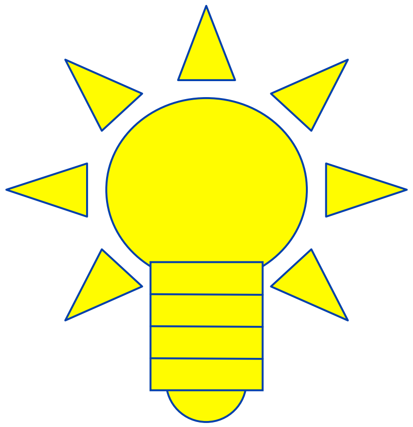
MonDe: Safe Updating through Monitored Deployment of New Component Versions

Alessandro Orso
Georgia Institute of Technology

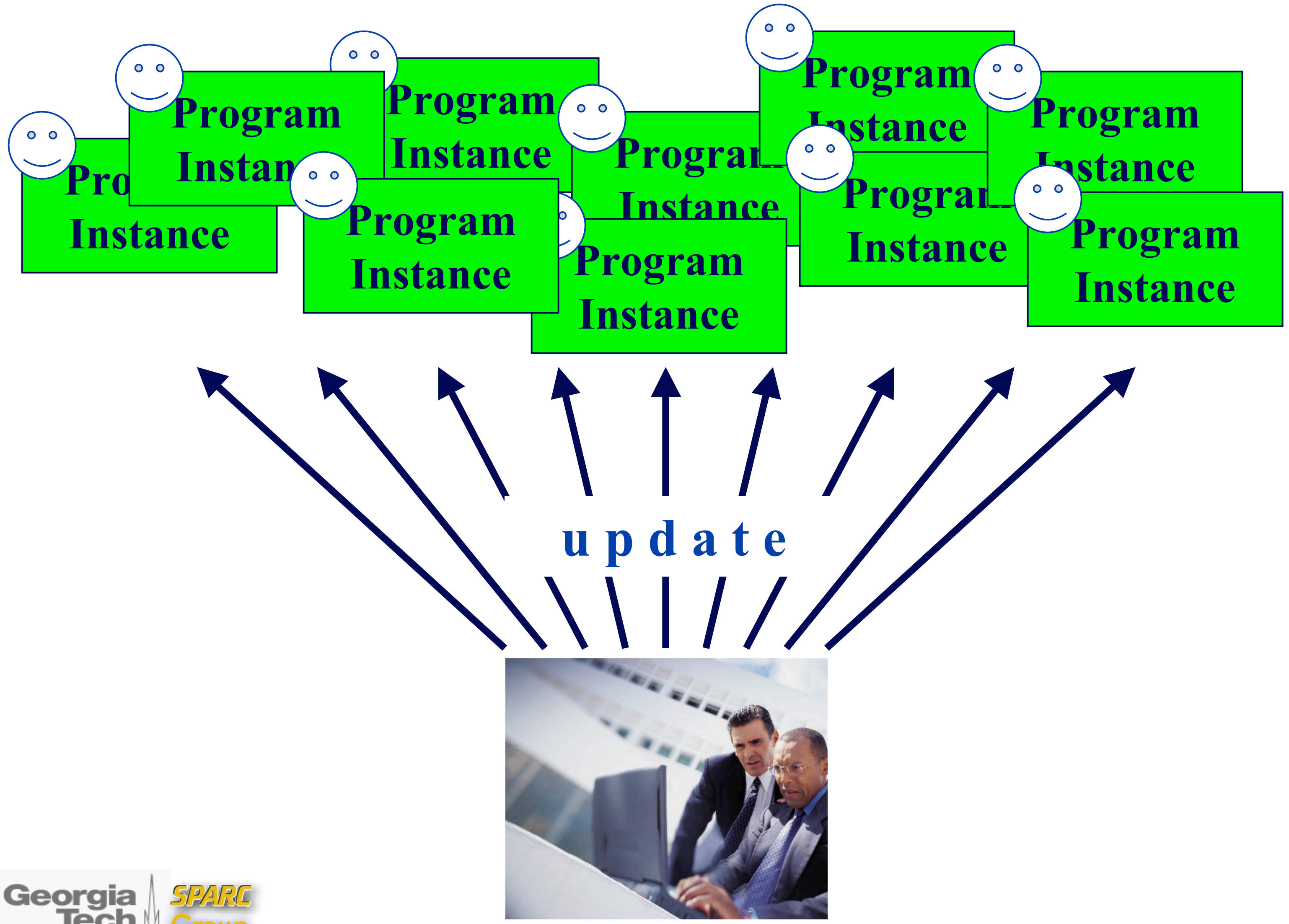
Jonathan Cook
New Mexico State University

This work was supported in part by NSF awards CCR-0306457, EIA-9810732, and EIA-0220590 to New Mexico State University and CCR-0205422, CCR-0306372, and CCR-0209322 to Georgia Tech.

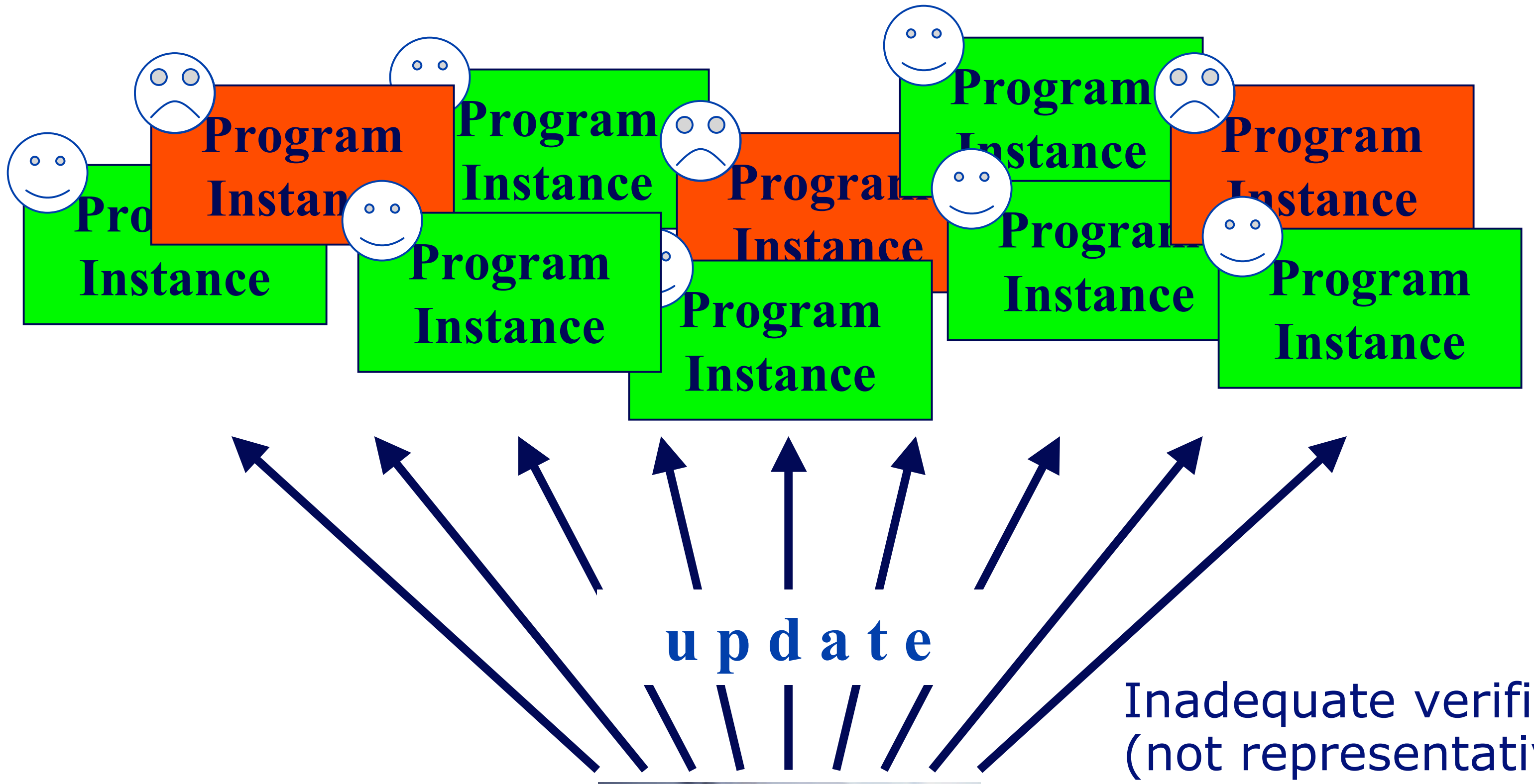
Idea Paper

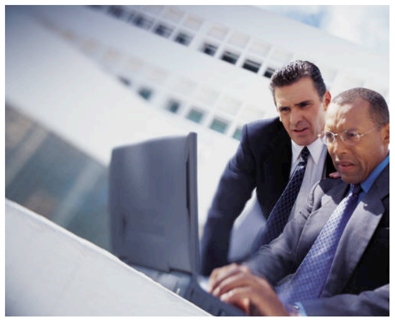


Software Updating



Software Updating





Inadequate verification (not representative)

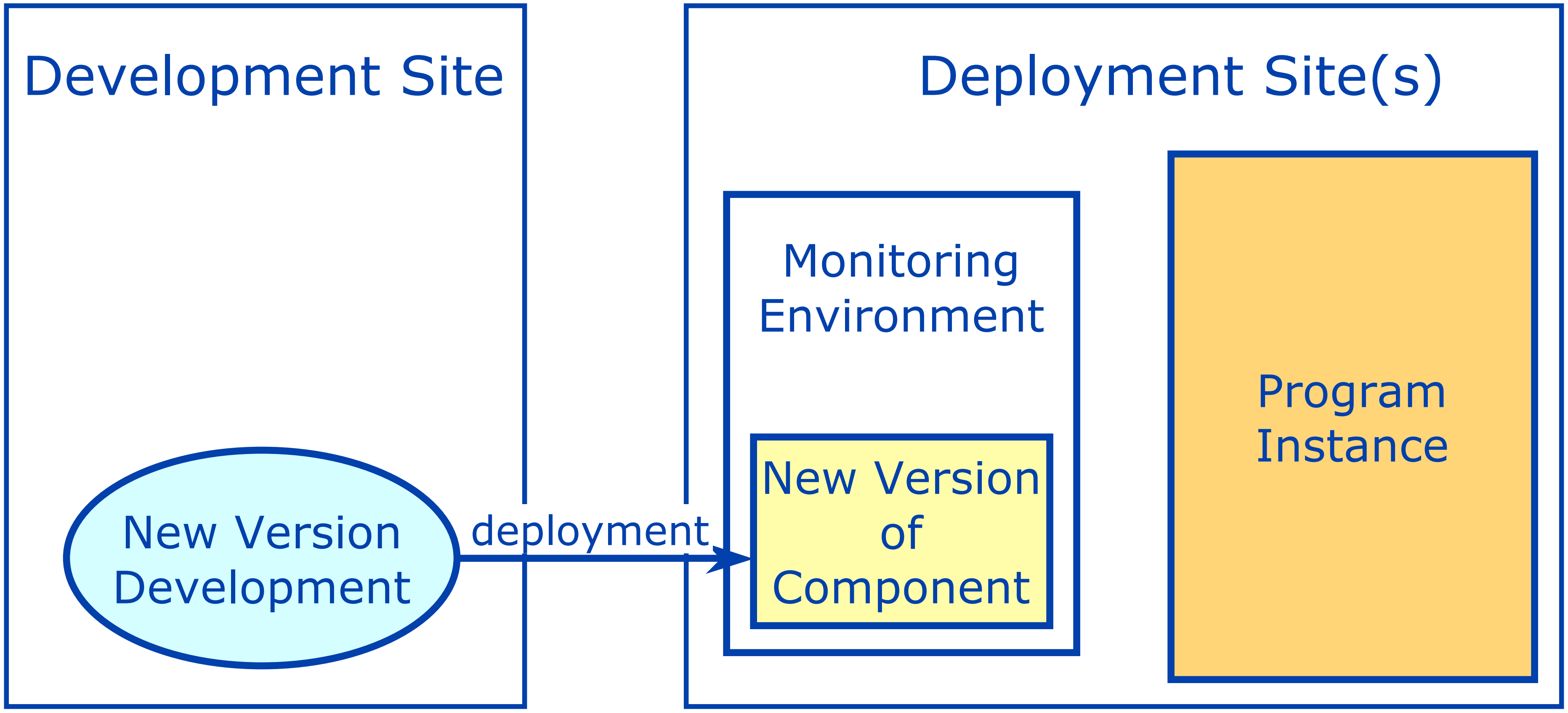
- User profiles unknown
- User configurations unknown
- Too many profiles/configs
- Hard to prioritize/focus testing effort

Proposed Solution: MonDe

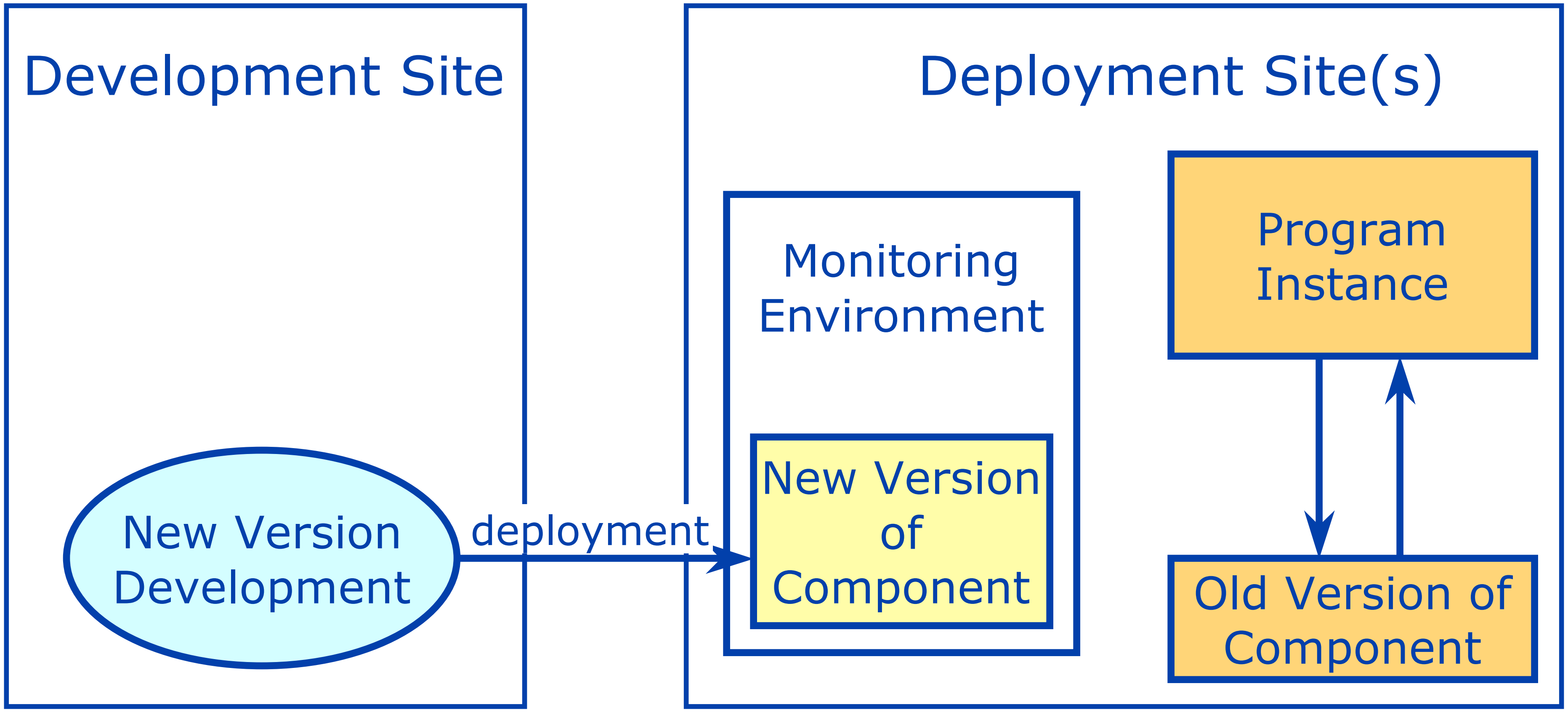
MonDe: Monitored Deployment

- Deploy updates at remote sites
- Run new version in a sandbox using actual workload
- Report the results back to developers

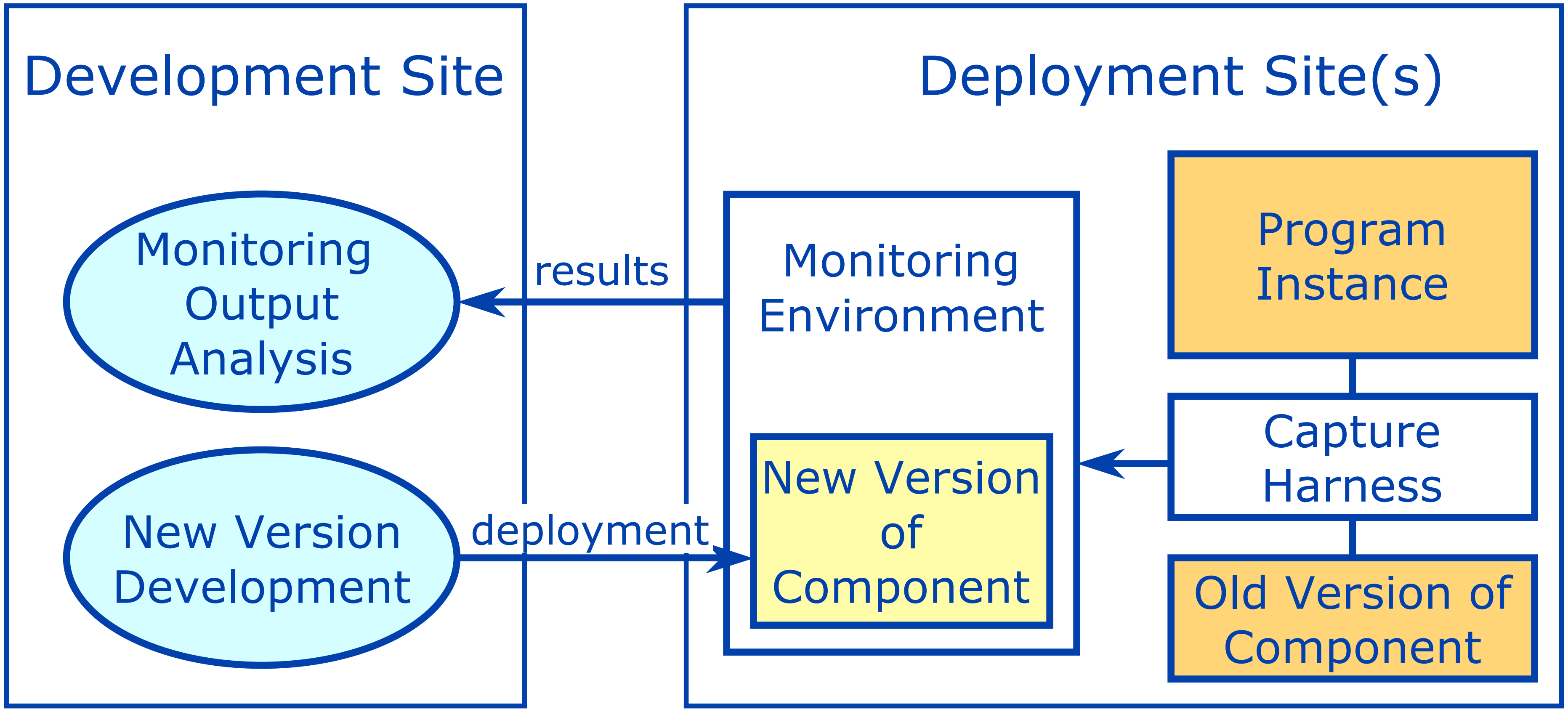
MonDe Framework



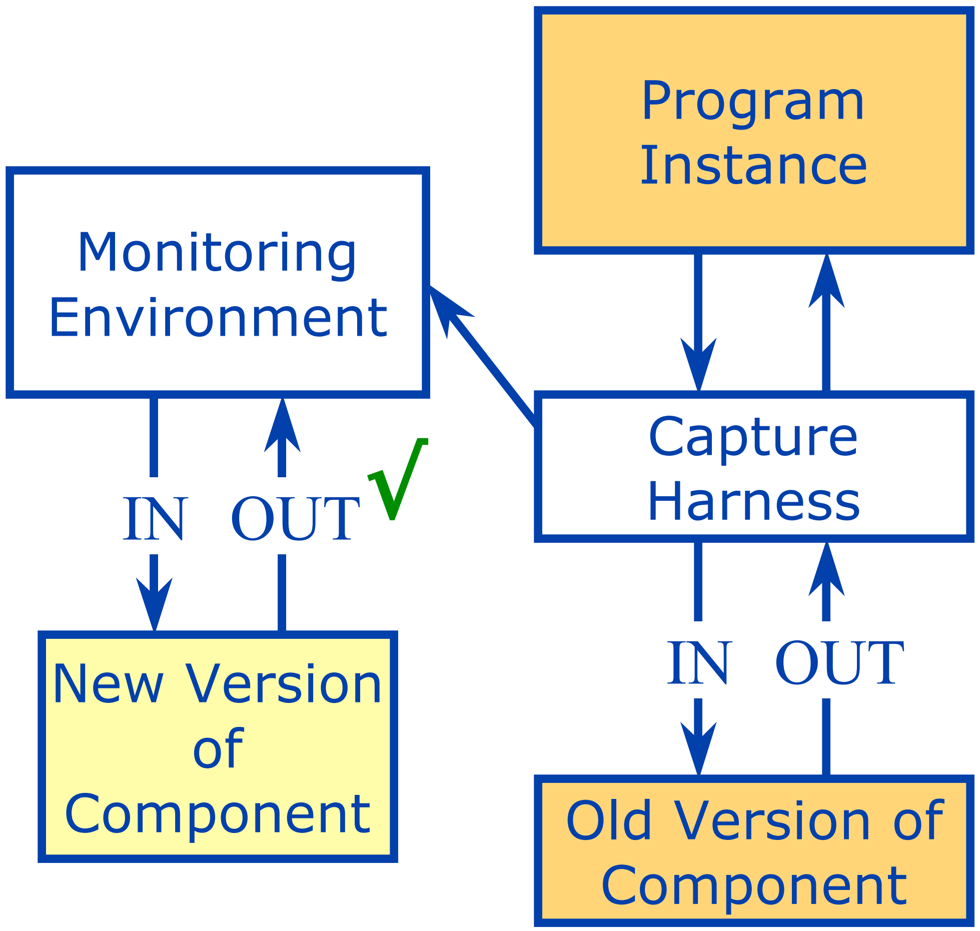
MonDe Framework



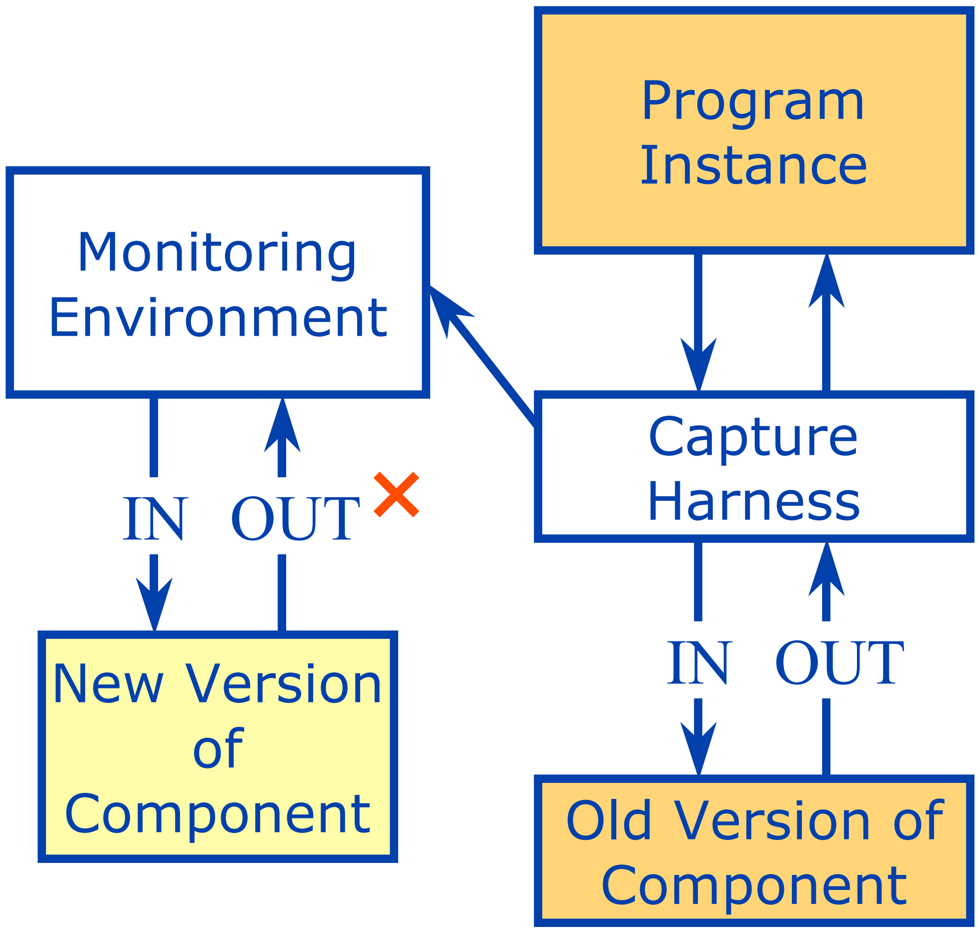
MonDe Framework



Capture Harness



Capture Harness



MonDe: Advantages

- Perform evaluation on real user data
- Leverage remote resources
- Protect user data privacy (mostly)
- Enable pre-processing of execution results
  - Avoid/limit false negatives (?)
  - Produce useful reports (?)

MonDe: Requirements

Capture capability

- Identify boundaries SW/new component
- Record interaction through boundaries

Execution and monitoring capability

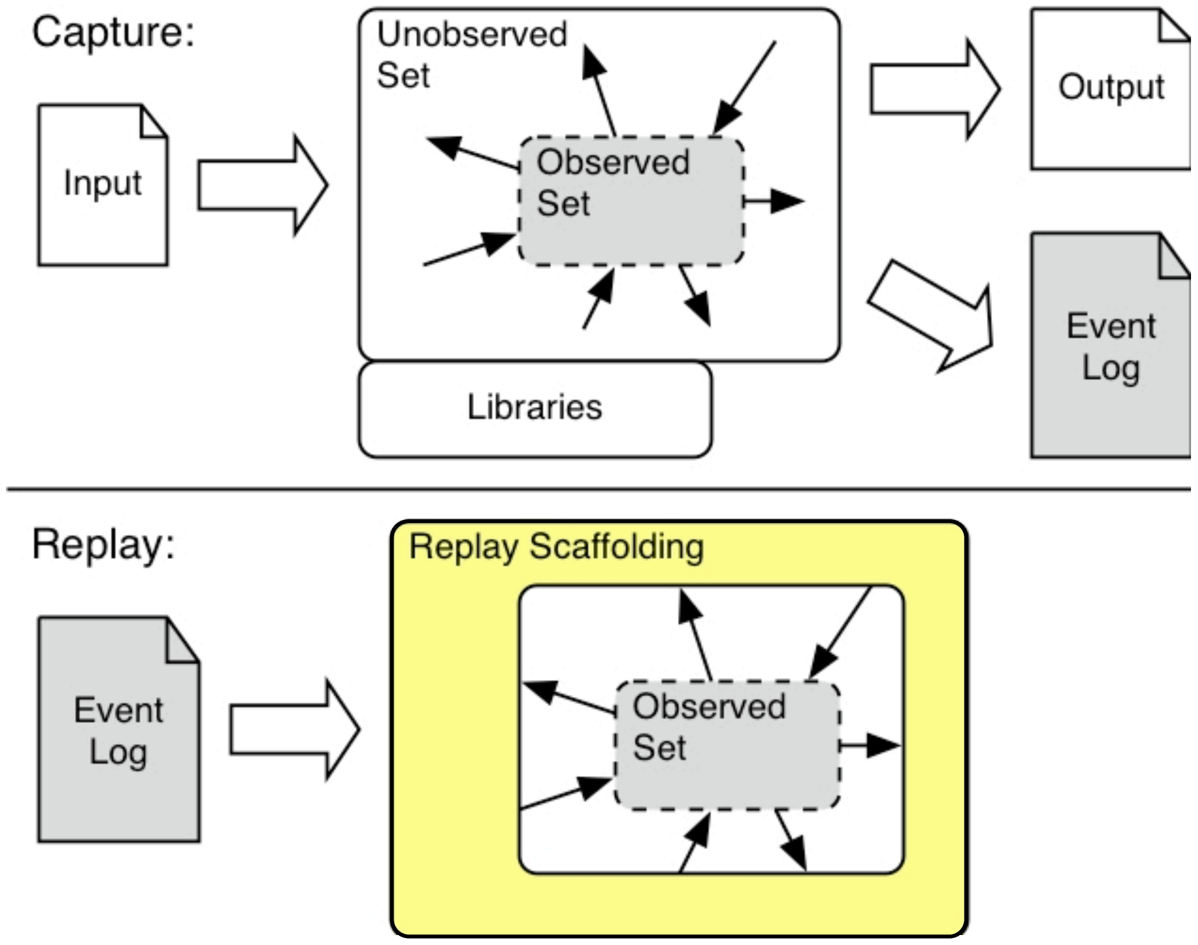
- Replay captured interactions in sandbox
- Observe and report results

⇒ Two approaches proposed

- $\boxed{\text{Offline (SCARPE)}}$
- Online (DDL)

SCARPE: Selective CApture and Replay of Program Executions

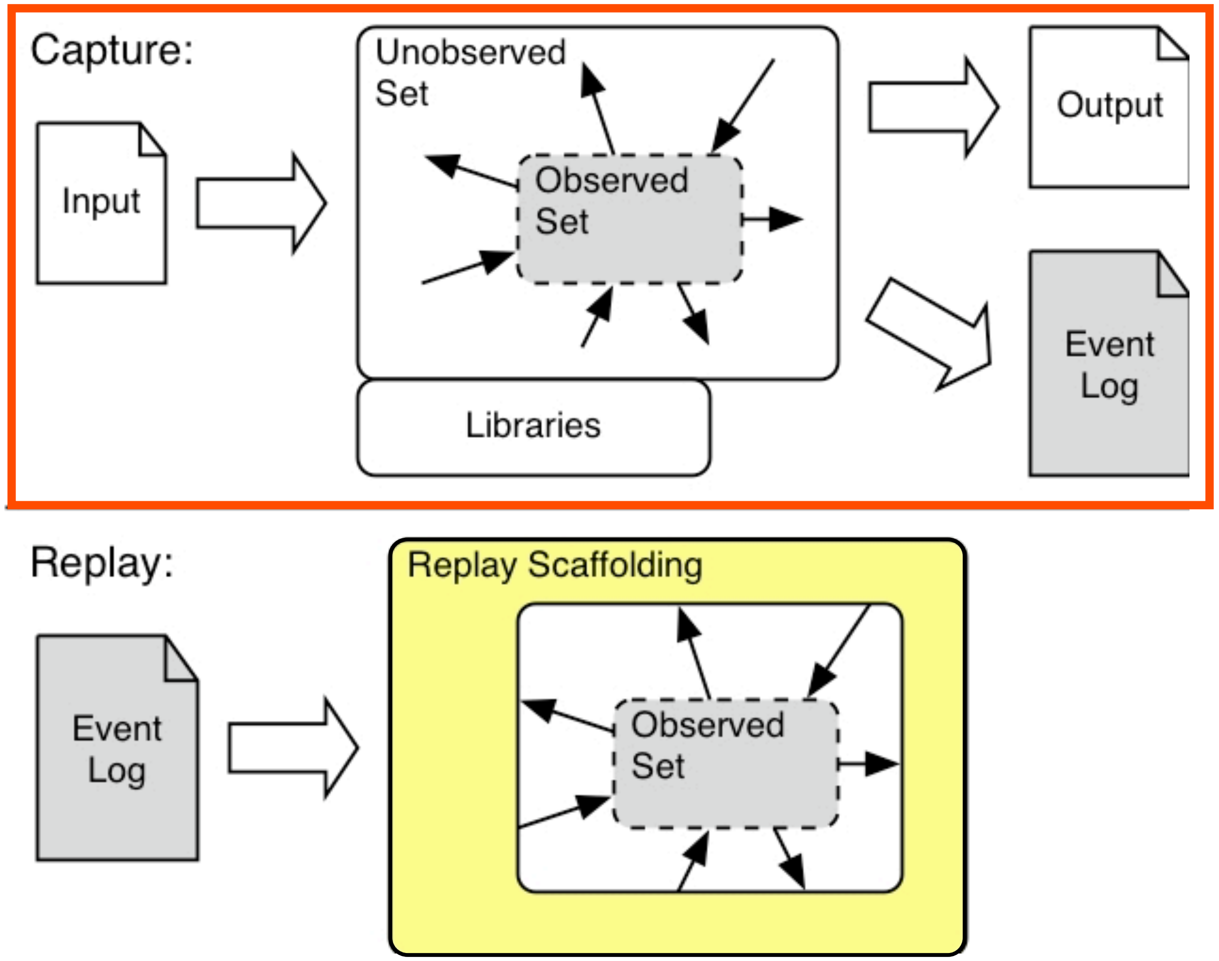
Defined for Java applications



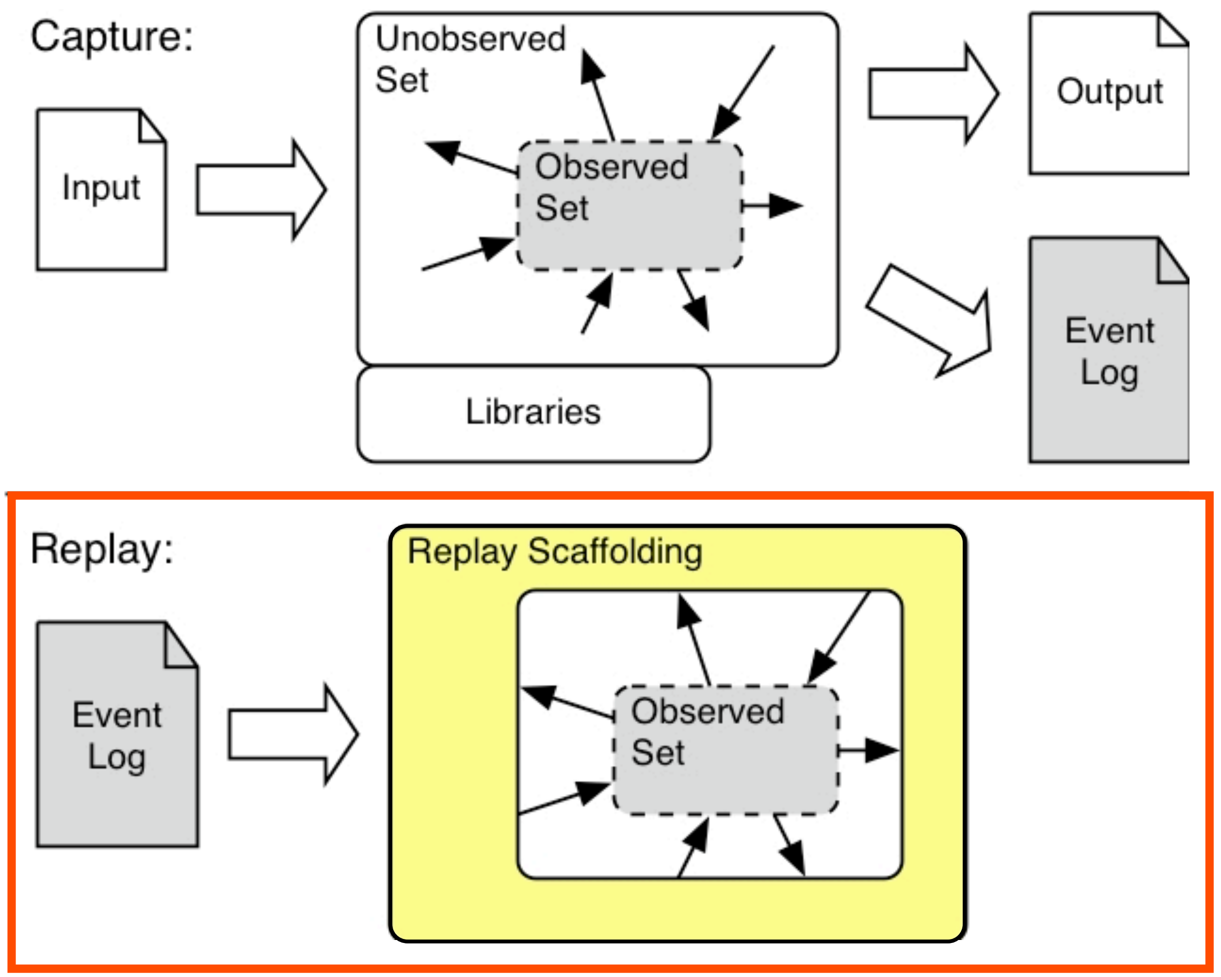
SCARPE: Capture Phase

- Input *observed set*
- Identify observed-set's boundaries
- Collect interactions and data across boundaries
  - method calls/returns
  - exceptions
  - field accesses

  => *event log*



SCARPE: Replay Phase

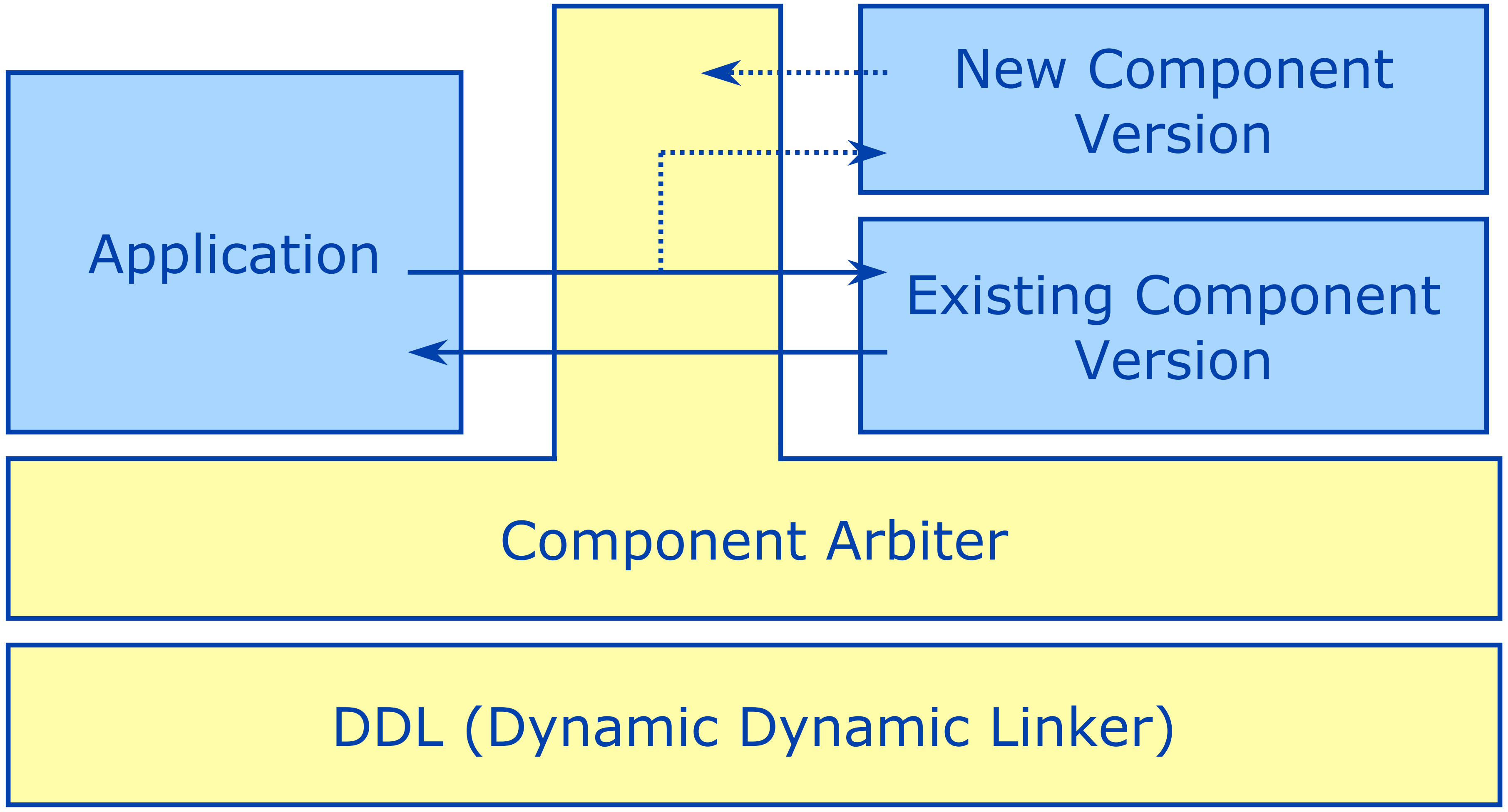


- Provide *replay scaffolding*
- Process *event log*
  - Create classes
  - Replay interactions

DDL: Dynamic Dynamic Linker

- Enables dynamic wrapper binding, and reconfiguration
- Harness for C++ captures:
  - incoming method invocations and returns
  - constructors and destructors
  - outgoing method/function invocations

DDL Online Monitoring



Conclusion

- MonDE for safe deployment of new versions
- Offline or online techniques possible
  - SCARPE and DDL

Open Issues

- Definition of oracles
  - What is a failure?
  - How can we filter?
- Identification of boundaries
  - Currently, hammocks, but other approaches possible (e.g., analyze how much flows across i/f, select low-flow cuts)
- Optimization of capture/interception
- Privacy issues

Questions?