A Journey with Model Based Design

Per Hagen Nielsen :: Plant Controls :: Electrical BoP & Hybrid:: Vestas Wind Systems A/S
We employ more than 24,400 people worldwide and have more than 35 years of experience with wind energy.

We have a total of 41,693 combined turbines under service, or more than 82 GW.

We have a total of 66,093 turbines or more than 97 GW of installed wind power capacity in 79 countries worldwide spanning six continents.

Vestas’ revenue for Q3 2018 was EUR 2,811 m.
With **wind** as the core energy source, Vestas’ hybrid power plant solutions leverage from the combination of wind power generation with either **solar PV** generation or **electrical storage**, or both of them.

Our department offers Power Plant Control software, hardware and models matching global grid code requirements: ActivePower, ReActivePower, Frequency, Power factor, Inertia, Voltage, FastRunBack control etc.
1. Several complex interaction control systems
2. Virtual synchronous machine requirements

More RE than 100% consumption
Power Plant Engineering

- Site Selection & Land Access
- Development
- Permits
- Engineering
- Construction
- Operations & Maintenance

Electrical models

Power Plant Controller

- WTGs
- Statcoms
- MSC / MSR
- Diesel Backup
- Energy Storage
- PV

Performance & Diagnostic
Pre-challenges before the Model Base Design approach. The Team in 2014 was approx 10 Engineers

- Power Engineers doing paper design documents
- Main design test platform was PSCAD (power system analyze tool)
- Design component wise, no fully true model in design framework

Waste of time & opportunities

- Software design/implementation from design documents
- Integration to full system discovered need for redesign iteration
- Handled both system software & application layer
- C++, Structured Text.

We needed one model one team
Prestudy for ModelBasedDesign using Simulink with C++ code generation

One man-month work

**Prestudy Scope**

- A representing selected control feature ported from existing code and legacy HW platform

**System Software integration**

- Investigate and choose layer split
- Represent system SW in Simulink: Comm layers, Sampling, Realtime system etc.
- New VxWorks & Embedded Intel Celeron platform

**Code review**

- Build trust in generated C++ code
- Examine "code generation report"
- Link between code and model in an easy way

**Site Test**

- Check performance
- Check debug possibility
Decision making – What do the management say yes to

TIP: Address everything but use only a few hours and move ahead… You won’t be more accurate anyway and you won’t regret

Governance
• Roles
• Responsible
• Stakeholder accept

Economy
• Training
• Toolboxes
• Maintenance
• Mathworx consults
• 1. Iteration overhead

Training
• Matlab
• Simulink
• Stateflow
• Architecture
• CodeGen
• GIT
• Merge
• Processes

Processes
• Feature Dev.
• Test
• Merge
• MAAB
• …

Externals
• Mathworks
• Consulting office ex 14days

Risc
• What can go wrong

Ambition
• Test
• Testmanager
• Testharness
• TimeToMarket
• Quality
• Scaling

...
TEAM ORGANIZATION CHANGES

Software engineers & Power engineers became **Control Engineers**
Challenges along the first 2 years

- Merge tool and GIT for concurrent development issues was underestimated.
- Team size increased up to 30 worldwide (Denmark, Sweden, Porto, Ukraine, Singapore)
- New way of working inefficiency under estimated (you’ll get it back later)
- Simulation and build speed exhausted on the platform (+20min compile & simulate)

Upgraded development platform 2 man-month + 14days Mathworks consulting

- Upgrade 2017b tools
- New internal development platform
  - Refactored layer design, build structure, dependency issues
  - Dataviewer usage
  - Accelerated executions, extended scripting
  - MathWorks Automotive Advisory Board (MAAB) rules introduction
From 2017 Jenkins build integration server
Test and Validation

• Usage of Mathworks ‘*Simulink Test*’ was not considered in the start, validation of design was following legacy procedure in Vestas eg. Lab test with generated code on real hardware and partly matlab test.

• In 2018 *Simulink Test* in form of Testmanager and TestSequences was introduced.
  • Every legacy written cases are baselined, so a “curve” of all signals outcome where recorded and checked when developer delivered to the GIT master stream with a low default tolerance. An easy quick check but not very intelligent so time was used on analyzing faulty faults. Also the setup is very sensitive to interface changes.
  • A more intelligent assessment for each case is necessary but takes time (*..a lot*), but how mission critical are your code ..... And front load your work, write assessment a dev time
  • All Tests runs at night on Jenkins integrations servers.
  • MAAB checks, Warning Error check, Test check, are reported through
Wrapping up

• Identify your current situation
  • Where are your product development?
  • Where would you like to be?

• Involve stakeholders
  • PreStudy
  • Get commitments
  • Governance & Economy
  • Don’t sell ModelBasedDesign on short term, takes years

• Use special attention to you continues integration workflow
Thanks for your interest

Q&A