|  |  |
| --- | --- |
| **Traditional** | **Agile** |
| * Assume that requirements can be known at the start of the project * Product Owner/User involvement with development team is minimal after requirements are approved * Schedule, budget, architecture and design for entire project can be created “up front” given requirements * Change managed formally. * Rework is bad * Lots of documentation * **Traditional strengths:**   Works well for projects with:   * Stable product definition * Well-understood technologies * Quality constraints stronger than cost & schedule * Supports technically weak staff * Provides structure * Good for overseas projects * **Traditional disadvantages** * Not flexible * Teams can end up following obsolete plans * Requirements difficult to define up front * Users tend to define as many requirements that they can think of * An over reliance on processes (that may not be working well or followed) * Can produce excessive documentation * Few visible signs of progress until the end | * Assume that requirements will emerge * Product Owner/User works closely with development team on a daily basis * Schedule, budget, architecture and design for project evolve as requirements emerge * Change is managed informally * Product built so you always have a working system and can “release” at any time. * Rework (refactoring) is good * Requirements are documented as “user stories” (a backlog of product features) * Implementation occurs in short ‘sprints’ * Pre- and post-sprint planning & acceptance activities * Teams are self organizing * Short (<30 minutes) daily standup meetings to communicate status & problems   **Agile strengths:**   * Guaranteed to meet customer needs (Product Owner determines the user stories to implement each sprint) * Unimportant requirements are not developed * Minimal documentation. Example: Requirements defined as user stories * Development team is trusted * Each sprint produces a potentially   shippable product  **Agile Disadvantages**   * Lack of long range requirements means minimal long range planning * Emerging requirements Requires management to delegate decision-making authority to the Scrum team. * Requires more senior developers * Some workers are not comfortable with the responsibility Scrum enables |

**Traditional and Agile measurement:**

|  |  |  |
| --- | --- | --- |
| Measurement | Traditional | Agile |
| Are we building the right product? | Managing Requirements | Customer Acceptance after each sprint |
| How long will the project take? | Big plan up front & earned value | Velocity & Burn Down Charts |
| How much will the project cost? | Big plan up front & earned value | Developers are designed 100% to project, so project cost is # of developers \* # sprints |
| Product Change | Needs to be managed | Natural part of meeting customer needs. Only need to measure # of stories |
| Rework and Refactoring | Rework is bad; Should be eliminated | Refactoring is a natural part of meeting customer needs. No need to measure |
| Quality | Process & product quality | Emphasis on product quality |