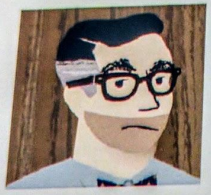

Twelve Patterns to Create Evolvable APIs
Mike Amundsen
API Academy / CA
@mamund
Drawings by Diogo Lucas
@diogoclucas


Mike Amundsen
@mamund

#mcaTravels



Mike Amundsen
@mamund

http://apiacademy.co


API


Changes Ahead


CHANGED
TRAFFIC
CONDITIONS

UNIVERSITY OF CALIFORNIA, IRVINE

Architectural Styles and the Design of Network-based Software Architectures

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Roy Thomas Fielding

2000

Dissertation Committee:
Professor Richard N. Taylor, Chair
Professor Mark S. Ackerman
Professor David S. Rosenblum

UNIVERSITY OF CALIFORNIA, IRVINE

Architectural Styles and the Design of Network-based Software Architectures

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Roy Thomas Fielding

2000

Dissertation Committee:
Professor Richard N. Taylor, Chair
Professor Mark S. Ackerman
Professor David S. Rosenblum

UNIVERSITY OF CALIFORNIA, IRVINE

Architectural Styles and the Design of Network-based Software Architectures

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

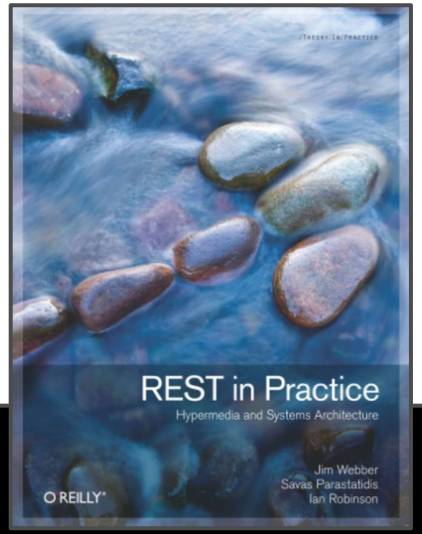
in Information and Computer Science

by

Roy Thomas Fielding

2000

Dissertation Committee:
Professor Richard N. Taylor, Chair
Professor Mark S. Ackerman
Professor David S. Rosenblum



breaking change

English [edit]

Noun [edit]

breaking change (*plural* **breaking changes**)

1. (*computing*) A change in one part of a software system that potentially causes other components to fail; occurs most often in shared libraries of code used by multiple applications

    *Not possible to fix old entries without a **breaking change**, so remap old to new in import lib.*

breaking change

English [edit]

Noun [edit]

breaking change (*plural* **breaking changes**)

1. (*computing*) A change in one part of a software system that potentially causes other components to fail; occurs most often in shared libraries of code used by multiple applications

    *Not possible to fix old entries without a **breaking change**, so remap old to new in import lib.*

See also [edit]

- backward compatibility

breaking change

Noun [edit]

backward compatibility (*usually uncountable*, *plural* **backward compatibilities**)

1. (*software*) Capability of interoperating with older systems.

Related terms [edit]

- backward compatible
- forward compatibility

es other
ultiple

*Not possible to fix old entries without a **breaking change**, so remap old to new in import lib.*

See also [edit]

- backward compatibility

breaking change

Noun [edit]

backward compatibility (*usually uncountable*, *plural* **backward compatibilities**)

1. (*software*) Capability of interoperating with older systems.

Rel

-
-

forward compatibility (*uncountable*)

1. (*software*) Capability of interoperating with anticipated future systems.

Related terms [edit]

- backward compatibility
- forward compatible

See

- backward compatibility

I want to be able to...

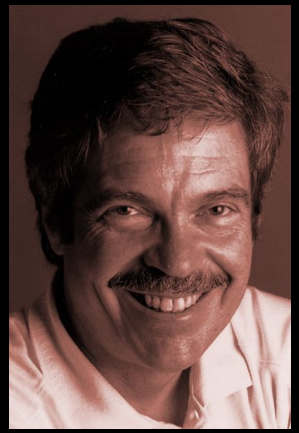
1. Change/Add Objects (payloads)
2. Change/Add Addresses (URLs)
3. Change/Add Actions (links and forms)

The OAA Challenge

1. Change/Add **O**bjects (payloads)
2. Change/Add **A**ddresses (URLs)
3. Change/Add **A**ctions (links and forms)

The OAA Challenge

1. Change/Add **O**bjects (payloads)
2. Change/Add **A**ddresses (URLs)
3. Change/Add **A**ctions (links and forms)

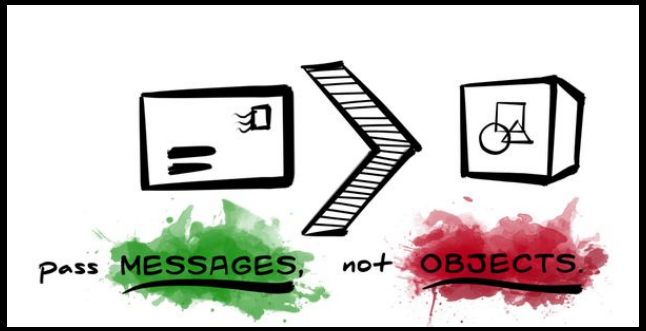


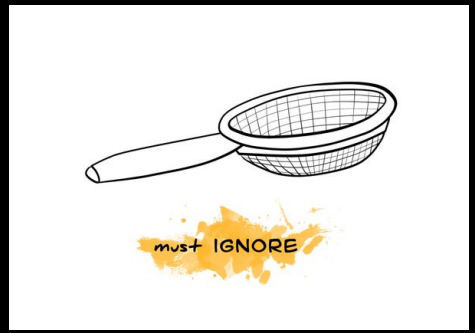
"[A] dynamic system that has extreme late binding in all aspects." -- Alan Kay, 2003

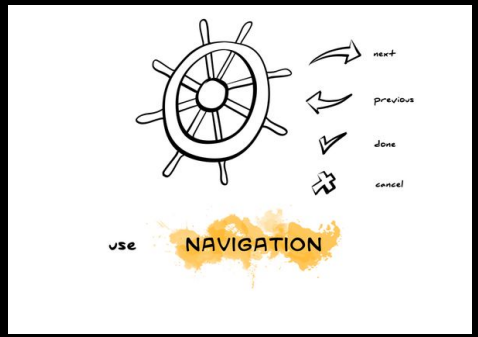
Evolvable API Patterns

Twelve Patterns for Evolvable APIs

Four Design Patterns
Four Basic Principles
Four Shared Agreements



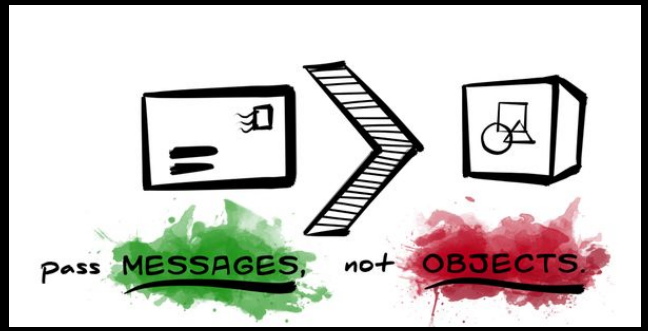


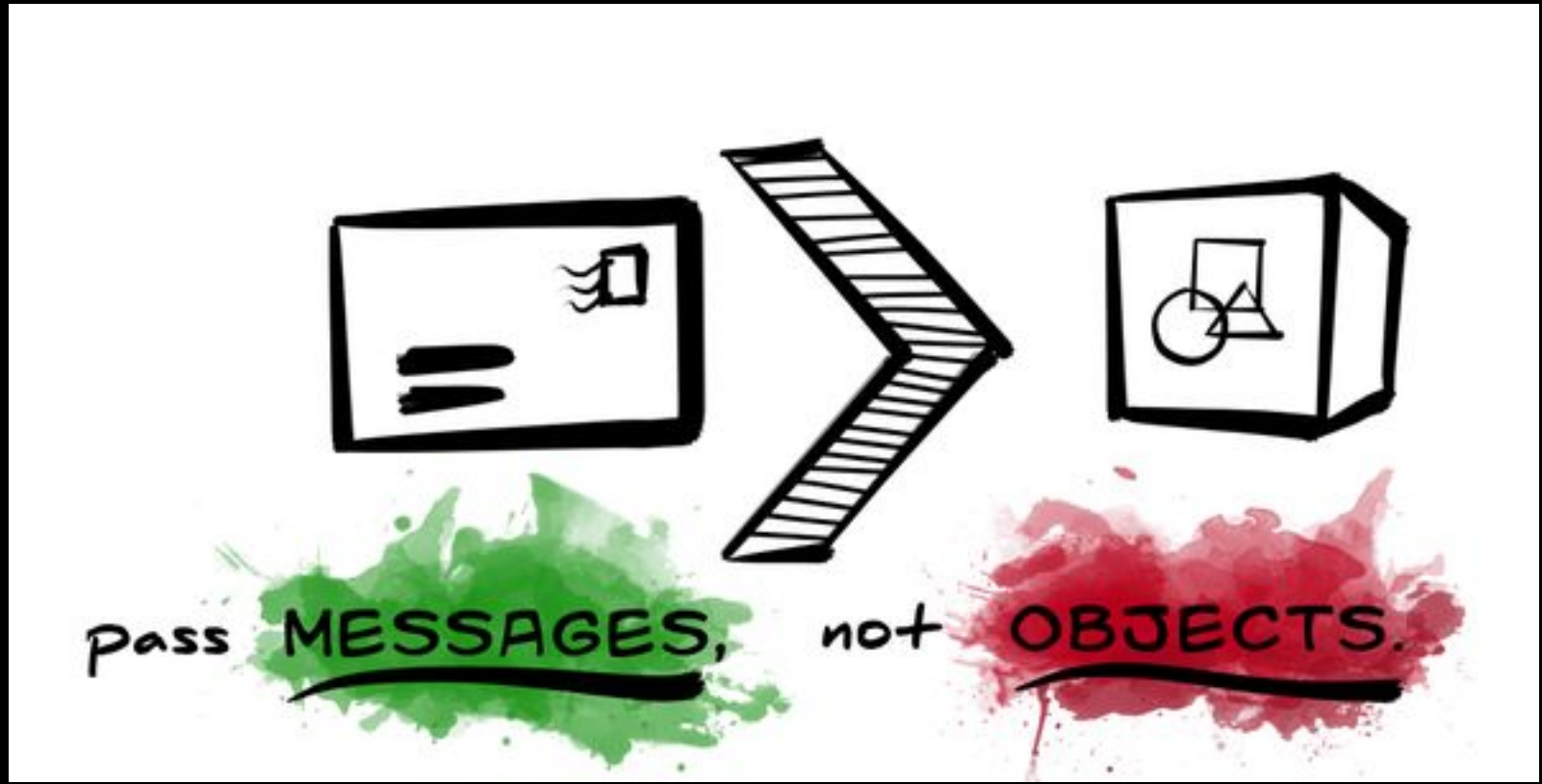


Design Patterns

Design Patterns

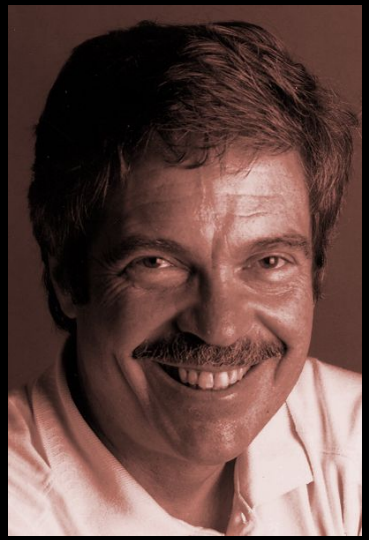
1. PASS MESSAGES, NOT OBJECTS
2. SHARE VOCABULARIES, NOT MODELS
3. USE THE REPRESENTOR PATTERN
4. PUBLISH PROFILES



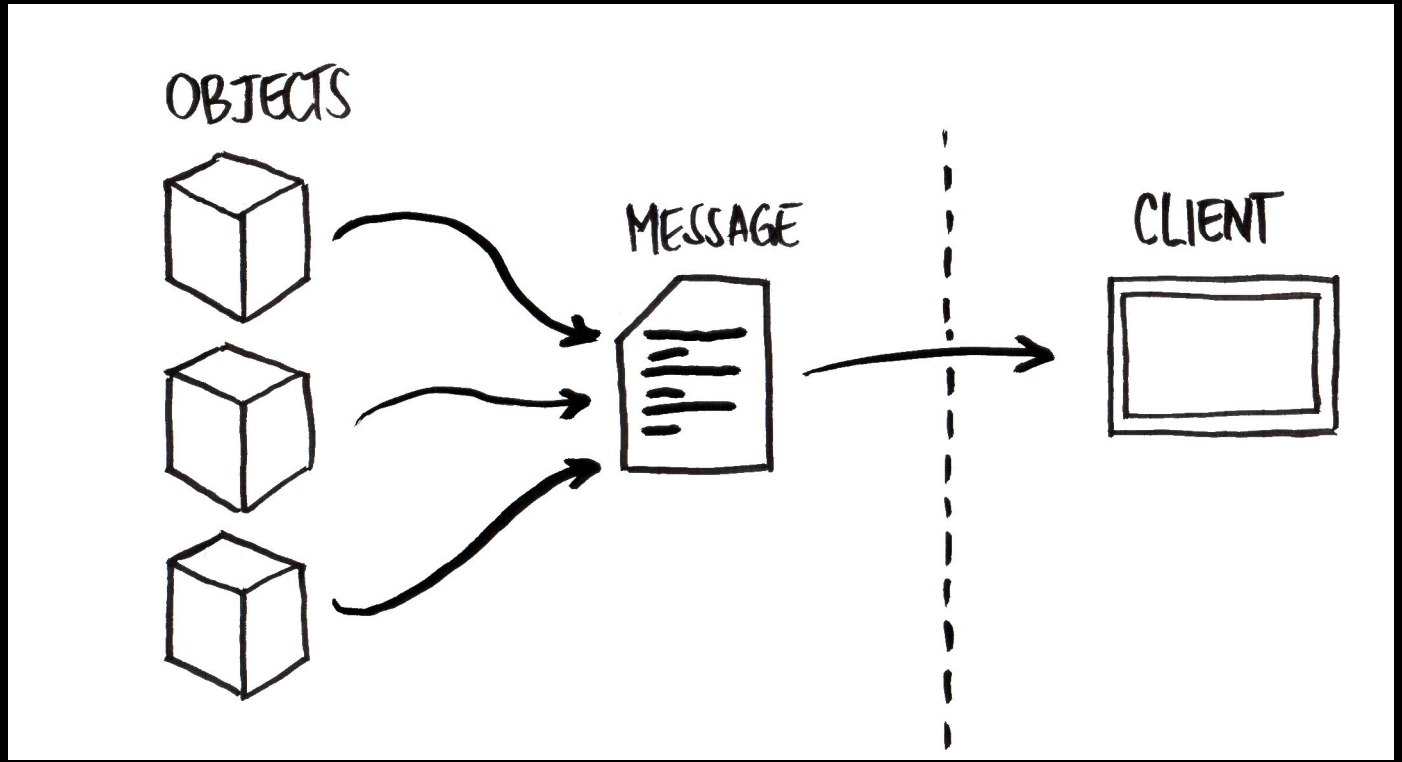

pass MESSAGES, not OBJECTS.

Pass Messages, Not Objects

"I'm sorry that coined the term 'objects' for this topic. The big idea is 'messaging'."

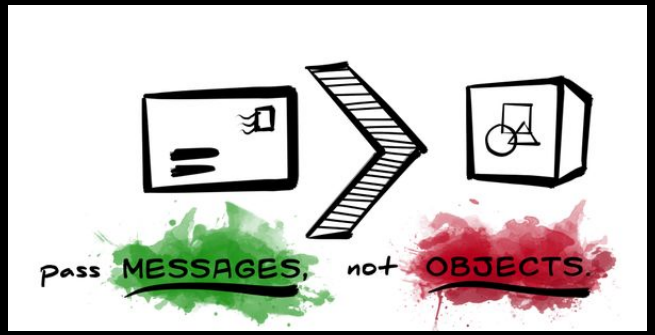


Alan Kay, 1998

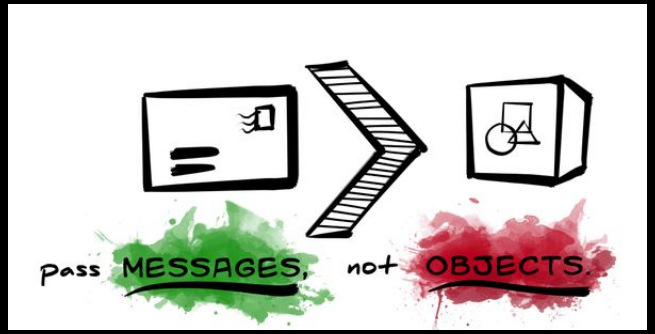

OBJECTS
MESSAGE
CLIENT

Pass Messages, Not Objects

Bodies SHOULD be sent using a highly-structured metadata-rich format such as:
HAL
Collection+JSON
Siren
UBER
Atom,
etc.



```
var responseObject = {};
var messageBody = "";
responseObject.customerSummary =
 dataStore.getCustomerSummary(custId);
responseObject.outstandingInvoices =
 dataStore.getInvoices(custId,status="outstanding");
responseObject.shippingStatus =
 dataStore.getOrdersInTransit(custId);

messageBody = messageTranslator(
 responseObject,
 "application/HTML"
);

HTTP.Send(messageBody);
```

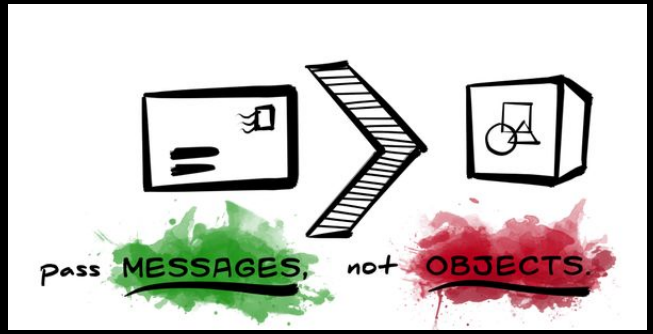


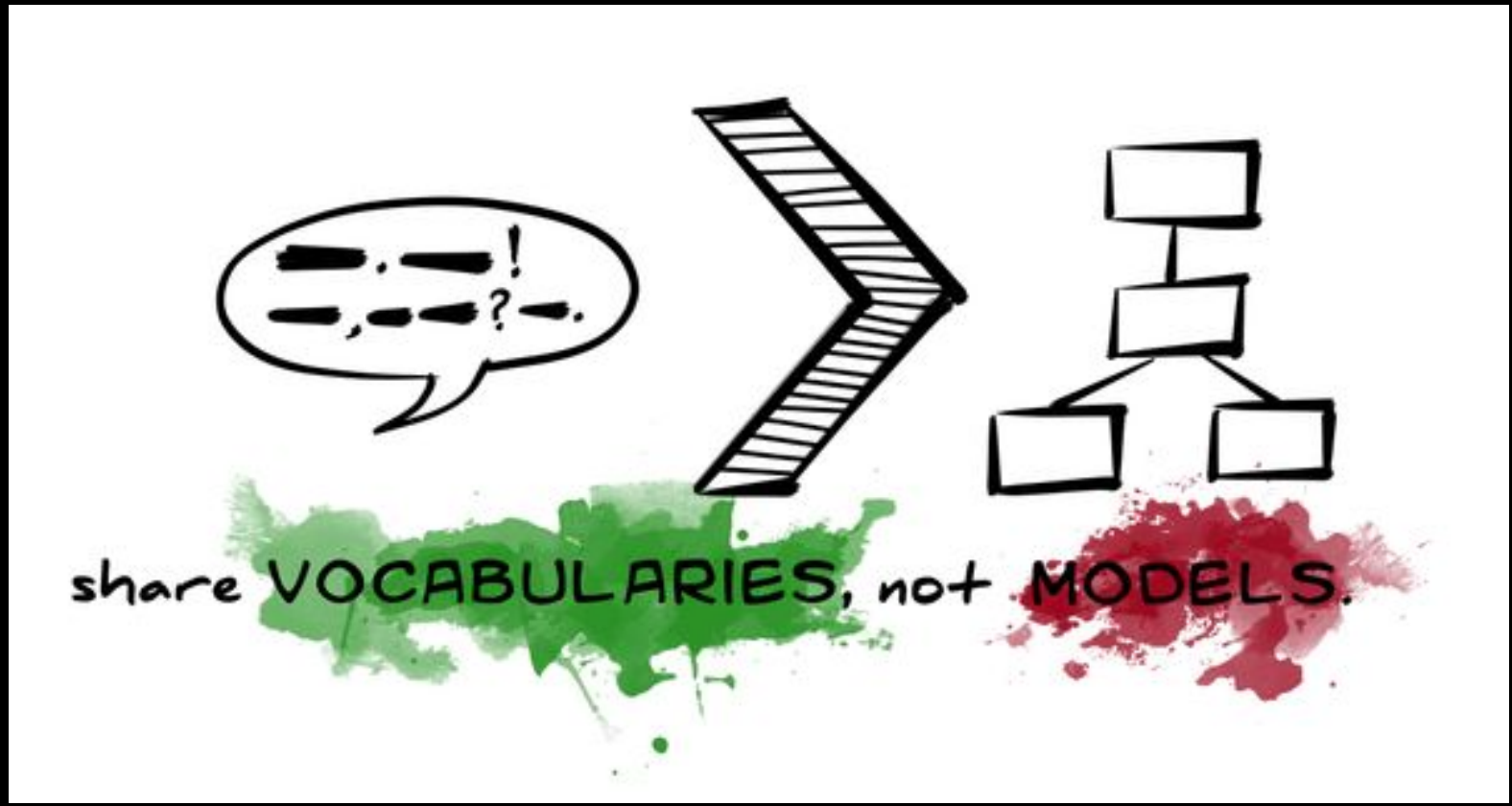
Pass Messages, not Objects

What problem does this solve?

I don’t need to share your object model to interact with you.

Machines can now manage their own internal models independently.

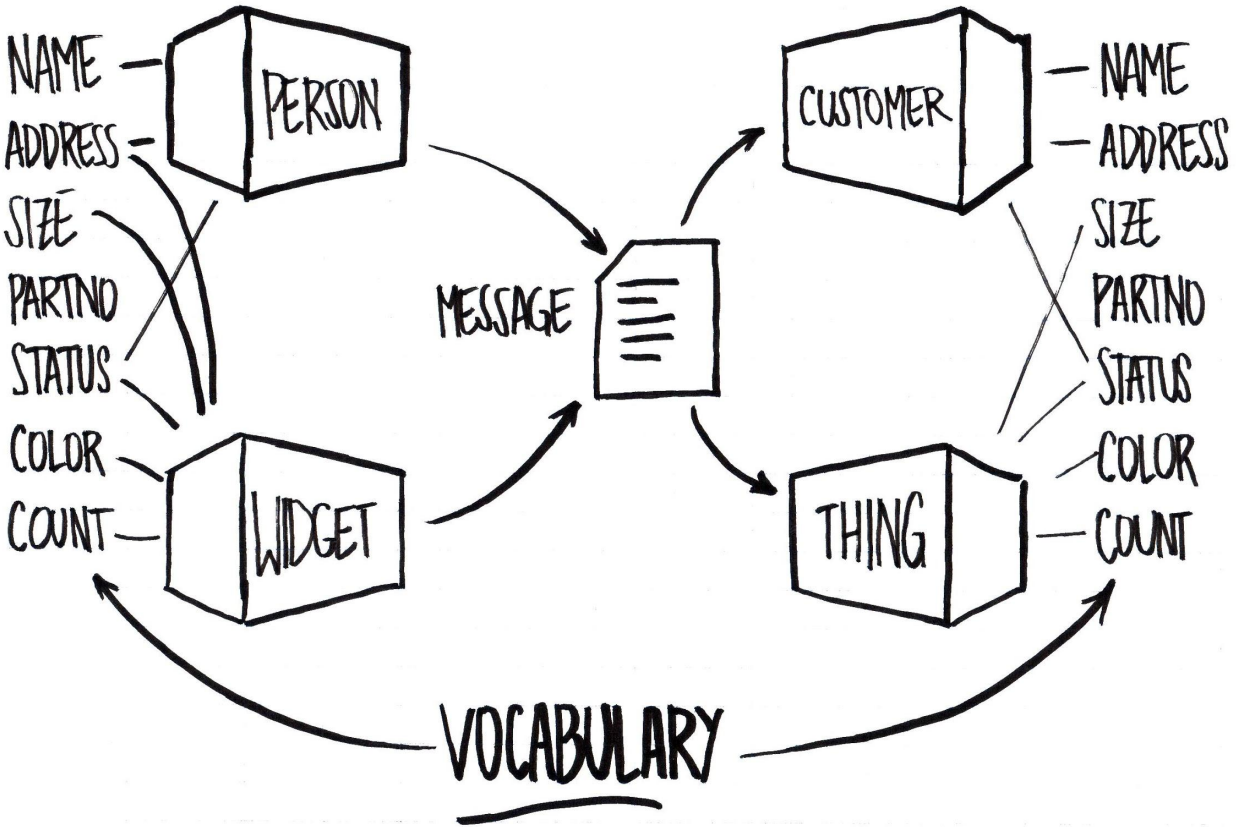



share VOCABULARIES, not MODELS.

Share Vocabularies, Not Models

"It is easier to standardize representation and relation types than objects and object-specific interfaces."

-- Roy Fielding

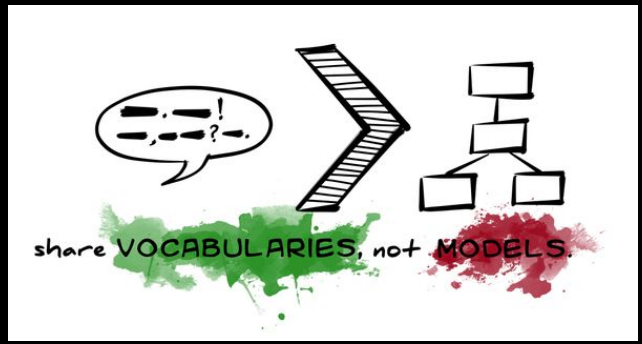


SHARE VOCABULARIES

Share Vocabularies, Not Models

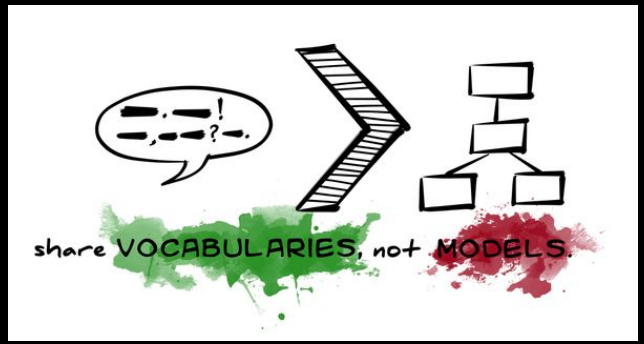
All messages SHOULD rely only on standardized identifiers (for data/action) based on shared vocabularies.

IANA Link Relation Values
Schema.org
Microformats
Dublin Core
Activity Streams



Shared Public Vocabulary

- Name
- Address
- Size
- Status
- Color
- Partno
- Count

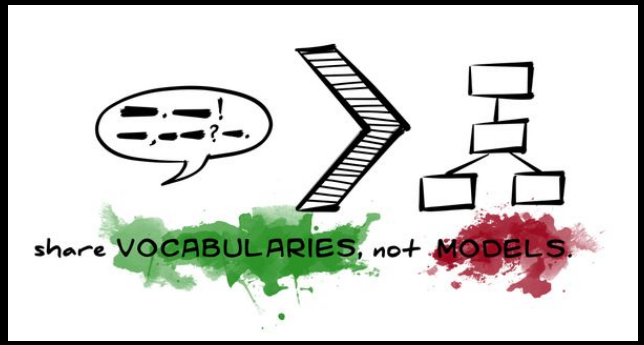


Shared Public Vocabulary

- Name
- Address
- Size
- Status
- Color
- Partno
- Count

Server's Internal Model

- Person
  - Name
  - Address
  - Status
- Widget
  - Address
  - Size
  - Status
  - Color
  - Count



Shared Public Vocabulary

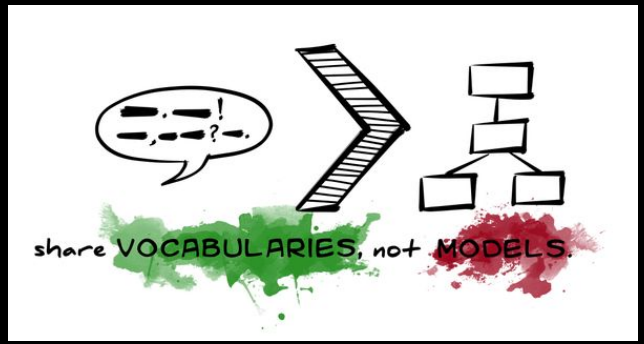
- Name
- Address
- Size
- Status
- Color
- Partno
- Count

Server's Internal Mode

- Person
  - Name
  - Address
  - Status
- Widget
  - Address
  - Size
  - Status
  - Color
  - Count

Client's Internal Model

- Customer
  - Name
  - Address
  - Status
- Thing
  - Size
  - Status
  - Color
  - Count

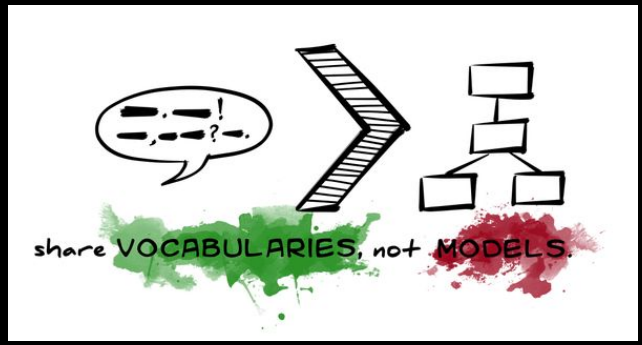


Share Vocabularies, Not Models

What problem does this solve?

Vocabulary is how we “evaluate and select”

Machines can now
evaluate and select without
direct human interaction.



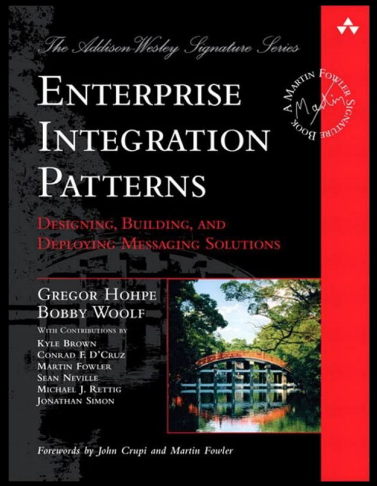
on behalf of...

use REPRESENTER

Use the Representor Pattern

*"Use a special filter, a **Message Translator**, between other filters or applications to translate one data format into another."*

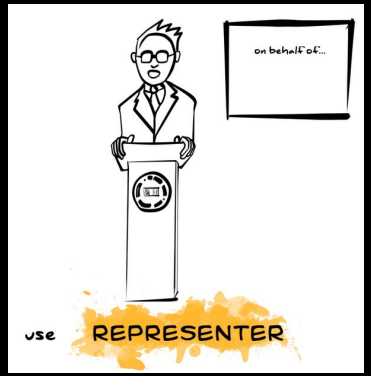
- Gregor Hohpe

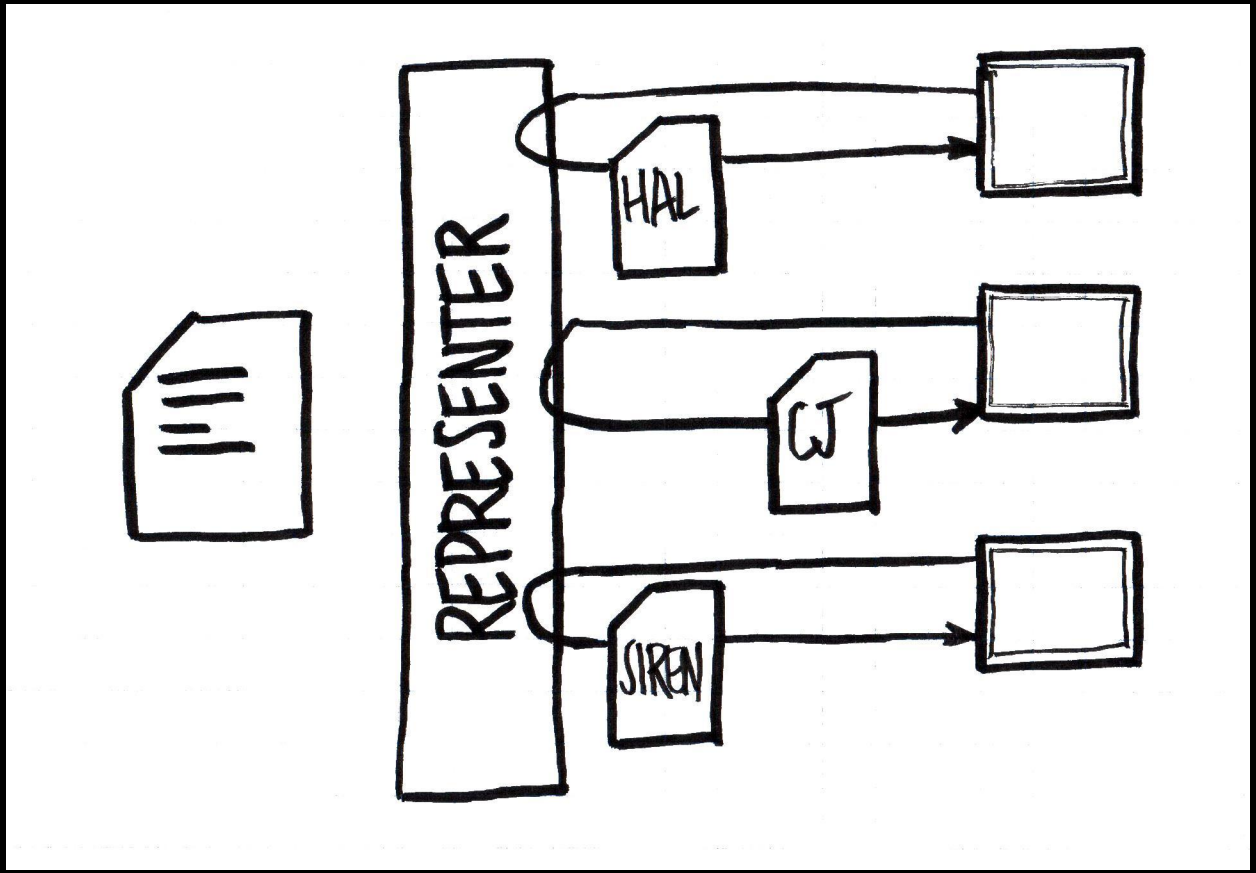


Use the Representor Pattern

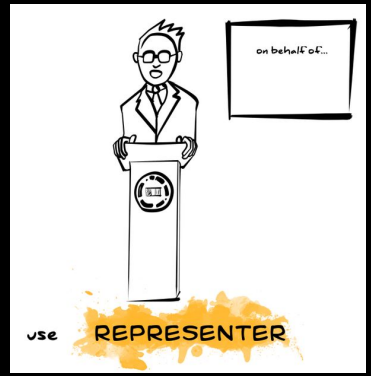
You SHOULD implement a message translator to convert internal models into public messages.

Standard Resource Model (WeSTL)
Strategy Messages Format Dispatch




REPRESENTER
HAL
CJ
SIREN

```
function representor(object, mimeType, root) {
  var doc;

  // clueless? assume JSON
  if (!mimeType) {
    mimeType = defaultFormat;
  }

  // dispatch to requested representor
  switch (mimeType.toLowerCase()) {
    case "application/vnd.wstl+json":
      doc = wstljson(object, root);
      break;
    case "application/json":
      doc = json(object, root);
      break;
    case "application/vnd.hal+json":
      doc = haljson(object, root);
      break;
    case "application/vnd.siren+json":
      doc = siren(object, root);
      break;
    case "application/vnd.collection+json":
      doc = cj(object, root);
      break;
    default:
      doc = cj(object, root);
      break;
  }

  return doc;
}
```

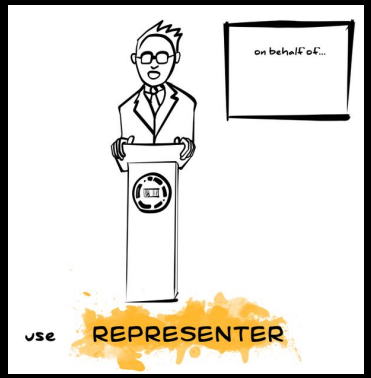


Use the Representor Pattern

What problem does this solve?

Sometimes we need to translate our conversations in order to communicate.

Machines can now “negotiate” the language of a conversation.



publish PROFILES

Publish Profiles

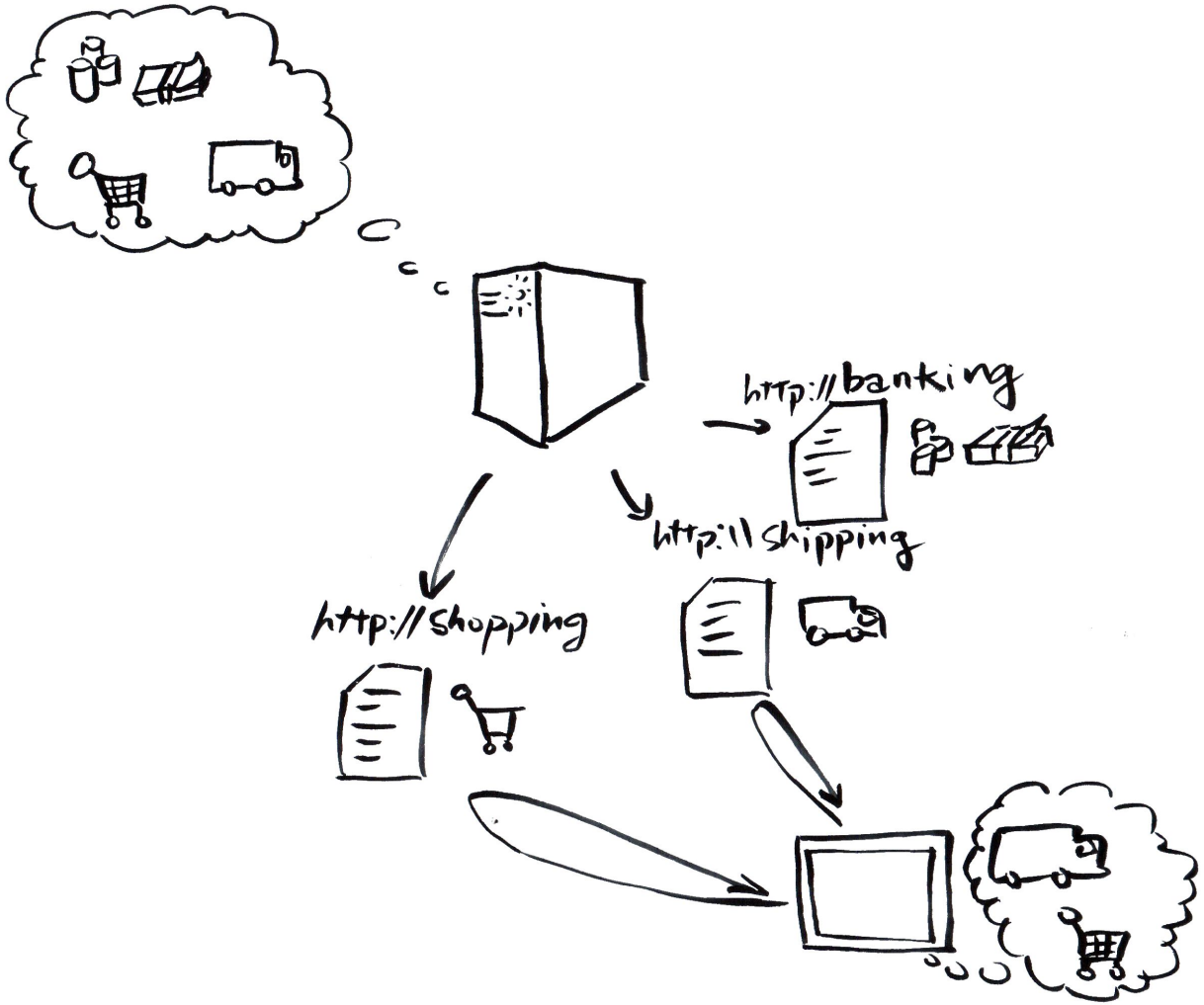
"Profiles provide a way to create a ubiquitous language for talking about APIs (resources) for both humans and machines."

-- Mark Foster

Publish Profiles

All messages SHOULD be accompanied by one or more PROFILE identifiers.

Define all possible data and actions
Use Profile Standard (RFC6906)
Servers emit profile URI
Clients validate profile URI


http://banking
http://shipping
http://shopping

```
#### Sucessful Profile Negotiation
*** REQUEST
GET /accounts HTTP/1.1
  Host: example.org
  Accept: application/vnd.uber+xml
  Link: <http://alps.io/banking/v3>;rel="profile">

***RESPONSE
HTTP/1.1 200 OK
  Content-Type: application/vnd.uber+xml
  Link: <http://alps.io/banking/v3>;rel="profile">

  <uber version="1.0">
    ...
  </uber>
```

```
#### Sucessful Profile Negotiation
*** REQUEST
GET /accounts HTTP/1.1
  Host: example.org
  Accept: application/vnd.uber+xml
  Link: <http://alps.io/banking/v3>;rel="profile">

***RESPONSE
HTTP/1.1 20
  Content-T
  Link: <ht

  <uber ver
    ...
  </uber>
```

```
#### Failed Profile Negotitation
*** REQUEST
GET /accounts HTTP/1.1
  Host: example.org
  Accept: application/vnd.uber+xml
  Link: <http://alps.io/banking/v4>;rel="profile">

***RESPONSE
HTTP/1.1 400 Bad Request
  Content-Type: application/vnd.uber+xml
  Link: <http://alps.io/banking/v2>;rel="profile">

  <uber version="1.0">
   <data id="error" text="Requested Profile Unsupported" />
  </uber>
```

products-alps.xml

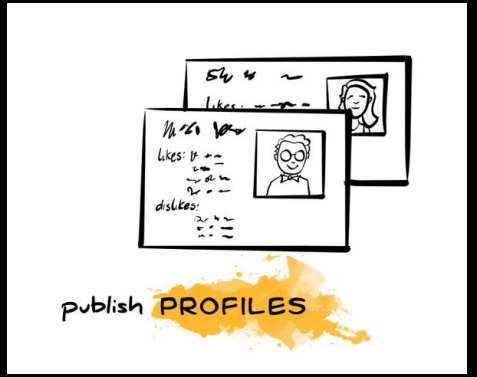
```xml
<alps version="1.0">
  <link rel="help" href="http://example.org/documentation/products.html" />
  <doc>
    This is a prototype product API.
  </doc>

  <!-- transitions -->
  <descriptor id="item" type="safe" rt="#product">
    <doc>Retrieve A Single Product</doc>
  </descriptor>

  <descriptor id="collection" type="safe" rt="#product">
    <doc>Provides access to all products</doc>
  </descriptor>

  <descriptor id="search" type="safe" rt="#product">
    <doc>Provides access to all products</doc>
    <descriptor href="#id" />
  </descriptor>

  <descriptor id="edit" type="idempotent" rt="#product">
    <doc>Updates A Product</doc>
    <descriptor href="#product" />
  </descriptor>

  <descriptor id="create" type="unsafe" rt="#product">
    <doc>Allows the creation of a new product</doc>
    <descriptor href="#product" />
  </descriptor>
```

Publish Profiles

What problem does this solve?

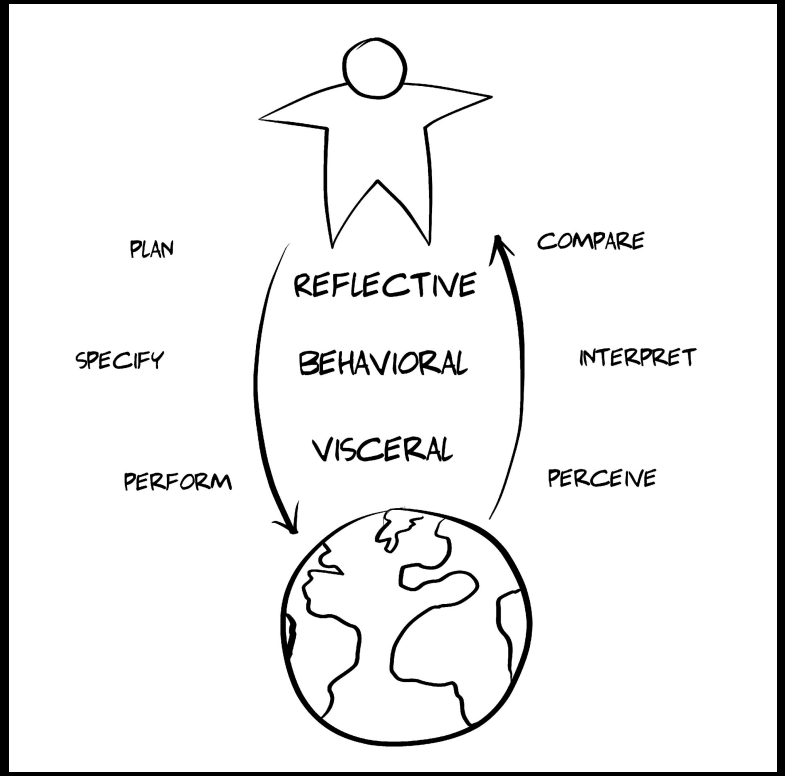
I need to know what we're talking about.

Machines can now
validate domain topics easily



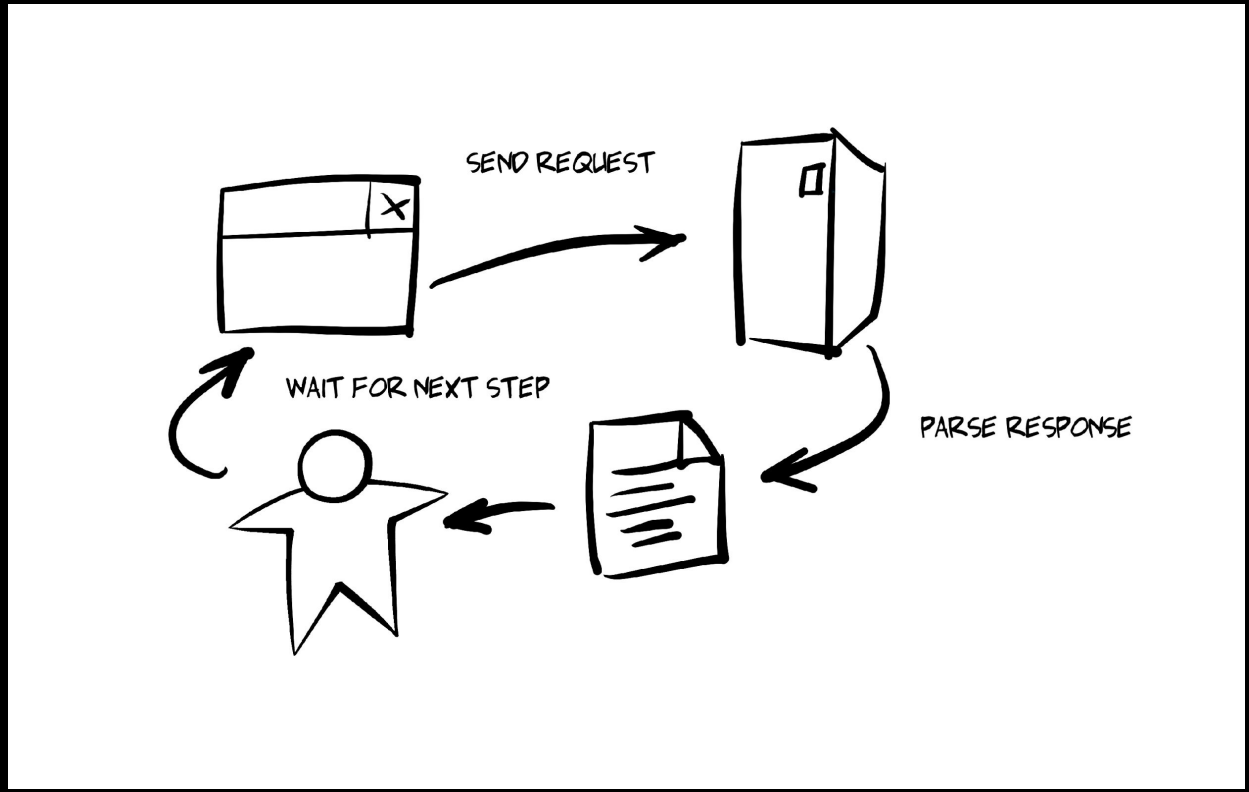
Messages

Norman's Action Lifecycle

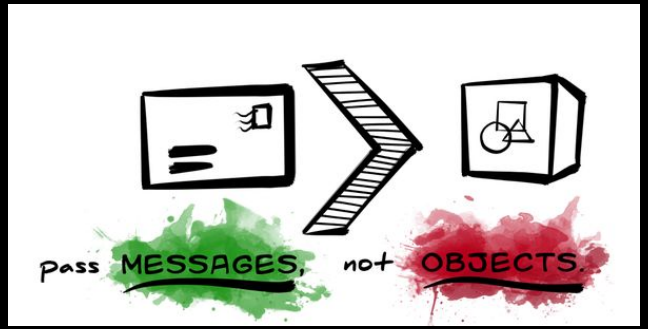


Donald Norman

Employing the RPW Loop



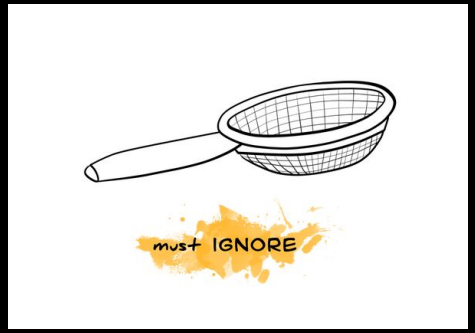
Design loosely-coupled interoperable services

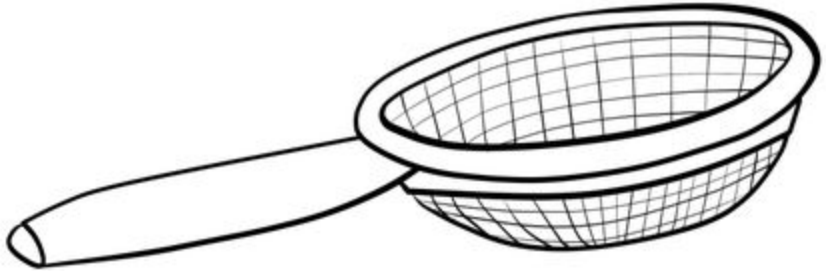


Basic Principles

Basic Principles

5. MUST IGNORE
6. MUST FORWARD
7. PROVIDE MRU
8. USE IDEMPOTENCE



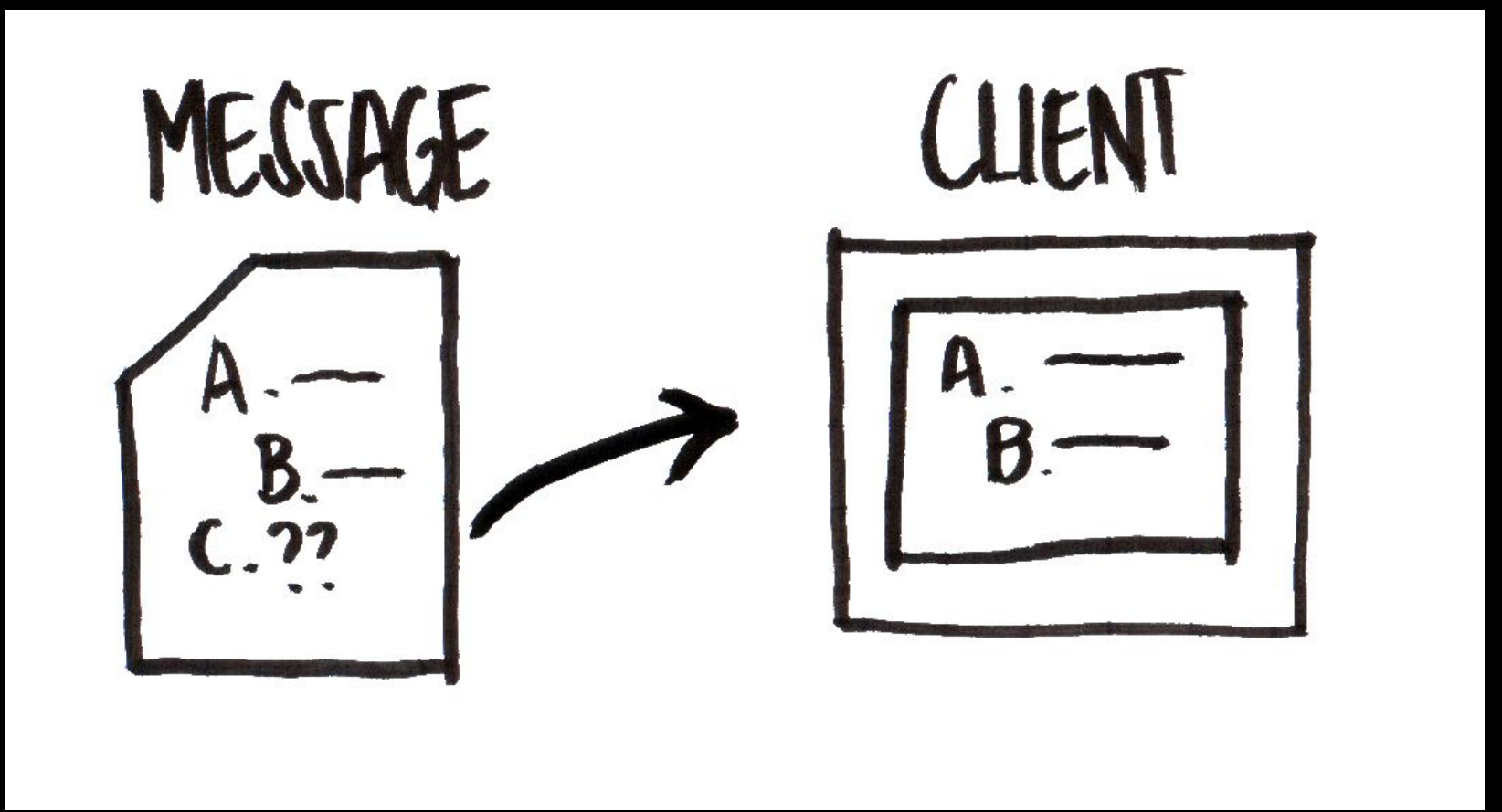


must IGNORE

Must Ignore

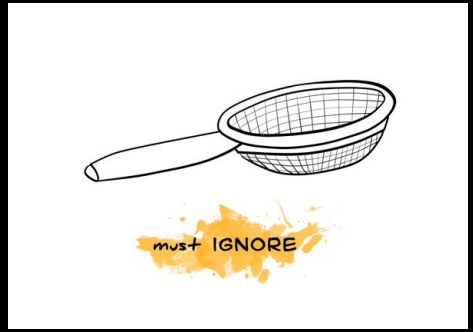
“The main goal of the MUST IGNORE pattern of extensibility is to allow backwards- and forwards-compatible changes.”

- David Orchard

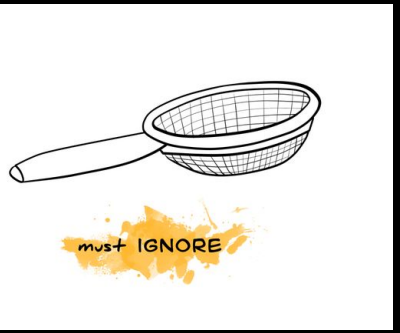

MESSAGE
A
B
C ??
CLIENT
A
B

Must Ignore

Clients MUST IGNORE any data/inputs that the client does not understand.



```
<script>
  /* incoming responseBody */
  {
     familyName: "Markov",
     givenName: "Shayne",
     dimensione_del_cappello: 12
  }
  ...
</script>

...

<!-- Rendering -->
<ul class="user">
  <li class="familyName">Markov</li>
  <li class="givenName">Shayne</li>
</ul>
```

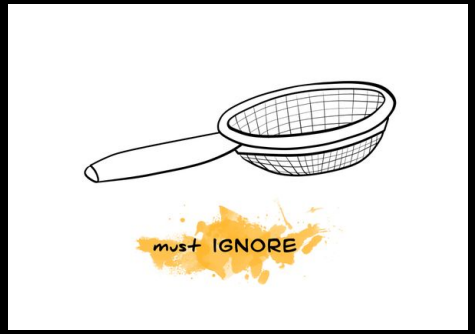


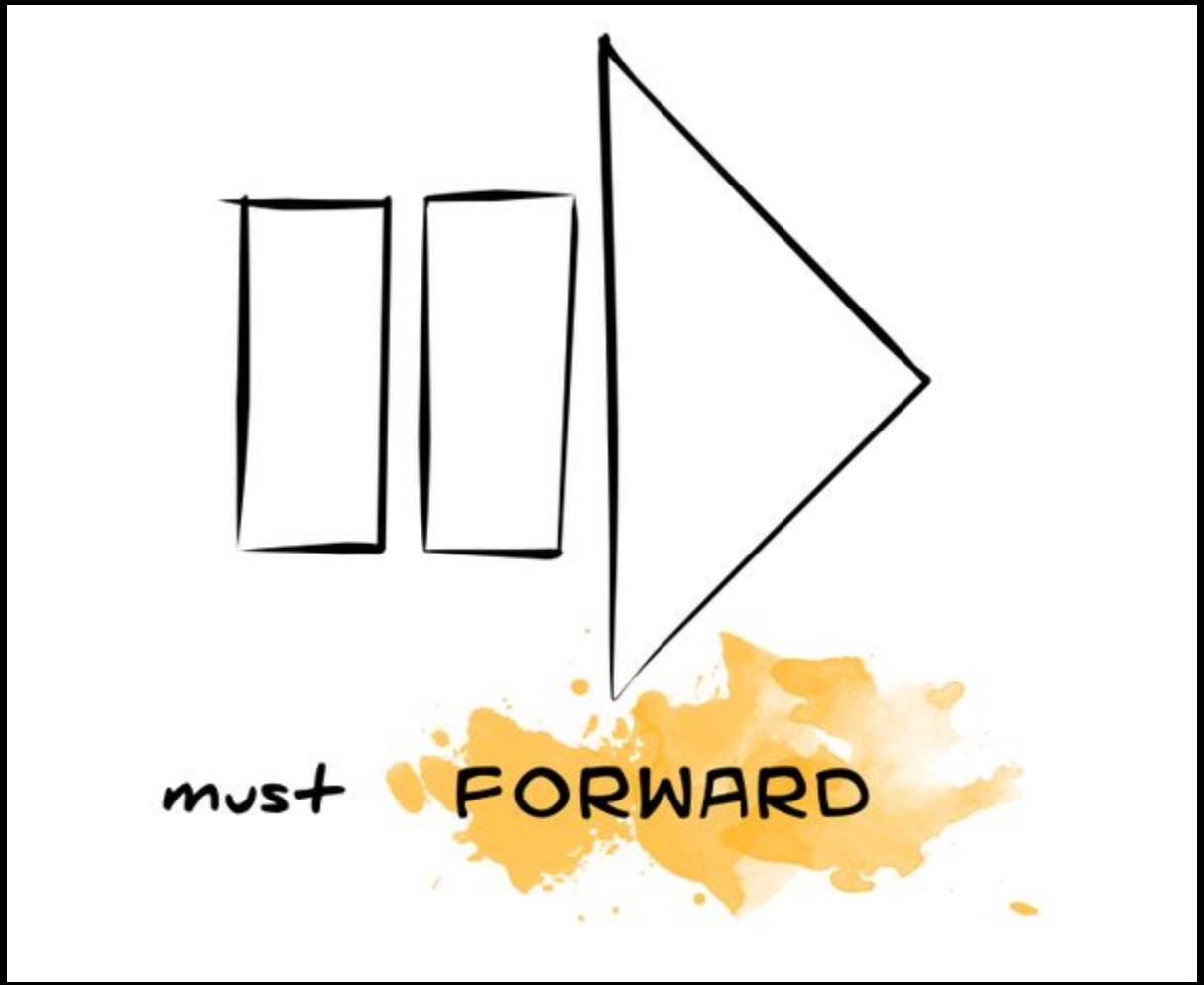
Must Ignore

What problem does this solve?

Ignoring what we don't understand lets us "do our own thing" w/o knowing everyone's job

Machines can now focus on their own job, not everyone's job.




must
FORWARD

MUST FORWARD

“A proxy MUST forward unrecognized header fields…”
-- RFC 7230

[Docs] [txt|pdf] [draft-ietf-httpbi...] [Diff1] [Diff2] [Errata]

PROPOSED STANDARD
Errata Exist

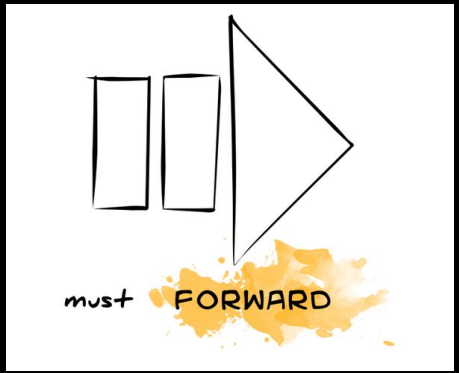
Internet Engineering Task Force (IETF) — R. Fielding, Ed., Adobe
Request for Comments: 7230 — J. Reschke, Ed., greenbytes
Obsoletes: 2145, 2616
Updates: 2817, 2818
Category: Standards Track — June 2014
ISSN: 2070-1721

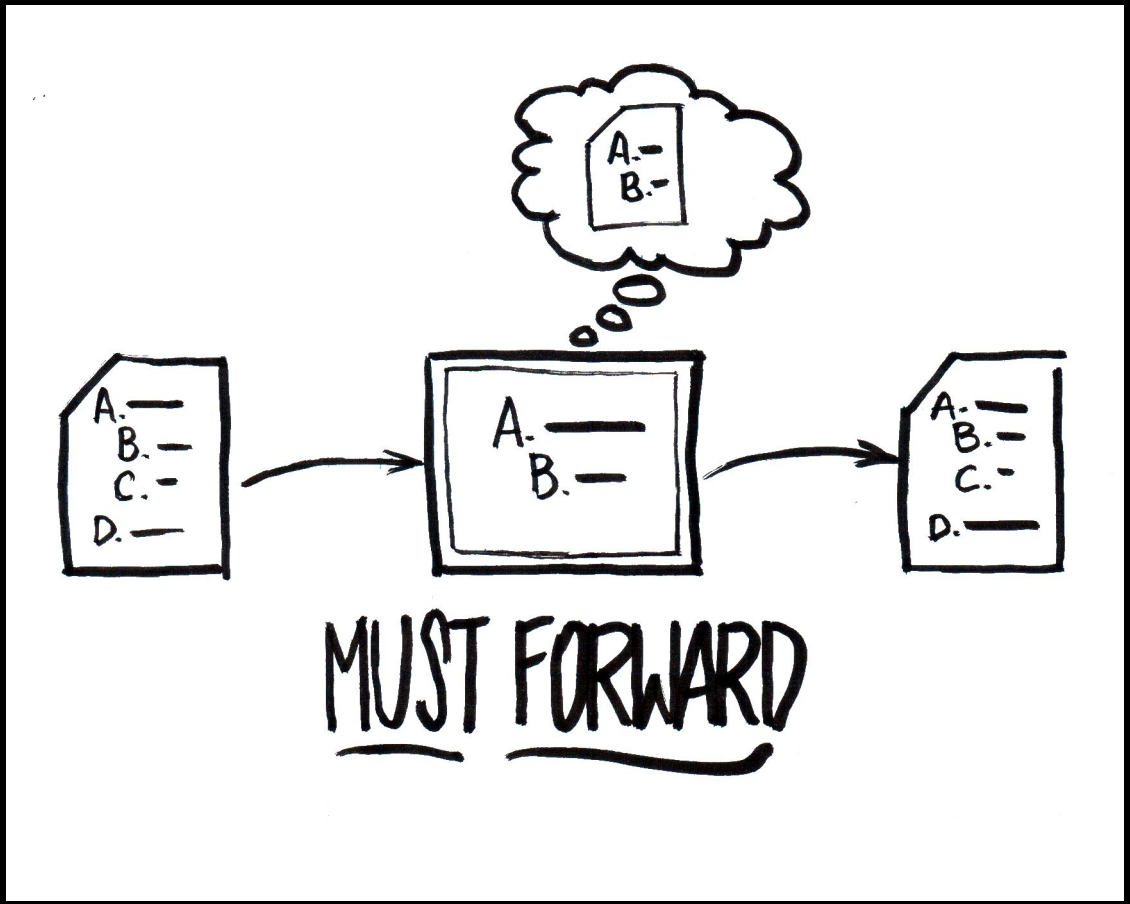
Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing

Abstract

The Hypertext Transfer Protocol (HTTP) is a stateless application-level protocol for distributed, collaborative, hypertext information systems. This document provides an overview of HTTP architecture and its associated terminology, defines the "http" and "https" Uniform Resource Identifier (URI) schemes, defines the HTTP/1.1 message syntax and parsing requirements, and describes related security concerns for implementations.

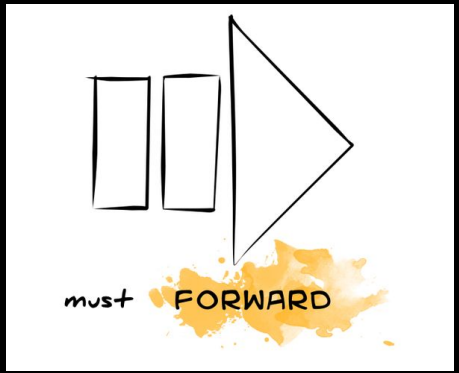
Status of This Memo



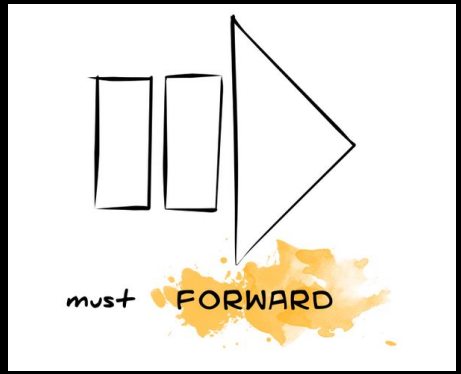

A.
B.
A.
B.
A.
B.
C.
D.
A.
B.
C.
D.
MUST FORWARD

Must Forward

Clients MUST FORWARD (unchanged) any input fields (URL or FORM) that the client does not recognize.



```
<ul class="user">
  <li class="familyName">Markov</li>
  <li class="givenName">Shayne</li>
  <li class="checksum">1qw2t3e5rt4u5</li>
</ul>

<script>
  ...
  function updateUser() {
    var fields = getInputs(class="user")
    for(var f in fields) {
      switch(f.name) {
        case "familyName":
          f.value="Markus";
          break;
        case "givenName":
          f.value="Ryane";
          break;
      }
    }
    updateForm(fields,class="update");
    updateForm.Send();
  }
  ...
</script>

<form class="update" action="..." method="post">
  <input type="hidden" name="checksum" value="" />
  <input type="text" name="familyName" value="" />
  <input type="text" name="givenName" value="" />
  <input type="submit" />
</form>
```

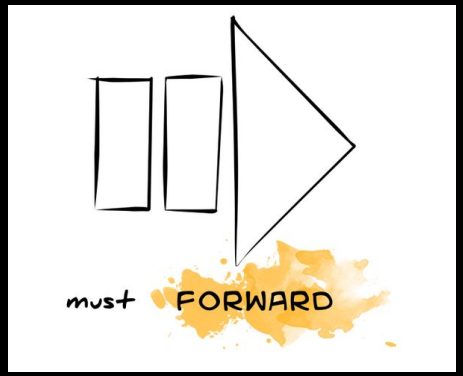


Must Forward

What problem does this solve?

We don’t edit for others around us.

Machines can now co-operate w/o full understanding of other’s work



provide MRU

Provide MRU (Most-Recently-Used)

"A feature of convenience allowing users to quickly see and access the last few used files and documents."
-- Wikipedia

Common menus in Microsoft Windows

From Wikipedia, the free encyclopedia

This is a **list of commonly used Microsoft Windows menus**.

Contents [hide]

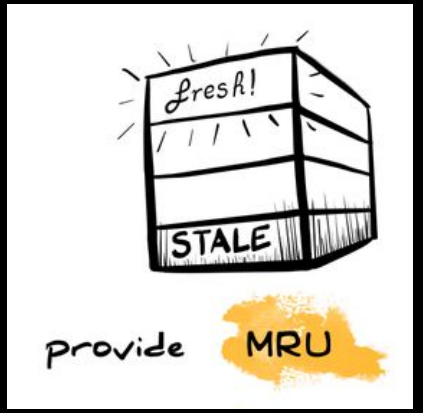
1 Microsoft menus
- 1.1 Most Recently Used menu
- 1.2 Properties menu
- 1.3 System menu

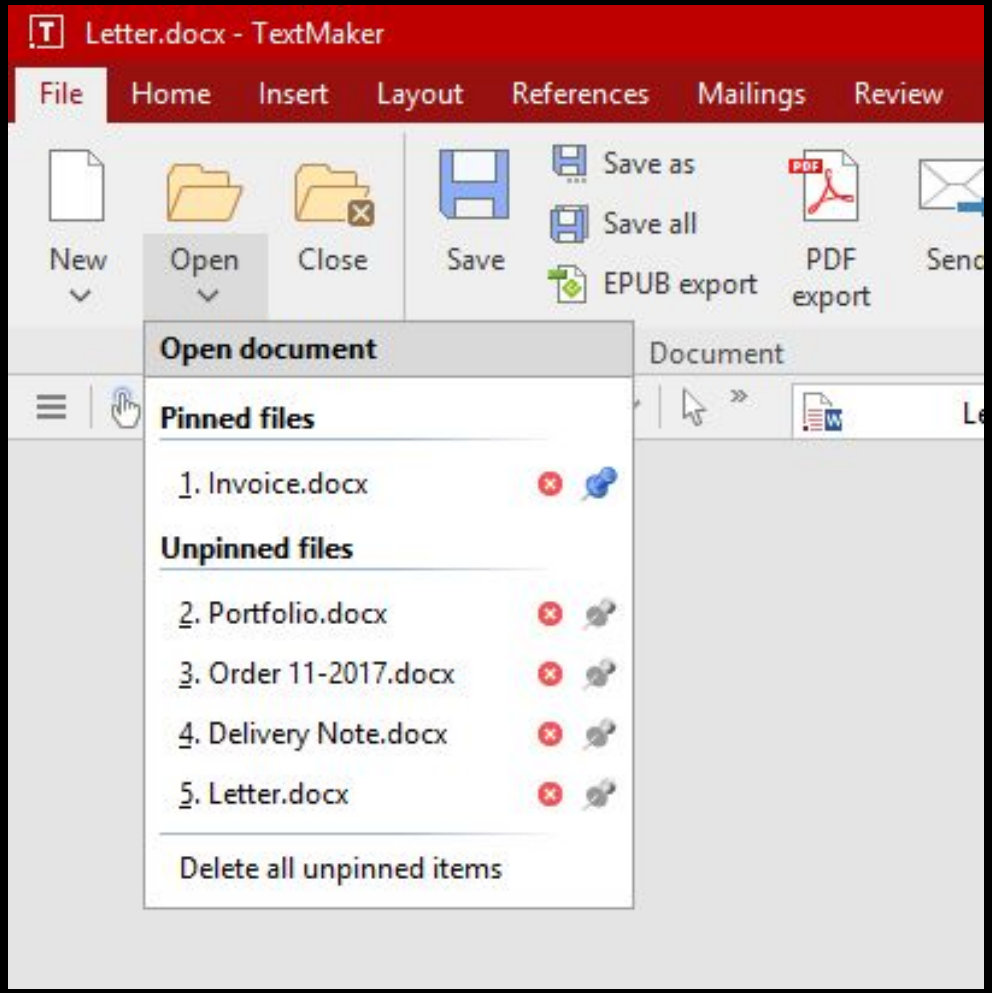
2 References

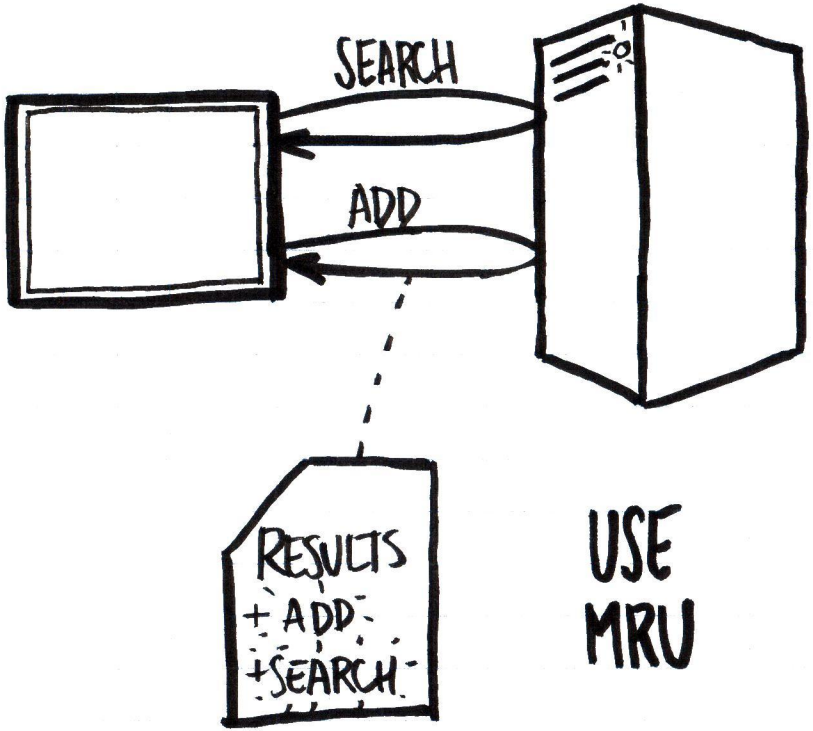
Microsoft menus [edit]

Most Recently Used menu [edit]

Most Recently Used (MRU) is a term used in computing to refer to the list of progr
quickly see and access the last few used files and documents, but could also be c

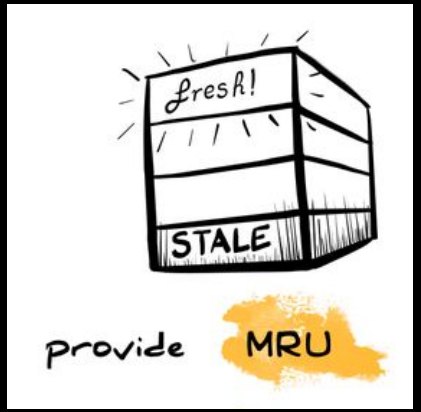



Letter.docx - TextMaker
File
Home
Insert
Layout
References
Mailings
Review
New
Open
Close
Save
Save as
Save all
EPUB export
PDF export
Document
Open document
Pinned files
1. Invoice.docx
Unpinned files
2. Portfolio.docx
3. Order 11-2017.docx
4. Delivery Note.docx
5. Letter.docx
Delete all unpinned items

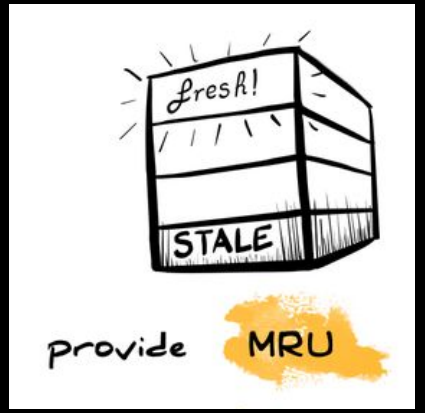

SEARCH
ADD
RESULTS
+ ADD
+ SEARCH
USE
MRU

Provide MRU

Services SHOULD return the most recently-used (MRU) LINKS and FORMS in all responses.

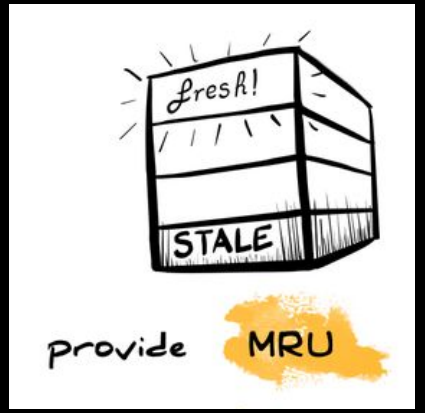


```
#### execute a SEARCH
*** REQUEST
GET /orders/search HTTP/1.1
  Host: example.org
  Accept: application/vnd.hal+json

*** RESPONSE
HTTP/1.1 200 OK
  Content-Type: application/vnd.hal+json
  Content-Length: XXXX

{
  "_links": {
    "self": {"href" : "..."},
    "search": {"href" : "/orders/search"}
    ...
}
```



```
#### execute a SEARCH
*** REQUEST
GET /orders/search HTTP/1.1
  Host: example.org
  Accept: application/vnd.hal+json

*** RESPONSE
HTTP/1.1 200 OK
  Content-Type: applic
  Content-Length: XXXX

{
  "_links": {
    "self": {"href" :
    "search": {"href"
    ...
}
```

```
#### execute an ADD
*** REQUEST
POST /orders/ HTTP/1.1
  Host: example.org
  Accept: application/vnd.hal+json

*** RESPONSE
HTTP/1.1 200 OK
  Content-Type: application/vnd.hal+json
  Content-Length: XXXX

{
  "_links": {
    "self": {"href" : "..."},
    "search": {"href" : "..."},
    "add": {"href" : "..."}
    ...
}
```

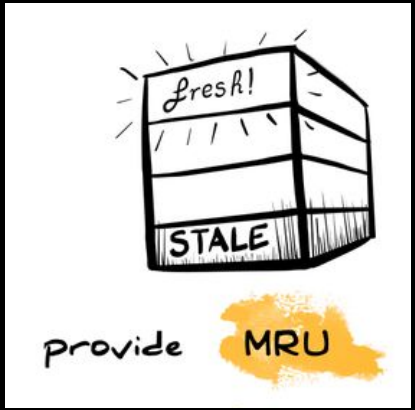


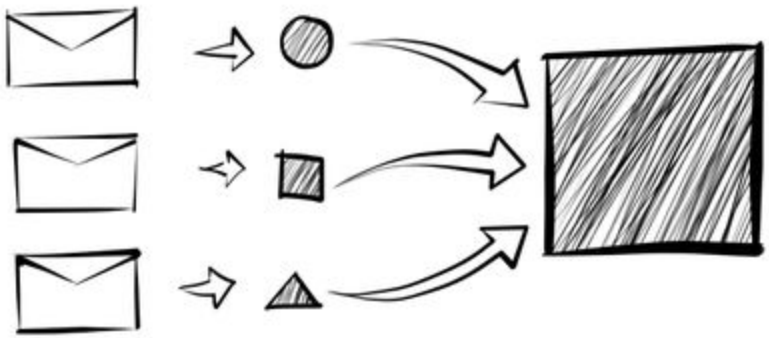
Provide MRU

What problem does this solve?

We need most-used tools close at hand

Machines can now find most-used affordances easily





use IDEMPOTENCY

Use Idempotence

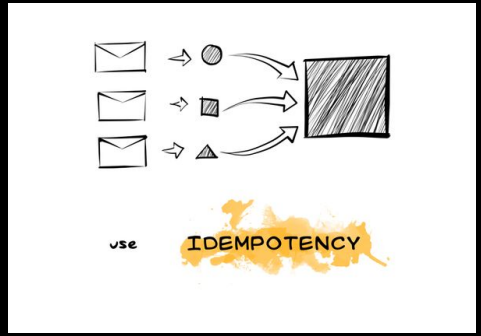
“Can be applied multiple times without changing the result beyond the initial application.”
-- Wikpedia

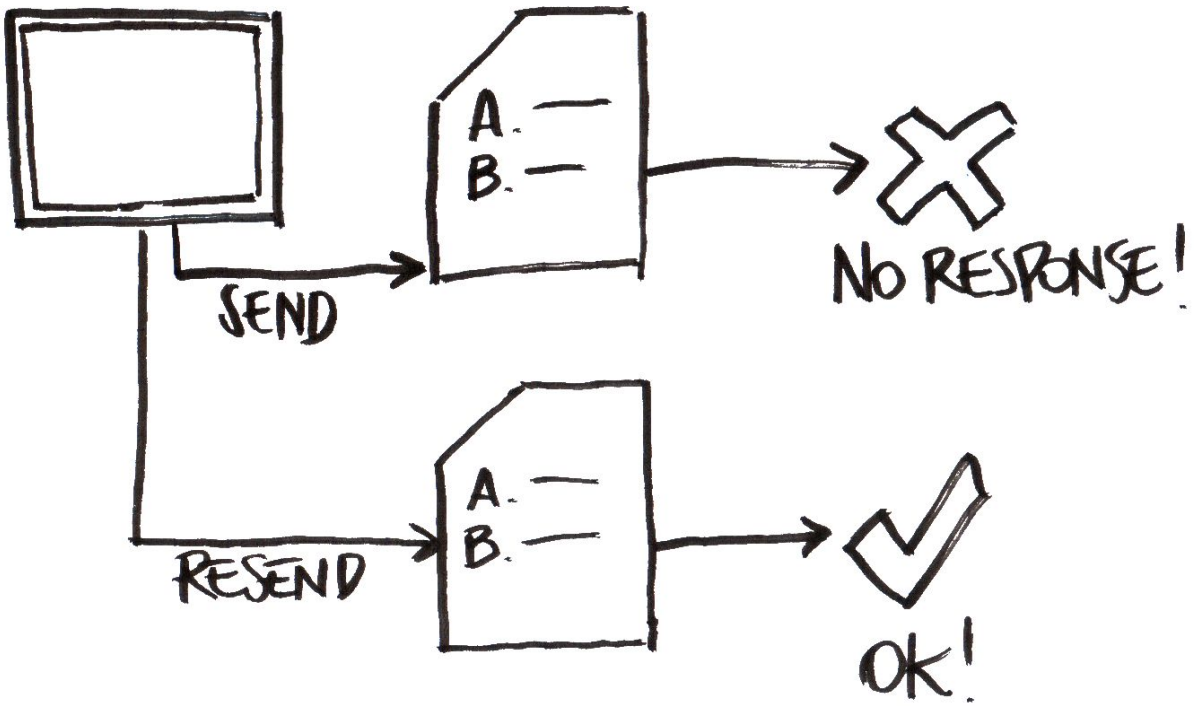
4.2.2. Idempotent Methods

A request method is considered "idempotent" if the intended effect on the server of multiple identical requests with that method is the same as the effect for a single such request. Of the request methods defined by this specification, PUT, DELETE, and safe request methods are idempotent.

Like the definition of safe, the idempotent property only applies to what has been requested by the user; a server is free to log each request separately, retain a revision control history, or implement other non-idempotent side effects for each idempotent request.

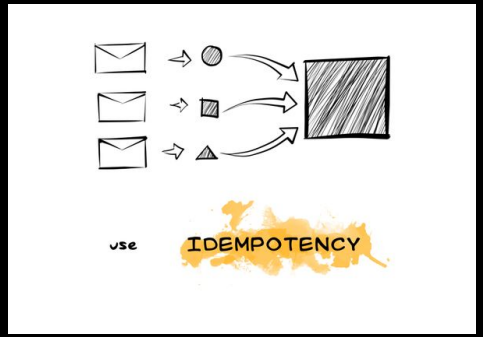
Idempotent methods are distinguished because the request can be repeated automatically if a communication failure occurs before the



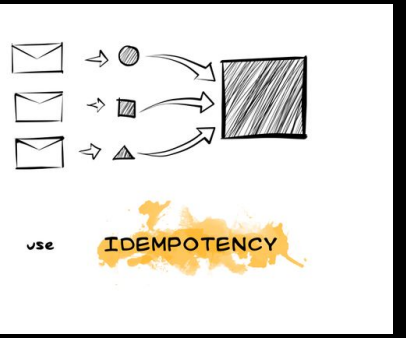

A.
B.
SEND
NO RESPONSE!
A.
B.
RESEND
OK!

Use Idempotence

All network requests SHOULD be idempotent in order to allow clients to safely repeat them when response is unclear.



```
var maxRequests = 5;
var options = {
  url: 'http://www.exmple.org/users/123',
  method: "put",
  body = user.formFields();
  timeout: 5000
}

function idempotentRequest(attempt){
  request(options, function(error,response,body){
    if(error){
      console.log(error);
      if(attempt==maxRequests)
        return;
      else
        idempotentRequest(attempt+1);
    }
    else {
      //do something with result
    }
  });
}

idempotentRequest(1);
```

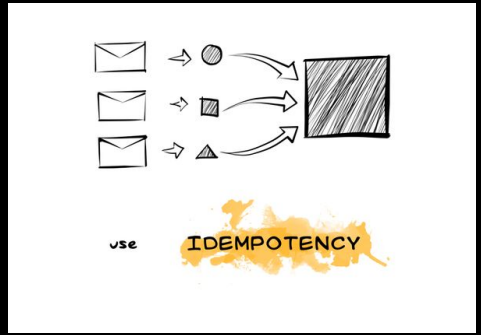


Use Idempotence

What problem does this solve?

If things didn’t work right the first time, we need to try again.

Machines can now safely “try again”



Networks

Programming the Network

There is no simultaneity at a distance!
-- Similar to the speed of light bounding information
-- By the time you see a distant object, it may have changed!
-- By the time you see a message, the data may have changed!

Pat Helland

"Data on the Inside vs. Data on the Outside, Helland (2005) http://cidrdb.org/cidr2005/papers/P12.pdf

Programming the Network

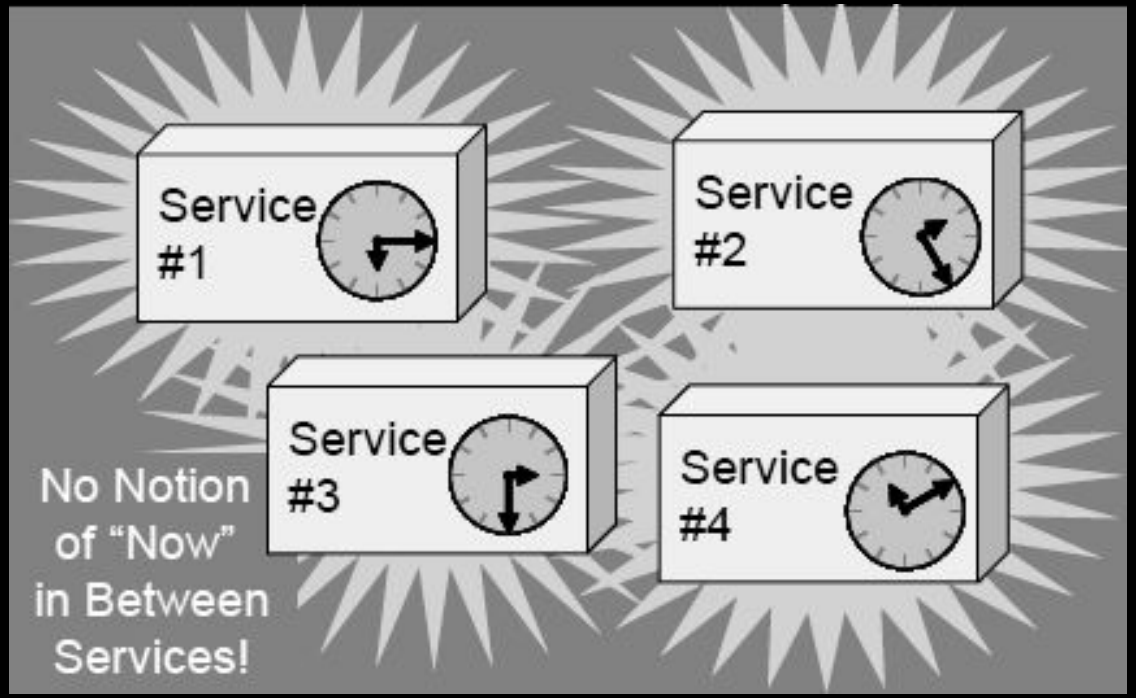
There is no simultaneity at a distance!
-- Similar to the speed of light bounding information
-- By the time you see a distant object, it may have changed!
-- By the time you see a message, the data may have changed!

Services, transactions, and locks bound simultaneity!
-- Inside a transaction, things are simultaneous
-- Simultaneity exists only inside a transaction!
-- Simultaneity exists only inside a service!

Pat Helland

"Data on the Inside vs. Data on the Outside, Helland (2005) http://cidrdb.org/cidr2005/papers/P12.pdf

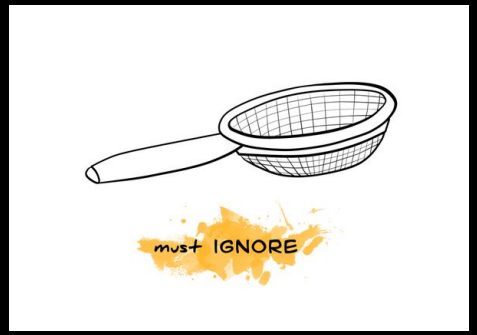
Programming the Network



Pat Helland

"Data on the Inside vs. Data on the Outside, Helland (2005) http://cidrdb.org/cidr2005/papers/P12.pdf

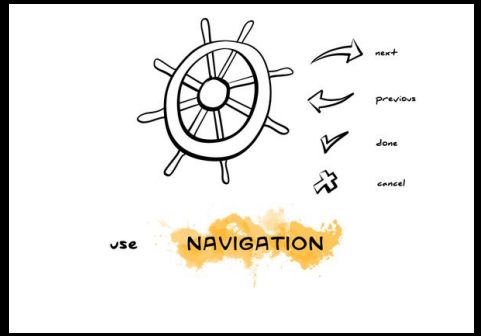
Build Network-Aware Implementations



Shared Agreements

Shared Agreements

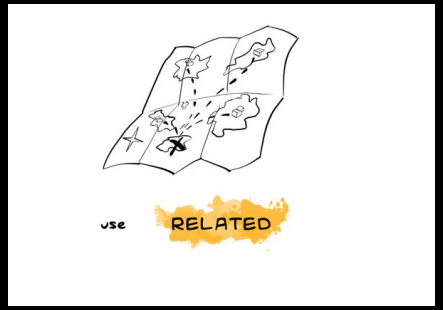
9. USE RELATED
10. USE NAVIGATION
11. PARTIAL SUBMIT
12. STATE WATCH

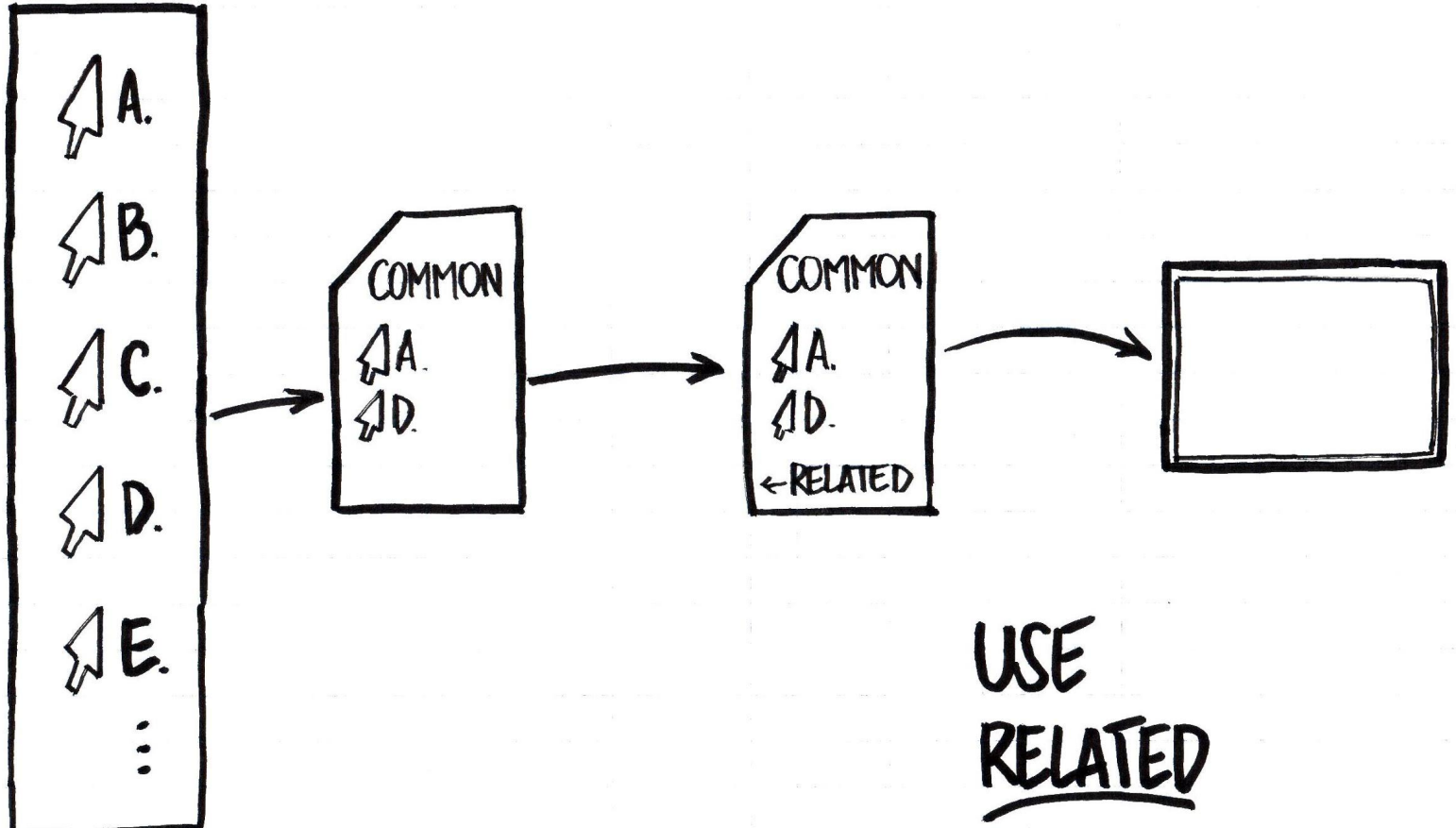


use RELATED

Use Related

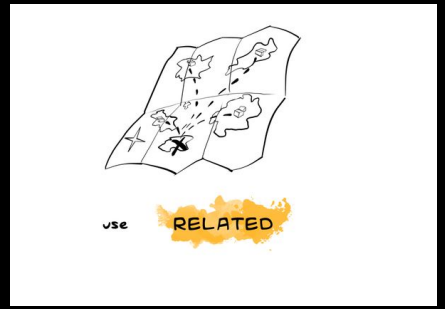
“By watching what you click on in search results, Google can learn that you favor particular sites.” – Danny Sullivan, 2009



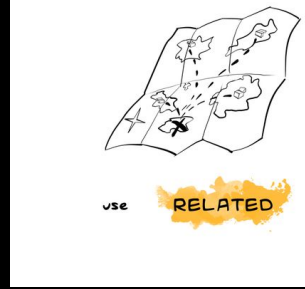

A.
B.
C.
D.
E.
COMMON
A.
D.
COMMON
A.
D.
←RELATED
USE
RELATED

Use Related

Services SHOULD return a RELATED LINK that responds with ALL the possible actions for this context.

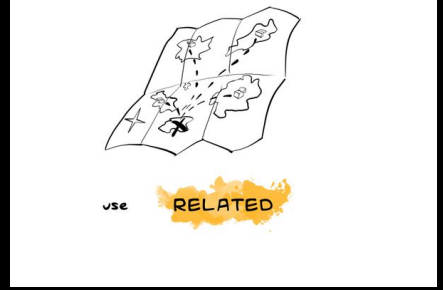


```
*** REQUEST
GET /orders/123 HTTP/1.1
  Host: example.org
  Accept: application/vnd.hal+json

*** RESPONSE
HTTP/1.1 200 OK
  Content-Type: application/vnd.hal+json
  Content-Length: XXXX

{
  "_links": {
    "self": {"href" : "..."},
    "approve": {"href" : "..."},
    "related": {"href" : "/orders/123?related"}
    ...
}
```



```
*** REQUEST
GET /orders/123 HTTP/1.1
  Host: example.org
  Accept: application/vnd.hal+json

*** RESPONSE
HTTP/1.1 200 OK
  Content-Type: applicat
  Content-Length: XXXX

{
  "_links": {
    "self": {"href" : ".
    "approve": {"href" :
    "related": {"href" :
    ...
}
```

```
*** RESPONSE
HTTP/1.1 200 OK
  Content-Type: application/vnd.hal+json
  Content-Length: XXXX

{
  "_links": {
    "self": {"href" : "..."},
    "approve": {"href" : "..."},
    "cancel": {"href" : "..."},
    "modify": {"href" : "..."},
    "transfer": {"href" : "..."},
    "review": {"href" : "..."},
    "rush": {"href" : "..."},
    "related": {"href" : "/orders/123?related"}
    ...
}
```

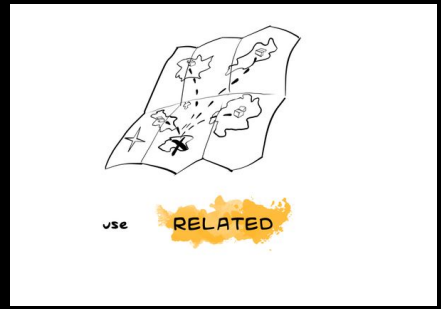


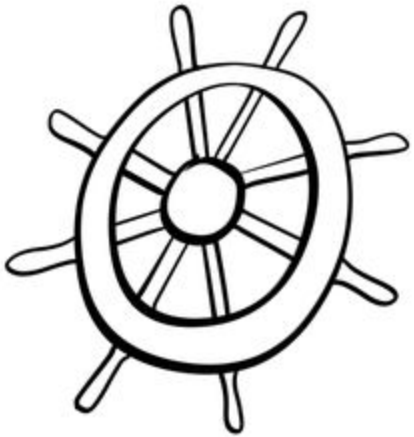
Use Related

What problem does this solve?

I can’t remember everything, need an easy way to look up instructions.

Machines can now “look up”
the available affordances.





next

previous

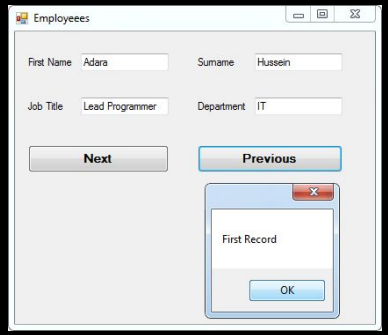
done

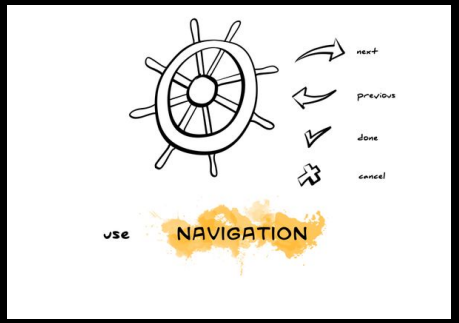
cancel

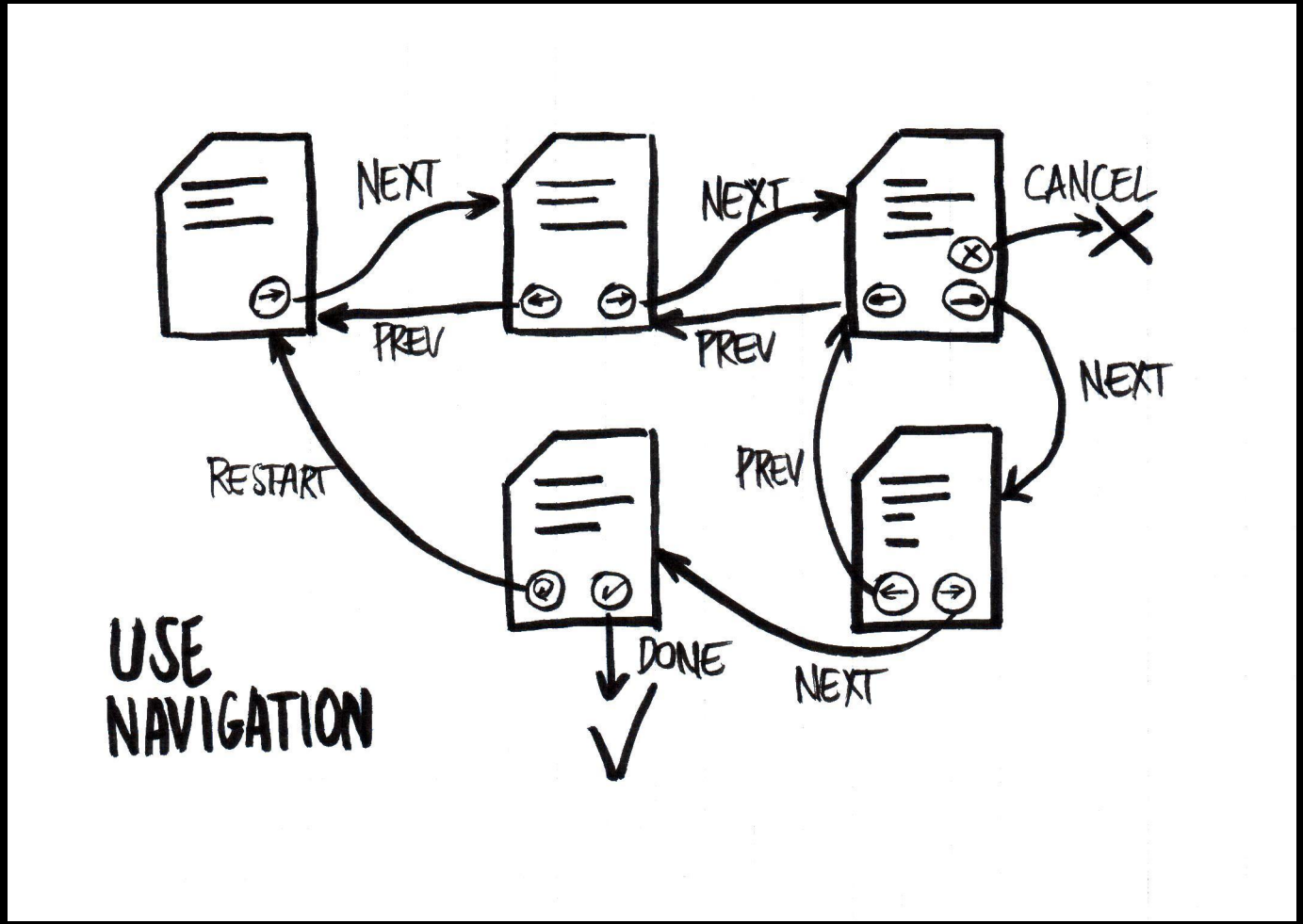
use NAVIGATION

Use Navigation

“To achieve a single goal which can be broken down into dependable sub-tasks.”
-- Design Patterns (@uipatterns)

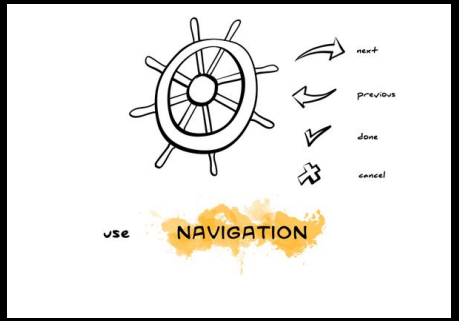




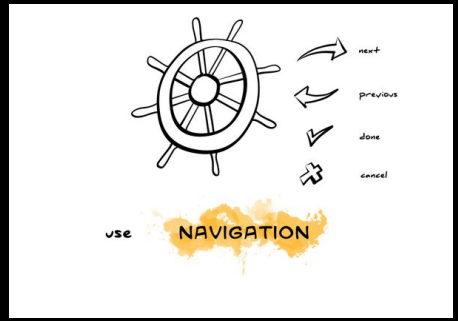

NEXT
NEXT
CANCEL
PREV
PREV
NEXT
RESTART
PREV
DONE
NEXT
USE
NAVIGATION

Use Navigation

Services SHOULD provide "next/previous" LINK to handle multi-step workflow with "cancel", "restart", & "done."



```
// evaulate options
var lookingFor = "next";
var msg = getCurrentResponseBody();
switch (lookingFor) {
  case "done":
    if(msg.findNavigation(lookingFor)) {
      processDone(msg);
    }
    break;
  case "cancel":
    if(msg.findNavigation(lookingFor)) {
      processCancel(msg);
    }
    break;
  case "restart":
    if(msg.findNavigation("restart")) {
      processRestart(msg);
    }
    break;
  case "previous":
    if(msg.findNavigation("previous")) {
      processPrevious(msg);
    }
    break;
  case "next":
    if(msg.findNavigation("next")) {
      processNext(msg);
```

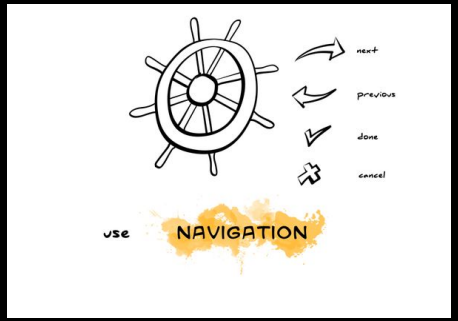


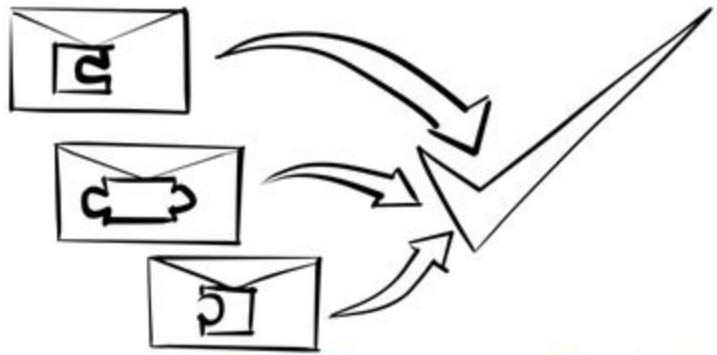
Use Navigation

What problem does this solve?

I can’t keep all the steps in my head

Machines can now navigate through a long series of steps safely.

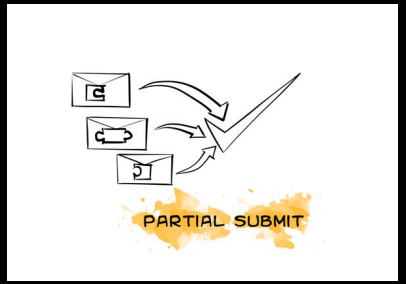


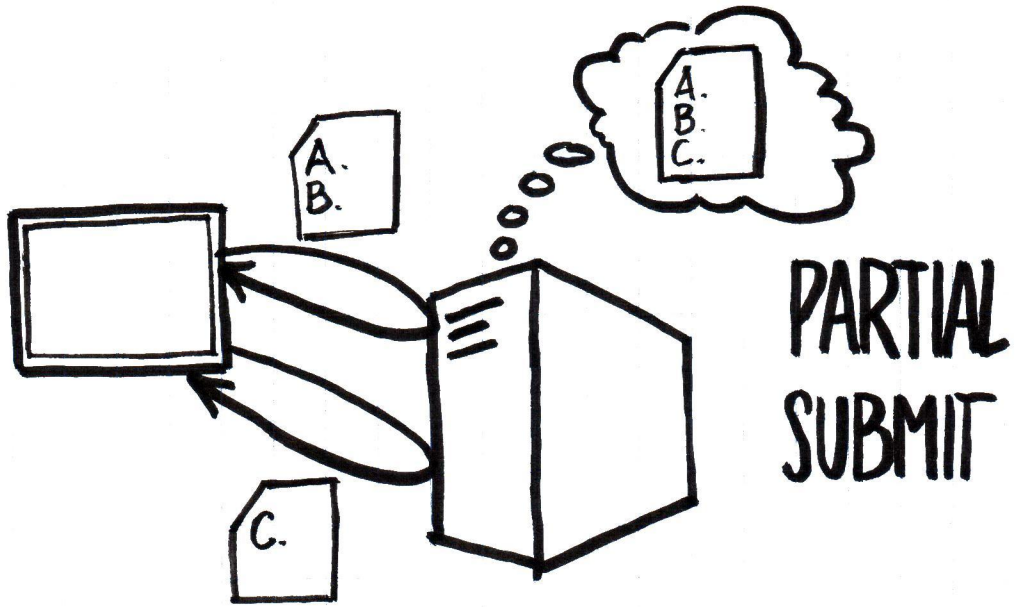


PARTIAL SUBMIT

Partial Submit

“Think of the actions as approximations of what is desired.”
-- Donald Norman

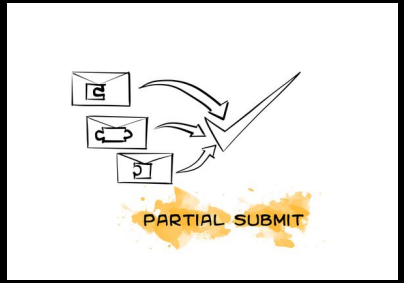




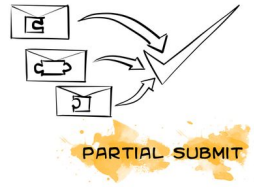
PARTIAL SUBMIT

Partial Submit

Services SHOULD accept partially filled-in FORM and return a new FORM with the remaining fields.



```
// partial submit processing
...
case "POST":
  neededInputs = processForm(suppliedInputs);
  if(neededInputs.length>0) {
    responseBody = generateForm(
      neededInputs,
      actions["done","cancel","restart","previous"]
    );
  }
  else {
    responseBody = generateResults();
  }
  break
...
```

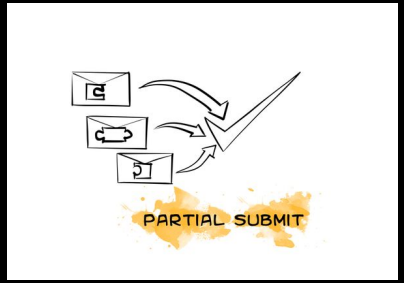


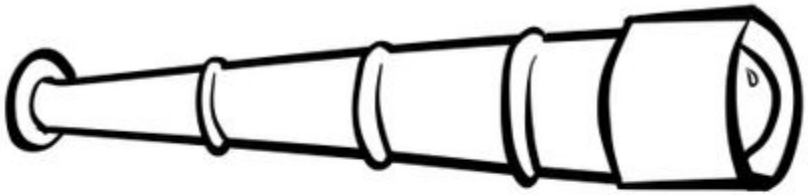
Partial Submit

What problem does this solve?

I sometimes only know part of the story.

Machines can now interact in small parts and not always be perfect.



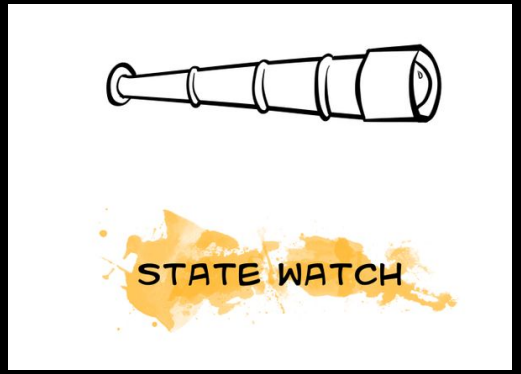


STATE WATCH

State Watch

“Data representing variables in a dynamical system…”

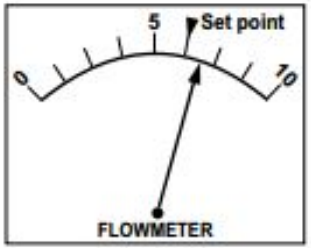
-- Jens Rassmussen



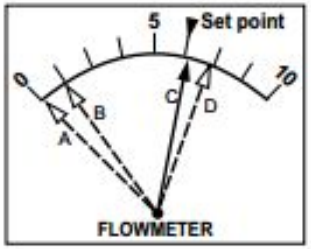
State Wate

“Data rep *a dynamical system…”*

Rasmussen



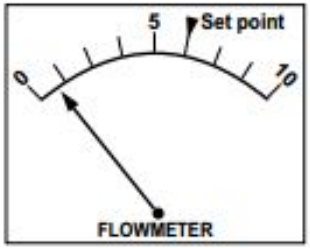
SIGNAL
- Keep at set point
- Use deviation as error signal
- Track continuously



SIGN

Stereotype acts

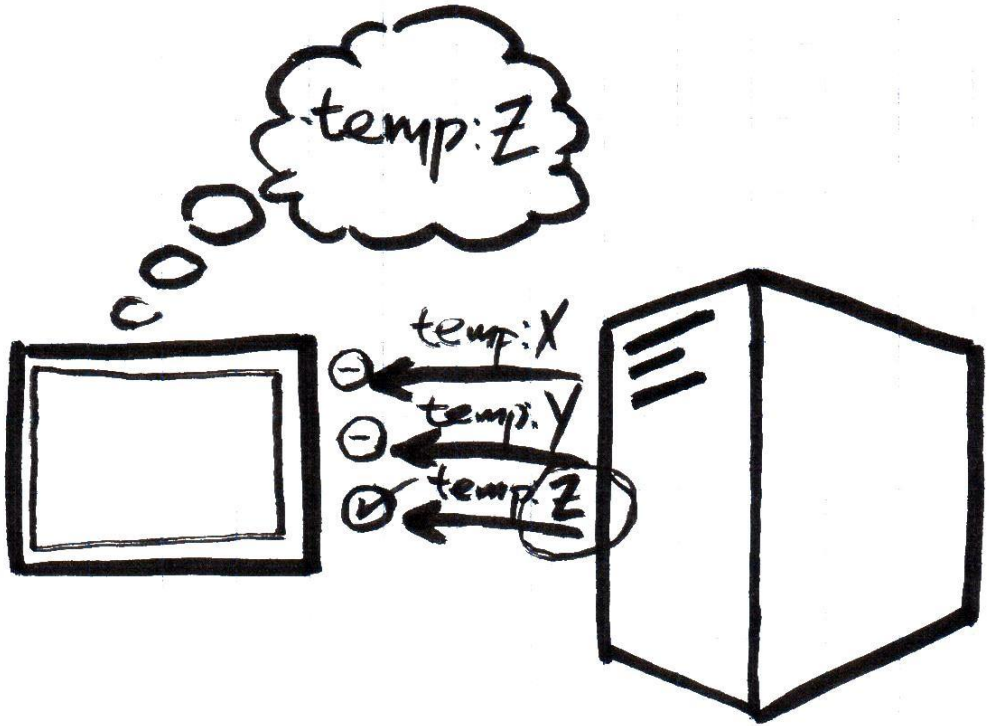
If Valve Open	If C, ok If D, adjust flow
If Valve Closed	If A, ok If B, recalibrate meter



SYMBOL

If, after calibration, is still B, begin to read meter and speculate functionally (could be a leak)

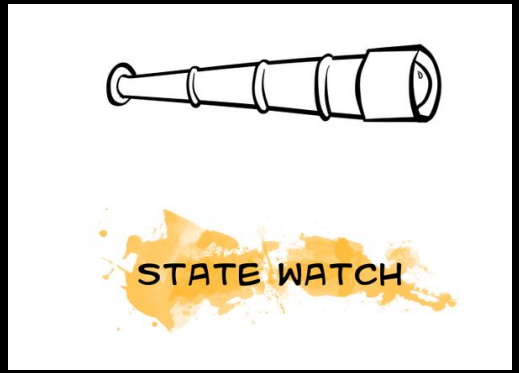
STATE WATCH



STATE WATCH

State Watch

Services SHOULD allow clients to subscribe to WATCH VALUES so that clients can determine "done."



```
*** REQUEST
POST /heat-mgmt HTTP/1.1
     Host: example.org
     Content-Type: application/x-www-form-urlencoded
     Accept: application/vnd.collection+json
     Prefer: state-watch="sensor5,temp13"

     sensor5=increase by .5c;

*** RESPONSE
HTTP/1.1 200 OK
     Content-Type: application/collection+json
     Preference-Applied: state-watch="sensor5,sensor13"
     Content-Location: /heat-mgmt

     {"collection" :
       {
         "items" : [
           ...
           {
             "href" : "/heat-mgmt/sensor5",
             data: [
               {"name" : "device", "value" : "sensor5"},
               {"name" : "reading", "value" : "14.5c"}
             ]
           }
           ...
         ]
       }
     }
```

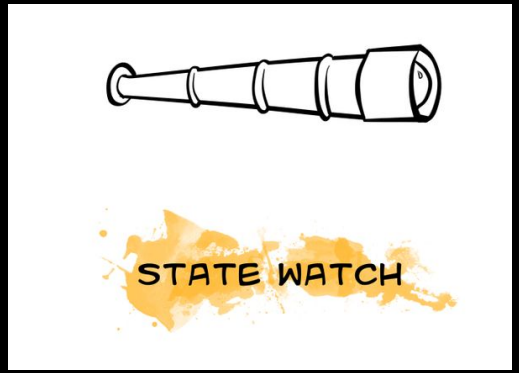
STATE WATCH

Use State Watch

What problem does this solve?

My boss doesn't always set my goals.

Machines can now set their own goals and act accordingly.



Hypermedia Affordance

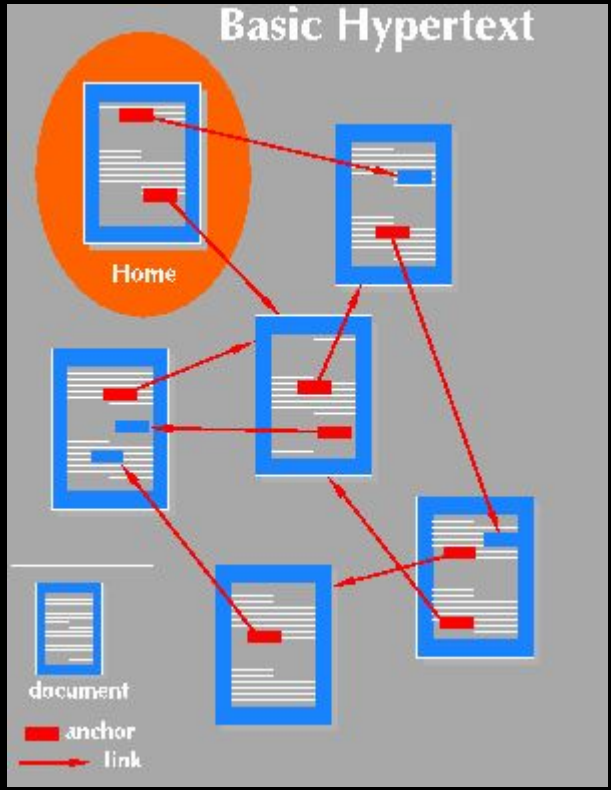
Ted Nelson's hyperlinks (1965)

The words hypertext, hyperlink, and hypermedia were coined by Ted Nelson around 1965.

Ted Nelson

What is Hypermedia?

[Hypermedia] *is not constrained to be linear. Hypertext is text which contains links to other texts.*



https://www.w3.org/WhatIs.html

Affordances

*"The **affordances** of the environment are what it offers ... what it provides or furnishes, either for good or ill.*

James Gibson, 1977

James Gibson

Ecological Approach to Visual Perception, Gibson, 1979

Affordances

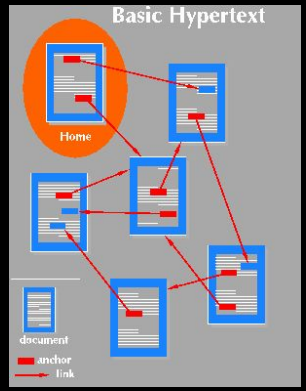
*"When I say Hypertext, I mean the simultaneous presentation of information and controls such that the information becomes the **affordance** through which the user obtains choices and selects actions."*

Roy Fielding, 2008

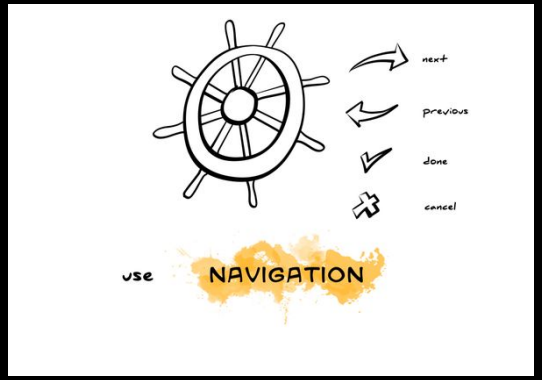
Roy Fielding

Architectural Styles and the Design of Network-based Software Architectures, Fielding, 2001

Affordances are the reason for hypermedia

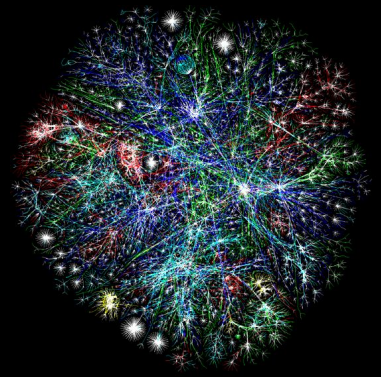


Enable Connected Services



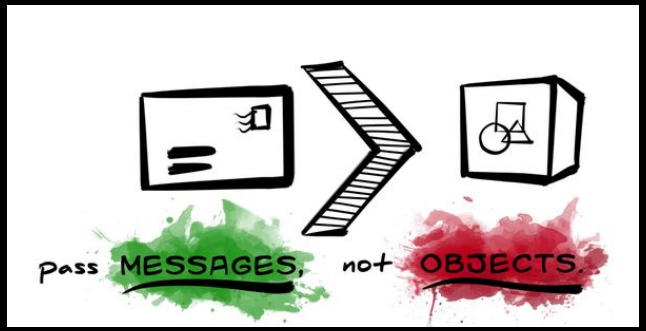
Summary

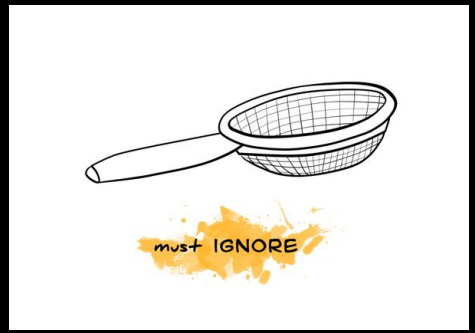
Programming the Network brings new challenges

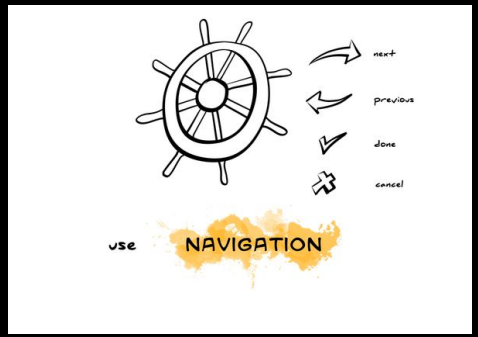


Twelve Patterns for Evolvable APIs

Four Design Patterns
Four Basic Principles
Four Shared Agreements

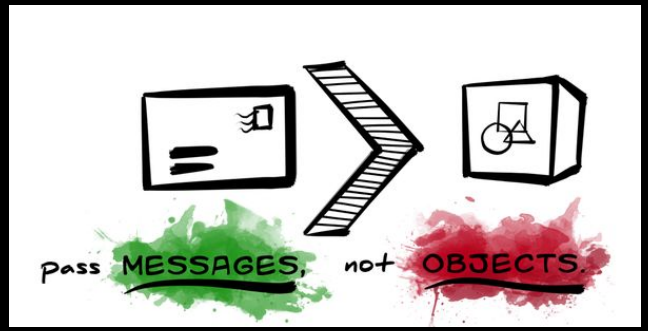






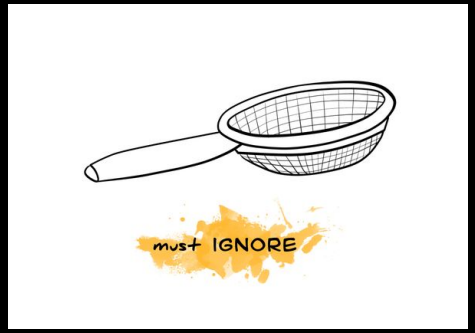
Design Patterns (for interop)

1. PASS MESSAGES, NOT OBJECTS
2. SHARE VOCABULARIES, NOT MODELS
3. THE REPRESENTOR PATTERN
4. PUBLISH PROFILES



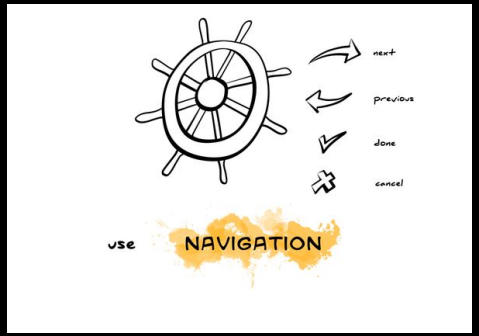
Basic Principles (for networks)

5. MUST IGNORE
6. MUST FORWARD
7. PROVIDE MRU
8. USE IDEMPOTENCE



Shared Agreements (for services)

9. USE RELATED
10. USE NAVIGATION
11. PARTIAL SUBMIT
12. STATE WATCH



README.md

Twelve Patterns Materials

References and examples for the patterns covered in my YOW 2017 talk: "Twelve Patterns for Evolvable APIs"

1. Pass Messages, Not Objects
2. Share Vocabularies, Not Models
3. Use the Representor Pattern
4. Publish Profiles
5. Must Ignore
6. Must Forward
7. Provide MRU
8. Use Idempotence
9. Related Links
10. Use Navigation
11. Partial Submit
12. State Watch

http://g.mamund.com/12-patterns


Changes Ahead


END
DETOUR


PLAN
STRATEGY
NOW


Twelve Patterns to Create Evolvable APIs
Mike Amundsen
API Academy / CA
@mamund
Drawings by Diogo Lucas
@diogoclucas