RESTful Clojure

Siva Jagadeesan

zolodeck

“I am getting frustrated by the number of people calling any HTTP-based interface a REST API”

zolodeck

“I am getting frustrated by the number of people calling any HTTP-based interface a REST API”

Roy T. Fielding

zolodeck

“I am getting frustrated by the number of people calling any HTTP-based interface a REST API”

Roy T. Fielding

Introduced and defined REST

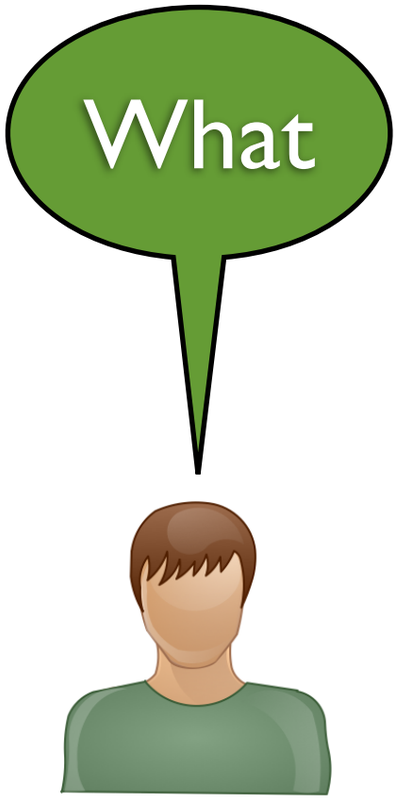
zolodeck

"so-called REST APIs ... choose some other buzzword for your API."

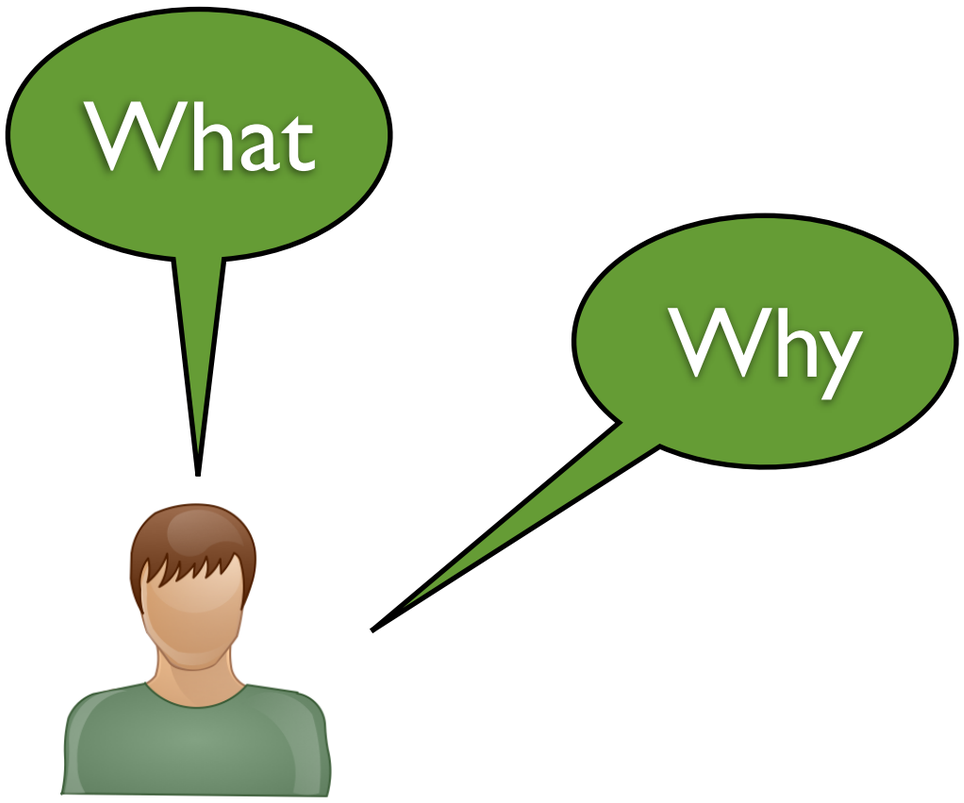
Roy T. Fielding

zolodeck

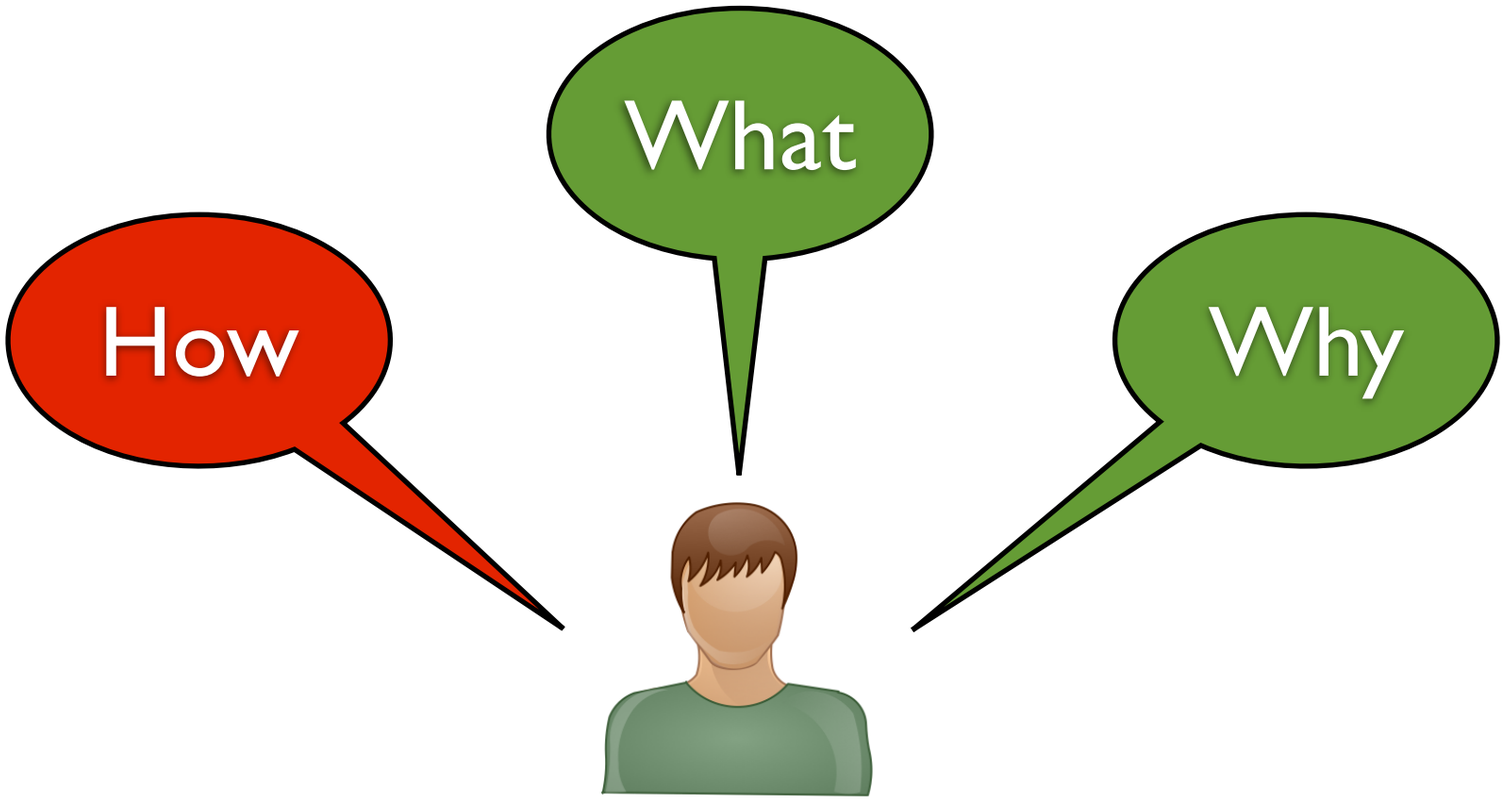
zolodeck



zolodeck



zolodeck



zolodeck

Who am I?

Siva Jagadeesan

Interests

- Clojure
- REST
- Datomic
- Continuous Deployment
- TDD
- Startups

About Me

- Founder & CTO, Zolo Labs Inc
- Director of Engineering, Runa Inc
- @sivajag
- siva@zololabs.com
- http://blog.zololabs.com
- http://techbehindtech.com

zolodeck

- first product from Zolo Labs
- Your digital assistant
- helps with professional networking and relationships
- *(and personal ones too!)*

zolodeck.com

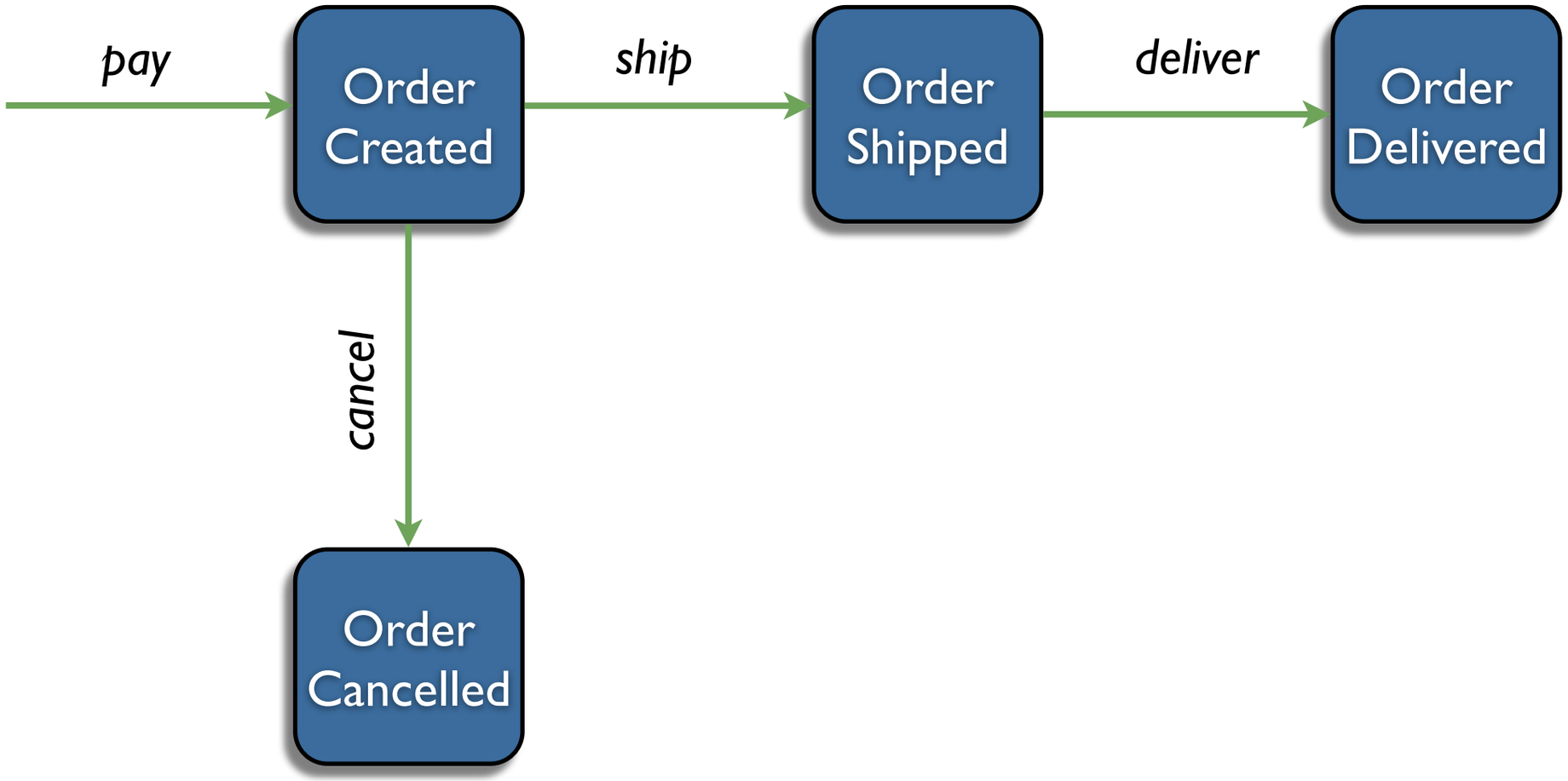
zolodeck

zolodeck

Example App

zolodeck

Order State Flow



zolodeck

OMS

zolodeck

Goals

- Maintainable
- Scalable
- Recoverable
- Loosely Coupled
- Secure

zolodeck

Goals

- Maintainable
- Scalable
- Recoverable
- Loosely Coupled
- ~~Secure~~

zolodeck

Lets build ...

zolodeck

API v1.0

- GET **/api**?action=create_order&products=p1,p2
- GET **/api**?action=list_orders
- GET **/api**?action=view_order&order_id=o1
- GET **/api**?action=cancel_order&order_id=o1

zolodeck

API v1.0 - core

```
(defroutes application-routes

  (GET "/api" [action & params] (web/json-response (api/perform action params)))

  (route/not-found "Page not found"))
```

zolodeck

API v1.0 - api

```
(defn perform [action params]
  (condp = (keyword action)
    :create_order (order/create-order params)
    :list_orders (order/find-all)
    :view_order (order/find-order (:order_id params))
    :cancel_order (order/delete (:order_id params))))
```

zolodeck

API v1.0
Use Cases

zolodeck

List Orders

```
(clj-http.client/get "http://localhost:4000/api" {:query-params {"action" "list_orders"}
                                                  :as :json})
```

```
{:body
 [{:status "PAID", :products ["p1" "p2" "p3"], :id "1"}
  {:status "DELIVERED", :products ["p3" "p6" "p9"], :id "3"}
  {:status "SHIPPED", :products ["p2" "p4" "p6"], :id "2"}],
 :status 200}
```

zolodeck

Find Existing Order

```
(clj-http.client/get "http://localhost:4000/api" {:query-params {"action" "view_order"
                                                                 "order_id" "1"}
                                                  :as :json})
```

```
{:body {:status "PAID", :products ["p1" "p2" "p3"], :id "1"},
 :status 200}
```

zolodeck

Cancel Order

```
(clj-http.client/get "http://localhost:4000/api" {:query-params {"action" "cancel_order"
                                                                 "order_id" "1"}
                                                  :as :json})
```

```
{:body {}, :status 200}
```

zolodeck

Find Cancelled Order

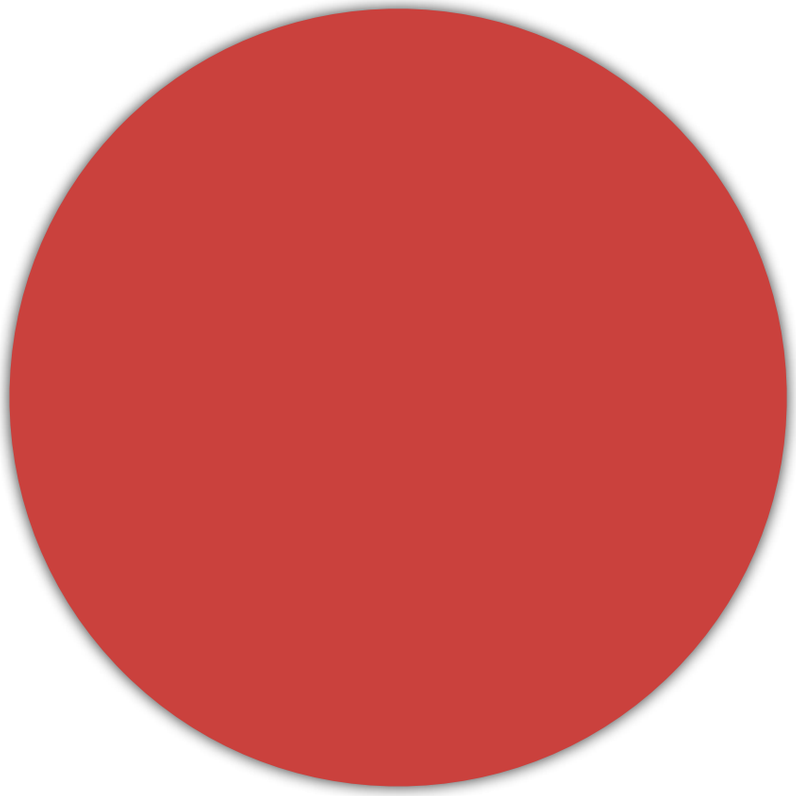
```
(clj-http.client/get "http://localhost:4000/api" {:query-params {"action" "view_order"
                                                                 "order_id" "1"}
                                                  :as :json})
```

```
{:body {:error "Order is not present with given ID"}, :status 200}
```

zolodeck

API v1.0
Goals Reached?

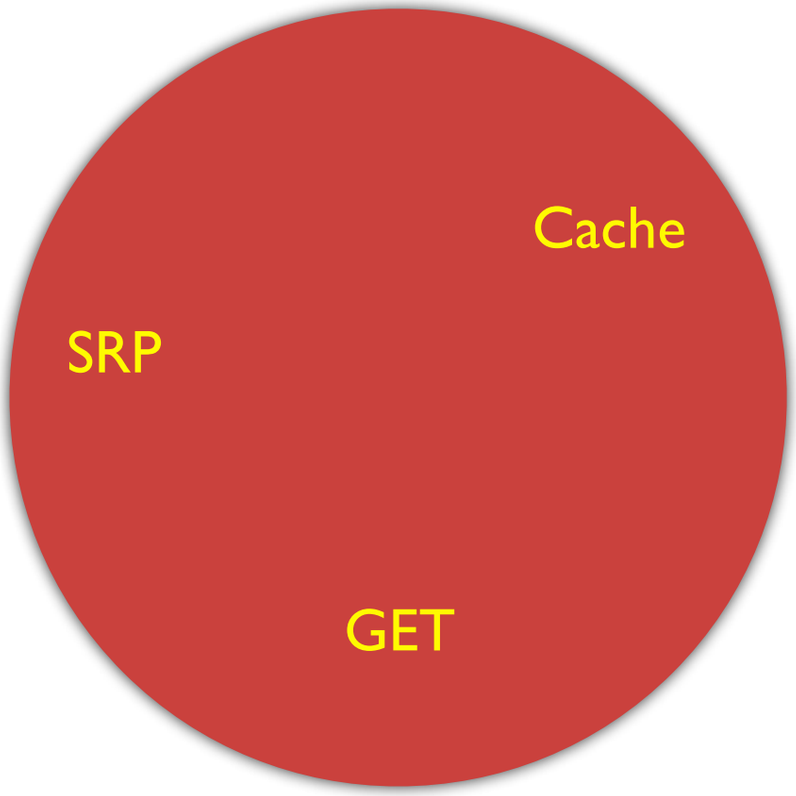
- Maintainable
- Scalable
- Recoverable
- Loosely Coupled



zolodeck

API v1.0
Goals Reached?

- Maintainable
- Scalable
- Recoverable
- Loosely Coupled



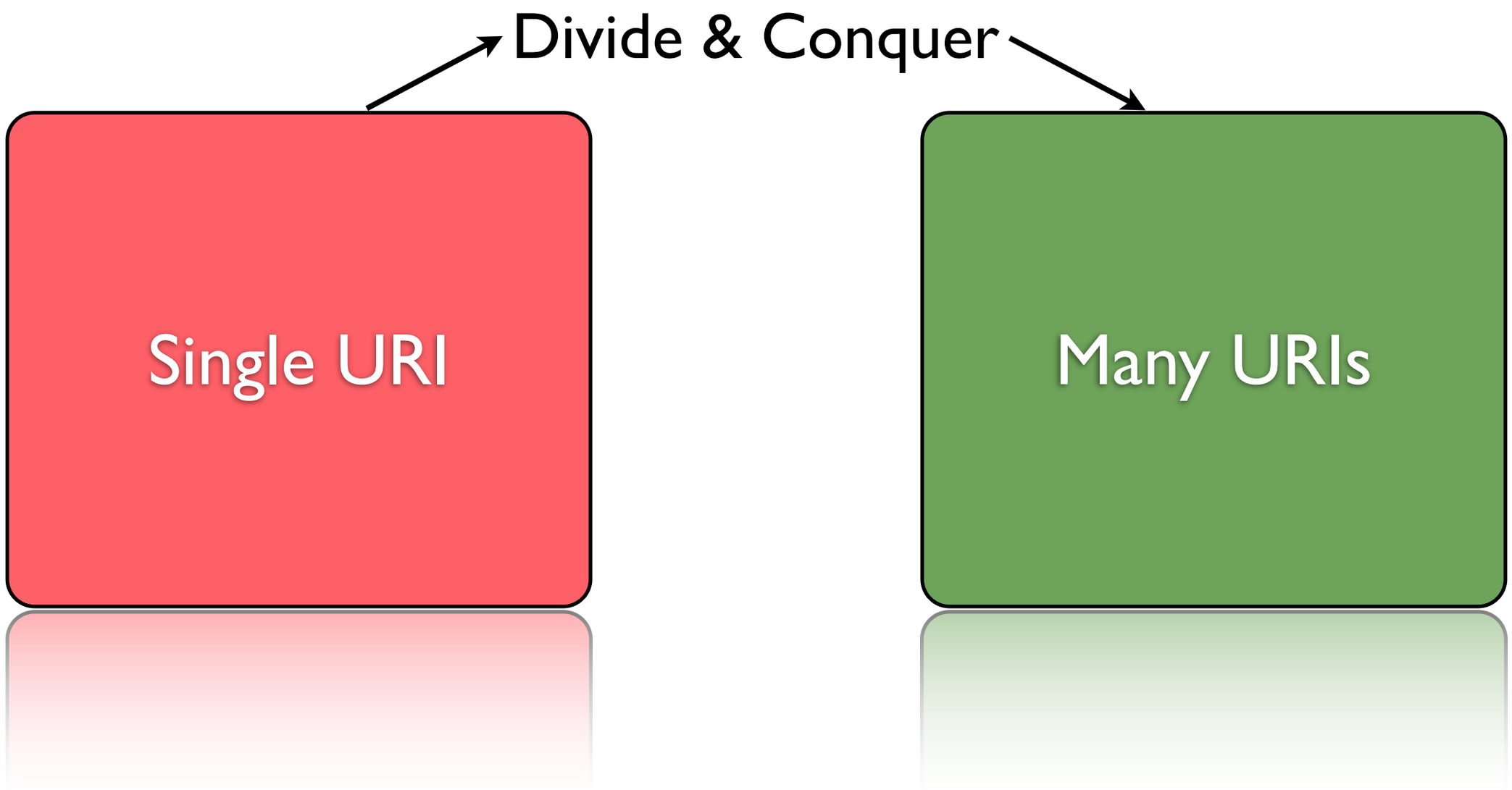
zolodeck

Improving API v1.0

Single URI

zolodeck

Improving API v1.0



zolodeck

zolodeck

Lets build v2.0

zolodeck

API v2.0

- GET **/create_order**?products=p1,p2,p3
- GET **/list_orders**
- GET **/view_order**?order_id=o1
- GET **/cancel_order**?order_id=o1

zolodeck

API v2.0 - core

```
(defroutes application-routes

  (GET "/create_order" [params] (web/json-response (api/create-order params)))

  (GET "/list_orders" [params] (web/json-response (api/list-orders)))

  (GET "/view_order/:order-id" [order-id] (web/json-response (api/view-order order-id)))

  (GET "/cancel_order/:order-id" [order-id] (web/json-response (api/cancel-order order-id)))

  (route/not-found "Page not found"))
```

zolodeck

API v2.0 - api

```
(defn create-order [params]
  (order/create-order params))

(defn list-orders []
  (order/find-all))

(defn view-order [id]
  (order/find-order id))

(defn cancel-order [id]
  (order/delete id))
```

zolodeck

API v2.0
Use Cases

zolodeck

Find Existing Order

```
(clj-http.client/get "http://localhost:4000/view_order/1" {:as :json})
```

```
{:body {:status "PAID", :products ["p1" "p2" "p3"], :id "1"},
 :status 200}
```

zolodeck

Cancel Order

```
(clj-http.client/get "http://localhost:4000/cancel_order/1" {:as :json})
```

```
{:body {}, :status 200}
```

zolodeck

Find Cancelled Order

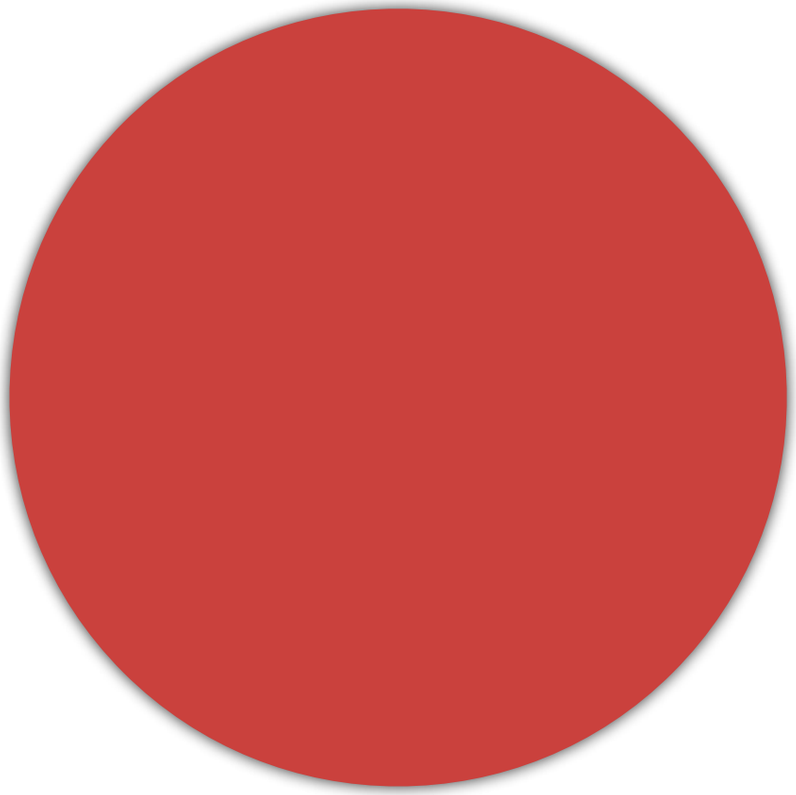
```
(clj-http.client/get "http://localhost:4000/view_order/1" {:as :json})
```

```
{:body {:error "Order is not present with given ID"}, :status 200}
```

zolodeck

API v2.0
Goals Reached?

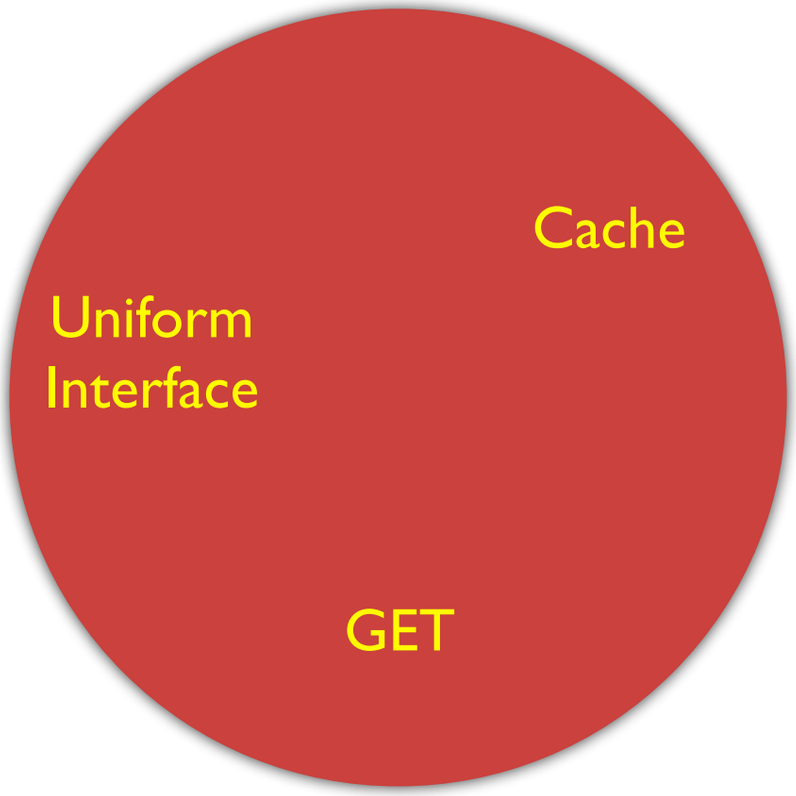
- Maintainable
- Scalable
- Recoverable
- Loosely Coupled



zolodeck

API v2.0
Goals Reached?

- Maintainable
- Scalable
- Recoverable
- Loosely Coupled



zolodeck

Find Cancelled Order

```
(clj-http.client/get "http://localhost:4000/view_order/1" {:as :json})
```

```
{:body {:error "Order is not present with given ID"}, :status 200}
```

zolodeck

Cancel Order

```
(clj-http.client/get "http://localhost:4000/cancel_order/1" {:as :json})
```

```
{:body {}, :status 200}
```

zolodeck

Improving API v2.0

URI encode
Operations

No uniform interface

zolodeck

Improving API v2.0

Do the same things
the same way

URI encode
Operations

No uniform interface

URIs

HTTP Status code

HTTP verbs

zolodeck

zolodeck

Lets build v3.0

zolodeck

API v3.0

- POST **/orders**
- GET **/orders**
- GET **/orders**/{order_id}
- DELETE **/orders**/{order_id}

zolodeck

HTTP Status Codes

1xx - Metadata

2xx – Success

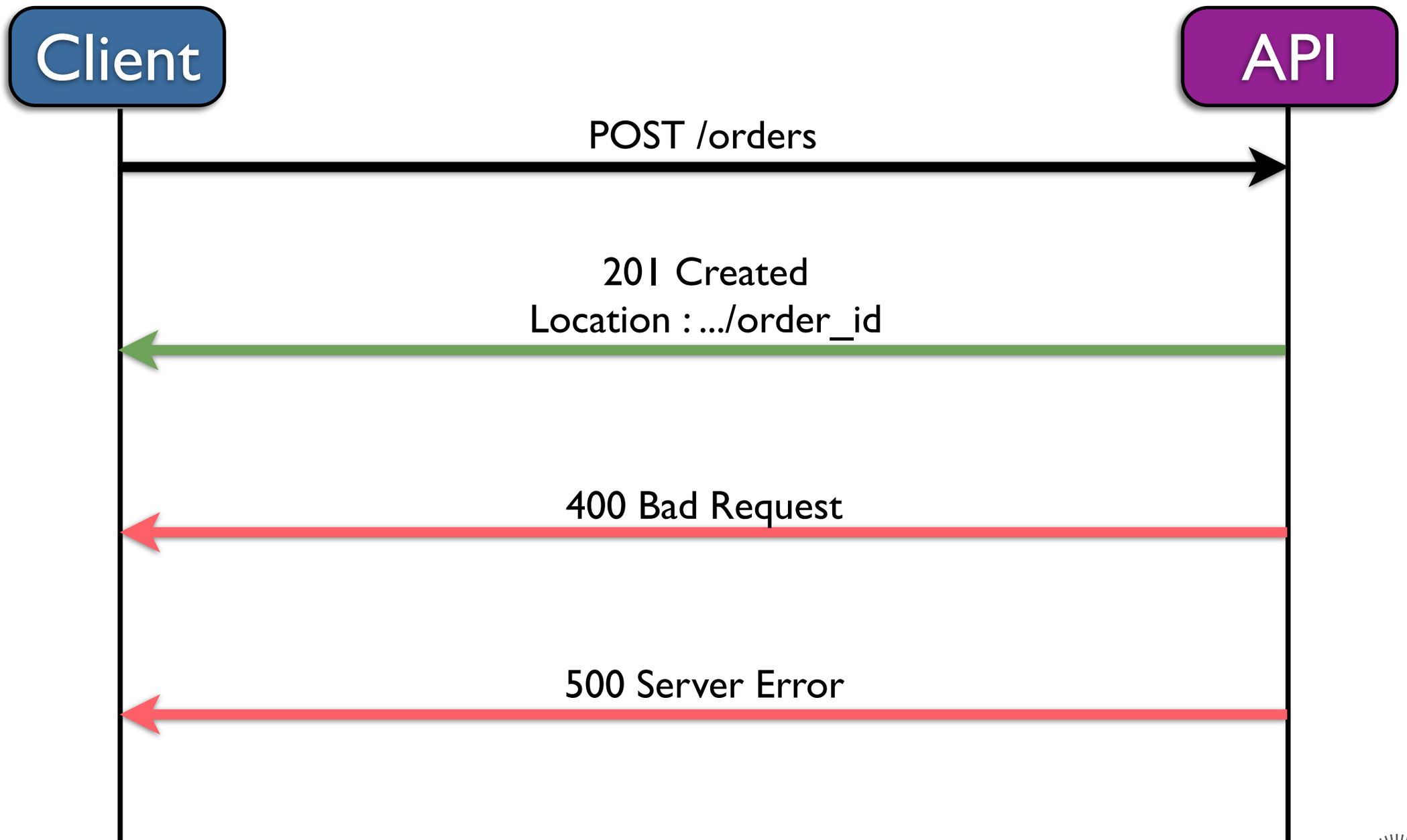
3xx – Redirection

4xx – Client error

5xx – Server error

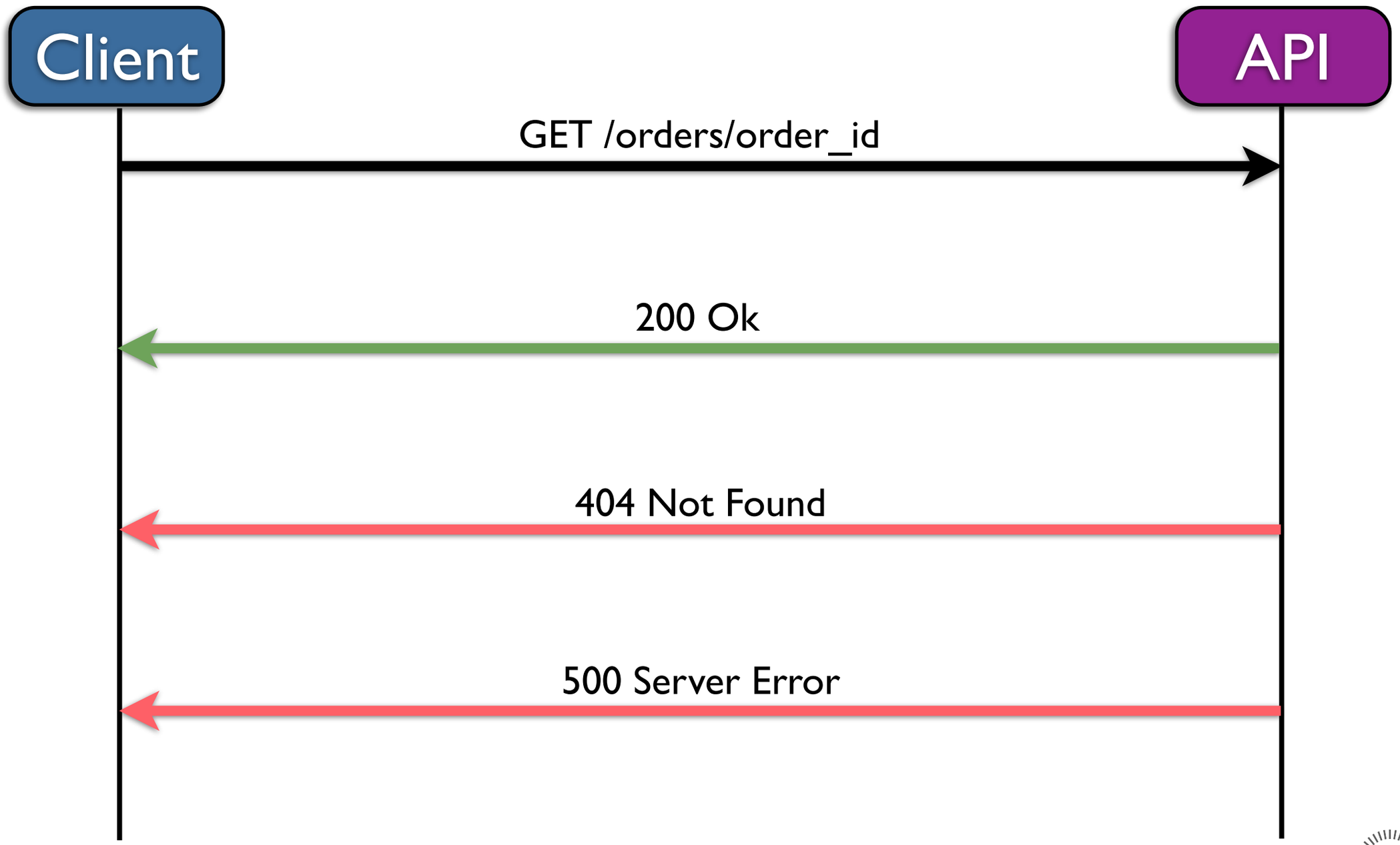
zolodeck

HTTP verbs
Create with POST



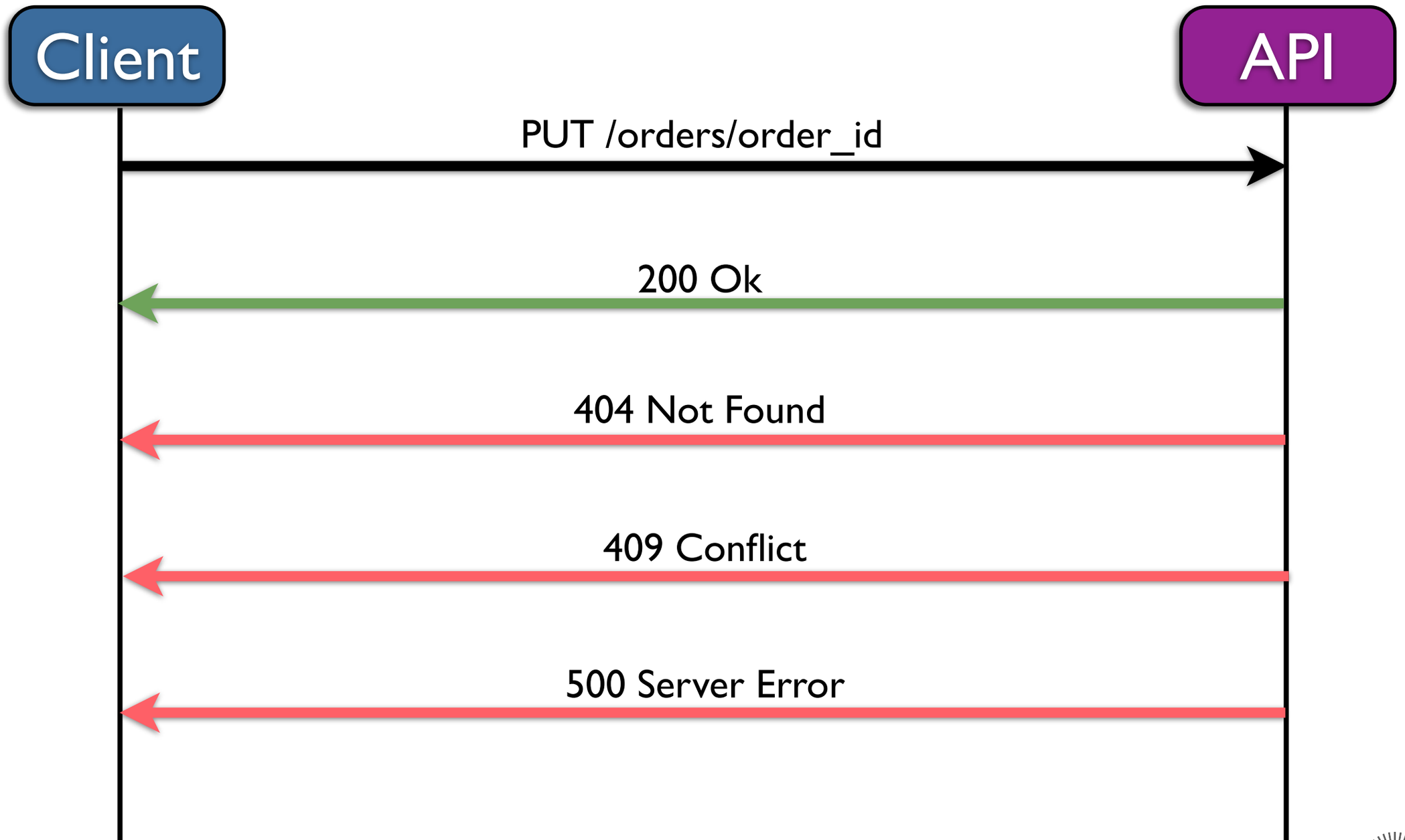
zolodeck

HTTP verbs
Read with GET



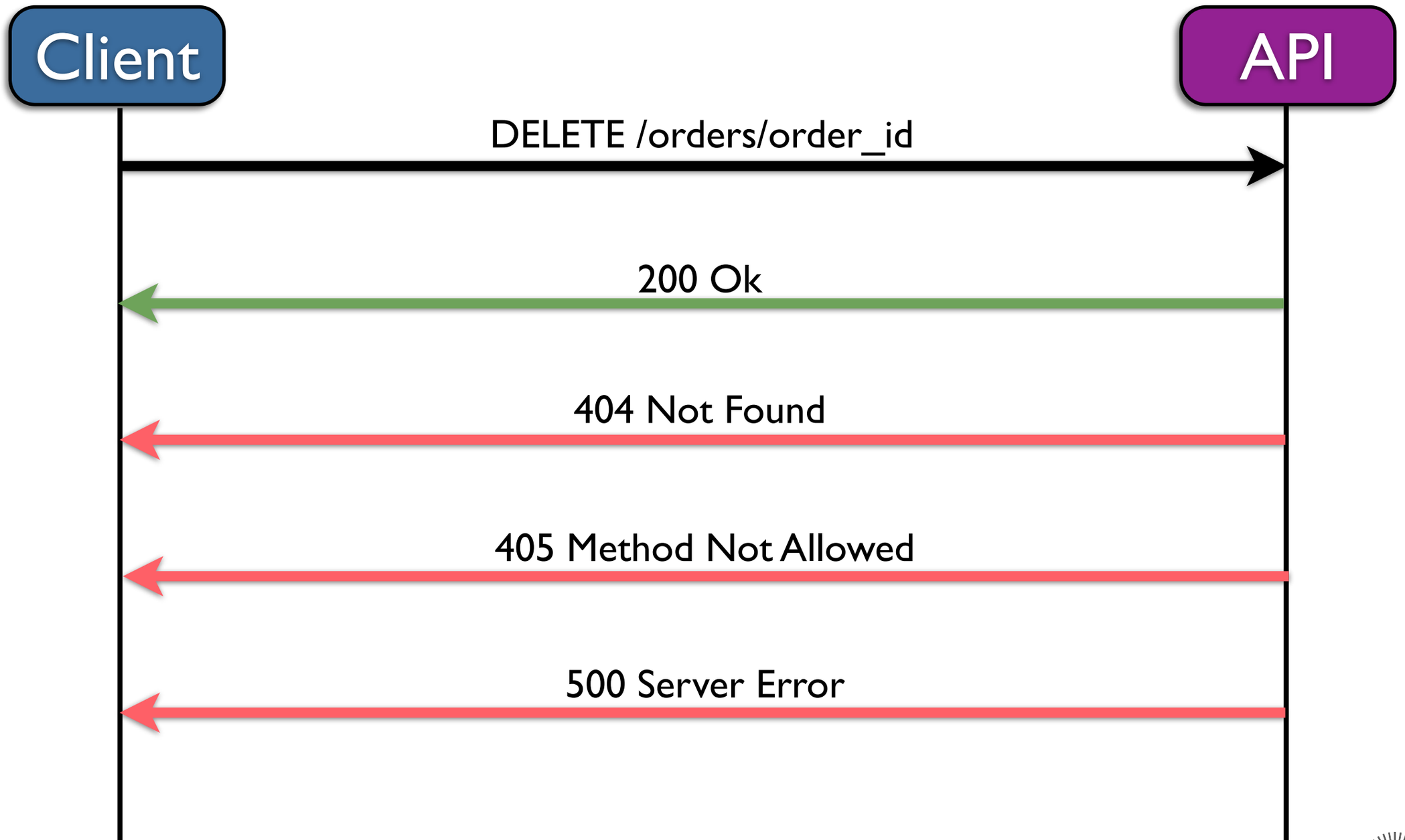
zolodeck

HTTP verbs
Update with PUT



zolodeck

HTTP verbs
Remove with DELETE



zolodeck

API v3.0 - core

```
(defroutes application-routes
  (POST "/orders" [params] (api/create-order params))
  (GET "/orders" [params] (web/json-response (api/list-orders)))
  (GET "/orders/:order-id" [order-id] (web/json-response (api/view-order order-id)))
  (DELETE "/orders/:order-id" [order-id] (web/json-response (api/cancel-order order-id)))
  (route/not-found "Page not found"))
```

zolodeck

API v3.0 - middleware

```
(defn wrap-error-handling [handler]
  (f [request]
    (try+
     (handler request)
     (catch [:type :not-found] e
       (error-response e 404))
     (catch [:type :bad-request] e
       (error-response e 400))
     (catch [:type :method-not-allowed] e
       (error-response e 405))
     (catch Exception e
       (json-response {:error (.getMessage e)} 500))))))
```

zolodeck

API v3.0 - api

```
(defn create-order [params]
  (let [o (order/create-order params)]
    {:status 201
     :headers {"Content-Type" "application/json; charset=utf-8"
               "Location " (order-url o)}
     :body {}}))

(defn list-orders []
  (order/find-all))

(defn view-order [id]
  (order/find-order id))

(defn cancel-order [id]
  (order/delete id))
```

zolodeck

Clojure Libraries

- liberator
- bishop
- https://github.com/zololabs/

API v3.0
Use Cases

zolodeck

Find Existing Order

```
(clj-http.client/get "http://localhost:4000/orders/1" {:as :json})
```

```
{:body {:status "PAID", :products ["p1" "p2" "p3"], :id "1"},
 :status 200}
```

zolodeck

Cancel Order

```
(clj-http.client/delete "http://localhost:4000/orders/1" {:as :json})
```

```
{:body {}, :status 200}
```

zolodeck

Find Cancelled Order

```
(clj-http.client/get "http://localhost:4000/orders/1" {:as :json})
```

```
{:body "{\"error\":\"Order is not present with given ID\"}",
 :status 404}
```

zolodeck

Cancel Shipped Order

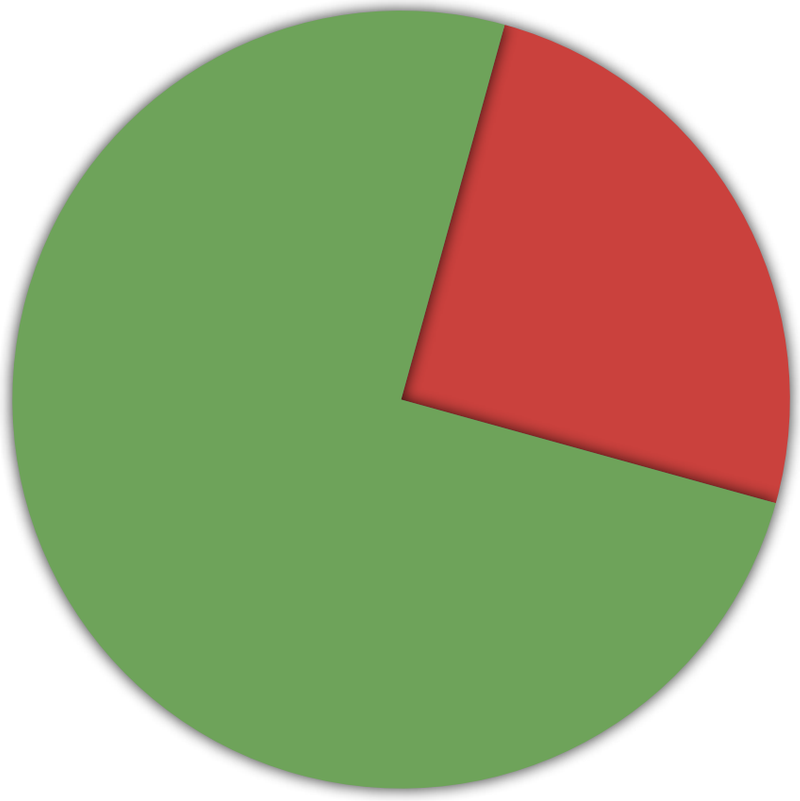
```
(clj-http.client/delete "http://localhost:4000/orders/2" {:as :json})
```

```
{:body "{\"error\":\"Order is already SHIPPED\"}", :status 405}
```

zolodeck

API v3.0
Goals Reached?

- Maintainable
- Scalable
- Recoverable
- Loosely Coupled



zolodeck

API v3.0

Is this RESTful?

- POST /orders
- GET /orders
- GET /orders/{order_id}
- DELETE /orders/{order_id}

zolodeck

API v3.0

Is this RESTful?

- POST /orders
- GET /or
- GET /or
- DELETE /orders/{order_id}

NO!

zolodeck

I don't care about buzzwords

API v3 is a good design

I am here to provide business value

zolodeck

The Other Side

zolodeck

TimeLine

zolodeck

TimeLine

API v1.0

zolodeck

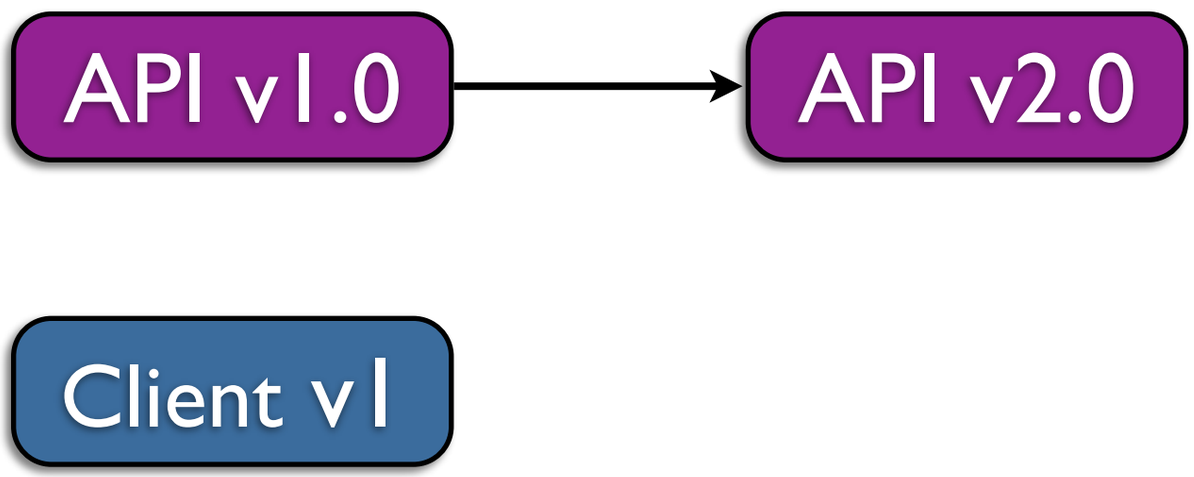
TimeLine

API v1.0

Client v1

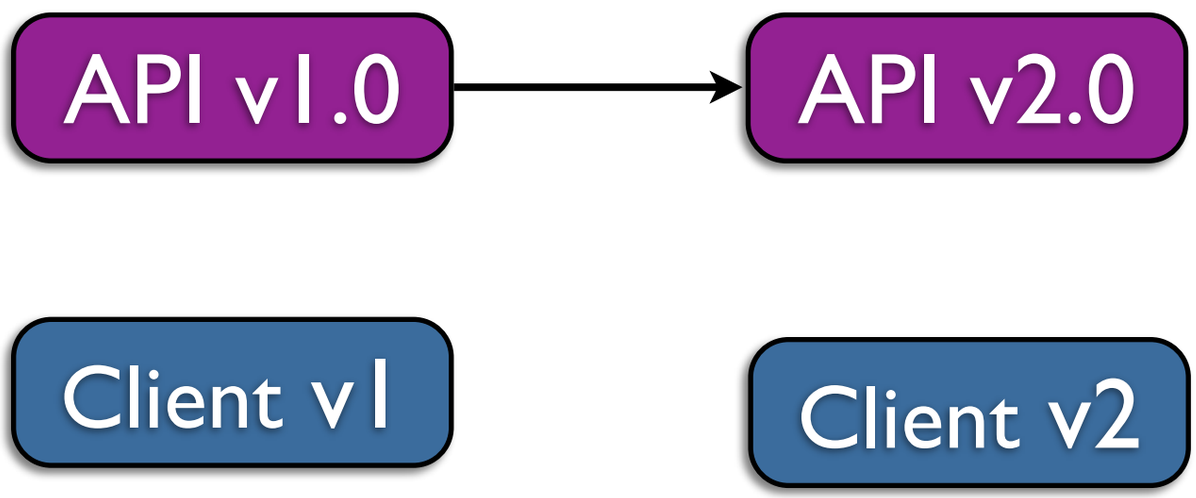
zolodeck

TimeLine



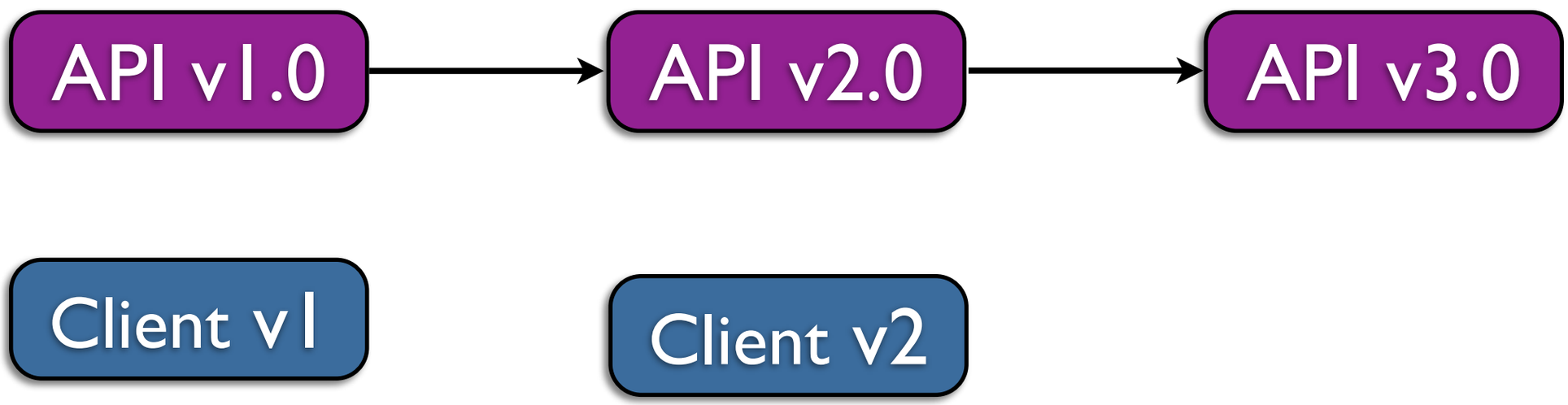
zolodeck

TimeLine



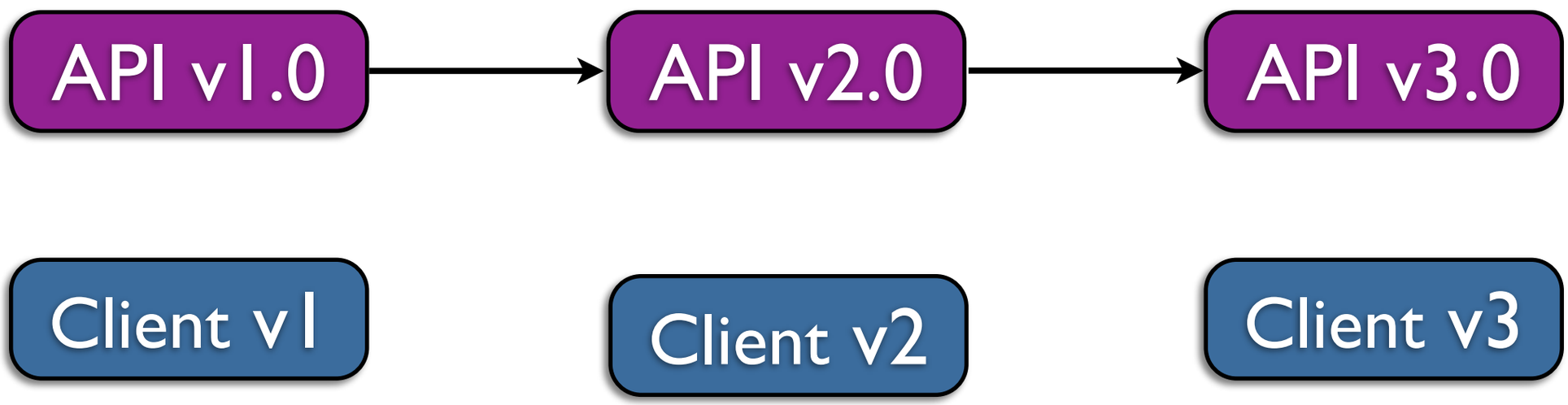
zolodeck

TimeLine



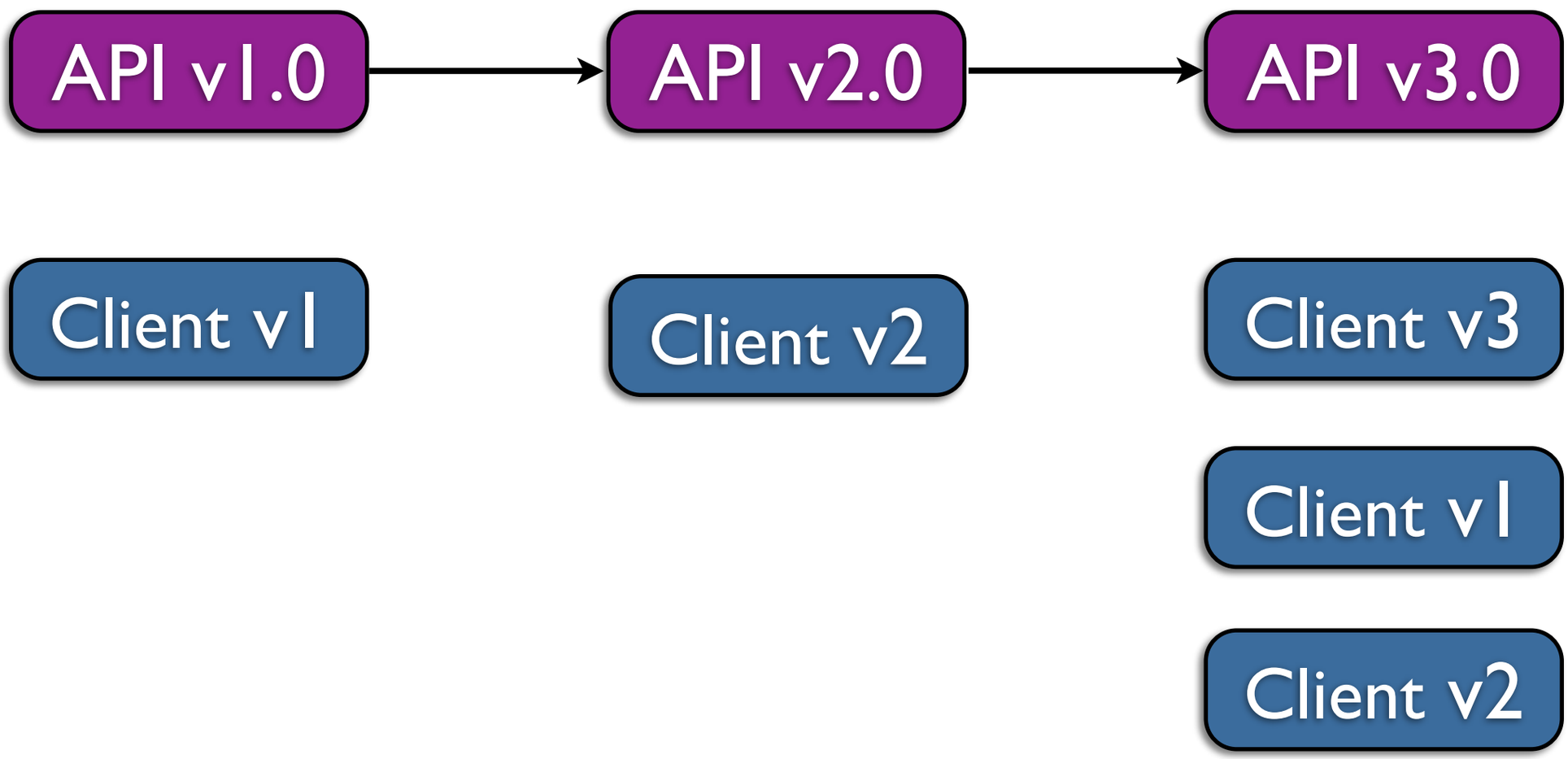
zolodeck

TimeLine



zolodeck

TimeLine



zolodeck

API v3.0

- POST **/orders**
- GET **/orders**
- GET **/orders**/{order_id}
- DELETE **/orders**/{order_id}

zolodeck

API v3.0

- GET /api?action=create_order&products=p1,p2,p3
- GET /api?action=list_orders
- GET /api?action=view_order&order_id=o1
- GET /api?action=cancel_order&order_id=o1
- GET /create_order?products=p1,p2,p3
- GET /list_orders
- GET /view_order?order_id=o1
- GET /cancel_order?order_id=o1
- POST /orders
- GET /orders
- GET /orders/{order_id}
- DELETE /orders/{order_id}

zolodeck

This is a dumb argument, Siva

We would have started in v3.0 ... so we won't have this problem

zolodeck

This is a dumb argument, Siva

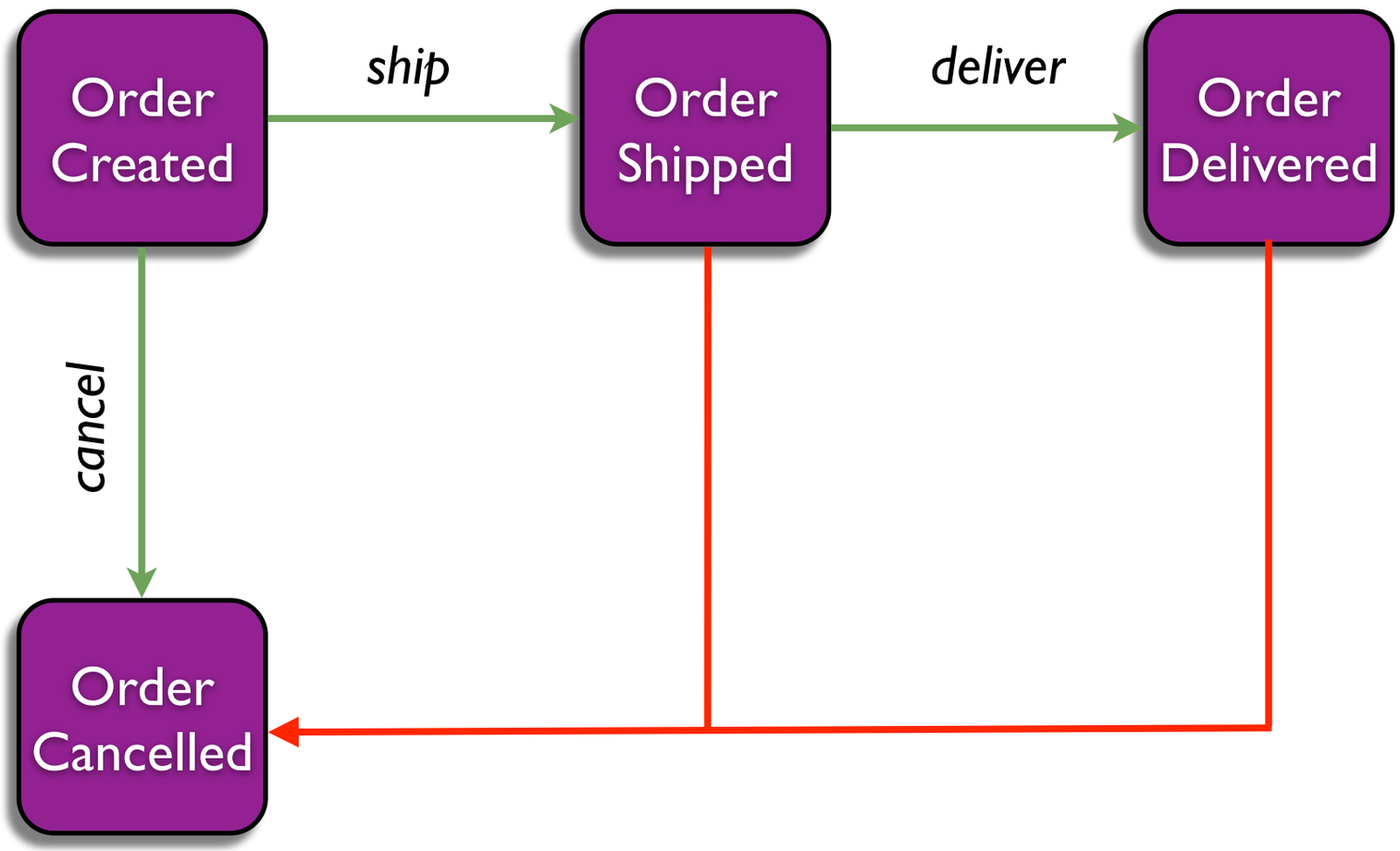
You still can't evolve this API

We would have started in v3.0 ... so we won't have this problem

zolodeck

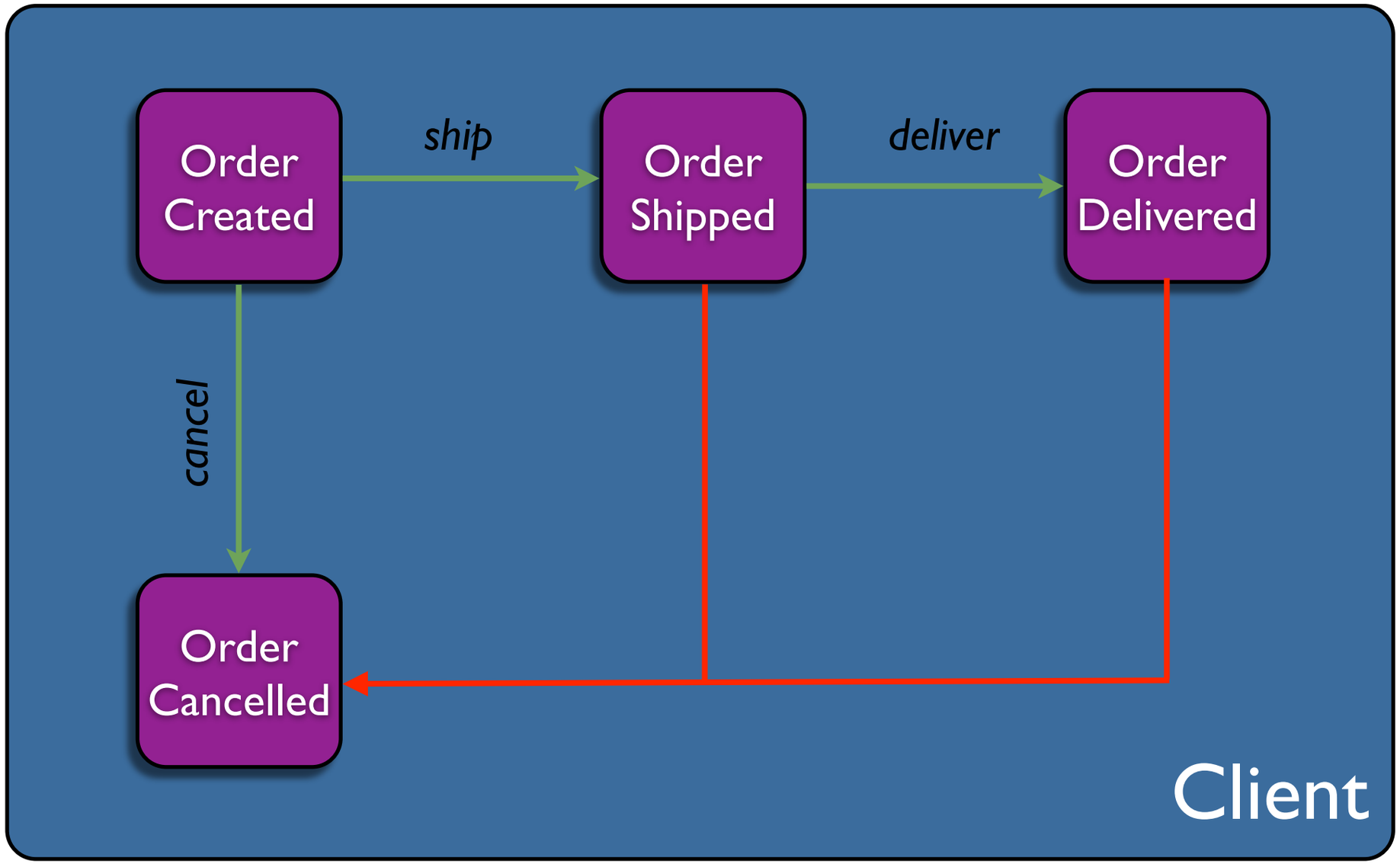
zolodeck

Cancel Order Logic



zolodeck

Cancel Order Logic

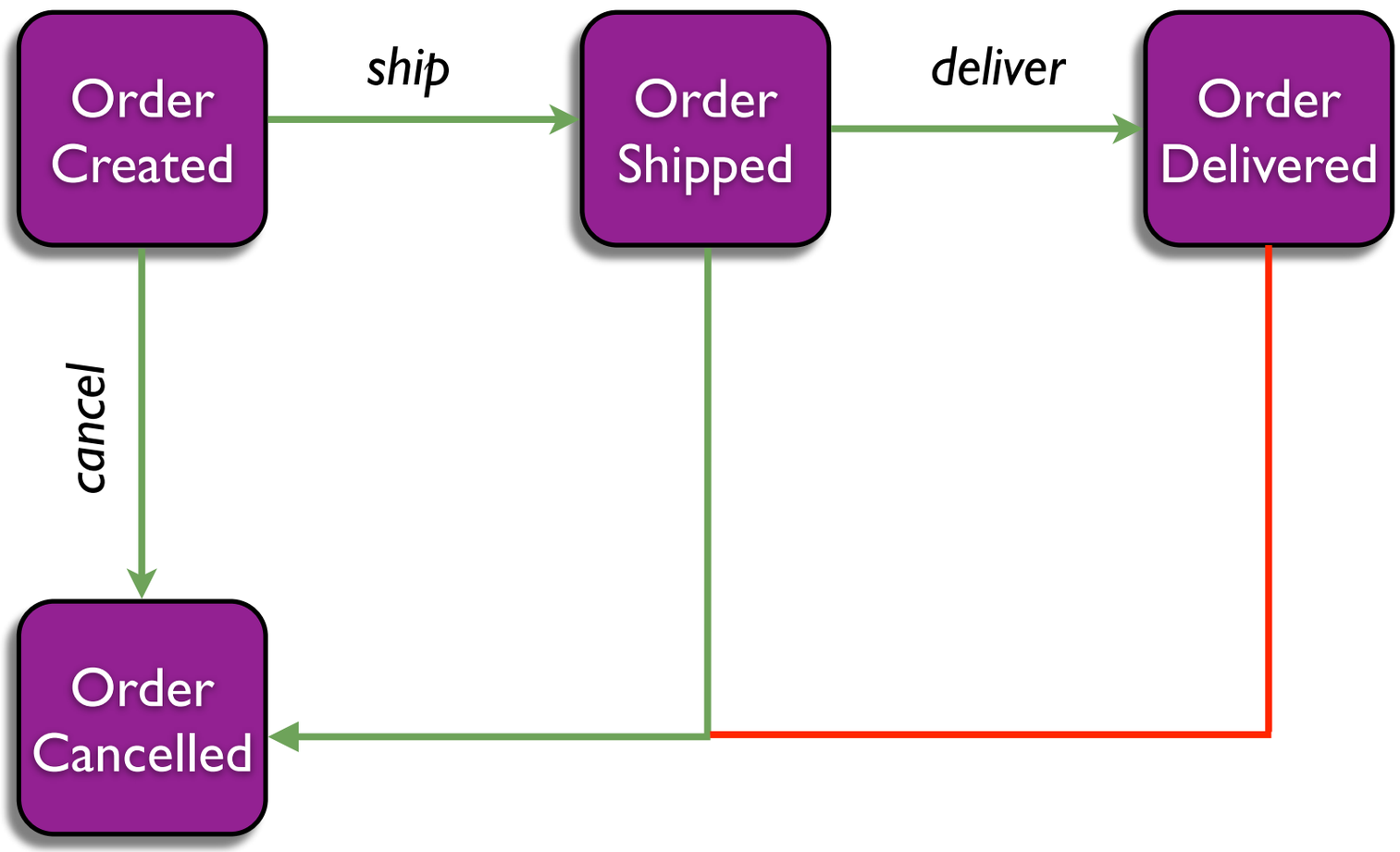


zolodeck

zolodeck

New

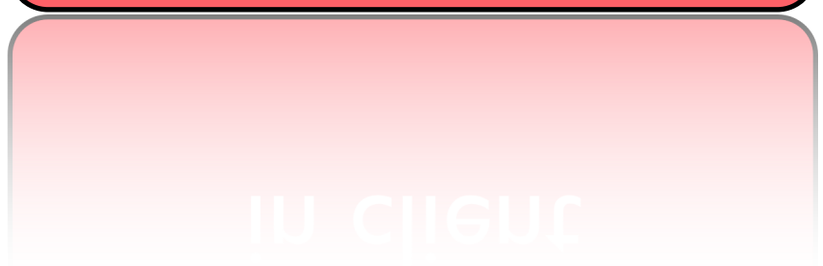
Cancel Order Logic



zolodeck

Improving API v3.0

Application state transition logic is in client



zolodeck

Improving API v3.0

Describe special behavior
in a standard way

Application state transition logic is in client

Send valid transitions to clients for a current resource

zolodeck

Lets Build API v4.0

zolodeck

API v4.0

no change from v3.0

- POST **/orders**
- GET **/orders**
- GET **/orders**/{order_id}
- DELETE **/orders**/{order_id}

zolodeck

v4.0 vs v3.0

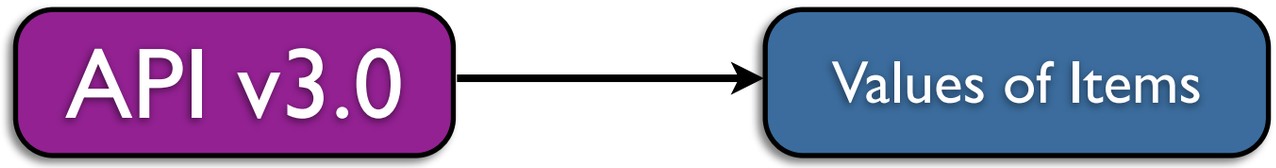
zolodeck

v4.0 vs v3.0

API v3.0

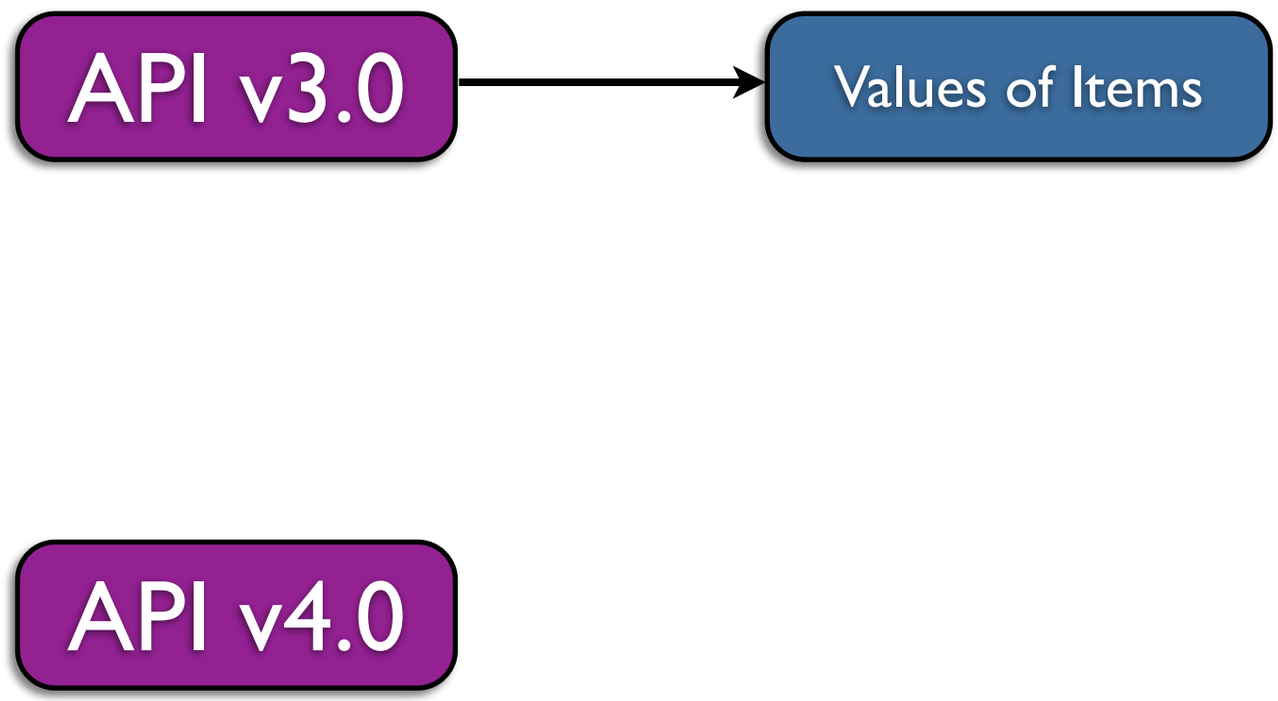
zolodeck

v4.0 vs v3.0



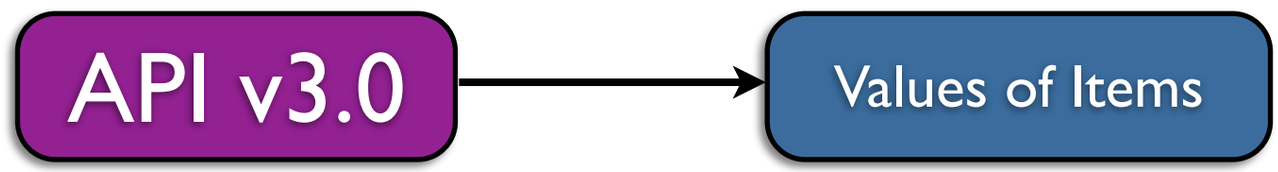
zolodeck

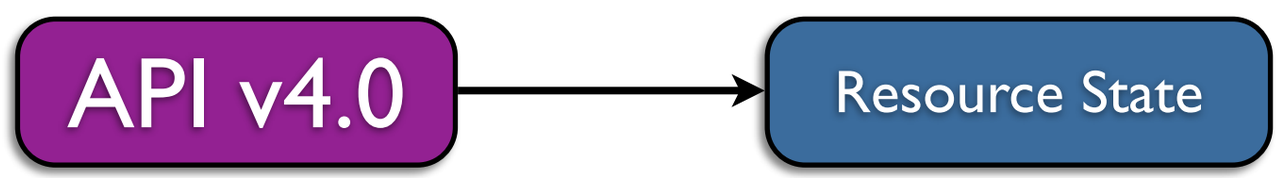
v4.0 vs v3.0



zolodeck

v4.0 vs v3.0





zolodeck

Resource State

Values of Items

Links - Related Resources

Links - Transitions

zolodeck

API v4.0
Use Cases

zolodeck

Find Existing Order

```
(clj-http.client/get "http://localhost:4000/orders/1" {:as :json})
```

```
{:status 200,
 :body
 {:status "PAID",
  :products ["p1" "p2" "p3"],
  :links
  [{:href "http://localhost:4000/orders", :rel "self"}
   {:href "http://localhost:4000/orders", :rel "cancel"}]
  :id "1"}}
```

zolodeck

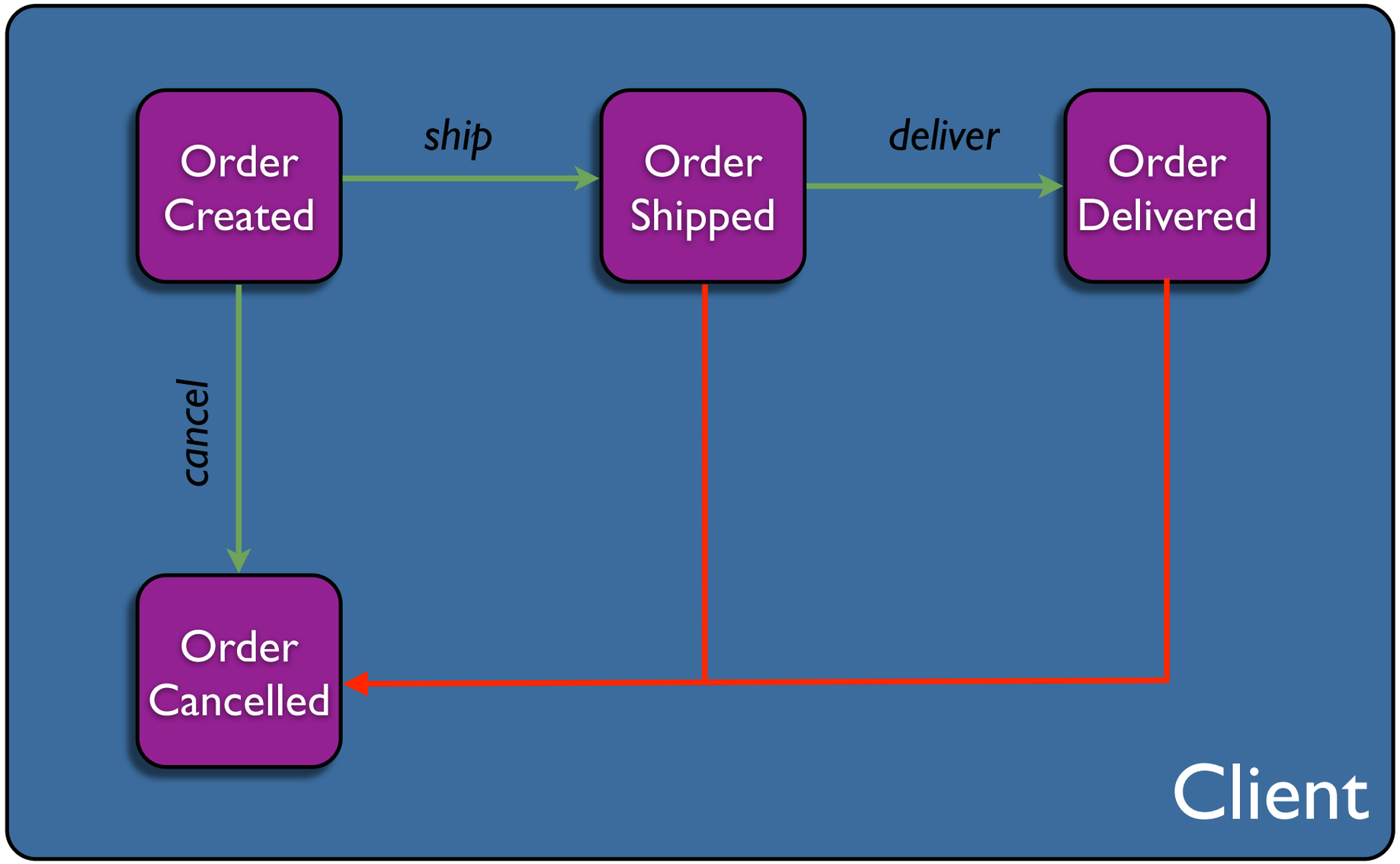
Find Shipped Order

```
(clj-http.client/get "http://localhost:4000/orders/1" {:as :json})
```

```
{:status 200,
 :body
 {:status "SHIPPED",
  :products ["p2" "p4" "p6"],
  :links [{:href "http://localhost:4000/orders", :rel "self"}],
  :id "2"}}
```

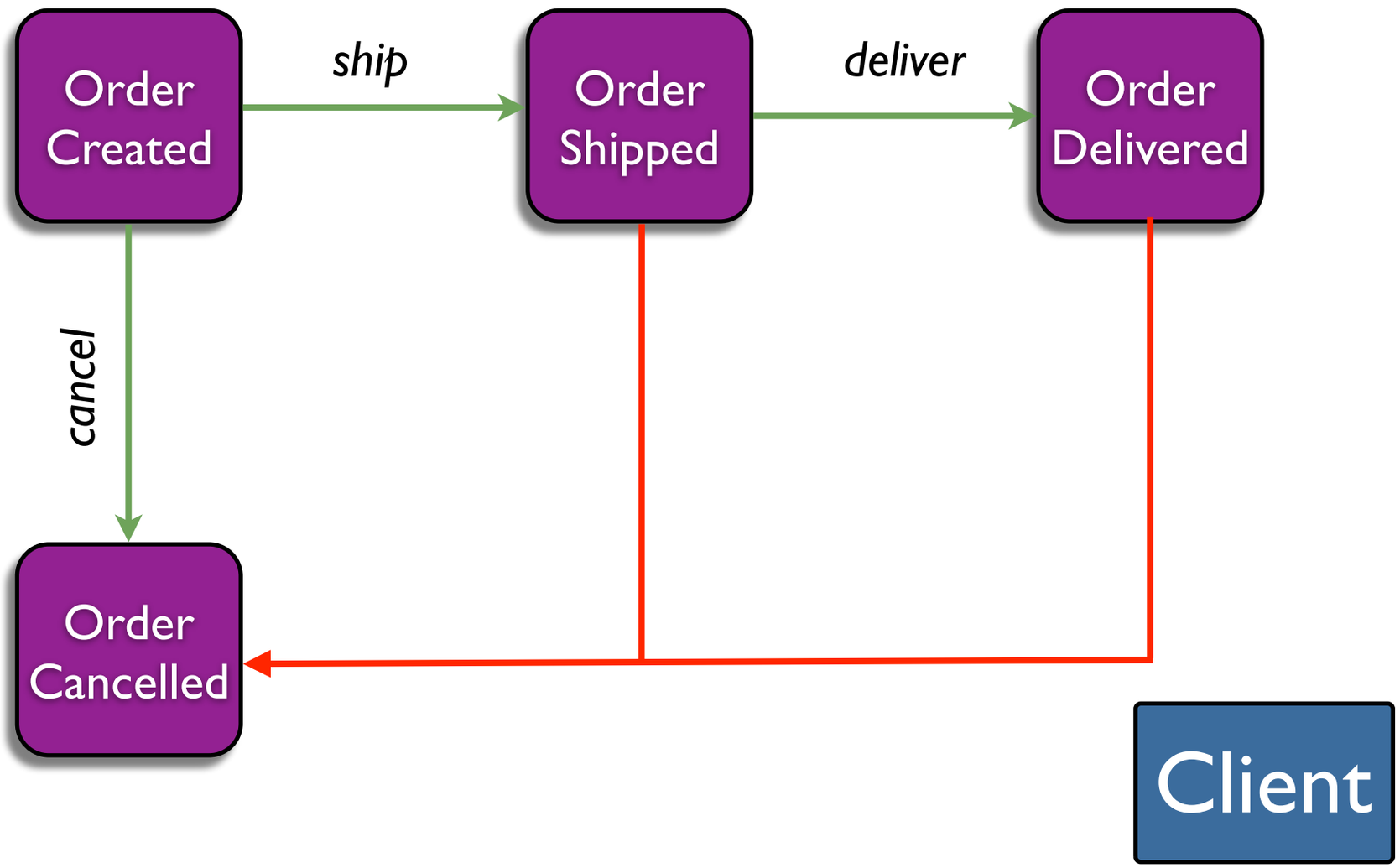
zolodeck

Cancel Order Logic



zolodeck

Cancel Order Logic



zolodeck

API v4.0
Goals Reached?

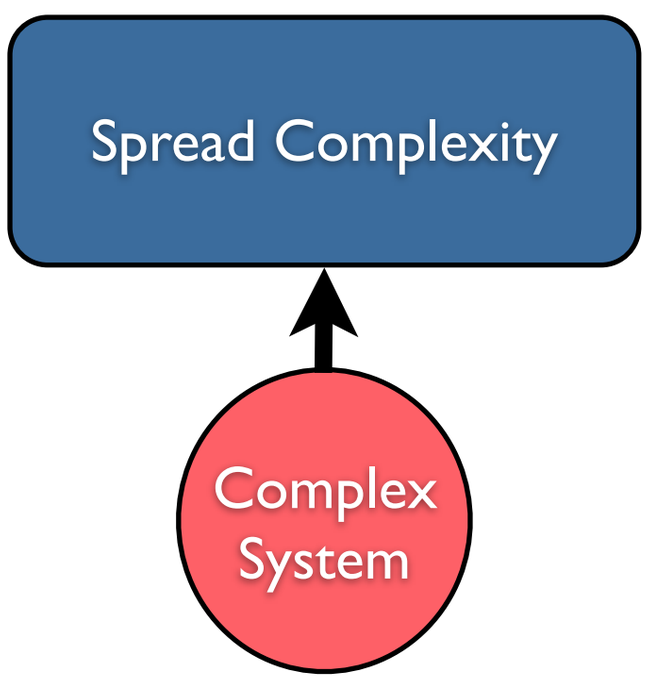
- Maintainable
- Scalable
- Recoverable
- Loosely Coupled

zolodeck

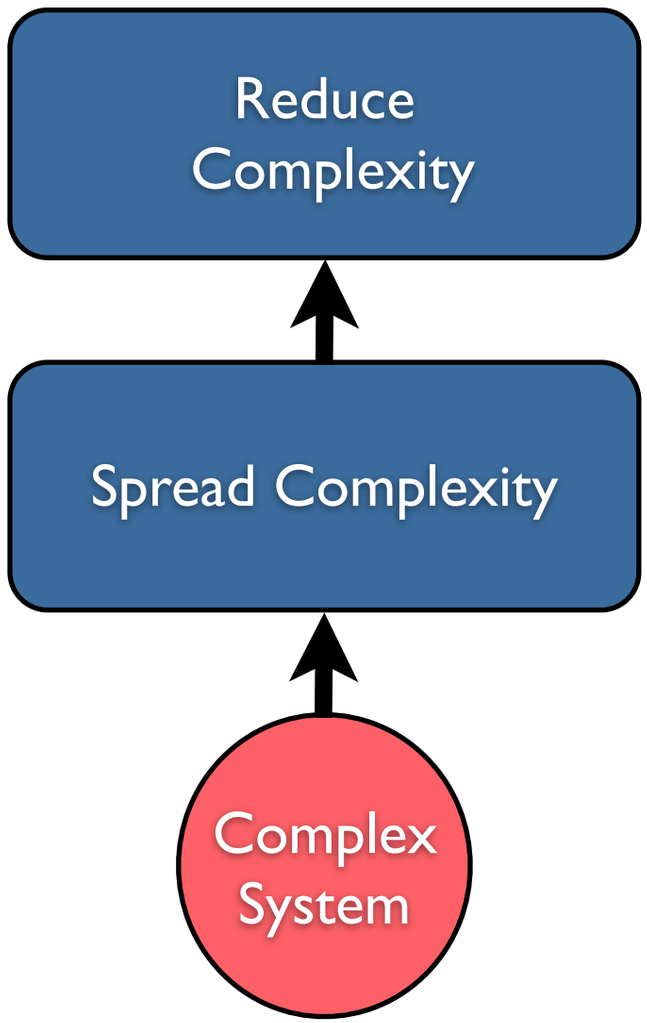
zolodeck

Complex System

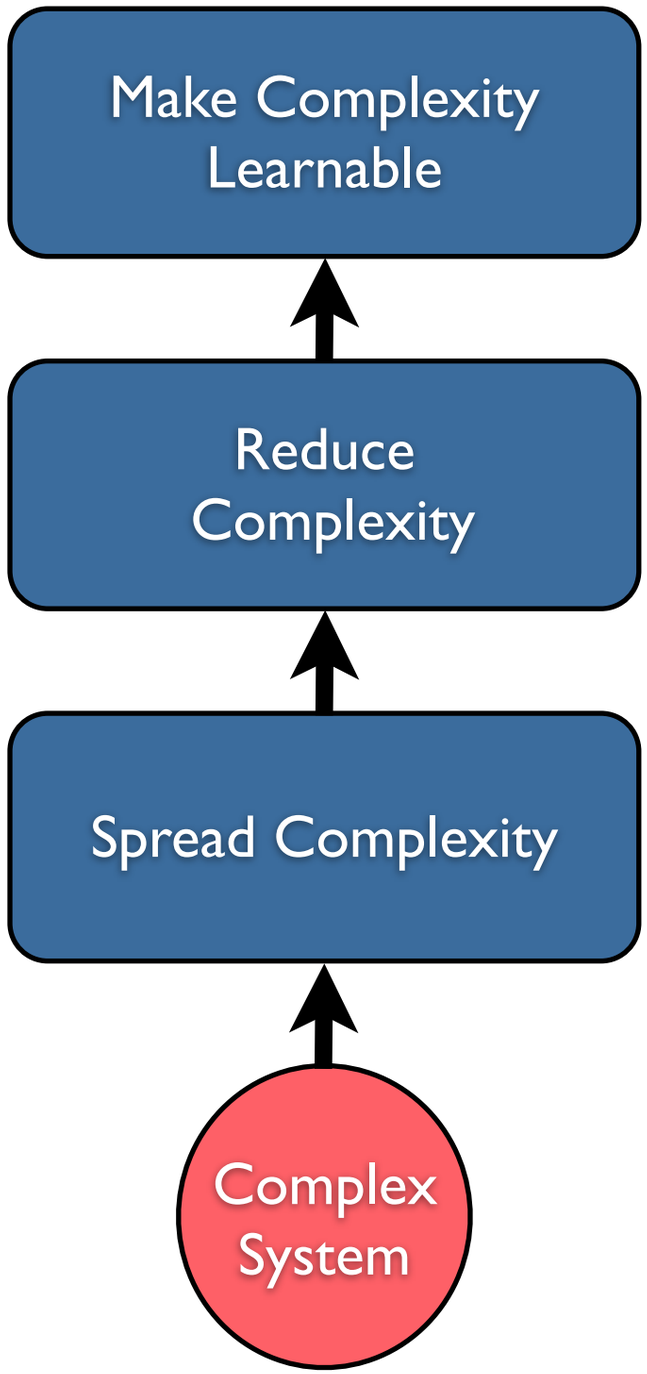
zolodeck



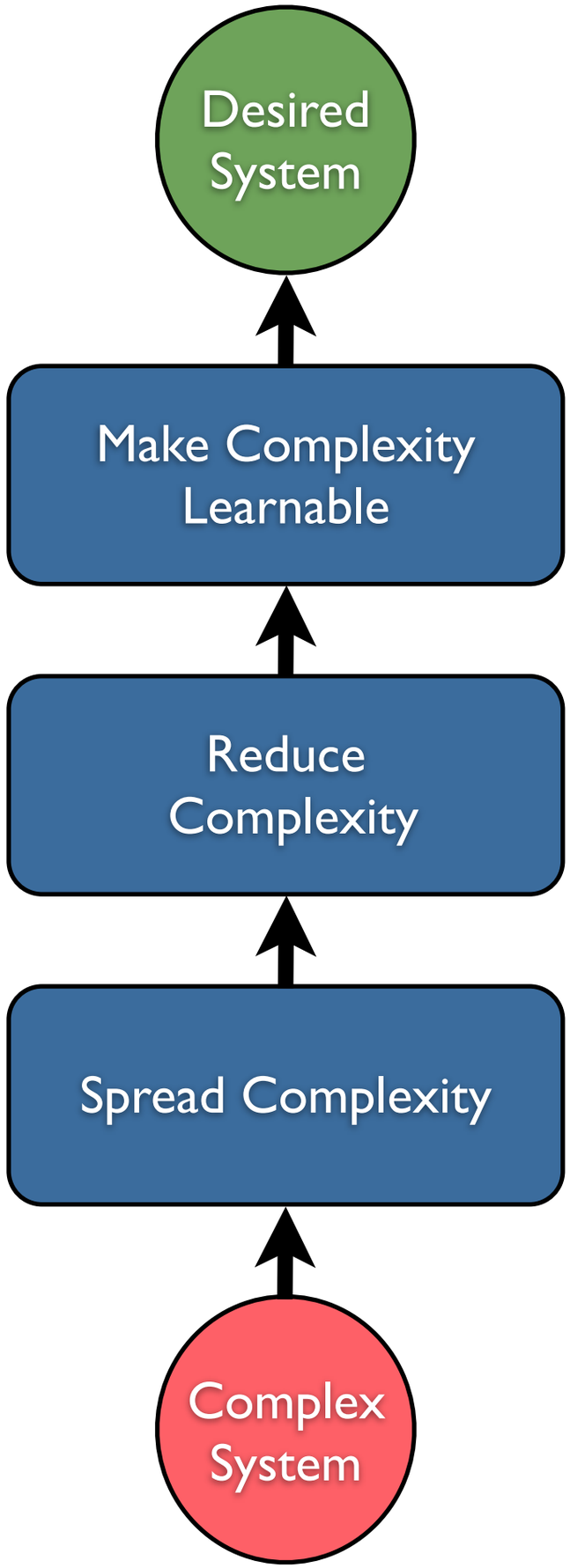
zolodeck



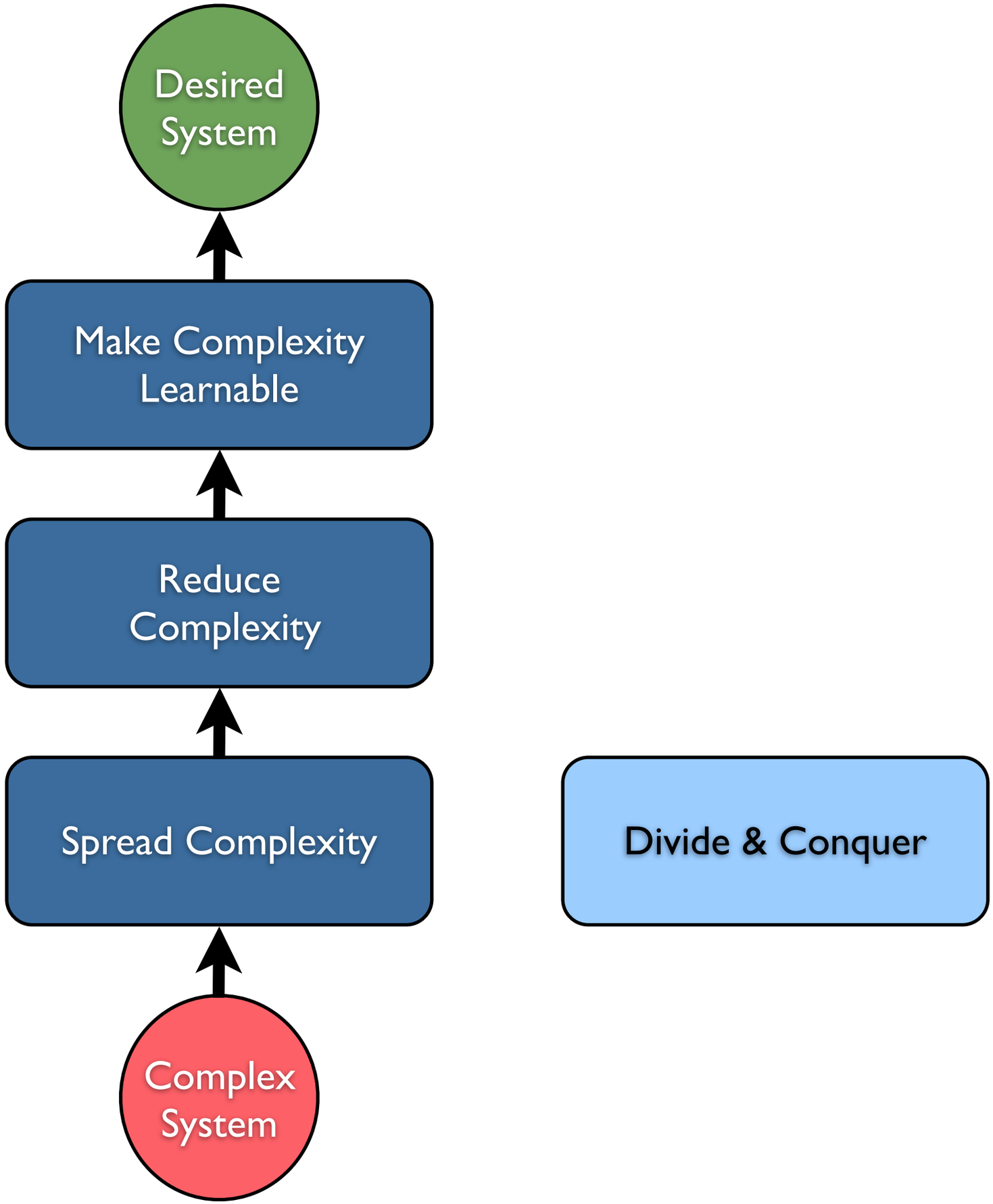
zolodeck


Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System

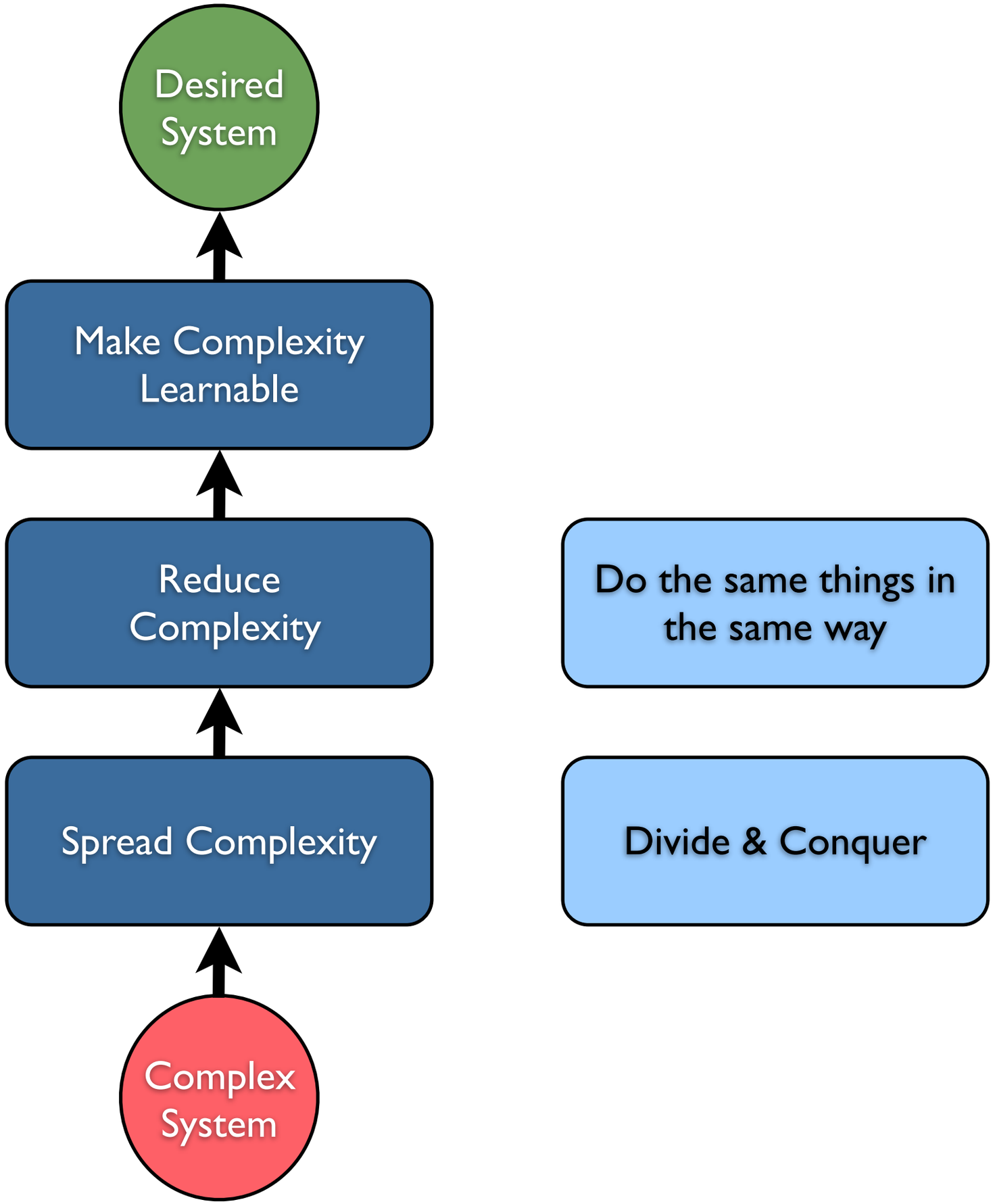
zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System

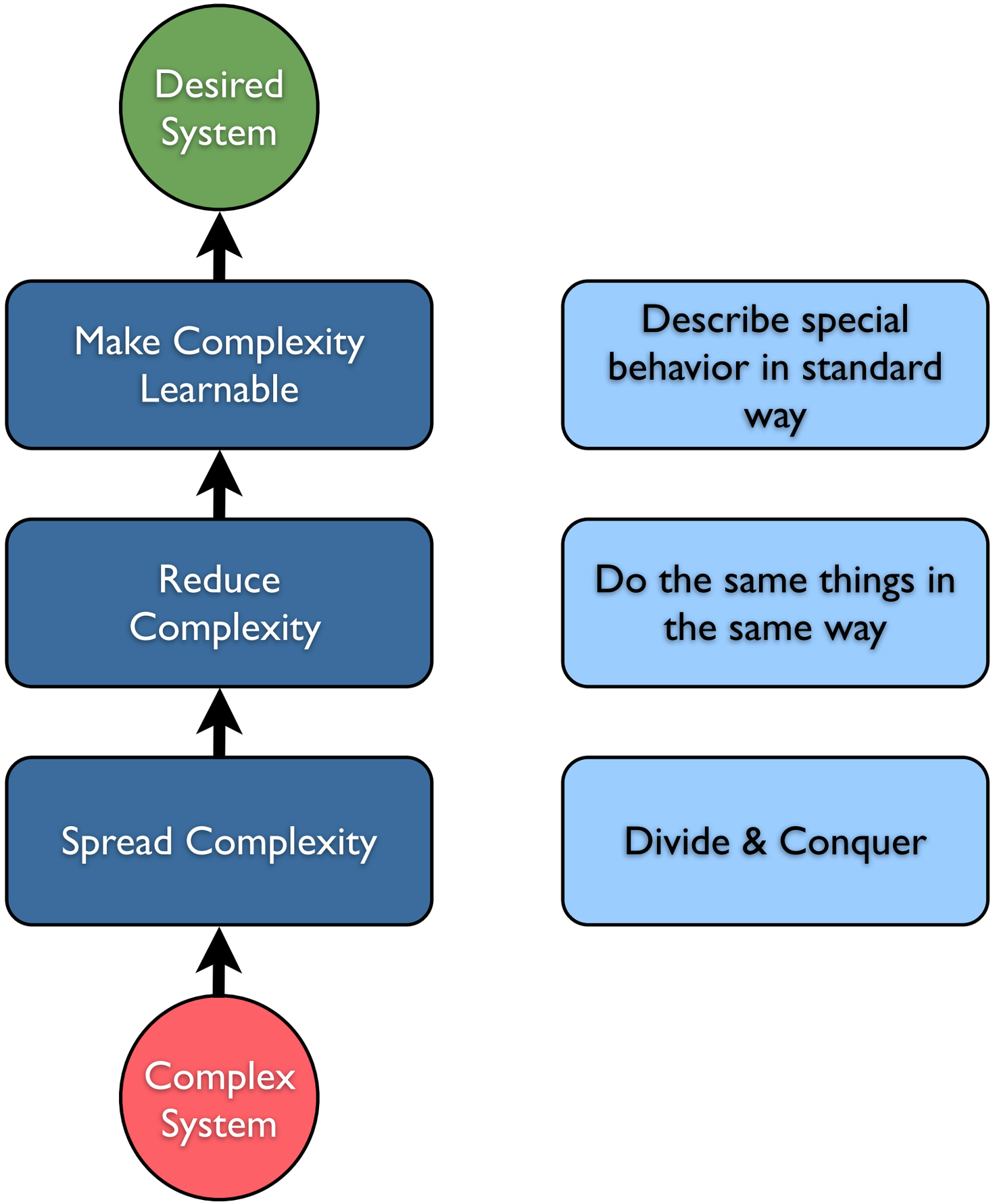
zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Divide & Conquer

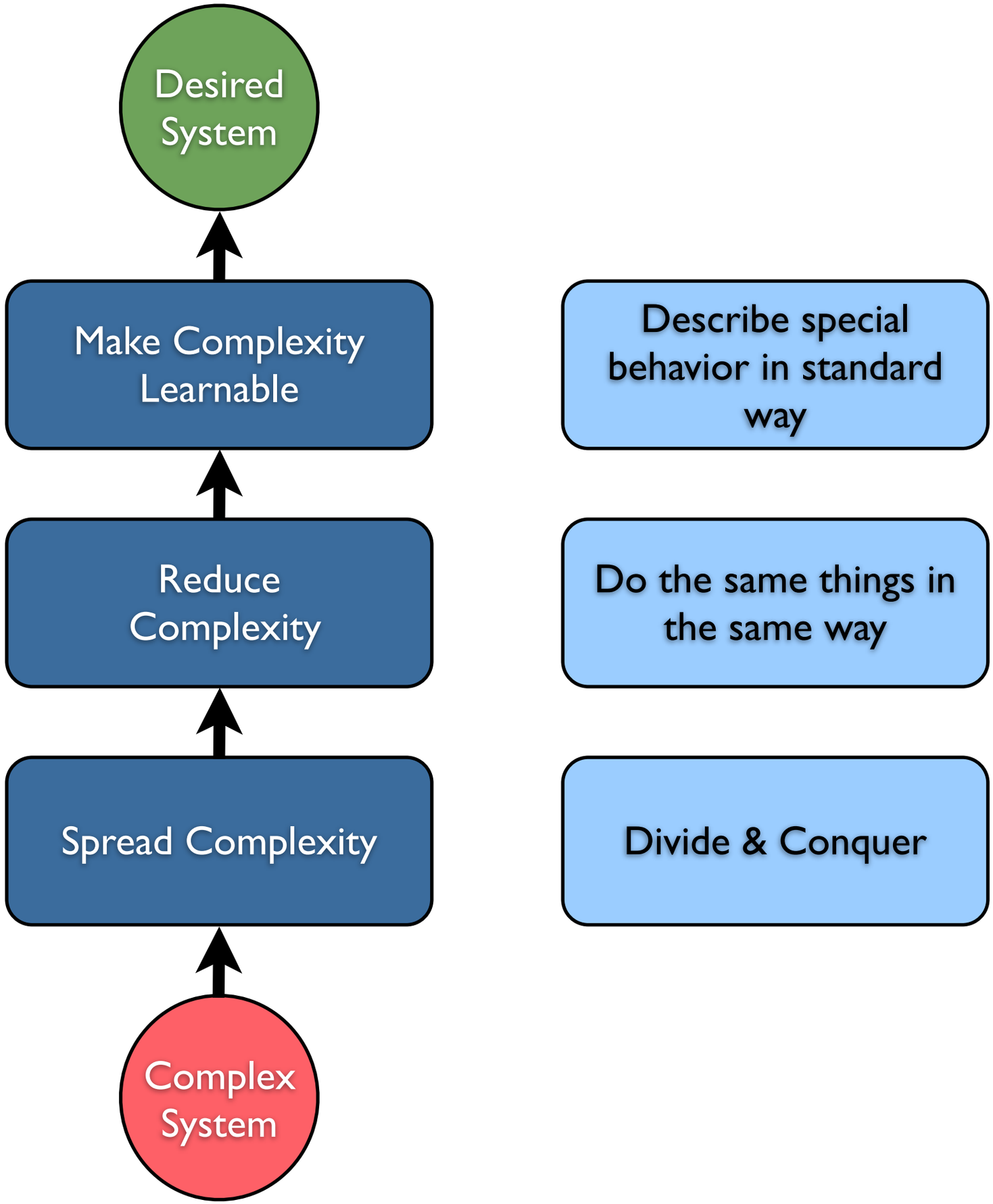
zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Do the same things in the same way
Divide & Conquer

zolodeck

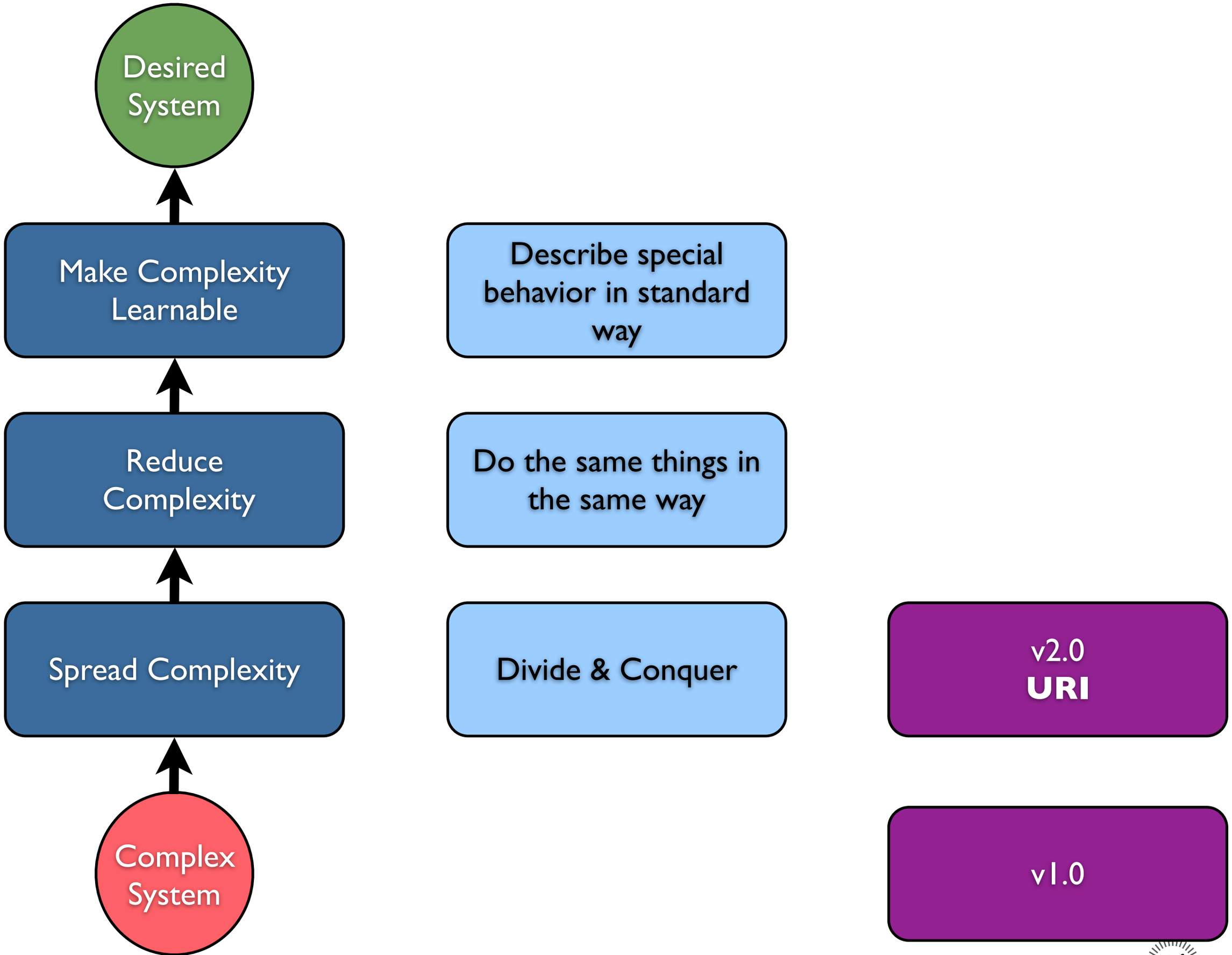

Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer

zolodeck

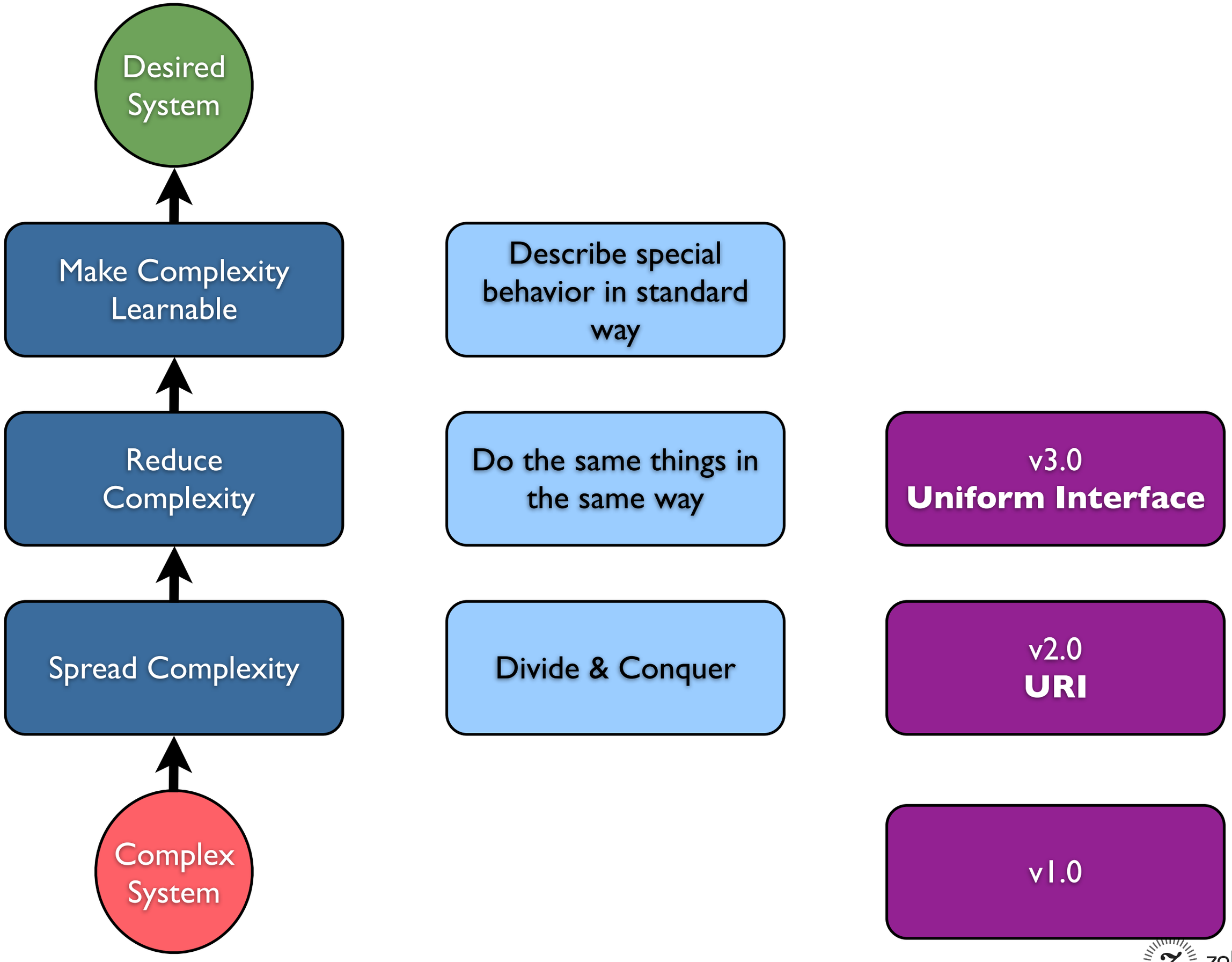


v1.0

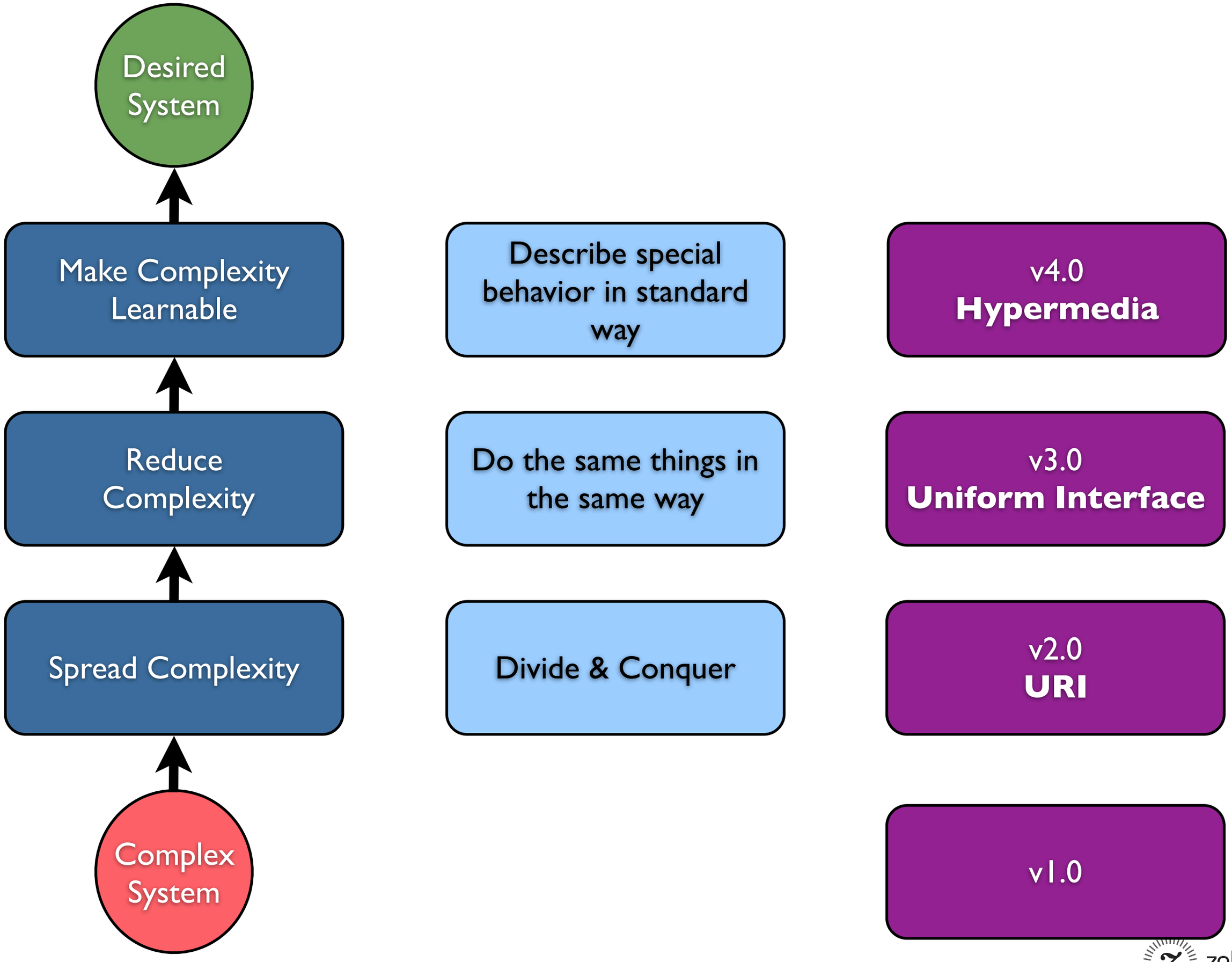
zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
v2.0
URI
v1.0

zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
v3.0
Uniform Interface
v2.0
URI
v1.0

zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
v4.0
Hypermedia
v3.0
Uniform Interface
v2.0
URI
v1.0

zolodeck

Richardson Maturity Model

zolodeck

Richardson Maturity Model

Level 0

The swamp of POX

zolodeck

Richardson Maturity Model

Level 1	URI
Level 0	The swamp of POX

zolodeck

Richardson Maturity Model

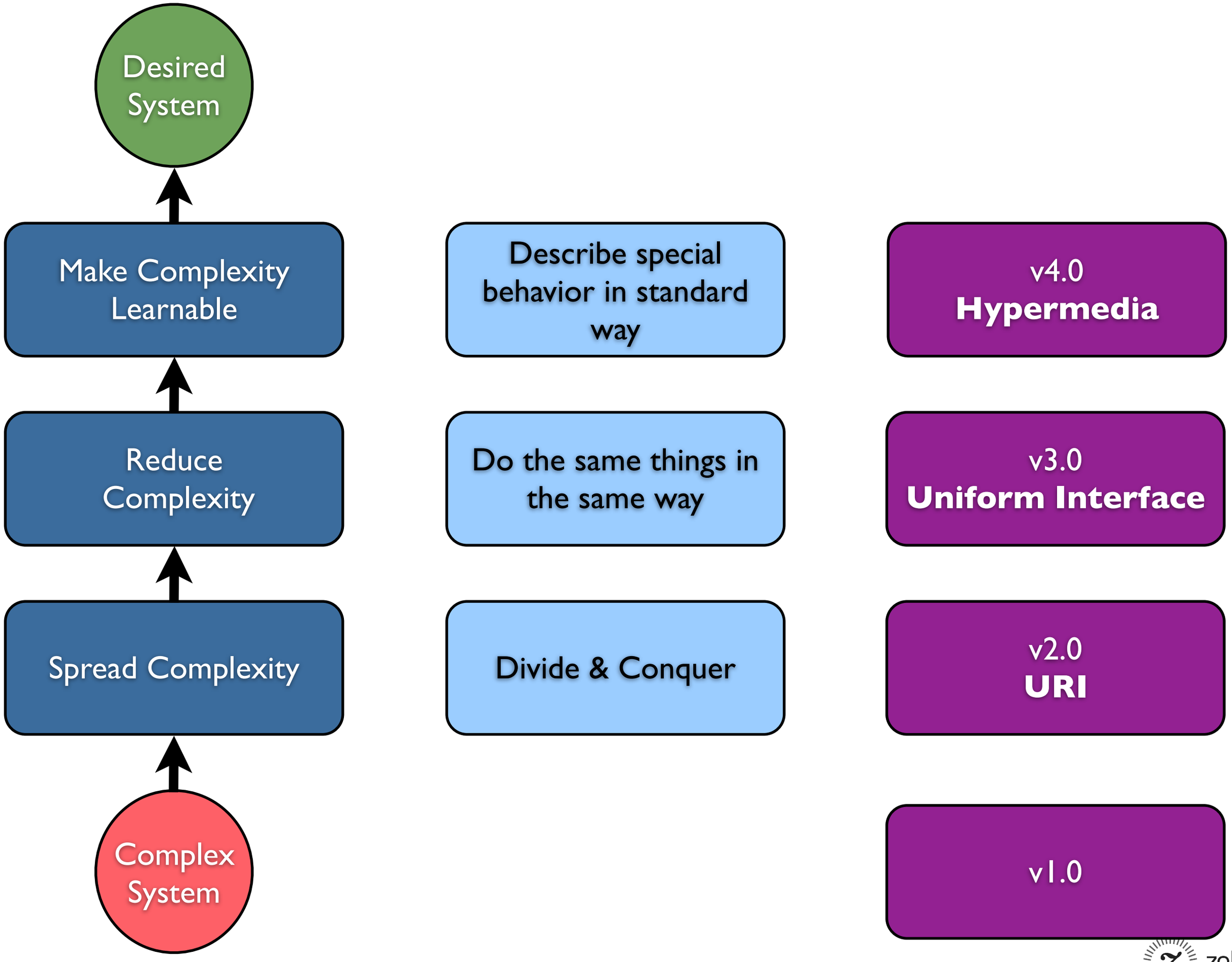
Level 2	HTTP
Level 1	URI
Level 0	The swamp of POX

zolodeck

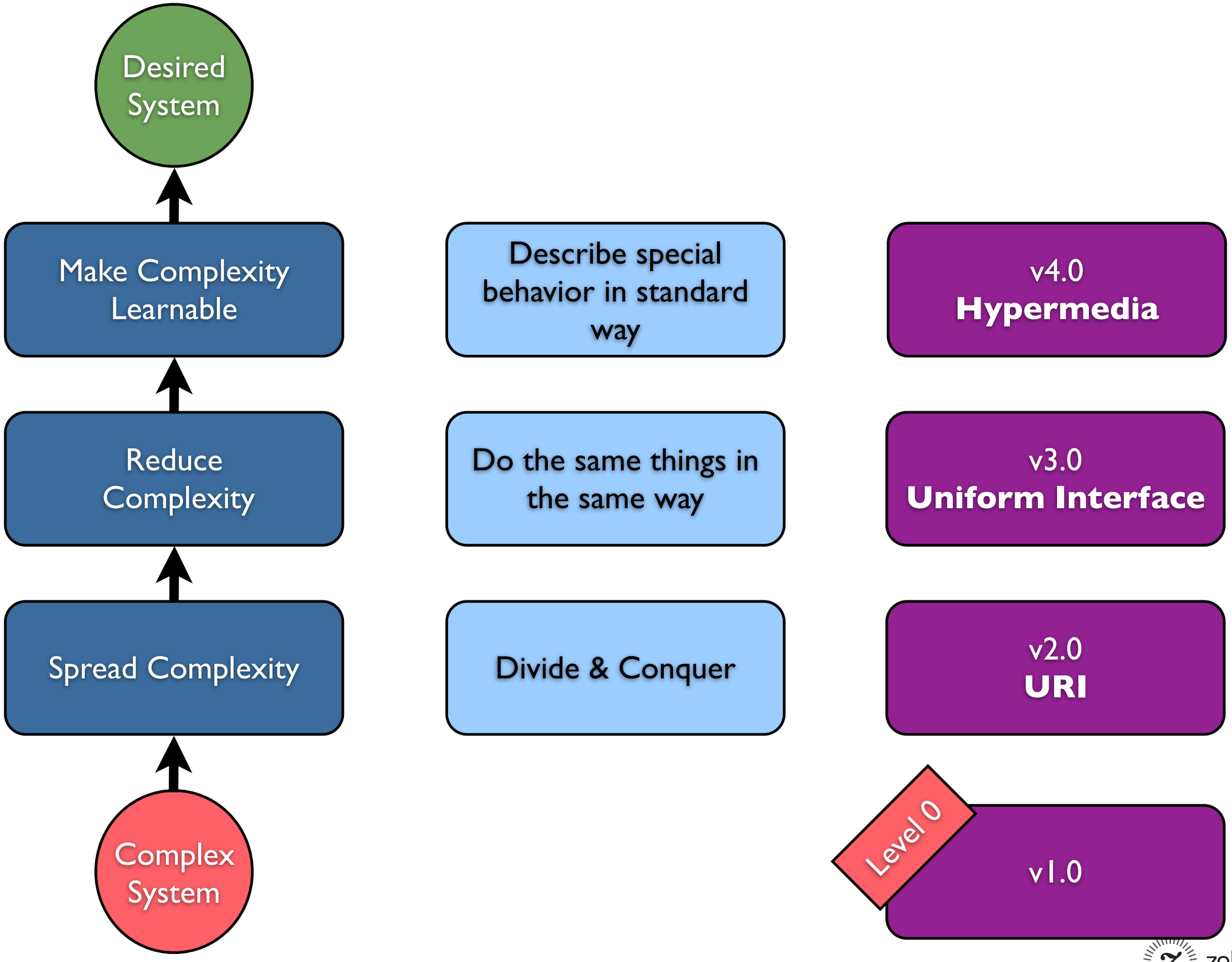
Richardson Maturity Model

Level 3	Hypermedia
Level 2	HTTP
Level 1	URI
Level 0	The swamp of POX

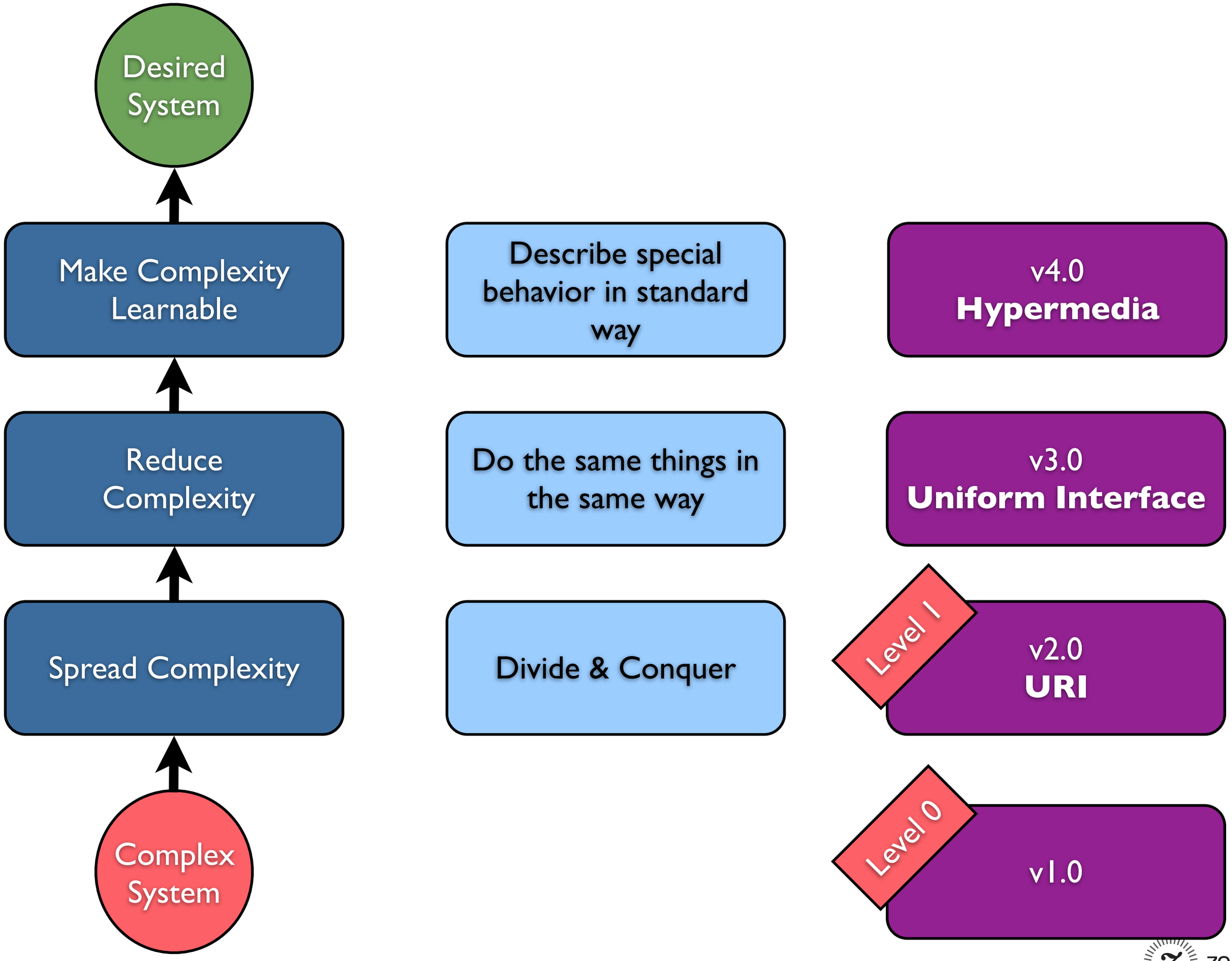
zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
v4.0
Hypermedia
v3.0
Uniform Interface
v2.0
URI
v1.0

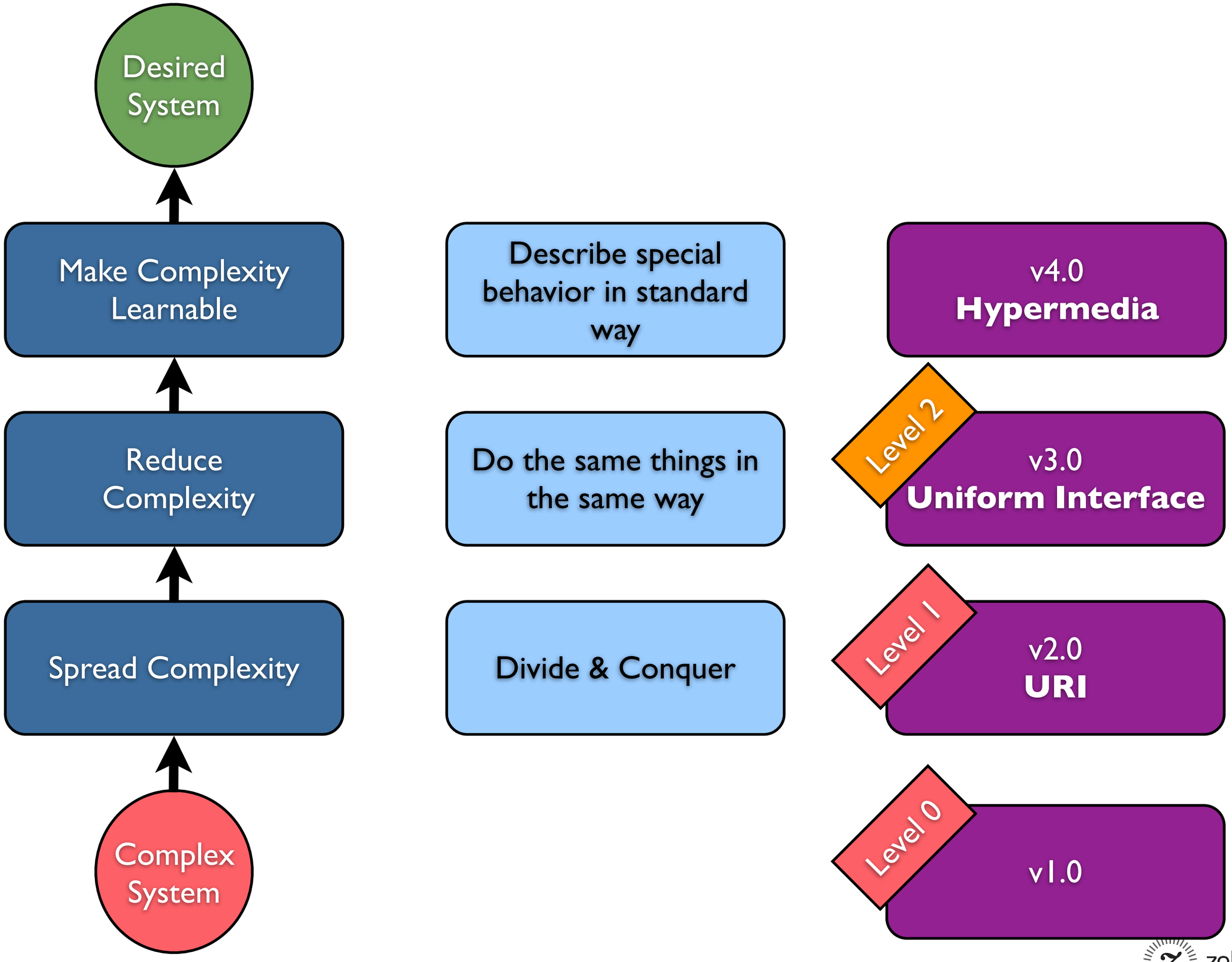
zolodeck


Desired System
Make Complexity Learnable
Describe special behavior in standard way
v4.0
Hypermedia
Reduce Complexity
Do the same things in the same way
v3.0
Uniform Interface
Spread Complexity
Divide & Conquer
v2.0
URI
Complex System
Level 0
v1.0

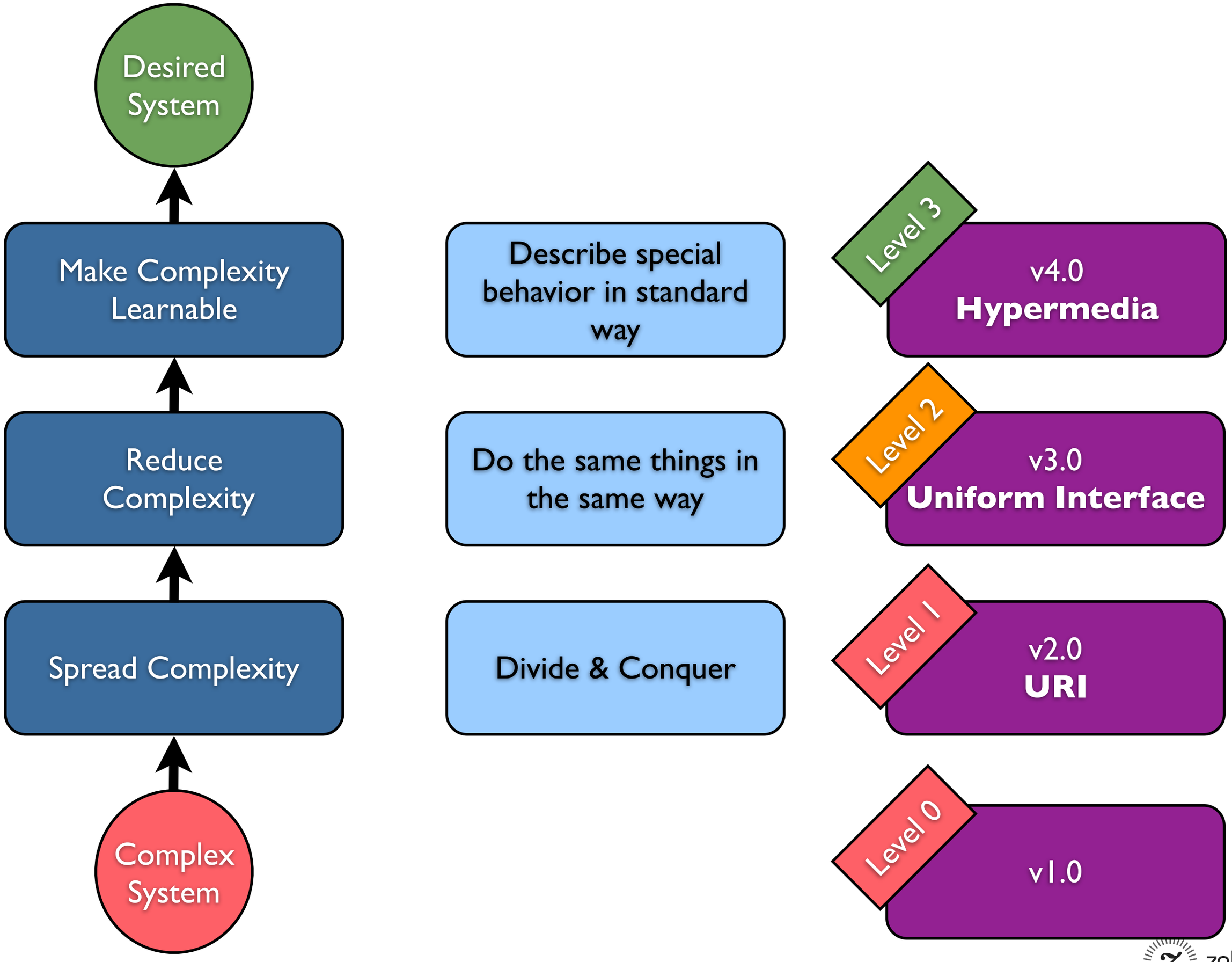
zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
v4.0
Hypermedia
v3.0
Uniform Interface
Level 1
v2.0
URI
Level 0
v1.0

zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
v4.0
Hypermedia
Level 2
v3.0
Uniform Interface
Level 1
v2.0
URI
Level 0
v1.0

zolodeck

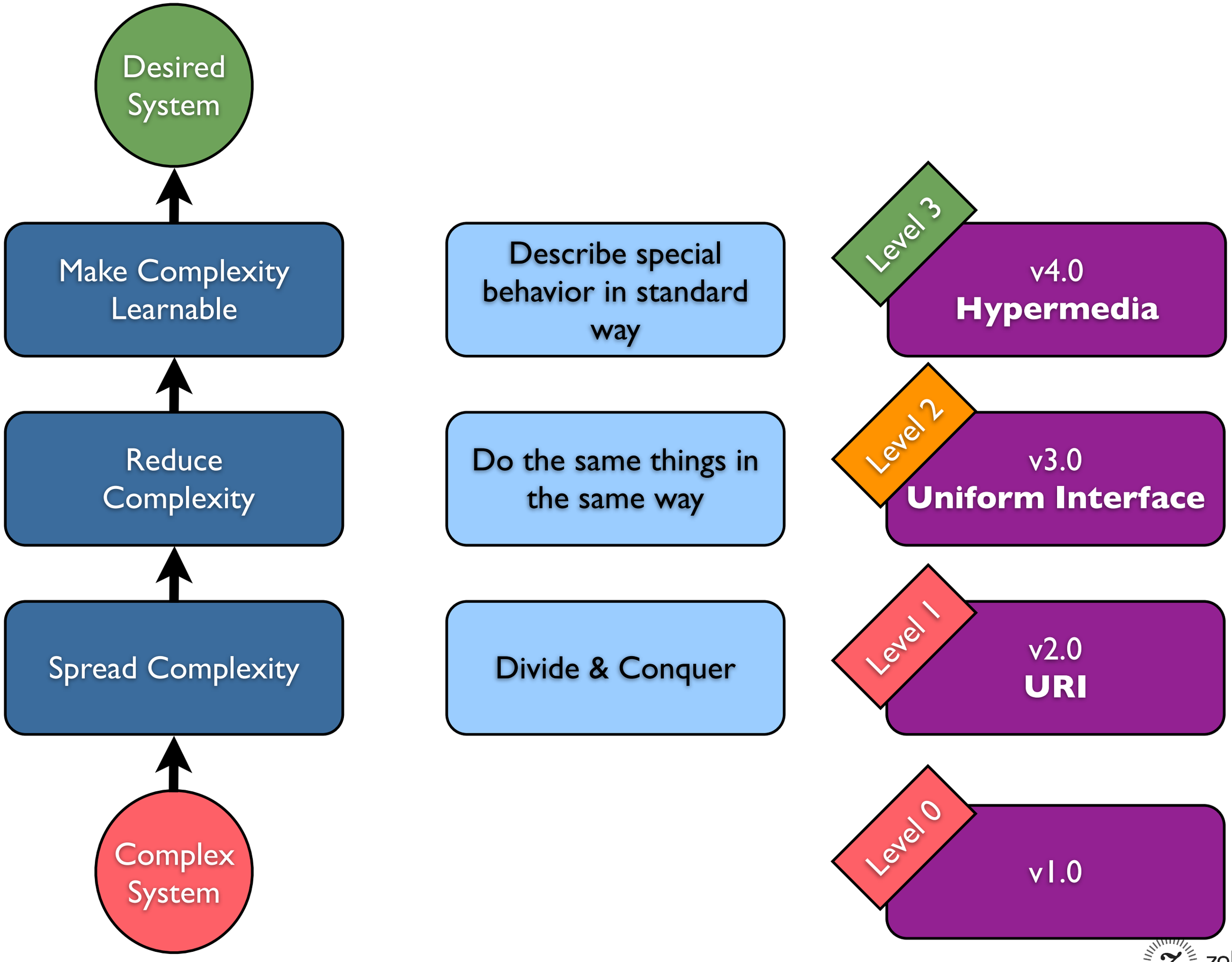

Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
Level 3
v4.0
Hypermedia
Level 2
v3.0
Uniform Interface
Level 1
v2.0
URI
Level 0
v1.0

zolodeck

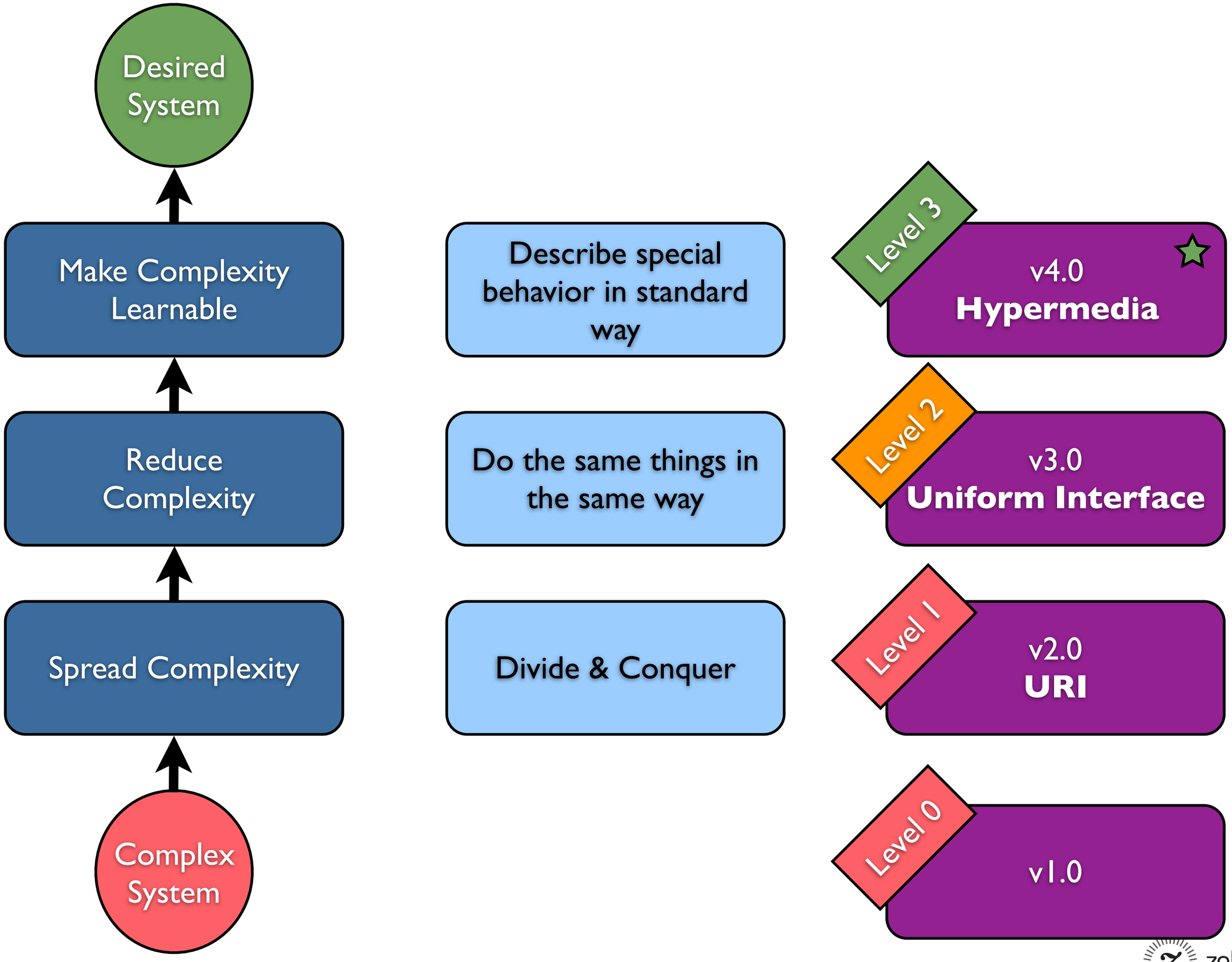
“if the engine of application state (and hence the API) is not being driven by hypertext, then it cannot be RESTful and cannot be a REST API. Period. ”

Roy T. Fielding

zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
Level 3
v4.0
Hypermedia
Level 2
v3.0
Uniform Interface
Level 1
v2.0
URI
Level 0
v1.0

zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
Level 3
v4.0
Hypermedia
Level 2
v3.0
Uniform Interface
Level 1
v2.0
URI
Level 0
v1.0

zolodeck

"Like many terms in software, REST gets lots of definitions, but since Roy Fielding coined the term, his definition should carry more weight than most."

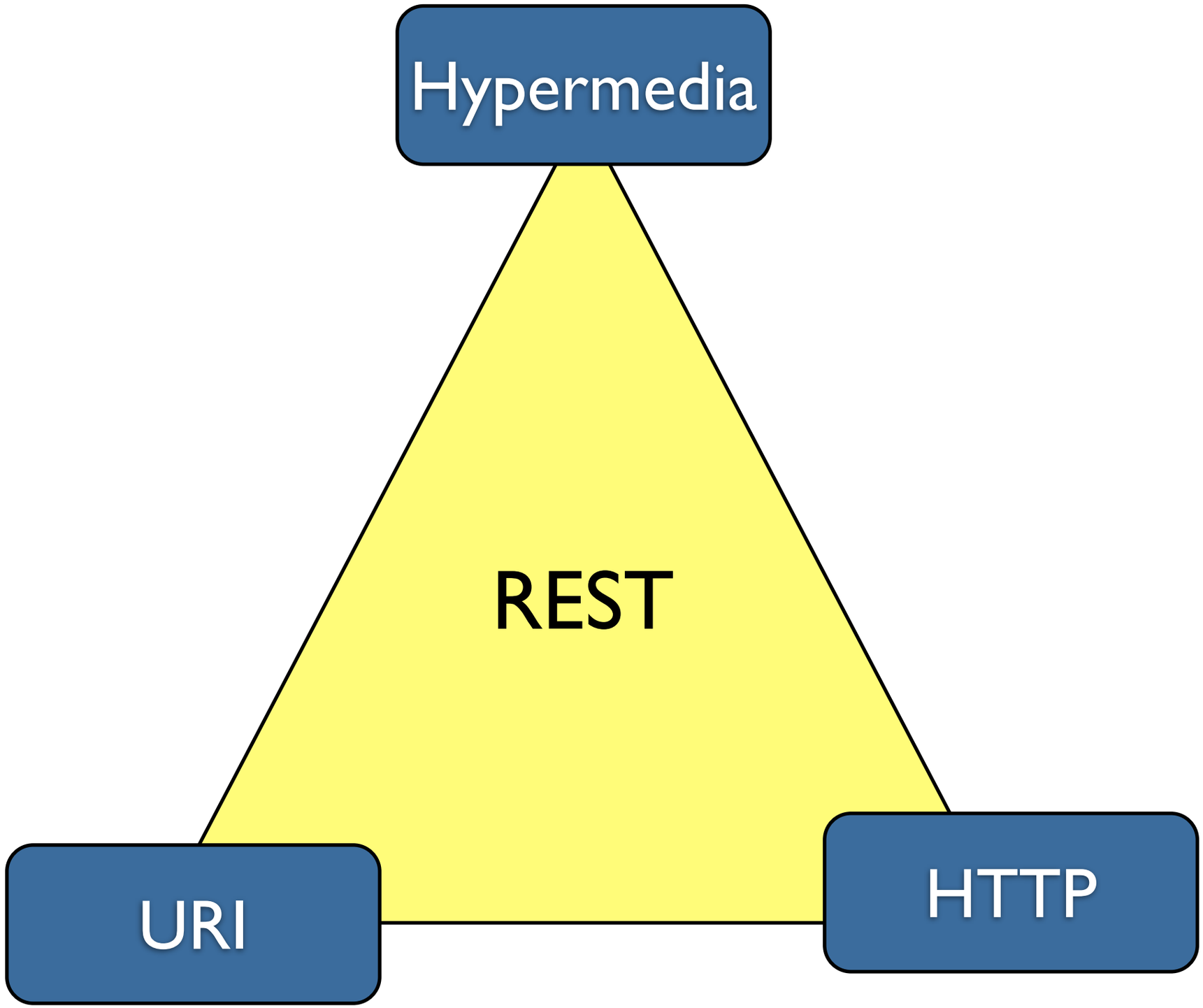
Martin Fowler

zolodeck

to sum up ...

zolodeck

What?



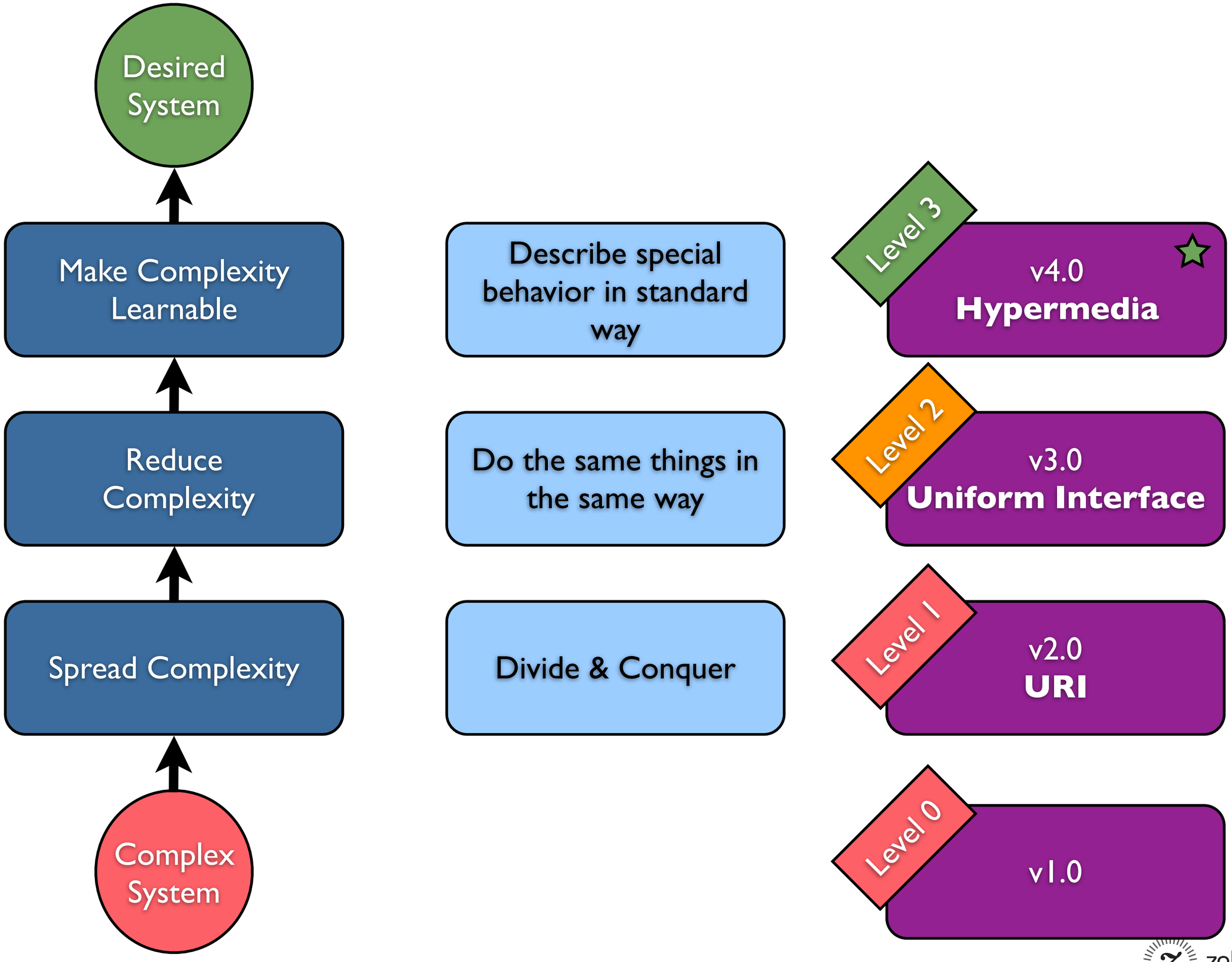
zolodeck

Why?

"REST ... every detail is intended to promote software longevity and independent evolution."

Roy T. Fielding

zolodeck


Desired System
Make Complexity Learnable
Reduce Complexity
Spread Complexity
Complex System
Describe special behavior in standard way
Do the same things in the same way
Divide & Conquer
Level 3
v4.0
Hypermedia
Level 2
v3.0
Uniform Interface
Level 1
v2.0
URI
Level 0
v1.0

zolodeck

Let us take the last step ...

zolodeck

Questions?

- @sivajag
- siva@zololabs.com
- http://blog.zololabs.com
- http://techbehindtech.com
- zolodeck.com

zolodeck