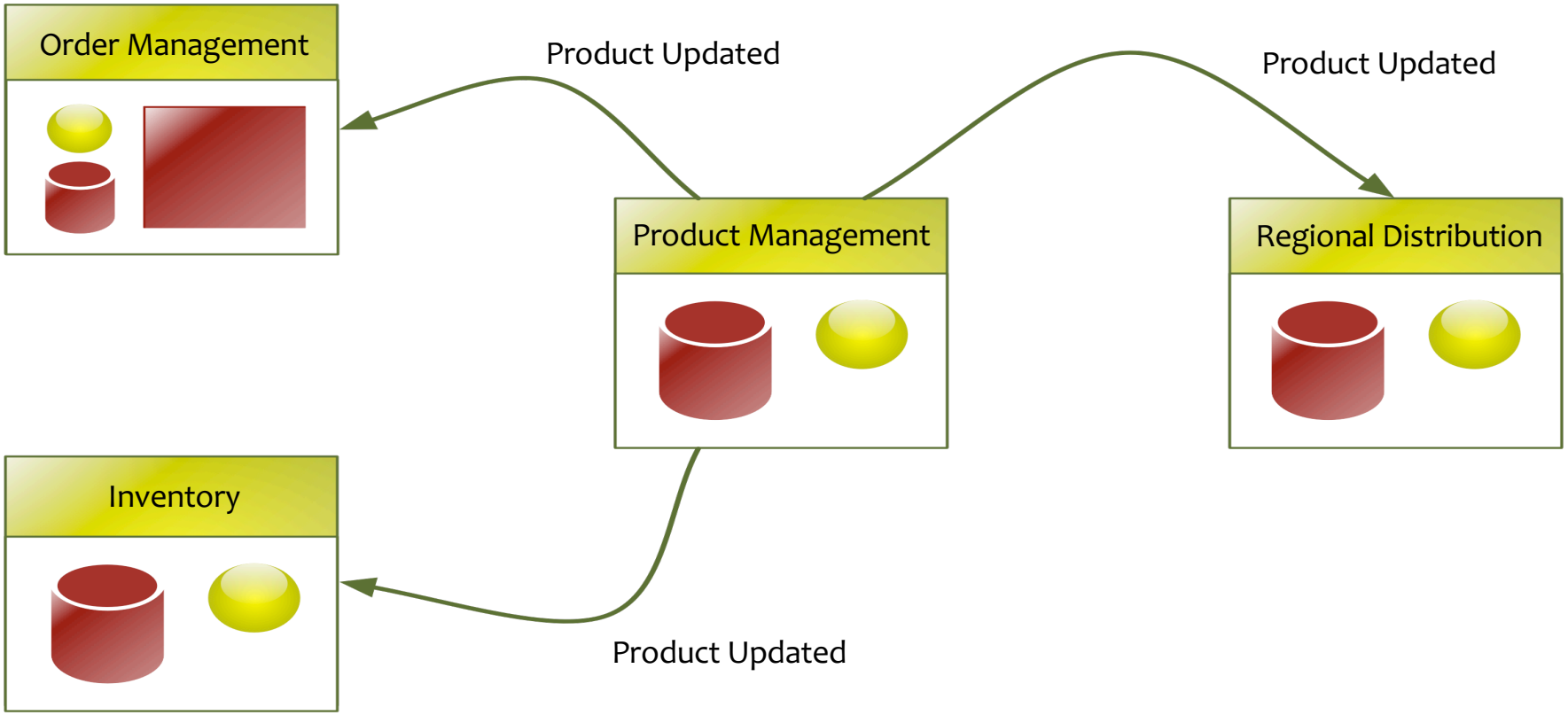

Microservices: Event-Driven Architecture

Services


Order Management
Product Management
Regional Distribution
Inventory

Events


Order Management
Product Updated
Product Management
Product Updated
Regional Distribution
Inventory
Product Updated

Implementation Options

Point-to-point

- Publisher maintains subscriber list
- Queues to reduce temporal coupling

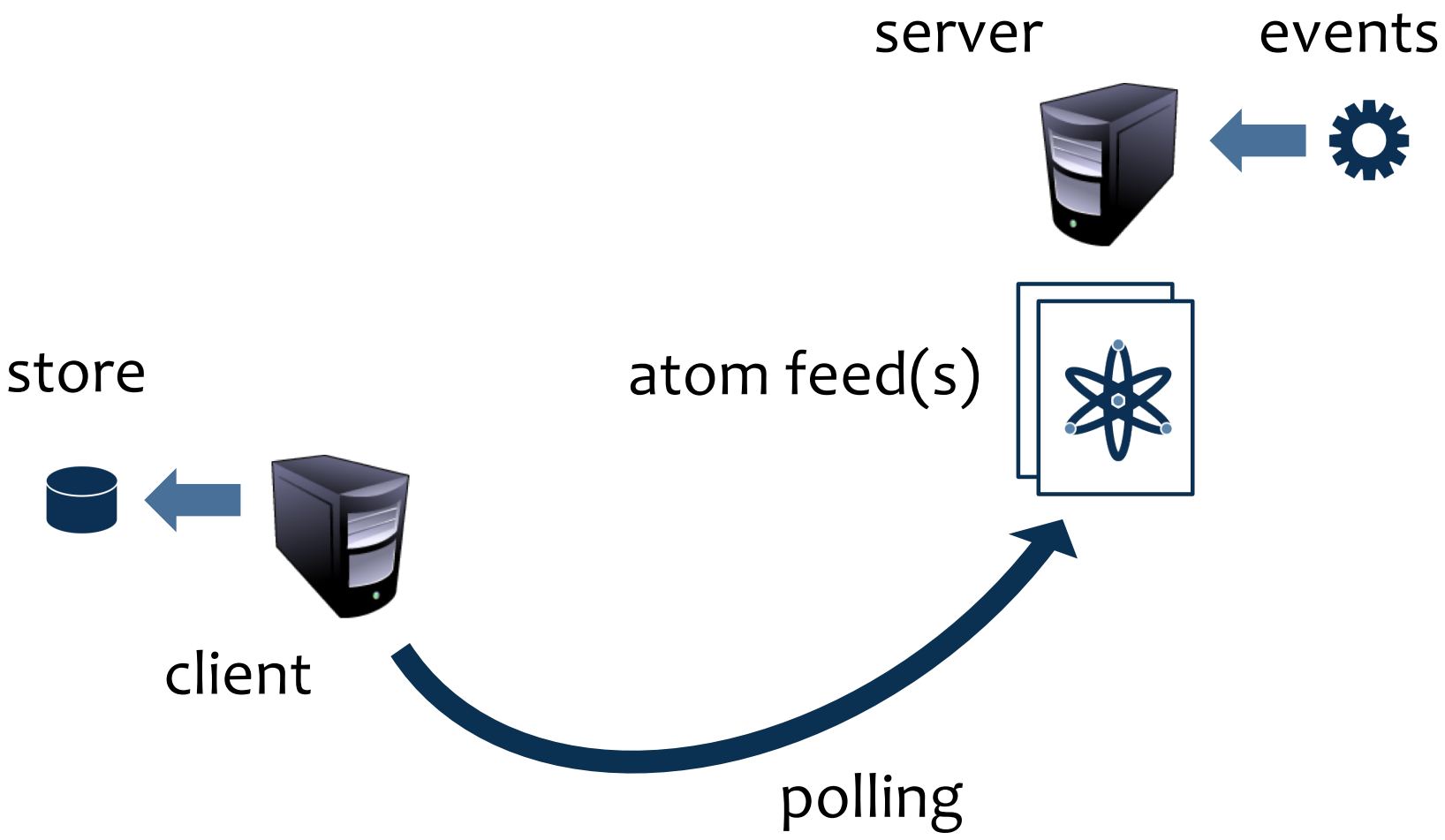
Bus

- Subscriptions and guaranteed delivery delegated to middleware
- Reduced location and temporal coupling

Consumers pull events

- Consumers poll publishers
- Guaranteed delivery delegated to consumers
- No list of subscribers to maintain

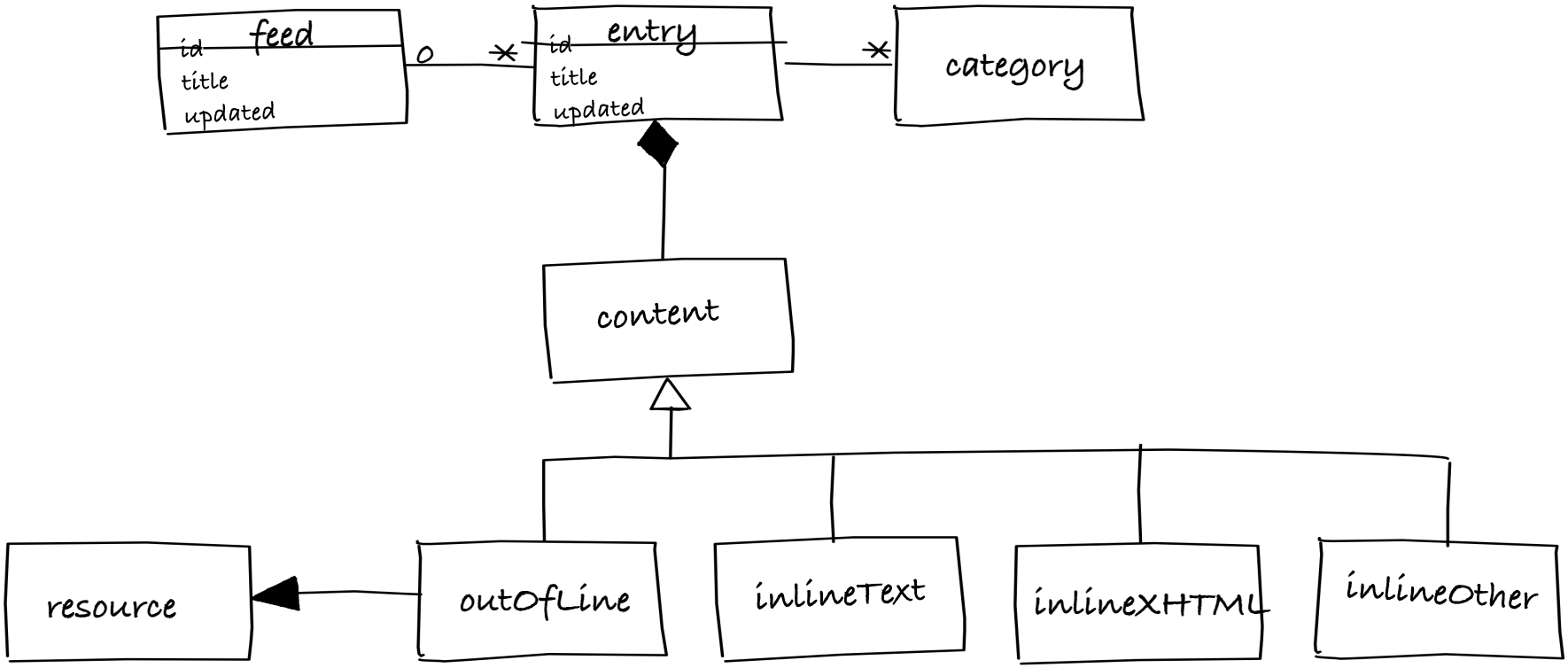
Event Feeds



Publish a Feed of Events

- Atom Syndication Format
  - Representation format for list of things
- Atom Publication Protocol
  - Application protocol for
    - editing Web resources
    - using HTTP transfer of
    - Atom-formatted representations

Atom model



Based on diagrams by Stefan Tilkov

Atom Feed

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Restbucks products and promotions</title>
  <id>urn:uuid:4aaf346c-1c9c-42cb-a006-5ff35d83c707</id>
  <updated>2010-11-29T04:38:09Z</updated>
  <author>
    <name>Product Catalog</name>
  </author>
  <generator>Product Catalog</generator>
  <link rel="self" href="http://localhost./updates" />
  <link rel="via" href="http://localhost./updates/20,29" />
  <link rel="prev-archive" href="http://localhost./updates/10,19" />
  <entry>
    <id>tag:restbucks.com,2010-11-29:23</id>
    <title type="text">event: 23</title>
    <updated>2010-11-29T04:38:07Z</updated>
    <category scheme="http://categories.restbucks.com/status" term="new"/>
    <link rel="self" href="http://localhost./updates/23" />
    <link rel="related" href="http://restbucks.com/products/9876" />
    <content type="application/vnd.restbucks+xml">
      <product xmlns="http://schemas.restbucks.com/product">
        <name>Costa Rica Tarrazu</name>
        <price>8</price>
        <size>1kg</size>
      </product>
    </content>
  </entry>
  ...
</feed>
```

Entries Represent Events

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Restbucks products and promotions</title>
  <id>urn:uuid:4aaf346c-1c9c-42cb-a006-5ff35d83c707</id>
  <updated>2010-11-29T04:38:09Z</updated>
  <author>
    <name>Product Catalog</name>
  </author>
  <generator>Product Catalog</generator>
  <link rel="self" href="http://localhost./updates" />
  <link rel="via" href="http://localhost./updates/20,29" />
  <link rel="prev-archive" href="http://localhost./updates/10,19" />
  <entry>
    <id>tag:restbucks.com,2010-11-29:23</id>
    <title type="text">event: 23</title>
    <updated>2010-11-29T04:38:07Z</updated>
    <category scheme="http://categories.restbucks.com/status" term="new"/>
    <link rel="self" href="http://localhost./updates/23" />
    <link rel="related" href="http://restbucks.com/products/24" />
    <content type="application/vnd.restbucks+xml">
      <product xmlns="http://schemas.restbucks.com/product">
        <name>Costa Rica Tarrazu</name>
        <price>8</price>
        <size>1kg</size>
      </product>
    </content>
  </entry>
  ...
</feed>
```

Entry = Event Metadata

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Restbucks products and promotions</title>
  <id>urn:uuid:4aaf346c-1c9c-42cb-a006-5ff35d83c707</id>
  <updated>2010-11-29T04:38:09Z</updated>
  <author>
    <name>Product Catalog</name>
  </author>
  <generator>Product Catalog</generator>
  <link rel="self" href="http://localhost./updates" />
  <link rel="via" href="http://localhost./updates/20,29" />
  <link rel="prev-archive" href="http://localhost./updates/10,19" />
  <entry>
    <id>tag:restbucks.com,2010-11-29:23</id>
    <title type="text">event: 23</title>
    <updated>2010-11-29T04:38:07Z</updated>
    <category scheme="http://categories.restbucks.com/status" term="new"/>
    <link rel="self" href="http://localhost./updates/23" />
    <link rel="related" href="http://restbucks.com/products/24" />
    <content type="application/vnd.restbucks+xml">
      <product xmlns="http://schemas.restbucks.com/product">
        <name>Costa Rica Tarrazu</name>
        <price>8</price>
        <size>1kg</size>
      </product>
    </content>
  </entry>
  ...
</feed>
```

Entry = Event Metadata + Snapshot of Entity State

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Restbucks products and promotions</title>
  <id>urn:uuid:4aaf346c-1c9c-42cb-a006-5ff35d83c707</id>
  <updated>2010-11-29T04:38:09Z</updated>
  <author>
    <name>Product Catalog</name>
  </author>
  <generator>Product Catalog</generator>
  <link rel="self" href="http://localhost./updates" />
  <link rel="via" href="http://localhost./updates/20,29" />
  <link rel="prev-archive" href="http://localhost./updates/10,19" />
  <entry>
    <id>tag:restbucks.com,2010-11-29:23</id>
    <title type="text">event: 23</title>
    <updated>2010-11-29T04:38:07Z</updated>
    <category scheme="http://categories.restbucks.com/status" term="new"/>
    <link rel="self" href="http://localhost./updates/23" />
    <link rel="related" href="http://restbucks.com/products/24" />
    <content type="application/vnd.restbucks+xml">
      <product xmlns="http://schemas.restbucks.com/product">
        <name>Costa Rica Tarrazu</name>
        <price>8</price>
        <size>1kg</size>
      </product>
    </content>
  </entry>
  ...
</feed>
```

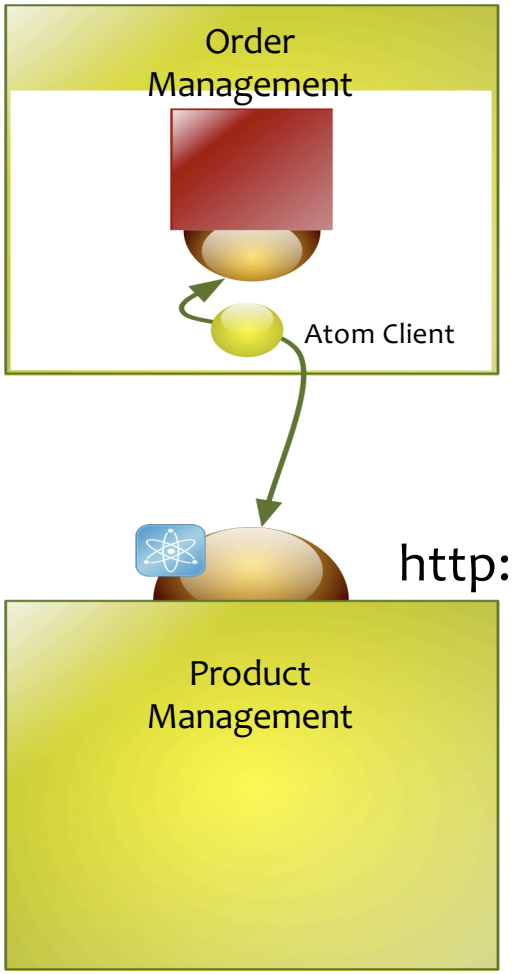
Head Feed Links

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Restbucks products and promotions</title>
  <id>urn:uuid:4aaf346c-1c9c-42cb-a006-5ff35d83c707</id>
  <updated>2010-11-29T04:38:09Z</updated>
  <author>
    <name>Product Catalog</name>
  </author>
  <generator>Product Catalog</generator>
  <link rel="self" href="http://localhost./updates" />
  <link rel="via" href="http://localhost./updates/20,29" />
  <link rel="prev-archive" href="http://localhost./updates/10,19" />
  <entry>
    <id>tag:restbucks.com,2010-11-29:23</id>
    <title type="text">event: 23</title>
    <updated>2010-11-29T04:38:07Z</updated>
    <category scheme="http://categories.restbucks.com/status" term="new"/>
    <link rel="self" href="http://localhost./updates/23" />
    <link rel="related" href="http://restbucks.com/products/24" />
    <content type="application/vnd.restbucks+xml">
      <product xmlns="http://schemas.restbucks.com/product">
        <name>Costa Rica Tarrazu</name>
        <price>8</price>
        <size>1kg</size>
      </product>
    </content>
  </entry>
  ...
</feed>
```

Archive Feed

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <title type="text">Restbucks products and promotions</title>
  <id>urn:uuid:88d7256c-593d-4154-a89f-1537c957119d</id>
  <updated>2010-11-29T04:38:09Z</updated>
  <author>
    <name>Product Catalog</name>
  </author>
  <generator>Product Catalog</generator>
  <link rel="self" href="http://localhost./updates/10,19" />
  <link rel="prev-archive" href="http://localhost./updates/0,9" />
  <link rel="next-archive" href="http://localhost./updates/20,29" />
  <archive xmlns="http://purl.org/syndication/history/1.0"></archive>
  <entry>
    <id>tag:restbucks.com,2010-11-29:19</id>
    <title type="text">event: 15</title>
    <updated>2010-11-29T04:37:59Z</updated>
    <category scheme="http://categories.restbucks.com/status" term="updated"/>
    <link rel="self" href="http://localhost./updates/15" />
    <link rel="related" href="http://restbucks.com/products/1234" />
    <content type="application/vnd.restbucks+xml">
      <product xmlns="http://schemas.restbucks.com/product">
        <name>Elephant Beans</name>
        <price>10</price>
        <size>1kg</size>
      </product>
    </content>
  </entry>
  ...
</feed>
```

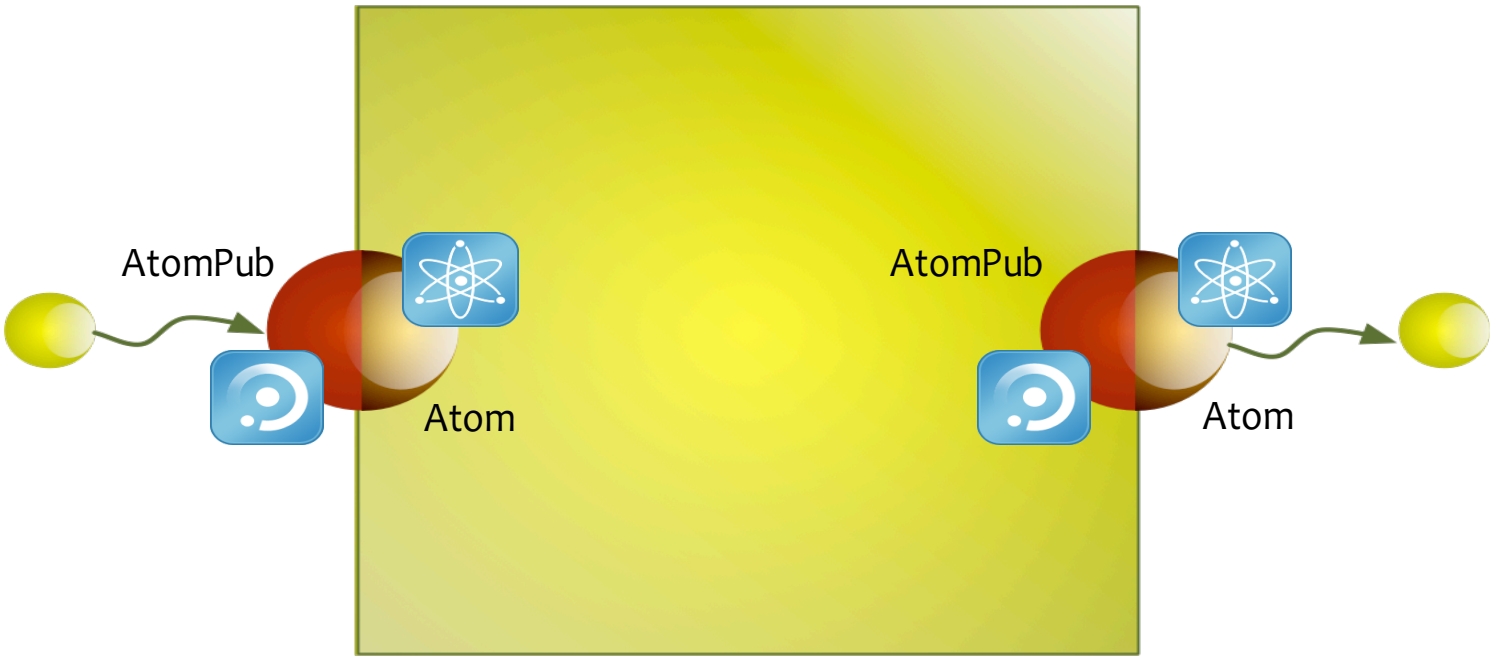
Polling an Atom feed



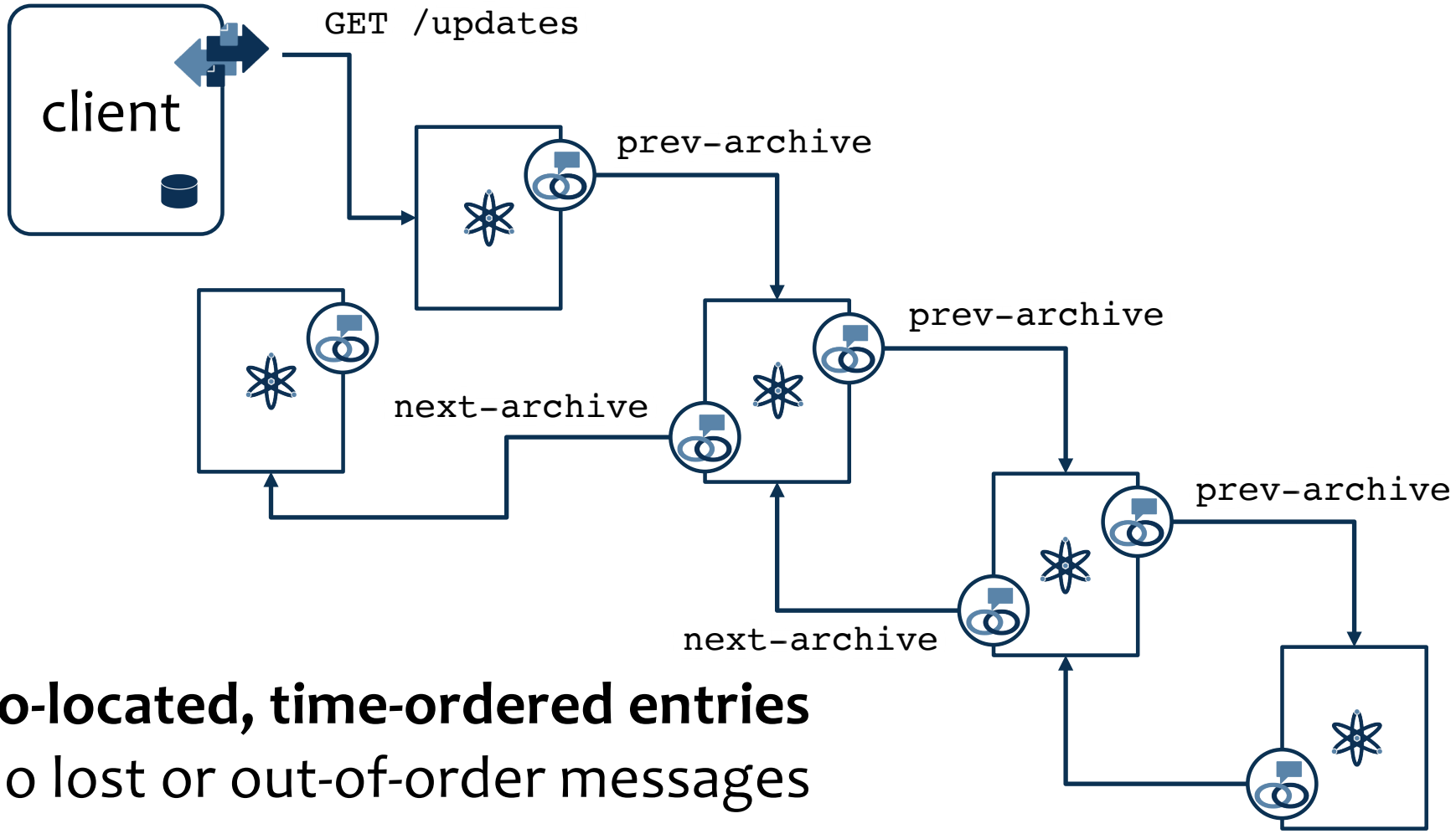
http://restbucks.com/products/notifications

Atom icons from http://sniker.blogage.de/

An Atom- and AtomPub-enabled service



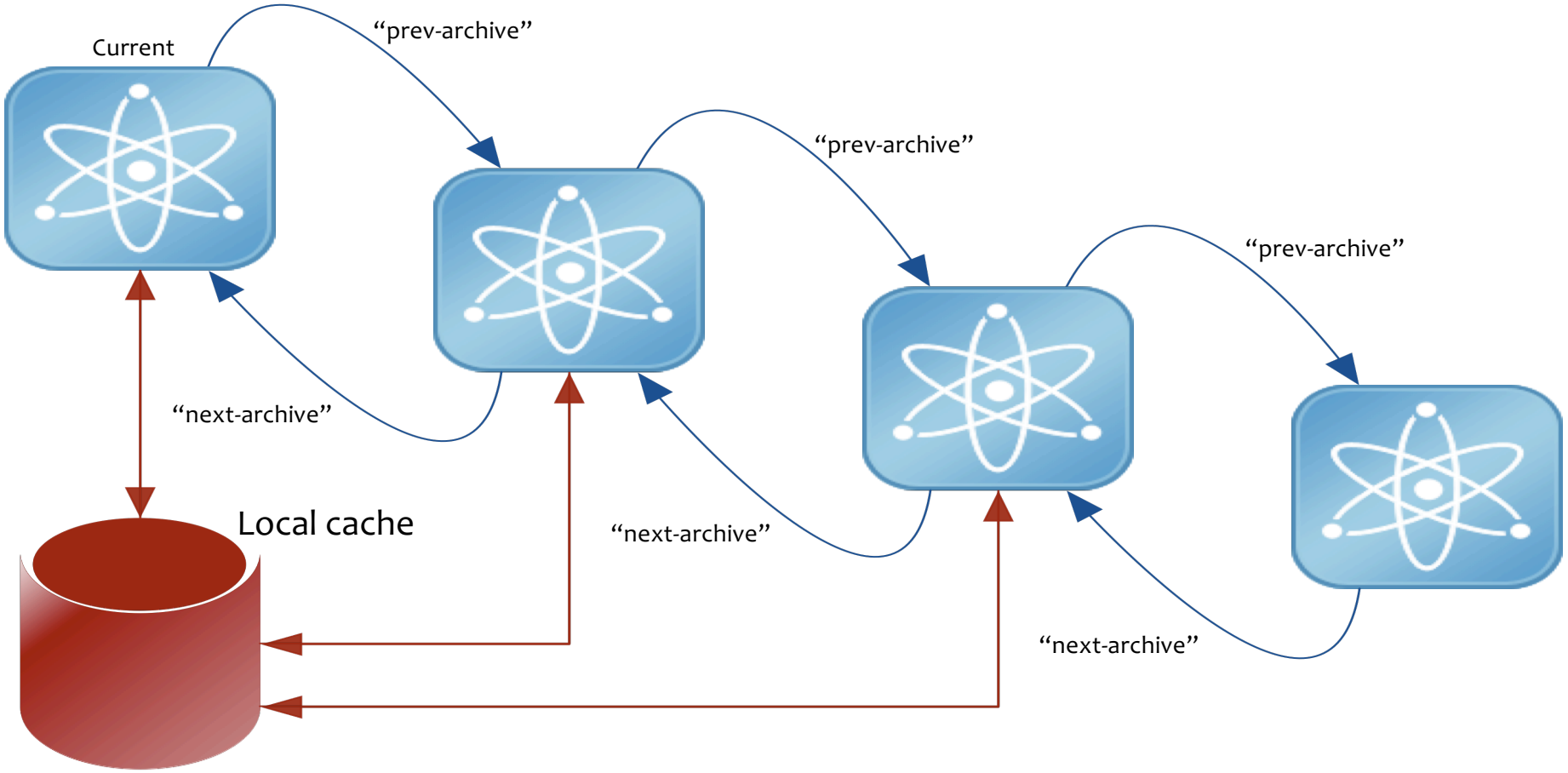
Feed Paging



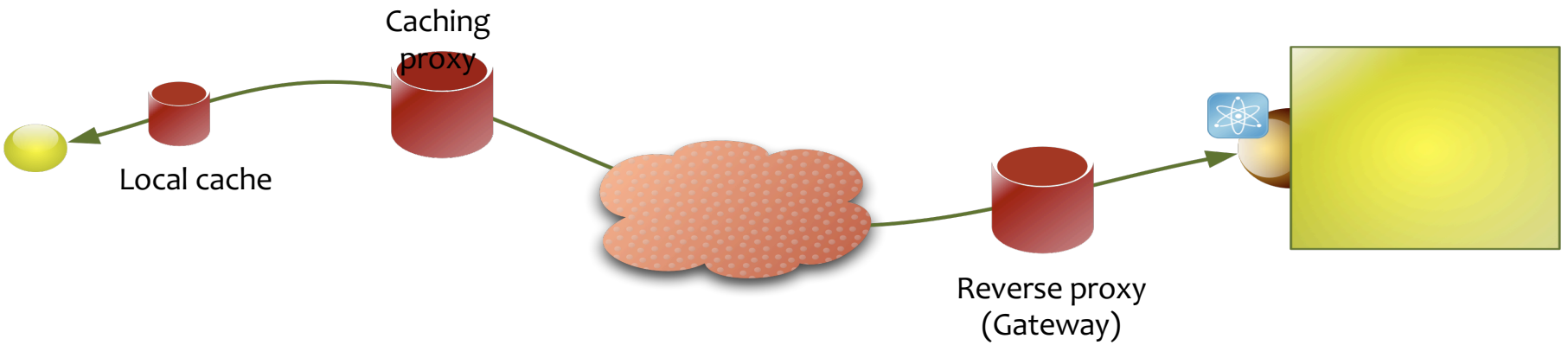
Co-located, time-ordered entries

No lost or out-of-order messages

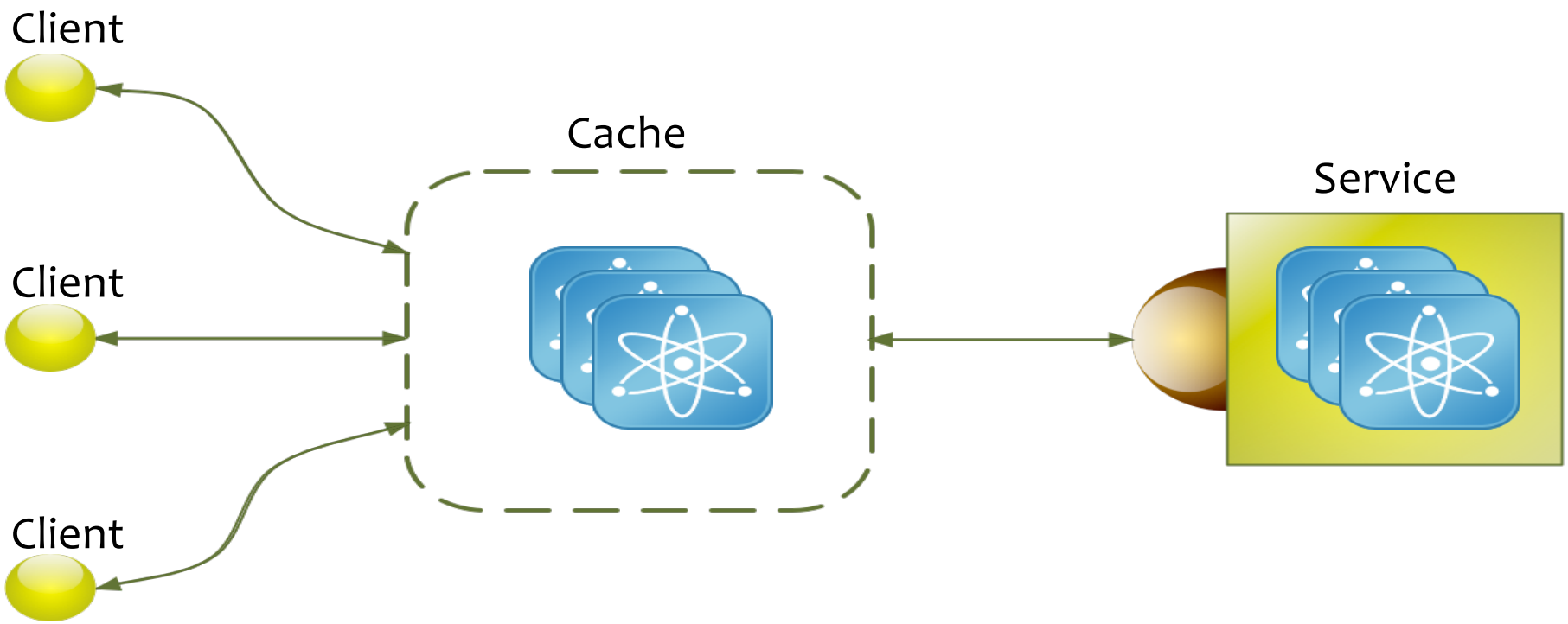
Caching archived feeds



Caches



Caching the bus



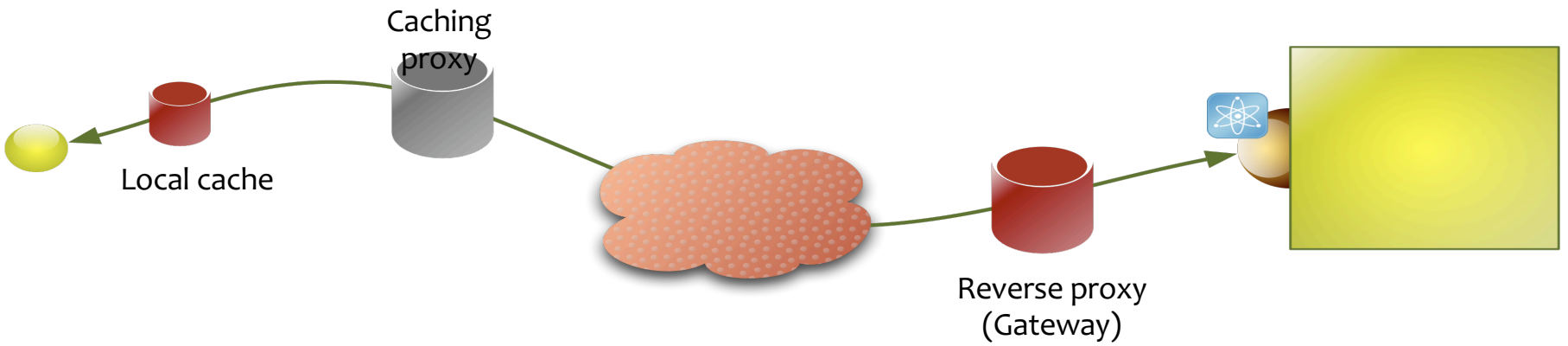
Protecting feeds

Request

```
GET /products/notifications HTTP/1.1
Host: restbucks.com
Authorization: Basic aWFuc3JvYmluc29uOlBhdHQzcm41==
```

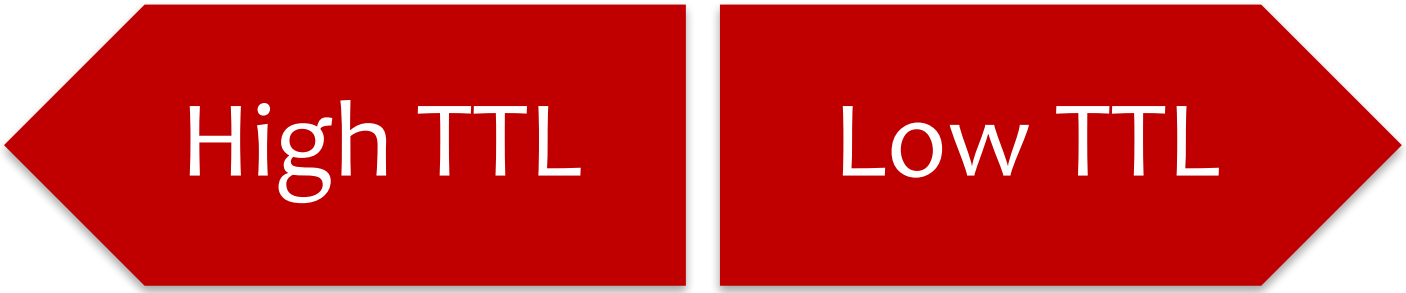
Response

```
...
Cache-control: public, no-cache
```



Caching dilemma

Efficient use of network resources

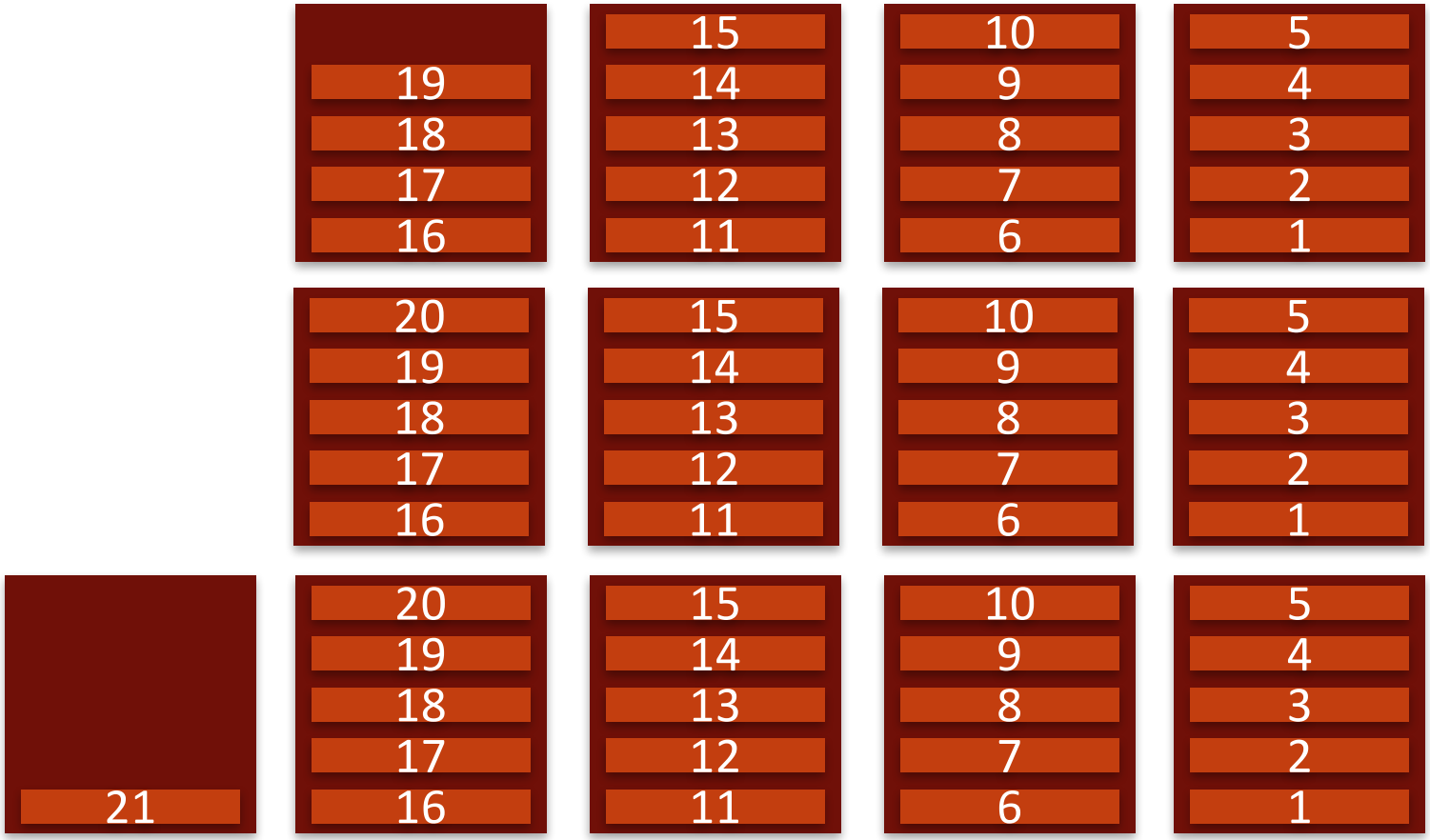


Publisher controls freshness of data

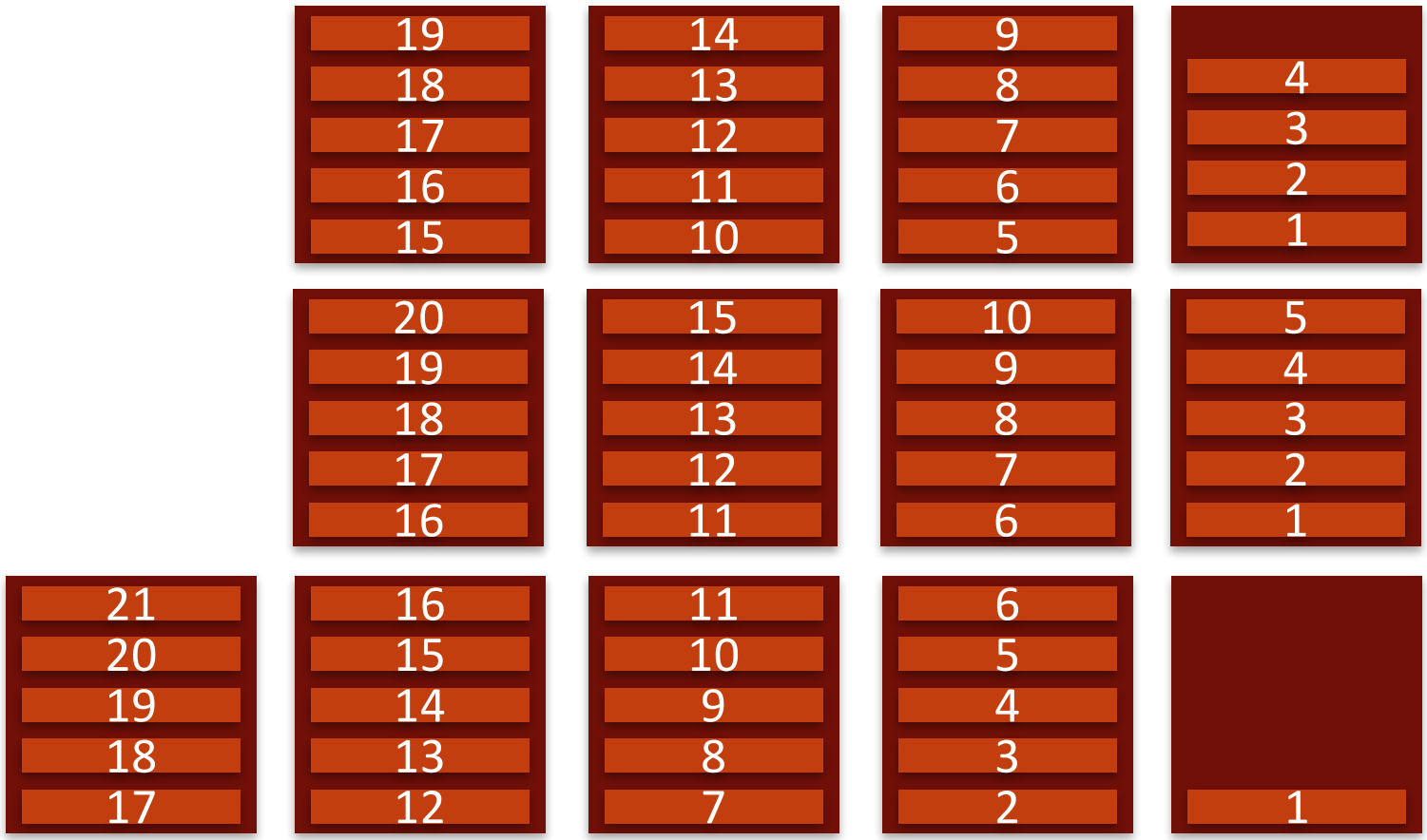
Implementation Tips

- Fill the current feed from zero to capacity before archiving
  - Allows for immutable archive feeds
  - `Cache-Control: max-age=2592000` (30 days)
- Separate consumption from production of feeds
  - Write on background thread
  - Serve from static files
- Push archive files to long-term storage
  - E.g. S3

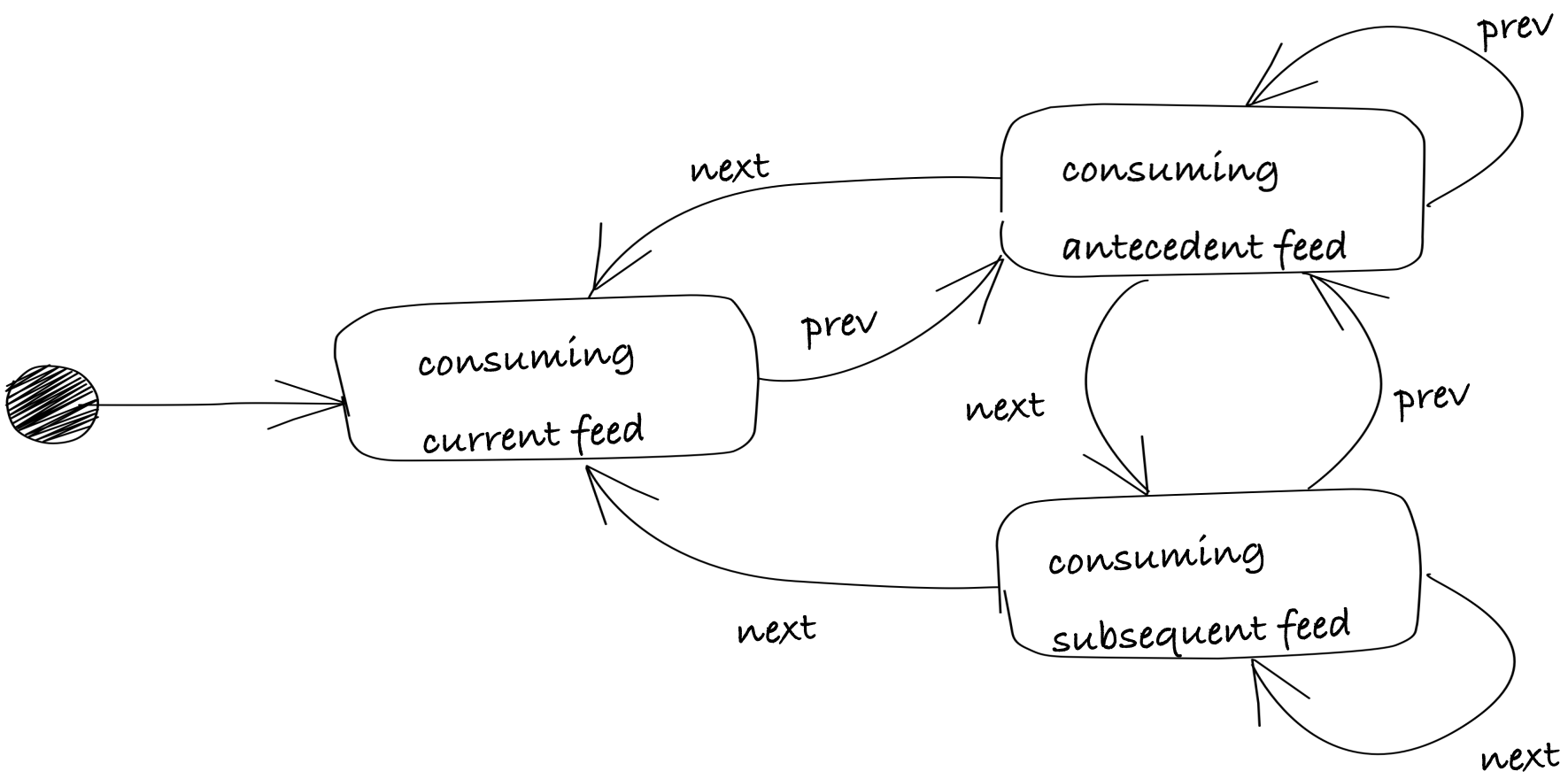
“Fixed window” feeds



“Sliding window” feeds



Domain application protocol



"application/atom+xml"

Atom Publishing Protocol

"An IRI of an editable Member Entry. When appearing within an atom:entry, the href IRI can be used to retrieve, update and delete the Resource represented by that Entry."

```
<link
    rel="edit"
    href="http://restbucks.com/products/notifications/2008/9/10/13"
    type="application/atom+xml;type=entry"/>
```

rel="edit"

Atom Syndication Format

Remember: "application/xml" is not your friend

"application/xml"

Processing model

?

?

It's XML…

Out-of-band processing model

Namespace

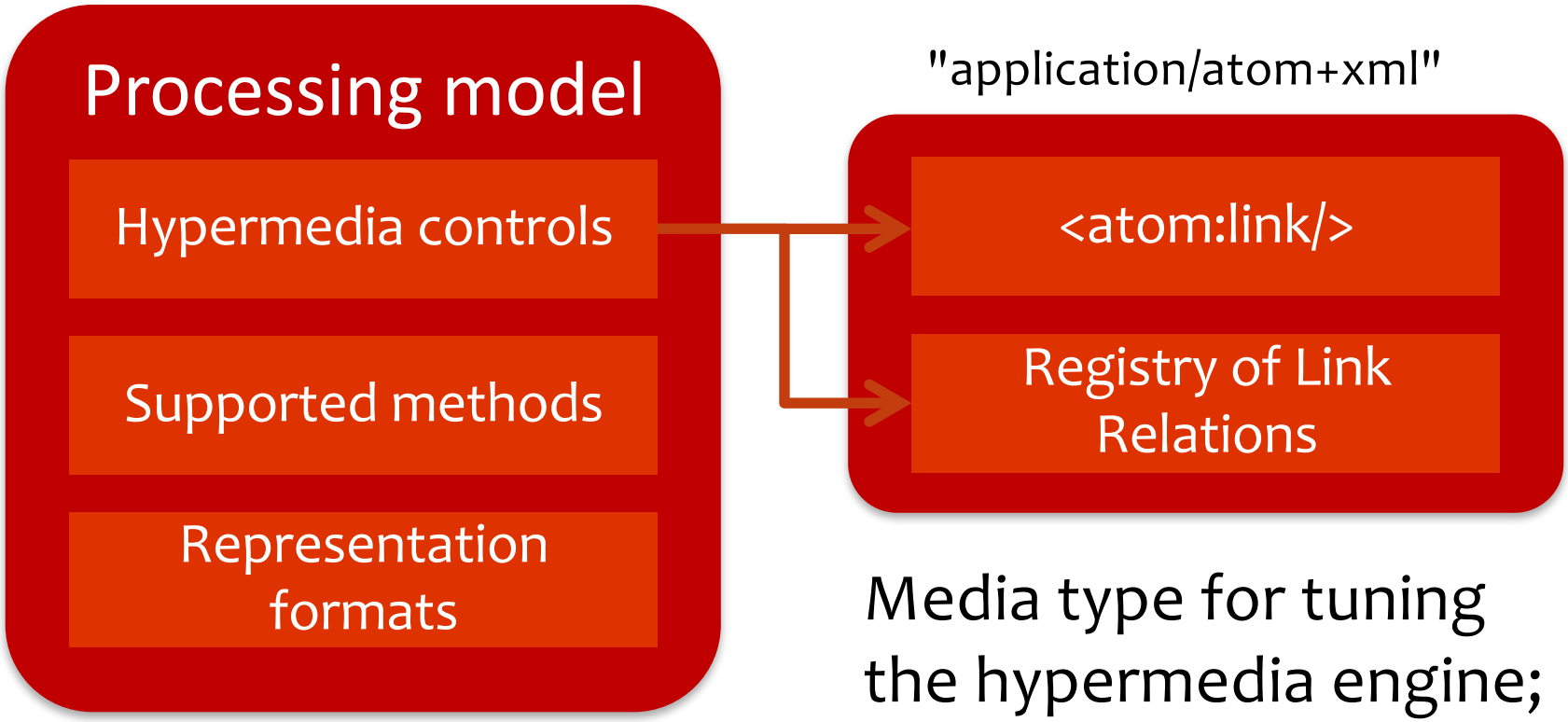
Application-specific hypermedia semantics

Documented operations

XML Schema, RELAX NG, etc

Custom media types

"application/vnd.restbucks+xml"



Media type for tuning the hypermedia engine; schema for structure

Different Caching Polices for Each Resource

Uri	Description	Caching
`/products/notifications`	Current	Short
`/products/notifications/{year}/{month}/{day}/{hour}`	Archive	Long
`/products/notifications/{entry-id}`	Notification	Long
`/products/{product-id}`	Product	Varies
`/products/{hardware-id}`	Promotion	Varies

Protocol Handoffs

- Use links and forms to advertise other protocols
  - Link to event archive/backup
    - Baseline state
    - Apply recent events on top
  - Queues and topics
    - JMS, AMQP, MSMQ
  - “Ping-and-Poll”
    - Subscribe for pings via, e.g., XMPP
    - On ping, poll the feed
    - Allows for infrequent polling, but low latencies when events do occur

Alternative: WebHooks

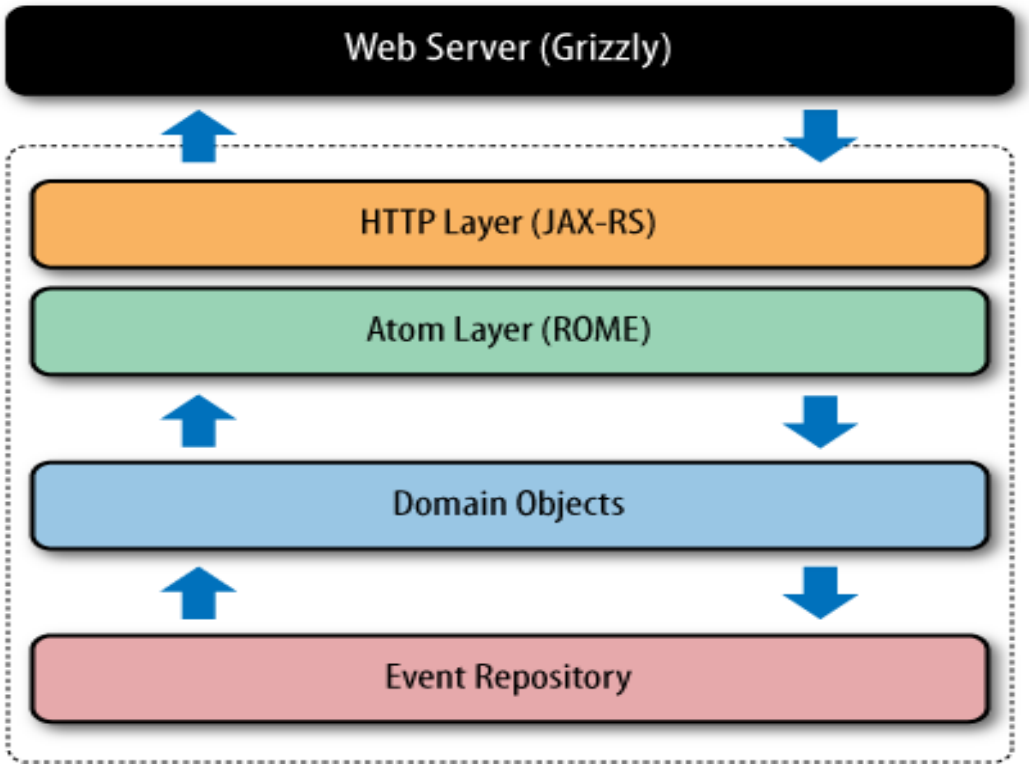
- Client subscribes one or more URIs for events
  - Server ***pushes*** events
- Pros
  - "Real-time" – no polling delays
  - No dependency on XML
    - Text, JSON, empty POST
  - Popular: Azure, BitBucket, Dropbox, GitHub, Instagram, MailChimp, PayPal, Pusher, Salesforce, Slack, Stripe, Trello, WordPress
- Cons
  - Variations in security, data formats, protocol
  - Server responsible for maintaining subscription list
  - Server responsible for guaranteed delivery
  - Possibility for out-of-order messages

Other Alternatives

- WebSockets
  - Streams of messages over TCP
  - HTTP request to switch protocols
  - Stateful connections
    - "Keep alive" messages, reopen closed connections
- Long Polling
  - "Hanging GET"
  - Do not supply `Content-Length`
  - Consumes connections/sockets
  - Good for short bursts of very frequent events


Example Java Implementation

Server-side architecture



Exposing a feed with JAX-RS

```
@GET
@Path("/updates/{startPos},{endPos}")
@Produces(MediaType.APPLICATION_ATOM_XML)
public Response getSpecificFeed(
  @PathParam("startPos") int startPos,
  @PathParam("endPos") int endPos) {

  if (invalidStartAndEndEntries(startPos, endPos)) {
    // Bad URI - the parameters don't align with our feeds
    return Response.status(Status.NOT_FOUND).build();
  }

  EventFeedGenerator generator = new EventFeedGenerator
    (uriInfo.getRequestUri(), ENTRIES_PER_FEED);
  Feed feed = generator.getArchiveFeed(startPos);

  return Response.ok().entity(stringify(feed))
    .header(CACHE_CONTROL_HEADER, cacheDirective
      (CachePolicy.getArchiveFeedLifetime()))
    .type(ATOM_MEDIA_TYPE).build();
}
```

Generating an Atom feed with ROME

```
private Feed getArchiveFeed(int startEntry) {
  Feed feed = new Feed();

  feed.setFeedType("atom_1.0");
  feed.setId("urn:uuid:" + UUID.randomUUID().toString());
  feed.setTitle(FEED_TITLE);
  feed.setAuthors(generateAuthorsList());

  List<Event> events = EventStore.current().getEvents(startEntry,
    ENTRIES_PER_FEED);
  feed.setEntries(createEntries(events));
  feed.setUpdated(newestEventDate(events));

  feed.getAlternateLinks().add(generateSelfLink(startEntry));
  feed.getAlternateLinks().addAll(
    generatePagingLinks(startEntry));

  return feed;
}
```

Add entries

```
private List<Entry> createEntries(List<Event> events) {
  ArrayList<Entry> entries = new ArrayList<Entry>();

  for(Event e : events) {
    final Entry entry = new Entry();
    entry.setId(e.getTagUri());
    entry.setTitle(e.getEventType());
    entry.setUpdated(e.getTimestamp());
    entry.setAlternateLinks(generateLinks(e));
    entry.setCategories(generateCategories(e));
    entry.setContents(generateContents(e));
    entries.add(entry);
  }

  return entries;
}
```

Generate paging links

```
private List<Link> generatePagingLinks(int currentFeedStart) {
  ArrayList<Link> links = new ArrayList<Link>();

  if(hasNewerFeed(currentFeedStart)) {
    Link next = new Link();
    next.setRel("next-archive");
    next.setType(ATOM_MEDIA_TYPE);
    next.setHref(generatePageUri(getServiceUri(),
      currentFeedStart + entriesPerFeed));
    links.add(next);
  }

  if(hasOlderFeed(currentFeedStart)) {
    Link prev = new Link();
    prev.setRel("prev-archive");
    prev.setType(ATOM_MEDIA_TYPE);
    prev.setHref(generatePageUri(getServiceUri(),
      currentFeedStart - entriesPerFeed));
    links.add(prev);
  }
  return links;
}
```

Client-side: consuming an event feed with Jersey and ROME

```
private Feed getFeed(URI uri) {
  // Jersey
  Client client = Client.create();
  ClientResponse response = client.resource(uri)
    .accept(ATOM_MEDIA_TYPE)
    .get(ClientResponse.class);

  String responseString = response.getEntity(String.class);

  // Rome code
  WireFeedInput wfi = new WireFeedInput();
  WireFeed wireFeed;
  try {
    wireFeed = wfi.build(new StringReader(responseString));
  } catch (Exception e) {
    throw new RuntimeException(e);
  }

  return (Feed) wireFeed;
}
```

Navigating links

```
private URI getPrevArchive(Feed feed) throws URISyntaxException {
  return getUriFromNamedLink("prev-archive", feed);
}

private URI getNextArchive(Feed feed) throws URISyntaxException {
  return getUriFromNamedLink("next-archive", feed);
}

private URI getUriFromNamedLink(String relValue, Feed feed)
  throws URISyntaxException {

  for (Object obj : feed.getOtherLinks()) {
    Link l = (Link) obj;
    if (l.getRel().equals(relValue)) {
      return new URI(l.getHref());
    }
  }
  return null;
}
```

Summary

- Event-driven systems quite do-able with Web
- Exactly-once delivery of messages practically possible (modulo FLP)
- Client pull, not server push, not middleware holding
- Entire event history can be held outside source server
- Entire event history can be navigated
- Caching provides availability, scale
- Latency trade-off is good: only cache the historic feeds!