Zowe “app store”

A pluggable network package manager registry architecture for “zwe”

Initial concept

- “zwe” can install packages of standardized format when provided their local path
- What if the path could be substituted for a query to an off-the-shelf package manager to retrieve the same package
  - Same zowe action takes place, but getting files on disk becomes more powerful
- Most package managers adhere to common functions, “install”, “upgrade”, “uninstall”, “search”. These would be useful in “zwe”.
- Most package managers do automatic dependency resolution
  - Installing becomes easier if you can install 3 dependencies in 1 operation.

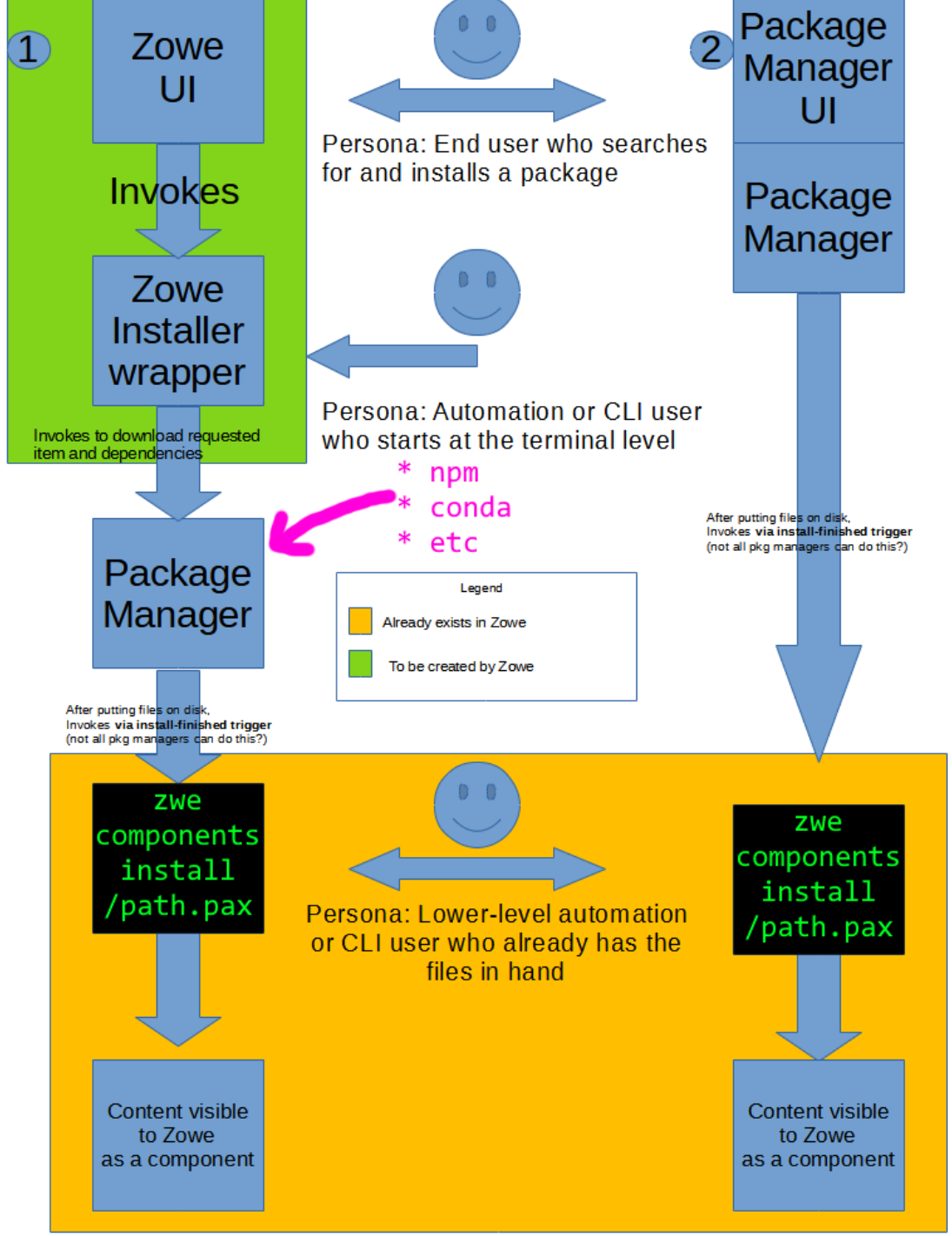
Which package managers are good on z/OS?

- z/OS requirements may rule out some technologies. We need:
  - Custom package registries. Offline access as an option. Getting files from untrusted internet places will not work. Curated internal networks are ideal, so they must be easy to create.
  - Simple to install – common dependencies and few of them
  - A version that exists for z/os
  - Can handle file tagging
  - SMPE will continue to exist. How does another package manager coexist?

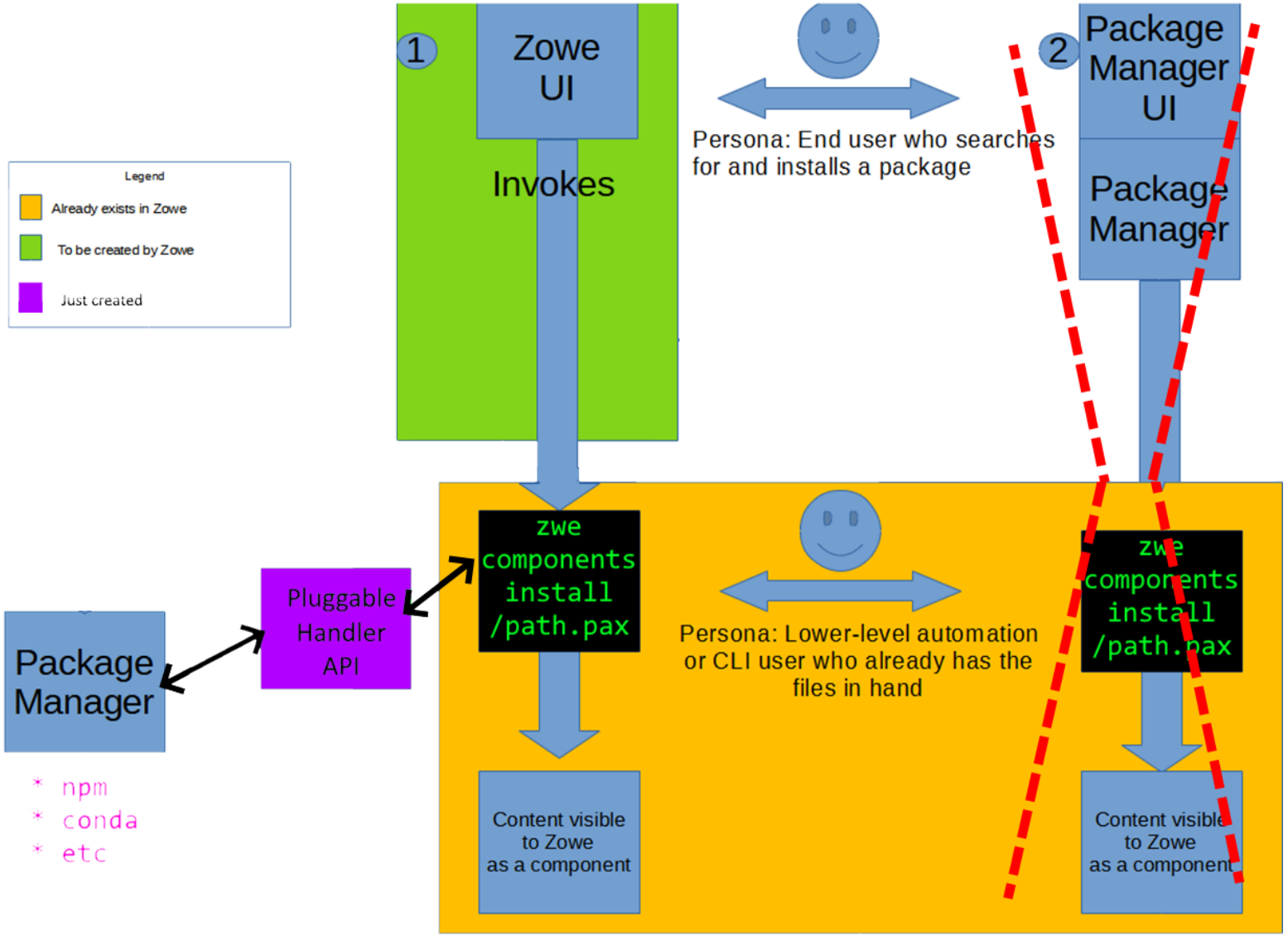
Which package managers are good on z/OS?

- Candidate package managers: npm and conda
- npm:
  - Most already have it, due to nodejs usage in zowe
  - Can setup LAN registries quickly (I did for the demo!)
  - Can namespace different packages to different registries (get company A's product from company A's registry, and product B from B?)
  - Assumes library is nodejs code… but accepts anything really.
  - Assumes ASCII…. But a package can just be a pax archive!
- conda:
  - Python-based, but python already on the platform
  - Also easy LAN registry setup
  - Multiple registries can coexist any way you want – even more capable than npm
  - Does not assume package language
  - Does not assume ASCII, but again a pax archive is fine too!

Initial implementation concept



What actually happened



Innovation week end result

- New “zwe” commands and functionality. “zwe components” can delegate to a “zowe extension registry handler”
  - npm as a handler is 100% functional in demo
  - No blockers on conda, it’s 25% done, just ran out of time this week ☺

zwe

- init…
- components
  - install -c zowe.yaml --component (path or query) [--handler npm] [--registry https//localhost:1234/] [--dry-run]
  - search -c zowe.yaml --component (query) [--handler npm] [--registry https//localhost:1234/]
  - uninstall -c zowe.yaml --component (name) [--handler npm] [--registry https//localhost:1234/] [--dry-run]
  - upgrade -c zowe.yaml --component (name|’all’) [--handler npm] [--registry https//localhost:1234/] [--dry-run]

Innovation week end result

- Any “zwe components” command can accept --handler and --registry to state which handler or registry to use for a command.
  - Defaults specified in zowe.yaml within zowe.extensionRegistry
- Install
  - Can accept a path or component name. If path, skips package manager. Existing behavior, no code change.
    - If component name, delegate to package manager. New “registry handler” API
  - Package managers may install more than 1 object if dependencies needed.
    - Zwe upgraded to handle this. Can do multiple install operations with a single input.
- Upgrade
  - Takes the name of an already installed component, or “all”
  - Upgrades all existing packages related to the given name, if dependencies exist
  - Essentially “install”, but for existing things.
- Uninstall
  - Takes name of already existing component
  - Removes the component, disables it in zowe.yaml, but doesn’t remove zowe.yaml customizations in case of future re-install
  - Package manager tracks if a package was only installed for the purpose of dependency, so uninstall may remove dependencies automatically, 1 uninstall may remove 3 things.
- Search
  - Takes a query. Can be a component name, id, tag, whatever a package manager accepts. Can be wildcard, can be versioned.
  - Prints out whatever the package manager prints out, whatever format.

Handler API – new code

- npm and conda work in zwe by “zowe extension registry handlers”
- Handlers can be built-in or 3rd parties can plug-in. They are found by giving their path in zowe.yaml config.
- They are ECMAScript2020-compatible JavaScript module code (NOT nodejs), run in the zwe scripting environment.
- Run with input/output environment variables.
- Input: ZWE_zowe_extensionDirectory
  - ZWE_CLI_PARAMETER_REGISTRY: In whatever format the handler understands
  - ZWE_CLI_REGISTRY_COMMAND: install | upgrade | uninstall | search
  - ZWE_CLI_REGISTRY_DRY_RUN: true | false
  - ZWE_CLI_PARAMETER_COMPONENT_NAME: A string
- Output: ZWE_CLI_PARAMETER_COMPONENT_FILE: A CSV of one or more paths (to install) or names (to uninstall), or ‘null’ if failure or nothing to do.

The code

- https://github.com/zowe/zowe-install-packaging/pull/2980
- Handler API
- npm handler
- schema update

This presentation is uploaded to the PR.

Futures

- Why not add it into Zowe right away?
  - It does nothing if not configured, adds no chores or dependencies
  - If nodejs present, configuration only takes a moment, and even registries are easy to set up on-prem: https://blog.bitsrc.io/how-to-set-up-a-private-npm-registry-locally-1065e6790796
- Setup zowe's own registry, and publish some stuff!
- Conda support will take a more few days, its much the same code to write as npm
- "real app store" means making a UI. Putting a UI on top of zwe and putting that in the Desktop and APIML seem like exciting next steps.

Futures 2

- More complex features could be added including
  - Ability for zwe to tell registry handler about components installed outside the package manager, so that packages can have dependencies satisfied regardless
  - zwe should check manifest.yaml (not pkg manager data) to see if a component depends upon a core component, and throw error if the core component is missing/disabled.
  - If a package manager can UNINSTALL packages during an upgrade (cleanup no-longer-needed dependencies) then zwe must turn “upgrade” into a hybrid install-and-uninstall operation
  - Test multiple registries coexisting… it might “just work” already.