Applying Earned Value Management to Agile Software Development Programs

Bob Hunt – Galorath Federal
Michael Thompson – Galorath Federal
Gordon Kranz - Enlightened Integrated Program Management
Agile 101 – How It Works

- Agile is a set of software development methods in which solutions evolve through collaboration
  - Integrated teams include PeopleSoft SMEs to configure the product, PeopleSoft Developers, Data Integration Developers, and Testers

- Development is iterative
  - The traditional software development phases are seen as continuous activities
  - Work is broken into smaller tasks
  - Multiple iterations may be required to release a product
  - Documentation is created as-built

- For each iteration, a working product is demonstrated to stakeholders

- Emphasizes value-driven approach
  - The usual project constraints still apply
  - Focusing on value allows most important functionality to be delivered first

- Technology agnostic
Agile is not a single method
# Agile 101 – Key Agile Terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agile Process; often Scrum</td>
<td>A framework for team collaboration on complex software projects. 1-10 people (have seen up to 20)</td>
</tr>
<tr>
<td>Sprint</td>
<td>A short multiple-week period where a team completely builds working, tested software. All phases of the SDLC are executed iteratively during a sprint – Analysis, Design, Code, Test. 1-6 weeks (have seen up to 13 weeks) <em>(13 results in 4 sprints per year)</em> (Time Boxing)</td>
</tr>
<tr>
<td>Feature</td>
<td>A set of specifications that can be shown in a user demonstration and oriented on system capabilities. Often the EVM “Work Package”</td>
</tr>
<tr>
<td>Epic</td>
<td>Epic is traditionally a large block of work that needs to be further decomposed.</td>
</tr>
<tr>
<td>Spike</td>
<td>A special type of feature/story used for research and prototyping activities, which can be functional or technical.</td>
</tr>
<tr>
<td>Backlog</td>
<td>A repository for all upcoming and uncompleted work. It consists of features to address user needs and deliver business benefits, as well as architectural features required to build the product.</td>
</tr>
</tbody>
</table>
Agile Implementation

To create working software products often and to demonstrate to the customer.

- Sprint Backlog
- Sprint Burndown
- Impediment List

Ensures holistic activities and iterative processes.
Hybrid Agile Development/Acquisition*

SCRUM roles are critical

*Sometimes called Water-Scrum-Fall
Agile/EVM Building Blocks*

* These “building blocks” are program specific and may be called by different names

EVM work Packages identified at Epic or Theme level
Sprints may be too low a level to track as a work package

Sprints
Cost Estimating often done at the Sprint Level

Release 1 (made up of multiple Themes/Increments

Epic 1
Feature 1
User Story 1

Epic 2
Feature 2
User Story 2

Epic 2
Feature 3
User Story 3

Theme/Increment 1

Requirements mapped to this level
The NDIA IPMD Guide objective:

- Provide tool-agnostic summary guidance and references for EVMS programs engaging Agile.
- Maintain Guide quality by following Version update protocol that represents multiple practitioner perspectives.

Earned Value Management Process

1. **Define the Work**
   - SOW - WBS - WBS dictionary

2. **Plan the Work**
   - IMP/IMS, Control Accounts/Work Packages, EV Methods

3. **Work the Plan**
   - Execute the tasks and activities in the plan

4. **Collect Results**
   - Actual labor and other direct costs

5. **Measure Performance Against the Plan**
   - Status EV and schedule

6. **Analyze Deviations**
   - Cost and Schedule Variances

7. **Institute Appropriate Corrective Actions**
   - Baseline Change Requests

8. **Engineering Change Proposals**

9. **Change Control**
Agile and EVM Integration “Touch Points”

- Organization
  - WBS, OBS, RAM
- Planning, Budgeting, Scheduling
  - IMP/IMS
  - Objective Criteria for Measure
  - Time Phased Budget
- Analysis and Management Reports
  - Progress Claims
  - Forecasting
  - Baseline changes
Agile – EVM Relationships

Agile to EVM Traceability

WBS

HCM
- Departments
  - Digital Signature
    - External Digital Certificates
      - WP.R2.RFC.0088
  - Disciplinary Actions
  - Duty Status
  - Foundation
  - Global Payroll Foundation
  - Hire/Rehire
    - Add a Person - Employee
      - WP.R2.RFC.0088
    - Job Data: HIR/REH
    - Modify a Person
      - WP.R2.RFC.0088
    - PAR: Request to Update Gender
    - Seniority Dates
    - Smart HR Transactions - New Hire

Hire/Rehire Total
- Mass Update
- Military Training
- Orders
- Overarching
- Person of Interest
  - Add a Person - POI
    - WP.R2.RFC.0088
- Physical Profiles
- Positions

Release Plan

HCM
- WP.R2.RFC.0007
- WP.R2.RFC.0013
- WP.R2.RFC.0016
- WP.R2.RFC.0018
- WP.R2.RFC.0021
- WP.R2.RFC.0084
- WP.R2.RFC.0086
- WP.R2.RFC.0088
  - Digital Signature
  - External Digital Certificates
  - Hire/Rehire
    - Add a Person - Employee
    - Modify a Person
  - Person of Interest
    - Add a Person - POI

1) WBS Decomposed to Features
2) Features Mapped to Releases and grouped into Work Packages
3) Work Packages mapped to the IMS

This chart can make Agile look like Waterfall – the key is mapping the Agile Process to Work Packages and he Work Packages to the Acquisition Process

IMS
Where Does EVM fit in?

- As long as there is a plan and product(s) EVM can be applied.
- The product backlog defines the product, and sprints are used to time phase the work.
- Status of each Sprint is rolled up to a level which in this example is the Control Account (CA).
  - Below the EPIC is the Feature at the Work Package (WP) level, which breaks the EPIC into functional packages.
  - Features are decomposed into Stories and Story points.
  - Sprints are statused by, in this case, Stories and Story Points, which are maintained in an Agile Program Management Tool.
- As Features are completed the percent complete is rolled up to the EPIC level.
Agile Baseline
Change is Inevitable, Minimizing the Impact is Critical

• Baseline Changes will happen, they happen to most programs, but they should be limited to as few as possible

• There is constant movement in an Agile program. The changes are divided into How do you account for those changes? First these questions need to be asked:
  • Will the change impact the Baseline?
  • Can the change take place within the existing timeframe of the Feature?
  • If the change impacts the Baseline, a BCR needs to be submitted
  • It all goes back to careful and thoughtful PLANNING and development of the program WBS.
Pulling the Thread

SOW / SOO / PBS

WBS /

IMP / MPS /

Understand the Work → Organize the Work → Plan the Work

Schedule the Work → Authorize, Baseline, Perform and Measure the Work

Release Plan / IMS

Product Backlog / CAP / WAD
### Agile Progress
Claiming what you earn

<table>
<thead>
<tr>
<th>Product</th>
<th>WBS - Product Decomposition</th>
<th>PLN</th>
<th>ACT</th>
<th>% CMPT</th>
<th>SPR T</th>
<th>EVM Hierarchy</th>
<th>Report Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release</td>
<td></td>
<td>8482</td>
<td>5742</td>
<td>76%</td>
<td></td>
<td>Agile</td>
<td>Agile Tool</td>
</tr>
<tr>
<td>WBS</td>
<td>Program X</td>
<td>2831</td>
<td>1574</td>
<td>78%</td>
<td></td>
<td>CA</td>
<td>IPMR Cost and Schedule</td>
</tr>
<tr>
<td>EPIC</td>
<td>Duty Status</td>
<td>77</td>
<td>72</td>
<td>94%</td>
<td></td>
<td>Summary</td>
<td></td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Personal Data</td>
<td>25</td>
<td>25</td>
<td>100%</td>
<td>1.1</td>
<td>WP</td>
<td>IPMR Schedule</td>
</tr>
<tr>
<td>Feature</td>
<td>SS: Personal Details - Addit</td>
<td>5</td>
<td>5</td>
<td>100%</td>
<td></td>
<td>1.1 Agile</td>
<td>Agile Tool</td>
</tr>
<tr>
<td>Feature</td>
<td>Direct Line Reports</td>
<td>20</td>
<td>20</td>
<td>100%</td>
<td></td>
<td>1.1 Agile</td>
<td>Agile Tool</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Activity Threads</td>
<td>27</td>
<td>27</td>
<td>100%</td>
<td>1.1</td>
<td>WP</td>
<td>IPMR Schedule</td>
</tr>
<tr>
<td>Feature</td>
<td>Immediate Approvers</td>
<td>26</td>
<td>26</td>
<td>100%</td>
<td></td>
<td>1.1 Agile</td>
<td>Agile Tool</td>
</tr>
<tr>
<td>Feature</td>
<td>Workflow Notifications</td>
<td>1</td>
<td>1</td>
<td>100%</td>
<td></td>
<td>1.1 Agile</td>
<td>Agile Tool</td>
</tr>
<tr>
<td>WBS</td>
<td>PayRoll Configuration and Development</td>
<td>1480</td>
<td>1262</td>
<td>85%</td>
<td></td>
<td>CA</td>
<td>IPMR Cost and Schedule</td>
</tr>
</tbody>
</table>

**Story Points Planned Vice Actual underpin progress claims in EV**
Approach Allows Following Progress Reports

• Earned Value Status at EPIC Level
• Progress against
  • % EPIC Completion
  • % Module Completion
  • % Feature Completion Total
  • % SSS Completion
• Metrics and Trends
  • Feature Burn Down / Up
  • EPIC Burn Down / Up
  • SSS Burn Down / Up
  • Sprint Velocity by Team and over time
• Supports EAC Development from a variety of angles for cross verification
## Notional Sprint Data

<table>
<thead>
<tr>
<th>Module</th>
<th>Date</th>
<th>Total Story Points</th>
<th>Total Story Points Planned</th>
<th>total Storypoints Completed</th>
<th>Velocity</th>
<th>Remaining Story Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCM Configuration and Development</td>
<td>3-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Determine Story Points</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7-Sep-16</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18-Sep-16</td>
<td>16</td>
<td>16</td>
<td>16.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sprint 2</td>
<td>3-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Determine Story Points</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7-Sep-16</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Sep-16</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-Sep-16</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18-Sep-16</td>
<td>16</td>
<td>16</td>
<td>16.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sprint 3</td>
<td>3-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Determine Story Points</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-Sep-16</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9-Sep-16</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13-Sep-16</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-Sep-16</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17-Sep-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18-Sep-16</td>
<td>21</td>
<td>18</td>
<td>15.667</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Notional Story Point* Burndown Chart

A statistical relationship may or may not be useful

* I am using Story Point and Feature Point as interchangeable terms
Notional Velocity Chart

- **Velocity Chart**
  - **Sprint #**: 1, 2, 3
  - **Story Points**: 0 to 25
  - **Legend**:
    - Orange: Total Story Points Planned
    - Gray: Story Points Complete
    - Yellow: Velocity

- **Graph Description**
  - Sprint 1: Total Story Points Planned = 15, Story Points Complete = 15
  - Sprint 2: Total Story Points Planned = 15, Story Points Complete = 15
  - Sprint 3: Total Story Points Planned = 20, Story Points Complete = 15

- **Key Attributes**
  - **Velocity**: Measured across sprints (1, 2, 3)
Summary

• Fixed Price and/or LOE contracts in the early phases should be written so that key “value-added” metrics are collected and reported during each increment.

• Estimators may have to employ a variety of software estimating methodologies within a single estimate to model the blended development approaches being utilized in today’s development environments.
  • An agile estimating process can be applied to each iteration/sprint.
  • Future Increments can be estimated based on most recent-successful IID performance.

• Cost estimators will have to scrutinize these programs like a schedule analyst might to determine the most likely IOC capabilities and associated date.
  • The number of increments are an important cost driver as well as an influential factor in uncertainty/risk modeling.
Summary

• All of the estimation methods are susceptible to error, and require accurate historical data to be useful within the context of the organization

• When developers and estimators use the same “proxy” for effort, there is more confidence in the estimate
Contact Information

• Bob Hunt
  • President, Galorath Federal
  • Email: BHunt@Galorath.com
  • Phone: 703.201.0651

• Mike Thompson
  • Email: mthompson@galorath.com
  • Phone: 301.904.1103

• Gordon Kranz
  • President, Enlightened Integrated Program Management
  • gmkranz@eipm-llc.com
  • 571.268.8168
BACK UP
## Documentation Traceability

### Contractor CWBS Matrix

![Contractor CWBS Matrix](image)

### CSDR CWBS Matrix

![CSDR CWBS Matrix](image)
EVM – Agile Traceability

<table>
<thead>
<tr>
<th>Epics</th>
<th>Features</th>
<th># of SSS Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPIC</td>
<td>Benefits 1</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Benefits 2</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Benefits 3</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Benefits 4</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Benefits 5</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Promotions</td>
<td>Benefits</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Hire / Rehire 1</td>
<td>Benefits</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Hire / Rehire 2</td>
<td>Benefits</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Hire / Rehire 3</td>
<td>Benefits</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Hire / Rehire 4</td>
<td>Benefits</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Hire / Rehire 5</td>
<td>Benefits</td>
</tr>
<tr>
<td>Sub-Epic</td>
<td>Hire / Rehire</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Benefits</td>
<td>Benefits</td>
</tr>
<tr>
<td>EPIC</td>
<td>Promotions 1</td>
<td>Benefits</td>
</tr>
</tbody>
</table>

Initial Features Priorities & LOE. Xls

INTEGRITY INNOVATION EXCELLENCE
IT Project Success

20% to 30% Failures

Why they fail

Figure 1. Distribution of Success and Failure Across Project Sizes

Source: Gartner (June 2012)
Five Estimating Methodologies

- **Methodology 1:** Since many Agile programs are fixed price, it is often just a matter of labor rates times quantity.
- **Methodology 2:** Simple Build-up approach based on averages can be defined as:
  - Sprint Team Size (SS) x Sprint length (Sp time) x Number of Sprints (# Sprints)
- **Methodology 3:** Structured approach based on established “velocity” – most often used internally by the developer since detailed/sensitive data are available to them.
- **Methodology 4:** Automated Models approach based on a size metric – which may be difficult to quantify.
- **Methodology 5:** Factor/Complexity approach based on data generated in early iterations.
**What to measure**

<table>
<thead>
<tr>
<th>Information Category</th>
<th>Measurable Concepts</th>
<th>Prospective Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1 Schedule and Progress</strong></td>
<td>Milestone completion</td>
<td>Mileston Dates</td>
</tr>
<tr>
<td></td>
<td>Critical Path Performance</td>
<td>Slack Time</td>
</tr>
<tr>
<td></td>
<td>Work Unit Progress</td>
<td>Requirements Traced, Requirements Tested, Problem Reports Opened, Problem Reports Closed, Reviews Completed, Change Requests Opened, Change Requests Resolved, Units Designed, Units Coded, Units Integrated, Test Cases Attempted, Test Cases Passed, Action Items Opened, Action Items Completed</td>
</tr>
<tr>
<td></td>
<td>Incremental Capacity</td>
<td>Components Integrated, Functionality Integrated</td>
</tr>
<tr>
<td><strong>2 Resources and Cost</strong></td>
<td>Personnel Effort</td>
<td>Staff Level, Development Effort, Experience Level, Staff Turnover</td>
</tr>
<tr>
<td></td>
<td>Financial</td>
<td>BCWS, BCWP, ACWP, Budget, Cost</td>
</tr>
<tr>
<td></td>
<td>Environmental/Support</td>
<td>Quality Needed, Quality Available, Time Available, Time Used</td>
</tr>
<tr>
<td><strong>3 Product Size &amp; Stability</strong></td>
<td>Physical Size/Stability</td>
<td>Database Size, Components, Interfaces, LOC</td>
</tr>
<tr>
<td></td>
<td>Functional Size</td>
<td>Requirements, Function Changes, Function Points</td>
</tr>
<tr>
<td><strong>4 Product Quality</strong></td>
<td>Functional Correctness</td>
<td>Defects, Age of Defects, Technical Performance</td>
</tr>
<tr>
<td></td>
<td>Maintainability</td>
<td>Time to Release, Cyclomatic Complexity</td>
</tr>
<tr>
<td></td>
<td>Efficiency</td>
<td>Utilization, Throughput, Response Time</td>
</tr>
<tr>
<td></td>
<td>Portability</td>
<td>Stand Compliance</td>
</tr>
<tr>
<td></td>
<td>Usability</td>
<td>Operator Errors</td>
</tr>
<tr>
<td></td>
<td>Realibility</td>
<td>MTTF</td>
</tr>
<tr>
<td><strong>5 Process Performance</strong></td>
<td>Process Compliance</td>
<td>Reference Maturity Rating, Process Audit Findings</td>
</tr>
<tr>
<td></td>
<td>Process Efficiency</td>
<td>Productivity, Cycle Time</td>
</tr>
<tr>
<td></td>
<td>Process Effectiveness</td>
<td>Defects Contained, Defects Escaping, Rework Effort, Rework Components</td>
</tr>
<tr>
<td><strong>6 Technology Effectiveness</strong></td>
<td>Technology Suitability</td>
<td>Requirements Coverage</td>
</tr>
<tr>
<td></td>
<td>Technology Volatility</td>
<td>Baseline Changes</td>
</tr>
<tr>
<td><strong>7 Customer Satisfaction</strong></td>
<td>Customer Feedback</td>
<td>Satisfaction Rating, Award Fee</td>
</tr>
<tr>
<td></td>
<td>Customer Support</td>
<td>Request for Support, Support Time</td>
</tr>
</tbody>
</table>

* Practical Software Measurement; McGarry, Card, Jones; Addison-Wesley 2002
Software Development

• While there are many approaches to Software Development, they can generally be placed into 2 categories:
  
  • **Plan Driven** – following a version of the Waterfall Development Process
  
  • **Iterative Driven** – following a “version” of the Agile Development Process

• Plan Drive programs have an assumption of some reliable/realistic size metric, for example:
  
  • Source Lines of Code (SLOC)
  
  • Function Points
  
  • Use Cases, User Stories, Web Pages
What is Agile Software Development?

• In the late 1990s, several methodologies received increasing public attention

• Each had a different combination of old, new, and transmuted old ideas, but they all emphasized:
  • Close collaboration between the programmer and business experts
  • Face-to-face communication (as more efficient than written documentation)
  • Frequent delivery of new deployable business value
  • Tight, self-organizing teams
  • And ways to craft the code and the team such that the inevitable requirements churn was not a crisis
Agile Release and Sprint Backlog

Agile methods can start earlier in the process because we are starting with a COTS product baseline.
Agile methods are generally used for projects of smaller scope.

Scaled Agile Framework (SAFe) builds from Agile methods to provide a construct for large-scale implementations.

A playbook tailored to particular software program will inform the development of the IMS and form the basis to guide teams.

- Introduces features to bridge the gap between epics (business processes) and sprints.
- Introduces spikes to explore various approaches to address key foundational decisions.
How to Define Work

---

1 - Requirements are defined based on the Business Process.

2 - Requirements are grouped into features aligned with the delivered software product.

3 - Features are decomposed into tasks to be executed in sprints.

---

- **Features will be classified as one of the following types:**
  - Framework: Foundational capabilities that need to be completed early as they will be leveraged throughout the solution.
  - Core: Capabilities support transactional processing. These include setup tables that maintain valid values and capabilities to manage employee records.
  - Self-Service: Any feature that has a self-service component is identified.
  - Information: Reports, queries, dashboards.
  - Data: Conversions and interfaces.
How to Apply Agile in a Non-Agile World

Plan → Analyze → Deploy → Run

SRR → SFR → IBR

Analyze

Requirements Analysis

Fit/Gap

Deploy

TRR → DIT → GAT → LUT → OPT → DEP

Operational Testing

Integration Testing

Run

Build

Detailed Design

Unit Testing

Develop

PDR → CDR → PLT

Agile Territory

Waterline
When performance is measured performance improves

Estimation processes are independent of tools

We will use these 10 steps in our Estimation Process Improvement Program (EPIP)
Fundamental Model Assumptions

Fundamental assumptions of most Software Estimating Models

• There is a fixed relationship between size and effort, e.g.
  \[(\text{Effort}^{\ast n}) \cdot \text{Time} = \text{Size/Technology}\]

• Results are then modified by current trends and analyses

• Total effort can be distributed by a mathematical model; e.g. Weibull, Rayleigh
### Methodology 5: Factor/Complexity Approach

<table>
<thead>
<tr>
<th>Scale</th>
<th>Functional Description</th>
<th>Effort Multipliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>- - -</td>
<td>Significantly less functionality to be delivered</td>
<td>0.5</td>
</tr>
<tr>
<td>- -</td>
<td>Moderately less functionality to be delivered</td>
<td>0.7</td>
</tr>
<tr>
<td>-</td>
<td>Slightly less functionality to be delivered</td>
<td>0.9</td>
</tr>
<tr>
<td>=</td>
<td>Functionality equivalent to Increment X</td>
<td>1.0</td>
</tr>
<tr>
<td>+</td>
<td>Slightly more functionality to be delivered</td>
<td>1.3</td>
</tr>
<tr>
<td>++</td>
<td>Moderately more functionality to be delivered</td>
<td>1.7</td>
</tr>
<tr>
<td>+++</td>
<td>Significantly more functionality to be delivered</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>Complexity Description</th>
<th>Effort Multipliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>- -</td>
<td>Significantly less complex</td>
<td>0.7</td>
</tr>
<tr>
<td>-</td>
<td>Slightly less complex</td>
<td>0.9</td>
</tr>
<tr>
<td>=</td>
<td>Complexity equivalent to Increment X</td>
<td>1.0</td>
</tr>
<tr>
<td>+</td>
<td>Slightly more complex</td>
<td>1.3</td>
</tr>
<tr>
<td>++</td>
<td>Significantly more complex</td>
<td>1.7</td>
</tr>
</tbody>
</table>

- These initial set of factors came from the environmental factor from traditional software cost models.
- Because each Increment is a mini project, use a Rayleigh or simple Beta Curve (such as a 60/50 Beta curve) to phase costs.