Marcel Weiher (@mpweiher)

Objective-SmallTalk

Objective-SmallTalk

- Embeddable SmallTalk language (Mac, iOS, Linux, Windows)
- Objective-C framework (peer/interop)
- Generalizes Objects+Messages to Components+Connectors
- Enable composition by solving Architectural Mismatch

Objective-SmallTalk

Architectural Mismatch

Architectural Mismatch

- Packaging mismatch
- Interactive programs vs. programming languages
- Programs vs. Systems, Computers do not compute

Packaging Mismatch

- Garlan et al ('95)
- Functionality available but not accessible
- Who has the thread of control?
- → Minimize assumptions
- → (see also: Hexagonal, Ports/Adaptors, Naked Objects)

Interactive Programs

- "Programs = Algorithms + Data Structurs + Architecture"
- y = f(x) v. interactive programs
- Transformational v. reactive
- Temperature Converter Example (Modularity '16)

The Gentle Tyranny of Call/Return

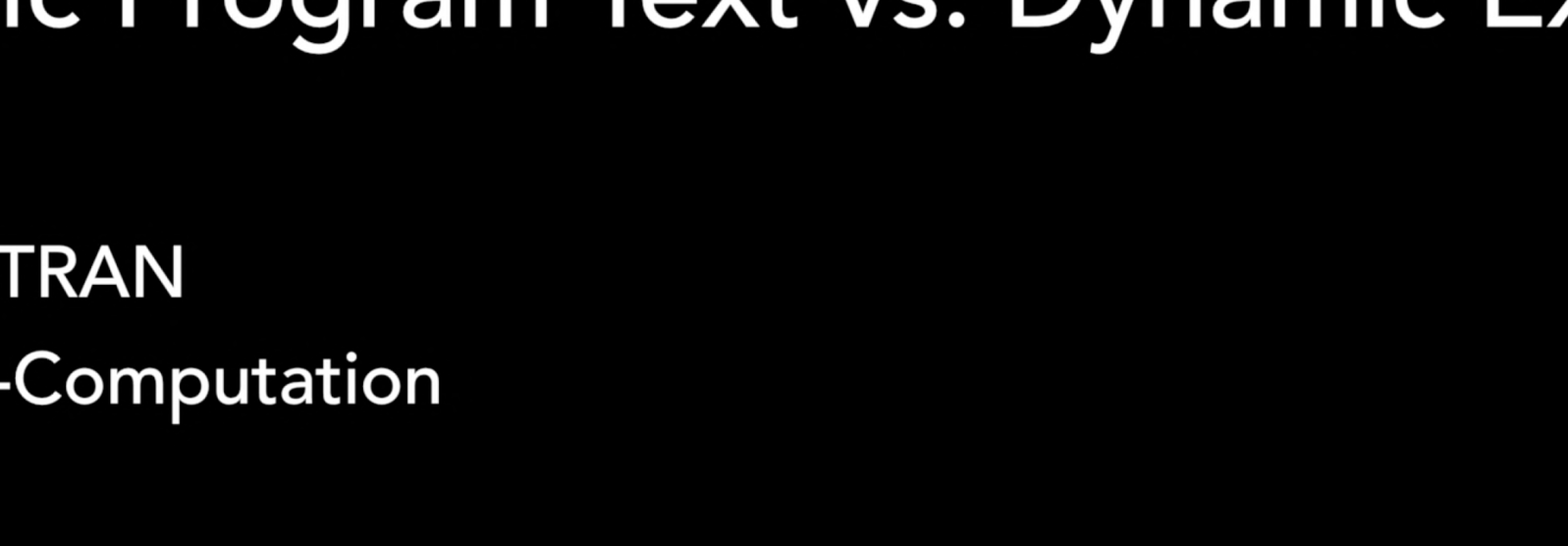
- Feymnan: we name everything just a little wrong
- Multiparadigm: Procedural, OO and FP!
- Guy Steele: it's no longer about completion
- Oscar Nierstrasz: we were told we could just model the domain
- Andrew Black: good OO students antropmorphise the objects

Static Program Text vs. Dynamic Execution

- FORTRAN

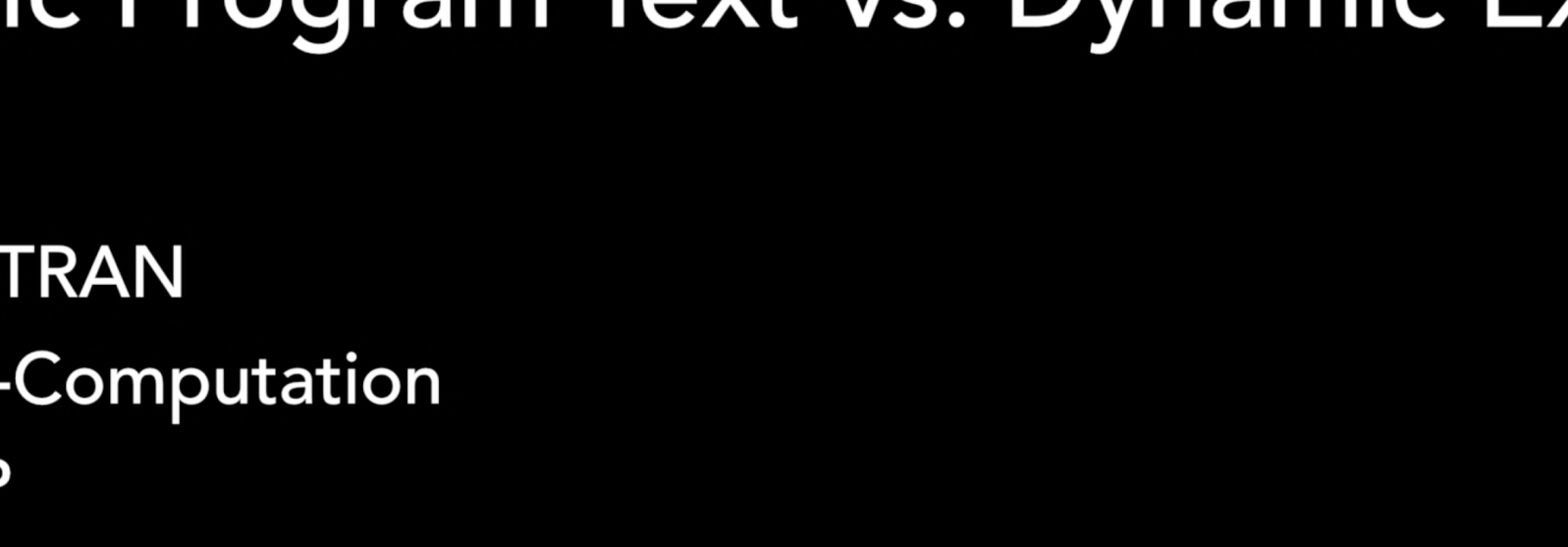
Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation



Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation
- OOP

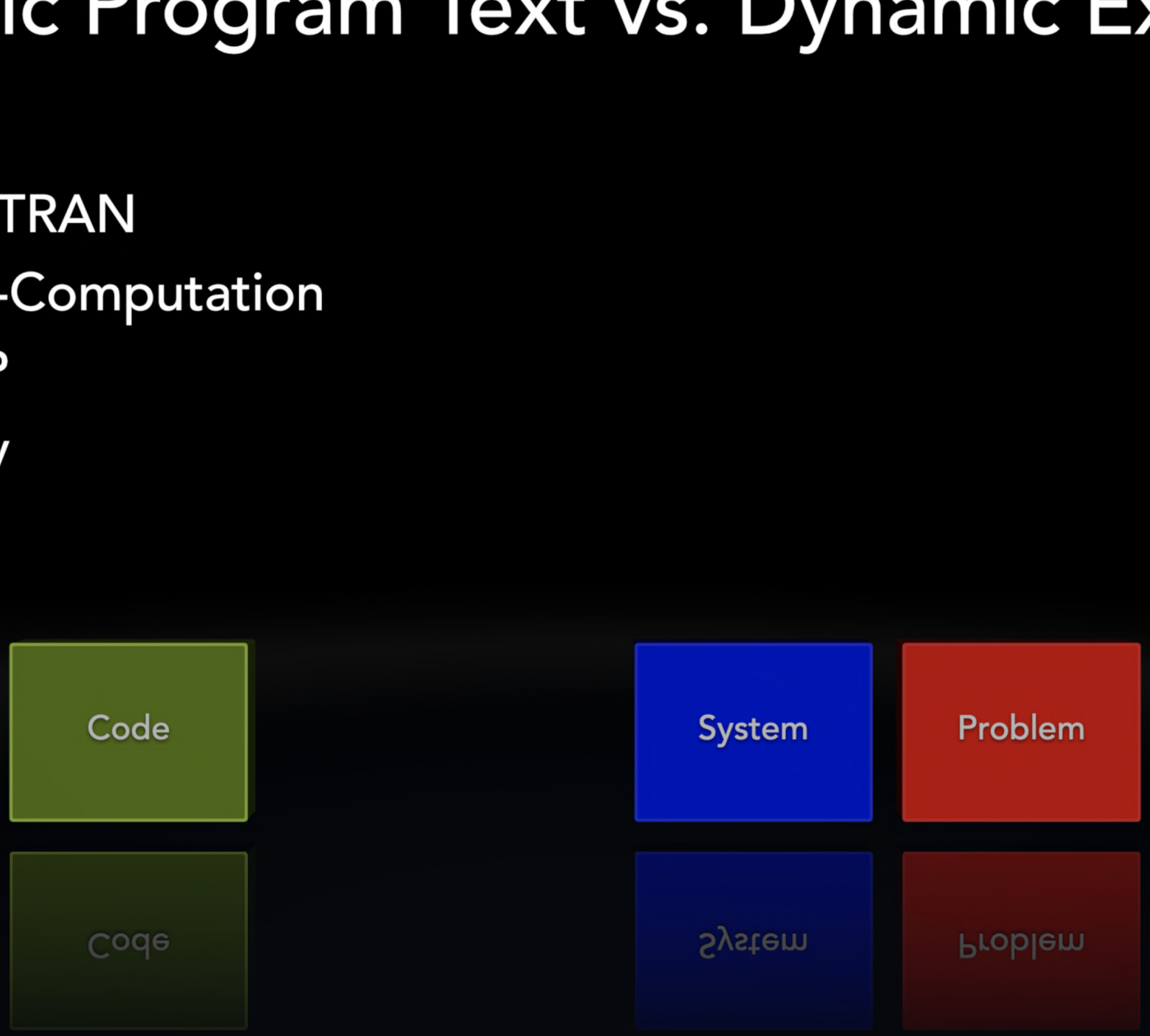


Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation
- OOP
- Ruby

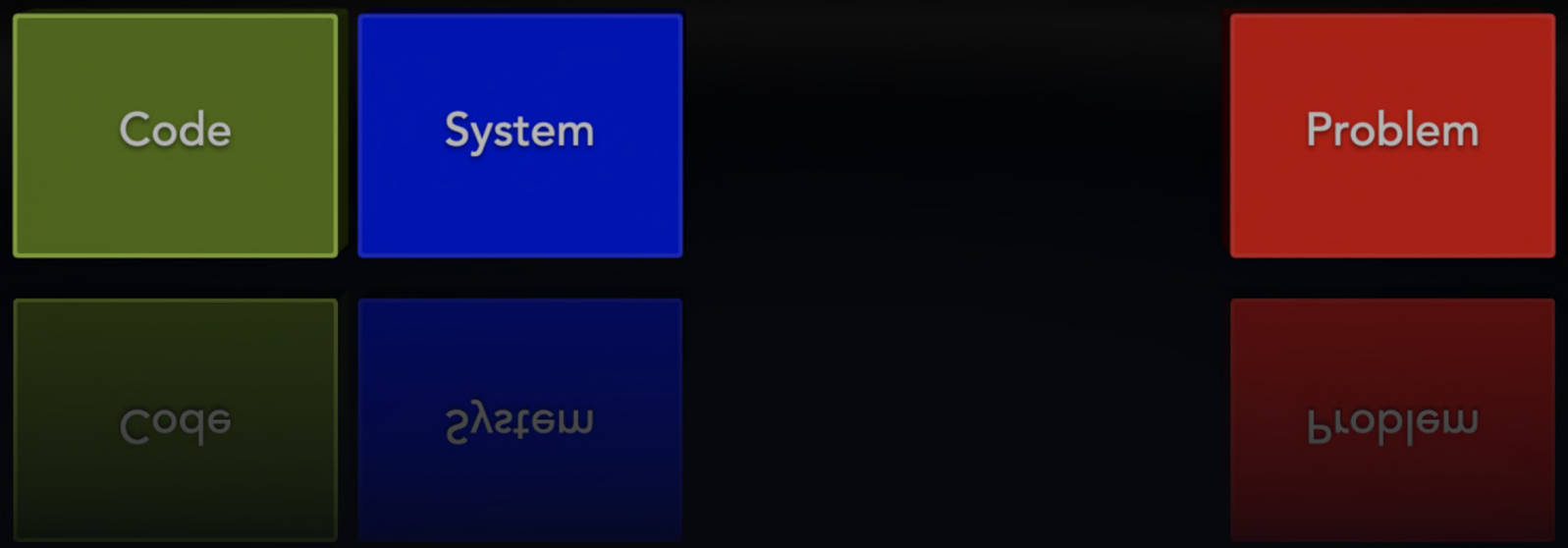
Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation
- OOP
- Ruby



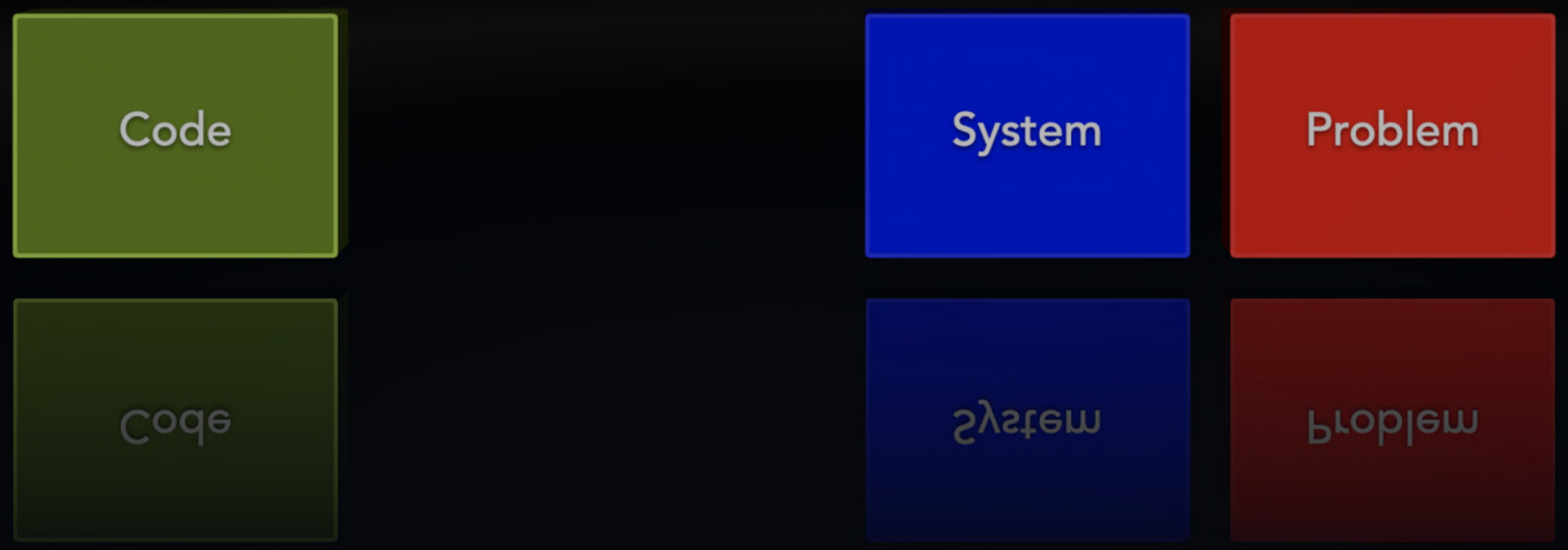
Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation
- OOP
- Ruby
- FP



Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation
- OOP
- Ruby
- FP



Static Program Text vs. Dynamic Execution

- FORTRAN
- Non-Computation
- OOP
- Ruby
- FP
- Objective-Smalltalk

Solution Ideas

- Alan Kay: figure out the metasystem
- Unicon: connectors ←→ programming language metaystem
- ST: decouple class hiearchy from implementation hierarchy
- → metasystem: conceptual class hierarchy of connectors

Objective-SmallTalk

Approach

Objective-SmallTalk: Approach

- Implement different architectural styles
- Add language support
- Inform the metasystem

Architectural Styles

- Objects and Messages
- Pipes and Filters
- Implicit Invocation
- (In-process) REST

Objects and Messages

- Objective-C compatible semantics
- Interpreted and native-compiled
- "C" using type annotations
- Higher Order Messaging
- Framework-oriented development
- Full platform integration

Objects and Messages

- Objective-C compatible semantics
- Interpreted and native-compiled
- "C" using type annotations
- Higher Order Messaging
- Framework-oriented development
- Full platform integration

Pipes and Filters

- Polymorphic Write Streams (DLS '19)
- #writeObject:anObject
- Triple Dispatch + Message chaining
- Asynchrony-agnostic
- Streaming / de-materialized objects
- Serialisation, PDF/PS (Squeak), Wunderlist, MS , To Do
- Outlook: filters generalise methods?

Implicit Invocation

- Notification Protocols
- Class adopts a Protocol to register for notifications

In-Process REST

- What real large-scale networks use
- Polymorphic Identifiers
- Stores
- Storage Combinators
- Used in a number of applications

Polymorphic Identifiers

- All identifiers are URIs
- var:hello := 'World!
- file:{env:HOME}/Downloads/site := http://objective.st
- slider setValueHolder: ref:var:celsius

Stores

- PIs are evaluated via stores
- Like in-process REST servers
- Or composable dictionaries
- var:hello <-> (schemes at:'var') at:'hello'

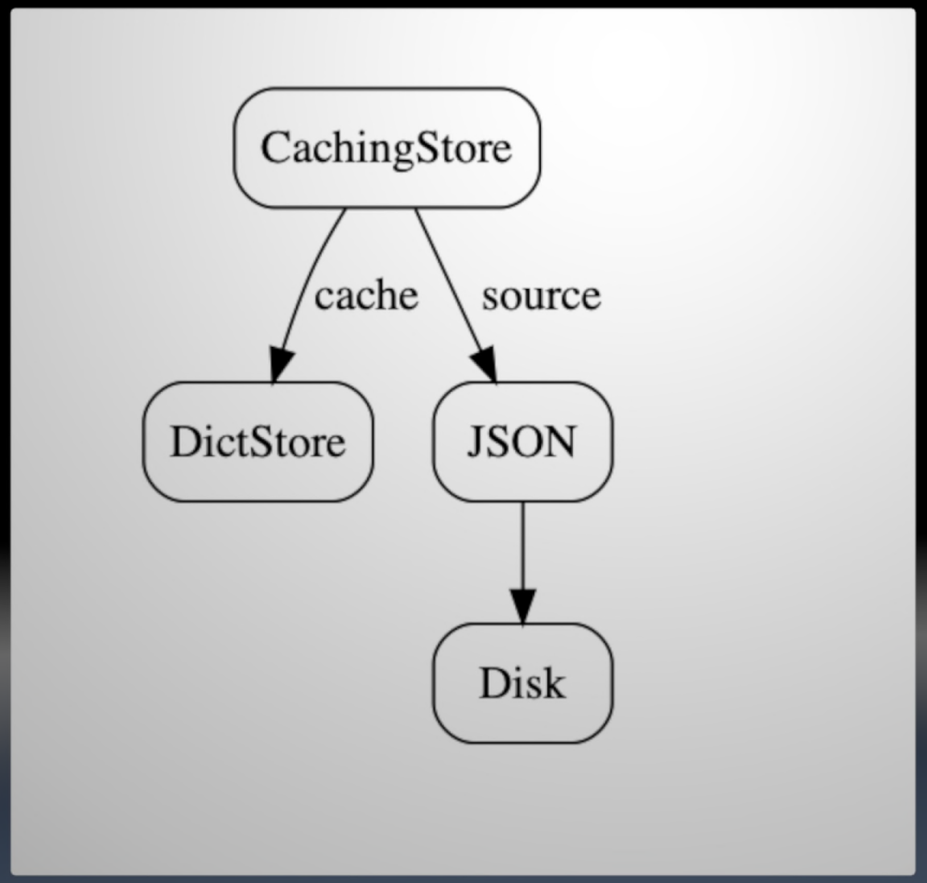
Example Stores

- Local variables, environment variables
- Filesystems, HTTP
- Databases: SQL/SQLite
- Other applications (via Apple Events)
- etc.

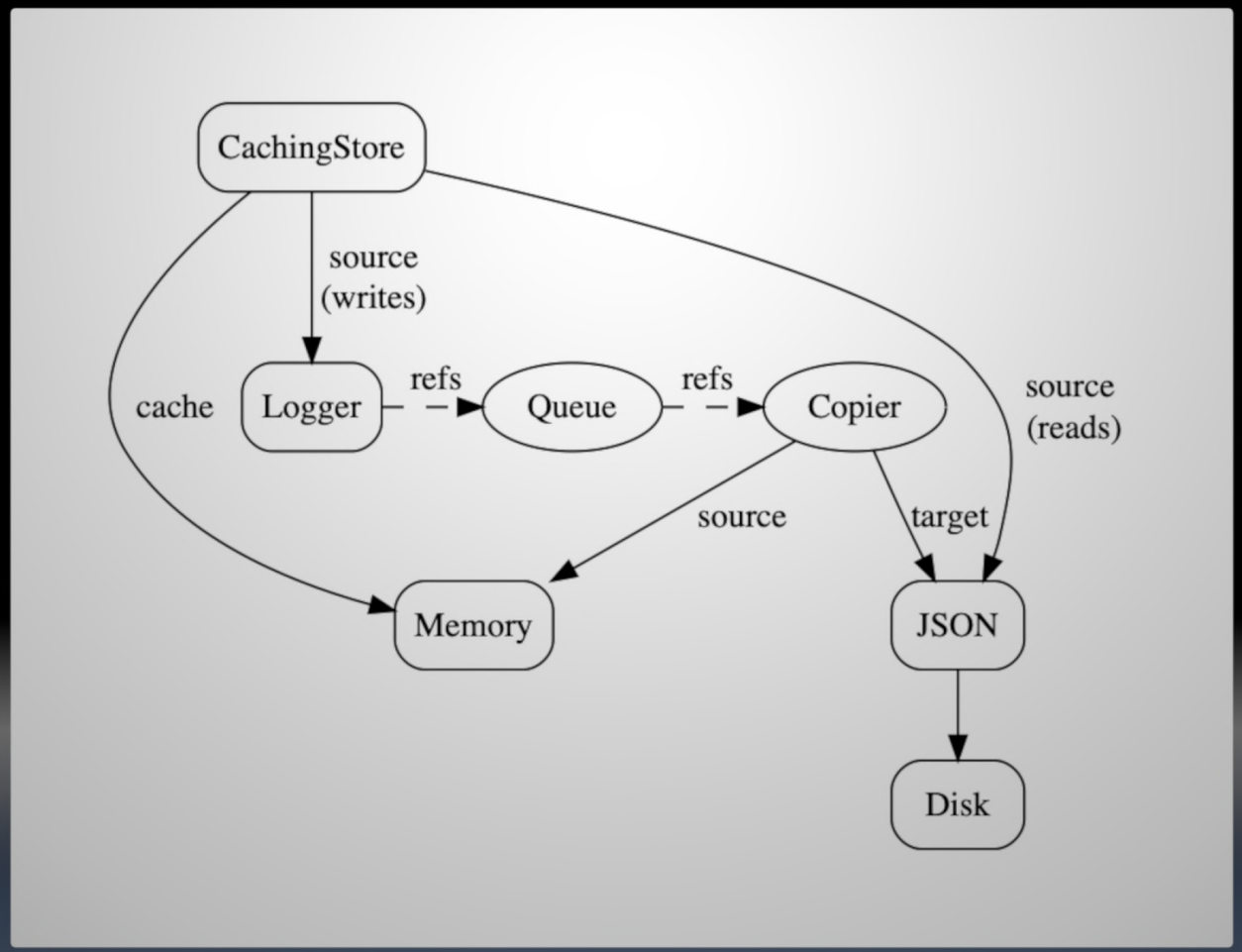
Storage Combinators

- Onward! '19
- Combinator exposes + consumes REST interfaces
- Uniform interface (REST) enables pluggability
- Narrow, semantically tight interface enables intermediaries
- 10x productivity/code improvments

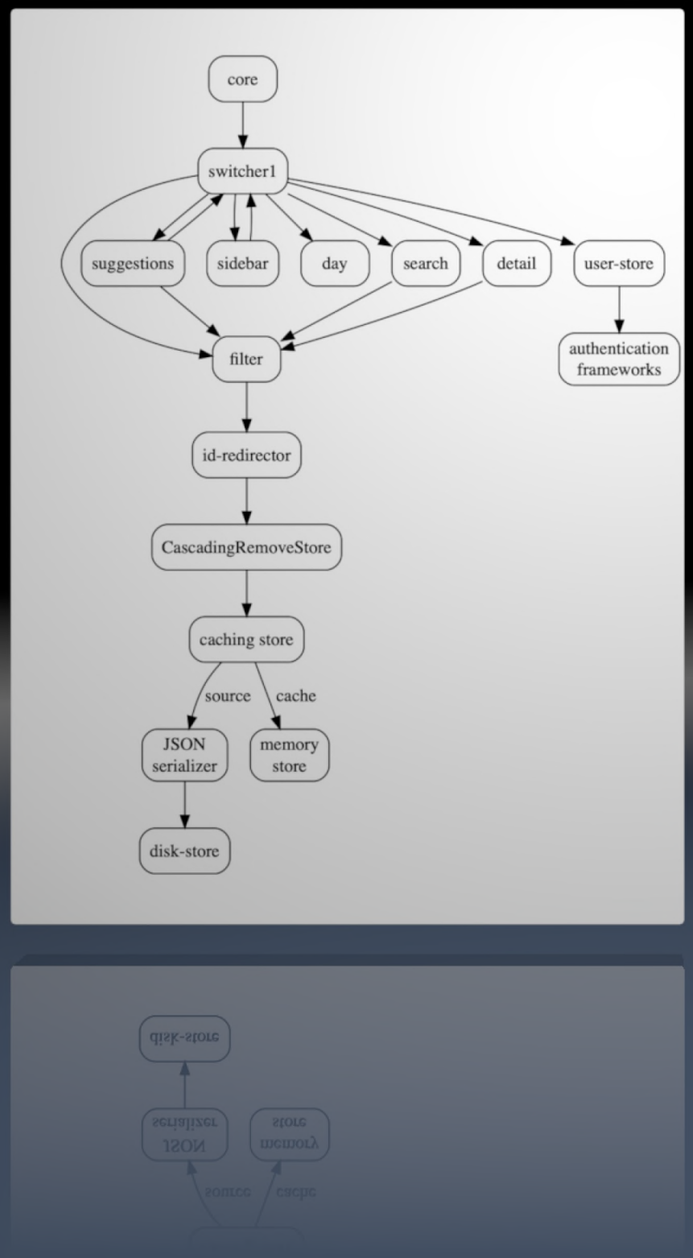
Simple Composed Store



Async Writer



To Do Store Hierarchy



Objective-SmallTalk

Language Support

Language support

- Class, method, protocol syntax
- Polymorphic Identifiers
- Composition "operator" →
- Permanent Assignment |=
- Stores and Filters as class templates
- Property Paths

Objective-SmallTalk

Demo

Objective-SmallTalk

Metasystem

Metasystem

- Apply/Eval → Connect/Run
- Class is subclass of Component
- Message, Interface etc. subclasses of Connector
- Components have ports, connectors roles

Objective-SmallTalk

Outlook

Outlook

- Port Stores and Polymorphic Write Streams
- Documentation / Sample Code
- Improve native compiler
- Tooling (Debugger)
- You! (http://objective.st)

Marcel Weiher (@mpweiher)

Q&A http://objective.st