FDIT:
PROJECT MANAGEMENT AND PERFORMANCE

Week 6

J Zander

Week 6 Overview

Learning Objectives:

- Review of Agile and Scheduling- In Practice
- Writing tasks and epics
- Writing risk and evaluating risk
- Prioritizing!
- Gantt charts and scheduling your work1

Scrum Overview

- Scrum is an **Agile framework** used to manage complex projects — especially in software and digital product development.

- It focuses on **iterative progress**, **team collaboration**, and **continuous improvement** through short work cycles called **sprints**.
-

Roles

ROLE	RESPONSIBILITY
Product Owner	Represents the customer and business; defines the vision and priorities. Owns the *Product Backlog*.
Scrum Master	Facilitates Scrum practices; removes roadblocks; coaches the team in Agile principles.
Developers	Cross-functional group that does the work — design, code, test, and deliver the product increment.

Scrum Artifacts

ARTIFACT	PURPOSE
PRODUCT BACKLOG	Ordered list of everything that might be needed in the product (features, fixes, ideas).
PRODUCT SPRINT	Subset of the Product Backlog chosen for the current Sprint, plus a plan for delivering it.
INCREMENT	The usable, potentially shippable product created during each Sprint.

Let’s work on our shared JIRA board:
Practice writing tasks (wording, level of granularity)
Practice BACKLOG PLANNING as a group

Common Risk Factors

1. Technical Risks
2. Project Management Risks
3. Resource Risks
4. Operational Risks
5. Strategic Risk
6. UX Risk (*)

Technical Risks (common)

- **System failures / downtime** → Server crashes, outages, or unstable architecture
- **Integration risks** → Compatibility issues between APIs, third-party services, or legacy systems.
- **Scalability** → Can the system handle increased users or data? (how do you know? Are there limits?)
- **Security vulnerabilities** → Data breaches, weak authentication, lack of encryption.
- **Unproven technology** → Using a new framework, tool, or platform that may not be stable.

Project Management Risks

- **Scope creep** → Uncontrolled expansion of requirements without resources.
- **Unclear requirements** → Ambiguity leading to rework or wasted effort.
- **Unrealistic timelines** → Pressure to deliver faster than feasible.
- **Budget overruns** → Underestimating costs for dev, QA, or infrastructure.
- **Dependency delays** → Reliance on external vendors, other teams, or approvals.

Resource Risks

- **Skill gaps** → Team lacks expertise in specific tech or methods.
- **Key-person dependency** → Reliance on one developer, designer, or PM.
  - (what if they get sick? Quit? Go on Vacation?)
- **Turnover / burnout** → High workload or stress leading to attrition.
- **Poor collaboration** → Misalignment between dev, design, QA, or stakeholders

Operational Risks

- **Data management issues** → Data loss, poor backups, or compliance failures.
- **Infrastructure failures** → Cloud misconfigurations, network downtime.
- **Process bottlenecks** → Manual workflows slowing down delivery.
- **Third-party risks** → SaaS provider downtime, API changes, or licensing issues.

Strategic Risks

- **Market shifts** → Competitor releases, tech obsolescence, or user expectations changing. (Let's talk about who manages this risk)
- **Regulatory & compliance** → GDPR, HIPAA, SOC 2, or other industry standards.
- **Customer adoption** → Risk that product doesn't meet user needs or gain traction.
- **Reputation risk/ Brand Damage** → Poor release quality harming trust.

UX Risks (poor usability)

- **Accessibility gaps** → Product not accessible (litigation and usability problem).
- **Performance issues** → Slow load times, lag, poor mobile optimization.
  - Performance is a FEATURE!
- **Usability challenges** → Confusing workflows, leading to abandonment.

- **NOTE: all other risks can also apply to UX. And Poor usability can certainly damage the brand!**

DEFINING/ WRITING RISKS

Writing good **risks** in a **risk assessment** is about being specific, measurable, and action-oriented — not just listing vague "things that might go wrong." It is also NOT PERSONAL.

What Is a Risk Statement?

- A risk statement describes a potential problem that *might* affect your project's success.

  Formula for writing a clear risk:

- "There is a risk that **[cause]** may lead to **[event/issue]**, resulting in **[impact/consequence]**."

Example:
There is a risk that **API rate limits from Google Calendar** may lead to **incomplete data sync**, resulting in **user frustration and reduced trust in the app.**

-

RISK ASSESSMENT

Risk	Mitigation Strategy
Third-party APIs become unavailable or change unexpectedly	Use API versioning and fallback data storage
Poor scalability leads to performance issues under load	Conduct load testing early and plan cloud autoscaling

RISK ASSESSMENT

-

Likelihood	Impact	Risk Level
Low (1-2)	Low (1-2)	Low
Med (3)	Med (3)	Moderate
High (4-5)	High (4-5)	Critical

Back to our shared project: Identify some risks

Planning Time

TRELLO
-Everyone go into Trello.
You can create Kanban and boards and sync to your calendar (Gantt).
Let’s do that now.

CPM: Critical Path Method

Identify the LONGEST path of dependent tasks to determine the shortest project duration.
-Need to have an idea of how long tasks will take (i.e., have done them before)
- Focus on efficiency (vs creativity)
- Identify a CRITICAL PATH

Key Vocabulary

Activities: tasks to be completed
Dependencies: tasks which depend on completion of other tasks
Duration: how long will they take?
Earliest Start (ES)/ Earliest Finish (EF)
Latest Start (LS)/ Latest Finish (LF)
Slack (buffer time) - amount of slack time without jeopardizing the project
Critical path: sequence of tasks that determine the minimum project duration

CPM: Critical Path Method

Identify the LONGEST path of dependent tasks to determine the shortest project duration.
-Need to have an idea of how long tasks will take (i.e., have done them before)
- Focus on efficiency (vs creativity)
- Identify a CRITICAL PATH
- Can we try this for our project?

In FIGJAM

1. Draw boxes for each task.
2. Connect them with arrows showing dependencies.
3. Label durations (e.g., 2 days, 5 days).
4. Let’s talk thru the following:
    - **ES & EF** (forward pass)
    - **LS & LF** (backward pass)
    - **Slack = LS – ES**
5. Highlight the **critical path** — the tasks with zero slack.

PERT: Program Evaluation and Review Technique

Estimate project time using *probabilistic* time estimates (optimistic, likely, pessimistic). (because we don't know how long it will take)

Activity durations are uncertain or variable.

Managing uncertainty and risk in time estimates.

Based on weighted average time using the formula: **(O + 4M + P) / 6**.

Expected project duration with a measure of uncertainty.

PERT

“You’re building a new prototype website. The project has 6 main tasks: planning, wireframing, design, coding, testing, and launch.”

For example…your Capstone!

Estimating Duration

"But what if we *don't know* exactly how long something will take?"

Each activity gets **three estimates:**

- **Optimistic (O)** – best case
- **Most Likely (M)** – normal case
- **Pessimistic (P)** – worst case

Expected Time (TE) = (O + 4M + P) / 6

Then you map out the GANTT chart/ similar the same way.