Monthly Cycle Overview

Alexander & Joey

Monthly Releases

How and When?

Why

- We believe we need to
  - move faster to service our partners and their non-Linaro commercial timelines
  - reduce the impact that the planning cycle has on engineering/execution
  - ensure blueprints are thought through and "ready to code" vs spending too much time doing "investigation"
  - produce rapid protoypes that can be used immediately
  - move to a continuous integration model with monthly milestone

The Schedule

Month	Finalizing	Planning	Release Date	Release
1	Month 2	Month 3	2011-06-30	11.06
2	Month 3	Month 4	2011-07-28	11.07
3	Month 4	Month 5	2011-08-25	11.08
4	Month 5	Month 6	2011-09-29	11.09
5	Month 6	Release & UDS	2011-10-27	11.10
6	Release	Months 1 & 2, Public Plan Reviews	2011-11-10	11.11

Component Versions & Milestones

- component versions and milestones should follow best practices from toolchain WG:
  - Milestone: $component-$version-$yyyy-mm
  - Version: $milestone-$buildcount
- Example:
  - Milestone:
    - gcc-linaro-4.6-2011-05
    - linux-linaro-2.6.38-2011-05
  - Version:
    - gcc-linaro-4.6-2011-05-2 (-> respin1)

Component Milestone Models

How do you release components?

- Release and Respin (Toolchain WG Model)
  - release 2 weeks before all-linaro release
  - keep milestone open for critical bug targeting
  - Platform integrates toolchain for all-linaro release
  - respin component for critical bug fixes

- Freeze and Release (Kernel WG Model)
  - milestone date is 1 week before all-linaro release
  - kernel tree enters freeze and delivers RC tarball
  - fixes any integration bugs found
  - only releases as part of the all-linaro release

Component Releases

- **Goal**: release components and LEBs in sync: last Thursday of the month
- Integration of components by Linaro Release Team with sufficient lead time (i.e. allow "baking")
  - Option 1: Release & Respin
  - Option 2: Freeze and Release
- Components use their: a) individual lead time agreed upon with Release Team, b) component milestone to schedule
  - Example 1: linux-linaro freezes 1 week before release
  - Example 2: gcc gets released 2 weeks before release
- Blueprints that deliver into a component get targeted against the component milestone

Component Delivery

- Delivery to Release Team happens as tarball at the component milestone date
- version of the tarball is $milestonename-1
    - gcc-linaro-4.5-2011.05-1
- integration teams integrate and validate delivered component
    - work is tracked in an integration blueprint owned by integration team
- bugs found during integration get escalated through release team and get targeted against the same component milestone
- once bugs get fixed, a new tarball is made and uploaded with the version $milestonename-2
    - gcc-linaro-4.5-2011.05-2

Component Delivery in LP

- All Blueprints implemented for this milestone must be targeted against the appropriate component milestone and status must be set to "Implemented"
- All Bugs fixed in the codebase of the released component milestone must be filed against the component project with the component milestone set
- Headline summary for delivered blueprints will be put into the Launchpad "Release Notes" field for each release
- ChangeLog of the component will be put into the launchpad milestone ChangeLog Field
- See the inspirational example

Operational Planning

And Blueprints!

Steering and Planning

- quarterly TSC steering meetings add, remove, re-prioritize TRs
    - TRs state steering directions that can be used to track a pipeline of deliverables that support that direction (backlog)
- quarterly Engineering sprints fill the backlog with prioritized blueprints (aka user-stories/building-blocks/deliverables)
- changes to the priority of backlog items can be submitted by TSC, Stakeholders, and Management to Engineering and they will be considered for future milestones.

Monthly Execution

- tech-lead executes the fine grained engineering planning in alignment with management and TSC
    - tech-lead approves and assigns a blueprint for drafting to bring it to a ready-to-code state
    - tech-lead approves implementation of ready-to-code blueprints
    - tech-lead accommodates change requests from TSC and stakeholder monthly
- as a rule of thumb an engineer should have assigned any point in time one blueprint for drafting and one blueprint for implementation

Benefits for Engineers and Leads

- A shorter planning horizon will help leads and engineers to stay focused on delivery
- less planning waste as work can hardly be planned out accurately for 6 month
- more flexibility to accommodate changes to priority and unexpected incoming work
- WG achievements can be better communicated by releasing monthly deliverables with real headlines
- Measurement of success shifts from counting work items to reviewing deliverables
- supports thinking in building blocks rather than long work item lists and learn how to deliver

What Now?

- Don't Panic
  - If you are all set for 11.11 you can continue as usual or ...
- Try Agile
  - Take a moment to think about your deliverables
  - Can your blueprint be split up in monthly, tangible deliverables?
  - make blueprints out of deliverables for which you know work items
  - add backlog entries for those that you don't know enough details yet

How to use blueprints traditionally

- Transform the TRs as good as possible in reasonable projects/blueprints up-front
- Plan 6 month work items; split work items up at milestones as checkpoint
  - To get extra cookies think whether the work item split can be done that it reaches an distributable, intermediate step and phrase a “micro” headline out of it
- give the checkpoint a one line summary by putting a "Headline: ..." in the work item block
- Try to release as soon as possible; always have something releasable!!

How to use blueprints the monthly agile way

- Maintain a backlog in the TR
  - Backlog items are building blocks towards delivering a TR
    - Building Blocks (aka User Stories)
    - Also a WI block with headline will do its job
- Multiple options to manage backlog
  - Ideas and Future blocks are listed in TR in priority order
  - drafting and ready to code blueprints are approved for a series
  - active blueprints are targeted against this or next month milestone

TSC Steering

How the TSC Steers Linaro

Agile Transition

- We are proposing an Agile transition
- We will
  - disconnect the planning and execution cycles
  - drop major releases like 11.11, 12.05, etc.
  - officially release monthly. Period.
  - treat TRs as backlog that get filled and reviewed at LDS
  - LDS happens four times a year
  - TSC happens four times a year
  - do further backlog planning during our team sprints
- To accomplish this, we'll need to modify how the TSC measures Linaro's performance.

Why the Change?

- Problem scope too big and unstable on a 6 monthly Linaro horizon
- Thinking in work items rather than building blocks does not yield concrete deliverables
- Lack of concrete deliverables makes it hard to interpret and communicate the value of engineering output and hard to claim either success or failure
- TSC Priorities and TRs are not broken down in clear deliverables rather long work items lists against a moving target
    - Due to changing technology and business priorities, TRs not immediately executed upon will go bad/spoil.
- Ad-hoc projects automatically eat into work item delivery of a part

Benefits for the TSC

- Allows you to continuously monitor concrete delivery of small and manageable building blocks for each TR instead of progress in a fluid list
- Allows you to continuously review and adjust priority of upcoming deliverables in the pipeline (backlog)
- Provides greater visibility into engineering activities by eliminating the detailed engineering work items from the status.linaro.org main dashboard replacing them with managable building blocks
- Overall, this change provides you with more transparency, more control on the engineering priorities and better means to communicate progress done in and outside of member organizations

TSC Responsibilities

- Steer Linaro
  - own the steering cycle
  - 3 month cadence
  - add, remove, reprioritize TRs
  - review backlog associated with TRs for feasibility and initial priority
  - review backlog state monthly and communicate changes identified to priority and direction
- NEW: Get opportunistically involved in Planning/Execution Cycle
  - monthly review of priority of items in the backlog
  - boost pet projects that are in the backlog

Engineering Responsibilities

- Operational Management
  - continuous backlog management
  - engineering allocation management
- Technical Execution
  - Filling backlog items with concrete engineering solutions (aka Work Items)
  - Implement and Release Deliverables from backlog
- Technical Cross Vendor Collaboration
  - advancing technical challenges to the TSC for a decision/statement of intent
  - facilitate community to solve difficult technical challenges (e.g. Power Management, Multimedia)

Measuring Performance

- We stop measuring performance based on work item completion
    - WIs are purely operational and service engineering management to monitor progress of work
    - WIs have an unknown impact on the delivery/goal; some will cause a delivery to fail; others are OK to postpone/drop. This becomes a management nightmare.
- We should measure Linaro on performance against deliverables (explicitly called out as BPs) instead of WIs.
    - one blueprint represents one deliverable

Terminology Changes

- We should use term "steering" and not "planning" for the TSC direction setting activities
  - TSC direction setting is done via managing TRs
- Pending deliverables/pipeline for a TR is called "the backlog"
- TRs will contain (in the whiteboard) a list of "stories" which document what a particular user, in a particular role, will be able to see/perform.
  - Those stories will become engineering blueprints with WIs.

FAQs

Answers to all your questions!

Connect vs LDS

Q: How will the changes to our cycle change the way we run our summits? Should Connect and Linaro@UDS be different, or run exactly the same, with a public schedule and session shuffling?

A: In general the two will be run the same. Connect will not have as many public people participating. LDS will have a lot of public people present. We expect both planning and hacking to occur at both of these meetings.

What if I miss a Milestone?

Q: If blueprints are meant to fit into a month, and I am working on a work item that was targeted against July but will only really be done in August, should I move the work item to a separate month?

Q: What if I miss a Milestone?

- If you have a task scheduled for a milestone and you miss it
  - Big Miss or Big Mess? Rethink the deliverable; split up in more blueprints and send back into backlog
  - Small Miss? Retarget the item to the following milestone

Will you be blocked on IS?

- Will need to put Due Dates in RT tickets that are needed for upcoming monthly releases and track them
- TLs will need to be proactive in getting this into the RT system as early as possible and monitoring the status
- Implementation and Deployment should be separate Blueprints. This allows you to complete the implementation and only the deployment would/could be blocked on an RT.

Building Blocks vs User Stories

Q: "Building Blocks" is to vague. Use "User Stories" instead.

A: Its intentionally vague as we don't know if user stories will apply everywhere. What we care about from release/steering point of view is just that we release something tangible rather than just a bunch of "meaningless" work items.

User Stories on the Whiteboard

Q: User stories on whiteboards may not work. It might be better to have user stories as BPs and then have the TR dependent upon the BPs.

A: Creating blueprints up front creates lots of "blueprint-noise"; also hard to define order of backlog/TR

2 Month Pre-Planning

Q: 2 month in advance planning is crossing the line of what's agile.

A: This early planning could really be helpful to think through the problems. It might yield better results so we should try it and see how it goes. We also mean that you don't draft things that are further ahead than 2 month. Obviously you can always draft something for next or even this month if it came in late and needs to get done ASAP.

Blueprint Dependencies Part 1

We should mark TRs in LP as dependent upon Engineering BPs. Once all the engineering BPs are done the TR will have no unmet dependencies and therefore we can mark it done. This may not always play out but it's a good practice to show the interdependencies between everything.

Blueprint Dependencies Part 2

Q: If you don't complete all the engineering BPs to complete a TR, what happens?

A: It doesn't matter if we fully complete a TR or not. What matters is that we accomplish the essence of the TR via completed deliverables/engineering BPs. The TSC looks at the backlog of a TR and the "completed" log and decides whether its worth continuing it; maybe under different overall priority etc.. Reasons for killing a TR might be that the TSC is happy enough with what was achieved, or the TR was deemed to be not worth further investiment due to changed priorities/environment. etc..

Monthly Milestone BPs

Q: Could we create a monthly milestone BP with high level deliverables listed like this?

A: Blueprints that contain Multi-team deliverables like we do in Android are a special best practice for how integration teams can manage their integration delivery commitments from supplying teams. So yes, you can but you should pilot it first to ensure it works well for your team.

Setting Engineering Priorities

Q: How to handle setting priorities for the engineering BPs?

A: we don't use BP priorities to manage the backlog order; this is done through ordering the backlog; priority of BPs are only decided when implementation is approved

A: This is decided during backlog review when the BP is approved for implementation. We want to strive to do first:

- high priority items
- those which are blockers to other BPs/WIs (i.e. dependencies)
- those which block other teams

My Blueprint is longer than a Month

Q: I'm working on a blueprint that doesn't quite fit into a month. What milestone do I target the blueprint to? Should I split the blueprint up further, or can blueprints span months?

A: Your blueprint is too big. Take a moment, step back, and think about what you are trying to accomplish and try to break up the blueprint into smaller pieces (aka "chunking"). Try to identify what tangible you can deliver in first month and focus on getting started on that.

Spreadsheeting TR Deliverables

A suggestion is to use a planning format like this which organizes TRs, Build Blocks, Stories, and potentially even work items.

Internal Team Coordination

Q: With the move to monthly milestones, the pace seems to have quickened in my team. How do I keep track of everything that's going on?
A: Daily Stand-up meetings are beneficial. If they seem a bit much for you, your team could sync up once a week to make sure the milestones are going to be hit (already doing this) and that everyone is going to produce something for the milestone.

Doing Non-TR Work

Q: How do we call out and justify working on non-TR activities such as fixing upstream patches, updating infrastructure, updating software libs, Fixing technical Debt., etc.

A: There seems to be a need to have at least one backlog for not-TR induced work - though most of things probably can always justified by one of our TRs. This is dependent on the team; for WGs it probably should rarely be the case they work on non-TR things; Platform Teams and LTs have other stakeholders like management and LTs that may drive some non-TR work. If you need to do non-TR work you'll want to say that 2/3 to 3/4s of the time per month is spent on TRs and the remainder is for "other" but important work.

Longer Term View

Q: Focusing on monthly deliveries biases us towards small, tactical features; how do we make sure we're still delivering the long-term solution?

A: The TSC via TRs define a mid to long term direction so they can also act as a roadmap.

TSC Reviews

Q: How does this affect requirements gathering from the TSC? Should we expect the TSC to only give us input every six months and collect deliverables monthly, or should we ask them for input more frequently?

A: The TSC reviews the backlog and status monthly. During this time they can also communicate updates to us. They provide us with new/update high level TRs four times a year. This includes reprioritizion of existing TRs and workload.

Component Integration Problems?

Q: I have a component that has an interdependency with another team's component. How do I ensure it doesn't break in Integration?

A: To ensure that integration goes smooth, component developers are encouraged to incorporate techniques in their flow that would discover integration issues early on, e.g.

  - qemu developer could monitor linux-linaro + 1 kernel version as linux-linaro regularly pulls in arm/ stuff from the next mainline kernel
  - kernel WG developers could test their features in qemu before landing and raise the issue with qemu developers early on

For Longer Term:

- Push to a continuous integration loop

Changes to status.l.org

Q: How will all this be visualized on status.l.org
A: Discussion still ongoing, but:

- focus might shift form 6 monthly work item burn down to monthly deliverables
  - short term view: what is coming this month, what is planned next month; what else is in backlog?
  - long term view: what was finished for TR X? what is currently worked on for TR X? What is next in the backlog?
- all views will be available broken down by Team, by TR and by deliverable
  - what is coming this month from team X, what is getting implemented for component Y, what will be get done for TR Z

End of Presentation

Thank you!

Improvement Ideas

Things to think about

Comments & Actions

- How frequently should we do Public Plan Reviews if we are moving to a complete monthly delivery cycle?
    - Not yet clear. Discussing at Exec Management level
- Please create a wiki page that contains
    - wg releases on this day
    - platform releases on this day
    - milestones are called X
    - series are called Y
    - best practice for planning/blueprints syntax/managing backlog etc.
    - e.g. Toolchain
    - etc
        - assigned Fathi & Andy (with asac & Joey)