Agenda

• Agile DevOps Transition
• Cost Estimating Approach and Challenges
• Agile Metrics
• Current Efforts and Beyond

Disclaimer: All Agile Data in this brief is Notional. Data does NOT reflect any actual contractor values or data provided to U.S. Government.
What does Division Y do?

- Supports array of programs under multiple contracts
- Varying customers: Both Domestic and Foreign
- Division’s software architecture old, tightly coupled and monolithic
The Way Ahead

• Moving towards service-based architecture
• Employing Agile Dev/Ops approaches, utilizing Agile Framework

• For Cost:
  • Revamped estimating to align with Agile methodologies and structure
  • Build dynamic models that facilitate capability with budget
  • Bridge multiple worlds between legacy and next-gen architectures

Through collaboration with budget, PM and other division teams, transitioning to Agile methodologies for Cost Estimation that better captures new structure
Primary focus of cost analysis, zooming in on Program Increment releases, discussions before and after Increments, and the analytics produced by each point.
“Welcome to Agile, where the stories are made up and the points don’t matter.”

– Drew Carey, probably
Where is the Data coming from?

- Program: X
- Contract: Multi-year
- Data collected by Contractor as part of rebaseline effort for charge codes
- Program initially established as Waterfall Development, restructured to incorporate more Agile principles in 2017
- Contractor collects data using Cloud Development Servers

Data dump provided quarterly to PM/FM/Cost (more ideally, monthly)
Product Breakdown

- **Epic - % Complete base on Feature roll-up**
- **Iteration (BCWP = % Story Points Complete)**
  - Included Features worked during Iteration
- **Features (% Complete based on User Stories)**
  - EVM BCWP based on % Complete calculations from Azure
- **User Stories (0 or 100% Complete)**
  - Supports accomplishment of Features
  - Defined entry / acceptance criteria for completion
  - Bug or User Story
- **Tasks (0 or 100% Complete)**
  - Activities to complete User Stories
  - User Story Credit taken when Tasks are closed

Different terminology provided by different contractors for Agile metrics
Contractor Story Point (SP) Approach

- Scoring Ground rules
  - 1 story point = half-day development & half-day test
  - Score using 0.5, 1, 3, 5, or 8 but go even higher
  - An item can refer to a user story or a bug

<table>
<thead>
<tr>
<th>Story Points</th>
<th>Size</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>Extra-Small</td>
<td>&lt; 1d</td>
</tr>
<tr>
<td>1</td>
<td>Small</td>
<td>~ 1d</td>
</tr>
<tr>
<td>3</td>
<td>Medium</td>
<td>1-3d</td>
</tr>
<tr>
<td>5</td>
<td>Large</td>
<td>3-5d</td>
</tr>
<tr>
<td>8</td>
<td>Extra-Large</td>
<td>5-8d</td>
</tr>
</tbody>
</table>

Story Point = Time (in 8 hour work days)

For this contractor, a Story Point is given a measure of time in day
## Primary Data and Analysis, So Far

<table>
<thead>
<tr>
<th>ID</th>
<th>Work Item Type</th>
<th>State</th>
<th>Closed Date</th>
<th>Value Area</th>
<th>Charge Code</th>
<th>Origin</th>
<th>Original Planned Story Points</th>
<th>Story Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>3962</td>
<td>User Story</td>
<td>Closed</td>
<td>10/19/2018</td>
<td>Architectural</td>
<td>Dev 6.2 Iteration</td>
<td>Planned</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3186</td>
<td>User Story</td>
<td>Closed</td>
<td>10/18/2018</td>
<td>Spike</td>
<td>Dev 6.2 Iteration</td>
<td>Planned</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>3191</td>
<td>Bug</td>
<td>Closed</td>
<td>10/23/2018</td>
<td>Spike</td>
<td>Mx 5.1 Iteration</td>
<td>Planned</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>2956</td>
<td>Bug</td>
<td>Closed</td>
<td>10/22/2018</td>
<td>Spike</td>
<td>Mx 5.1 Iteration</td>
<td>Unplanned</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>2848</td>
<td>Bug</td>
<td>Closed</td>
<td>2/20/2019</td>
<td>Business</td>
<td>Dev 7.2 Iteration</td>
<td>Rework</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>3133</td>
<td>User Story</td>
<td>Closed</td>
<td>2/22/2019</td>
<td>User</td>
<td>Mx 6.4 Iteration</td>
<td>Planned</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>3533</td>
<td>User Story</td>
<td>Closed</td>
<td>3/25/2019</td>
<td>User</td>
<td>Dev 7.1 Iteration</td>
<td>Planned</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1139</td>
<td>User Story</td>
<td>Closed</td>
<td>3/25/2019</td>
<td>Enabler</td>
<td>Mx 6.4 Iteration</td>
<td>Planned</td>
<td>1.5</td>
<td>5</td>
</tr>
</tbody>
</table>

- **ID**: Identification value assigned to anything put into Azure, assigned in order of input
- **Work Item Type**: Epic, Feature, User Story, Bug
- **State**: Shows whether the item being worked on is brand new, closed or pending
- **Closed Data**: Applies only to closed item, day and time in which user story / bug was resolved
- **Value Area**: Category of Work Item and where it most benefits
- **Origin**: Planned, Unplanned, Rework
- **Charge Code**: Specific iteration charge code item/work being done is being charged to
- **Original Planned Story Points**: Number of Story Points/Time Engineers initially assess for each user story / bug
- **Story Points**: Number of Story Points/Time Engineers actually took to finish each user story / bug
- Also given in Quarterly Data deliveries: Title, Area Path, Iteration Path, BCWP
Work Items: Planned vs. Unplanned by Type

<table>
<thead>
<tr>
<th>Work Item</th>
<th>Q3 2019 Count</th>
<th>Origin</th>
<th></th>
<th></th>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Planned</td>
<td>Unplanned</td>
<td>Rework</td>
<td>Blank</td>
<td></td>
</tr>
<tr>
<td>User Story</td>
<td>1566</td>
<td>186</td>
<td>43</td>
<td>1058</td>
<td>2853</td>
<td></td>
</tr>
<tr>
<td>Bug</td>
<td>1293</td>
<td>1603</td>
<td>548</td>
<td>375</td>
<td>3819</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>2859</td>
<td>1789</td>
<td>591</td>
<td>1433</td>
<td>6672</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Work Item</th>
<th>Q4 2019 Count</th>
<th>Origin</th>
<th></th>
<th></th>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Planned</td>
<td>Unplanned</td>
<td>Rework</td>
<td>Blank</td>
<td></td>
</tr>
<tr>
<td>User Story</td>
<td>2524</td>
<td>318</td>
<td>102</td>
<td>1425</td>
<td>4369</td>
<td></td>
</tr>
<tr>
<td>Bug</td>
<td>1611</td>
<td>1749</td>
<td>882</td>
<td>251</td>
<td>4493</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>4135</td>
<td>2067</td>
<td>984</td>
<td>1676</td>
<td>8862</td>
<td></td>
</tr>
</tbody>
</table>

- Within data, the children of a Feature are made up of User Stories or Bugs
- In the following slides, the analysis done is focusing on both the User Stories and Bugs together (as Work Items) that have been completed

From a top level, is the contractor planning accordingly the load of new Work Items added between Quarters?
Work Item: Status Changes

- 928 total new tasks/work items added between Quarters
- 974 work items were closed (570 of them added and then closed, 404 closed out from previous Q3 tasks)
  - 5 Closed became a Ready To Work
  - 28 Active, 37 Investigate, 96 New, 73 Proposed, 90 Ready to Triage, 60 Ready To Work, 8 Resolved, 12 Reviewed, and 6 Test had status changed to Closed
- 180 Work Items classified as ‘New’ in Q3 remained so in Q4

Is the contractor limiting technical debt while continuing to add new Work Items?
Work Item: Status Changes - Value Areas

Key Takeaways:

• Of the Work Items closed after Q3 2019, 39% were for the User

• Compared to the history up until Q3 2019:
  • Enabler Work Items added in Q4 2019 were generally proportional
  • User Work Items added increased in Percentage
  • Architectural and Business decreased

When adding new Work Items, is each area of importance being consistently addressed, or is one area being over- or under-serviced?
Planned vs. Actual Story Points: Moving Average

Using a 3 Iteration Moving Average, we see that Development continues to have fewer Actual Story Points compared to what was initially proposed. Maintenance also continues this trend as well. Starting at Iteration 7.1, it appears towards the tail end of the data we start to see a more accurate prediction of story points given the actuals.

Are the Development teams giving themselves enough buffer in order to handle Unplanned Work Items or Bugs before starting the Iteration?
Development: Work Item Counts (by Origin)

- Consistently, Planned Work Items have always been significantly higher than Unplanned or Rework Items.
- However, a particular area to be concerned about in Development are the Unplanned Work Items that take up time/resources that would be better used on new requirements.

Is the Developer maintaining a good ratio of Planned vs. Unplanned? Are they building issues out of solutions?
In contrast to Development, Maintenance tended to have a more even distribution of Planned/Unplanned Work Items. With Unplanned Work Items significantly outnumbering the Planned in a few iterations. However, this is consistent with the Maintenance process and requires less scrutiny.

Is the Maintenance work being done consistent, or is their significant volatility in volume?
Averages of Original vs. Actual Story Points

On average, we see that the Contractor’s original estimate for the amount of time to finish a Planned Work Item is greater than the actual time spent fixing the Work Item for both development and maintenance.

Only for Maintenance Rework was their original estimate for the amount of time to complete a Work Item less than what the actual time was (so it took them longer).

In general, they were fairly accurate in predicting the average amount of time to complete certain tasks when they were unplanned or rework. Depending on when the input the Planned vs. Actual Story Point values could be the rationale for this lack of difference.

In total, contractor underestimating time for Planned work and generally on the mark for the rest.
Development: Actual SPs vs. Number of User Stories/Bugs

- Average time spent per Work Item remained fairly consistent throughout each iteration, hovering around the 2.0 Story Point mark.
- Number of Work Items was fairly consistent per iteration, with only a significant dip coming in 5.X Iterations.

Is the actual time being spent per Work Item remaining consistent throughout the Iterations?
Maintenance: Actual SPs vs. Number of User Stories/Bugs

- Average time spent per Work Item remained fairly consistent throughout each Iteration, hovering around the 1.9 Story Point mark.
- Number of Work Items was fairly consistent per iteration, with only a significant increase coming in 5.X Iterations (contrast to that of Development).

Maintenance shows a bit more volatility in time spent per Work Item.
Release Level Agile Metrics
Metric Tracking Software

• As of Q2 FY20, Program Office has worked with Contractor to stand up progress tracking software for Development Teams

• Product Owners, System Architects and other members of the agile structure, feed and make updates to the requirements, backlog items, etc.

• Currently tracking various Program levels
Program: Epic

- Good source to track progress of program on the Epic level
- Completion %’s also provided in output based on Epic Owner input
- Feature level tends to be identical in analysis/usability

Tracking progress on Epic level can provide more useful metrics for big picture / senior leadership

### Current Status Counts of Program Epics

<table>
<thead>
<tr>
<th>PID</th>
<th>Created Date</th>
<th>Last Activity Date</th>
<th>eStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID.0001</td>
<td>1/1/2019</td>
<td>7/30/2019</td>
<td>Analysis</td>
</tr>
<tr>
<td>PID.0002</td>
<td>1/2/2019</td>
<td>7/31/2019</td>
<td>Analysis</td>
</tr>
<tr>
<td>PID.0003</td>
<td>1/3/2019</td>
<td>8/1/2019</td>
<td>Analysis</td>
</tr>
<tr>
<td>PID.0004</td>
<td>1/4/2019</td>
<td>8/2/2019</td>
<td>Analysis</td>
</tr>
<tr>
<td>PID.0005</td>
<td>1/5/2019</td>
<td>10/1/2019</td>
<td>Analysis</td>
</tr>
<tr>
<td>PID.0006</td>
<td>1/6/2019</td>
<td>1/5/2020</td>
<td>Analysis</td>
</tr>
<tr>
<td>PID.0007</td>
<td>1/7/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0008</td>
<td>1/8/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0009</td>
<td>1/9/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0010</td>
<td>1/10/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0011</td>
<td>1/11/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0012</td>
<td>1/12/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0013</td>
<td>1/13/2019</td>
<td>2/1/2020</td>
<td>Rejected</td>
</tr>
<tr>
<td>PID.0014</td>
<td>1/14/2019</td>
<td>10/30/2019</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0015</td>
<td>1/15/2019</td>
<td>1/30/2020</td>
<td>Approved</td>
</tr>
<tr>
<td>PID.0016</td>
<td>1/16/2019</td>
<td>9/2/2019</td>
<td>Done</td>
</tr>
<tr>
<td>PID.0017</td>
<td>1/17/2019</td>
<td>7/1/2019</td>
<td>Done</td>
</tr>
<tr>
<td>PID.0019</td>
<td>1/19/2019</td>
<td>7/1/2019</td>
<td>Done</td>
</tr>
<tr>
<td>PID.0020</td>
<td>1/20/2019</td>
<td>9/11/2019</td>
<td>Done</td>
</tr>
<tr>
<td>PID.0021</td>
<td>1/21/2019</td>
<td>9/12/2019</td>
<td>Review</td>
</tr>
<tr>
<td>PID.0022</td>
<td>1/22/2019</td>
<td>9/13/2019</td>
<td>Review</td>
</tr>
</tbody>
</table>
Current data flow allows for a traceability between Epic and Features. Tracing through the data clearly shows some disconnects in data input (see Completion %). As more substantial time passes, will be able to track status changes on the Epic/Feature/Defect Report (DR) levels.

Feature level status updates provide a more tangible deliverable being monitored.
One of the important aspects of the current maintenance efforts in MPS are the various DR fixes that have prioritization and severity.

- Similar analysis potential as the Epic/Feature traceability suggests
- Critical in categorizing and ranking needs based on severity and desire
- Time and analysis should open opportunity to estimate timeboxes based on DR quantity, or vice-a-versa
- 30 DRs have no assigned Desirability value
Future of Metrics and Agile within Division Y
Where are we now?

- Working with Senior Leadership, incorporating new estimating techniques with Program Management
- Past methodologies incorporated predominately SLOC counts for major Software Development
- Moving towards FTE based estimates formed from the agile structure
- Using Agile Metrics to better track current efforts. Point out deficiencies or uncharacteristic trends
- Work with Product Owners, Engineering, and other Senior Leadership to prioritize backlog and determine Best/Worst case scenarios based on Project Scope and Velocity
- Attend Program Increment Planning events to both stay in and keep in the loop regarding velocity of Dev Teams and Backlog status

Not an Exhaustive List!

Solution Level
- Solution Architect
- Sys Engineers
- Solution Engineer
- Test (Ops, Automated)
- Solution Management

Program Level
- Sys Architect
- Sys Engineer
- Release Engineer
- Test (Ops, Automated)
- Product Management

Development Team
- Developers
- Testers
- Scrum Master
- Product Owner
- Acq Support
Establishing Guardrails

• Provide context, not control -- Allow decision making at the lowest possible levels while ensuring alignment with organizational goals

• Should there be a focus on keeping pliable guardrails regarding software teams’ work?

• Metrics to keep track of:
  • New work being added
  • Old work being completed/removed
  • Velocity of teams
  • Focus of teams

• Prioritizing work based on origination:
  • Top-driven from larger backlog vs.
  • Side-driven from program vs.
  • Software Team-driven requirements

Metrics and Context of Requirements are critical in optimizing Agile Development
Way Ahead: Cry a little, Try a lot

• Continue to analyze the data coming in as fast as possible
• Learn what works, learn from what doesn’t
• Will never be perfect

“Sucking at something is the first step to becoming sorta good at something” – Jake the Dog
Questions?
Back-up
Average Cost per Story Point by Iteration (Development)

- Cost Actual for each iteration provided in every data drop provided by Contractor
- Above charts show the distribution of the Cost of Iterations per User Story/Bug completed
- Shows a ramp up / ramp down in terms of money spent per Iteration. Cost/SP remained consistent, indicating a consistent amount of Story Points being completed per Iteration

Number of Story Points completed generally associated with increased money spent
Average Cost per Story Point by Iteration (Maintenance)

- Similar to Development, fairly consistent Cost/Story Point throughout the Iterations
- Peaks of Actual Dollars being spent are met with lulls in the corresponding Iteration for Development

Money spent per Story Point remained very consistent, aside from one outlier
Building from Foundation

- Improve metrics being delivered
  - From Contractor
    - Receive metrics to clearly show what comprises Epics and Features
    - Provide feedback on potentially erroneous data
    - Capture FTE counts and charge codes associated per Feature/User Story
  - From Metric Tracking Tools
    - Process for inputting progress of Epics/Features/DRs and making sure they are stacking together properly
    - Diligence in updating and removing out-of-date items
    - Improving metric delivery to senior leadership
  - Additional Data Overall
    - Start/End/Last Modified Dates on the Work Item/User Story level
    - Lead Time, how long it takes to be delivered, on all levels
    - User Satisfaction/Value Added
    - Drilled down categories for Features/Epics
- Additional programs are in the process of getting on new contracts, switching to an Agile development process
  - Currently working with the programs to have similar Agile Metrics reporting and receive access to metric tracking tools

Continue to seek out new ways to improve/ease processes under Agile to every part of Program
Government monitored Guardrails for all Items being developed must be approached delicately.
Agile Keywords

- **Epic** – container for a Solution development initiative large enough to require analysis, the definition of a Minimum Viable Product, and financial approval prior to implementation

- **Enablers** – support the activities needed to extend the architecture capabilities to provide future business functionality

- **Features** – service that fulfills a stakeholder need

- **Iteration** – standard, fixed-length timebox, where teams deliver incremental value in the form of working, tested software and systems (2-3 weeks)

- **Pre-and Post-PI Planning** – events used to prepare for and follow up Program Increments

- **Product Owner** – member of team responsible for defining User Stories and prioritizing the backlog

- **Program Increment (PI)** – timebox during which a release delivers software value

- **Solution** – product, services, or systems delivered to customer