# Initial Email:

* How does work get into the product backlog?
  + A product backlog item (PBI) is defined as something that delivered business value to production and extends the abilities of a product
  + We only want to put such items in a backlog.
  + Defects are the only exception. The teams that create the defects should fix them.
  + All initiatives get prioritized against each other before teams are given PBIs to work on.
  + Backlogs could not use the SRC ticketing system but arrive directly from Program Analysts adding new Strategic Themes and Portfolio Epics in the proper backlog.
* If the teams must perform other kinds of maintenance/support work (updating databases with new data, refreshing some service, rebooting some service, etc.) how will they get that work?
  + Maintenance work should not go into the team’s backlog. It doesn’t meet the PBI criteria/definition.
  + Should groups use a ticketing process to submit those items?
  + Should new maintenance and support tickets be prioritized (in something like a KANBAN board)?
  + If a team’s members must participate in being support teams beyond tackling the defects they create, then their sprinting productivity would be proactively be reduced by the normal 60 hours per sprint so they can focus on pulling high priority items from the ticketing KANBAN board to solve. (Someone would need to decide how much to systemically reduce their 60 hours to set the consistent expectation: you will be allocated 50 hours to work on sprint items and spend 10 hours on KANBAN support items every two weeks).

Discussion Notes below.

## How does work get into the product backlog?

* New initiatives
  + Definition:
    - Initiatives to add functionality to products and services through development often requested by the business
      * Example: Updates to a business process to benefit service workflows or client value
    - Initiatives to upgrade services often through software/system upgrades often requested by IT but for the benefit of the business including dependency coordination on effort and timing (technical enabler)
      * Example: Upgrading a database server software (MYSQL, Oracle, SQLServer, etc) to increase stability and performance of business solutions
  + Straight into backlogs via intake
  + Sprinting failed tests (“Bugs”)
    - If a piece of work is tested by another team, and it fails, that “bug” is returned to the team that created it and is currently building the Feature
      * Example: Integration
        + Bugs become “stories” within an open Feature
        + “Features” do not close until they are integrated and ready to go to production
      * Example: Bugs promoted to Defects for post-deployment
        + If a work-around might exist, then it is promoted from a “bug” to a “defect” to be fixed later (value decision based on impact and other factors)
  + Enablers initiatives:
* New Defects
  + Definition: *Functionality that has escaped (or bugs promoted to defects intentionally) to production*
  + Routed to the team **who “created them” or “is responsible” for an area (one of the team(s) handling that capability)**
    - Show up right away or much later
  + Defect Triage
    - “Found” defects are turned into “tickets” in IT Ops for initial triage
      * Example – found defect: A user submits a defect and IT Ops would help figure out who will resolve the problem
        + This gets turned into a Rally Defect PBI in the Team backlog and is prioritized in Sprinting along with Sprinting hours (comes out of Sprinting hours and not maintenance)
        + Real production defects **do not go into Maintenance** backlogs.

Examples:

If a defect is discovered in production and needs to be solved immediately, then it gets prioritized OVER some other work already in the sprint (product work)

If a defect can wait, it is prioritizing WITH other enhancements within a future sprint (again, product work and not maintenance)

* + - * Example – confused expectation: A user submits a “defect” for functionality that *is actually performing “as designed”* (not really a defect but requires an enhancement) but they need/expect it to work another way, IT Ops communicates the misclassification and closes the non-defect.
      * Example – small enhancement (should not stay in maintenance):
        + They might re-submit what they previously classified as a “defect” but now as an enhancement
        + sometimes it is submitted directly as a “small enhancement”

This needs to be inevitably routed to Product Management for Agile Teams (and not through maintenance)

The goal is to not continue putting enhancements through the SCR ticketing system, however in the transition period, we would continue to use the SCR process until we achieve a complete Agile transformation (i.e. all delivery in Agile) and a level of Agile maturity

In many cases, Business Analysts may triage these enhancement request to get routed to the product backlog with prioritization without immediately handing it off to Product Management.

* + - **Org Readiness** may be helping to route new/confused expectations to New Initiatives. Challenges include:
      * Analysis of impact
      * Attempting to not introduce back-doors unprioritized to teams
      * Informal at the moment when moving potential initiatives to the “agile intake” side of the house as an enhancement
* Difference between a bug and a defect
  + Bugs are problematic code that is not yet in production and not tracked in Rally as a formal “defect”
  + Defects are problematic code that is in production and tracked in Rally as a formal “defect”

## If the teams must perform other kinds of maintenance/support work (updating databases with new data, refreshing some service, rebooting some service, etc.) how will they get that work?

* **Hidden Roles and maintenance**
  + **Emergency maintenance (example: security patch) (SCR scenario) - unpredictable**
    - **DevOps might handle the actual patch**
    - **Teams may participate in testing**
    - **Whomever is participating in testing would need to handle this as a priority**
  + Maintenance definitions: (SCR scenario)
    - Business Maintenance: Scheduled or non-scheduled work requested by the business
    - Examples:
      * Data updated where the business does not have the ability to make those changes on their own
    - Technical Maintenance: Scheduled or non-scheduled work need to maintain systems and support, ensure security, etc.
    - Examples:
      * Updating software to newer, supported versions, completing security maintenance as requested by Optum, etc.
  + Ensure that maintenance work is financed appropriately
  + Ensure teams work on maintenance in a consistent manner across teams
  + Kanban items should be associated with dates needed, but only when applicable.
  + Option for approach:
    - Assumption:
      * A team of people (not agile teams) would manage the prioritization for the maintenance Kanban
    - Approaches
      * Team members would have consistently reduced “sprinting” hours per employee (example: 5 hours for maintenance, per sprint – pre carved out)
      * New maintenance work would be added to a “maintenance” Kanban board (not associated with the team’s sprint/iteration backlog)
    - Maintenance Kanban board would need practice rules like:
      * Team members cannot spend time on Kanban items beyond their maintenance hours (ex: 5 hours max per sprint)
      * Team members pull from the Kanban, and work is not “pushed” to team members
      * Sprint planning does not attempt to steal or double-plan hours allocated to maintenance
  + First approach (decision):
    - Ask team members to pre-allocate hours for the Maintenance Kanban to remove those available hours from the coming sprint
      * Team member would review the kanban and select work and determine hours each sprint (this is done sprint-by-sprint)