Managing Agile Budgets

Creating an Initial Budget

⇒ Calculate the cost for scrum team by using an hourly rate for each team member.

Multiply each team member's hourly rate by his available hours per week by no. of weeks in your sprint to calculate ~~Total~~ scrum team's per sprint cost.

⇒ Cost of additional resources also need to be included.

Creating a Self funding Project

TABLE 13-4 Sample Scrum Team Budget for a Two-Week Sprint

Team Member	Hourly Rate	Weekly Hours	Weekly Cost	Sprint Cost (2 Weeks)
Don	$80	40	$3,200	$6,400
Peggy	$70	40	$2,800	$5,600
Bob	$70	40	$2,800	$5,600
Mike	$65	40	$2,600	$5,200
Joan	$85	40	$3,400	$6,800
Tommy	$75	40	$3,000	$6,000
Pete	$55	40	$2,200	$4,400
Total		280	$20,000	$40,000

Creating a self-funding project

A big benefit of agile projects is the capability to have a self-fund

TABLE 13-5

Income from a Traditional Project with a Final Release after Six Months

Month	Income Generated	Total Project Income
January	$0	$0
February	$0	$0
March	$0	$0
April	$0	$0
May	$0	$0
June	$0	$0
July	$100,000	$100,000

In Table 13-5, the project created $100,000 in income after six months of development. Now compare the income in Table 13-5 to the income generated in Table 13-6.

TABLE 13-6

Income from a Project with Monthly Releases and a Final Release after Six Months

Month/Release	Income Generated	Total Project Income
January	$0	$0
February	$15,000	$15,000
March	$25,000	$40,000
April	$40,000	$80,000
May	$70,000	$150,000
June	$80,000	$230,000
July	$100,000	$330,000

In Table 13-6, the project generated income with the first release. By the end of six months, the project had generated $330,000 — $230,000 more than the project in Table 13-5.

Using Velocity to determine long range Costs

⇒ Multiply the cost per sprint by the number of sprints the Scrum team needs to complete the product backlog

for ex. 40000 Rs. per sprint × 50 sprint = 2,000,000
Remaining Cost

⇒ Lowering Cost by increasing Velocity

⇒ Lowering Cost by reducing time

Managing Team Dynamics and Communication

Managing Team Dynamics and Communication

Managing Agile Team Dynamics

Becoming scy managing & scy organizing

- Scrum Teams are directly accountable for deliverables.
- Scy management means people are professional, motivated dedicated enough to commit to a job.
- Needs Trust & Respect within the team.

Supporting the Team : The Servant-leader

Characteristics of Servant Leader.

1. Listening
2. Empathy
3. Healing
4. Awareness
5. Persuation
6. Conceptualization
7. Foresight
8. Stewardship
9. Commitment to the growth of people
10. Building Community

⇒ Working with a dedicated Team

Benefits (1) Prevent Distractions

(2) Dedicated scrum teams have fewer distractions, thus fewer mistakes.

(3) People know what they will be working on everyday

(4) Able to innovate more on ~~the~~ projects

(5) More likely to be happy in their jobs

(6) Accurate calculation of team's velocity.

Working with Cross-functional Team

⇒ Everyone on team is willing to pitch in on different parts of the project as much as possible.

How can I contribute Today ?

How can I expand my contribution in the future ?

⇒ Cross functionality gives opportunity to learn new skills by working on areas outside of their expertise.

⇒ Allows people to share knowledge

⇒ Eliminate Single point of failure

Limiting - Development Team Size

⇒ · Ideal size is 3 - 9 people.

⇒ Provides enough diverse skills for all tasks.

⇒ Easy interaction & make decision by consensus

① • An emphasis on technical excellence & good design

⇒ Through self management, self organization.

② quality Dev. Techniques (Extreme Programming dev. approaches)

ⓐ TDD (Test Driven Development)

ⓑ Pair Programming : Working in group of two

one is in tactical Role

another is in strategic / navigational role,

looking ahead & providing in moment feedback.

Managing projects with dislocated teams

In *A Scrum Handbook* (Scrum Institute Training Press), Jeff Sutherland describes three models of distributed scrum teams:

» **Isolated scrums:** With isolated scrums, individual scrum teams have collocated scrum team members, but each scrum team is in a separate geographic location and works separately. Product development with isolated scrums has only code-level integration; that is, the different teams don't communicate or work together but expect the code to work when it is time to integrate each module due to organizational coding standards. Isolated scrums tend to struggle because different people interpret coding standards differently.

- **Distributed scrum of scrums:** With a distributed scrum of scrums model, scrum teams are in different locations, like in isolated scrums. To coordinate work, scrum teams hold a *scrum of scrums* — a meeting of multiple scrum masters — to integrate on a daily basis.
- **Integrated scrums:** Integrated scrum teams are cross-functional, with scrum team members in different locations. A scrum of scrums still occurs but face-to-face communication is lost.

TABLE 14-3 **Success of Collocated and Dislocated Scrum Teams**

Team Location	Success Percentage
Collocated scrum team	83%
Dislocated but physically reachable	72%
Distributed across geographies	60%

"Agile Adoption Rate Survey Results" (Scott W. Ambler, Ambysoft, Copyright © 2008)

Managing Quality and Risk

TABLE 15-1 Traditional versus Agile Quality

Quality Management with Traditional Approaches	Quality Dynamics with Agile Approaches
Testing is the last phase of a project before product deployment. Some features are tested months after they were created.	Testing is a daily part of each sprint and is included in each requirement's definition of done. You use automated testing, allowing quick and robust testing every day.
Quality is often a reactive practice, with the focus mostly on product testing and issue resolution.	You address quality both reactively, through testing, and proactively, encouraging practices to set the stage for quality work. Examples of proactive quality approaches include face-to-face communication, pair programming, and established coding standards.
Problems are riskier when found at the end of a project. Sunk costs are high by the time teams reach testing.	You can create and test riskier features in early sprints, when sunk costs are still low.
Problems or defects, sometimes called *bugs* in software development, are hard to find at the end of a project, and fixes for problems at the end of a project are costly.	Problems are easy to find when you test a smaller amount of work. Fixes are easier when you fix something you just created, rather than something you created months earlier.
Sometimes, to meet a deadline or save money, teams cut the testing phase short.	Testing is assured on agile projects because it is part of every sprint.

of this chapter, we state that quality and risk are closely related. The

Proactive quality Practices

1. An emphasis on technical excellence & good design
2. Incorporation of quality-specific development techniques in to product creation
3. Daily communication between the dev. team & prod owner
4. Acceptance criteria built in to user stories
5. face to face comm^{n} and collocation
6. Sustainable development
7. Regular inspection & ~~adaptation~~ adaptation of work & behavior

① An emphasis on technical excellence & good design

⇒ Through self management, self organization.

② quality Dev. Techniques (Extreme Programming dev. approaches)

ⓐ TDD (Test Driven Development)

ⓑ Pair Programming : Working in group of two

one is in tactical Role

another is in strategic / navigational role, looking ahead & providing in moment feedback.

(c) Peer Review : Reviewing one another's role.

(d) Collective Code Ownership

⇒ everyone on the team can create, change or fix

⇒ Speedup development, encourage innovation.