Mobile Application Programming

Networked Data Models
### REST

- **Representational State Transfer (REST)**
- **Software architectural style** used in protocols like HTTP

<table>
<thead>
<tr>
<th>Resource</th>
<th>GET</th>
<th>PUT</th>
<th>POST</th>
<th>DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collection URI, such as <a href="http://example.com/resources/">http://example.com/resources/</a></td>
<td>List the members of the collection, complete with their member URIs for further navigation. For example, list all the cars for sale.</td>
<td>Meaning defined as &quot;replace the entire collection with another collection&quot;.</td>
<td>Create a new entry in the collection where the ID is assigned automatically by the collection. The ID created is usually included as part of the data returned by this operation.</td>
<td>Meaning defined as &quot;delete the entire collection&quot;.</td>
</tr>
<tr>
<td>Element URI, such as <a href="http://example.com/resources/7HOU57Y">http://example.com/resources/7HOU57Y</a></td>
<td>Retrieve a representation of the addressed member of the collection expressed in an appropriate MIME type</td>
<td>Update the addressed member of the collection or create it with the specified ID.</td>
<td>Treats the addressed member as a collection in its own right and creates a new subordinate of it.</td>
<td>Delete the addressed member of the collection.</td>
</tr>
</tbody>
</table>

See chapter 5 of Roy T. Fielding’s doctoral dissertation and REST article on Wikipedia
What if something goes wrong?
Errors and Asynchrony

- Error handling with synchronous requests is easy - tell the user the request failed and *suggest a solution*
- With asynchronous communications, the user has *moved on*, doing other things
- Showing a simple error here leaves the user *unsure* how much of their work is corrupted
- The typical solution, “Try again later”, you can build into the system yourself by way of a *transactional cache*
Network Architecture
Network Architecture

Diagram showing the relationships between View, Controller, and Model.
Network Architecture
Network Architecture
Network Architecture
Network Architecture
Network Architecture
Network Architecture
Network Architecture

Data Model

Model Rules
Logic
Data Validation

Session

Cache

Server
Network Layer

- **Point of contact** with the server
- Represents **app’s perception** of the server’s interface
- Typically has **one method for each** server end point
- **Converts** between app language objects (Swift) and server protocol (JSON, XML, binary, etc)
- Contains **no state** - implement as a static class
Session

- Stores the state of the network relationship
- Authentication Token
- Current Device ID, User ID, Game ID, etc.
- May be just a few variables in the model or an object
- Can be used as the point of contact with the network layer if the relationship with the server is highly stateful
- Coordinates with the network cache to see when data needs to be updated from the server or just returned
Data Cache

- Prevents asking for the server for data more often than is needed
- Stores server responses or data with expiration times
- Makes up part of the persistent state of the data model
- Can be implemented in many ways
  - Custom storage of model data
  - Cached responses from the server
  - Implicit in the network requests (transparent)
Model Interface

- Access to model data that is only on the server at any time should be asynchronous.
- Use callback-style request methods for data that include a way to return errors in retrieving the data.
- Call the callback on the listener an appropriate thread.
Another Solution
Transactional Cache

- Divide the data in the system into two parts:
  - Server Cache - The data most recently received from the server in previous successful requests
  - Local Modifications - Requests by the user to update server information that have not yet been successfully transmitted to the server
- An operation queue can be used to continually attempt to send local modifications to the server
- Show the user a combined version of the server cache with local modifications applied, if appropriate