Resource-Oriented Architecture Patterns

Brian Sletten
Bosatsu Consulting, Inc.

SYNTHESIS LECTURES ON SEMANTIC WEB #1

MORGAN & CLAYPOOL PUBLISHERS

ABSTRACT

SAMPLE ABSTRACT

In quantum computing, where algorithms exist that can solve computational problems more efficiently than any known classical algorithms, the elimination of errors that result from external disturbances or from imperfect gates has become the “holy grail,” and a worldwide quest for a large scale fault-tolerant and computationally superior quantum computer is currently taking place. Optimists rely on the premise that, under a certain threshold of errors, an arbitrary long fault-tolerant quantum computation can be achieved with only moderate (i.e., at most polynomial) overhead in computational cost. Pessimists, on the other hand, object that there are in principle (as opposed to merely technological) reasons why such machines are still inexistent, and that no matter what gadgets are used, large scale quantum computers will never be computationally superior to classical ones. Lacking a complete empirical characterization of quantum noise, the debate on the physical possibility of such machines invites philosophical scrutiny. Making this debate more precise by suggesting a novel statistical mechanical perspective thereof is the goal of this project.

KEYWORDS

SAMPLE KEYWORDS

computational complexity, decoherence, error-correction, fault-tolerance, Landauer’s Principle, Maxwell’s Demon, quantum computing, statistical mechanics, thermodynamics

Contents

CHAPTER 1

Informational Patterns

1.1 INTRODUCTION

The collection of patterns identified here are loosely categorized as those that serve information-sharing and manipulation purposes. The Information Resource pattern itself really needs no motivation. It has an established reference implementation called The World Wide Web[1]. The logically-named, interlinked HTML documents became *The Way* of sharing information with collaborators, partners, clients and the general public. As we grew discontent with static documents, we found ways to make them dynamic and to reflect the contents of our data stores in user-friendly and navigable ways. This in term fueled a hunger for zero installation dynamic applications through the *Universal Client*, a.k.a. the browser.

This migration from static documents to dynamic applications certainly involved new technologies (e.g. AJAX, JavaScript libraries, etc.), but it was enabled based on the underlying pieces of logically-named *Uniform Resource Locators* (URLs), the *Hypertext Transfer Protocol* (HTTP) and *Hypertext Markup Language* (HTML).

These technologies combined quite seamlessly to bring us a platform usable by scientists, knowledge workers, grandparents and children. It was not an accident that this happened. Tim Berners-Lee, Roy Fielding, Dan Connolly, Dave Raggett and others from the *Internet Engineering Task Force* (IETF) and *World Wide Web Consortium* (W3C) toiled to create a platform unencumbered by patents but displaying the architectural properties of loose-coupling, scalability, flexibility and extensibility.

The part that still requires a surprising amount of motivation is to think about these technologies within an organization's firewalls as well. We forget that Tim's original proposal was written in the context of solving the information sharing needs of a single (albeit complex) organization, the European Organization for Nuclear Research (CERN).

In his storied proposal[2], he said:

> If a CERN experiment were a static once-only development, all the information could be written in a big book.Berners-Lee (1989)

Clearly this was not the case, however. In trying to do fundamental science, new participants, new strategies, new results and new experiments needed to be accepted constantly. Given that CERN brought researchers from around the world and interacted with

[1] Or, these days, "The Web".
[2] Deemed "vague but exciting" by his supervisor

other academic, commercial and governmental institutions, no social or organizational policies could force standardization amongst all the participants. A technological solution that embraced change was proposed. After sitting idle

1.2 INFORMATION RESOURCE

1.2.1 INTENT

The Information Resource pattern is the basic building block of the Web. It provides an addressable, resolvable relatively course-grained piece of information. It should generally have a stable identifier associated with it and optionally support content negotiation, updates and deletions.

1.2.2 MOTIVATION

1.2.3 CONSEQUENCES

Document Data Service

1.3 COLLECTION RESOURCE

1.3.1 INTENT

The Collection Resource is a form of the Information Resource pattern that allows arbitrary partitioning of the information space. To keep client resources from having to know too much about the server layout, these resources should provide the mechanism for pagination across the collection.

1.3.2 MOTIVATION

1.3.3 CONSEQUENCES

Partition Pagination

1.4 NON-INFORMATION RESOURCE

1.4.1 INTENT

The Non-Information Resource is a basic building block of the Semantic Web. It provides an addressable, resolvable identifier representing a resource that is not otherwise directly accessible. It can be used for people, organizations, concepts or any term or relationship from a domain of interest.

1.4.2 MOTIVATION

1.4.3 CONSEQUENCES

Concept

1.5 NAMED QUERY

1.5.1 INTENT

The Named Query pattern generalizes the idea of providing an identifier for a reusable query. The naming process might be explicit or implicit based on ad hoc, but identifiable, query structures. By giving the query identity, it becomes reusable, shareable, cacheable and perhaps the source of a new data source.

1.5.2 MOTIVATION

1.5.3 CONSEQUENCES

SPARQL Endpoint

CHAPTER 2

Applicative Patterns

2.1 INTRODUCTION

2.2 GUARD

2.2.1 INTENT

The Guard pattern serves as a protection mechanism for another resource, service or data source. Client requests are funneled through a resource to force a specific authentication and authorization strategy.

2.2.2 MOTIVATION

2.2.3 CONSEQUENCES

NetKernel? OAuth 2?

2.3 THROTTLE

2.3.1 INTENT

The Throttle pattern is a variation of the Guard pattern where the protection scheme involves scale rather than security. It is designed specifically to provide a resource to shape the request traffic into a predictable and sustainable load.

2.3.2 MOTIVATION

2.3.3 CONSEQUENCES

NetKernel Throttle in AWS

2.4 OVERLAY

2.4.1 INTENT

2.4.2 MOTIVATION

2.4.3 CONSEQUENCES

2.5 TRANSFORMATION

2.5.1 INTENT

The Transformation pattern is a generalization over the approach of producing new resources to extract content from or transform the shape of content from another service or resource. As with the Named Query pattern, doing so can induce reusability and cacheability from sources that otherwise do not support such properties. It can also be used to add content negotiation to sources that do not support it.

2.5.2 MOTIVATION

2.5.3 CONSEQUENCES

RDFa Distiller

CHAPTER 3

Procedural Patterns

3.1 INTRODUCTION

3.2 GATEWAY

3.2.1 INTENT

The Gateway pattern is notionally a combination of the Information Resource, Named Query and Tranformation patterns. The distinction is that it is generally intended for a specific client use case. Rather than a server resource being established for general use, the Gateway pattern establishes a proxy for a client interested in an orchestration, data aggregation, content extraction or other processing of one or more backend sources.

3.2.2 MOTIVATION

3.2.3 CONSEQUENCES

ql.io

3.3 CURATED URI

3.3.1 INTENT

The Curated URI pattern establishes a stewardship commitment to the clients of a named resource. Rather than relying on the good graces of a resource provider to keep an identifier stable over time, a new identifier is chosen to be redirected toward another endpoint. The target resource can be moved as long as the redirection is maintained without having a negative impact on clients.

3.3.2 MOTIVATION

3.3.3 CONSEQUENCES

PURL

3.4 HEATMAP

3.4.1 INTENT

The Heatmap pattern grows out of the dual nature of an addressable resource identifier. As a name, it affords disambiguation in a global information space at a very fine granularity. As a handle, it represents the interface to the resource itself. Metadata captured about how this resource is used, accessed, failure rates, etc. represent crucial business intelligence. This information can be used to identify business opportunities, operational planning, dynamic routing and more.

3.4.2 MOTIVATION

3.4.3 CONSEQUENCES

3.5 WORKFLOW

3.5.1 INTENT

The Workflow pattern encodes a series of steps into a resource abstraction where the client learns what options are available through the resource representation. The server is in charge of enabling and disabling state transitions based on the context of the requests, the client choices and other inputs.

3.5.2 MOTIVATION

3.5.3 CONSEQUENCES

Bibliography

T. Berners-Lee, "Information Management : A Proposal," http://www.w3.org/History/1989/proposal.html.

Author's Biography

YOUR NAME

Your Name Your short bio goes here...