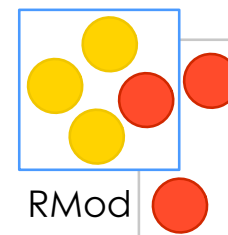


# Remote Debugging and Reflection in Resource Constrained Devices

Nikolaos Papoulias - December 2013





# Outline

- ✧ Introduction
- ✧ Related Work
- ✧ Contributions
- ✧ Implementation
- ✧ Validation
- ✧ Conclusion & Future Work



# Context - Constrained Devices



- ✦ Cannot locally support an IDE & Dev-Tools
- ✦ Have different HW/SW Configurations from Dev-Machines



# Context - Debugging Constrained Devices

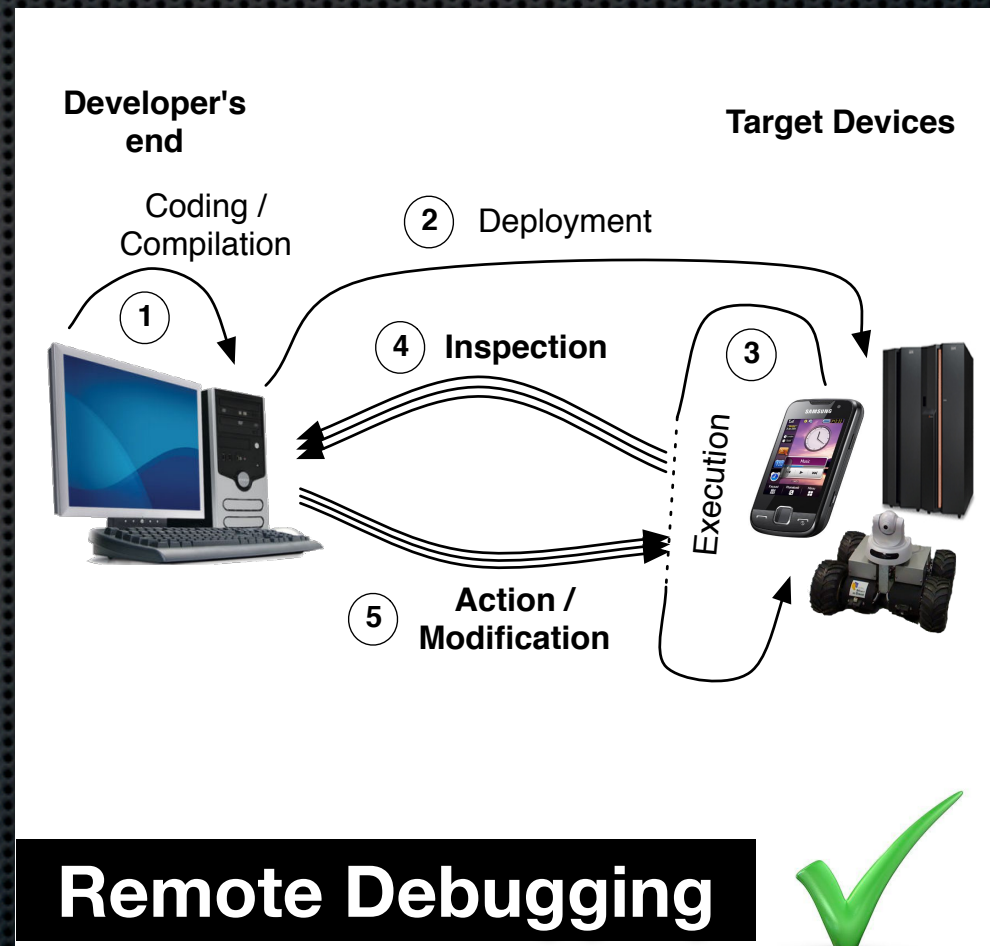
- ✧ ~~Emulators~~
- ✧ ~~Post-Mortem Analysis~~





# Context - Debugging Constrained Devices

- ✦ ~~Emulators~~
- ✦ ~~Post-Mortem Analysis~~





# Remote Debugging

**Is a solution that is distributed in a nature:**

- ✦ **Impact on productivity** due to re-deployments [ZeroTurnAround 2011]
- ✦ **Lacks facilities otherwise** available in a local setting (e.g O-Centric Debugging [Ressia 2012b])



# Research Questions

- ✦ *What are the properties of an ideal remote debugging solution ?*
- ✦ **Given these properties which model for remote debugging can exhibit them ?**
- ✦ **What are the trade-offs between this ideal model and a real world implementation ?**



# Thesis Statement - Properties

An ideal remote debugging solution should support





# Interactiveness

*the ability to **incrementally update** all parts of a remote application without losing the running context (i.e without stopping the application).*

*Add/Rem **Packages**, Add/Rem **Classes**, Add/Rem **Fields**,  
Edit **Hierarchy**, Add/Rem **Methods***



# Instrumentation

*the ability to **alter the semantics of a running process** in order to assist debugging*

**Method/Statement** Execution, **Class** Instantiation/Field Read/Write  
**Object** Read/Write/Send/Receive/Argument/Store/Interact



# Distribution

*the ability of a debugging solution **to adapt its framework** while debugging a remote target*

***No-Distribution** / **Fixed** Middleware /  
**Extensible** Middleware / **Adaptable** Middleware*



# Security

*the availability of prerequisites  
for security mechanisms such as*  
***authentication and access restriction***

***Internal / External / Target-Side / Client-Side***










# Outline

- ✧ Introduction
- ✧ Related Work
- ✧ Contributions
- ✧ Implementation
- ✧ Validation
- ✧ Conclusion & Future Work



# Related Work - Overview

 JAVA JPDA	 .NET	 GDB	 DCE	 JREBEL	 SMALLTALK	 BIFROST
---	---	--	--	---	--	--

✦ Interactiveness (**6**)

✦ Instrumentation (**13**)




✦ Distribution (**+++**)

✦ Security (**4**)

1	1	1	6	6	6	6
4	4	5	4	4	3	12
+	++	+	+	+	-	-
3	4	2	3	3	0	0

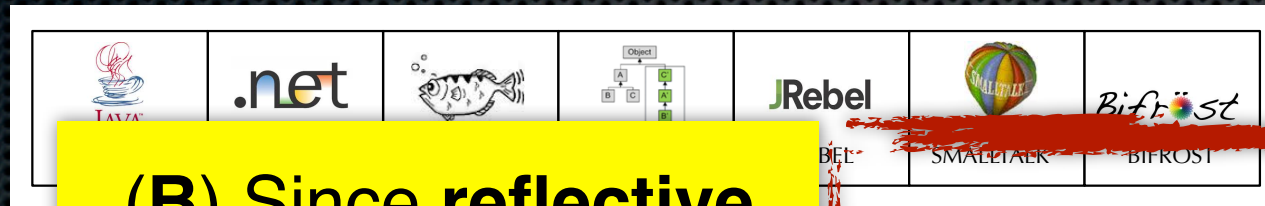


# Related Work - Overview

						
					SMALLTALK	BIFROST
1	(A) None of the existing solutions met all of our criteria				6	6
4					3	12
+	++	+	+	(B) But reflective debugging proved superior to other approaches in a local setting		
3	4	2	3			



# Related Work - Overview

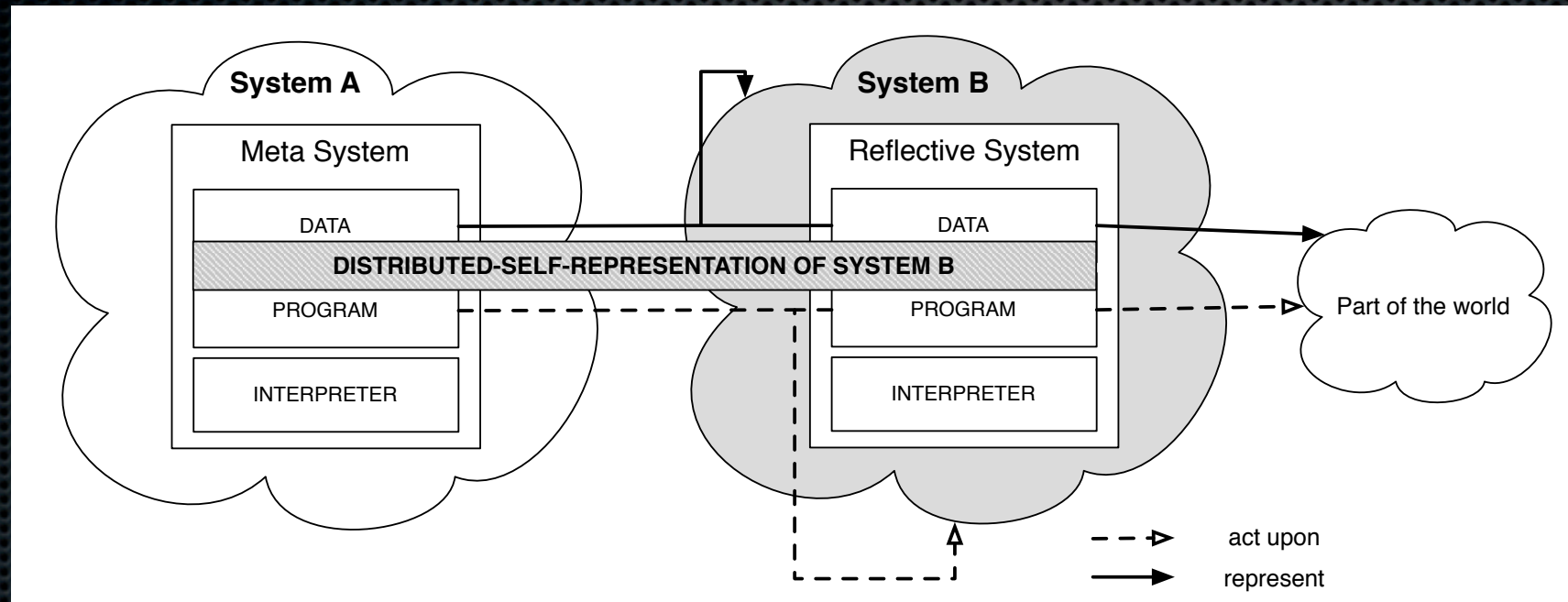


**(B)** Since **reflective debugging** proved superior to other approaches in a local setting

**(C)** Investigate **Remote Reflection** Design Patterns



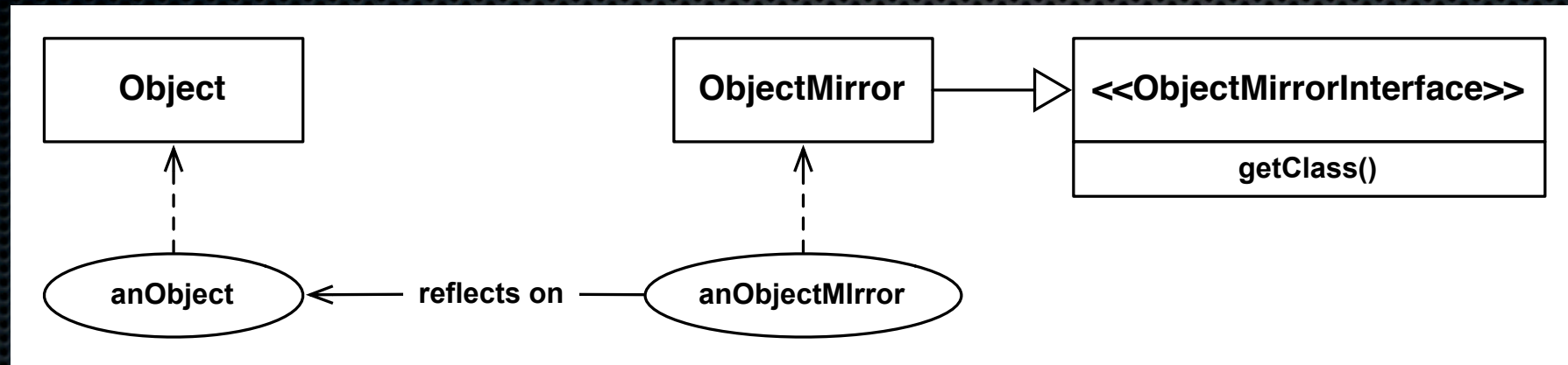
# Remote Reflection



***Remote Proxy / Remote Facade / Mirrors***



# Mirrors - Explicit Meta-Objects



- ✦ **Encapsulation**
- ✦ **Stratification**
- ✦ **Ont. Correspondance**

Mirror on: anObject  
***Indirection***



# Design Patterns - Criteria

- ✦ Extensibility, Re-use
- ✦ Distribution
- ✦ Identity *[Bracha 2010]*
- ✦ Meta-recursion *[Denker 2008]*

**Mirrors can be seen as an extension to both the remote proxy and the remote facade patterns**



# Mirrors - Open Issues

- ✦ **Mirrors and the Problem of State** - debugging meta-information in cohesive language kernels
- ✦ **Mirrors and Intercession** - advanced instrumentation support while debugging



# Outline

- ✧ Introduction
- ✧ Related Work
- ✧ Contributions
- ✧ Implementation
- ✧ Validation
- ✧ Conclusion & Future Work



# Our Proposals

## MetaTalk

- ✦ A **solution to the problem of Reflective-Data** [Maes 1987b] in the context of mirrors [Bracha 2004]
- ✦ The **definition of a model** for remote debugging that can exhibit the properties of: **interactiveness, instrumentation distribution and security**

## Mercury



# Our Proposal

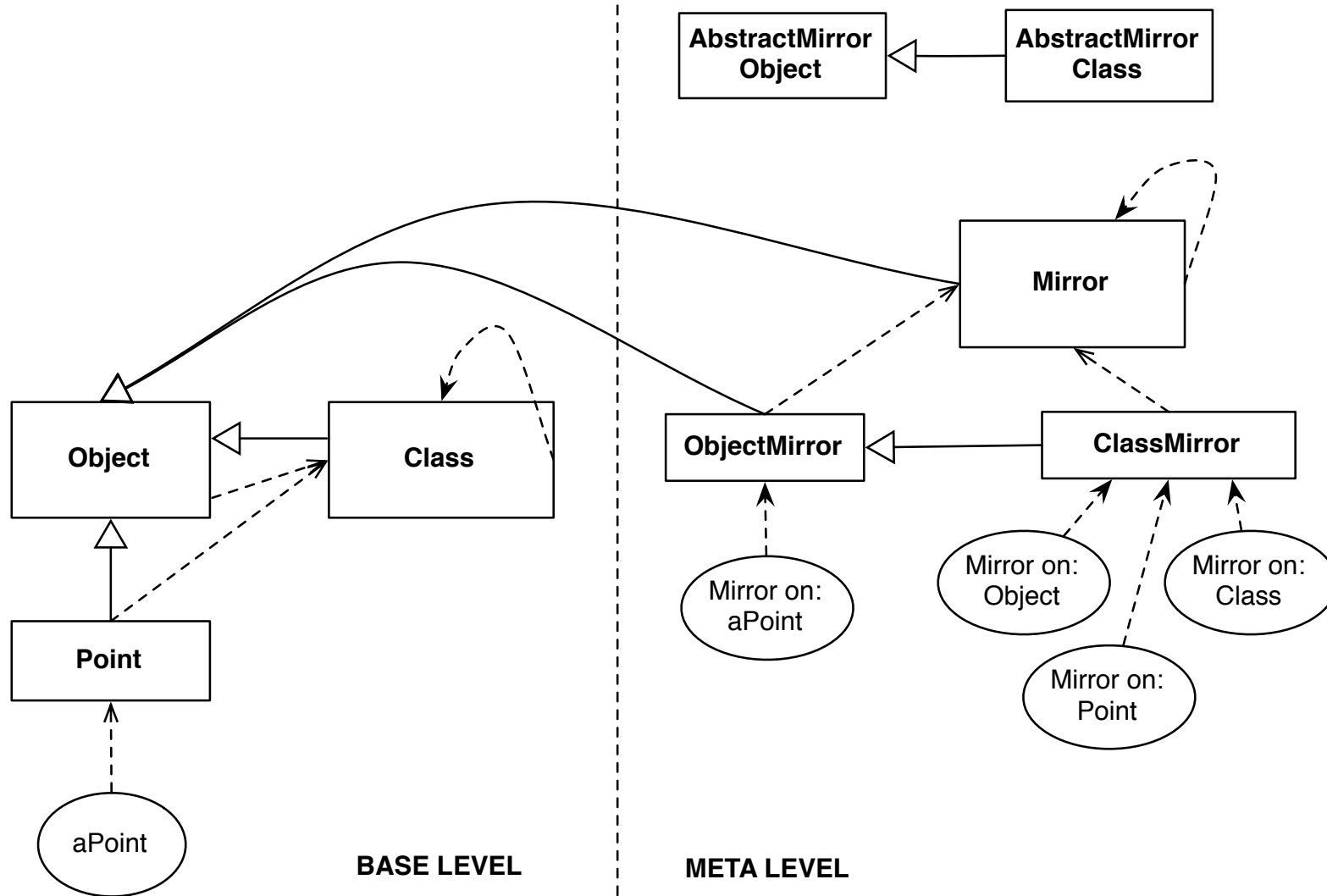
## MetaTalk

From a language design perspective meta-objects should be both:

- ✧ **Pluggable** as mirrors are
- ✧ and **State-Full** as 3-KRS meta-objects



# MetaTalk Kernel





# MetaTalk Implementation

- ✦ **MetaTalk-VM** is written in Pharo
- ✦ **MetaTalk-Compiler** relies on Petit-Parser
- ✦ **Object-Model** inspired by ObjVLisp

<http://www.squeaksource.com/MetaTalk/>



# Our Proposals

## MetaTalk



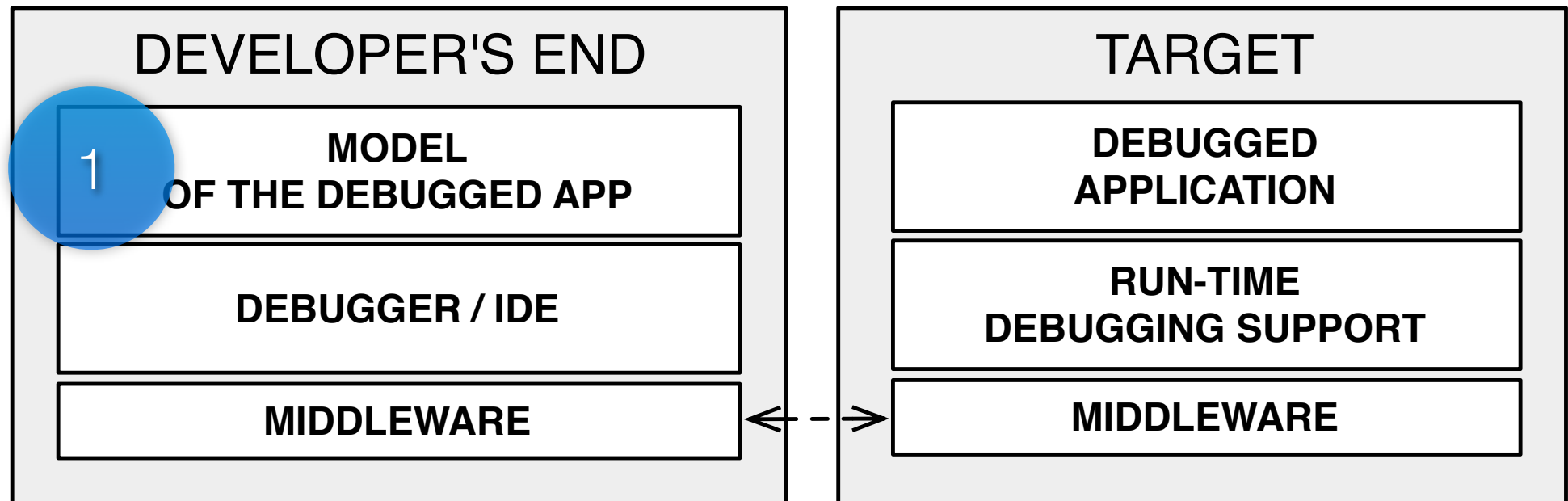
- ✦ A **solution to the problem of Reflective-Data** [Maes 1987b] in the context of mirrors [Bracha 2004]
- ✦ The **definition of a model** for remote debugging that can exhibit the properties of: **interactiveness, instrumentation distribution and security**

## Mercury



# Overview

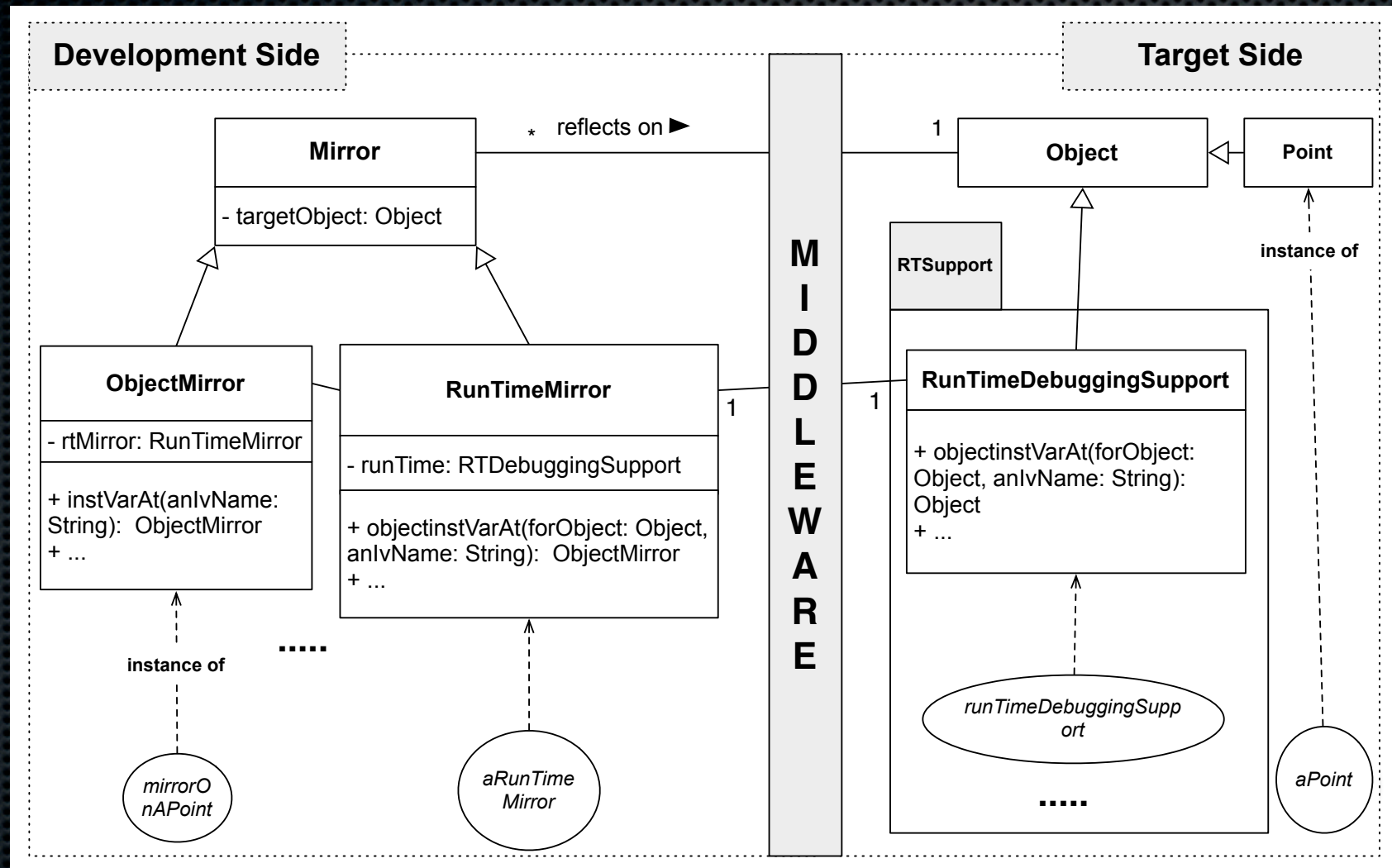
# Mercury



- ✦ Interactiveness - *through a mirror-based remote meta-level that is causally connected to its target*

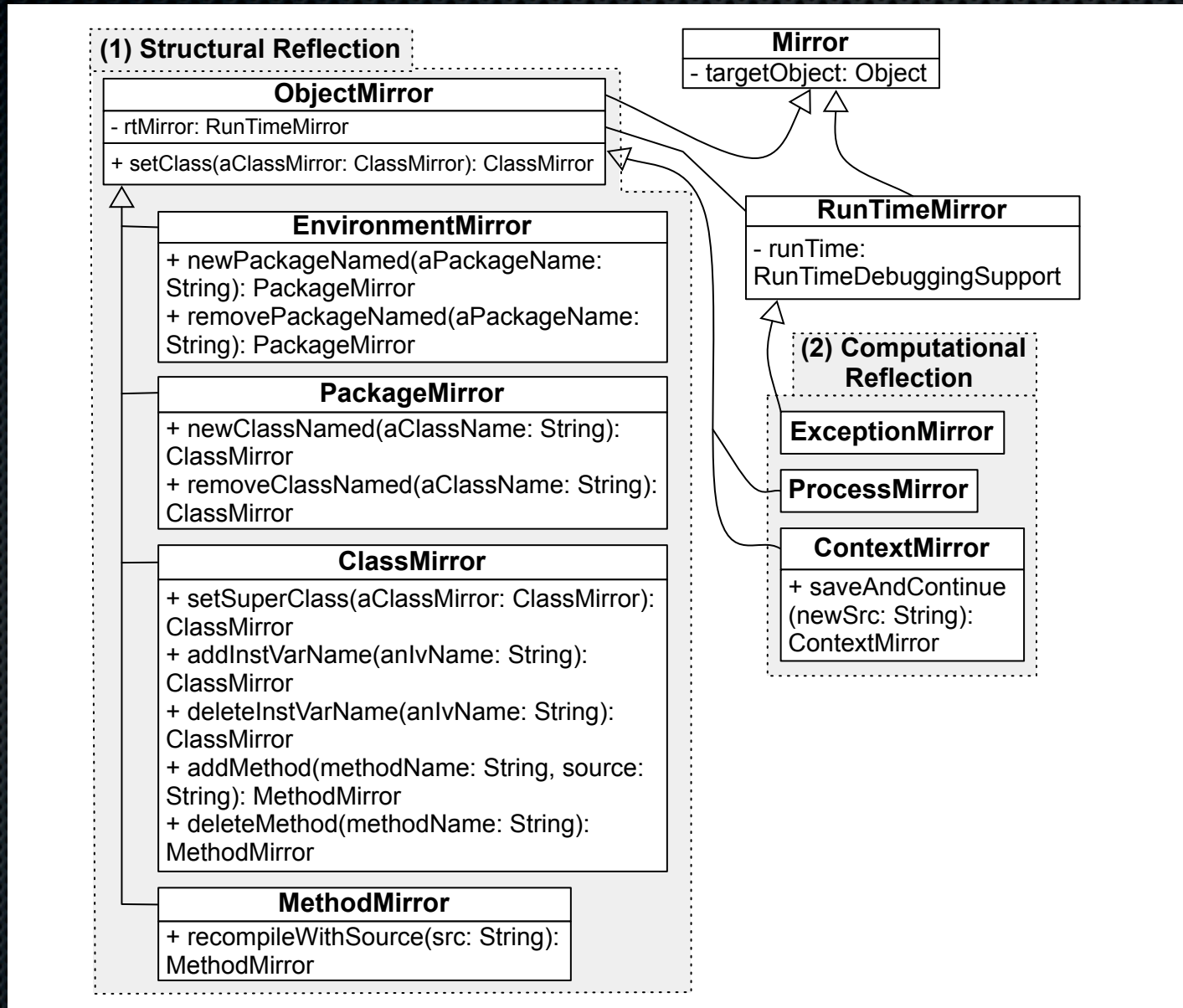


# Core Model



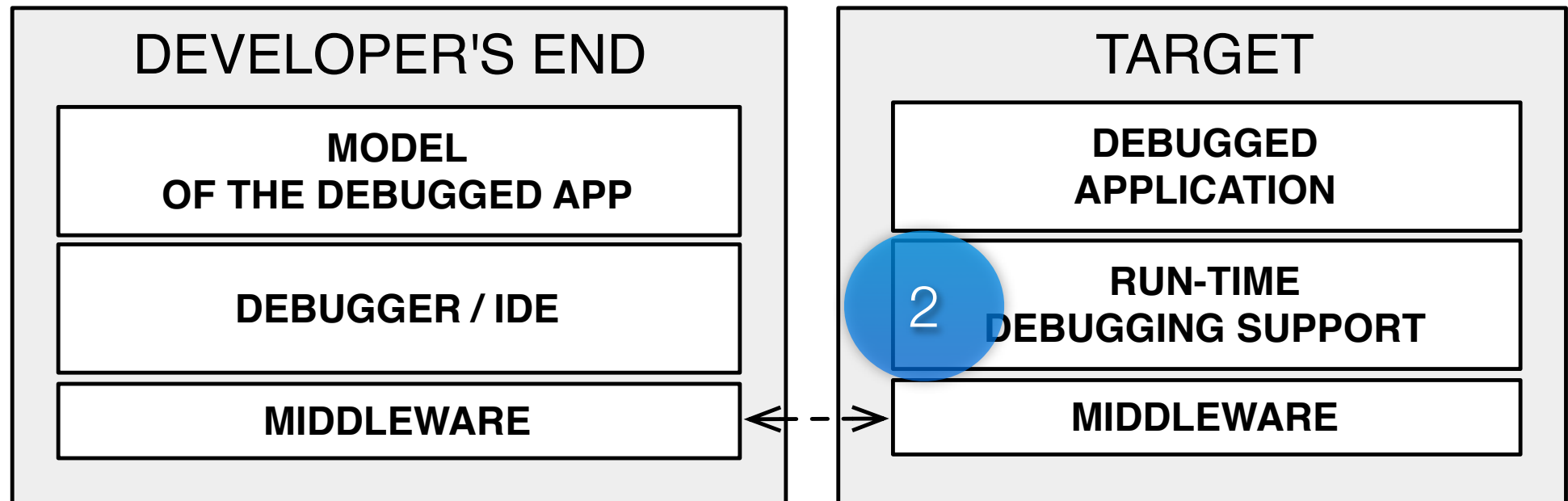


# Interactiveness





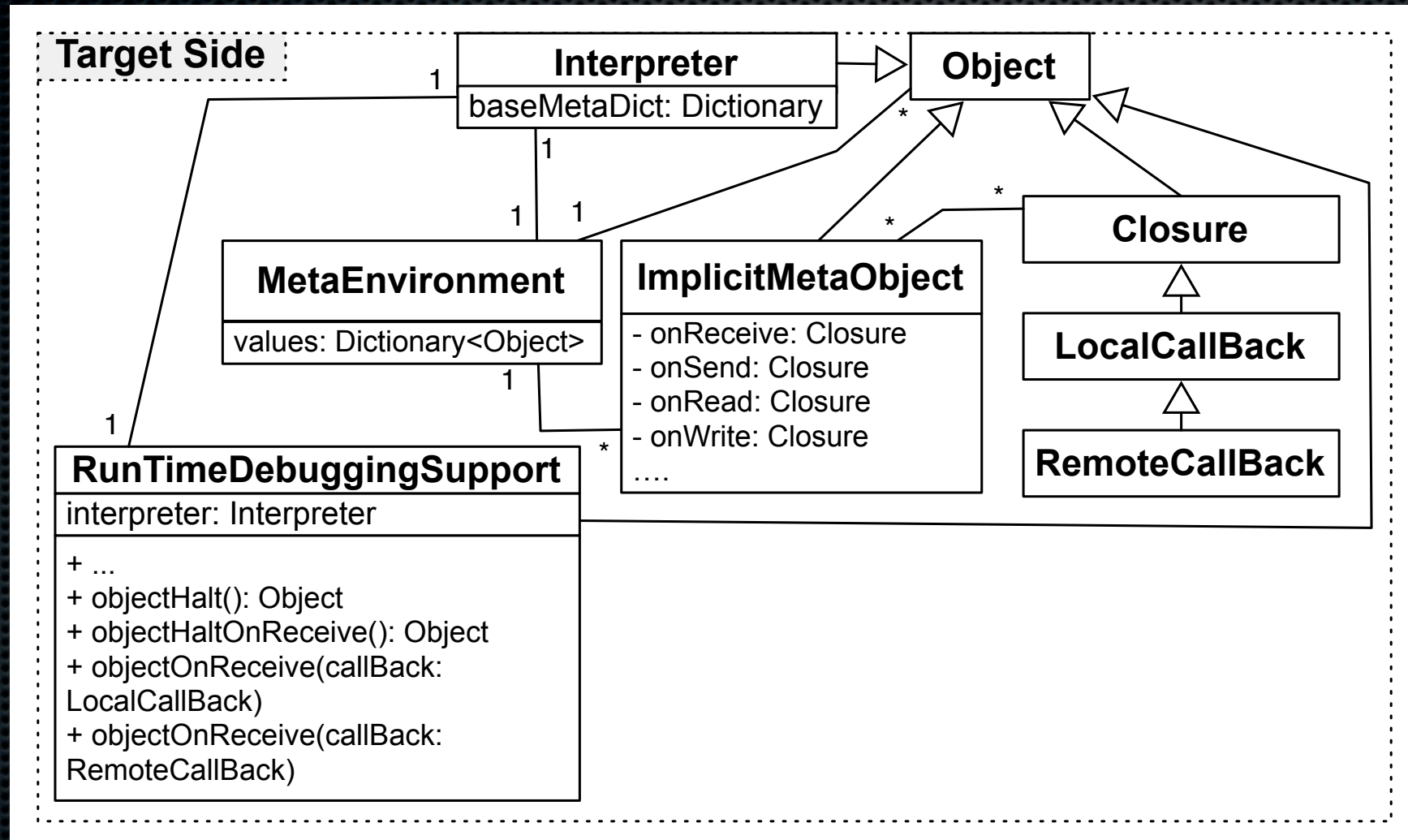
# Our Proposal - Overview



- ✦ Instrumentation - *through reflective intercession by reifying the underlying execution environment*

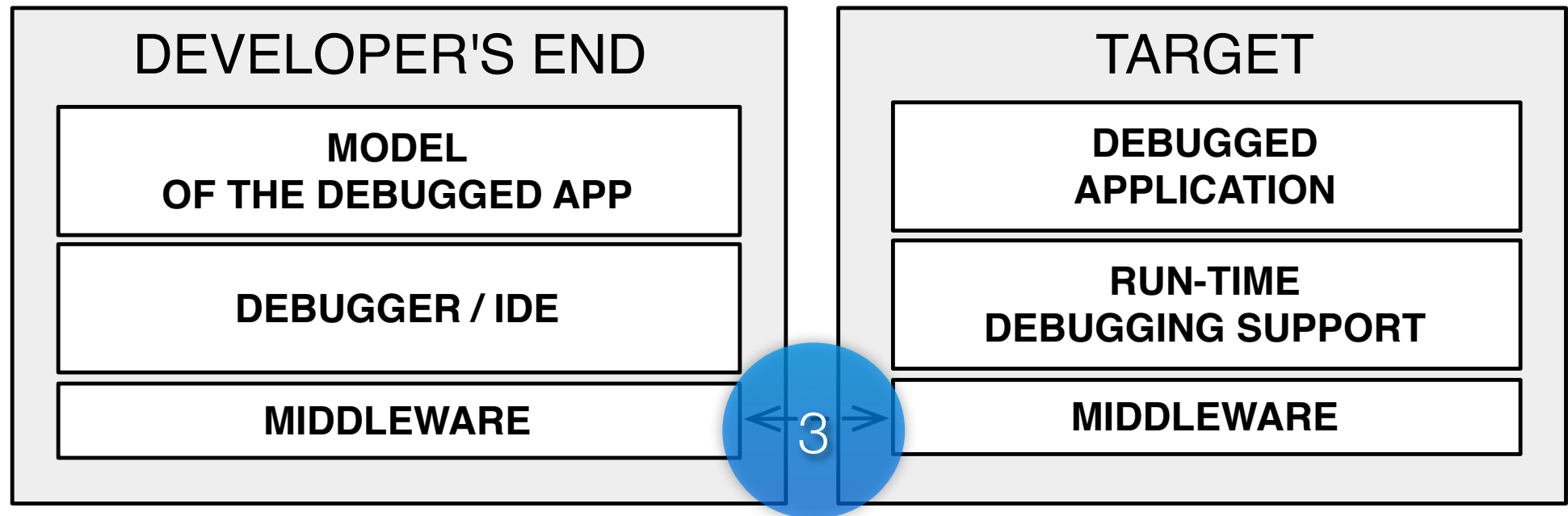


# Instrumentation





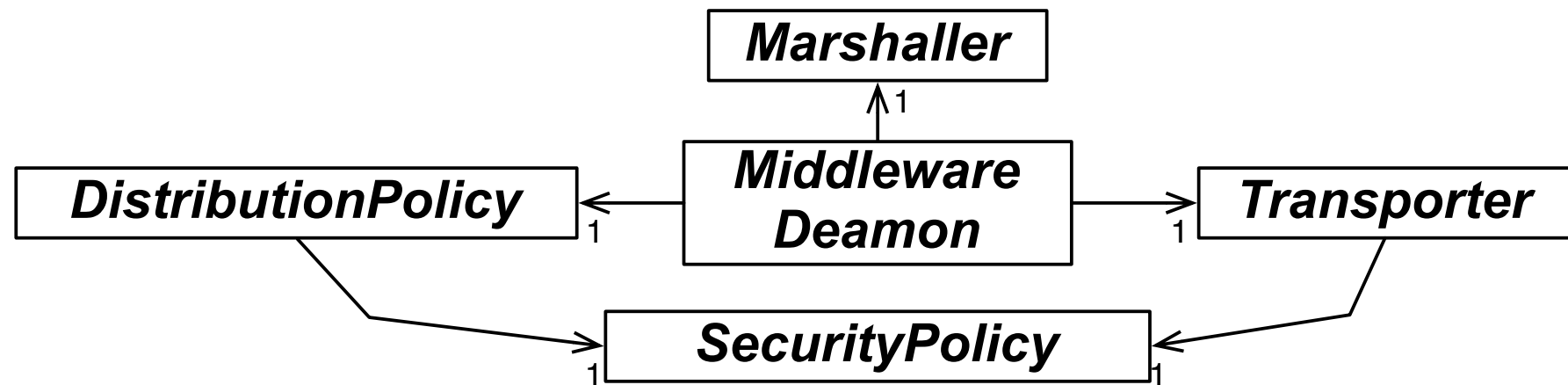
# Our Proposal - Overview



- ✦ Distribution - *through an Adaptable Middleware*

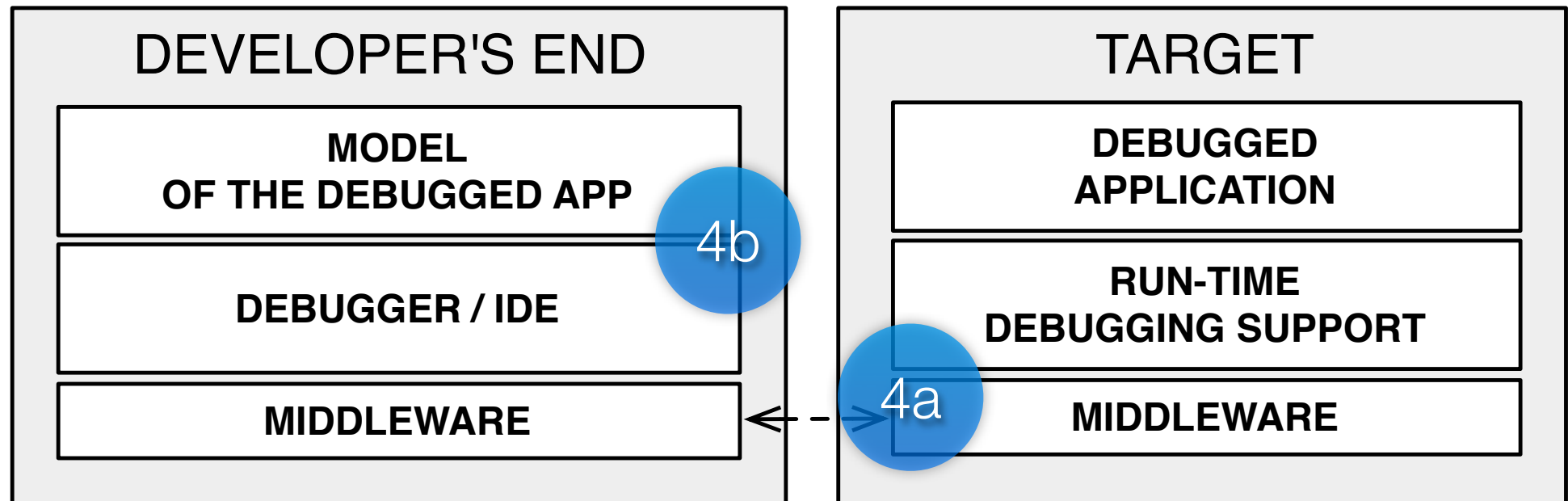


# Distribution





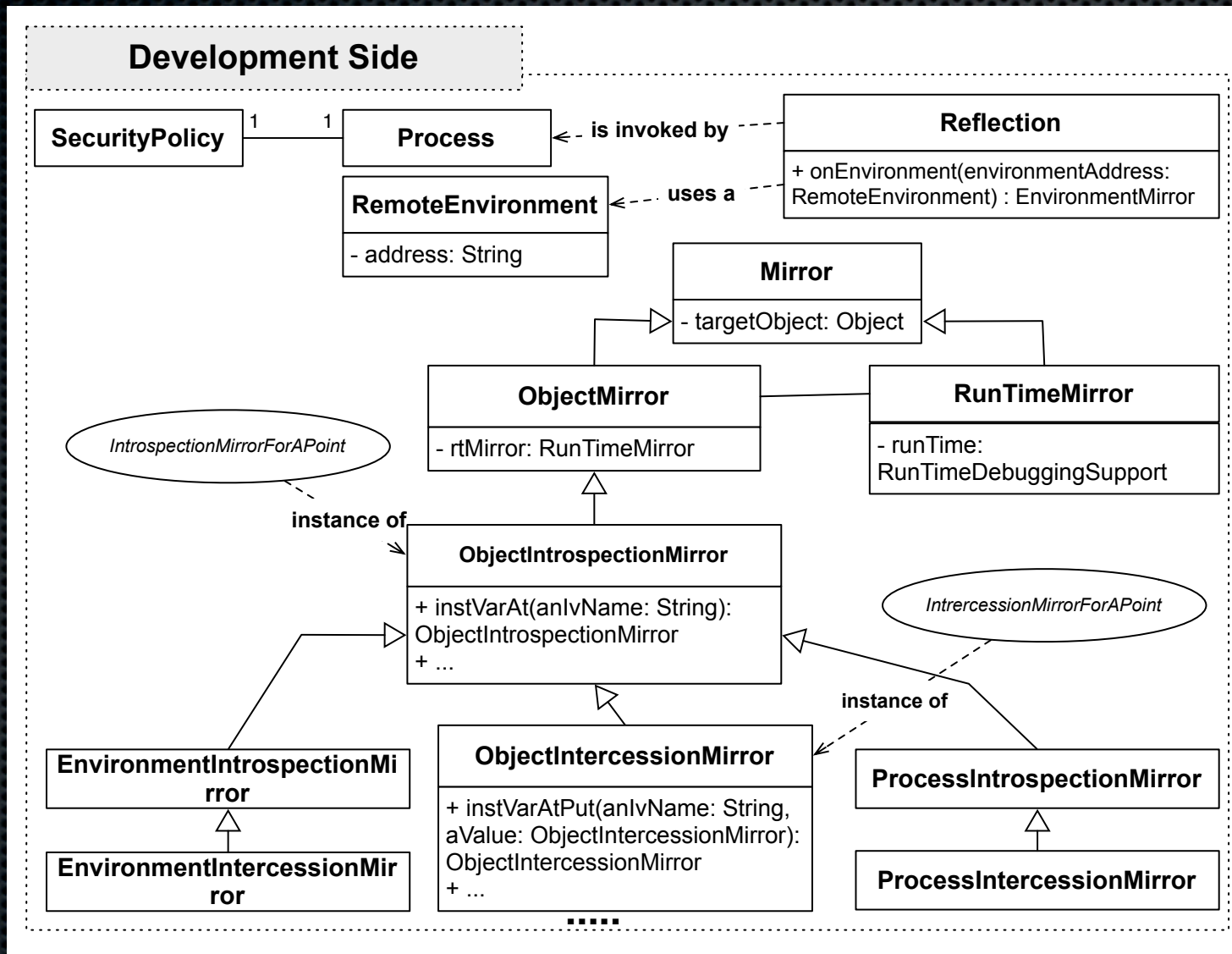
# Our Proposal - Overview



- Security - *security by decomposing and authenticating access to reflective facilities*











# Security





# Comparison

	 JPDA	 .NET	 GDB	 DCE	 JRebel JREBEL	 SMALLTALK	 BIFROST	 MERCURY
Interactiveness ( <b>6</b> )	1	1	1	6	6	6	6	6
Instrumentation ( <b>13</b> )	4	4	5	4	4	3	12	13
Distribution ( <b>+++</b> )	+	++	+	+	+	-	-	+++
Security ( <b>4</b> )	3	4	2	3	3	0	0	4



# Outline

- ✧ Introduction
- ✧ Related Work
- ✧ Contributions
- ✧ Implementation
- ✧ Validation
- ✧ Conclusion & Future Work



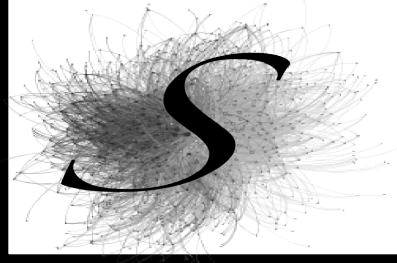
# Implementation Overview



**Mercury-Core**

**Mercury-Ui**

(Alexandria)



**Seamless**

**MetaStackVM**



- ✦ <http://ss3.gemstone.com/ss/Mercury-Prototype.html>
- ✦ <http://ss3.gemstone.com/ss/Seamless.html>
- ✦ <http://ss3.gemstone.com/ss/mSVM.html>



# Meta-Recursion - *mStackVM*

Interpreter

on: MessageReceived

for: anObject

do: [:reifications |

~~anObject incrementMessageCounter.~~

anObject

perform: reifications selector

withArguments: reifications arguments]

Ooops !





# MetaStackVM - Reflectogram

Interpreter

on: MessageReceived

for: anObject

do: [:reifications :reflectogram |

reflectogram disable.

anObject incrementMessageCounter.

reflectogram enable.

reflectogram

returnValue: reflectogram defaultAction]



# MetaStackVM - Reflectogram

Interpreter

on: Message

for: anObject

do: [:reification

reflect

anObj

reflect

reflect

return

## Reflectogram

- + ..
- + enable
- + disable
- + remove
- +...
- + defaultAction
- + returnValue:
- + ...
- + processMetaLevel
- + objectMetaLevel
- + ...
- + obj:perform:

counter.

defaultAction]



# Implementation Trade-offs

Supporting Interactiveness and Instrumentation



- ✦ ***Through Local Reflection***
- ✦ ***Through Virtual-Machine support***
- ✦ ***Through Byte-code manipulation***



# Benchmark

*Benchmark based  
on Tanter [Tanter 2003]*

- ✦ No Instrumentation
- ✦ Disabled Instrumentation
- ✦ Enabled Instrumentation

 Bifrost	 Mercury
1x	1x
1x	1x
35x	8x

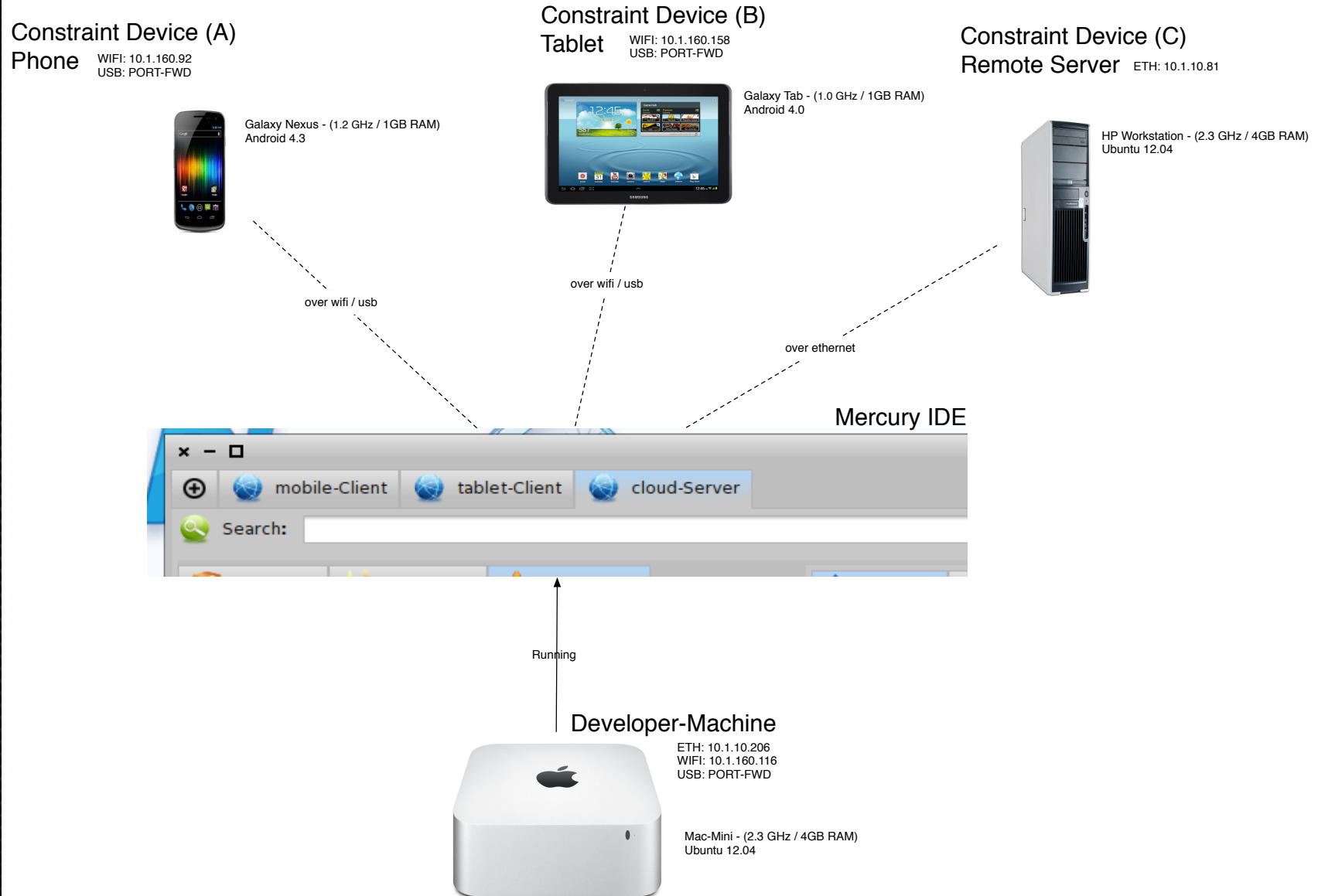


# Outline

- ✧ Introduction
- ✧ Related Work
- ✧ Contributions
- ✧ Implementation
- ✧ Validation
- ✧ Conclusion & Future Work

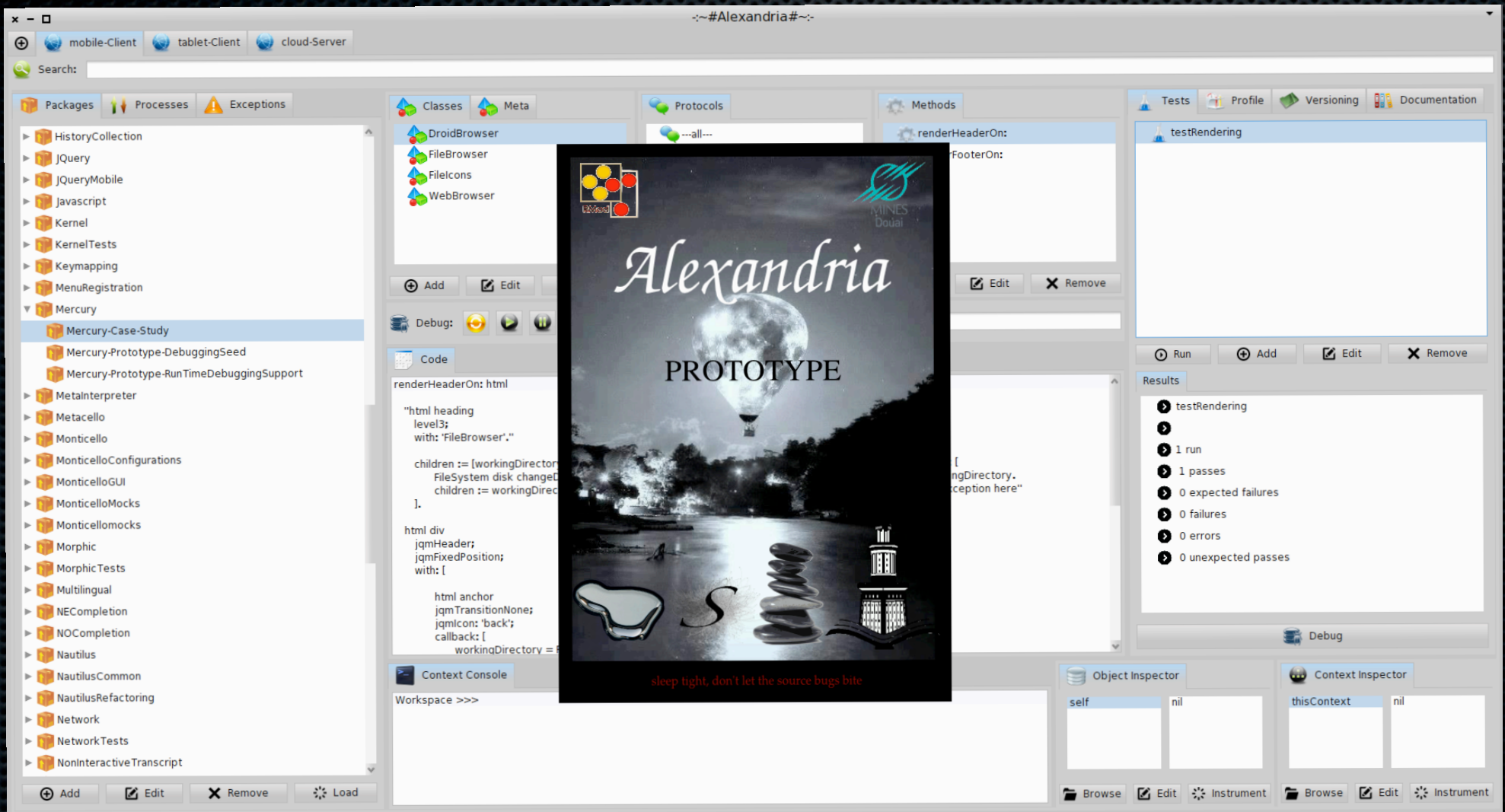


# Experimental Setting





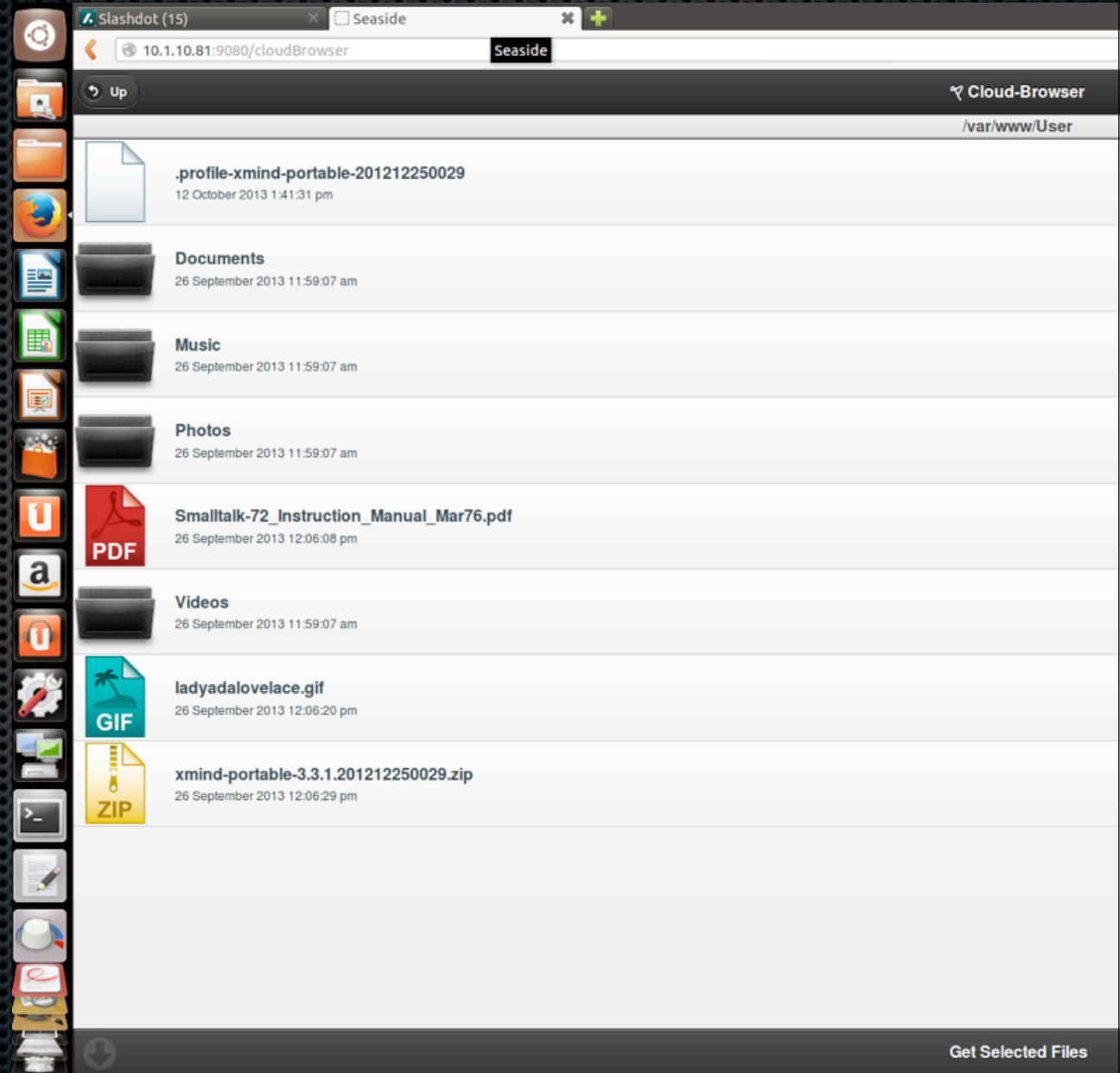
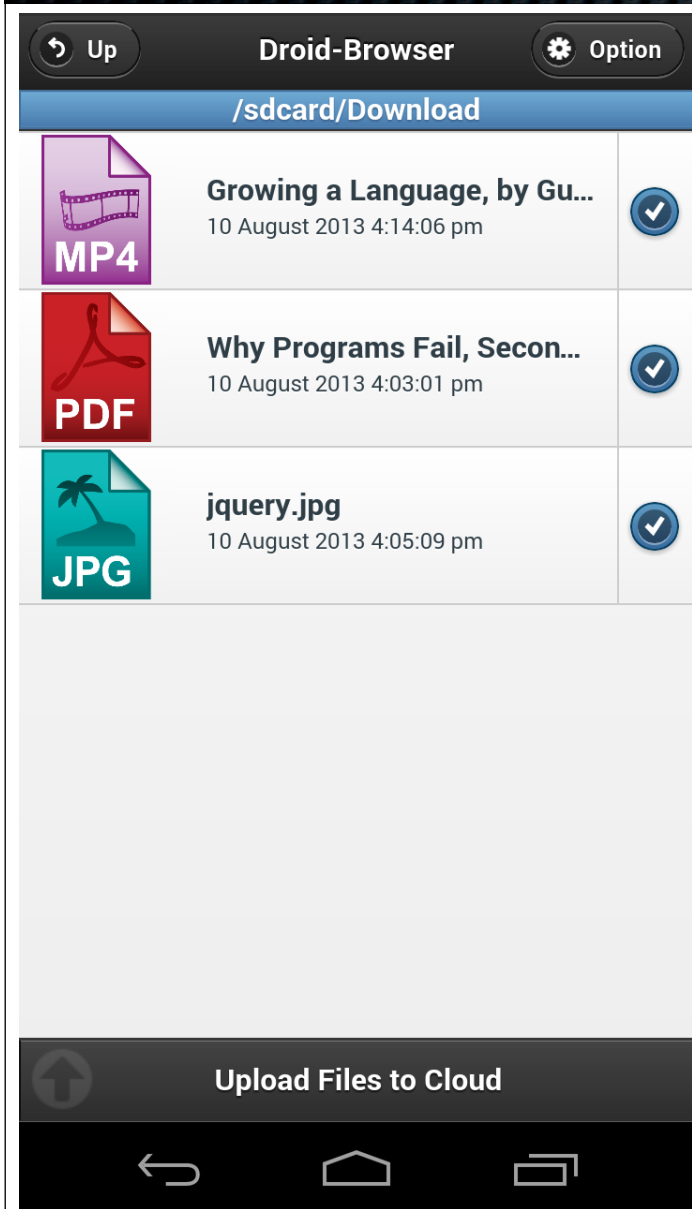
# Alexandria





# Experimental Setting

47





# Objectives

- **Verify the applicability** of Mercury for these constrained debugging targets.
- **Illustrate how** a debugging session benefits from Mercury's properties

Case-Study I

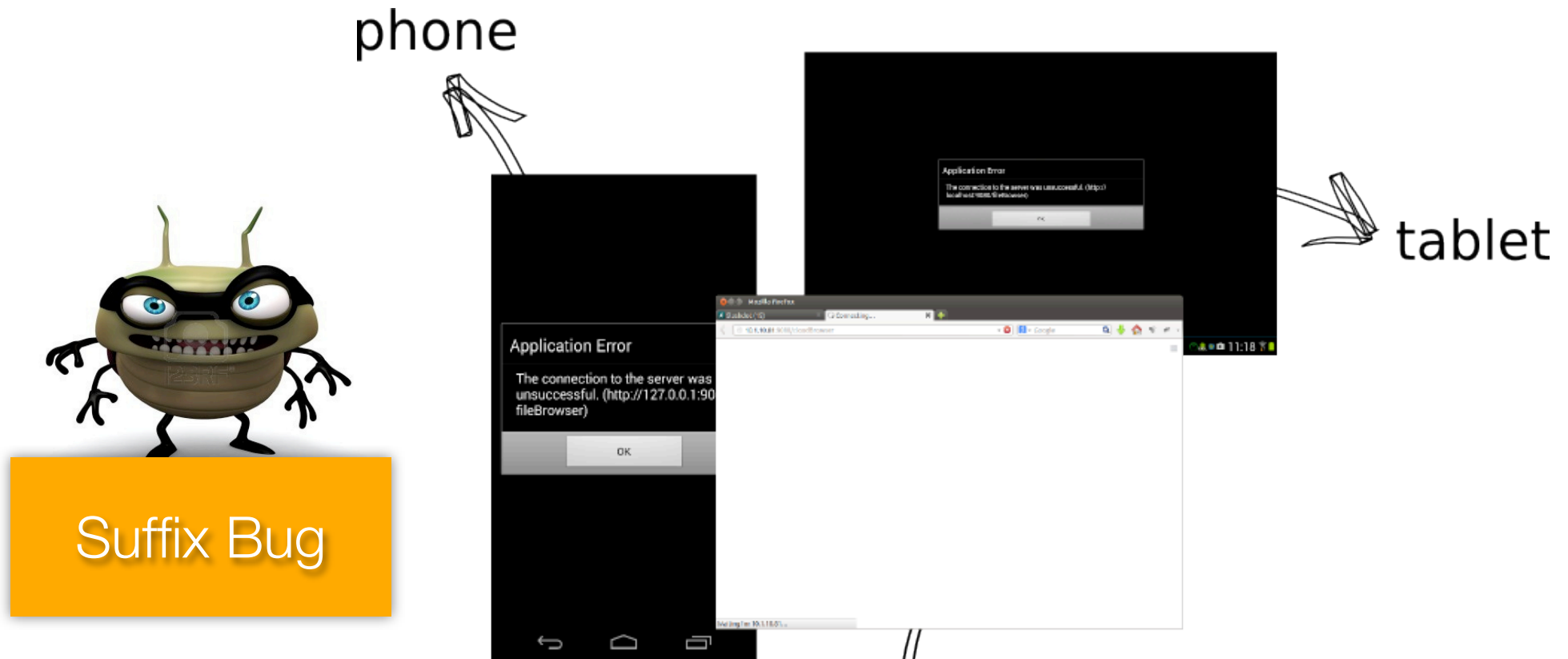
**Remote Agile  
Debugging**

Case-Study II

**Remote Object  
Instrumentation**



# Remote Agile Debugging



Phone: '/charger'

Tablet: '/default.prop'

Server: '/User/.profile-xmind-portable-201212250029'



# Remote Agile Debugging

## Test Device (A)

WiFi: 10.1.160.92  
USB: PORT-FWD



Galaxy Nexus - (1.2 GHz / 1GB RAM)  
Android 4.3

## Constraint Device (B) Tablet

WiFi: 10.1.160.158  
USB: PORT-FWD

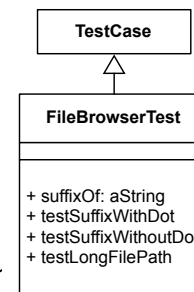
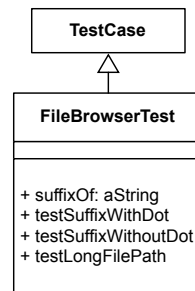
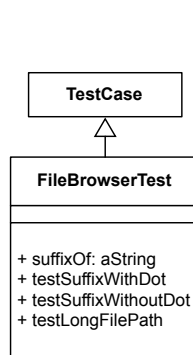


Galaxy Tab - (1.0 GHz / 1GB RAM)  
Android 4.0

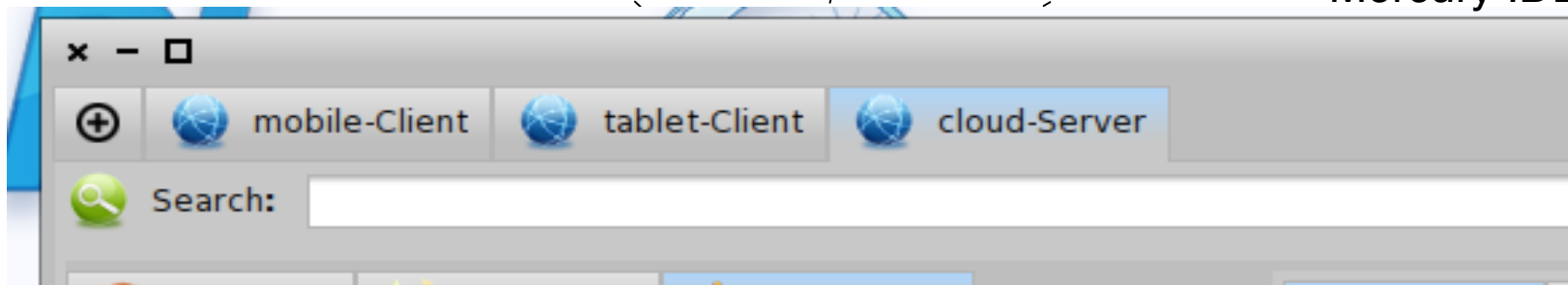
## Constraint Device Remote Server



HP Workstation  
Ubuntu



Mercury IDE





# Remote Agile Debugging

51

The screenshot displays the Xcode IDE interface during a test run. On the left, the 'Tests' pane lists three test cases: 'testSuffixWithDot', 'testSuffixWithoutDot', and 'testHiddenFilePath'. Below this, the 'Results' pane shows a summary: '1 run', '1 passes', '0 expected failures', '0 failures', '0 errors' (circled in red), and '0 unexpected passes'. On the right, the 'Exceptions' pane lists several 'NotFound' errors and two 'TestFailure: Assertion failed' messages. A stack trace below shows the call path: 'FileBrowserTest(TestCase)>>signalFailure:', 'FileBrowserTest(TestCase)>>assert:', and 'FileBrowserTest>>testSuffixWithDot'. The 'Code' pane shows the source code for 'testSuffixWithDot', with the line 'self assert: (self suffixOf: 'filename.ext') = 'ext'' highlighted. Hand-drawn arrows indicate the flow of information: one arrow points from the 'TestFailure' messages in the Exceptions pane to the '0 errors' in the Results pane, labeled 'Re-produced Errors as Failed Assertions'; another arrow points from the 'TestFailure' messages to the 'Initial Error' text, labeled 'Initial Error'; and a third arrow points from the 'Initial Error' text to the 'Re-produced Errors from test cases' text.

Tests

- testSuffixWithDot
- testSuffixWithoutDot
- testHiddenFilePath

Run Add Edit Remove

Results

- testSuffixWithoutDot
- 1 run
- 1 passes
- 0 expected failures
- 0 failures
- 0 errors
- 0 unexpected passes

Debug

Exceptions

- NotFound: [i | (self at: i) = dot] not found in Interval
- NotFound: [i | (aString at: i) = dot] not found in Interval
- NotFound: [i | (aString at: i) = dot] not found in Interval
- NotFound: [i | (aString at: i) = dot] not found in Interval
- TestFailure: Assertion failed
- TestFailure: Assertion failed

Initial Error

Re-produced Errors from test cases

Stack

- FileBrowserTest(TestCase)>>signalFailure:
- FileBrowserTest(TestCase)>>assert:
- FileBrowserTest>>testSuffixWithDot

Code

```
testSuffixWithDot
self assert:
    (self suffixOf: 'filename.ext') = 'ext'
```



# Objectives

- **Verify the applicability** of Mercury for these constrained debugging targets.
- **Illustrate how** a debugging session benefits from Mercury's properties

Case-Study I

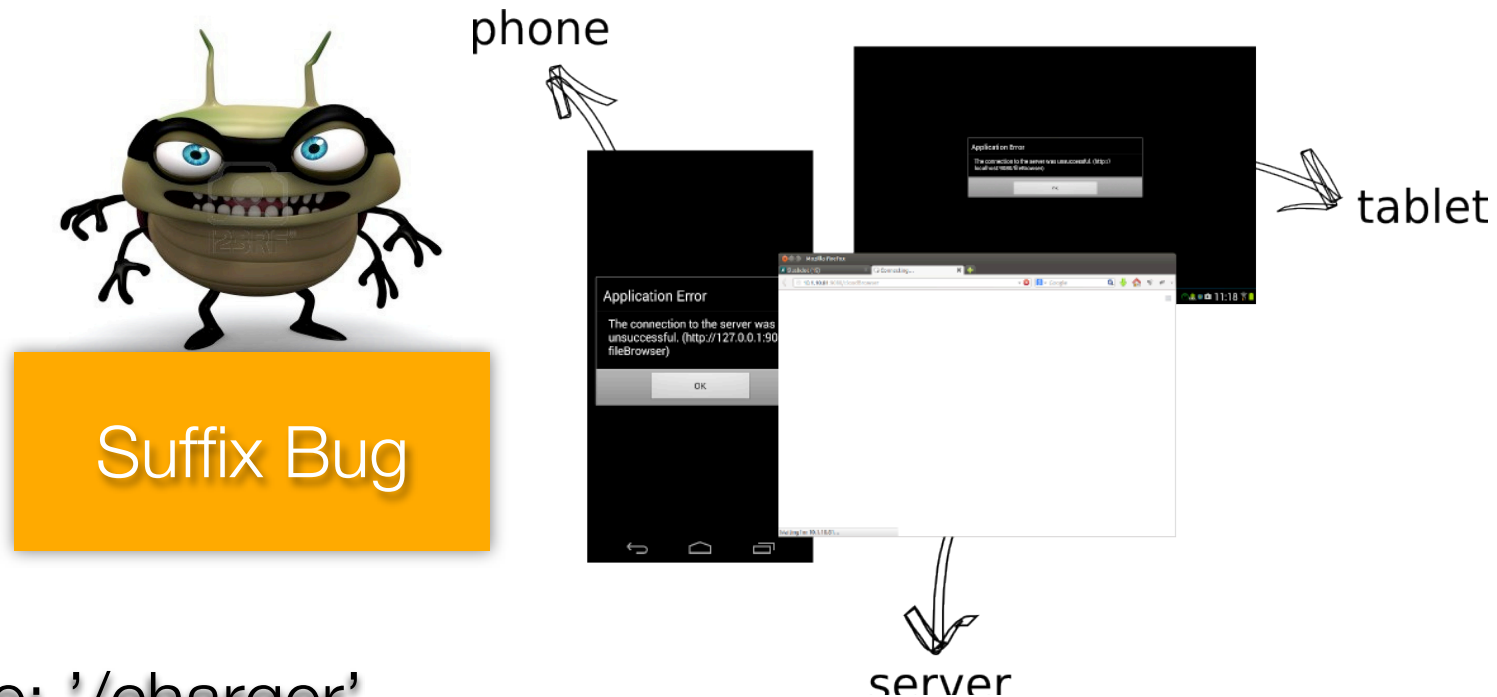
**Remote Agile  
Debugging**

Case-Study II

**Remote Object  
Instrumentation**



# Remote Object Instrumentation



Phone: '/charger'

Tablet: '/default.prop'

Server: '/User/.profile-xmind-portable-201212250029'



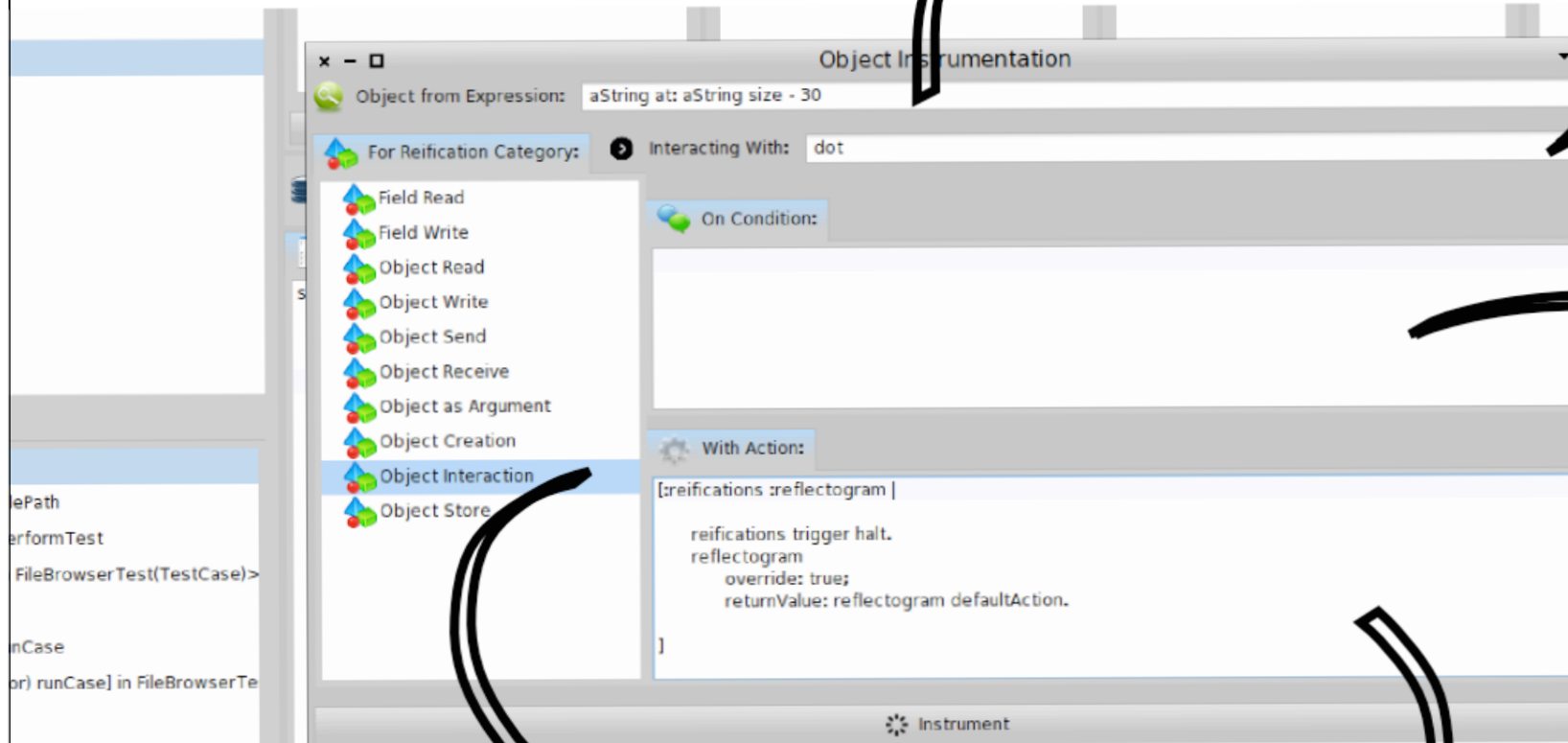


# Remote Object Instrumentation

Mirror of ②  
Target Object

③

Mirror of  
Interacting  
Object



① Semantical  
Event

Optional  
Condition  
Of Meta-Ac

④ Meta-Action



# Remote Object Instrumentation

① Assertion Halted on Semantic Event

Stack

- Character(Object) >> halt
- [t1 :t2 | t1 trigger halt.t2 override: true; returnValue: ...]
- [reifications: reflectogram | (reifications message argument ...]
- Interpreter >> object:metaReceived:
- [i | (aString at: i) = dot] in [(aString size to: 1 by: -1) detect ...]**
- [reach | (aString value: eachIfTrue: [^ each] nil) in Interval(C ...]

```
suffixOf: aString
"assumes that I'm a file name, and answers my suffix, the part after the last dot"
| dot dotPosition |
dot := FileDirectory dot.
dotPosition := [(aString size to: 1 by: -1) detect: [ i | (aString at: i) = dot ]] on: NotFound do: [
^ aString copyFrom: dotPosition + 1 to: aString size
```

Object Inspector

self	value
self	\$.
value	thi

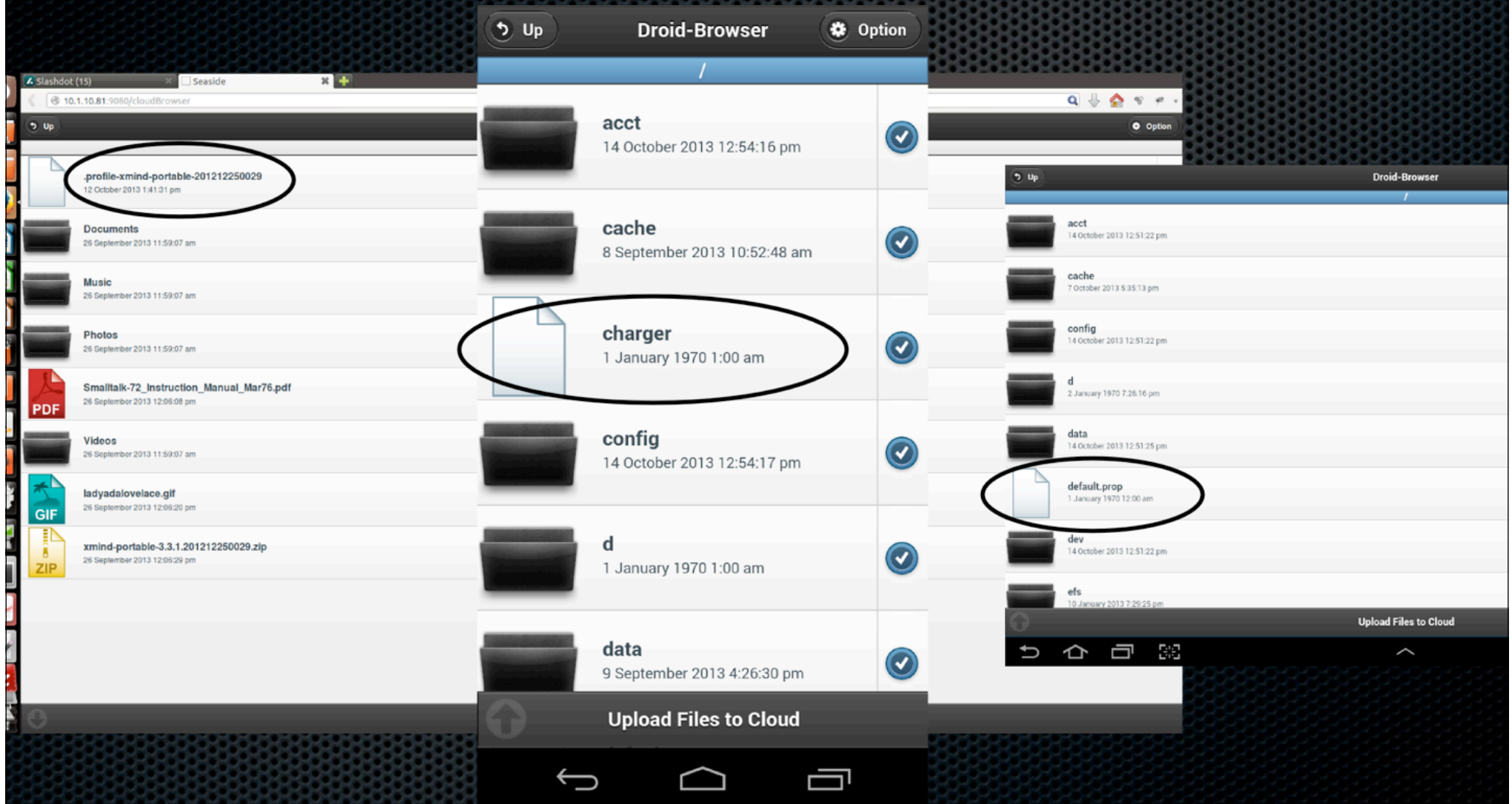
Context Inspector

thisContext	dotPosition	i	dot	aString
thisContext	dotPosition	i	dot	aString

② Involving two interacting remote objects



# Results





# Outline

- ✧ Introduction
- ✧ Related Work
- ✧ Contributions
- ✧ Implementation
- ✧ Validation
- ✧ Conclusion & Future Work



# Summary - Contributions

- **Identification** of four desirable properties for remote debugging: ***interactiveness, instrumentation, distribution and security.***
- The **definition of a model** for remote debugging (**Mercury**) that exhibits these desirable properties.
- A **solution to the problem of Reflective-Data** [Maes 1987b] in the context of mirrors [Bracha 2004] (**MetaTalk**)



# Contributions

- The **reification** of a previously illustrative notion (that of the **reflectogram** [Tanter 2003])
- **Prototype implementation** of our model **for remote debugging** in the context of reflective languages.
- **Implementation of an adaptable middleware** [David 2002] for supporting distribution (**Seamless**).
- **Implementation of a dedicated VM** for Pharo (**MetaStackVM**) for advanced intercession facilities.



# Diffusion of Results

- **Submitted/Published:** Nikolaos Papoulias, Noury Bouraqadi, Marcus Denker, Stéphane Ducasse and Luc Fabresse. **Towards Structural Decomposition of Reflection with Mirrors**. In Proceedings of International Workshop on Smalltalk Technologies (IWST'11), Edinburgh, United Kingdom, 2011. 105
- **Conference Talk:** Nikolaos Papoulias. **Seamless -- Let a thousand systems bloom**. 20th International Smalltalk Conference, Ghent, Belgium, 2012.
- **To Be Submitted:** Nikolaos Papoulias, Noury Bouraqadi, Luc Fabresse, Marcus Denker and Stéphane Ducasse. **Mercury: Live Remote Debugging in Reflective Languages**. To be submitted in The Journal of Object Technology.
- **To Be Submitted:** Nikolaos Papoulias, Stéphane Ducasse, Marcus Denker, Guillermo Pollito, Noury Bouraqadi and Luc Fabresse. **MetaTalk: Designing a Language with a Pluggable Meta-Level**. To be submitted in The Journal for Universal Computer Science.
- **Invited Chapter:** Nikolaos Papoulias. **Seamless: an Adaptable Middleware Solution**. Invited Chapter to be submitted for the forthcoming book Pharo in the Enterprise, by Square Bracket Associates.



# Future Work

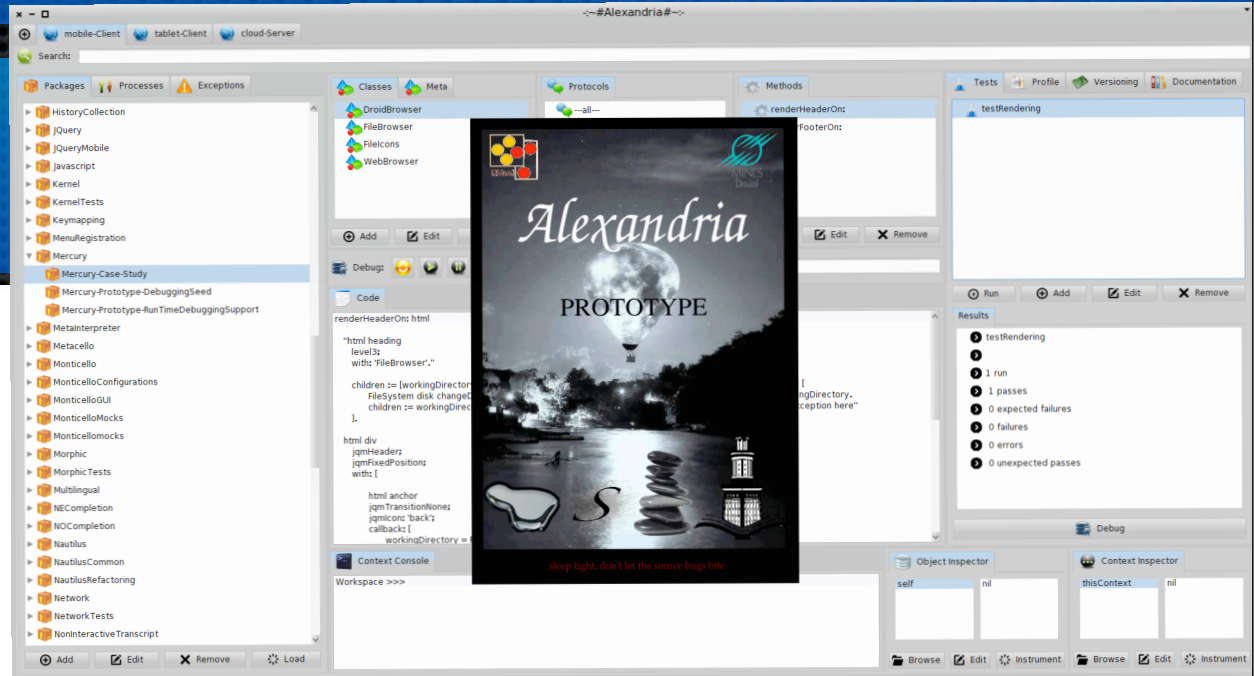
- ✦ *Language and **Virtual-Machine Debugging** in the Same Model*
- ✦ *Integration of **Automated** Debugging Techniques (e.g delta-debugging) in **Developer-Driven Debugging***



# Mercury

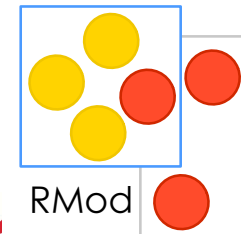
# Thank you !

## MetaTalk



**Mercury-Core**

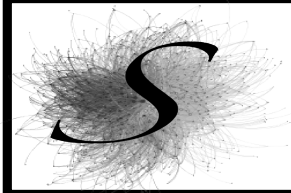
**Mercury-Ui**  
(Alexandria)



RMod

*Inria*

Nick Papoulias  
2013



**Seamless**

**MetaStackVM**





**Some time left ?  
Groovy !!**



# Collaborations

- ***Seamless as a library*** for the Continuous integration services of Pharo.
- ***The MetaTalk model as a case-study*** for bootstrapping OO - languages.
- ***Mercury integration*** with the PhaROS robotic middleware (on-going effort).



# Design Patterns - Mirrors

- ✦ Explicit meta-object
- ✦ Abstract class / Interface
- ✦ Factory
- ✦ Facade and Bridge



# Conditional Meta-Action

```
[:reifications :reflectogram |
```

```
    reifications trigger halt.
```

```
    reflectogram
```

```
        override: true;
```

```
        returnValue: reflectogram defaultAction.
```

```
]
```



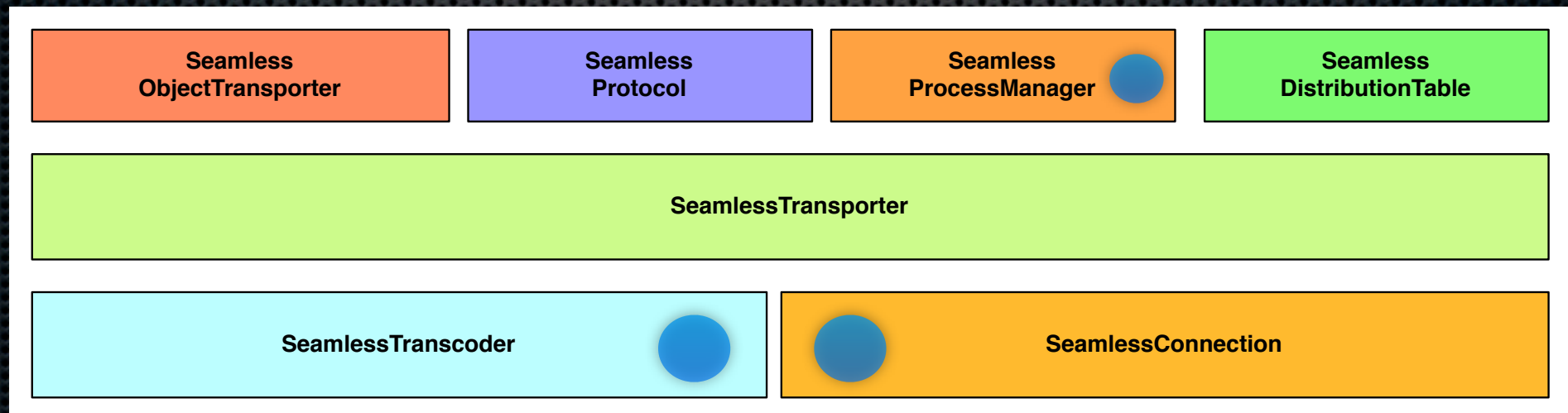
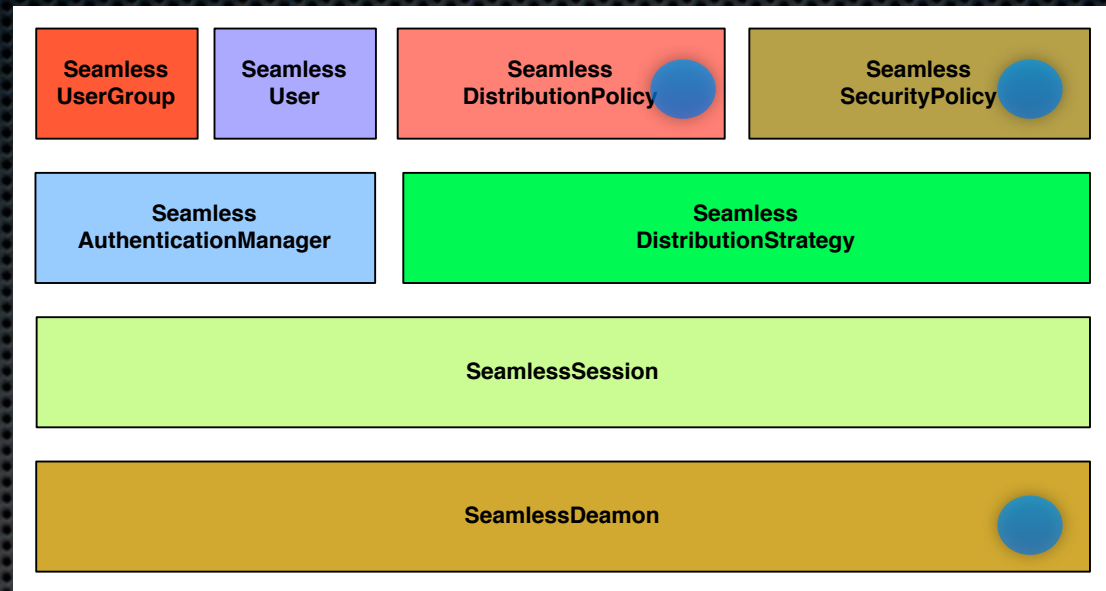
# Seamless Initialization

SeamlessDeamon class>>newDefaultWithGlobalAccess

```
^ self new
  buildWithTransporterClass: SeamlessSocketStreamTransporter
  transcoderClass: SeamlessFuelTranscoder
  proxyClass: SeamlessFastDNUProxy
  garbageCollectorClass: SeamlessDefaultGarbageCollector
  andAuthenticationManager: ((SeamlessAuthenticationManager new)
    addGroup: [...]
    withPolicy: (SeamlessDistributionPolicy
      newWithEntryPoint: [...]
      classesToPassByValue: [...]
      classesToPassByShallowCopy: [...]
      andSecurityPolicy: (SeamlessSecurityPolicy
        newWithClassesToPassByReference: [...]
        classesNotToReference: [...]
        includingMessages: [...]
        excludingMessages: [...]);
      addUser: [...] withPassword: [...] inGroup: [...])).
```



# Seamless





# ***JMercury** - Our model on top of Java*

- ✦ **JPDA + DCE VM** (Interactiveness)
- ✦ **Reflex / ASM / JavaAssist**  
(Instrumentation)
- ✦ **Cajo Project** (Distribution)
- ✦ **Decomposed Hierarchy** of Mirrors / Closer  
integration with **SecurityManager** (Security)



# Emulators - Field Experience

- ✦ **iPhone/Android emulators** (different models - versions of OSes - gyroscopes - touch gestures ...)
- ✦ **Car-Team experience -- RoboShop 2013 Demo** (unanticipated changes - people walking by - glass walls ...)