Object Flow Analysis – Taking an Object-centric View on Dynamic Analysis

Adrian Lienhard[1], Stéphane Ducasse[2] and Tudor Gîrba[1]

[1] *Software Composition Group, University of Bern, Switzerland*
[2] *LISTIC, University of Savoie, France*

A missing aspect of OO dynamic analysis

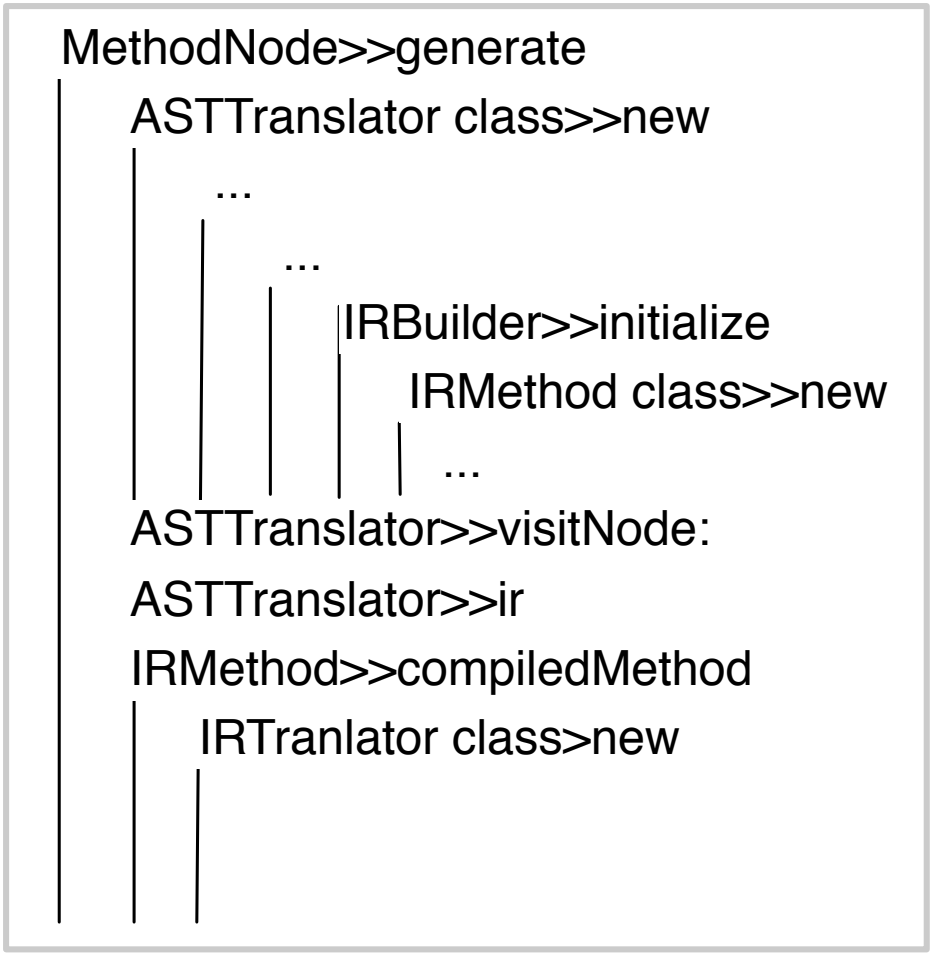
Typical characterization of OO program execution:

1) message passing

2) object interrelationships

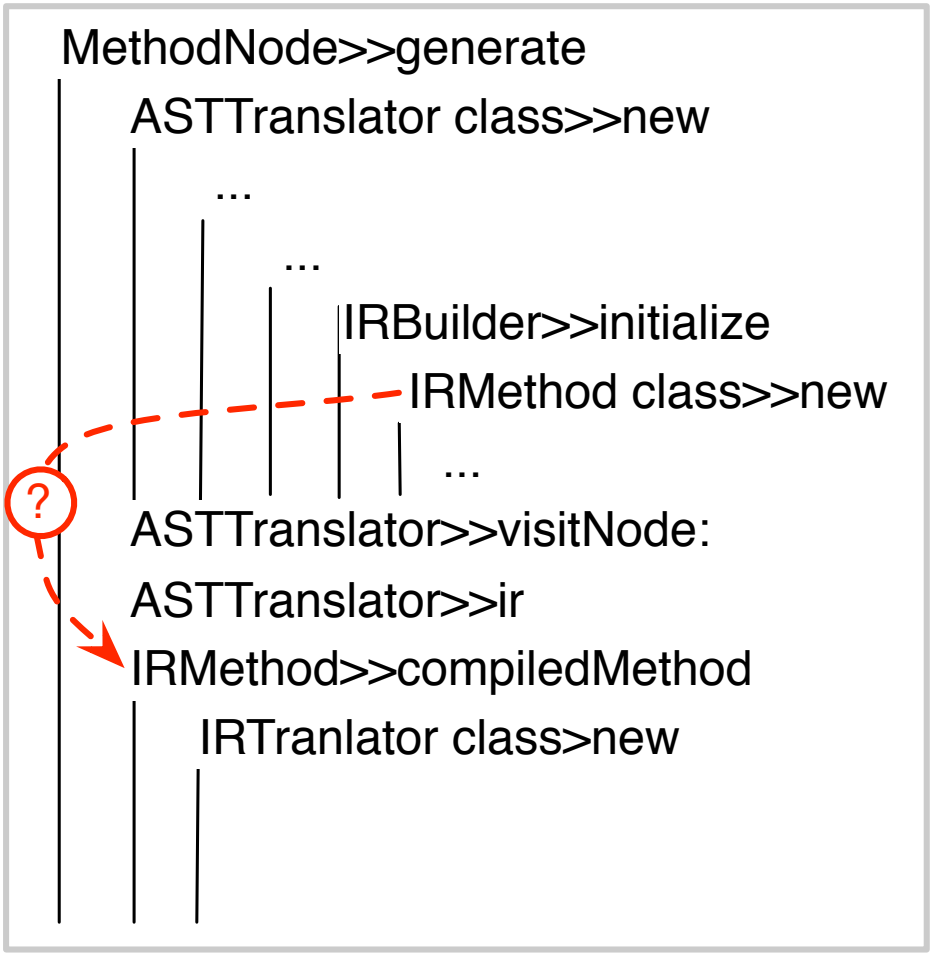
How are objects passed throught the system?

1) the method execution perspective

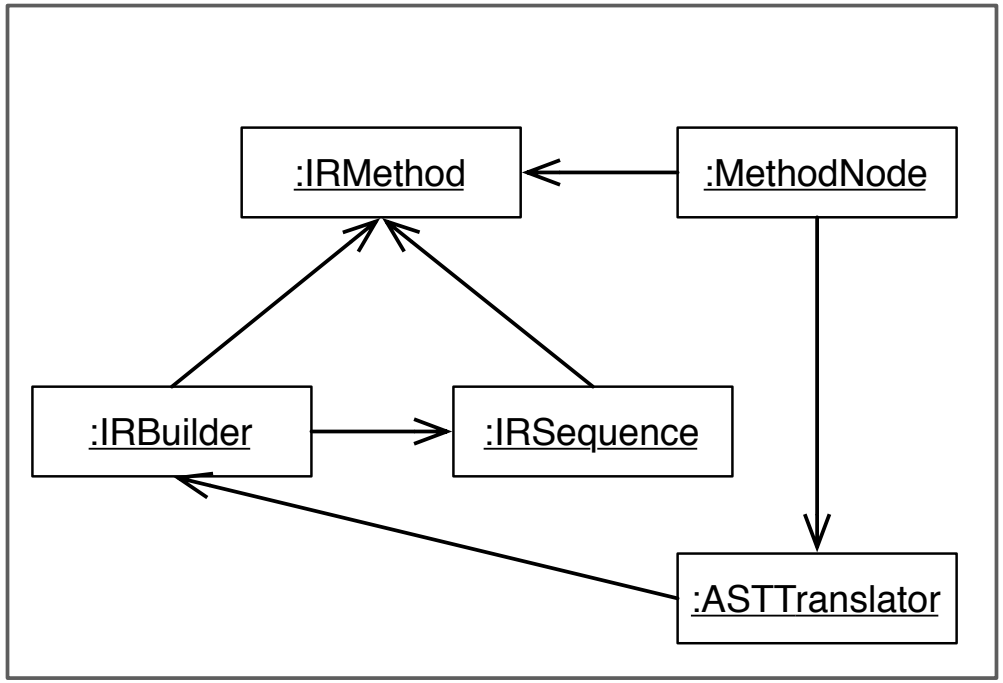


1) the method execution perspective

"How is the IRMethod instance passed to MethodNode?"

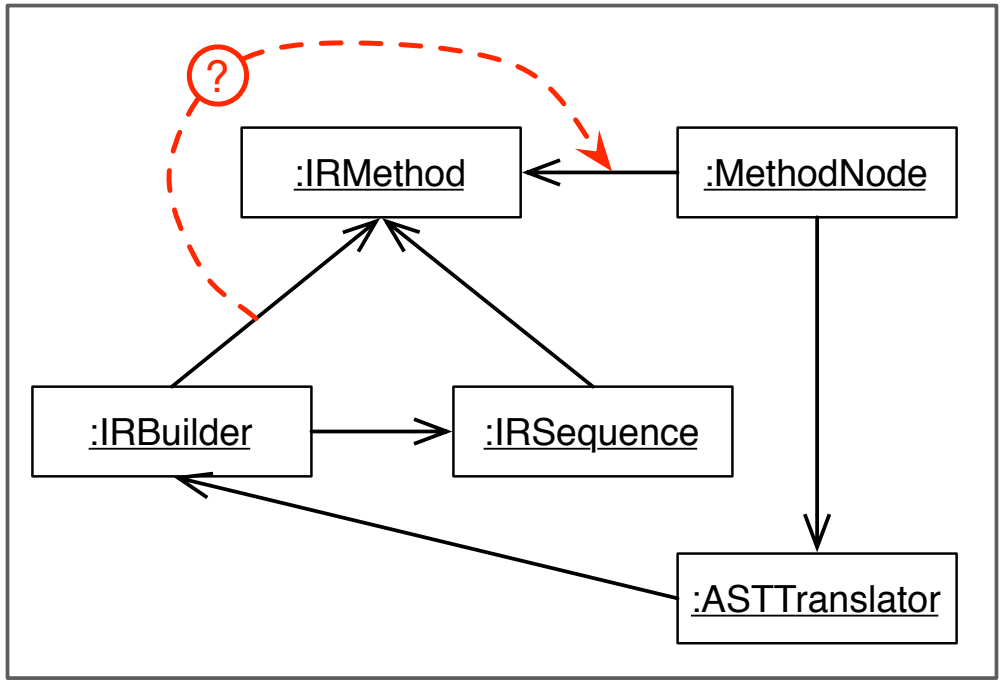


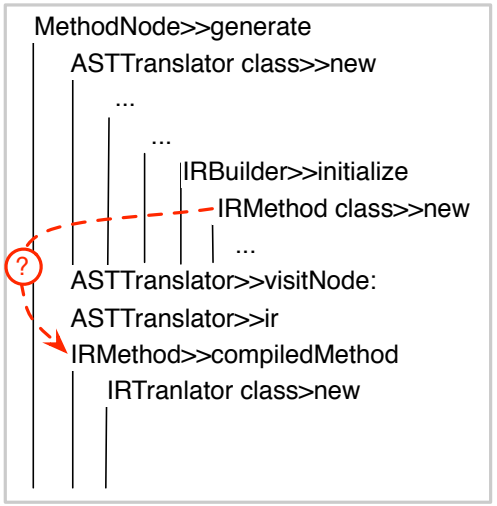
2) the object interrelationship perspective



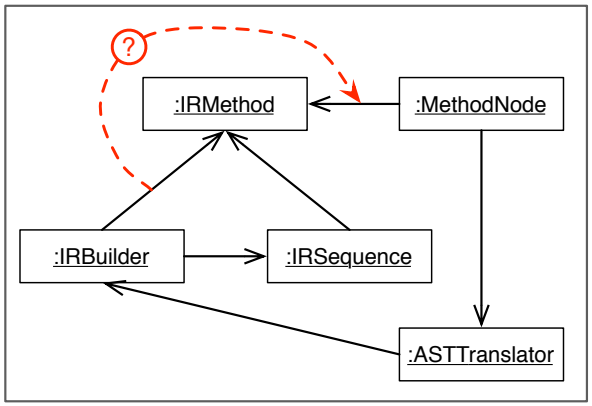
2) the object interrelationship perspective

"How is the IRMethod instance passed to MethodNode?"





interdependency?



Object Flow Analysis

track the transfer of object references

Object Flow Analysis

a) capture all references of an object

b) track the transfer of references

```
IRBuilder>>initialize
  ir := IRMethod new.
  ...

IRBuilder>>startNewSequence
    newSequence := IRSequence new.
    newSequence  method: ir.

IRSequence>>method: aMethod
  method := aMethod.

ASTTranslator>>ir
  ^ builder ir.

MethodNode>>generate
    ast := ASTTranslator new visitNode: self.
  ir := ast ir.
  ^ ir compiledMethod.

IRMethod>>compiledMethod
    ^ compiledMethod := IRTranslator new
        interpret: self;
        compiledMethod.

IRTranslator>>interpret: ir
    ...
```

instantiation

field store

field read

argument
field store

return

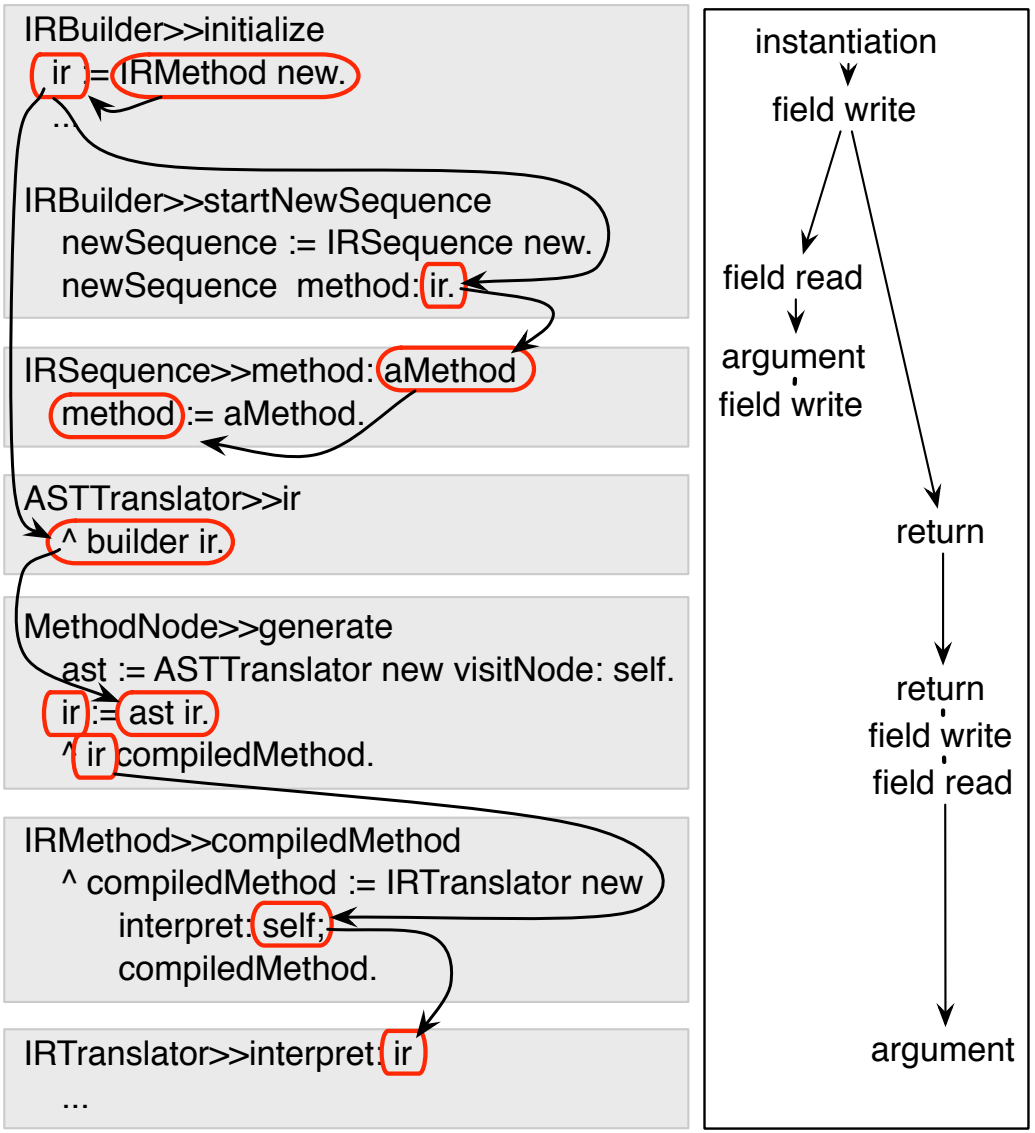
return
field store
field read

argument

Object Flow Analysis

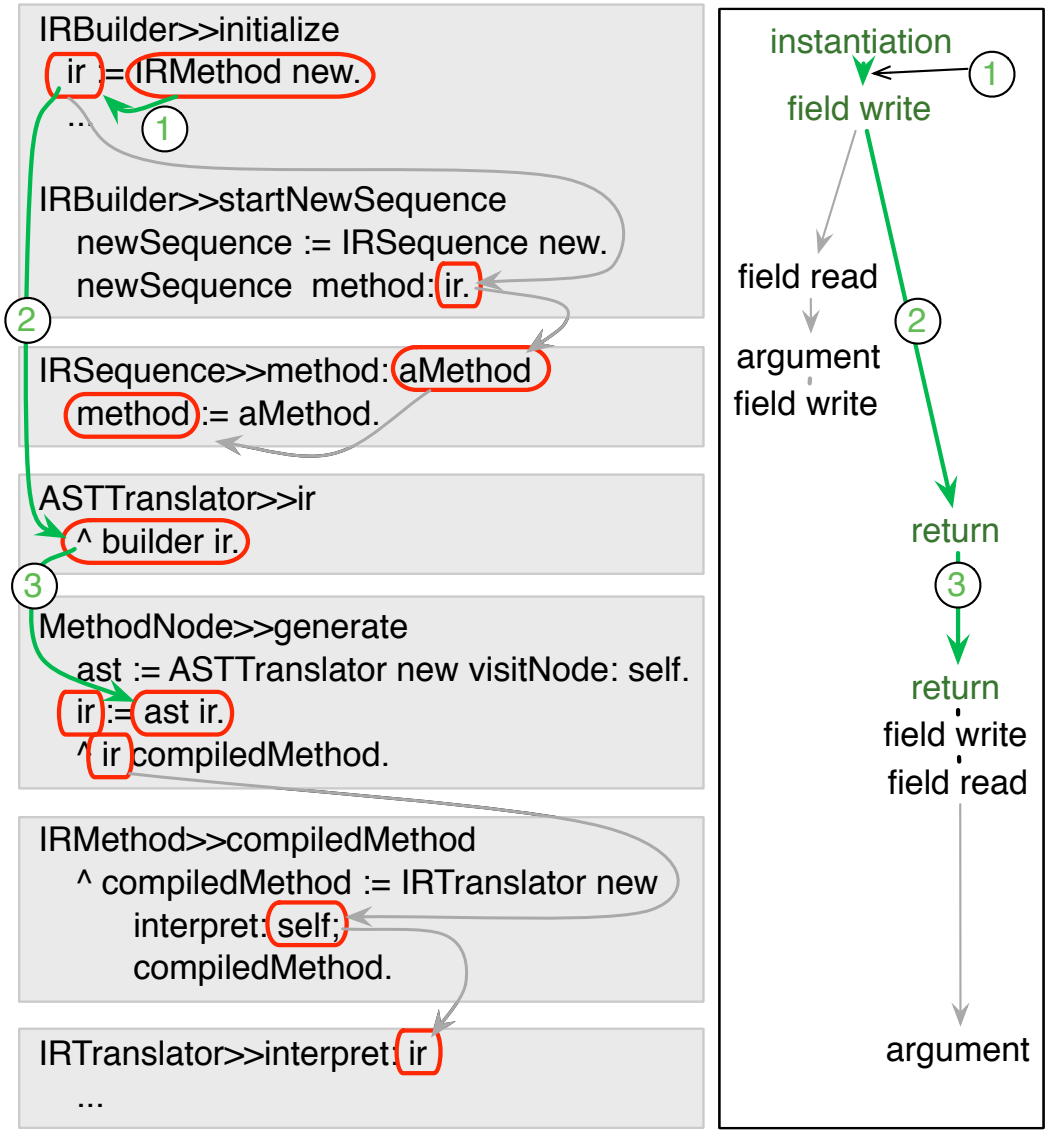
a) capture all references of an object

b) track the transfer of references



Object Flow Analysis

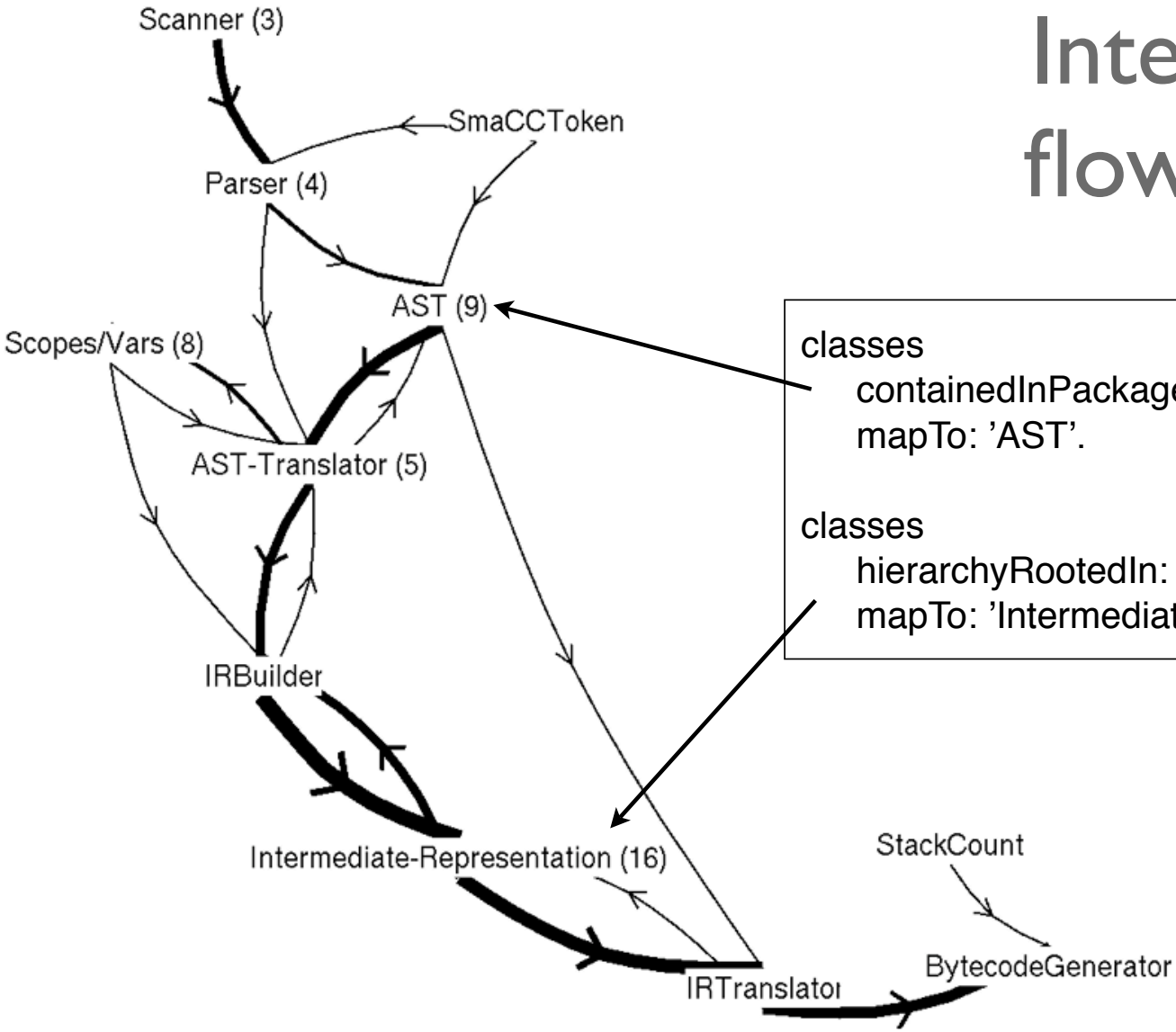
"How is the IRMethod instance passed to MethodNode?"



Application

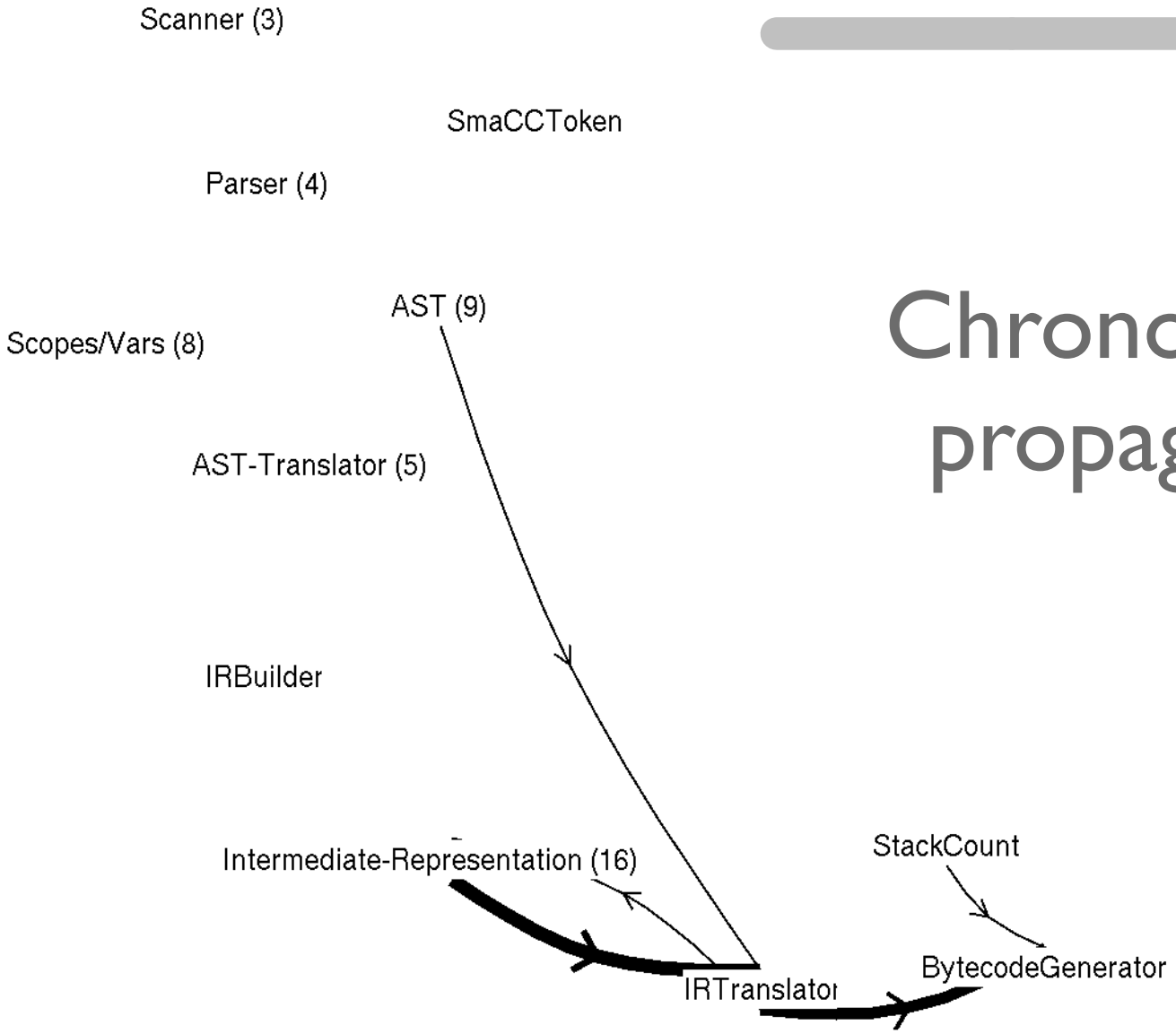
How do classes exchange objects?

Inter-unit flow view

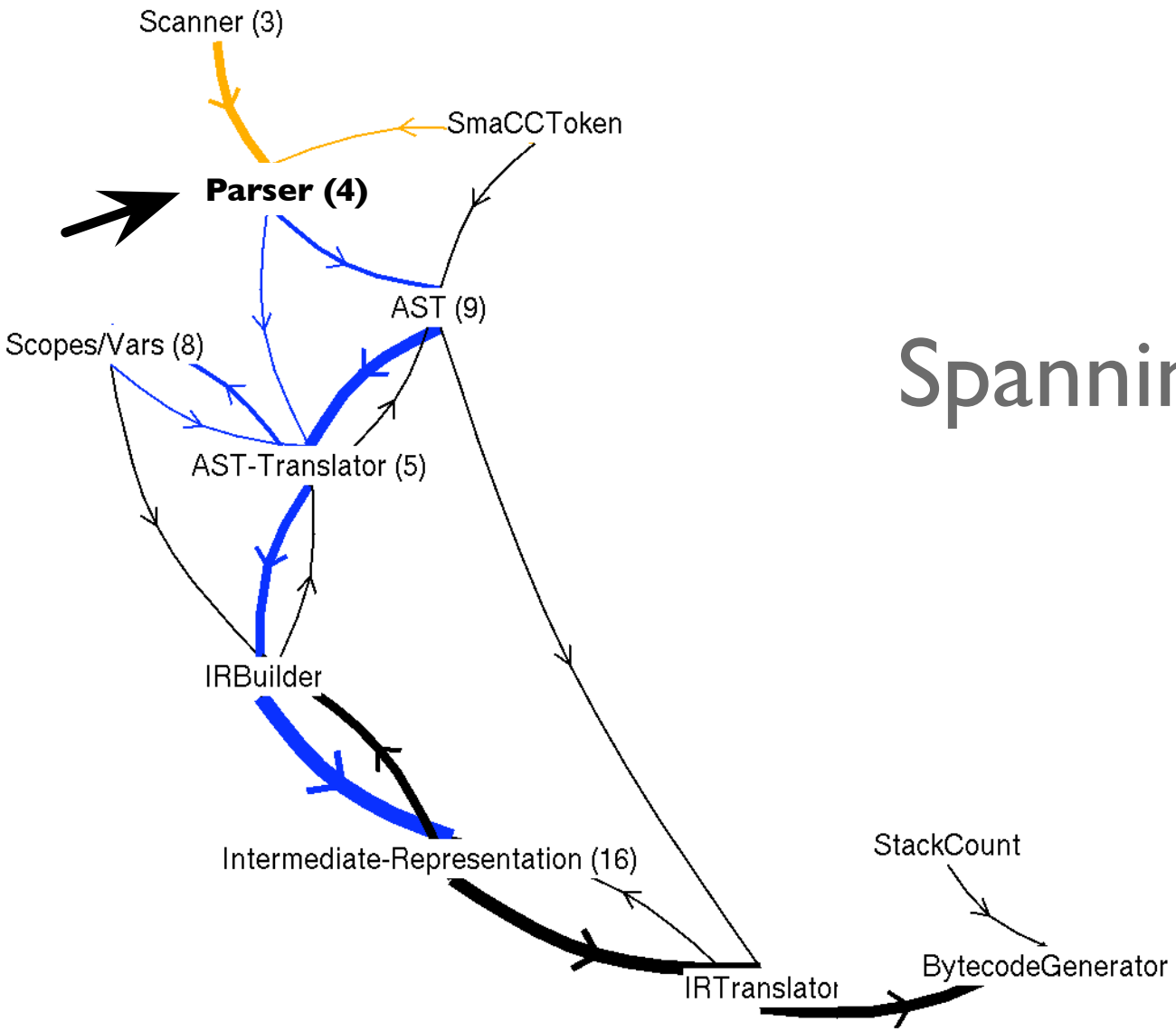


classes
containedInPackage: 'AST-Nodes'
mapTo: 'AST'.

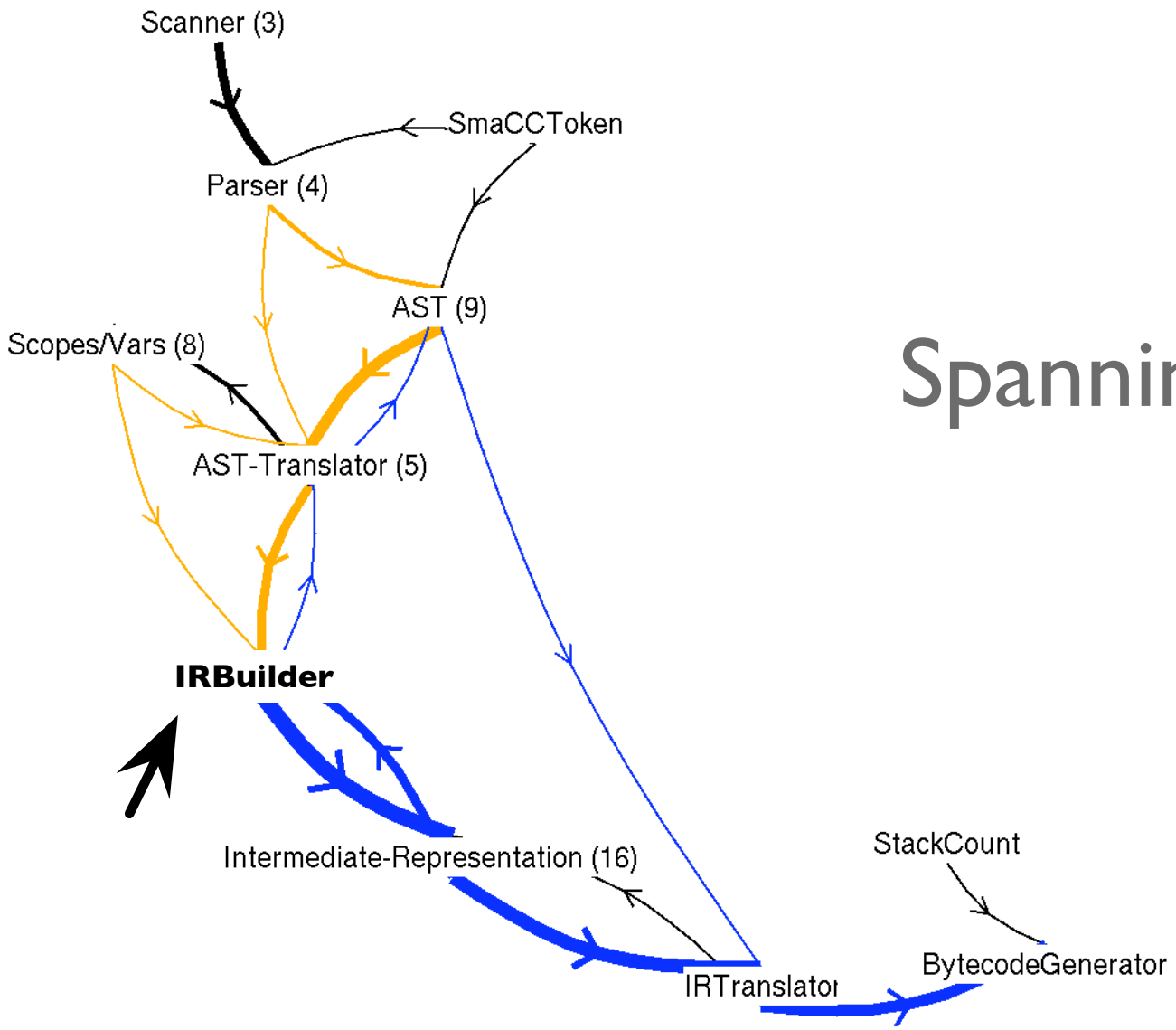
classes
hierarchyRootedIn: 'IRInstruction'
mapTo: 'Intermediate-Representation'



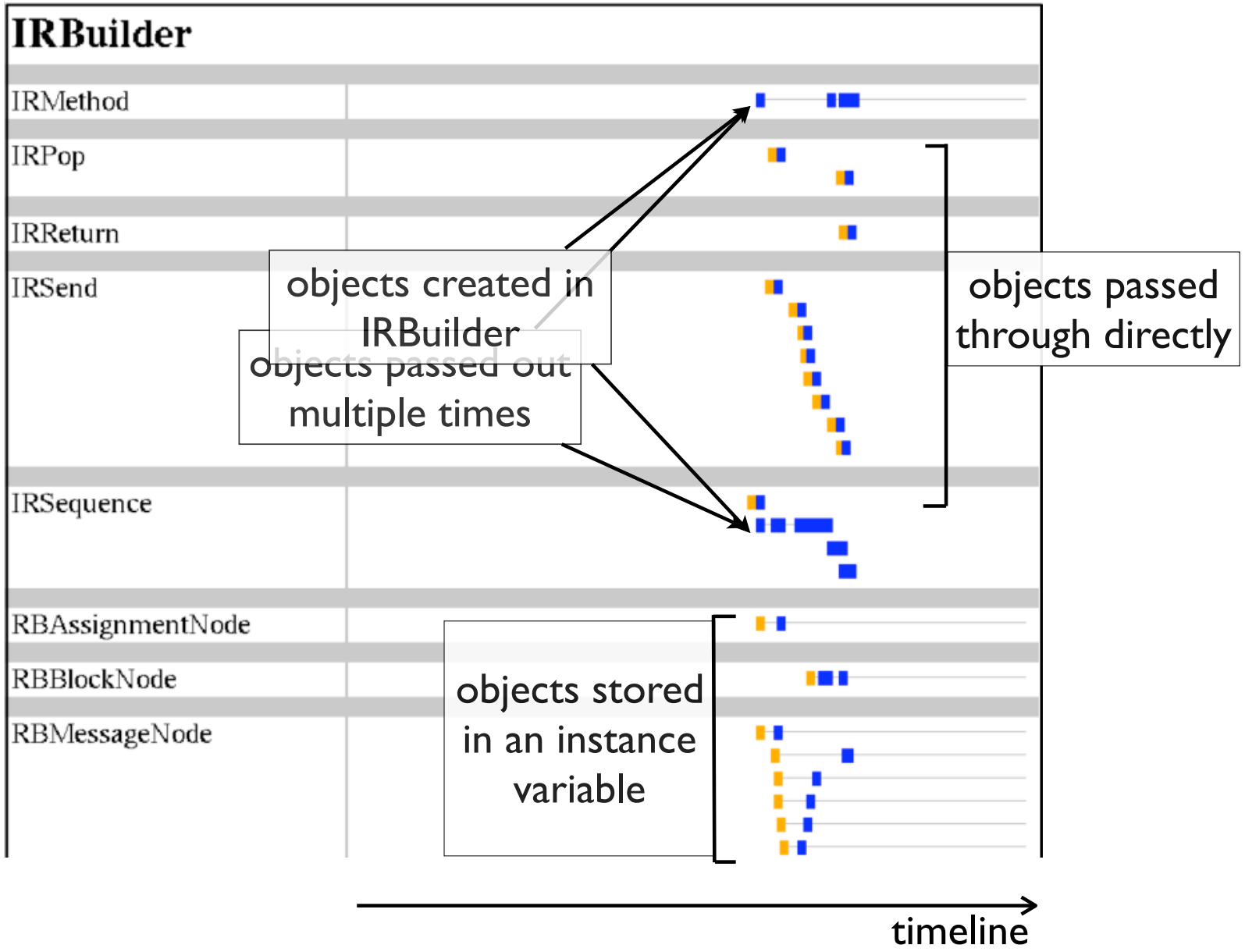
Chronological propagation



Spanning flows

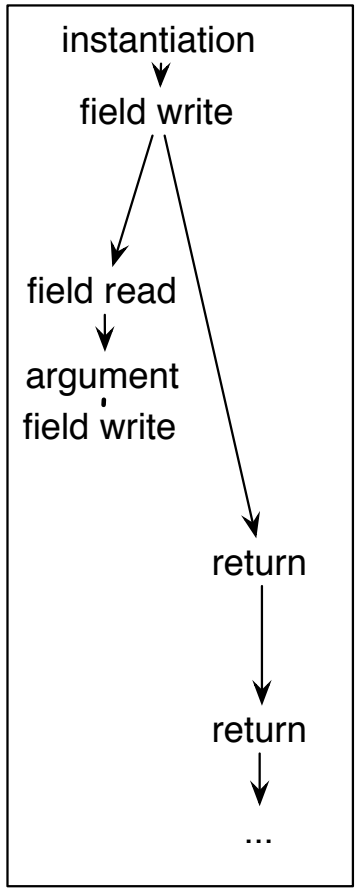


Spanning flows


IRBuilder
IRMethod
IRPop
IRReturn
IRSend
IRSequence
RBAssignmentNode
RBBlockNode
RBMessageNode
objects created in IRBuilder
objects passed out multiple times
objects passed through directly
objects stored in an instance variable
timeline

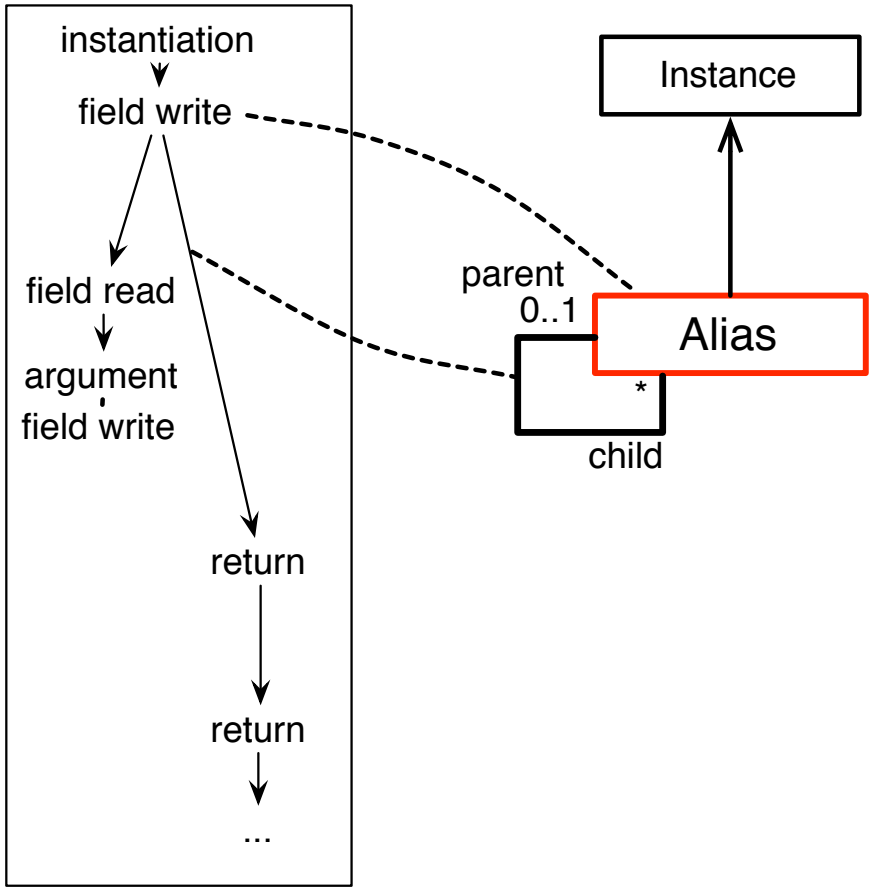
Representing object flows

```
IRBuilder>>initialize
  ir := IRMethod new.
  ..

IRBuilder>>startNewSequence
    newSequence := IRSequence new.
    newSequence  method: ir.

IRSequence>>method: aMethod
   method := aMethod.

ASTTranslator>>ir
   ^ builder ir.

MethodNode>>generate
    ast := ASTTranslator new visitNode: self.
   ir := ast ir.
    ^ ir compiledMethod.
```

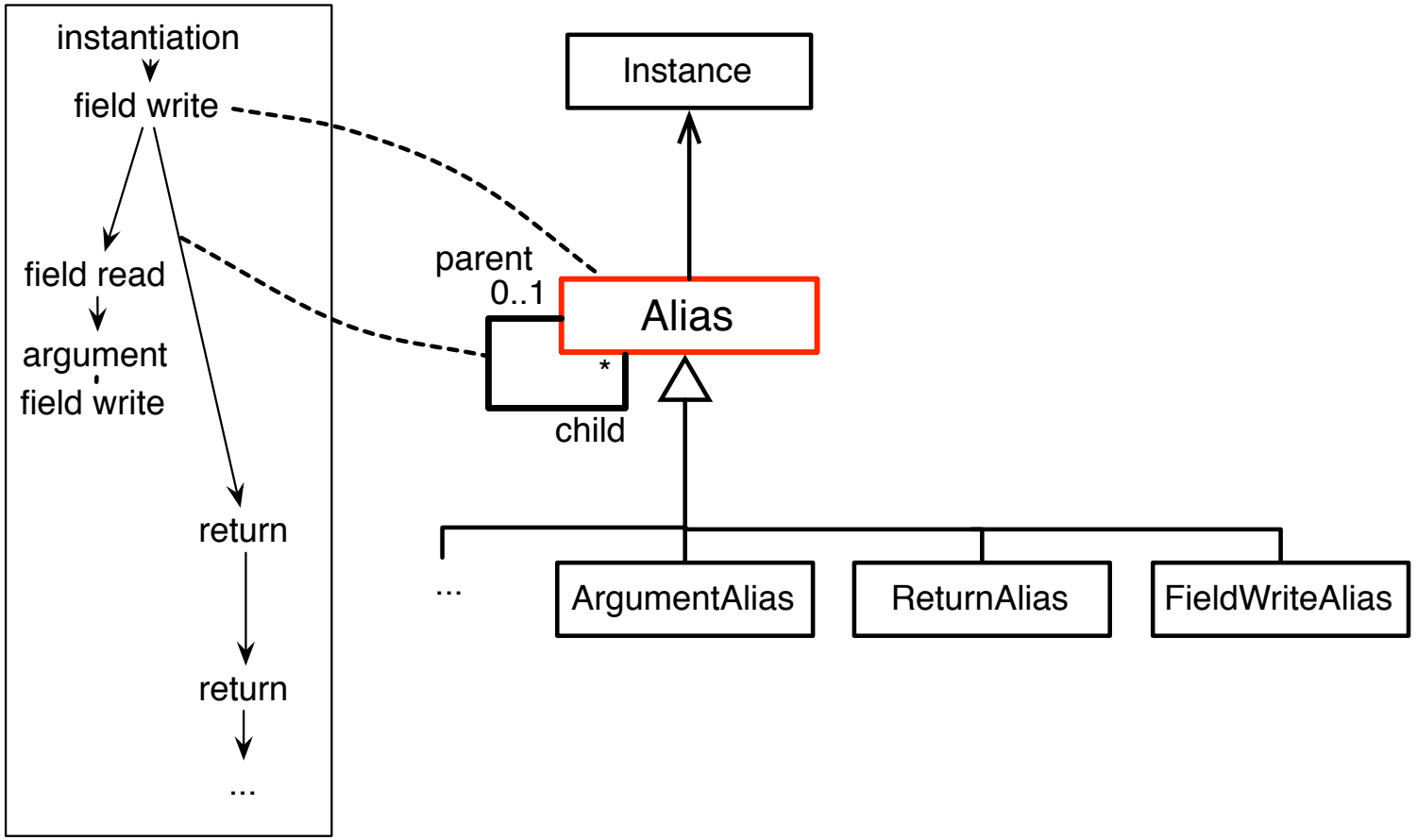


Representing object flows

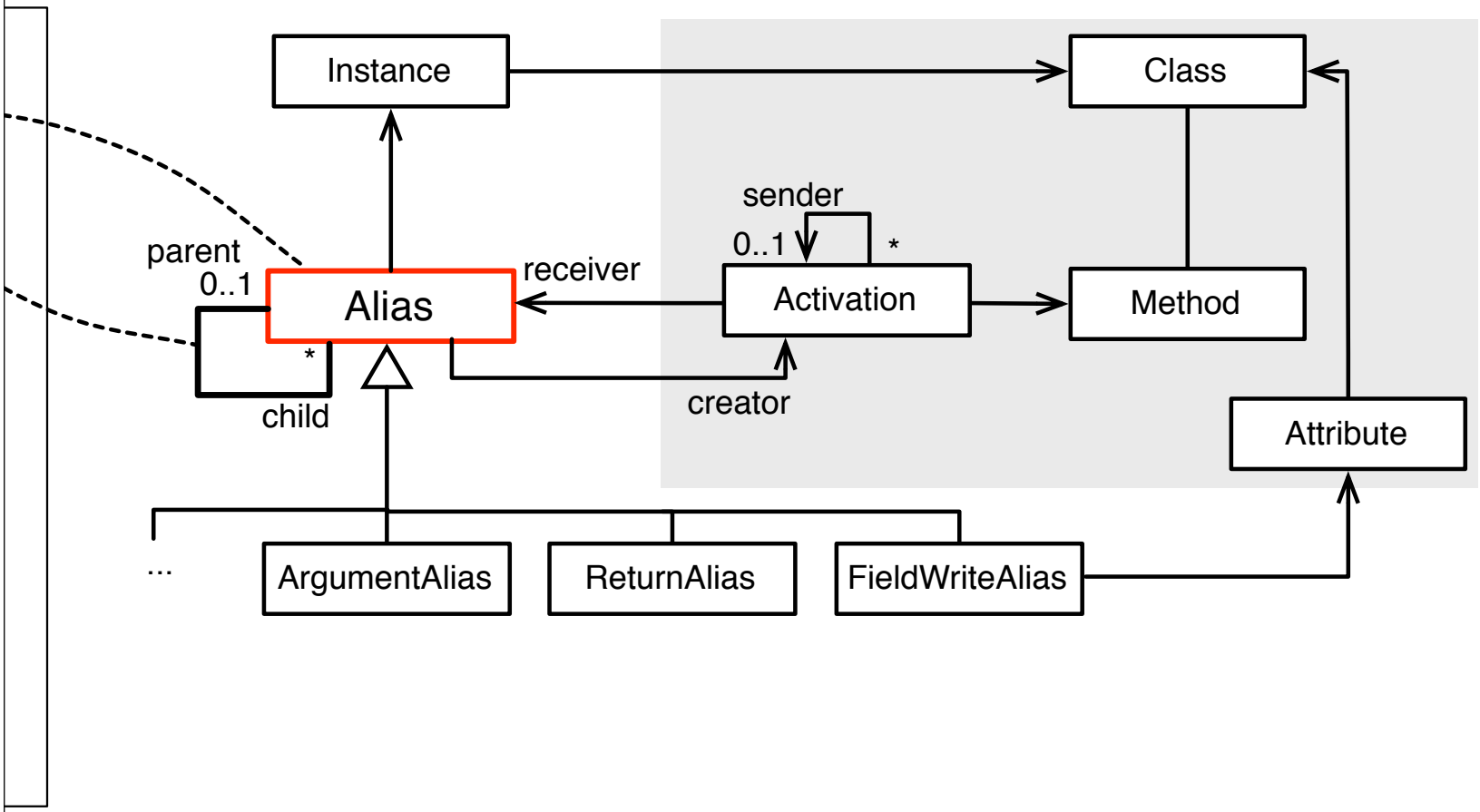
```
IRBuilder>>initialize
  ir := IRMethod new.
  ..

IRBuilder>>startNewSequence
    newSequence := IRSequence new.
    newSequence  method: ir.

IRSequence>>method: aMethod
   method := aMethod.

ASTTranslator>>ir
   ^ builder ir.

MethodNode>>generate
    ast := ASTTranslator new visitNode: self.
   ir := ast ir.
    ^ ir compiledMethod.
```



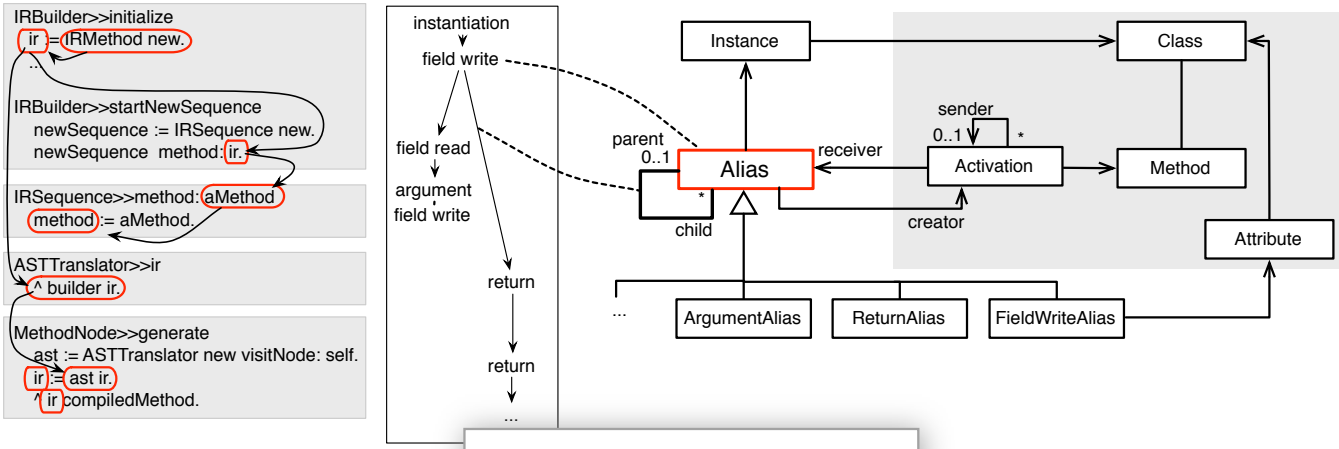
Representing object flows



Representing object flows



Summary



Questions?