



# The Pharo Debugger and Debugging tools

## Advances and Roadmap

**Steven Costiou**

**Valentin Bourcier**



# This talk

A two-part talk

- **Part 1 - Advances and Roadmap**

- The team and its organization
- The latest improvements
- The roadmap

- **Part 2 - Demos**

# Research objective

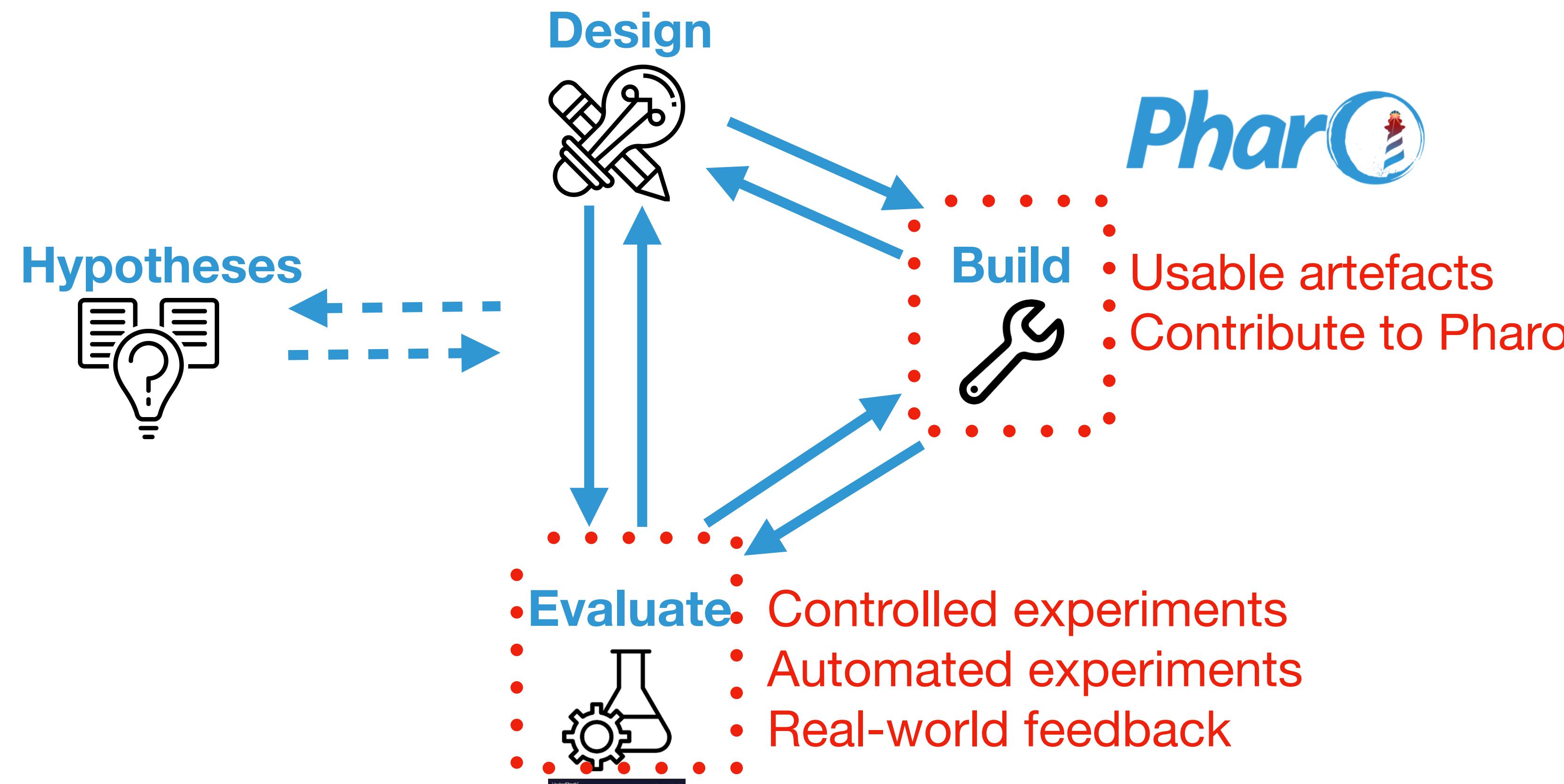
Discover and understand what are the best debugging methods for object-oriented systems so that we can **build better debugging tools** in order **to facilitate debugging** (faster, easier, cheaper)



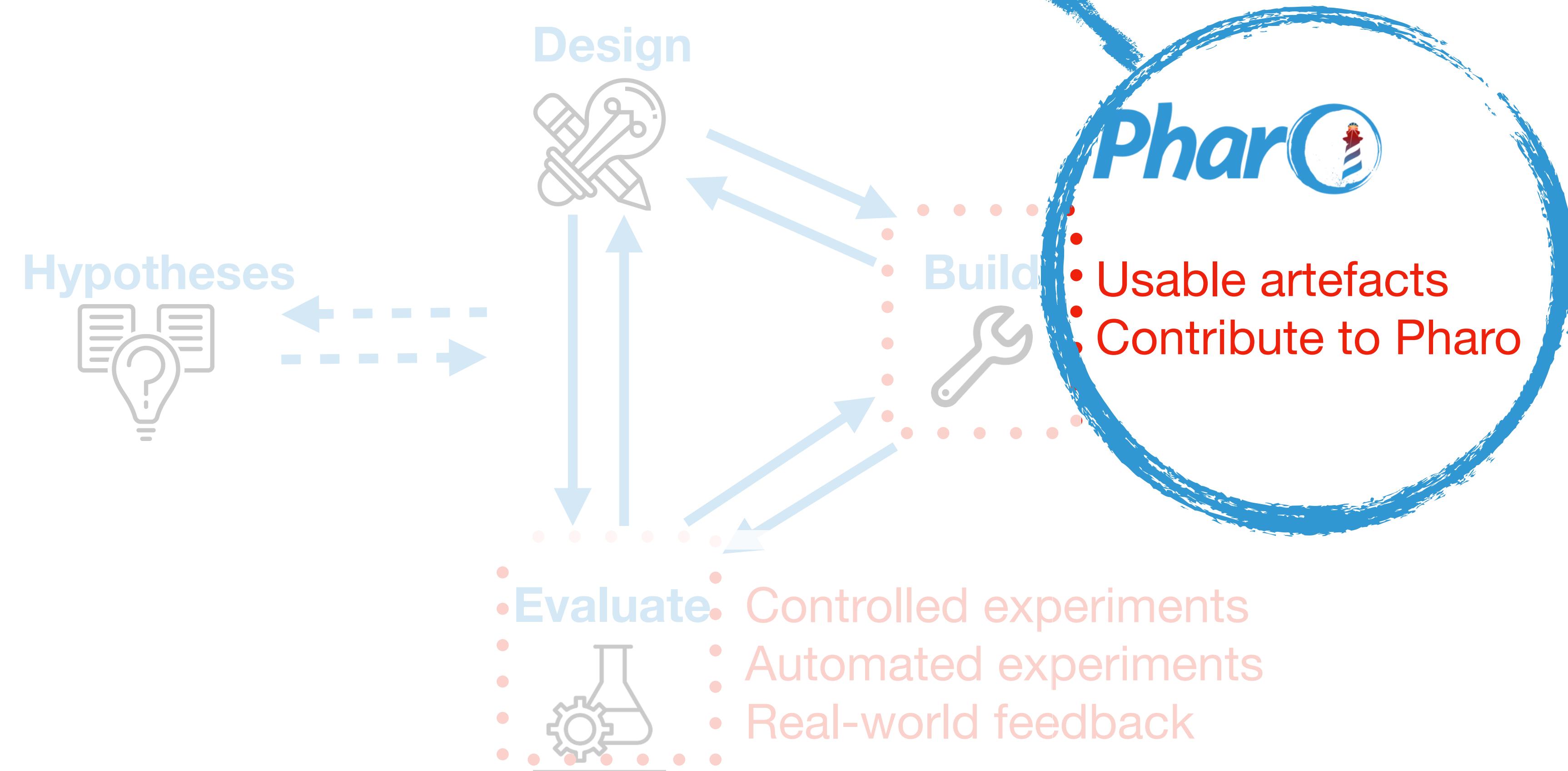
# Research topics

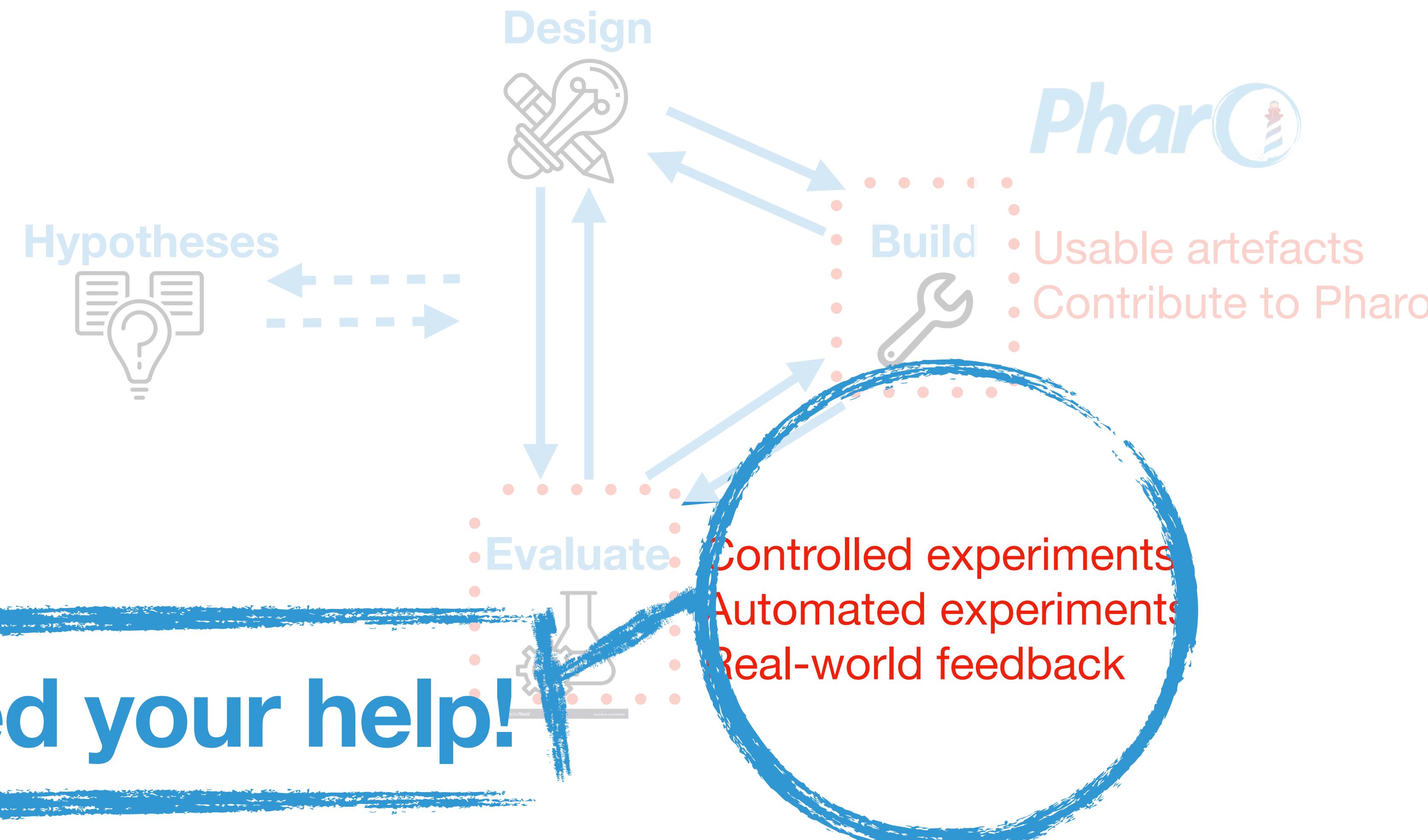
- Time-Traveling Debugging
- Object-Centric Debugging
- Language Support for Tool Implementation
  - Modular debugging infrastructures
  - Reflection techniques
  - Frameworks and tools

# Research method



# We need a strong debugger!





We need your help!



# Research Experiments

What you should expect if you participate

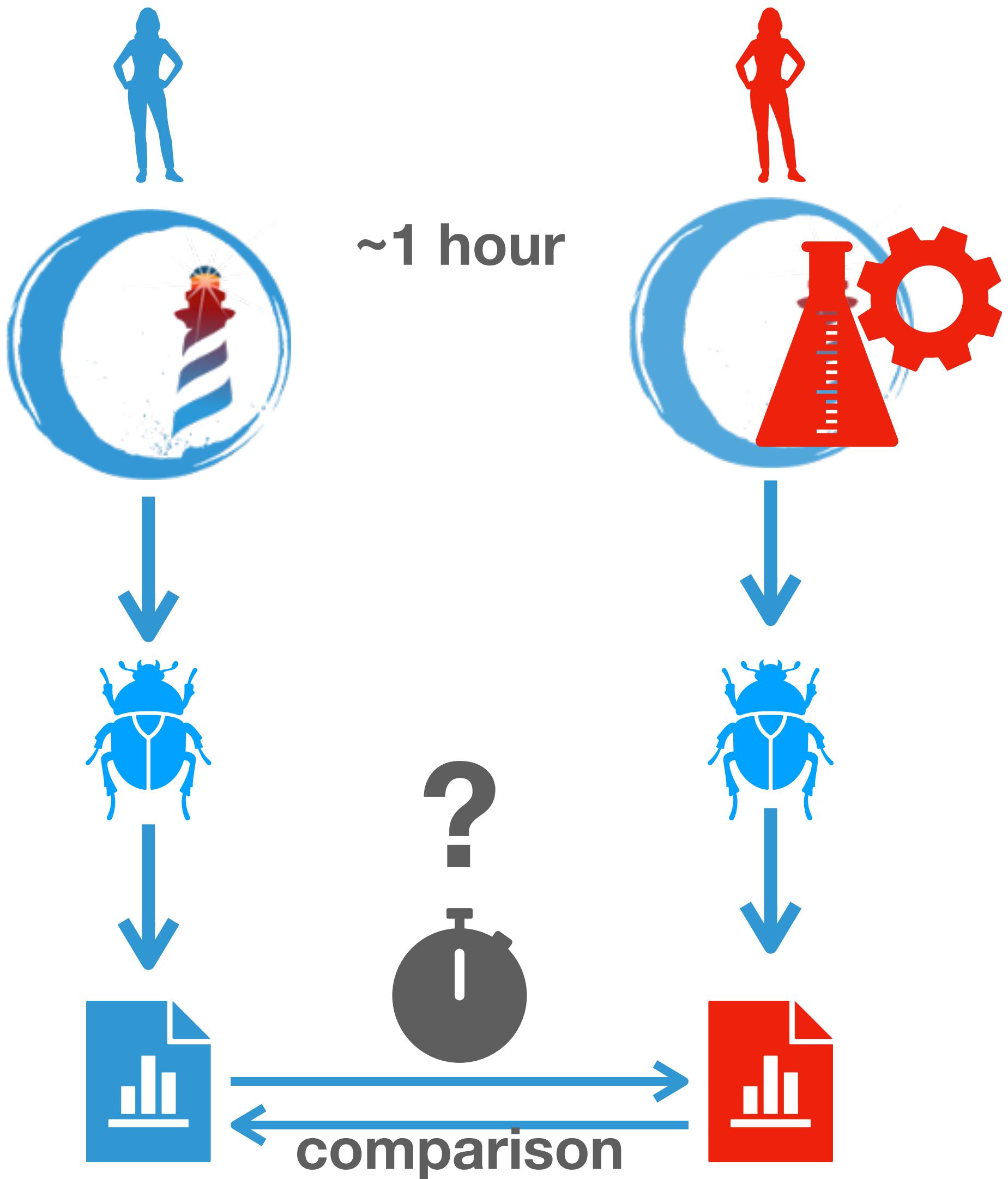
- **Surveys and interviews**
- **Empirical evaluations**
  - Between participants design
  - Within participants design
  - Data collection



# Research Experiments

What you should expect if you participate

- Surveys and interviews
- Empirical evaluations
  - Between participants design
  - Within participants design
- Data collection

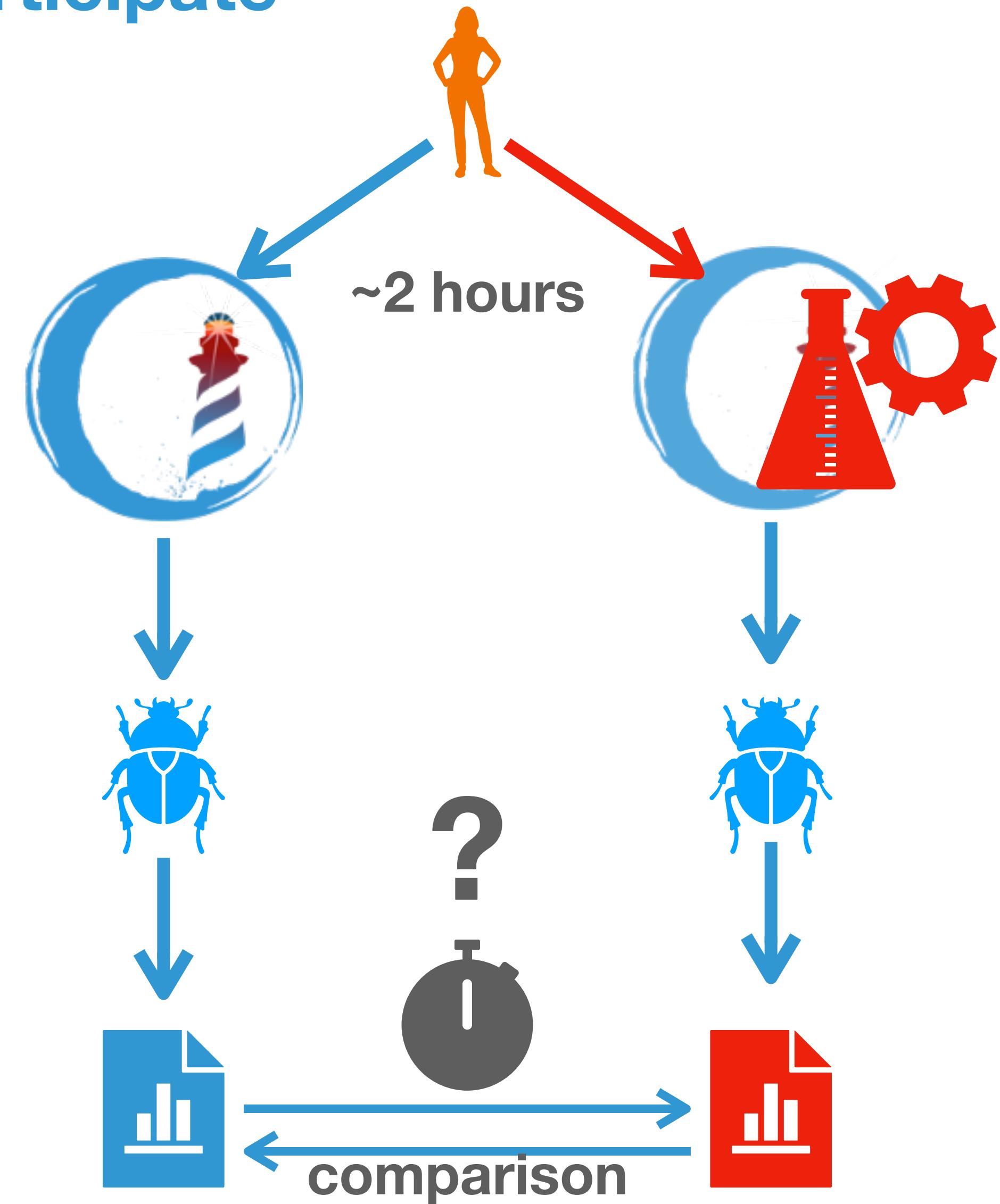




# Research Experiments

What you should expect if you participate

- Surveys and interviews
- Empirical evaluations
- Between participants design
- Within participants design
- Data collection

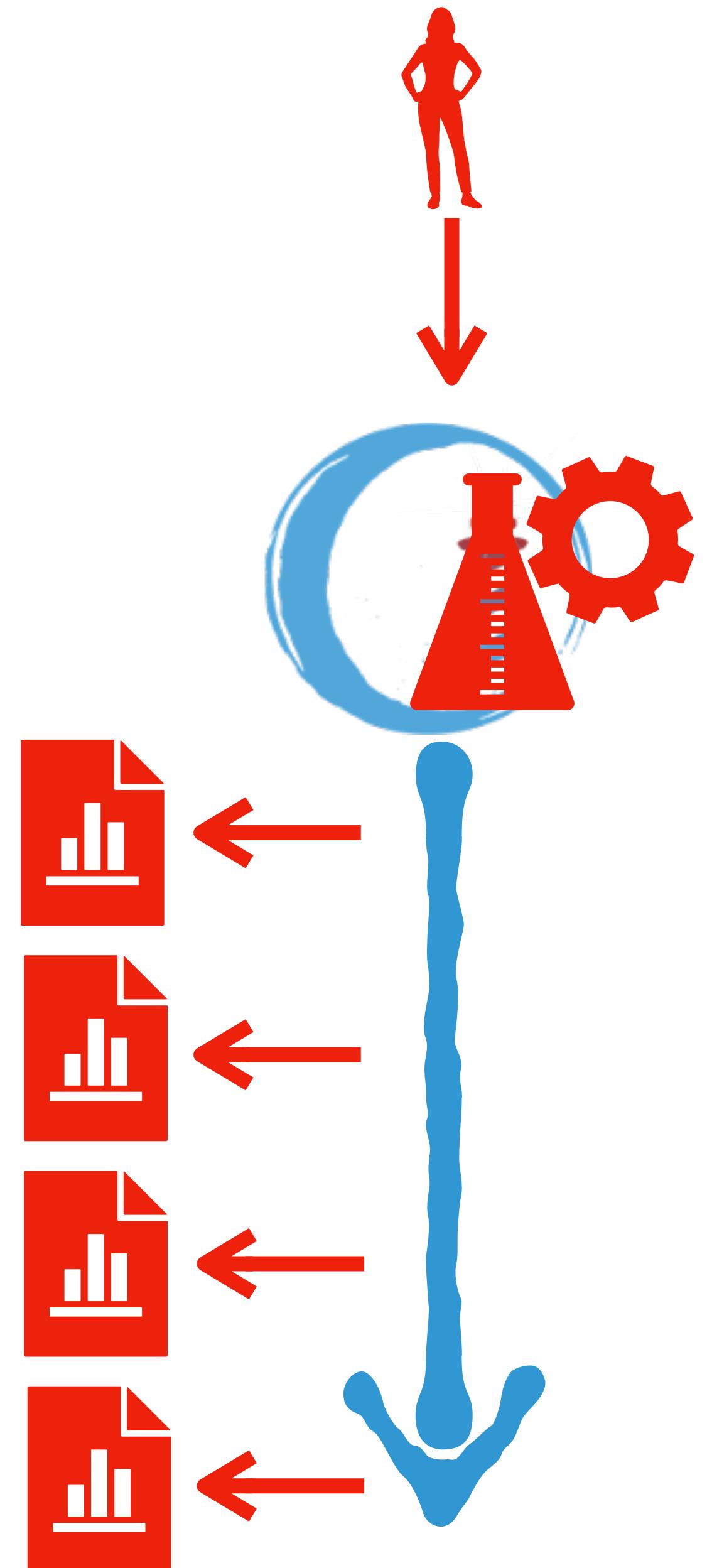




# Research Experiments

What you should expect if you participate

- Surveys and interviews
- Empirical evaluations
  - Between participants design
  - Within participants design
  - Data collection





# Research Experiments

What you should expect if you participate

- **Surveys and interviews**
- **Empirical evaluations**
  - Between participants design
  - Within participants design
  - Data collection
- **Anonymous automated data collection**
  - The data goes into a secure server hosted at Inria
  - Once the data is in, we can't tell it's yours



# Next Experiments

Would you want to participate?

- **Debugger Driven Development / XTDD**
- **Object-Centric Breakpoints**
- **Time-Traveling Object-Centric Debugging**



# The Debugging Department



**Steven Costiou**  
Researcher



**Adrien Vanègue**  
Engineer



**Valentin Bourcier**  
PhD Student



# The Debugging Department



**Steven Costiou**  
Researcher



**Adrien Vanègue**  
Engineer



**Valentin Bourcier**  
PhD Student



**THALES**

 **SCHMIDT**  
GmbH  
Ingenieurbüro für Bauwesen

**anr<sup>®</sup>**



# Improvements and new tools

Since 2020

- **Bugfixes**

- 136 bug fixes
- 36 issues open with « debugger » somewhere in the issue description or discussion
- Tests!

- **Infrastructure improvements**

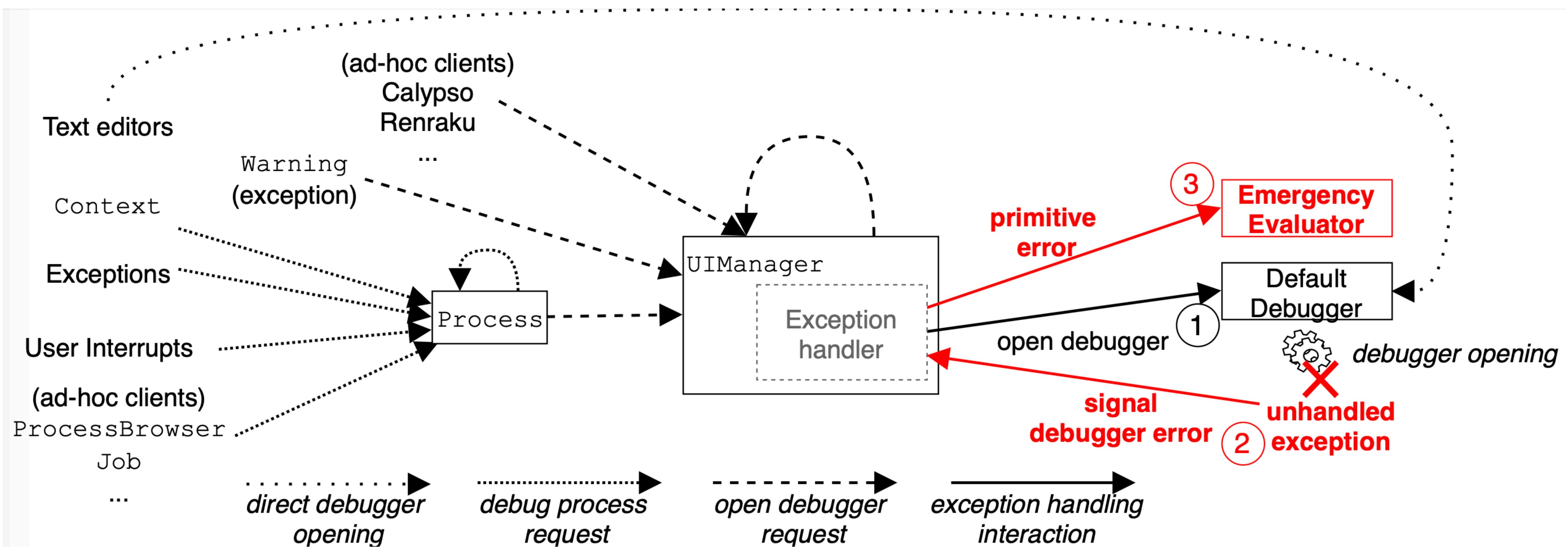
- **Architectural improvements**

- **New tools**



# Infrastructure

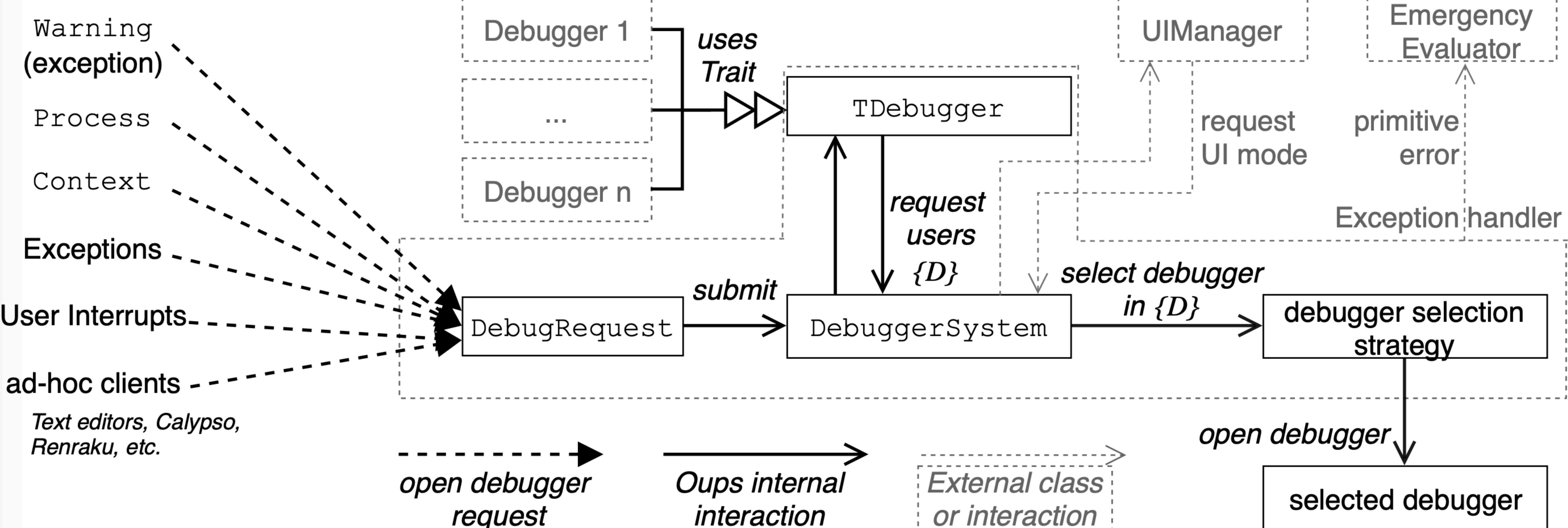
## Before





# Infrastructure

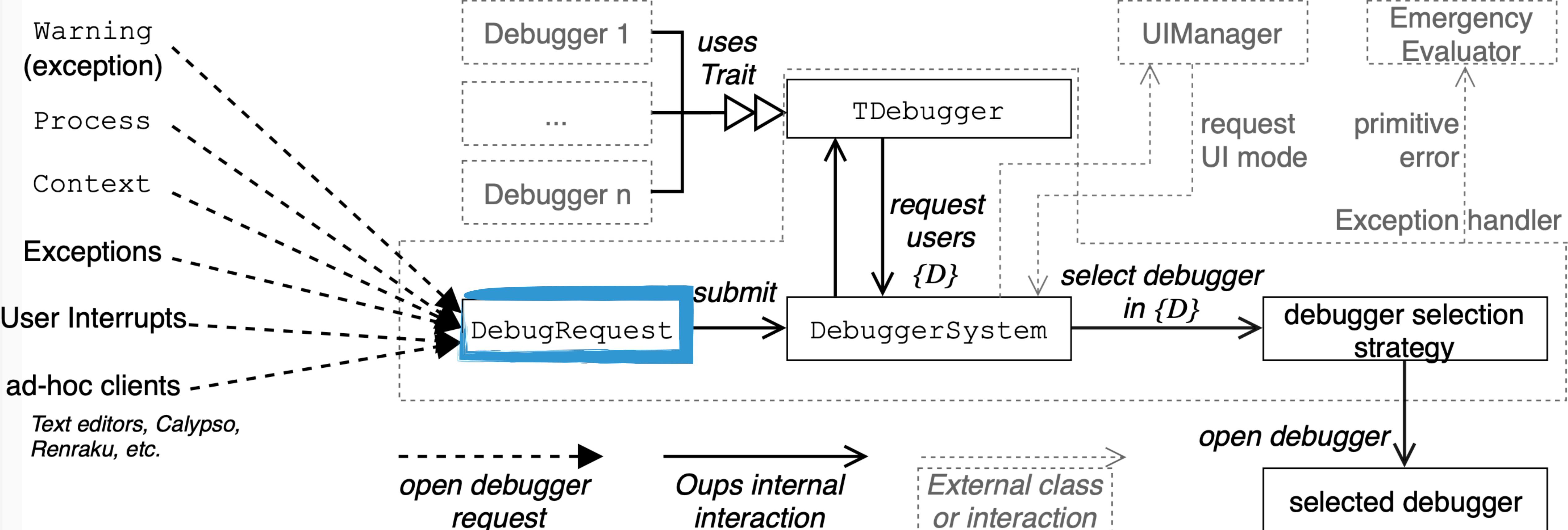
## Now





# Infrastructure

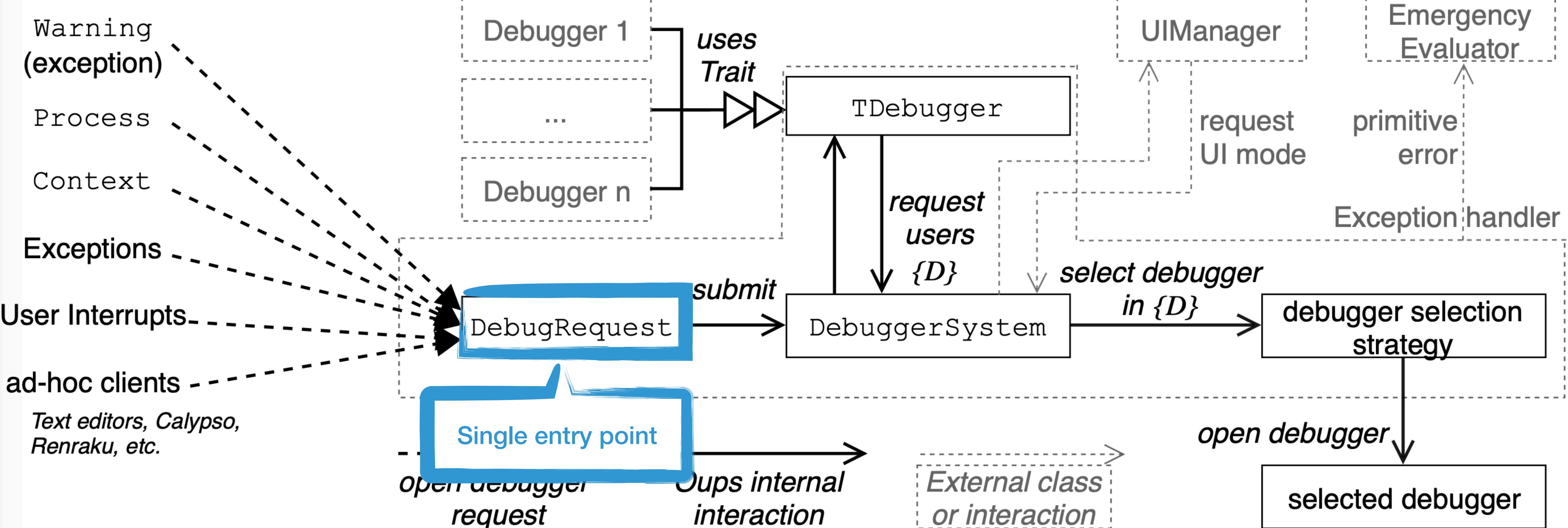
## Now





# Infrastructure

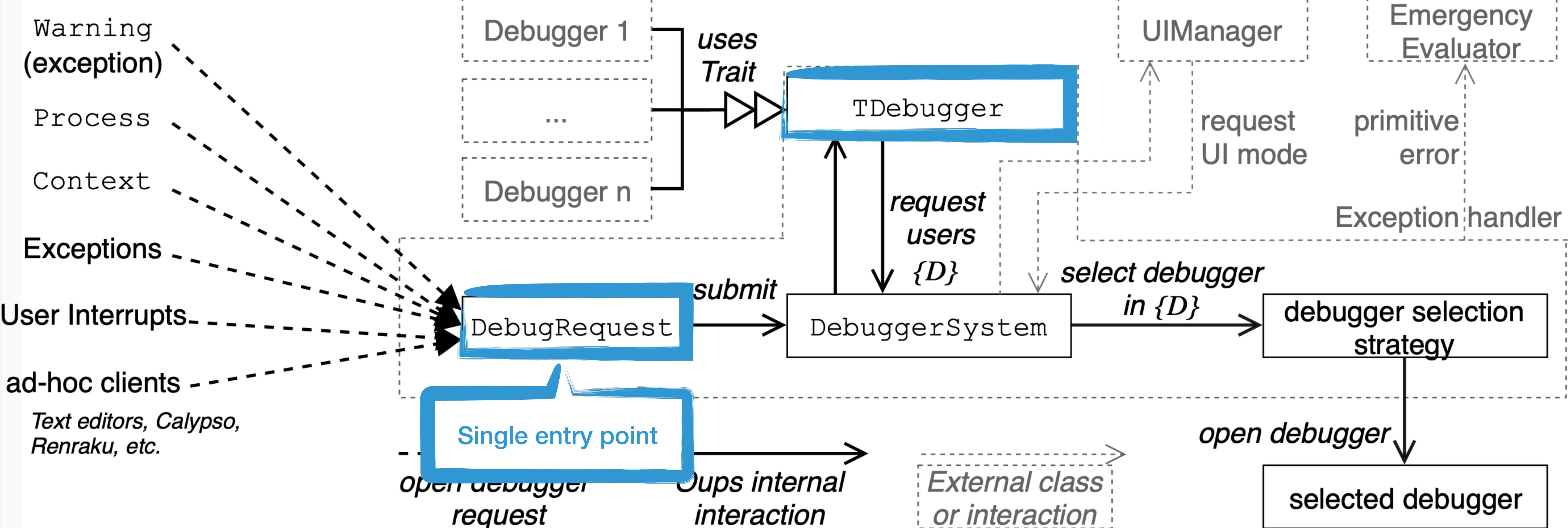
## Now





# Infrastructure

## Now

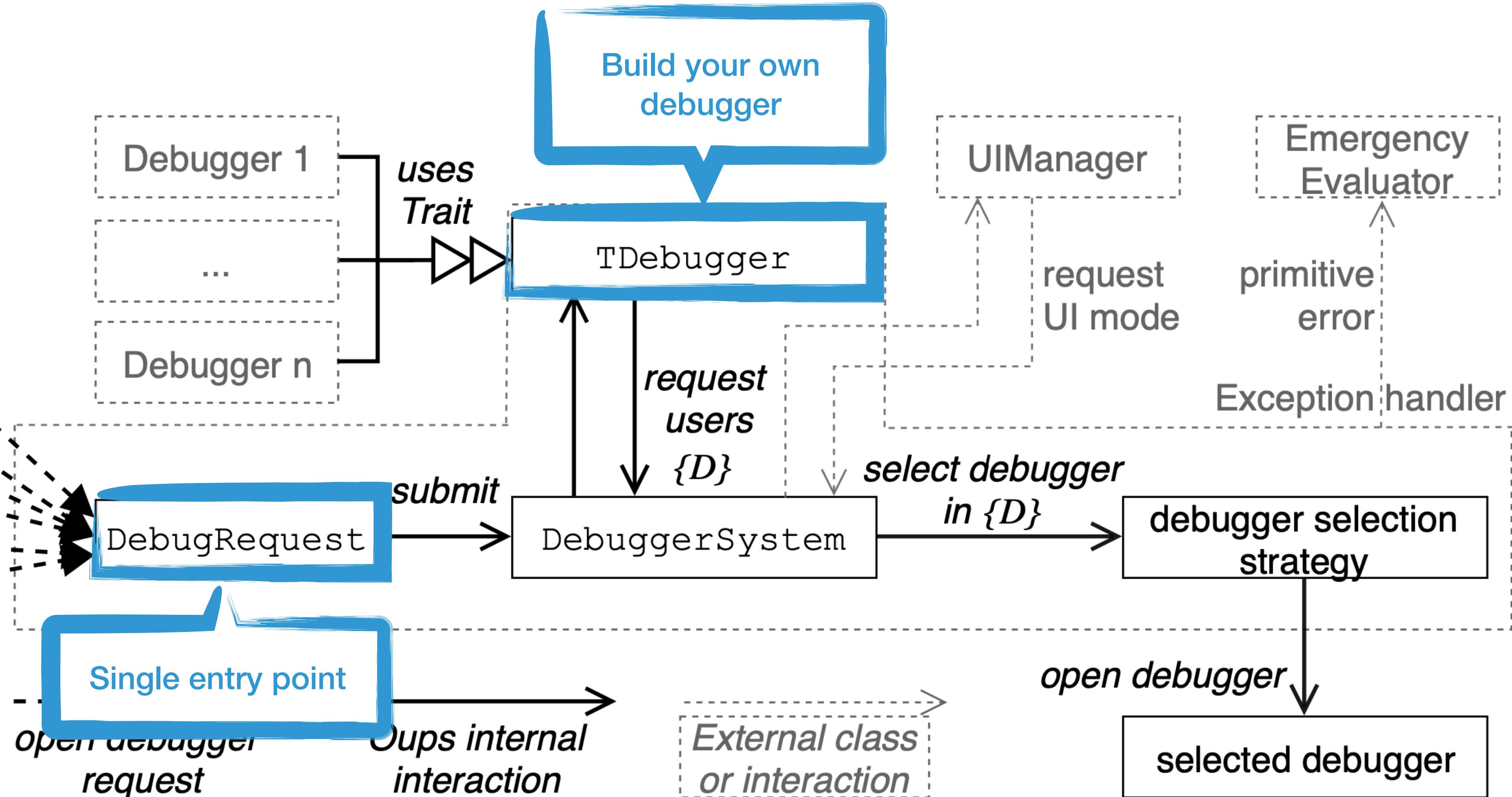




# Infrastructure

## Now

Warning (exception)  
Process Context  
Exceptions  
User Interrupts  
ad-hoc clients  
*Text editors, Calypso, Renaku, etc.*

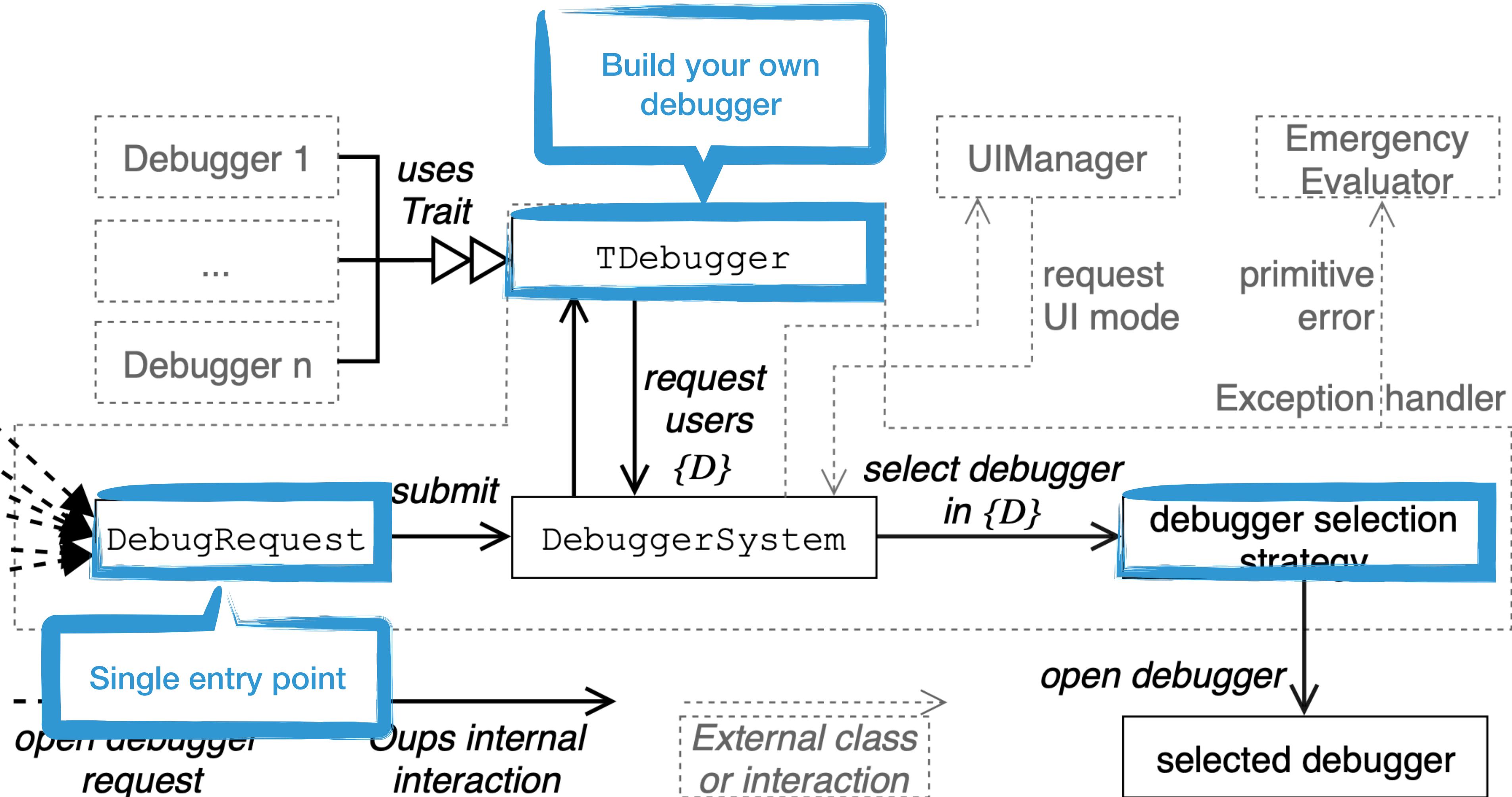




# Infrastructure

## Now

Warning (exception)  
Process Context  
Exceptions  
User Interrupts  
ad-hoc clients  
*Text editors, Calypso, Renaku, etc.*

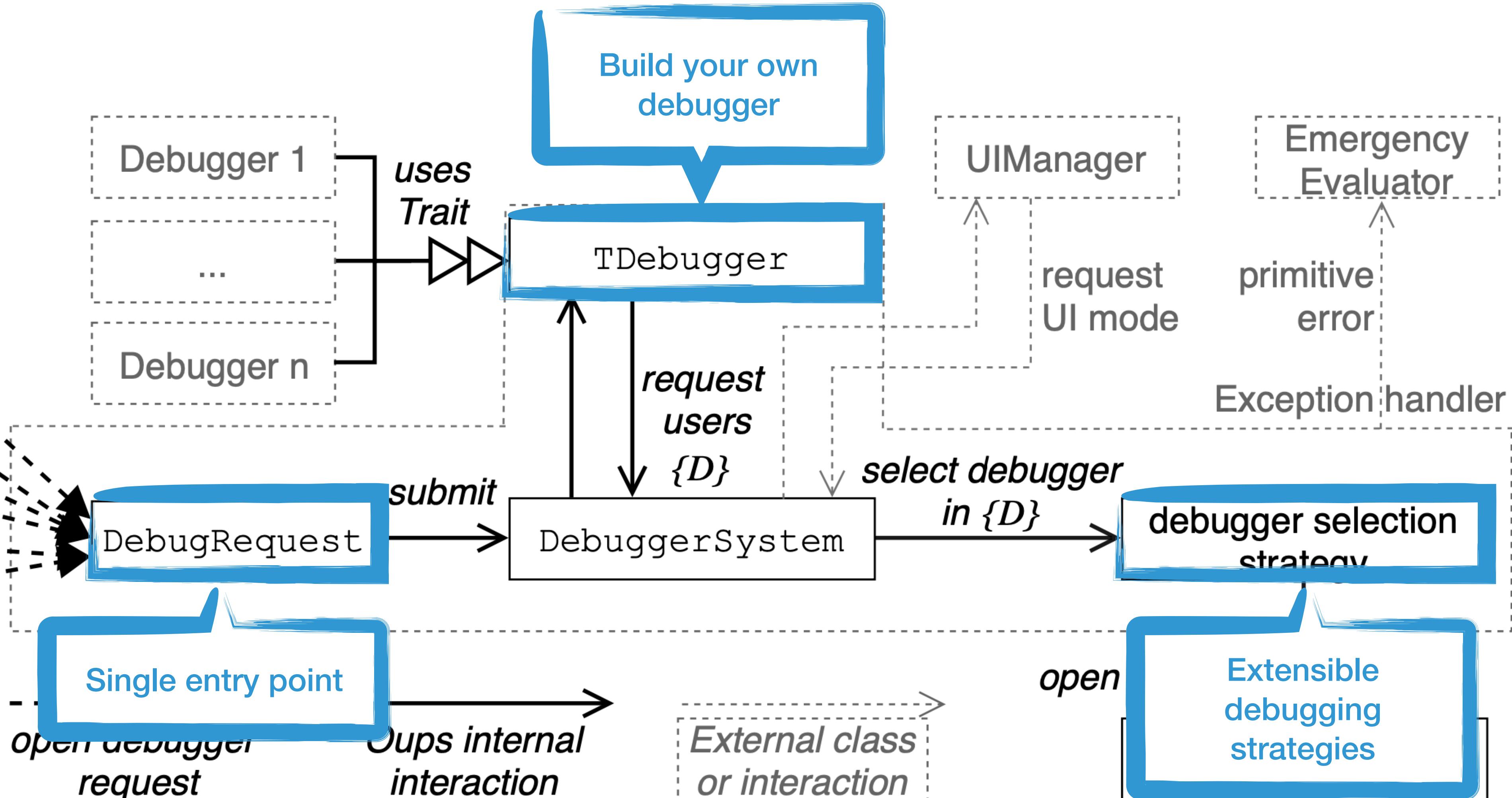




# Infrastructure

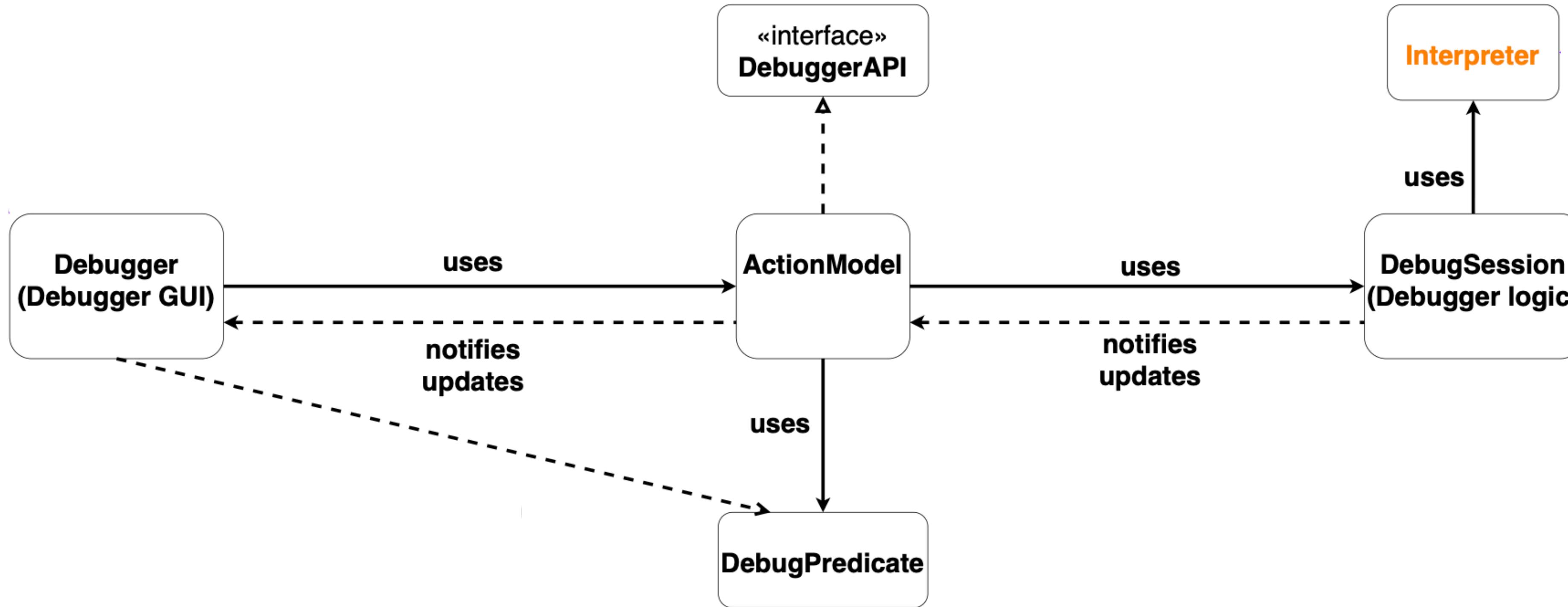
## Now

Warning (exception)  
Process Context  
Exceptions  
User Interrupts  
ad-hoc clients  
*Text editors, Calypso, Renaku, etc.*



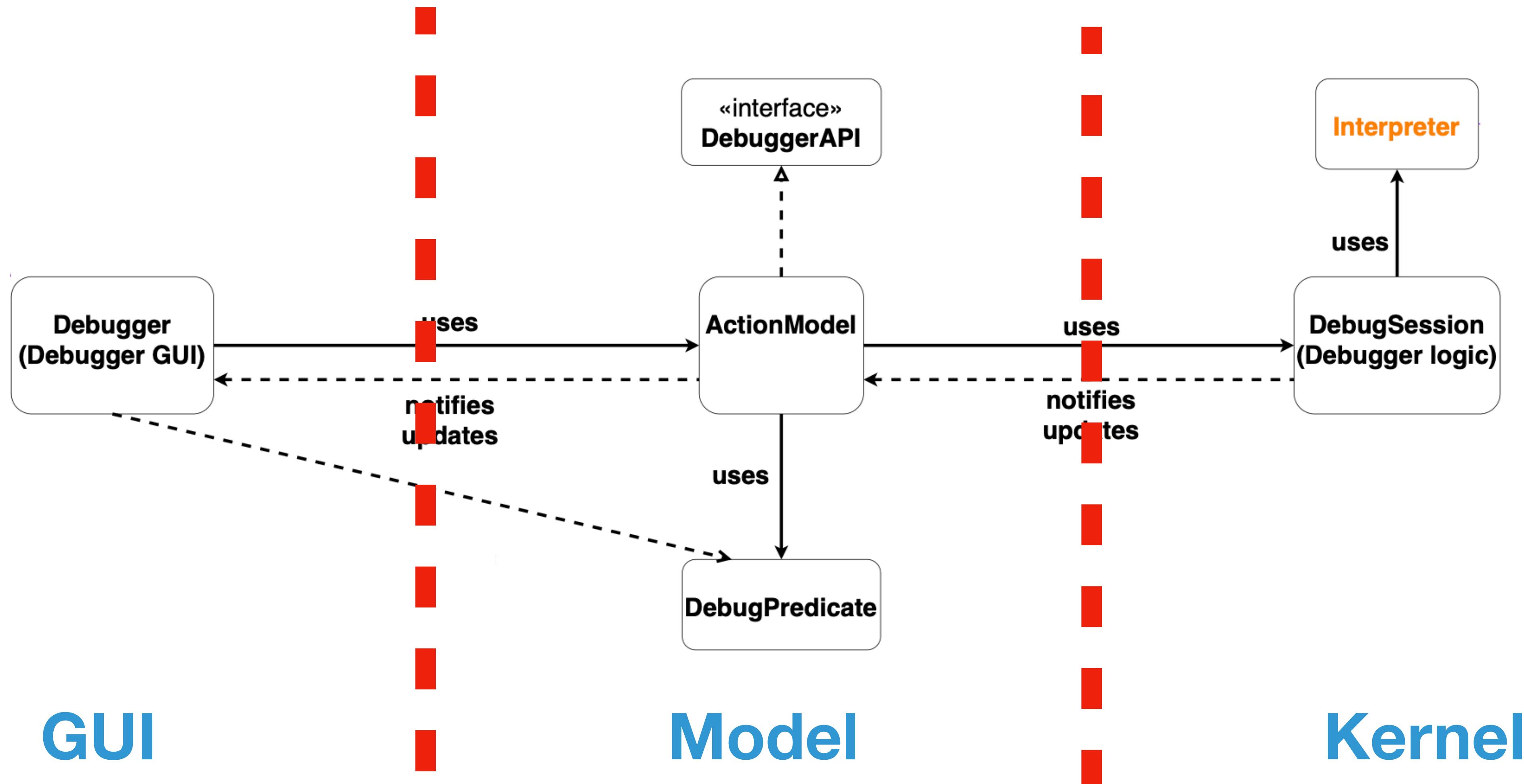


# Architectural improvements





# Architectural improvements



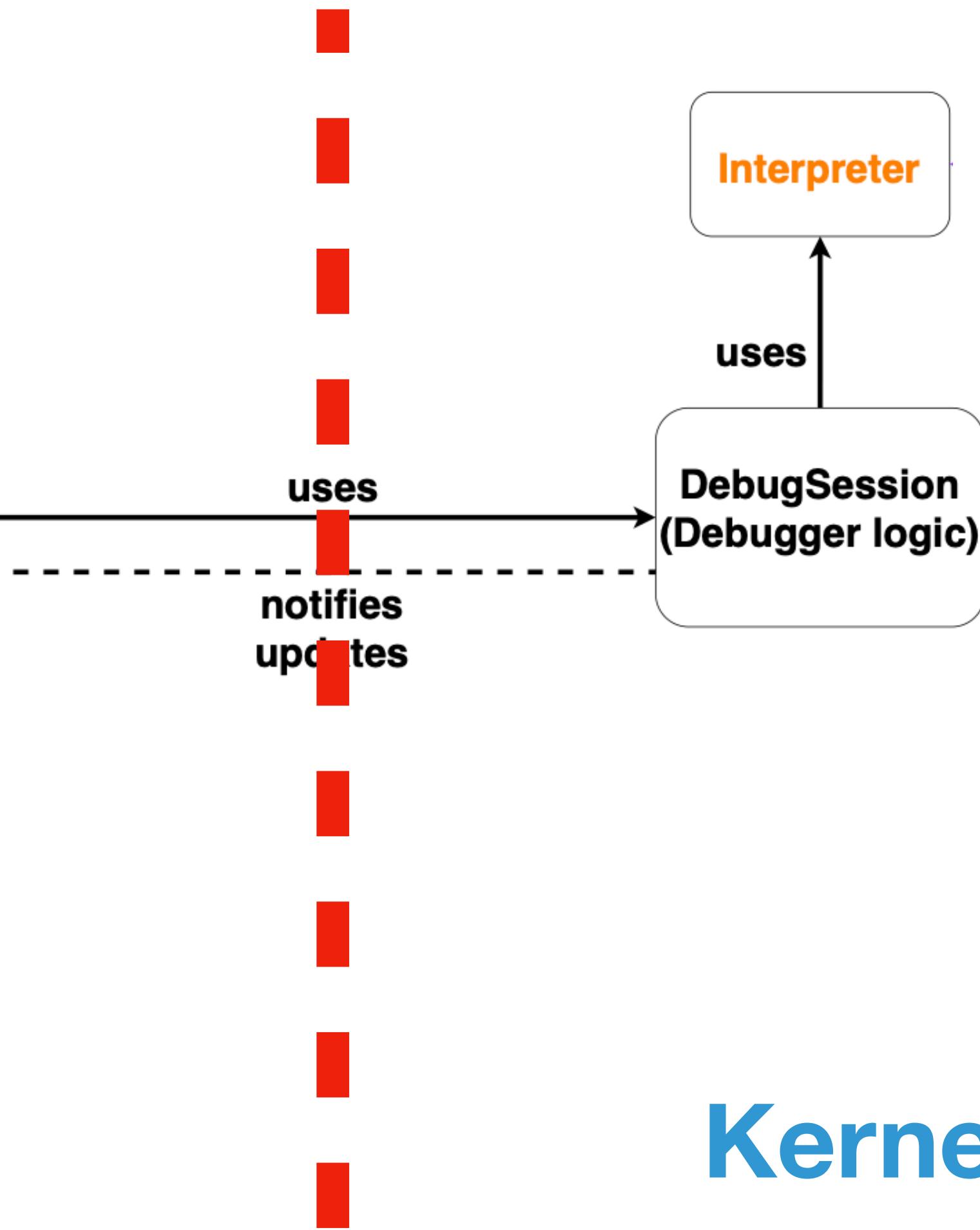
# Improvements

SindarinDebuggerTest (Object) >> halt [Kernel]

SindarinDebuggerTest (Object) >> **halt** [Kernel]

SindarinDebuggerTest >> **testChangingPcToNonExistingBytecodeOffsetGoesToPreviousPcWithExistingBytecodeOffset** [Sindarin-Tests]

```
1 testChangingPcToNonExistingBytecodeOffsetGoesToPreviousPcWithExistingBytecodeOffset
2
3     | scdbg newNode |
4     scdbg := SindarinDebugger debug: [
5         self methodWithDoubleAssignment ].
6
7     scdbg step.
8     "pc of b := 1 from `a:= b:= 1` This is associated to the pc of a storeIntoTemp
9     bytecode, of length 2 bytes. So we add 1 to get a pc that is in the middle of the
10    bytecode"
11    newNode := scdbg methodNode statements first value.
12    newPc := (scdbg methodNode firstPcForNode: newNode) + 1.
13
14    self assert: (scdbg methodNode sourceNodeForPC: newPc) identicalTo: newNode.
15    self halt.
16    scdbg pc: newPc.
17
18    self assert: scdbg node equals: newNode.
19    self assert: scdbg pc equals: newPc - 1.
```

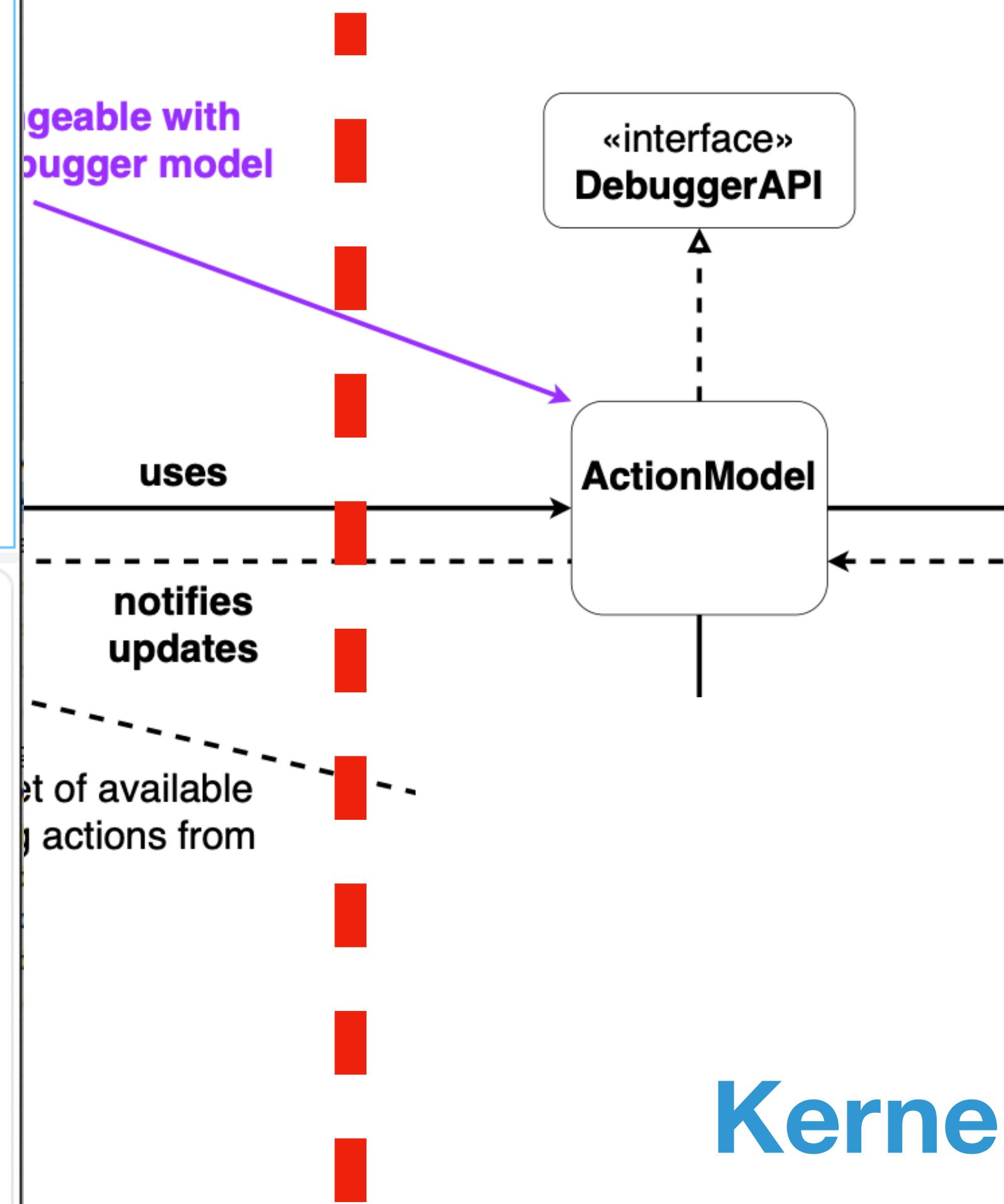


Kernel

# Improvements

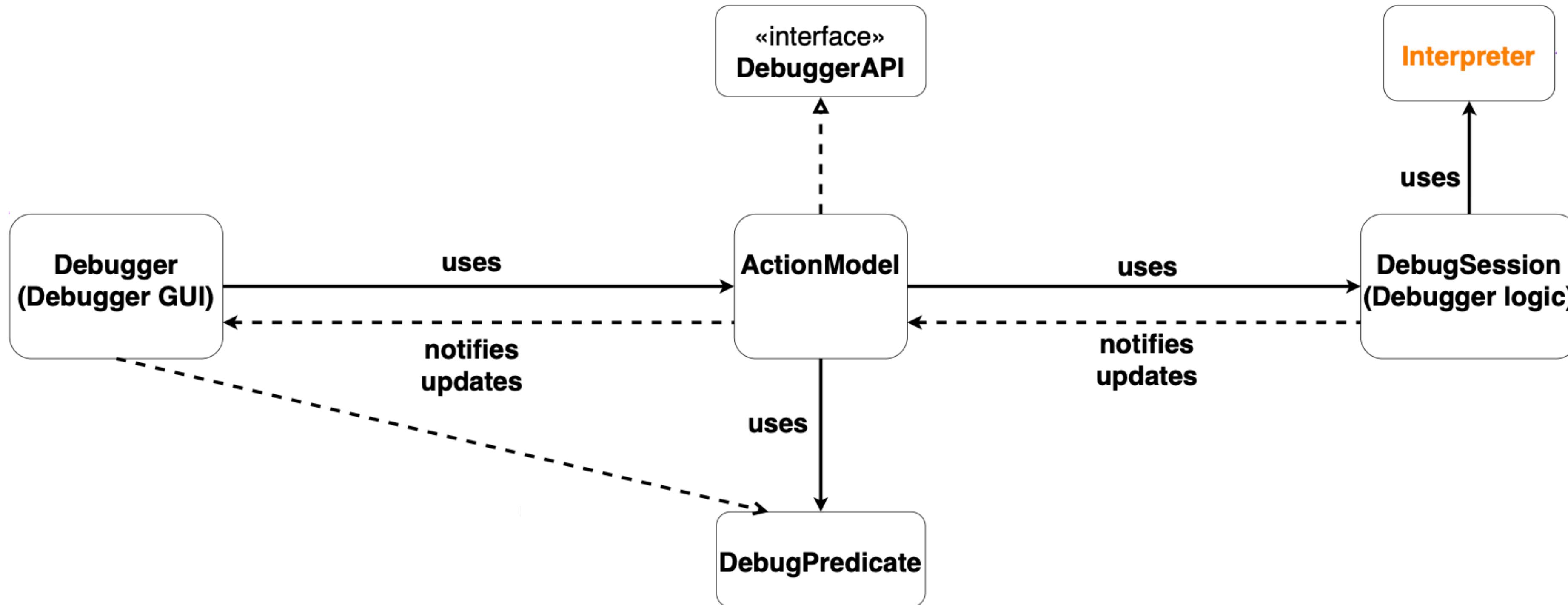
Kernel

```
SindarinDebuggerTest (Object) >> halt [Kernel]  
  
SindarinDebuggerTest (Object) >> halt [Kernel]  
  
SindarinDebuggerTest >> testChangingPcToNonExistingBytecodeOffsetGoesToPreviousPcWithExistingBytecodeOffset [Sindarin-Tests]  
  
SindarinDebuggerTest (TestCase) >> performTest [SUnit-Core]  
SindarinDebuggerTest (TestCase) >> runCase [SUnit-Core]  
FullBlockClosure (BlockClosure) >> ensure: [Kernel]  
SindarinDebuggerTest (TestCase) >> runCase [SUnit-Core]  
FullBlockClosure (BlockClosure) >> ensure: [Kernel]  
SindarinDebuggerTest (TestCase) >> runCase [SUnit-Core]  
SindarinDebuggerTest >> runCaseManaged [Sindarin-Tests]  
TestResult >> runCaseForDebug: [SUnit-Core]  
FullBlockClosure (BlockClosure) >> on:do: [Kernel]  
TestResult >> runCaseForDebug: [SUnit-Core]  
  
1 testChangingPcToNonExistingBytecodeOffsetGoesToPreviousPcWithExistingBytecodeOffset  
2  
3     | scdbg newNode |  
4     scdbg := SindarinDebugger debug: [  
5         self methodWithDoubleAssignment ].  
6  
7     scdbg step.  
8     "pc of b := 1 from `a:= b:= 1` This is associated to the pc of a storeIntoTemp  
bytecode, of length 2 bytes. So we add 1 to get a pc that is in the middle of the  
bytecode"  
9     newNode := scdbg methodNode statements first value.  
10    newPc := (scdbg methodNode firstPcForNode: newNode) + 1.  
11  
12    self assert: (scdbg methodNode sourceNodeForPC: newPc) identicalTo: newNode.  
13    self halt.  
14    scdbg pc: newPc.  
15  
16    self assert: scdbg node equals: newNode.  
17    self assert: scdbg pc equals: newPc - 1.
```



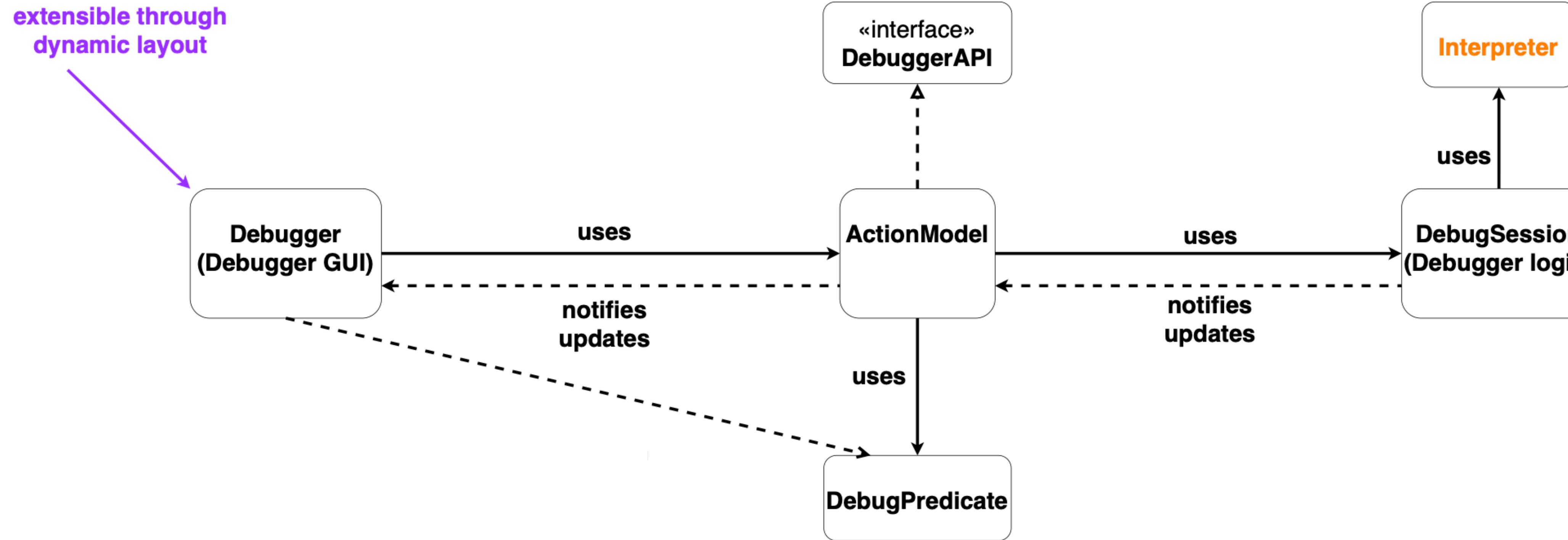


# Architectural improvements



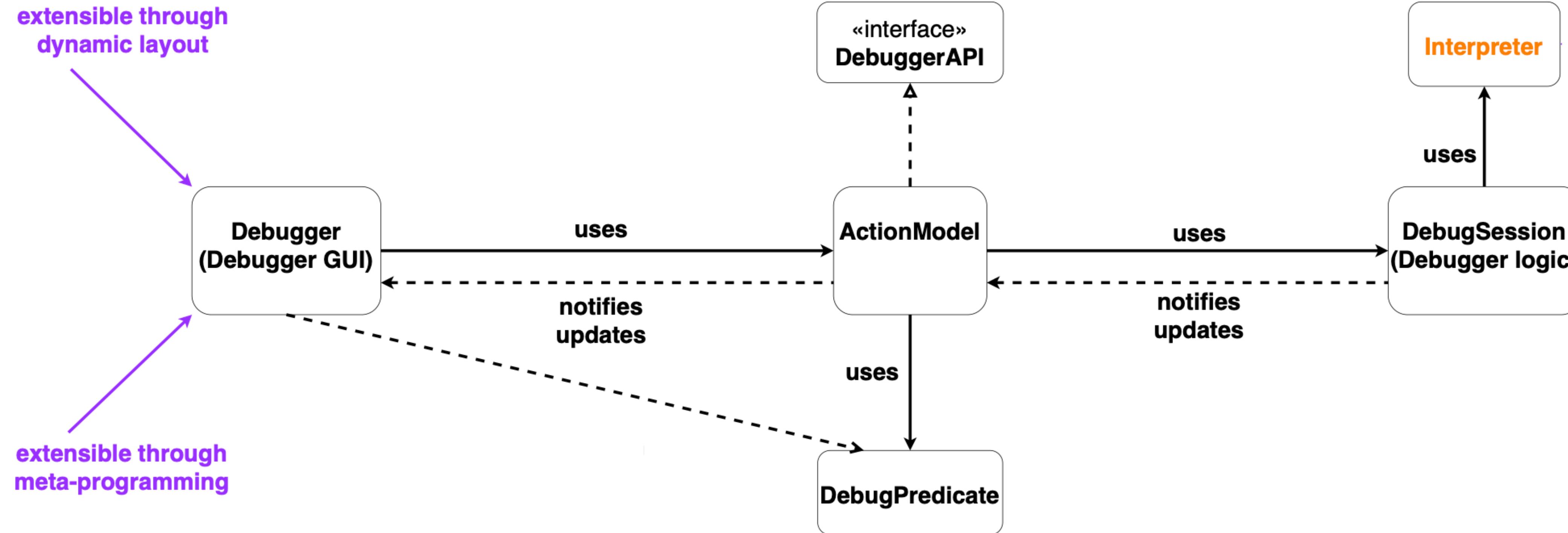


# Architectural improvements



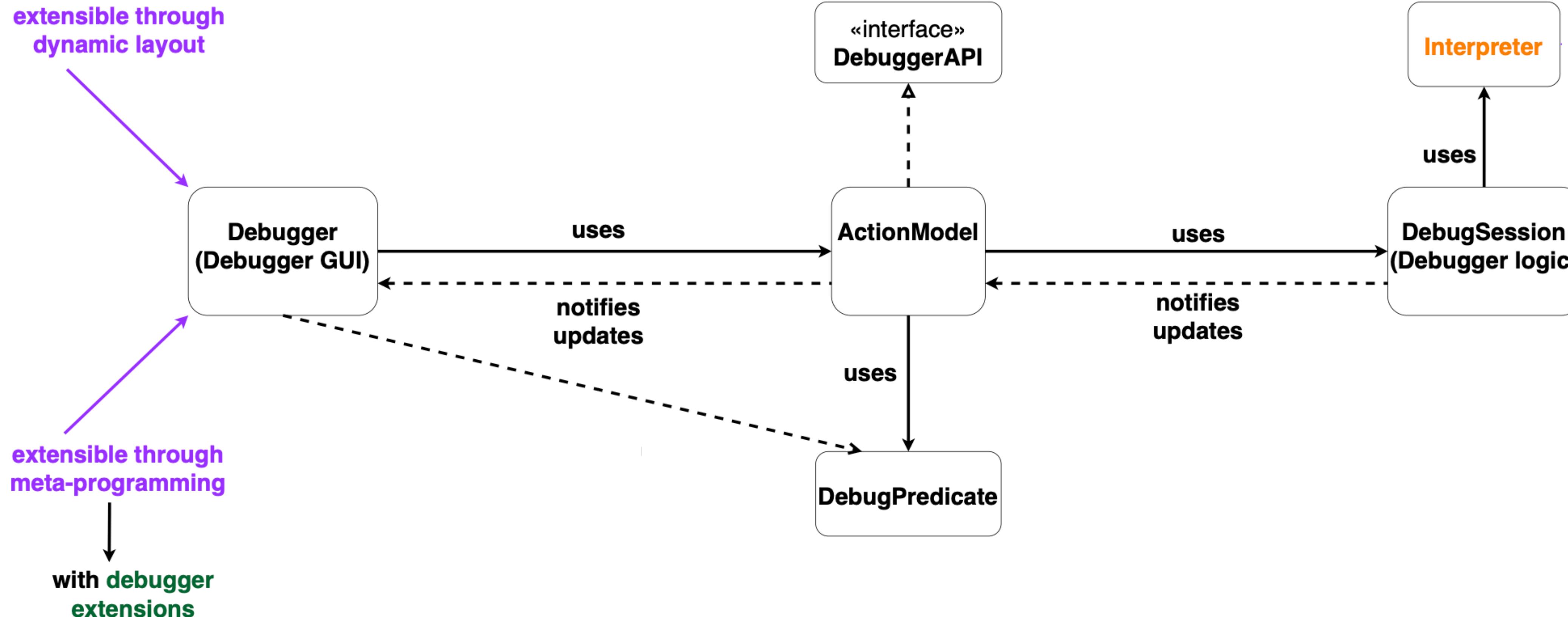


# Architectural improvements





# Architectural improvements





# Architectural improvements

#asFloat was sent to nil

Stack

Class	Method	Package
BATextParagraphSpan	[ :anAttribute   anAttribute	Bloc-Alexandrie
Array (SequenceableCollection)	do: [ :index   aBlock value: (self at: index)]	Collections-Abstract
BATextParagraphSpan	[ aeCanvas pathTranslate: self	Bloc-Alexandrie
AeCairoContext	restoreStateAfter: [ ]	Alexandrie-Cairo

Bytecode

```
43 <40> pushTemp: 0
44 <4C> self
45 <42> pushTemp: 2
46 <70> send: at:
47 <7A> send: value:
48 <D8> pop
49 <42> pushTemp: 2
50 <51> pushConstant: 1
51 <60> send: +
52 <D2> popIntoTemp: 2
53 <E1 FF ED ED> jumpTo: 38
57 <58> returnSelf
```

Variable

Variable	Value
Σ stackTop	2
⌚ aBlock	[ :anAttribu
Σ index	2

extensible through dynamic layout

Debug (Debug)

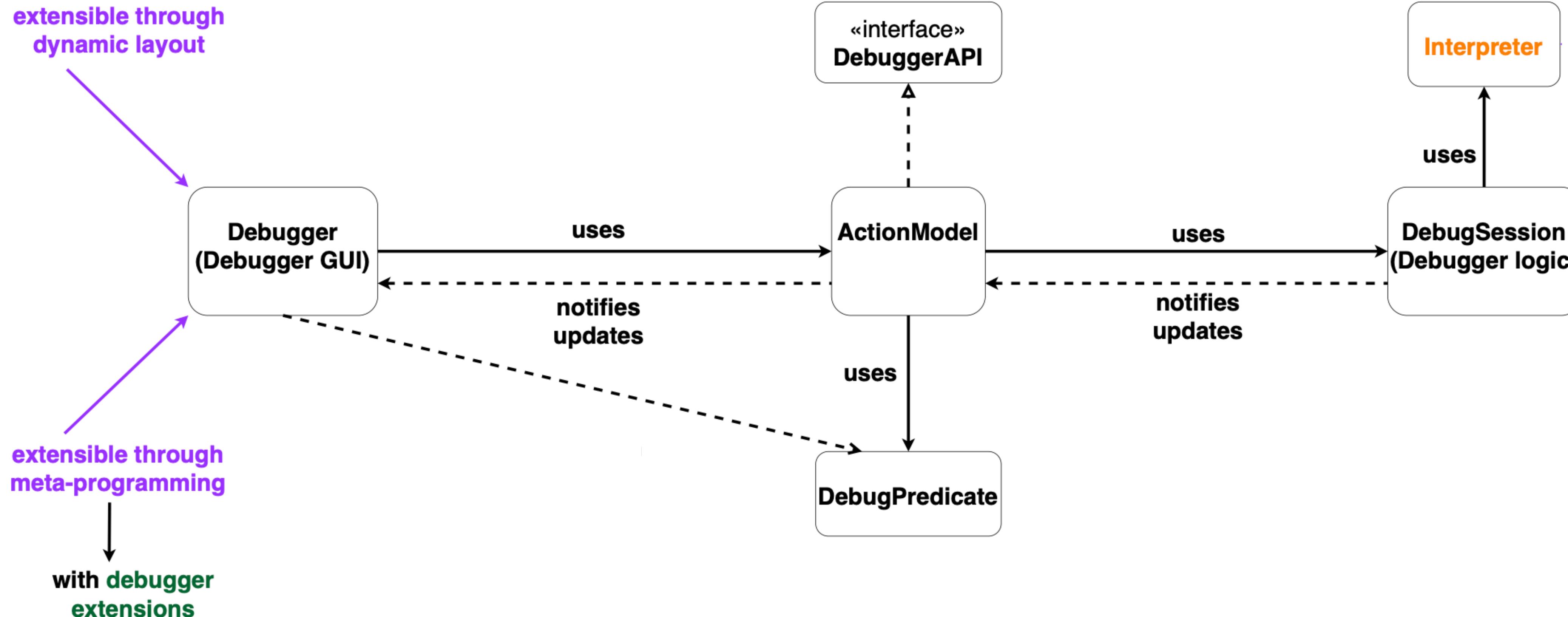
extensible through meta-programming

with debugger extensions

```
1 do: aBlock
2 "Refer to the comment in Collection|do:."
3 1 to: self size do:
4   [:index | aBlock value: (self at: index)]
```

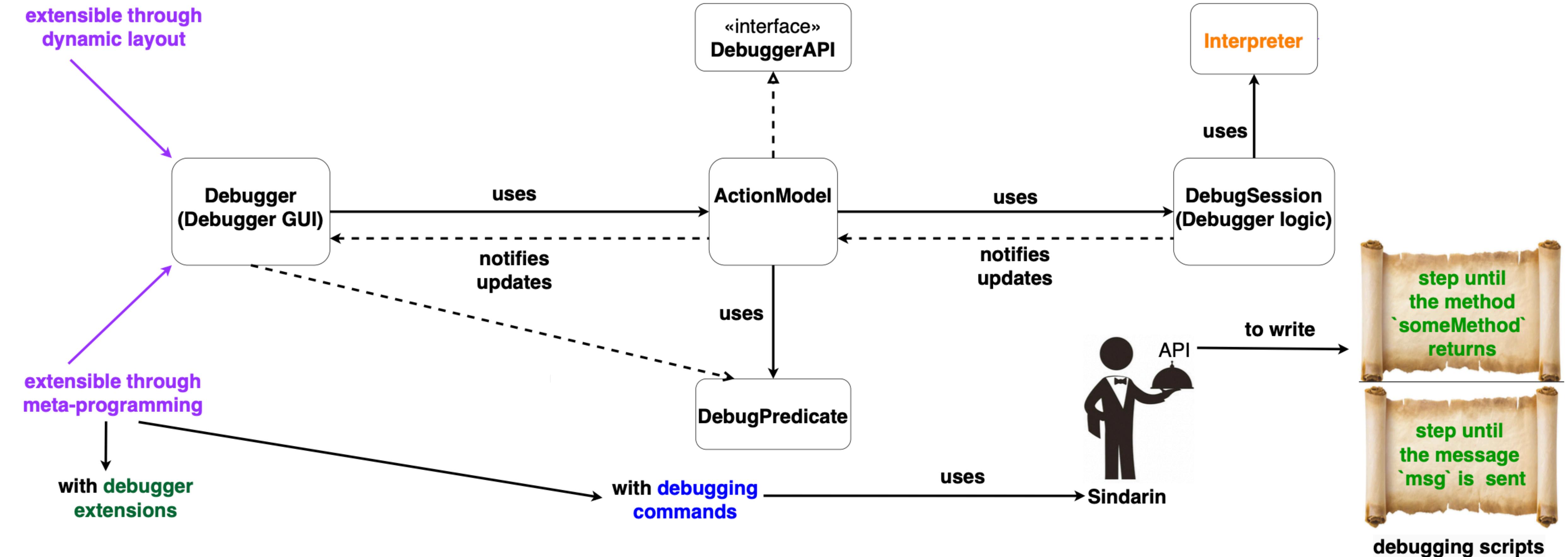


# Architectural improvements





# Architectural improvements





# Architectural improvements

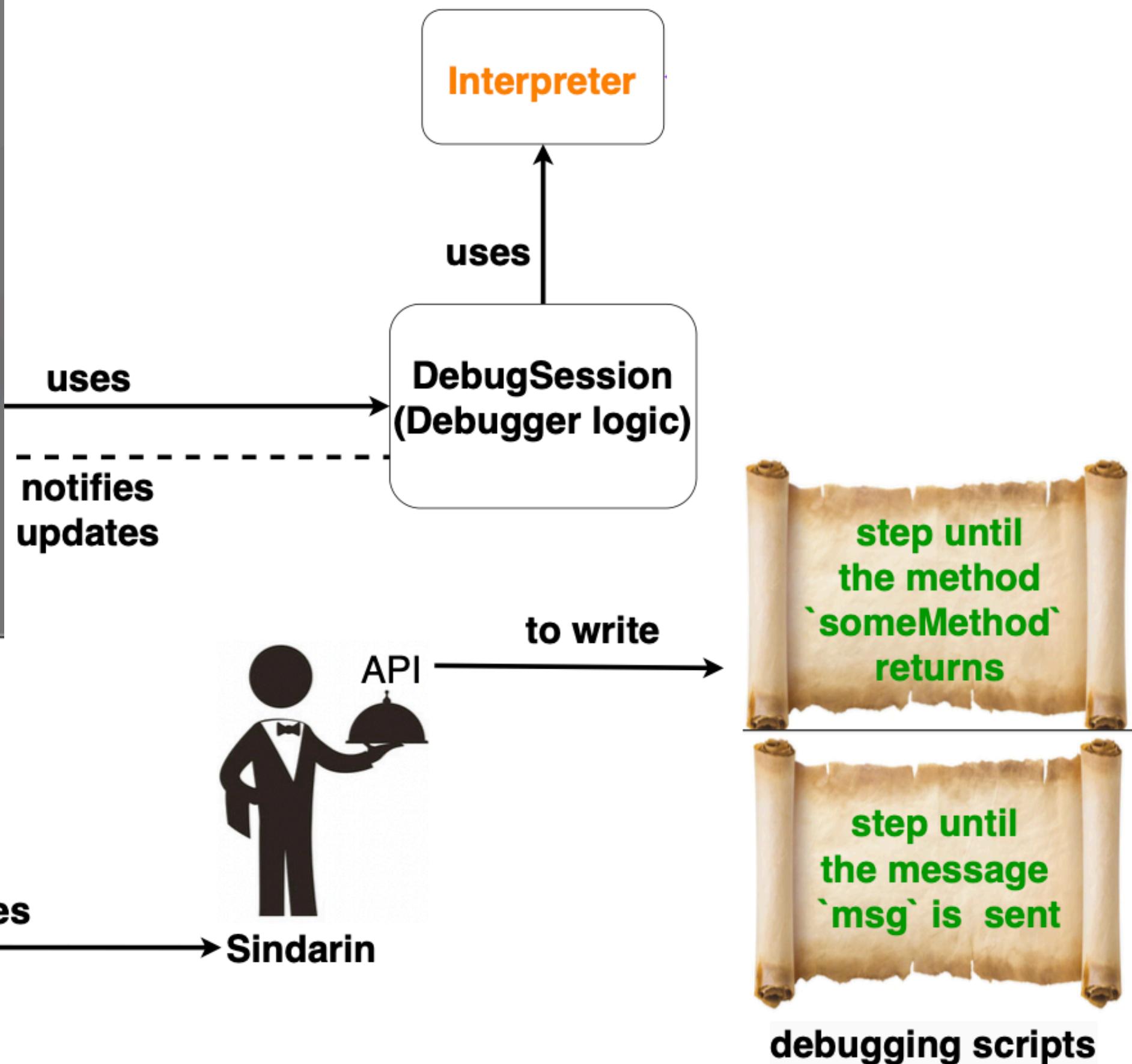
The screenshot shows a debugger interface with a sidebar listing classes, methods, and packages. The main area displays a step history and a code editor with the following code:

```
1 do: aBloc
2 "Refer" To return Next call in class Next call in receiver [Exp.] Skip up to (stop before) Skip
3 1 to: self size do:
4   [:index | aBlock value: (self at: index)]
```

The 'Advanced Step' button is highlighted with a red box. A red line connects the 'Advanced Step' button to the code editor, indicating extensibility through meta-programming.

extensible through  
meta-programming

with debugger  
extensions





# Custom debugging commands

[Jump to caret](#)

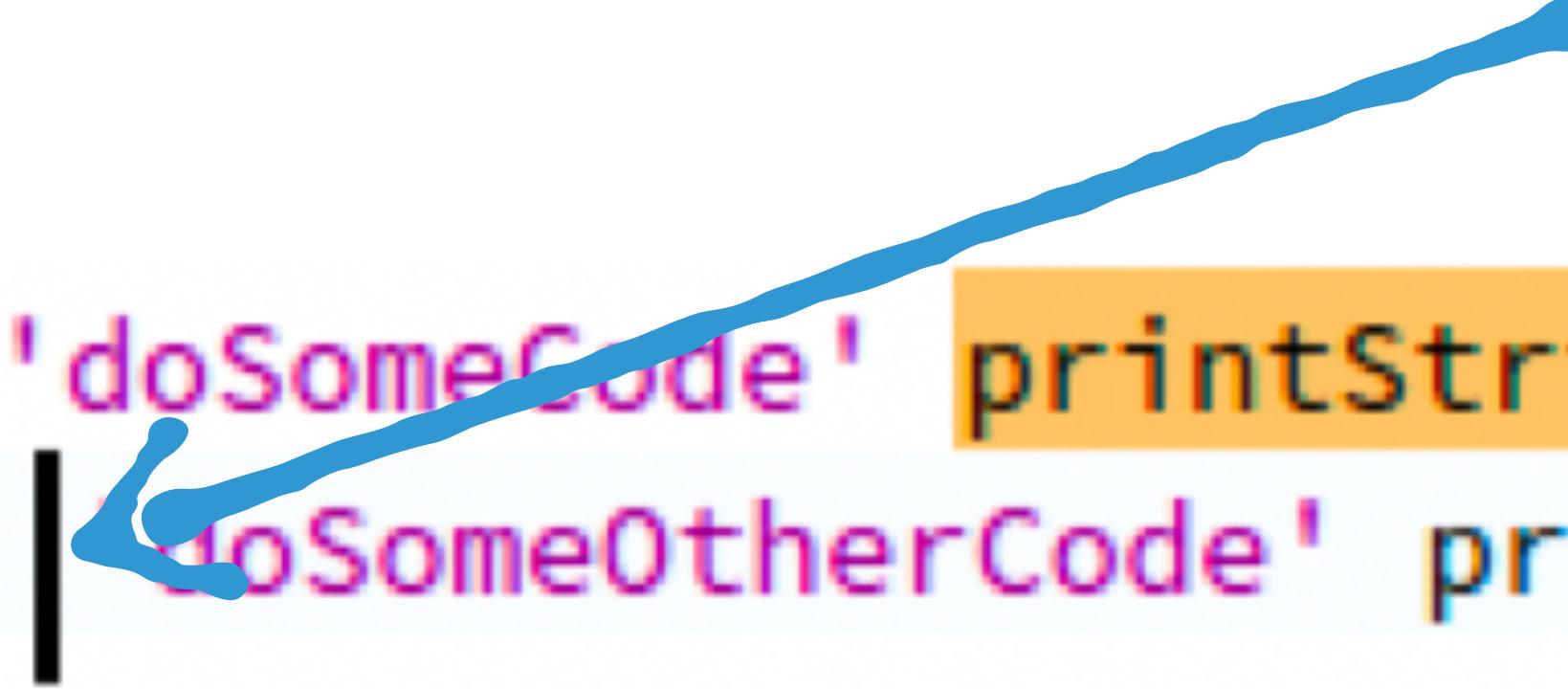
```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```



# Custom debugging commands

Jump to caret

```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```



A blue arrow originates from the word "printString" in the first line of code and points to the position of the opening bracket "[" in the second line of code, indicating where the debugger should jump.



# Custom debugging commands

Jump to caret

place caret

```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```



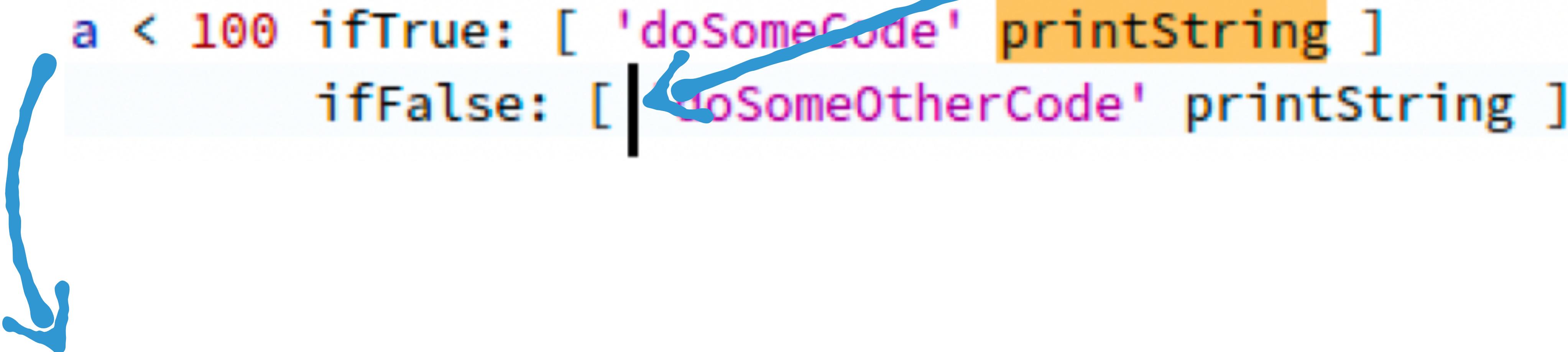


# Custom debugging commands

Jump to caret

place caret

```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```



A hand-drawn blue arrow originates from the bottom left, points upwards towards the word 'printString' in the first code block, and then continues straight to point at the opening bracket of the second code block. A second, shorter blue arrow points directly at the opening bracket of the second code block.



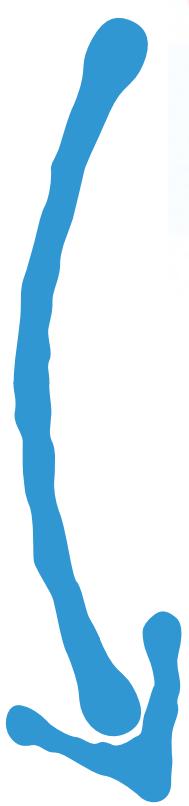
# Custom debugging commands

Jump to caret

jump

```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```

place caret





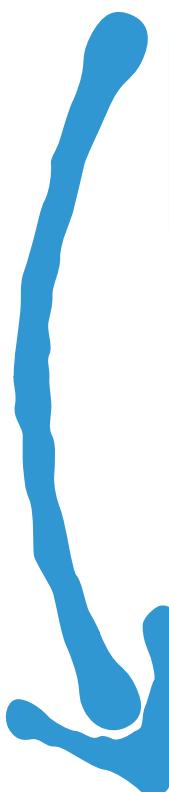
# Custom debugging commands

Jump to caret

jump

```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```

place caret



```
a < 100 ifTrue: [ 'doSomeCode' printString ]  
ifFalse: [ 'doSomeOtherCode' printString ]
```



# Roadmap

Until ESUG 2024

- More infrastructure improvements
- Emergency debugger
- Meta-Object Protocol
- UX redesign (with Thales)
- Remote debugger
- Documentation



**DEMOS**