The Yocto Project - Where We're Going and What's Next

Philip Balister
philip@opensdr.com

Open SDR

June 30, 2023

1. Introduction
2. Five Year Plan
3. Five Year Plan Details
   - Patchtest
   - Toaster
   - Core Workflow
   - Project Tooling
   - Meta-OpenEmbedded
   - Security
   - VSCode IDE Integration
   - Binary Distribution
   - Layer Setup
4. Call to Action

whoami

- Linux user since kernel 0.95a
- OpenEmbedded since 2006
- GNU Radio since 2008
- Yocto Project since 2010
- Industry experience
  - Brushless DC motors
  - Electronic PCB contract assembly
  - Steel mill automation
  - Graduate School
  - Embedded Linux for Software Defined Radio

Why was the Yocto Project Created

- By 2010, Embedded Linux was taking off
- OpenEmbedded use is skyrocketing
  - Developers are getting jobs
  - Users and developers have conflicting needs
- This leads to the creation of the Yocto Project
- The Yocto Project is the first Linux Foundation Collaborative Project
- Founding members include 8 companies and OpenEmbedded

What is the Yocto Project

- Linux Foundation’s first collaborative project
- Three membership levels: Silver, Gold and Platinum
- Thirty members and OpenEmbedded
- Member dues support the project’s operations
    - Documentation
    - Patch integration and testing for OpenEmbedded-Core
    - Infrastructure: autobuilder, git, etc
    - Bug tracking
    - Testing the Poky reference distribution on reference hardware
    - Advocacy and public outreach
    - Managing contractors
    - Developing new funding streams

Impact of the Yocto Project

- 18 of 24 stands at Embedded Open Source Summit (6/23) have some Yocto Project dependency

Five Year Plan Development

- Trying to answer the question, What should the project look like in five years?
- Identifying where the project lacks maintainers
- Ensure project sustainability
- Improve best practices
- Improve the project resilience to loss of maintainers and funding
- Attract new members to the Yocto Project

Five Year Plan Execution

- After creating the plan, we developed estimates for
  - Number of developers and time
  - Of course this leads to dollars
- Developers are in short supply, people working full time for companies
- Yocto Project uses member dues to fund some work already
  - Pays Richard Purdie, Yocto Project Architect
  - Paying developers test patches and report issues found on autobuilder
  - Paying a person to work on documentation
- Conclusion: Expand project membership to fund work

Patchtest - Background

- The Yocto Project workflow is based on patches sent via email
- Patchwork is used to manage patches sent to mailing lists
- Patchtest implemented on top of patchwork
  - Ran publicly for a while
  - Users found automatic feedback very helpful
- But, Patchwork had to be updated, this broke Patchtest
- No one had time to update Patchtest

Patchtest - Plans

- Update patchtest to work with new patchwork
- Run basic tests locally and give developers feedback before contributing
- This reduces maintainers workload by catching common issues
- Tests run may be updated based on developer feedback
- End goal, make it easier for people to submit patches

Toaster - Background

- Toaster is a web-based interface to bitbake
- Based on data collected by user interface designers
- Graphical interface benefits new users
- bitbake core-image-sato seems simple, but what is inside
- Also useful to experienced users to look into the build process

Toaster - Plans

- Ensure toaster works today
- Update automated tests to make sure it keeps working
- Increase user base
- Work with users to work out areas to improve capability

Core Workflow - Background

- Building everything from source does take time
- Many people are building the same configurations
- The Yocto Project does support sharing build output

Core Workflow - Plans

- Public shared state from the autobuilder exists
- For widespread use, a proper Content Delivery Network is required
- Some additional features and tooling needs developing

Project Tooling - Background

- Over the years several tools to automate common developer tasks
  - recipetool
  - devtool
- Other tools are fundamental, like pseudo
- Common theme, maintainers moved to other projects

Project Tooling - Plans

- Work on existing bug backlog
- Add support for modern languages with their own packaging systems (Rust, Go, etc)
- Add new features

Meta-Openembedded - Background

- In the beginning, OpenEmbedded Classic was one layer with all the recipes
- Maintainence challenges drove the creation of the Yocto Project
- OpenEmbedded-core reduced recipes to build basic systems that are testable
- Many of the remaining recipes ended up in meta-openembedded
- Much less testing and less focused maintainers
- But still widely used in real world systems!

Meta-Openembedded - Plans

- Add automated CVE checking, already in openembedded-core
- Enable Automatic Upgrade Helper
- Identify recipes that need ptests
- Mirror sources for LTS releases
- Identify obsolete recipes and remove them (and how to re-add)
- Improve reproducible builds

Security - Background

- Security is more than just running automated tools
- Yocto Project needs processes in place to handle security issues
- Tools to support security analysis of build artifacts, like SBOM
- No dedicated project personnel focused on security

Security - Plans

- Identify and document security process for the Yocto Project
- Implement additional security infrastructure and processes
- Create and staff a security team
- Support development of SPDX 3.0 draft standard

VSCode IDE Integration - Background

- All the cool people use IDE’s
- Potential to lower the perceived complexity using bitbake
- Integrate with build environment
- Support both application and recipe developers

VSCode IDE Integration - Plans

- Complete support for writing recipes
- Automated testing of vscode plugins
- Extend support to work with SDKs for application developers
- Support testing and debugging on qemu and target hardware
- Review what the user base requires

Binary Distribution - Background

- Yocto Project creates tools to build source based distributions
- In many cases we need can build multiple images from the same build
- Shared state has the binaries, but isn't as convenient as standard package formats
- Once open a time there was a web based tool that created images for hardware

Binary Distribution - Plans

- Creating deployable images quickly helps people prototype
- This is harder than it looks
- What features should the binary distribution include?
- What machines should be supported?
- Should it produce containers?

Layer Setup - Background

- OpenEmbedded Classic solved this by having all recipes in one layer
- Minimal useful system has openembedded-core and a BSP
- Typical systems add 2-5 more layers
- Many existing solutions, repo, combo layer, git submodules, bitbake feature
- If you speak to 3 developers you will get 6 solutions

Layer Setup - Plans

- This is the hardest problem presented today
- Many use cases, CI configurations, developer setups, end users ...
- Track layer revisions and build configuration
- Challenging people problem!

Call to Action - Companies

- Join the Yocto Project
- Dedicate resources to support the project
- Review the RFQ for five year plan tasks and bid on them

Call to Action - Individuals

- Talk to your employer about joining the Yocto Project
- Spend some spare time contributing to the project
- Review the RFQ for five year plan tasks and bid on them

Questions?

BOF (Birds of a Feather flock together) South Hall 3B at 1450

Birds of a feather flock together is an English proverb. The meaning is that beings (typically humans) of similar type, interest, personality, character, or other distinctive attribute tend to mutually associate.