code for all these queries is contained in the Configuration Wizard, and can be generated by navigating to the wizard Query Source tab and pushing the Generate Queries Source Code Listing button there. A sample of that commented source code is provided by Appendix C.

4 Appendices

4.1 Appendix A: Wizard External Interfaces Specification—Sample Extract

Use Case	Tool to Agent	Agent to Tool	Agent to User	User to Agent	IO #	IO Name (File Name for Files)	Table Name in Integrated Report	IO Type	Col #	Column Name (Exchanged Table Heading)	Meaning and Notes
Configure Pattern	X				IO 0	Export Tables	N/A	XLS File	1		Integrated rept of Pattern Tables from repository: Name of Tbl, or "#" for Tbl Hdr Row, or Tbl Row Nos., or empty to delimit each Tbl section of integrated report.
Configure Pattern	X				IO 0	Export Tables	N/A	XLS File	2-N		Tbl Col Hdgs, or Tbl Col Data; Contains IO 1,2,3,18-22,29-30
Configure Pattern	X				IO 1	Pattern Features File	Pattern Features File	CSV File	1	Name	
Configure Pattern	X				IO 1	Pattern Features File	Pattern Features File	CSV File	2	Configuration Rule	
Configure Pattern	X				IO 2	Pattern Feature Attributes File	Pattern Feature Attributes File	CSV File	1	Owner	Feature Name
Configure Pattern	X				IO 2	Pattern Feature Attributes File	Pattern Feature Attributes File	CSV File	2	Name	Feature Attribute
Configure Pattern	X				IO 2	Pattern Feature Attributes File	Pattern Feature Attributes File	CSV File	3	Applied Stereotype	Feature PK Flag ("Feature Attribute" or "Feature Primary Key")
Configure Pattern	X				IO 2	Pattern Feature Attributes File	Pattern Feature Attributes File	CSV File	4	Possible Values	Possible values, using colon (:) delimiters within single cell list
Configure Pattern	X				IO 3	Pattern Features-Interactions File	Pattern Features-Interactions File	CSV File	1	Type (Role B)	Feature Name
Configure Pattern	X				IO 3	Pattern Features-Interactions File	Pattern Features-Interactions File	CSV File	2	FPK Value	FPK Value
Configure Pattern	X				IO 3	Pattern Features-Interactions File	Pattern Features-Interactions File	CSV File	3	Type (Role A)	Interaction Name
Configure Pattern	X				IO 3	Pattern Features-Interactions File	Pattern Features-Interactions File	CSV File	4	IPK Rule	Interaction PK Rule
Configure Pattern	X				IO 18	Pattern Interactions-Roles File	Pattern Interaction Roles	CSV File	1	Type (Role B)	Interaction Name
Configure Pattern	X				IO 18	Pattern Interactions-Roles File	Pattern Interaction Roles	CSV File	2	Type (Role A)	Role Name
Configure Pattern	X				IO 18	Pattern Interactions-Roles File	Pattern Interaction Roles	CSV File	3	<<HasRole>> IPK Value	RPK Rule; guillemets portion is optional
Configure Pattern	X				IO 18	Pattern Interactions-Roles File	Pattern Interaction Roles	CSV File	4	<<HasRole>> RPK Rule	IPK Value; guillemets portion is optional
Configure Pattern	X				IO 19	Pattern Role Attributes File	Pattern Role Attributes File	CSV File	1	Owner	Logical System Name
Configure Pattern	X				IO 19	Pattern Role Attributes File	Pattern Role Attributes File	CSV File	2	Name	Attribute Name

Use Case	Tool to Agent	Agent to Tool	Agent to User	User to Agent	IO #	IO Name (File Name for Files)	Table Name in Integrated Report	IO Type	Col #	Column Name (Exchanged Table Heading)	Meaning and Notes
Configure Pattern	**X**				IO 20	Pattern Role-Des Compons File	Pattern Functional Role Allocation	CSV File	1	Type (Role B)	Des Compon Name
Configure Pattern	**X**				IO 20	Pattern Role-Des Compons File	Pattern Functional Role Allocation	CSV File	2	Type (Role A)	Role Name
Configure Pattern	**X**				IO 20	Pattern Role-Des Compons File	Pattern Functional Role Allocation	CSV File	3	<<FunctionalRoleAllocation>> IPPK Value	PSPK Rule; guillemets portion is optional
Configure Pattern	**X**				IO 20	Pattern Role-Des Compons File	Pattern Functional Role Allocation	CSV File	4	<<FunctionalRoleAllocation>> Configuration Rule	RPK Value; guillemets portion is optional
Configure Pattern	**X**				IO 21	Pattern Des Compons Attributes File	Pattern Des Compons Attributes File	CSV File	1	Owner	Des Compon Name
Configure Pattern	**X**				IO 21	Pattern Des Compons Attributes File	Pattern Des Compons Attributes File	CSV File	2	Name	Des Compon Attribute

4.2 Appendix B: Control Program Source Code—Sample Extract

```
'---------------------------------------
'
' Pattern Configuration Wizard Control Program
'
'    Controls overall flow of pattern configuration process, including queries
'    Obtains local pattern from repository tool
'    Returns configured model, generated from pattern,back to repository tool
'    See also wizard Interface Specification, built into wizard.
'    See also Pattern Configuration Wizard user documentation
'    See also query code and documentation, built into wizard.
'
'  Copyright 2021, 2022 System Sciences, LLC.
'  Available under Creative Commons CC SA BY public licensing.
'
'---------------------------------------
Dim SinglePatternFileEnabled, FileLoggingEnabled, BackgroundQueryEnabled As Long
Dim NumOfFileTypes, MaxTableRows, InputFilesQty, IntegratedReptFileRowNum, FirstFileRowNum, LastFileRowNum As Integer
Dim TableName, TableRangetoExtract, SepFileName, ControlFile, ColID, RowID As String
Dim TableRange As Range
Dim ConfigureInterfaces, ConfigureCouplings, ConfigureStates, ConfigureFMEA As Boolean

Sub Macro5()
'------------------------------
'
'   Button B1: Refresh Configurable Pattern from Model Repository
'
'-------------------------------
Dim Msg As String
Dim LogPath As String
Dim SourceFile, DestinationFile As String
Dim MyFSO As FileSystemObject
' Warn user to consider saving configuration input values before proceeding further.
   response = MsgBox("This action will blank out previously entered pattern Configuration Input Values. You can Save Configuration Input Values first,  using Button 7, and later Retrieve Configuration Input Values, using Button 8.
Proceed with pattern refresh data loading anyway? ", vbYesNo)
 If response = vbNo Then
    Exit Sub
 End If
ColID = "C"     ' Set column and row of progress log output on control sheet
RowID = 8
FirstRowID = 8  ' Set range of log space to be cleared prior to logging.
LastRowID = 130
ClearProgressLog FirstRowID, LastRowID, ColID  ' Clear the progress log
ProgressLog RowID, ColID, "Pattern Template Loading Started:", 0  ' Start logging.
```

```
'Refresh check box status for file  logging and including configuration of interfaces, states, couplings, FMEA
Macro10
Macro12
Macro13
Macro14
Macro15
Macro16
'Disable_Background_Refresh
BackgroundQuery = False
'------------------------------------------------
'
' Set Full Path Names to Pattern Input Files, Based on User Admin of Input File Locs
'
'------------------------------------------------
SourceFilesDirectory = ThisWorkbook.Sheets("Control").Cells(8, "L")
InputFilesQty = 22   ' Number of pattern input file types
IntegratedReptFileRowNum = 2    ' Row number in file util directory tab of integrated pattern input report file
FirstFileRowNum = 3   '  Number of first row in file util directory tab of individual pattern input file names
LastFileRowNum = FirstFileRowNum + InputFilesQty - 1 ' Number of last row in file util directory tab
MaxTableRows = 3000  ' Max number of rows allowed in a pattern input table from pattern.
For FileNo = 1 To InputFilesQty
   SourceFileFullName = SourceFilesDirectory & "\" & ThisWorkbook.Sheets("tableConfig").Cells(FileNo + 2, "B")
   ThisWorkbook.Sheets("tableConfig").Cells(FileNo + 2, "C") = SourceFileFullName '
Next FileNo
'----------------------------------------------------
'
' There are two options for incoming pattern data tables, depending on optioning of pattern repository--
'          Option 1: Multiple CSV files of one pattern table per file, or . . .
'          Option 2:  A single integrated XLS file containing all the pattern tables
'              Selection of one of these options is by wizard user on wizard control tab
' Check status of single versus multiple pattern input file control selection
'
'-----------------------------------------------------
If (ActiveSheet.CheckBoxes("Check Box 12").Value = 1) Then
       SinglePatternFileEnabled = True
       ProgressLog RowID, ColID, "Single Integrated Pattern File Xfer Mode", 0
  Else: SinglePatternFileEnabled = False
       ProgressLog RowID, ColID, "Multiple Pattern File Xfer Mode", 0
End If
'--------------------------------------------------------
'
' For Option 2 above, following routine generates CSV input pattern files (same as Option 1) from single XLS file
'
'---------------------------------------------------------
If SinglePatternFileEnabled = "True" Then
```

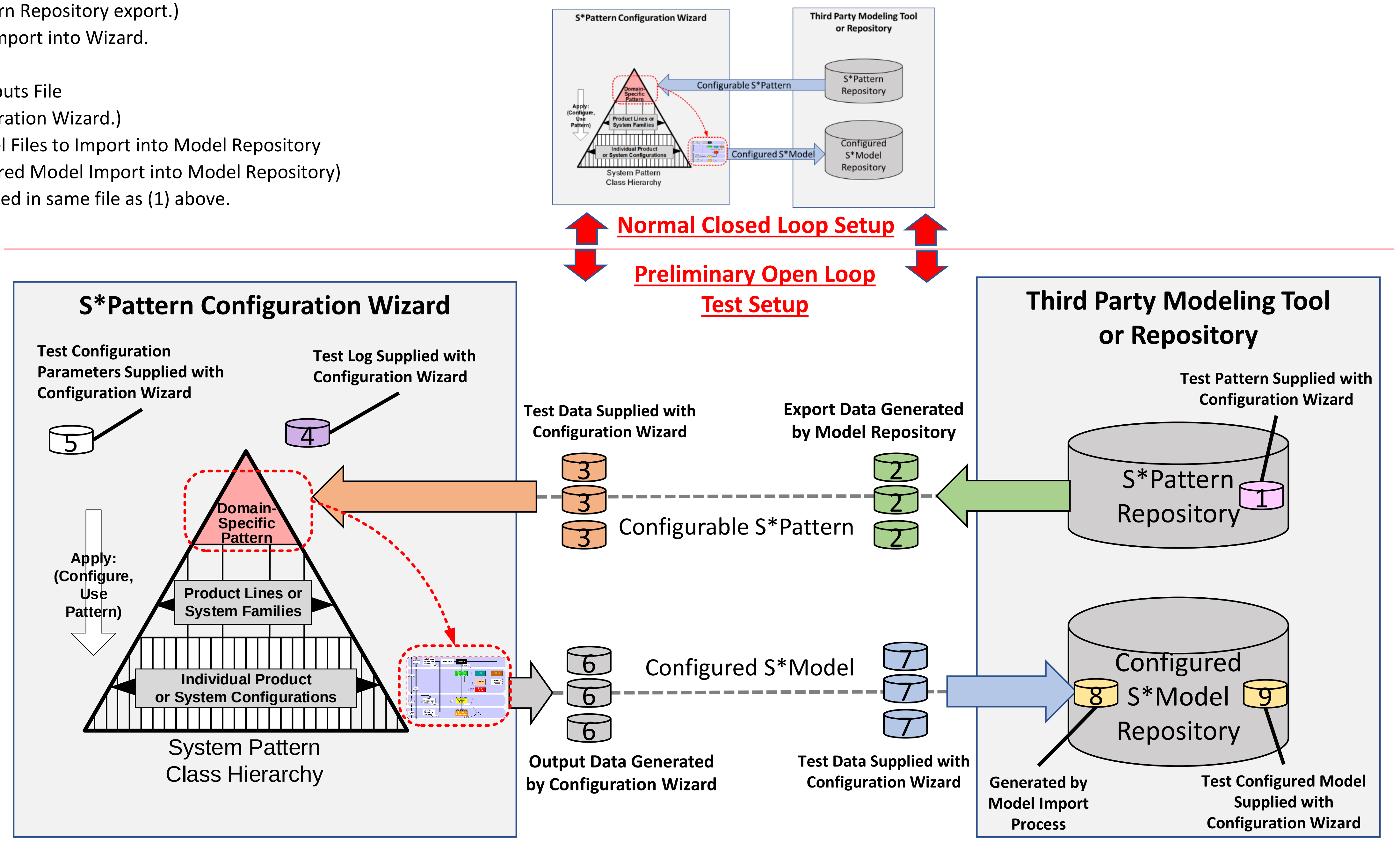
```
' Load the integrated tables report file for subsequent separation of its tables into files.

   ControlFile = ActiveWorkbook.Name       ' Remember name of wizard to return to it below
   Application.ScreenUpdating = False      ' Suspend screen updating
   Worksheets("IO 0 Integrated Pttrn Table Set").Range("A:Z").Clear  ' Clear report input sheet early in case following fails
   SourceFileFullName = SourceFilesDirectory & "\" & ThisWorkbook.Sheets("tableConfig").Cells(IntegratedReptFileRowNum, "B")
   ThisWorkbook.Sheets("tableConfig").Cells(IntegratedReptFileRowNum, "C") = SourceFileFullName '
   SingleInputFileName = ThisWorkbook.Sheets("tableConfig").Cells(IntegratedReptFileRowNum, "C") '  Get name of single integrated rept file to load
   SingleInputShortName = ThisWorkbook.Sheets("tableConfig").Cells(IntegratedReptFileRowNum, "B") ' Get simple file name for file close cmd to follow
   Workbooks.Open SingleInputFileName      'Open single integrated tables spreadsheet file
   Worksheets("Report").Range("A:Z").Copy  'Copy the integrated tables file for use in later table separation.
   Application.DisplayAlerts = False       ' Disable user dialog for closing
   Workbooks(SingleInputShortName).Close SaveChanges:=False
   Workbooks(ControlFile).Activate         ' Switch to wizard
   Worksheets("IO 0 Integrated Pttrn Table Set").Range("A:Z").PasteSpecial Paste:=xlPasteAll 'Load to IO 0 Tab for table parsing
   Application.DisplayAlerts = True        ' Re-enable user dialog after closing and pasting
   Selection.NumberFormat = "@" ' format copied table as text for the parsing that will follow
' Count the number of non-blank rows in integrated table items count report, for progress log
   NumRowsNotBlank = 0 ' in case of error
   On Error GoTo DoneCounting
   NumRowsNotBlank = Worksheets("IO 0 Integrated Pttrn Table Set").Range("A:A").Cells.SpecialCells(xlCellTypeConstants).Count
DoneCounting: On Error GoTo 0   ' turn off error trapping
   ProgressLog RowID, ColID, "Integrated Rept File Loaded", NumRowsNotBlank  'Log integrated file has been loaded
' Loop through the list of table/file names to be parsed from the single integrated file.
   For FileNo = 1 To InputFilesQty
      TableName = ThisWorkbook.Sheets("tableConfig").Cells(FileNo + FirstFileRowNum - 1, "D") 'Name of incoming table to find in integrated file.
      SepFileName = ThisWorkbook.Sheets("tableConfig").Cells(FileNo + FirstFileRowNum - 1, "C") 'Name of separated file to created from incoming table
      MetamodelSeg = ThisWorkbook.Sheets("tableConfig").Cells(FileNo + FirstFileRowNum - 1, "E") 'Name of Metamodel Segment this table is in.

         If (MetamodelSeg = "Interfaces" And ConfigureInterfaces = False) Then GoTo NextFile  ' Check Wizard User options for not used parts of metamodel
         If (MetamodelSeg = "Couplings" And ConfigureCouplings = False) Then GoTo NextFile
         If (MetamodelSeg = "States" And ConfigureStates = False) Then GoTo NextFile
         If (MetamodelSeg = "FMEA" And ConfigureFMEA = False) Then GoTo NextFile

      ExtractTable TableName, SepFileName 'Find table in incoming file and write to separated file
      Application.ScreenUpdating = True
      DoEvents  ' This DoEvents smooths out the user log viewing experience, but adds on the order of 10 seconds to the overall button 1 loading run
      Application.ScreenUpdating = False
NextFile:     Next FileNo  ' Advance to next table/file type to be parsed from the single integrated set
   Application.ScreenUpdating = True
End If 'End of table extraction and file generation process.
'---------------------------------------------------------------
```

4.3 Appendix C: Detail Query Source Code—Sample Extract

```
/////////////////////////////////////////////////////////////////////////////////////
//
//  This query produces populated Requirement Statements by an Inner Join between (Populated Interactions + Populated Roles) and (Pattern Interaction-Role-Requirements Rules)
//
////////////////////////////////////////////////////////////////////////////////////
//
//  Begin with pattern configuration rules Interaction-Role-Requirement table;
//  Perform inner join with table of populated interactions and populated roles
//  Initially the join checks for match of Roles and Interactions with pattern, but not yet check of IPK and RPK values
//
// Access method used for source data chosen to minimize later load times
//
let
   #"Source" = Excel.CurrentWorkbook(){[Name="Pattern_Interactions_Roles_Requirements_File"]}[Content],
   #"Source2" = Excel.CurrentWorkbook(){[Name="Populated_Roles_and_RPKs"]}[Content],
   #"Merged Queries" = Table.NestedJoin(Source, {"Interaction Name", "Role Name"}, Source2, {"Interaction Name", "Role Name"}, "Source2", JoinKind.Inner),
   #"Removed Other Columns" = Table.SelectColumns(#"Merged Queries",{"Interaction Name", "IPK Value", "Role Name", "RPK Value", "RSPK Rule", "Requirement ID", "Requirement Statement","Source2"}),
   #"Expanded Populated Roles and RPKs" = Table.ExpandTableColumn(#"Removed Other Columns", "Source2", {"IPK Value", "RPK Value"}, {"IPK Value.1", "RPK Value.1"}),
//  There should be no blank rows at this point, but remove any blank rows for robustness sake.
   #"Removed Blank Rows" = Table.SelectRows(#"Expanded Populated Roles and RPKs", each not List.IsEmpty(List.RemoveMatchingItems(Record.FieldValues(_), {"", null}))),
//
//  Check for IPK rule satisfied, checking both the case of IPK = *ANY* and the case of IPK = anything else; filter out any exceptions:
//
   #"Filtered Rows with IPK Rule=*ANY*" = Table.SelectRows(#"Removed Blank Rows", each [IPK Value] = "*ANY*"),
   #"Filtered Rows with IPK Value=IPK Rule" = Table.SelectRows(#"Removed Blank Rows", each [IPK Value] = [IPK Value.1]),
   #"Combined Rows" = Table.Combine({#"Filtered Rows with IPK Value=IPK Rule",  #"Filtered Rows with IPK Rule=*ANY*" }),
//
//  Check what remains (from above filtering) for RPK rule also satisfied, checking both the case of RPK = *ANY* and the case of RPK = anything else; filter out any exceptions:
//
   #"Filtered Rows with RPK Rule=*ANY*" = Table.SelectRows(#"Combined Rows", each [RPK Value] = "*ANY*"),
   #"Filtered Rows with RPK Value=RPK Rule" = Table.SelectRows(#"Combined Rows", each [RPK Value] = [RPK Value.1]),
   #"Combined Rows2" = Table.Combine({#"Filtered Rows with RPK Value=RPK Rule",  #"Filtered Rows with RPK Rule=*ANY*" }),
//
//  For surviving rows, generate Req Stmt PK (RSPK) value, by using RSPK Rule in pattern on populated IPK and RPK values.
//  RSPK Rule may contain any one of the following: "", "IPK", "RPK", "RIPK", "/<text string>/", "IPK/<text string>/", "RPK/<text string>/", "RIPK/<text string>/".
//  Begin by extracting the populated IPK value, if the RSPK rule contains "IPK":
//
   #"Added Conditional Column" = Table.AddColumn(#"Combined Rows2", "IPK Flag", each if Text.Contains([RSPK Rule], "IPK") then [IPK Value.1] else ""),
//  Then extract the populated RPK value, if the RSPK rule contains "RPK":
   #"Added Conditional Column1" = Table.AddColumn(#"Added Conditional Column", "RPK Flag", each if Text.Contains([RSPK Rule], "RPK") then [RPK Value.1] else ""),
//  Then extract the text string value, if the RSPK rule contains a text string within "/" delimiters, extract:
   #"Inserted Text Between Delimiters" = Table.AddColumn(#"Added Conditional Column1", "Text Between Delimiters", each Text.BetweenDelimiters([RSPK Rule], "/", "/"), type text),
//  Then form delimiter characters "-" or "", to separate concatenated value fields if they are not blank:
   #"Added Conditional Column2" = Table.AddColumn(#"Inserted Text Between Delimiters", "Delimiter1", each if [IPK Value.1] = "" then "" else if [RPK Value.1] = "" then "" else "-"),
//  Then extract a concatenated IPK-RPK value, if the RSPK rule contains "RIPK":
   #"Added Conditional Column3" = Table.AddColumn(#"Added Conditional Column2", "RIPK Flag", each if Text.Contains([RSPK Rule], "RIPK") then [RPK Value.1]&[Delimiter1]&[IPK Value.1] else ""),
   #"Added Conditional Column4" = Table.AddColumn(#"Added Conditional Column3", "Delimiter2", each if [IPK Flag]&[RPK Flag]&[RIPK Flag] = "" then "" else if [Text Between Delimiters] = "" then "" else "-"),
//  Then form RSPK value from above intermediates:
   #"Inserted Merged Column" = Table.AddColumn(#"Added Conditional Column4", "RSPK Value", each Text.Combine({[IPK Flag]&[RPK Flag]&[RIPK Flag], [Delimiter2] , [Text Between Delimiters]}), type text),
//  Generate columns with [RSPK] value, owning Interaction, qualified names, RTF reference, relationship owning package
   #"Added Conditional Column5" = Table.AddColumn(#"Inserted Merged Column", "Configured Requirement ID", each if [RSPK Value] <> "" then [Requirement ID] & " [" & [RSPK Value] & "]" else [Requirement ID]),
   #"Added Conditional Column6" = Table.AddColumn(#"Added Conditional Column5", "IPK In Brackets", each if [IPK Value.1] = "" then "" else " [" & [IPK Value.1] & "]"),
   #"Added Custom" = Table.AddColumn(#"Added Conditional Column6", "Owning Interaction", each "02 Configured Model::03 Interaction Framework::" & [Interaction Name] & [IPK In Brackets]),
   #"Added Conditional Column7" = Table.AddColumn(#"Added Custom", "Configured Requirement ID, Qualified Name", each [Owning Interaction] & "::" & [Configured Requirement ID]),
//    #"Added Custom1" = Table.AddColumn(#"Added Conditional Column7", "Configured Requirement, Qualified Name", each [Owning Interaction] & "::" & [Requirement Statement]),
   #"Added Conditional Column8" = Table.AddColumn(#"Added Conditional Column7", "Configured Interaction", each if [IPK Value.1] = "" then [Interaction Name] else [Interaction Name] & " [" & [IPK Value.1] & "]" ),
   #"Added Conditional Column9" = Table.AddColumn(#"Added Conditional Column8", "Configured Role", each if [RPK Value.1] = "" then [Role Name] else [Role Name] & " [" & [RPK Value.1] & "]" ),
   #"Added Custom2" = Table.AddColumn(#"Added Conditional Column9", "Configured RTF, Qualified Name", each "02 Configured Model::03 Interaction Framework::"&[Configured Interaction]&"::Perform " & [Configured Role] & " in " & [Configured
Interaction]),
   #"Added Custom3" = Table.AddColumn(#"Added Custom2", "Owning Package", each "02 Configured Model::03 Interaction Framework"),
//  Housecleaning--remove remaining columns and duplicate rows; reorder columns
   #"Removed Columns" = Table.RemoveColumns(#"Added Custom3",{"Interaction Name", "IPK Value", "Role Name", "RPK Value", "IPK Value.1", "RPK Value.1", "IPK Flag", "Text Between Delimiters", "Delimiter2", "RSPK Rule", "RPK Flag", "Delimiter1", "RIPK
Flag",  "IPK In Brackets", "Configured Interaction", "Configured Role"}),
   #"Removed Duplicates" = Table.Distinct(#"Removed Columns"),
   #"Reordered Columns" = Table.ReorderColumns(#"Removed Duplicates",{"Configured Requirement ID", "Requirement Statement", "Owning Interaction", "Configured Requirement ID, Qualified Name","Configured RTF, Qualified Name", "Owning Package",
"Requirement ID", "RSPK Value"})
in
   #"Reordered Columns"
```

There are six test files supplied with the Configuration Wizard, numbered in the diagram:

1. S*Pattern File for Modeling Tool
2. N/A (Generate using S*Pattern Repository export.)
3. Importable Pattern Files to Import into Wizard.
4. Expected Wizard Log Results
5. Retrievable Configuration Inputs File
6. N/A (Generate using Configuration Wizard.)
7. Importable Configured Model Files to Import into Model Repository
8. N/A (Generate using Configured Model Import into Model Repository)
9. Configured S*Model, contained in same file as (1) above.


S*Pattern Configuration Wizard
Third Party Modeling Tool or Repository
Configurable S*Pattern
S*Pattern Repository
Configured S*Model
Configured S*Model Repository
Domain-Specific Pattern
Product Lines or System Families
Individual Product or System Configurations
Apply: (Configure, Use Pattern)
System Pattern Class Hierarchy
Normal Closed Loop Setup
Preliminary Open Loop Test Setup
S*Pattern Configuration Wizard
Test Configuration Parameters Supplied with Configuration Wizard
Test Log Supplied with Configuration Wizard
5
4
Domain-Specific Pattern
Apply: (Configure, Use Pattern)
Product Lines or System Families
Individual Product or System Configurations
System Pattern Class Hierarchy
Test Data Supplied with Configuration Wizard
Export Data Generated by Model Repository
3
3
3
Configurable S*Pattern
2
2
2
6
6
6
Configured S*Model
7
7
7
Output Data Generated by Configuration Wizard
Test Data Supplied with Configuration Wizard
Third Party Modeling Tool or Repository
Test Pattern Supplied with Configuration Wizard
S*Pattern Repository
1
Configured S*Model Repository
8
9
Generated by Model Import Process
Test Configured Model Supplied with Configuration Wizard

4.5 Appendix E: Pattern File Names and Paths Table

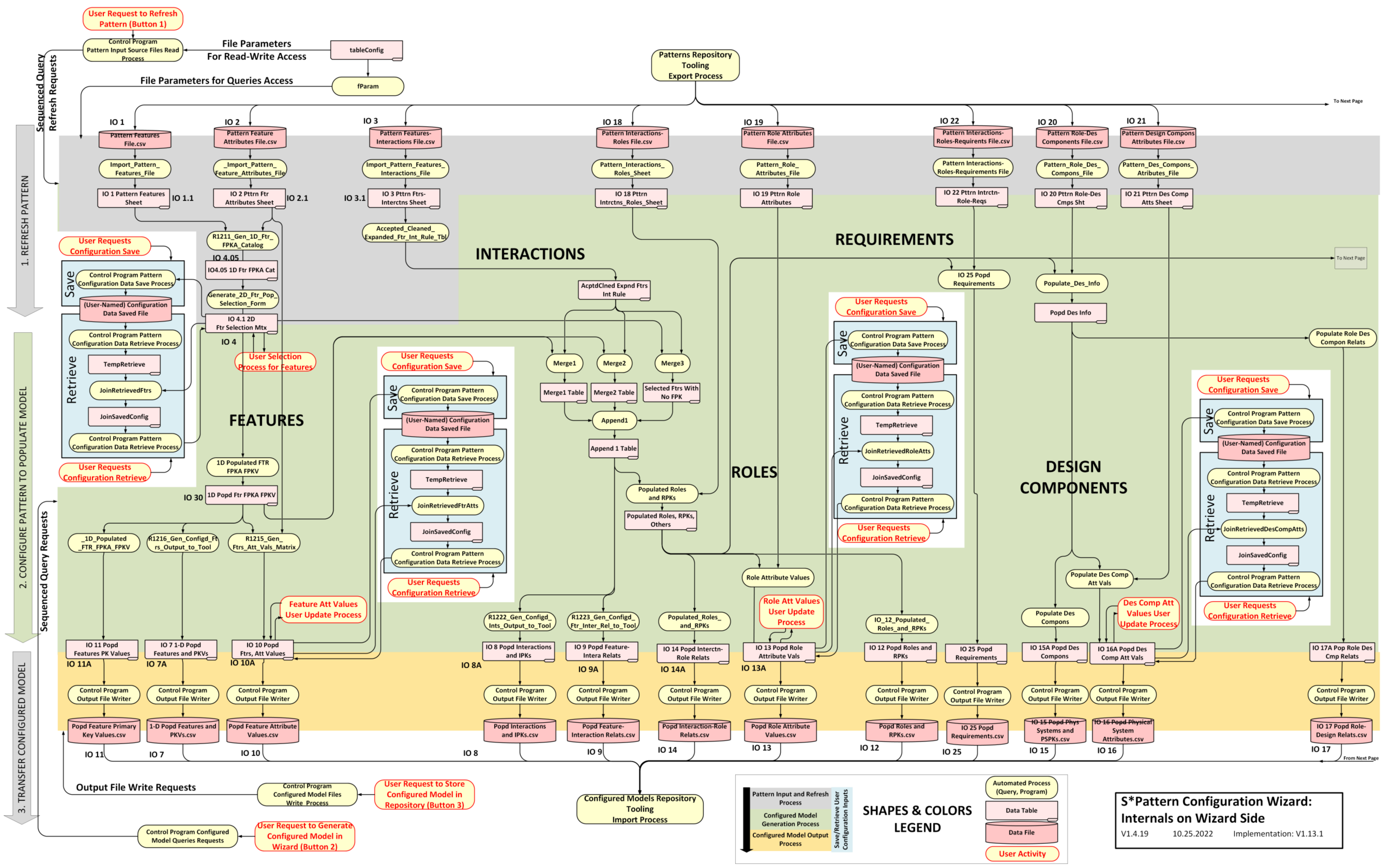
This table is built into a tab of the Configuration Wizard. It contains the file names of pattern files importable into the Configuration Wizard, generated by the Pattern Repository Export. The first row is the name of the Integrated Pattern Tables File, whose loading brings in all the required pattern data in one integrated file. The subsequent rows are the names of the individual files when the import is not from a single integrated file, along with their names when integrated into the single integrated file case. The Segment column identifies the overall class of information for control of the scope of the pattern content being imported. The path column is generated by the Configuration Wizard, based on the user's entry of the pattern import path name.

Index	FileType	Path	Integrated Pattern Report Section Name	Metamodel Segment
	Export Tables.xlsx	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Export Tables.xlsx		
1	Pattern Features File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Features File.csv	Pattern Features File	Base
2	Pattern Feature Attributes File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Feature Attributes File.csv	Pattern Feature Attributes File	Base
3	Pattern Features-Interactions File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Features-Interactions File.csv	Pattern Features-Interactions File	Base
4	Pattern Interactions-Roles File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Interactions-Roles File.csv	Pattern Interaction Roles	Base
5	Pattern Role Attributes File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Role Attributes File.csv	Pattern Role Attributes File	Base
6	Pattern Role-Des Compons File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Role-Des Compons File.csv	Pattern Functional Role Allocation	Base
7	Pattern Des Compons Attributes File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Des Compons Attributes File.csv	Pattern Des Compons Attributes File	Base

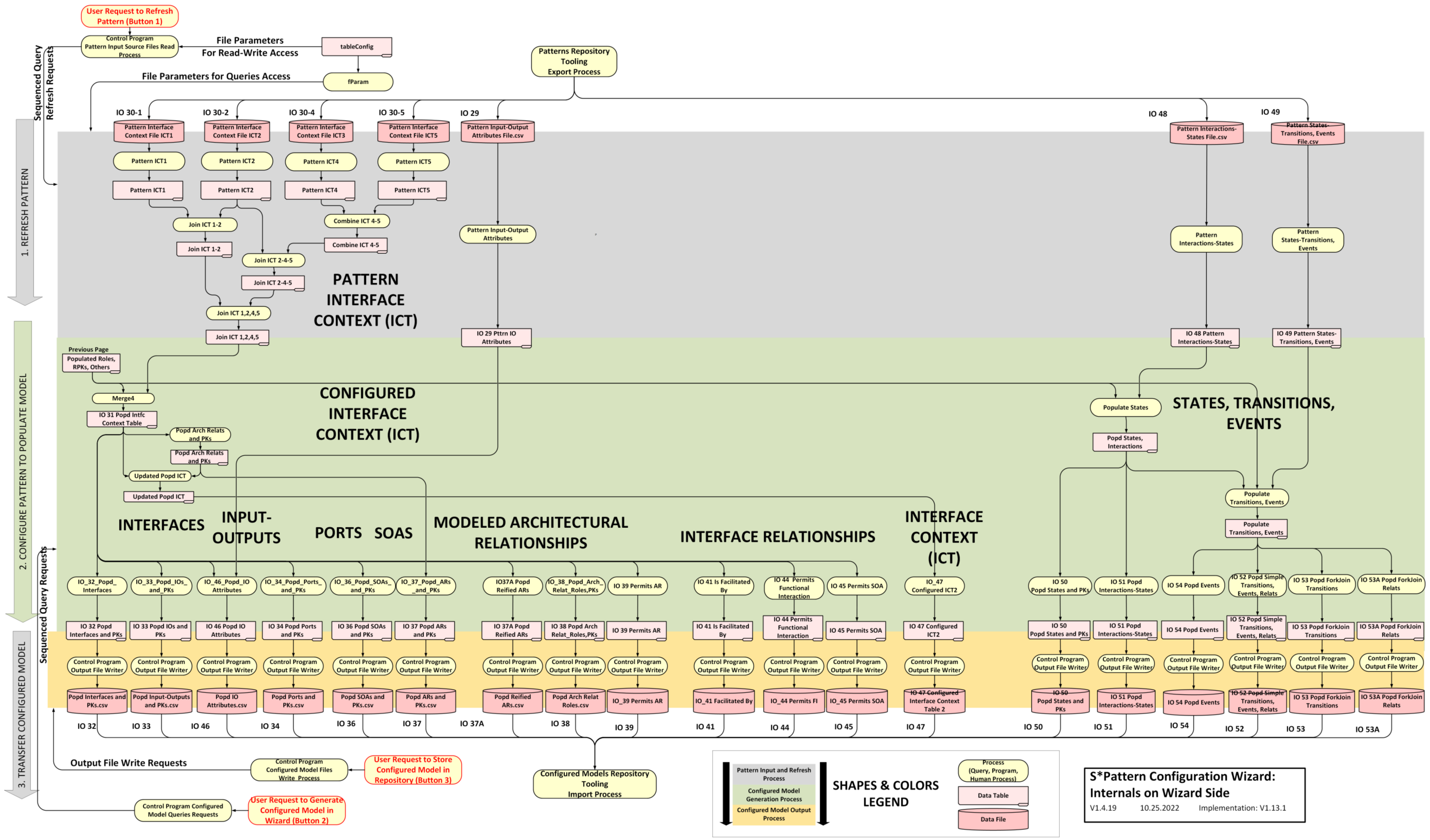
Index	FileType	Path	Integrated Pattern Report Section Name	Metamodel Segment
8	Pattern Interactions-Roles-Requirements File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Interactions-Roles-Requirements File.csv	Pattern Requirements	Base
9	Pattern Input-Output Attributes File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Input-Output Attributes File.csv	Pattern Input-Output Attributes File	Interfaces
10	Pattern ICT1 File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern ICT1 File.csv	Pattern Interface Context File ICT1	Interfaces
11	Pattern ICT2 File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern ICT2 File.csv	Pattern Interface Context File ICT2	Interfaces
12	Pattern ICT4 File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern ICT4 File.csv	Pattern Interface Context File ICT4	Interfaces
13	Pattern ICT5 File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern ICT5 File.csv	Pattern Interface Context File ICT5	Interfaces
14	Pattern Interactions-States File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Interactions-States File.csv	Pattern Interactions-States File	States
15	Pattern States-Transitions, Events File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern States-Transitions, Events File.csv	Pattern States-Transitions, Events File	States
16	Pattern Attribute Couplings File.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Attribute Couplings File.csv	Pattern Attribute Couplings	Couplings
17	Pattern Failure Impacts.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Failure Impacts.csv	Pattern Failure Impacts	FMEA

Index	FileType	Path	Integrated Pattern Report Section Name	Metamodel Segment
18	Pattern Counter Requirements.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Counter Requirements.csv	Pattern Counter Requirements	FMEA
19	Pattern Failure Impacts-Counter Requirements.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Failure Impacts-Counter Requirements.csv	Pattern Failure Impacts-Counter Requirements	FMEA
20	Pattern Failure Modes.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Failure Modes.csv	Pattern Failure Modes	FMEA
21	Pattern Failure Modes-Counter Requirements.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Failure Modes-Counter Requirements.csv	Pattern Failure Modes-Counter Requirements	FMEA
22	Pattern Failure Mode Context.csv	C:\Users\WSchindel\Documents\Docs\System Sciences, LLC\Tech\Systematica\STM Methodology\STMTemplatesWorksheets\Workbook\2020-21 Wkbk Experiments\File IntchgTstData--PwrCnvrt\FilesPowerCnvrtr08.28.2022\Pattern Failure Mode Context.csv	Pattern Failure Mode Context	FMEA

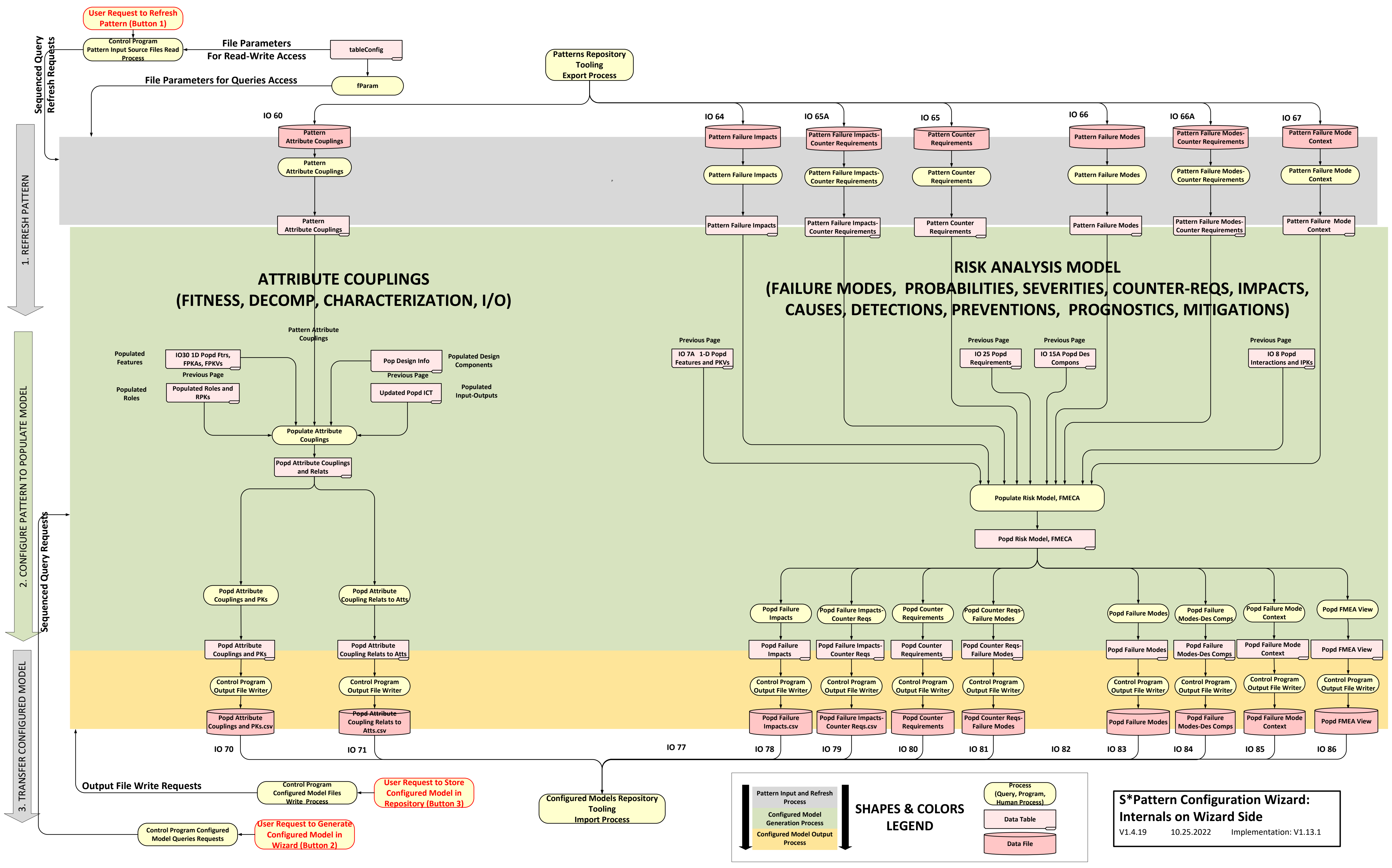
4.6 Appendix F: Data Flow and Queries Architecture Diagrams



User Request to Refresh Pattern (Button 1)
Control Program Pattern Input Source Files Read Process
File Parameters For Read-Write Access
tableConfig
File Parameters for Queries Access
fParam
Sequenced Query Refresh Requests
Patterns Repository Tooling Export Process
To Next Page
1. REFRESH PATTERN
2. CONFIGURE PATTERN TO POPULATE MODEL
3. TRANSFER CONFIGURED MODEL
IO 1
Pattern Features File.csv
Import_Pattern_Features_File
IO 1 Pattern Features Sheet
IO 1.1
IO 2
Pattern Feature Attributes File.csv
_Import_Pattern_Feature_Attributes_File
IO 2 Pttrn Ftr Attributes Sheet
IO 2.1
IO 3
Pattern Features-Interactions File.csv
Import_Pattern_Features_Interactions_File
IO 3.1
IO 3 Pttrn Ftrs-Interctns Sheet
Accepted_Cleaned_Expanded_Ftr_Int_Rule_Tbl
IO 18
Pattern Interactions-Roles File.csv
Pattern_Interactions_Roles_Sheet
IO 18 Pttrn Intrctns_Roles_Sheet
IO 19
Pattern Role Attributes File.csv
Pattern_Role_Attributes_File
IO 19 Pttrn Role Attributes
IO 22
Pattern Interactions-Roles-Requirents File.csv
Pattern Interactions-Roles-Requirements File
IO 22 Pttrn Intrctn-Role-Reqs
IO 20
Pattern Role-Des Components File.csv
Pattern_Role_Des_Compons_File
IO 20 Pttrn Role-Des Cmps Sht
IO 21
Pattern Design Compons Attributes File.csv
Pattern_Des_Compons_Attributes_File
IO 21 Pttrn Des Comp Atts Sheet
R1211_Gen_1D_Ftr_FPKA_Catalog
IO 4.05
IO4.05 1D Ftr FPKA Cat
Generate_2D_Ftr_Pop_Selection_Form
IO 4.1 2D Ftr Selection Mtx
IO 4
User Selection Process for Features
User Requests Configuration Save
Save
Control Program Pattern Configuration Data Save Process
(User-Named) Configuration Data Saved File
Retrieve
Control Program Pattern Configuration Data Retrieve Process
TempRetrieve
JoinRetrievedFtrs
JoinSavedConfig
User Requests Configuration Retrieve
INTERACTIONS
REQUIREMENTS
FEATURES
ROLES
DESIGN COMPONENTS
AcptdClned Expnd Ftrs Int Rule
Merge1
Merge2
Merge3
Merge1 Table
Merge2 Table
Selected Ftrs With No FPK
Append1
Append 1 Table
Populated Roles and RPKs
Populated Roles, RPKs, Others
JoinRetrievedFtrAtts
JoinRetrievedRoleAtts
JoinRetrievedDesCompAtts
IO 25 Popd Requirements
Populate_Des_Info
Popd Des Info
Populate Role Des Compon Relats
To Next Page
1D Populated FTR FPKA FPKV
IO 30
1D Popd Ftr FPKA FPKV
_1D_Populated_FTR_FPKA_FPKV
R1216_Gen_Configd_Ftrs_Output_to_Tool
R1215_Gen_Ftrs_Att_Vals_Matrix
Feature Att Values User Update Process
Sequenced Query Requests
IO 11 Popd Features PK Values
IO 11A
IO 7 1-D Popd Features and PKVs
IO 7A
IO 10 Popd Ftrs, Att Values
IO 10A
R1222_Gen_Configd_Ints_Output_to_Tool
R1223_Gen_Configd_Ftr_Inter_Rel_to_Tool
Populated_Roles_and_RPKs
Role Attribute Values
Role Att Values User Update Process
IO_12_Populated_Roles_and_RPKs
Populate Des Compons
Populate Des Comp Att Vals
Des Comp Att Values User Update Process
IO 8 Popd Interactions and IPKs
IO 8A
IO 9 Popd Feature-Intera Relats
IO 9A
IO 14 Popd Interctn-Role Relats
IO 14A
IO 13 Popd Role Attribute Vals
IO 13A
IO 12 Popd Roles and RPKs
IO 25 Popd Requirements
IO 15A Popd Des Compons
IO 16A Popd Des Comp Att Vals
IO 17A Pop Role Des Cmp Relats
Control Program Output File Writer
Popd Feature Primary Key Values.csv
1-D Popd Features and PKVs.csv
Popd Feature Attribute Values.csv
Popd Interactions and IPKs.csv
Popd Feature-Interaction Relats.csv
Popd Interaction-Role Relats.csv
Popd Role Attribute Values.csv
Popd Roles and RPKs.csv
IO 25 Popd Requirements.csv
IO 15 Popd Phys Systems and PSPKs.csv
IO 16 Popd Physical System Attributes.csv
IO 17 Popd Role-Design Relats.csv
IO 11
IO 7
IO 10
IO 8
IO 9
IO 14
IO 13
IO 12
IO 25
IO 15
IO 16
IO 17
From Next Page
Output File Write Requests
Control Program Configured Model Files Write Process
User Request to Store Configured Model in Repository (Button 3)
Configured Models Repository Tooling Import Process
Control Program Configured Model Queries Requests
User Request to Generate Configured Model in Wizard (Button 2)
Pattern Input and Refresh Process
Configured Model Generation Process
Configured Model Output Process
Save/Retrieve User Configuration Inputs
SHAPES & COLORS LEGEND
Automated Process (Query, Program)
Data Table
Data File
User Activity
S*Pattern Configuration Wizard: Internals on Wizard Side
V1.4.19
10.25.2022
Implementation: V1.13.1



User Request to Refresh Pattern (Button 1)
Control Program Pattern Input Source Files Read Process
File Parameters For Read-Write Access
tableConfig
File Parameters for Queries Access
fParam
Patterns Repository Tooling Export Process
Sequenced Query Refresh Requests
1. REFRESH PATTERN
IO 30-1
IO 30-2
IO 30-4
IO 30-5
IO 29
IO 48
IO 49
Pattern Interface Context File ICT1
Pattern Interface Context File ICT2
Pattern Interface Context File ICT3
Pattern Interface Context File ICT5
Pattern Input-Output Attributes File.csv
Pattern Interactions-States File.csv
Pattern States-Transitions, Events File.csv
Pattern ICT1
Pattern ICT2
Pattern ICT4
Pattern ICT5
Join ICT 1-2
Combine ICT 4-5
Join ICT 2-4-5
Join ICT 1,2,4,5
Pattern Input-Output Attributes
Pattern Interactions-States
Pattern States-Transitions, Events
PATTERN INTERFACE CONTEXT (ICT)
IO 29 Pttrn IO Attributes
IO 48 Pattern Interactions-States
IO 49 Pattern States-Transitions, Events
2. CONFIGURE PATTERN TO POPULATE MODEL
Previous Page Populated Roles, RPKs, Others
Merge4
IO 31 Popd Intfc Context Table
Popd Arch Relats and PKs
Updated Popd ICT
CONFIGURED INTERFACE CONTEXT (ICT)
Populate States
Popd States, Interactions
STATES, TRANSITIONS, EVENTS
Populate Transitions, Events
INTERFACES
INPUT-OUTPUTS
PORTS
SOAS
MODELED ARCHITECTURAL RELATIONSHIPS
INTERFACE RELATIONSHIPS
INTERFACE CONTEXT (ICT)
IO_32_Popd_Interfaces
IO_33_Popd_IOs_and_PKs
IO_46_Popd_IO Attributes
IO_34_Popd_Ports_and_PKs
IO_36_Popd_SOAs_and_PKs
IO_37_Popd_ARs_and_PKs
IO37A Popd Reified ARs
IO_38_Popd_Arch_Relat_Roles,PKs
IO 39 Permits AR
IO 41 Is Facilitated By
IO 44 Permits Functional Interaction
IO 45 Permits SOA
IO_47 Configured ICT2
IO 50 Popd States and PKs
IO 51 Popd Interactions-States
IO 54 Popd Events
IO 52 Popd Simple Transitions, Events, Relats
IO 53 Popd ForkJoin Transitions
IO 53A Popd ForkJoin Relats
IO 32 Popd Interfaces and PKs
IO 33 Popd IOs and PKs
IO 46 Popd IO Attributes
IO 34 Popd Ports and PKs
IO 36 Popd SOAs and PKs
IO 37 Popd ARs and PKs
IO 37A Popd Reified ARs
IO 38 Popd Arch Relat_Roles,PKs
IO 39 Permits AR
IO 41 Is Facilitated By
IO 44 Permits Functional Interaction
IO 45 Permits SOA
IO 47 Configured ICT2
IO 50 Popd States and PKs
IO 51 Popd Interactions-States
IO 54 Popd Events
IO 52 Popd Simple Transitions, Events, Relats
IO 53 Popd ForkJoin Transitions
IO 53A Popd ForkJoin Relats
Control Program Output File Writer
3. TRANSFER CONFIGURED MODEL
Sequenced Query Requests
Popd Interfaces and PKs.csv
Popd Input-Outputs and PKs.csv
Popd IO Attributes.csv
Popd Ports and PKs.csv
Popd SOAs and PKs.csv
Popd ARs and PKs.csv
Popd Reified ARs.csv
Popd Arch Relat Roles.csv
IO_39 Permits AR
IO_41 Facilitated By
IO_44 Permits FI
IO_45 Permits SOA
IO 47 Configured Interface Context Table 2
IO 50 Popd States and PKs
IO 51 Popd Interactions-States
IO 54 Popd Events
IO 52 Popd Simple Transitions, Events, Relats
IO 53 Popd ForkJoin Transitions
IO 53A Popd ForkJoin Relats
IO 32
IO 33
IO 46
IO 34
IO 36
IO 37
IO 37A
IO 38
IO 39
IO 41
IO 44
IO 45
IO 47
IO 50
IO 51
IO 54
IO 52
IO 53
IO 53A
Output File Write Requests
Control Program Configured Model Files Write Process
User Request to Store Configured Model in Repository (Button 3)
Configured Models Repository Tooling Import Process
Control Program Configured Model Queries Requests
User Request to Generate Configured Model in Wizard (Button 2)
Pattern Input and Refresh Process
Configured Model Generation Process
Configured Model Output Process
SHAPES & COLORS LEGEND
Process (Query, Program, Human Process)
Data Table
Data File
S*Pattern Configuration Wizard: Internals on Wizard Side
V1.4.19
10.25.2022
Implementation: V1.13.1



User Request to Refresh Pattern (Button 1)
Control Program Pattern Input Source Files Read Process
File Parameters For Read-Write Access
tableConfig
File Parameters for Queries Access
fParam
Sequenced Query Refresh Requests
Patterns Repository Tooling Export Process
1. REFRESH PATTERN
IO 60
Pattern Attribute Couplings
IO 64
Pattern Failure Impacts
IO 65A
Pattern Failure Impacts-Counter Requirements
IO 65
Pattern Counter Requirements
IO 66
Pattern Failure Modes
IO 66A
Pattern Failure Modes-Counter Requirements
IO 67
Pattern Failure Mode Context
ATTRIBUTE COUPLINGS (FITNESS, DECOMP, CHARACTERIZATION, I/O)
RISK ANALYSIS MODEL (FAILURE MODES, PROBABILITIES, SEVERITIES, COUNTER-REQS, IMPACTS, CAUSES, DETECTIONS, PREVENTIONS, PROGNOSTICS, MITIGATIONS)
2. CONFIGURE PATTERN TO POPULATE MODEL
Sequenced Query Requests
Pattern Attribute Couplings
Populated Features
IO30 1D Popd Ftrs, FPKAs, FPKVs
Previous Page
Populated Roles
Populated Roles and RPKs
Pop Design Info
Previous Page
Populated Design Components
Updated Popd ICT
Populated Input-Outputs
Populate Attribute Couplings
Popd Attribute Couplings and Relats
Previous Page
IO 7A 1-D Popd Features and PKVs
Previous Page
IO 25 Popd Requirements
Previous Page
IO 15A Popd Des Compos
Previous Page
IO 8 Popd Interactions and IPKs
Populate Risk Model, FMECA
Popd Risk Model, FMECA
Popd Attribute Couplings and PKs
Popd Attribute Coupling Relats to Atts
Popd Failure Impacts
Popd Failure Impacts-Counter Reqs
Popd Counter Requirements
Popd Counter Reqs-Failure Modes
Popd Failure Modes
Popd Failure Modes-Des Comps
Popd Failure Mode Context
Popd FMEA View
3. TRANSFER CONFIGURED MODEL
Control Program Output File Writer
Popd Attribute Couplings and PKs.csv
Popd Attribute Coupling Relats to Atts.csv
Popd Failure Impacts.csv
Popd Failure Impacts-Counter Reqs.csv
Popd Counter Requirements
Popd Counter Reqs-Failure Modes
Popd Failure Modes
Popd Failure Modes-Des Comps
Popd Failure Mode Context
Popd FMEA View
IO 70
IO 71
IO 77
IO 78
IO 79
IO 80
IO 81
IO 82
IO 83
IO 84
IO 85
IO 86
Output File Write Requests
Control Program Configured Model Files Write Process
User Request to Store Configured Model in Repository (Button 3)
Configured Models Repository Tooling Import Process
Control Program Configured Model Queries Requests
User Request to Generate Configured Model in Wizard (Button 2)
Pattern Input and Refresh Process
Configured Model Generation Process
Configured Model Output Process
SHAPES & COLORS LEGEND
Process (Query, Program, Human Process)
Data Table
Data File

S*Pattern Configuration Wizard: Internals on Wizard Side

V1.4.19 10.25.2022 Implementation: V1.13.1