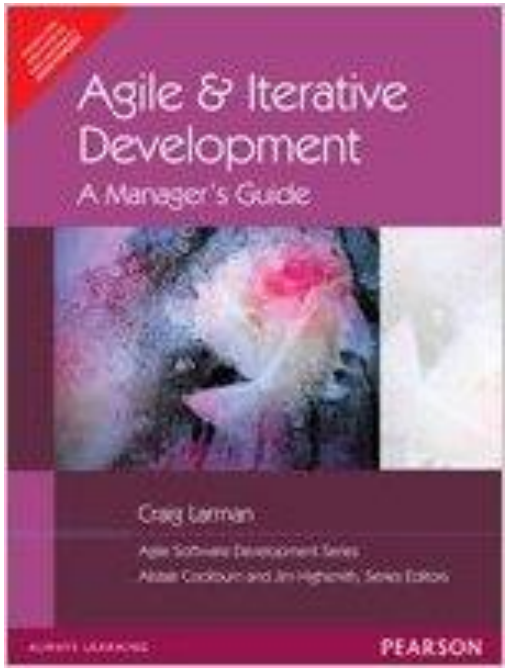
BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Agile Methodologies

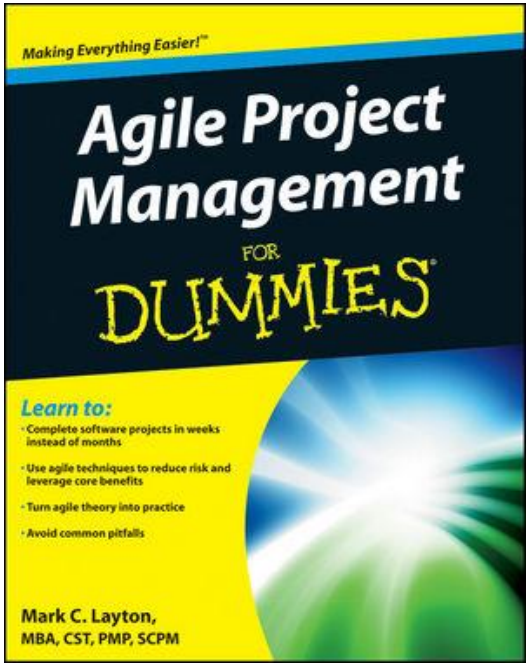
- Prof K G Krishna

Text/Reference Books

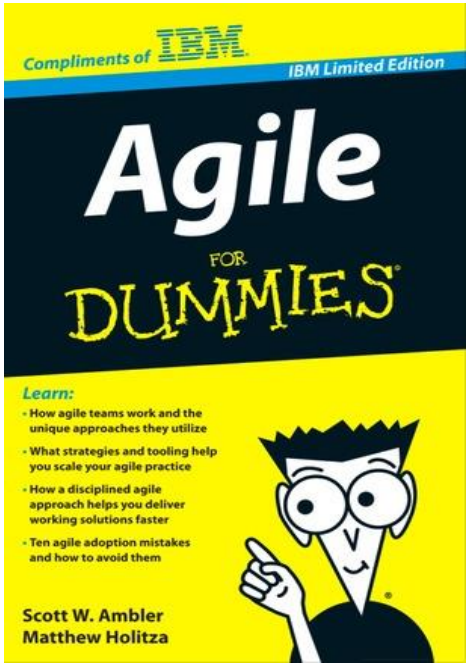
T1



T2



Compliments of IBM



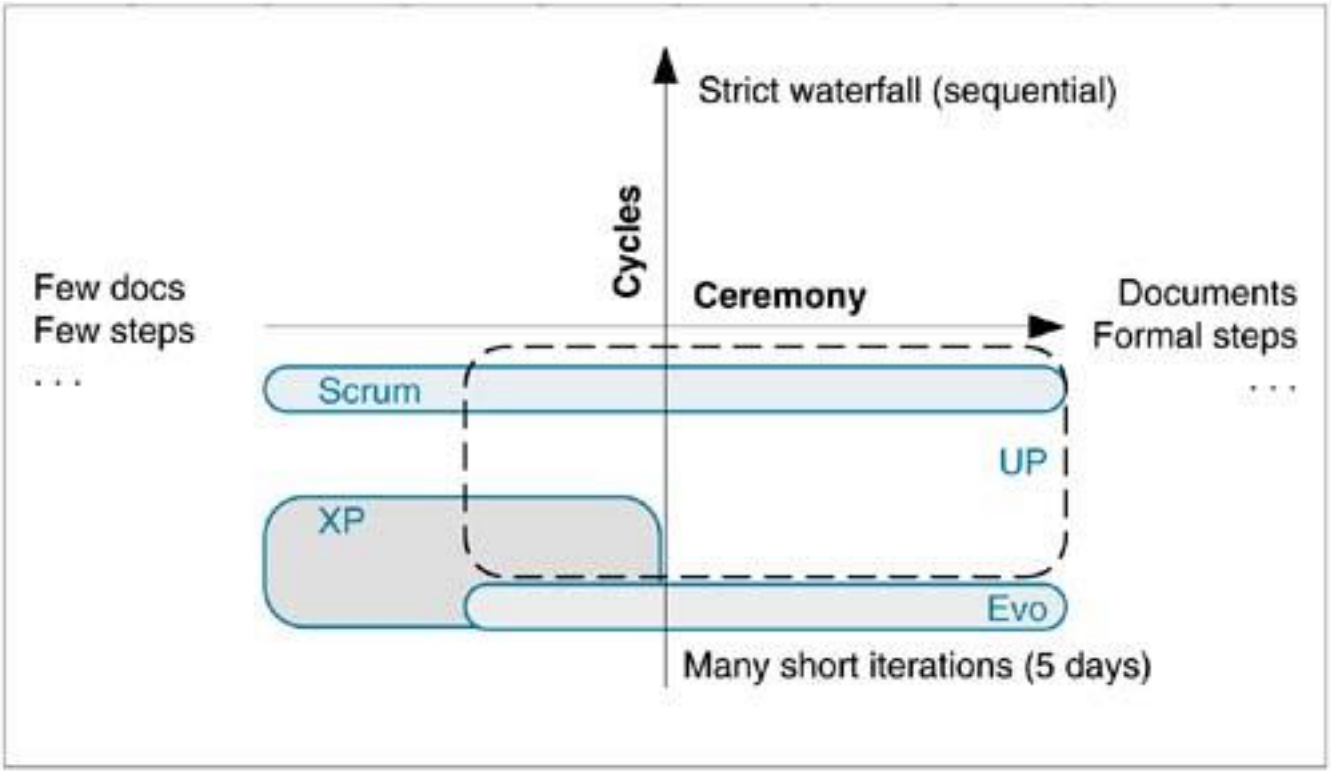
➔ As this field is evolutionary, the student is advised to stay tuned to the current and emerging practices by referring to their own organization's documentation as well as Net sources

Topics

Overview of Agile Methodologies

- SCRUM
- Extreme Programming (XP)
- Test-Driven Development (TDD)

SCRUM on '*Ceremony – Cycles*' Scale



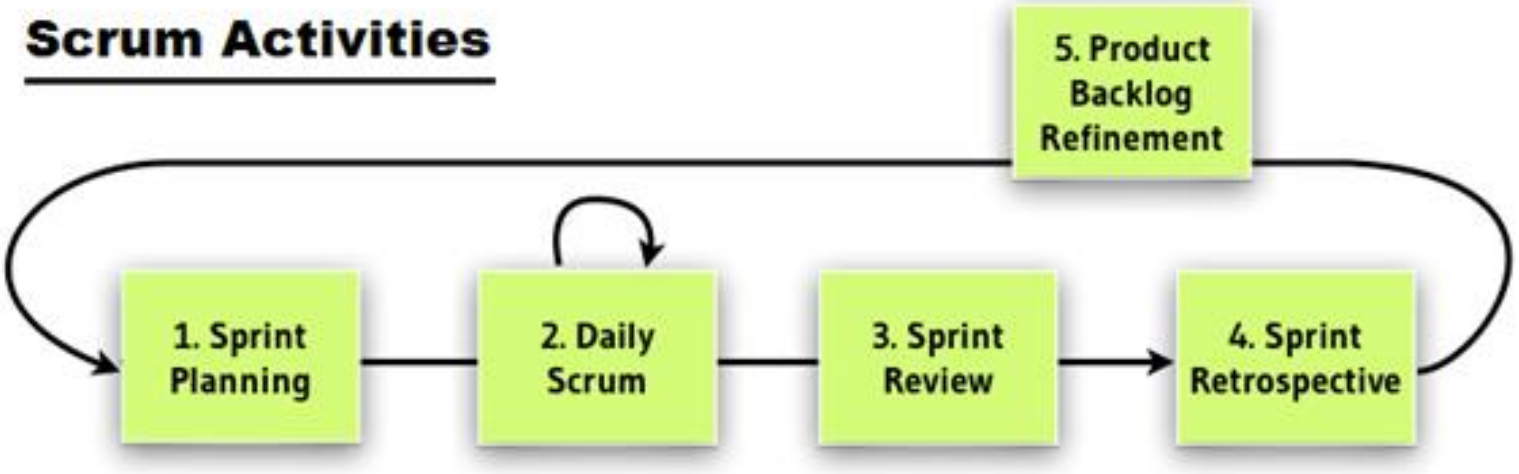
Source: T1-Chap7

SCRUM Lifecycle

PRE-GAME		DEVELOPMENT	RELEASE
PLANNING	**STAGING**		
Purpose: - establish the vision, set expectations, and secure funding	**Purpose**: - identify more requirements and prioritize enough for first iteration	**Purpose**: - implement a system ready for release in a series of 30-day iterations (Sprints)	**Purpose**: - operational deployment
Activities: - write vision, budget, initial Product Backlog and estimate items - exploratory design and prototypes	**Activities**: - planning - exploratory design and prototypes	**Activities**: - Sprint planning meeting each iteration, defining the Sprint Backlog and estimates - daily Scrum meetings - Sprint Review	**Activities**: - documentation - training - marketing & sales - . . .

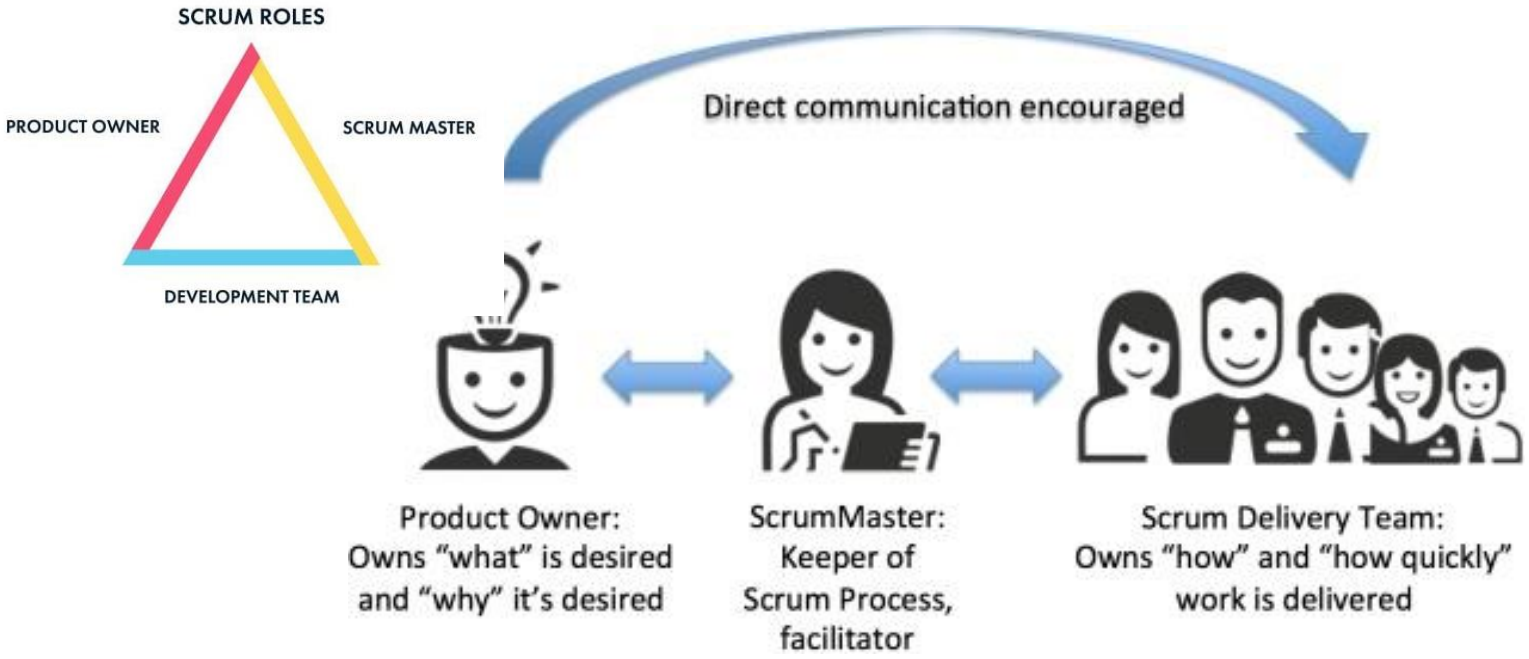
Source: T1-Chap7

SCRUM Activities

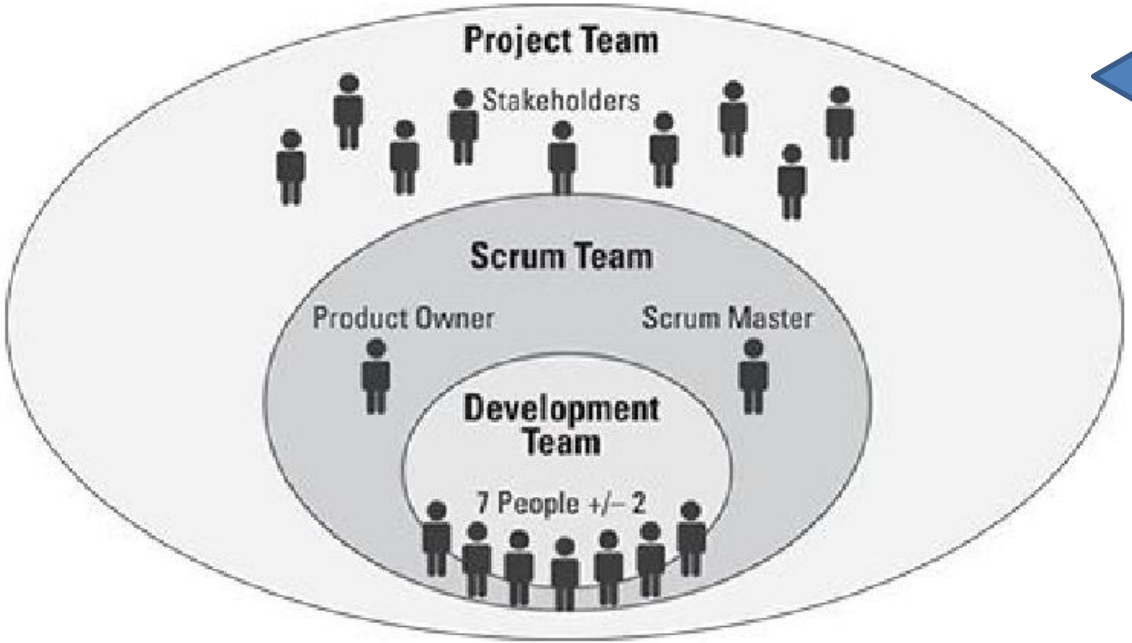


Source courtesy: www.snyxius.com/how-to-run-scrum-planning-meeting-like-pro

SCRUM – The *Key Players*



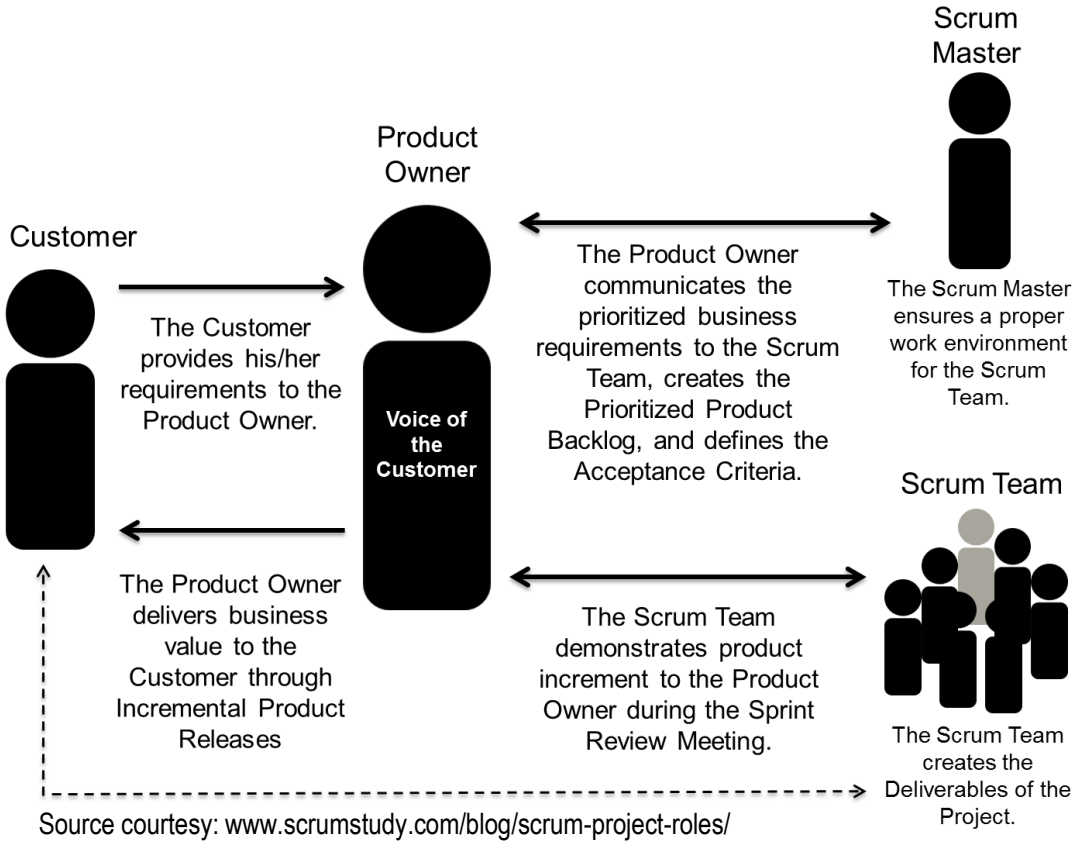
The ‘Extended’ SCRUM Team



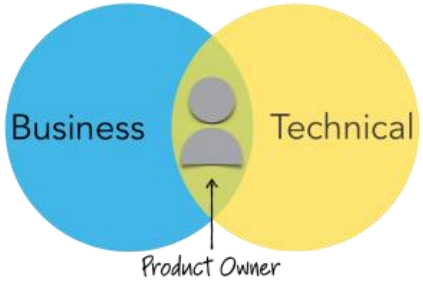
+ Agile Coach

Source: (T2)

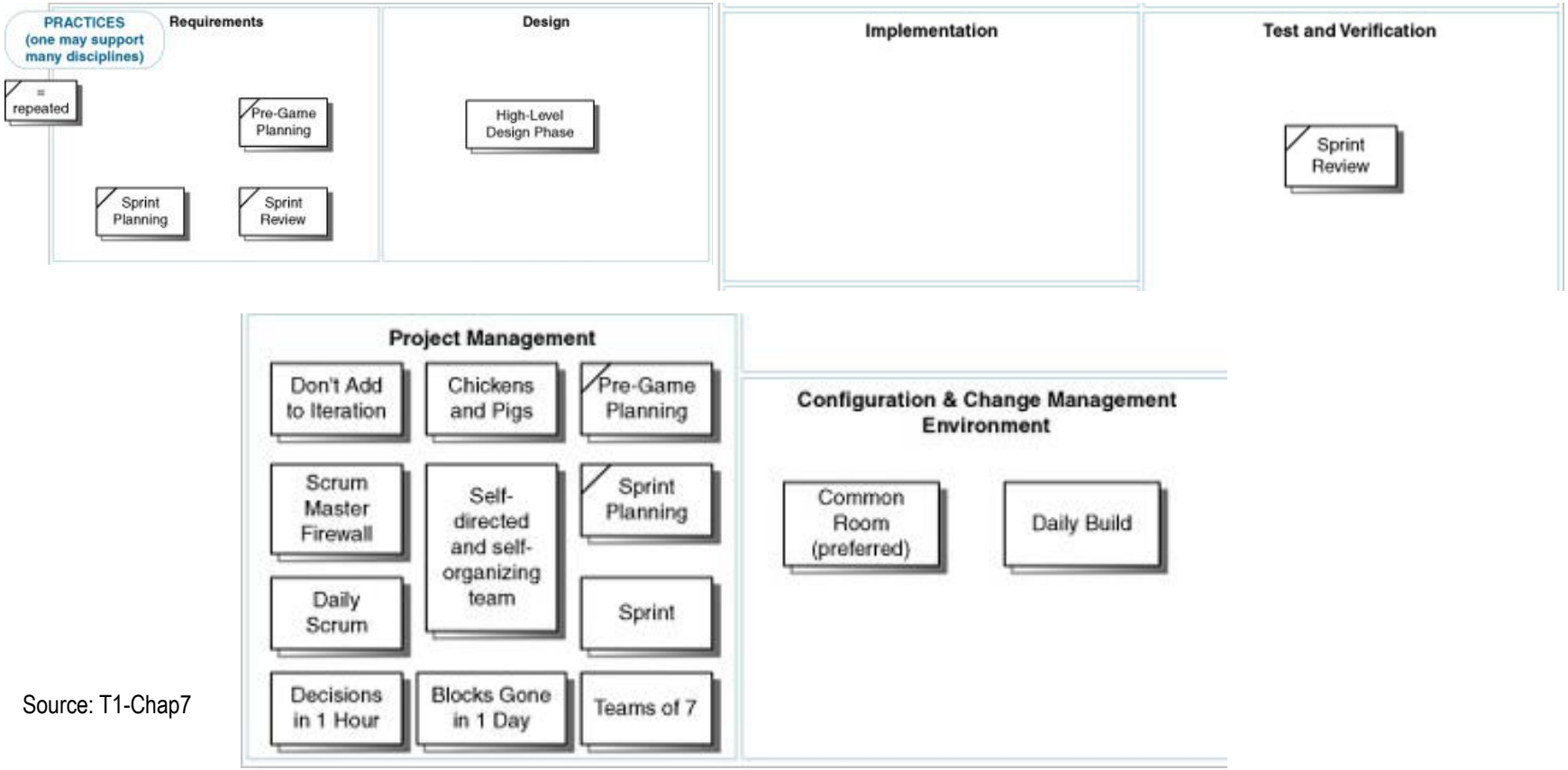
SCRUM Project Roles



Source courtesy: www.scrumstudy.com/blog/scrum-project-roles/



SCRUM Practices



Source: T1-Chap7

The Daily SCRUM Meeting –
The Heartbeat of the Project

- Daily Standup Meetings (1 – 7 Members, 15 – 20 minutes)
- Typical Questions Answered:
  - What have you done since the last Scrum?
  - What will you do between now and the next Scrum?
  - What is getting in the way (blocks) of meeting the iteration goals?
  - Any tasks to add to the Sprint Backlog? (missed tasks, not new requirements)
  - Have you learned or decided anything new, of relevance to some of the team members? (technical, requirements, …)
  - …
- Non-Team Members (‘chickens’) Can’t Ask Questions – they just listen
- White-board Meeting / Tele-Conf Call
- Shared Responsibility and Team Cohesiveness is maintained

SCRUM Artifacts

Product Backlog

Requirement	Num	Category	Status	Pri	Estimate
log credit payments to AR	17	feature	underway	5	2
process sale-simple cash scenario	232	use case	underway	5	60
slow credit payment approval	12	issue	not started	4	10
sales commission calculation	43	defect			
lay-away plan payments	321	enhance			
PDA sale capture	53	technology			
process sale-credit pmt scenario	235	use case			

Sprint Backlog

Task Description	Originator	Responsible	Status	Hours of work remaining					
				6	7	8	9	10	
				362	322	317	317	306	
Meet to discuss the goals and	JM	JM/SR	Completed	20	10	0	0	0	
Move Calculations out of	TL	AW	Not Started	8	8	8	8	8	
Get GEK Data		TN	Completed	12	0	0	0	0	
Analyse GEK Data - Title		GP	In Progress	24	20	30	25	20	
Analyse GEK Data - Parcel		TK	Completed	12	12	12	12	12	
Define & build Database		BR/DS	In Progress	80	80	75	60	52	

Source: T1-Chap7

SCRUM Values

("At the end, It's the People that Matters the Most")

- **Commitment**: The Team given Authority and Autonomy to decide; The Product Owners commits to Product Backlog; The Scrum Master commits not to introduce new work items till Iteration is complete
- **Focus**: The Scrum Master ensures the Team is not distracted; commits Resources and removes Roadblocks if any
- **Openness**: Product Backlogs and Daily Progress of Work Items are Visible to the entire Team
- **Respect**: Diversity of Individual Strengths and Weaknesses; facilitate Self-directed Teams; Teams empowered to seek/hire resources
- **Courage**: The Team has the courage to take Decisions adaptively; Management is supportive by empowering Teams

SCRUM Drawbacks (discountable however!)

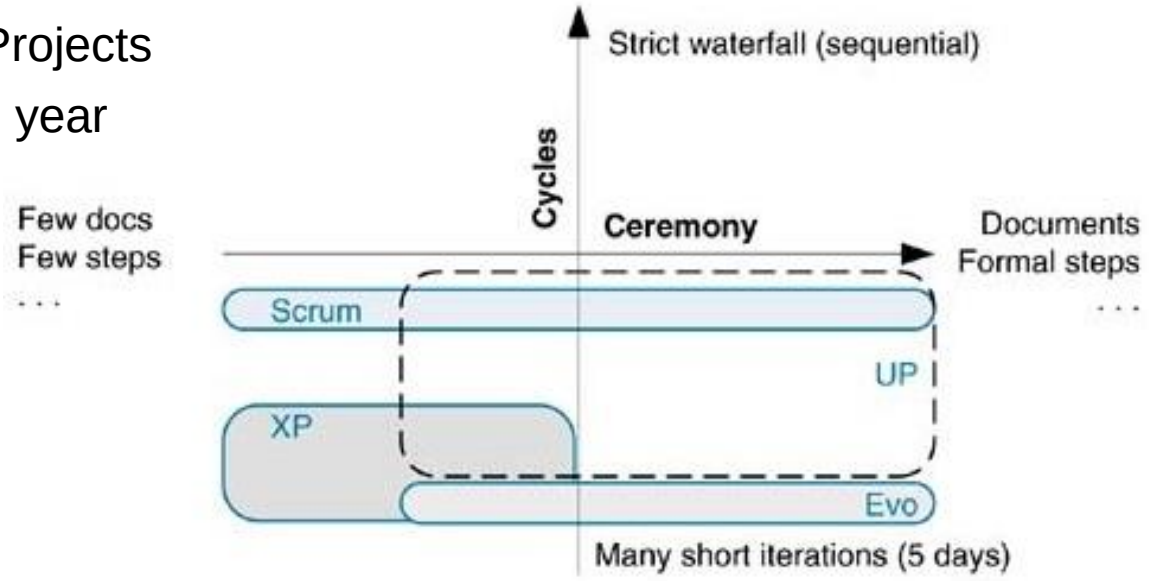
- **Active Engagement of the Customer**: the customer has to be continually involved in the project; however, while this can be seen as an advantage, it requires a lot of time and effort to manage the process
- **High Visibility may lead to Scope Creep**: as the Scrum methodology makes problems more visible, any problems arising at the end of each phase can also lead to "scope creep".
- **Small Teams losing Focus on the Big Picture**: while Scrum encourages small teams, it implies that larger teams may have to be broken down into smaller sizes which could result in the smaller teams loosing sight of the overall project focus.
- **Increase of Project Cost**: Scrum requires a certain level of training for all users which could increase the overall cost of the project.
- …

Extreme Programming (XP) – A Set of Skillful Practices:

➔➔

Extreme Programming (XP) – Core Values

- Communication, Simplicity, Feedback, Courage
- “Informal” (low on ‘ceremony’, usage of *Story Cards*)
- Small Teams (<10)
- Non-mission-critical Projects
- Delivery Schedule <1 year



Source: T1-Chap8

XP Lifecycle

EXPLORATION	PLANNING	ITERATIONS TO FIRST RELEASE	PRODUCTIONIZING	MAINTENANCE
Purpose: - Enough well-estimated story cards for first release. - Feasibility ensured.	**Purpose**: - Agree on date and stories for first release.	**Purpose**: - Implement a tested system ready for release.	**Purpose**: - Operational deployment	**Purpose**: - Enhance, fix. - Build major releases
Activities: - prototypes - exploratory proof of technology programming - story card writing and estimating	**Activities**: - Release Planning Game - story card writing and estimating	**Activities**: - testing and programming - Iteration Planning Game - task writing and estimating	**Activities**: - documentation - training - marketing - . . .	**Activities**: - May include these phases again, for incremental releases.

Source: T1-Chap8

XP Core Practices (12)

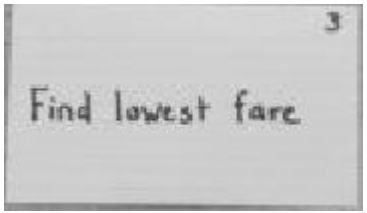
1. Planning Game
2. Small, Frequent Releases
3. System Metaphors
4. Simple Design
5. Testing
6. Frequent Refactoring
7. Pair Programming
8. Team Code Ownership
9. Continuous Integration
10. Sustainable Pace
11. Whole Team Work together
12. Coding Standards

Pitfalls of XP

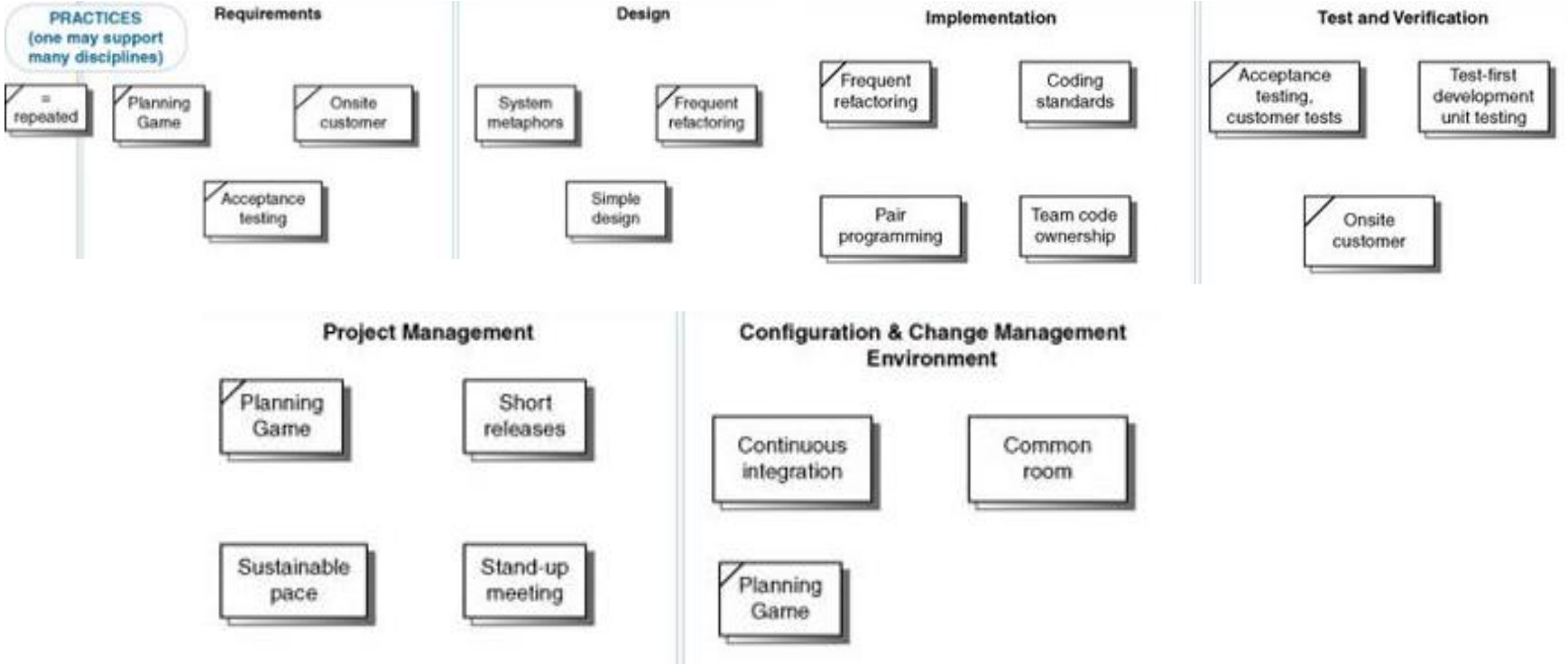
- Requires presence of Onsite Customer or his/her Proxy
- Relies on Informal oral understanding – difficult to ramp-up to speed new members or in large projects
- XP practices are highly interdependent and tightly coupled—we can’t be selective in any
- No ‘Standard’ means of documenting design
- Pair programming may not be favoured by all – *programmers are loners!*
- Lack of Focus on Architecture (relies on simple, if not fragile design and refactoring)

Extreme Programming (XP) Work Products

- Minimalist approach towards Requirements Gathering: Story Cards (sketches, visuals, post-it notes,…) representing *Features (not Use-cases/Scenarios)* – just cue-cards for discussion
- Task-list: containing Stories gathered for the Iteration (Task-card); Task-effort ~ 1-2 days
- Visible Wall Graphs for all to communicate with each other—progress of Stories, Test-cases, etc. using Simple Metrics



XP Practices



Source: T1-Chap8

XP – Other Common Practices/Values

- Embrace Change with Onsite Customer
- Volunteering rather than by Assignment
- Light Modeling (Just enough to get stared)
- Minimal Documentation
- Daily Metrics (few vitals) for Progress Tracking
- Incremental Infrastructure (no upfront investment)
- Daily Standup Meetings
- Simplicity – “Do the simplest thing that could possibly work.”
- Communication promoted through Pair-programming

“The practices are what you do. The values are how you decide if you are doing it right.”

Test Driven Development (TDD):
A Test-before-Code XP Practice

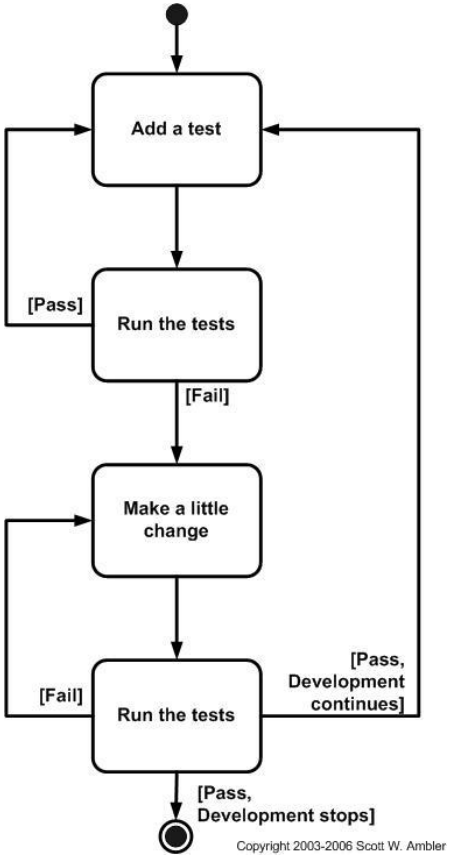
➔➔

Test-Driven Development

- Writing Test-cases before Coding (Unit Tests are written before writing the Code to be tested)
- Adoption of Short Iterative Development Cycle & Automated-Test Suites
- Eliminates “coder bias”

- “test with a purpose" - know why you are testing something and to what level it needs to be tested
- “achieve 100% coverage test” – every single line of code is tested
- “well-written unit-tests provide working specification of functional code” (code ~ documentation, tests ~ specifications)
- “proxies”: (unit-tests ~ design specifications, acceptance-tests ~ requirements)

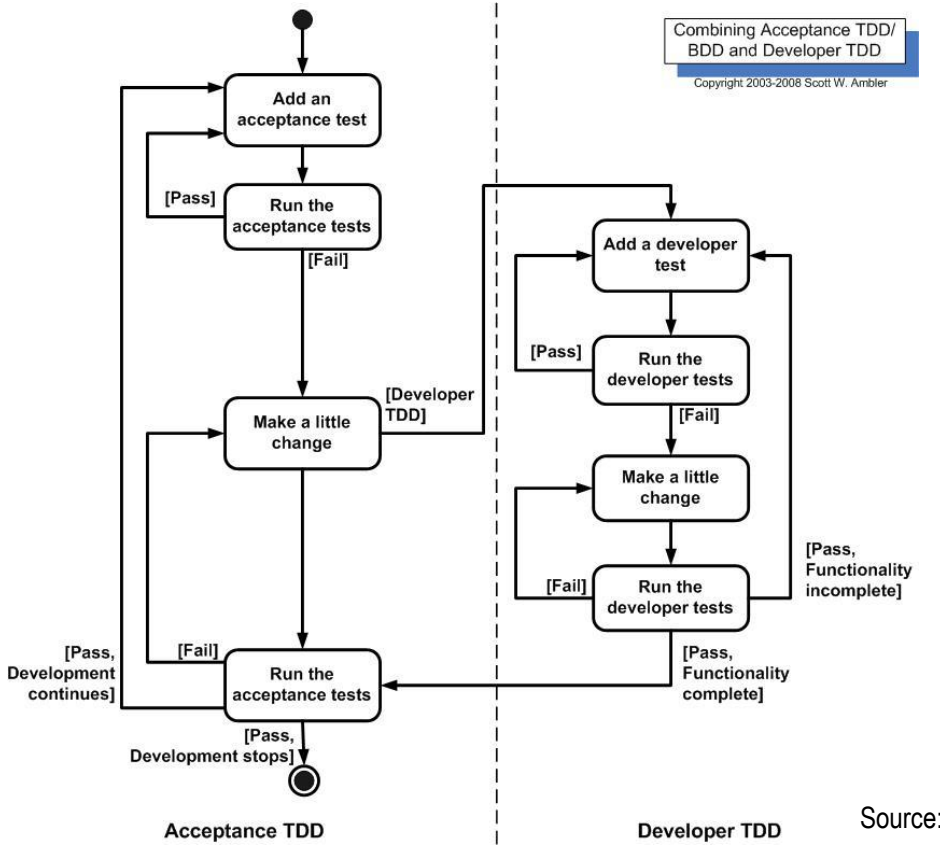
Test-Driven Development (TDD) =
Test-First Development (TFD) + Refactoring



- Evolutionary Approach to Development
- Write a Test-case first that fails before you write new functional Code; Coding evolves to fulfil those Test-cases, and then Refactor the Code
- Is TDD an Agile Requirements Capture technique or Programming Technique (than a Validation technique)?
- A Way to arrive at Clean Specifications!
- Does NOT replace traditional Testing, ensures effective Unit Testing

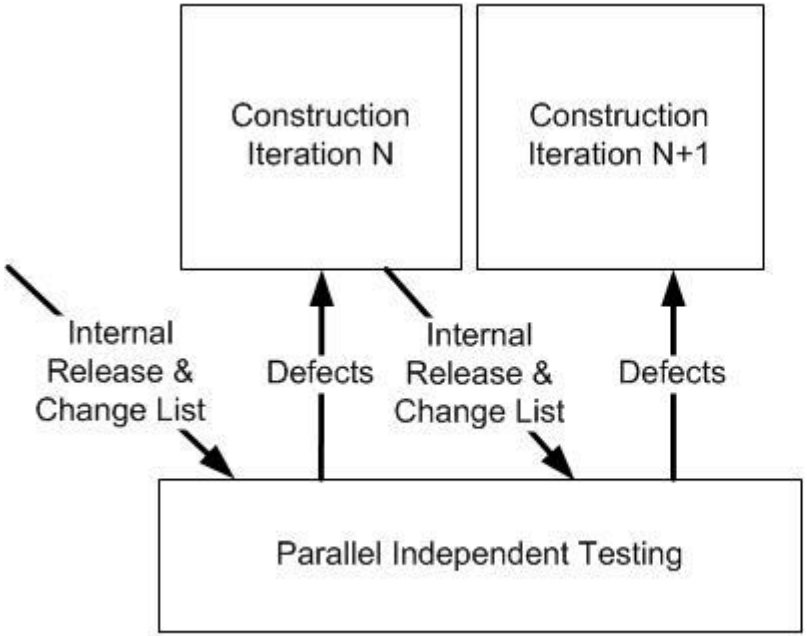
CUnit
DUnit (Delphi)
DBFit
DBUnit
DocTest (Python)
Googletest
HTMLUnit
HTTPUnit
JMock
JUnit
Moq
NDbUnit
NUnit
OUnit
PHPUnit
PyUnit (Python)
SimpleTest
TestNG
TestOoB (Python)
Test::Unit (Ruby)
VBUnit
XTUnit
xUnit.net

TDD: Acceptance TDD, Developer TDD



Source: http://agiledata.org/essays/tdd.html

TDD is NOT an End to Itself...



Confirmatory Testing
- Acceptance TDD
- Developer TDD
- Developer regression testing

Independent Testing:
- Pre-production integration testing
- Other expensive/skillful testing

Agile Methodologies - Summary

- SCRUM is a Process in Agile Methodology which is a creative combination of an Iterative and Incremental Methods; it is prescriptive in nature and has well-defined Roles
- XP – set of Practices that take Programming to the Extreme, i.e. just enough, just-in-time with minimalist approach; relies heavily on people
- TDD – an XP technique turns traditional Code Development upside-down, i.e. write test first and then write (just enough) code to fulfil that test and move-on to write the next test, then code, etc.; relies on availability of automated test tools

Thank You