

[Link to 2021 ACAMP Wiki](#)

Title: COnfigure Manage configuration

Advance CAMP Wed. Oct 6, 2021

Room - Climbing Tower

CONVENER: Allan Kim

MAIN SCRIBE: Laura and lots and lots of other people because you are all super generous with your typing :)

ADDITIONAL CONTRIBUTORS:

of ATTENDEES: 15

DISCUSSION:

Session Scoping

Allan remarked that the configuration of COnfigure Manage is mostly through the GUI, though some configurations can be more complicated to manage, maintain, and setup this way. What have other deployers used to manage changes and the care and feeding of a installation?

Strategies for managing COnfigure Manage configurations

- SNAPSHOT A CA deployer has been taking a snapshot of the database - perhaps not the end solution (*Benn O*)
 - Standup an instance
 - take a snapshot of it
 - Make changes / test ideas / play around
 - Restore back to your snapshot
 - If you make config changes that have to be preserved, you need to make sure you re-snapshot
- TEMPLATE CO (*Benn O*)
 - Create a CO & configure it the way that you want it
 - Use that CO as your base template for creating other COs

- Challenge: Won't update existing COs as you make changes.
 - Develop proper tooling for this?
- FUTURE IDEAS (perhaps with the Framework migration update):
 - YAML or JSON export/import
 - [CO-1407](#) Copy Configuration to JSON
 - Export useful: run a diff between environments
 - Q: what format is most helpful?
 - Scott: manages over dev/test/prod tiers - can be difficult migrating configuration from dev to test to prod. If it works in test, he'll collaborate with an admin or carry to prod himself - is largely a manual process
 - Interested in both export and import
 - JSON first, YAML second.
 - "Migration Plugin" ([CO-2239](#))
 - Use the API to set up parts of the configuration
 - Benn: metadata can be tricky - but at framework level, tooling already knows how to deal with this -- maybe a few UI driven approaches
 - External tool with REST API?
 - V2 maybe
 - Scott: interested in declarative ways to evolve the state configuration - lots of clicks otherwise; would like to be able to do this in (e.g.) a YAML file.
 - Drive COmanage with Ansible playbook would be happy-making (popular idea!)
 - Process for upgrading (e.g. from version 3 to 4)
 - Documentation of changes to YAML files ... so they could be made
 - Plugin? Try it - and iterate.
 - Might be nice to unify code base where configuration / other shells can be unified
 - Allan: Export is very helpful (first priority) - way to persist and compare configs (and back up in a human-readable way)
 - Is this tooling intended for sysadmins or admins within the application?
 - Scott: wouldn't expect CO admins to deal with this; would be aimed at those who run the platform.
 - Very helpful for comparing configurations when an admin breaks things! E.g. dump an export and compare it to how it was previously configured
 - Benn: reporting - generally useful tool: show config changes within a window of time, maybe a plugin, maybe filter by CO/timestamp/exclude noisy tables/etc ([CO-2238](#))
- How are deployers deploying? Containers? VMs? Bare metal?

- Scott: mixture. A lot of groups have moved to containers (Docker run ... to Kubernetes) - but a lot running on VMs. None we are aware of on bare metal anymore.
 - Containers well supported, but COmanage project is not expecting to go exclusively to containers (in the near future)
 - Dual support model for indefinite future (containers / VMs)
 - Benn: Most common deployments are to EC2, some kind of container is a close second

Some underlying questions

- When choosing the direction for future ideas, which approaches would gain traction with users?
- DB-level functions that require privileges
 - only needed for setting up a new SQLserver provisioning;
 - extended attributes (legacy - these update a table dynamically under the hood - this is going to be replaced by ad hoc attributes which are handled in a way that doesn't require a DB update);
 - schema management stuff (maybe exceptions for upgrades)
 - Matchgrid (for COmanage Match)
- HA Configuration?

Size impacts

- Number of CO Person records? - COmanage team testing environment using a local postgres (not optimized)
- the PHP apps have a known way of scaling (AWS...)
 - Challenges index reconstructions
 - (60,000 CO Person records) Amazon DB T3 Large was too large - T3 Small was good enough. Is not an app that punishes the DB significantly.
- Background processing (through jobs) can help with processing challenges

Match in the wild?

-