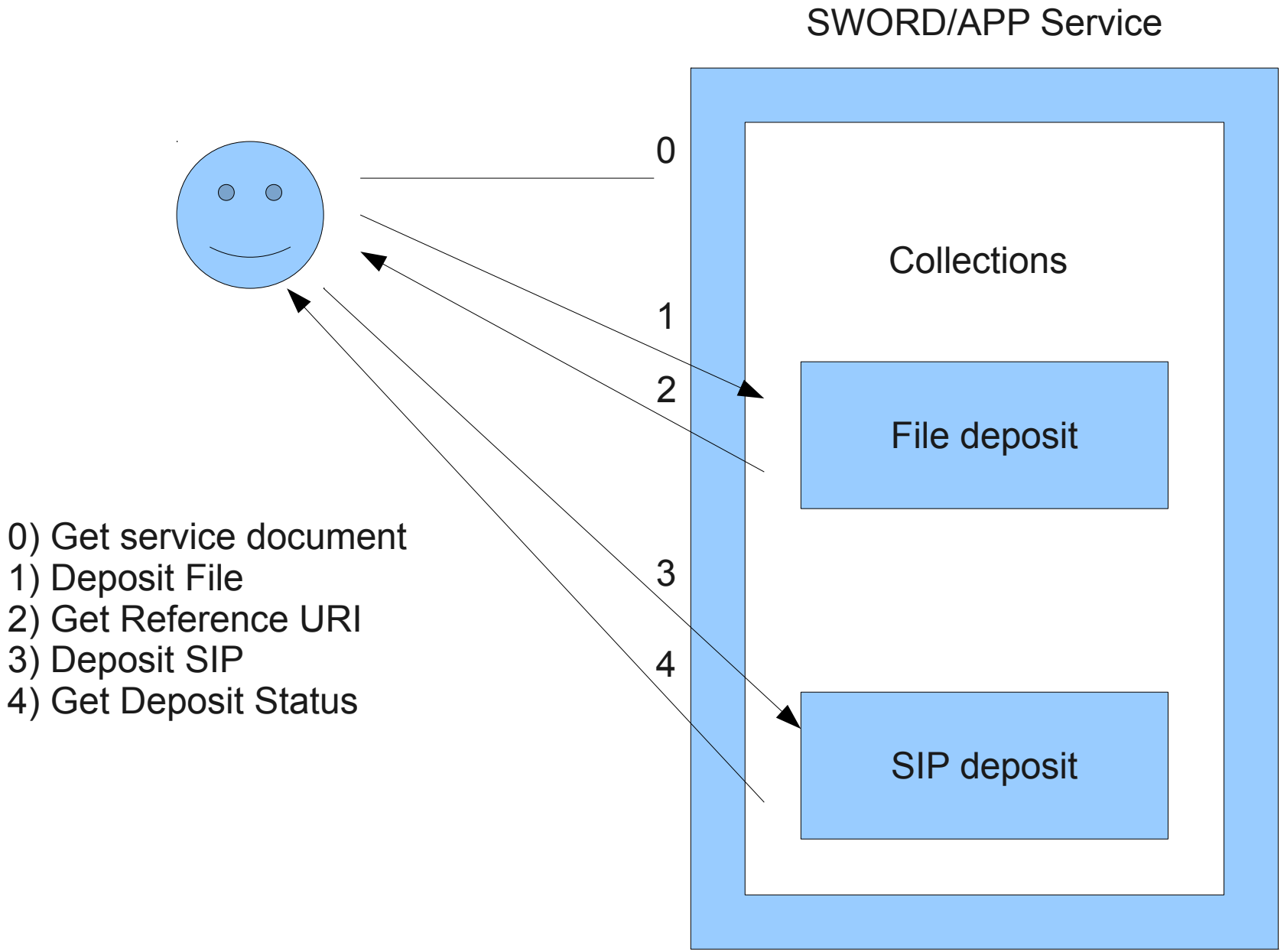
SWORD/APP Service
Collections
File deposit
SIP deposit
0
1
2
3
4
0) Get service document
1) Deposit File
2) Get Reference URI
3) Deposit SIP
4) Get Deposit Status

0) Service document

- http://dataconservancy.org/deposit/
- Specified by AtomPub
  - Enumerates depositable collections, organized into “workspaces”
  - Enumerates their accepted content types
- With SWORD extensions
  - Enumerates the accepted packaging
  - Lists other characteristics – policy, treatment, whether mediation is allowed, etc

```
<service xmlns:app>
   <sword:version>1.3</sword:version>
   <sword:maxUploadSize>12345678</sword:maxUploadSize>
   <workspace>
      <atom:title>Basic Deposit</atom:title>
      <collection href="http://dataconservancy.org/deposit/sip">
        <atom:title>SIP deposit</atom:title>
        <accept>application/xml</accept>
        <sword:acceptPackaging>
http://dataconservancy.org/schemas/dcp/1.0
        </sword:acceptPackaging>
        <sword:mediation>false</sword:mediation>
      </collection>
      <collection href="http://dataconservancy.org/deposit/file">
        <atom:title>File Upload</atom:title>
        <accept>*/*</accept>
        <sword:mediation>false</sword:mediation>
      </collection>
   </workspace>
</service>
```

- Links to SIP and file collections
- Specifies Dcp/XML as the only packaging for SIPs
- Specifies any file accepted for file upload

File Deposit & Response

- Optional – used when one has private content (somewhere) and wants to upload it to the data conservancy server
- Opaque URI is returned. Used for referring to file content in a subsequent SIP. Actually, an entire status document is returned, which contains the URI
- Uploaded file is considered temporary until it is accepted into the DCS through reference by an accepted SIP.

1) File Deposit

File upload is simply a standard http post, with no special requirements. However, any checksums will eventually be checked (but not necessarily right away)

```
POST /deposit/file
Host:dataconservancy.org
Content-Type: image/jpg
Content-Disposition: filename=one.jpg
Content-MD5: GcJfVu207OPHDUTQgSSGsA==
Digest:SHA=4TGfZP5L/llIKhQ48q7p6nxYSuo=,TIGER=/MJk32csHNHEGFX6
iHOD3MogE/nAR4rx
```

Response will return an atom entry document

```
HTTP/1.1 202 Accepted
Content-Type: application/atom+xml; charset="utf-8"
X-dcs-src: urn:dataconservancy.org:file/00123
Location: http://dataconservancy.org/deposit/file/00123
```

2) Response

- Atom entry document
- atom:content contains link to dcp document, containing a File entity representing the uploaded file
- The **src** attribute of this file entity is what the client needs to use to refer to the file in a SIP.
  - This is also conveniently in the response header “X-dcs-src”
- atom:link rel=alternate contains a link to a “status” document, containing an atom feed of all current events associated with that file in the ingest pipeline.

```
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:sword="http://purl.org/net/sword/">
  <id>deposit:/fileUpload/00123</id>
  <content type="application/xml"
    src="http://localhost:8080/dcs-integration-deposit/content/fileUpload/00123" />
  <title type="text">Deposit 00123</title>
  <updated>2010-08-02T14:12:50.926Z</updated>
  <author>
    <name>Depositor</name>
  </author>
  <summary type="text">ingesting</summary>
  <link
    href="http://localhost:8080/dcs-integration-deposit/status/fileUpload/00123"
    type="application/atom+xml; type=feed" rel="alternate" title="Processing Status" />
  <sword:treatment>Deposit processing</sword:treatment>
</entry>
```

- The 'content' link is dcp xml document containing a file entity and associated events.
- The important part is the X-dcs-src header, or the 'src' attribute on the content file entity. The rest can be disregarded by the client
- Or the file entity may be copied wholesale into the sip
- Provided values (filename, fixity checksums, mime type) are reflected in the entity
- Additional computed digests may be included
- Any associated events are included in the content dcp as well.
- Retrieving the atom doc at a later point might reveal changes. Http headers and atom “updated” will be updated accordingly.
- This entry will **not** appear in any feed. Thus, clients who care to check back need to hold on to the url

Questions so far

- Do we need to include a “self” link. That way, clients could keep a copy of the atom doc, and know how to retrieve a newer copy of it.

alternate/status document

- Atom feed containing dcs Events
- Events mapped onto Atom entry semantics
- The following are equivalent:

```
<entry>
  <id>http://dataconservancy.org/y1p-26</id>
  <updated>2010-07-01T15:40:02.289Z</updated>
  <title type="text">deposit</title>
  <content type="text">myOutcome</content>
  <summary type="text">myDetail</summary>
  <link href="test:/ref1" rel="related" />
  <link href="test:/ref2" rel="related" />
</entry>

<Event>
  <eventType>deposit</eventType>
  <eventDate>2010-07-01T15:40:02.289Z</eventDate>
  <eventTarget ref=”test:/ref1” >
  <eventTarget ref=”test:/ref2” >
  <eventDetail>myDetail</eventDetail>
  <eventOutcome>myOutcome</eventOutcome>
</Event>
```

3,4 SIP deposit and response

- SIP needs to be “self consistent” - contain links to entities within the SIP, or within the DCS
- References to any Files are made through the 'src' value. File entities in the SIP referencing uploaded content should use the 'src' value returned during the upload process.
- SIP is checked for basic schema validity before being accepted for conditional deposit
- One accepted package format for now: dcp xml 1.0

3) SIP deposit

```
POST /deposit/sip
Host: dataconservancy.org
Content-Type: application/xml
X-Packaging: http://dataconservancy.org/schemas/dcp/1.0
```

And the response is an atom entry document, of identical Semantics to that produced by file deposit

```
HTTP/1.1 202 Accepted
Content-Type: application/atom+xml; charset="utf-8"
Location: http://dataconservancy.org/deposit/sip/00abc
```

4) Response

- As for files, the 'content' element represents a dcp. In the case of ingest, this contains the entire set of submitted SIP entities, plus any additional events/entities added during ingest processing.
- At any time, retrieving the atom entry doc and inspecting the content dcp will reflect the current state of each entity – incorporating any transformations that ocurred during ingest processing

4) Response

- rel=alternate link points to an atom feed document containing a feed of all events submitted or generated during ingest
- Headers and updated dates are updated to reflect reality
- If a client wishes to determine the id assigned to an entity, it can either introspect in the dip, or the events.
- Every id assignment creates an event, whose detail contains the old and new identifiers, for example

Overview

- Two relevant maven projects:
  - dc-deposit: Contains core java-based deposit API, as well as a sword server based on Abdera.
    - Very general. Does not involve DCS semantics or entities.
  - dcs-ingest: Contains the ingest framework, services, and an implementation of the deposit API which allows file and sip deposits into the ingest pipeline

dcs-deposit-core

- Primary deposit interface (slightly abridged):

```
public interface DepositManager {

	public DepositInfo deposit(InputStream content,
				String contentType,
				String packaging,
				Map<String, String> metadata);

	public DepositInfo getDepositInfo(String id);
}

public interface DepositInfo {
	public DepositDocument getDepositContent();
	public DepositDocument getDepositStatus();
	public boolean hasCompleted();
	public boolean isSuccessful();
}
```

dcs-deposit-sword-server

- Extends abdera server framework to add Sword support
- DefaultSWORDProvider: assembles workspaces of sword-enabled collections
- SWORDCollectionAdapter: Abstract abdera collection adapter with additional sword semantics
- DepositManagerAdaptor: concrete implementation of SWORDCollectionAdapter which wraps an implementation of DepositManager, and takes care of all the swordid details.

dcs-deposit-status

- Should probably be re-named
- Contains simple servlets for serving DepositDocument content derived from a DepositInfo
- Contains 'StatusResourceLocator' for generating URLs to appropriate place (i.e. the simple status servlet). Used for generating 'content' and 'alternate' link urls.