CMPT 276 Midterm Review

Dr. Jack Thomas
Simon Fraser University
Fall 2020

Midterm Format

- 1 Hour, timed, one attempt – no resubmission.
- Available on SFU Canvas on Wednesday, October 28th.
- Must be completed that day during the scheduled class time.
- If that isn’t possible, **NOTIFY ME ASAP!**

Security & Academic Integrity

- Canvas logs your IP address.
- By necessity, the midterm is open book, and the questions reflect this.
- All standard rules about plagiarism and citing apply here as they do in your assignments.

- Don’t share answers!

Content Overview

- The first eight units of the course, found on the Notes page of the course website.
- At least one question drawing on your knowledge of Java and Android.
- Some parts of the assignments (how to use Git), but not others (the requirements document in assignment 3).

How to Review Course Topics

1. Consult your notes!
2. Review the lecture slides.
3. Follow up with the recorded lectures where clarification would help.
4. Check out the sample assignment solutions and Dr. Fraser’s videos.
5. Maybe discuss them with your new project groupmates?

1. Introduction to Software Engineering

- The definition of Software Engineering
- Objectives of Software Engineering
- The Software Process Activities:
  - Specification
  - Development
  - Validation
  - Evolution

1. Introduction to Software Engineering

- Essential Attributes of Good Software
  - Maintainability
  - Dependability & Security
  - Efficiency
  - Acceptability
- Types of Applications?
- General Software Issues
  - Diverse types of systems.
  - Changing environments.
  - Security and trust.
- Diversity of Needs

2. Version Control

- Motivation
- Understanding the Topology
  - Working Directory, Local Repo, Remote Repo
- The Basic Workflow
  - Setting Up
  - Making Changes
  - Pulling Changes from Others
  - Merging Changes Together

2. Version Control

- Merge vs. Lock
- The Purpose of a Tag
- Good Etiquette:
  - Clear comments, frequent commits, don’t break the build.
- Issues & Branches
- Terminal Commands? SSH keys?

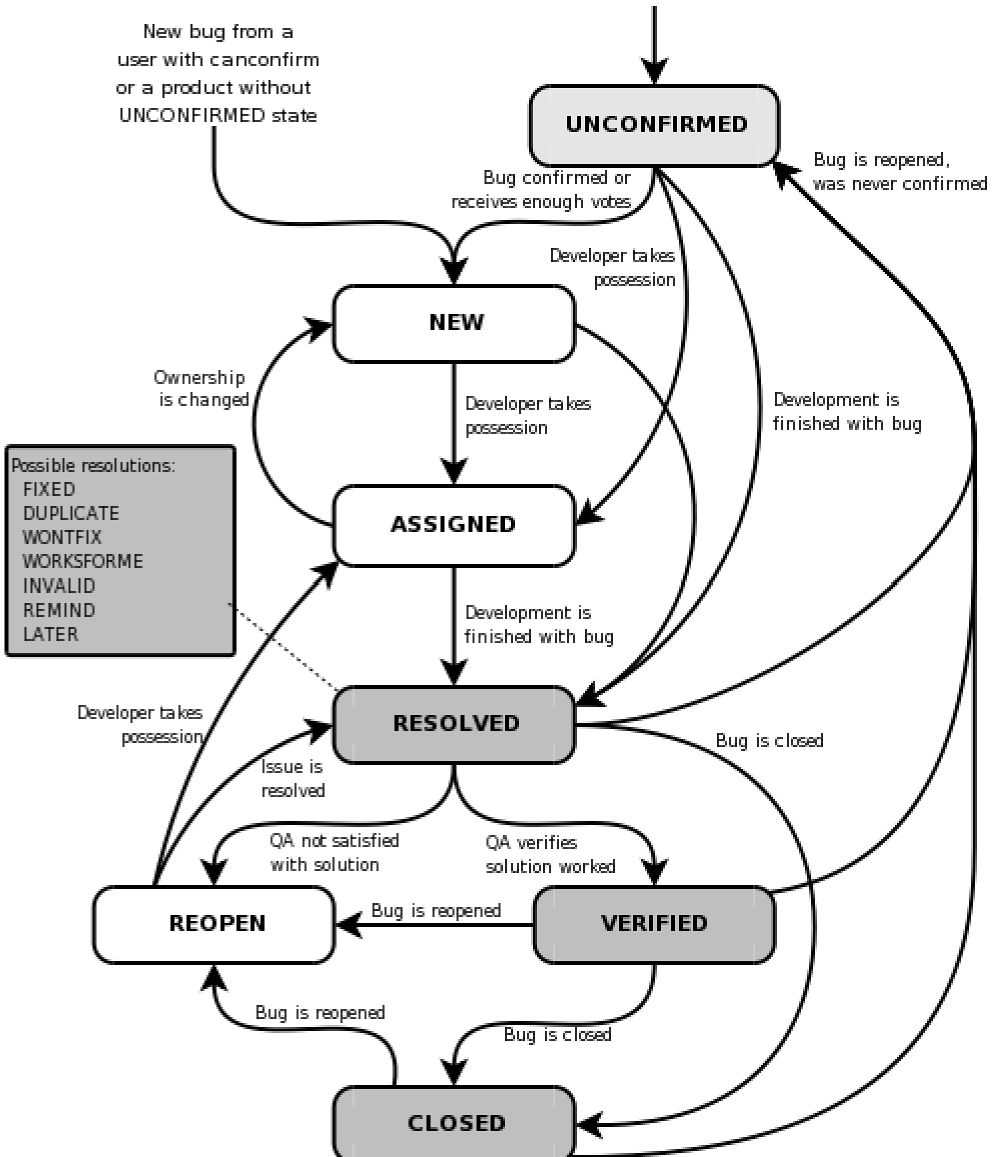
3. Testing

- Acceptance and Unit Testing
- White Box vs. Black Box
- How to write a JUnit Test

3. Testing

- Partition Testing and Guideline-Based Testing
  - Equivalence Classes
- Code Coverage and Test Quality
- Writing Good Bug Reports

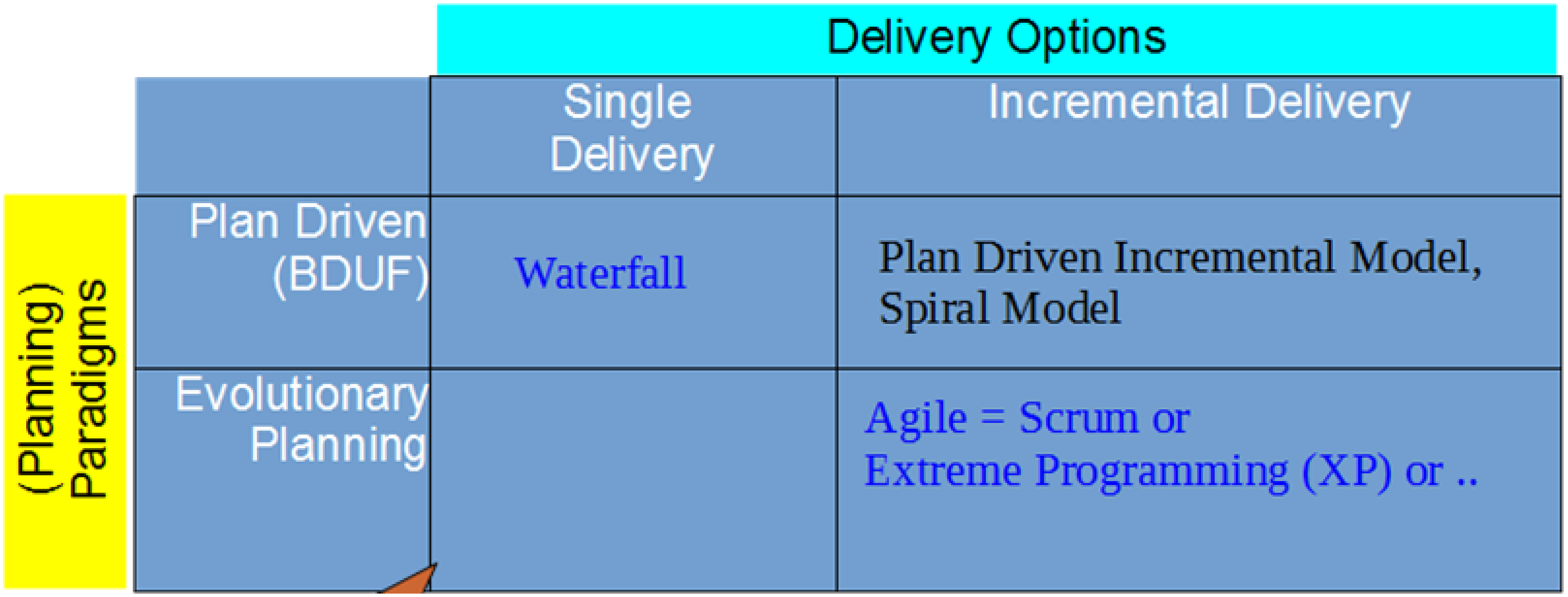
Imagine if I made you fill this out


New bug from a user with canconfirm or a product without UNCONFIRMED state
UNCONFIRMED
Bug is reopened, was never confirmed
Bug confirmed or receives enough votes
Developer takes possession
NEW
Ownership is changed
Developer takes possession
Development is finished with bug
Possible resolutions: FIXED DUPLICATE WONTFIX WORKSFORME INVALID REMIND LATER
ASSIGNED
Development is finished with bug
Developer takes possession
RESOLVED
Bug is closed
Issue is resolved
QA not satisfied with solution
QA verifies solution worked
REOPEN
Bug is reopened
VERIFIED
Bug is reopened
Bug is closed
CLOSED

4. Software Processes

- The idea behind a software process model.
- The previously-introduced four activities:
  - Specification
  - Design and Implementation
  - Validation
  - Evolution
- How they combine to produce different models (one after the other, interleaved, in parallel).

4. Software Processes



		Delivery Options	
		Single Delivery	Incremental Delivery
(Planning) Paradigms	Plan Driven (BDUF)	Waterfall	Plan Driven Incremental Model, Spiral Model
	Evolutionary Planning		Agile = Scrum or Extreme Programming (XP) or ..

4. Software Processes

- Waterfall Model
  - Plan-driven, documentation-heavy
- Agile
  - Planning-driven, lightweight
- Incremental Development
  - Frequent releases, code rot
- Refactoring

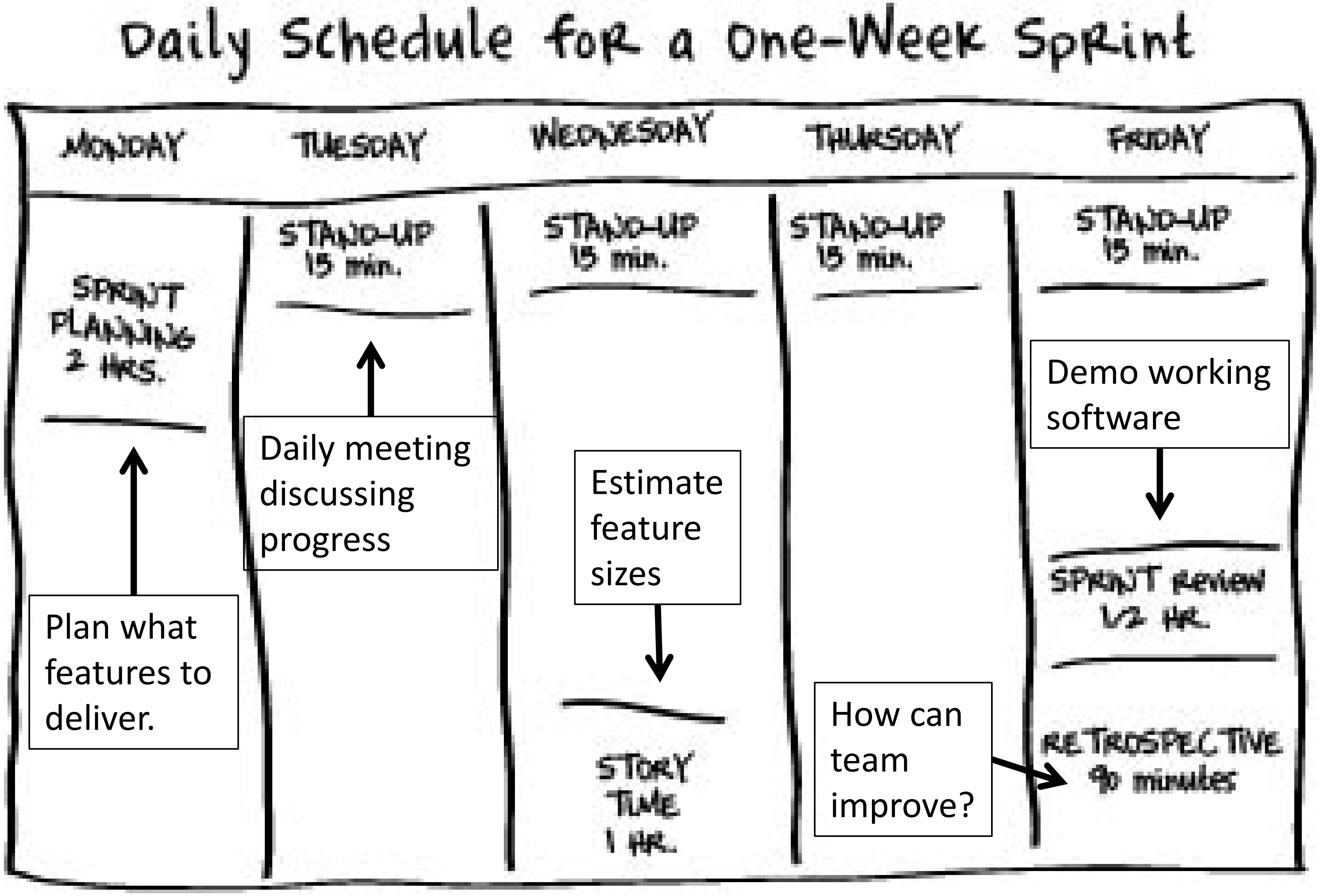
5. Change Risk

- Understanding the cost of change, change avoidance, and change tolerance.
- Prototypes
- Incremental Development & Delivery

6. SCRUM

- Backlogs, Iterations (“Sprints”), Ceremonies
- Scrum Roles:
  - Scrum Master
  - Product Owner
  - Repository Manager
  - Team Member

6. SCRUM


Daily Schedule for a One-Week Sprint
MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRIDAY
SPRINT PLANNING 2 HRS.
STAND-UP 15 min.
STAND-UP 15 min.
STAND-UP 15 min.
STAND-UP 15 min.
STORY TIME 1 HR.
SPRINT REVIEW 1/2 HR.
RETROSPECTIVE 90 minutes
Plan what features to deliver.
Daily meeting discussing progress
Estimate feature sizes
How can team improve?
Demo working software

7. Agile

- Plan-Driven vs. Planning-Driven
- Values (Inspect & Adapt)
  - Individuals and Interactions
  - Working Software
  - Customer Collaboration
  - Responding to Change
- Principles
  - Customer Involvement
  - Incremental Delivery
  - People, not Processes
  - Embrace Cahnge
  - Maintain Simplicity

7. Agile

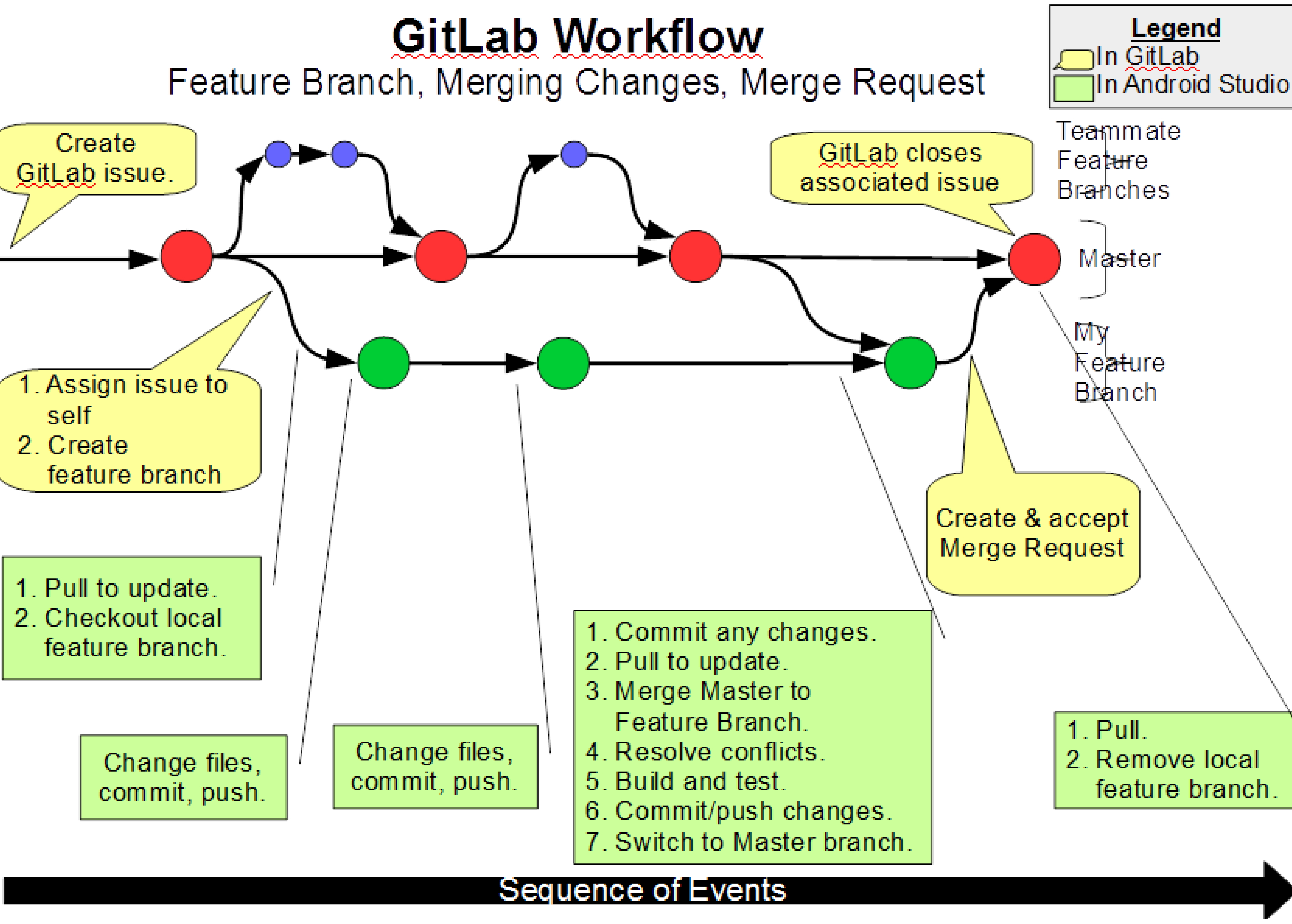
- Applicability
- Benefits & Drawbacks
- Extreme Programming
  - Pair programming
  - Test-Driven Development

8. The GitLab Workflow

- Creating Issues on GitLab
- Managing Branches for Features and Bugs
- How to handle a Merge Request

GitLab Workflow

Feature Branch, Merging Changes, Merge Request



Android & Java

- Familiar with the basic syntax.
- Understand Model/View separation.
- Apply Object-Oriented Design and Encapsulation.
- Know how to use Contexts, Intents, SharedPreferences.
- Not going to ask about .xml or the graphical interface parts of Android Studio.