API Architecture Outline

Here is an outline of possible content for a Medbiquitous API Architecture document. I included questions that we need to discuss or research. I would like to discuss this outline in the Monday TSC call and hopefully get volunteers to look into some of the questions.

1. Principles and Objectives
    a. Light-weight
    b. Easy to implement
    c. Secure
    d. Extensible (method extensibility/parameters, upward compatibility, extensible payload)
2. Interaction
    a. REST
    b. Payload format (need to decide what is required)
        i. JSON
        ii. XML
        iii. Both (design of new data model must think about both. We can require providers to support content negotiation.)
    c. REST Guidelines
    d. JSON Guidelines (what can this be based on?)
3. Granularity
    a. Individual item response (JSON preferred?)
    b. Document fragment response (XML well defined subset of existing MedBiq standard documents)
    c. Need additional allowable value definitions like controlled vocabularies?
4. API signatures
    a. PURLs - how to host, construct, querying vs persistence, item potency, concurrency
    b. /medbiq/api/…
5. Transactions
6. Attachments (do we need in V1? How to do in JSON?)
7. Notifications (do we need this? Wait for use cases? What are notifications in this context?)
8. Security
    a. HTTPS, PKI
    b. Updated single sign-on guidelines
        i. API oriented
        ii. Latest standards
        iii. Guidelines, not requirements
    c. Access Control, Labels (do we need for V1? Guidelines)
    d. Audit (Guidelines, do we need for V1?)
    e. Digital Signatures
        i. Just include signature or assure no tampering and non-repudiation?
        ii. How to include in JSON? Possibly - http://self-issued.info/docs/draft-jones-json-web-signature-json-serialization-01.html
    f. Privacy (deals with data that is stored or retrieved, also part of Access Control - is this needed for V1?)
9. Extensibility
10. Standard exceptions
    a. Do we have some that cross all APIs, like unknown id?
    b. Standard exception format

Note on Content Negotiation: the following text is form our REST Web services design guidelines:

1.1 Determining Which Representation to Use

When a service supports more than one output format, the service may support the accept parameter or Accept http header in order to allow the client to select the format to use.

To determine what format the client wants for the representation, the client should do the following:

- Look for an `accept` URI parameter. This parameter's value should follow the same format as the Accept header
- Else, look for the Accept http header

Clients should use the Accept header as the preferred indicator of format. The URI parameter is provided as a convenience for clients that have limited header support.

If the format is unsupported, then the REST service should return an error. If no content header is present but there is body content along with the request, then the service should use the same content type for the response. If the request contains no body and no content header, then the service should select a default. The default for MedBiquitous is XML.