Nonlinear Model-Based Adaptive Robust Controller Design for Hydraulic Winch in Oil and Gas Wireline Operation:
Model based design with docker container

Fanping Bu

Nov. 16, 2021

Digital Technology
Innovate. Engage. Deliver

SLIIC

SCHLUMBERGER
INDUSTRIAL INTERNET
CENTER

Outline

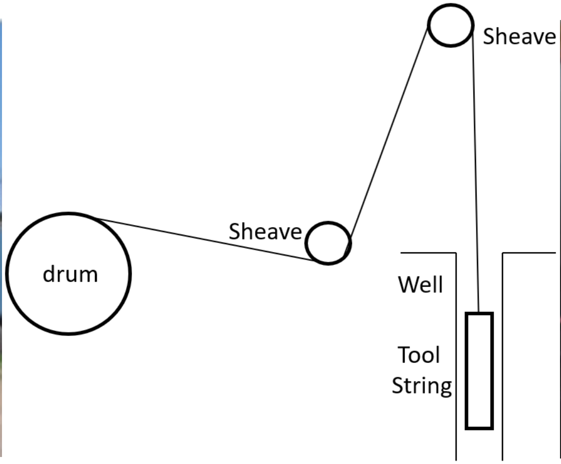
Digital Technology
Innovate. Engage. Deliver

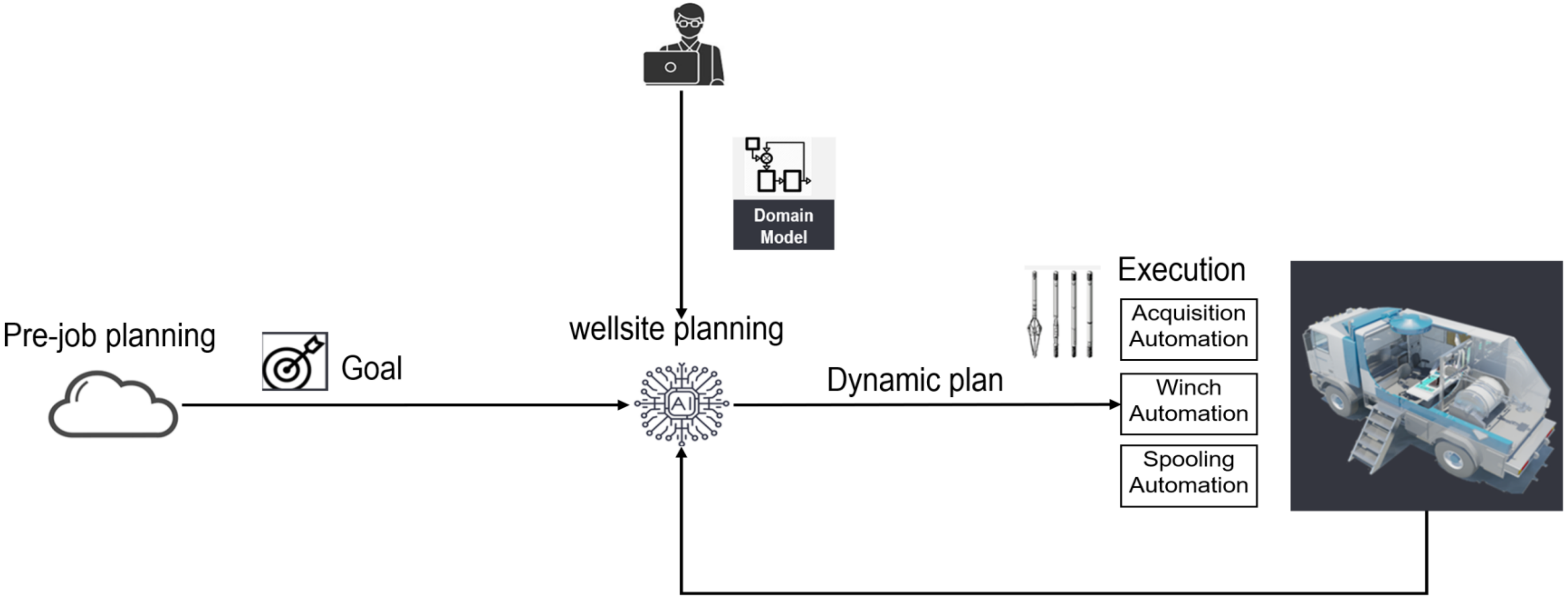
- ❑ Project background
- ❑ Model based design and docker
- ❑ Winch control hardware setup
- ❑ Winch system model development
- ❑ Winch controller design
- ❑ Controller software architecture, implementation and testing
- ❑ Conclusion

Schlumberger

Autonomous wireline operation

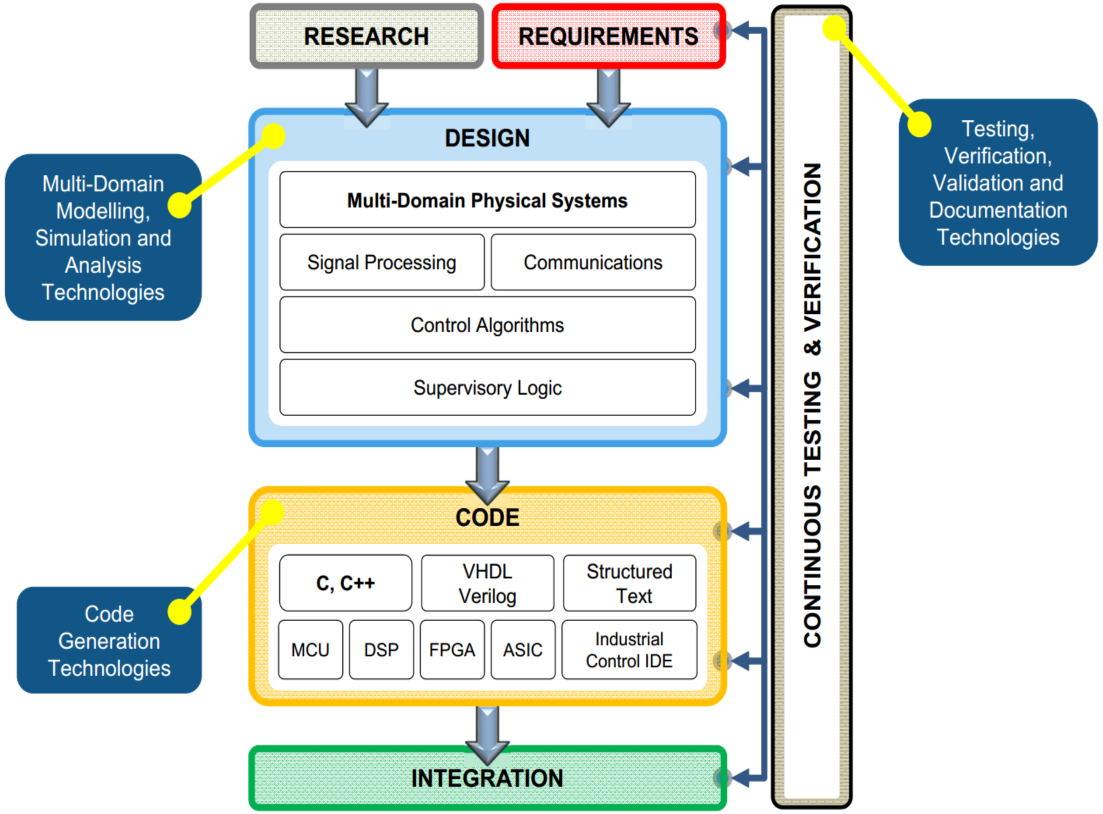
Digital Technology
Innovate. Engage. Deliver

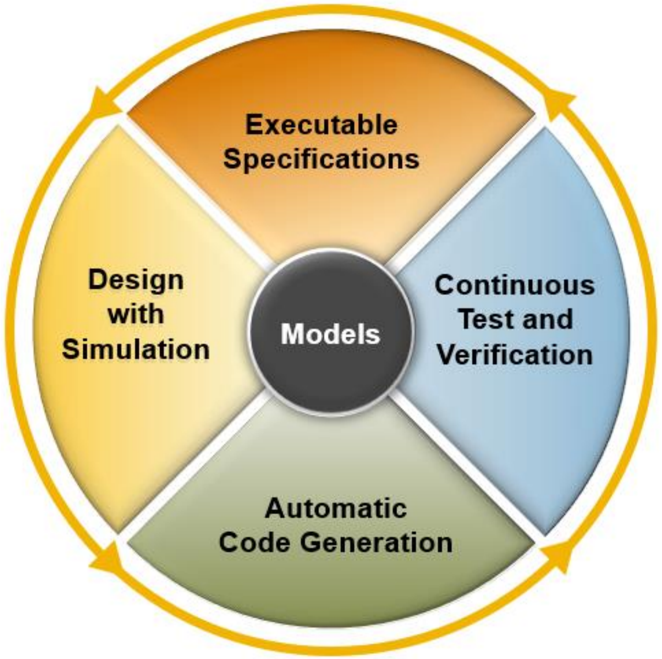


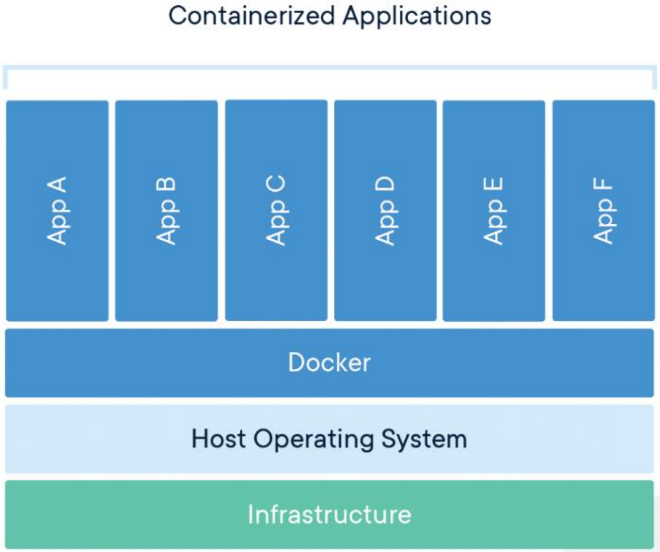


Model based design and docker container

Digital Technology
Innovate. Engage. Deliver



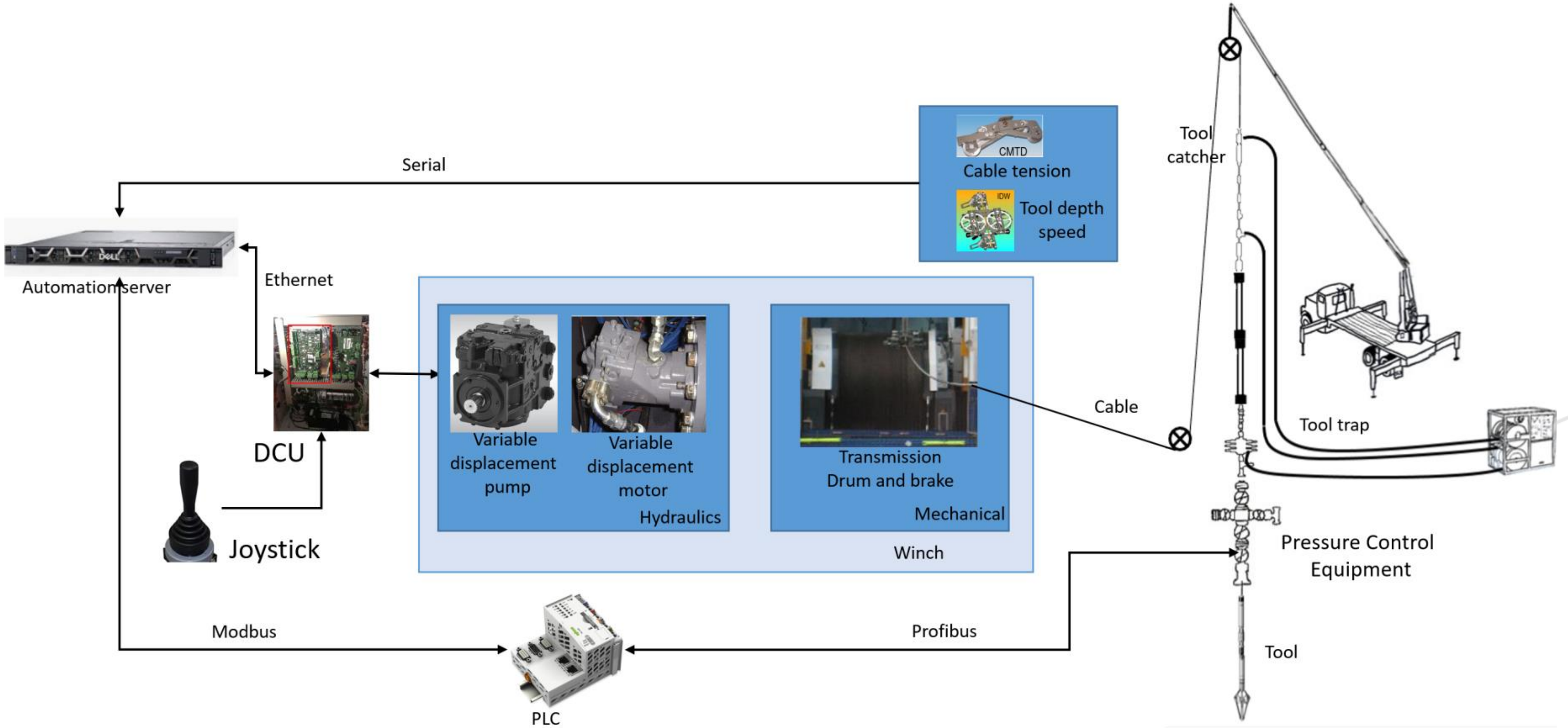




Slides from Mathworks presentation

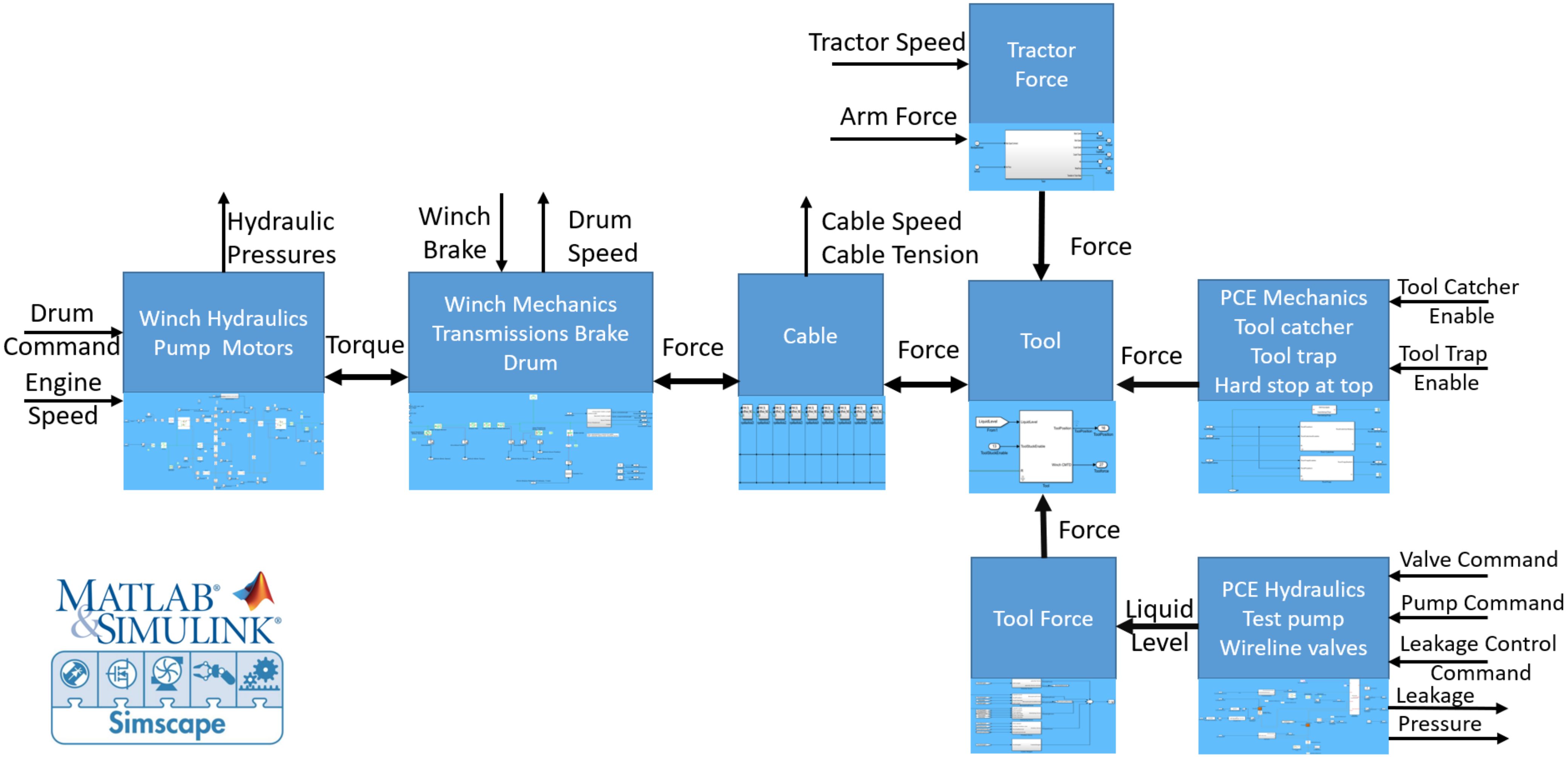
Schlumberger

Current hardware setup for winch control



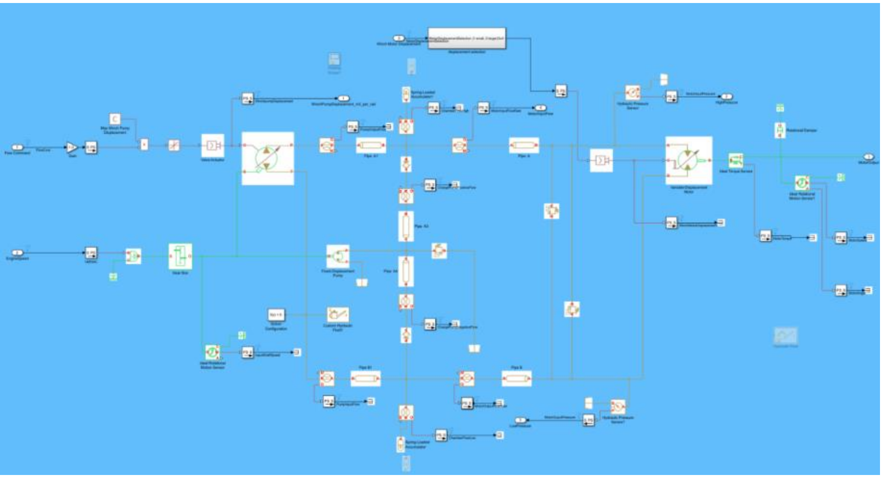
Schlumberger

Equipment model and simulator: overview

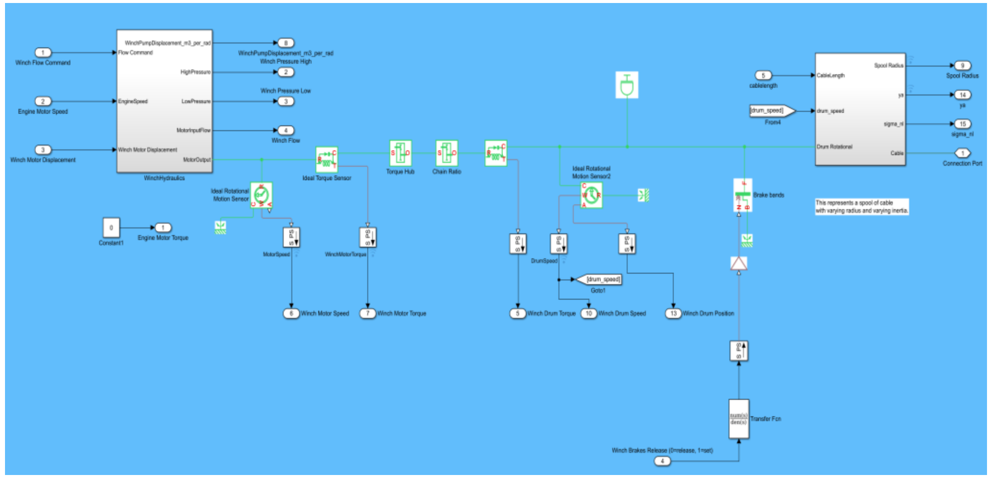


Equipment model and simulator

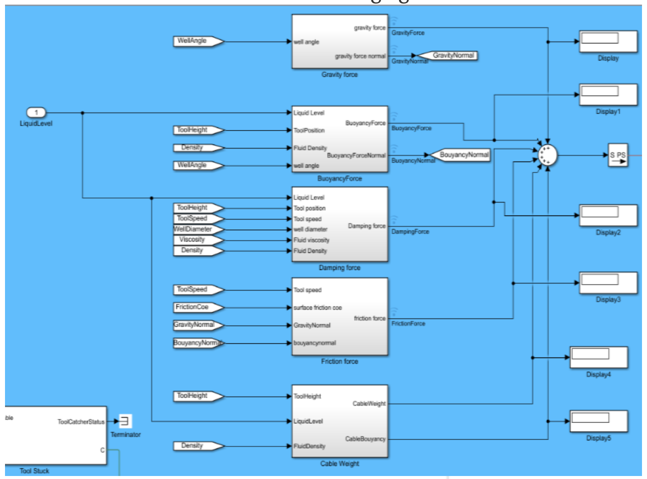
Digital Technology
Innovate. Engage. Deliver



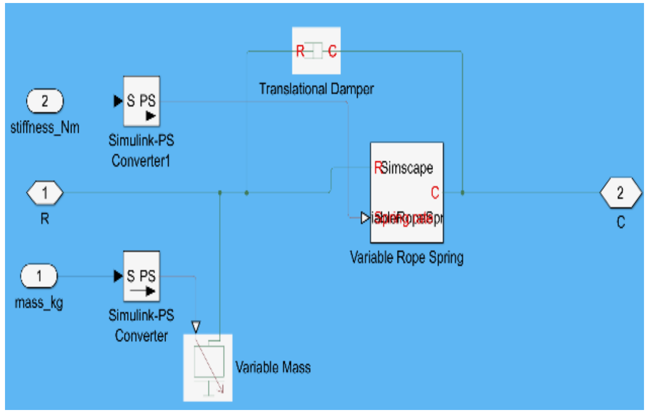
Hydraulics



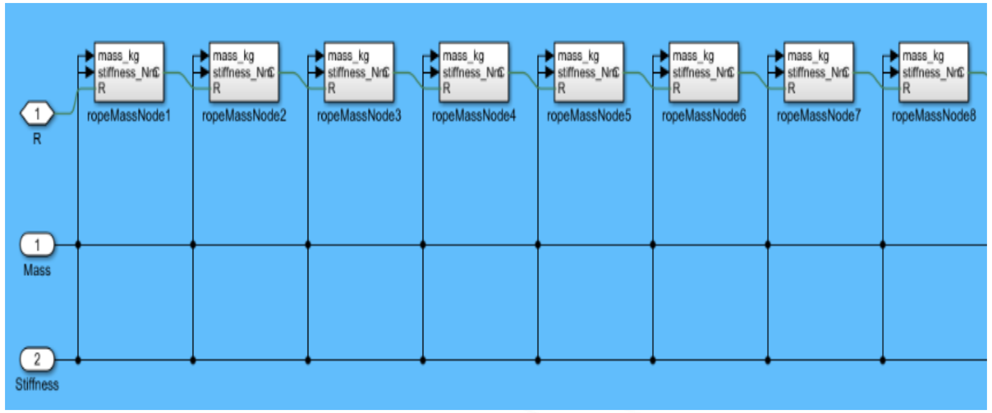
Transmission and drum



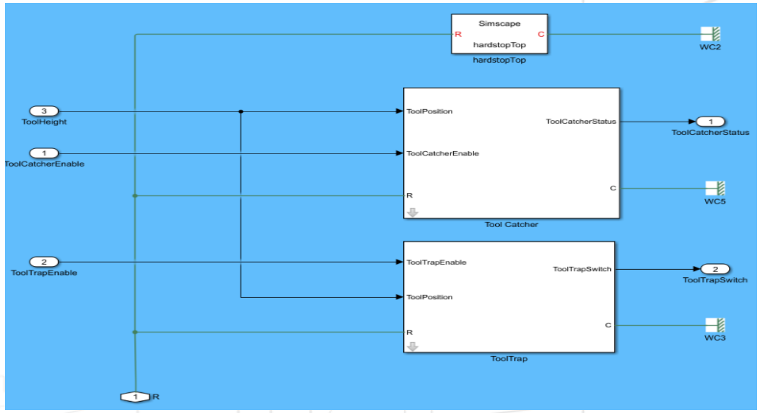
Tool force



Cable node



Cable

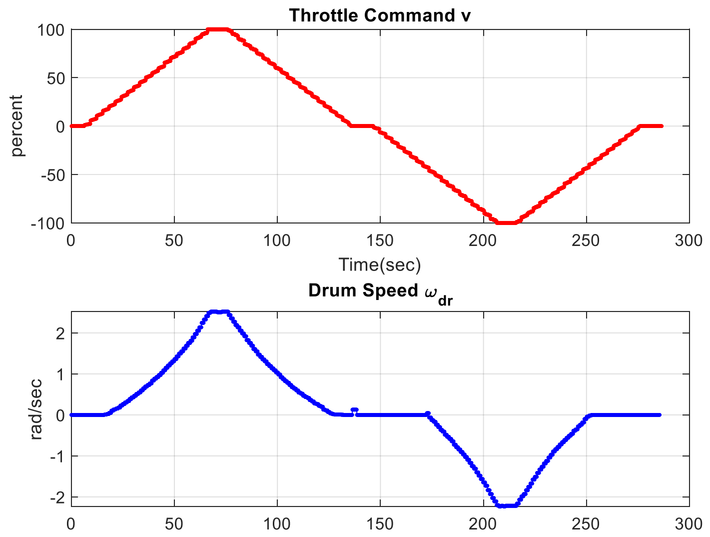


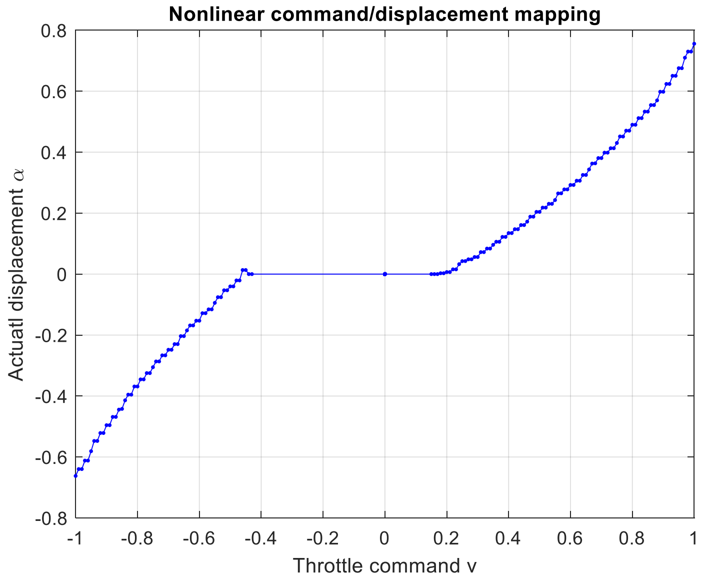
Tool Trap and Catcher

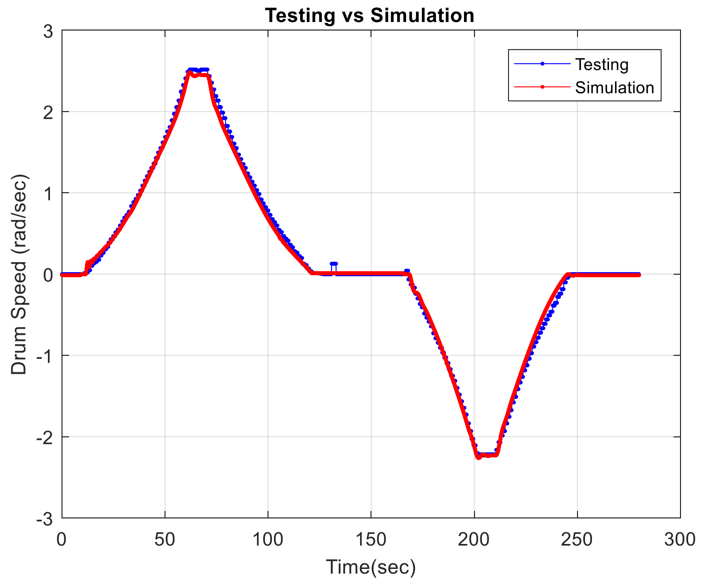
Schlumberger-Private

Schlumberger

Model calibrations

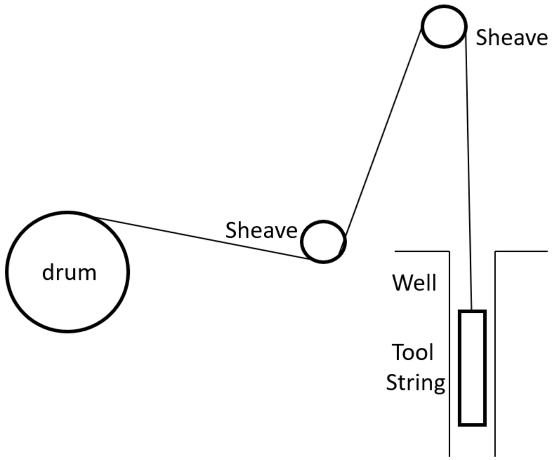


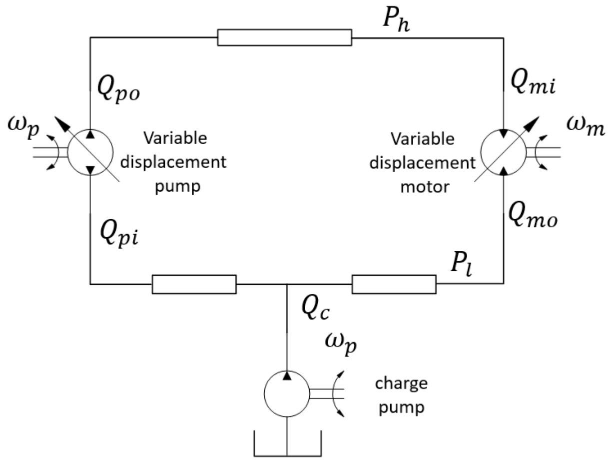


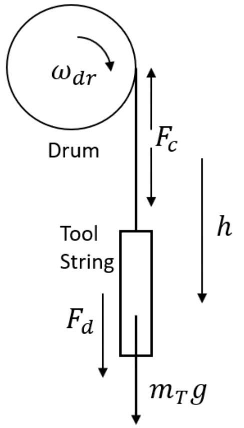


Winch controller design: design system model

Digital Technology
Innovate. Engage. Deliver







Hydraulic Flow

Variable displacement pump

$$Q_{pi} = D_p \omega_p \quad D_p = G_p \alpha \quad \alpha = f(v)$$

$$v \in [-1,1] \quad \alpha \in [-1,1]$$

$$Q_{po} = \eta_{vp} Q_{pi}$$

Variable displacement motor

$$Q_{mo} = D_m \omega_m$$

$$Q_{mi} = Q_{mo}/\eta_{vm}$$

$$T_m = D_m (P_h - P_l)\eta_{tm}$$

Charge pump

$$Q_c = D_c \omega_p$$

Pressure dynamics

$$\frac{V_h}{\beta_e}\dot{P}_h = Q_{po} - Q_{mi} - Q_{lh}$$

$$\frac{V_l}{\beta_e}\dot{P}_l = Q_{mo} + Q_c - Q_{pi} - Q_{ll}$$

Motor torque

$$T_m = D_m (P_h - P_l)\eta_{tm}$$

Winch Mechanics

Drum

$$\omega_r = \omega_m / n$$

$$T_r = n T_m$$

$$(J_r + J_c)\dot{\omega}_r = T_r - T_L + T_d$$

$$T_L = F_c R$$

Tool

$$(\rho h + m_T)\ddot{h} = (\rho h + m_T) g - F_c + F_d$$

Design model in state space

load pressure $P_L = P_h - P_l$

$x = [x_1, x_2, x_3]^T = [h, \dot{h}, P_L]^T$,

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \frac{A_m}{m_s} x_3 + \frac{(\rho x_1 + m_T)}{m_s} g + \theta_1 + \tilde{F}_d$$

$$\dot{x}_3 = \frac{\beta_e}{V_T}(C_p \omega_p \alpha - C_m x_2 - D_c \omega_p) + \theta_2 + \tilde{Q}_L$$

$$\alpha = f(v)$$

Schlumberger

Winch controller design: ARC controller design using backstepping

Digital Technology
Innovate. Engage. Deliver

Step 1: A desired load pressure α_2 is designed for the system load pressure x_3 such that the tool motion x_1 will follow the desired motion trajectory x_{1d}.

$$\alpha_2 = \alpha_{2a} + \alpha_{2s} \quad \alpha_{2s} = \alpha_{2s1} + \alpha_{2s2}$$

$$\alpha_{2a} = \frac{m_s}{A_m}\left(-\frac{\rho x_1 + m_T}{m_s} g - \hat{\theta}_1 + \dot{x}_{2eq}\right)$$

$$\alpha_{2s1} = -\frac{m_s}{A_m} k_2 z_2$$

Step 2: Synthesize a control law α_3 for the actual pump displacement α such that the load pressure x_3 will track the virtual control function α_2 designed in the first step.

$$\alpha_3 = \alpha_{3a} + \alpha_{3s} \quad \alpha_{3s} = \alpha_{3s1} + \alpha_{3s2}$$

$$\alpha_{3a} = \frac{1}{C_p \omega_p}\left[C_m x_2 + D_c \omega_p + \frac{V_T}{\beta_3}\left(-\hat{\theta}_2 + \dot{\alpha}_{2c} - \frac{\omega_2}{\omega_3}\frac{A_m}{m_s} z_2\right)\right]$$

$$\alpha_{3s1} = -\frac{1}{C_p \omega_p}\frac{V_T}{\beta_3} k_3 z_3$$

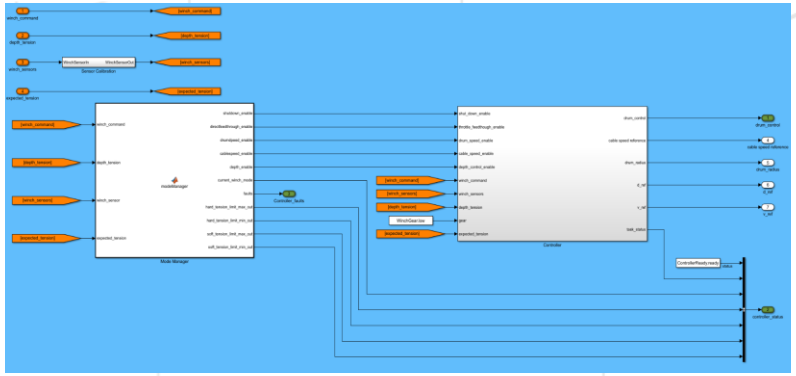
Step 3: The actual control command v can be calculated from inversed nonlinear mapping $v = f^{-1}(\alpha_3)$

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \frac{A_m}{m_s} x_3 + \frac{(\rho x_1 + m_T)}{m_s} g + \theta_1 + \tilde{F}_d$$

$$\dot{x}_3 = \frac{\beta_e}{V_T}(C_p \omega_p \alpha - C_m x_2 - D_c \omega_p) + \theta_2 + \tilde{Q}_L$$

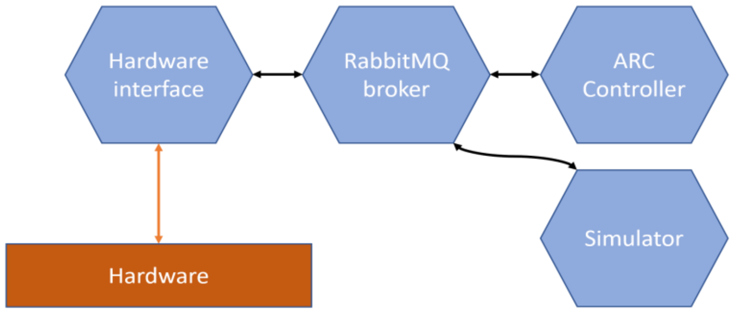
$$\alpha = f(v)$$

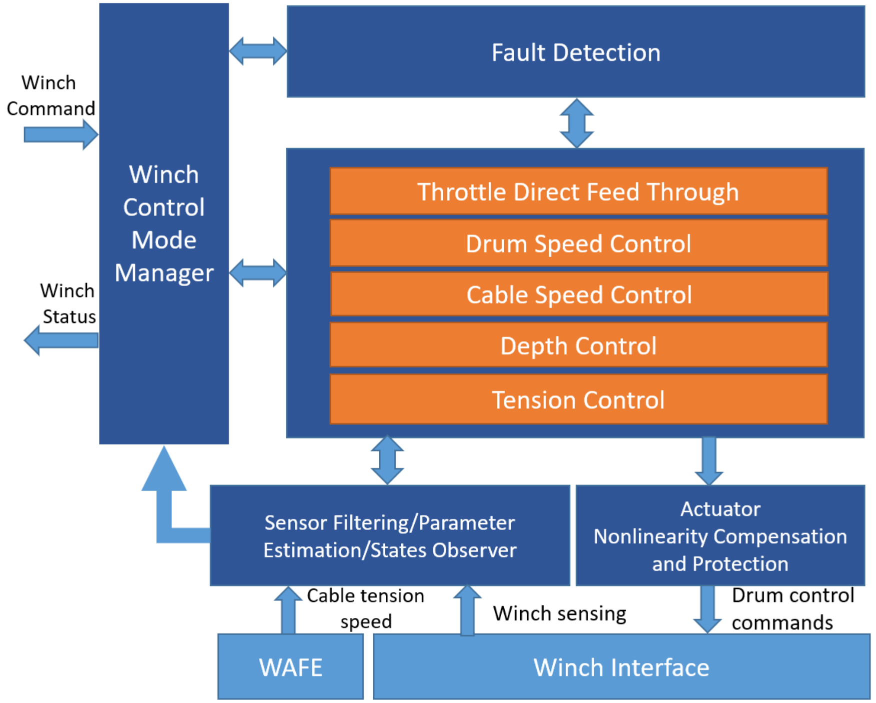


Schlumberger

Winch controller implementation

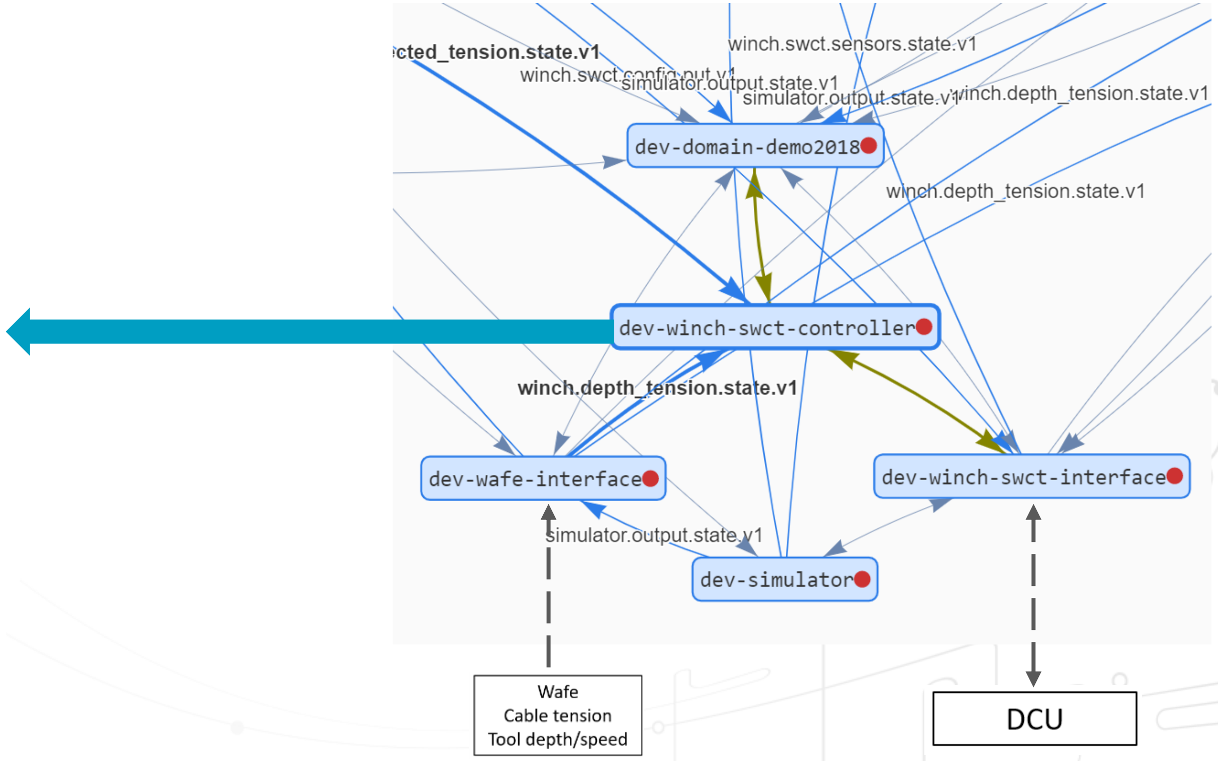
Digital Technology
Innovate. Engage. Deliver





RabbitMQ

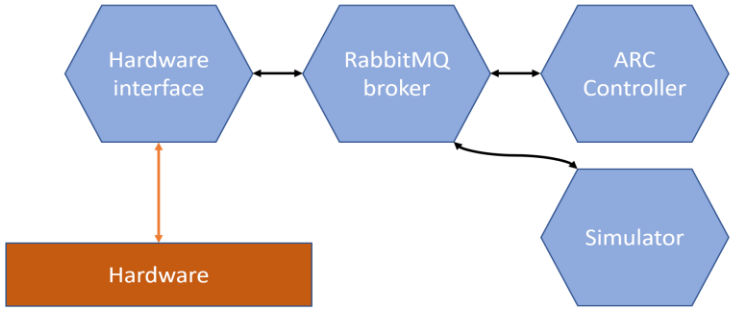
docker



Schlumberger

Build container services from Simulink/SimScape

Digital Technology
Innovate. Engage. Deliver



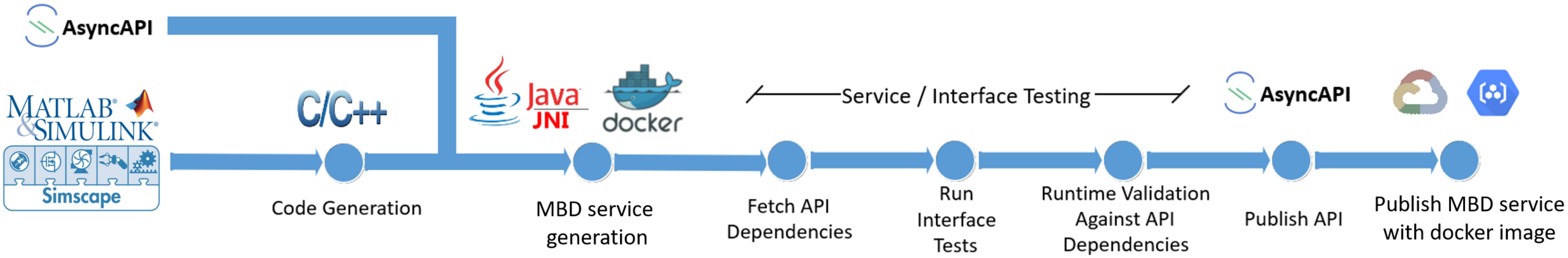
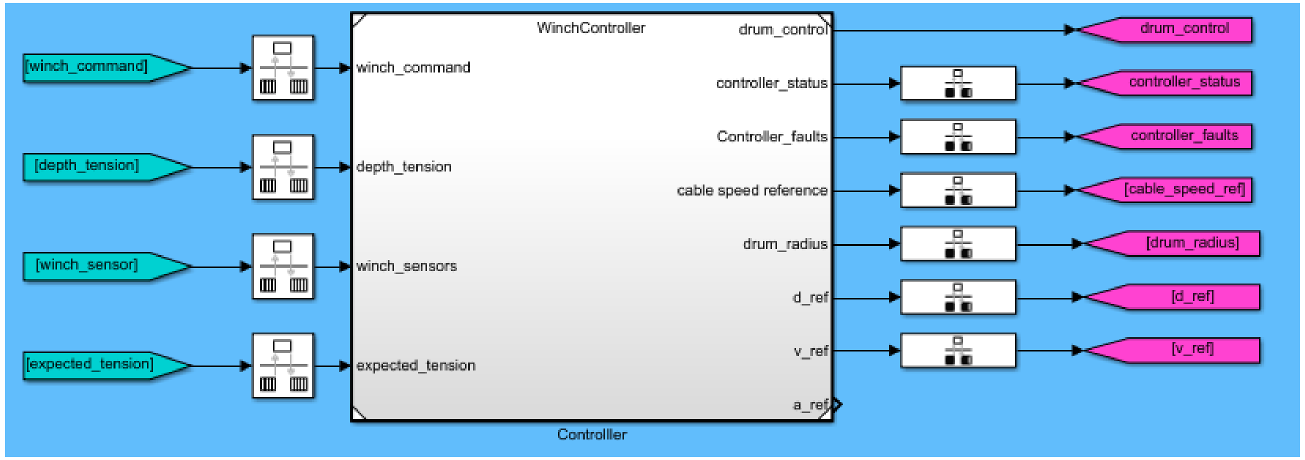


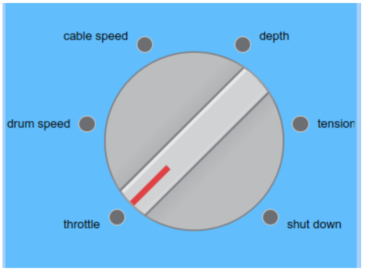
Figure courtesy of Ken Ditlefsen's poster

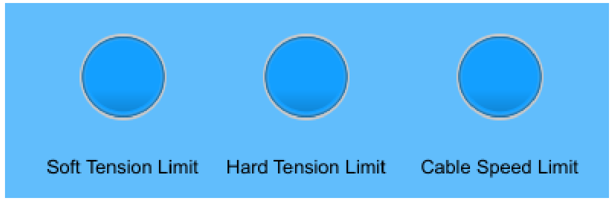
Schlumberger

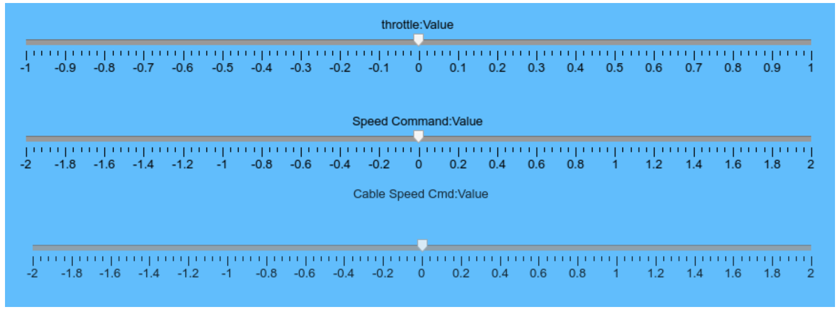
Model-in-loop testing

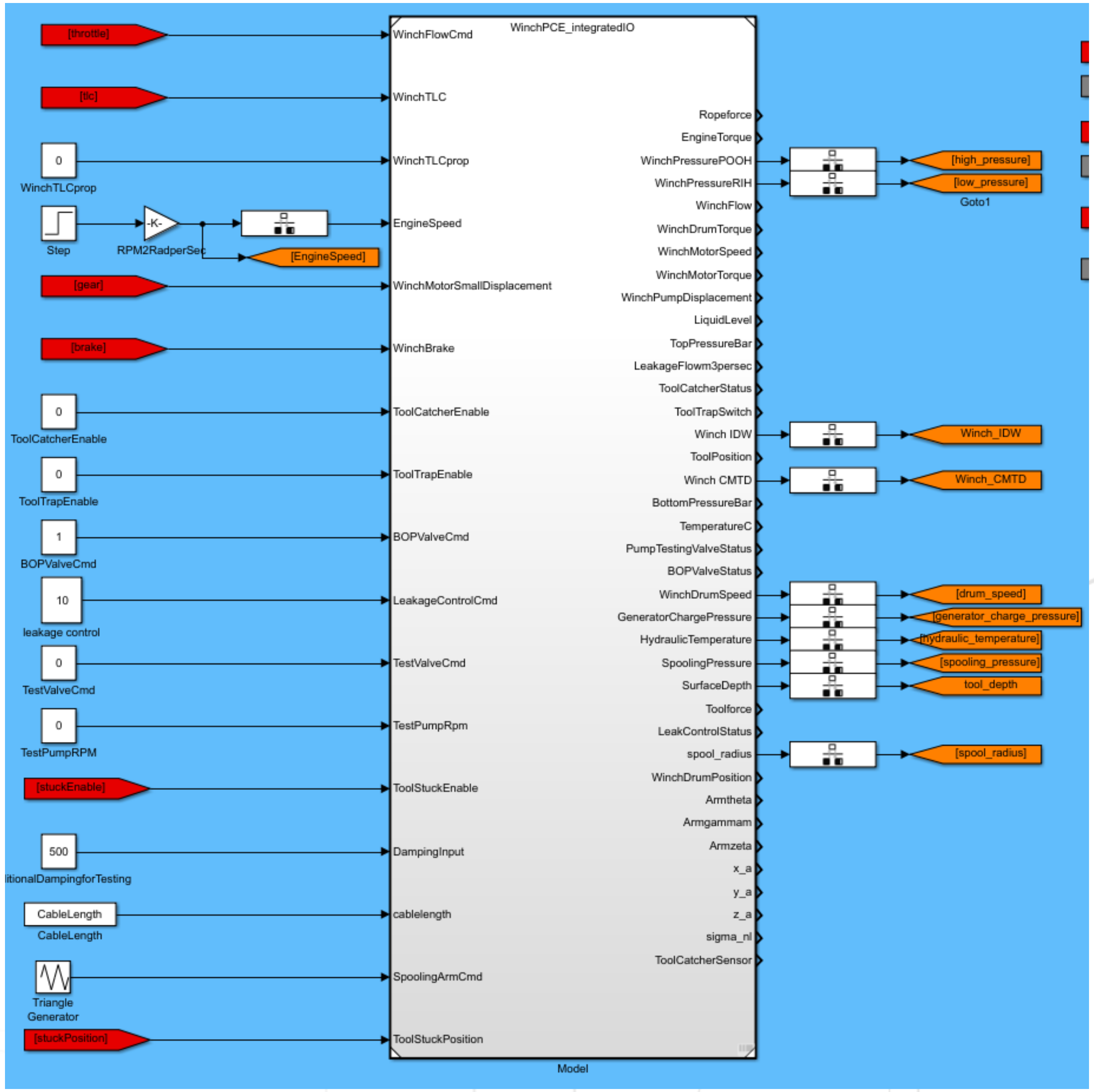
Digital Technology
Innovate. Engage. Deliver











Schlumberger

Software-in-loop testing

Back to Dashboard
Status
Changes
Build Now
Full Stage View
Open Blue Ocean
Embeddable Build Status

Build History trend

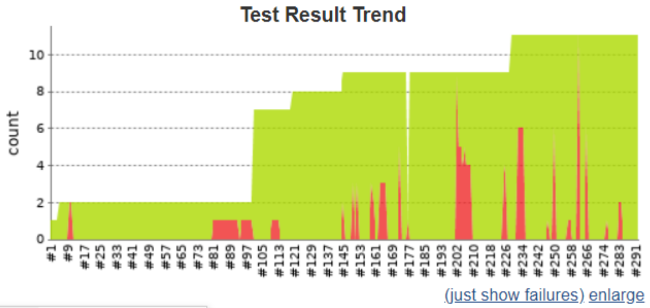
find

#292	May 2, 2019 6:50 PM
#291	Apr 29, 2019 5:11 PM
#290	Apr 26, 2019 9:28 PM
#289	Apr 26, 2019 8:15 PM
#288	Apr 24, 2019 5:49 PM

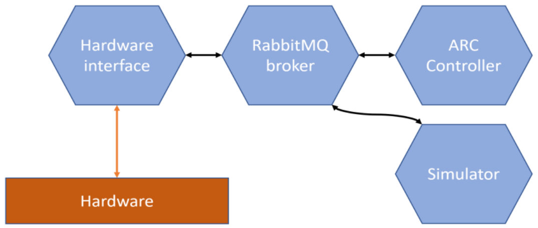
Pipeline dev-integration-tests_wafe-swct

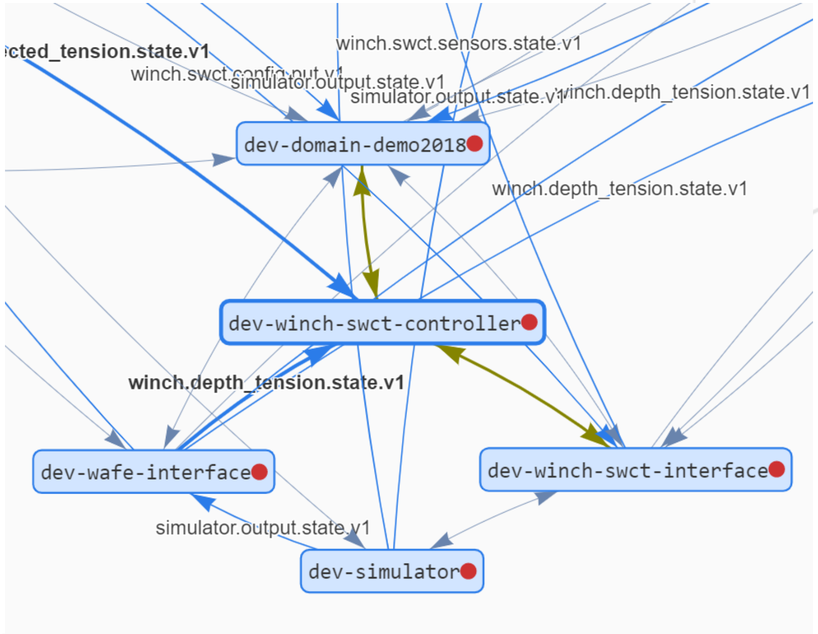
Recent Changes

Stage View



	Declarative: Checkout SCM	Prepare	Fetch Image Dependencies	Integration Test
Average stage times: (Average full run time: ~11min 47s)	678ms	1s	13s	10min 42s
#292 May 02 13:50 2 commits	1s	1s	13s	11min 40s

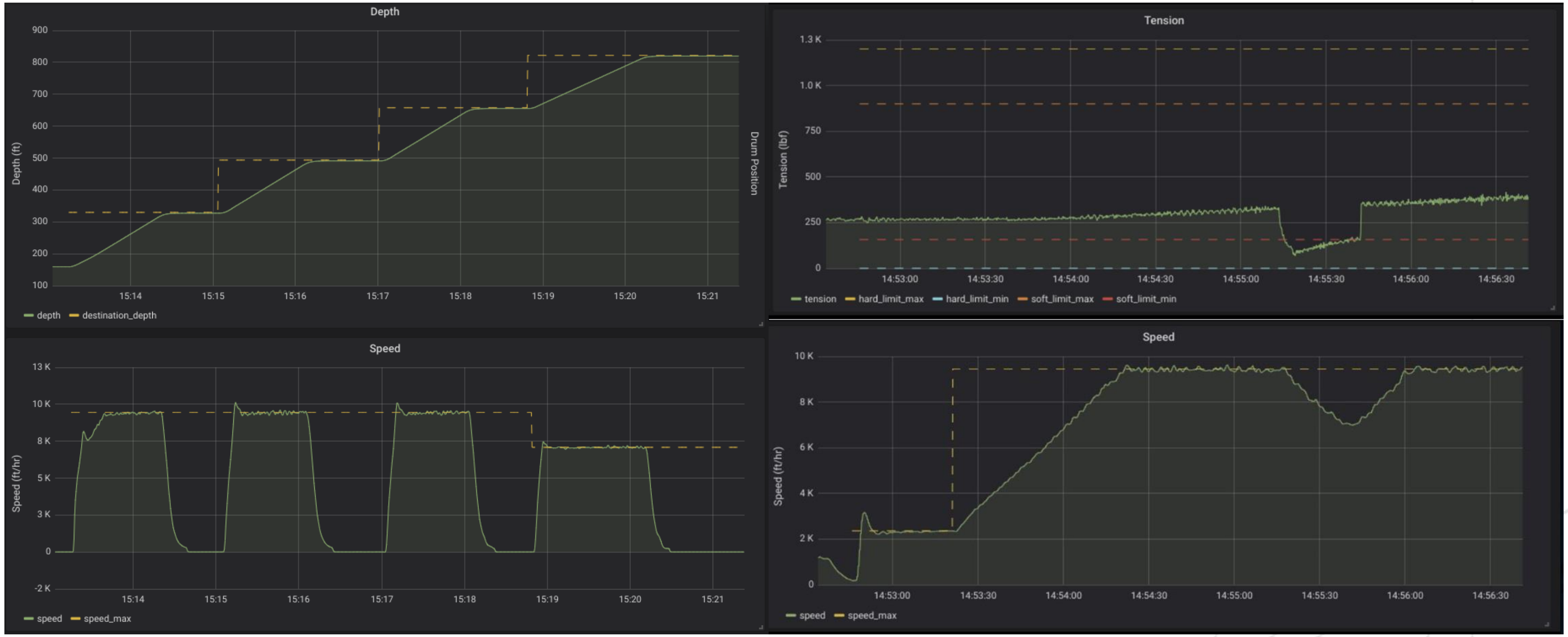





```python
@pytest.mark.timeout(600)
def test_ctrl_depth(evertest, depth_cmd, winch_mode_cmd, tension_offset_cmd, simulator_input_cmd):
    """
    Try to run in hole and pull out of hole by sending depth commands to dev-winch-swct-controller.
    Verify command response by reading dev-wafe-interface output.
    """
    start_depth = evertest.get_value(WAFE_INTF_DEPTH_TENSION_STATE_TOPIC, 'depth', timeout=1)
    assert start_depth is not None
    depth_cmd['depth_min'] = start_depth - 1 # m
    depth_cmd['depth_max'] = start_depth + 1 # m
    depth_cmd['cable_speed_limit_min'] = 0 # m/s
    depth_cmd['cable_speed_limit_max'] = 1 # m/s
    assert evertest.publish_msg(WINCH_CTRL_DEPTH_CMD_TOPIC, depth_cmd)
```

Truck Testing

Digital Technology
Innovate. Engage. Deliver



Schlumberger

Conclusion

Digital Technology
Innovate. Engage. Deliver

- ❑ Multi-domain physical system modeling using SimScape/Simulink
  - ❑ Using existing libraries
  - ❑ Customized component modeling using SimScape language
- ❑ Control algorithm design using Simulink/Stateflow
  - ❑ Together with other tools (Control system Toolbox, Simulink Design Optimization) offered by Mathworks to speed up controller design
- ❑ MIL simulation and testing
  - ❑ Quick what-if analysis and automation using Simulink Test
- ❑ Code generation for both controller and system model(Simulator)
  - ❑ Eliminate manual coding and speed up implementation
- ❑ Containerized controller and simulator
  - ❑ Facilitate CI/CD
  - ❑ Democratize model simulation
- ❑ Build and test automation (SIL) using Jekins/Azure devops
  - ❑ Automatic build and regression test

Schlumberger