Agile Planning for Software Products

⚠ A Task

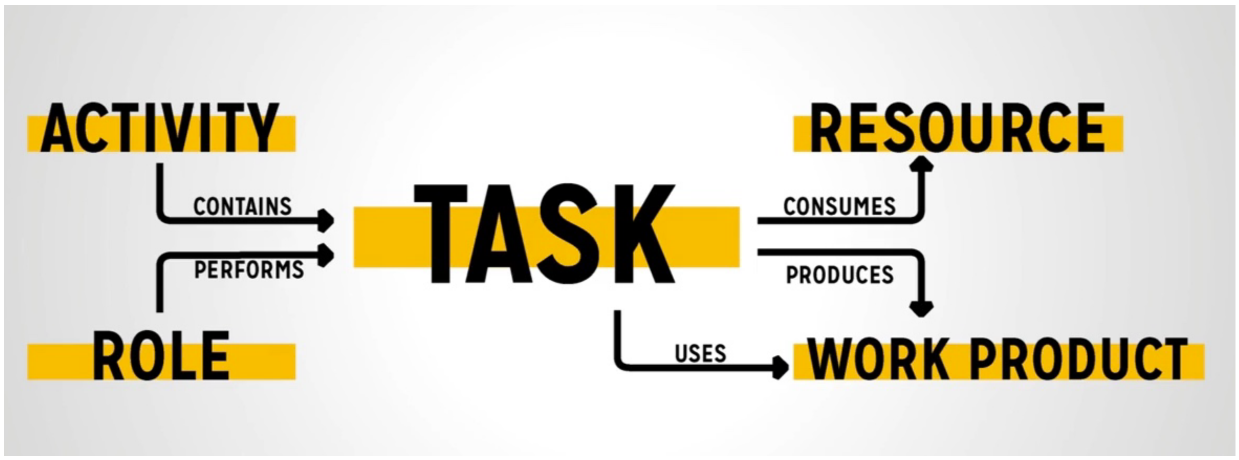
↳ a small, manageable step of a project to be completed

⚠ A Role

↳ a duty that a person takes on or plays

⚠ work Product

↳ an output produced by a task or a process



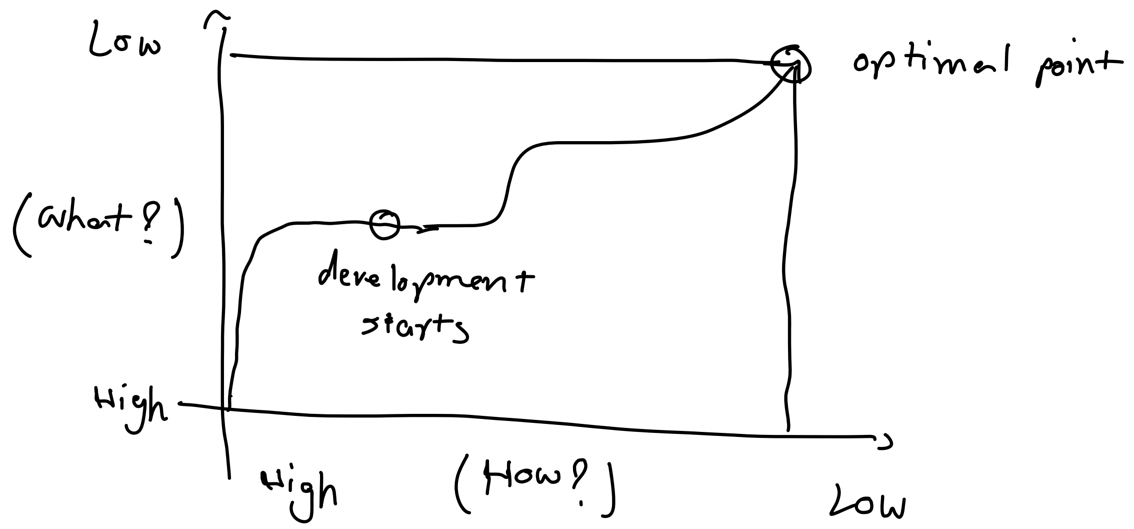
⚠ schedule

↳ The mapping of tasks to a timeline

⚠ milestones

↳ an internal checkpoint to measure progress. They're not time based, but event or action based

⚠ Uncertainity space diagram



⚠ A Work Breakdown Structure (WBS) takes one large work product & breaks it down into smaller, manageable work products, into a hierarchy

⚠ Estimate

↳ a guess for the time it will take for your development team to complete a task

⚠ Target

↳ a point in the schedule to meet. This is almost an ideal deadline

⚠ Commitments

↳ what you are agreeing to deliver

⚠ A Task Estimate

↳ an approximation of how long a task will take.
should be based on previous work

⚠ Story Points are uniteless and relative

⚠ velocity = work accomplished / length of the sprint

⚠ In velocity-driven development, each sprint is planned based on the amount of work that is being completed, or velocity achieved in previous sprints

⚠ Time boxing is a way for software teams to compartmentalize the work which they have planned for themselves & leave room for reflection on their progress

↳ The general term for something being built in a restricted time period

⚠ A basic gantt chart consists of tasks & dates

⚠ Iteration planning

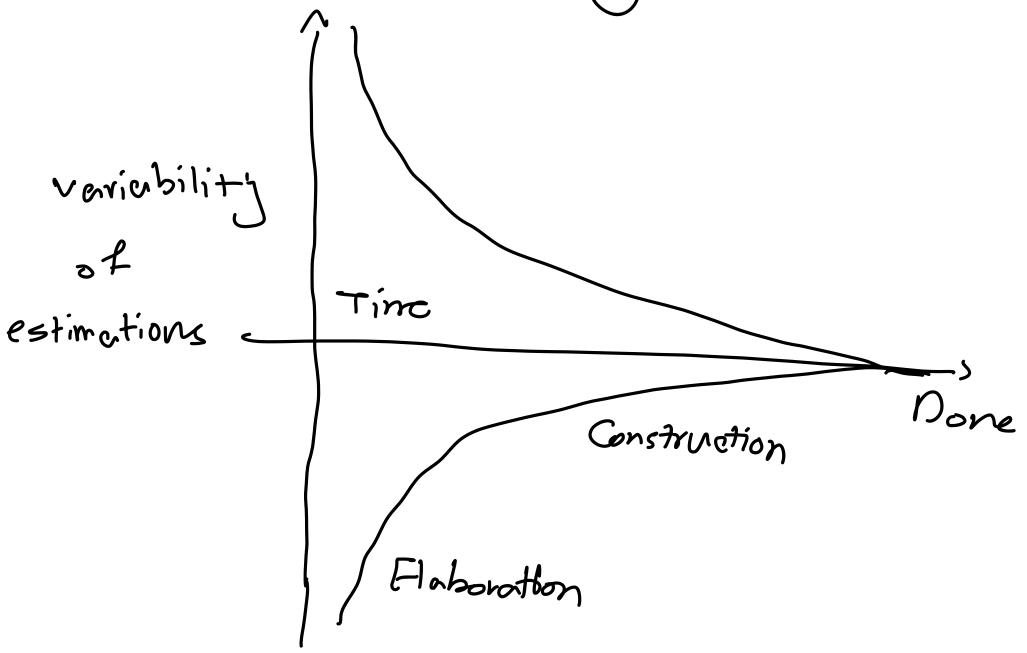
↳ is fine grained, dealing with tasks as the pieces of work to do

⚠ Release planning

↳ is more coarse grained, dealing with user stories as the pieces of work to do

  ↳ assigning user stories to planned sprints within your project

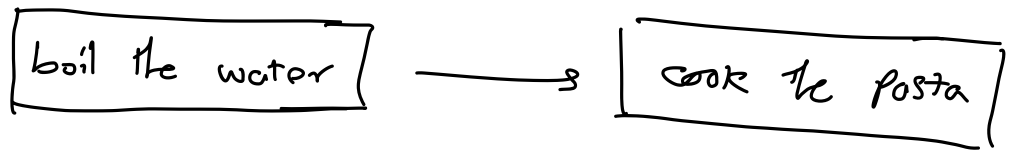
⚠ The Cone of Uncertainity



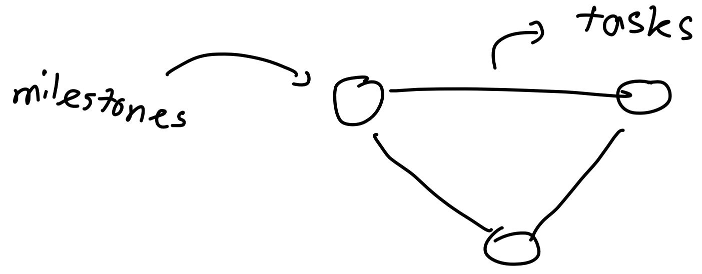
⚠ Types of task dependencies

- start - start
- start - finish
- finish - start

⚠ A. CPM is a visual way to organize task dependencies



⚠ A PERT chart is a visual representation of a project



⚠ iteration planning

- generate realistic commitment based on velocities, task estimates, and available time

⚠ Anti-Pattern

- A commonly occurring situation in a project that comes with negative consequences

⚠ Anti-Patterns in Agile development

- analysis Paralysis
  - getting stuck in the specification phase of your project
  - How to resolve : releasing incrementals
- putting the cart before the horse
  - placing too much emphasis on a part of the project that should be done later
  - How to resolve : focus on what must be done NOW!
- Group think
  - accepting the poor idea which is popular
  - How to avoid : generate solutions to problems silently

⚠ anti-patterns in teams

- over engineering
- gold plating
- vendor lock-in
- silos
- Backlogging

⚠ anti-Patterns in development

- view graph engineering
- Fire Drill
- Death March

⚠ anti-Patterns in management

- micro management
- seagull manager

⚠ anti-Patterns in indivisual developers

- loose Canon
- intellectual violence

⚠ A Risk

- → something that could potentially cause your project to fail
  - → scope risks
  - → technology risks
  - → customers & stakeholders risk
  - → Personnel risk

⚠ Impact vs Likelihood Matrix

- → a 2D representation of the amount of influence a risk has on your project. It shows you what to focus your efforts on preventing

⚠ It's better to first work on high value - high risk features