Comparing and Contrasting the PP and PMC Process Areas of CMMI v 1.2 and Scrum

Aldo Dagnino, Andrew Cordes, and Karen Smiley

ABB Inc. US Corporate Research Center

CMMI Technology Conference and User Group
November 12-15, 2007
Hyatt Regency Tech Center, Denver CO
SCRUM is based on the principle that software development is an empirical process, not a defined process, and that if you try to manage an empirical process with a system designed for defined processes, you are doomed to fail.

SCRUM expects the unexpected and control is exercised through frequent inspection and adaptation.

SCRUM is a borrowed term from Rugby.

It claims to be CMMI Maturity Level 3 compliant.
Scrum Flow

Product Backlog (Prioritized To Do List) -> Sprint Planning -> Selected Product Backlog -> Sprint Planning 2

Product Owner (VISION!) -> Reflection -> New Functionality

Sprint Backlog

Sprint Daily and Monthly
Phases

- Planning
  - Initial requirements
  - Initial release planning
  - Architectural and business vision
  - Establishment of ROI gauges with initial funding

- Staging
  - Addition of non-functional requirements to project backlog
  - Route maps

- Development
  - 30-day sprints
  - 1 day bursts of implementation

- Release
  - Enough usable functionality is developed, or
  - Release date is achieved
Establish Estimates

Scrum

Product backlog: prioritized product features desired by the customer

Sprint backlog: (30 days) tasks to create Increment. Lifecycle is evolutionary.

Estimates are established for sprint backlog

CMMI

Top level WBS: work packages, task descriptions

Evolutions can be also considered in the definition of the project lifecycle

Estimates are determined for complete WBS
Develop a Project Plan

Scrum

**Budget** is agreed at the beginning of the project and revised at each iteration. Product Owner manages ROI; business decision at the end of the sprint on project continuation. **Sprints** deliver results every 30 days. **Daily Scrum meetings** held every 24 hours.

**Project risks** are analyzed and evaluated daily.

**Project data** primarily includes **product backlog**, **sprint backlog**, and **product increment**.

**Project resources** discussed at **Sprint Planning Meeting** and **Daily Scrums**.

**Product backlog** and **sprint backlog** act as the project plan.

CMMI

**Establish budget and schedule** and maintain it.

**Identify project risks**.

**Plan for management of project data**.

**Plan for project resources**, knowledge and skills, and planning for stakeholder involvement.

Establish and maintain the overall **Project Plan**.
Obtain Commitment to the Plan

**Scrum**

- **Product backlog and sprint backlog** act as the project plan
- **Sprint planning** reconciles resources. **Daily Scrum** meetings revises resource needs
- **Plan commitment** obtained at Sprints and Daily Scrums; team as a whole owns the sprint plan, and commits to meet it

**CMMI**

- **Review all plans** that affect the project
- **Reconcile work** and resource levels
- **Obtain plan commitment**
**Monitor Project Against Plan**

**Scrum**

- **Use backlog** to monitor progress and commitments.
- **Volunteerism** used as work load balancing.
- Discuss **risks** and **issues** and remove impediments at Daily Scrum meetings.
- Monitoring done at **Daily Scrum** and **Sprint planning meetings**.
- **Stakeholder involvement** monitored at Daily Scrum meetings.
- Progress and milestone reviews conducted during the **Daily Scrum** and **Sprint planning meetings**.

**CMMI**

- **Monitor project planning indicators** of project progress.
- Monitor **commitments**.
- Monitor project **risks**.
- Monitor **data management**.
- Monitor stakeholder involvement.
- Conduct **progress and milestone** reviews.
## Manage Corrective Action to Closure

### Scrum

- **Corrective actions** taken and managed to closure primarily at **Daily Scrum meetings**

### CMMI

- Analyze issues to determine **corrective actions**
- Take **corrective actions** on identified issues.
  - Manage **corrective actions** to closure
Scrum "Home Ground"

- Meeting highly volatile customer needs (requirements)
- Managing tighter dependencies among team members' work
- Having co-located teams
- Coping with shorter planning horizons
- Building "just enough" software functionality, and then ending the project
- Having a small development team
ABB
Power and productivity for a better world™