Running Pharo on the GemStone VM

James Foster
VP of Finance & Operations, GemTalk Systems LLC

ESUG 2017 – Maribor, Slovenia

4 September 2017

Agenda

- **GemStone/S Introduction**
- Replacing Base Class Libraries
- Questions

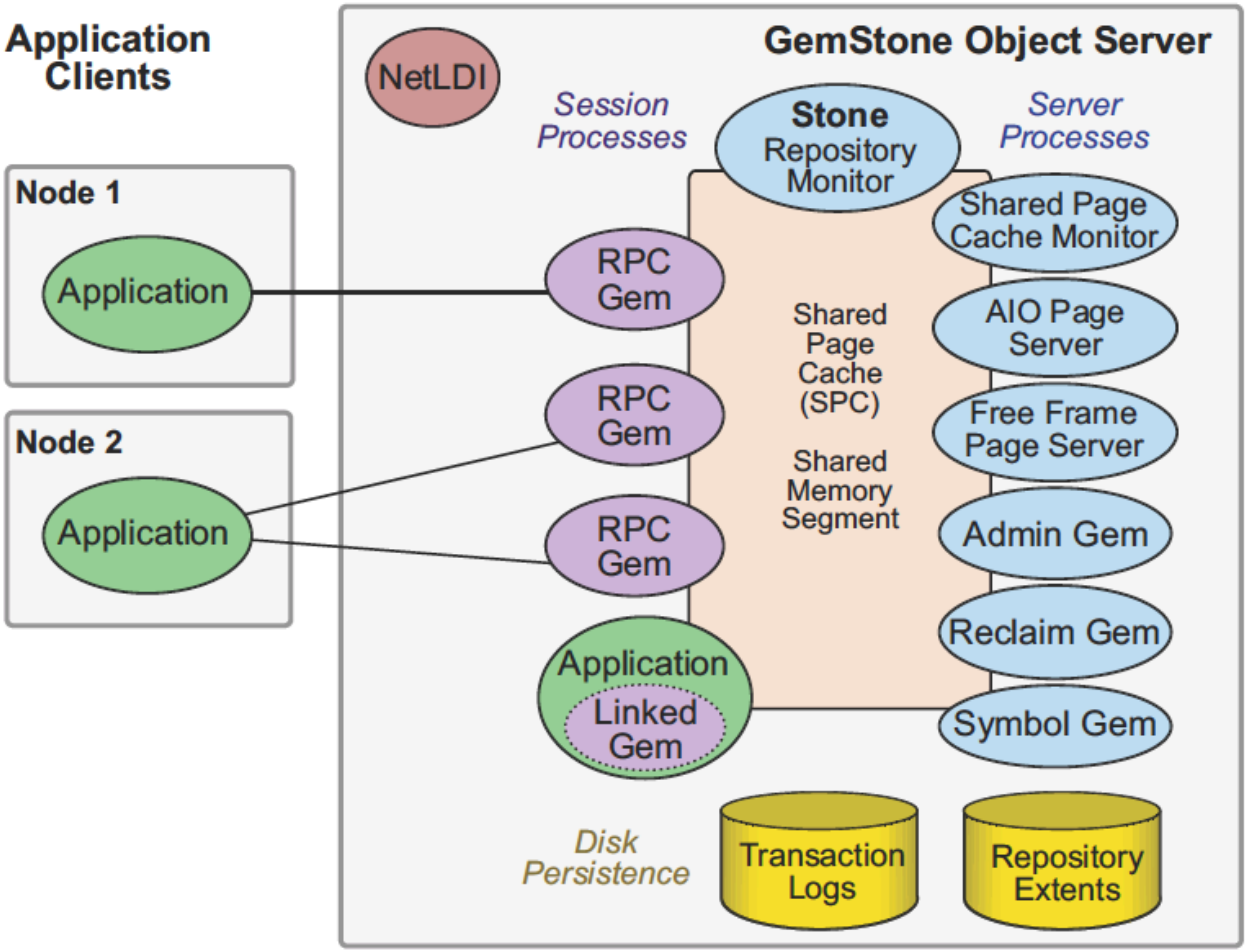
Limitations of traditional Smalltalks

- Object space (image) must fit in (virtual) RAM
- Object space visible to one (single-threaded) VM
- Sharing objects between VMs is difficult
  - Convert to non-object format (binary, XML, SQL)
  - No built-in object identity for multiple exports
- Object state is lost when VM exits

Welcome to the magical world of GemStone

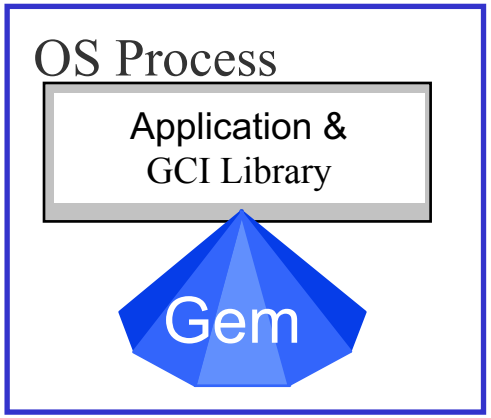
- Object space limited by disk, not RAM
- Object space shared across multiple VMs on multiple hosts
- Transactional persistence

GemStone/S Architecture

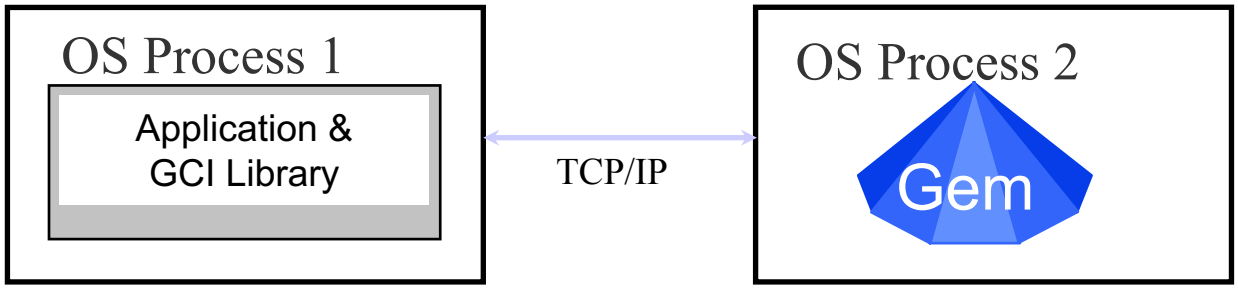


Gem Types

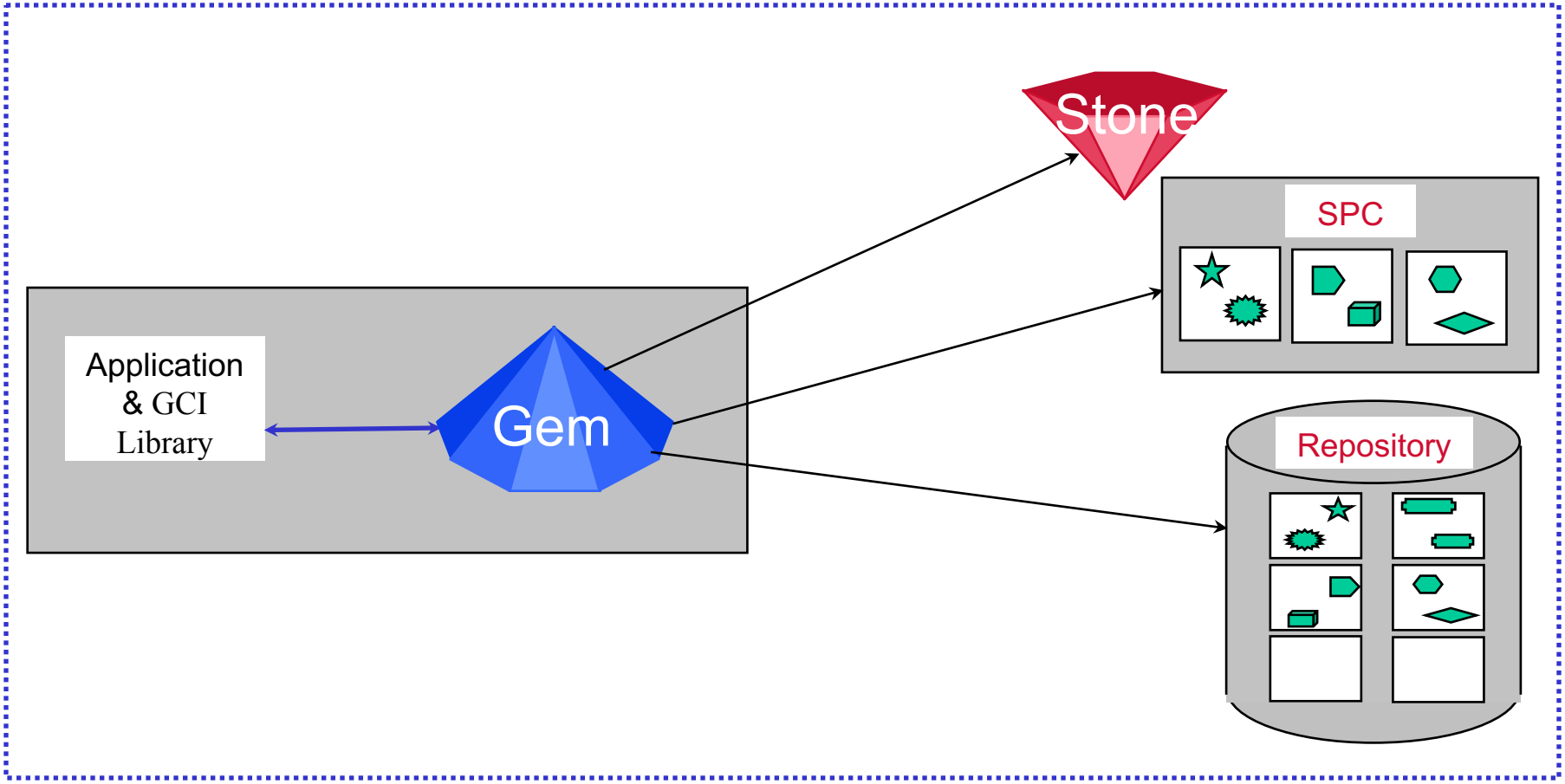
- Linked Gem
  - Gem in application process space



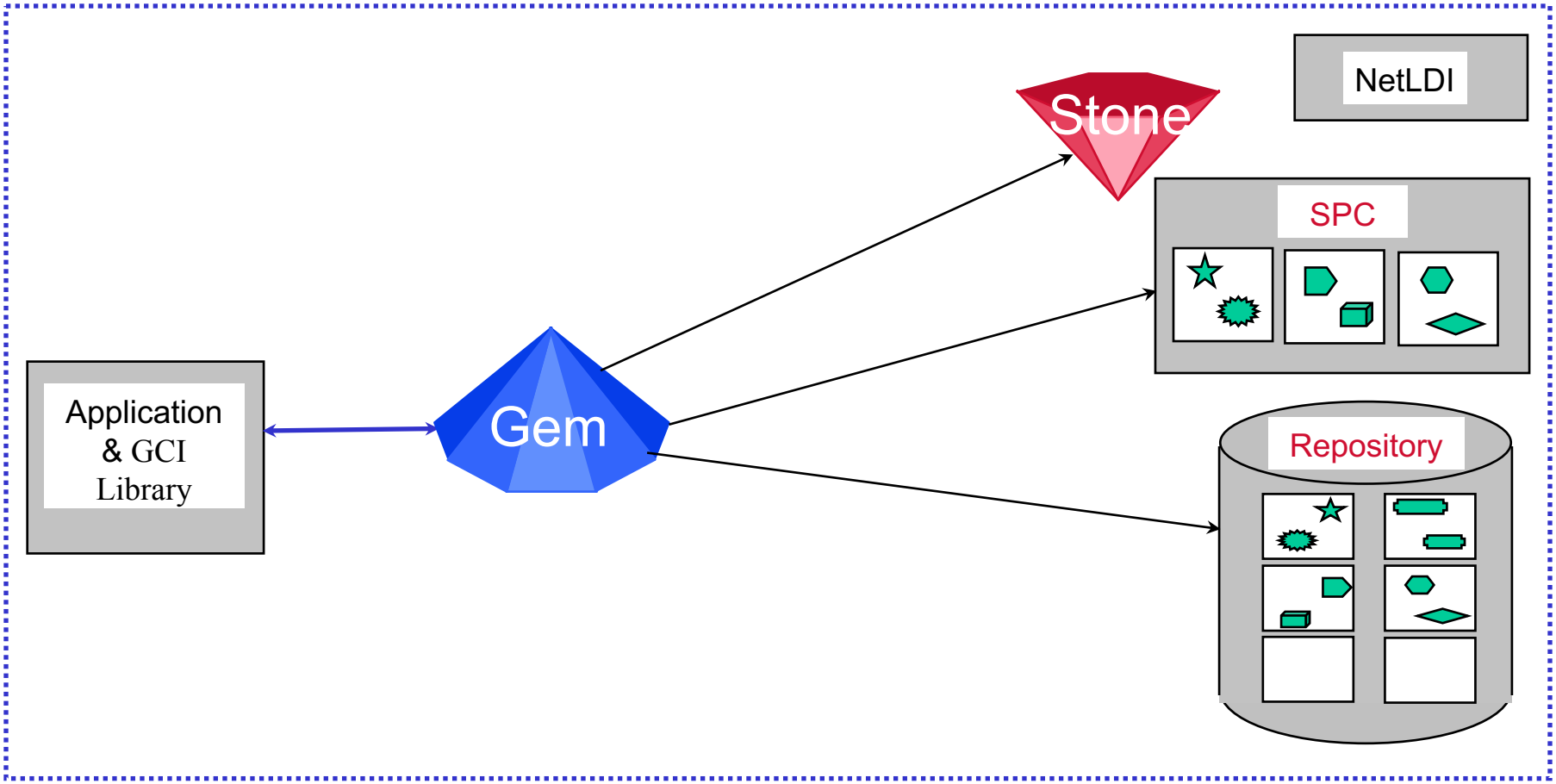
- Remote Procedure Call (RPC) Gem
  - GCI library in application space
  - Gem has separate process



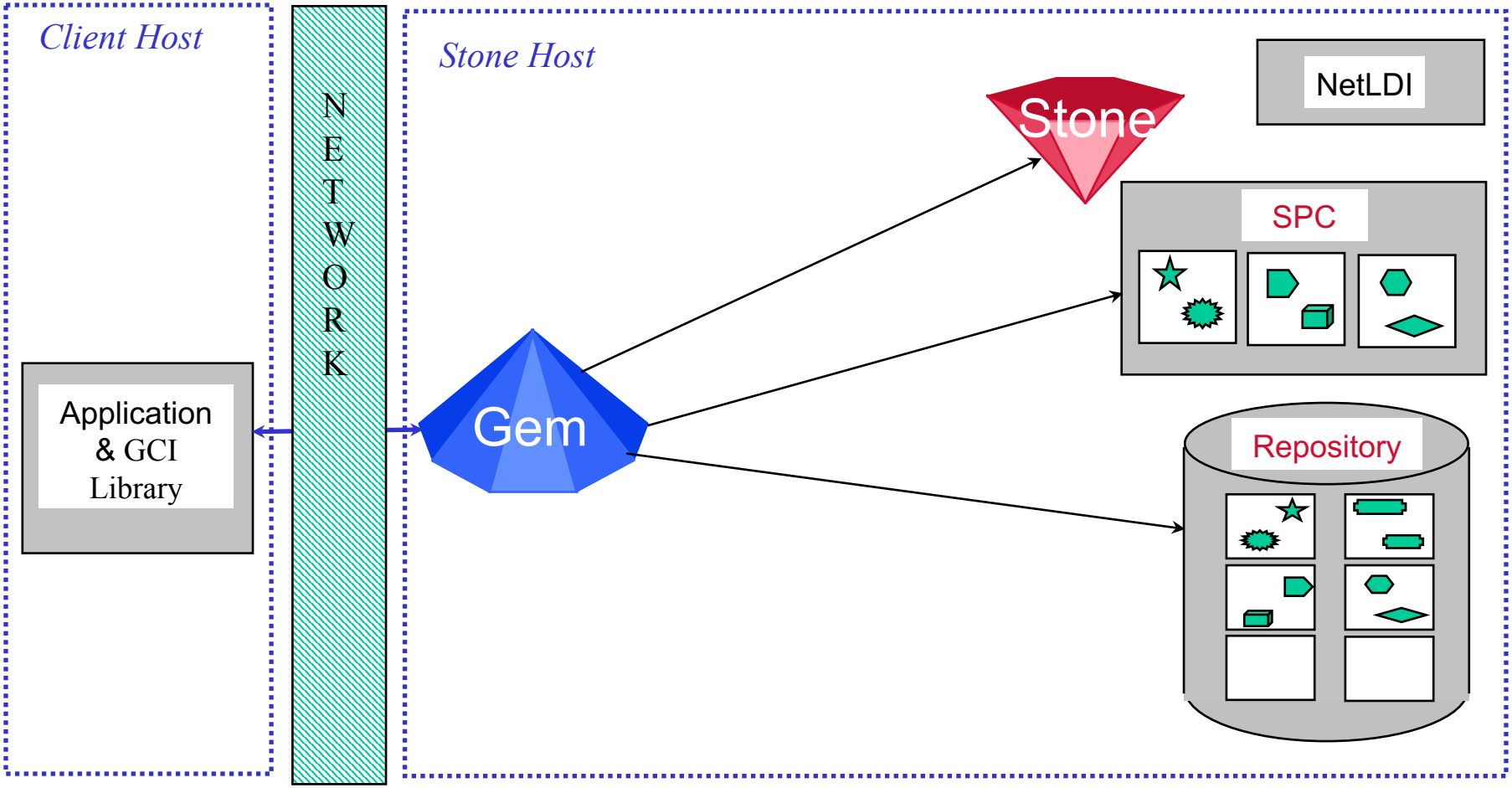
One-Machine Process Locations (Linked Gem)



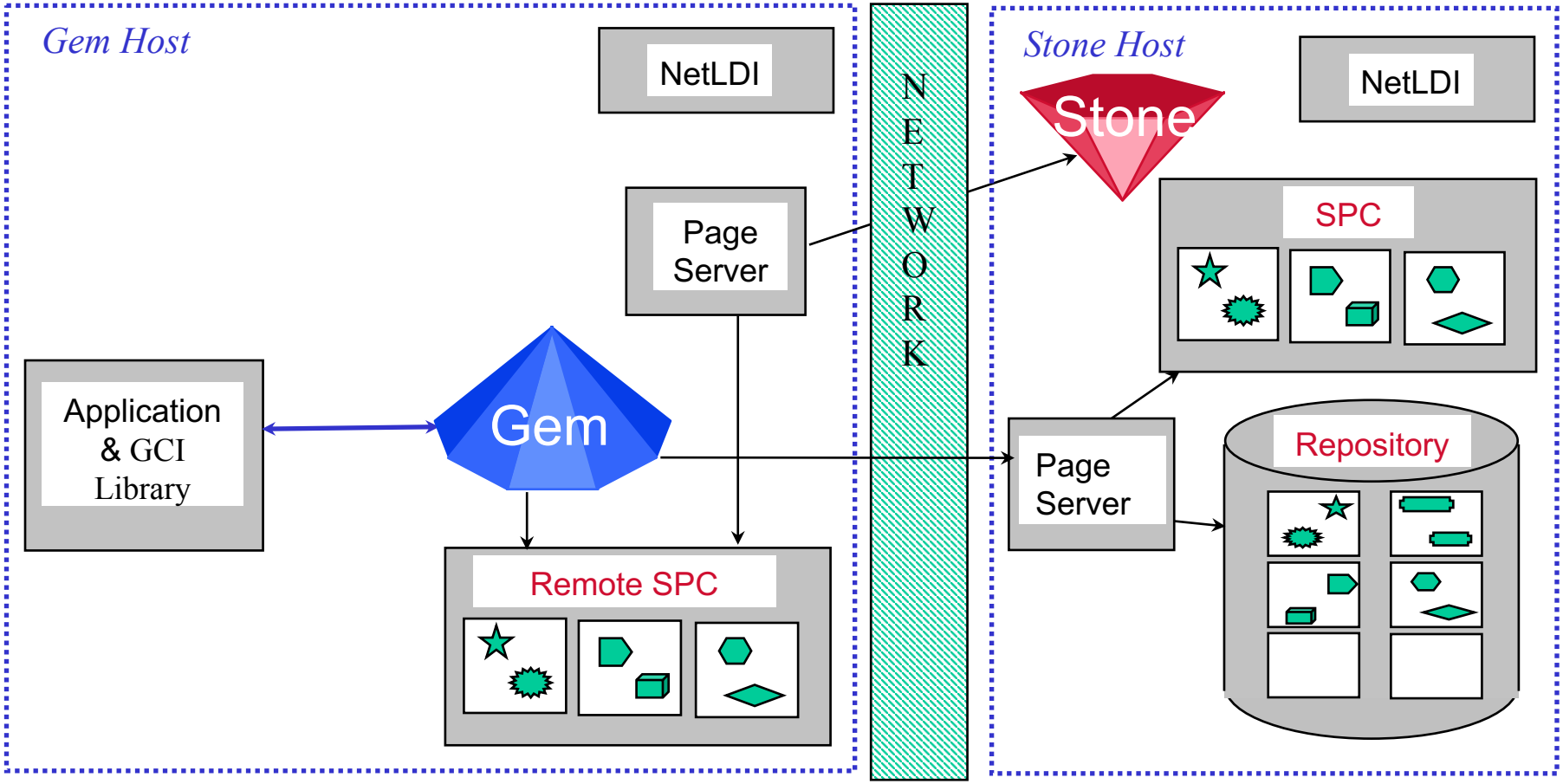
One-Machine Process Locations (RPC Gem)



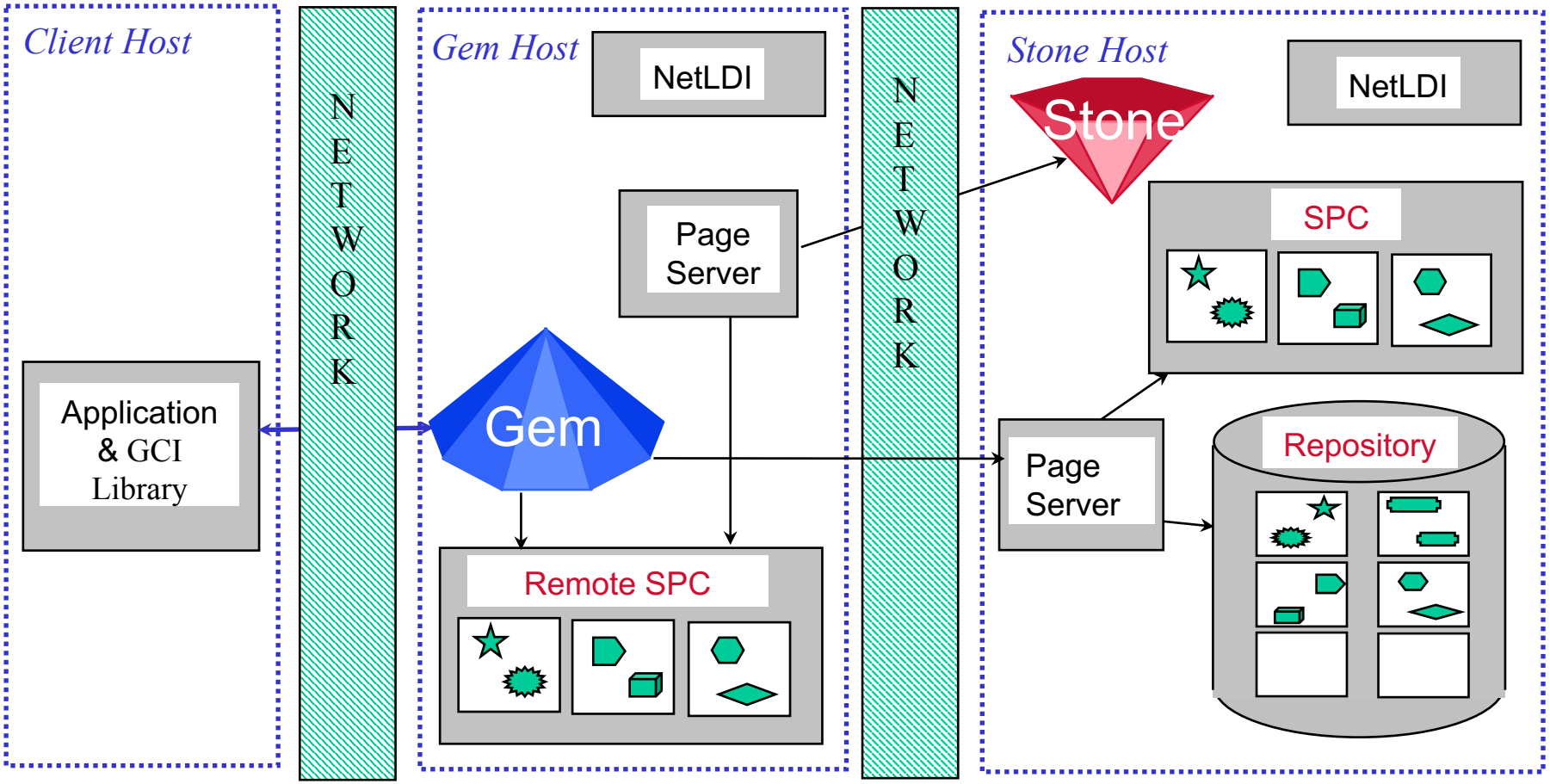
Two-Machine Process Locations (Gem on Stone Host)



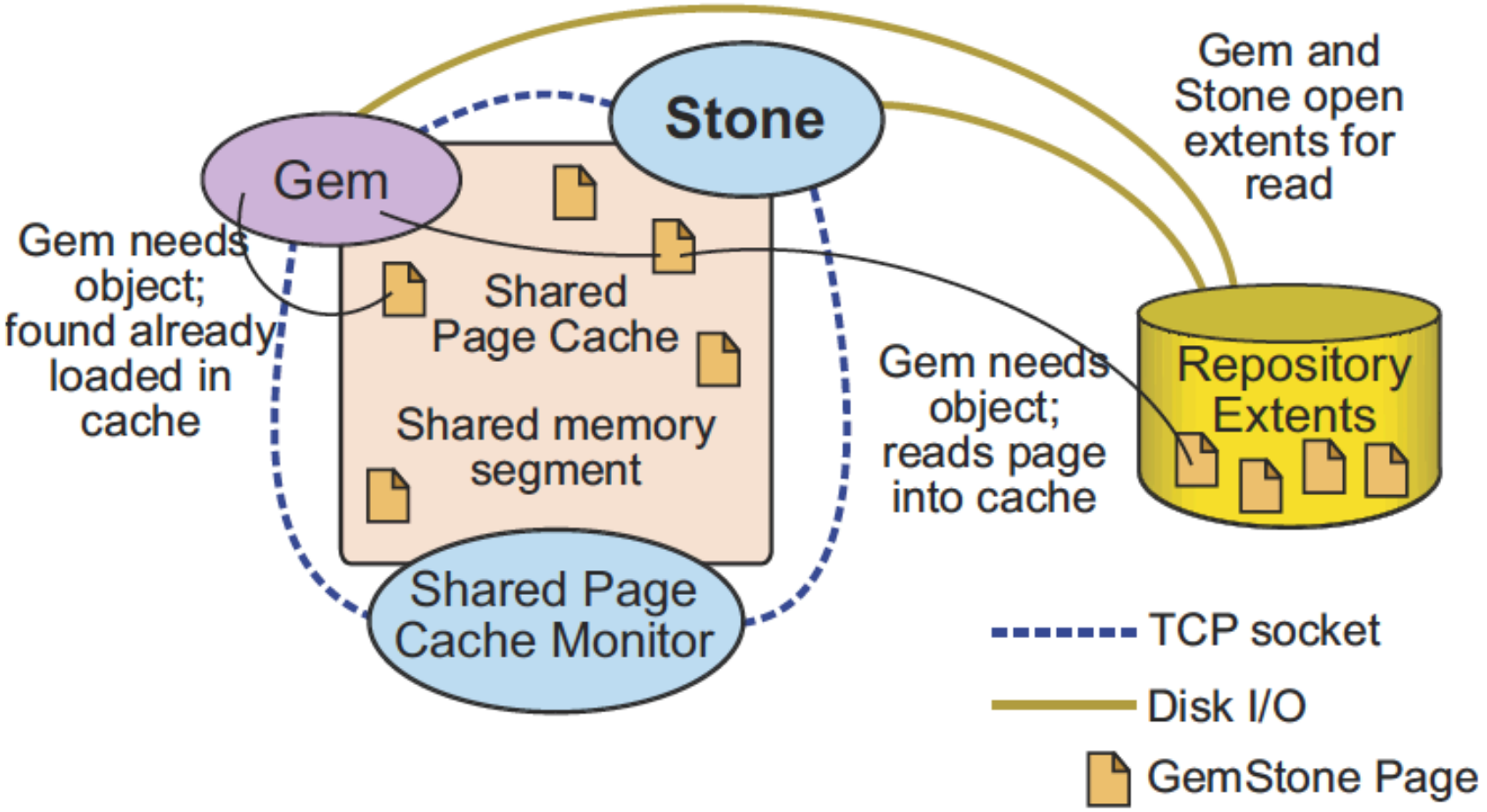
Two-Machine Process Locations (Gem Remote from Stone)



Three-Machine Process Locations



Shared Page Cache



Agenda

- GemStone/S Introduction
- **Replacing Base Class Libraries**
- Questions

Replacing Base Class Libraries

- Smalltalk was intended to allow exploration
- Portability
  - ANSI
  - Grease
- Can we go to a lower level?

- One approach to learn more

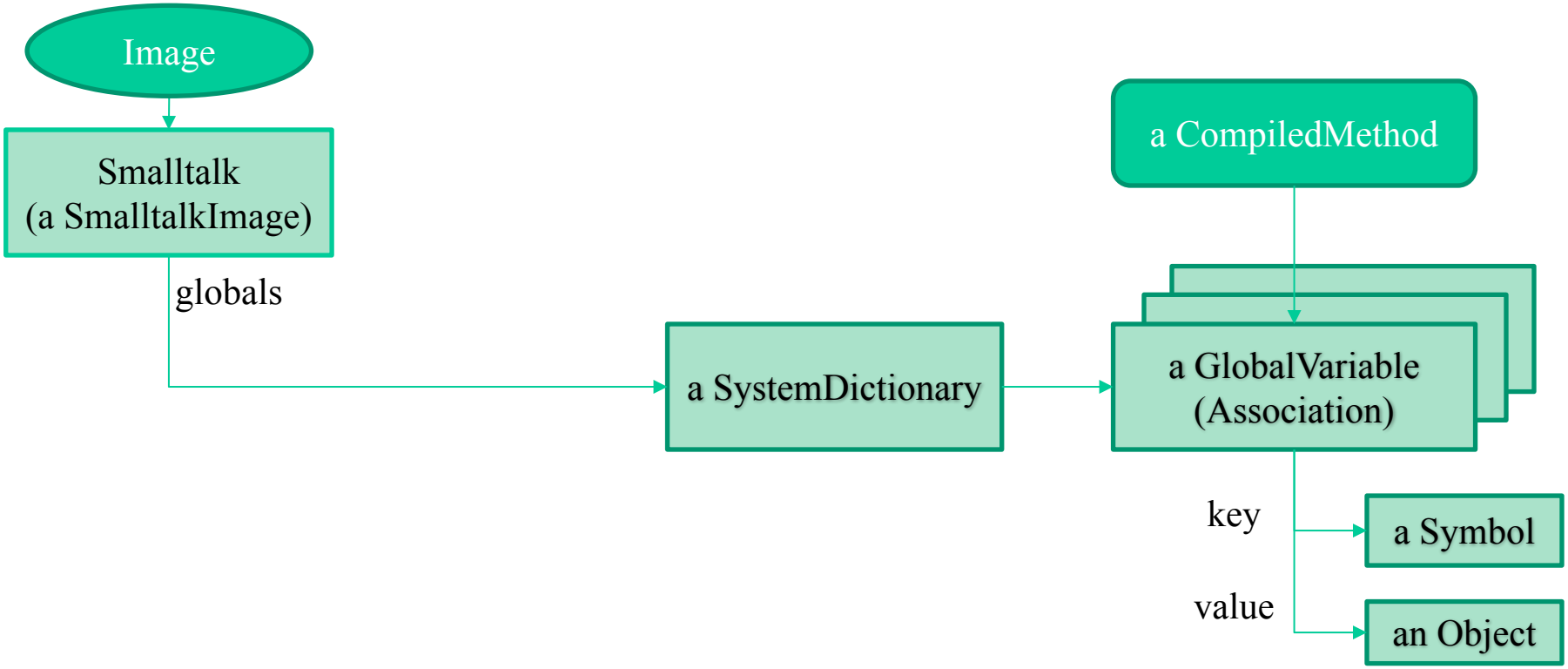
Option 1: Complete Replacement

- Not possible in most Smalltalks
  - Creating methods requires code
- GemStone schema editing with Topaz
  - Build process starts with a few base classes
  - But no methods for any class (initially)
- Challenges
  - Difficult to debug (most tools are internal)
  - VM has knowledge of schema

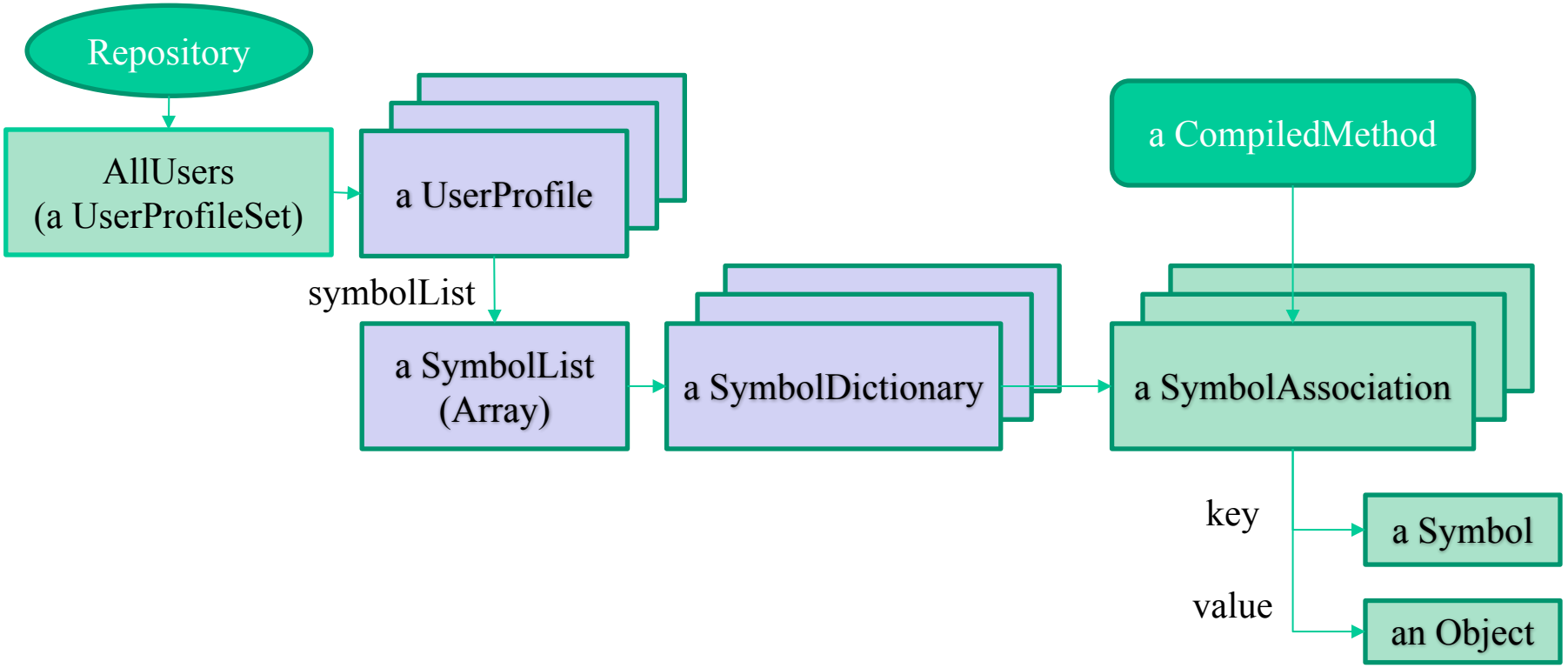
Option 2: Parallel Class Hierarchy

- GemStone's namespace model allows separate hierarchy

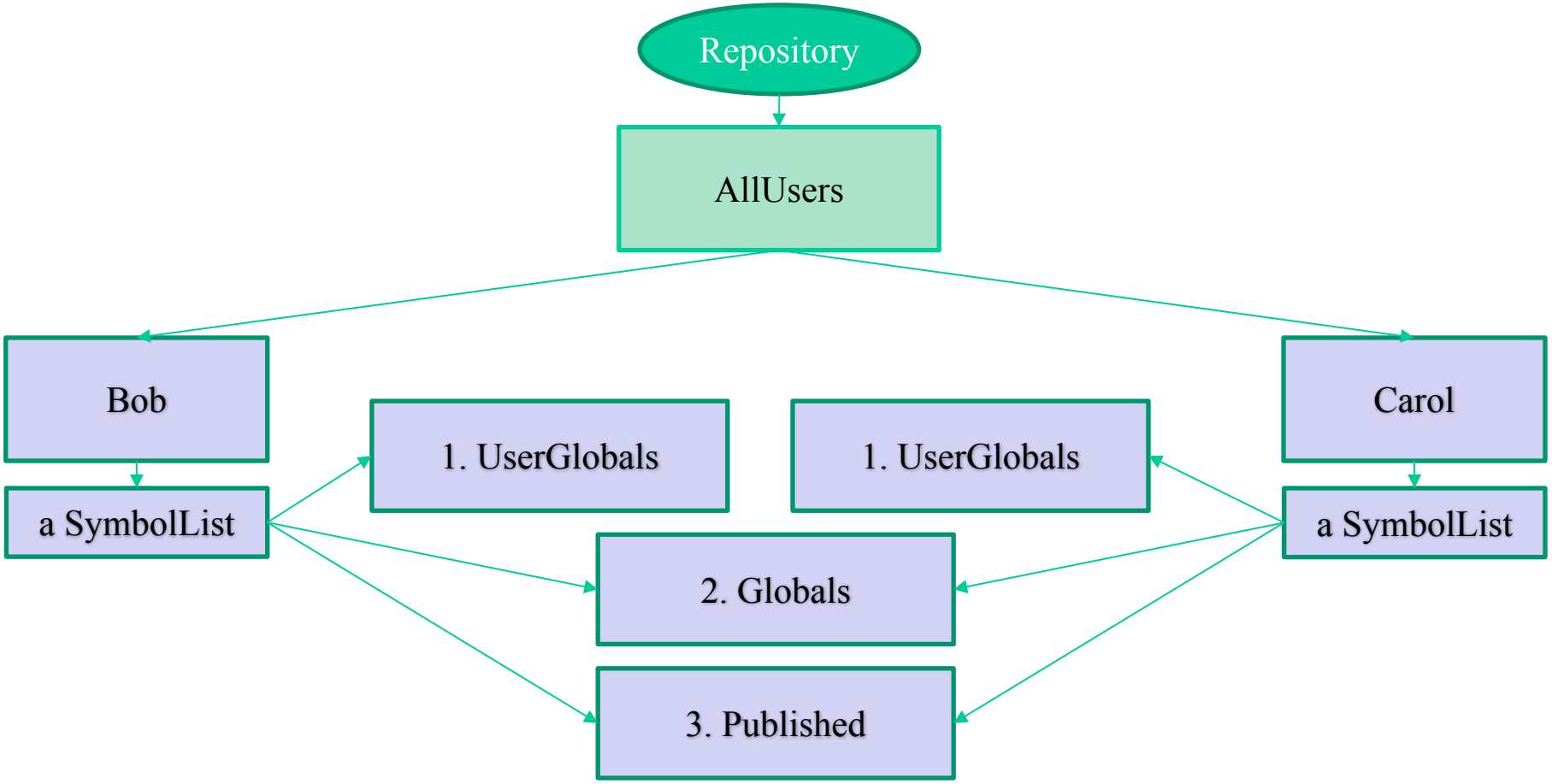
Pharo Global Lookup



GemStone/S Global Lookup

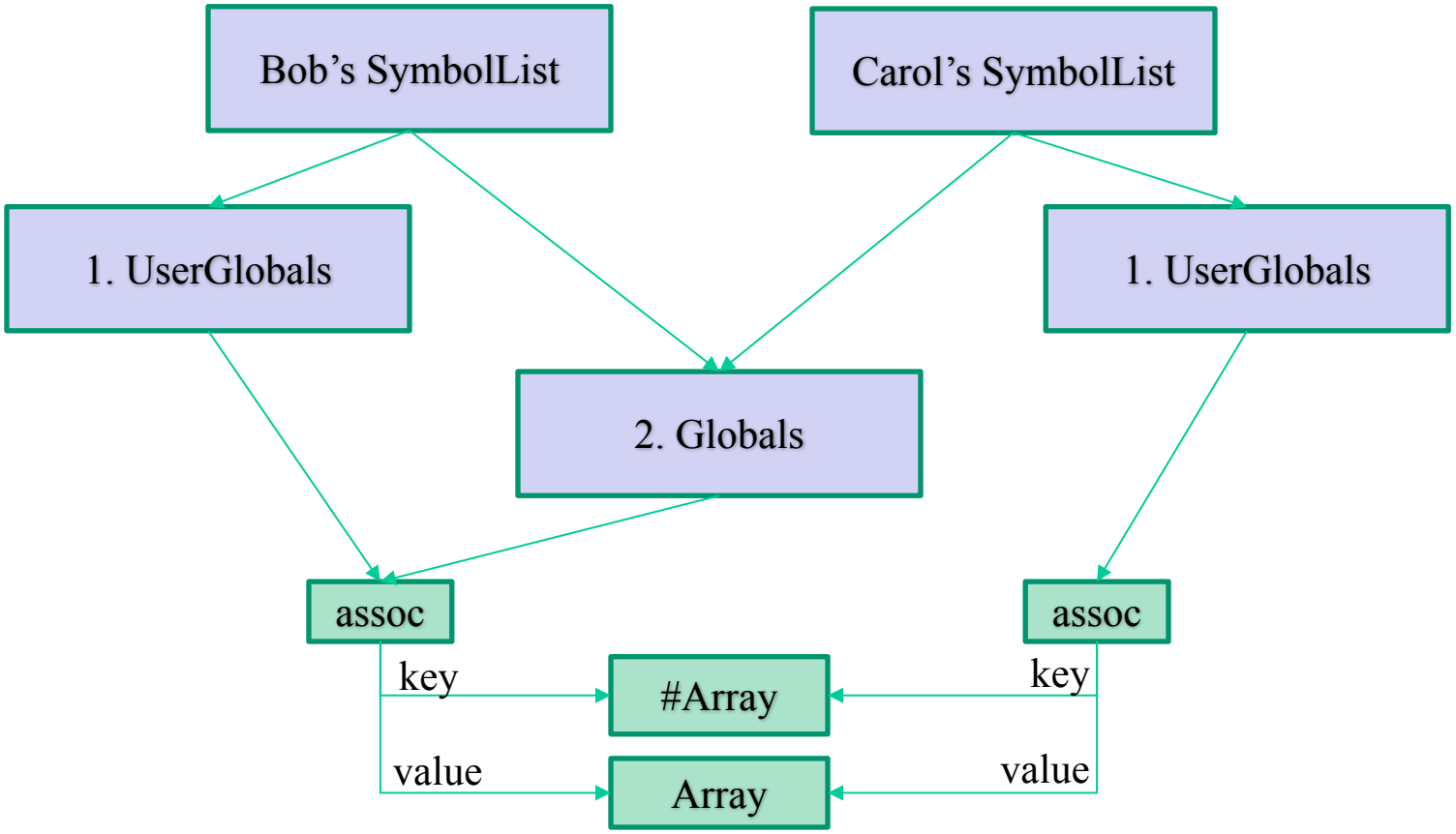


Unique or Shared SymbolDictionary Instances

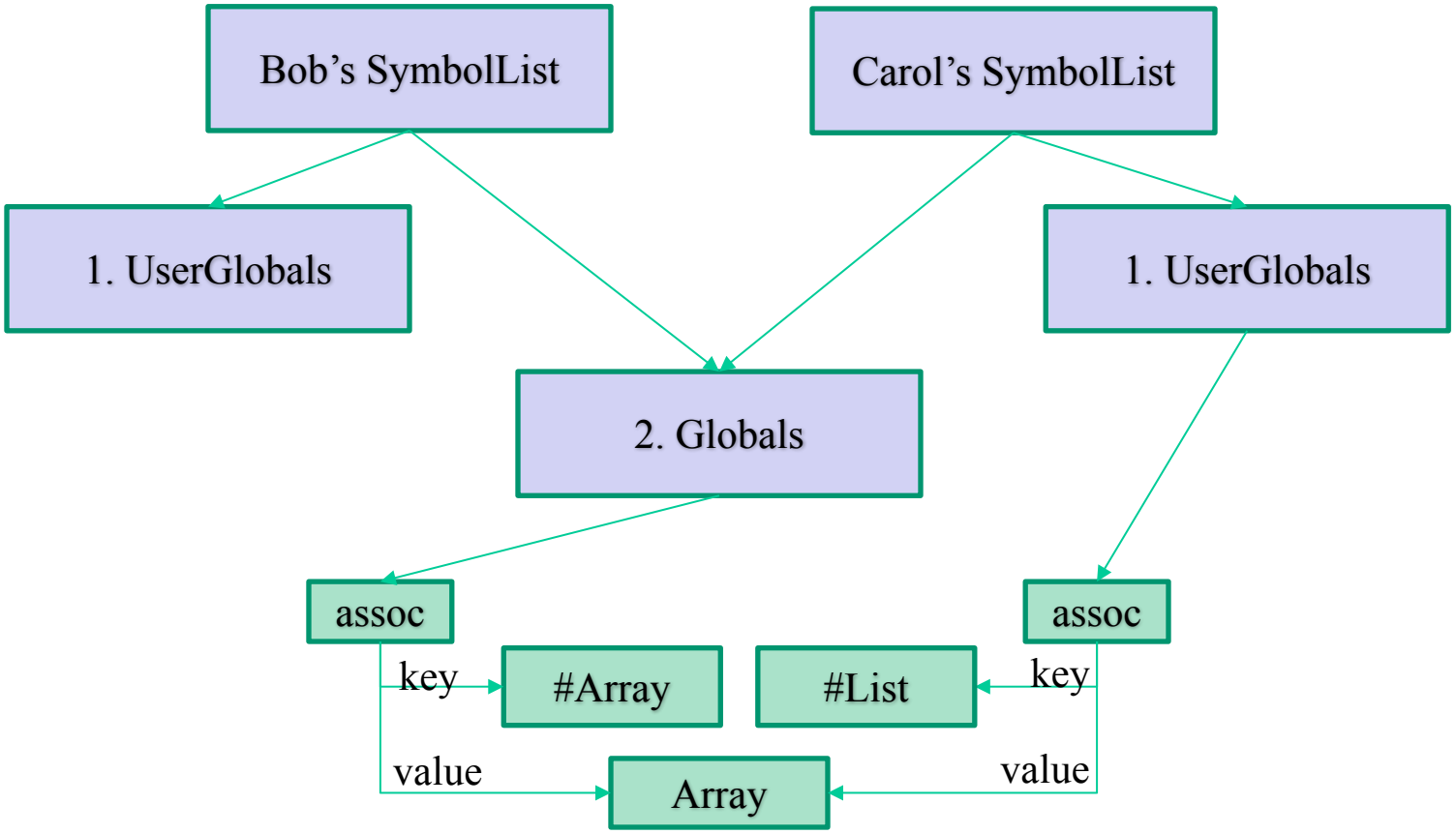


SymbolDictionary instances

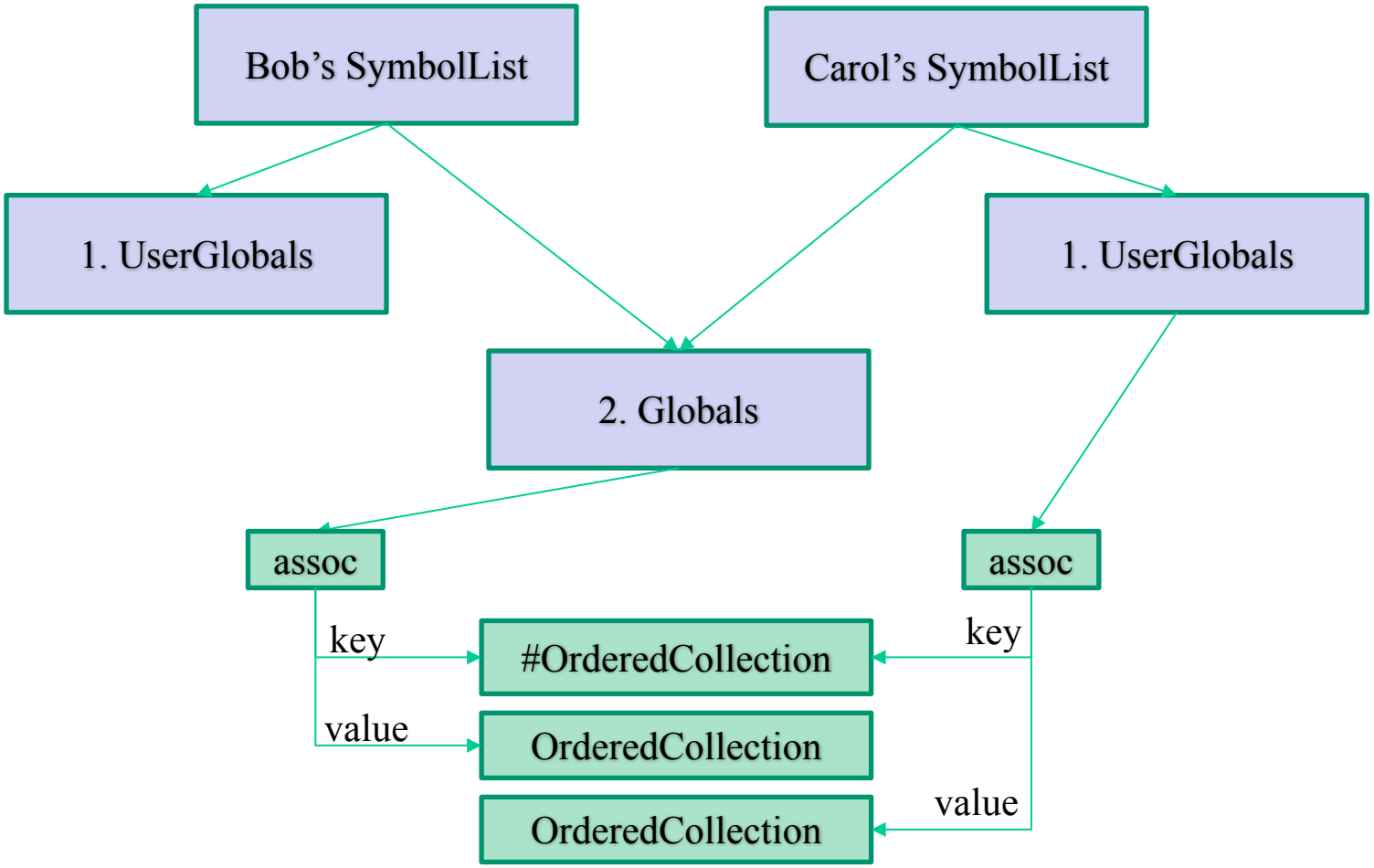
Unique or Shared SymbolAssociation Instances



Unique Keys


Bob's SymbolList
Carol's SymbolList
1. UserGlobals
1. UserGlobals
2. Globals
assoc
assoc
key
#Array
#List
key
value
Array
value

Unique Values



Problems with Parallel Class Hierarchy

- GemStone's namespace model allows separate hierarchy
- But complications exist:
  - Literals
  - Classes known to the VM

Complication: Literals (and their superclasses)

- Array: #()
- BlockClosure: []
- Boolean: true, false
- ByteArray: #[1 2 3]
- Character: $a
- Float: 1.23
- SmallInteger: 42
- String: ‘Smalltalk’
- Symbol: #Array
- UndefinedObject: nil

Complication: Classes Known to the VM

- Behavior, Class, Metaclass
- Exception, MessageNotUnderstood, ZeroDivide, …
- Pragma
- Process
- ProcessorScheduler

- **So, we need to use (some) base classes**

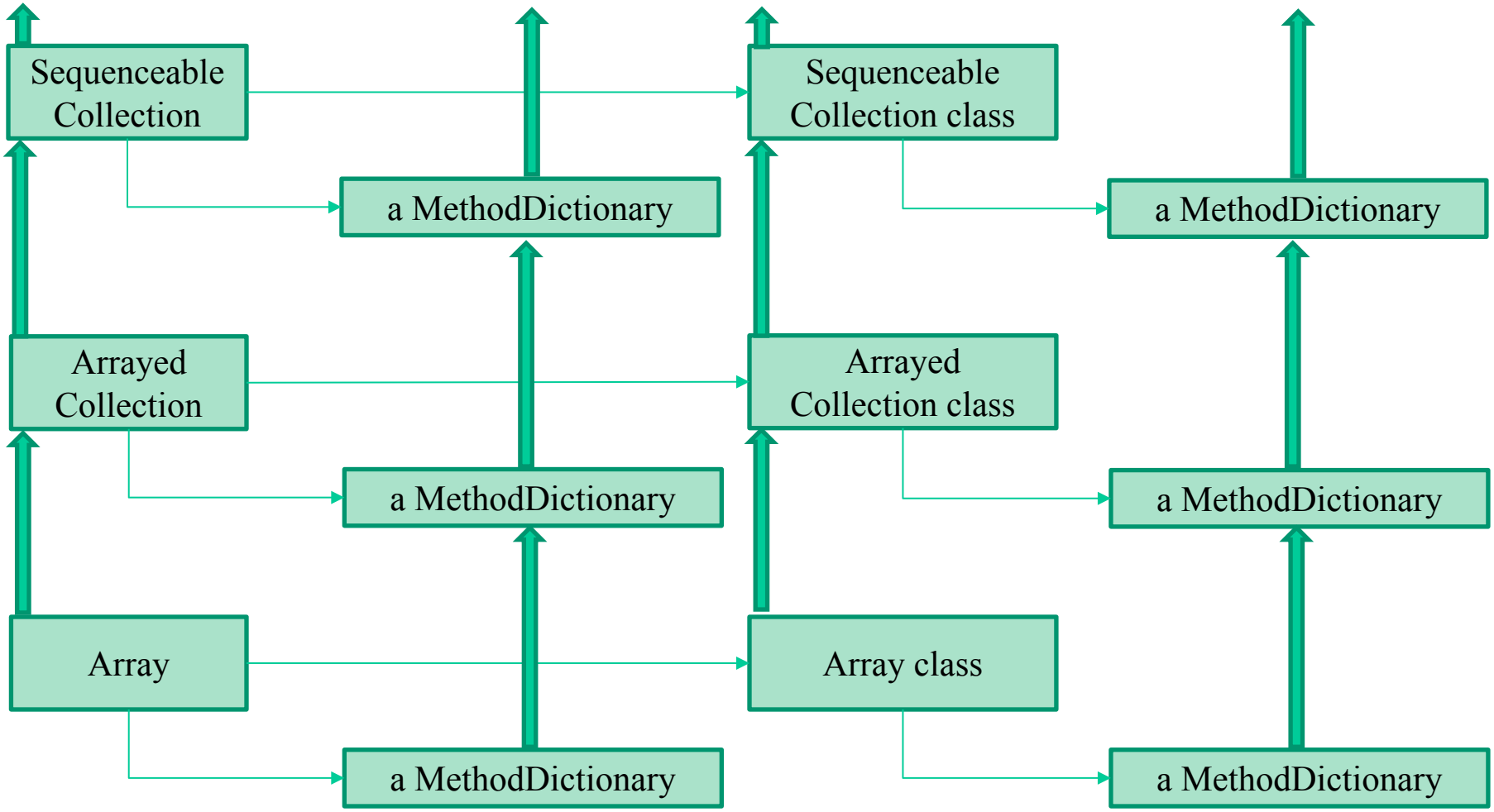
Problem: Conflicting Implementations

- Array>>printOn:
  - Pharo
    - #(1 2 3) printString '#(1 2 3)'
  - GemStone
    - #(1 2 3) printString 'anArray(1, 2, 3)'

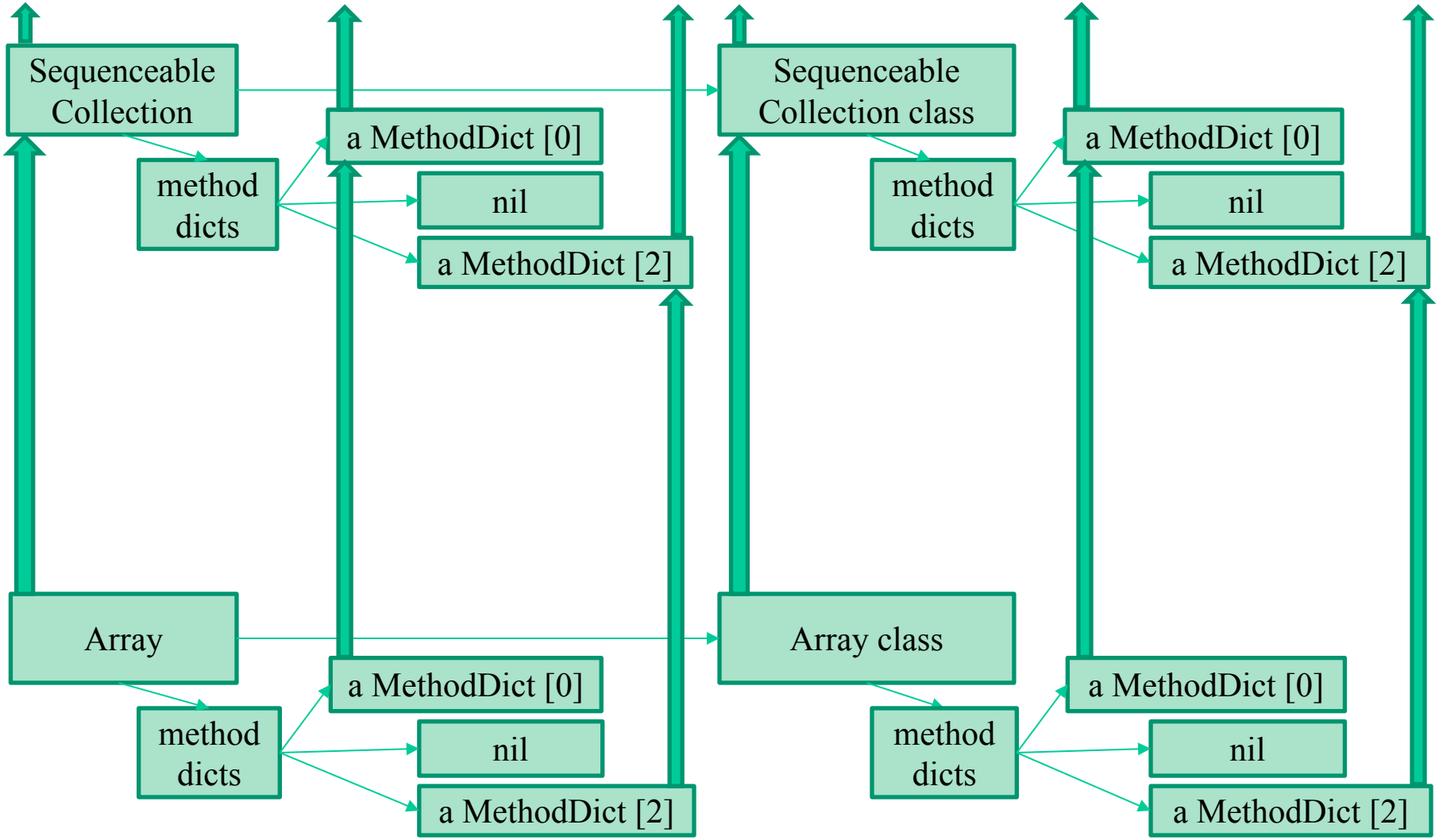
Option 3: Parallel Methods *in* GemStone Classes

- Each GemStone class has a *collection* of MethodDictionary instances
  - Methods are compiled into an “environment”
  - Message sends to same environment (by default)
- Method environments:
    0 = GemStone/S (default)
    1 = Maglev (reserved for Ruby)
    2+ are for others
  - We use 2 for Pharo

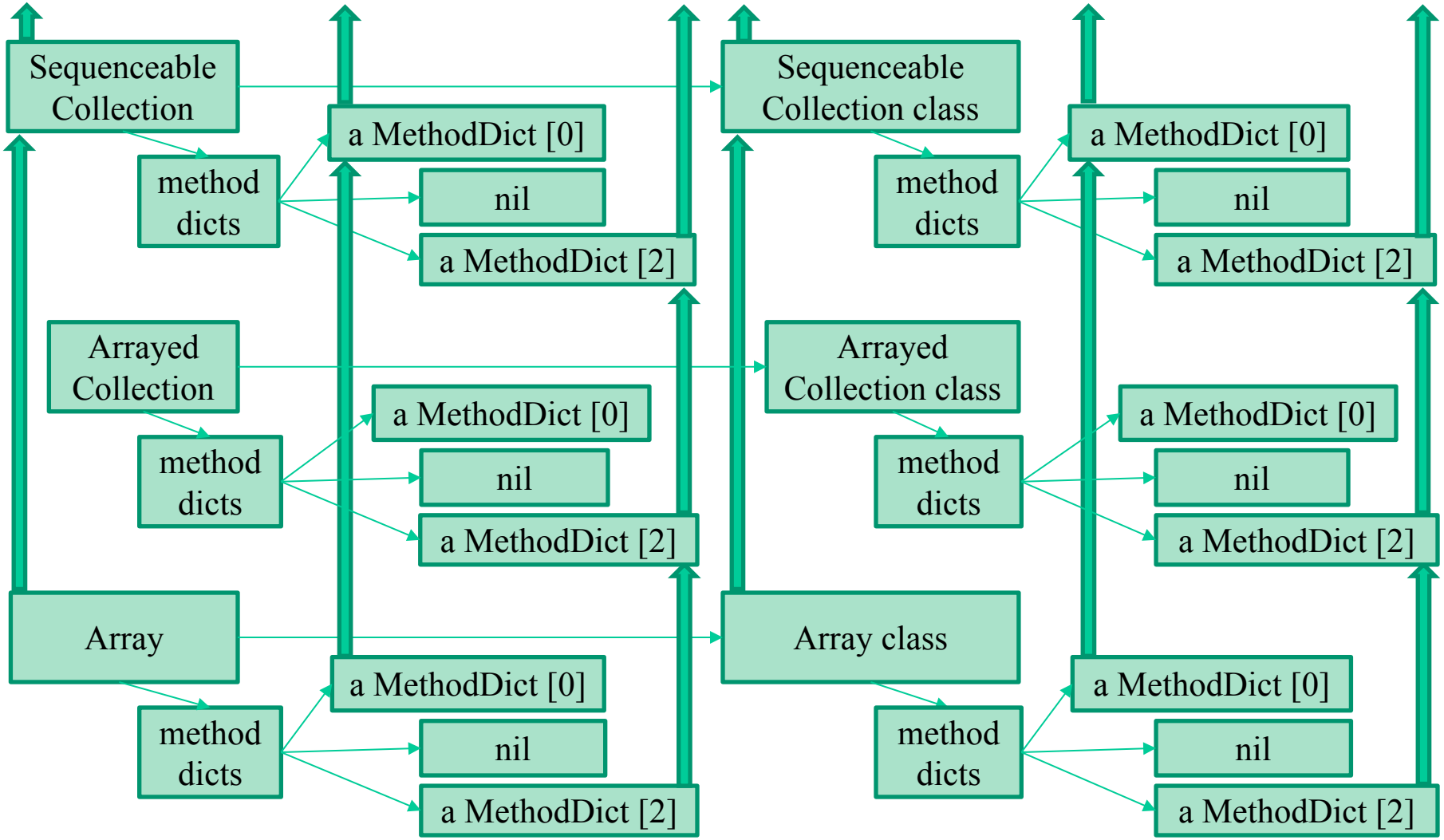
Pharo Method Dictionaries



GemStone Method Dictionaries - 1

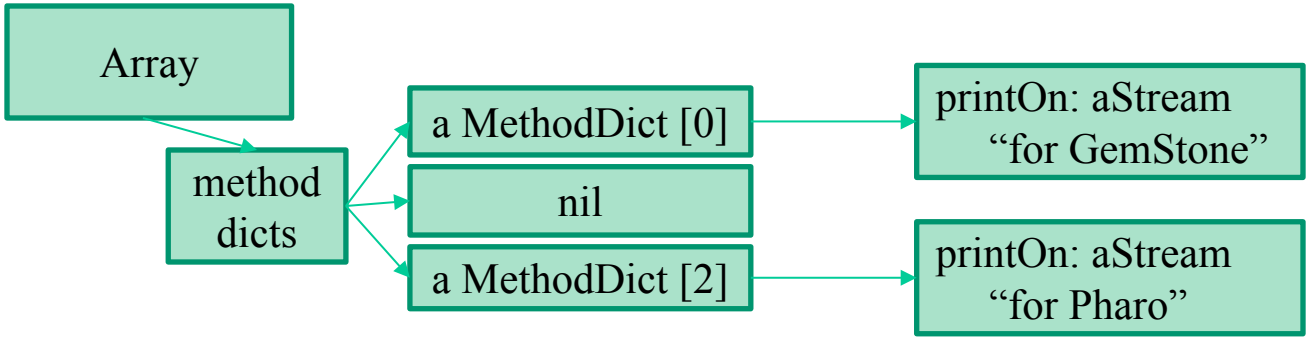


GemStone Method Dictionaries - 2



Sample Messages

- Implicit
  - “in env 0” #(1 2 3) printString 'anArray(1, 2, 3)'
  - “in env 2” #(1 2 3) printString '#(1 2 3)'
- Explicit
  - #(1 2 3) @env0:printString 'anArray(1, 2, 3)'
  - #(1 2 3) @env2:printString '#(1 2 3)’



Problems with Using GemStone Classes

- GemStone class might have different schema
  - OrderedCollection in Pharo
    - Instance variables: (array firstIndex lastIndex)
  - OrderedCollection in GemStone
    - No instance variables
  - Since OrderedCollection is not known to the compiler, we don't have to use GemStone's class

Options

1. Replace all classes
   But schema is wrong in some cases
   And we would like to use tools during development
2. Use parallel classes exclusively
   But compiler & VM know about some GemStone classes
3. Use parallel methods exclusively
   But schema is different for some classes
4. **Hybrid**
   Use GemStone classes when necessary
   Use parallel classes otherwise
   Use method environment 2 for everything

Tools

- Topaz
  - GemStone's command-line interface
- GemBuilder for Smalltalk (GBS)
  - GemStone's GUI IDE for VA & VW
- Others
  - tODE
  - gt4gemstone
  - Jade

Issues

- Incomplete Class Types
  - GemStone has in-memory Ephemerons, but no Weak references
  - No 32-bit objects
- GemStone compiler is in VM
  - Not as easy to experiment with new syntax
- Low-level objects may need extensive rewrite
  - BlockClosure, Process, ProcessorScheduler, etc.

Demo

- http://files.pharo.org/get-files/60/pharo-minimal.zip
- http://downloads.gemtalksystems.com/pub/GemStone64/3.4.0-Alpha4/
- https://github.com/jgfoster/PharoGs/tree/james/james

Questions?

James G. Foster
VP of Finance & Operations

GemTalk Systems LLC
15220 NW Greenbrier Pkwy., Suite 240
Beaverton, Oregon, 97006
Voice & Fax: +1 503 766 4714
james.foster@gemtalksystems.com

www.gemtalksystems.com