Continuous Modeling with MATLAB and Microsoft Azure DevOps

Wojciech Halabis
Test and Validation Engineer
1. Introduction
2. Motivation
3. Solution
4. Microsoft Azure DevOps
5. Use Cases
6. Conclusion
Introduction about myself

I am 33 years old.

Degree in Electronics (B.Eng.) and Engineering Physics (M.Sc.).

Professional software development since 2013.

Current position: Test Automation in IoT (Architectural design, Test specification and implementation, continuous testing…).

Working at CLAAS since August 2018.

5+ years experience in modular software design using “git flow”.
CLAAS Company presentation

1. Introduction
CLAAS E SYSTEMS Company presentation

1. Introduction

Connectivity & Data Management

Steering, Precision Applications & Terminals

Process Automation

Base Components & Architecture
1. Introduction

2. Motivation

3. Solution

4. Microsoft Azure DevOps

5. Use Cases

6. Conclusion
Who may be interested in this presentation

Are your models complex?

Are there 10+ people working on the same model?

Are those people working at different locations, even countries?

Do you feel that your teams develop several times similar stuff, e.g. libraries?

Have you experienced problems during integration of MATLAB Simulink models?

Do you want to adopt agile methodologies?
Best practices for modular mode design

Use model reference for unit-level models.

Pick a strategy for grouping units into features.

Classify models according to safety requirements.

Avoid algorithmic content at the integration level.

Use model metrics to monitor unit complexity.

Define the role of model architects and integrators.
1. Introduction
2. Motivation
3. **Solution**
4. Microsoft Azure DevOps
5. Use Cases
6. Conclusion
Taking the approach of other industries

These problems are not new, there are a lot of engineers that have been struggling for years with them.

So... why don’t we use some of their approaches?

Do you know who I’m talking about?

SOFTWARE ENGINEERING
1. Introduction

2. Motivation

3. Solution

4. Microsoft Azure DevOps

5. Use Cases

6. Conclusion
Covering the entire application lifecycle with Microsoft Azure DevOps

It provides version control.

It provides reporting and requirements management.

It provides project management, for both agile software development and waterfall teams.

It provides automated build, testing and release management.
Snapshots – Home page

Welcome to this project!

Our Goal
The main purpose of this wiki is to enable a seamless transition into version control for MATLAB models. All the information related to this topic will be found here. you don't need to search anywhere else! We hope you get enough information, transparency and if not, you can tell us or much better, feel free to appoint new entries to this wiki

Filter pages by title

- Introduction to git
- How To's:
  - Installation of dSpace
  - Git process
  - TFS Admins page
  - Gitignore explained
  - Installation of git
  - Git using Matlab
  - Git using Git Extensions
  - Git using Visual Studio

Meeting notes

MATLAB_Simulink_Styleguide
Snapshots – Git repositories

- Files in the repository
- "Commits" or history of the model repository
- Responsible of the commit
Snapshots – Knowledge base

Enable Source Control in MATLAB

Open MATLAB -> Preferences, go to MATLAB -> General -> Source Control select "Enable MathWorks source Control Integration"
Putting your models in version control system

Encourage collaboration.

Maintain properly our versions.

Restore previous versions.

Understand all the time what is happening.

ALL OF THAT JUST BY INTEGRATING PROPERLY A VERSION CONTROL SYSTEM IN YOUR COMPANY
Faster development with appropriate git tool

As a command line tool

With a git GUI

Integrated
Developing models using the “git flow”

The “master branch” is a permanent branch which always reflects a production-ready state. Only “merges” or “pull requests” are allowed according to the git-flow.

Major changes should be implemented in separated feature branches.

Minor changes should be implemented in develop branch.
Traceability out of the box

REQ: a function that calculates max of two values

Automatic creation of feature-branches for the model, important for almost automated release notes creation
1. Introduction
2. Motivation
3. Solution
4. Microsoft Azure DevOps
5. Use Cases
6. Conclusion
Example of modular design with git submodules

“Welding” for your interfaces

“Nuts and bolts” for your interfaces
5. Use Cases

Externalization and reuse of model units

OEM

Tyre supplier company

Supplier company “A”

Engine supplier company

git repository

OEM – Department “A”

OEM – Department “B”

OEM
Development efforts based on safety requirements of model units

5. Use Cases

git repository

Safety critical model unit

Non safety critical model unit
1. Introduction
2. Motivation
3. Solution
4. Microsoft Azure DevOps
5. Use Cases
6. Conclusion
Where are we now at CLAAS

Different departments are already working using this methodology.

Growing number of machine Simulink models are being built on top of git submodules.

Departments that have developed a modelling process without git, are being trained and will adopt this methodologies in the near future.
References

[8] https://thenounproject.com/victorulerz
[12] https://thenounproject.com/IconTrack/
[14] https://thenounproject.com/Flatart
[16] https://thenounproject.com/AlfredoCreates
[17] https://thenounproject.com/coquet_adrien
Thank you for your attention.

More information: ces-frontoffice@claas.com

wojciech.halabis@claas.com