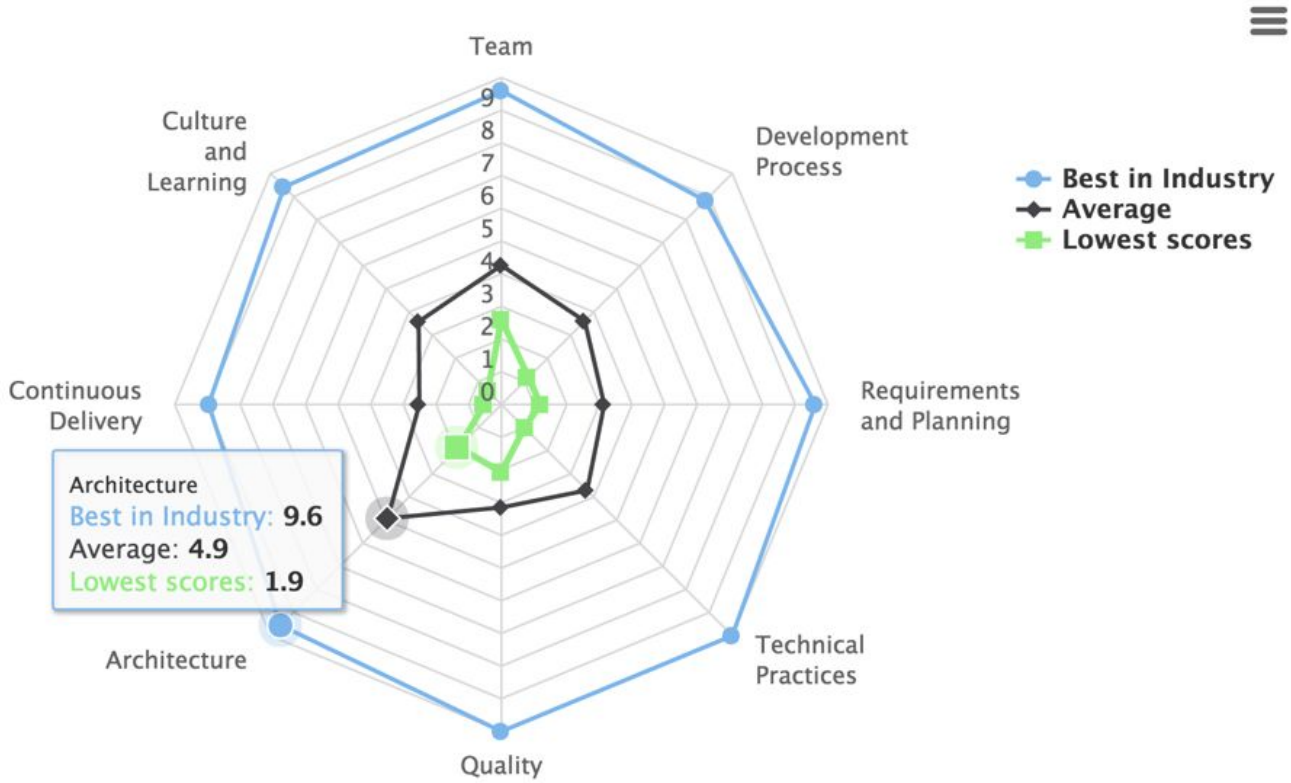
HYPR
Uncode. Unleash

Hypermedia API

Design Patterns

HYPR
Uncode. Unleash

Build for Speed with Callaghan Innovation



HYPR
Uncode. Unleash

Build for Speed

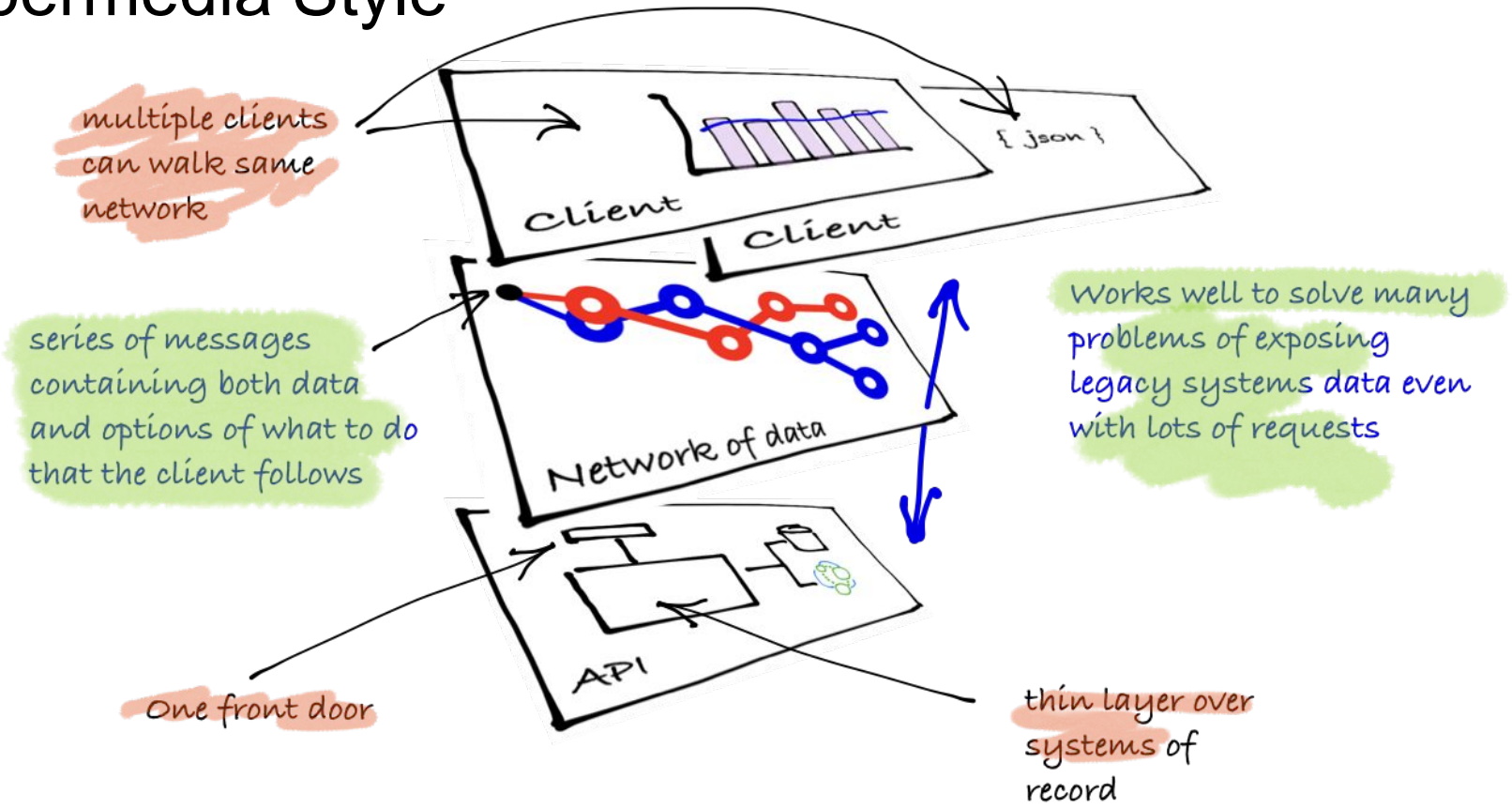
HYPR
Uncode. Unleash

Industry Issues We've Seen

- Trying to Scale: Heavy load on server and database
  - But hard to make progress
  - Monolithic persistence, database as point of integration
- Slow down in getting features done
- Technical debt
  - Tangled code, ui state, business logic, persistence
  - Hard to evolve
  - Untestable or untested architecture
- Version 2 -> problems

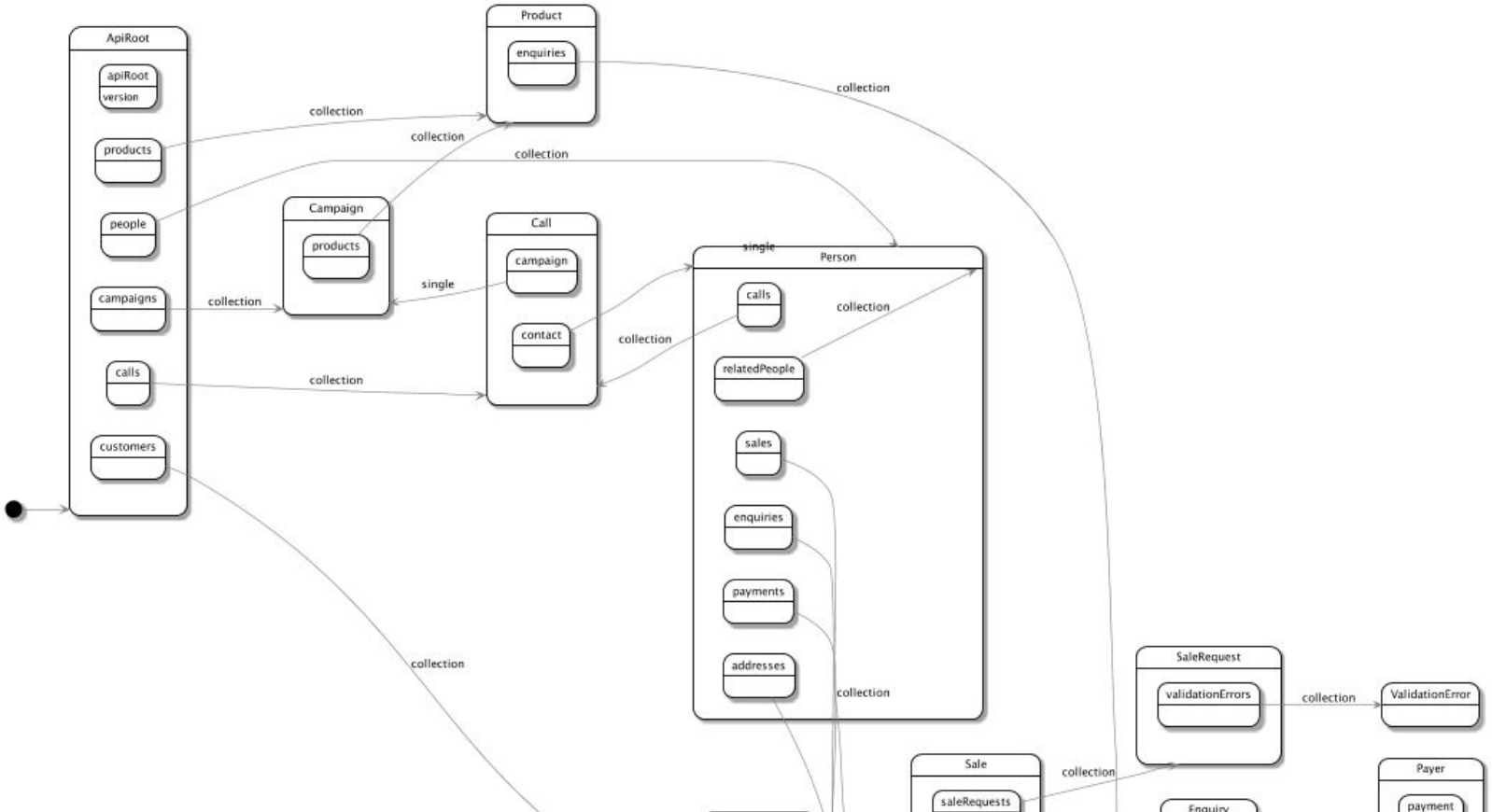
HYPR
Uncode. Unleash

Hypermedia Style

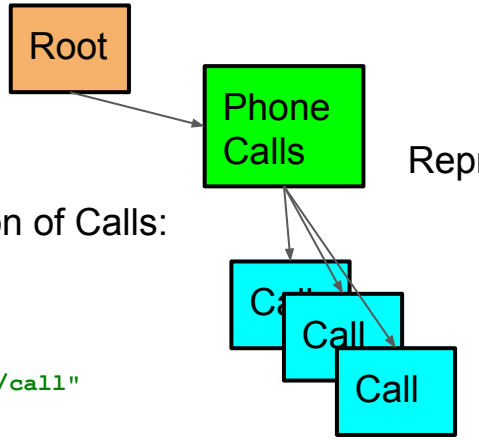


HYPR
Uncode. Unleash

Network of Data - A Domain Graph



Micro Format



Representation of a Collection of Calls:

```
{
 "links": [
   {
     "rel": "self",
     "href": "https://example.com/call"
   },
   {
     "rel": "up",
     "href": "https://example.com/"
   },
   {
     "rel": "search",
     "href": "https://example.com/call/search"
   },
   {
     "rel": "create-form",
     "href": "https://example.com/call/create-form"
   }
 ],
 "items": [
   {
     "id": "https://example.com/call/68",
     "title": "Complete"
   }
 ]
}
```

Representation of a Single Call Resource:

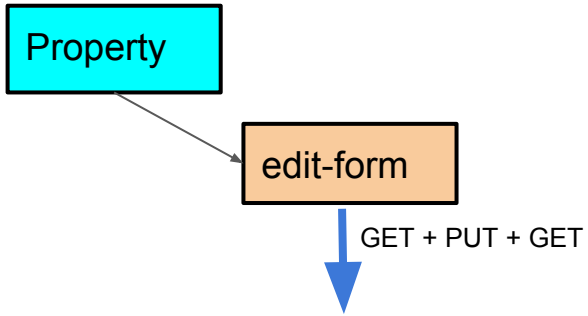
```
{
 "links": [
   {
     "rel": "self",
     "href": "https://example.com/call/68"
   },
   {
     "rel": "up",
     "href": "https://example.com/"
   },
   {
     "rel": "edit-form",
     "href": "https://example.com/call/68/edit-form"
   },
   {
     "rel": "contact",
     "href": "https://example.com/person/117"
   },
   {
     "rel": "campaign",
     "href": "https://example.com/campaign/0"
   }
 ],
 "start": "2016-03-18T09:37:02.827",
 "end": "2016-03-18T09:38:30.98",
 "state": "Complete",
 "outcome": "AnswerMachine",
 "moreInformationPreference": "NotSet"
}
```

HYPR
Uncode. Unleash

Micro Format

FormRepresentation in JSON:

```
{
 "links": [
   {
     "rel": "self",
     "href": "https://example.com/property/2/edit-form"
   },
   {
     "rel": "up",
     "href": "https://example.com/property/2"
   }
 ],
 "items": [
   {"id": "titleNumber", "type": "string"},
   {"id": "legalDescription", "type": "string"}
 ]
}
```



FormRepresentation in HTML:

```
<html>
    <head>
      <title>Form</title>
      <link rel="self"
            href="https://example.com/property/2/edit-form" />
      <link rel="up"
            href="https://example.com/property/2" />
    </head>
    <body>
      <form action='https://example.com/property/2/edit-form'
            method='POST'
            enctype='application/x-www-form-urlencoded'>
        <label for='titleNumber'>TitleNumber:</label>
        <input id='titleNumber'
               type='text'
               name='TitleNumber'></input>
        <label for='legalDescription'>LegalDescription:</label>
        <input id='legalDescription'
               type='text'
               name='LegalDescription'></input>

        <input type="submit" value="Submit">
      </form>
    </body>
</html>
```

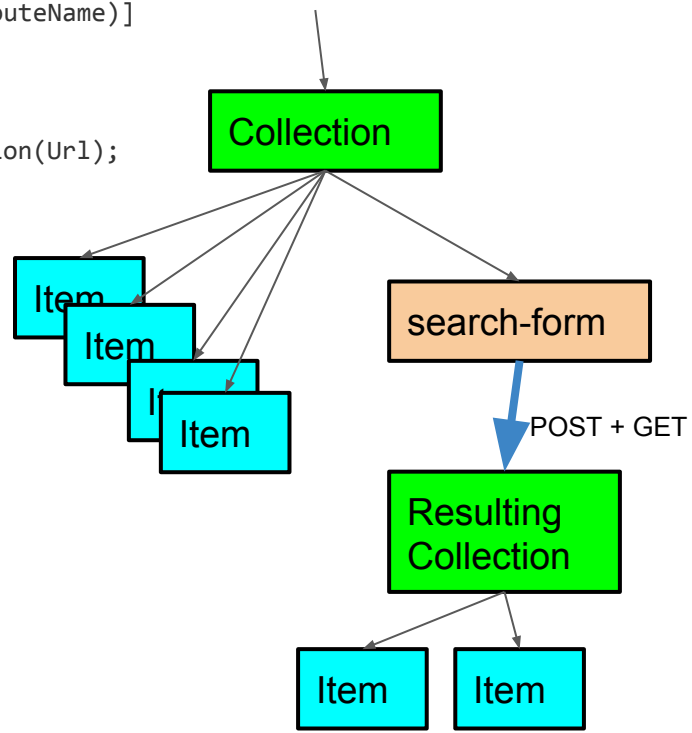
Note: Edited for readability

1. Small Scale Patterns

HYPR
Uncode. Unleash

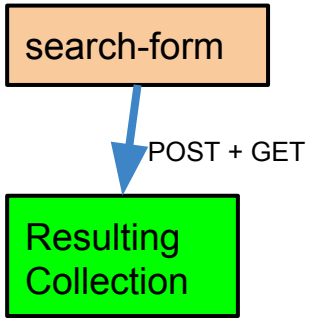
1.1 Search

```
[GET("stockLowNotification/search", RouteName = StockLowNotificationSearchRouteName)]
public FormRepresentation GetSearchForm()
{
    return new StockLowNotificationRepresentation().ToSearchFormRepresentation(Url);
}

[POST("stockLowNotification/search")]
public HttpResponseMessage Search([FromBody] SearchRepresentation criteria)
{
    return Url
       .MakeStockLowNotificationListUri(criteria.Search)
       .MakeCreated(Request, "Search resource created");
}
```



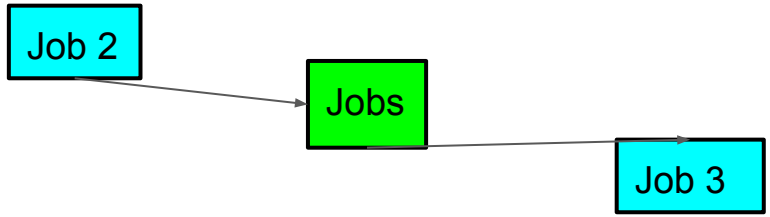
Search

```
{
 "message": "The resource has been created",
 "status": "Search resource created",
 "id": "https://example.com/stockLowNotification/searchresult?q=a"
}
```




```
[GET("stockLowNotification/searchresult", RouteName = StockLowNotificationListRouteName)]
public FeedRepresentation GetSearchResults([FromUri(Name = "q")] string search)
{
    return _stockLowNotificationRepository
        .GetByName(search)
        .ToSearchResultRepresentation(Url, search);
}
```

1.2 Recursion




```
{
 "links": [
   {
     "rel": "self",
     "href": "https://example.com/job/2"
   },
   {
     "rel": "up",
     "href": "https://example.com/product/2/job"
   },
   {
     "rel": "edit-form",
     "href": "https://example.com/job/2/edit-form"
   },
   {
     "rel": "jobs",
     "href": "https://example.com/job/2/job"
   }
 ],
 "description": "An endless job..."
}
```

```
{
 "links": [
   {
     "rel": "self",
     "href": "https://example.com/job/3"
   },
   {
     "rel": "up",
     "href": "https://example.com/job/2/job"
   },
   {
     "rel": "edit-form",
     "href": "https://example.com/job/3/edit-form"
   },
   {
     "rel": "jobs",
     "href": "https://example.com/job/4/job"
   }
 ],
 "description": "One job after another..."
}
```

HYPR
Uncode. Unleash

Recursion

```
[GET("job/{jobId:int}/job", RouteName = JobsUriFactory.JobsOnJobRouteName)]
public FeedRepresentation GetJobsOnJob(int jobId)
{
    return _jobRepository
        .GetByJob(jobId)
        .ToRepresentationOnJob(Url, jobId);
}

[GET("job/{jobId:int}/job/create-form", RouteName = JobsUriFactory.JobOnJobCreateFormRouteName)]
public FormRepresentation GetJobCreateForm(int jobId)
{
    return new JobRepresentation().ToCreateFormRepresentationOnJob(Url, jobId);
}
```

2. Medium Scale Patterns

HYPR
Uncode. Unleash

2.1 Network of Data as an Anti-Corruption Layer

- Clean domain model
- API maps between that model and the ugly backend
- A single change through the front-end could led to several databases being changed
- The mappings evolve as the backend is cleaned up

HYPR
Uncode. Unleash

2.2 Client-Side Hydration Strategies

- UI needs certain data to be loaded
  - Some eagerly, some lazy
- ORM approach
- We hydrate the required subgraph of the network of data
  - Recursively

HYPR
Uncode. Unleash

Client-side Hydration

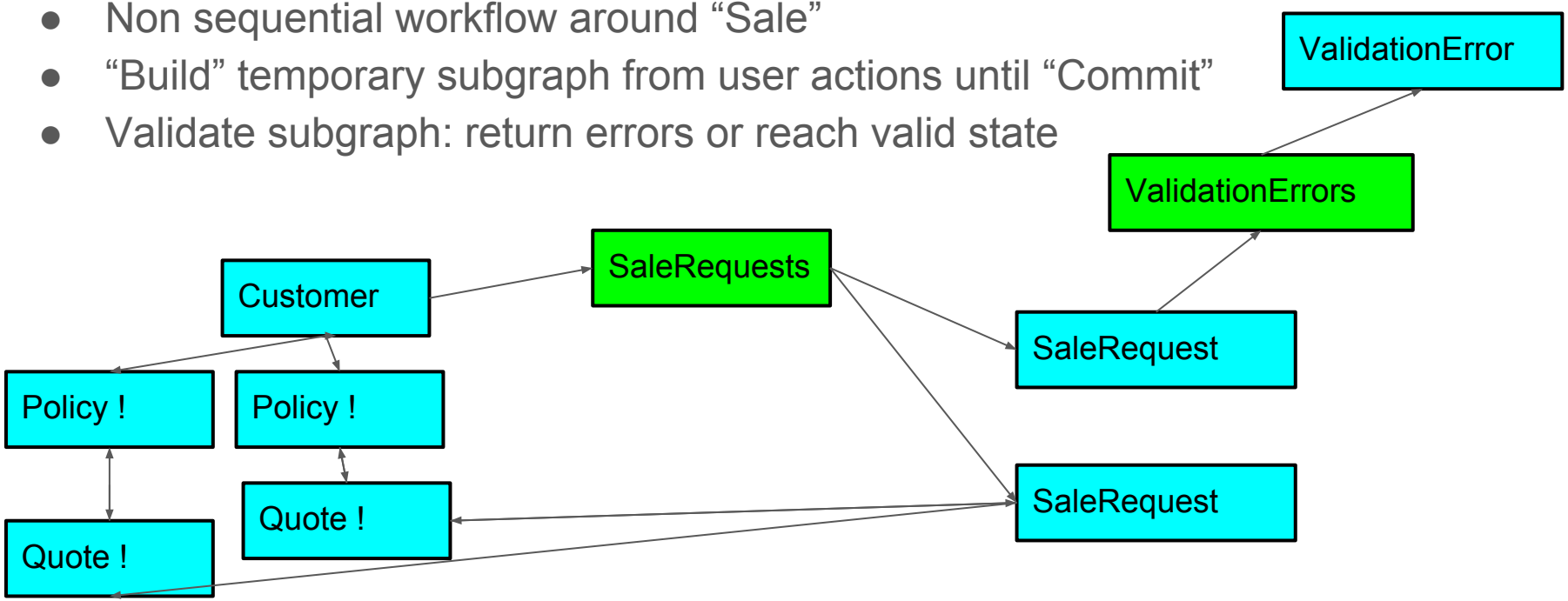
```
export class Person extends RestDomain {
    @mobx calls: CollectionRepresentation<Call>;
    @mobx relatedParties: CollectionRepresentation<Party>;
  ….
   hydrateTransitively() {
        return this
            .hydrateRelationships([addresses_, notes_], {alsoHydrateChildren: true})
            .then(()=> this.hydrateRelationships([relatedParties_, calls_, sales_, policyEnquiries_],
                {transitiveHydrate: true}));
    }
}

const calls_ = {name: 'calls', toMany: true, optional: true, make: () => Call.make};
const relatedParties_ = {...};
const addresses_ = {...};
...
```

HYPR
Uncode. Unleash

2.3 Business Transactions in REST

Network of data contains data and state (**Re**presentational **S**tate **T**ransfer)

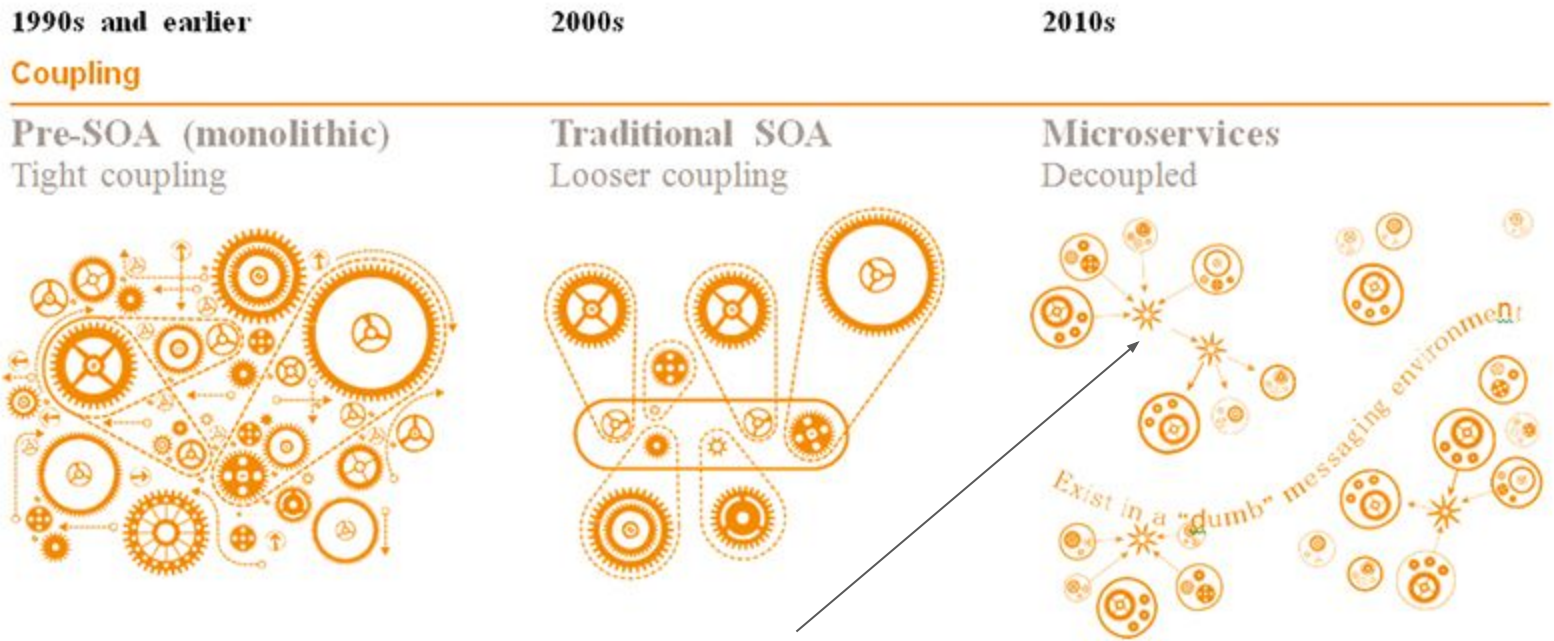
- Non sequential workflow around “Sale”
- “Build” temporary subgraph from user actions until “Commit”
- Validate subgraph: return errors or reach valid state



3. Large Scale Patterns

HYPR
Uncode. Unleash

3.1 Hypermedia & Microservices

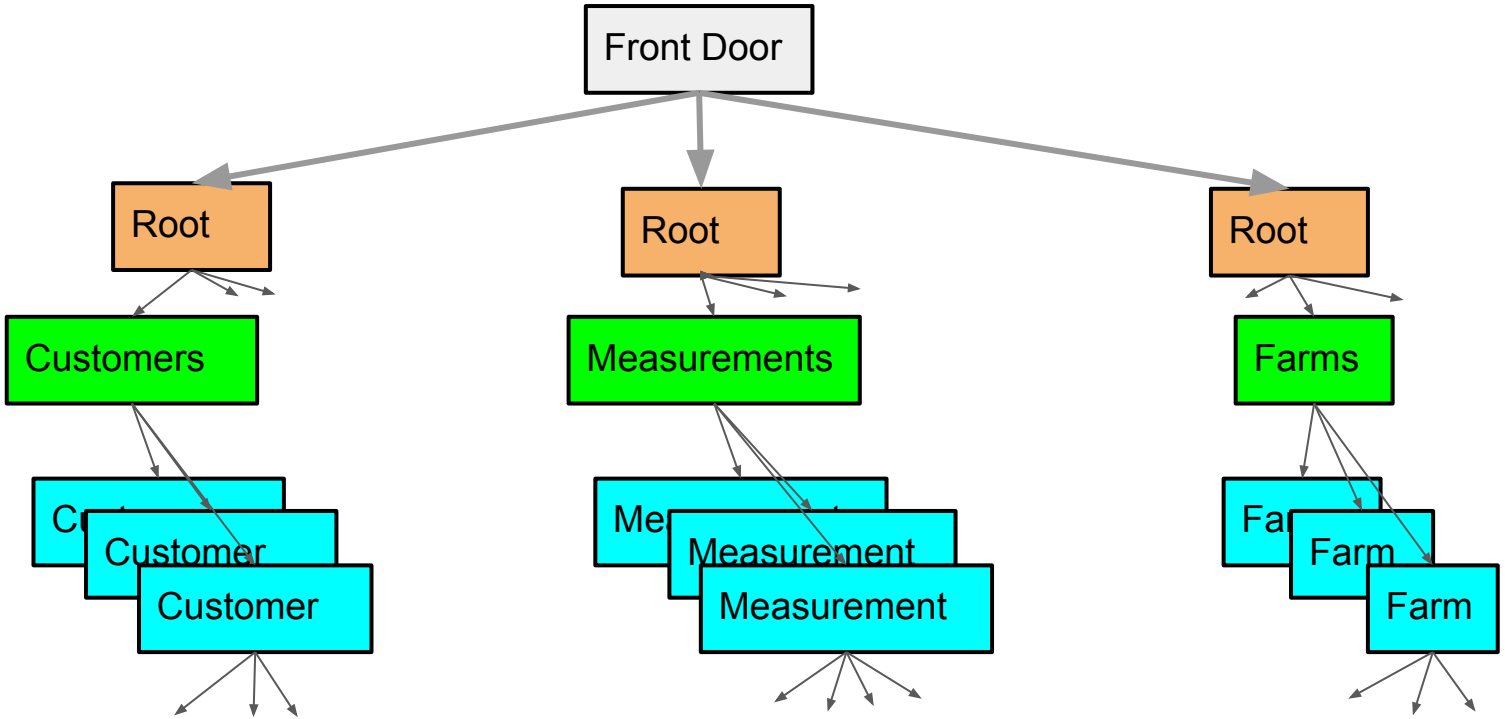


This looks like a graph!

http://www.pwc.com/us/en/technology-forecast/2014/cloud-computing/features/microservices.html

HYPR
Uncode. Unleash

Its graphs all the way down...



HYPR
Uncode. Unleash

Questions?

Gareth Evans

gareth.evans@hypr.co.nz

@gareth__evans

Rick Mugridge

rick.mugridge@hypr.co.nz

HYPR
Uncode. Unleash

Hypermedia Pros

- Secure - one front door
- Workflow and data means smaller clients
- Simple - typically 80% less code
- Fast - async loading
- Scaleable - clients do more work, caching
- Microformat - more general/simpler clients
- Version resilience
- Separation of concerns (front and back end)
- Encourages composition (small reusable components)
- Refactoring strategy