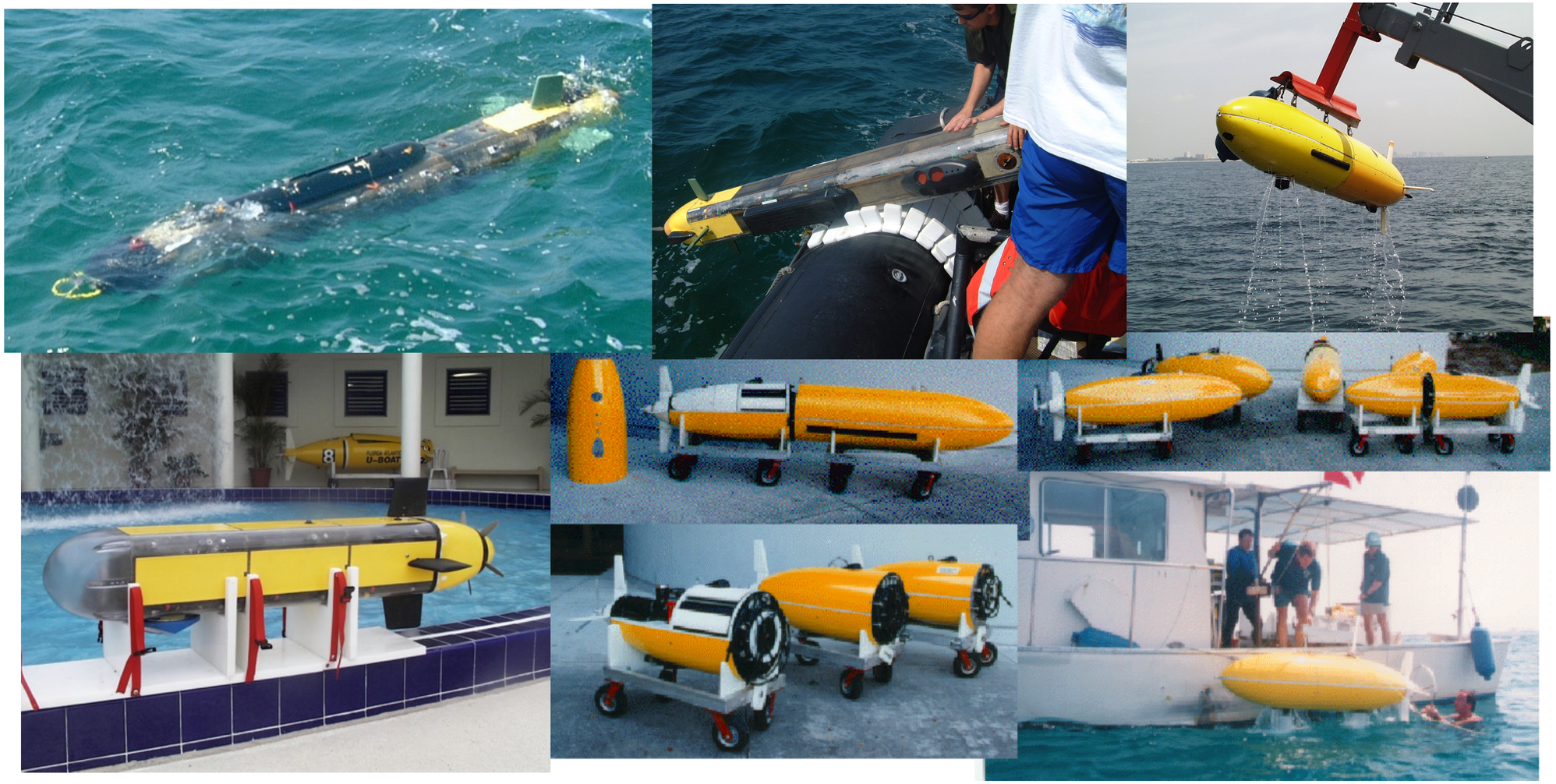
Asynchronous Micro-Services in Python with Ioflo and Bottle

OpenWest 2016

Samuel M Smith Ph.D.
Founder
Prosapien LLC
sam@prosapien.com

Autonomous Underwater Vehicles



Asynchronous Flow-Based Programming Framework

github.com/ioflo/ioflo
ioflo.com
sam@ioflo.com
github.com/ioflo/ioserve
github.com/ioflo/ioflo_manuals

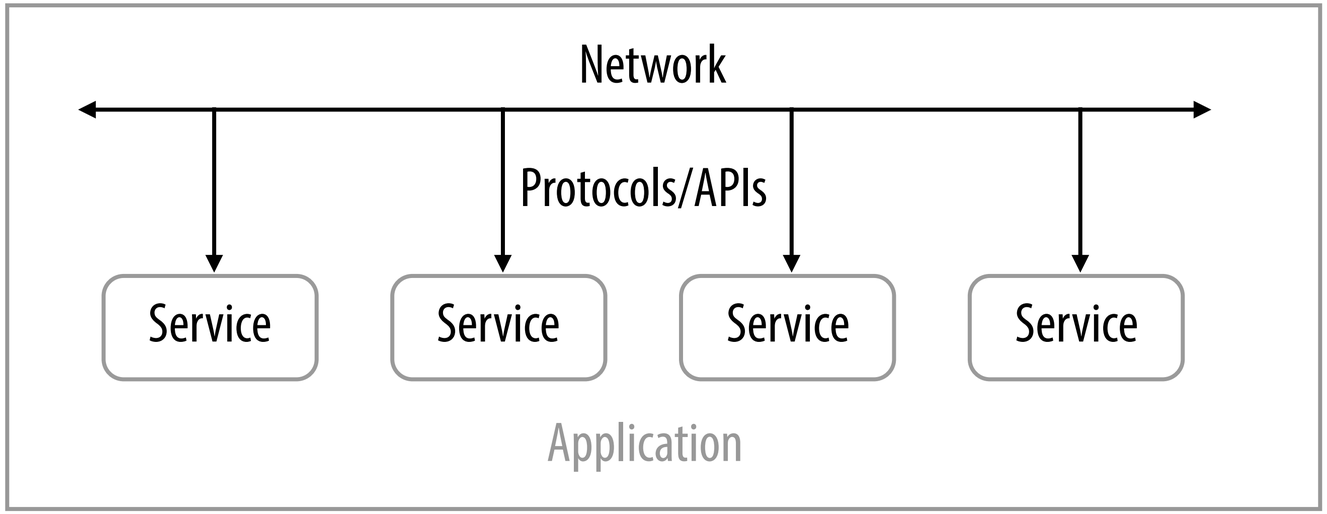
WSGI Micro Web Framework
Ideal for ReST/Micro-Services

http://bottlepy.org
https://github.com/bottlepy/bottle

like Flask only lighter

What is a Micro-Service Application?

Application composed of small single purpose services connected via lightweight protocols



Enables internal language agnosticism but external shared protocol (http/ReST/json)

Increased dev team autonomy and agility

Easier scaling

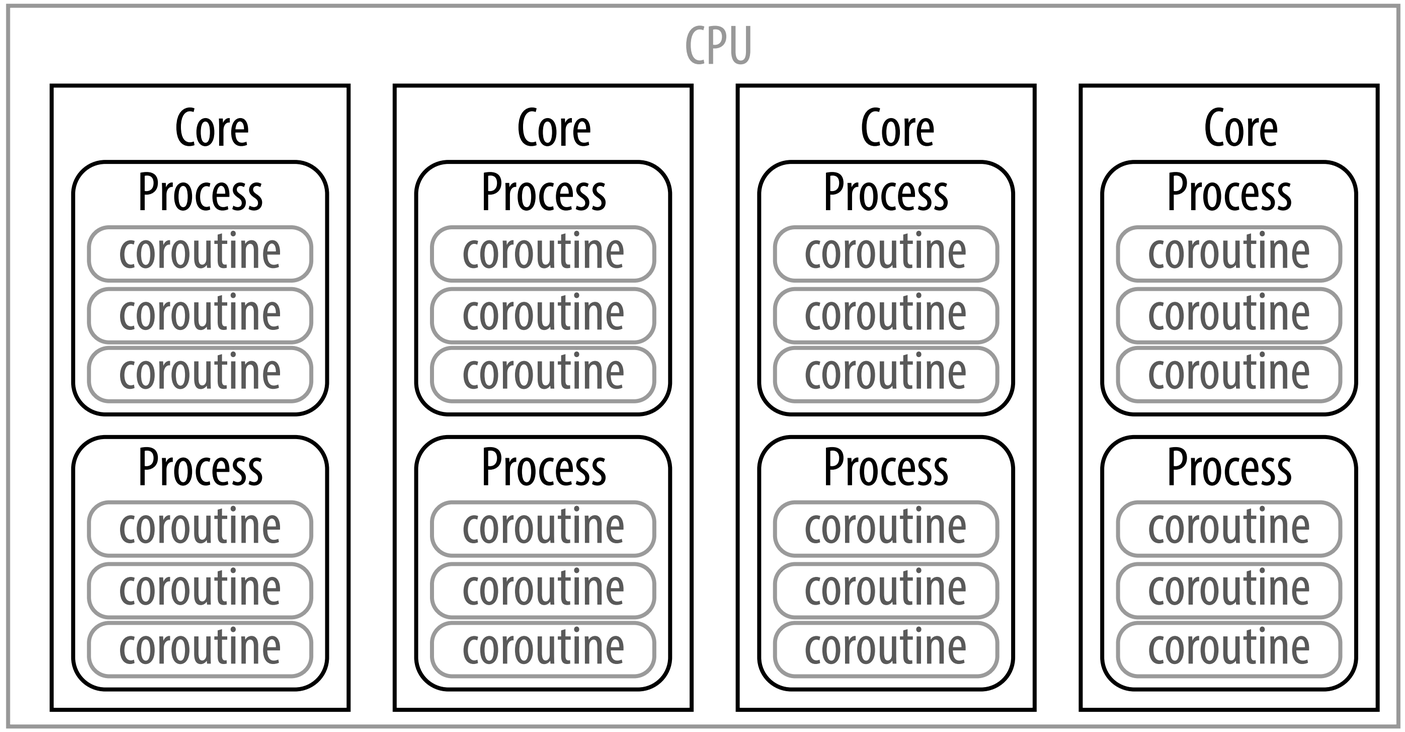
Decreased coupling Enhanced cohesion

Producer/Consumer architecture

"High Replacement Independence"

What is Asynchronous Processing?

Cooperative logical concurrency within a single process/thread plus nonblocking IO.



Advantages:

Performant, 1 process per core, minimizes context switch overhead >15%

Simple, no resources contentions, locks, semaphores, everything is atomic

Scalable, granularity matches logical concurrency not arbitrary host/process boundaries

Asynchronous Processing Approaches

Greenlets, Micro-threads, gevent, stackless python

Emulates conventional multi-thread/process model, join calculus, sleepy, continuous logic flow,

Event Loop Callbacks, Twisted, Node.js

Medusa/Reactor pattern, callback hell, discontinuous logic flow

Deferred, Python 3.5 asyncio futures, promises

Thread pool like scheduler, discontinuous logic flow

Yielded Delegated, generators/coroutines, Python 3.5 asyncio coroutines

Yielding scheduler, delegating coroutines, yieldy, continuous logic flow

Nested hybrid coroutines, Ioflo

Nested contextual yielding/delegating schedulers, continuous logic flow

Flow-Based Programming (FBP)

- Originated in 1970's by J.P. Morrison, contemporary with OOP & FP
- Relatively unknown but a lot of interest recently
- Think general-purpose data-flow programming
- Use it via a programming style, pattern, paradigm, or framework, not a language
- FBP is a simplifying unifying paradigm
- Distributed concurrent applications benefit most from FBP
- An FBP architecture may still be really useful even when not using an FBP Framework
- An FBP mindset may provide unique solution insights even when using OOP or FP

Flow-Based Programming Frameworks

Javascript

NoFlo

Flux

constructables.es

Python

ioflo

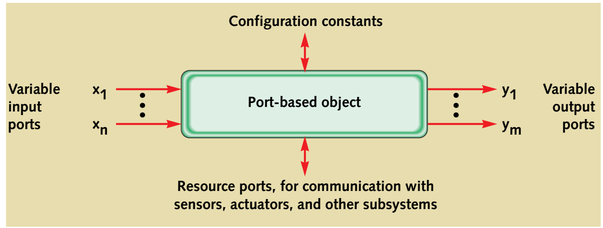
Pypes

PyF

Other

Expecco

PBO



More or less FBP

Java VM

Apache STORM

Spark

IBM InfoSphere Streams

Microsoft Azure Event Hubs

Python

ROS

Open Source Robotics Foundation

ERLANG

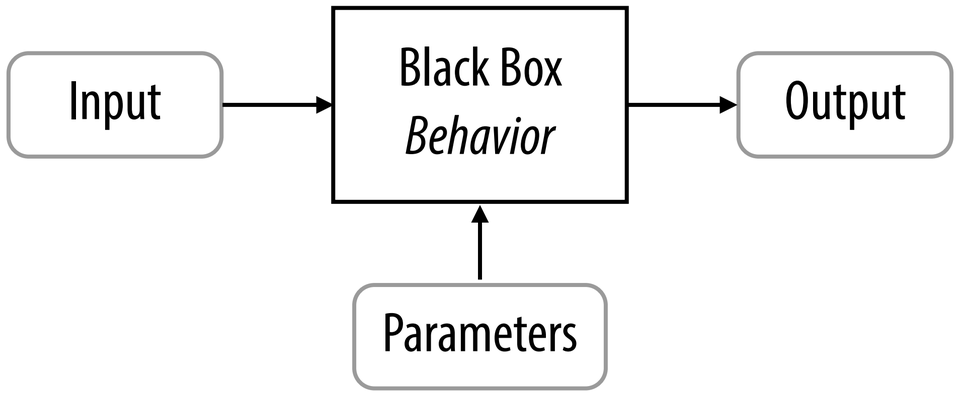
Other

MatLab Simulink

NATIONAL INSTRUMENTS LabVIEW

What is Flow-Based Programming (FBP)

A behavior transforms its input(s) into its output(s)

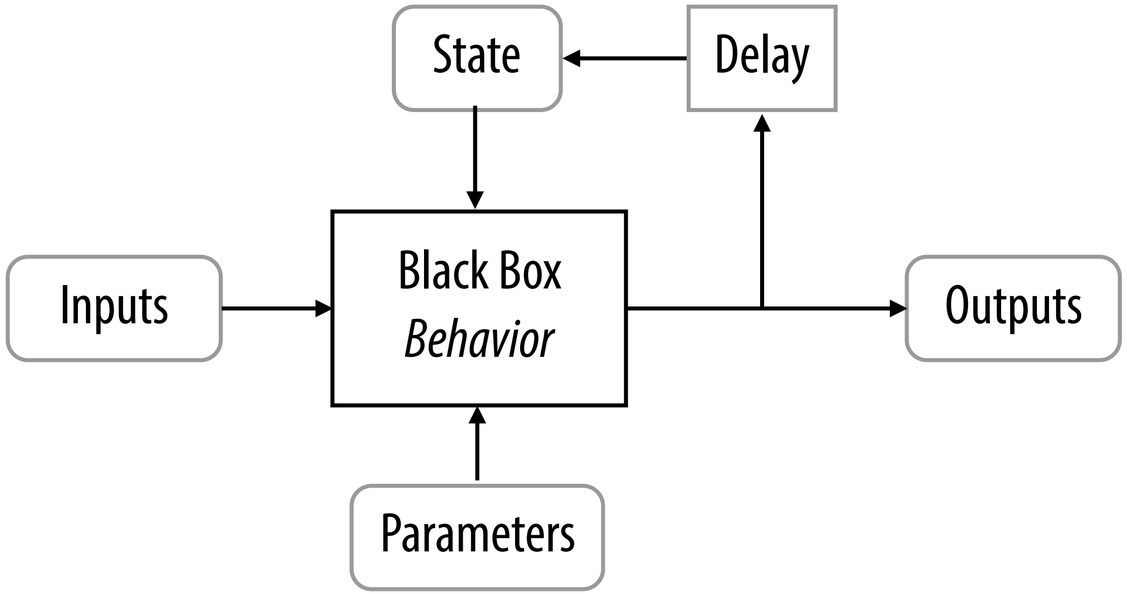


Inputs may also be parameters that modify the transformation

A behavior transforms its inputs governed by its parameters into its outputs

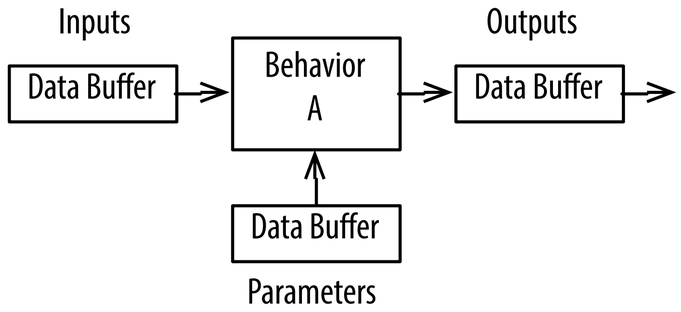
What is Flow-Based Programming (FBP)

Outputs may be fed back as state to modify the transformation

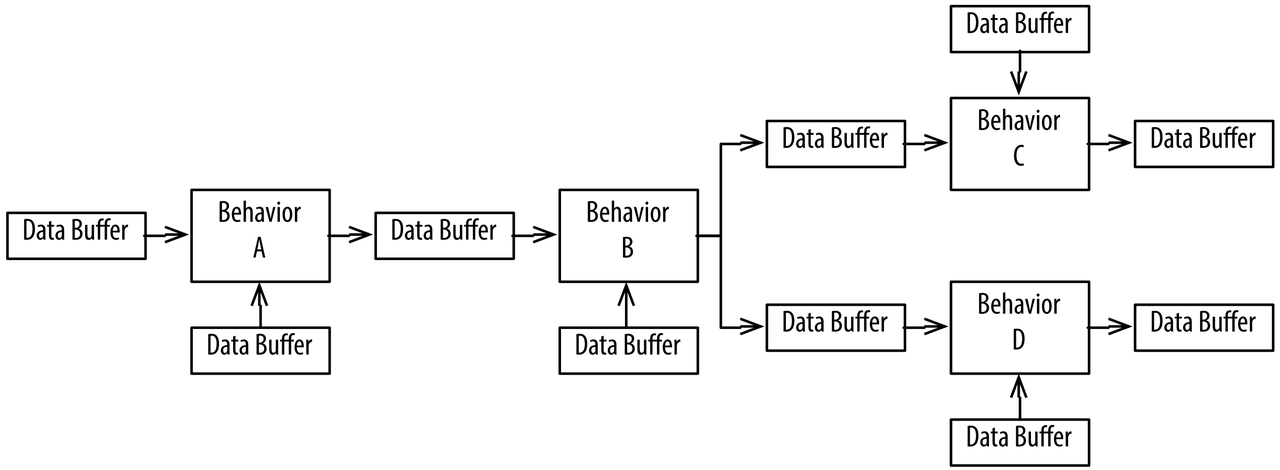


A behavior transforms its inputs and past outputs as governed by its parameters into its outputs

What is Flow-Based Programming (FBP)

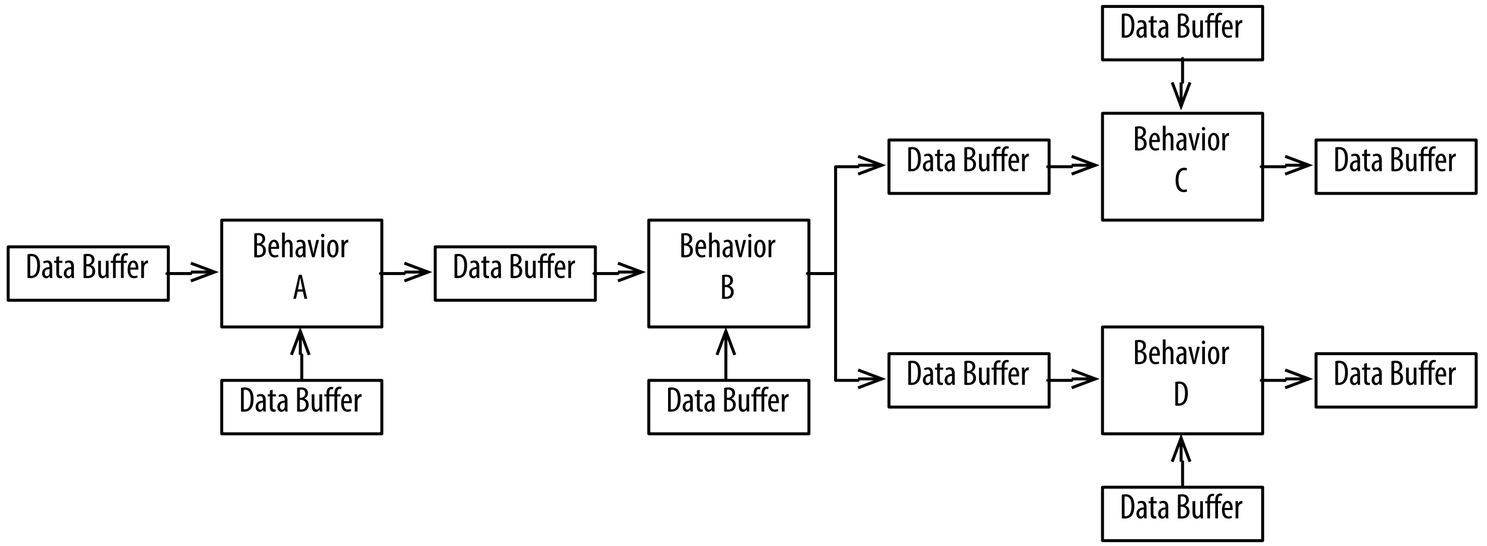


Behaviors can be reconnected endlessly in different networks without any internal changes



FBP applications (programs) are networks of asynchronous behaviors
which exchange data across externally defined connections

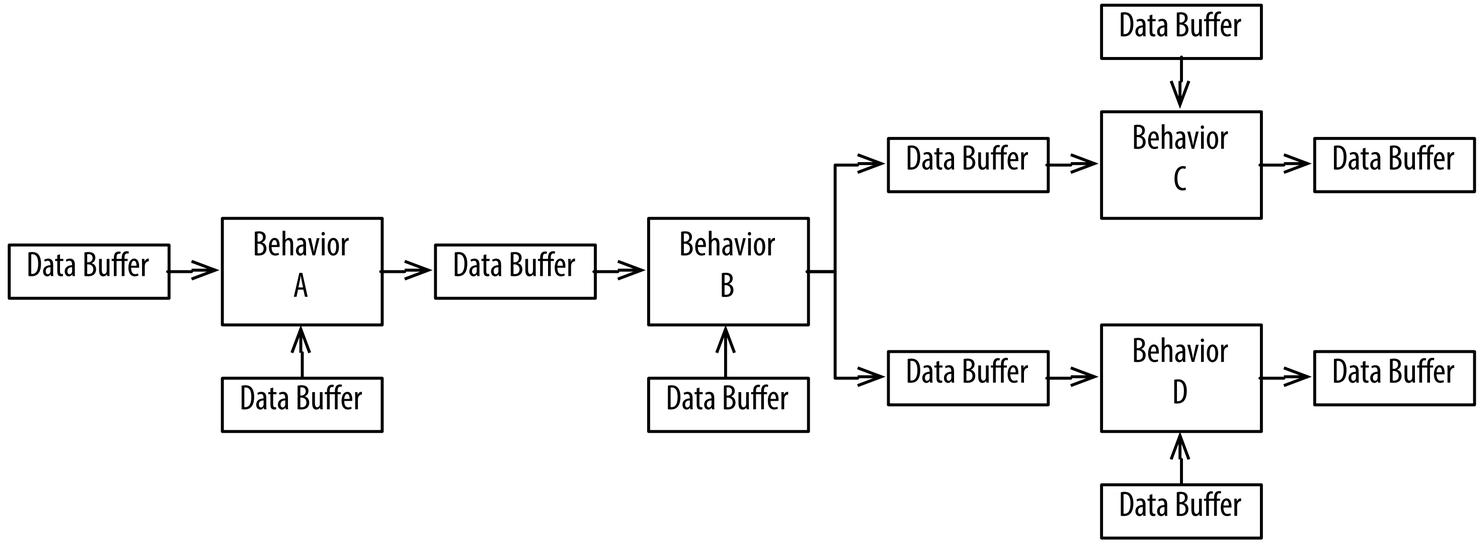
What is Flow-Based Programming (FBP)



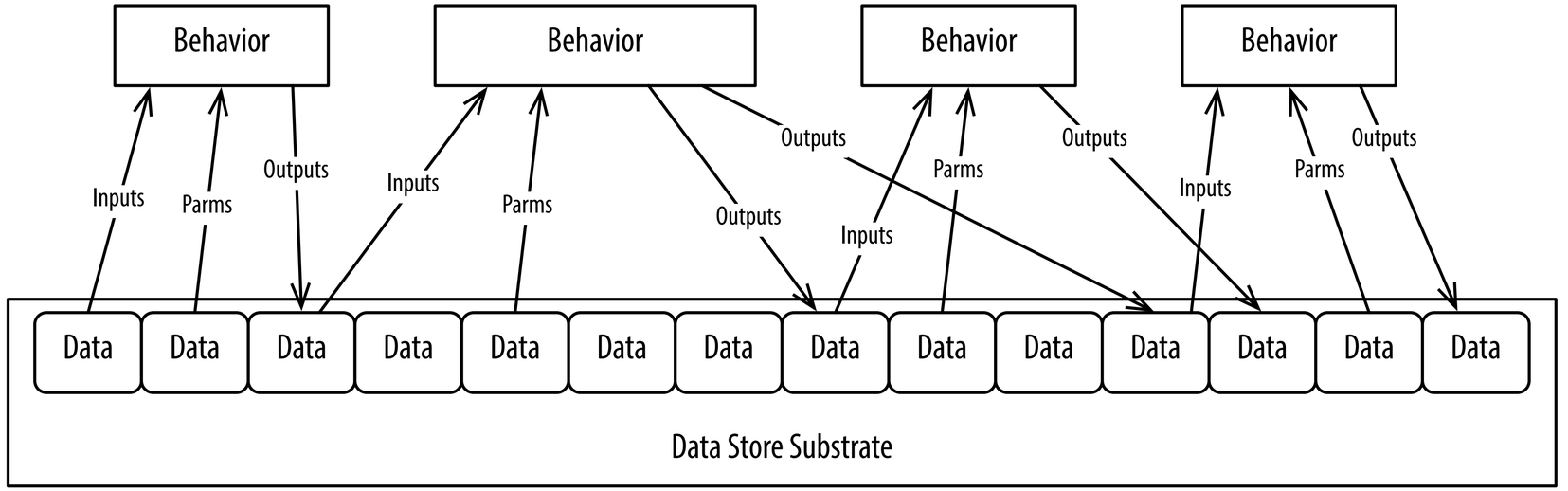
- Programming is the declarative composition of networks of behaviors
- FBP = Data Flow Oriented + Component Oriented
- FBP = FP-ish + OOP-ish

FBP Variant with DataStore Substrate

From this



To this



- Adds configurability, observability, traceability, replayability

Replacement Independence

Behaviors (components) can be externally connected without any internal changes.

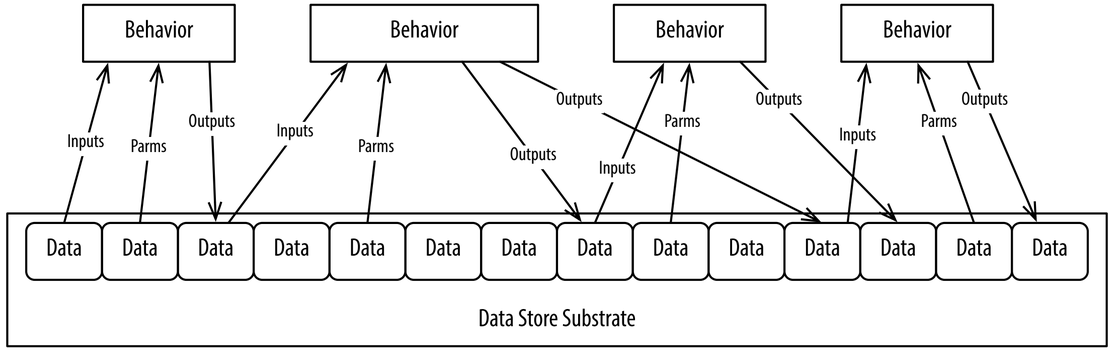
Composition of complex networks/graphs is conceptually simple

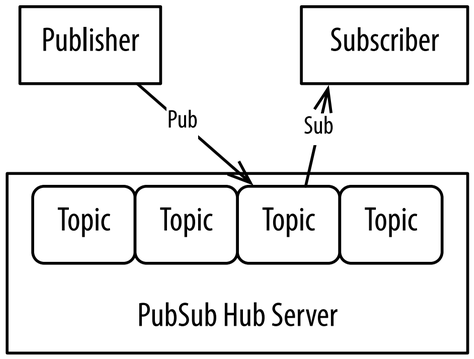
Because partitioning occurs intra-process/intra-host instead of inter-process/inter-host, distribution of behaviors across processor resources does not change behavior internals

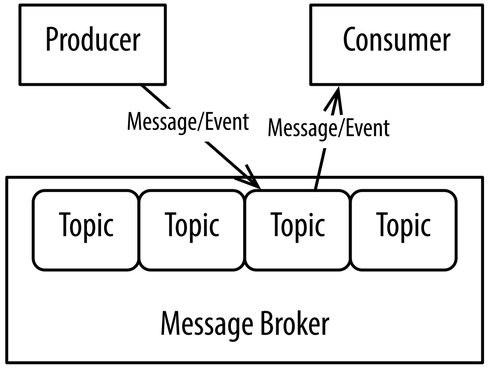
Replacement Independence = Dependency Minimization

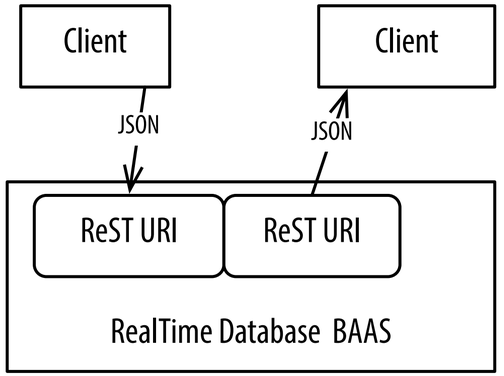
Replacement Independence = Flexibility, Robustness, Mobility

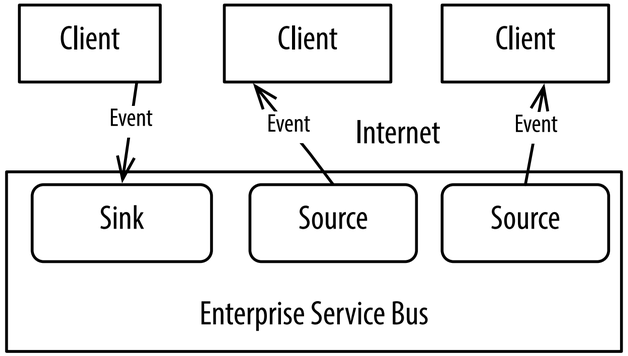
Meta Architecture Patterns

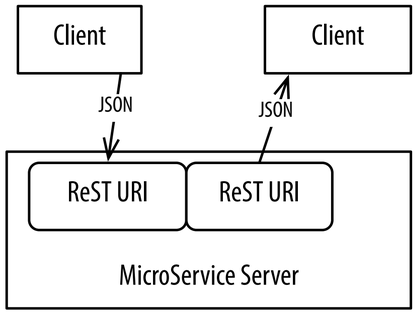


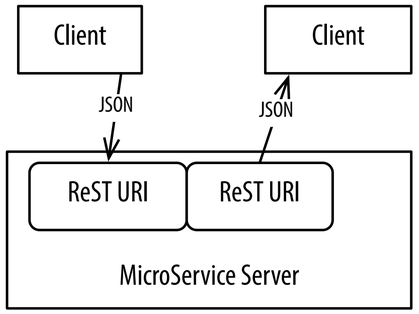


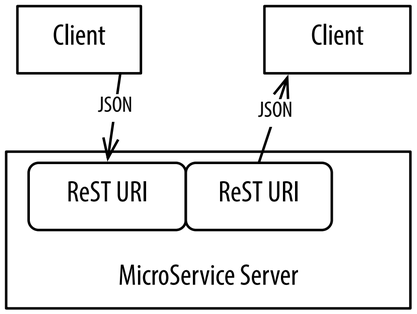


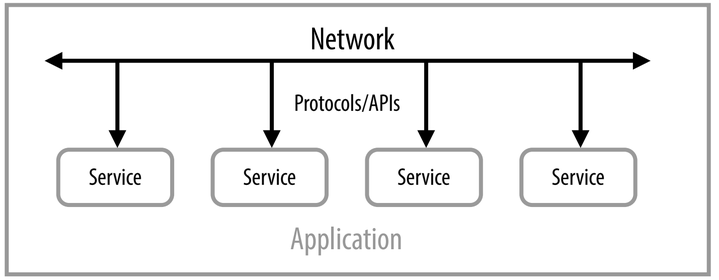








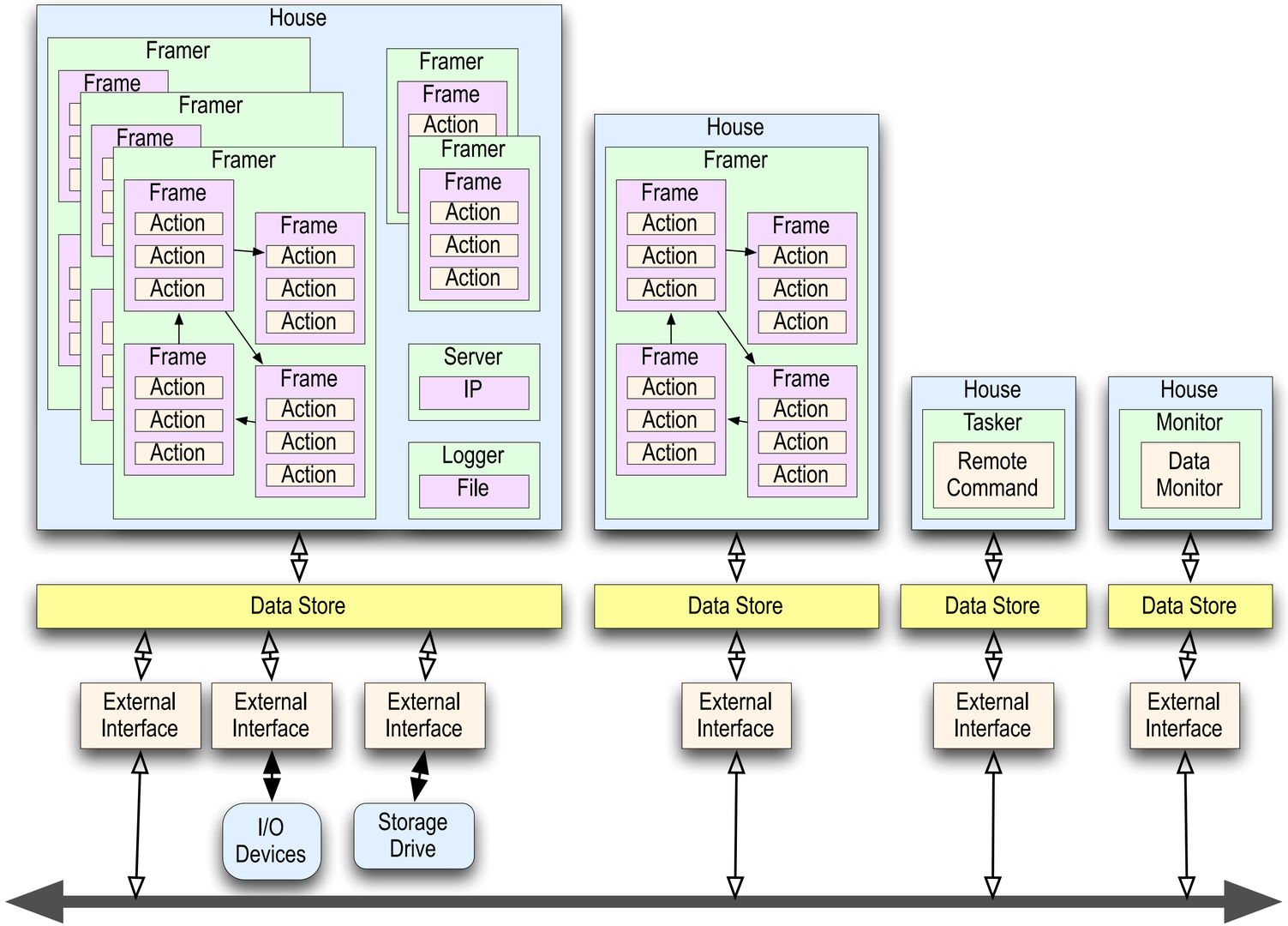




ioflo

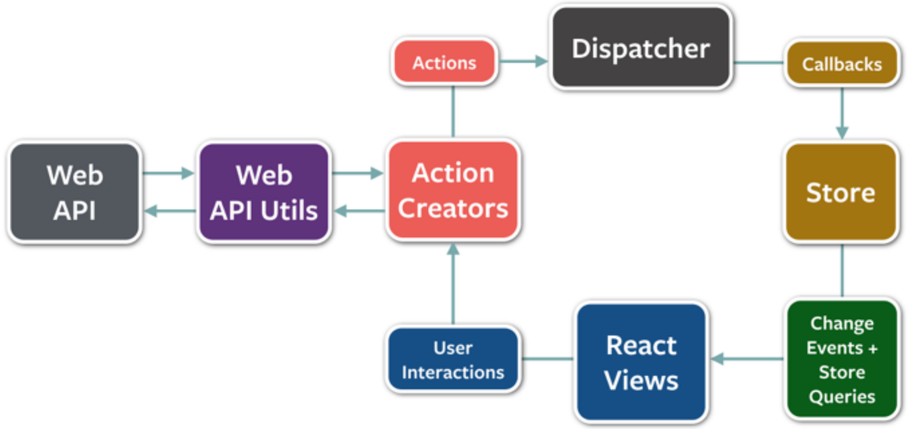
FloScript

Big Picture



What is a Flow-Based Programming Framework

- Analogous to web application framework.
- Provides syntactic sugar and/or graphical editors to facilitate the composition and scheduling of behavior networks/graphs



- Begins where many other distributed application architectures are striving to end up.

```
house box1
   framer vehiclesim be active first vehicle_run
      frame vehicle_run
         do simulator motion uuv

   framer mission be active first northleg
      frame northleg
         set elapsed to 20.0
         set heading to 0.0
         set depth to 5.0
         set speed to 2.5
         go next if elapsed >= goal

      frame eastleg
         set heading to 90.0
         go next if elapsed >= goal

      frame southleg
         set heading to 180.0
         go next if elapsed >= goal

      frame westleg
         set heading to 270.0
         go next if elapsed >= goal

      frame mission_stop
         bid stop vehiclesim
         bid stop autopilot
         bid stop me

   framer autopilot be active first autopilot_run
      frame autopilot_run
         do controller pid speed
         do controller pid heading
         do controller pid depth
         do controller pid pitch
```

DSL

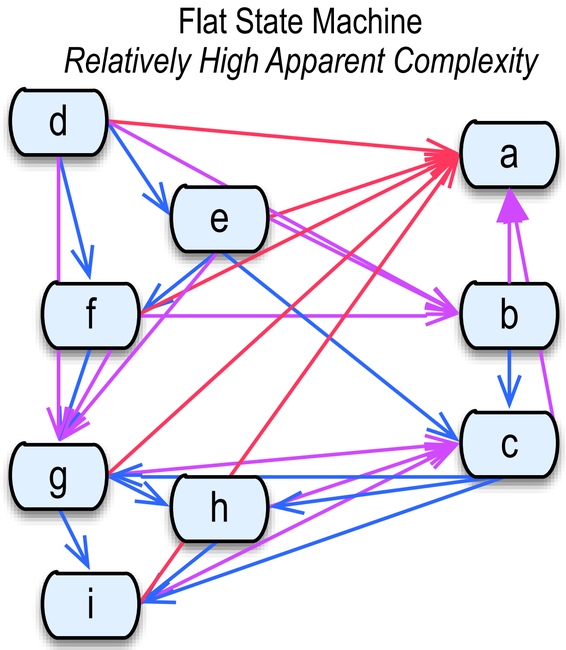
Marshaling, Configuration, Logic, Flow, Execution, and Scheduling

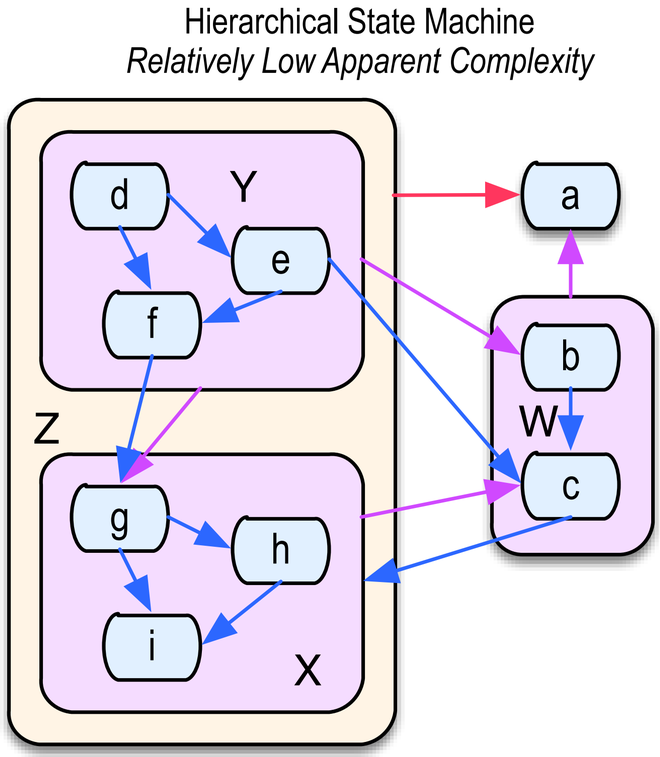
all in one place

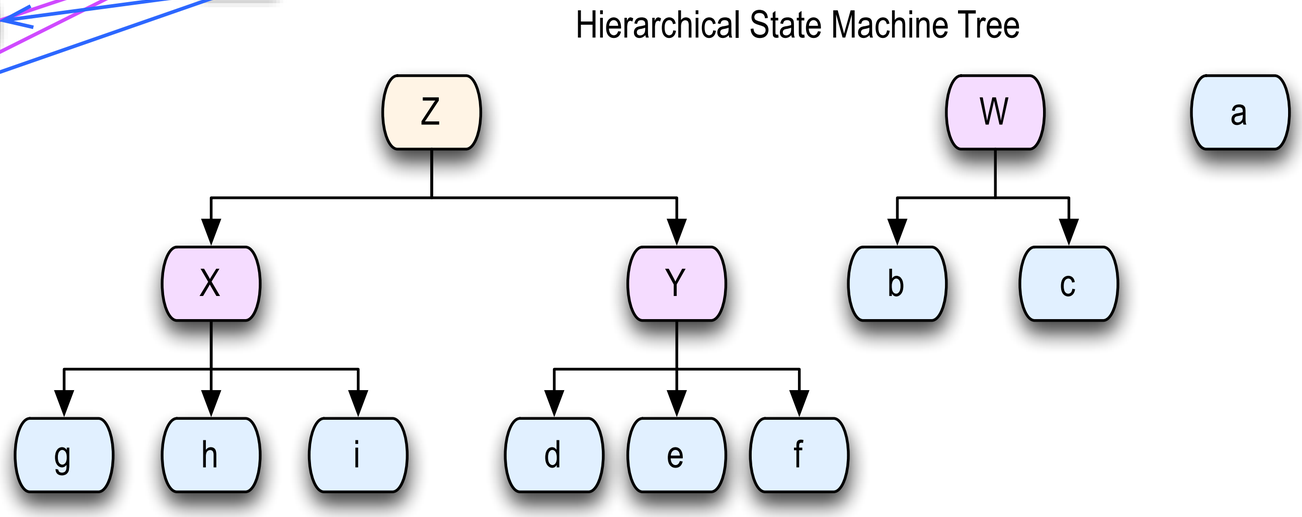
Convenient Declarative Syntax (*3rd Generation*)

- Unified Scheduling and Execution
- Integrated Pub/Sub, Messaging,
- Observable/Traceable
- Nested Concurrent Contexts

Hierarchical State

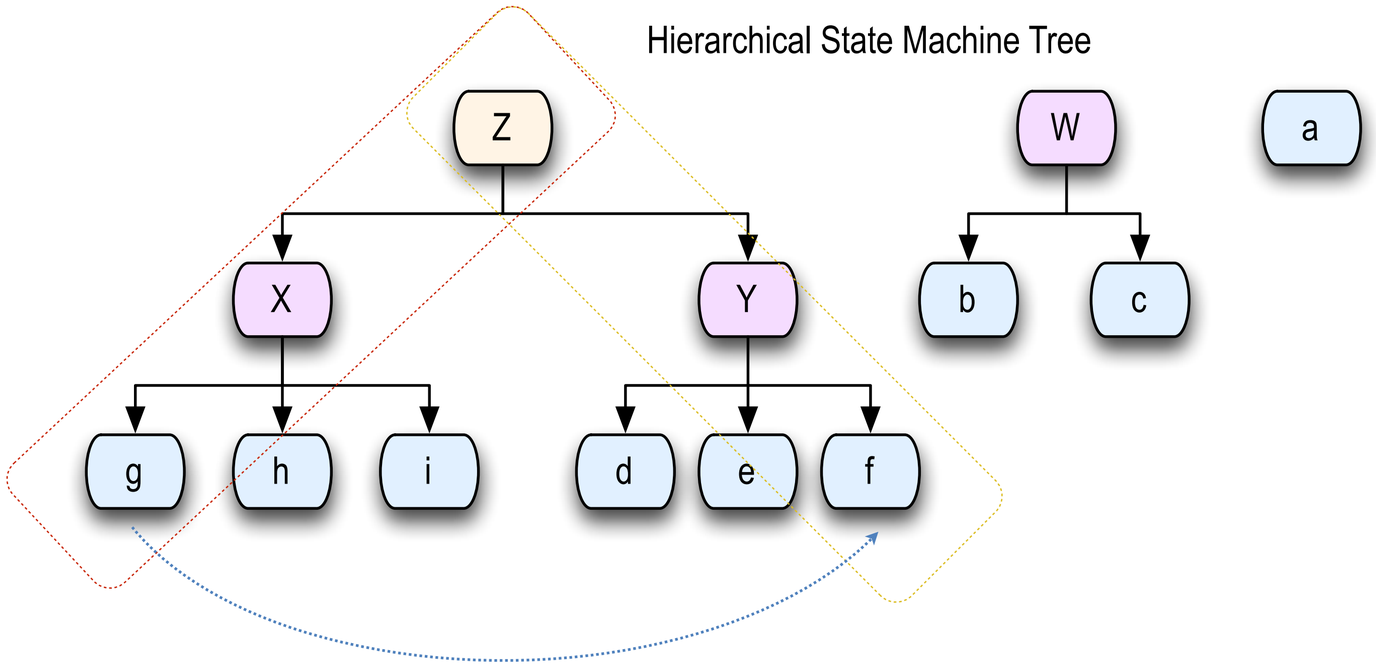






Transition = Change in Framing Context

```
go foobar if elapsed >= goal
```

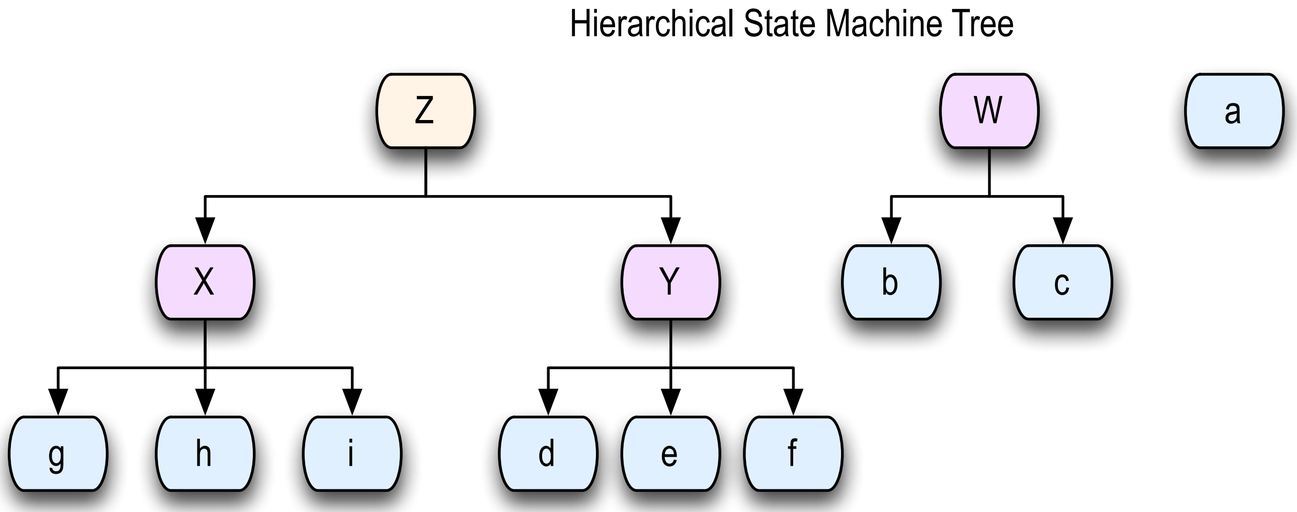


FloScript

Dichotomous Priorities

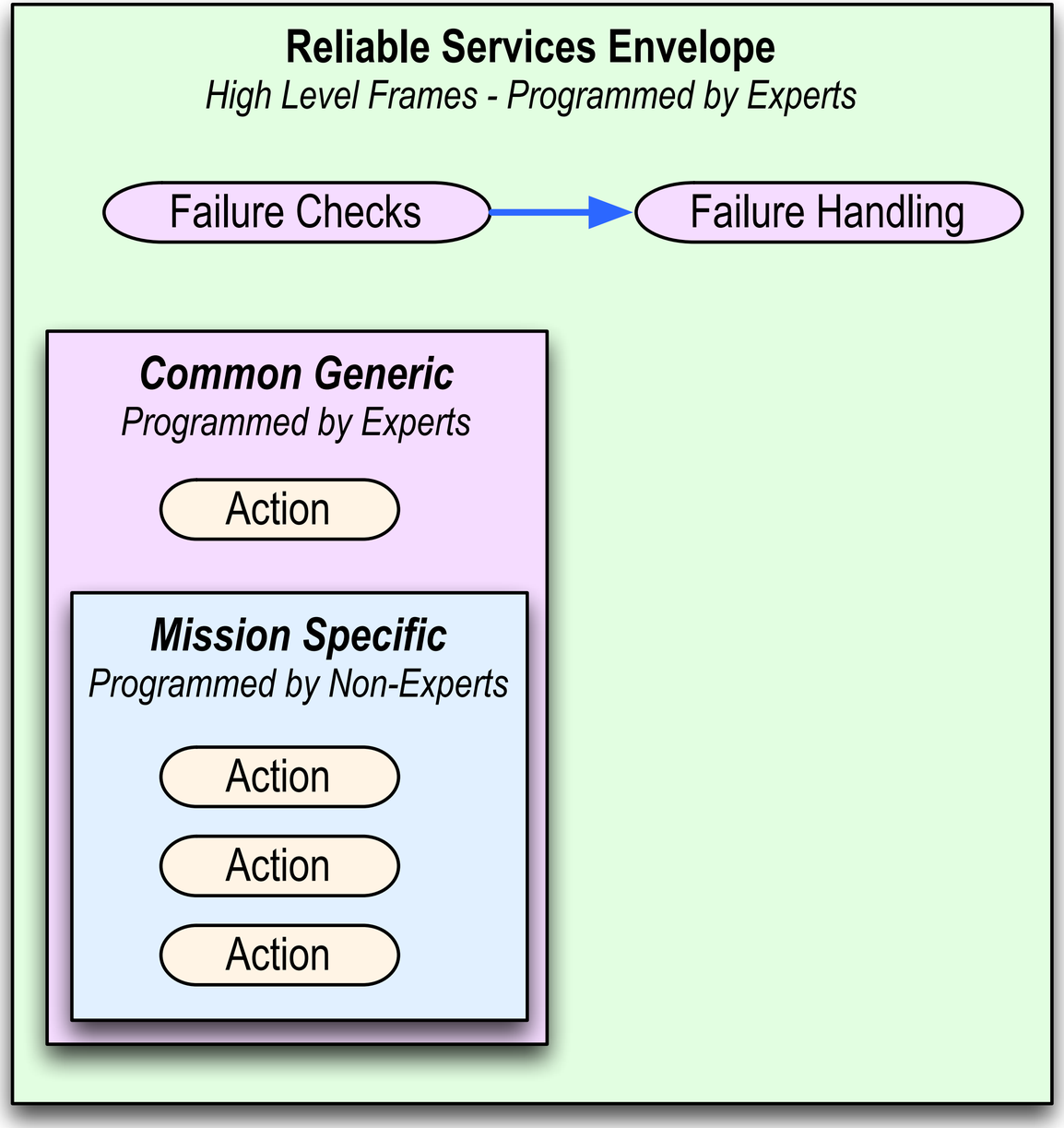
FloScript

Transitions
- Change Context, Propriety
- Priority is Top-Down



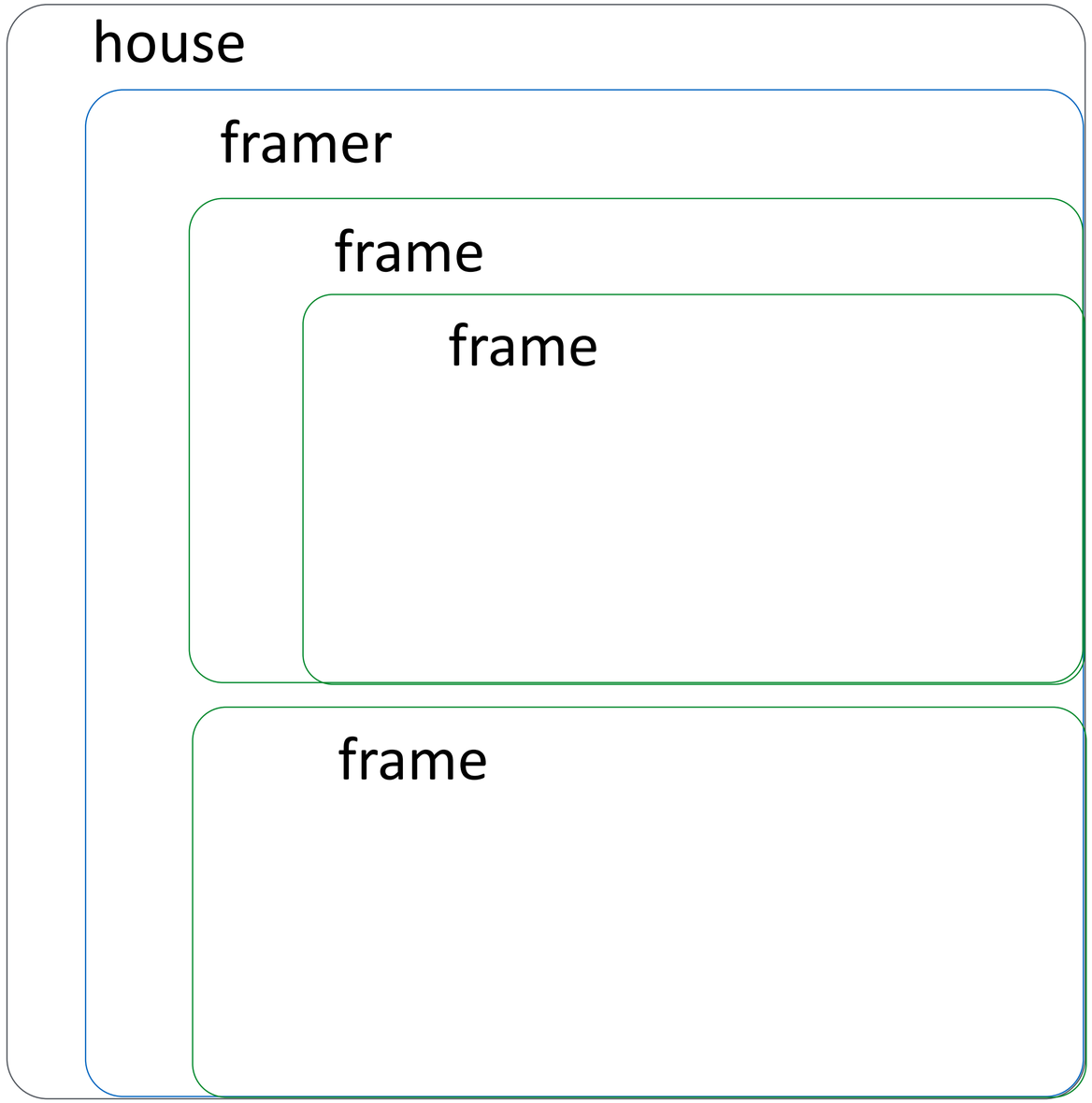
Actions
- Determine Contextual Behavior, Specificity
- Priority is Bottom-Up

Reliability

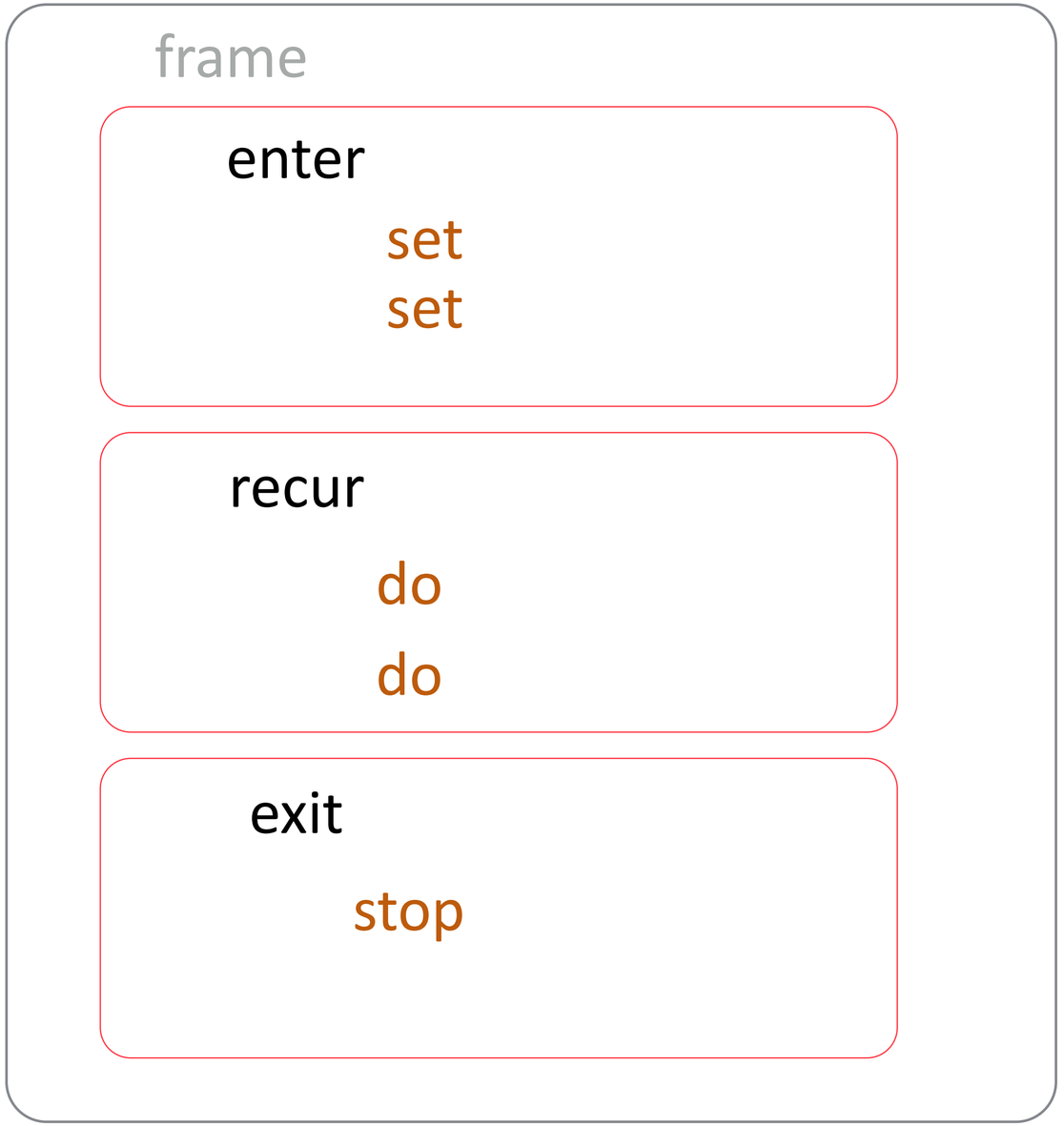


Contexts:

Frame of Reference "*Framing*"



Action Execution "*Actioning*"



FloScript

Incremental Encapsulation

Framing *concurrency*

Framers

Auxiliary Framers

Master/Slave Framers

Nested Frames

FloScript

Incremental Encapsulation

Actioning *components*

FloScript Verbs

Ioflo library Python Actions/Behaviors for do verb objects

Custom Python coded do verb objects

Custom coded Python C Extensions for do verb objects

FloScript Line Oriented, Contextual, Declarative Configuration Language

not block oriented

Declaration Sentence: `verb [object] [prepositional-phrases]`

```
done
frame build in setup
set speed to 5
framer create at 0.5 be active first build
```

`[adjectives …] kind`

Verb Object: `do salt eventer`

Creating a Do verb object/behavior "deed" with decorator

```
def doify(name,
          base=None,
          registry=None,
          parametric=None,
          inits=None,
          ioinits=None,
          parms=None):
    """ Parametrized decorator function that converts the decorated function
        into an Actor sub class with .action method and with class name name
        and registers the new subclass in the registry under name.
        If base is provided then register as subclass of base.
        Default base is Doer

        The parameters registry, parametric, inits, ioinits, and parms if provided,
        are used to create the class attributes for the new subclass
    """
```

Example

```
@doify('IoserveServerClose', ioinits=odict(valet=""))
def ioserveServerClose(self, **kwa):
    """
    Close server in valet

    Ioinit attributes:
        valet is a Valet instance

    Context: exit

    Example:
        do ioserve server close at exit
    """
    if self.valet.value:
        self.valet.value.servant.close()

        console.concise("Closed server '{0}' at '{1}'\n".format(
                               self.valet.name,
                               self.valet.value.servant.eha))
```

Valet WSGI HTTP Server

```
class Valet(object):
    """
    Valet WSGI Server Class
    """
    Timeout = 5.0  # default server connection timeout

    def __init__(self,
                 app=None,
                 reqs=None,
                 reps=None,
                 servant=None,
                 store=None,
                 name='',
                 bufsize=8096,
                 wlog=None,
                 ha=None,
                 host=u'',
                 port=None,
                 eha=None,
                 scheme=u'',
                 timeout=None,
                 **kwa):
        """
        Initialization method for instance.
        app = wsgi application callable
        reqs = odict of Requestant instances keyed by ca
        reps = odict of running Wsgi Responder instances keyed by ca
        servant = instance of Server or ServerTls or None
        store = Datastore for timers

        kwa needed to pass additional parameters to servant

        if servantinstances are not provided (None)
        some or all of these parameters will be used for initialization

        name = user friendly name for servant
        bufsize = buffer size
        wlog = WireLog instance if any
        ha = host address duple (host, port) for local servant listen socket
        host = host address for local servant listen socket, '' means any interface on host
        port = socket port for local servant listen socket
        eha = external destination address for incoming connections used in TLS
        scheme = http scheme u'http' or u'https' or empty

        """
```

Patron HTTP Client

```
lass Patron(object):
    """
    Patron class nonblocking HTTP client connection manager
    """
    def __init__(self,
                 connector=None,
                 requester=None,
                 respondent=None,
                 store=None,
                 name='',
                 uid=0,
                 bufsize=8096,
                 wlog=None,
                 hostname='127.0.0.1',
                 port=None,
                 scheme=u'',
                 method=u'GET',  # unicode
                 path=u'/',  # unicode
                 headers=None,
                 qargs=None,
                 fragment=u'',
                 body=b'',
                 data=None,
                 fargs=None,
                 msg=None,
                 dictable=None,
                 events=None,
                 requests=None,
                 responses=None,
                 redirectable=True,
                 redirects=None,
                 **kwa):
```

```
"""
        Initialization method for instance.
        kwa needed to pass other init parameters to connector

        connector = instance of Outgoer or OutgoerTls or None
        requester = instance of Requester or None
        respondent = instance of Respondent or None

        if either of requester, respondent instances are not provided (None)
        some or all of these parameters will be used for initialization

        name = user friendly name for connection
        uid = unique identifier for connection
        bufsize = buffer size
        wlog = WireLog instance if any
        host = host address or hostname of remote server
        port = socket port of remote server
        scheme = http scheme
        method = http request method verb unicode
        path = http url path section in unicode
               path may include scheme and netloc which takes priority
        qargs = dict of http query args
        fragment = http fragment
        headers = dict of http headers
        body = byte or binary array of request body bytes or bytearray
        data = dict of request body json if any
        fargs = dict of request body form args if any
        msg = bytearray of response msg to parse
        dictable = Boolean flag If True attempt to convert body from json
        events = deque of events if any
        requests = deque of requests if any each request is dict
        responses = deque of responses if any each response is dict
        redirectable = Boolean is allow redirects
        redirects = list of redirects if any each redirect is dict
        """
```

Backup Slides

Universal Name-Spaced Pub/Sub Interface for Intra application communication

high replacement independence

Addressing Modes:

direct indirect-absolute

```
put name sam eyes blue into .person.detail
```

indirect-relative-implied indirect-relative-root

```
set speed by blue in myapp.setup
```

direct indirect-relative-frame

```
put name sam eyes blue into detail of frame start
```

direct indirect-relative-framer

```
put true into good of framer
```

FloScript

References

http://ioflo.com
https://github.com/ioflo
https://github.com/ioflo/ioflo_manuals

http://www.jpaulmorrison.com/fbp/
http://www.jpaulmorrison.com/fbp/links_external.html
https://flowbasedprogramming.wordpress.com/article/flow-based-programming/
http://www.amazon.com/Flow-Based-Programming-2nd-Application-Development/dp/1451542321/ref=sr_1_1?ie=UTF8&qid=1427910581&sr=8-1&keywords=flow+based+programming

http://wiki.ros.org/ROS/Tutorials

Flow-Based Programming Resources

Flow-Based Programming, 2nd Ed. May 14, 2010

http://www.jpaulmorrison.com/fbp/

http://www.jpaulmorrison.com/fbp/links_external.html

https://flowbasedprogramming.wordpress.com/article/flow-based-programming/

Port Automata/ Port Based Objects

Port Automata and the Algebra of Concurrent Processes, Steenstrup & Arbib 1982

SoftwareComponentsForRealTime, Stewart 2000

RAET Reliable Asynchronous Event Transport

https://github.com/RaetProtocol/raet

- RAET
  - Micro threaded architecture with non-blocking I/O
  - Micro threaded Event Pub/Sub separate from socket based transport layer
  - UDP sockets
  - Better observability and management of performance under load
  - Transactions
  - Unix domain sockets for interprocess communications

Dependency Inversion Principle

Dependency Inversion Principle, Martin 1996

Bad Software Design: Software that fulfills its requirements but exhibits any or all of:

Rigidity: Hard to change because each change affects other parts of the system.

Fragility: Making a change causes other parts of the system to break.

Immobility: Hard to disentangle in order to reuse in another application.

DIP:

HIGH LEVEL MODULES SHOULD NOT DEPEND UPON LOW LEVEL MODULES.

BOTH SHOULD DEPEND UPON ABSTRACTIONS.

ABSTRACTIONS SHOULD NOT DEPEND UPON DETAILS.

DETAILS SHOULD DEPEND UPON ABSTRACTIONS.

Its all about dependency management, duh !!!

Dependency Inversion Principle Transcendence

Bad Software Design: Software that fulfills its requirements but exhibits any or all of:

Rigidity: Hard to change because each change affects other parts of the system.

Fragility: Making a change causes other parts of the system to break.

Immobility: Hard to disentangle in order to reuse in another application.

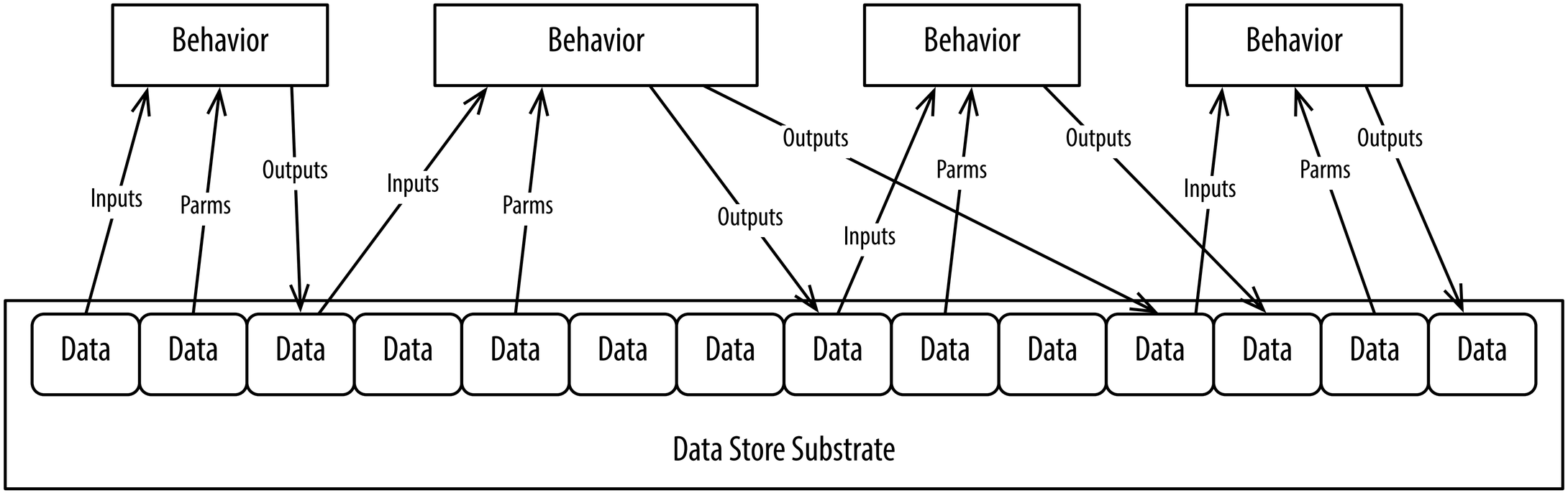
Flow-Based Programming transcends the DIP thusly:

THERE ARE NO MODULES, JUST COMPONENTS

THERE ARE NO ABSTRACTIONS OR DETAILS JUST DATA

COMPONENTS DEPEND ON DATA, NOT OTHER COMPONENTS

One Dependency: DATA



Complexity Management

Real Complexity = number of dependencies between elements of a software system

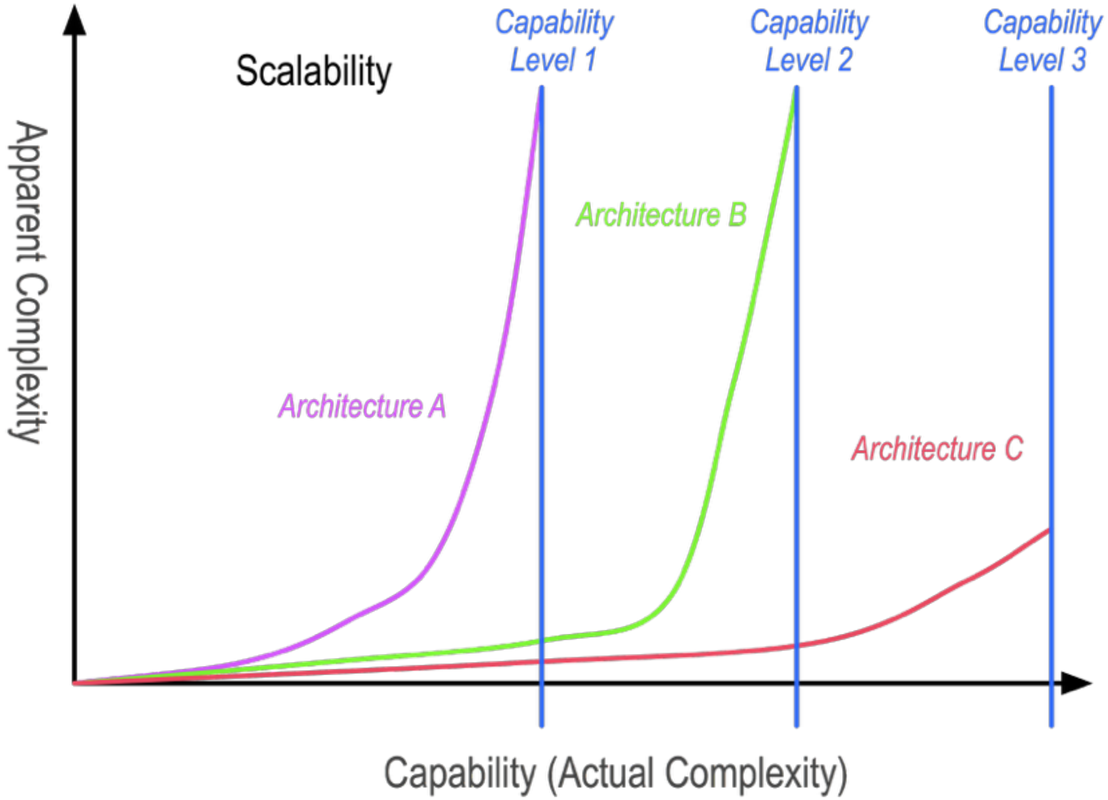
Apparent Complexity = number of dependencies that programmers must manage in order to make meaningful enhancements to software functionality

Perceived Risk = peril the programmer faces when attempting to add meaningful enhancements to software functionality.

Scalability

Higher levels of capability increase *real complexity* and may increase *apparent complexity*.

The principle limitation is programmer capacity not computational capacity



A given architecture/development team has a given maximal apparent complexity capacity.

When the actual apparent complexity starts to reach the maximal capacity, development becomes painful.