#10 Workflow State Modeling

Jeremy Friesen (Notre Dame)

- Mediated deposit: nothing goes in to repository without being viewed
- Were going to do workflow app for ETDs, but others came up, made Sipity
- 2 workflows in production, 3rd being discussed for Irish journal
- Additional one for internal projects

Esmé Cowles (Princeton)

- 1 app - book digitization, ASM workflow
- uploading of images, review, published state, can flag for attention/takedown
- hang things off of states, have some notifications, want to personalize more
- 1 other app with different workflow, have needs to plug them in to different models

Lynette Rayle (Cornell)

- 1 app, basic workflow
- 3 fields that users can control: 1 can set workflow state, 1 can assign it to user(s), 1 field can comment
- states are configured in a YAML file
- will have more complex needs

David Trujillo (UCSD)

- interested in describing workflows and Sufia UI

Joe Atzberger (Stanford)

- very complex automated workflow
- different steps, sub-steps, majority are automated
- due to amount of writes affecting solr, split this off into own separate service
- model has workflow datastream, dependency on remote service
- when to reindex

Ryan Wick (Oregon State University)

- basic workflow in Oregon Digital: single review step
- looking at more options
- Sufia-based IR will have more workflow, desire mediated deposit

Mark Bussey (DCE)

Matt Critchlow (UCSD)

Jeremy:
role-based actions, map to states
normalized way to describe workflows in Sipity to JSON
struggling with AASM, FSM

Types of workflows:
- basic steps from New Work to Item in repository
- Publication workflow - create work, add metadata, review, publish - performed by humans
- Automated workflow
- Exceptions workflow - already published but needs review, DMCA takedowns, etc.
- Editing/Reviewing already published works

Sipity

work areas - ex: ETDs
submission windows - ex: self-deposit can be open window
work type - ex: master's thesis

work areas:
https://github.com/ndlib/sipity/blob/master/app/data_generators/sipity/data_generators/work_areas/ulra_work_area.config.json
work types:

JSON gets evaluated and written in to DB
graph gets built from database
role has all of these permissions
can assign person or actor to role at strategy level, or work level
for UI, required actions are pulled out to Sipity checkboxes
every work area and submission has a processing entity
- other notification options? dashboard note, text, etc.
  - feasible but not done yet
  - delivery subsystem is isolated

  currently doesn't write to Fedora until publish
  could direct write earlier
  ID is minted early, helps with writes and JSON transport

  when validator is on ActiveRecord, tougher to migrate data

Esmé
https://github.com/pulibrary/plum/blob/master/app/models/state_workflow.rb
aasm is a gem
can build state machines into object
chose to make them their own objects
declare states, declare transitions
can add guards to see if a transition is allowed
model has state property, calls StateValidator - state_validator.rb
renderer displays radio buttons
notifications done in callback
  currently 1-way workflow, could add regressions
  may add additional workflows

? where to persist?
plum puts it in the AF property
sipity puts it on the processing entity

plum using cancan
sipity using pundit API, devise and cancan not included

Hull's workflow looks very similar to Plum

How to work this in to Sufia
possibly work pieces in to CurationConcerns

Draft flag/status
Embargos vs. Lease - more restrictive -> less restrictive

UI vs. JSON
multiple custom workflows
start with minimum viable, probably JSON

Esmé will reach out to Jeremy about modeling what Plum has in JSON
write code to parse JSON to generate state machine instead of hardcoding
Goal for moving to JSON is to allow more complex workflows and better sharing