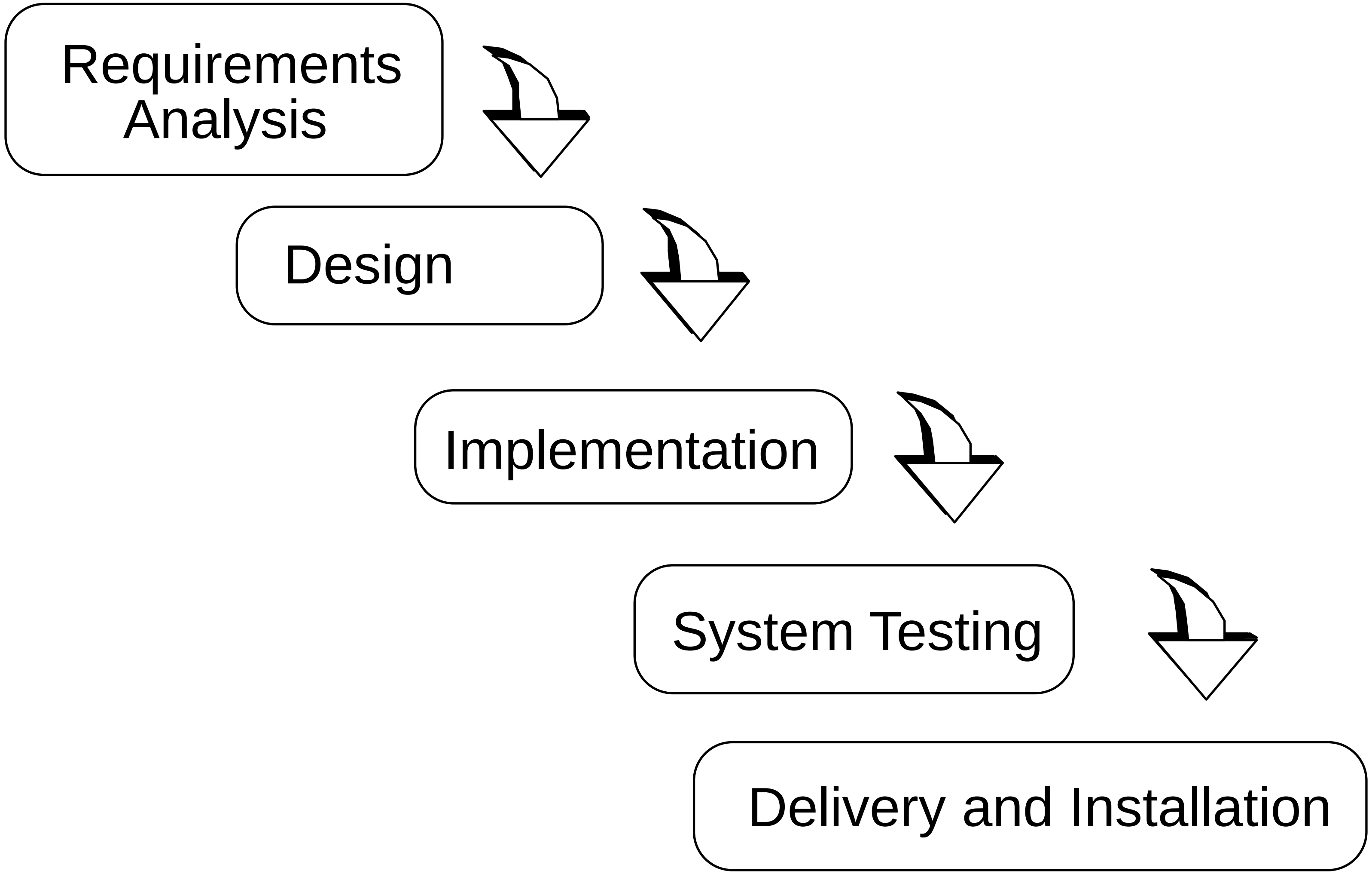
CSCI 4050/6050
Software Engineering

Software

Project Management

Dr. Eman Saleh eman.saleh@uga.edu

How It Should Proceed (Ideally)



Requirements Analysis

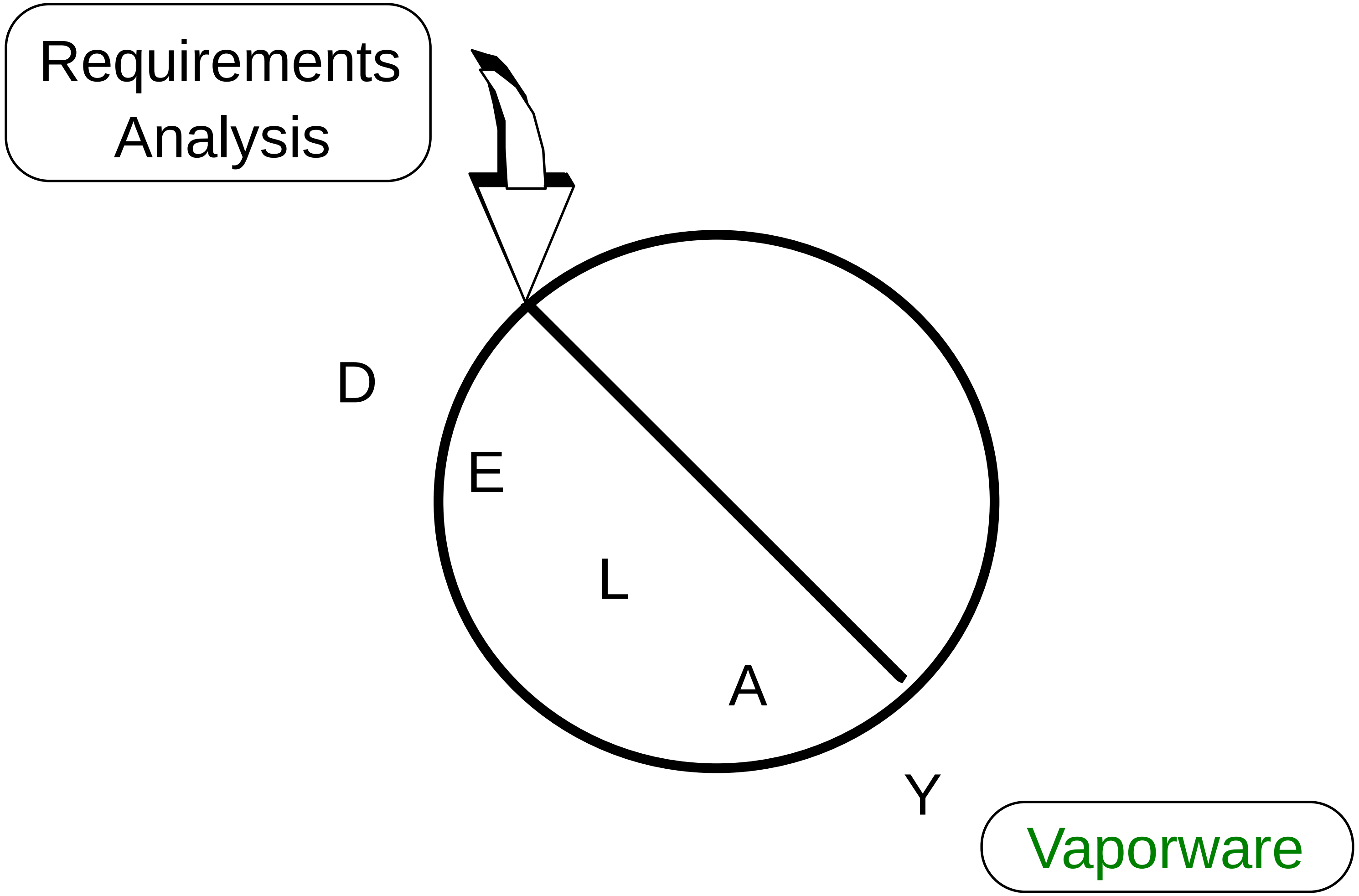
Design

Implementation

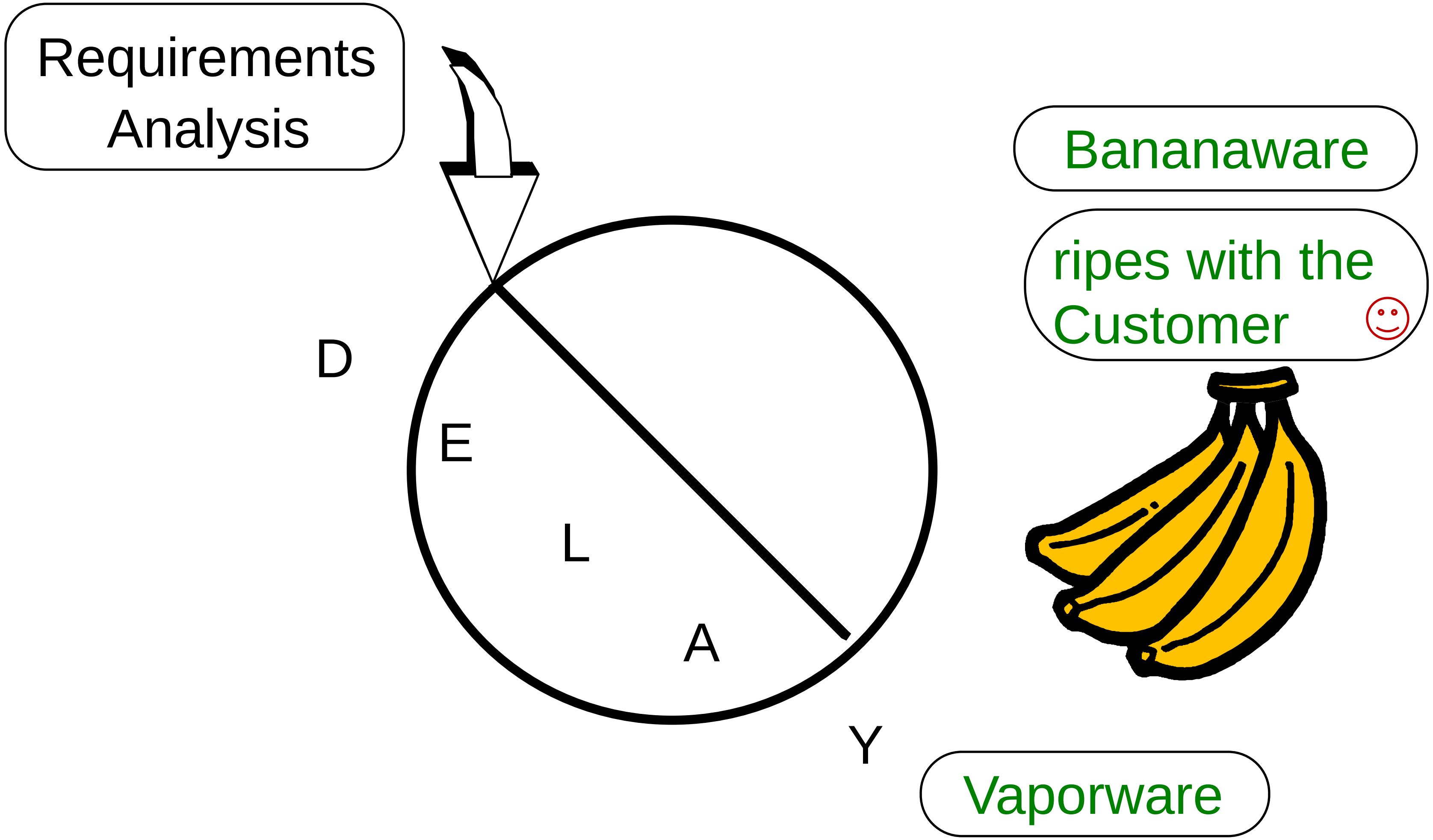
System Testing

Delivery and Installation

What frequently happens...



How it often goes



Requirements Analysis

Bananaware

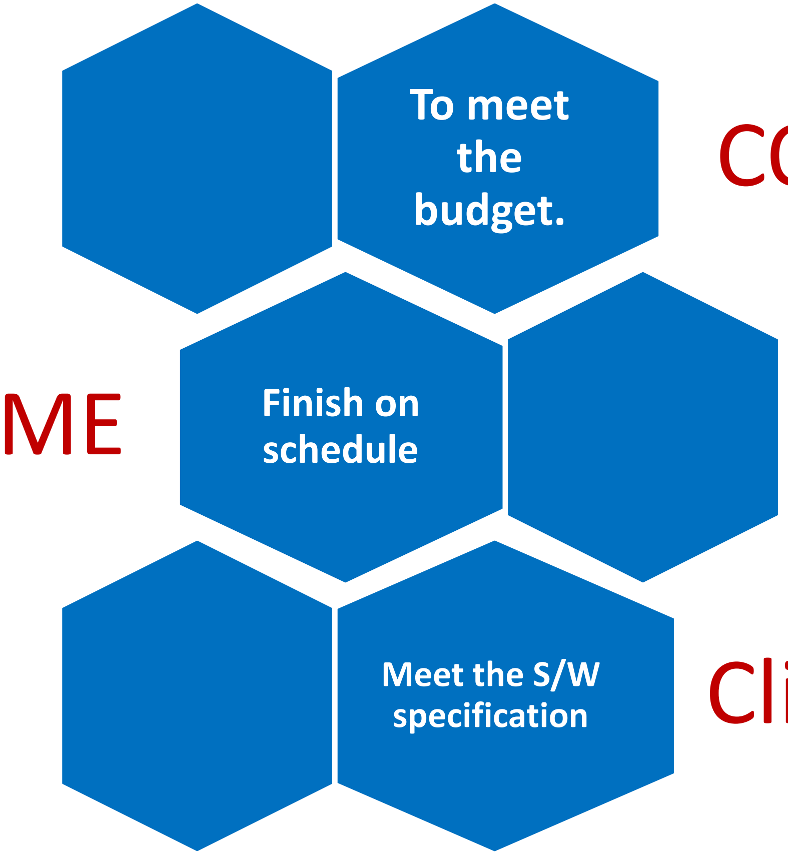
ripes with the Customer ☺

Vaporware

Laws of Project Management

- Projects progress quickly until they are 90% complete
  - Then they remain at 90% complete forever
- If the project content is allowed to change freely, the rate of change will exceed the rate of progress
- Project teams detest progress reporting because it manifests their lack of progress
- Murphy's laws:
  - "When things are going well, something will go wrong"
  - "When things just can't get worse, they will"
  - "When things appear to be going better, you have overlooked something."

Each project has 3 interrelated goals



To meet the budget.

COST

TIME

Finish on schedule

Meet the S/W specification

Client

Project Definition

- A project is an undertaking, limited in time, to achieve a set of goals that require a concerted effort
- A project includes:
  - A set of deliverables to a client
  - A schedule
  - Technical and managerial activities required to produce and deliver the deliverables
  - Resources consumed by the activities (people, budget)
- Focus of project management:
  - Administration of resources
  - Accountability maintenance
  - Reacting to change
  - Making sure the goals are met.

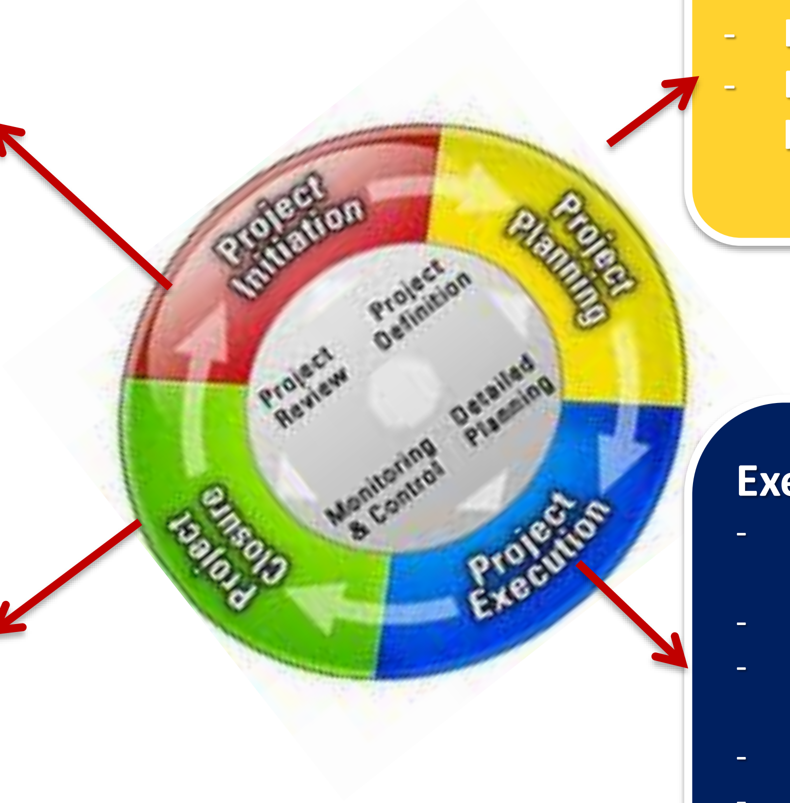
Project Phases

Initiation:

- Assemble the team.
- Define the expectations.
- Define the scope.

Planning:

- Identify the tasks.
- Develop the schedule and budget.



Closing:

- Deliver and submit the project.
- Write a report representing the evaluation of the project and the development staff.

Execution:

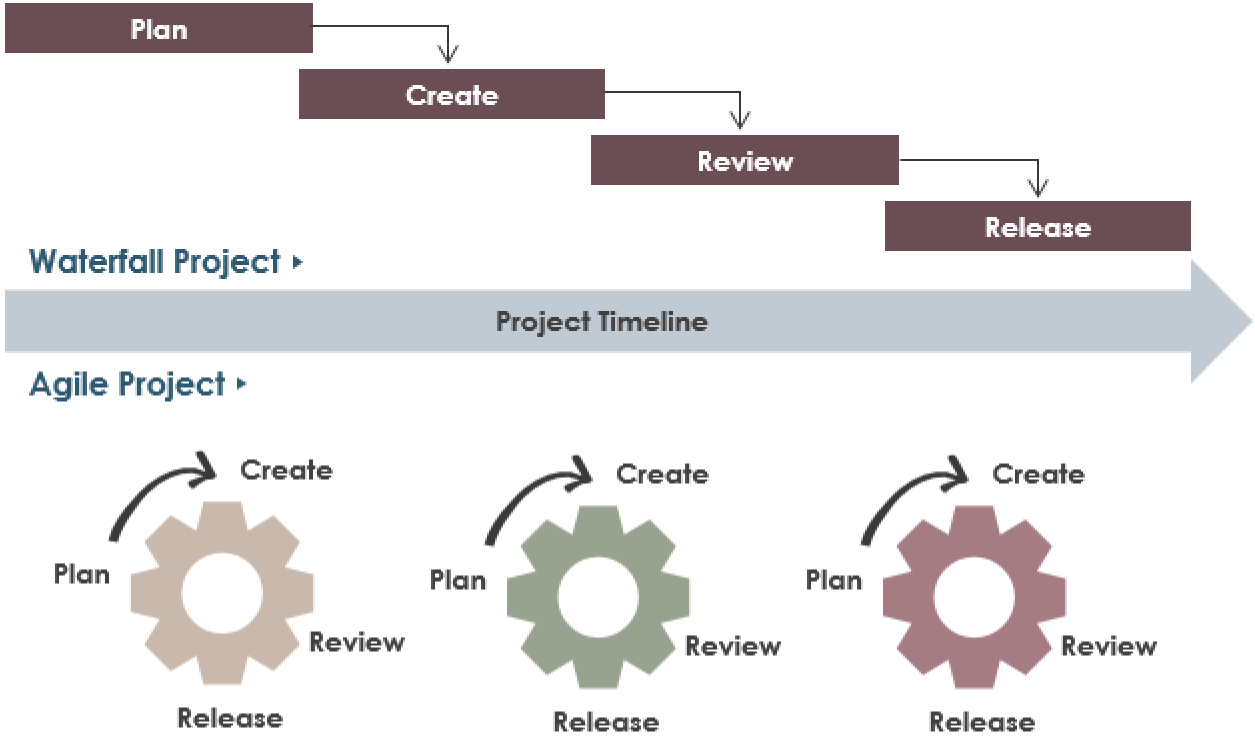
- Develop the project to accomplish the goal.
- Controlling:
- Monitor the changes to the project.
- Make corrections.
- Adjust the schedule.
- Leading the team staff.

Traditional Project Management

The traditional Project Management (waterfall) approach is linear where all the phases of a process occur in a sequence.

The entire project is planned upfront without any scope for changing requirements, such as Waterfall. The approach depends on predictable tools and predictable experience.

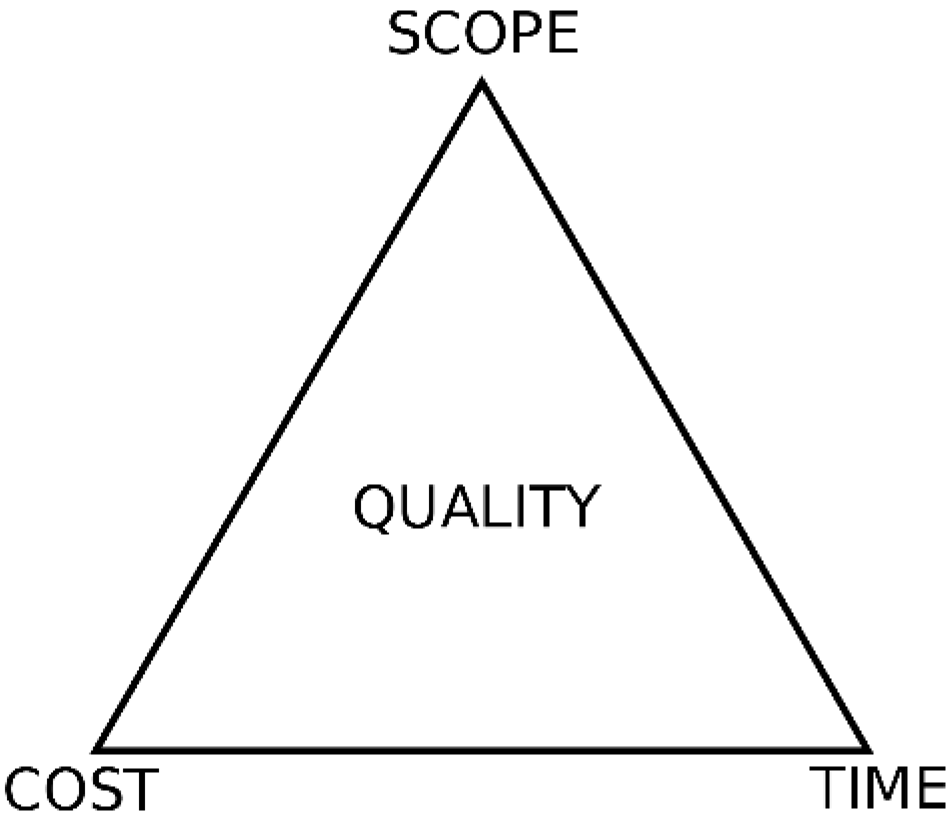
- Agile project management is iterative and aims at constantly incorporating user feedback and continuous releases with every iteration of software development. Every task output is a product you’re selling to stakeholders. Team and work structures are designed around creating things that are directly useful to the customer or client.



Picture from visualParadigm.com

Iron Triangle

- It contends that:
- The quality of work is constrained by the project's budget, deadlines and scope (features).
- The project manager can trade between constraints.
- Changes in one constraint necessitate changes in others to compensate or quality will suffer.



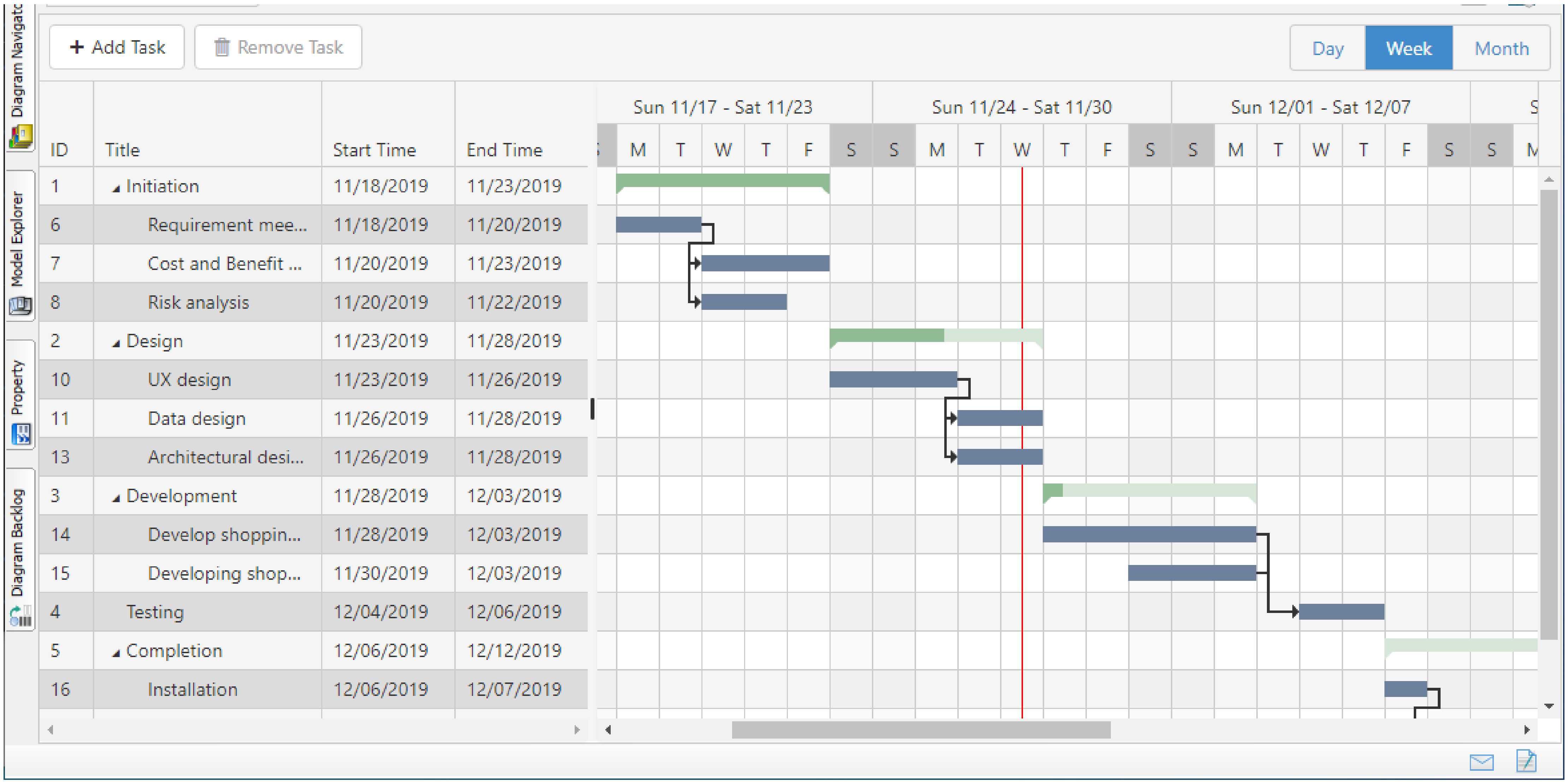
The Project Management Triangle is considered insufficient as a model of project success because it omits crucial dimensions of success including impact on stakeholders, learning and user satisfaction.

Iron Triangle in Plan-Driven and Agile Sense

- Plan-driven: Completing the scope within budget and time, may reduce the quality.

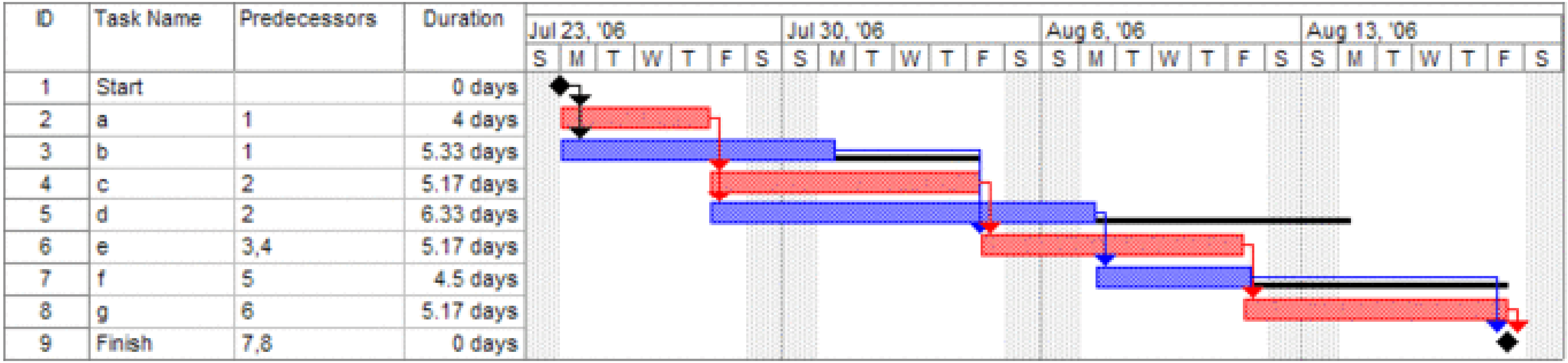
- Agile: More focus on Quality even if it would lead to reducing the scope.

Gant chart is used for project scheduling



ID	Title	Start Time	End Time
1	Initiation	11/18/2019	11/23/2019
6	Requirement mee...	11/18/2019	11/20/2019
7	Cost and Benefit ...	11/20/2019	11/23/2019
8	Risk analysis	11/20/2019	11/22/2019
2	Design	11/23/2019	11/28/2019
10	UX design	11/23/2019	11/26/2019
11	Data design	11/26/2019	11/28/2019
13	Architectural desi...	11/26/2019	11/28/2019
3	Development	11/28/2019	12/03/2019
14	Develop shoppin...	11/28/2019	12/03/2019
15	Developing shop...	11/30/2019	12/03/2019
4	Testing	12/04/2019	12/06/2019
5	Completion	12/06/2019	12/12/2019
16	Installation	12/06/2019	12/07/2019

Gant chart is used for project scheduling

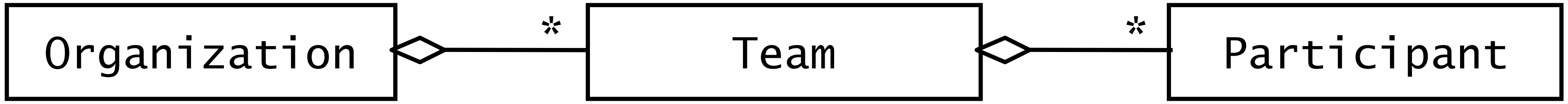


ID	Task Name	Predecessors	Duration
1	Start		0 days
2	a	1	4 days
3	b	1	5.33 days
4	c	2	5.17 days
5	d	2	6.33 days
6	e	3,4	5.17 days
7	f	5	4.5 days
8	g	6	5.17 days
9	Finish	7,8	0 days

Project Organization

- A project organization defines the relationships among resources, in particular the participants, in a project

- A project organization should define
  - Who decides (decision structure)
  - Who reports their status to whom (reporting structure)
  - Who communicates with whom (communication structure)

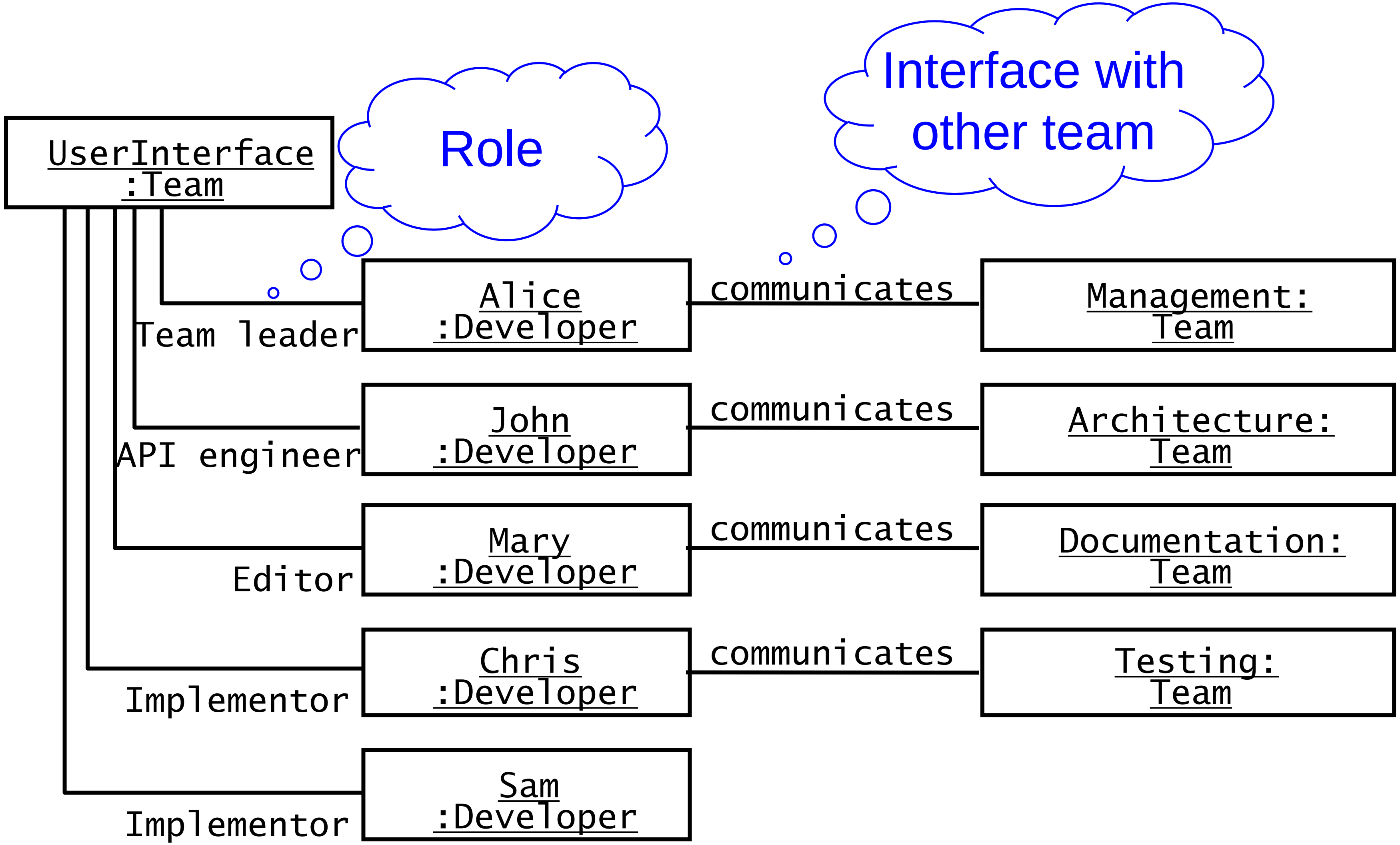


Reporting vs. Communication

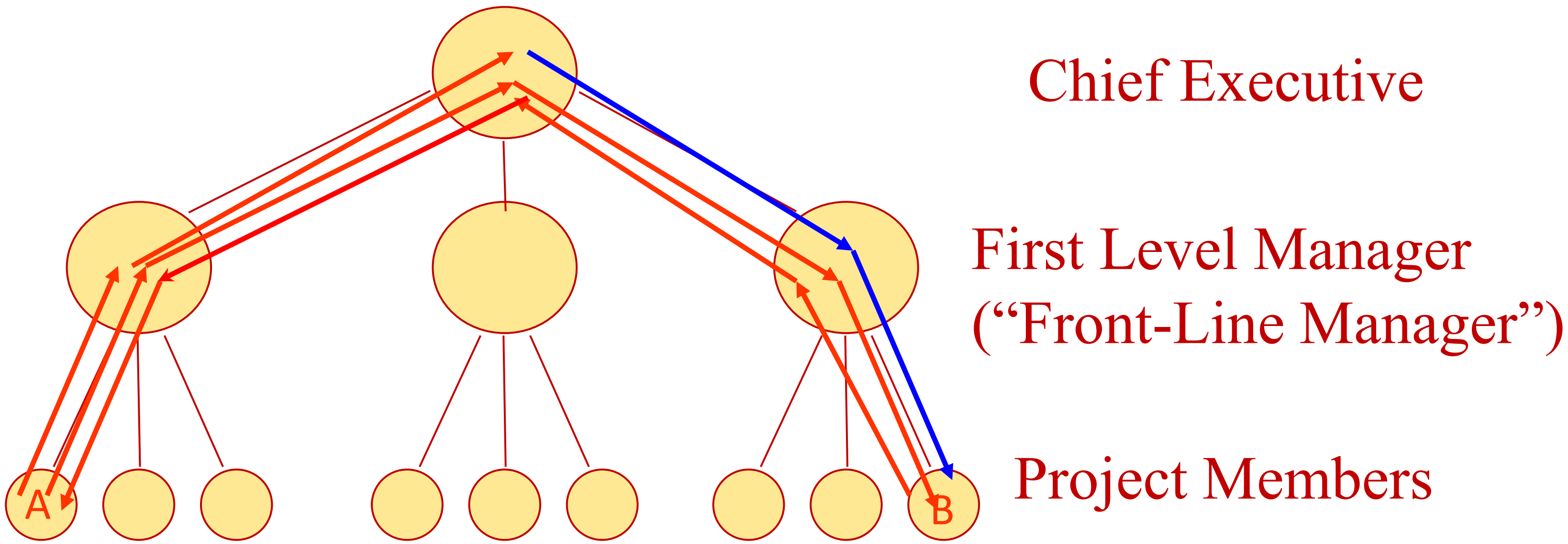
- Reporting supports project management in tracking project status
  - What work has been completed?
  - What work is behind schedule?
  - What issues threaten project progress?

- Reporting along the hierarchy is not sufficient when two teams need to communicate
  - Effective communication structure is needed
  - A participant from each team is responsible for facilitating communication between both teams
  - Such participants are called **liaison**

Example of a Communication Structure



Hierarchical Project Organization

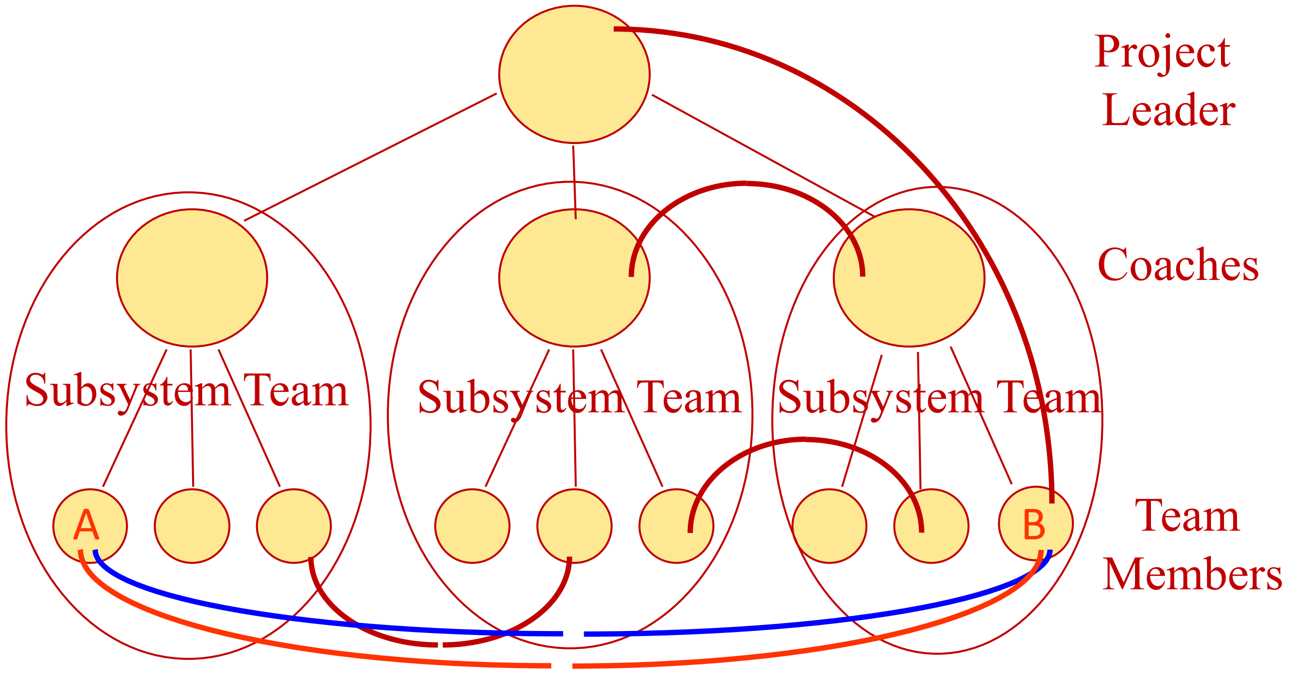


A wants to talk to B: Information Flow

A wants to make sure B does a certain change: Controlflow

Basis of organization:
Complicated information and control flow
across hierarchical boundaries

Peer-To-Peer Communication



A wants to talk to B: Simple Information Flow

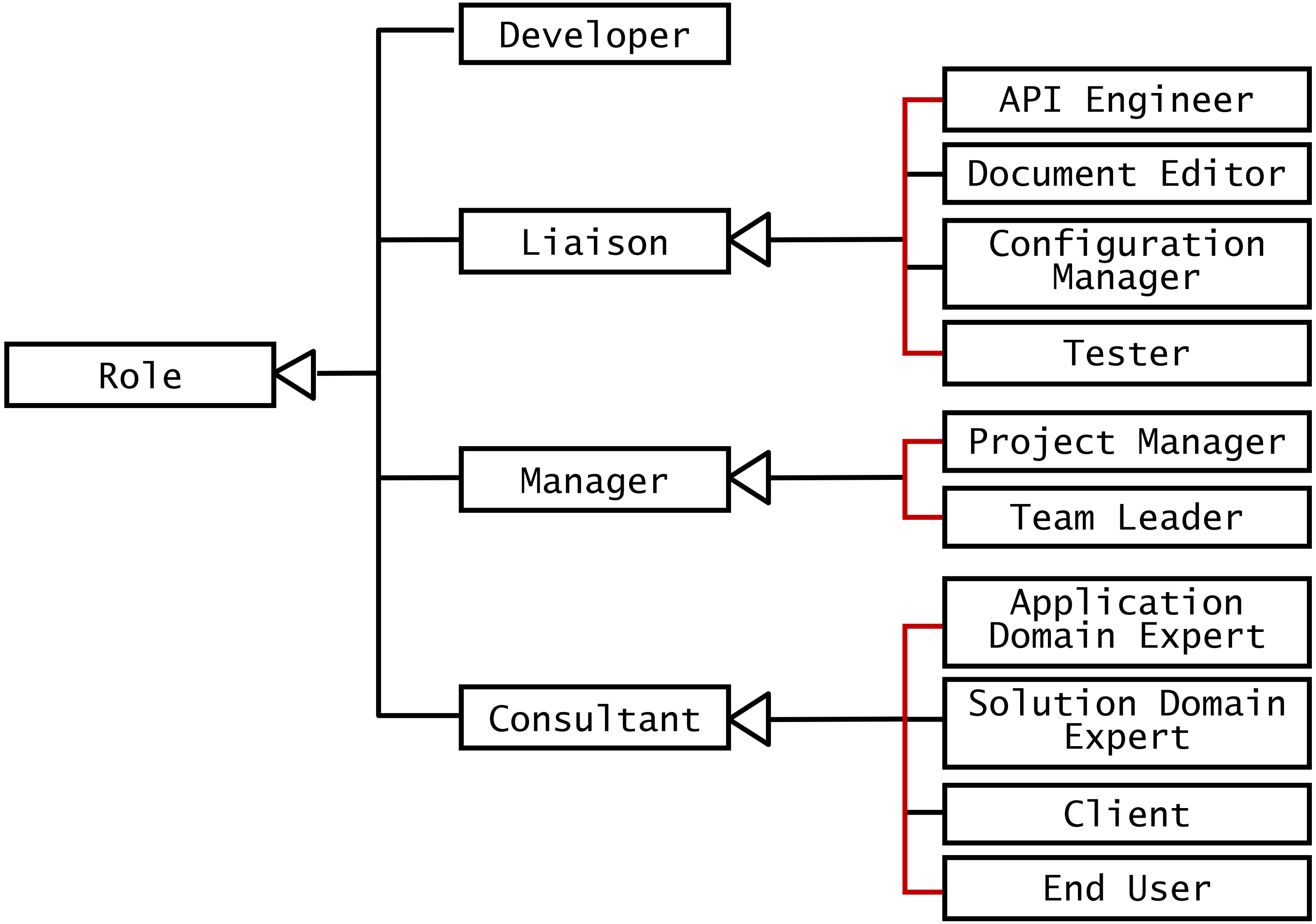
A wants to make sure B does a certain change: Simple Controlflow

Basis of organization:
Nonlinear information flow across dynamically formed units

Roles

- A **role** defines a set **responsibilities** (“to-dos”); Examples:
  - Role: Manager/Leader
    - Form team and assign roles
    - Create schedule of meetings, deliverable deadlines
    - Control the project progress
  - Role: System architect
    - Ensure consistency in design decisions, system architecture, subsystem interfaces
    - Formulate system integration strategy
  - Role: Developer
    - Implement/write code
  - Role: Tester (QA)
    - Write tests
    - Report failures
    - Check if bug fixes address a specific failure
  - Role: Liaison
    - Facilitate communication between two teams.

Types of Roles in Software Organizations



Possible Mappings of Roles to Participants

- One-to-One
  - Ideal but rare
- Many-to-Few
  - Each project member assumes several "hats"
  - Danger of over-commitment
  - Need for load balancing
- Many-to-"Too-Many"
  - Some people don't have significant roles
  - Lack of accountability
  - Loosing touch with project

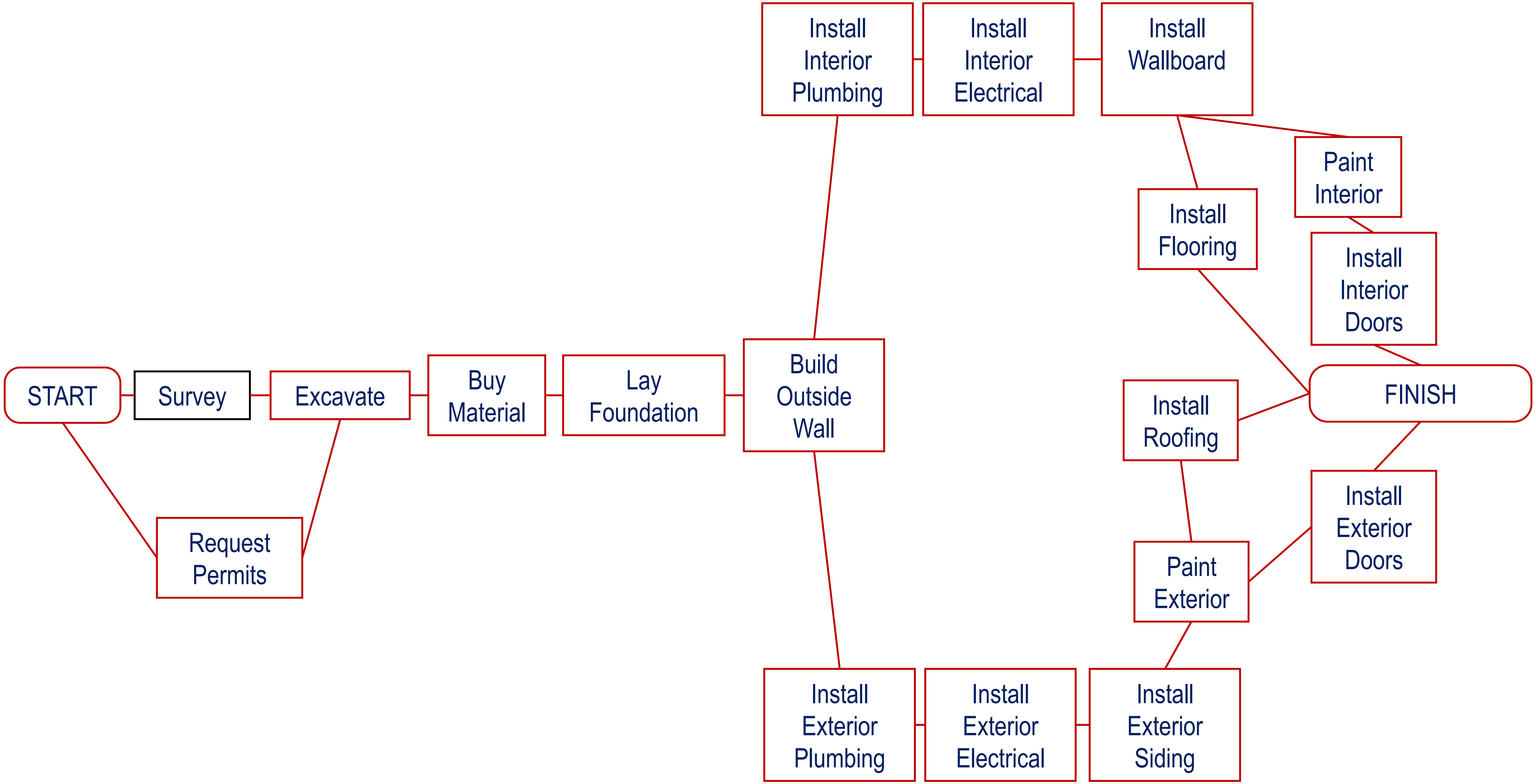
Task

- A **task** describes the smallest amount of work tracked by management
- Typically 3-10 working days effort

- Tasks descriptions
  - Role
  - Work product
  - Start date
  - Planned duration
  - Required resources

Example

Example: Tasks for Building a House



Tasks and Work Packages

- A task is specified by a work unit (package)
  - Description of work to be done
  - Preconditions for starting, duration, required resources
  - Work products to be produced, acceptance criteria for it
  - Risks involved
- A task must have completion criteria
  - Includes the acceptance criteria for the work products (deliverables) produced by the task.

Work Products

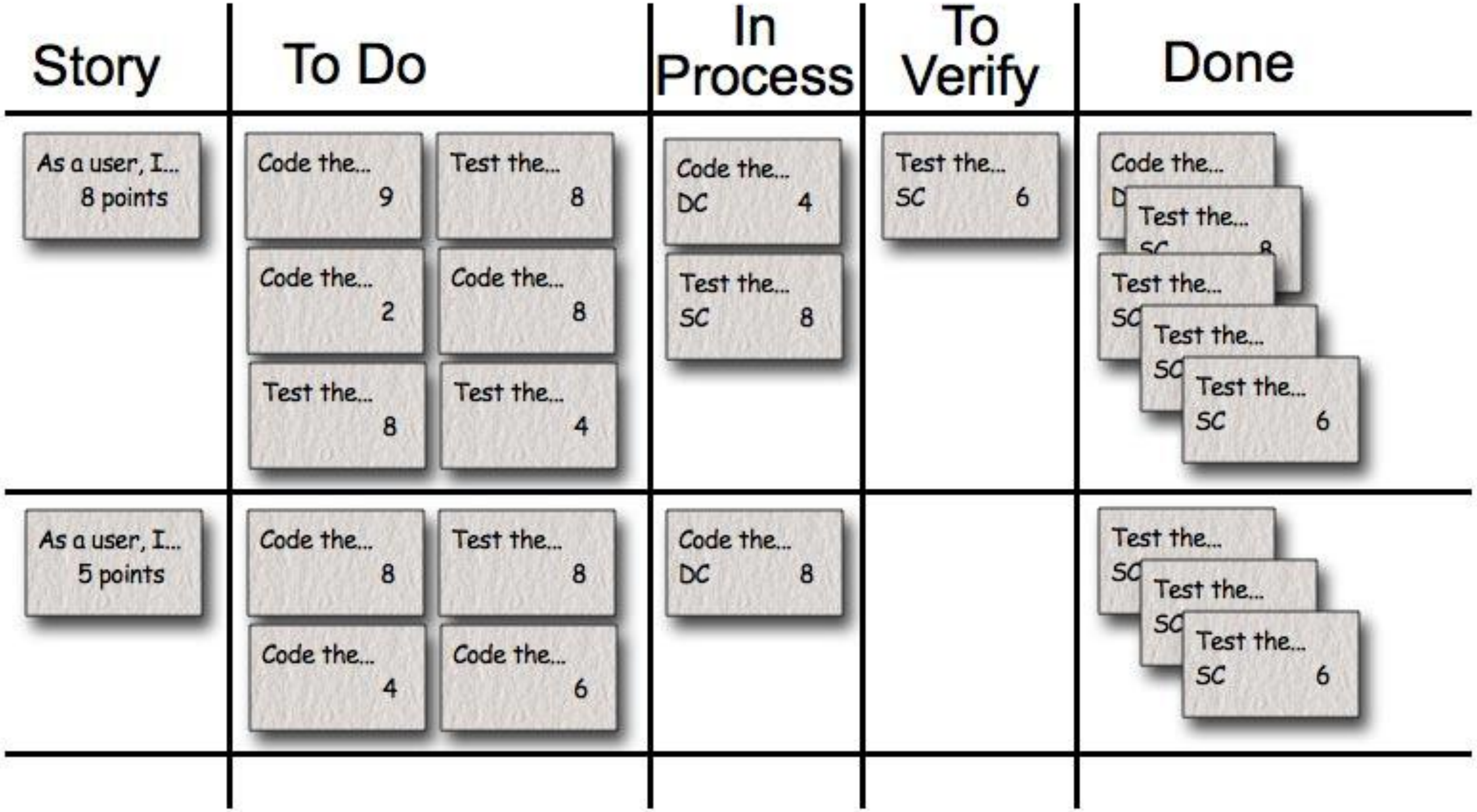
- A work product is a visible outcome of a task
- Examples
  - A document
  - A review of a document
  - A presentation
  - A model (of a domain/subsytem/architecture)
  - A unit of code
  - A test report
- Work products delivered to the customer are called **deliverables**.

Task Sizes

- Tasks are decomposed into sizes that allow monitoring
  - You may not know how to decompose the problem into tasks at first
  - Depends on the nature of work and how well task is understood.
- Finding the appropriate size is crucial
  - To-do lists from previous projects
  - Each software development activity identifies more tasks and modifies existing ones.

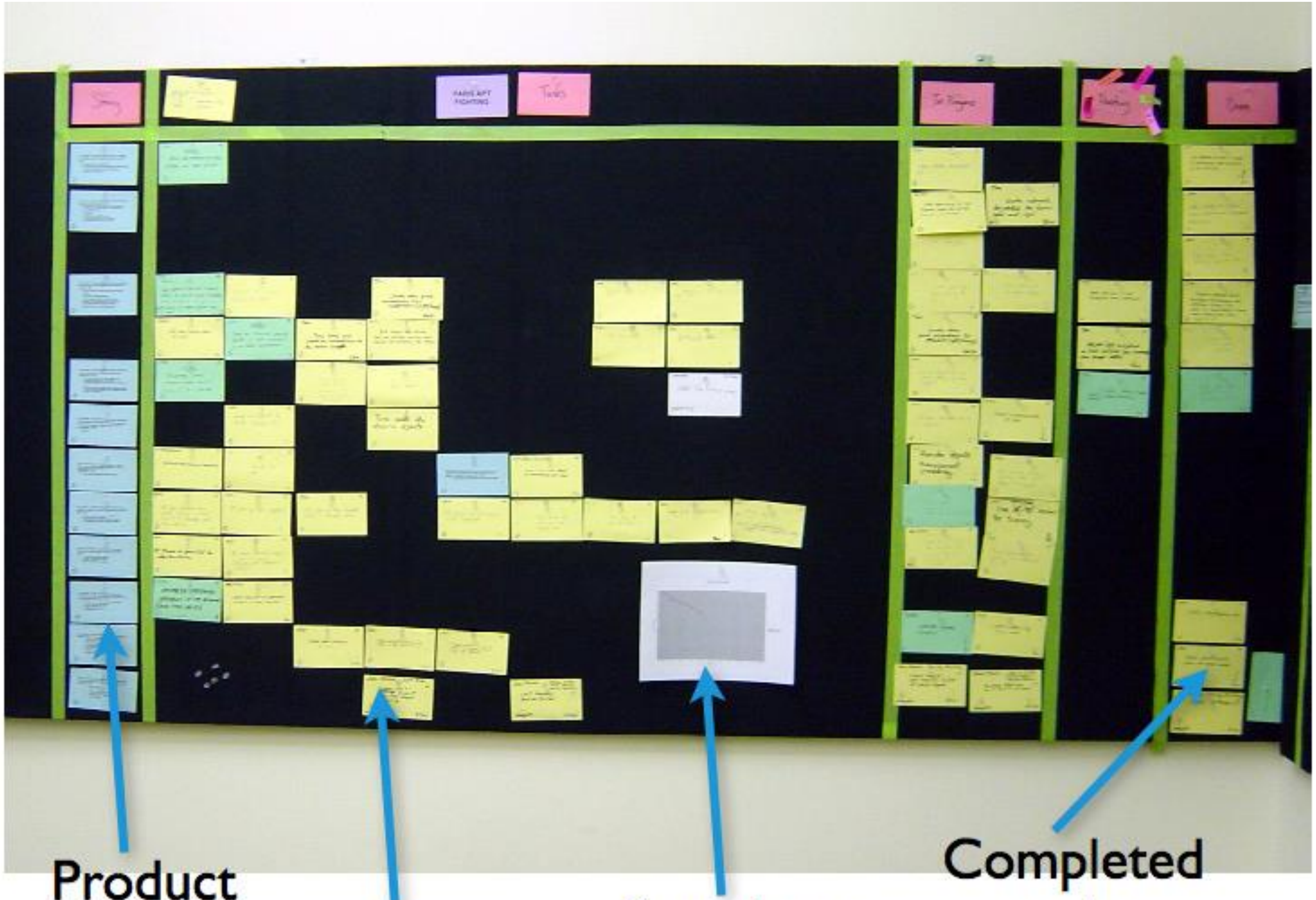
Tasks in Scrum

Scrum task board



Story	To Do	In Process	To Verify	Done
As a user, I... 8 points	Code the... 9 Test the... 8 Code the... 2 Code the... 8 Test the... 8 Test the... 4	Code the... DC 4 Test the... SC 8	Test the... SC 6	Code the... D Test the... SC 8 Test the... SC Test the... SC Test the... SC 6
As a user, I... 5 points	Code the... 8 Test the... 8 Code the... 4 Code the... 6	Code the... DC 8		Test the... SC Test the... SC Test the... SC 6

From: www.mountaingoatsoftware.com

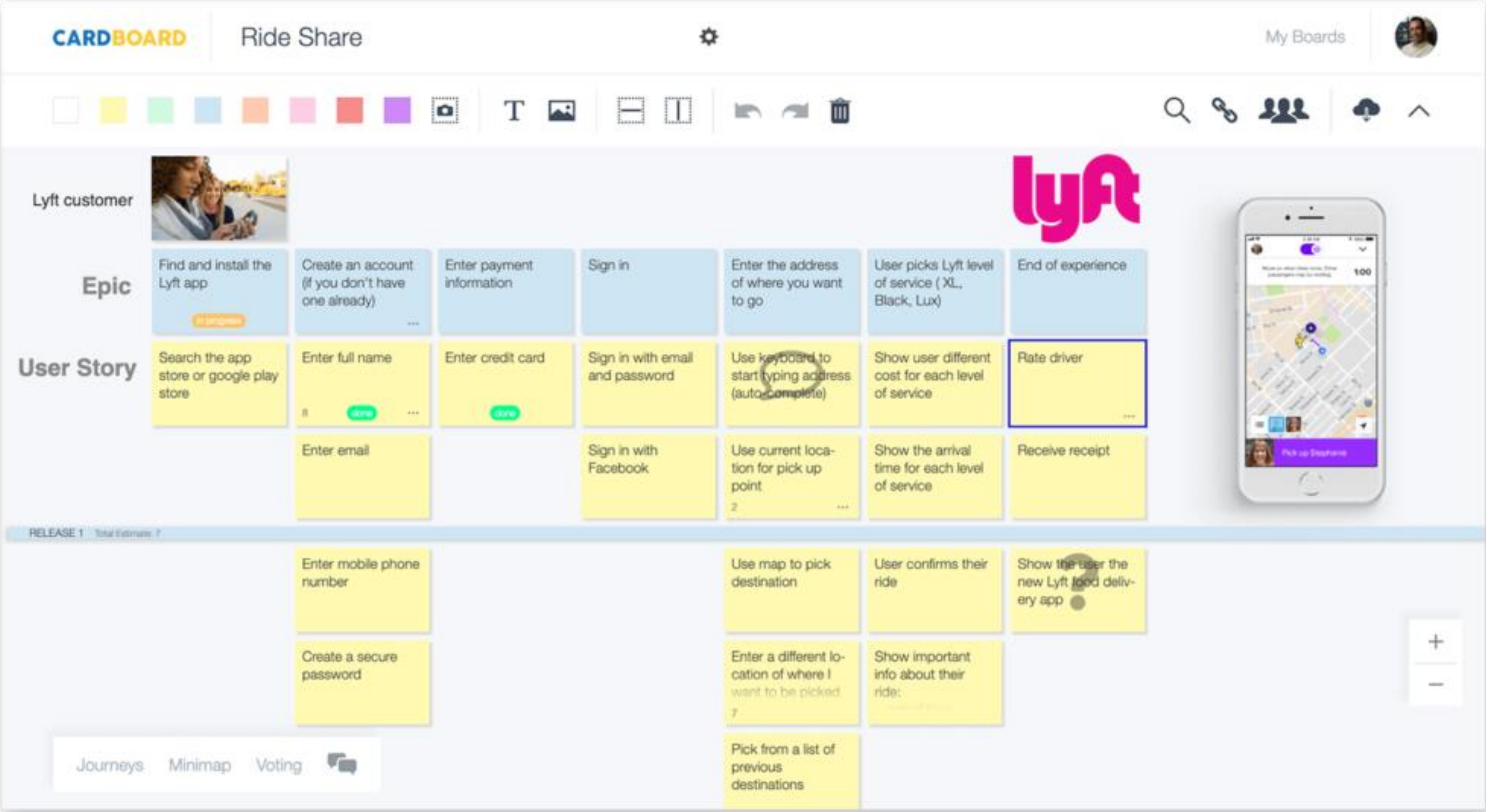


Product backlog

Tasks to do

Burndown chart

Completed tasks

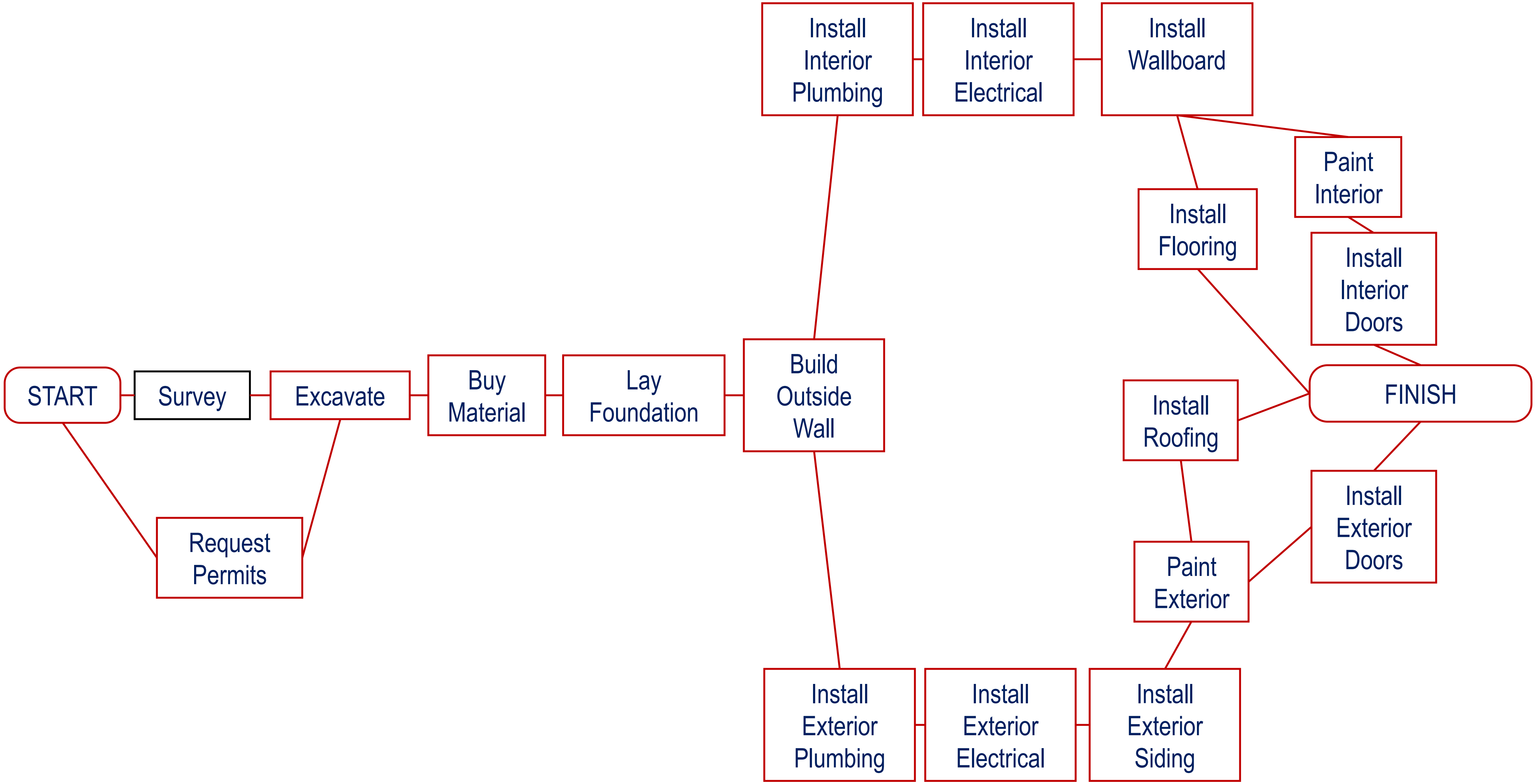


Lyft customer							
Epic	Find and install the Lyft app in progress	Create an account (if you don't have one already)	Enter payment information	Sign in	Enter the address of where you want to go	User picks Lyft level of service (XL, Black, Lux)	End of experience
User Story	Search the app store or google play store	Enter full name 8 done	Enter credit card done	Sign in with email and password	Use keyboard to start typing address (auto-complete)	Show user different cost for each level of service	Rate driver
		Enter email		Sign in with Facebook	Use current location for pick up point 2	Show the arrival time for each level of service	Receive receipt
RELEASE 1 Total Estimate: 7							
		Enter mobile phone number			Use map to pick destination	User confirms their ride	Show the user the new Lyft food delivery app
		Create a secure password			Enter a different location of where I want to be picked 7	Show important info about their ride:	
					Pick from a list of previous destinations		

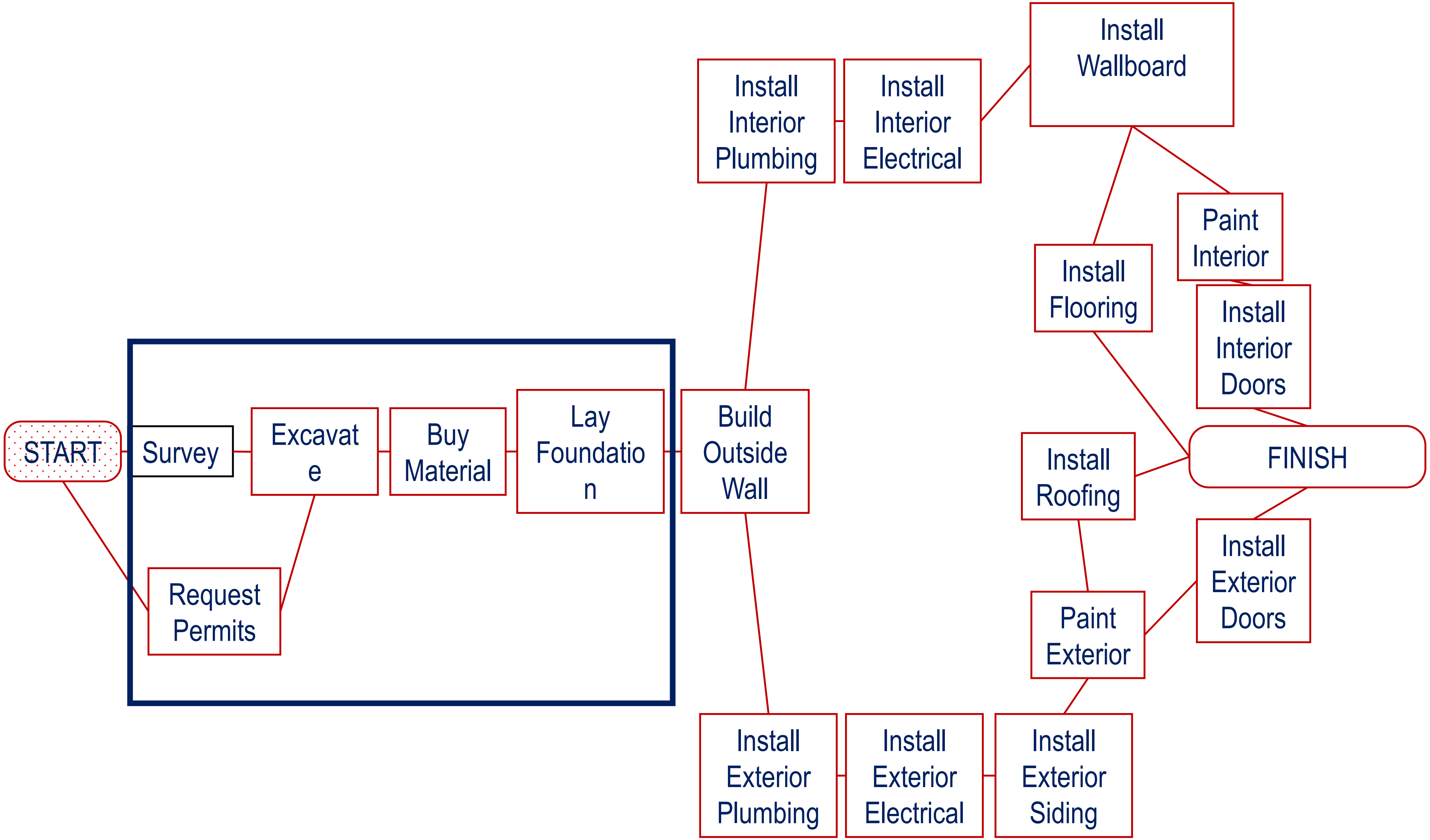
Activities

- Major unit of work
- Culminates in a major project milestone:
  - Scheduled event used to measure progress
  - Internal checkpoints should not be externally visible
  - A project milestone usually produces a baseline
- Activities are often grouped again into higher-level activities with different names:
    - Phase 1, Phase 2 …
    - Step 1, Step 2 …
- Allows separation of concerns
- Precedence relations can exist among activities
  - Example: “A1 must be executed before A2”

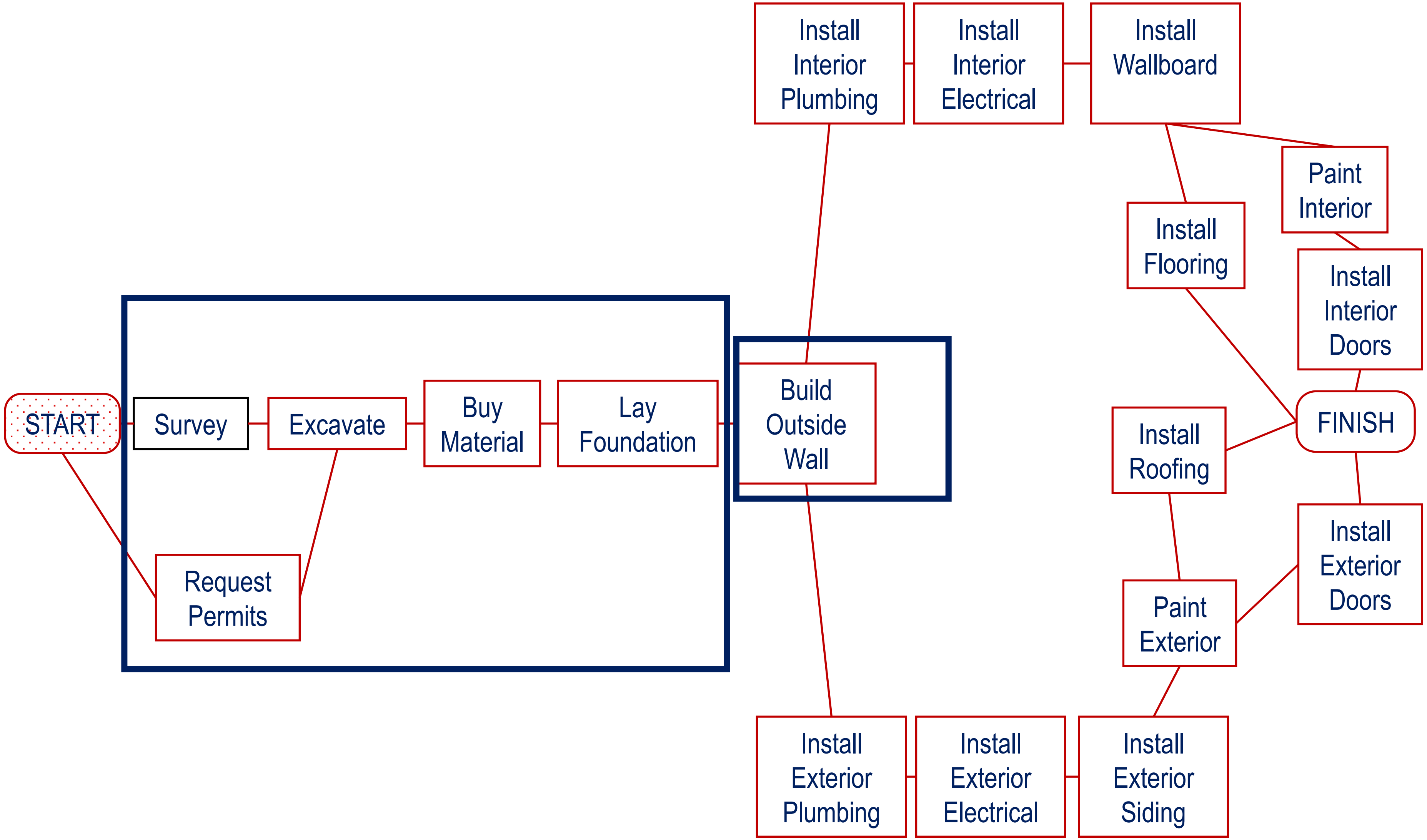
Example: Activities for Building a House



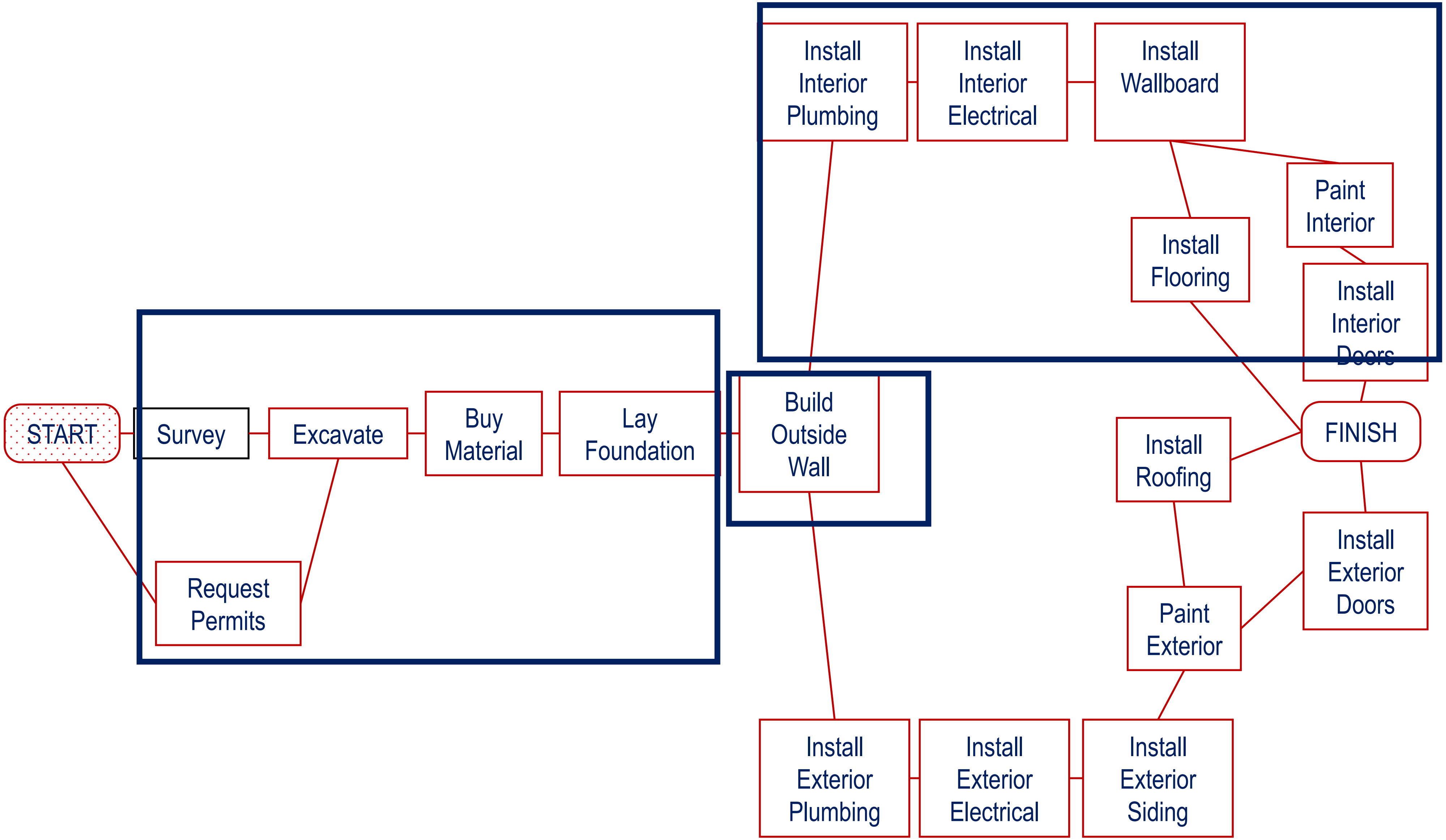
Example: Activities for Building a House



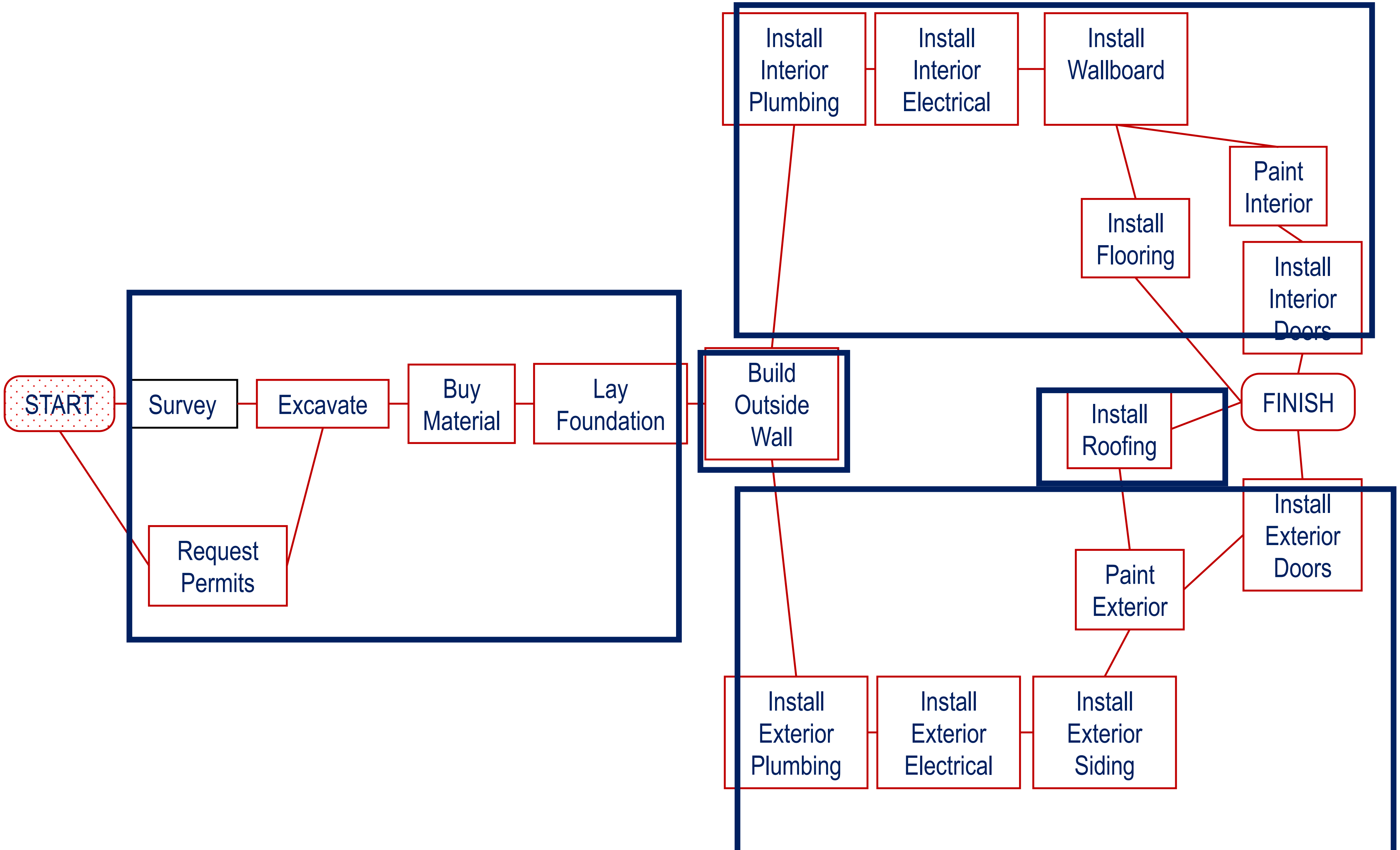
Example: Activities for Building a House



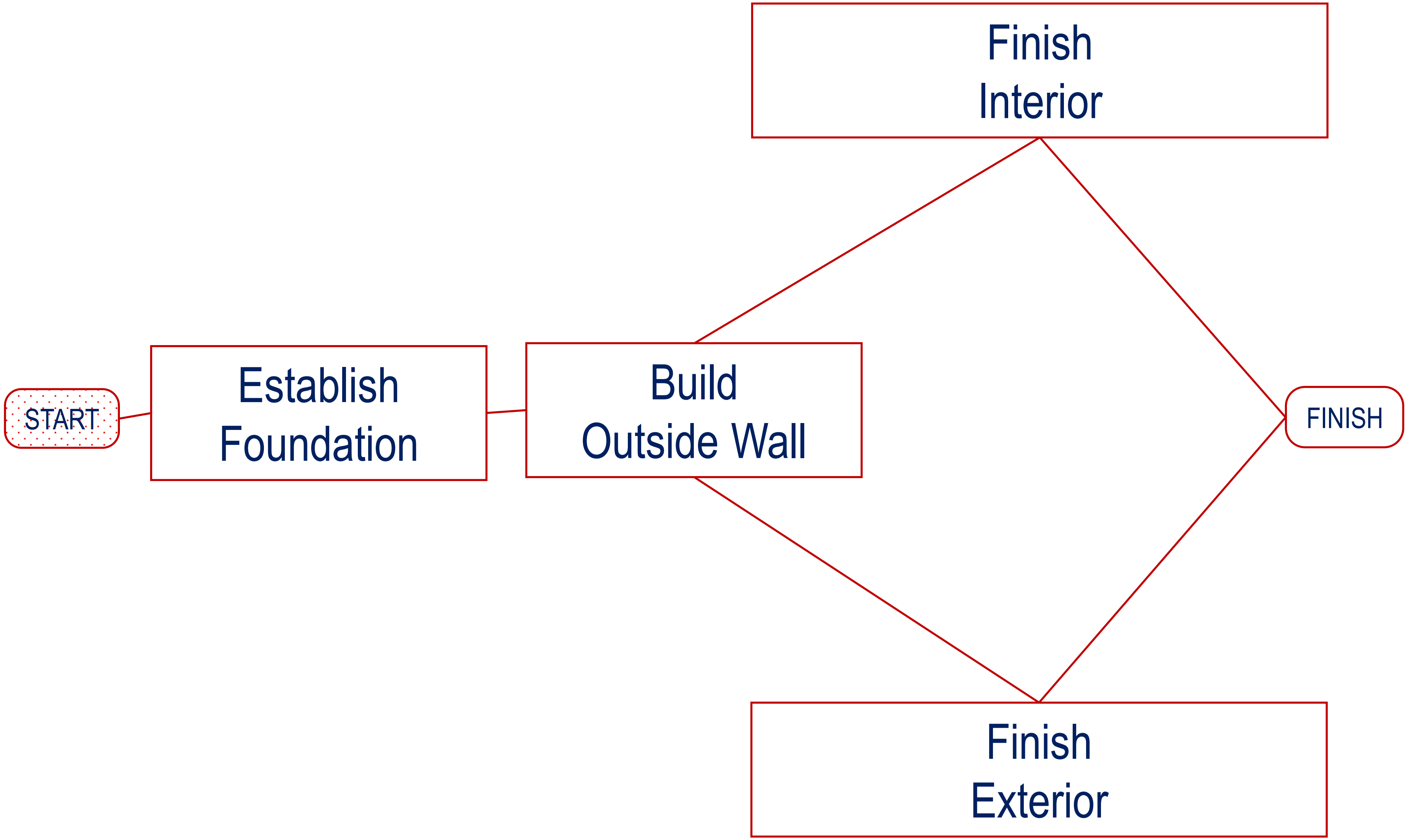
Example: Activities for Building a House



Example: Activities for Building a House



Example: Activities for Building a House



Software Engineering Activities

- Planning
- Requirements Elicitation
- Analysis
- System Design
- Object Design
- Implementation
- Testing
- Delivery

Communication is critical

- In large system development efforts, you will spend more time communicating than coding
- A software engineer needs to learn the so-called soft skills:
  - Collaboration
    - Negotiate requirements with the client and with members from your team and other teams
  - Presentation
    - Present a major part of the system during a review
  - Management
    - Facilitate a team meeting
  - Technical writing
    - Write part of the project documentation.

Communication Event vs. Mechanism

Communication event

- Information exchange with defined objectives and scope
- *Scheduled*: Planned communication
  - Examples: weekly team meeting, review, daily scrum
- *Unscheduled*: Event-driven communication
  - Examples: problem report, request for change, clarification

Communication mechanism

- Tool or procedure that can be used to transmit information
- *Synchronous*: Sender and receiver are communicating at the same time
- *Asynchronous*: Sender and receiver are not communicating at the same time.

Planned Communication Events

Problem Definition

- Objective: Present goals, requirements and constraints
- Example: Client presentation
- Usually scheduled at the beginning of a project

Project Review: Focus on system models

- Objective: Assess status and review the system model
- Examples: Analysis review, system design review
- Scheduled around project milestones and deliverables

Client Review: Focus on requirements

- Objective: Brief the client, agree on requirements changes
- The first client review is usually scheduled after analysis phase.

Planned Communication Events

Walkthrough (Informal)

- Objective: Increase quality of subsystem
- Example
  - Developer informally presents subsystem to team members (“peer-to-peer”)
- Scheduled by each team

Inspection (Formal)

- Objective: Compliance with requirements
- Example
  - Demonstration of final system to customer (Client acceptance test)
- Scheduled by project management

Planned Communication Events

Status Review

- Objective: Find deviations from schedule and correct them or identify new issues
- Example
  - Status section in regular weekly team meeting

Brainstorming

- Objective: Generate and evaluate large number of solutions for a problem
- Example
  - Discussion section in regular weekly team meeting.

Planned Communication Events

Release

- Objective: Baseline the result of each software development activity
- Examples:
  - Software Project Management Plan
  - Requirements Analysis Document
  - System Design Document
  - Beta version of software
  - Final version of software
  - User Manual
- Usually scheduled after corresponding activity (“phase”)

Postmortem Review

- Objective: Describe Lessons Learned
- Scheduled at the end of the project

Unplanned Communication Events

Request for clarification

- The bulk of communication among developers, clients and users
- Example: A developer may request a clarification about an ambiguous sentence in the problem statement.

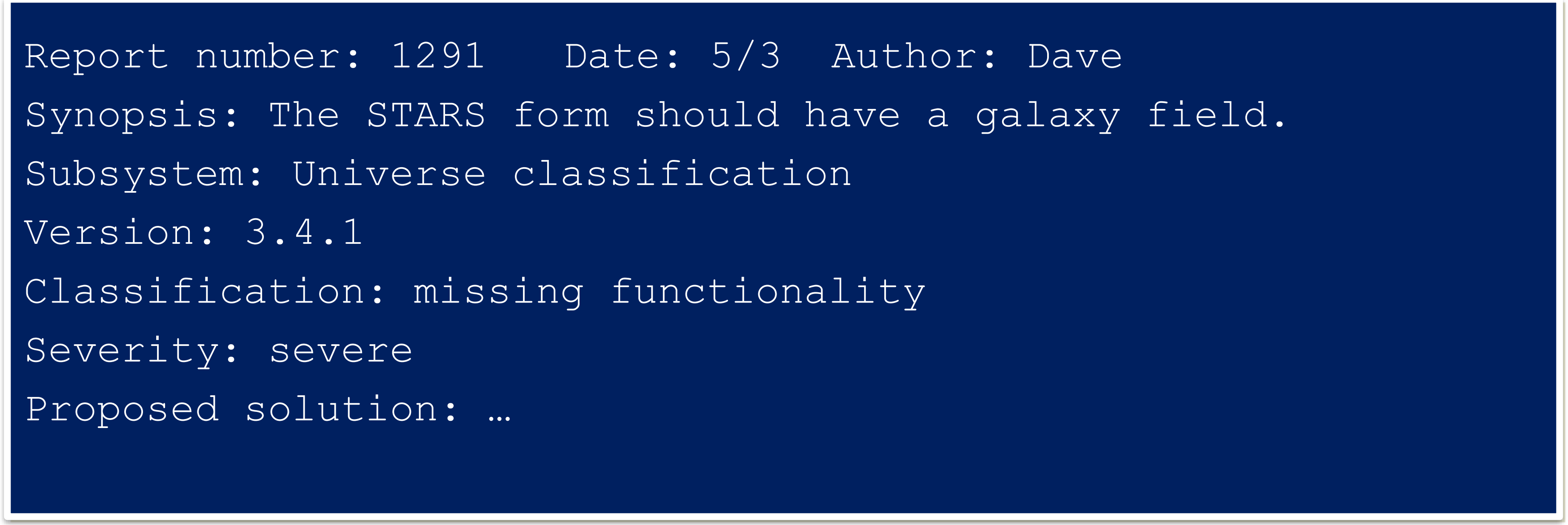
```
From: Alice
Newsgroups: vso.discuss
Subject: SDD
Date: Wed, 2 Nov 9:32:48 -0400

When exactly would you like the System Design Document?
There is some confusion over the actual deadline: the schedule
claims it to be October 22, while the template says we have until
November 7.
Thanks,-Alice
```

Unplanned Communication Events

Request for change

- A participant reports a problem and proposes a solution
- Change requests are often formalized when the project size is substantial
- Example: Request for additional functionality

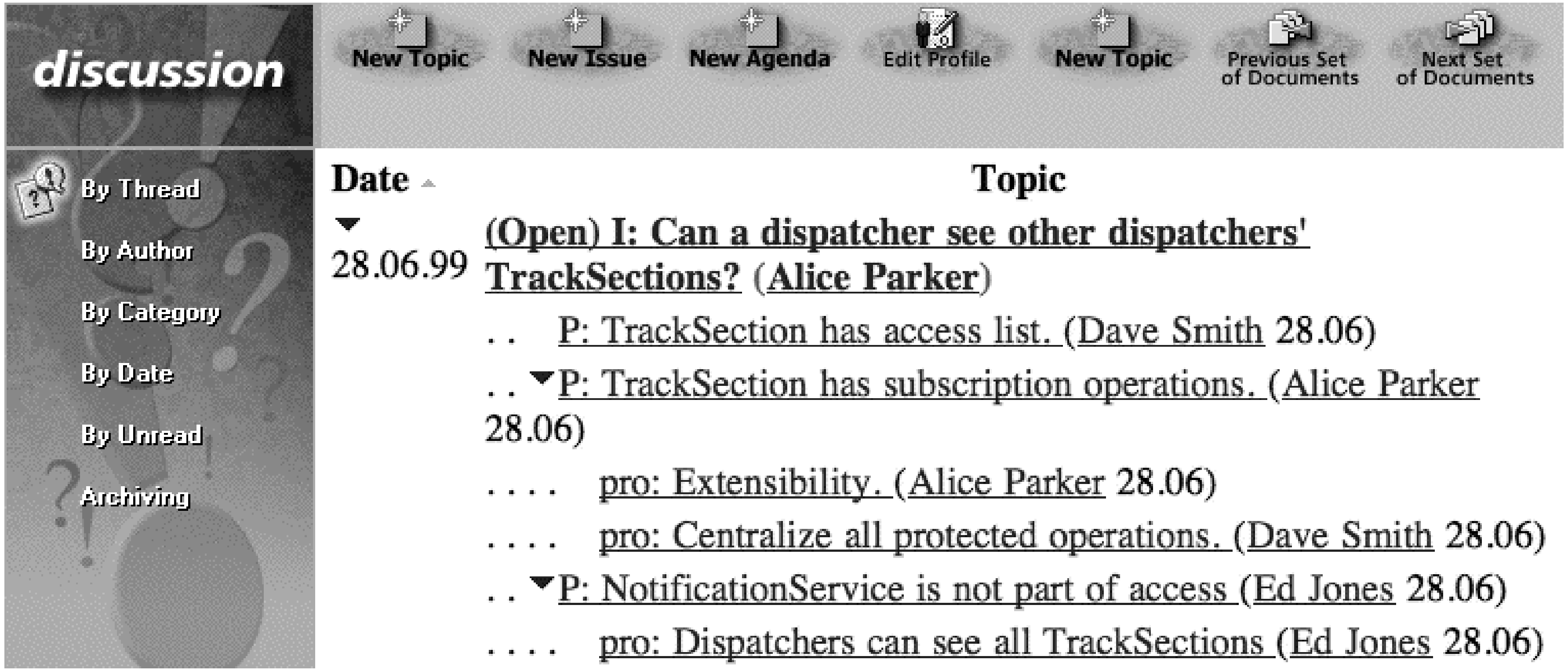


```
Report number: 1291   Date: 5/3  Author: Dave
Synopsis: The STARS form should have a galaxy field.
Subsystem: Universe classification
Version: 3.4.1
Classification: missing functionality
Severity: severe
Proposed solution: …
```

Unplanned Communication Events

Issue resolution

- Selects a single solution to a problem for which several solutions have been proposed
- Uses issue base to collect problems and proposals.



Date	Topic
28.06.99	(Open) I: Can a dispatcher see other dispatchers' TrackSections? (Alice Parker)
	. . P: TrackSection has access list. (Dave Smith 28.06)
	. . P: TrackSection has subscription operations. (Alice Parker 28.06)
	. . . . pro: Extensibility. (Alice Parker 28.06)
	. . . . pro: Centralize all protected operations. (Dave Smith 28.06)
	. . P: NotificationService is not part of access (Ed Jones 28.06)
	. . . . pro: Dispatchers can see all TrackSections (Ed Jones 28.06)

Synchronous Communication Mechanisms

- Hallway conversation
  - Supports: Unplanned conversations, Request for clarification, request for change
  - + Cheap and effective for resolving simple problems
  - – Information loss, misunderstandings are frequent
  - – Usually, no record of the communication

- Meeting (face-to-face, phone, video conference, Google Chat)
  - Supports: Planned conversations, client review, project review, status review, brainstorming, issue resolution
  - + Effective for issue resolution and consensus building
  - – High cost (people, resources), low bandwidth.
  - – Minutes are very helpful

Asynchronous Communication Mechanisms

- E-Mail
  - Supports: Release, change request, brainstorming
  - + Ideal for planned communication and announcements
  - – E-mail taken out of context can be misunderstood, sent to the wrong person, or lost
- Newsgroup
  - Supports: Release, change request, brainstorming
  - + Suited for discussion among people who share a common interest; cheap (shareware available)
  - – Primitive access control (often, you are either in or out)
- World Wide Web (Portal)
  - Supports: Release, change request, inspections
  - + Provide the user with a hypertext metaphor: Documents contain links to other documents.
  - – Does not easily support rapidly evolving documents.

Typical Initial Communication Activities in a Software Project

- Understand problem statement
- Join a team
- Schedule and attend team status meetings
- Join the communication infrastructure.

Understand the Problem Statement

- The **problem statement** is developed by the client
  - Also called scope statement
- Usually, initiates a software project

Ingredients of a Problem Statement

- Current situation
  - The problem to be solved
  - Description of one or more scenarios
- Requirements
  - Functional and nonfunctional requirements
  - Constraints (“pseudo requirements”)
- Target environment
  - The environment in which the delivered system has to perform a specified set of system tests
- Project schedule
  - Major milestones that involve interaction with the client including deadline for delivery of the system
- Client acceptance criteria
  - Criteria for the system tests.

Join a Team

- During the project definition phase, the project manager forms a team for each subsystem
- Additional cross-functional teams are formed to support the subsystem teams
- Each team has a team leader
- Other roles can include (many other roles exist!)
  - Software architect
  - Programmer
  - Configuration manager
  - API-Liaison
  - Technical writer
- The responsibilities of the team and the responsibilities each member must be defined to ensure the team success.

Attending Team Status Meetings

- Important part of a software project: The regular team meeting (weekly, daily,…)
- Meetings are often perceived as pure overhead
- Important task for the team leader:
  - Train the teams in meeting management
    - Announce agendas
    - Write minutes
    - Keep track of action items
  - Show value of status meeting
  - Show time-saving improvements.

Join the Communication Infrastructure

- A good communication infrastructure is the backbone of any software project
  - Web-Portal, e-mail, Newsgroups, Lotus Notes
- Learn to use the appropriate communication mechanism for the information at hand
  - The appropriateness of mechanisms may depend on the organizational culture.
- Register for each communication mechanism which is used by the software project
  - Get an account, get training
- Questions to ask:
  - Are meetings scheduled in a calendar?
  - Does the project have a problem reporting system?
  - Do team members provide peer reviews in meetings or in written form?

Join the Communication Infrastructure

- Establish meeting times

  May want to use:

When is Good:	http://whenisgood.net
Doodle:	http://doodle.com

- Have a clear meeting agenda
- Establish communication methods
  - Phone call, email, text message, Google chat, etc.
- Keep minutes; at the least, have a short summary written, available to all
- Keep a record of status meetings

Homework

- Read Chapter 3 from the textbook