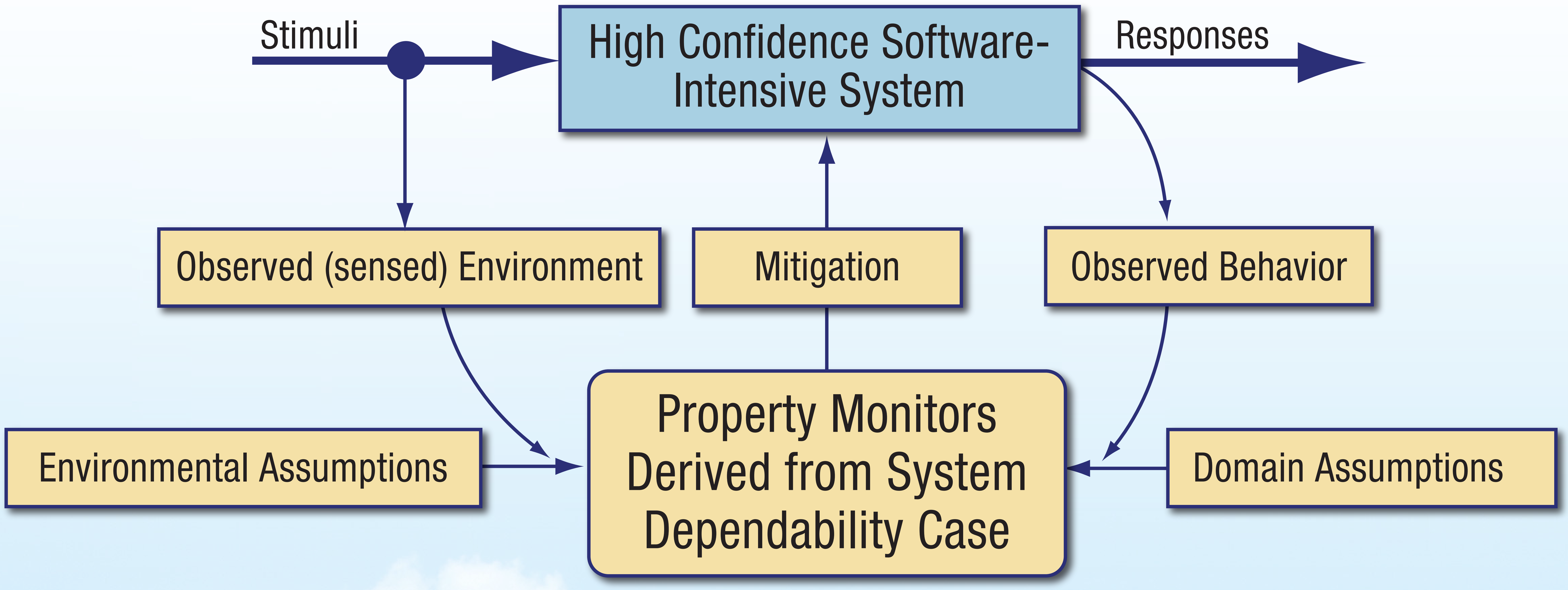
National Aeronautics and Space Administration

Software Health Management

Paul S. Miner and Eric G. Cooper, NASA Langley Research Center


Stimuli
High Confidence Software-Intensive System
Responses
Observed (sensed) Environment
Mitigation
Observed Behavior
Environmental Assumptions
Property Monitors Derived from System Dependability Case
Domain Assumptions

PROBLEM

Software-related incidents

- Failure in the data bus that monitors fuel levels and flow resulted in one engine losing power and another beginning to fluctuate; faulty software logic prevented a working backup computer from being selected
- Two occurrences of a single engine thrust rollback during takeoff because a flawed software algorithm computed an erroneous N1 (fan speed) command
- Defective software program provided incorrect data about the aircraft's speed and acceleration resulting in a harrowing roller-coaster ride zooming 3,000 feet upward, followed by a steep dive, followed by another steep climb
- Loss of cockpit power that shut down avionics systems including radio and transponder preventing the pilot from issuing a Mayday call. Because of a software design flaw, an action for restoring power was not shown to the flight crew due to it's position on a list

APPROACH

Explore software health management in the context of system level dependability cases by developing a framework that enables

- Explicit claims of system (and subsystem) requirements including assumptions about the application domain and environment in which the system is to operate
- Evidence that software satisfies these explicit claims under the stated domain assumptions
- Architectural principles, enforced by hardware mechanisms, that ensure that software behavior dependencies are traceable; and
- Mechanisms for correctly composing software systems from trusted components within the constraints imposed by the architectural principles