R2013a Lessons Learned

Table Contents

- Table Contents
- How Lessons Learned are Managed
- What is a Lesson Learned
- What is Not a Lesson Learned
- Things We Should Keep Doing
- Things We Should Change
    - Do Better
        - Build-Test System Reliability/Performance Puts Daily build cycle and Release cycle at Risk
        - Ticket/Schedule System Process are not well defined and those that are not followed
        - Release Process
        - Miscellaneous
    - Start Doing
    - Stop Doing

How Lessons Learned are Managed

GMAT lessons learned include things that we did well and should keep doing, and large scale things we should be doing to improve the software or our project.   Lessons learned are each discussed by the team and if we decide there is a real issue, we require a plan for improvement.   To make sure we are efficiently handling lessons learned, here are some high level guidelines for creating them.

What is a Lesson Learned

Lessons learned are issues that cause significant problems or could have caused significant problems, or are issues where we did something that significantly improved the software or our project.   Lessons learned require group discussion and probably a change in team habits, process or strategy.

Lessons learned satisfy one the following criteria:

- Issue that is putting the project at greater risk than necessary
- Issue that is causing significant inefficiency
- Issue that is significantly lowering quality

What is Not a Lesson Learned

A lesson learned is not a minor annoyance, a tweak to an existing process, or something that can be resolved between team members in the everyday process of getting work done. Team members should bring these types of issues up at meetings, or work them among the team members involved.

A minor issue, (i.e. not a lessons learned), satisfies one of these criteria:

- Tweak to an existing process
- Minor annoyance or gripe
- Can be resolved by just picking up the phone, or discussing via email, or weekly meeting
- Does not require significant change in habits or processes

Things We Should Keep Doing

- SPH: Keep working so well together!
- SPH: Broad regression tests
- SPH: Process: Ticket -> Specs -> Dev -> Test -> User Docs
- SPH: Group/Feature area leads
- SPH: Focus on quality
- SPH: CCB prioritization of work items
- JJKP: Defined process

Things We Should Change

Do Better

Build-Test System Reliability/Performance Puts Daily build cycle and Release cycle at Risk

- SPH: Improve test system reliability. Test systems should fail less than once a month.
    - SPH: Get two machines set up to run the tests
    - SPH Schedule maintenance work in advance, if possible
    - SPH Start system early enough to be monitored
    - SPH: Machine should be located on-center, so VPN is not needed
    - JJKP: Something keeps causing the script test system to crash when loading results. If this continues, we may need to investigate changing how the reporting code works. (This might be a good idea anyway, it takes a LONG time).
    - Bottom line: SCALABILITY. We have almost 11k tests, and both the filtering mechanism and the reporting mechanism aren't behaving well with this size.
    - SPH: We should consider farming out GUI tests across more machines to reduce the time to get results. Currently it takes about 5 days to know the state of a build. 3 to run and a 1-2 to analyze. This makes RC cycles take a long time.
    - SPH: We were not running certain types of regression tests!!! This can't happen!!
    - SPH: Put in waiver to avoid overnight reboot of machines
- Build system needs to be on a machine more people can access.

Ticket/Schedule System Process are not well defined and those that are not followed

- SPH: Critical path issues in ticket system put us at risk and we got lucky
    - Lots of tickets waiting for clarification or analysis until last month of 10 month release cycle. We got lucky here for the most part. At least one bug just coulnd't be fixed because of lack of time to solve the problem.
    - Lot's of work for SPH and JJKP tracked outside of ticket system. The result is that P2 items in those lists get attention before P1 release items. For example, we sent a lot of time on SBIR support, but didn't start reviewing end user docs until 3 days before scheduled app freeze even though over half the doc was ready weeks before.
- DJC/SPH: Some ticket quality problems happen too often.
    - When submitting a bug, include scripts or detailed list of procedures, and error messages.
    - When partially closing issues: Be sure to include text explaining the current status of the issue so that when a developer returns to it, it is clear where things were left. (See for example GMT-2686. I should have noted where things were left, but didn't.)
    - f you have a script that shows the issue, always include it even if it seems trivial.
    - Avoid "whack a mole" issues, where the developer will fix the presented issue only to see it reopened with a description of "this one is just like the original issue." Reopening once is okay, but if the issue shows signs of being systemic, then change the description to clue the developer in that it is likely systemic, or open a new issue.
- SPH: Ticket system has so many trivial tickets, important issues get lost in the weeds. Need a process to streamline ticket system (keep a clean backlog related to goals), define what is and is not a ticket, and how the backlog is a streamlined list of necessary work. CCB potentially wastes valuable senior team member time reviewing trivial items, early ideas/suggestions.

Release Process

- JJKP: Some Visual Freeze things don't actually need to be in visual freeze.
    - Updating destination addresses in GMAT.ini
    - Updating TestComplete link tests in GMAT.ini
    - Note that if new buttons/links need to be *added*, these do become Viz Freeze items.
    - Branching strategy improvement to avoid commit issues for people working on other releases

Miscellaneous

Start Doing

- SPH: Leverage code from heritage systems
- SPH: Consider using more third party libraries like SPICE or Asset Importer to avoid duplication of solved problems
- SPH: Be better at directly responding to GSFC flight projects.. help missions.
- JJKP: Do better at PR, outreach, etc.
    - Better outreach of current activities (blog posts, announcements, etc.)
    - Presenting at conferences, post sessions, etc.
    - Swag: coffee mugs, T-shirts, etc.
    - Outreach to universities, organizations, centers

    - Colloquia, seminars, talks
    - Outreach to HQ/missions

Stop Doing

- SPH: Spreading ourselves too thin. Don't do more projects and improvements than we can do well.