

# TwinOps

## Digital Twins Meet DevOps

### Introduction

Cyber-Physical Systems (CPS) exhibit multiple engineering, verification and validation (V&V), and testing challenges. In this project, we aimed at reducing the time to get first test results by leveraging state-of-the-art system and software engineering approaches.

TwinOps explored the interplay between three core technologies:

- *Model-Based Engineering (MBE)*: model-based engineering relies on models as first-class abstraction of a system to support engineering activities;
- *DevOps*: an organizational effort to support continuous delivery of software through a better coupling between (Dev)elopment and (Op)erations activities;
- *Digital Twins*: an infrastructure to support system monitoring and diagnosis in real-time and enable continuous system improvement.

### Achievements

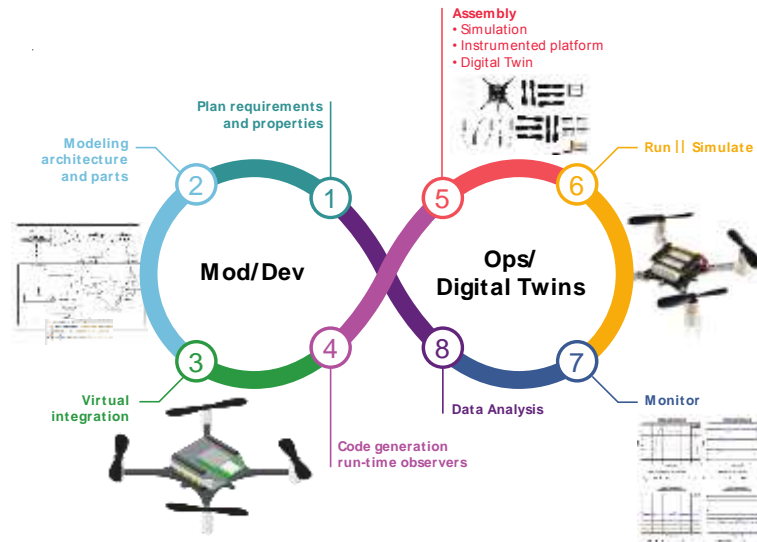
SEI delivers a ModDevOps exemplar

ModDevOps extends DevOps through MBE and its V&V and code generation capabilities. We demonstrate how MBE enables rapid system prototyping through a DevOps cycle.

SEI enhances analysis and testing process for systems architects who build software-intensive CPS with the *TwinOps* process

TwinOps builds on ModDevOps and Digital Twins to collect data on a system at runtime, and compare it to other engineering artifacts: model simulation and analysis. This comparison enables rapid system diagnosis.

# Model-Based Engineering brings early V&V and code generation to DevOps automation, or ModDevOps



### Approach

ModDevOps is defined as an abstract process using OMG SysML. This captures the key steps of the process as a collection of use cases, block diagrams, and activities.

⇒ Each project will adapt ModDevOps to its own problem/solution spaces

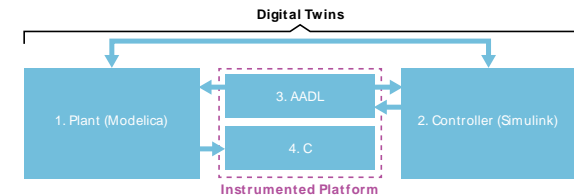
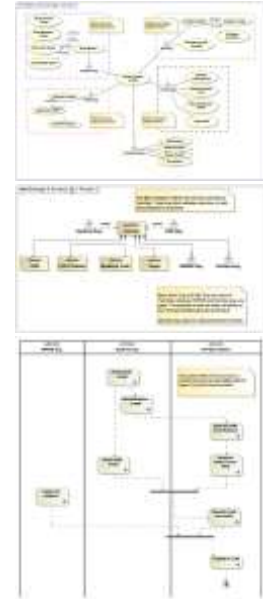
TwinOps is an instance of ModDevOps tailored for CPS. It combines

- AADL modeling for CPS architecture
- Simulink or C for the functional code
- Modelica for modeling the environment

The definition of the process as SysML models guides engineering phases:

- Orchestrate modeling, code generation, and compilation
- Continuous integration/continuous deployment used to deploy the system on the target, using Azure IoT cloud-based solutions

Code generation from model enables multiple scenarios: deployment on target and digital twins to support various operating scenarios.



Copyright 2020 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

Internal use:\* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use:\* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

\* These restrictions do not apply to U.S. government entities.

DM20-0864