yocto PROJECT

It's not an embedded Linux distribution –
It creates a custom one for you.

HOB V2 Plan

Contributor:
Dongxiao, Jessica, Ke, Lianhao, Shane

Reviewed by:
Josh, Richard, Saul

yocto PROJECT · THE LINUX FOUNDATION

Agenda

- Plan
- Requirement
- Design

Plan

- 1.2
  - Provide accurate data (runtime depends, size etc.)
  - Support package removal
  - Incremental UI enhancement (progress bar etc.)
  - Start bitbake client/server model restructure
- 1.3
  - Finish bitbake client/server model restructure
  - Start WebUI front end
- 1.4
  - Finish WebUI front end

Example usage scenario of 1.2 deliverable

Kyle is the chief engineer of embedded systems development team. He wants using HOB to evaluate the possibility of building NAS solution with Yocto. After launching the HOB, he start to selects the “core-image-basic” as “base image” and select extra samba, NFS related recipes. He then clicks the button “Build Package”. HOB shows the progress bar for the building. Seeing it may need quite some time to finish, Kyle decides to have a coffee first.

When Kyle is back, the package build is done. The size of all packages and image are also shown. He notices the image size exceed his planed size limitation, and he also finds some packages are actually not necessary. So Kyle decide to de-select those packages from image. During the process, HOB occasionally reminds him that the to-be-removed package is depended by other packages, and asks for confirm. Kyle finds it is quite helpful since it prevents removing desired package by mistake.

Finally the package de-selection is done, and Kyle is satisfied with the image size HOB showed, so he clicks the “Build Image” button to construct the image from package. Soon the image is ready, he then clicks the button “Run in QEMU” to play with image in QEMU. Kyle is happy to find that samba and NFS works pretty well, which is good news for him to go to next step…

Requirement

Requirement

- Provide accurate data
  - accurate runtime dependency, packages list , size
- Support package removal
- UI incremental enhancement
  - Add “progress bar”
  - Add build output
  - Add button to run image in qemu

Requirement (Cont.)

- Well defined bitbake server/client model and modular reusable interface
  - Meeting the requirement of OSV: if OSV want to develop its own UI frontend, they can reuse most of the API, and focus on the upper UI design instead of the underlying infrastructure.

Requirement (Cont.)

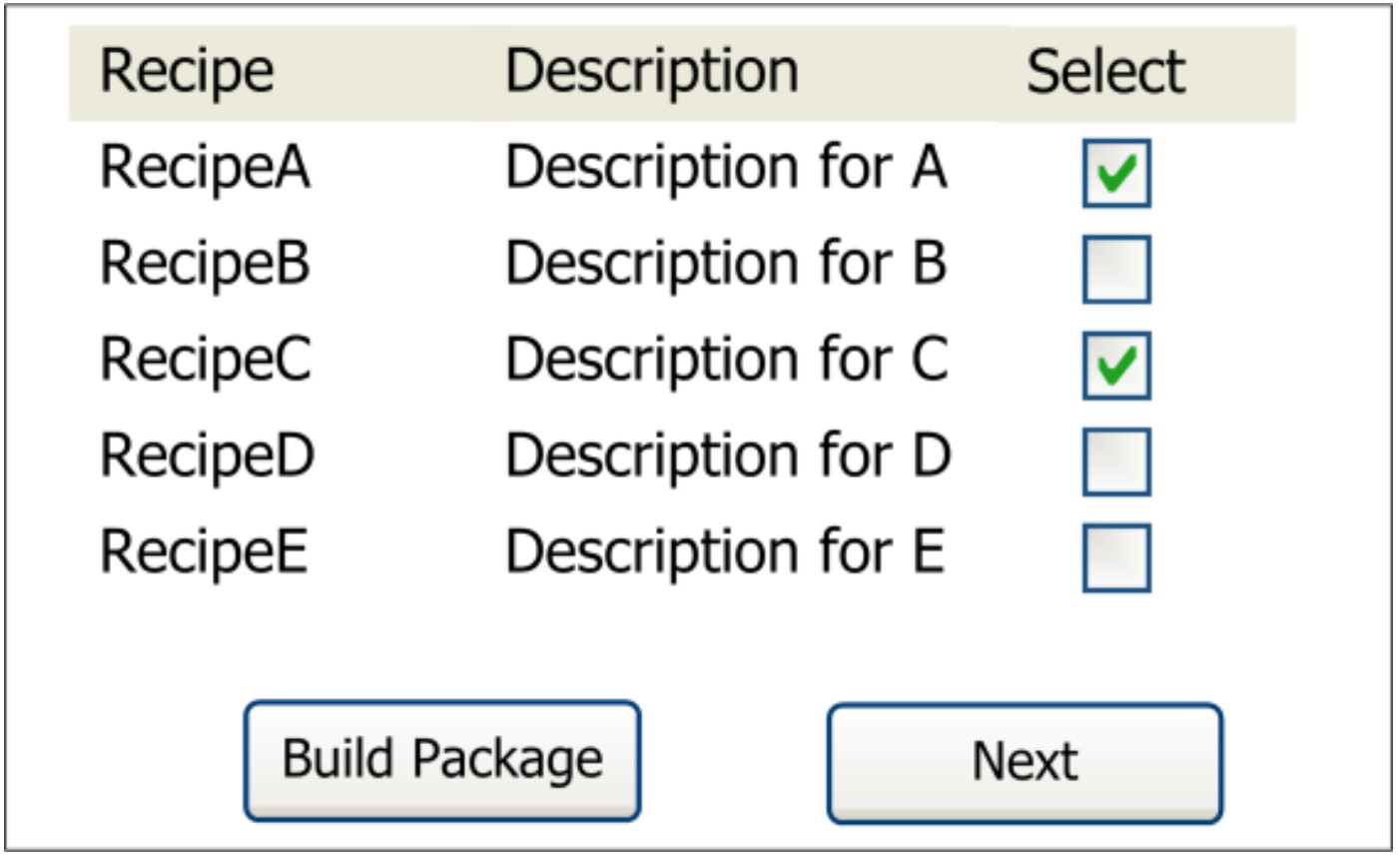
- New frontend: e.g. web style UI
  - Meeting requirement of Persona FAE: considering customer want to a video player demo. the FAE can access the web from windows, and customize the image with video player added, and download the image from web, burning it to device to show demo with customer.

Design

Req1: Accurate Data

- Accurate data
  - runtime dependency, packages list , size …
  - only available after packages are built
- Proposal: two build stage build for accurate data
  - Stage 1: build package
    - User select recipes, and build package
  - Stage 2: build image
    - User select already-built packages, and build image

Two Stage Build

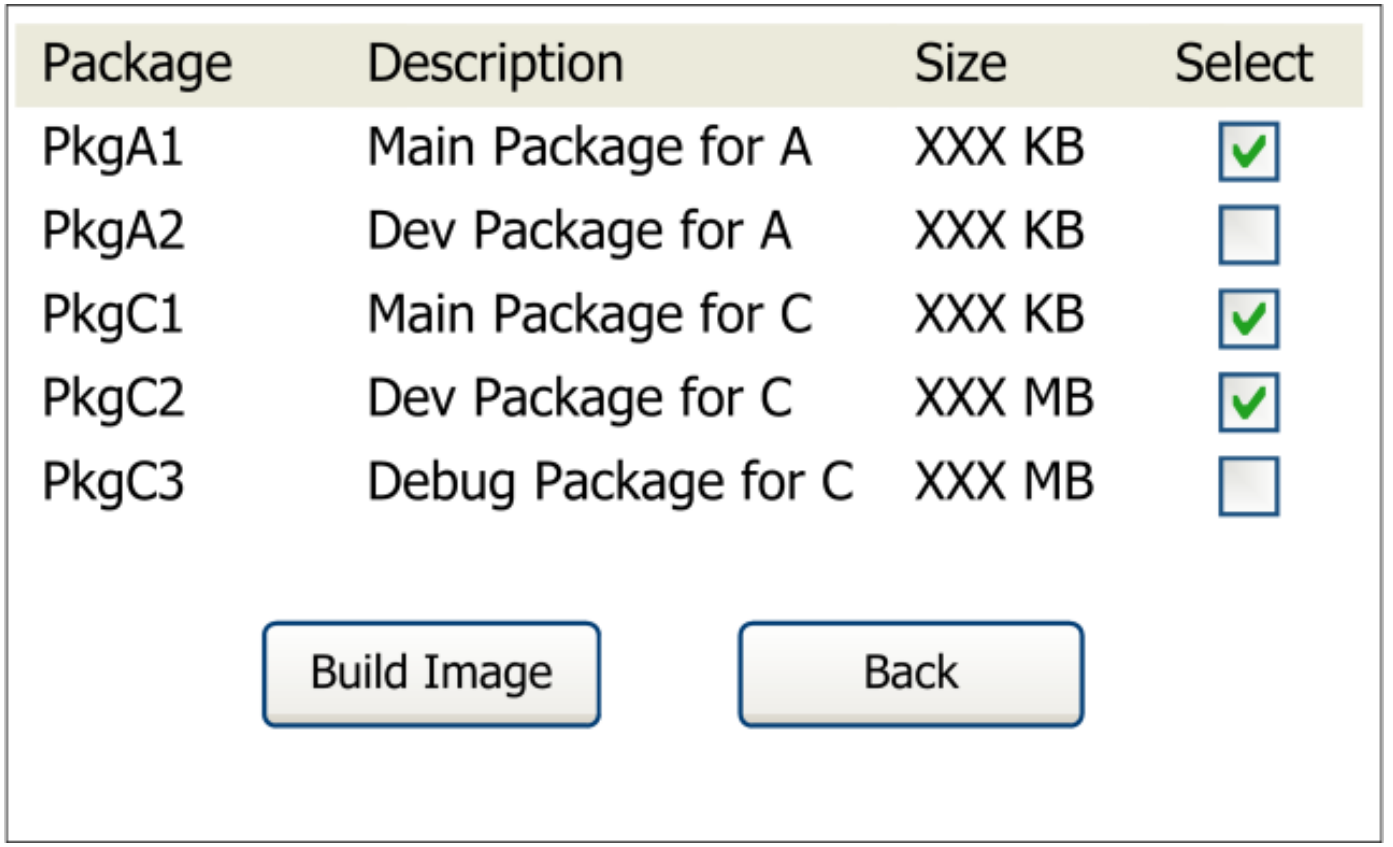


Stage 1: Build Package

Comment: provide recipe groups for user to select, e.g. group by the section field, or group by the metadata directory? TBD

Note: The UI is conceptual, and only for idea illustration. The actual UI will likely be different

Two Stage Build (Cont.)



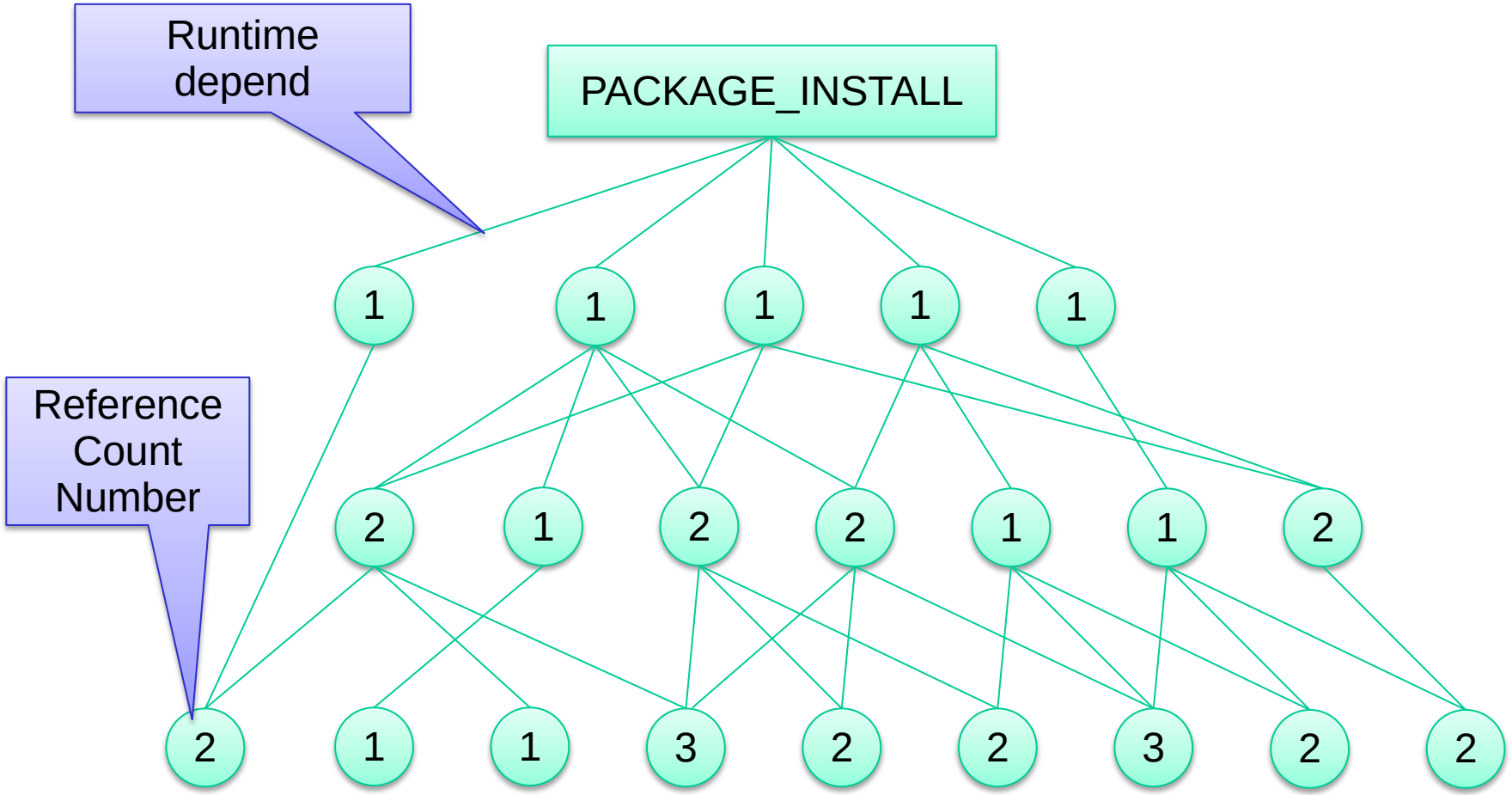
Stage 2: Build Image

Comment: before image creation, show the estimated image size. After image is done, show the image size, image path, etc.

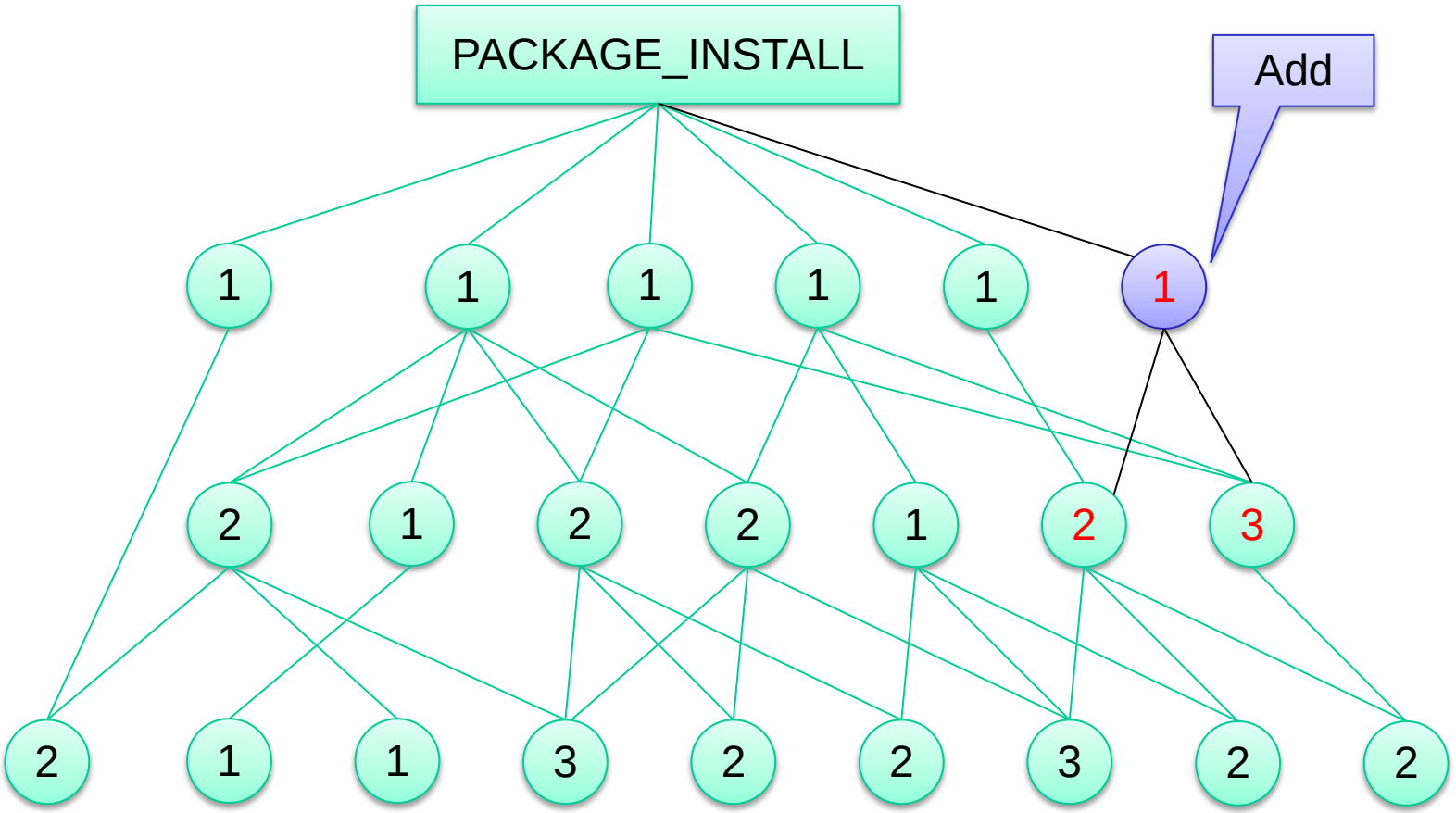
Req2: Package Removal

- Prerequisite: accurate runtime dependency
- Reference count based approach
  - Packages tree with runtime dependency relation
  - Each package has reference count of how many packages runtime depend on it
  - Package removal base on its reference count

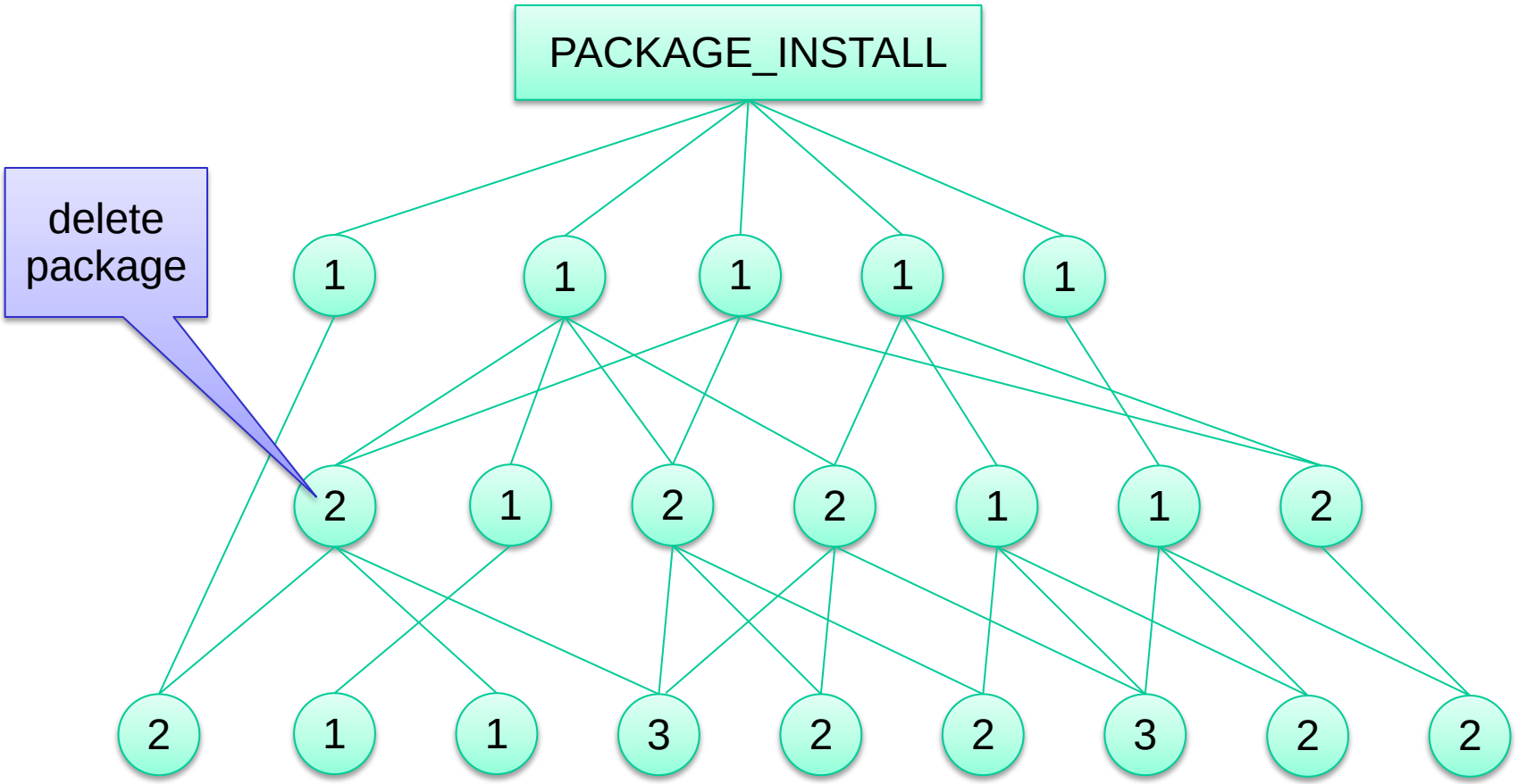
Package Tree



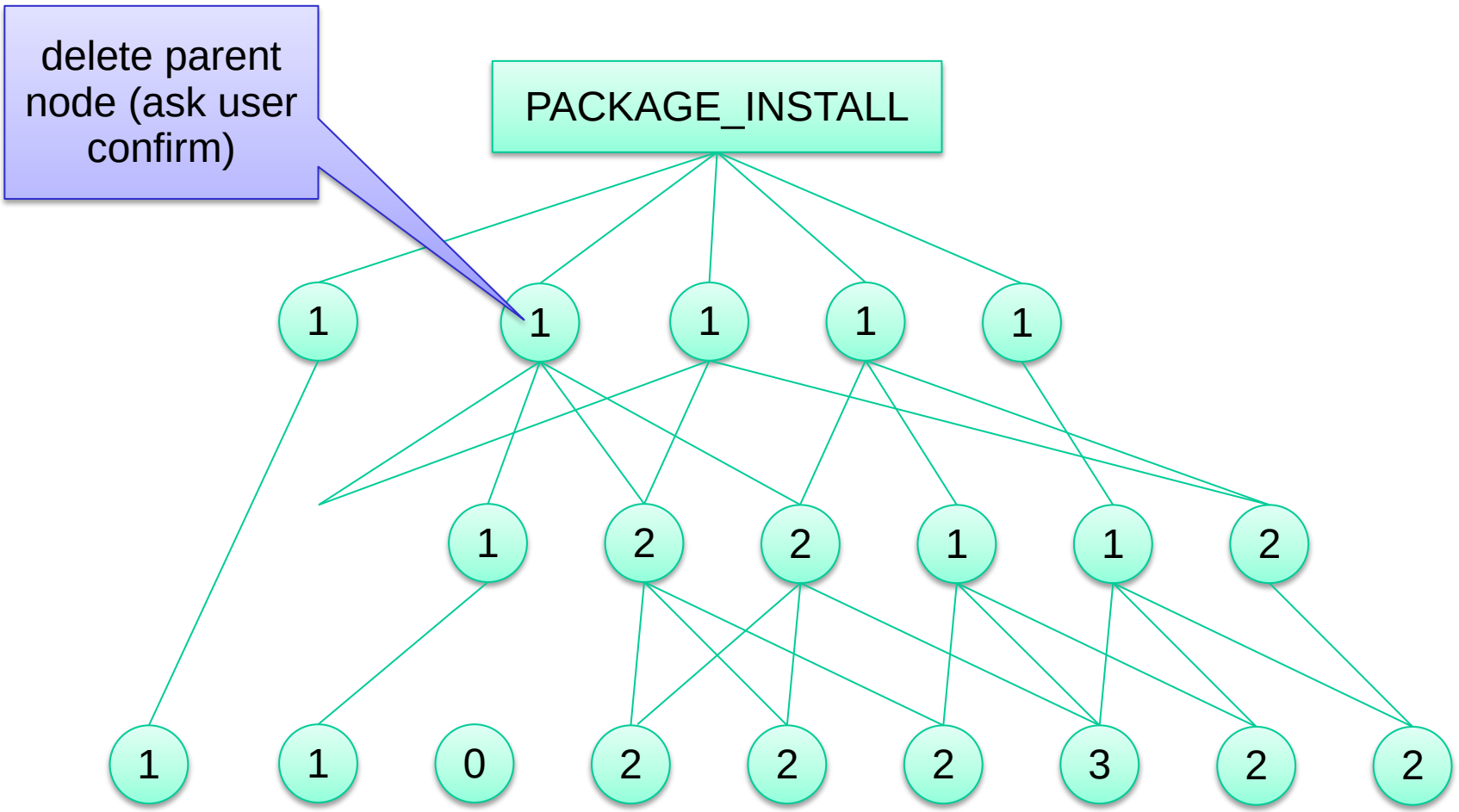
Package Add



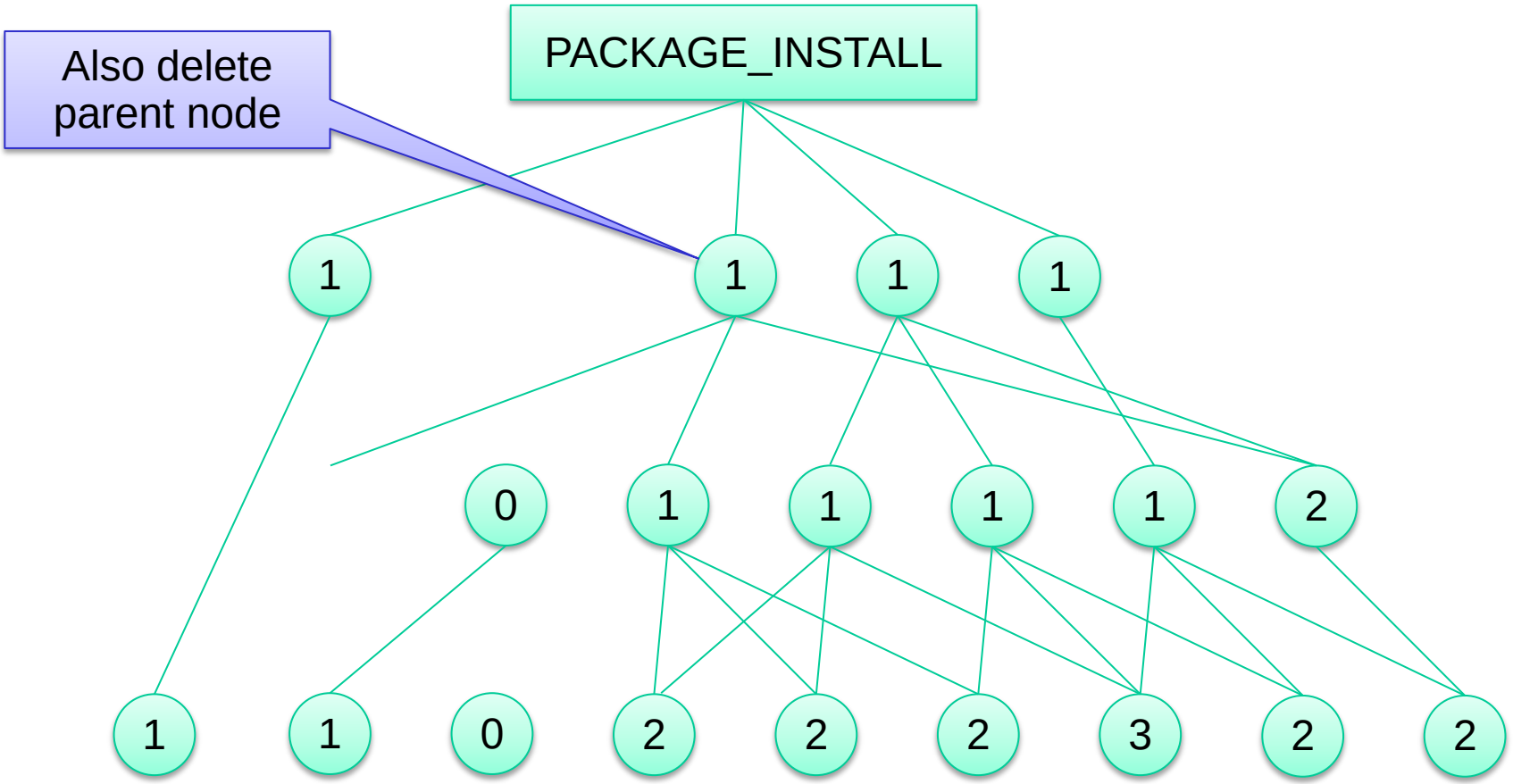
Package Delete



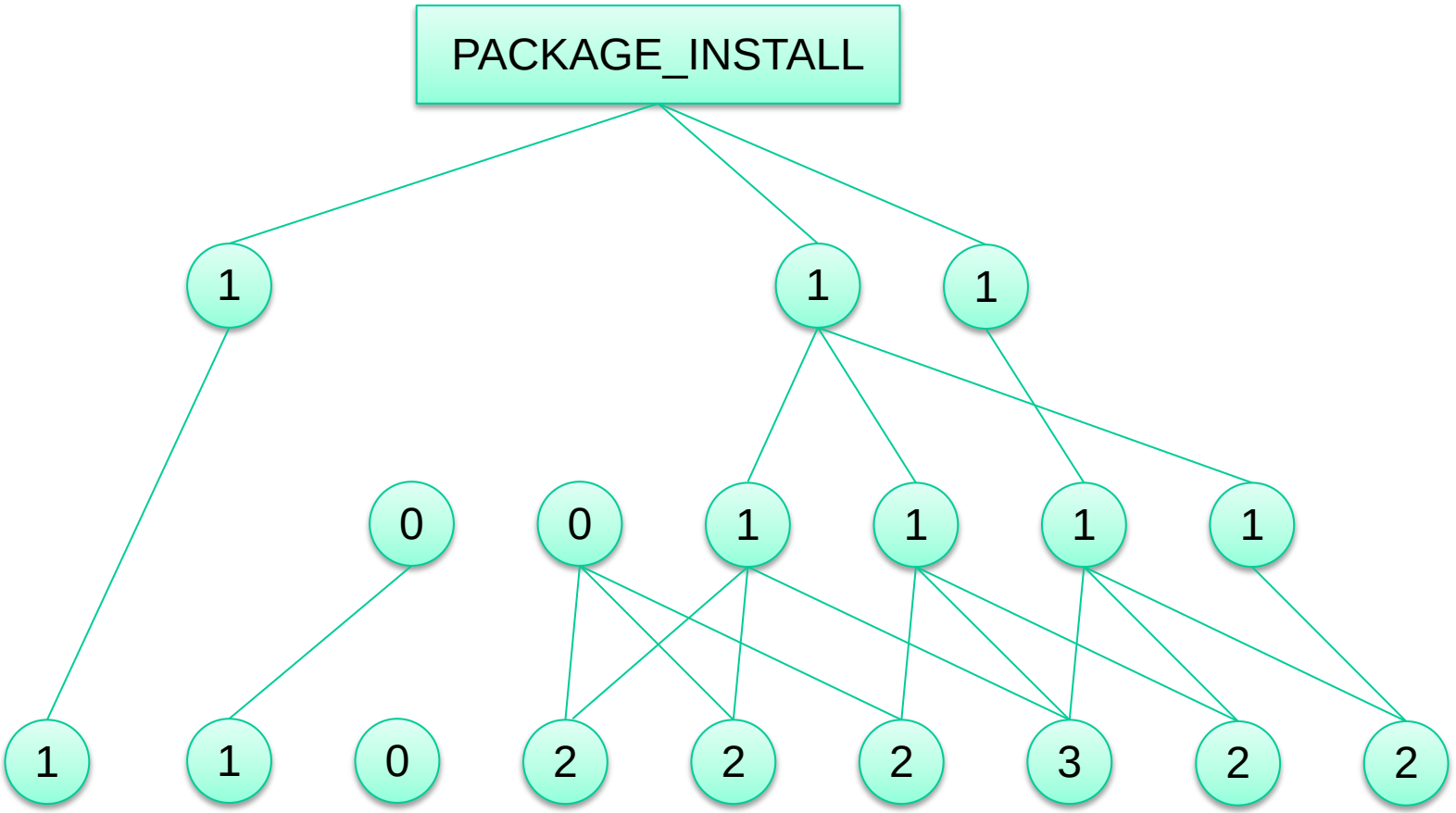
Package Delete (Cont.)



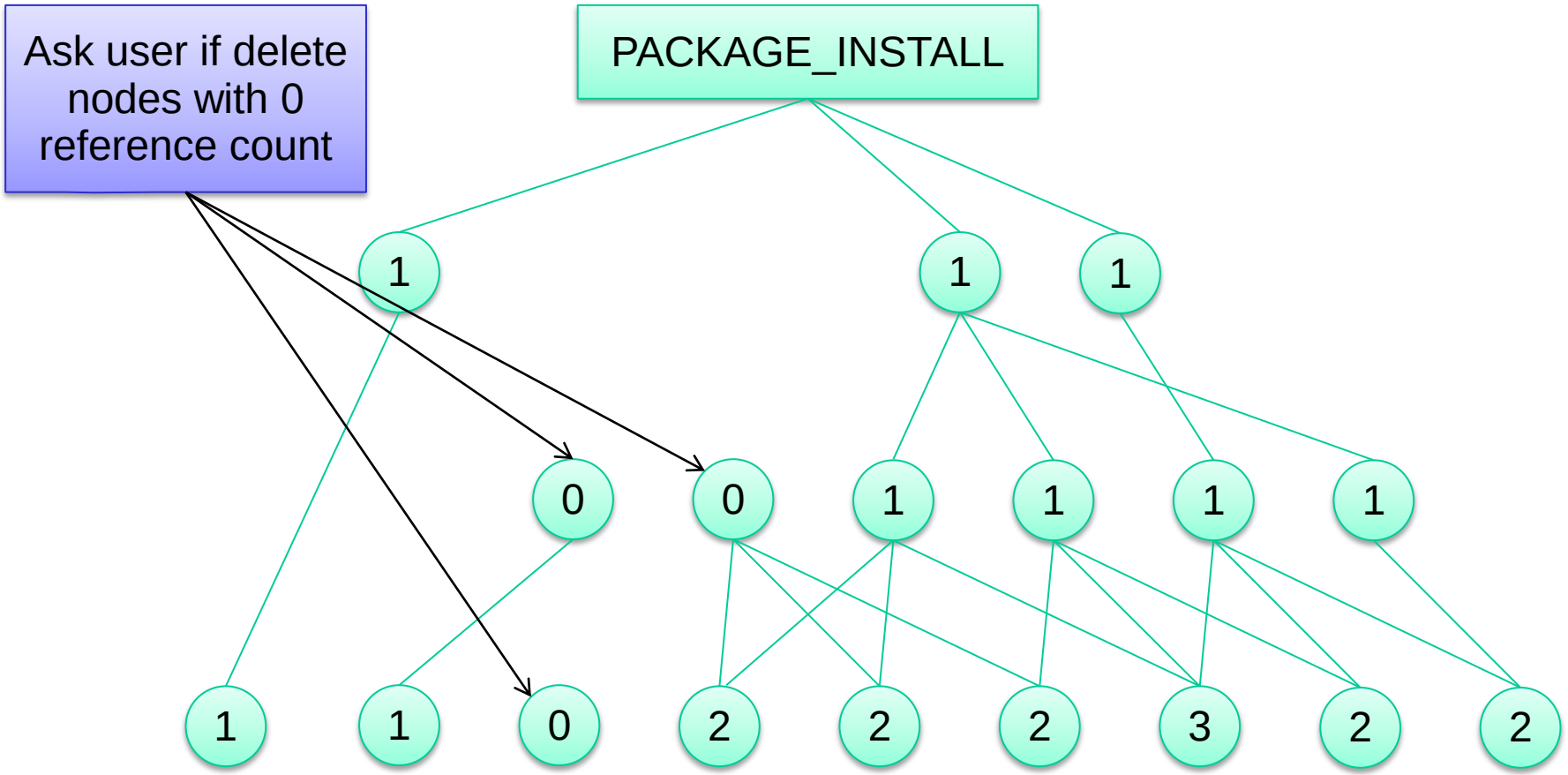
Package Delete (Cont.)



Package Delete (Cont.)

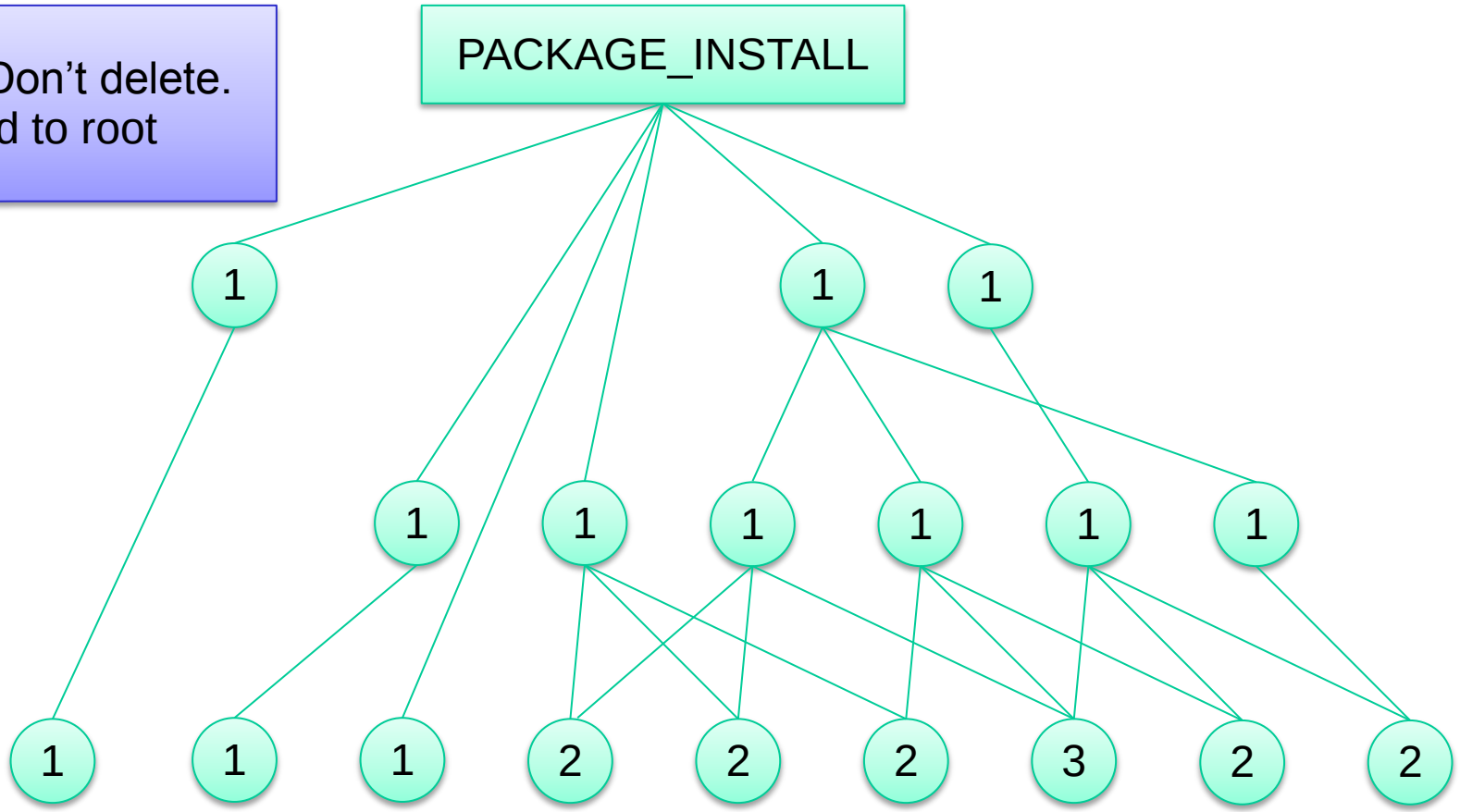


Package Delete (Cont.)

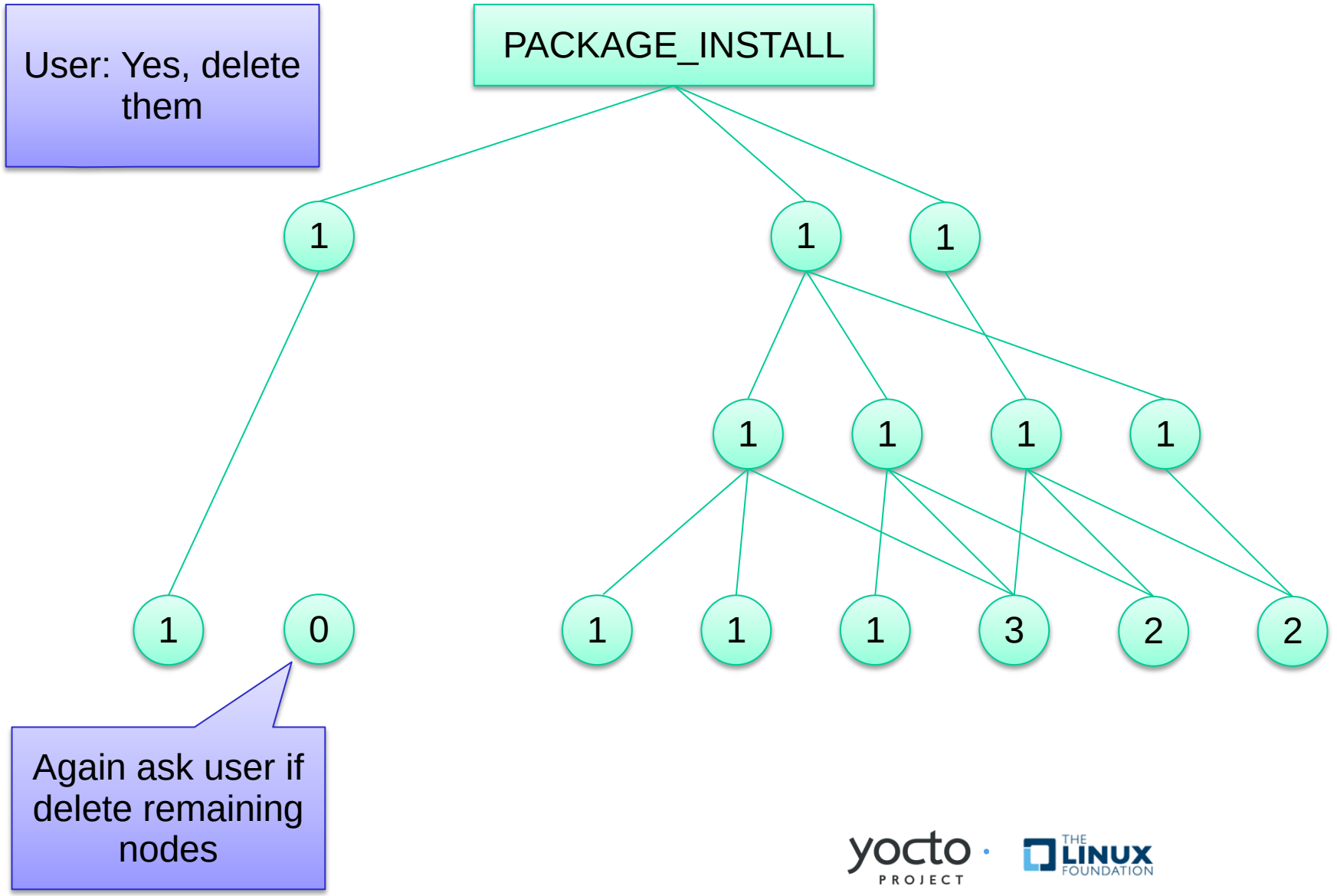


Package Delete (Cont.)

User: Don’t delete.
Add to root



Package Delete (Cont.)



Req3: UI Incremental Enhancement

- Progress Bar
  - use the task number to show the percentage of finished task
- Build Output
  - Show user the build log or error in friend way
- Add button of “run image in QEMU”

Req4: bitbake restructure

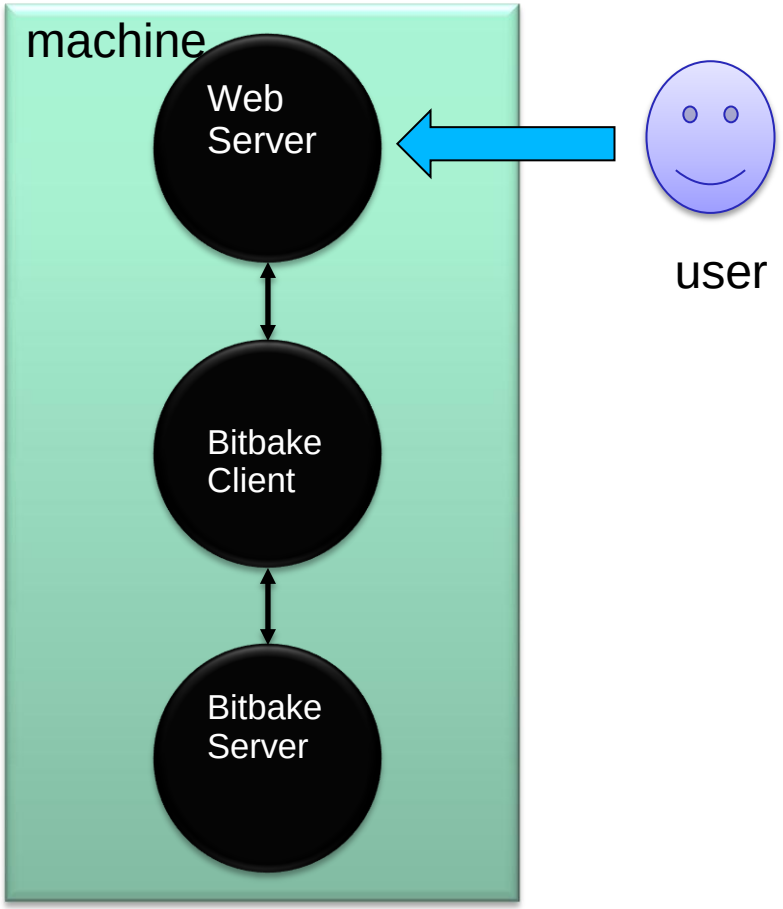
- Make bitbake client and server standalone
  - Currently, bitbake client/server model lack of some key features, e.g. user authentication
  - Not suitable for the usage model where client and server are in different machines
  - Restructure: add missing key feature for standalone client and server

Req4: bitbake restructure (Cont.)

- Modular reusable client/server API
  - Currently, the HOB client/server interface is HOB specific, not very easy to be reused by other frontend
  - Restructure: redefine interface and provide a well defined modular API
  - Analyzing the current UI's (knotty, hob, depexp, ADT plugin, etc) to createuse-case stories for the API

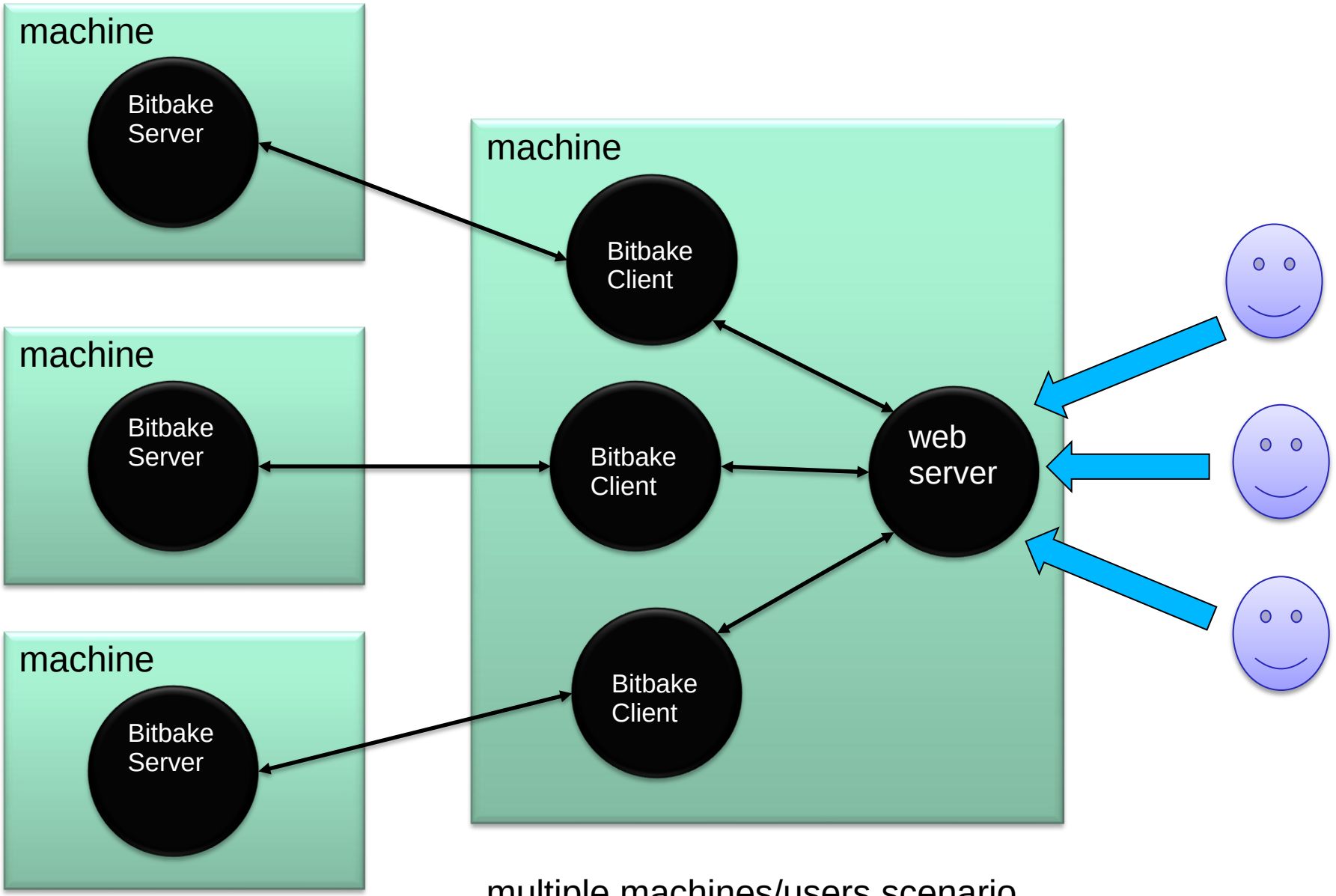
Req5: Web UI (Post 1.2)

- Web Server：
  - provide HTML frontend
  - create bitbake client
- Bitbake Client
  - forward command to bitbake server
  - forward result to web server
- Bitbake Server
  - response command



Single machine scenario

Req5: Web UI (Cont.)



multiple machines/users scenario

Backup

Related bug tracking

- Implement user-wide configuration file

  http://bugzilla.pokylinux.org/show_bug.cgi?id=1441

Image Deployment

- When image is ready, how to deploy it?
  - Via USB-Key, dd'ed to a HDD, copied as a ramfs to a SD Card, over the network somehow to a network boot (PXE)?
  - Ideally it should be done by another separate tool, and HOB call it upon completion
  - Also think about plugin architecture, and the deployment tool can be a HOB plug in.