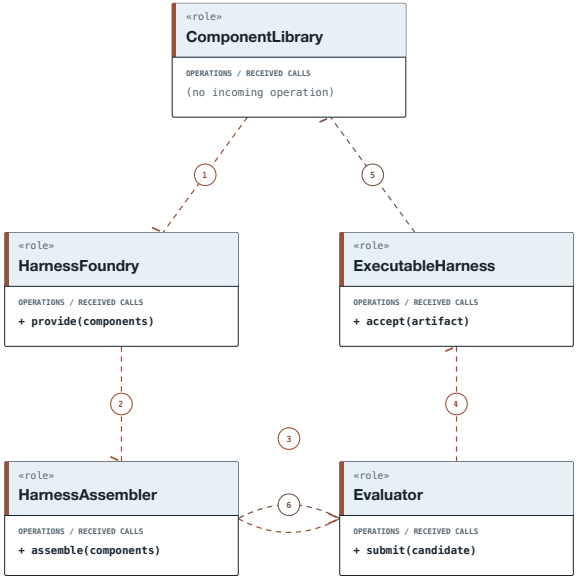
# E05 Composable harness foundry

**Evolution / UML CLASS VIEW / ANALYTICAL MODEL**

Construct and evolve reusable harness components.

**UML CLASS VIEW**

dashed arrow: call dependency · brown: return/failure · analytical, not an observed trace




**POINTCUT / PATTERN INTERFACE**

| | |
|---|---|
| CONTEXT | Harness behavior is assembled from independently typed components. |
| TRIGGER | Components are selected, composed, or replaced. |
| STRUCTURE | Component interface, composition builder, evaluator, and artifact registry. |
| EFFECT | The built harness is executable and its component provenance is recoverable. |

**CONTRACT / REQUIRED BEHAVIOR**

| | |
|---|---|
| REQUIRES | Components have compatible contracts and composition operators. |
| ENSURES | Invalid composition is rejected before execution. |
| INVARIANT | Reusing a component does not silently reuse privileged state. |
| VIOLATION | Build conflicts, incompatible versions, and evaluation failure are explicit. |

**RELATIONSHIPS / ANALYTICAL CALLS**

number · caller → callee · message

1 ComponentLibrary → HarnessFoundry provide(components)
2 HarnessFoundry → HarnessAssembler assemble(components)
3 HarnessAssembler → Evaluator submit(candidate)
4 Evaluator → ExecutableHarness accept(artifact)
5 ExecutableHarness ↩ ComponentLibrary report(reusable)
6 Evaluator ↩ HarnessAssembler feedback(failure)

**VARIATION / TEST / NEGATIVE CONTROL**

| | |
|---|---|
| VARIATION | Typed processors, foundry DSL, component search, and immutable builders. |
| TEST_OBLIGATION | Test composition, incompatible component, rebuild, and provenance. |
| NEGATIVE_CONTROL | Separate helper functions without component contracts are insufficient. |