I’M NOT HERE
HELLO
design process

- goal: maintain distributed management but create integrations
- high level requirements > use cases - surveys, interviews
- three types of meetings: tech archive stakeholder, LTS, ecosystem meetings to work out and iterate on design
- University of Houston (Bayou City DAMS), Rutgers University, University of Toronto, Rockefeller Center
- reached out to IT, contractors
integration hub
digital repository backend

github: digitaldu-backend

redis

elasticsearch

node.js
digital repository frontend
github: digitaldu-frontend

UV

IIIF
International Image Interoperability Framework

elasticsearch

KALTURA

node.js
<table>
<thead>
<tr>
<th>advantages</th>
<th>dis(s)advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>lower resource requirements</strong> - reduce burden of one dept having to manage entire stack</td>
<td><strong>loss of control</strong></td>
</tr>
<tr>
<td><strong>independence</strong> - economic, technological, legal. separation of concerns - department can manage the way they want to for most of their own purposes. parts can be swapped out</td>
<td><strong>integration requires coordination</strong>, dependent on others to “play their part”</td>
</tr>
<tr>
<td><strong>deep collective knowledge</strong>, no black boxes</td>
<td><strong>knowledge is siloed</strong> per department</td>
</tr>
<tr>
<td><strong>abundant mentality</strong> towards partners - invested in each others’ success</td>
<td><strong>additional potential points of failure</strong> (such as departmental roadblocks)</td>
</tr>
<tr>
<td><strong>greater efficiency</strong></td>
<td><strong>risk if responsibility</strong> for integration <strong>unclear</strong></td>
</tr>
</tbody>
</table>
integration considerations

really hard to test until integrations are all finished

each system does not perform at the same speed

working within each system’s limitations - what you can do with the API they provide

doesn’t really reduce the maintenance problem (just distributes it)

monitoring is essential
lets work together?

Some ideas:

- Common architectural patterns? Maybe think through something like the OpenSystemsInterconnection model for common standards for integration?
- Develop reusable workflows?
- Develop modular integrations?
- Work through pain points?
online all the time

Email kim.pham60@du.edu

Twitter @tolloid

Do not check out my soundcloud