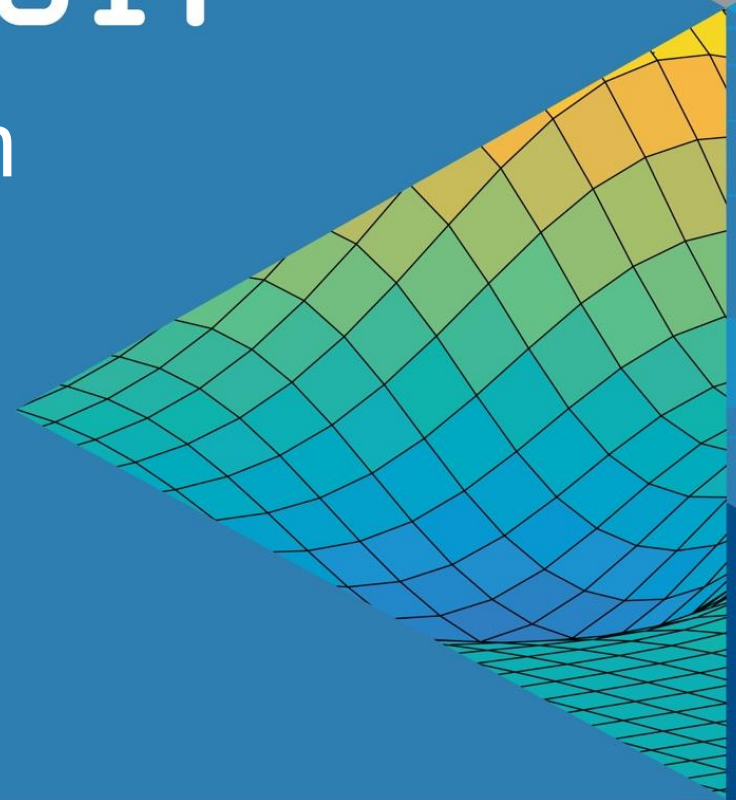
MATLAB EXPO 2017

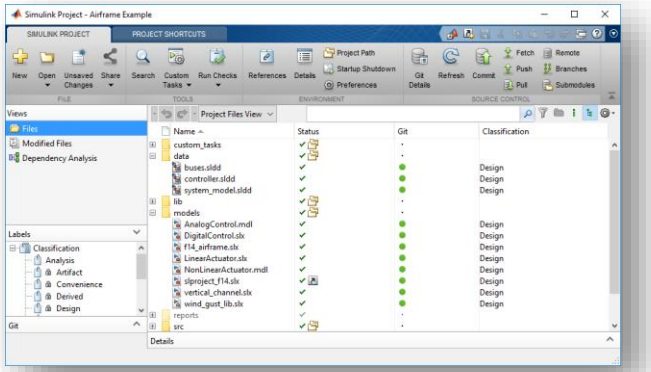
Team-Based Collaboration in Simulink

Sonia Bridge

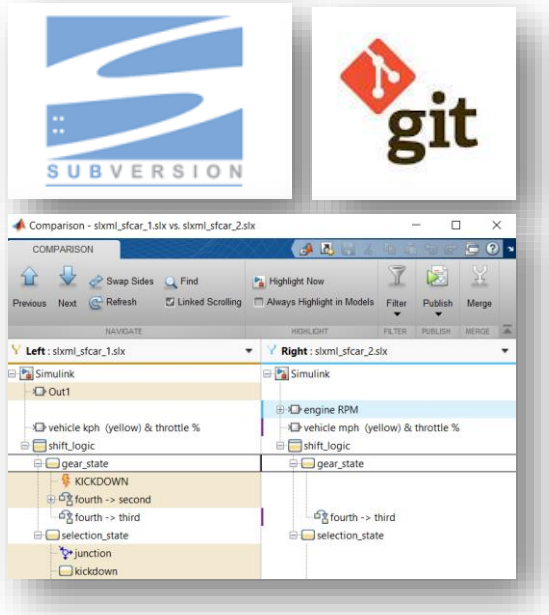


*Create **tools** that make it **easy** for teams to manage the full lifecycle of their **Model-Based Design** projects*

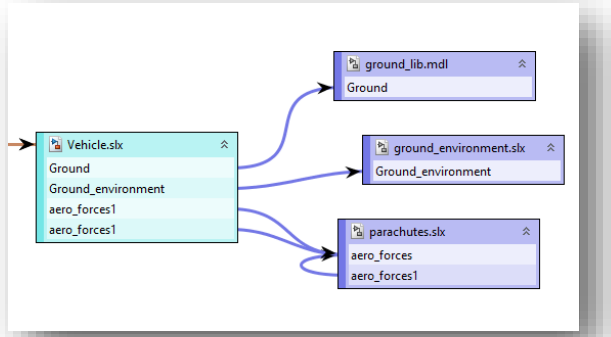
Collaborate



Integrate



Analyse



Common Challenges

How to:

- Create a more efficient team-based environment?
- Effectively componentize system designs including data?
- Track design changes?
- Use source control functionality within Simulink?
- Associate project-level information with files?
- Utilise automation to maximise efficiency in enforcing best practices?
- Share work within the group and outside the group?
- Transfer knowledge across projects?

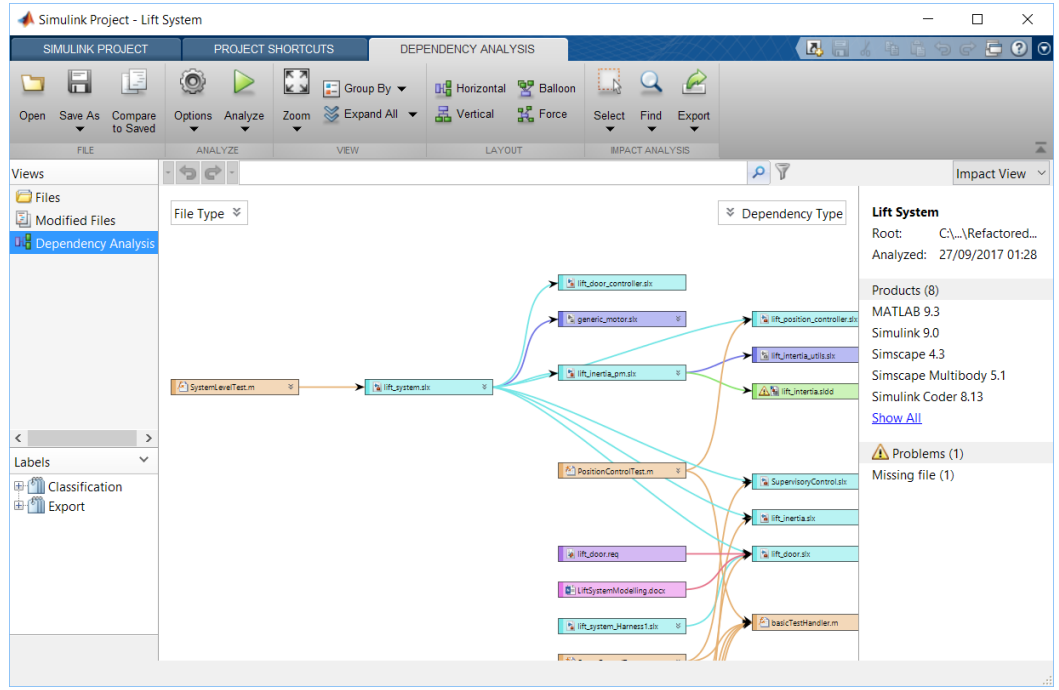
Simulink Project

Foundation technology supporting efficient Model-Based Design in teams

- Enables sharing of work
- Ensures a project is complete (files, data)
- Ensures consistent environment across a team (MATLAB path, data, slprj location, …)
- Supports different entry points and sharing best practices via shortcuts
- Enables associating project-level information with files via labels
- Part of Simulink (first shipped in R2011b)

Supports advanced users

- Impact analysis: what is the impact of a change?
- Rich APIs to script and customize



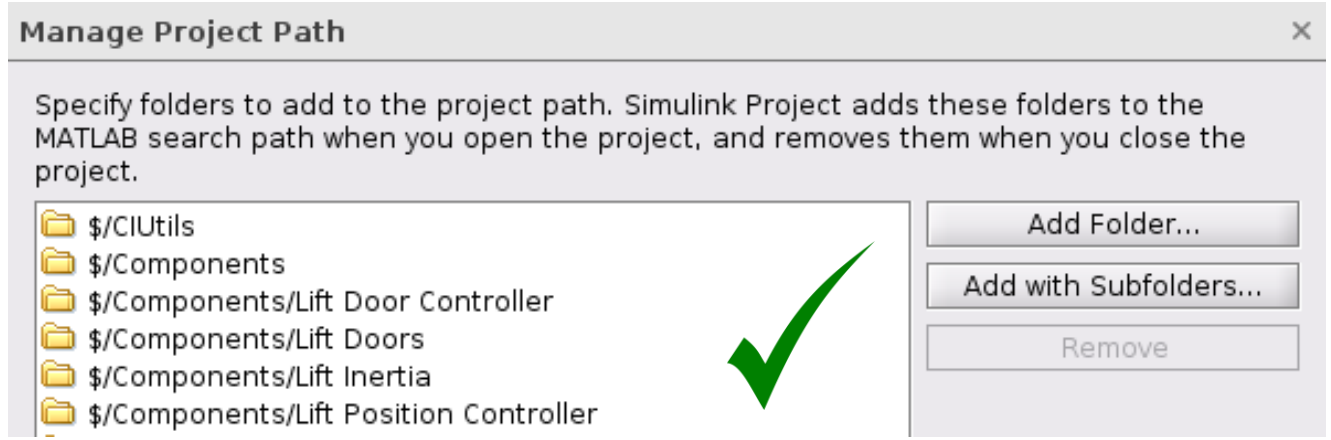
Getting started with an existing project

Simulink project “mistake-proofs” your team environment

No more MATLAB code required to manage

- MATLAB Path via UI
- Locations for generated files (“slprj”)

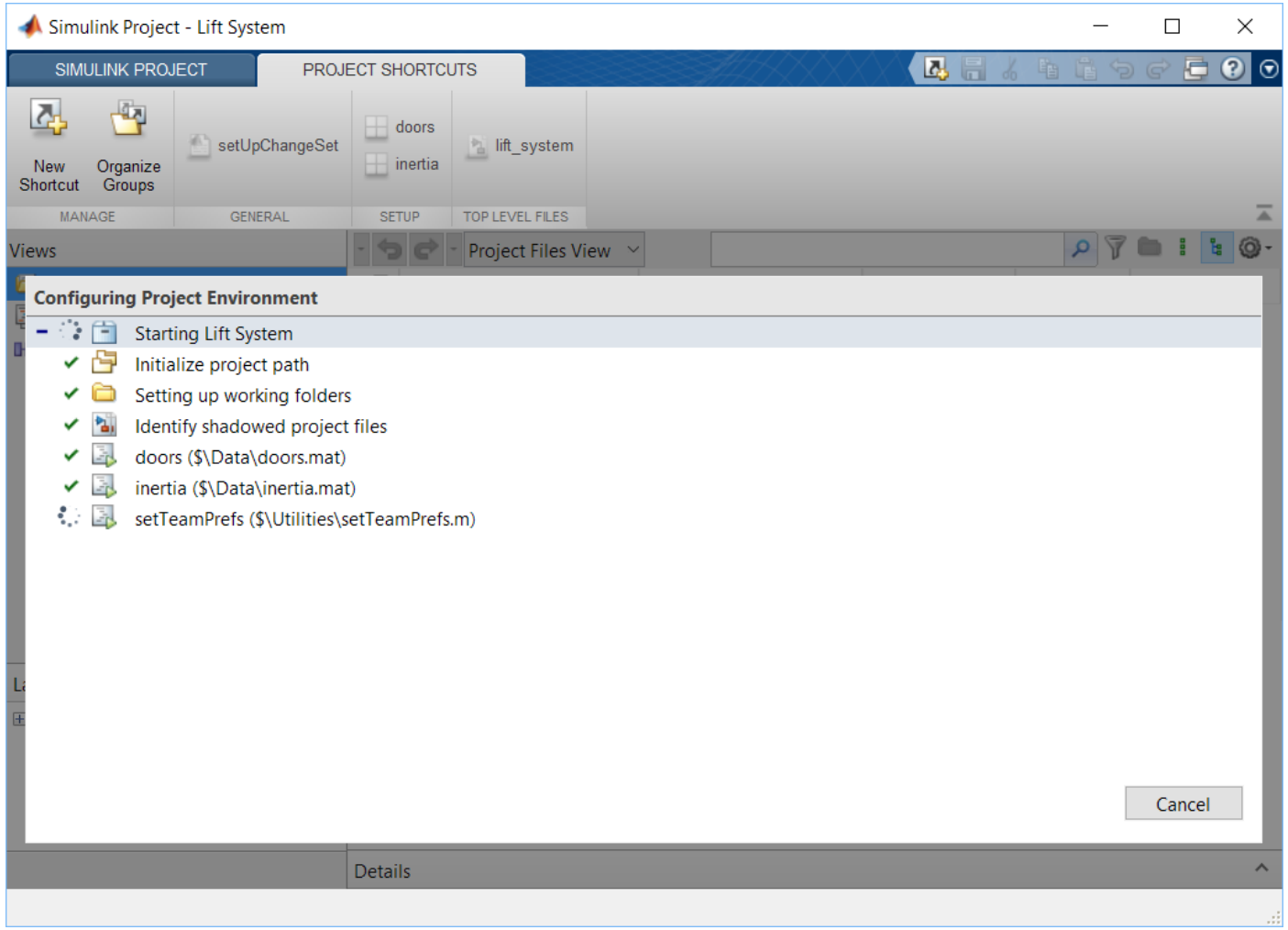
“I’m going to try my project on the new Linux cluster”



Simulink Projects Shortcuts

- Make it easy for *any* engineer (not just the engineer who created the project) to:
  - Find important files
  - Find and execute important or common operations
    - Make the top-level model in the project a shortcut
  - All debuggable
- Optionally set tasks to run at project start-up or shutdown
  - Provides formal mechanism for running initialization scripts
  - Makes it easier to ensure the symmetric shutdown scripts are called

Task Automation – Configuring Project Environment



- Robustly configure the team environment
- For everyone
- Automatically

Using Simulink Projects to Create a Consistent Cross-Team Environment

- Benefits:
  - Everyone on the team has the same environment
  - New team members can get started more quickly
  - Less wasted time debugging discrepancies

Integration with Source Control

How do people share and manage projects?

At an SAE webinar on “Model-Based Engineering”, question asked:

Q: “How do you manage the files and data within your projects?”

1. Named folders (“project_v1”, “project_v2”, etc.)
2. Source Control tool
3. Application Lifecycle Management (ALM) tool

How do people share and manage projects?

Majority use COTS tools for managing work & sharing information

- Source control
- Application Lifecycle Management (ALM)

Surprise was the number just using the file system

- Doesn’t scale well
- Doesn't support team work
- So why were they doing it?

Source Control Integrations

Microsoft Team Foundation Server (TFS) integration available now from MathWorks File Exchange

SUBVERSION

PERFORCE
Version everything.

git

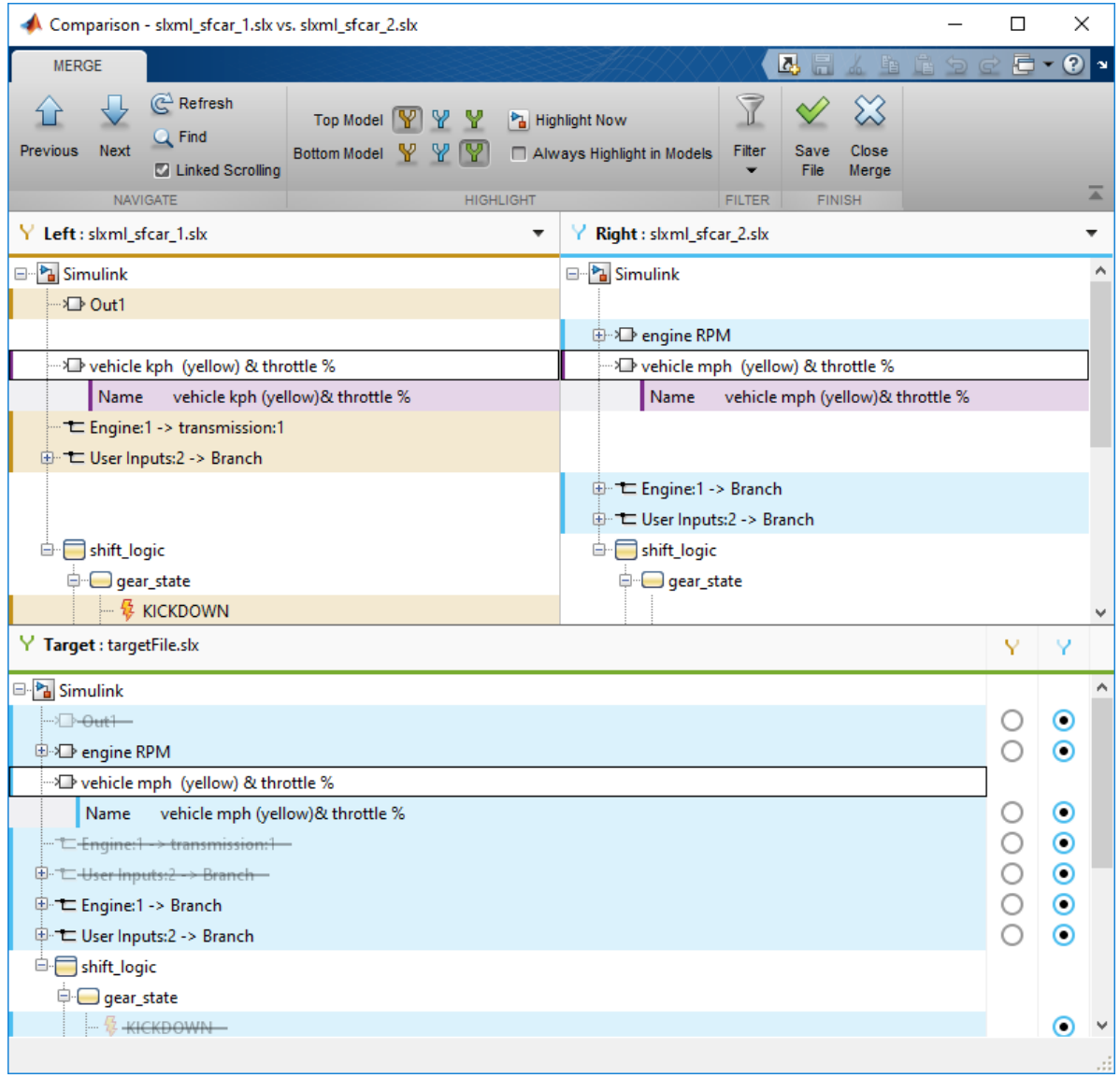
Team Foundation Server 2012
Microsoft

Compare and Merge Simulink Models

Simplified comparison and merge workflow for Simulink models

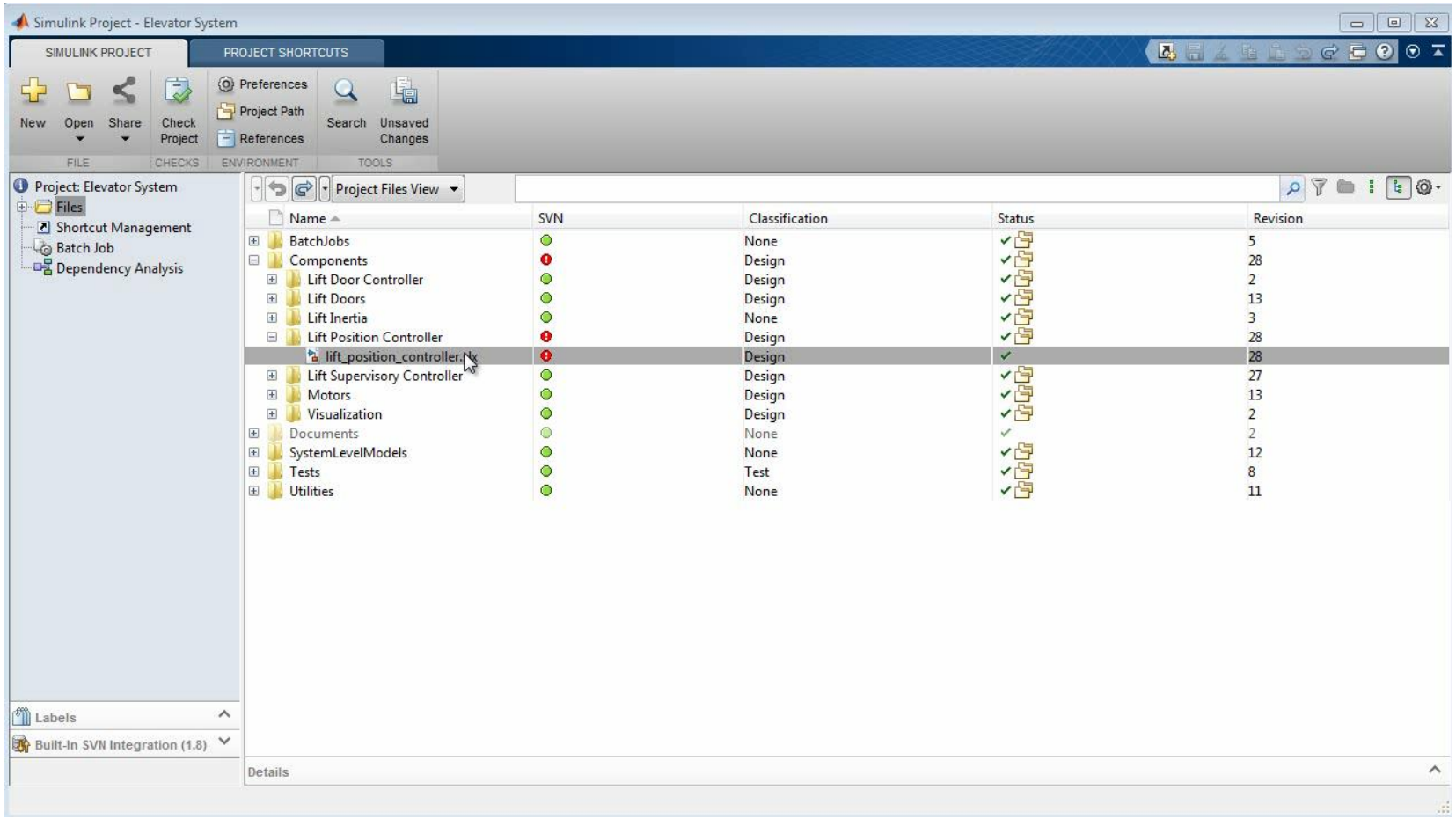
- Comparison and merge available with Simulink
- Easily select changes to merge into new target model file
- Highlight changes in the Simulink editor
- Launch comparison from the MATLAB desktop, current folder browser, command line, or source control
- Create reports for archiving and review

```
» slxml_sfcar
```



MATLAB EXPO 2017

Integrating Work from Different Engineers via Merge

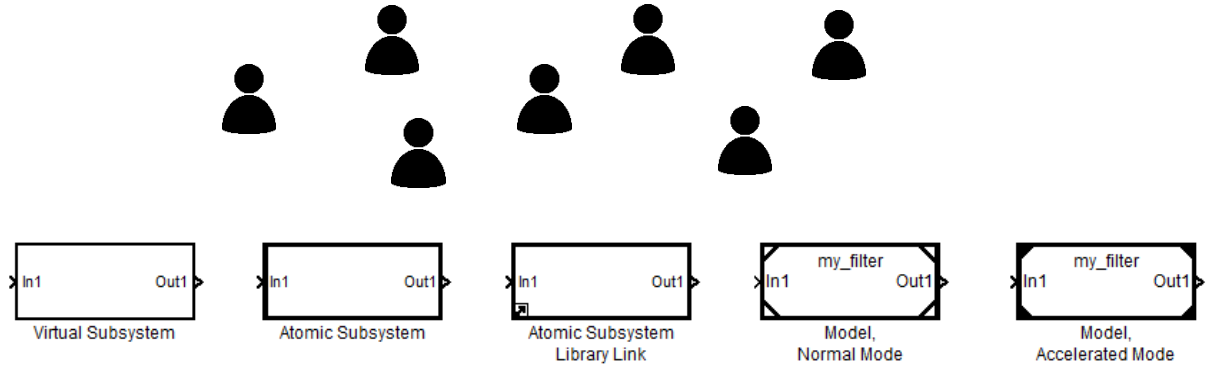


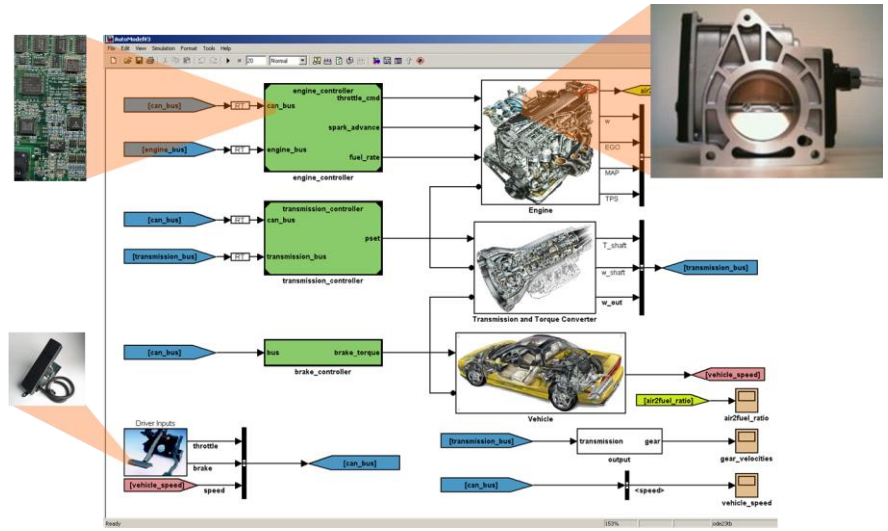
- Supports concurrent engineering
- Lets you concentrate on design

Componentization

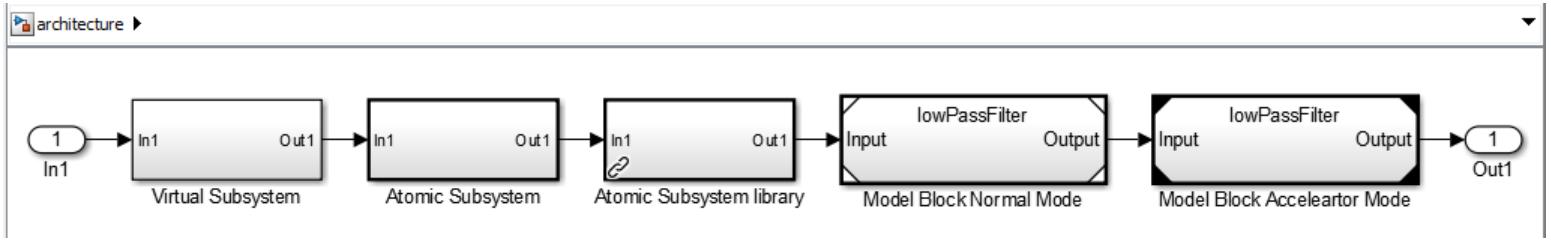
Complex Design Development through Componentization

- Supporting team-based workflows
  - Faster modular development
  - More effective verification
  - Increased reusability





Simulink Architectural Components



- Virtual subsystem
  - Graphical component – The contents are flattened to the level of the parent system during execution.
- Atomic subsystem
  - Simulink executes all blocks as a unit before executing the next block
  - Context dependent so inherits properties such as dimensions and data types from the parent model
- Model block
  - Executed as a unit
  - Context independent so doesn't inherit properties from parent model

Component selection strategy

- Virtual and Atomic Subsystems
  - When scalability is not an issue
  - When the atomic subsystem boundary is acceptable
  - During early development of the system

- Model Reference
  - When scalability is needed
  - When hard interfaces are critical
  - To enable concurrent teamwork and unit testing

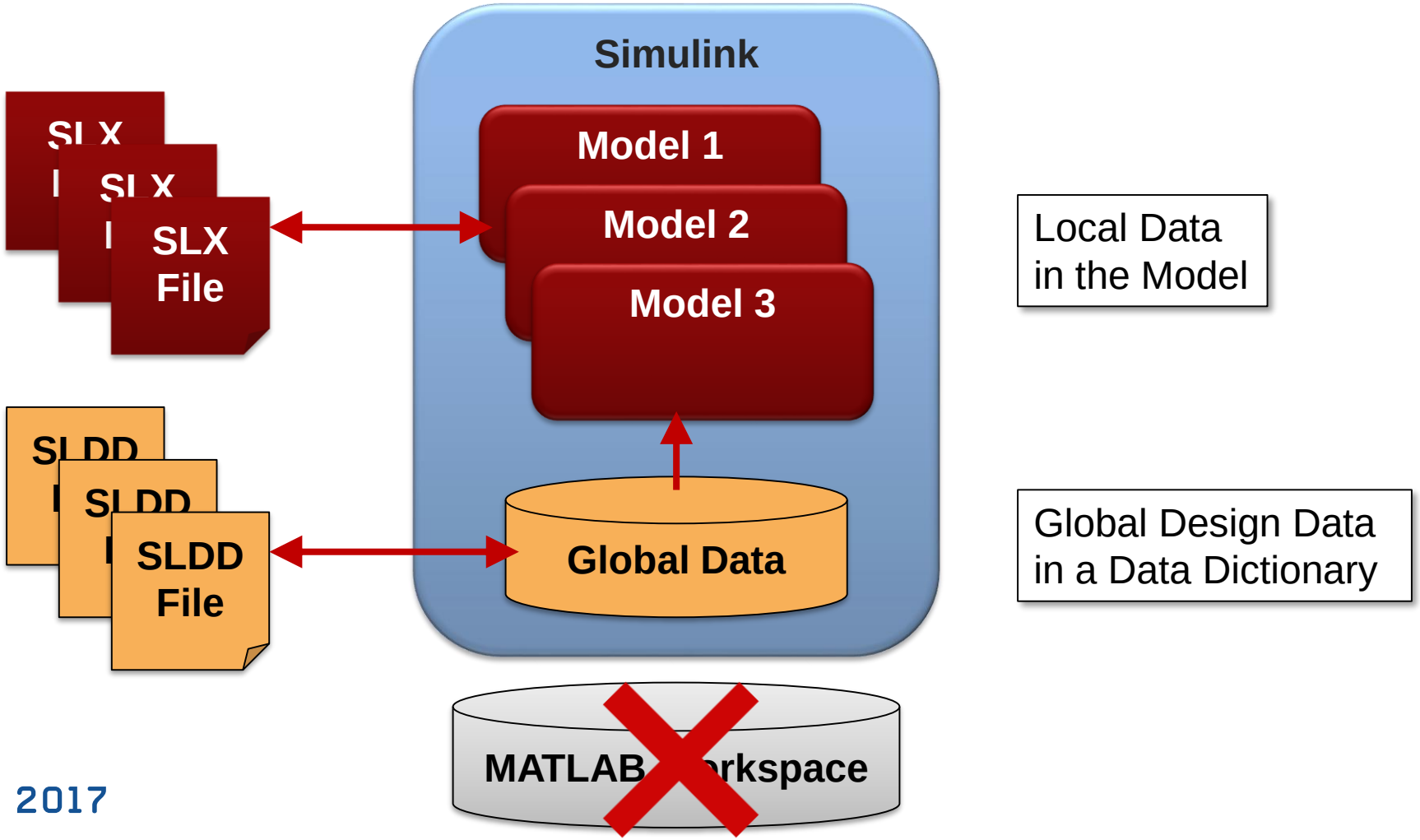
- Library Components
  - Reused utility functions

Component-Based Modelling

- Criteria for componentization:
  - Base the component boundaries on those of the real system
  - Define components distinctly so that only one engineer at a time needs to edit a component.
  - Subdivide components that are too big and those that could become too big as the design is elaborated.
- Recognize that there is no silver bullet
  - Experience is key here as well
- Start discussing this early in your project
  - What should be the criteria for componentization?
  - Who owns which component?

Partitioning Design Data

Executable Specification = Algorithm + **Data**



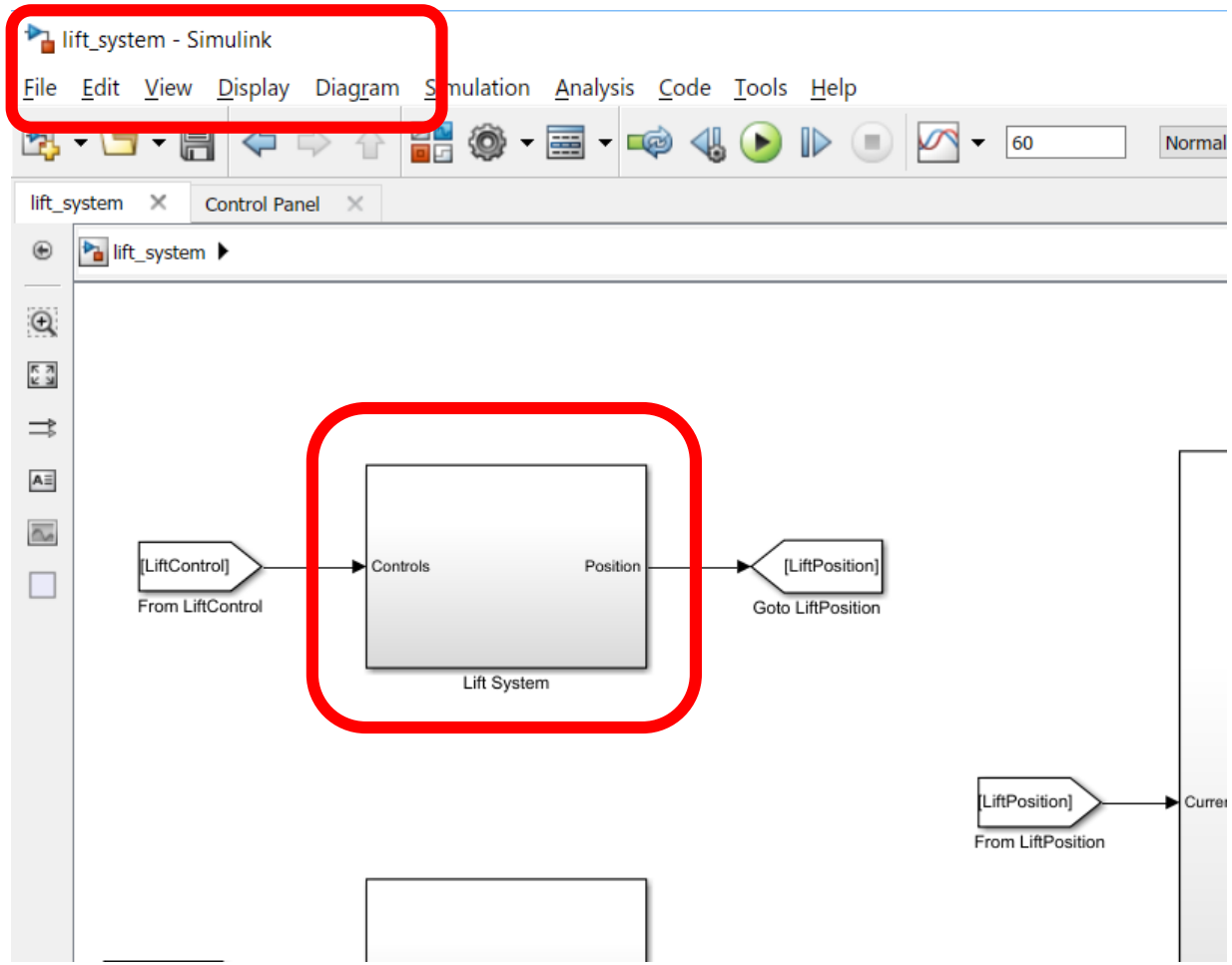
Why Simulink Data Dictionary?

Base Workspace Limitations

- Mixed with MATLAB data
- Lack of organization
- Lack of change detection
  - What changed?
  - How did it change?
  - Can’t revert changes
- Where did it come from?
- Lack of data persistence
- Conflict resolution issues

Simulink Data Dictionary

- Separate
- Partitioning
- Change detection
  - Shows changed items
  - Differencing
  - Revert
- Traceability
- Data persistence
- Conflict resolution



- This subsystem has same name as parent model
- Probably not the best name
- What is it..?

Demo

- Refactor into a new Model Reference
- Advisor helps automate/mistake proof the process
- Dependency analysis helps ensure we do not “lose” this new component
- Refactoring support for renaming
- Find dependencies to help work out why there are some other components with poorly chosen names (like “lift_intertia”)

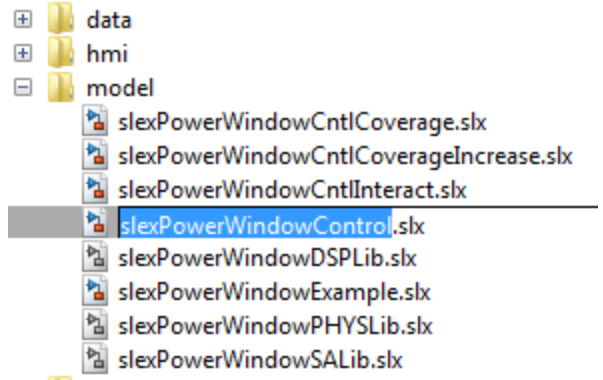
Simulink Project: Automatic Renaming

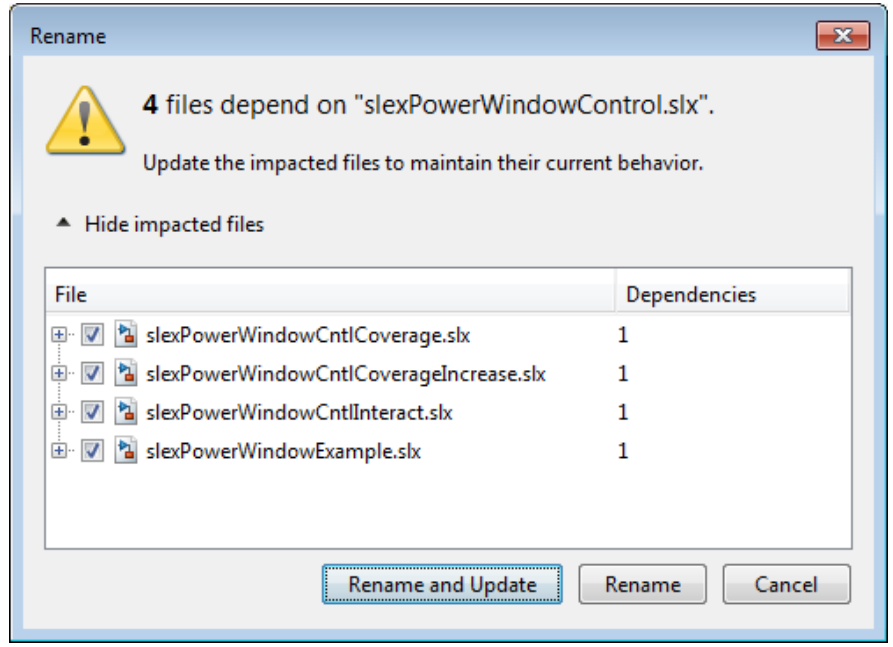
Automatically update files impacted by renaming, removing and deleting project files.

- Update model references and library links when renaming Simulink models.
- Update MATLAB code and model/block callbacks when renaming m/mlx files.
- Warn when deleting a file that is used by other files in the project.
- Update the MATLAB path when adding models or code files to the project.

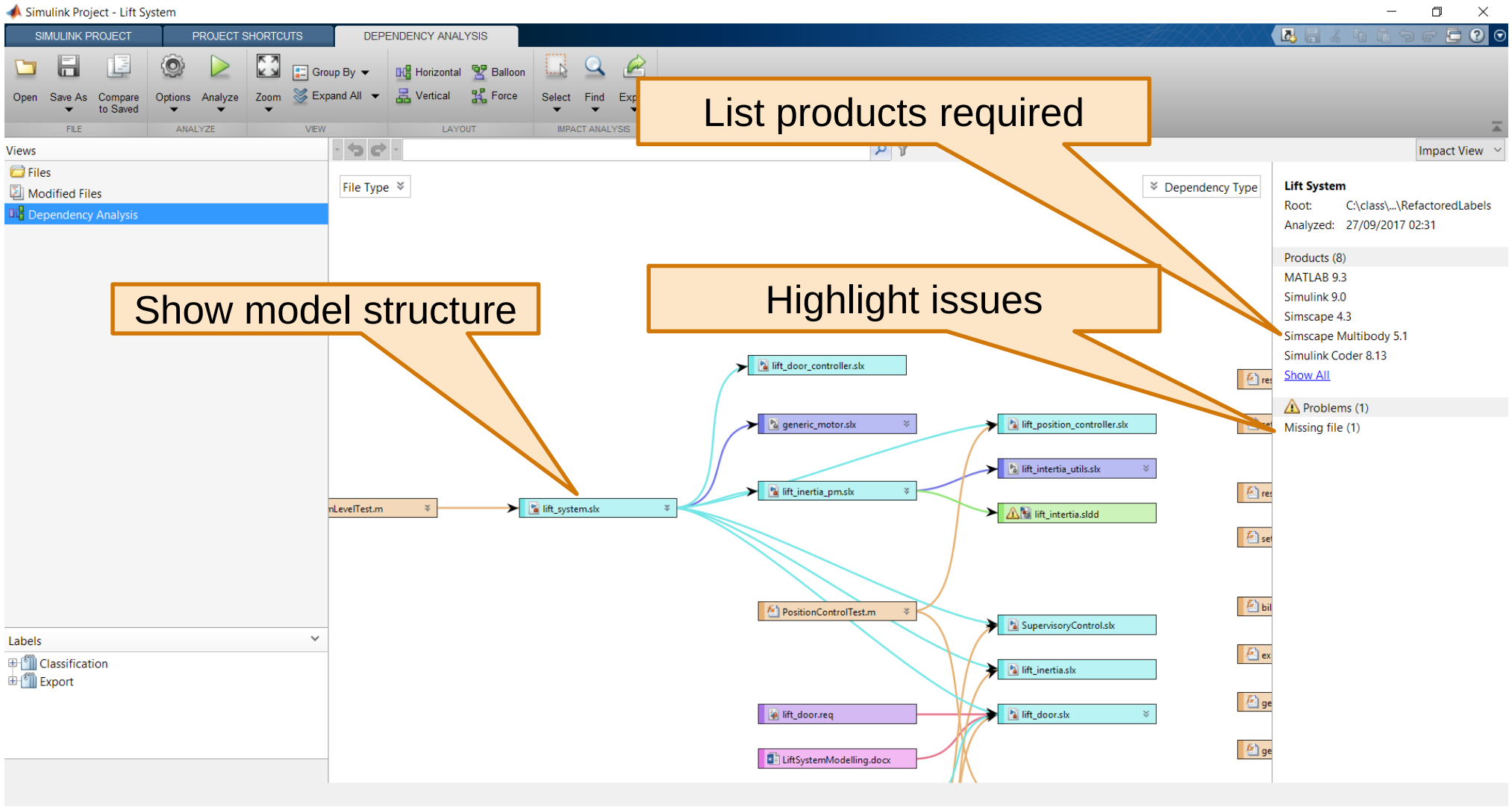
```
» slexPowerWindowStart
```

Rename, remove or delete a file from the project.





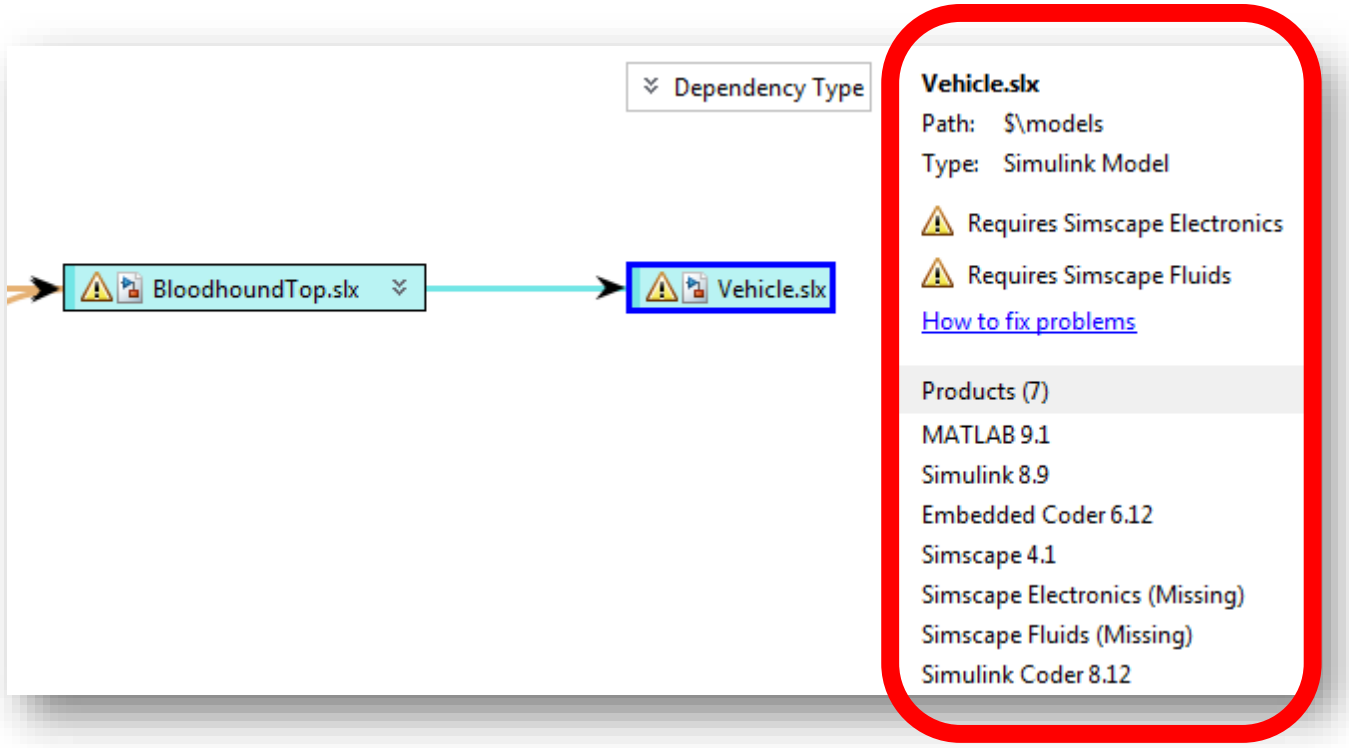
Dependency Analysis – Modular Development



Highlight Missing Products Required by a Project

Find the products needed to use a project

- Dependency analysis reports the products required by a project
- Products that are not installed shown as “(Missing)”.
- Files that use missing products show a warning icon. Click the file to see the missing products in the side panel.
- Open the model to get links to download missing products



Using labels to share and store information

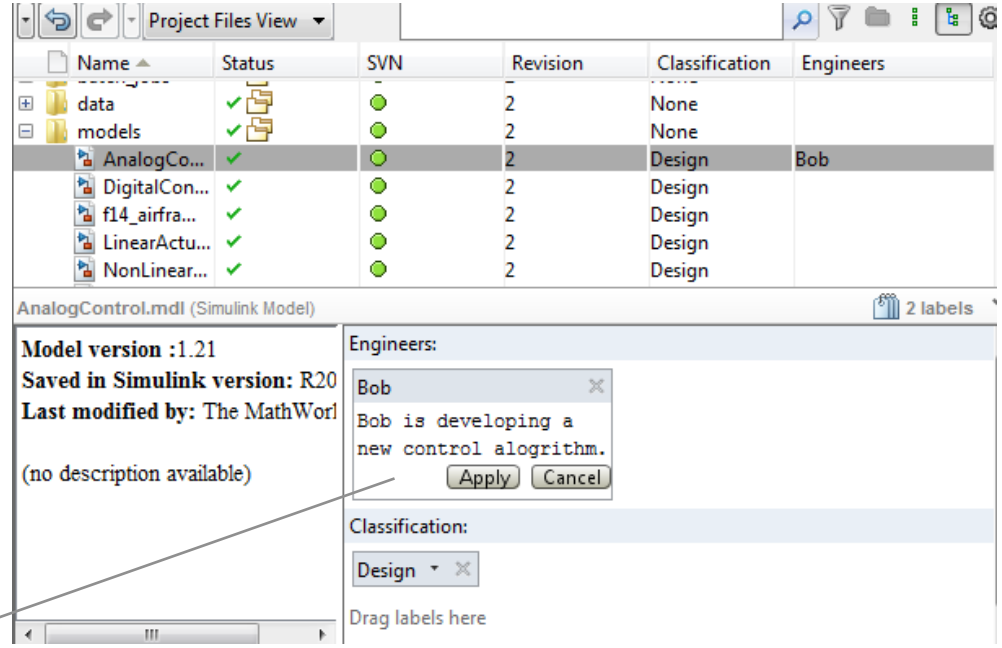
Using Labels to Add Information to the Project

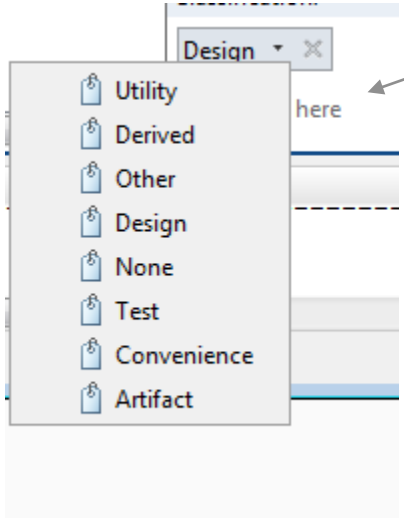
- Done lots of work to understand what the different parts are
- Wouldn’t it be nice to record that so others do not have to repeat this?
- What are labels?
- Apply some labels to the project

Simulink Project Labels

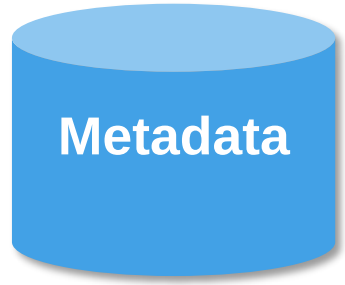
Easily add, modify and view labels attached to a file.

- Easily see and edit label data for all labels attached to a file.
- Use drag and drop to add labels.
- Easily switch between single-valued labels.





Note on Metadata

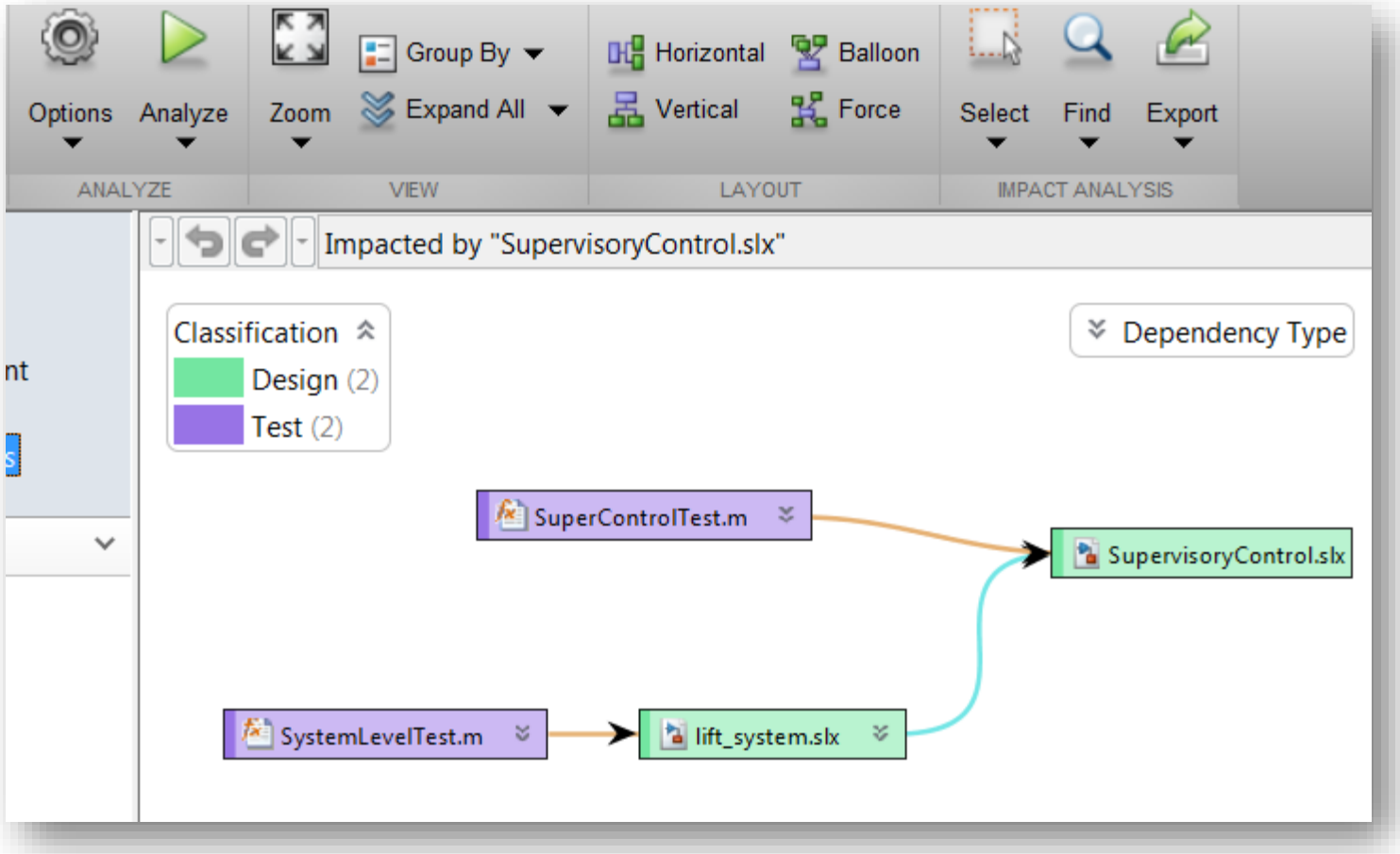


- What do we mean by metadata?
  - Wikipedia: “Data about data”
  - MathWorks: “Data about files”

- Data that is about the file, not (necessarily) part of it. For example:
  - FuelType = Gas, Diesel
  - ReleaseStatus = Research, Prototype, Production, Sunset
  - SecurityClassification = Unclassified, Protected, Restricted, Confidential
  - FileClassification = Design, Derived, Artefact
  - TestedWith = R2010b, R2011a, R2011b, ...
  - Coverage Metric = 84%

- Metadata can change without the file it relates to *having* to change.

Labels + Dependency analysis = Impact Analysis



- "What is the impact of changing the supervisory control model?"
- "What tests do I need to run to verify those changes?"
- All accessible from command-line API for full automation

More options for automation

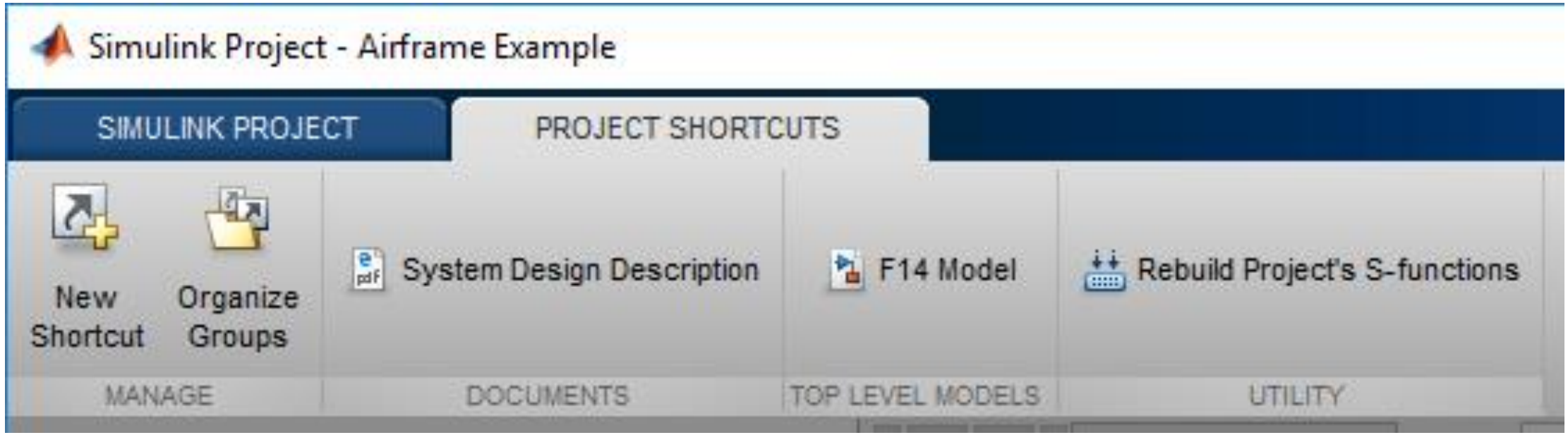
Why Automate?

- Automated Processes Get Done
  - Regularly (if needed)
  - Repeatable
  - Can be done by anyone

- Corollaries
  - Manual processes are often infrequently done
  - Can be subject to variation
  - Perhaps only one person can do them

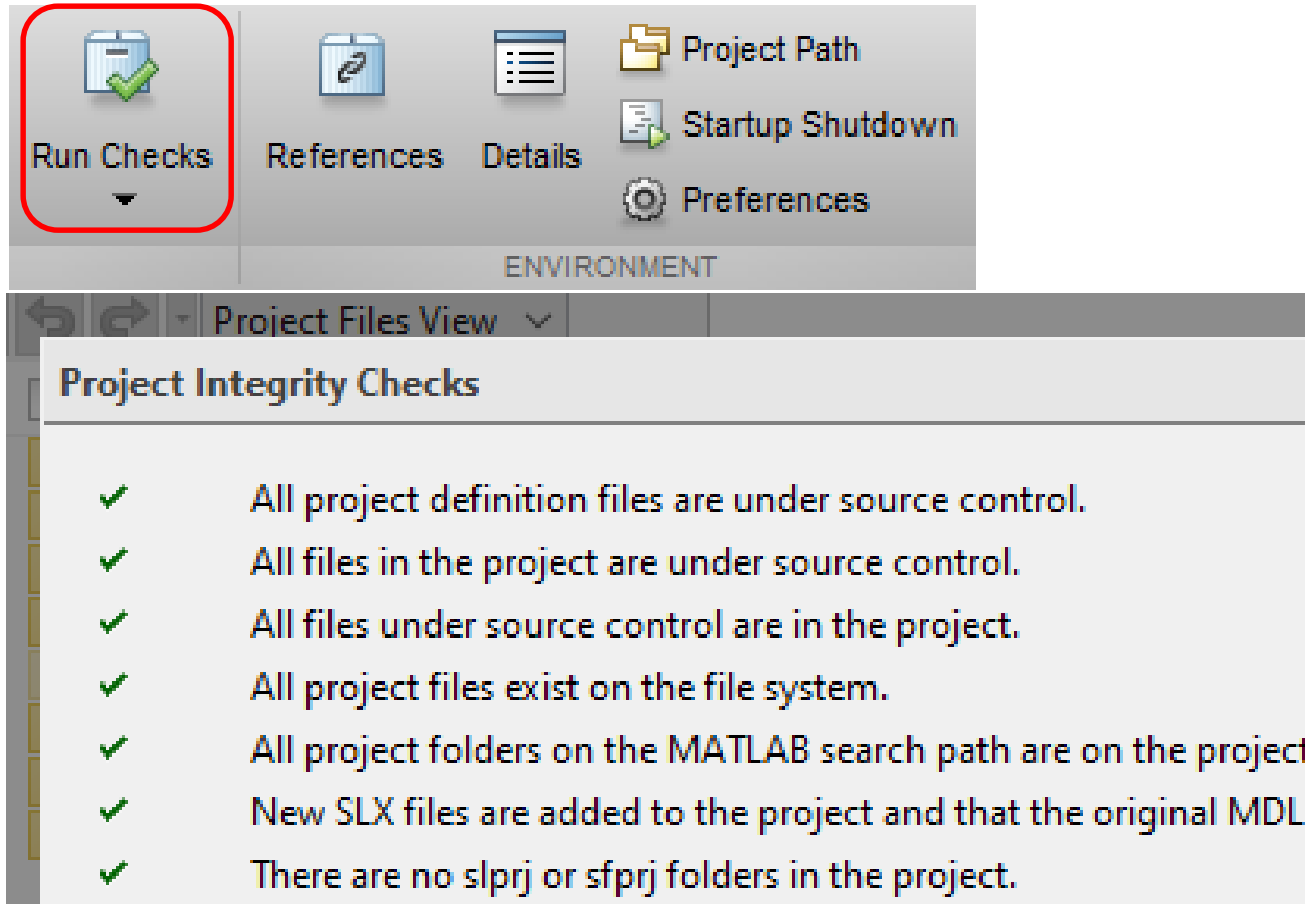
How can Automation in Simulink Project help?

- Now anyone can rebuild the S-Functions
  - (or run the tests; generate code; publish the reports; import and validate test data; …)
  - Even at 8:34pm on a Friday night; on a testing trip; …
- Groups help provide structure
  - Group by type; or by job function (project manager group; testing group)



Automation Options in Simulink Projects

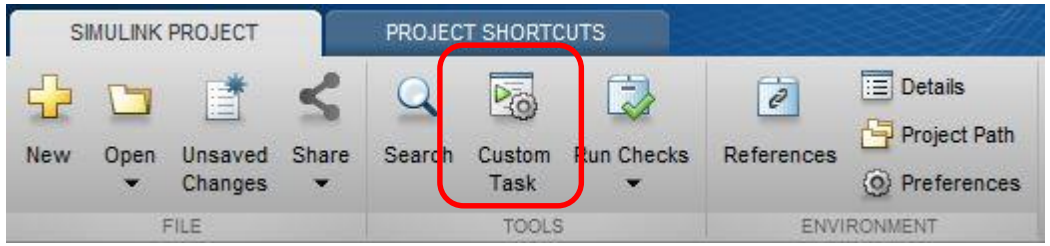
- Build-in “best practice” support
  - Project Checks
  - Growing list of our own “gotchas”

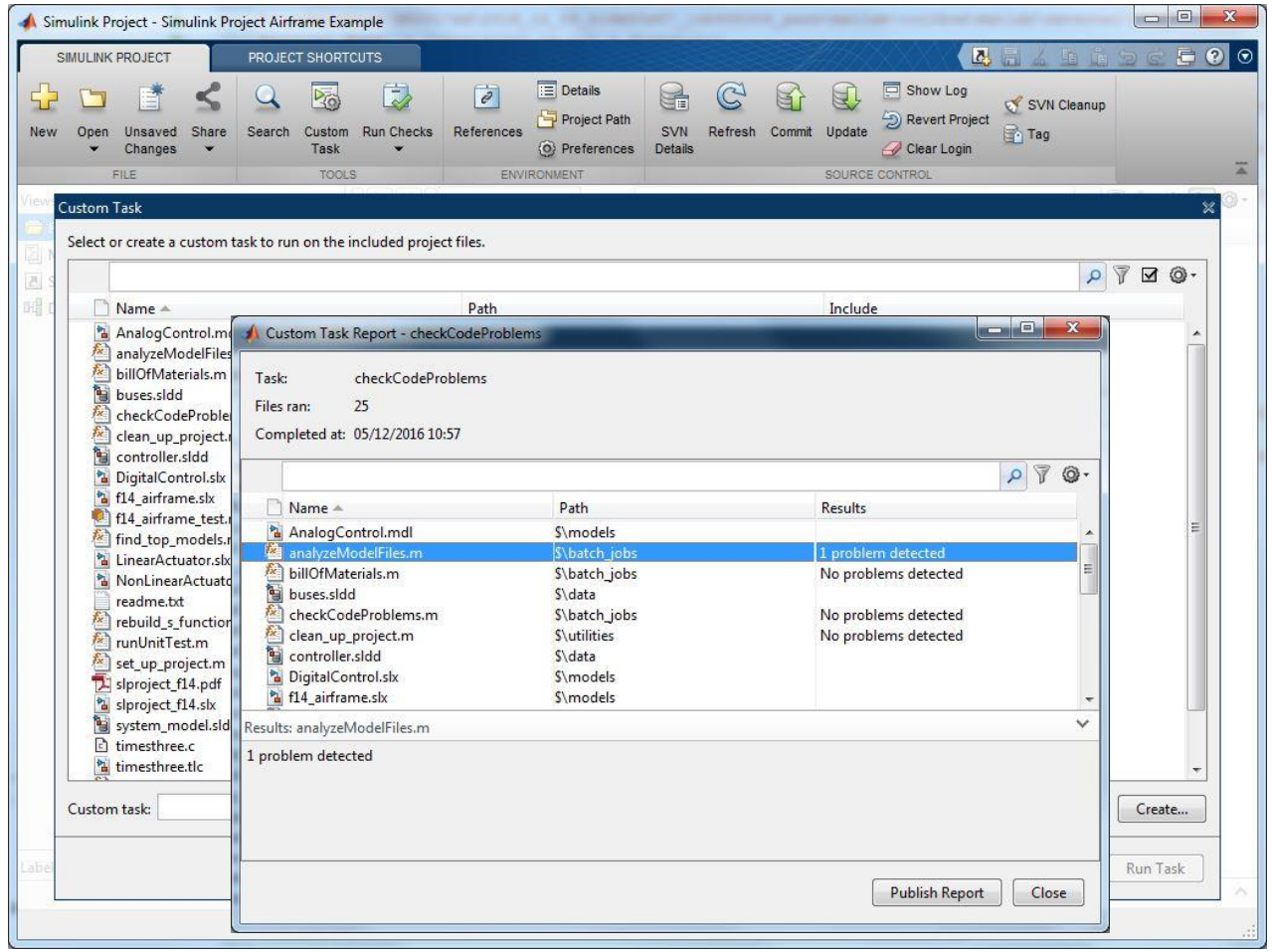


Run Custom Tasks and Create Reports

Open custom task control from the toolstrip

- Select custom functions and files more easily
- View sets of results side-by-side
- Generate reports from custom task results





- Note: Custom tasks were known as “batch jobs” in releases before R2017a

```
» sldemo_slproject_batchjobs
```

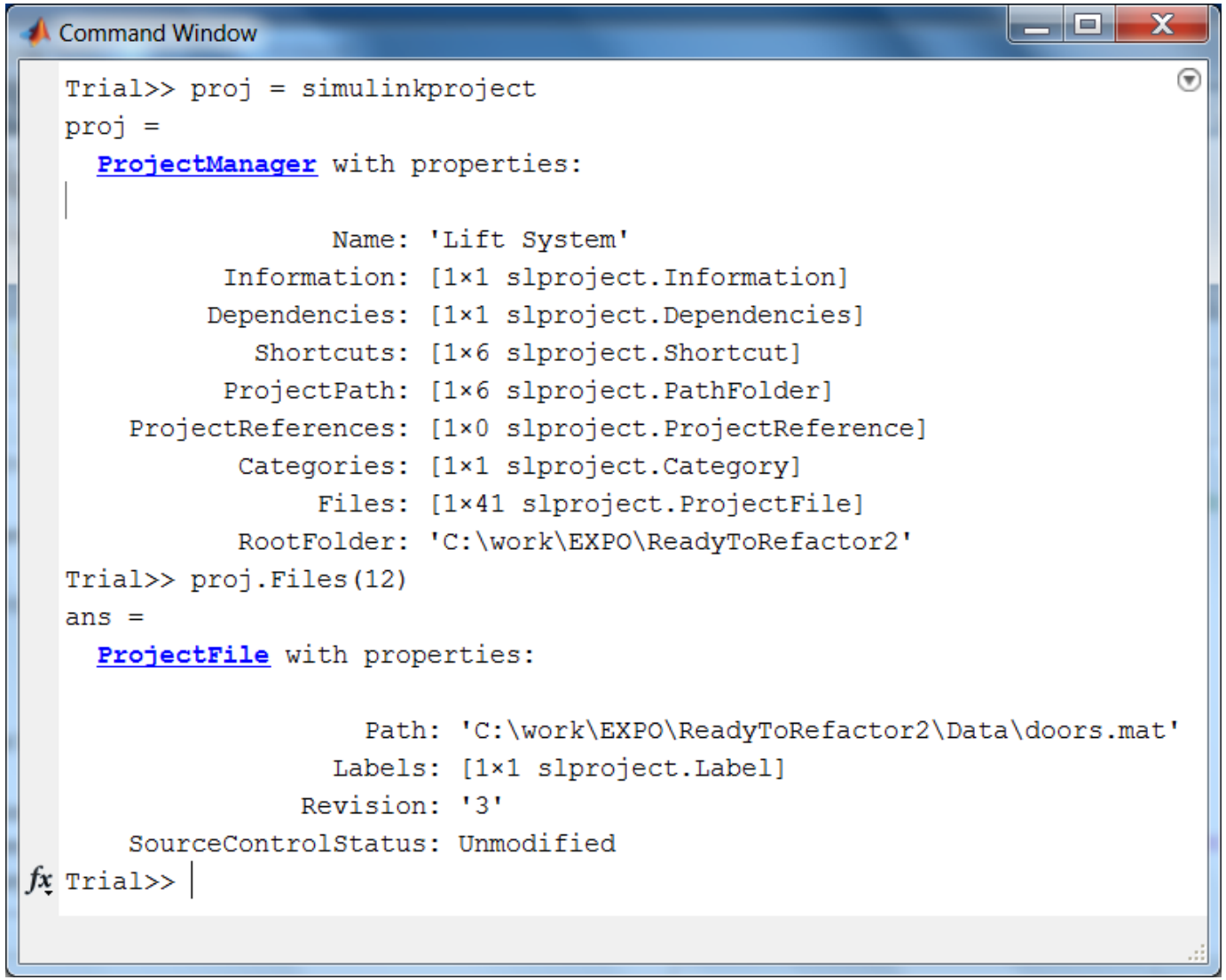
Example Custom Task

displayMATLABVersion.m

```
1   function result = displayMATLABVersion(file)
2   %displayMATLABVersion  Display the MATLAB version used to save this model
3   try
4       info = Simulink.MDLInfo(file);
5       result = info.ReleaseName;
6   catch E %#ok<NASGU>
7       result = 'Not a block diagram';
8   end
9   end
```

- Very small amount of code required
- Common patterns
  - Is this a file of type X?
  - Does this file have a label from category X with value Y?

Simulink Project API



- Easily access information for the project
- Add, remove, inspect files and labels

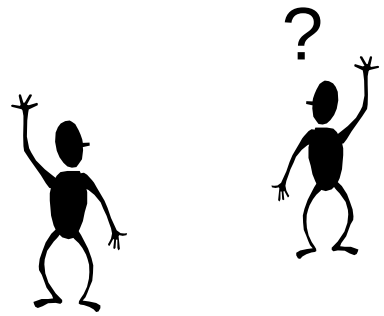
If under source control,

- See source control information for files
- Get the list of modified files

More options for sharing

Most Common Challenge in Sharing Work

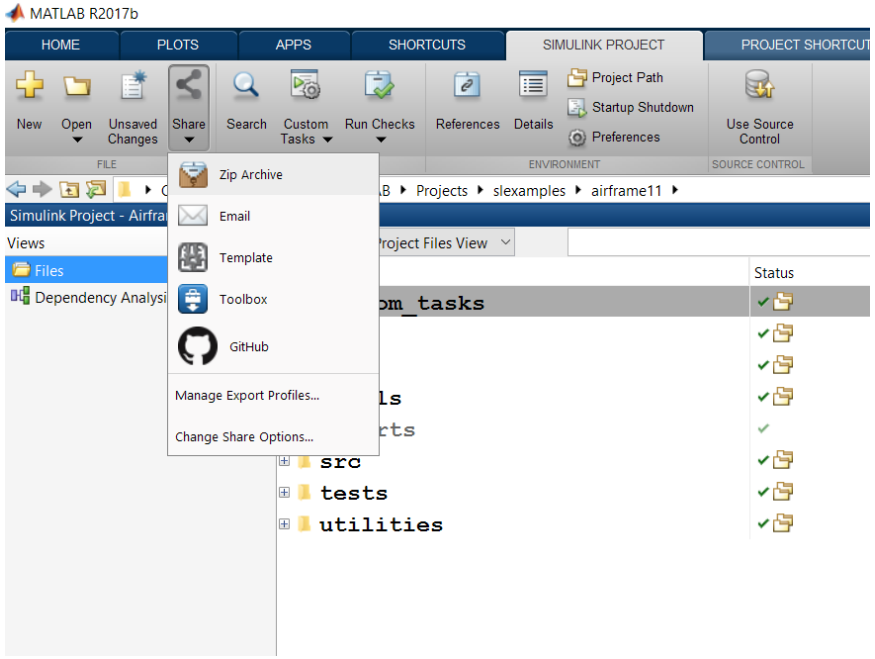
"It works on my computer, just not on yours…"



Common causes:

- Incomplete set of files
- Different environment
  - (software versions, MATLAB path, …)
- Wrong data loaded
- What do I do to get started?

Sharing work outside source control



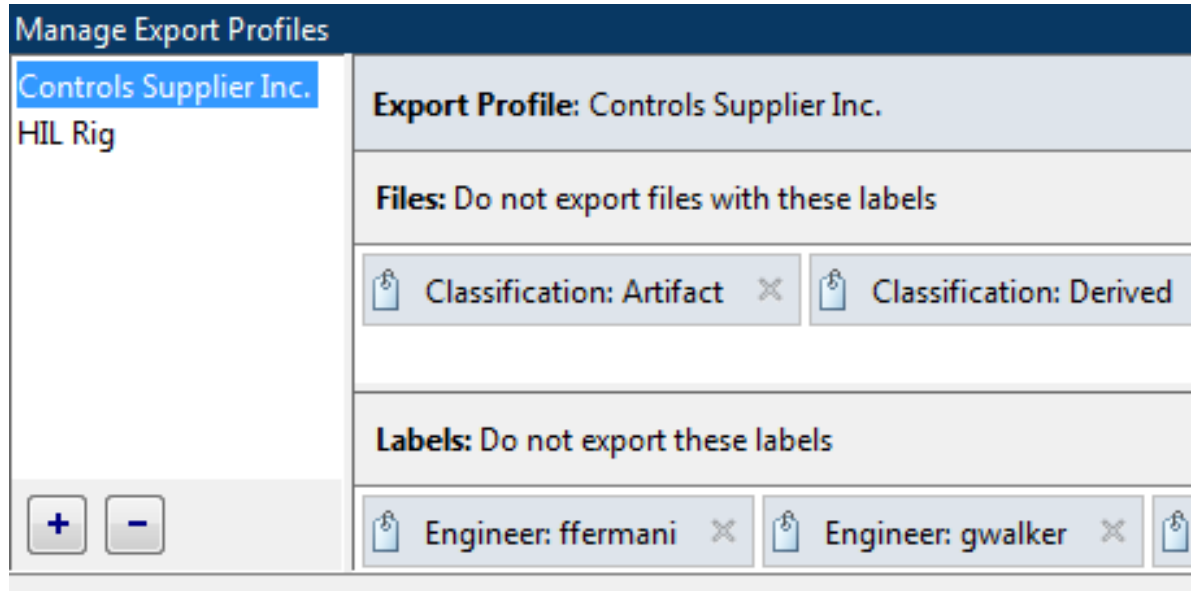
Simulink Project has built in capabilities for sharing

- GitHub
  - Collaborative sharing
  - Expect to make changes together
- Archive file
  - Fast sharing of “what I am doing now”
  - “Delivery” workflows:
    - Send a package of work
    - Work independently
    - Receive a package of work back

How much to share?

- Typically do not want to share all my project with a supplier or customer
- Reduce to the minimum to
  - Avoid sharing IP I want to keep in-house
  - Keep it simple

- Create “Export Profiles” to manage which files are exported from project
  - Uses project labels to set up exclusion rules
  - Set up many profiles for different workflows
    - Sharing to supplier (share only what is needed)
    - Share to customer (shield my IP)
    - Share to HIL rig (no tests, doc, requirements)
    - Etc.

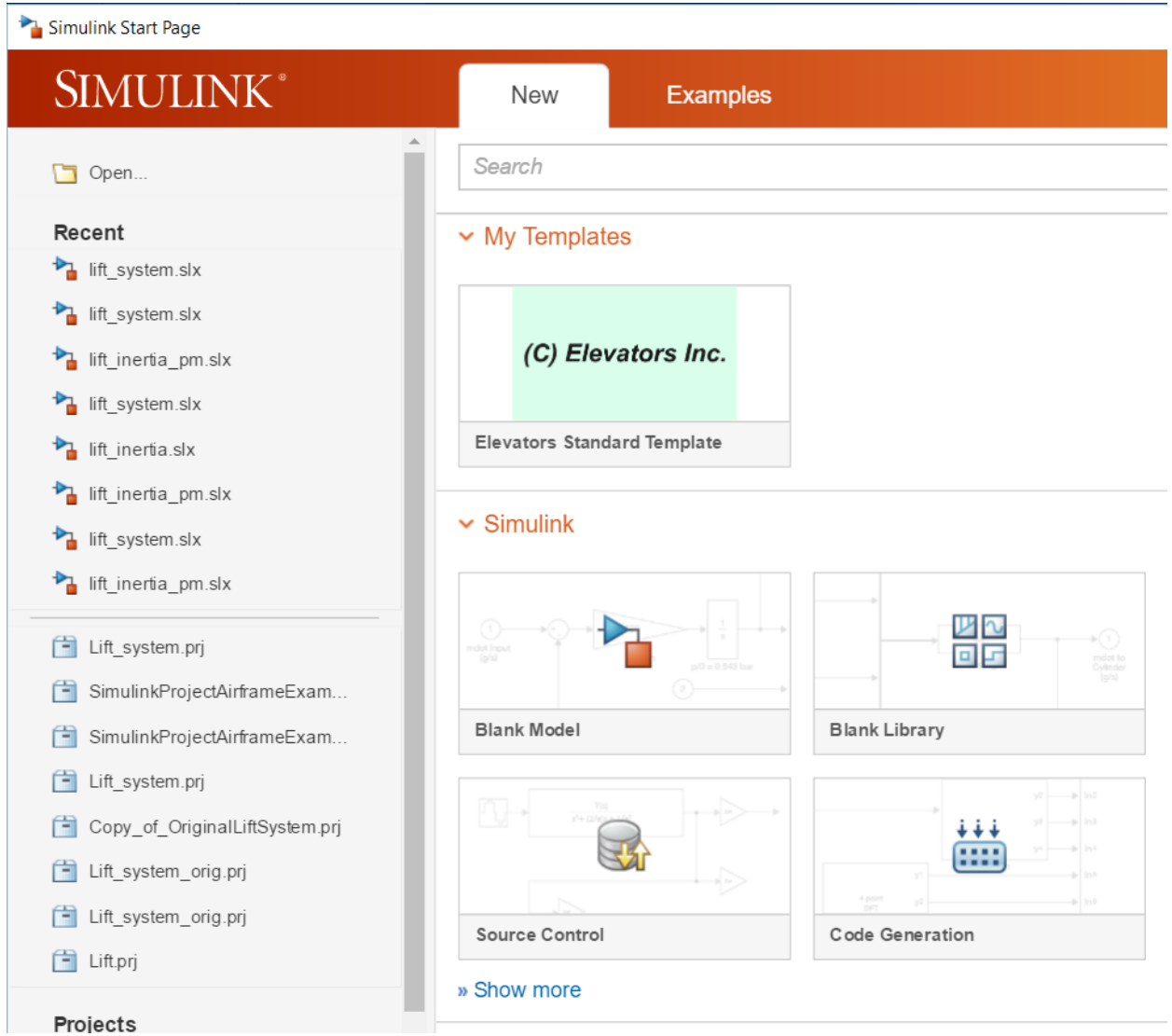


Knowledge transfer

Model Templates

Build models using design patterns that serve as starting points to solve common problems

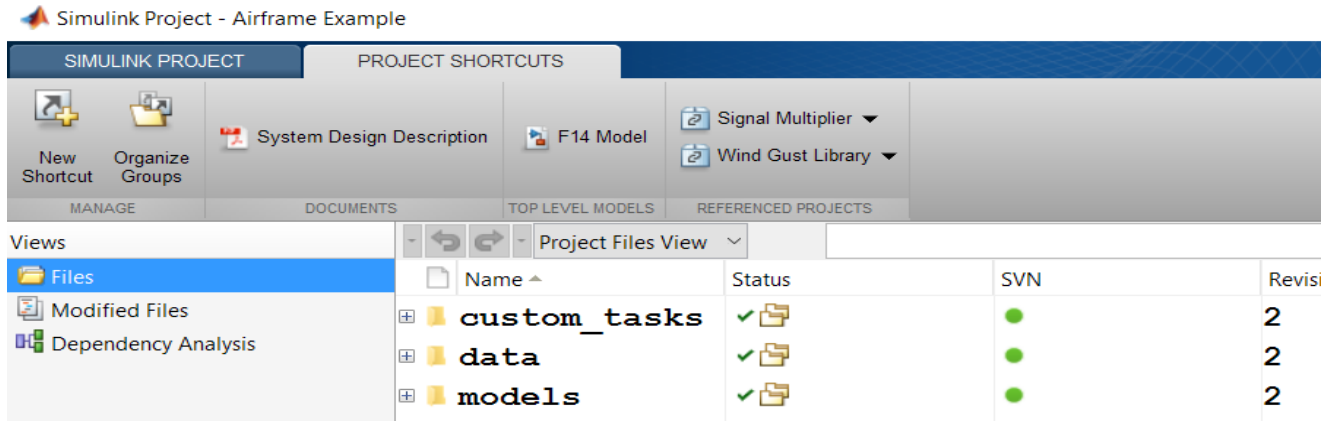
- Use shipped templates to get started with building models or create custom templates to from a Simulink model
  - Avoid problem of corrupting original file when creating a new model
- Avoid repetitive tasks when starting out to build a new model
- Enforce a standard process for building models for the entire team or organization

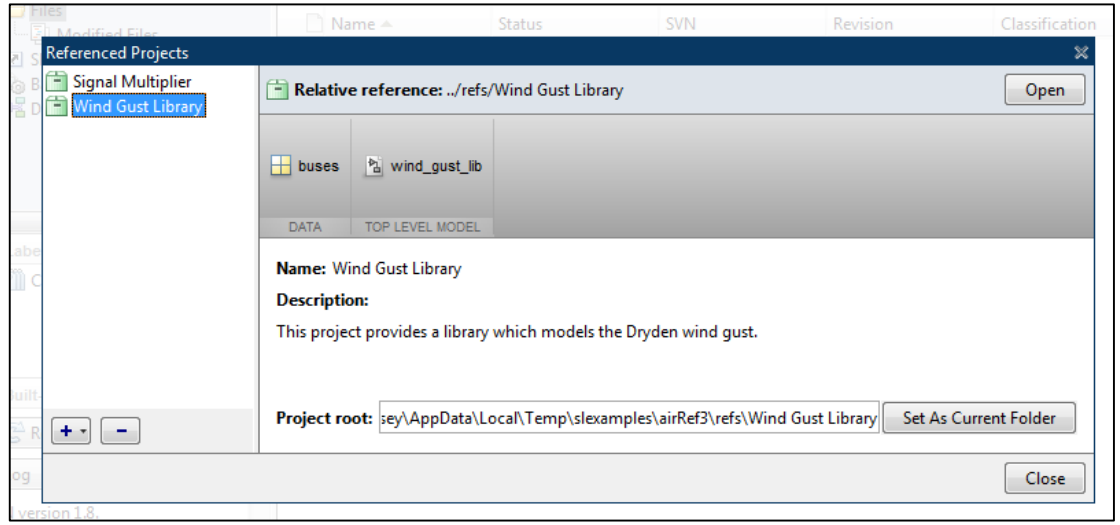


Projects can reference other projects

Componentize large modelling projects

- Develop reusable components using projects
- Flexible referencing:
  - Relative
  - Absolute
- Extract folders to referenced projects
- Deep hierarchies are supported



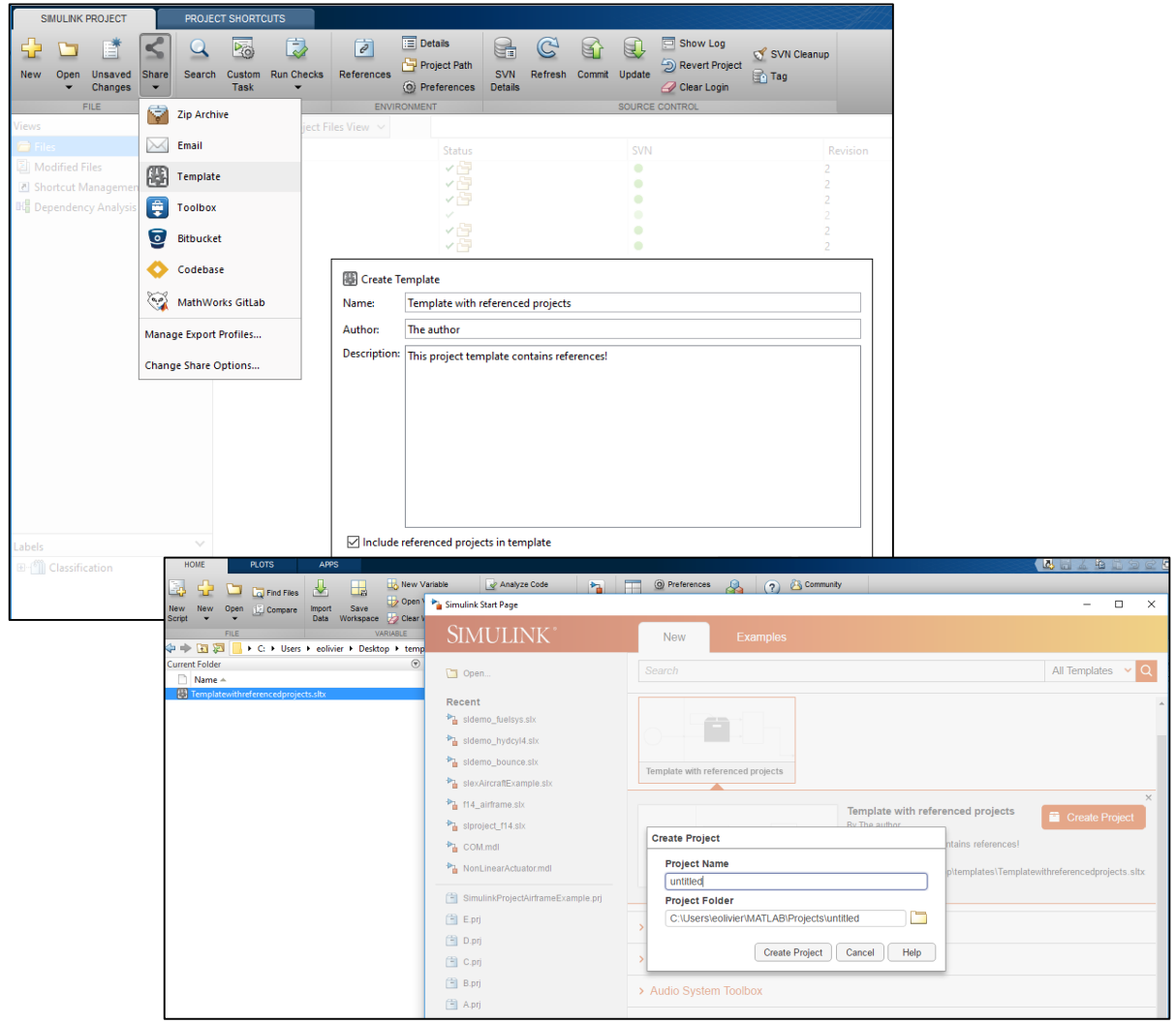



```
» sldemo_slproject_airframe_references
```

Include References in Templates for Sharing and Reuse

Template with references

- Start from a project with references
- Create a template including the references
- Save it on the MATLAB path or double click it to see it in the start page
- Create a new project based on the template



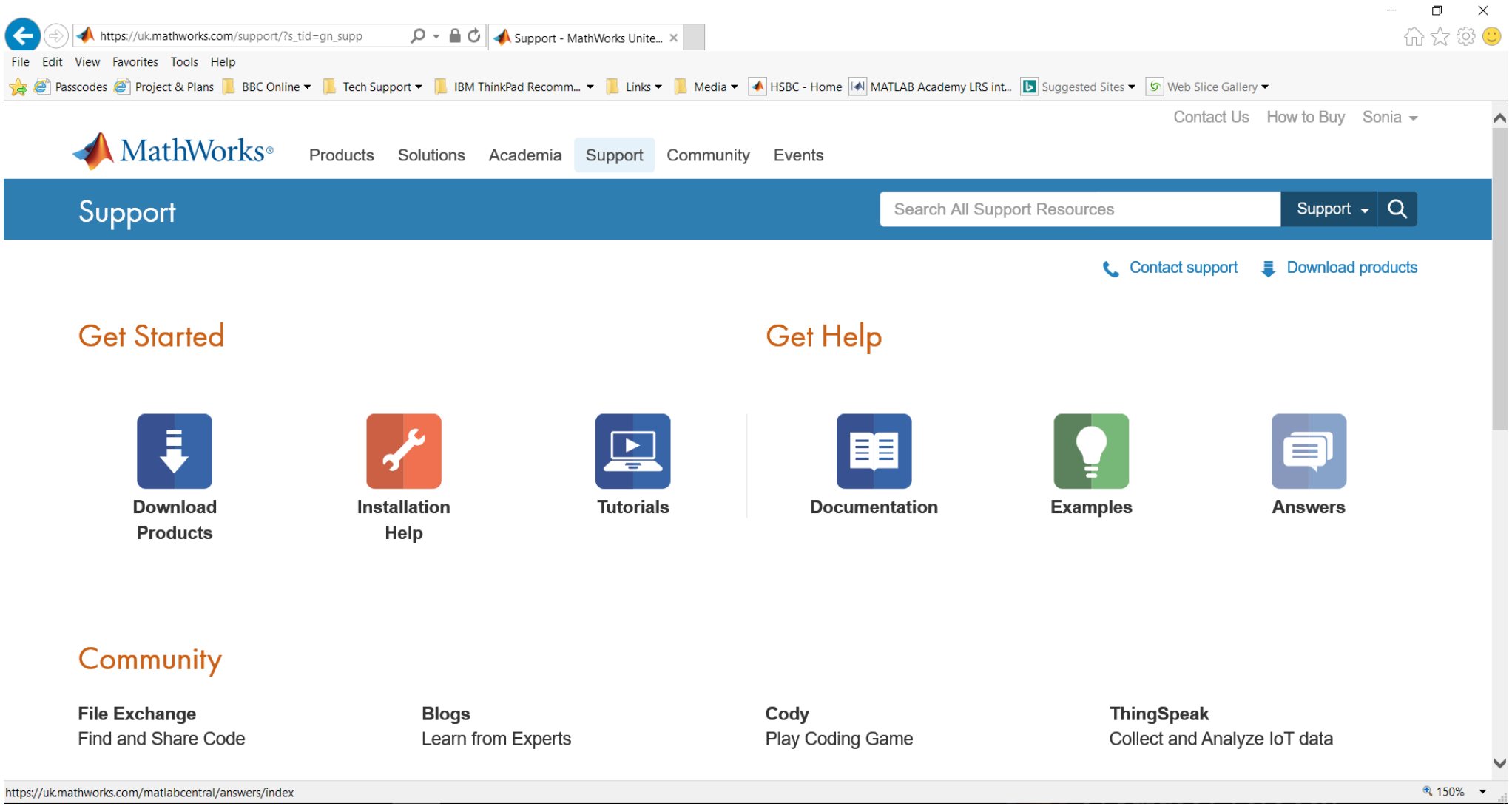
Summary

- Common challenges addressed!
  - Structured/ Common Environment
  - Graphical Dependency Analysis
  - Source Control Integration
  - Automation of common tasks
  - Options for sharing work
  - Parallel development workflows
  - Knowledge retention
- Simulink Projects for efficient team collaboration workflows
- Try it Today!

Additional Resources

- Documentation
  - Project Management
- Example
  - Using a Simulink Project
- Tutorials
  - Try Simulink Project Tools with the Airframe Project
  - Create a New Project to Manage Existing Files
- Training
  - Simulink Model Management and Architecture
- Consulting
  - Proven Solutions from MathWorks Consulting Services

Support


https://uk.mathworks.com/support/?s_tid=gn_supp
Support - MathWorks Unite...
File Edit View Favorites Tools Help
Passcodes Project & Plans BBC Online Tech Support IBM ThinkPad Recomm... Links Media HSBC - Home MATLAB Academy LRS int... Suggested Sites Web Slice Gallery
Contact Us How to Buy Sonia
MathWorks
Products Solutions Academia Support Community Events
Support
Search All Support Resources
Support
Contact support
Download products
Get Started
Download Products
Installation Help
Tutorials
Get Help
Documentation
Examples
Answers
Community
File Exchange
Find and Share Code
Blogs
Learn from Experts
Cody
Play Coding Game
ThingSpeak
Collect and Analyze IoT data
https://uk.mathworks.com/matlabcentral/answers/index
150%

Q & A