

Web Applications in Clojure and ClojureScript with Pedestal

Brenton Ashworth
Cognitect



What you're in for

Focus on client

What kind of application?

An example

Pedestal overview

The future

Interactive applications

continuous two-way transfer of information

Interactive applications

receive inputs from multiple sources

coordination on the client

large, long-running applications

Why something new?

we need more than just a language

existing libraries and frameworks are either

not well suited for this kind of application

rely heavily on mutation and OO concepts

Example application

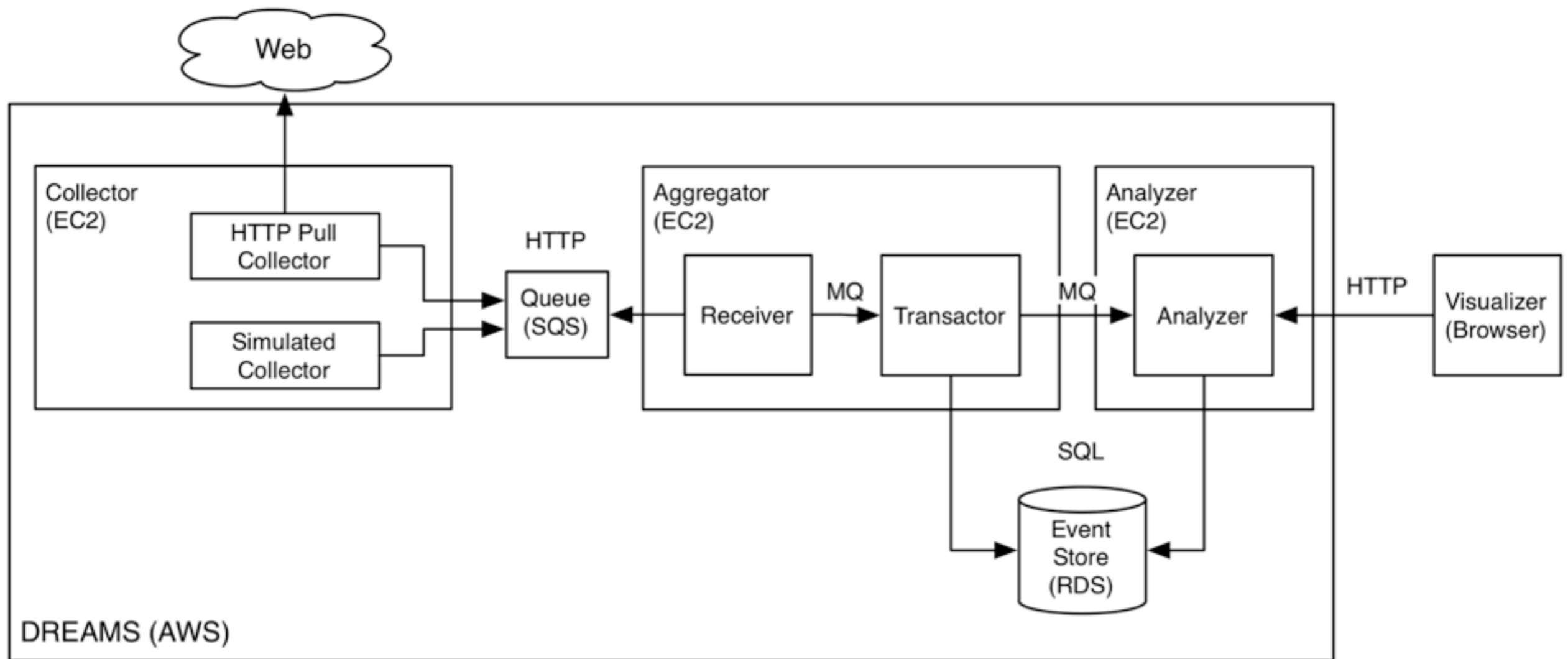
high volume sales funnel

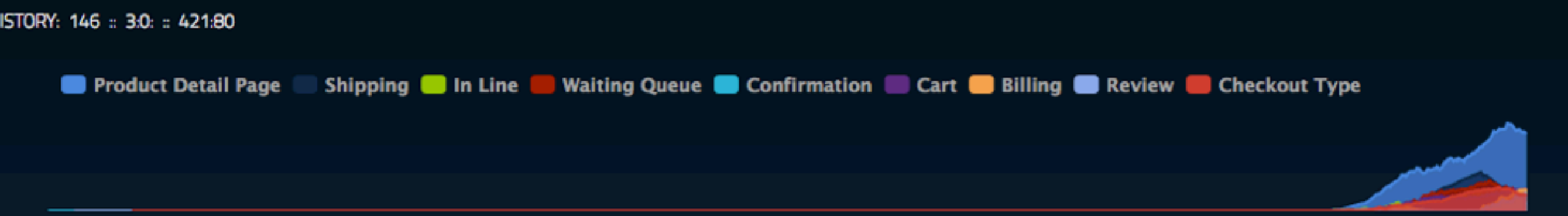
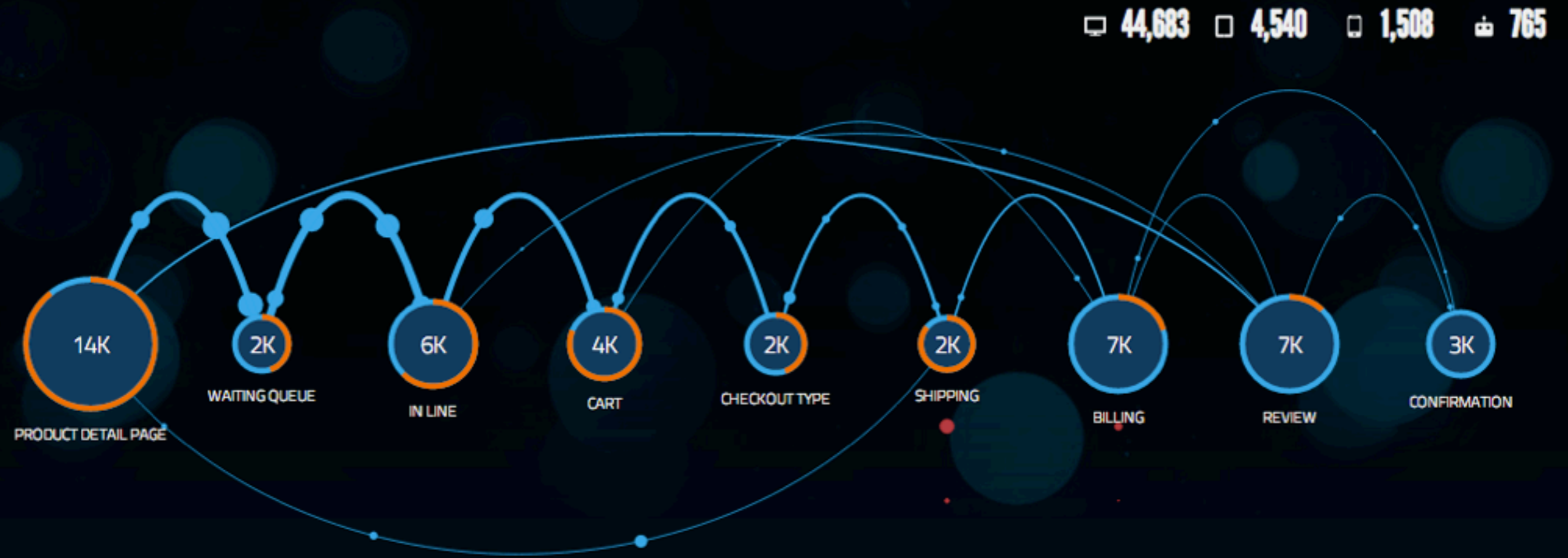
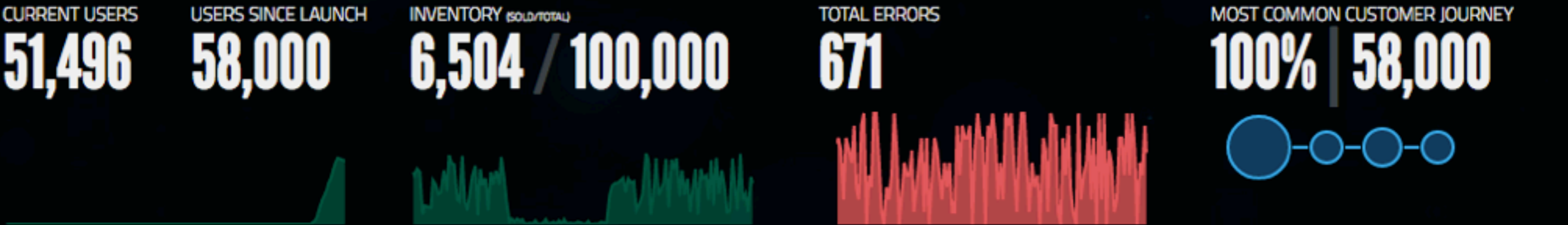
would like to see how shoppers are
progressing in near real time

need to react to problems in near real time

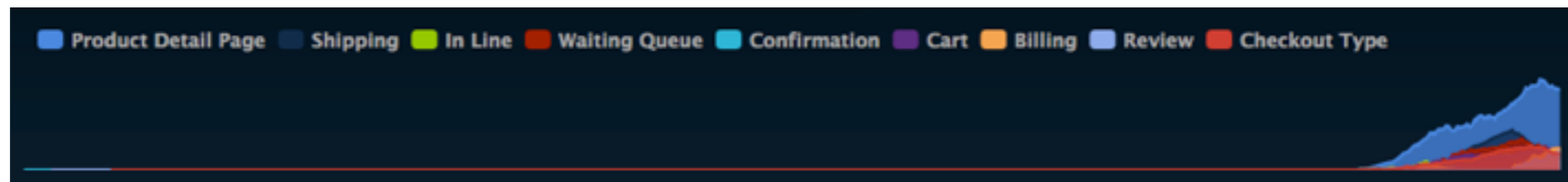
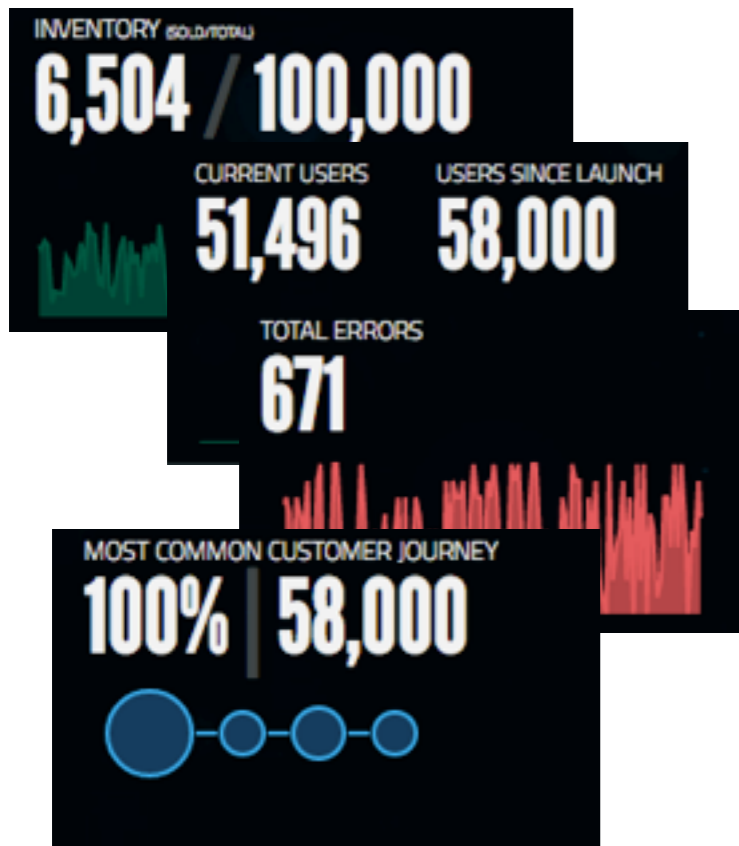
~1000 events per second

Example application





Widgets



Pedestal provides

Separation of concerns

Information model

State transition model

Dataflow

Separation of concerns

Rendering

Information model & logic

Services

Separation of concerns

Rendering

Information model & logic

Services

with messages and queues

Message examples

```
{msg/type :add-name msg/topic [:states :cart] :size 42}
```

op

target

arguments

```
[ :node-create [:states :cart :size] :map]
```

```
  [ :value [:states :cart :size] 20 42]
```

Information model

a standard way to organize and store
information

Information model

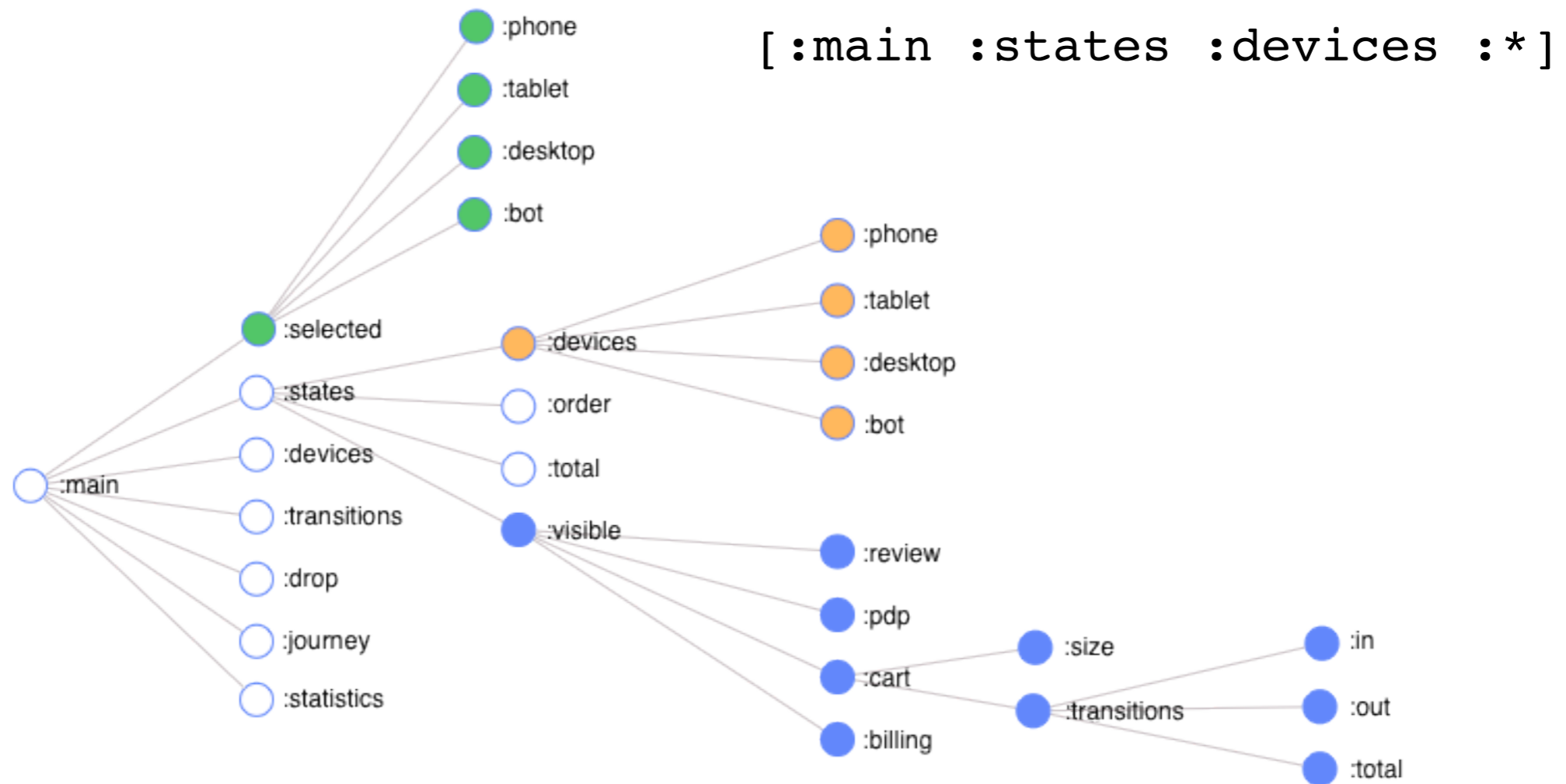
tree (nested map)

basis information

derived information

information which supports the UI

Information model



[:main :states :visible :cart :size]

State transitions

an orderly succession of states

associate cause and effect

State transitions

one atom vs many atoms

would prefer to have one atom

...and see exactly what has changed

State transitions

Pedestal provides

- consistent state transitions

- fine-grained change reporting

Dataflow

disconnect functions from how they are executed

declarative inputs and outputs

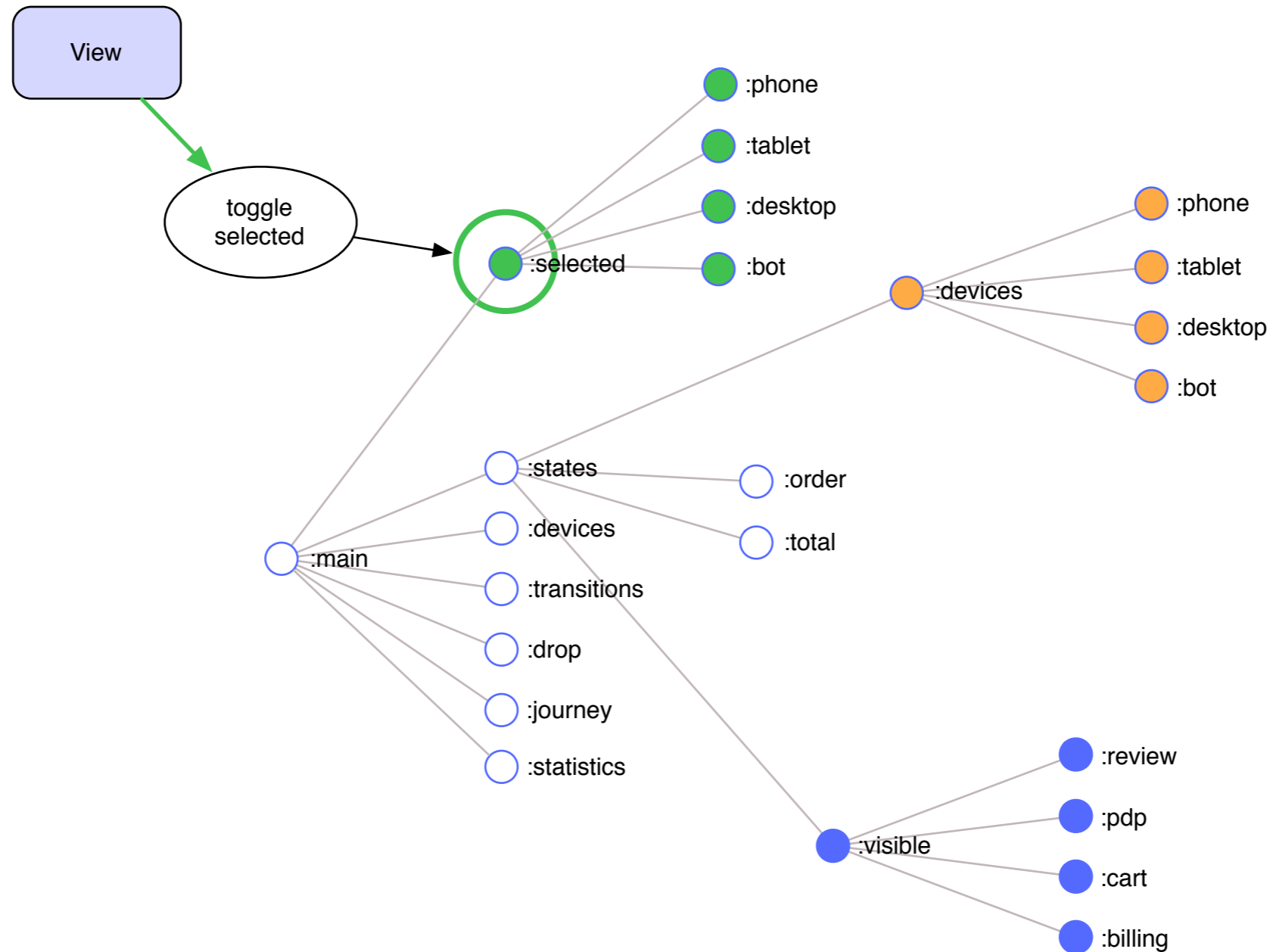
automatic (minimal) propagation

no explicit pub/sub or conditionals

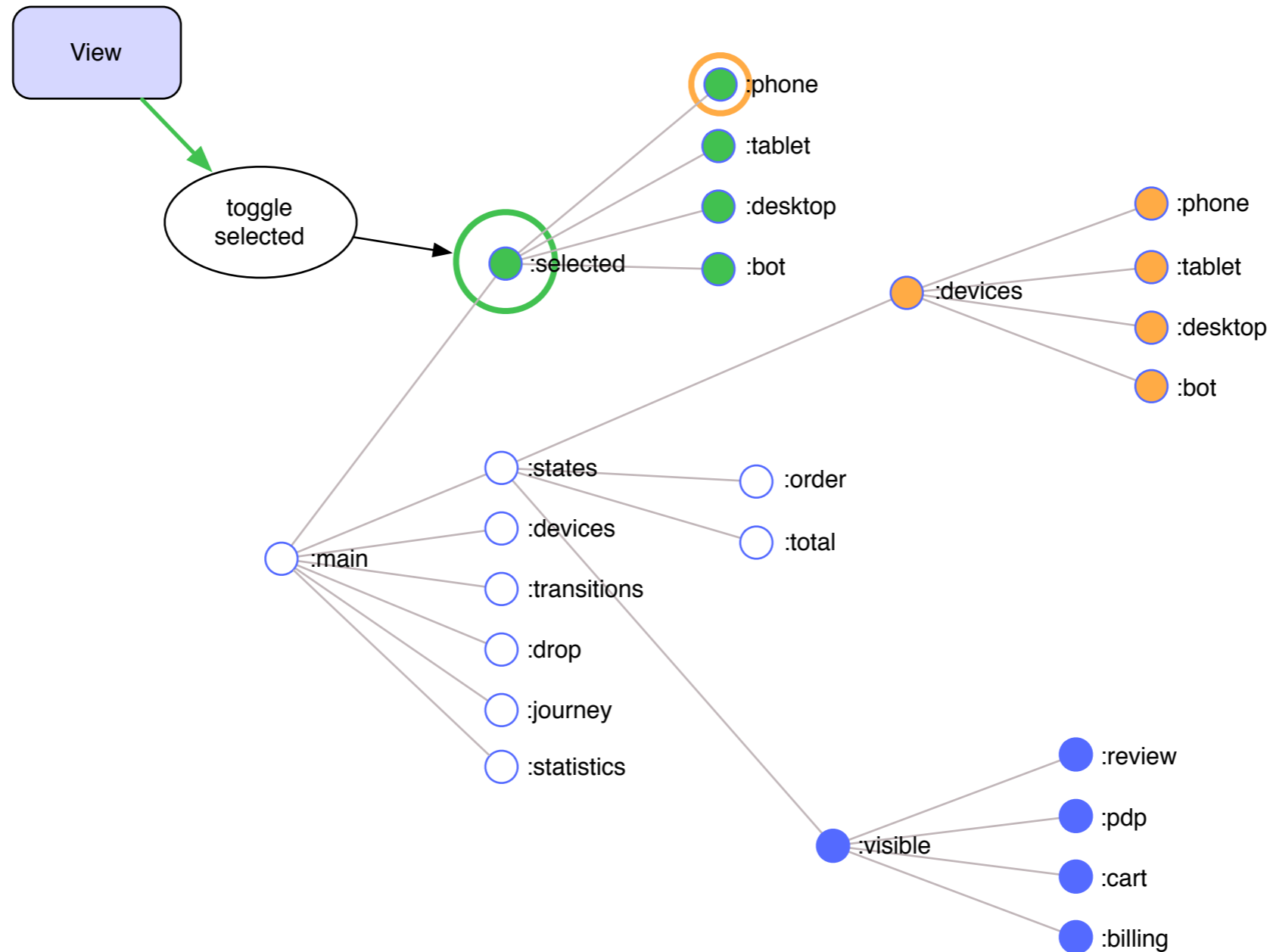
Dataflow

a great way to encode data dependencies
promotes adding code rather than updating

Dataflow

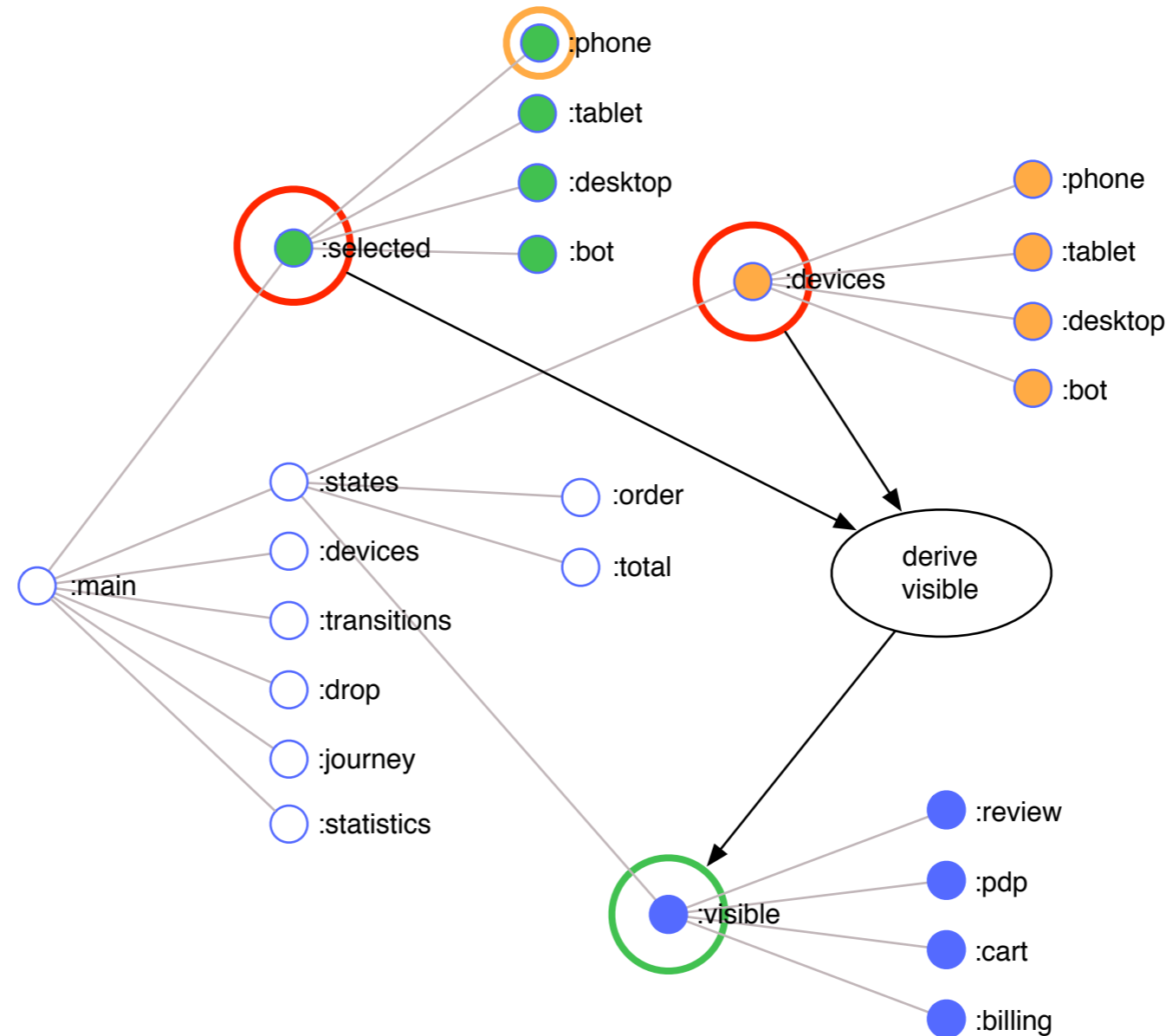


Dataflow



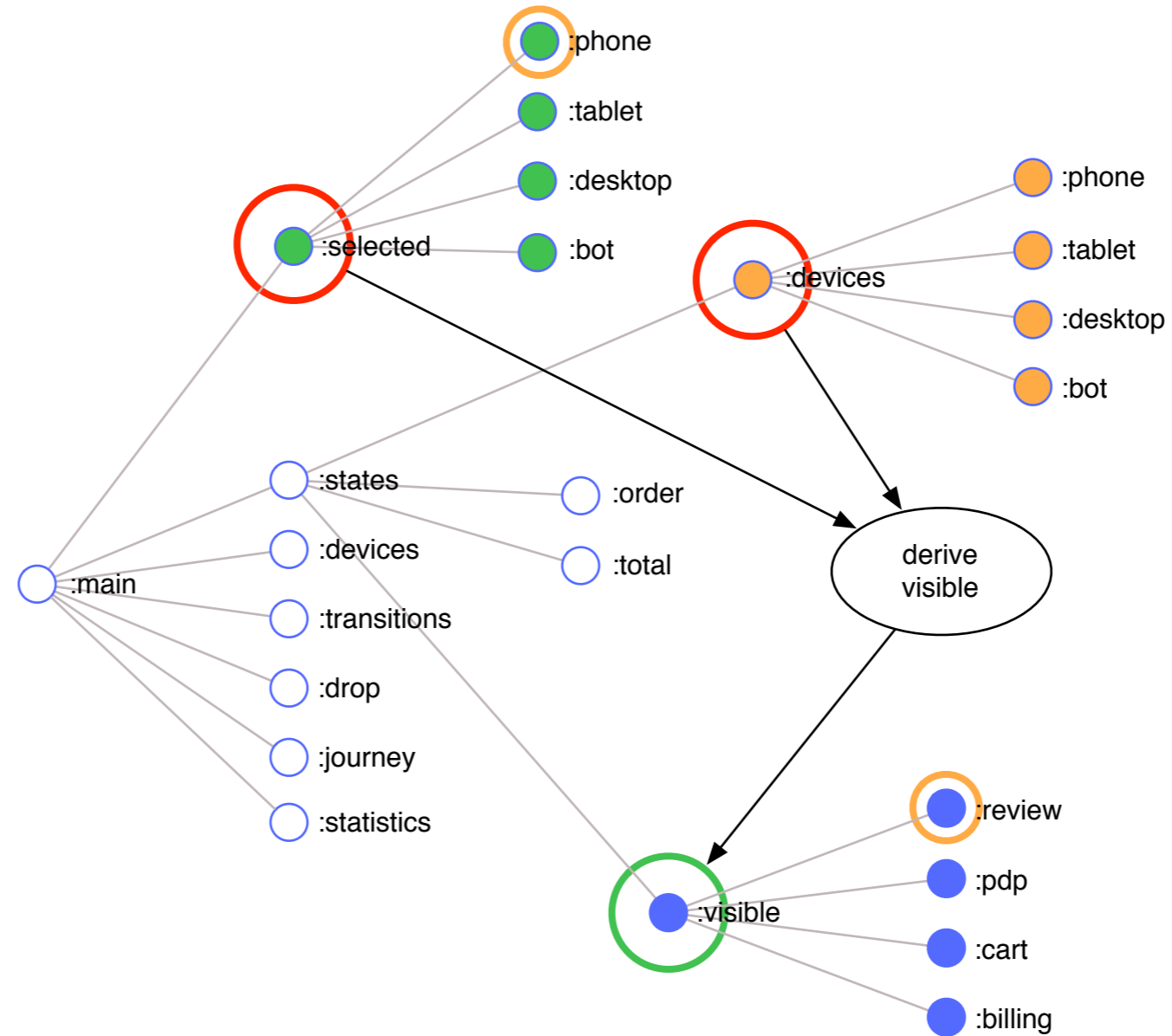
Dataflow

View



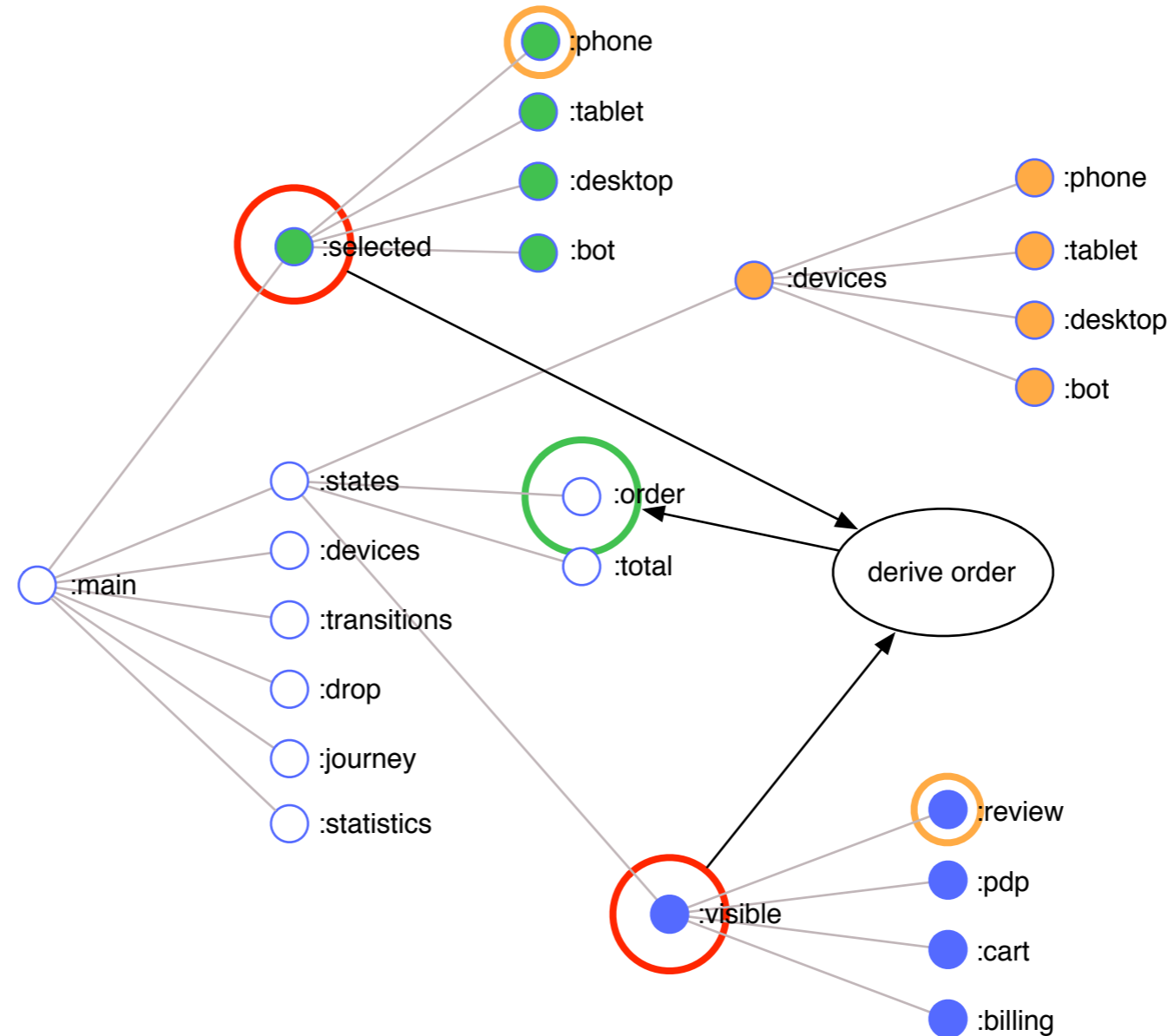
Dataflow

View



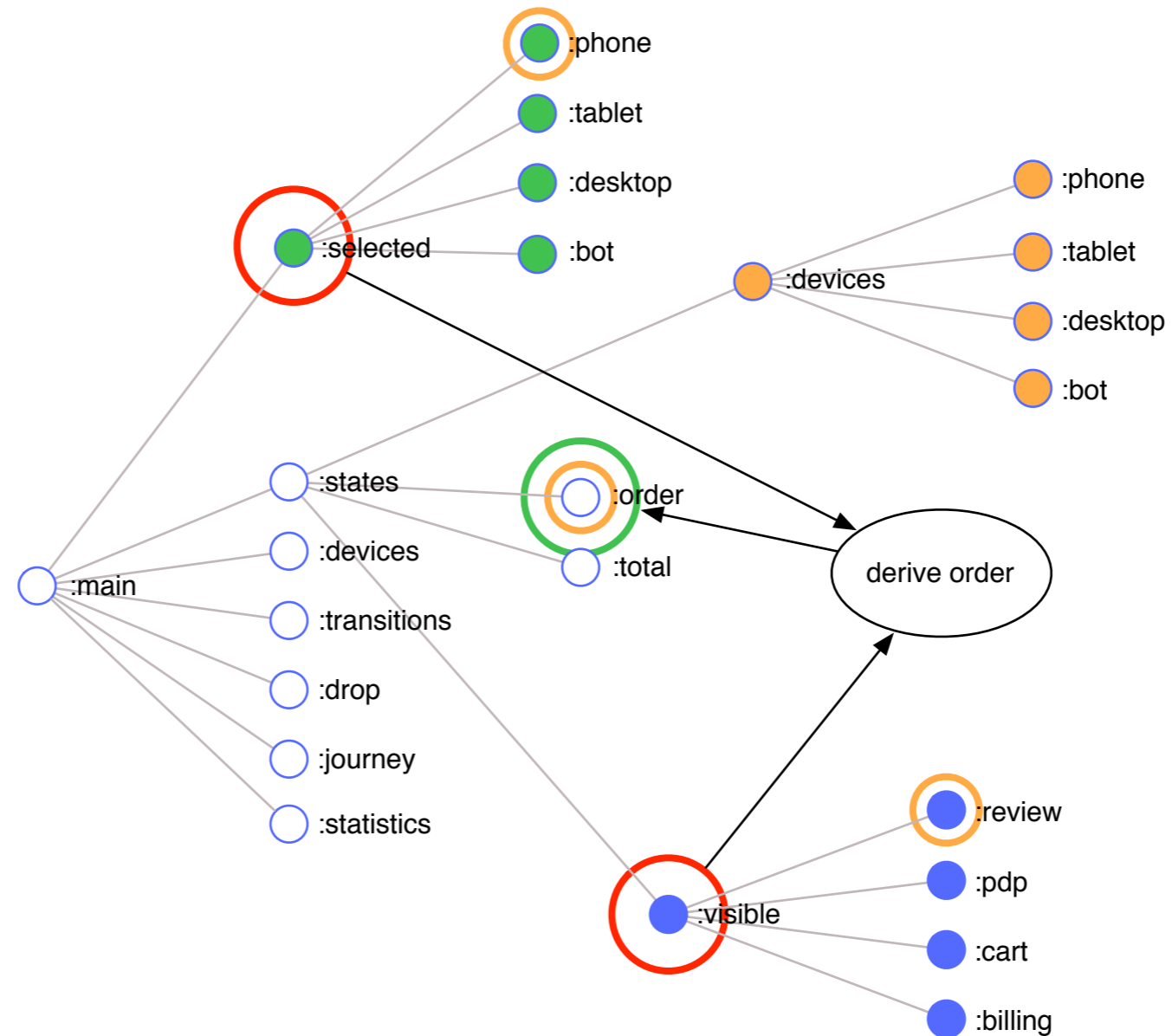
Dataflow

View

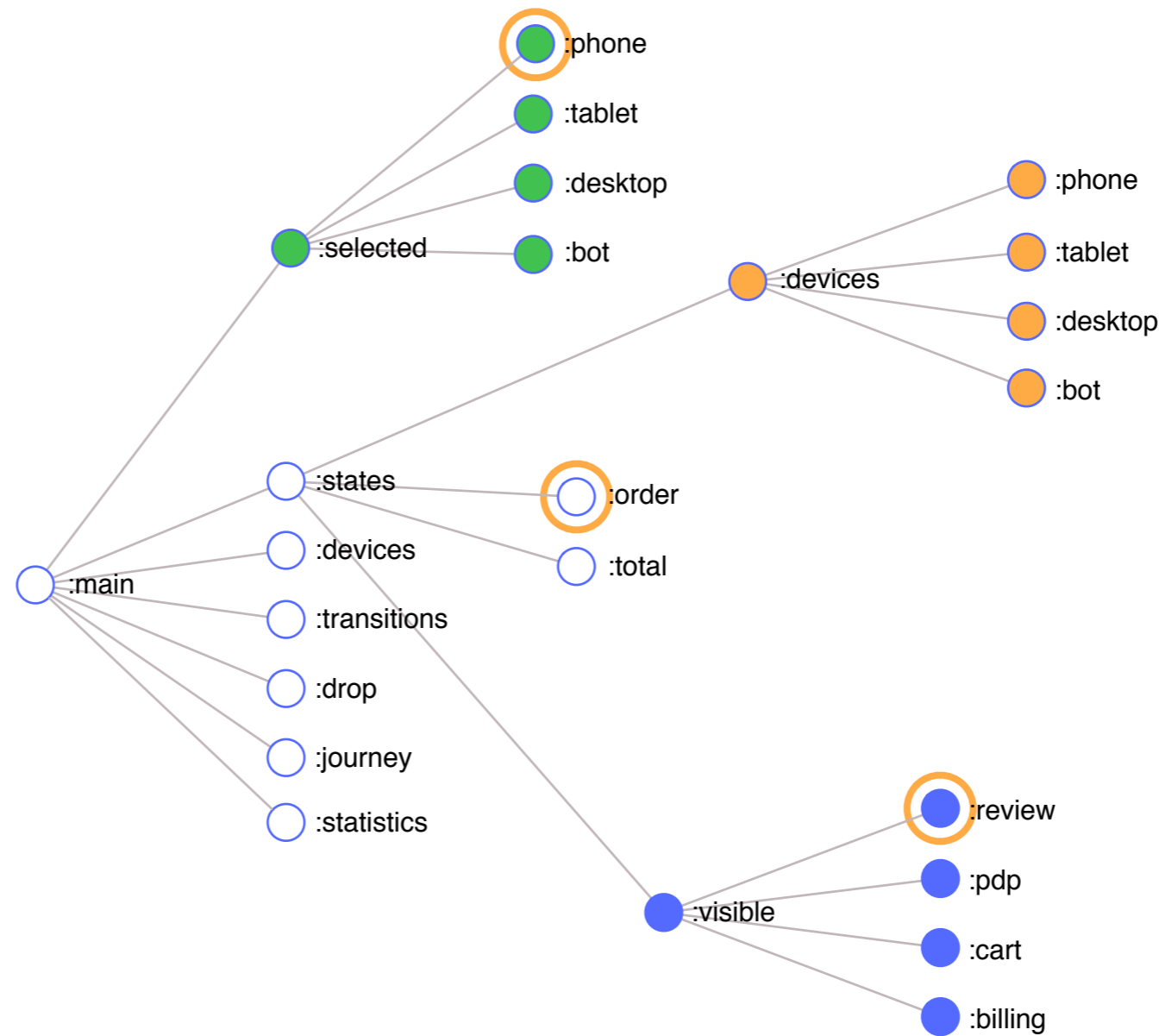


Dataflow

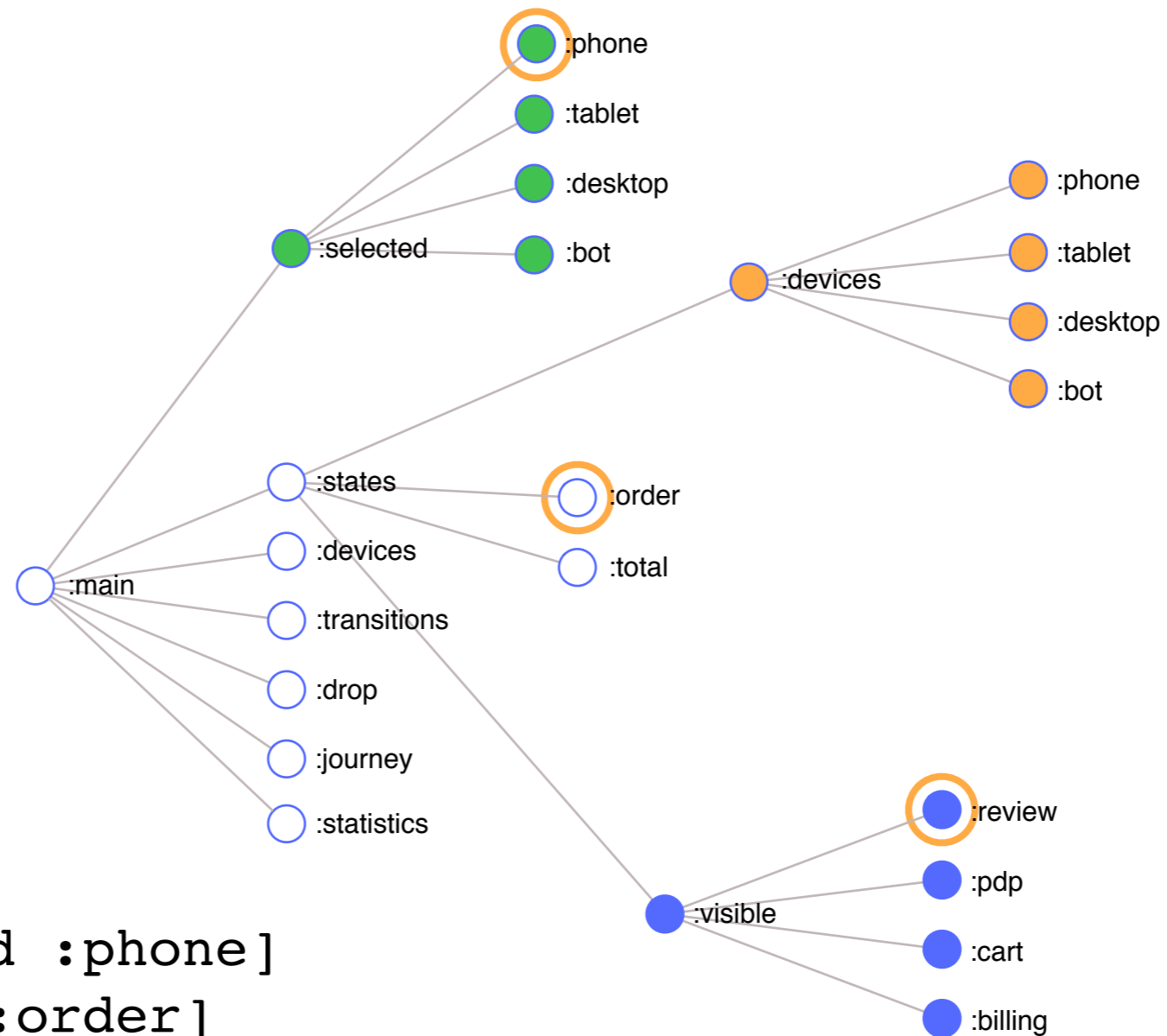
View



Dataflow

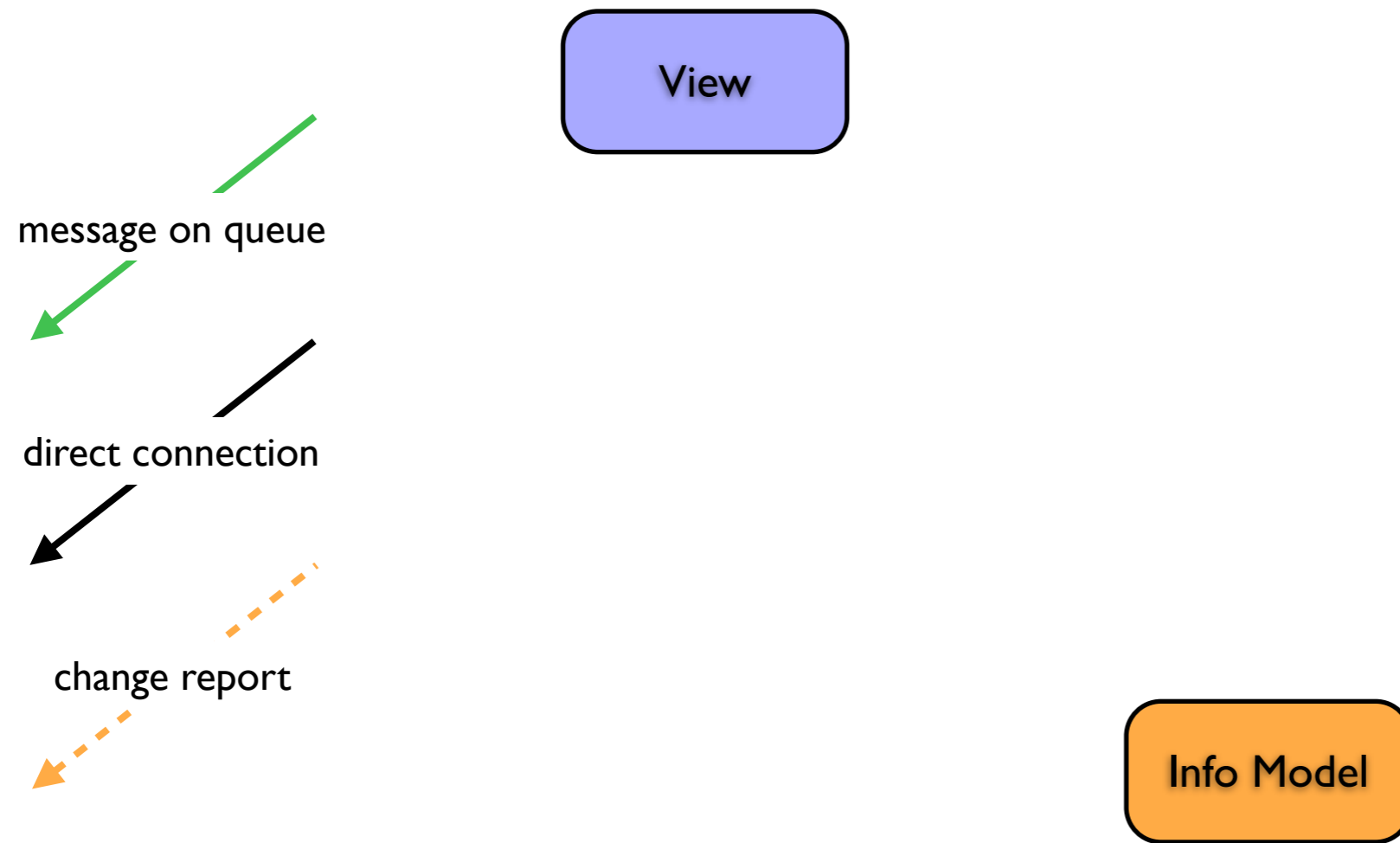


Change report

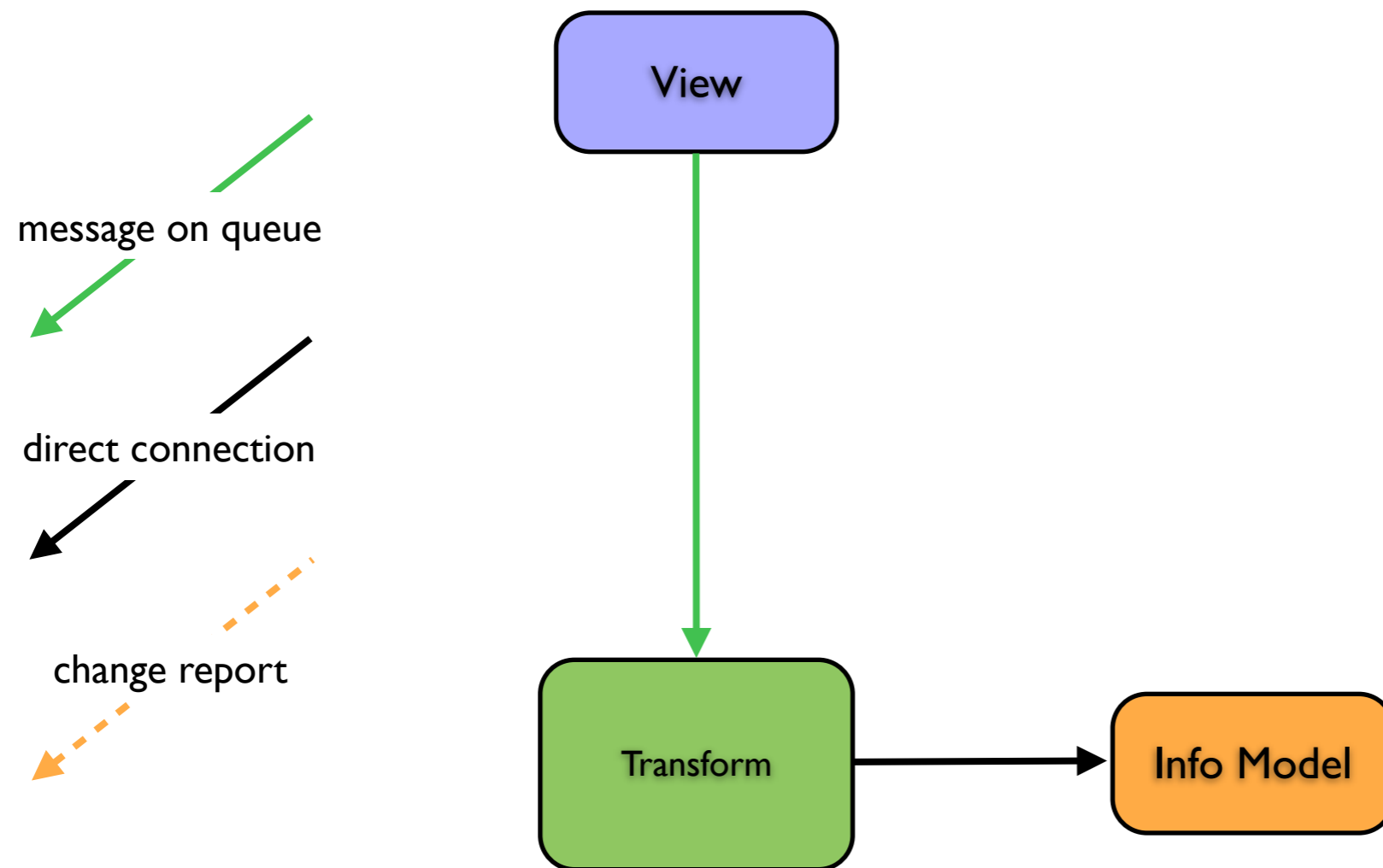


```
[[:main :selected :phone]  
[:main :states :order]  
[:main :states :visible :review]]
```

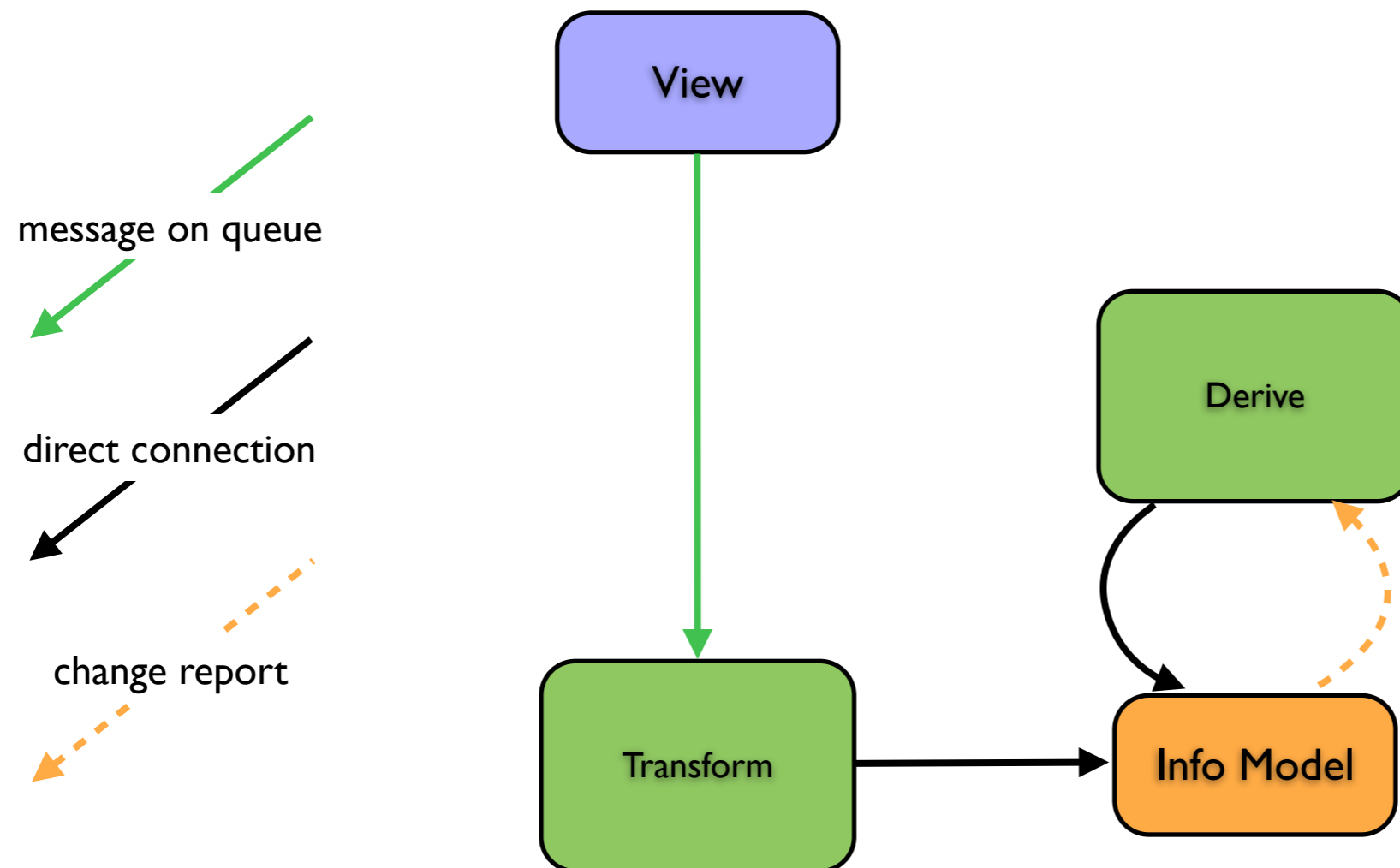
Roundtrip



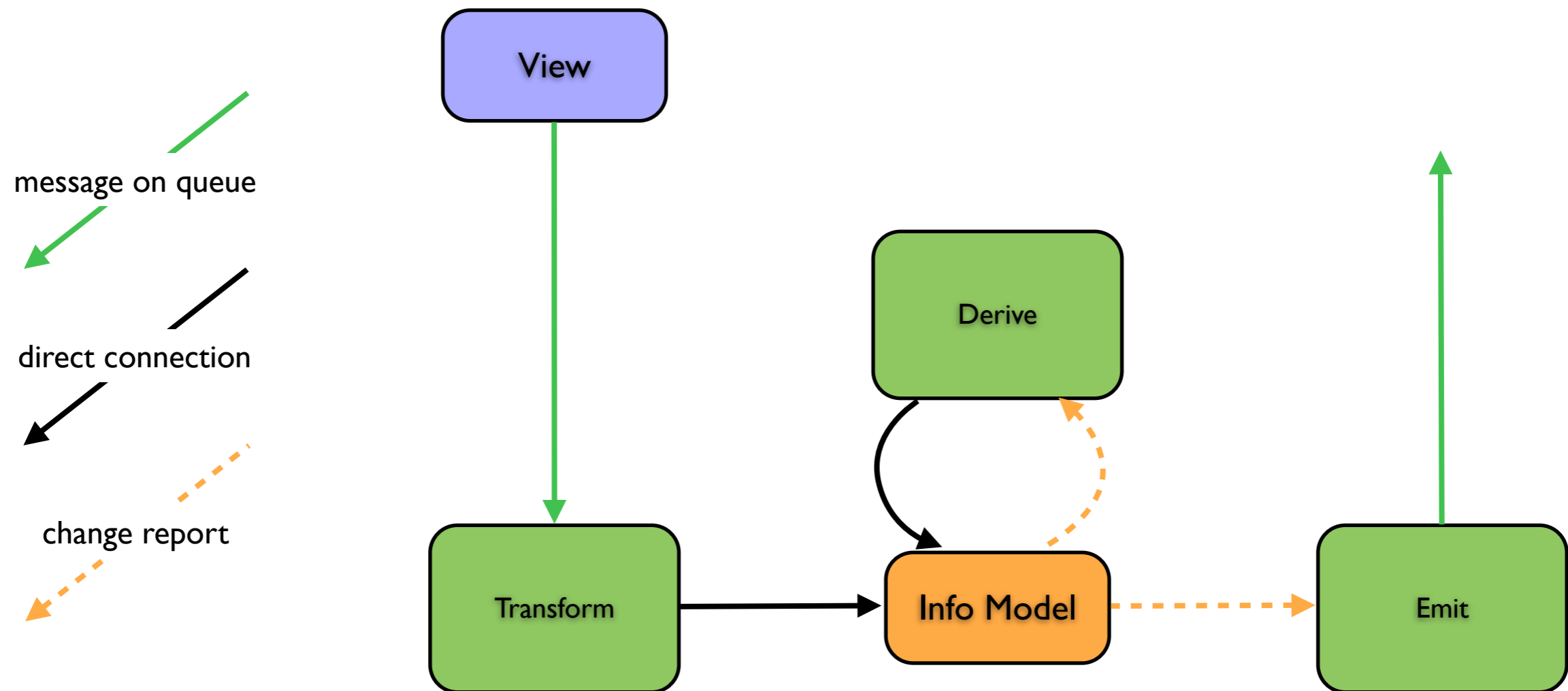
Roundtrip



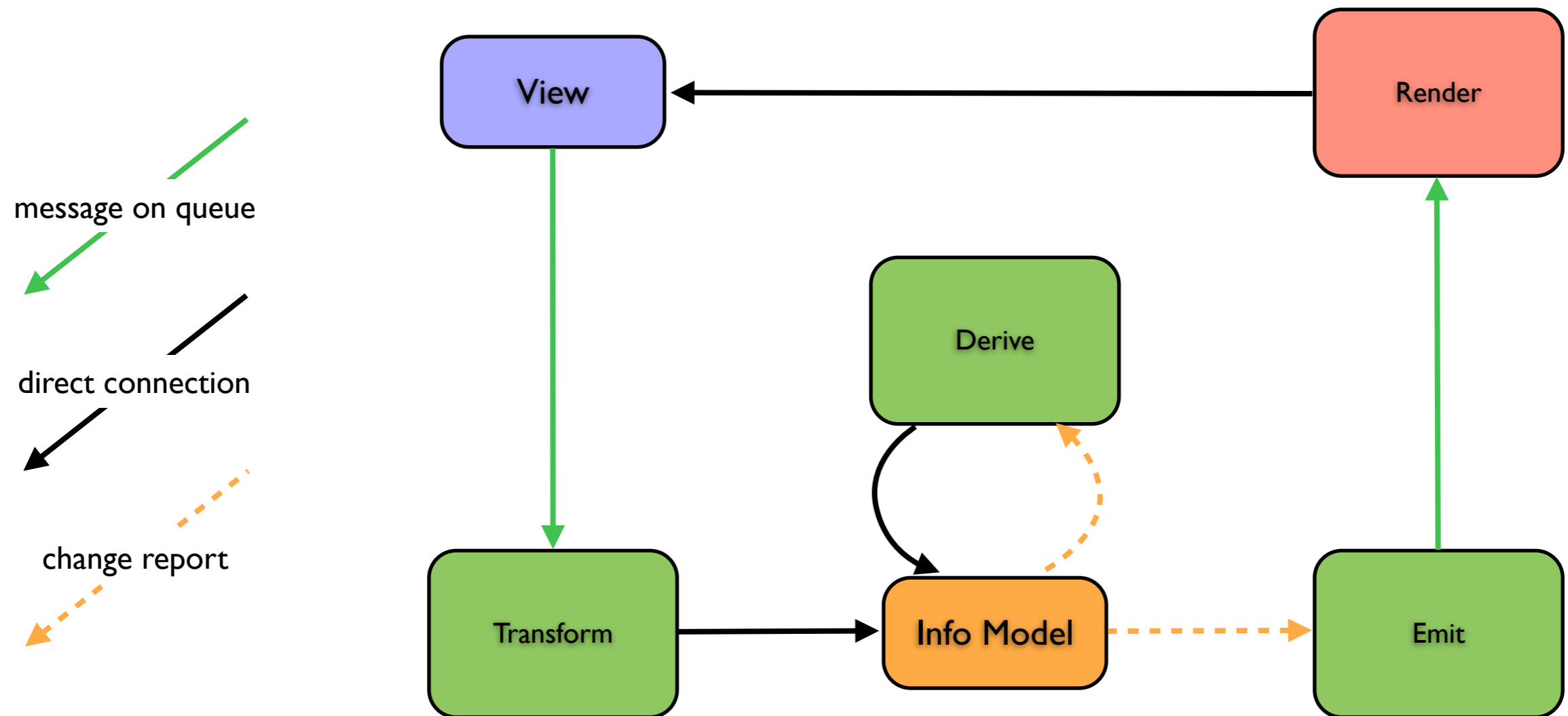
Roundtrip



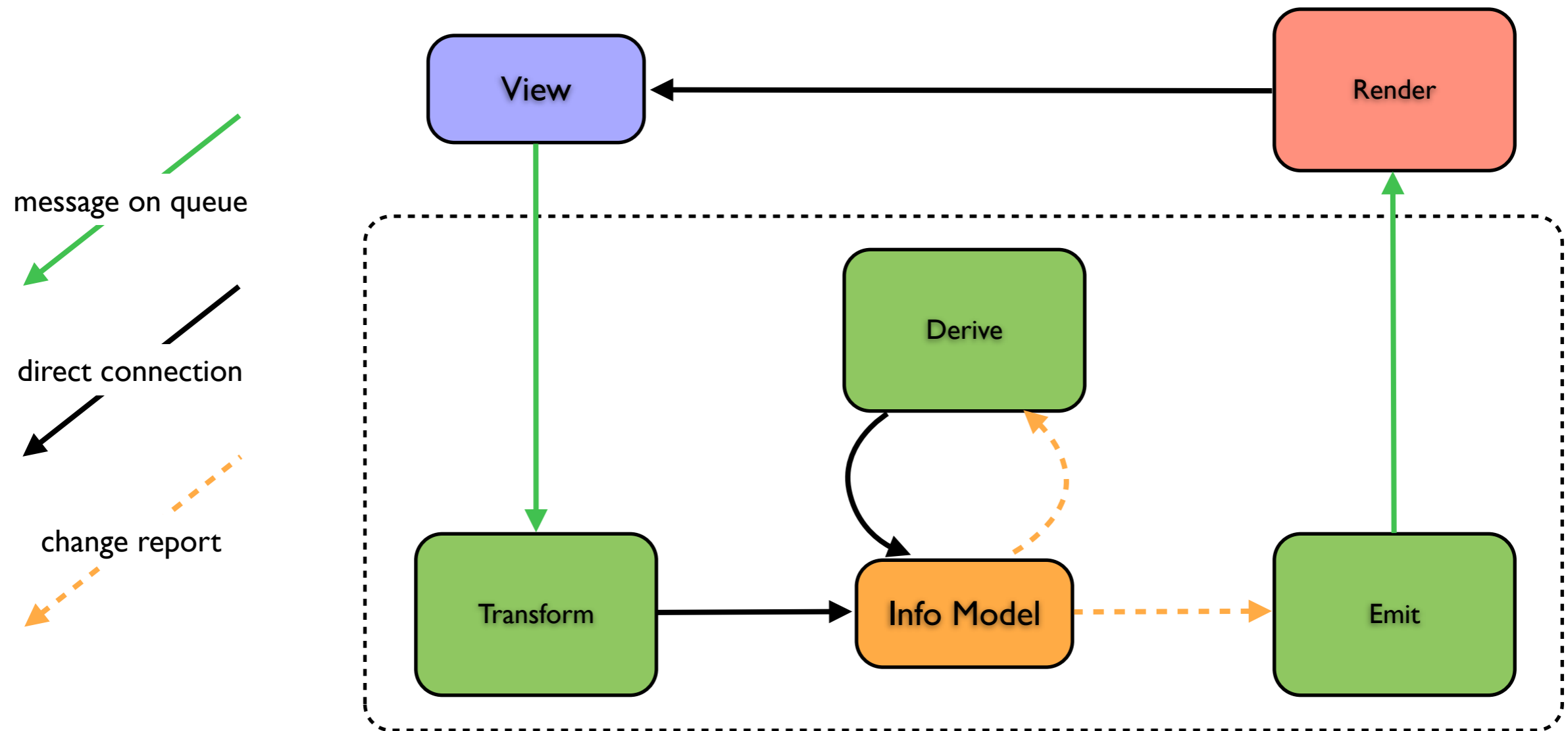
Roundtrip



Roundtrip



Roundtrip



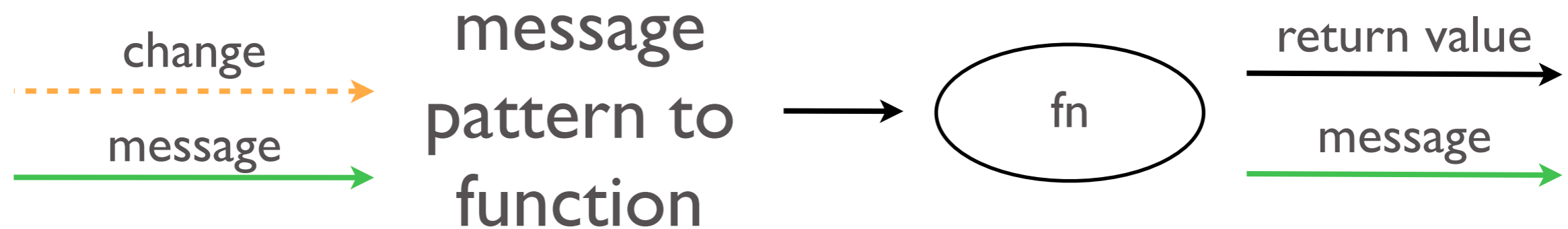
core.async

channels

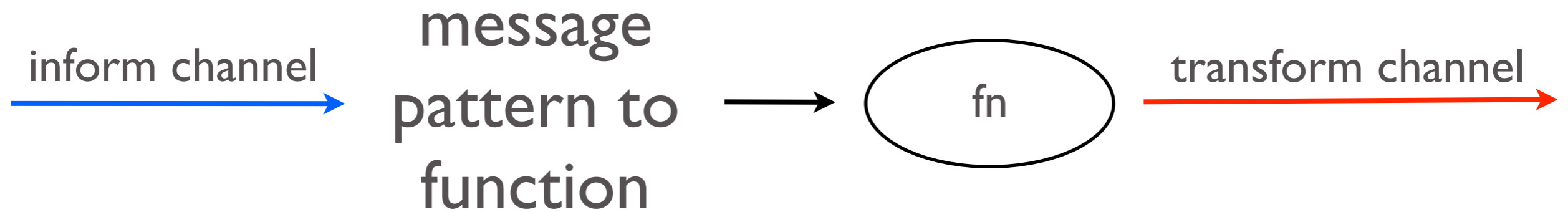
logical threads

blocking

Current message processing



New message processing

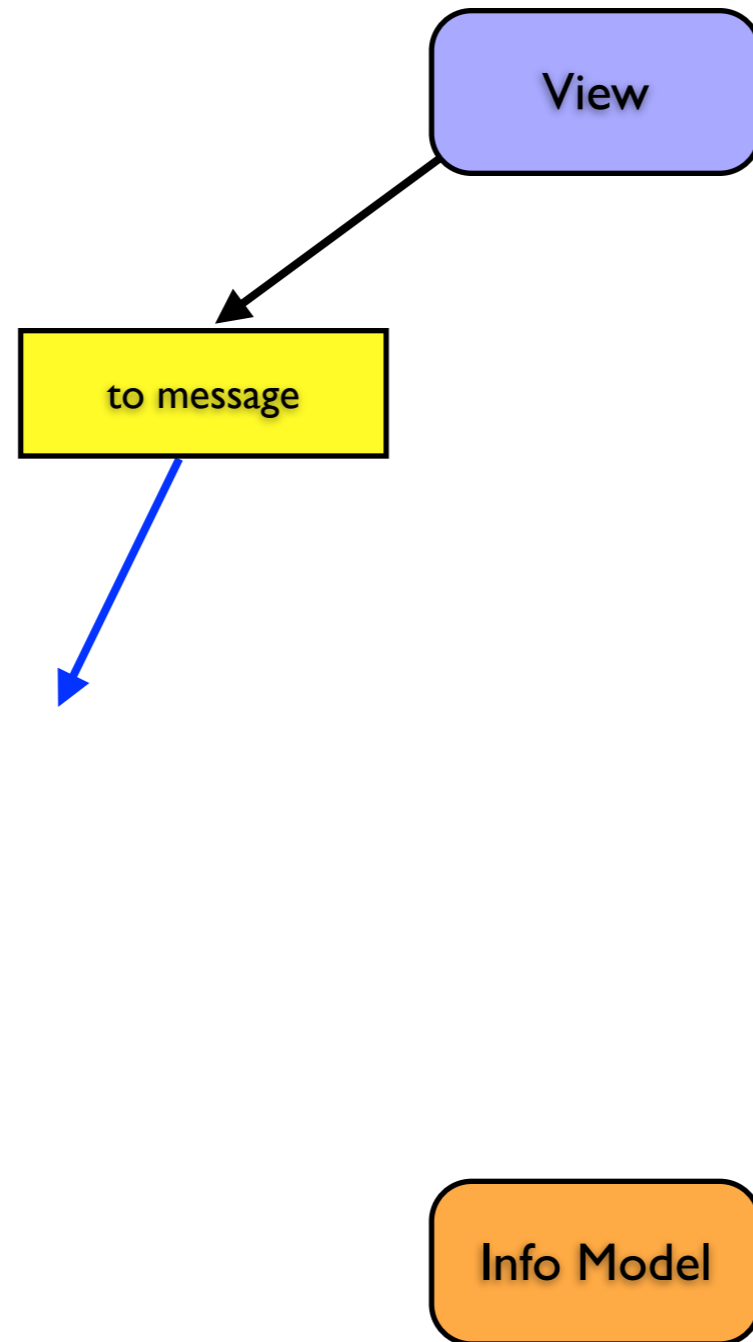


The future

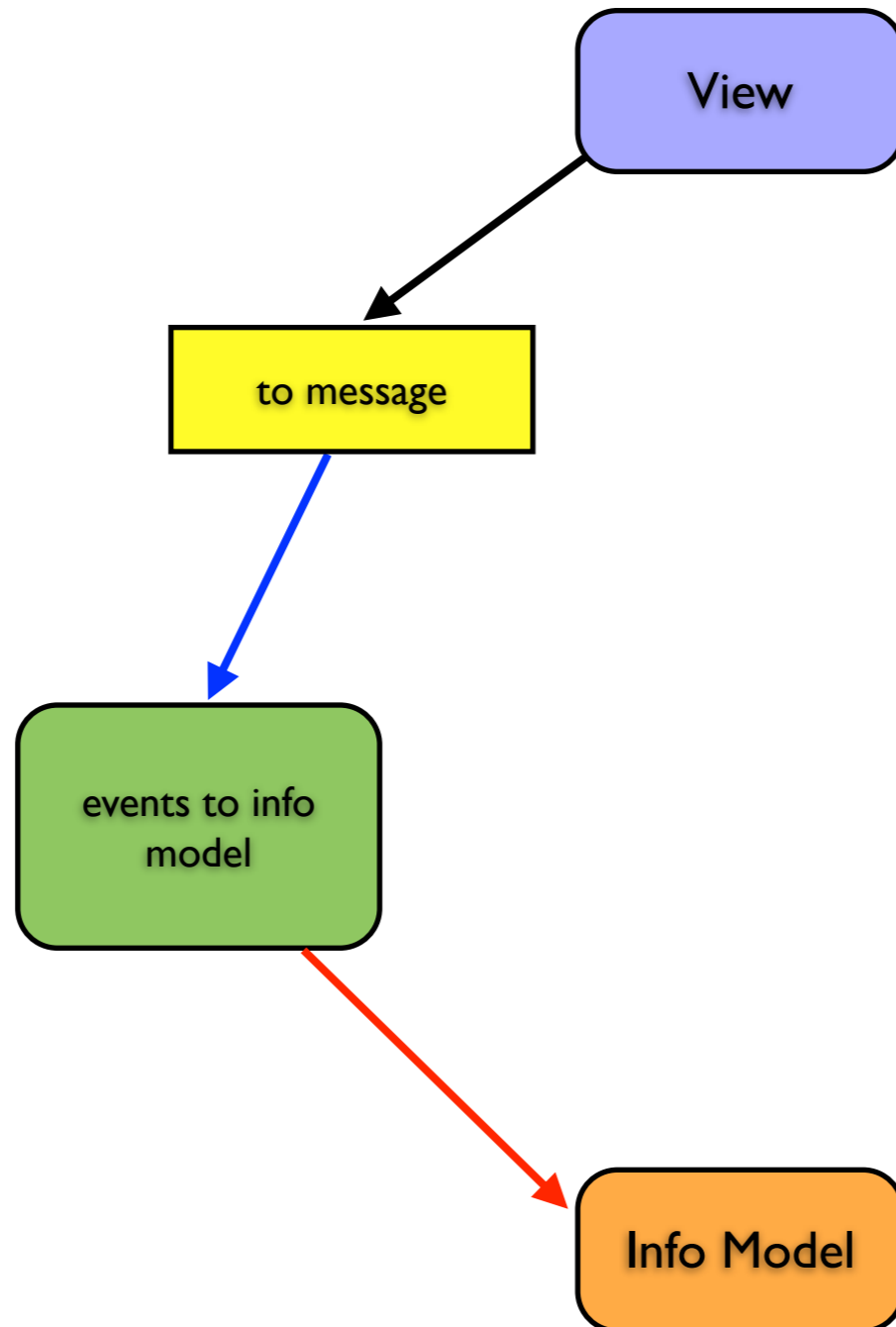
View

Info Model

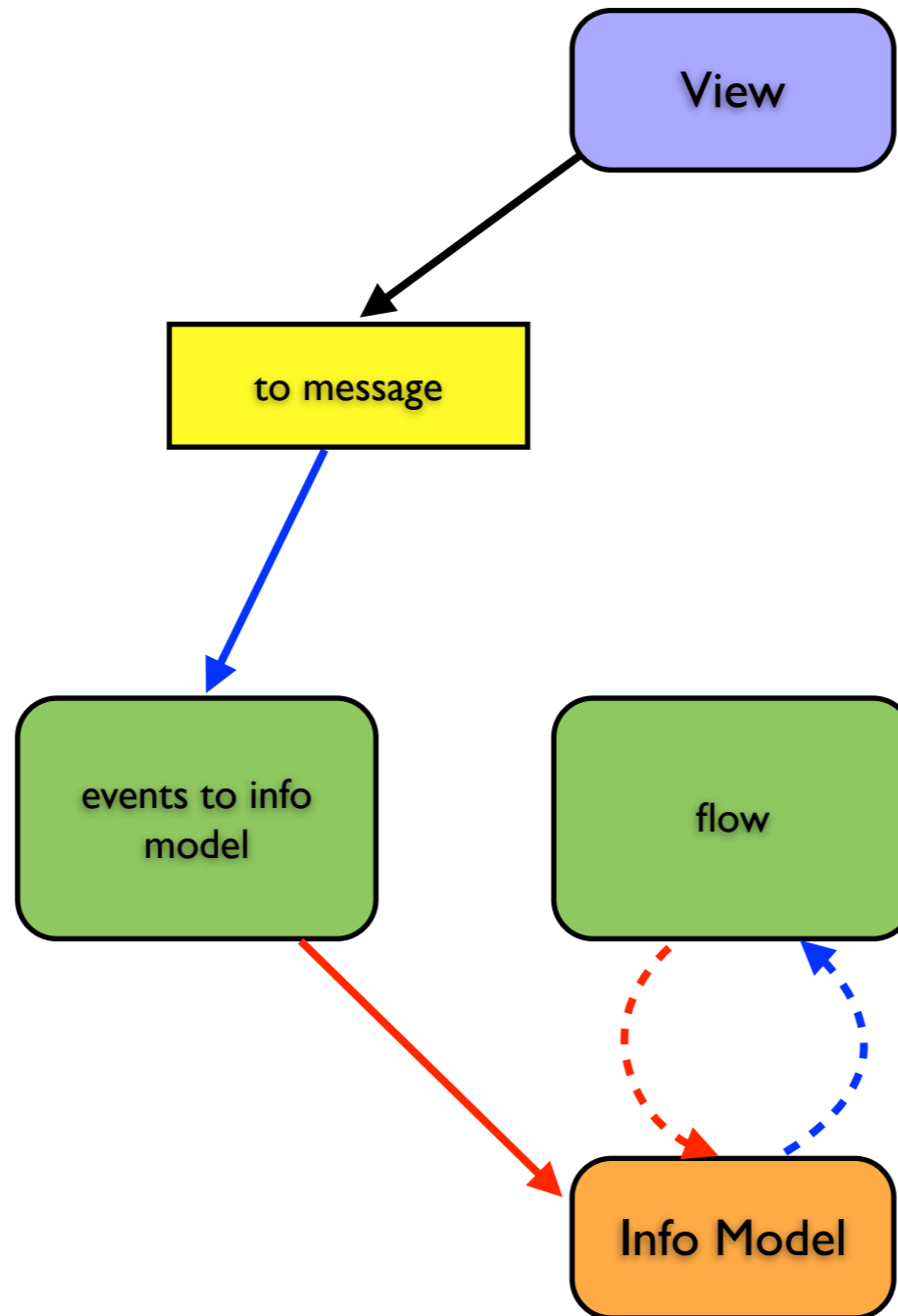
The future



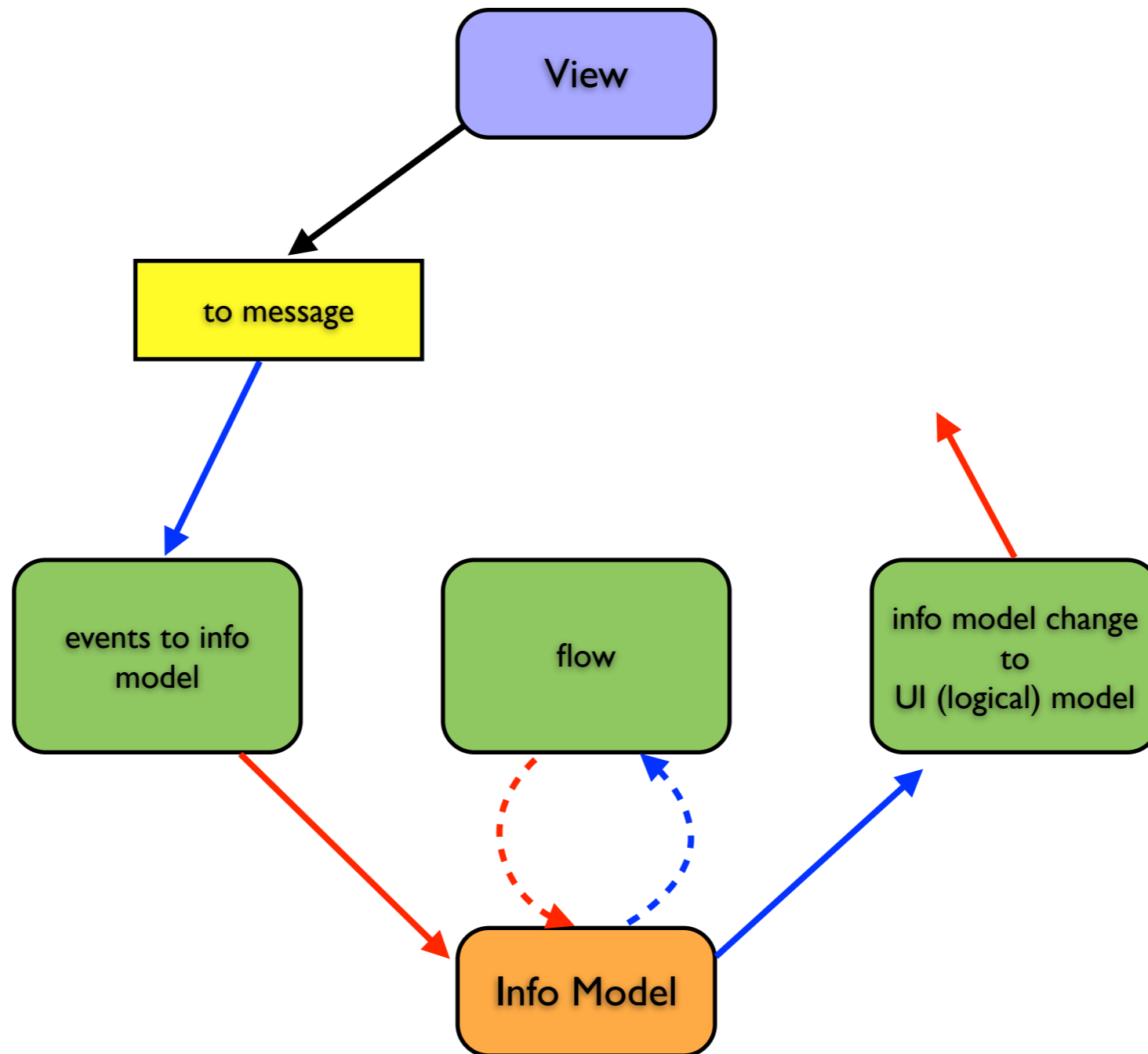
The future



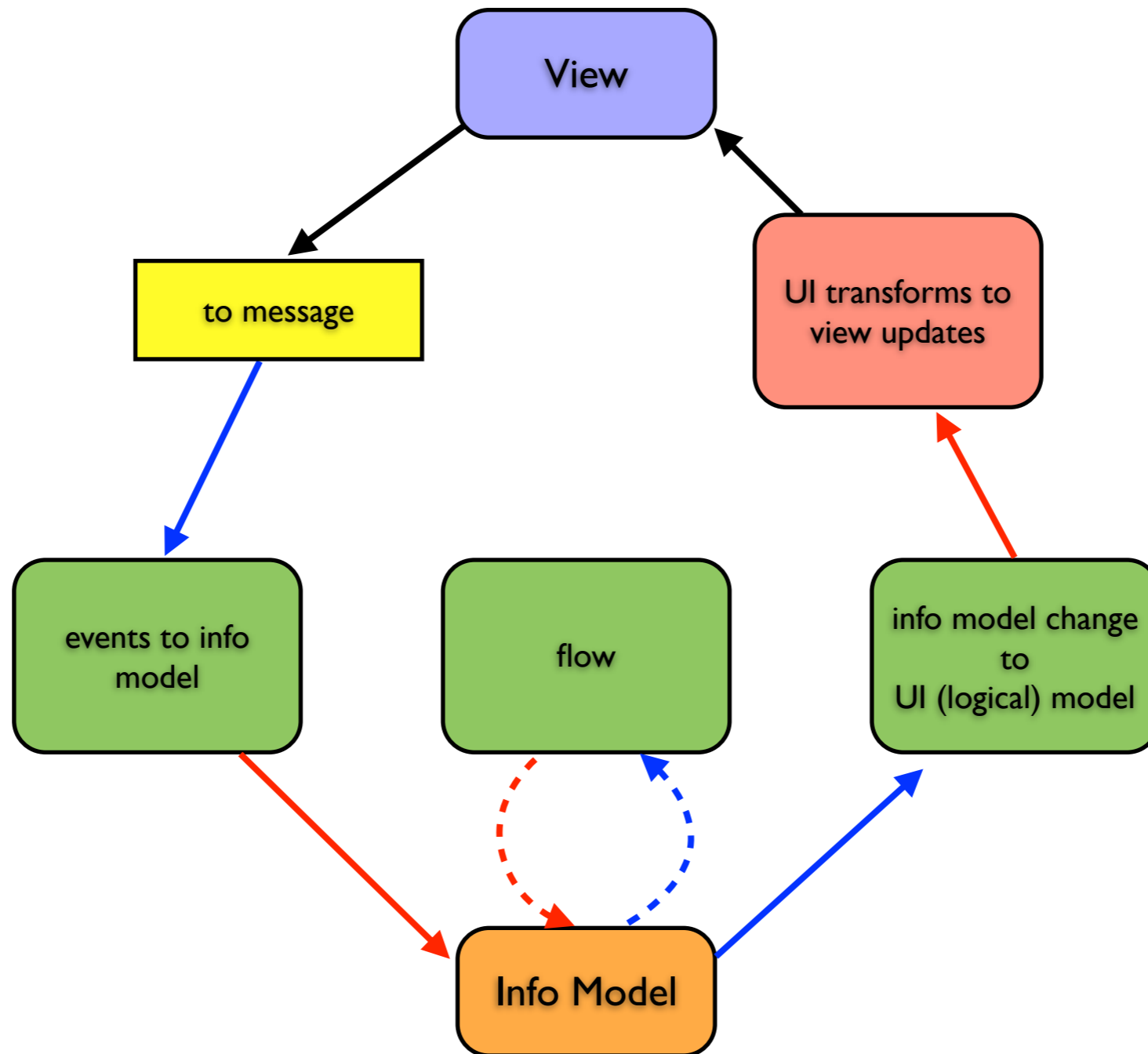
The future



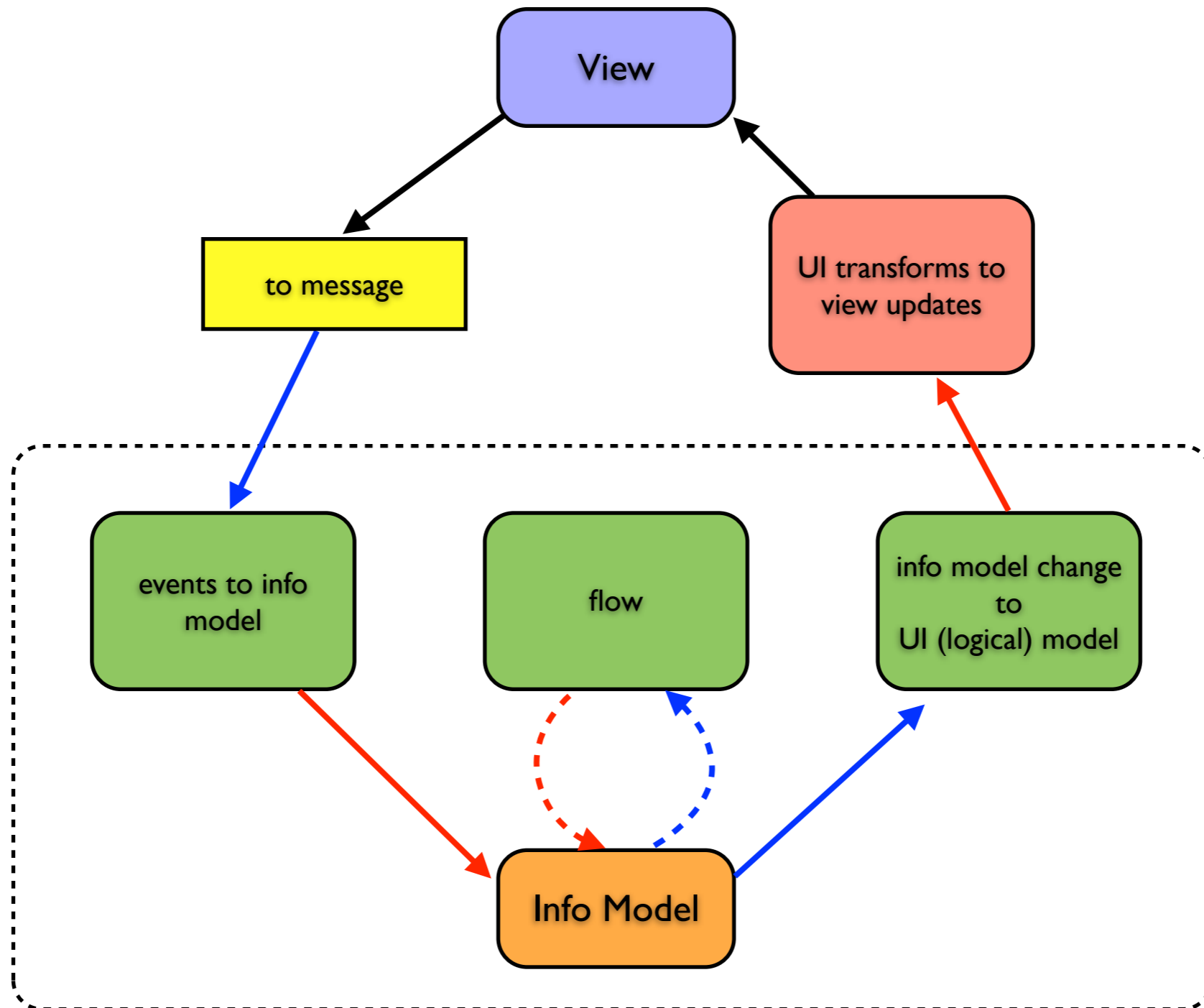
The future



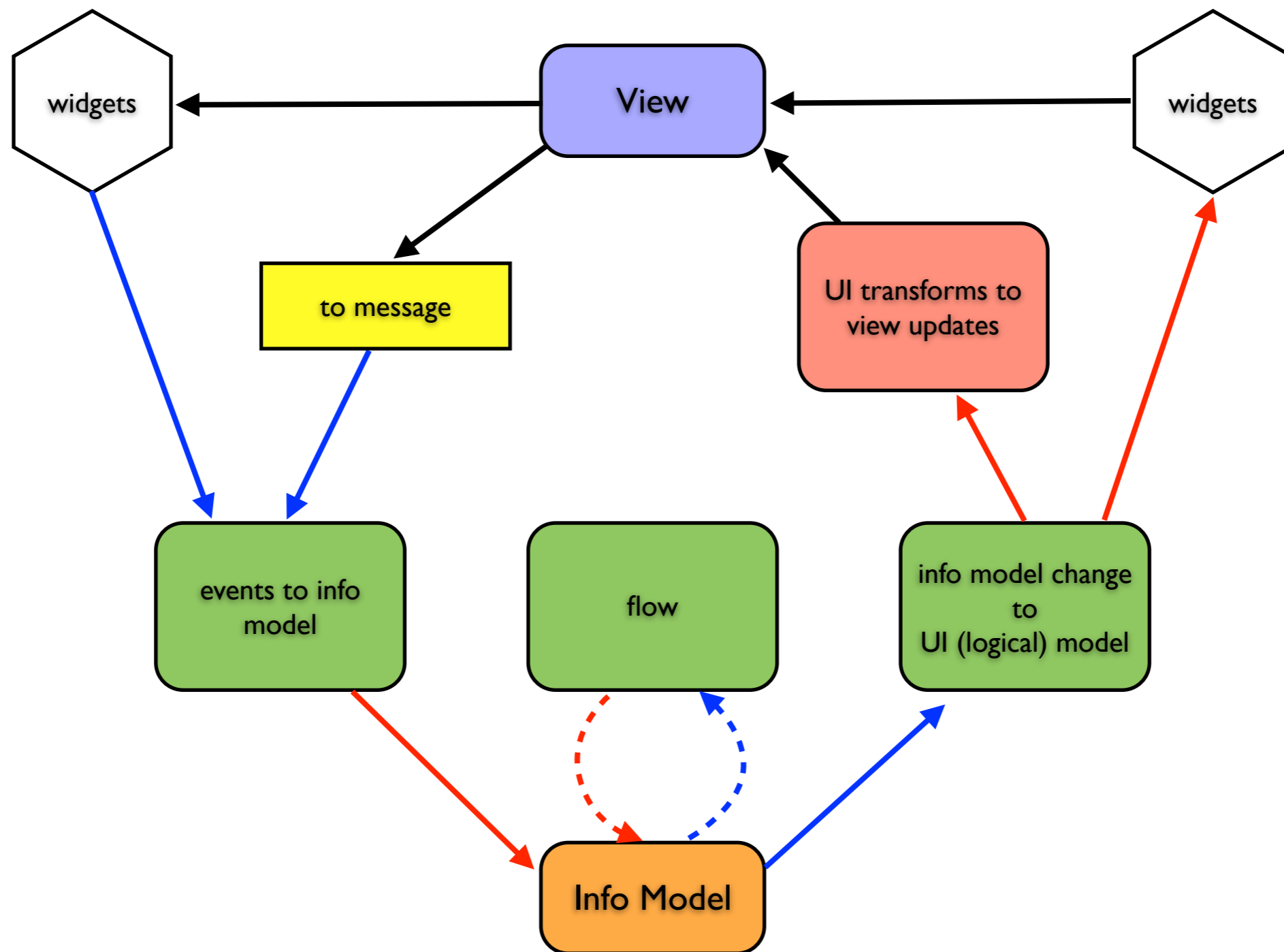
The future



The future



The future



Thank You

<https://github.com/pedestal>

