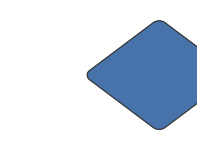
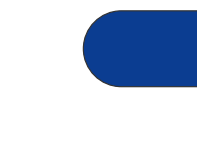
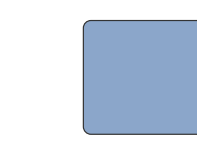


Flowchart Legend

-  Test Condition
-  Test Environment
-  Configuration Step

Environments and Definitions

1. Unit-Level Test Environment - Used for executing and validating isolated FSW (flight software) logic (e.g., command parsing or internal functions) without involving any simulation model or dynamic hardware behavior.

2. Model-Only Simulation Environment - Includes a standalone simulation model (e.g., a physics or hardware model) for validating component behavior independent of FSW.

3. Component Wrapper (FSW + Model Only) - Combines FSW and a simulation model for a specific component. Useful when testing the software's interaction with simulated hardware behavior, but without external or dynamic inputs.

4. Component Wrapper with Environmental Inputs - Extends the wrapper setup by adding either static or dynamic environmental models (e.g., preconfigured input files or orbital simulators like 42) to simulate time-varying or mission-relevant inputs.

5. Subsystem Simulation Environment - Simulates a full subsystem composed of multiple components (e.g., ADCS with sensors and actuators). Used to validate internal logic and interactions within the subsystem.

6. Full System Simulation Environment - Provides a complete, integrated spacecraft simulation, including all subsystems, environmental models, and ground software. Designed for mission-like, end-to-end test scenarios.

Environment Setup Notes

• **Environments 1–3** (Unit Test to Wrapper w/ Model): Can be run locally, entirely within an IDE using software stubs, basic drivers, or mock inputs. These environments are typically created by extracting and isolating the target flight software code from the full stack and running it independently for focused validation.

• **Environment 4** (Wrapper w/ Environmental Inputs): Still executable locally, using either seamlessly integrated scripted inputs or external simulators (e.g., 42). While scripted feeds can be embedded directly into the wrapper, orbital simulators require additional interfacing, introducing moderate complexity. Use an orbital simulator when environmental conditions, like sunlight, attitude, or orbit, must evolve automatically during the simulation or are difficult to coordinate through static input files.

• **Environments 5–6** (Subsystem & Full System Sims): Typically involve orchestrated runtimes, middleware, multiple containers or VMs, full FSW execution, all simulators, and often ground software, set up across mission-like infrastructures.

Note: While this flowchart offers a structured guide for environment selection, it is not exhaustive. Setups should be tailored to match their scenario's goals, fidelity needs, and constraints.

****Start simple and layer tools only when scenario needs justify them****