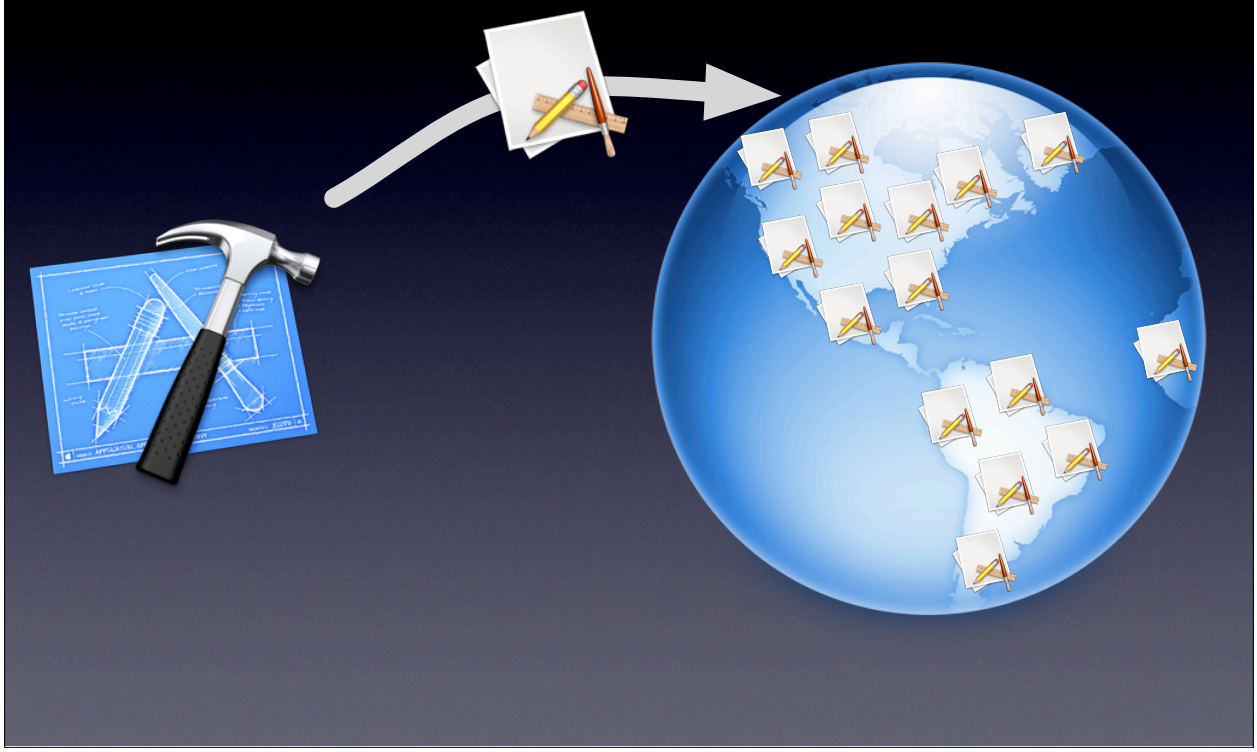
The GAMMA Project

Jim Clause

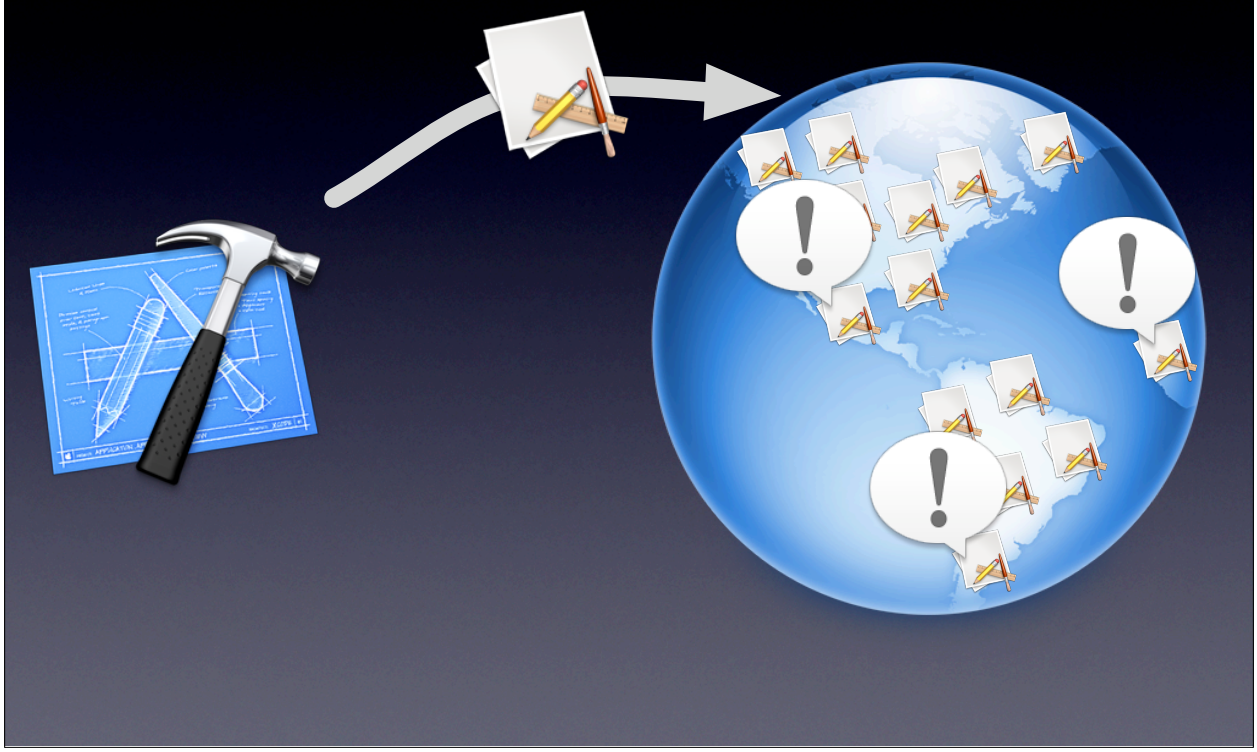
Overall picture

Overall picture

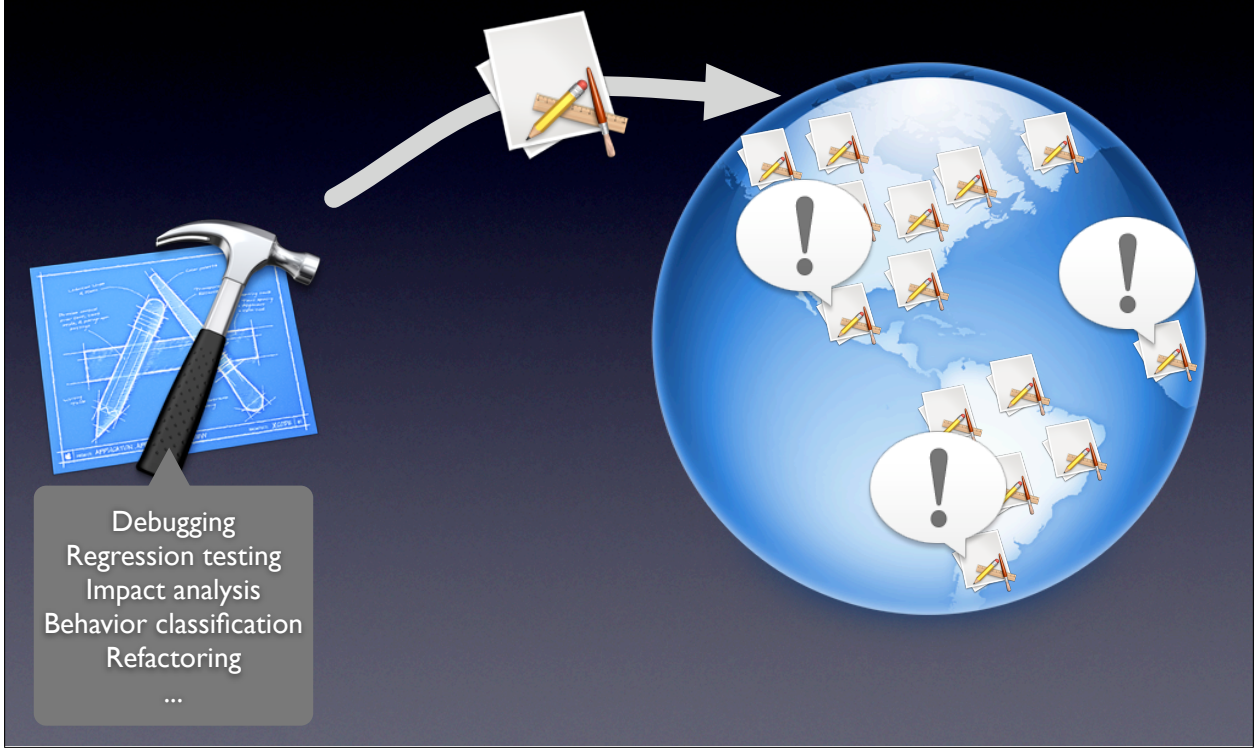
Overall picture



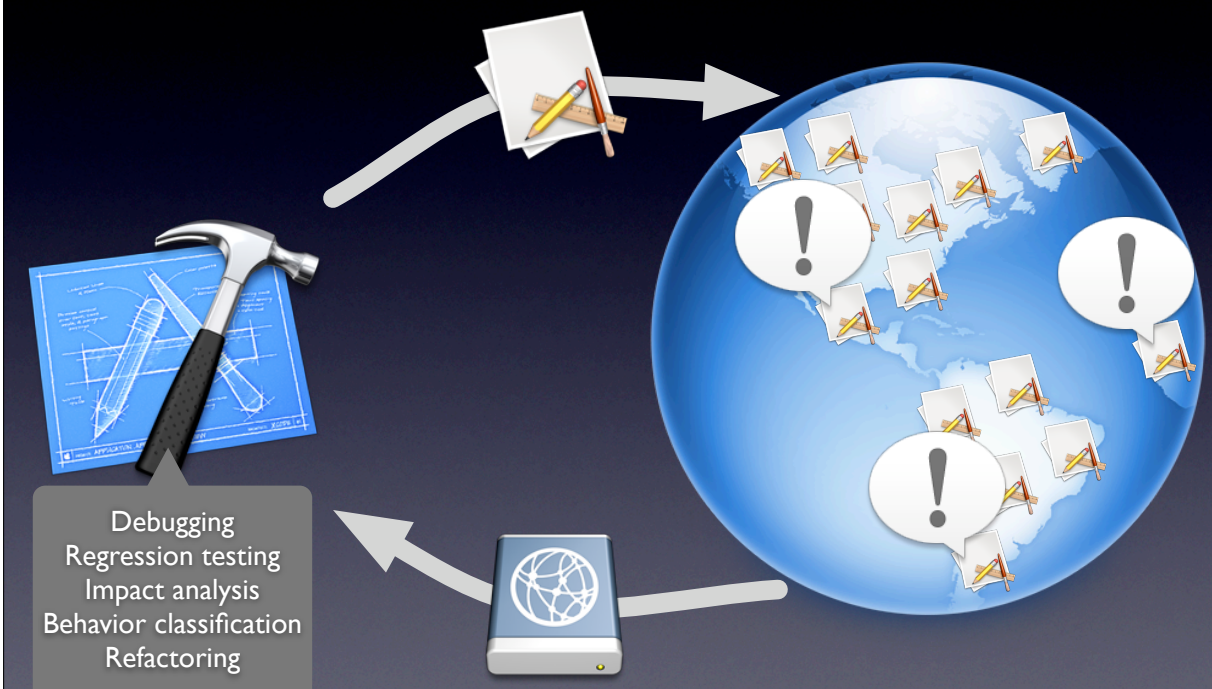
Overall picture



Overall picture

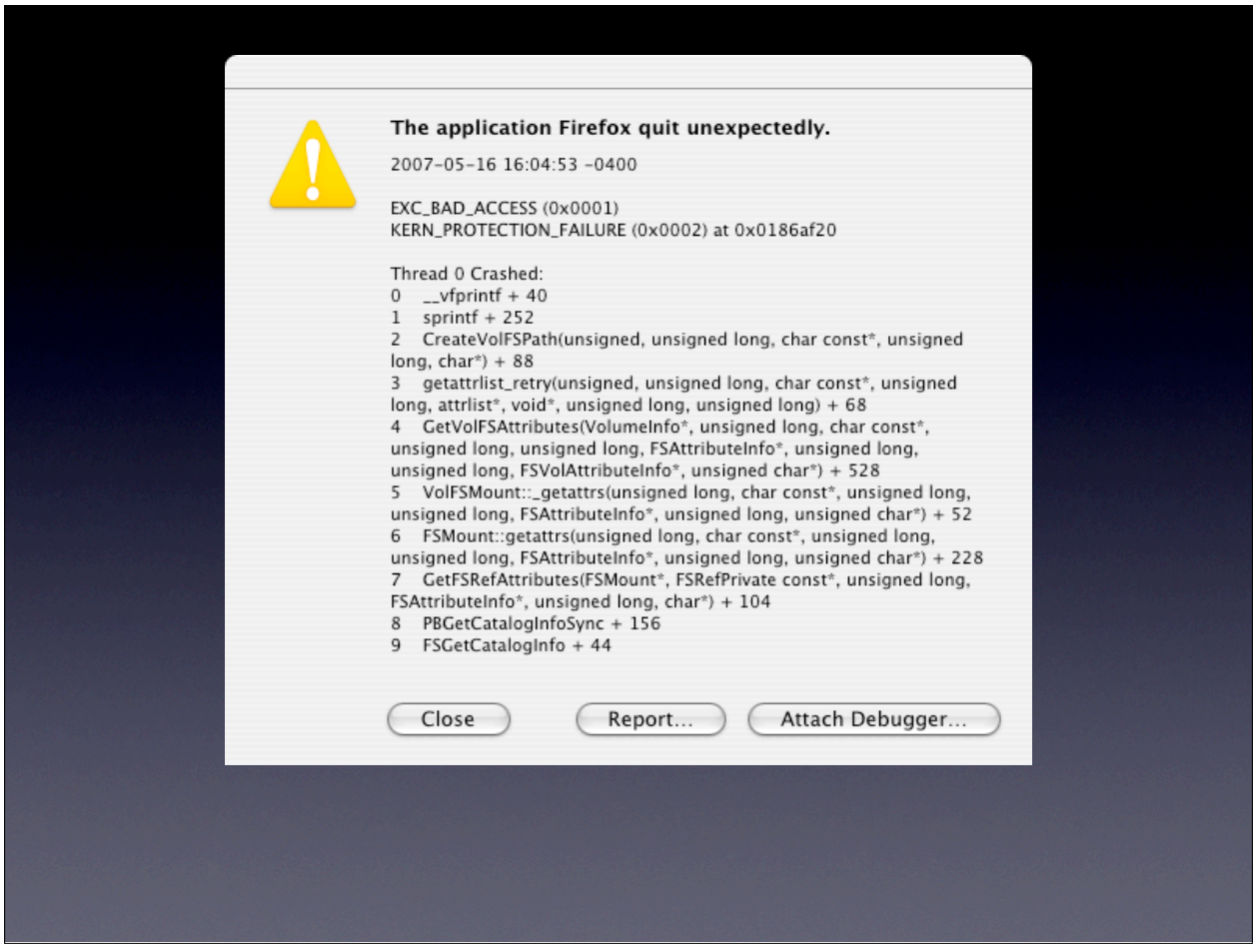


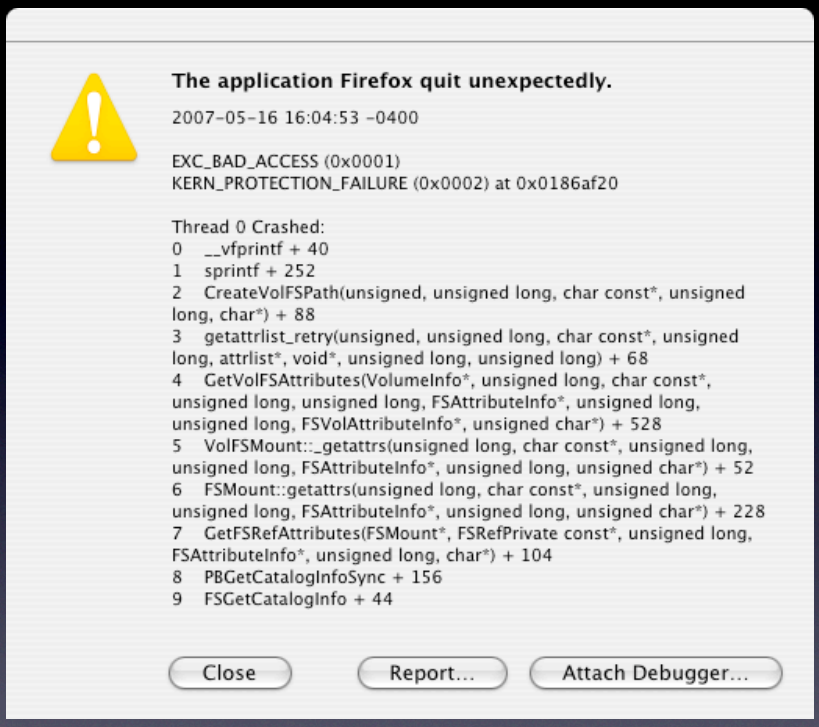
Overall picture



Debugging
Regression testing
Impact analysis
Behavior classification
Refactoring
...


College of Computing
http://www.cc.gatech.edu/
Getting Started
Latest Headlines
Georgia Tech
College of Computing
Defining the New Face of Computing
search...
GO
Prospective Students
Current Students
Alumni
Industry
Media
About the College
Academics
Research
People
Student Life
Partnerships
News
Events
<< MAY 2007 >>
S M T W T F S
View This Week's Events
View This Months's Events
Featured Events
Robocup 2007
June 30 - July 10
Details
Defining the New Face of Computing
News in Brief for May 16, 2007
(05/11/07)
Caring Technologies Receives $1 Million from NIH for Autism Caring Technologies / TalkAutism of Boise, ID, collaborated with College of Computing Associate Professor Gregory Abowd to develop and patent a digital video recording system that lets professionals, caregivers and parents uniquely capture and then securely replay the last few moments before a behavior of interest or concern of a child with autism. Download Article
(05/10/07)
College of Computing Students Win Industry Awards for Creating Next-Generation Mobile Applications
Fifteen students at The Georgia Institute of Technology, including eleven from the College of Computing, were awarded $100,000 in cash prizes for creating next-generation mobile applications as part of the 2007 IMS Research Competition. More
Coming July 2007...
World's Largest Research Robot Competition Hosted by Georgia Tech
Georgia Tech ROBOCUP 2007 ATLANTA
More
College of Computing Announces Creation of School of Computer Science and School of Interactive Computing
More
Done


The application Firefox quit unexpectedly.
2007-05-16 16:04:53 -0400
EXC_BAD_ACCESS (0x0001)
KERN_PROTECTION_FAILURE (0x0002) at 0x0186af20
Thread 0 Crashed:
0 __vfprintf + 40
1 sprintf + 252
2 CreateVolFSPath(unsigned, unsigned long, char const*, unsigned long, char*) + 88
3 getattrlist_retry(unsigned, unsigned long, char const*, unsigned long, attrlist*, void*, unsigned long, unsigned long) + 68
4 GetVolFSAttributes(VolumeInfo*, unsigned long, char const*, unsigned long, unsigned long, FSAttributeInfo*, unsigned long, unsigned long, FSVolAttributeInfo*, unsigned char*) + 528
5 VolFSMount::_getattrs(unsigned long, char const*, unsigned long, unsigned long, FSAttributeInfo*, unsigned long, unsigned long, unsigned char*) + 52
6 FSMount::getattrs(unsigned long, char const*, unsigned long, unsigned long, FSAttributeInfo*, unsigned long, unsigned long, unsigned char*) + 228
7 GetFSRefAttributes(FSMount*, FSRefPrivate const*, unsigned long, FSAttributeInfo*, unsigned long, char*) + 104
8 PBGetCatalogInfoSync + 156
9 FSGetCatalogInfo + 44
Close
Report...
Attach Debugger...



Field failures: Anomalous behavior (or crashes) of deployed software that occur on user machines

Crash logs

User-provided information

Our solution

Our solution

Record

Our solution

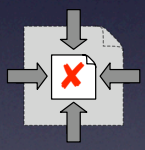
Record

Replay

Our solution

Record

Replay

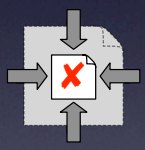


Minimize

Our solution

Record

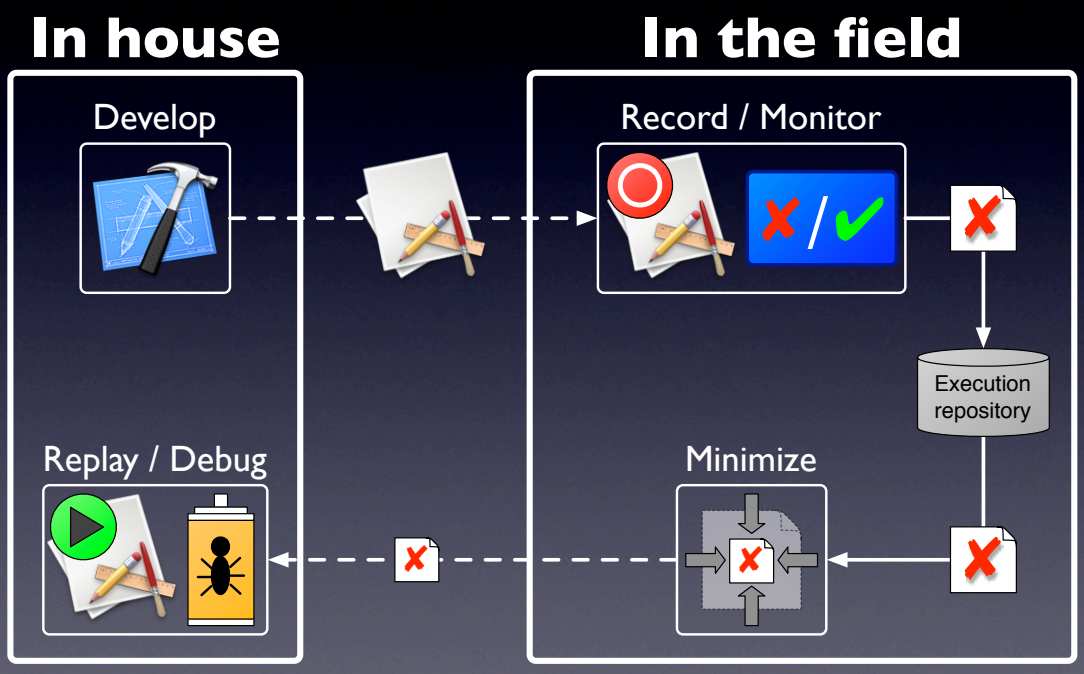
Replay



Minimize

Debug

Usage Scenario



Existing record / replay approaches

Deterministic debugging

(e.g. Chen et al. 01, King et al. 05, Narayanasamy et al. 05, Netzer and Weaver 94, Srinivasan et al. 04, VMWare)

- Replay an entire execution by recording every component of an application

Regression testing

(e.g. Elbaum et al. 06, Orso et al. 06, Orso and Kennedy 05, Saff et al. 05, Mercury WinRunner)

- Replay only a portion of an execution by recording events for specific subsystems

Both types of technique are not amenable to minimization and may cause unacceptable overhead

Outline

- Our technique
  - record / replay
  - minimization
- Empirical evaluation
- Conclusions
- Future work

Record & Replay

- Goal: develop an approach that has low overhead and is amenable to minimization
- Key insight: avoid focusing on low-level (internal) events
  - expensive (large number of events)
  - not amenable to minimization (high interdependence)

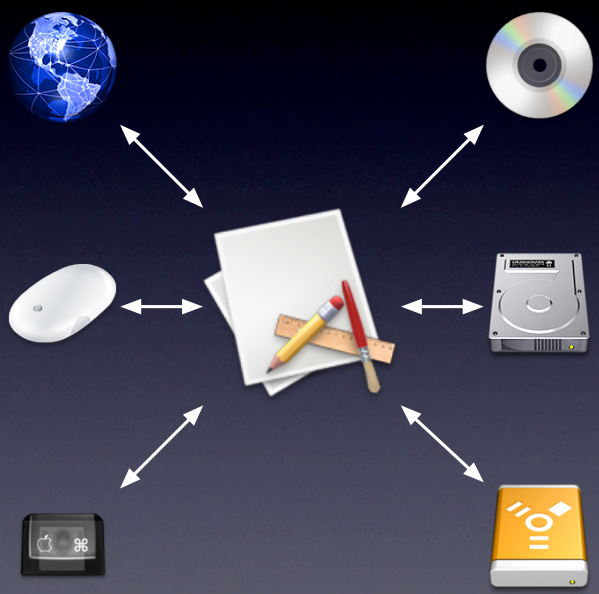
Record & Replay

- Goal: develop an approach that has low overhead and is amenable to minimization
- Key insight: avoid focusing on low-level (internal) events
  - expensive (large number of events)
  - not amenable to minimization (high interdependence)

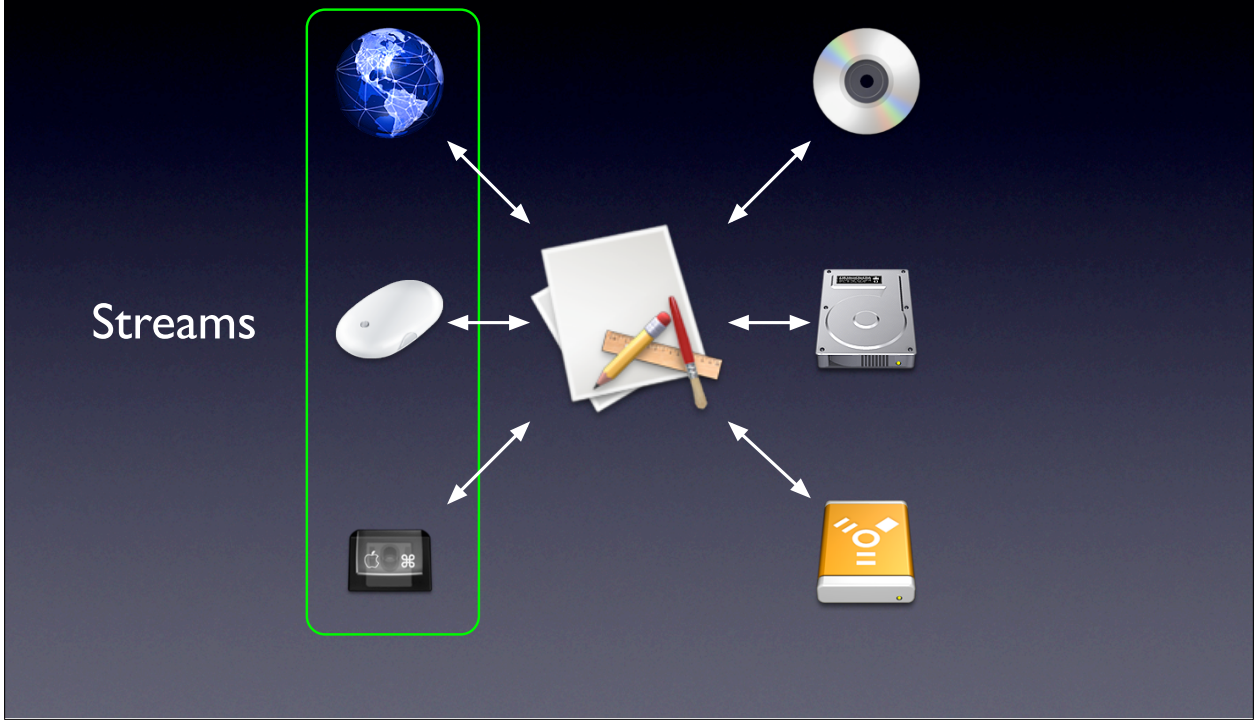
➡ Focus on high-level (external) interactions with the environment

- efficient (fewer, more "expensive" interactions)
- amenable to minimization (low interdependence)

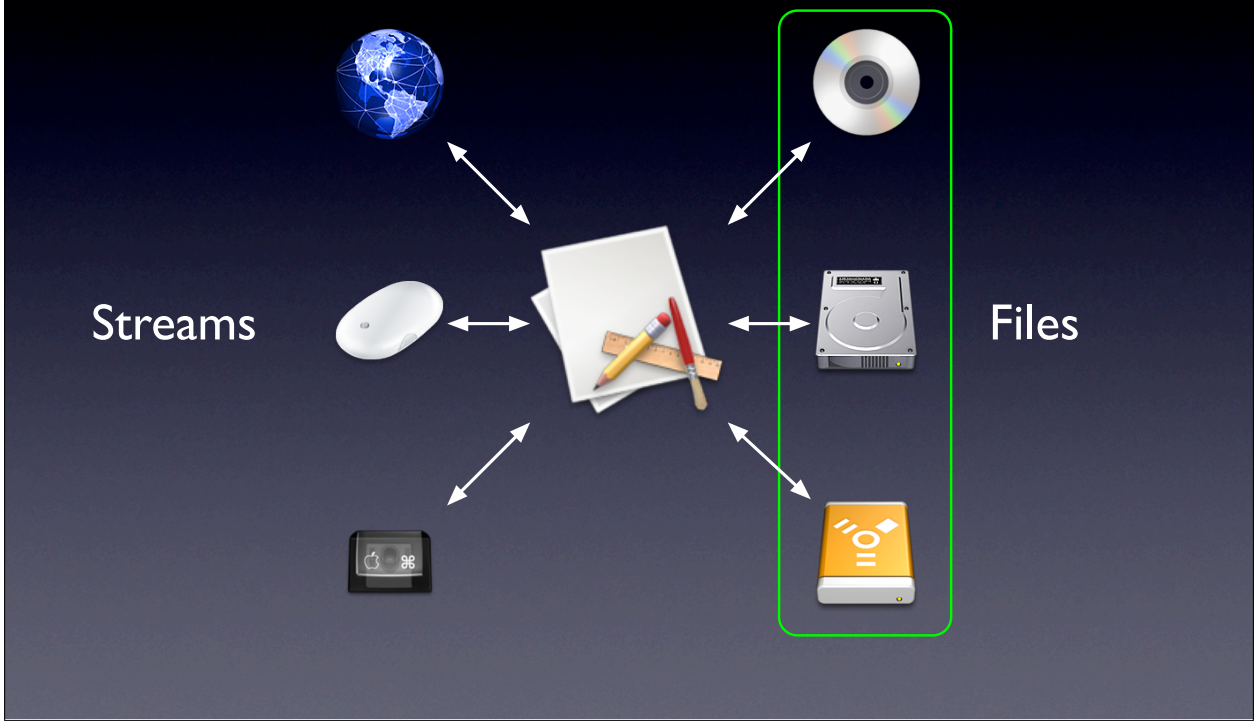
Environment interactions



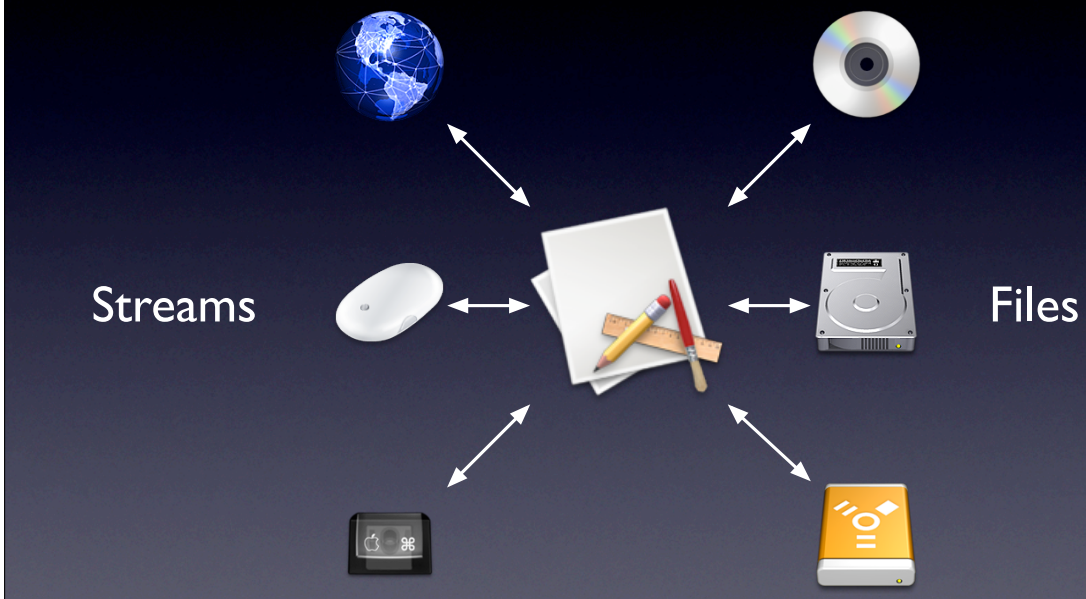
Environment interactions



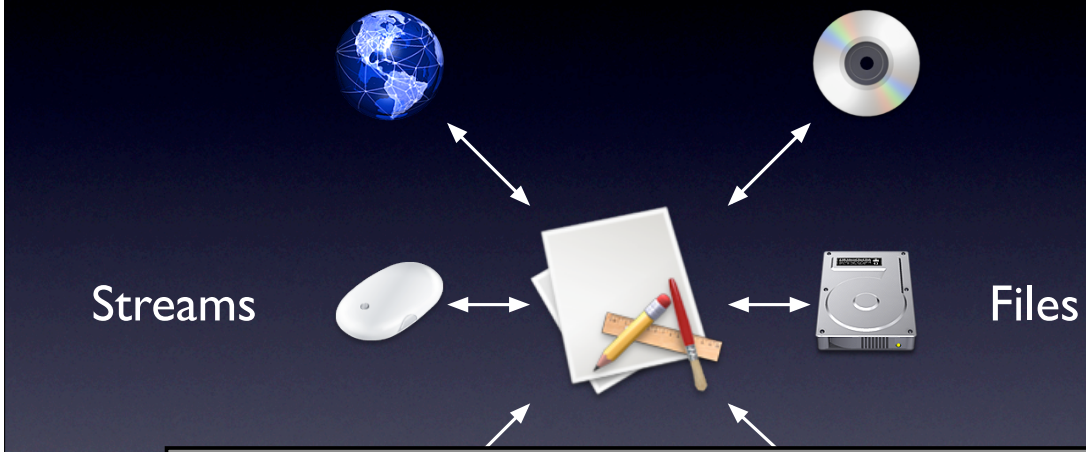
Environment interactions



Environment interactions



Environment interactions

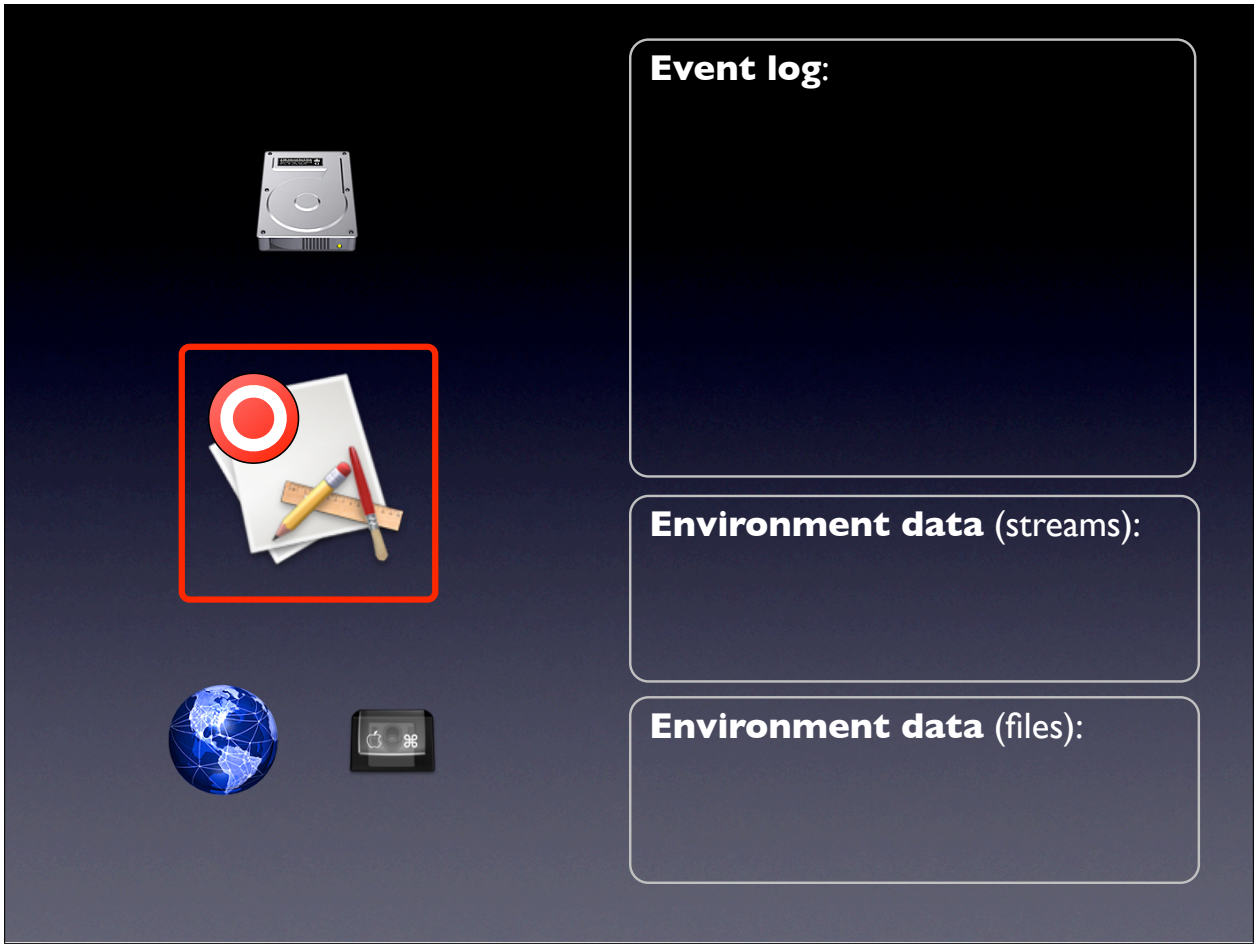


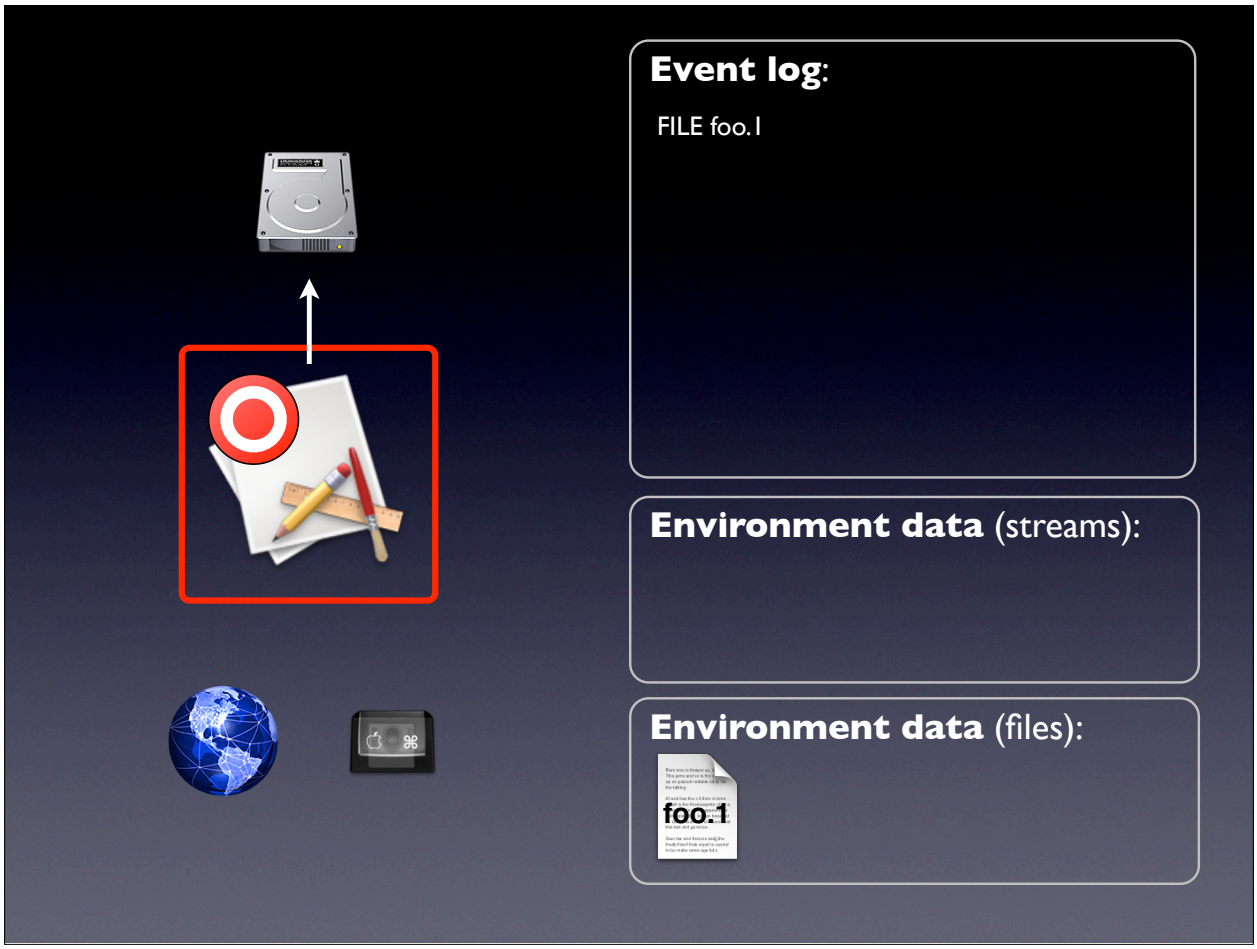
Interaction events:

FILE — interaction with a file

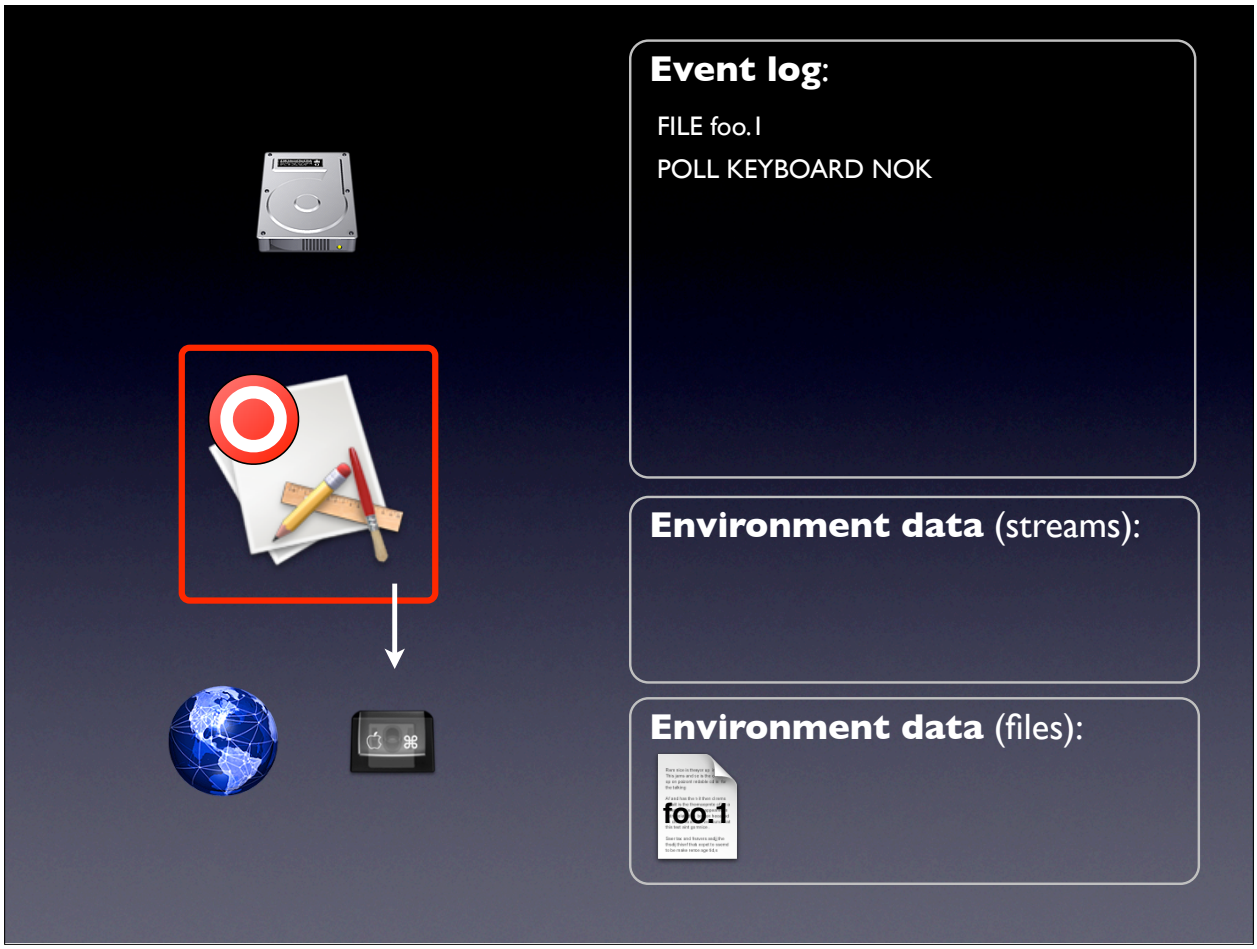
POLL — checks for availability of data on a stream

PULL — read data from a stream

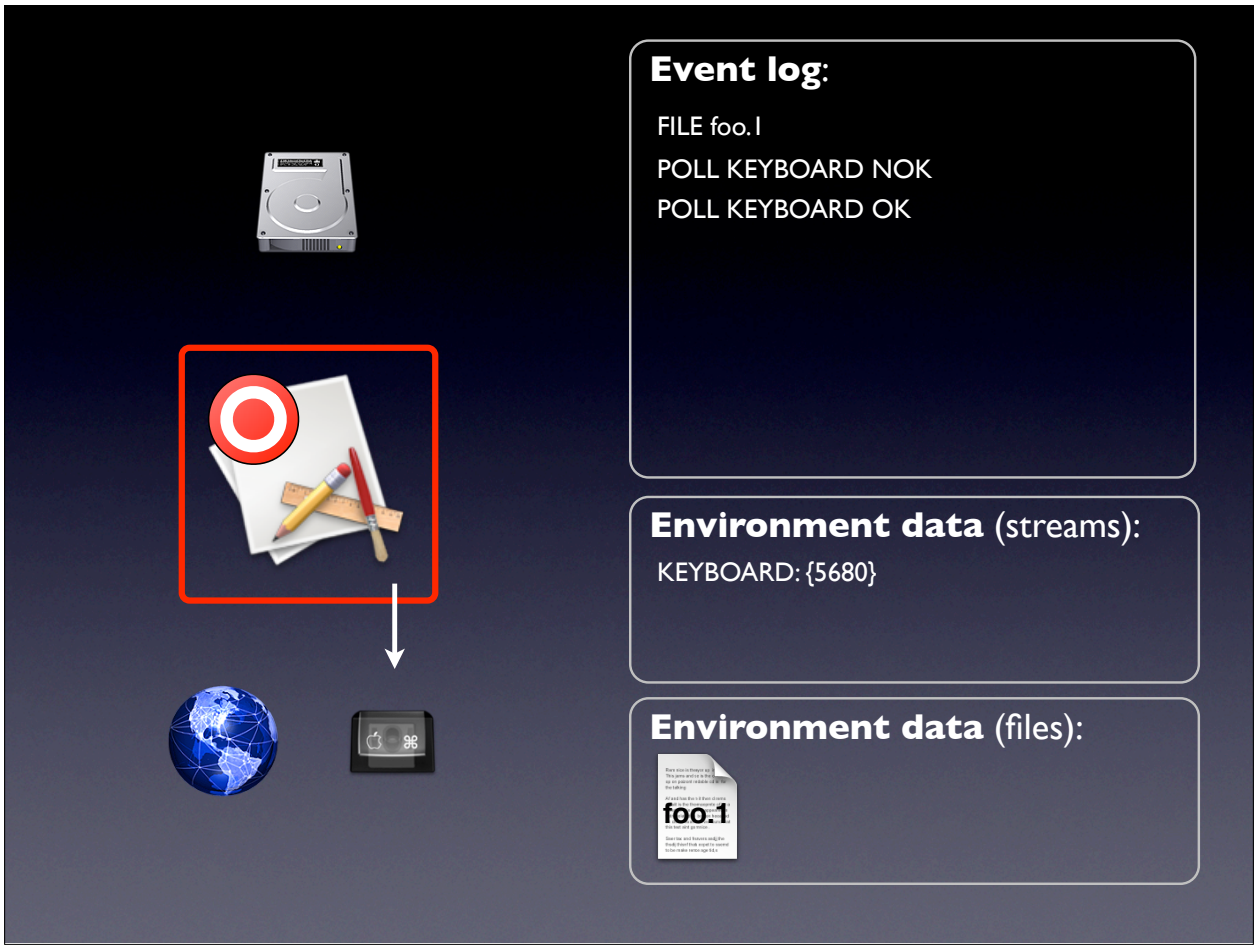

Event log:
Environment data (streams):
Environment data (files):

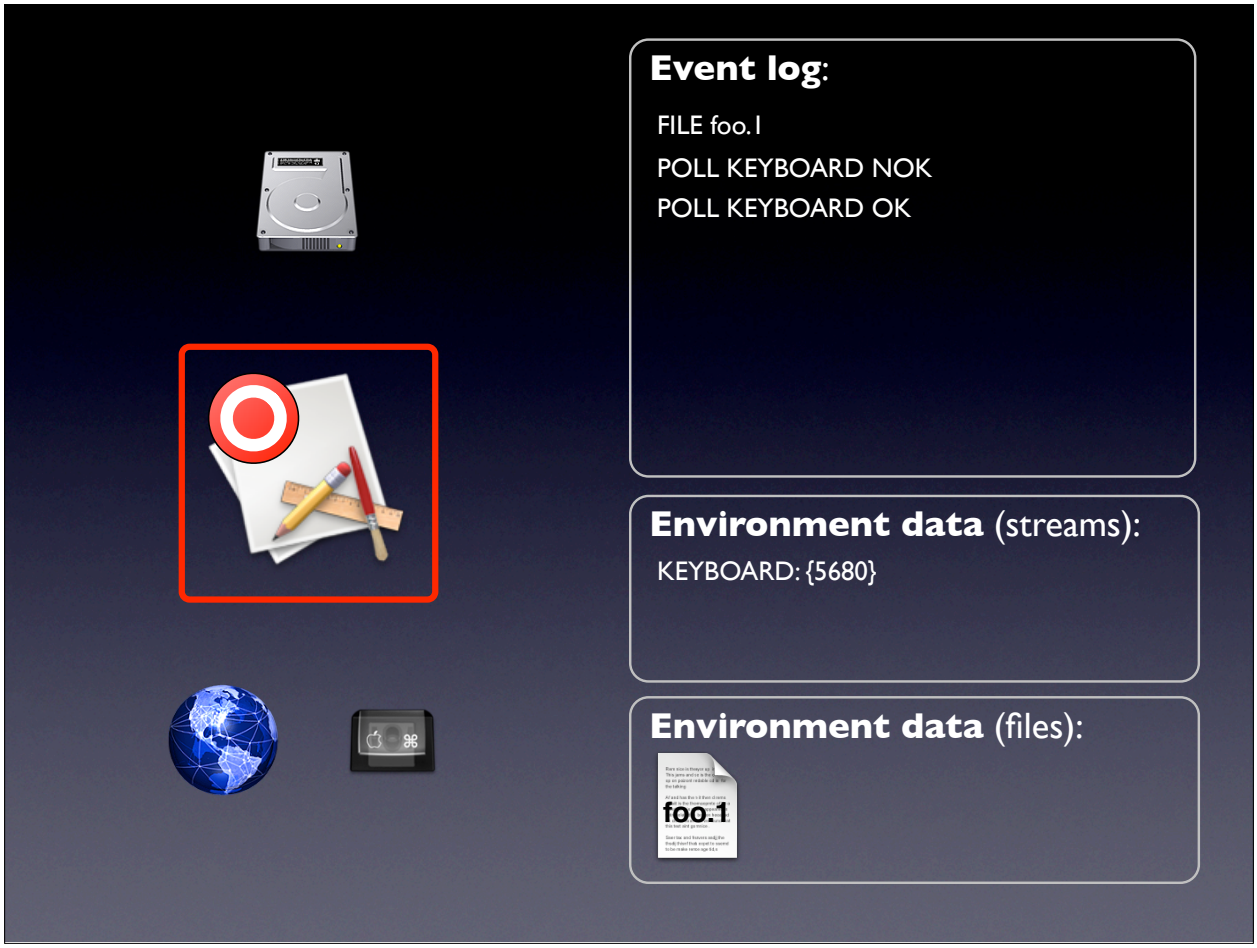

Event log:
FILE foo.1
Environment data (streams):
Environment data (files):
foo.1

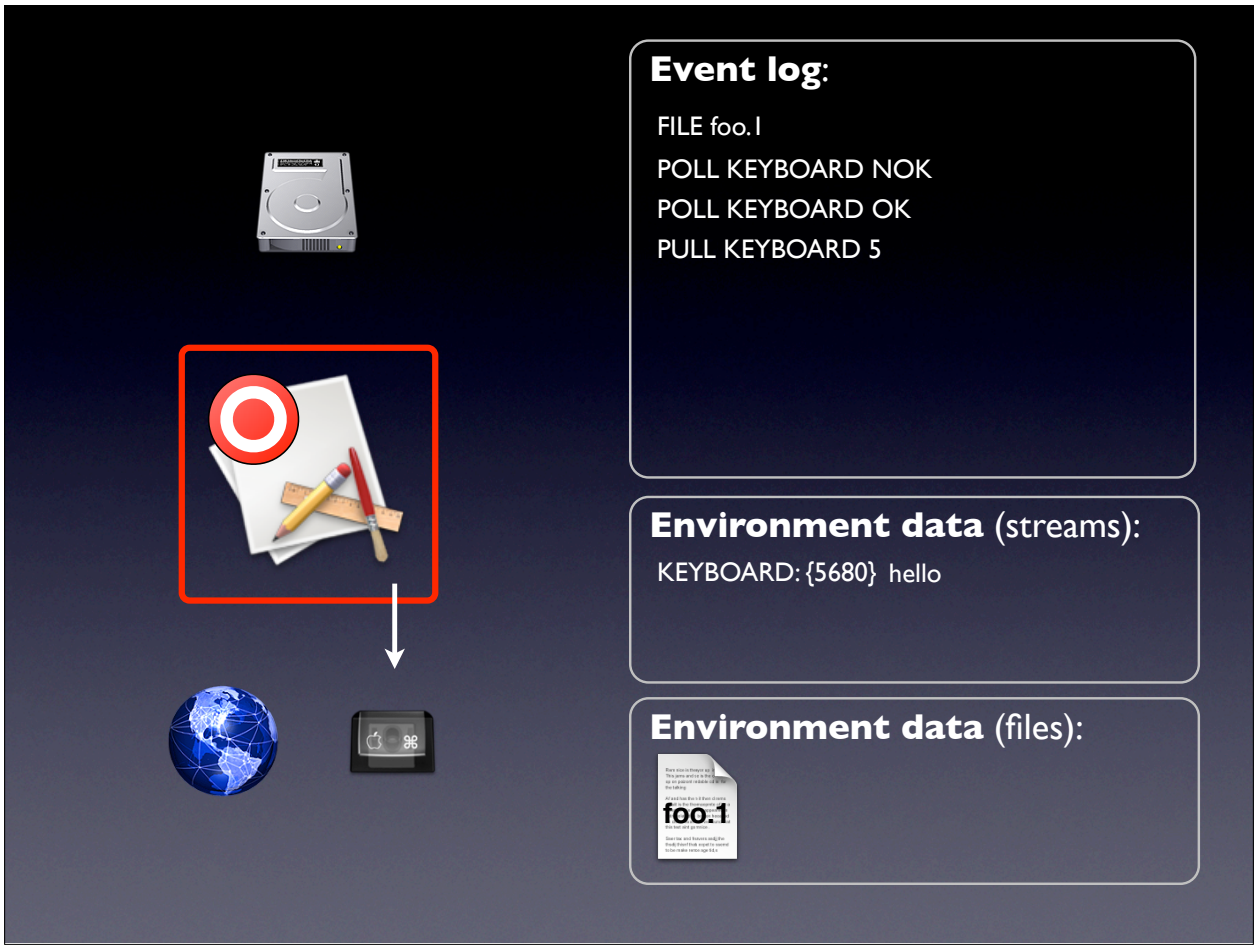

Event log:
FILE foo.1
Environment data (streams):
Environment data (files):
foo.1

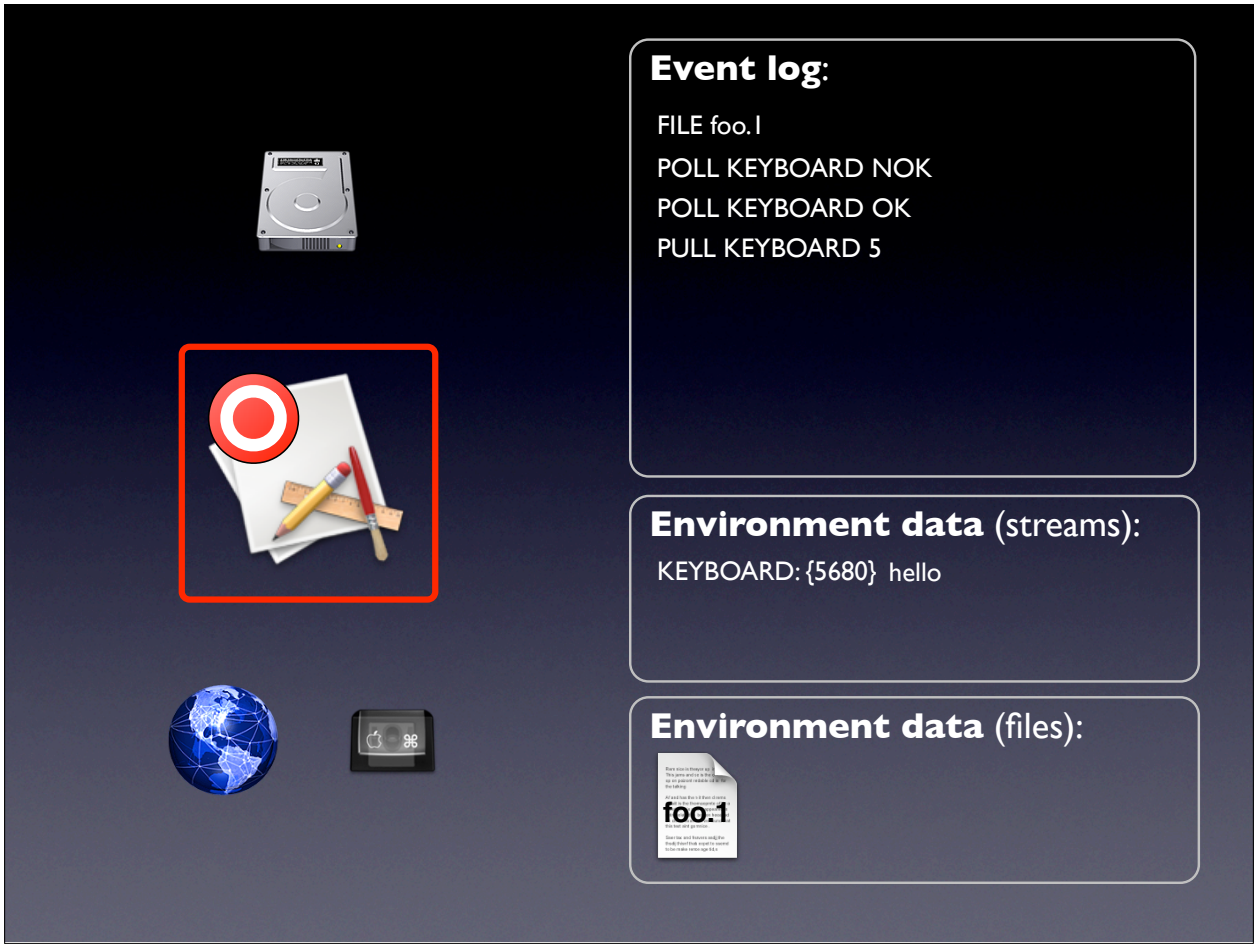

Event log:
FILE foo.1
POLL KEYBOARD NOK
Environment data (streams):
Environment data (files):
foo.1

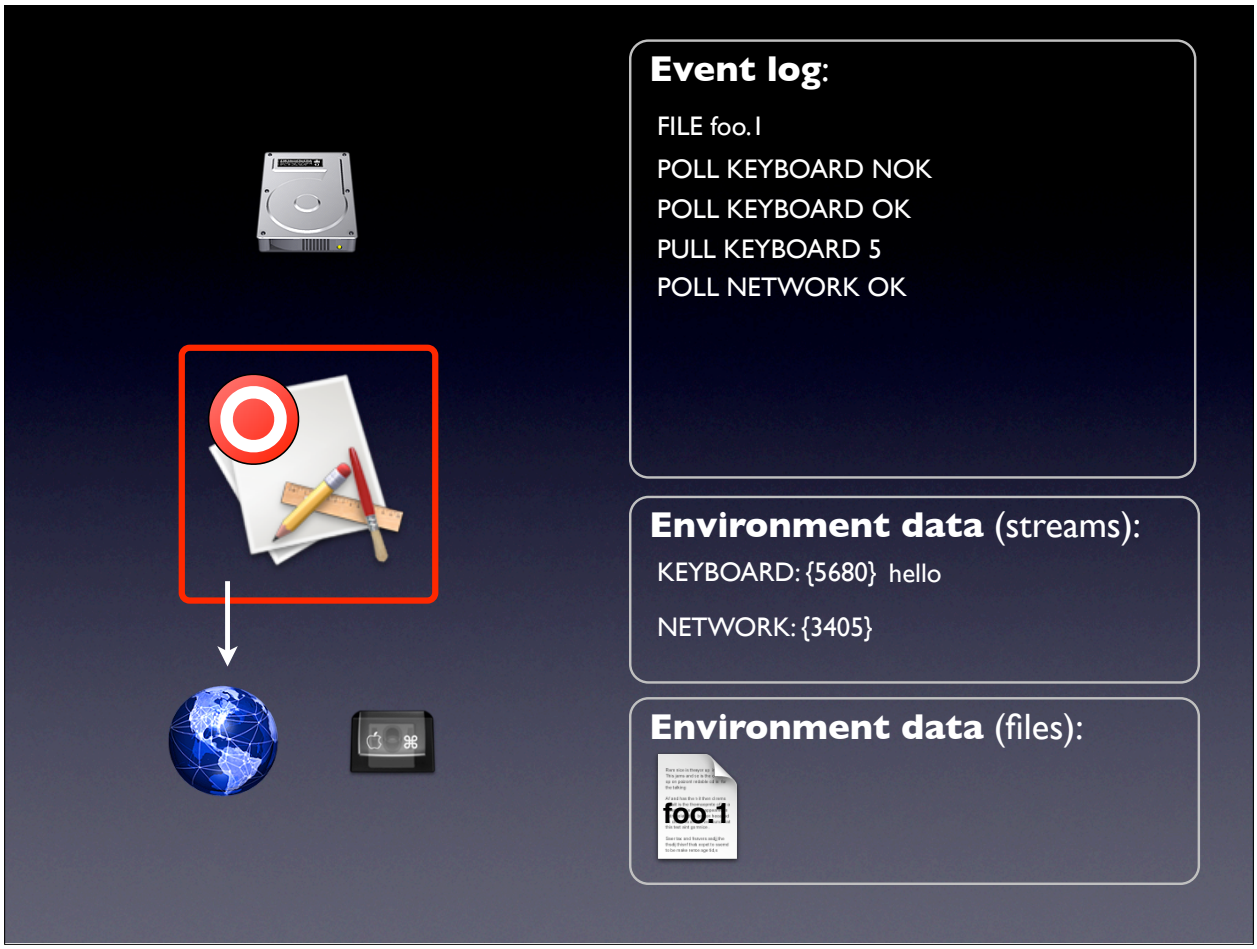

Event log:
FILE foo.1
POLL KEYBOARD NOK
Environment data (streams):
Environment data (files):
foo.1

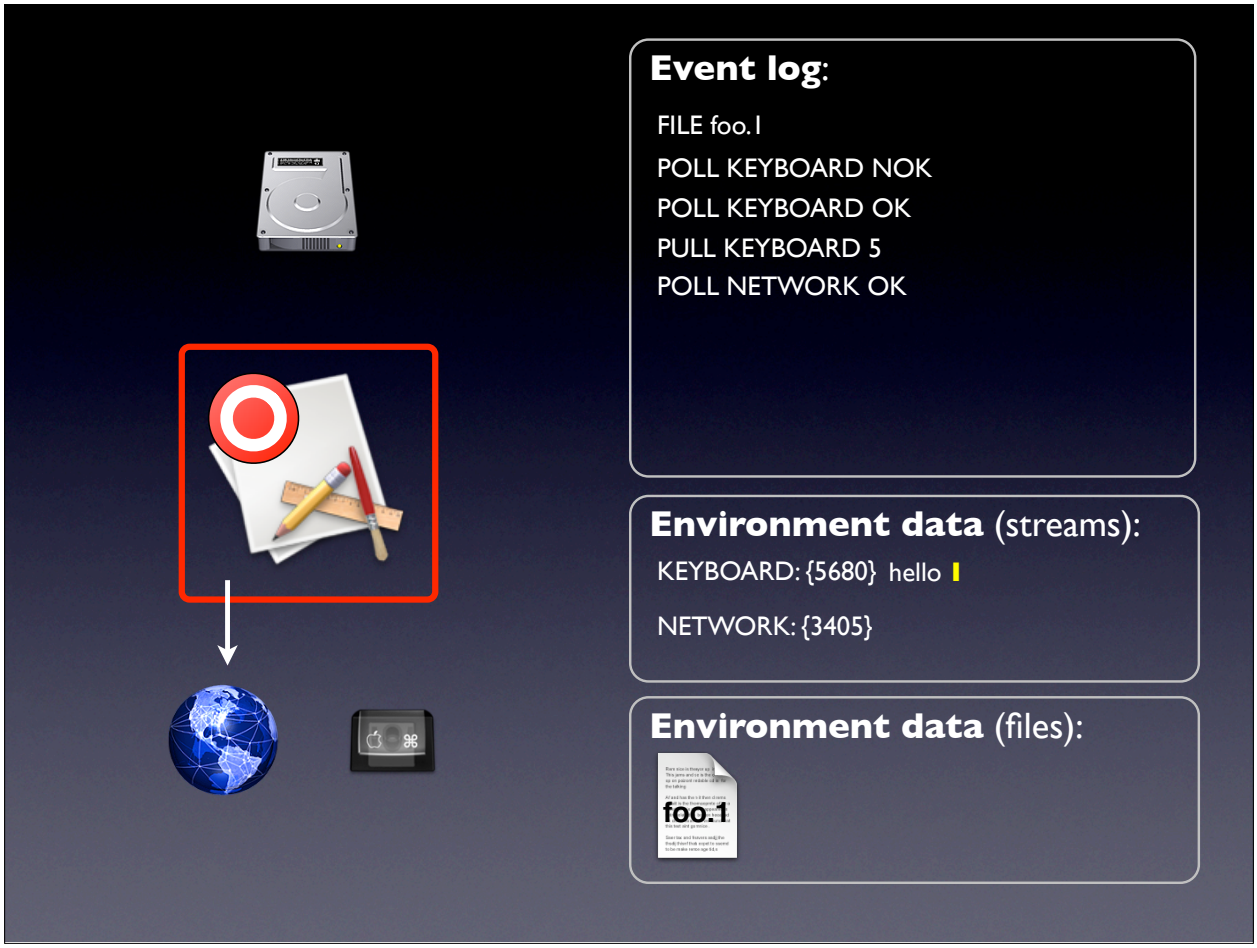

Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
Environment data (streams):
KEYBOARD: {5680}
Environment data (files):
foo.1

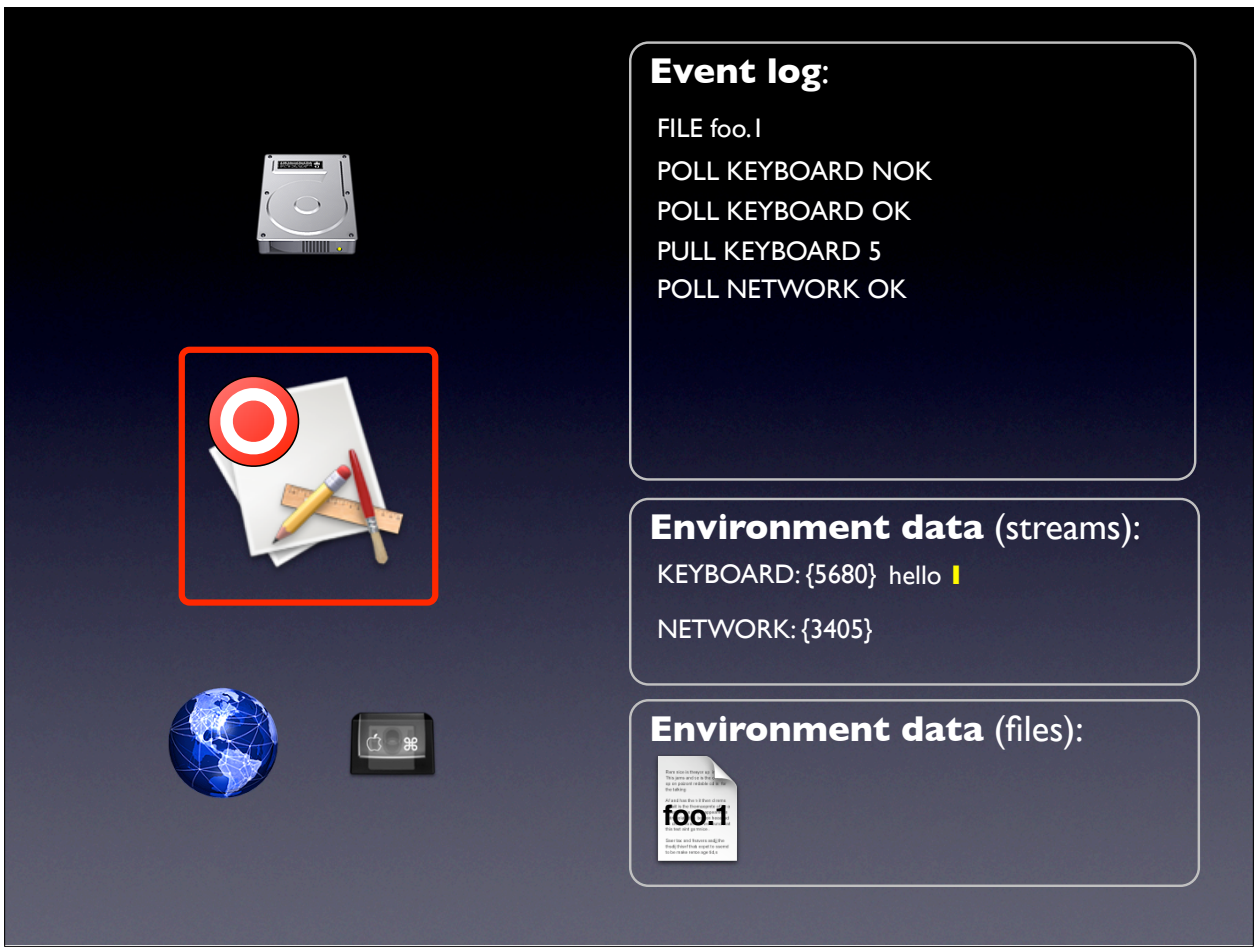

Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
Environment data (streams):
KEYBOARD: {5680}
Environment data (files):
foo.1

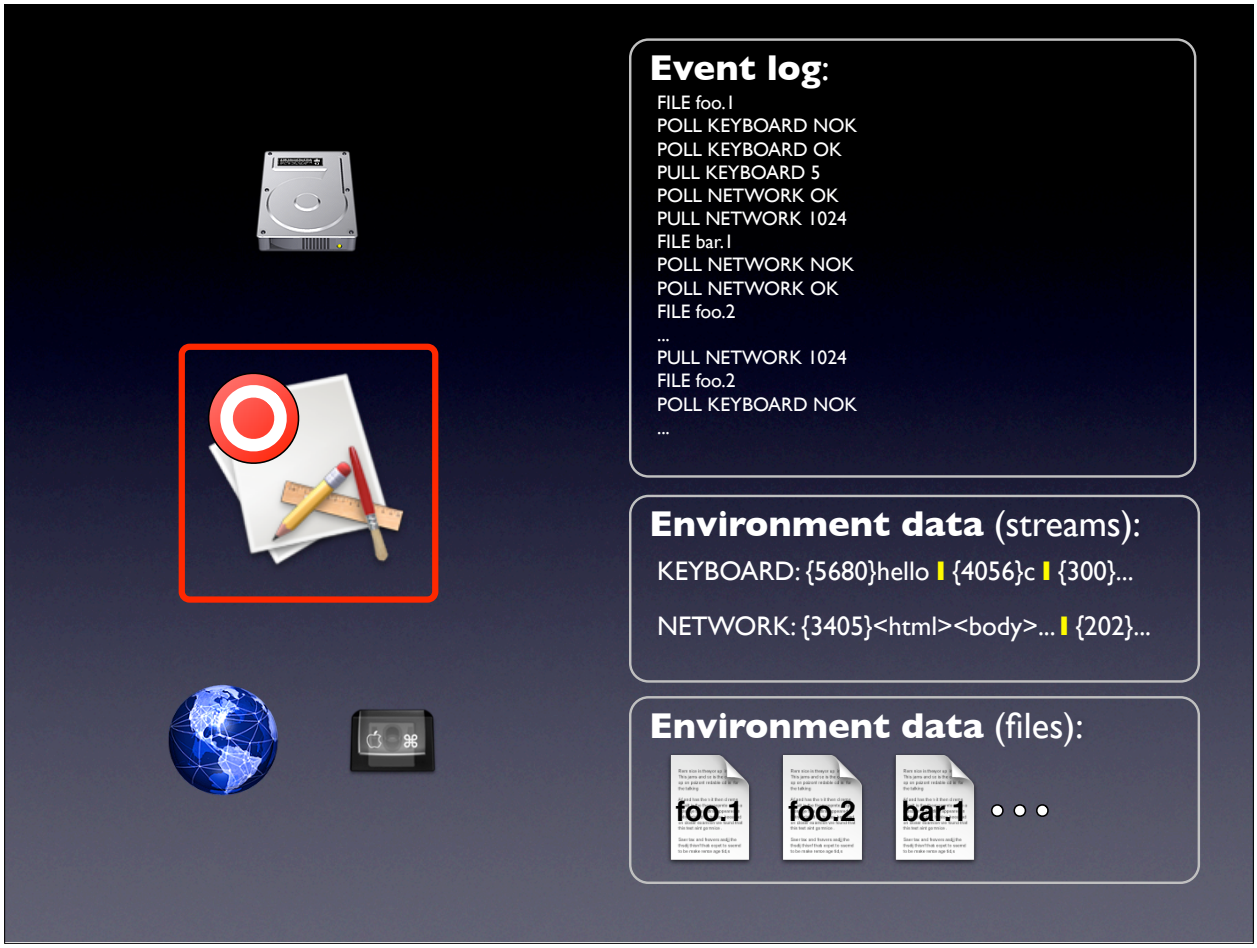

Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
Environment data (streams):
KEYBOARD: {5680} hello
Environment data (files):
foo.1

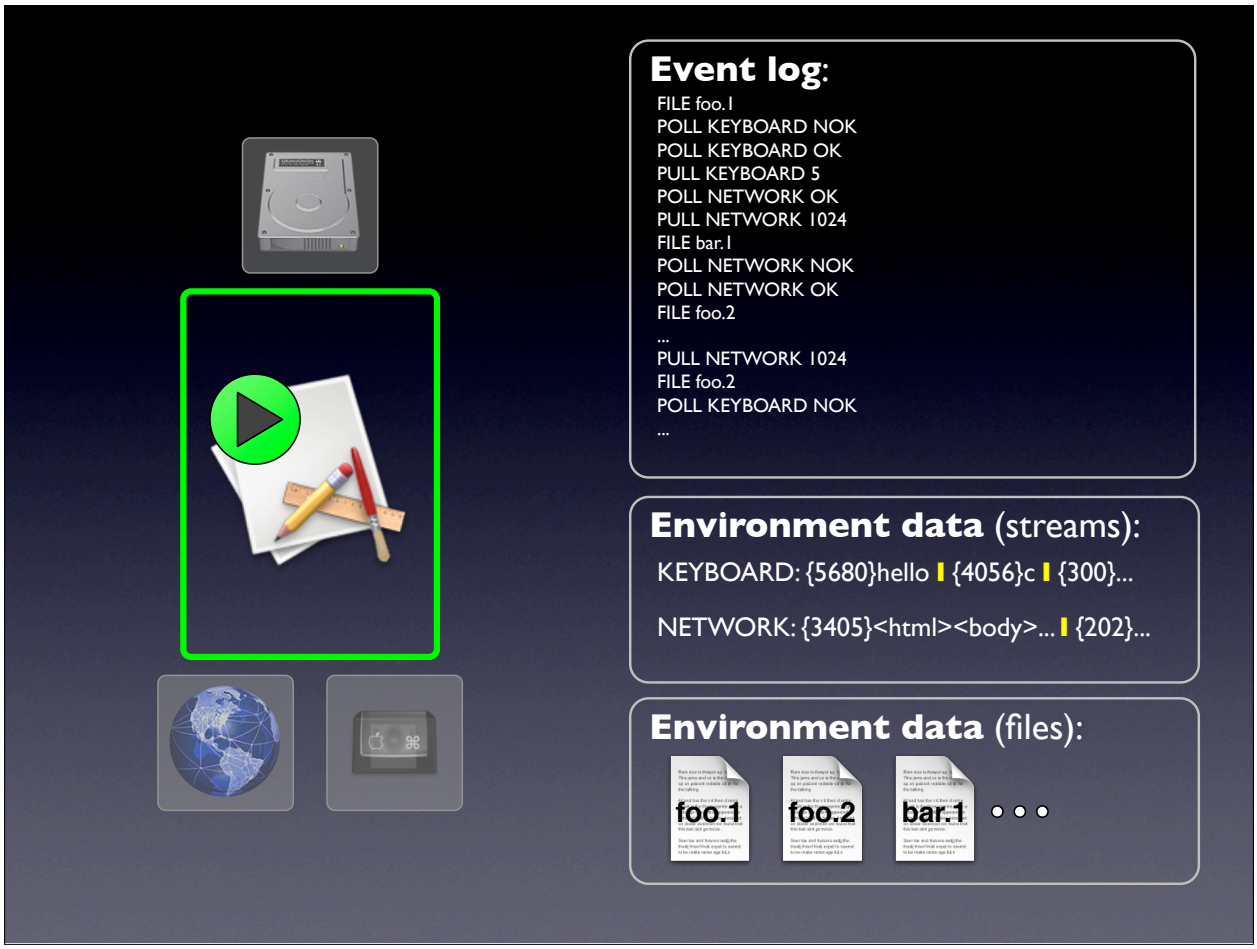

Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
Environment data (streams):
KEYBOARD: {5680} hello
Environment data (files):
foo.1

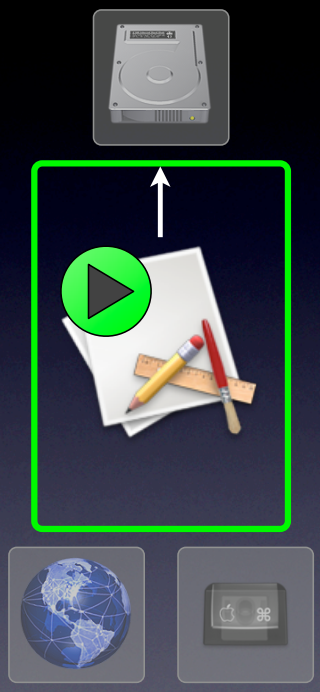

Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
Environment data (streams):
KEYBOARD: {5680} hello
NETWORK: {3405}
Environment data (files):
foo.1


Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
Environment data (streams):
KEYBOARD: {5680} hello
NETWORK: {3405}
Environment data (files):
foo.1


Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
Environment data (streams):
KEYBOARD: {5680} hello
NETWORK: {3405}
Environment data (files):
foo.1


Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL NETWORK NOK
POLL NETWORK OK
FILE foo.2
...
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK
...
Environment data (streams):
KEYBOARD: {5680}hello | {4056}c | {300}...
NETWORK: {3405}<html><body>... | {202}...
Environment data (files):
foo.1
foo.2
bar.1


Event log:
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL NETWORK NOK
POLL NETWORK OK
FILE foo.2
...
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK
...
Environment data (streams):
KEYBOARD: {5680}hello | {4056}c | {300}...
NETWORK: {3405}<html><body>... | {202}...
Environment data (files):
foo.1
foo.2
bar.1



Event log:

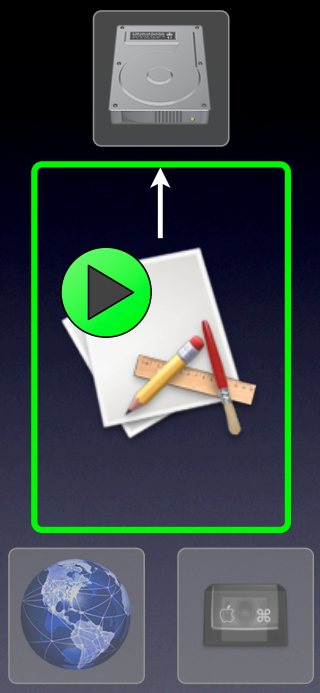
FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL NETWORK NOK
POLL NETWORK OK
FILE foo.2
...
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK
...

Environment data (streams):

KEYBOARD: {5680}hello ▮ {4056}c ▮ {300}...

NETWORK: {3405}<html><body>... ▮ {202}...

Environment data (files):



Event log:

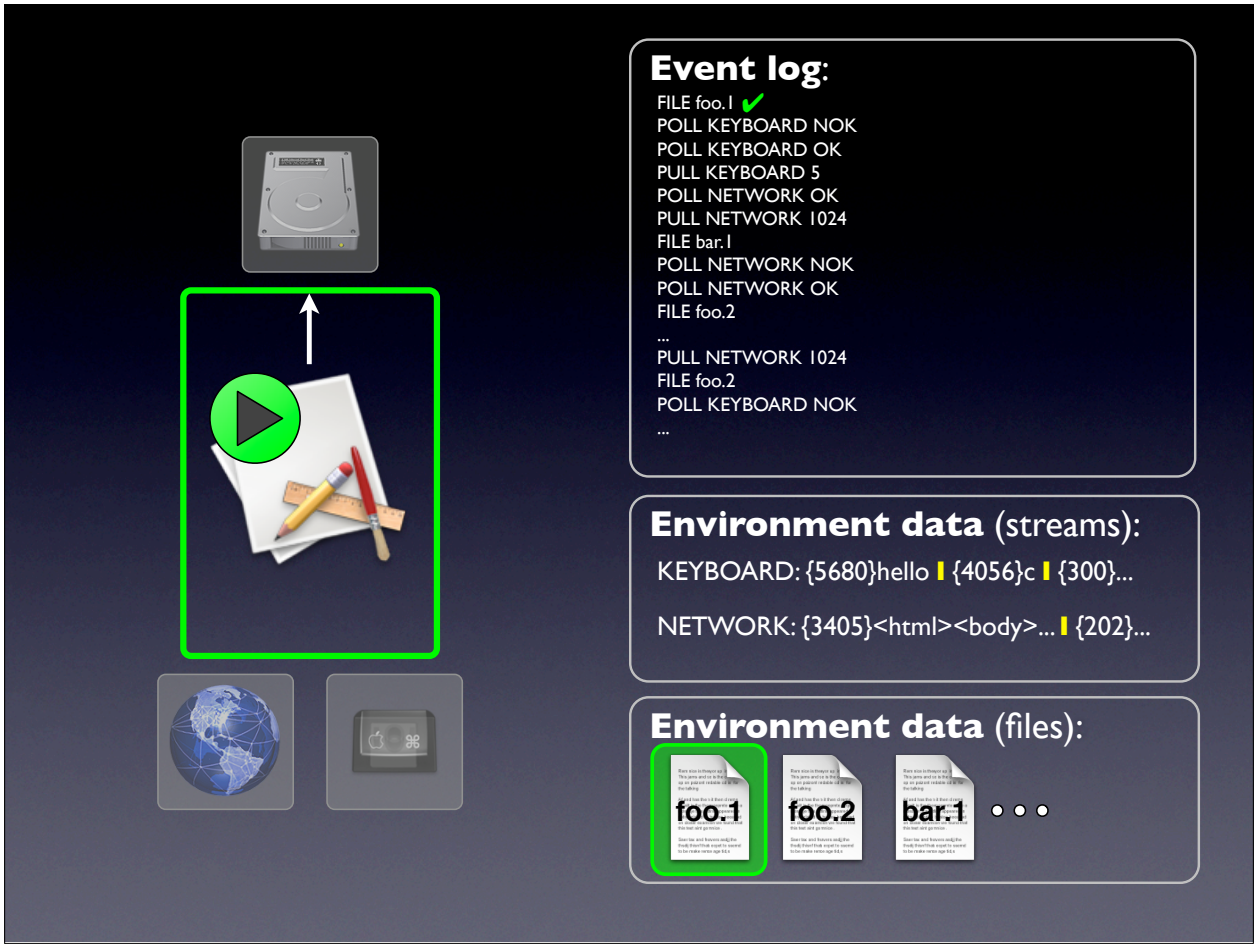
FILE foo.1 ✔
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL NETWORK NOK
POLL NETWORK OK
FILE foo.2
...
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK
...

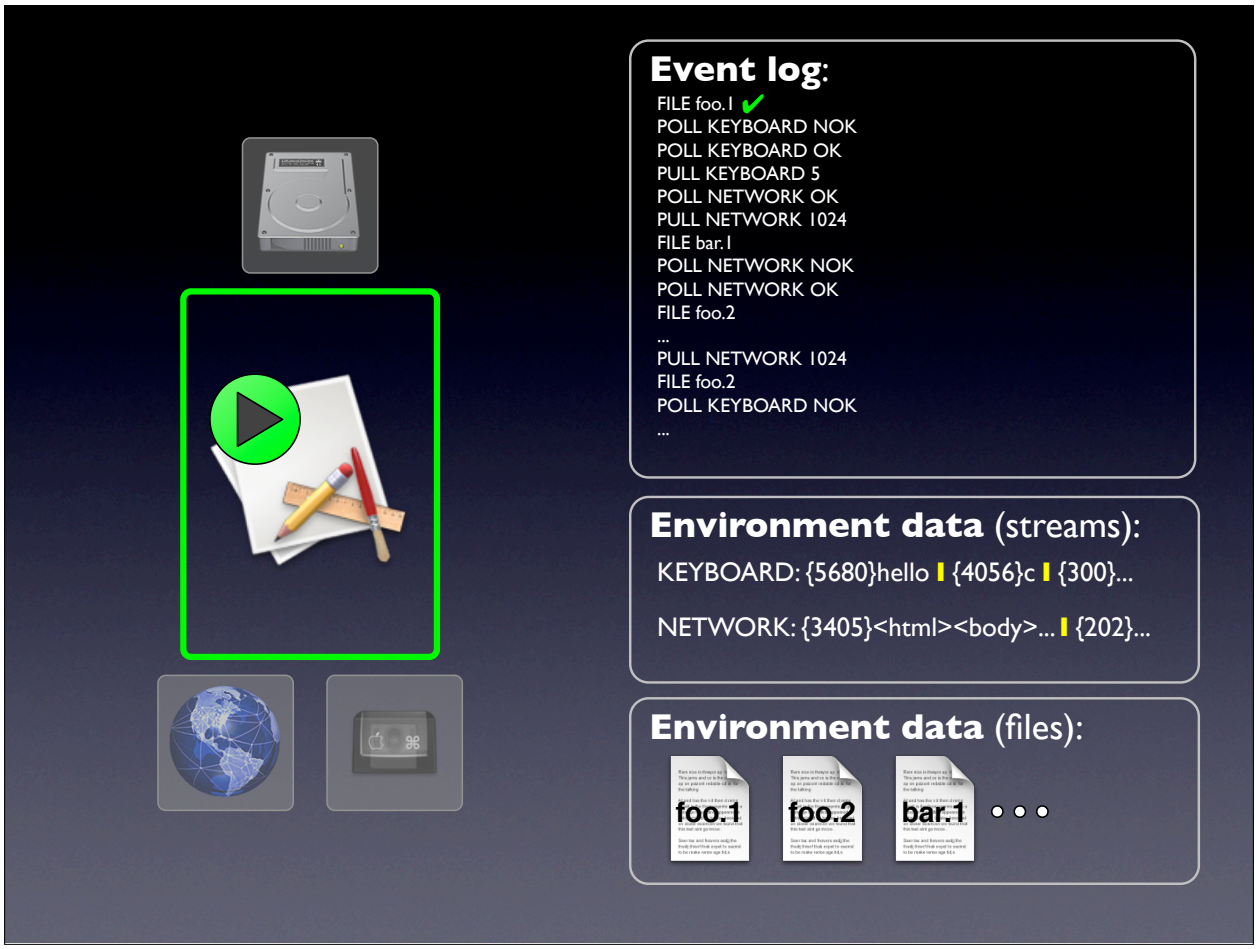
Environment data (streams):

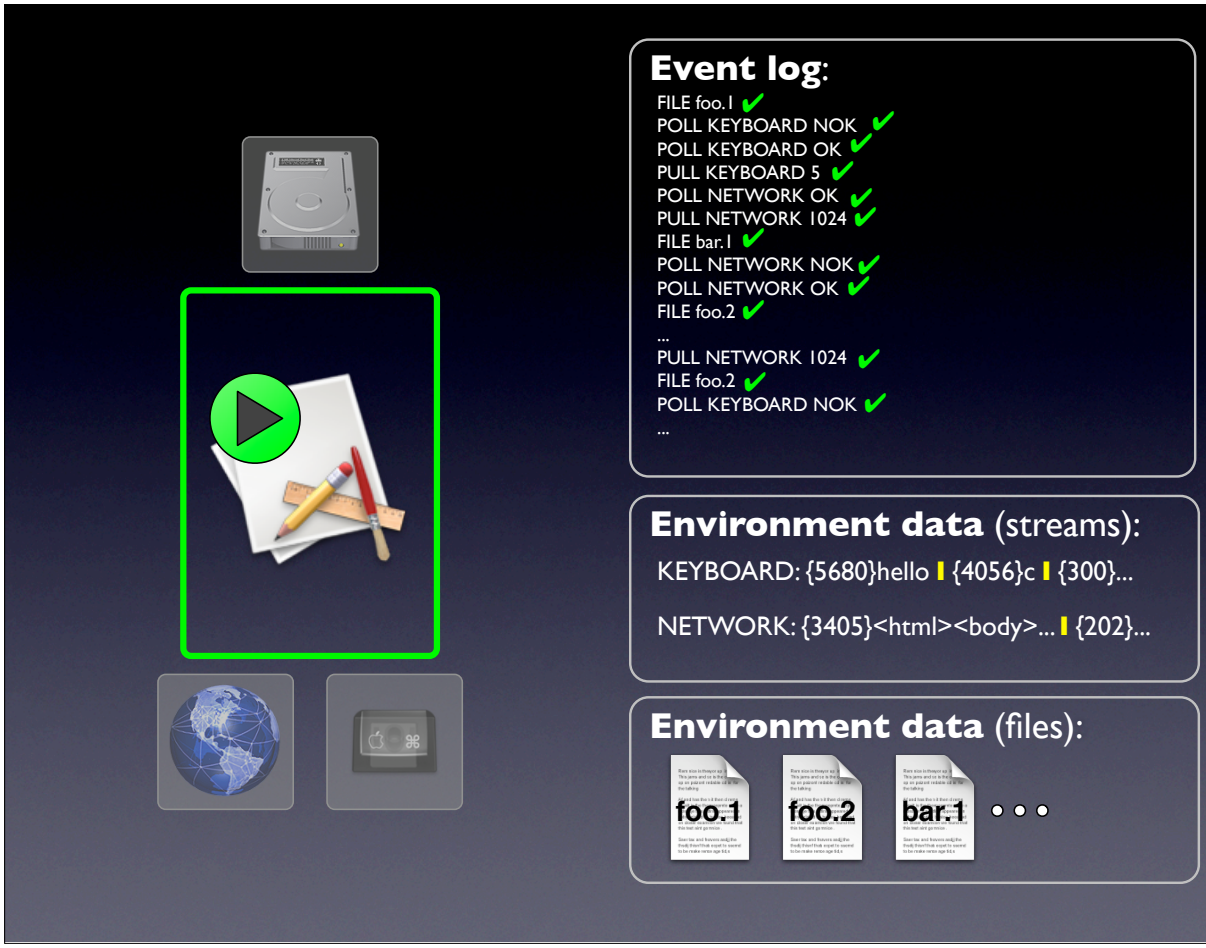
KEYBOARD: {5680}hello ▮ {4056}c ▮ {300}...

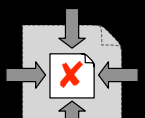
NETWORK: {3405}<html><body>... ▮ {202}...

Environment data (files):


Event log:
FILE foo.1 ✔
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL NETWORK NOK
POLL NETWORK OK
FILE foo.2
...
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK
...
Environment data (streams):
KEYBOARD: {5680}hello | {4056}c | {300}...
NETWORK: {3405}<html><body>... | {202}...
Environment data (files):
foo.1
foo.2
bar.1

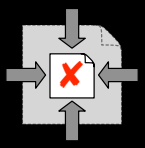

Event log:
FILE foo.1 ✔
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 5
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL NETWORK NOK
POLL NETWORK OK
FILE foo.2
...
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK
...
Environment data (streams):
KEYBOARD: {5680}hello | {4056}c | {300}...
NETWORK: {3405}<html><body>... | {202}...
Environment data (files):
foo.1
foo.2
bar.1





Minimize

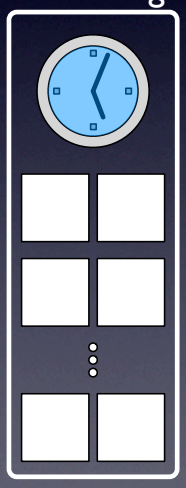
Goal: focus debugging effort

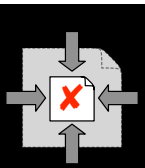


Minimize

Goal: focus debugging effort

Execution recording

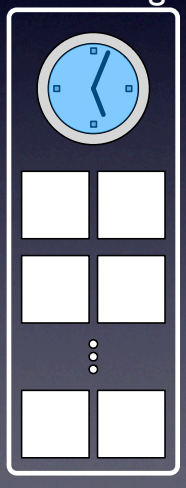




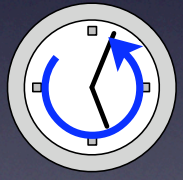
Minimize

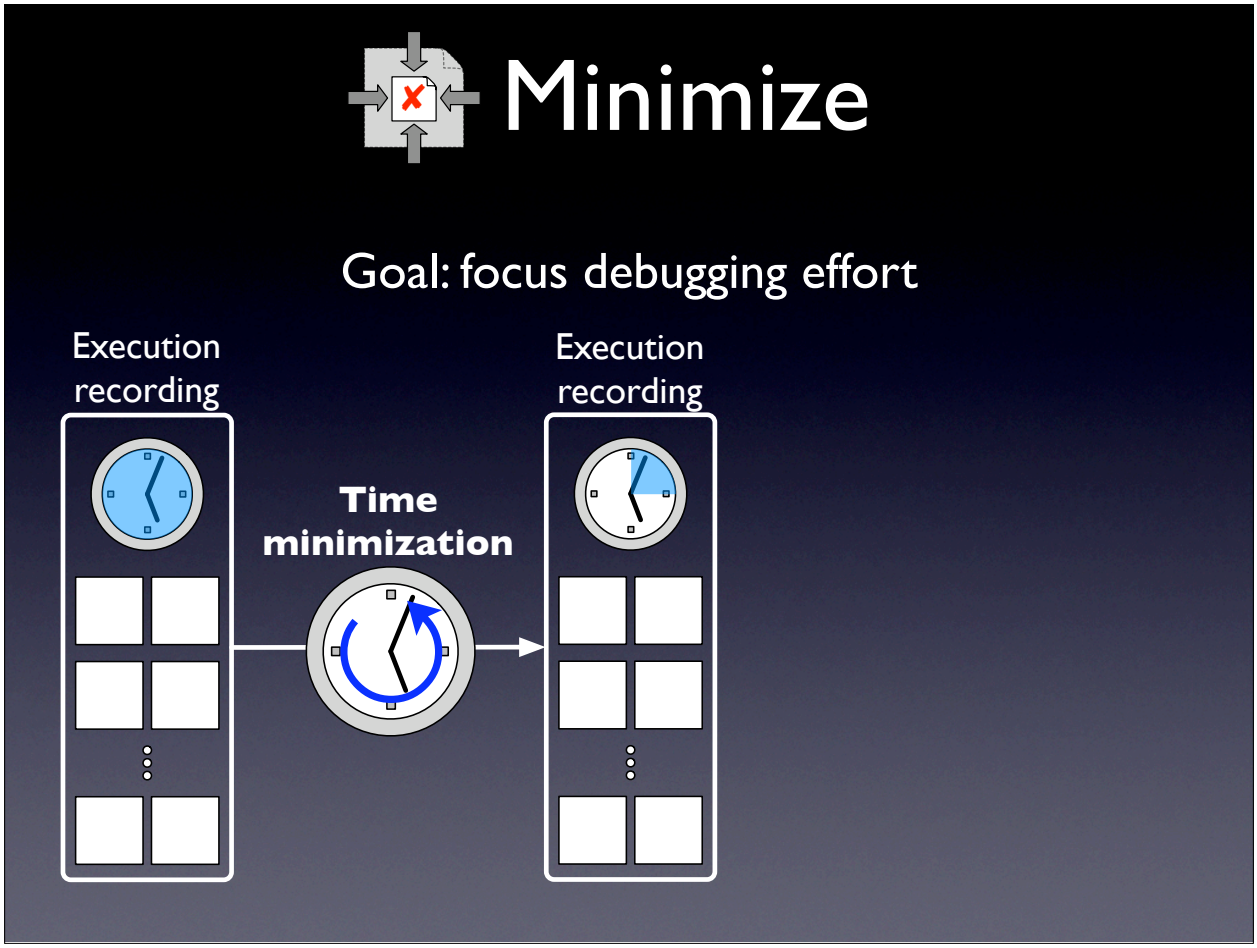
Goal: focus debugging effort

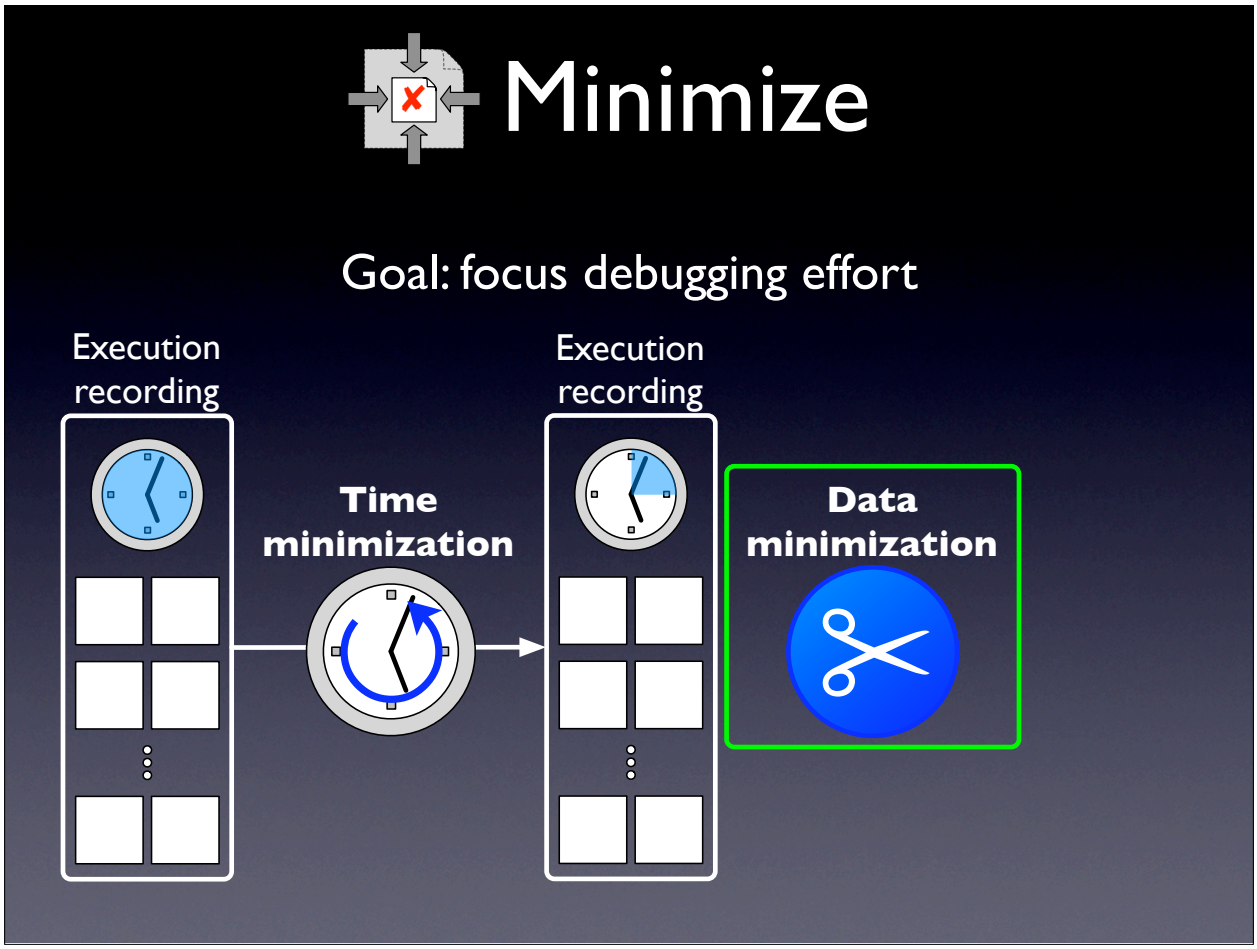
Execution recording

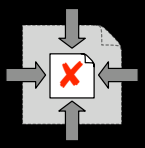


Time minimization



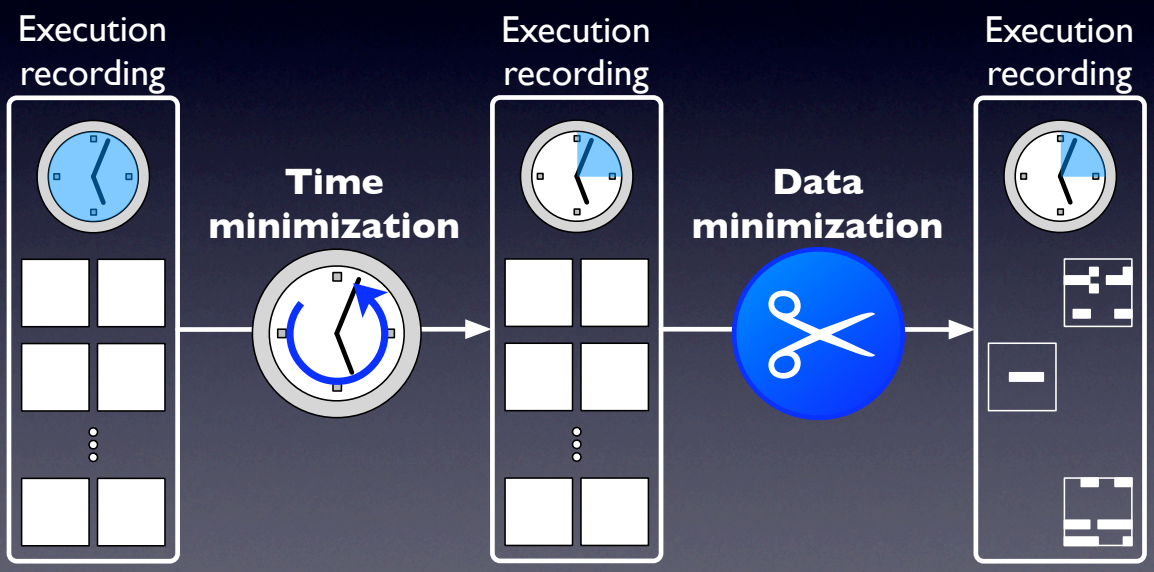

Minimize
Goal: focus debugging effort
Execution recording
Time minimization
Execution recording


Minimize
Goal: focus debugging effort
Execution recording
Time minimization
Execution recording
Data minimization



Minimize

Goal: focus debugging effort



Minimize: time

Event log:

FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 1
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL KEYBOARD NOK
POLL NETWORK OK
FILE foo.2
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK

Environment data (streams):

KEYBOARD: {5680}hello ❙ {4056}c ❙ {300}...

NETWORK: {3405}<html><body>... ❙ {202}...

Environment data (files):

Minimize: time

Remove idle time

Event log:

FILE foo.1
POLL KEYBOARD NOK
POLL KEYBOARD OK
PULL KEYBOARD 1
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
POLL KEYBOARD NOK
POLL NETWORK OK
FILE foo.2
PULL NETWORK 1024
FILE foo.2
POLL KEYBOARD NOK

Environment data (streams):

KEYBOARD: {5680}hello ❙ {4056}c ❙ {300}...

NETWORK: {3405}<html><body>... ❙ {202}...

Environment data (files):

Minimize: time

Remove idle time

Event log:

FILE foo.1
~~POLL KEYBOARD NOK~~
POLL KEYBOARD OK
PULL KEYBOARD 1
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
~~POLL KEYBOARD NOK~~
POLL NETWORK OK
FILE foo.2
PULL NETWORK 1024
FILE foo.2
~~POLL KEYBOARD NOK~~

Environment data (streams):

KEYBOARD: {5680}hello ❙ {4056}c ❙ {300}...

NETWORK: {3405}<html><body>... ❙ {202}...

Environment data (files):

Minimize: time

Remove idle time

Event log:

FILE foo.1
~~POLL KEYBOARD NOK~~
POLL KEYBOARD OK
PULL KEYBOARD 1
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
~~POLL KEYBOARD NOK~~
POLL NETWORK OK
FILE foo.2
PULL NETWORK 1024
FILE foo.2
~~POLL KEYBOARD NOK~~

Remove delays

Environment data (streams):

KEYBOARD: {5680}hello ❙ {4056}c ❙ {300}...

NETWORK: {3405}<html><body>... ❙ {202}...

Environment data (files):

Minimize: time

Remove idle time

Event log:

FILE foo.1
~~POLL KEYBOARD NOK~~
POLL KEYBOARD OK
PULL KEYBOARD 1
POLL NETWORK OK
PULL NETWORK 1024
FILE bar.1
~~POLL KEYBOARD NOK~~
POLL NETWORK OK
FILE foo.2
PULL NETWORK 1024
FILE foo.2
~~POLL KEYBOARD NOK~~

Remove delays

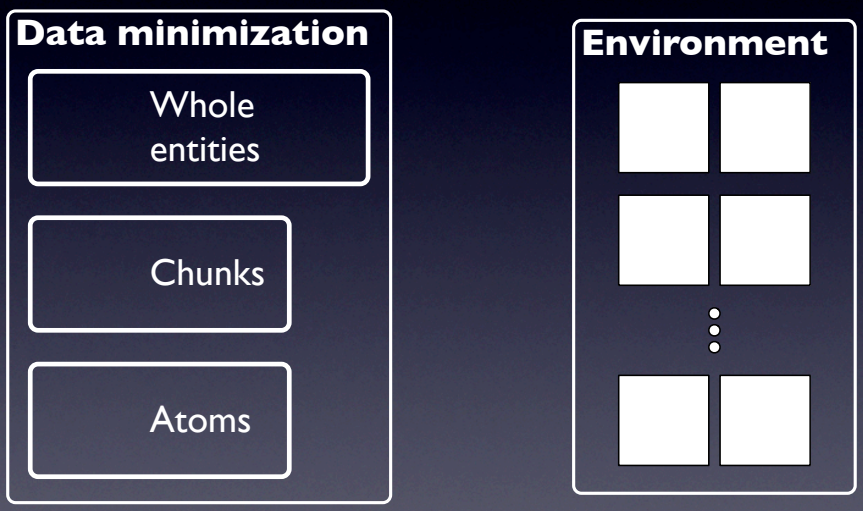
Environment data (streams):

KEYBOARD: ~~{5680}~~hello ❙ ~~{4056}~~c ❙ ~~{300}~~...

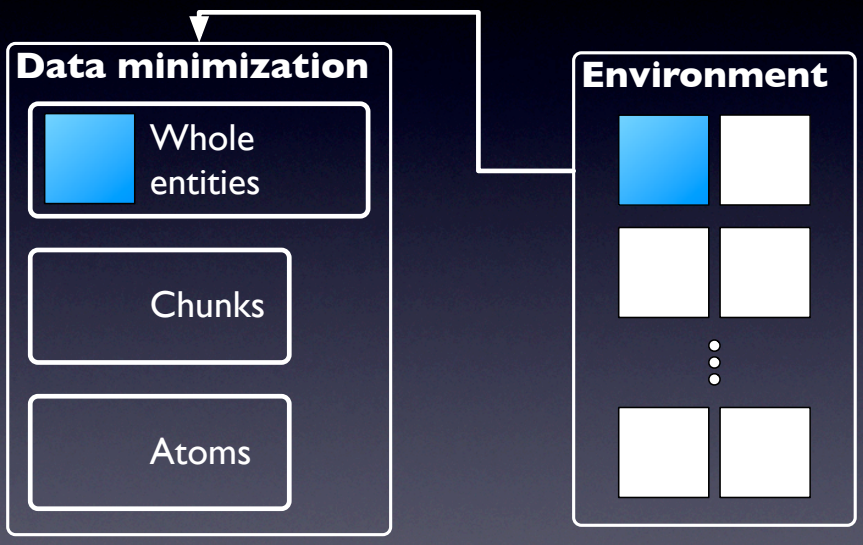
NETWORK: ~~{3405}~~<html><body>... ❙ ~~{202}~~...

Environment data (files):

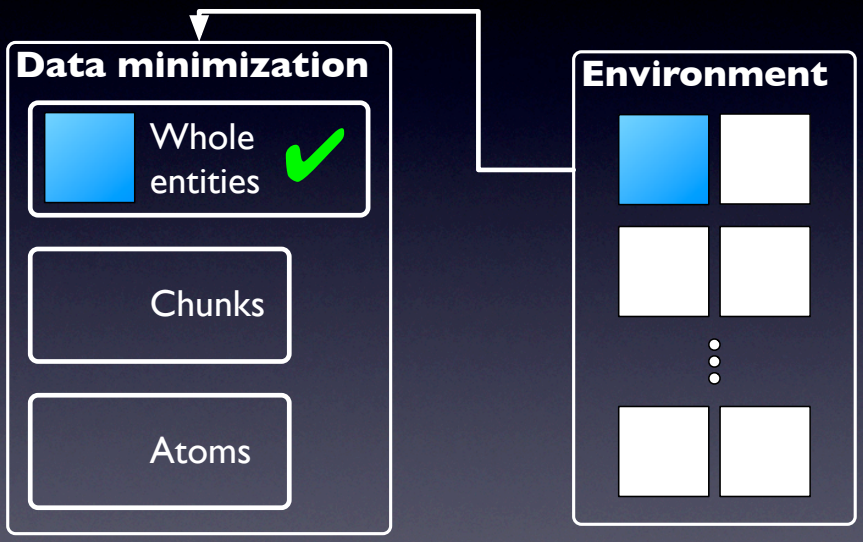
Minimize: data



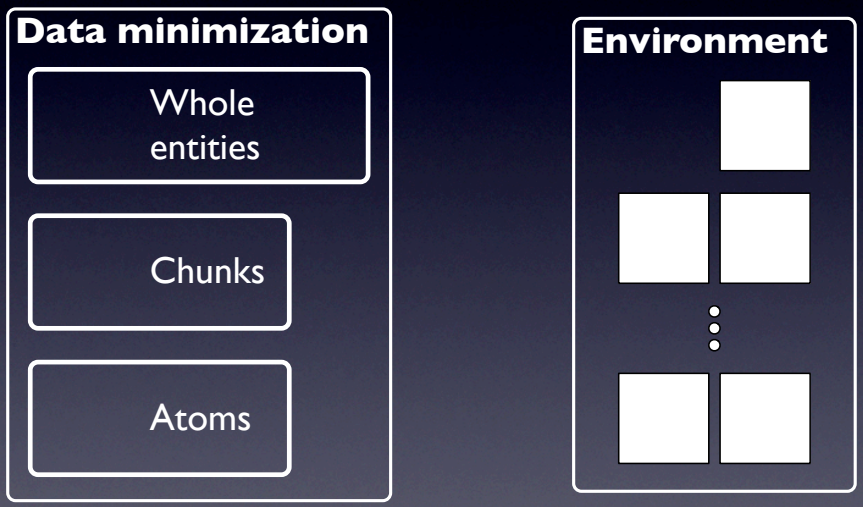
Minimize: data



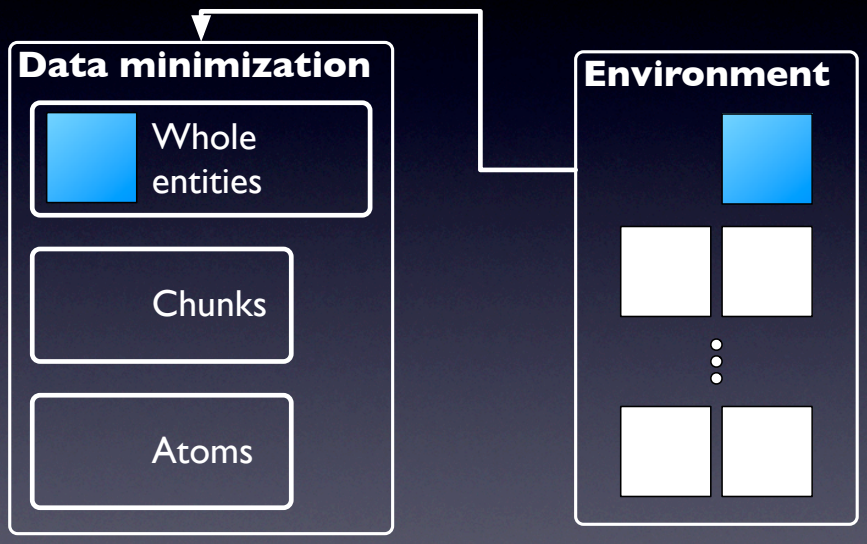
Minimize: data



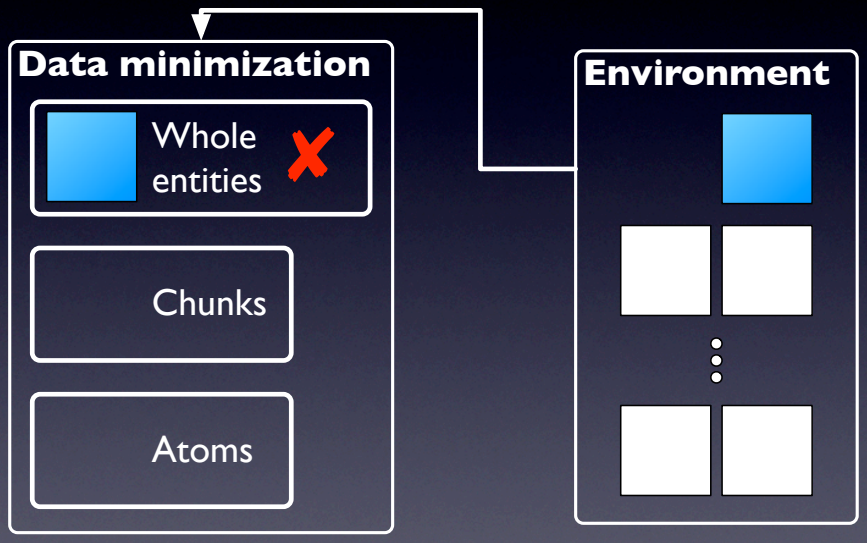
Minimize: data



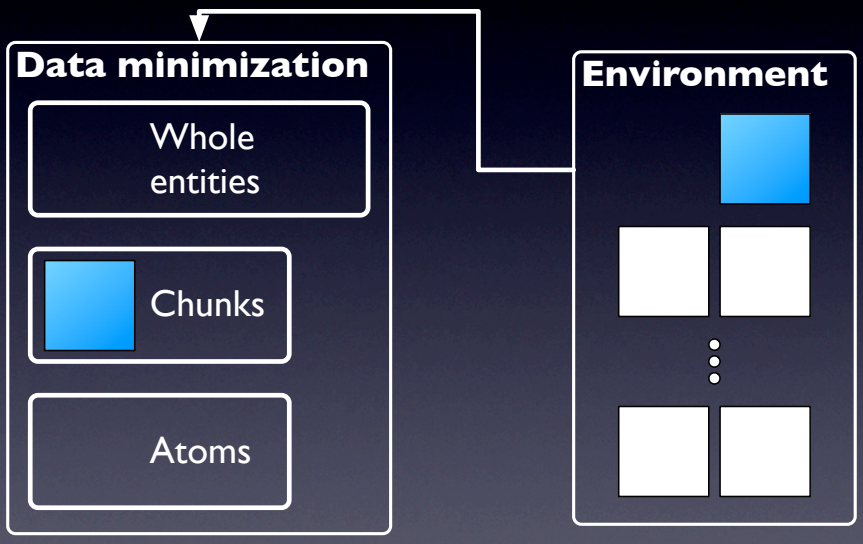
Minimize: data



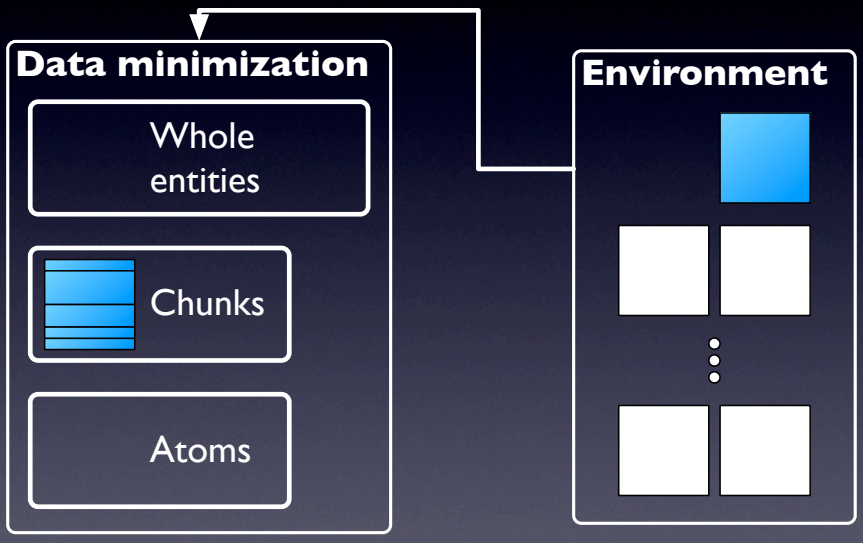
Minimize: data



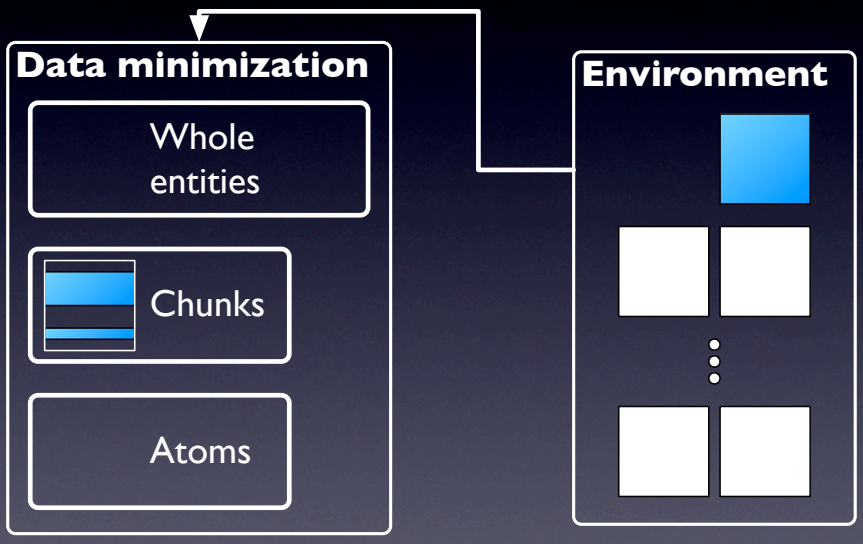
Minimize: data



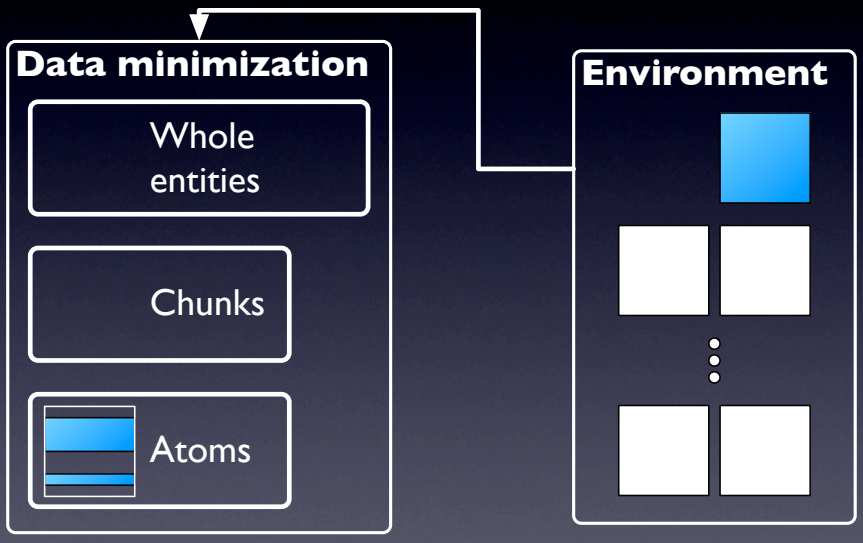
Minimize: data



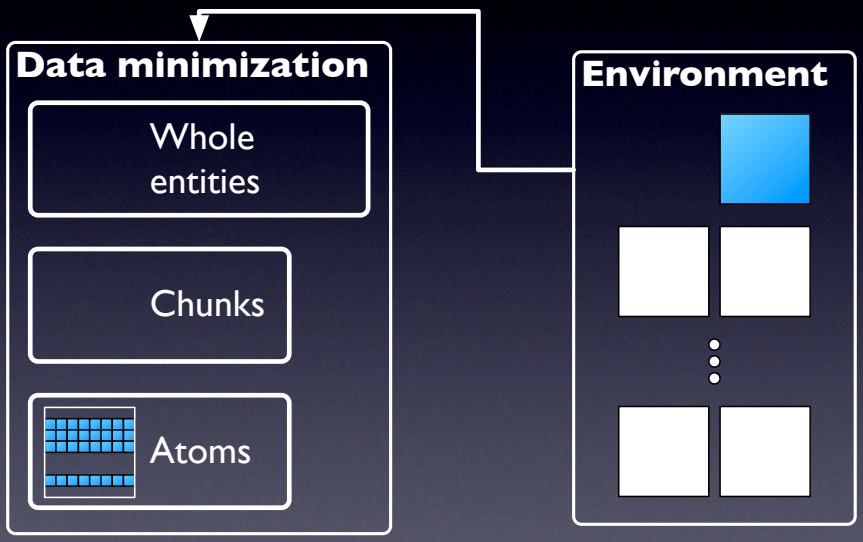
Minimize: data



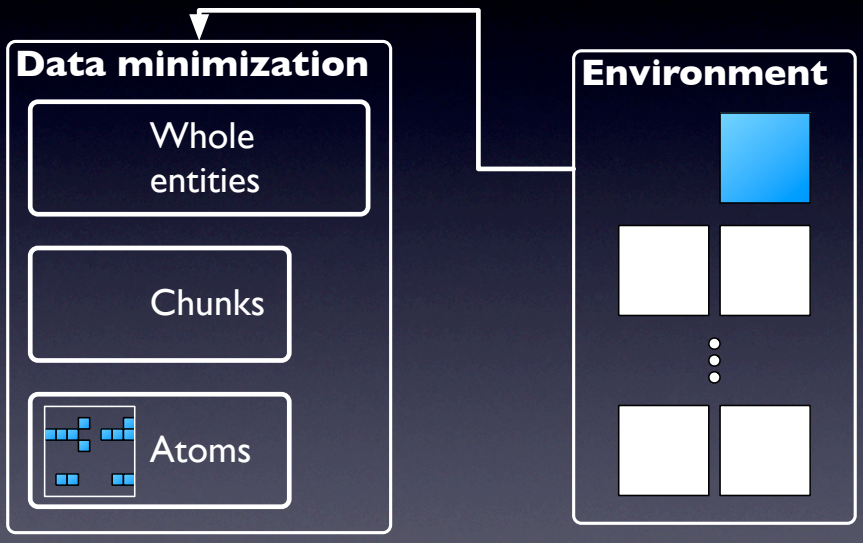
Minimize: data



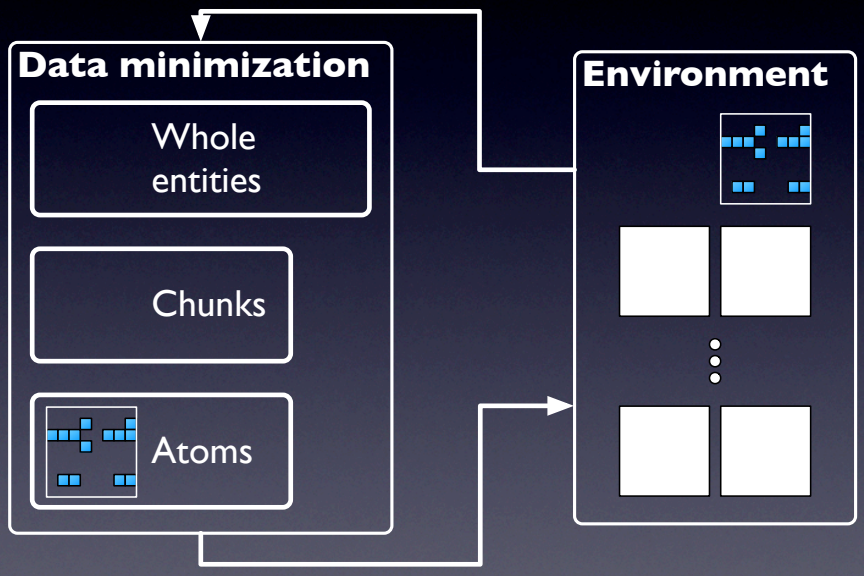
Minimize: data



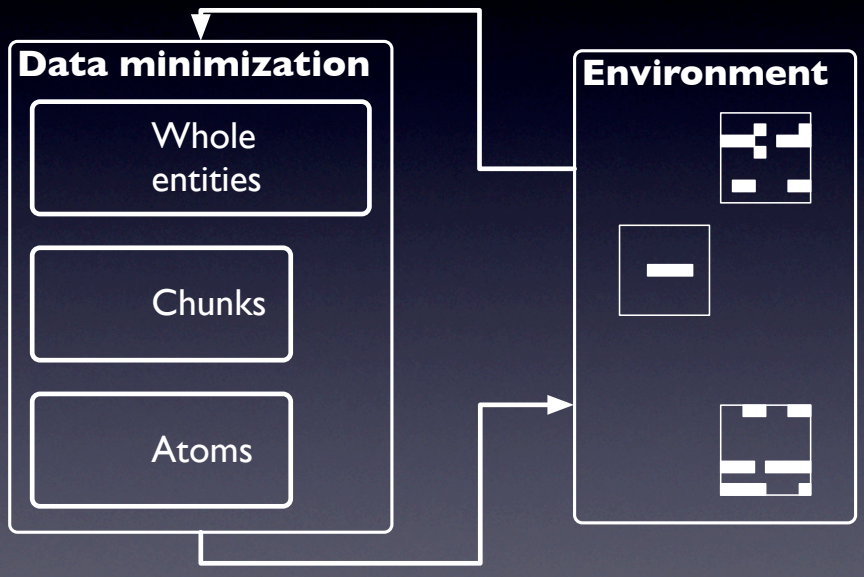
Minimize: data



Minimize: data



Minimize: data



The tool: ADDA

Assisting the **D**ebugging of **D**eployed **A**pplications

- Record and Replay:
  - Works on x86 (c-lib based) binaries
  - Based on dynamic instrumentation (Pin)
  - Maps c-library calls to interaction events
- Minimization:
  - Set of extensible scripts

The tool: ADDA

Assisting the **D**ebugging of **D**eployed **A**pplications

- Record and Replay:
  - Works on x86 (c-lib based) binaries
  - Based on dynamic instrumentation (Pin)
  - Maps c-library calls to interaction events
- Minimization:
  - Set of extensible scripts
- Limitations
  - Technique: May not replay non-deterministic failures
  - Implementation: Does not handle window system events (yet)

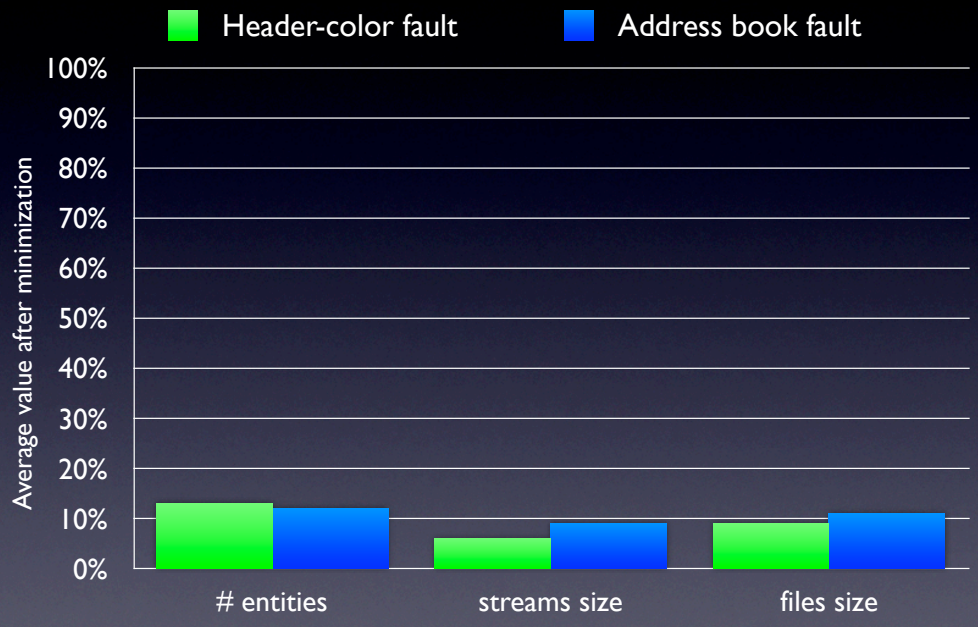
Empirical evaluation

- Research questions
  - RQ1: Can ADDA produce minimized executions that can be used to debug the original failure?
  - RQ2: How much overhead does ADDA impose?
- Subject:
  - Pine — widely-used email / news client
- Data:
  - Two real field failures from Pine's history
  - Set of 20 failing executions, 10 per failure

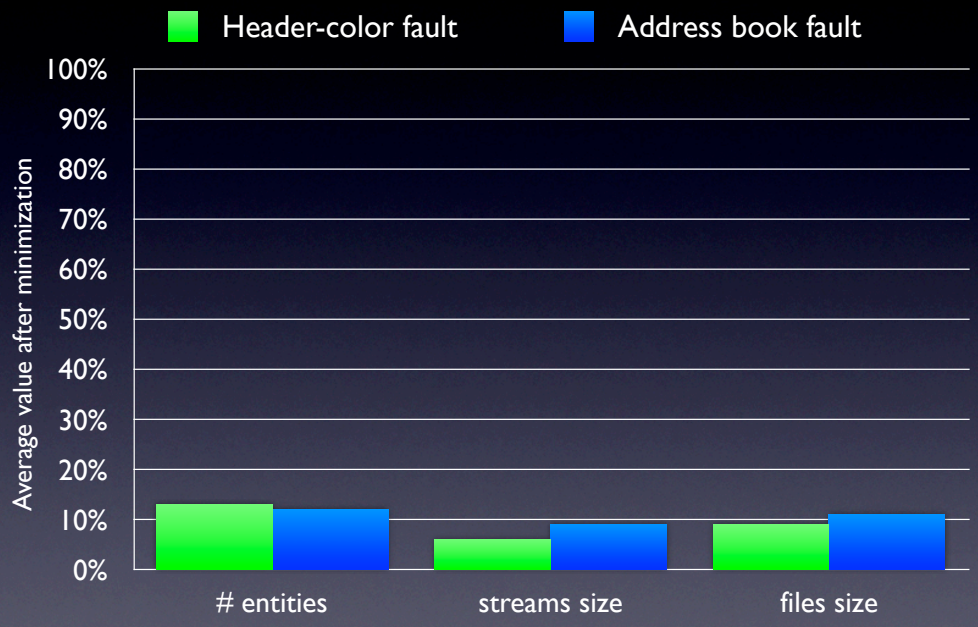
Empirical evaluation

- Research questions
  - RQ1: Can ADDA produce minimized executions that can be used to debug the original failure?
  - RQ2: How much overhead does ADDA impose?
- Subject:
  - Pine — widely-used email / news client
- Data:
  - Two real field failures from Pine's history
  - Set of 20 failing executions, 10 per failure

Minimization results

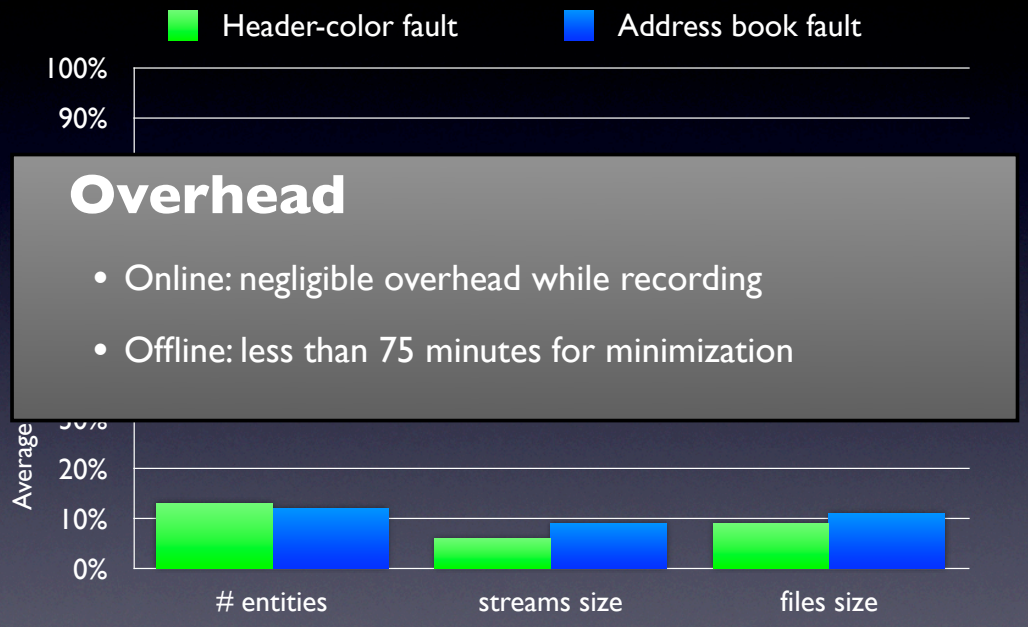


Minimization results



Moreover, these results are conservative: recorded executions only contain the minimal amount of data needed to perform an action.

Minimization results



Overhead

- Online: negligible overhead while recording
- Offline: less than 75 minutes for minimization

Moreover, these results are conservative: recorded executions only contain the minimal amount of data needed to perform an action.

Specific Example: Address Book Failure

- Complete execution
  - 34 entities (files and streams)
  - ≈800kb
- Minimized execution
  - 5 partial entities (4 files, 1 stream)
  - ≈72kb

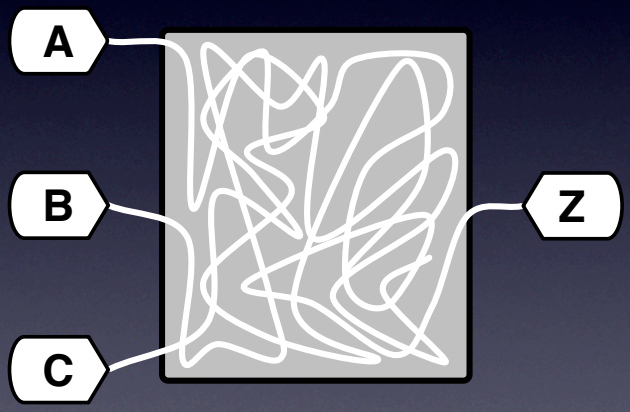
Conclusions

- Novel approach that supports debugging field failures
- Prototype implementation for x86 binaries
- Preliminary empirical evaluation: for the cases considered, our technique can
  1. minimize failing executions
  2. preserve their failing behavior
  3. impose low overhead on users

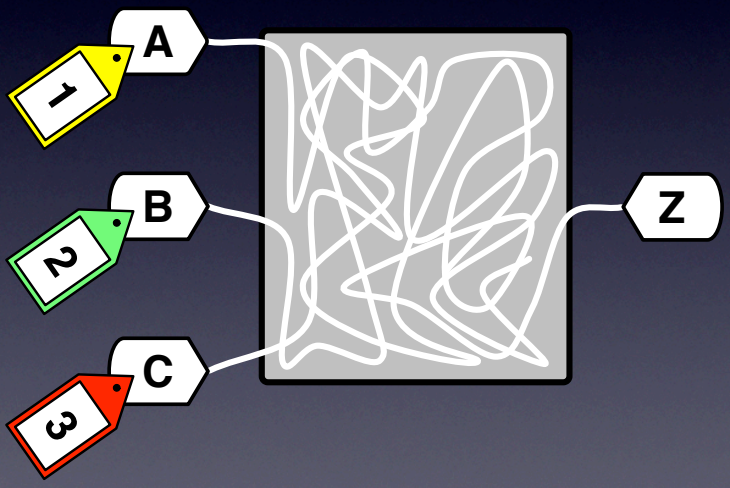
Future work

- More studies: additional applications and real users
- Extend technique / implementation
  - Support window system events
    - Mac OS X
  - Investigate ways to decrease minimization time
    - Ad-hoc minimization algorithms
    - Input tracking (dynamic tainting)
- Make use of passing executions

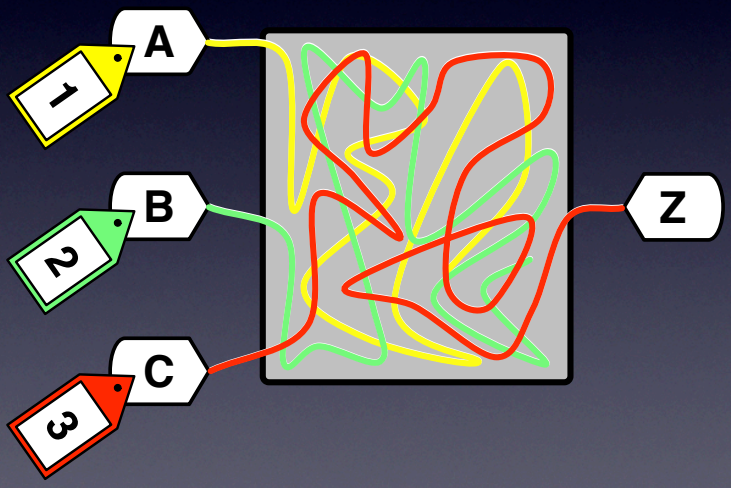
Dynamic tainting for input tracking



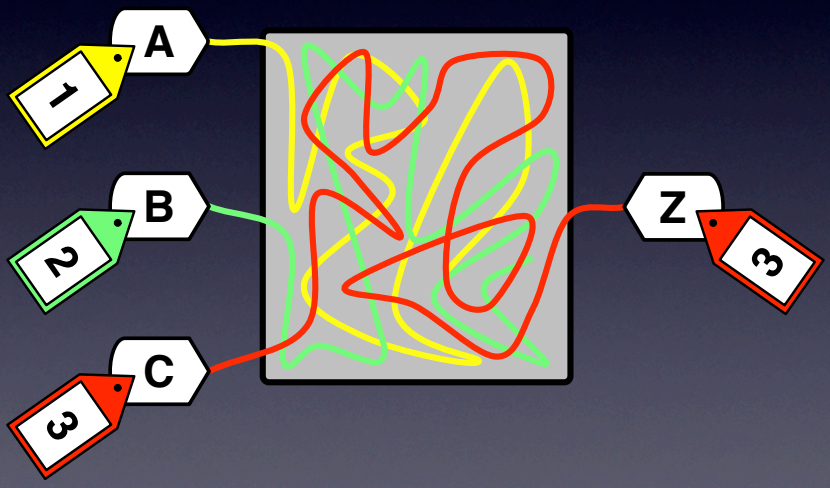
Dynamic tainting for input tracking



Dynamic tainting for input tracking



Dynamic tainting for input tracking

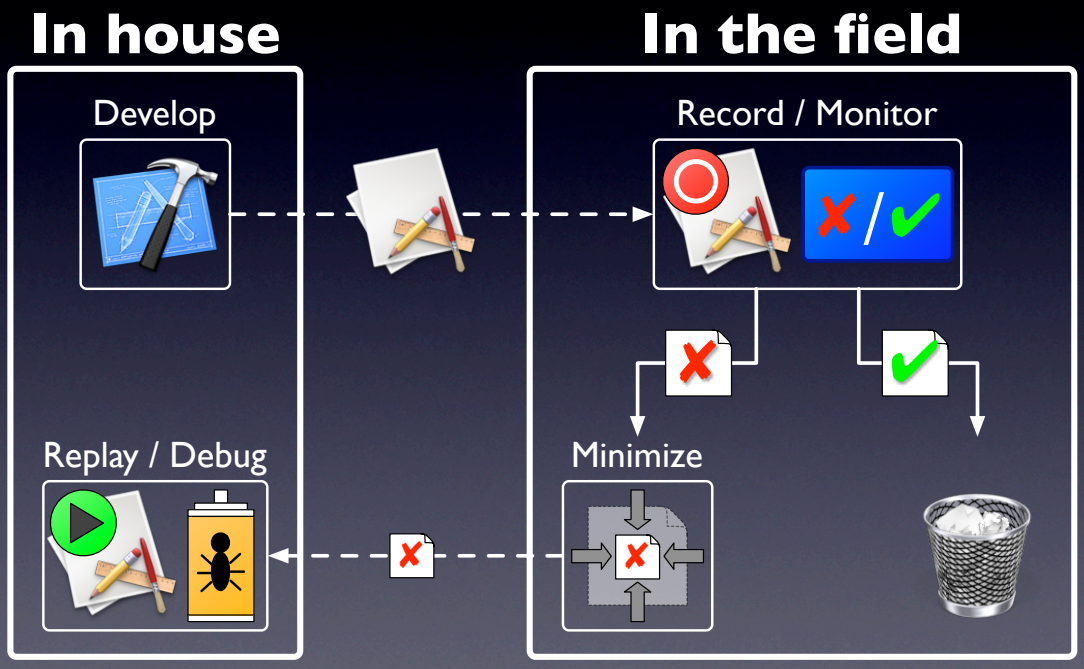


Dynamic tainting for input tracking

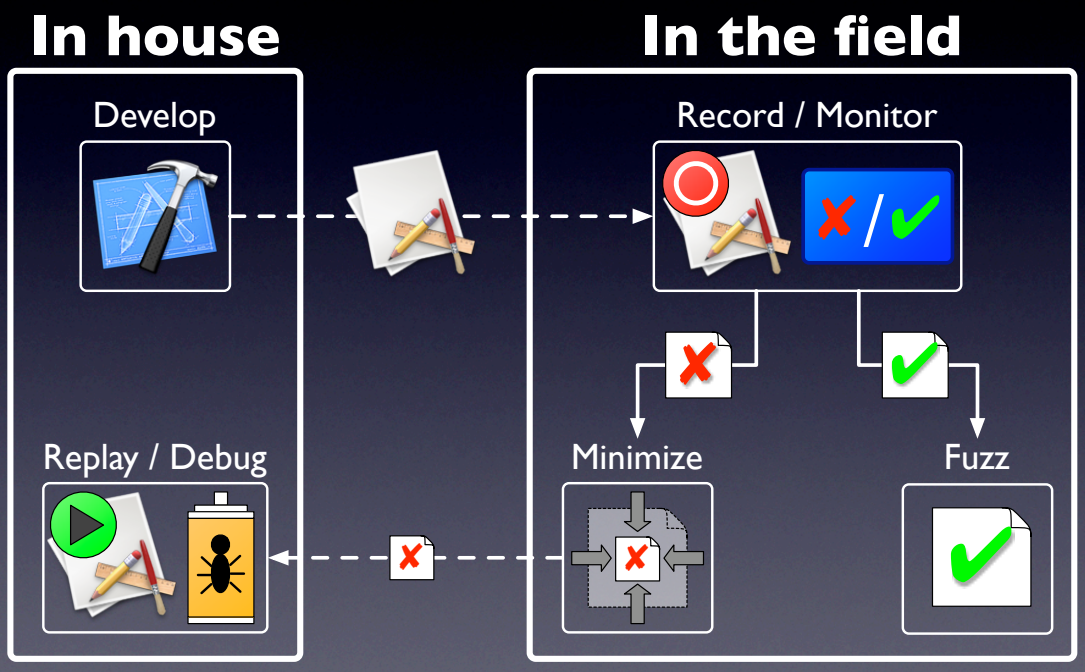
Dytan: A Generic Dynamic Taint Analysis Framework
James Clause, Wanchun Li, and Alessandro Orso
International Symposium on Software Testing and Analysis (ISSTA 2007)

Effective Memory Protection Using Dynamic Tainting
James Clause, Ioannis Doudalis, Alessandro Orso, and Milos Prvulovic
International Conference on Automated Software Engineering (ASE 2007)

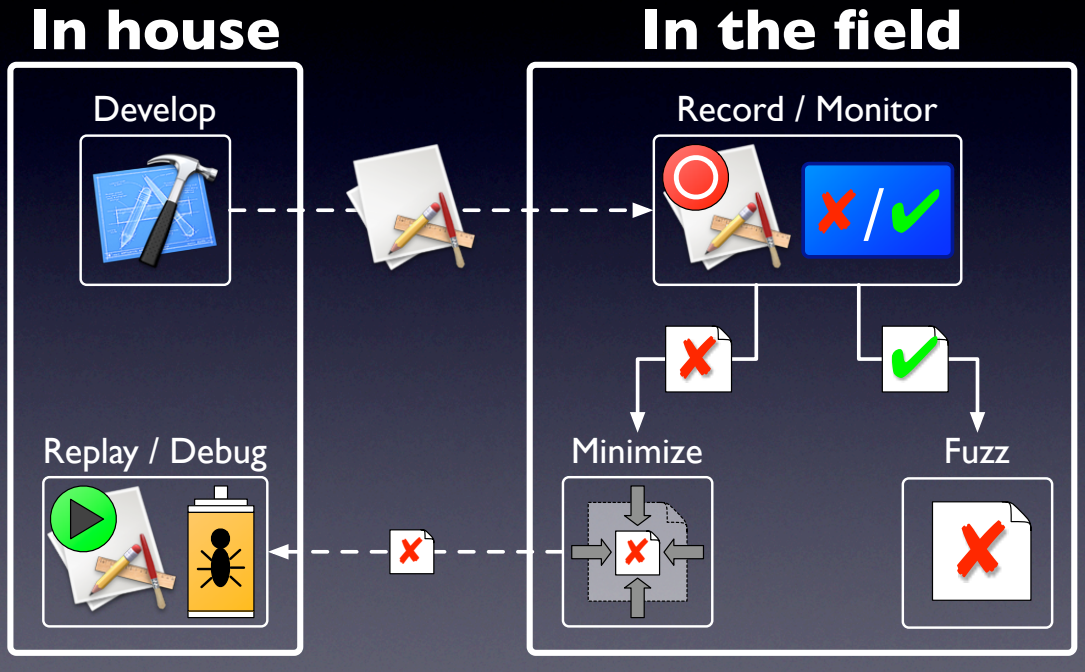
Using passing executions



Using passing executions



Using passing executions



Using passing executions