

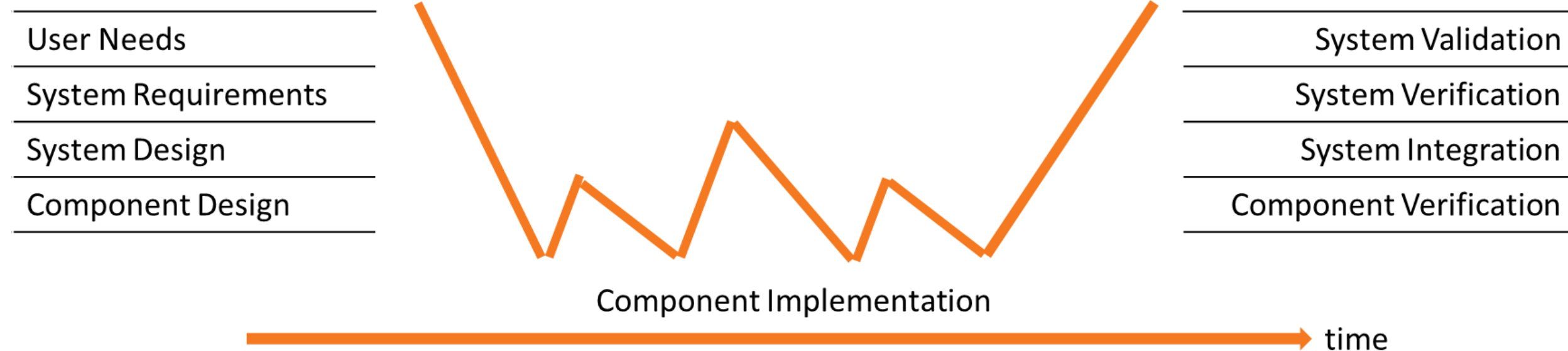


Preventing and Diagnosing System Integration Issues

Using an open-source approach: Component Modeling & Analysis (ComMA)

CHALLENGES

- Many integration issues arise from unclear interfaces between components
 - Problems with 3rd party components
 - Components developed by different teams



- Integration issues might require iterating on the component or even the system design
- Hard to predict when iteration ends, and full integration is achieved

CAUSES

- Interface descriptions are often minimal
 - Typically, listing only call signatures
 - A document-based engineering approach
 - Can lead to ambiguity and imprecision

- Commonly missing elements include
 - The allowed order of calls
 - Expected timing behavior
 - Constraints on data

SOLUTION DIRECTION

- Models include commonly missing elements
 - Behavior by protocol state machine
 - Contract between client(s) and server; allowed sequences of calls
- Constraints
 - Timing; specifying timing intervals between calls
 - Data; valid value ranges for call parameters
- Unambiguous, precise models can mitigate integration problems

COMMA DESIGN MODELS

- The Domain Specific Languages (DSLs) incorporate previously missing elements
- The DSLs use an interface/implementation technology agnostic notation

SIGNATURE

```
commands
Status PowerOn
int TakePicture(timestamp)
signals
PowerOff
notifications
Click
```

TIME & DATA CONSTRAINTS

```
timing constraints
TC1 in state On command TakePicture - [ 10.0 ms .. 75.0 ms ]
-> notification Click
TC2 in state Off command PowerOn and reply(Status::OK)
-> [ .. 100.0 ms ] between events
```

INTERFACE (BEHAVIOR)

```
initial state Off {
  transition trigger: PowerOn
  do: reply(Status::OK) next state: On
OR
  do: reply(Status::Failed) next state: Off
}
state On {
  transition trigger: TakePicture(timestamp)
  do: reply(*) Click next state: On
  transition trigger: PowerOff next state: Off
```

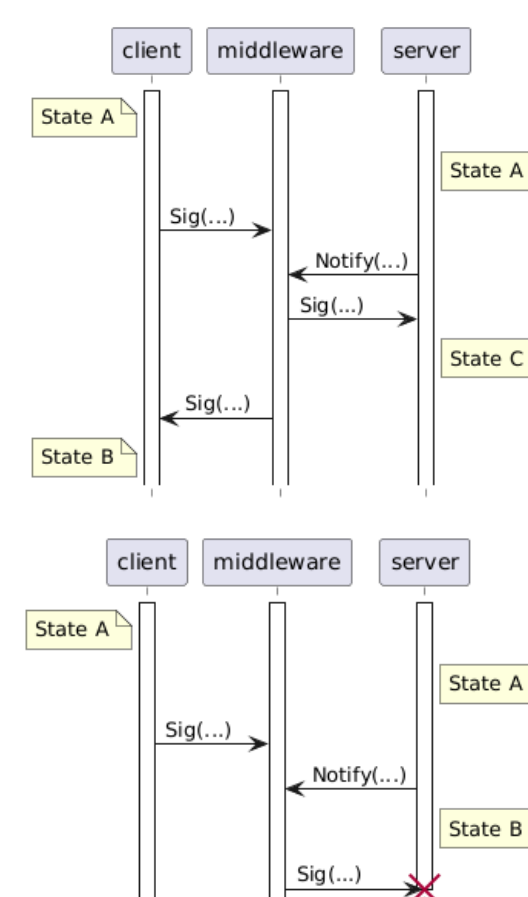
COMPONENTS

```
component componentCamera
provided port ILens lensPort
provided port ISystem systemPort
functional constraints
// SetZoom and Autofocus are only allowed when system is on
FC1 {
  use events
  command systemPort::PowerOn
```

THE COMMA APPROACH

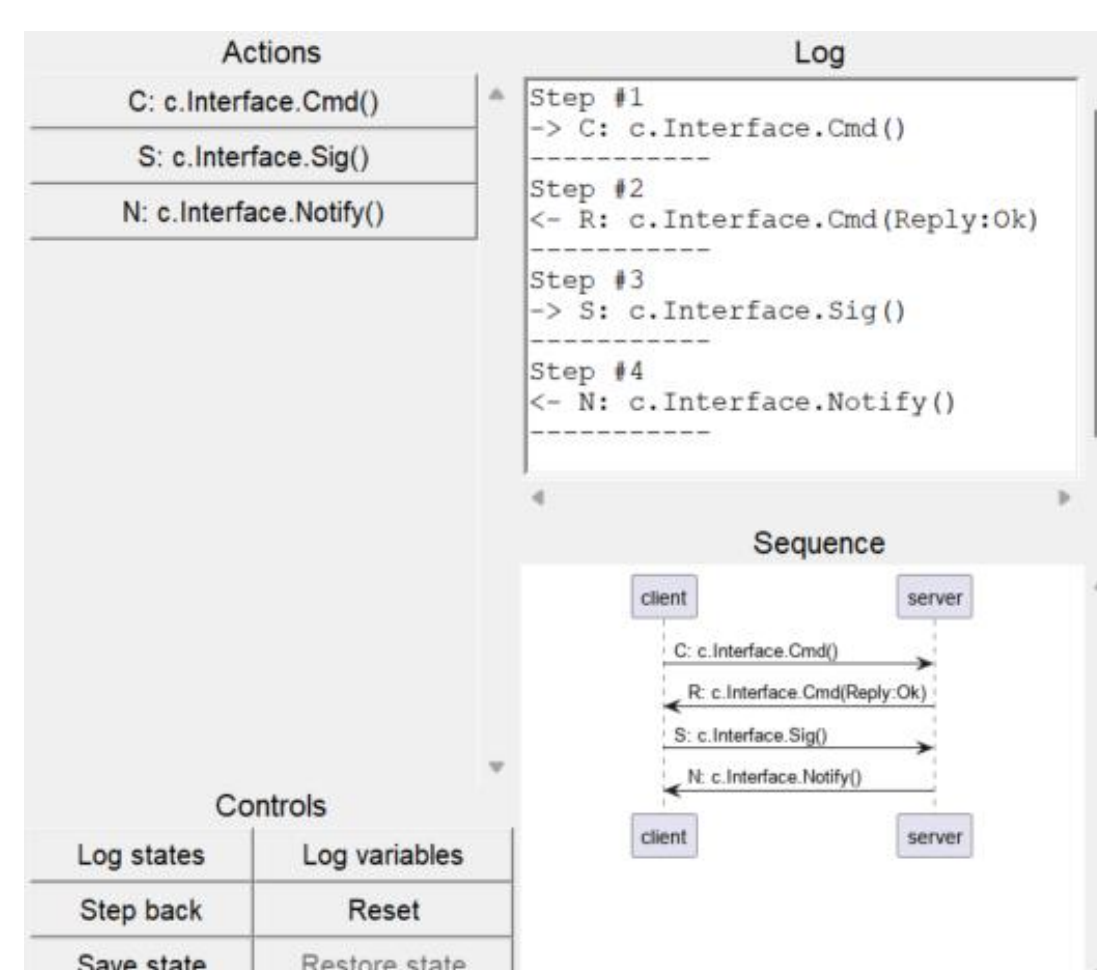
MODEL QUALITY CHECKS (VERIFICATION)

- Goal: detect interface design errors before implementation begins
- Quality of interface model
 - Unreachable states: if there is no sequence of transitions to get to that state
 - Deadlock states: if no transitions can be executed in that state
 - Sink states: if there is no transition that allows to leave that state
 - Livelock states: if there is no sequence of transitions to get to the "home" state
- Quality of interactions
 - Race conditions: if both a client and a server can independently make a choice to execute a transition



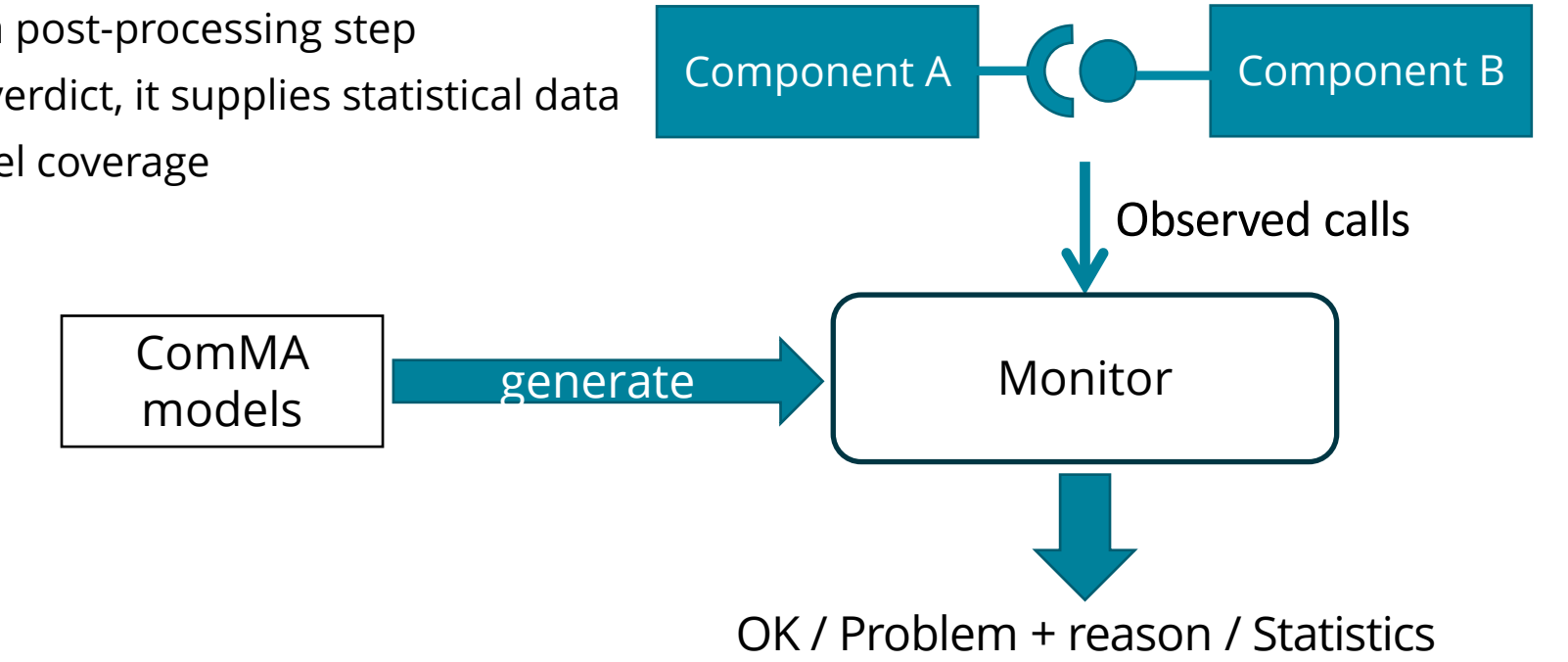
SIMULATOR (VALIDATION)

- Goal: identify design model errors before the implementation phase
- Explore an **interface** or **component** model's execution behavior



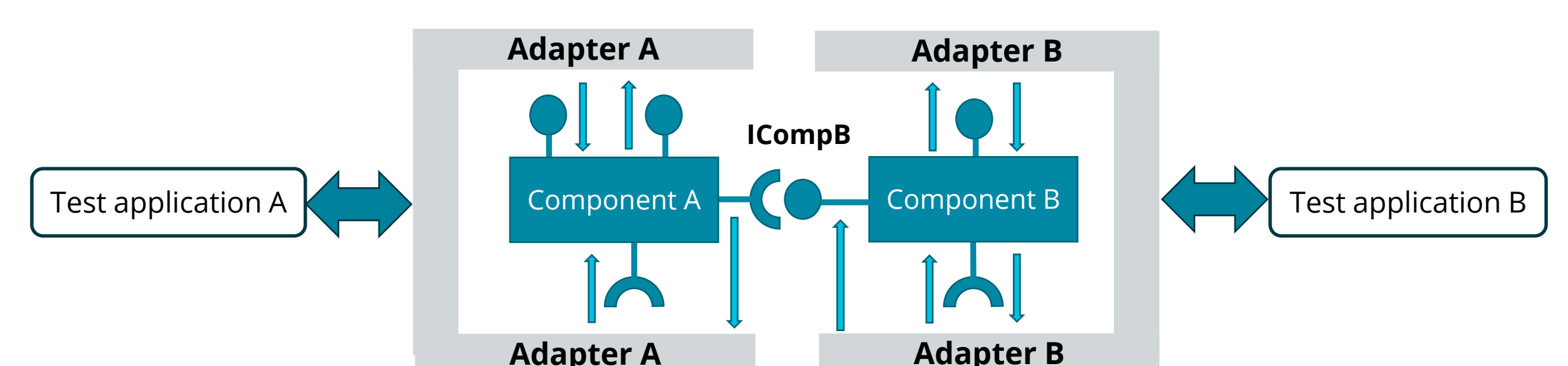
OFFLINE RUNTIME MONITORING (DURING & POST-INTEGRATION)

- Goal: investigate integration issues during and post-integration
- Offline runtime monitoring of stored execution logs
 - Translators exist for OpenAPI/AsyncAPI and supported proprietary technologies to monitor input
 - Performed as a post-processing step
 - Alongside the verdict, it supplies statistical data
 - Such as model coverage



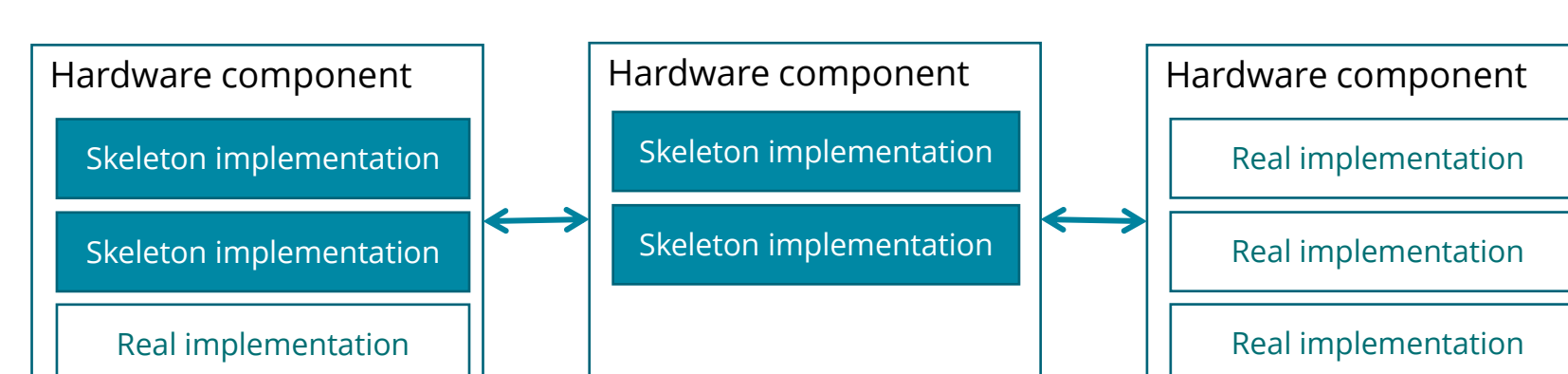
ONLINE MODEL-BASED TESTING (PRE-INTEGRATION)

- Goal: test components pre-integration to prevent issues during the integration phase
- Online MBT for confidence testing, i.e., for checking the ICompB interface (see figure below)
- Test applications are automatically generated from **interface** and/or **component** models
- Adapters are automatically generated for OpenAPI/AsyncAPI and supported proprietary technologies



SKELETON IMPLEMENTATIONS (EARLY INTEGRATION)

- Goal: verify program flows by early integration with real and skeleton implementations
- Generate a skeleton implementation from component model
 - Component model specifies partial behavior, typically the main program flows
 - Skeleton implementations comply with specified timing budgets



This functionality is not yet included in the open-source release

- CommaSuite is an Eclipse Foundation project



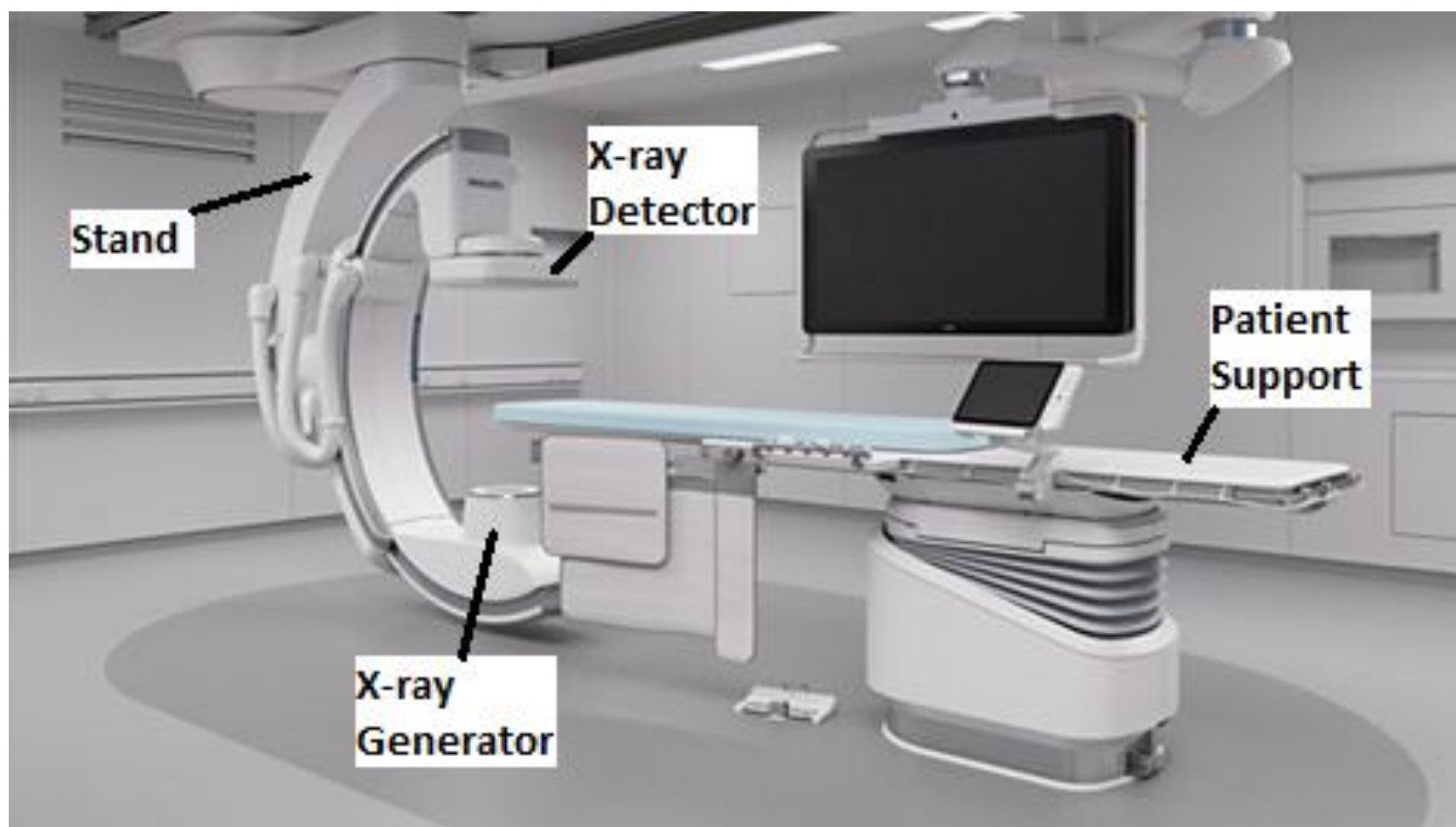
- <https://eclipse.dev/comma>





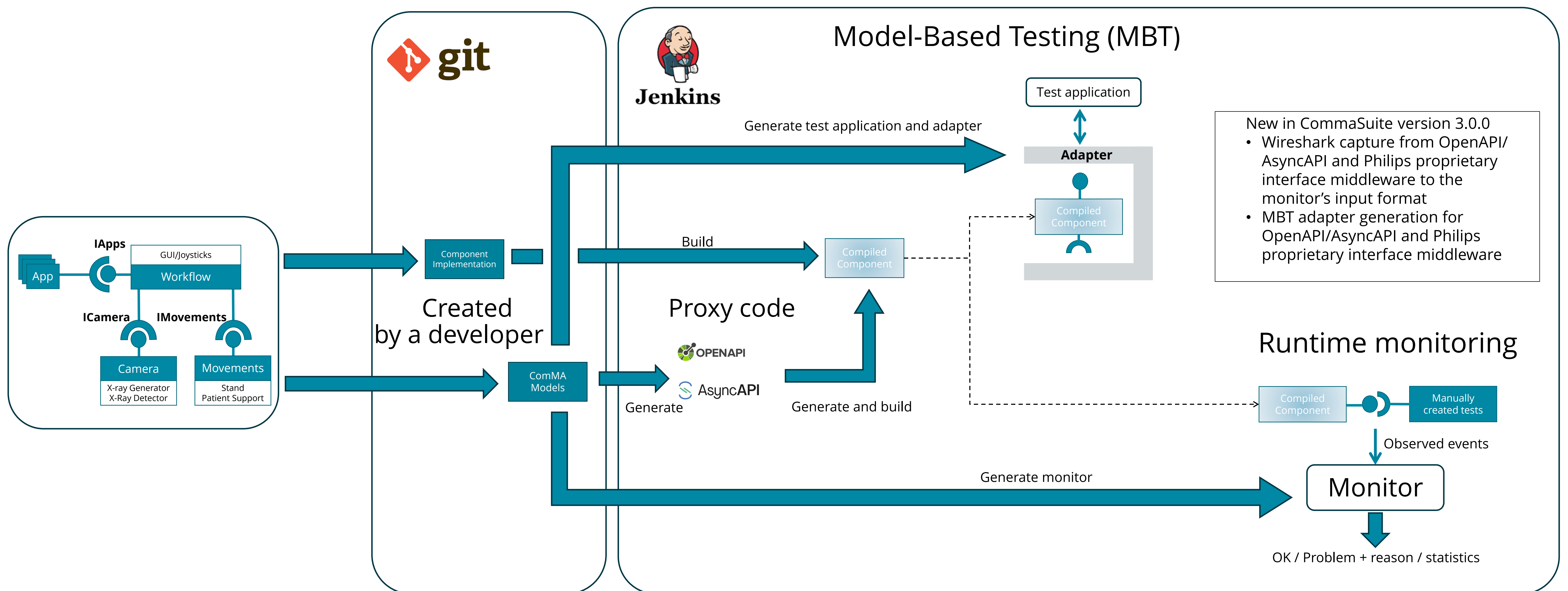
A DECADE OF INNOVATION WITH COMMA AT PHILIPS IGT

Component Modeling & Analysis (ComMA): 2015 - 2025



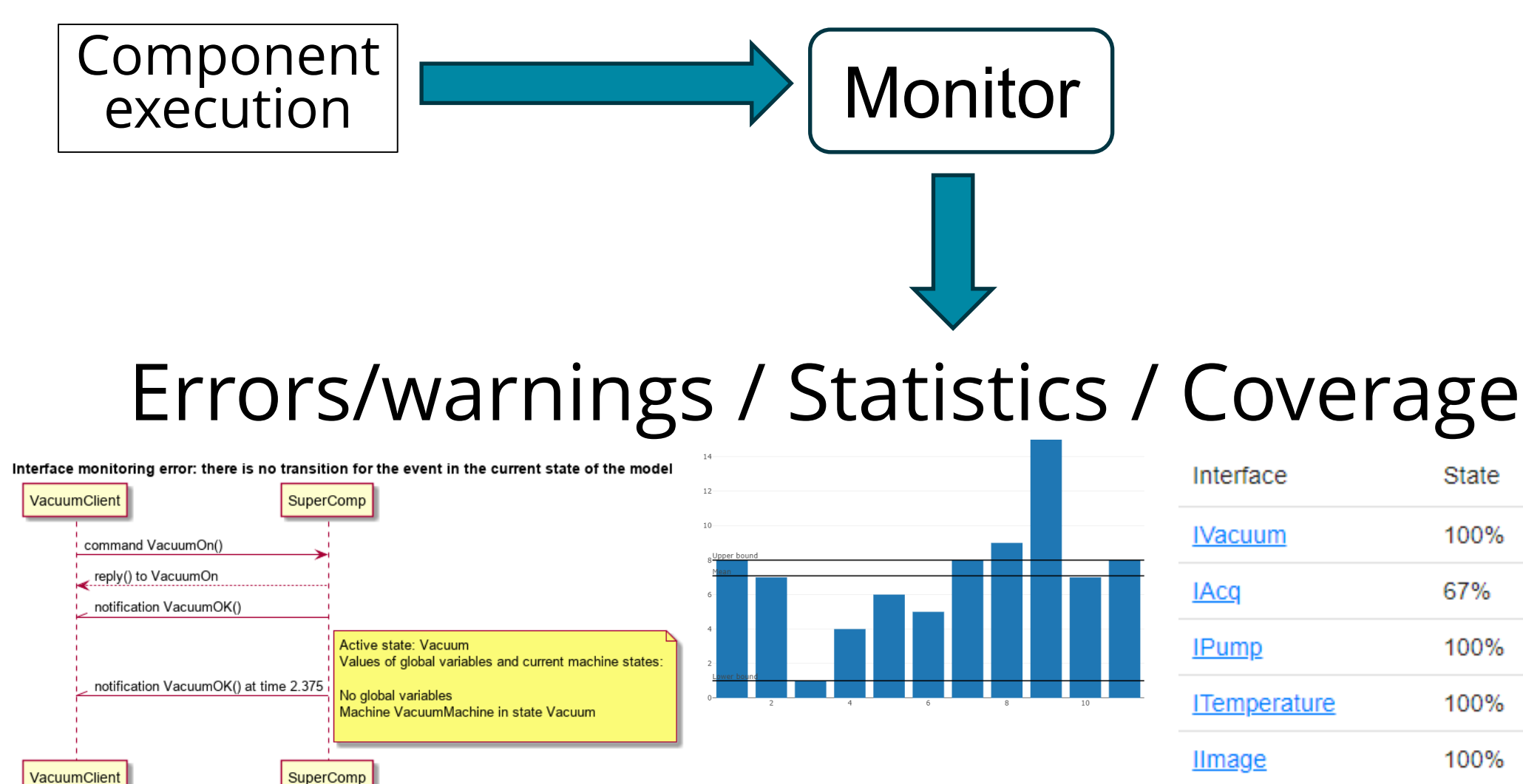
- **Context**
 - Components are developed by multiple teams, parties, and suppliers
- **Challenge**
 - Integrating all components
- **Solution**
 - Proxy code generation
 - Offline runtime monitoring of components
 - Online model-based testing of components

COMMA'S ADOPTION AT PHILIPS IGT



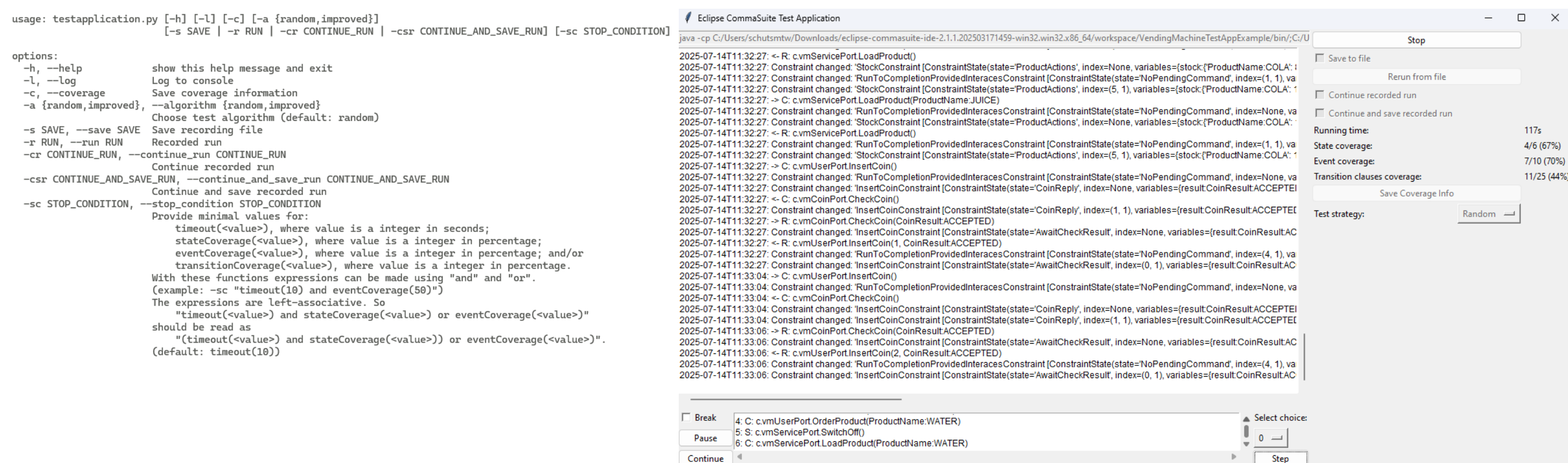
RUNTIME MONITORING

Command-line (for CI/CD automation)



MODEL-BASED TESTING

Command-line (for CI/CD automation) / Graphical user interface (for developer laptop)



Partners:

PHILIPS

Contact

mathijs.schuts@tno.nl

www.esi.nl