Getting Things Done with REST

http://ianSrobinson.com
@ianSrobinson
Getting Things Done
hinges. Listening at the door, you hear strange
mutterings and the clatter of what could be pots and
pans. Whatever is in there, there are several of
them. Do you want to go through the door (turn to
199) or turn back (turn to 237)?

79
The passageway ends in front of you in a dead end
If you wish to search for secret passageways, turn to
337. If not, return to the crossroads at 207.

80
The key fits the lock and opens the door. You find
yourself in a large boathouse. Various beams, in
different stages of construction, are lying around.
Apart from the door behind you, there is another in
the north wall. As you enter, the Skeletons stop
their work and crane their bony necks around to
look at you. They pick up planks of wood and ham-
mers and advance towards you. These are five of
them. Do you:

Smile nervously and back out of
the door into the passage? Turn to 129
Tell them you’re come about
buying a boat? Turn to 131
Tell them you’re their new boss
and order them back to work? Turn to 195
Draw your sword and prepare for
battle? Turn to 140

595–597
The old man raises a hand. “Not so fast!”
He has a small pocketbook in his hands.
He opens it and shows you the contents:
several coins, a map, a small scroll.
He offers to sell you the scroll for
20 gold pieces. What will you do?

913
Try something else from your
backpack? Turn to 122

195
This is a rather unlikely story, considering that they
see few humans around. Nevertheless, Skele-
tons are pretty dim – you know this and that’s why
you tried the story. Roll one die. If you roll 1 or 2,
they don’t believe you and keep on advancing. Turn
to 140.

A 3 or 4 means that they aren’t sure, and send two of
their number off through the north door whilst the
rest hold you at bay with their weapons. Turn to
161.

A roll of 5 or 6 means they’ve believed you and they
all get back to work! Turn to 9. Add 2 Luck points.

196
You search the room. Try as you may you cannot
find the secret switch to open the door in the book-
shelf – the old man must have locked it from the
outside. You do find 5 gold pieces in a drawer in the
table. You decide to return to the junction to the
south. Turn to 286

197
At the top of the stairs the passage turns sharply to
Executing a specialism in a generalized way

**Warlock of Firetop Mountain Protocol**

- Go north
- Defeat goblin
- Take key
- Unlock door
- Go east
- Solve riddle

**Fighting Fantasy Transfer Protocol**

- Numbered prose paragraphs
- Multiple choices keyed to numbered paragraphs
Hypermedia
As
the
Engine
of
Application
State
Procurement process

- Customer
  - Request quote
  - Order
  - Confirm order
  - Cancel
  - Pay

- Supplier
CHAPTER 2: INTRODUCING RESTBUCKS: HOW TO GET A COFFEE, WEB STYLE

Although Restbucks stems from modest roots, its coffee quality and increasingly positive reputation help it to continue to grow. To help scale the business, Restbucks decides to hire a cashier to speed things up. With a cashier busy handling the financial aspects of the operation, the barista can concentrate on making coffee. The customer’s interactions aren’t affected, but Restbucks now needs to coordinate the cashier’s and barista’s tasks with a low-ceremony approach using sticky notes. The interactions (or protocol) between the cashier and the barista remain hidden from customers. Now that we’ve got two moving parts in our coffee shop, we need to think about how to encapsulate them, which leads to the scenario shown in Figure 2-5.

By implementing this scheme, Restbucks decouples ordering and payment from the coffee preparation. In turn, it is possible for Restbucks to abstract the inner workings of the shop through a façade. While the customer gets the same good coffee, Restbucks is free to innovate and evolve its implementation behind the scenes.
Server-side Development
Divide and conquer

home  rfq  quote/123  order-form/123

202 Accepted

order/987

303 See Other

order/987

pay/xyz  confirm/xyz
Resources look after themselves

- Quote
  - created

- Order
  - created
  - awaiting payment
  - paid
  - cancelled

- Payment
  - created
  - paid
  - settled
  - cancelled
[ServiceContract]
public class Quote
{
    private readonly UriFactory uriFactory;
    private readonly IQuotationEngine quoteEngine;

    public Quote(UriFactory uriFactory, IQuotationEngine quoteEngine)
    {
        this.uriFactory = uriFactory;
        this.quoteEngine = quoteEngine;
    }

    [WebGet(UriTemplate = "/{id}" ReSharper: Formatted)]
    public Shop Get(string id, HttpRequestMessage request, HttpResponseMessage response)
    {
        //Get quotation from quotation engine
        //Add HTTP headers to response
        //Return entity body
    }
}
[Test]
public void ShouldReturn404NotFoundWhenGettingQuoteThatDoesNotExist()
{
    var id = Guid.Empty;

    var quoteEngine = MockRepository.GenerateStub<IQuotationEngine>();
    quoteEngine.Stub(e => e.GetQuote(id))
        .Throw(new KeyNotFoundException());

    var response = new HttpResponseMessage();

    var quoteResource = new Quote(DefaultUriFactory.Instance, quoteEngine);
    quoteResource.Get(
        id.ToString("N"),
        new HttpRequestMessage(),
        response);

    Assert.AreEqual(HttpStatusCode.NotFound, response.StatusCode);
}
public class Quote
{
    private readonly IQuotationEngine quoteEngine;

    ...

    [WebGet]
    public Shop Get(string id, HttpRequestMessage request, HttpResponseMessage response)
    {
        Quotation quote;
        try
        {
            quote = quoteEngine.GetQuote(new Guid(id));
        }
        catch (KeyNotFoundException)
        {
            return null;
        }

        ...
    }
}
public void ResponseShouldExpire7DaysFromDateTimeQuoteWasCreated()
{
    var id = Guid.Empty;

    DateTimeOffset createdDateTime = DateTime.Now;
    var expiryDateTime = createdDateTime.AddDays(7.00).UtcDateTime;

    var quoteEngine = MockRepository.GenerateStub<IQuotationEngine>();
    quoteEngine.Stub(e => e.GetQuote(id)).Return(
        new Quotation(id, createdDateTime, new LineItem[]{})�);

    var response = new HttpResponseMessage();

    var quote = new Quote(DefaultUriFactory.Instance, quoteEngine);
    quote.Get(
        id.ToString("N"),
        new HttpRequestMessage{Uri = new Uri("http://localhost/quote/"),
            response});

    Assert.AreEqual(expiryDateTime, response.Headers.Expires);
}
public class Quote
{

    [WebGet]
    public Shop Get(string id, HttpResponseMessage request, HttpResponseMessage response)
    {
        //Retrieve quote
        ...

        response.StatusCode = HttpStatusCode.OK;
        response.Headers.CacheControl = new CacheControl {Public = true};
        response.Headers.Expires = quote.CreatedDateTime
            .AddDays(7.0).UtcDateTime;

        ...
    }
}
[ServiceContract]
[UriTemplate("quote", "{id}")]  
public class Quote 
{  
    private readonly UriFactory uriFactory;  
    private readonly IQuotationEngine quoteEngine;  

    public Quote(UriFactory uriFactory, IQuotationEngine quoteEngine)  
    {  
        this.uriFactory = uriFactory;  
        this.quoteEngine = quoteEngine;  
    }  

    [WebGet]  
    public Shop Get(string id, HttpRequestMessage request, HttpResponseMessage response)  
    {  
        Quotation quote;  
        try  
        {  
            quote = quoteEngine.GetQuote(new Guid(id));  
        }  
        catch (KeyNotFoundException)  
        {  
            return null;  
        }  

        response.StatusCode = HttpStatusCode.OK;  
        response.Headers.CacheControl = new CacheControl {Public = true};  
        response.Headers.Expires = quote.CreatedDateTime.AddDays(7.0).UtcDateTime;  

        var baseUri = uriFactory.CreateBaseUri<Quote>(request.Uri);  

        return new ShopBuilder(baseUri, quote.LineItems.Select(  
            li => new LineItemToItem(li).Adapt()))  
            .AddLink(new Link(uriFactory.CreateRelativeUri<Quote>(quote.Id),  
                              RestbucksMediaType.Value, LinkRelations.Self))  
                .AddLink(new Link(uriFactory.CreateRelativeUri<OrderForm>(quote.Id),  
                                     RestbucksMediaType.Value, LinkRelations.OrderForm)).Build();  
    }  
}
Resources := domain

shop

request for quote

quotes

quote

order form

orders

order

cancellation

payment
<shop xmlns:rb="http://relations.restbucks.com/
    xml:base="http://restbucks.com/
    xmlns="http://schemas.restbucks.com/shop">
  <items>
    <item>
      <description>coffee</description>
      <amount measure="g">125</amount>
      <price currency="GBP">1.25</price>
    </item>
  </items>
  <link rel="self"
    type="application/vnd.restbucks+xml"
    href="quote/68cff6e75a09474fa0098c9393aa6d4e" />
  <link rel="rb:order-form"
    type="application/vnd.restbucks+xml"
    href="order-form/68cff6e75a09474fa0098c9393aa6d4e" />
</shop>
Order form

```xml
<shop xml:base="http://restbucks.com/">
  xmlns="http://schemas.restbucks.com/shop">
  <model id="order" xmlns="http://www.w3.org/2002/xforms">
    <instance>
      <shop xml:base="http://restbucks.com/">
        xmlns="http://schemas.restbucks.com/shop">
        <items>
          <item>
            <description>coffee</description>
            <amount measure="g">125</amount>
            <price currency="GBP">1.25</price>
          </item>
        </items>
      </shop>
    </instance>
    <submission resource="/orders/?c=12345&amp;s=325"
      method="post"
      mediatype="application/vnd.restbucks+xml" />
  </model>
</shop>
```
Resources adapt the domain for hypermedia clients
Problem: URI redundancy

Routes

RouteTable.Routes.AddServiceRoute<Quote>("quote", configuration);
RouteTable.Routes.AddServiceRoute<OrderForm>("order-form", configuration);

Methods

[WebGet(UriTemplate = "{id}"),]
public Shop Get(string id, HttpRequestMessage request,
    HttpResponseMessage response)
{
    ...

Links

return new ShopBuilder(new Uri("http://restbucks.com/"))
    .AddItem(new Item("coffee beans", new Amount("g", 250)))
    .AddLink(new Link(
        new Uri("quote/" + quoteId, UriKind.Relative),
        RestbucksMediaType.Value, LinkRelations.Self))
    .AddLink(new Link(
        new Uri("order-form/" + quoteId, UriKind.Relative),
        RestbucksMediaType.Value, LinkRelations.OrderForm))
    .Build();
Solution: UriFactory

```csharp
[UriTemplate("quote", "{id}" )]
public class Quote
{
}

[Test]
public void UriFactoryExample()
{
    var uriFactory = new UriFactory();
    uriFactory.Register<Quote>();

    Assert.AreEqual(  
        new Uri("quote/1234", UriKind.Relative),  
        uriFactory.CreateRelativeUri<Quote>(1234));

    Assert.AreEqual(  
        new Uri("http://restbucks.com/quote/1234"),  
        uriFactory.CreateAbsoluteUri<Quote>(  
            new Uri("http://restbucks.com"), 1234));

    Assert.AreEqual(  
        new Uri("http://restbucks.com/") ,  
        uriFactory.CreateBase Uri<Quote>(  
            new Uri("http://restbucks.com/quote/1234")));
}
```
public void RegisterResourcesFor(Assembly assembly)
{
    var register = typeof(UriFactory).GetMethod("Register",
        BindingFlags.Instance | BindingFlags.NonPublic);

    var types = from t in assembly.GetTypes()
        where t.GetCustomAttributes(typeof(UriTemplateAttribute),
            false).Length > 0
            select t;

    types.ToList().ForEach(t =>
    {
        var genericMethod = register.MakeGenericMethod(new[] {t});
        genericMethod.Invoke(this, null);
    });
}
[ServiceContract]
[UriTemplate("quote", "{id}")]}
public class Quote { ...

[WebGet]
public Shop Get(string id, HttpRequestMessage request, HttpResponseMessage response) { ...

    var baseUri = uriFactory.CreateBaseUri<Quote>(request.Uri);

    return new ShopBuilder(baseUri, quote.LineItems.Select(
        li => new LineItemToItem(li).Adapt()))
        .AddLink(new Link(uriFactory.CreateRelativeUri<Quote>(quote.Id),
            RestbucksMediaType.Value, LinkRelations.Self))
        .AddLink(new Link(uriFactory.CreateRelativeUri<OrderForm>(quote.Id),
            RestbucksMediaType.Value, LinkRelations.OrderForm)).Build();
}
Client-side Development
The application is in the eye of the client

Clients apply resources to achieve an application goal
Client-side state machine: buckets of rules

- Started
  - Quote
    - Requested
    - Goods
      - Ordered
        - Order
          - Confirmed
            - Cancelled
            - Paid
            - Error
Rules trigger events; hypermedia controls handle events

Client

Receive response

Apply rules

Activate control

```xml
<shop xml:base="http://restbucks.com/
 xmlns="http://schemas.restbucks.com/shop">
 <model id="order" xmlns="http://www.w3.org/2002/xforms">
   <instance>
     <shop xml:base="http://restbucks.com/
 xmlns="http://schemas.restbucks.com/shop">
       <items>
         <item>
           <description>coffee</description>
           <amount measure="g">125</amount>
           <price currency="GBP">1.25</price>
         </item>
       </items>
     </shop>
   </instance>
   <link rel="self" type="application/vnd.restbucks+xml"
 href="quote/68cff6e75a09474fa0098c9393aa6d4e" />
 </model>
</shop>
```
A bucket of rules

public class Started : IState
{
    public IState NextState(IClientCapabilities httpClient)
    {
        var rules = new Rules(
            When
                .IsRequired(response => prevResponse.ContainsForm(Forms.Rfq))
                .Invoke(actions => actions.SubmitForm(Forms.Rfq))
                .Return(new[] {
                    On.Status(HttpStatusCode.Created)
                        .Do((response, vars) => new QuoteRequested(response, vars))
                }),
            When
                .IsRequired(response => prevResponse.ContainsLink(Links.Rfq))
                .Invoke(actions => actions.ClickLink(Links.Rfq))
                .Return(new[] {
                    On.Status(HttpStatusCode.OK)
                        .Do((response, vars) => new Started(response, vars))
                }));

        return rules.Evaluate(previousResponse, stateVariables, httpClient);
    }

    public bool IsTerminalState { get { return false; } } }
Get homepage

Request

GET /shop/ HTTP/1.1
Accept: application/vnd.restbucks+xml
Host: restbucks.com

Response

HTTP/1.1 200 OK
Cache-Control: max-age=86400, public
Content-Length: 285
Content-Type: application/vnd.restbucks+xml
Date: Wed, 11 May 2011 17:27:22 GMT

<shop xmlns:rb="http://relations.restbucks.com/"
     xml:base="http://restbucks.com/
     xmlns="http://schemas.restbucks.com/shop">
  <link rel="rb:rfq prefetch"
         type="application/vnd.restbucks+xml"
         href="request-for-quote/" />
</shop>
Second rule fires

```csharp
public class Started : IState
{
    public IState NextState(IClientCapabilities httpClient)
    {
        var rules = new Rules(
            When
            .IsTrue(response => prevResponse.ContainsForm(Forms.Rfq))
            .Invoke(actions => actions.SubmitForm(Forms.Rfq))
            .Return(new[]{
                On.Status(HttpStatusCode.Created)
                .Do((response, vars) => new QuoteRequested(response, vars))
            })),
            When
            .IsTrue(response => prevResponse.ContainsLink(Links.Rfq))
            .Invoke(actions => actions.ClickLink(Links.Rfq))
            .Return(new[]{
                On.Status(HttpStatusCode.OK)
                .Do((response, vars) => new Started(response, vars))
            }));

        return rules.Evaluate(previousResponse, stateVariables, httpClient);
    }

    public bool IsTerminalState { get { return false; } }
}
```
Get request for quote form

Request
GET /request-for-quote/ HTTP/1.1
Accept: application/vnd.restbucks+xml
Host: restbucks.com

Response
HTTP/1.1 200 OK
Cache-Control: max-age=86400, public
Content-Length: 385
Content-Type: application/vnd.restbucks+xml
Date: Wed, 11 May 2011 22:12:52 GMT

<shop xml:base="http://restbucks.com/"
    xmlns="http://schemas.restbucks.com/shop">
    <model id="request-for-quote"
        schema="http://schemas.restbucks.com/shop"
        xmlns="http://www.w3.org/2002/xforms">
        <instance />
        <submission resource="quotes/"
            method="post"
            mediatype="application/vnd.restbucks+xml" />
    </model>
</shop>
public class Started : IState
{
    public IState NextState(IClientCapabilities httpClient)
    {
        var rules = new Rules(
            When
                .IsTrue(response => prevResponse.ContainsForm(Forms.Rfq))
                .Invoke(actions => actions.SubmitForm(Forms.Rfq))
                .Return(new[]{
                    On.Status(HttpStatusCode.Created)
                    .Do((response, vars) => new QuoteRequested(response, vars))}),
            When
                .IsTrue(response => prevResponse.ContainsLink(Links.Rfq))
                .Invoke(actions => actions.ClickLink(Links.Rfq))
                .Return(new[]{
                    On.Status(HttpStatusCode.OK)
                    .Do((response, vars) => new Started(response, vars))}));

        return rules.Evaluate(previousResponse, stateVariables, httpClient);
    }

    public bool IsTerminalState { get { return false; } }
Alternate homepage (inlined form)

Request
GET /shop/ HTTP/1.1
Accept: application/vnd.restbucks+xml
Host: restbucks.com

Response
HTTP/1.1 200 OK
Cache-Control: max-age=86400, public
Content-Length: 470
Content-Type: application/vnd.restbucks+xml
Date: Wed, 11 May 2011 17:25:31 GMT

<shop xml:base="http://restbucks.com/
   xmlns="http://schemas.restbucks.com/shop">
   <link rel="search" type="application/opensearchdescription+xml"
       href="search/" />
   <model id="request-for-quote"
       schema="http://schemas.restbucks.com/shop"
       xmlns="http://www.w3.org/2002/xforms">
       <instance />
       <submission resource="quotes/" method="post"
           mediatype="application/vnd.restbucks+xml" />
   </model>
</shop>
public class Started : IState
{
    public IState NextState(IClientCapabilities httpClient)
    {
        var rules = new Rules(
            When
                .IsTrue(response => prevResponse.ContainsForm(Forms.Rfq))
                .Invoke(actions => actions.SubmitForm(Forms.Rfq))
                .Return(new[]
                {
                    On.Status(HttpStatusCode.Created)
                    .Do((response, vars) => new QuoteRequested(response, vars))
                })),
            When
                .IsTrue(response => prevResponse.ContainsLink(Links.Rfq))
                .Invoke(actions => actions.ClickLink(Links.Rfq))
                .Return(new[]
                {
                    On.Status(HttpStatusCode.OK)
                    .Do((response, vars) => new Started(response, vars))
                }));

        return rules.Evaluate(previousResponse, stateVariables, httpClient);
    }

    public bool IsTerminalState { get { return false; } }
}
Running the client

```csharp
var variables = ...

var state = new Uninitialized(variables);
var nextState = state.NextState(ClientCapabilities.Instance);

while (!nextState.IsTerminalState)
{
    nextState = nextState.NextState(ClientCapabilities.Instance);
}
```
Result: a hypermedia system

- **client**
- **server**
- **logical resources** (uri + uniform interface)
- **physical resources** (domain)
- **representation** e.g. XML, JSON document
- **request**
- **response**
Questions?

http://ianSrobinson.com
@ianSrobinson