estimation techniques

Panin Alexei

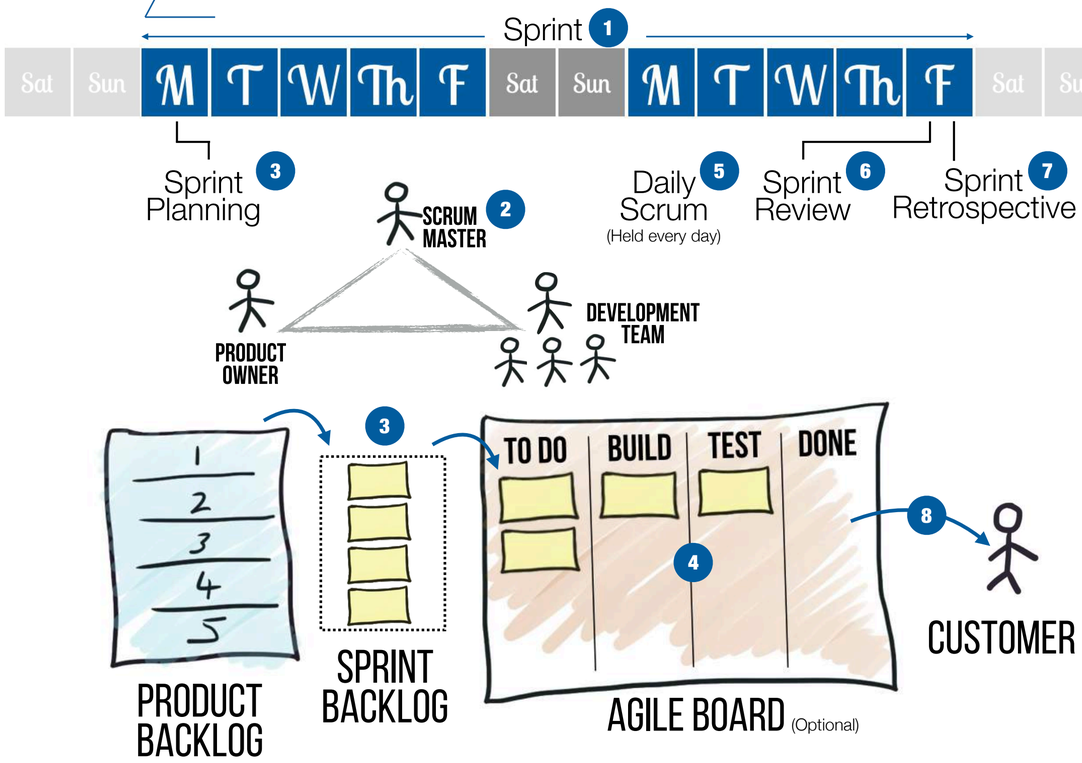
recap - previous lecture

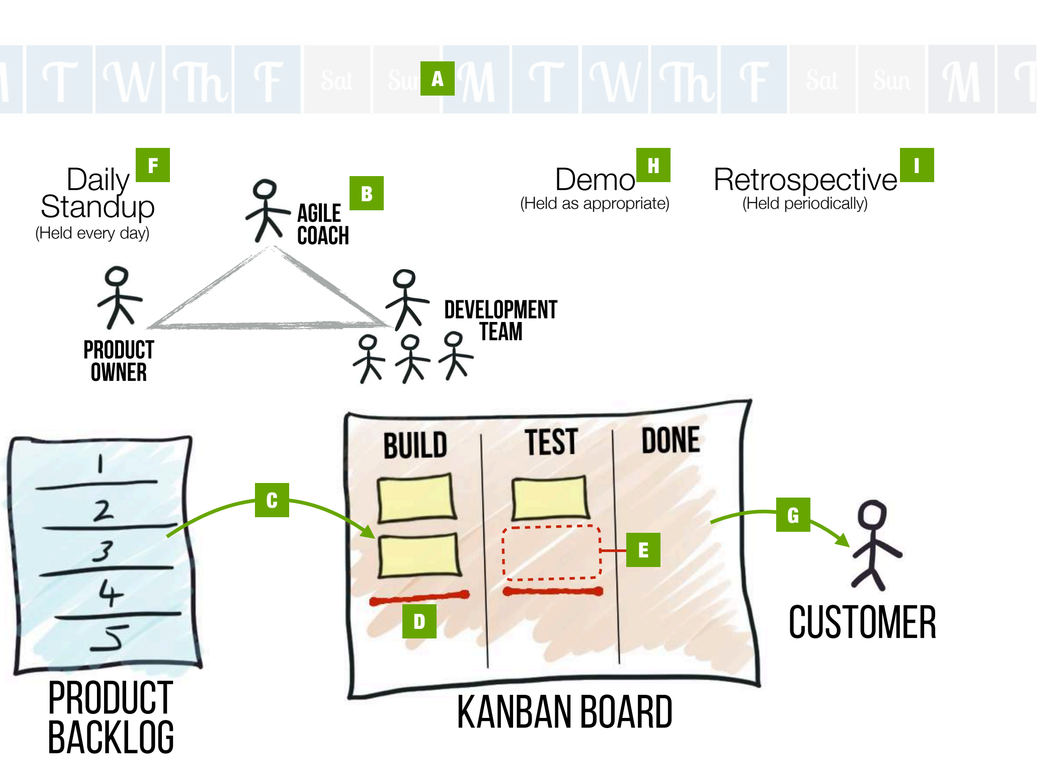
https://www.youtube.com/watch?v=rIaz-l1Kf8w

SCRUM vs KANBAN

SCRUM vs KANBAN CHEAT SHEET

Watch the video:
http://bit.ly/2jDxyUh





1. **Scrum** Teams work in a series of **Sprints** of 1, 2 (most common), 3 or 4 weeks duration.
2. It is the job of the **Scrum Master** to help the *Product Owner*, the *Development Team* to develop and maintain good habits.
3. Each Sprint starts with a **Sprint Planning Meeting** - facilitated by the *Scrum Master* and attended by the *Product Owner* and the *Development Team* and (optionally) other *Stakeholders*. Together they select high priority items from the **Product Backlog** that the *Development Team* can commit to delivering in a single Sprint. The selected items are known as the **Sprint Backlog**.
4. The *Development Team* works on items in the Sprint Backlog ***only*** for the duration of the Sprint. In all but exceptional circumstances, new issues must wait for the next Sprint.
5. The **Daily Scrum** (aka Daily Huddle, Daily Standup) is a short standup meeting attended by the *Scrum Master*, the *Product Owner* and the *Development Team*.
6. A review of the Sprint. Often includes a demo of new features to *Stakeholders*.
7. An examination of what went well, what could be improved, etc. Aim: to make each Sprint more efficient and effective than the last.
8. At the end of the Sprint, completed items are packaged for release to live. (Note that some teams release more often than this.) Any incomplete items are returned to the Product Backlog.

- **A** **Kanban** is a *continuous* process. (cf. Scrum's *periodic* Sprint.)
- **B** It is the job of the **Agile Coach** (if present - not all Kanban teams have one) to help the *Product Owner* and the *Development Team* to develop and maintain good habits.
- **C** Items are "***pulled***" directly from the **Product Backlog.**
- **D** Each column has a strict Work in Progress (**WIP**) limit. The WIP limits ensure that items move across the board in the shortest possible time.
- **E** An empty - or nearly empty - column is a signal to the *previous* column to send another item. This is the "pull" system in action.
- **F** The **Daily Standup** is a short standup meeting attended by the *Agile Coach*, the *Product Owner* and the *Development Team*.
- **G** Each item is packaged for release as soon as it is ready.
- **H** A demonstration of new functionality to *Stakeholders*.
- **I** A look at what went well, what could be improved, etc. Aim: to improve the process.

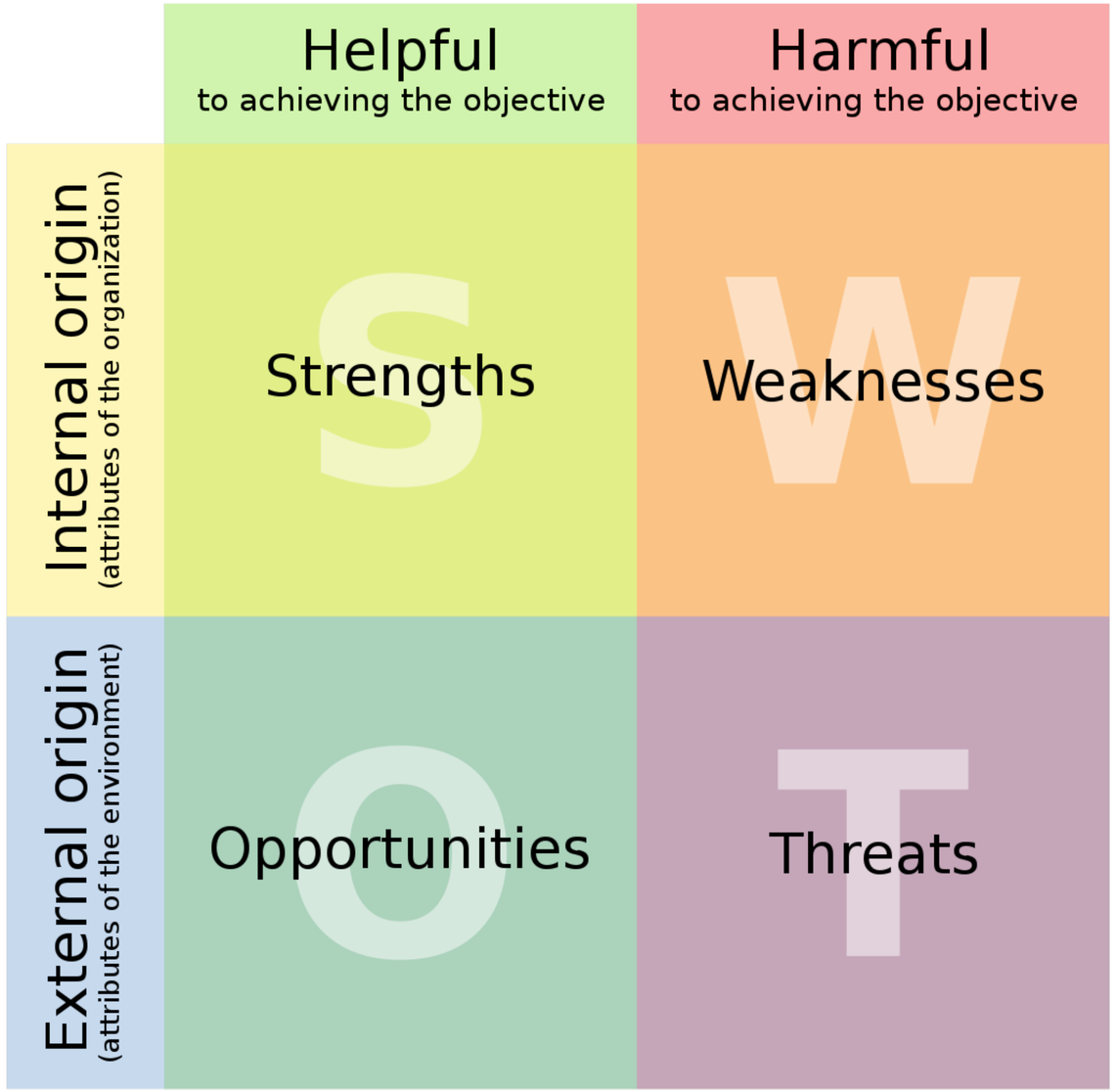
DEVELOPMENT THAT PAYS

SWOT

SWOT ANALYSIS

SWOT analysis is a strategic planning and strategic management technique used to help a person or organization identify Strengths, Weaknesses, Opportunities, and Threats related to business competition or project planning. It is sometimes called situational assessment or situational analysis.

Source, Wikipedia



- **strengths**: characteristics of the business or project that give it an advantage over others

- **strengths**: characteristics of the business or project that give it an advantage over others

- **weaknesses**: characteristics that place the business or project at a disadvantage relative to others

- **strengths**: characteristics of the business or project that give it an advantage over others

- **weaknesses**: characteristics that place the business or project at a disadvantage relative to others

- **opportunities**: elements in the environment that the business or project could exploit to its advantage

- **strengths**: characteristics of the business or project that give it an advantage over others
- **weaknesses**: characteristics that place the business or project at a disadvantage relative to others
- **opportunities**: elements in the environment that the business or project could exploit to its advantage
- **threats**: elements in the environment that could cause trouble for the business or project

the 3 major parts to project estimation

the 3 major parts to project estimation

effort

the 3 major parts to project estimation

effort

cost

the 3 major parts to project estimation

effort cost resource

top-down

the 3 major parts to project estimation

effort cost resource

top-down bottom-up

the 3 major parts to project estimation

effort cost resource

estimation techniques

absolute value estimation (ex. hours)

estimation techniques

absolute value estimation (ex. hours)

relative value estimation (ex. SP)

estimation techniques

absolute value estimation (ex. hours)

relative value estimation (ex. SP)

data driven estimation

estimation techniques

absolute estimation

(hours, days, weeks, months, minutes)

strengths

- Accuracy Improvement: Hour-based estimation offers a finer-grained perspective on effort, potentially enhancing accuracy for tasks.

strengths

- Accuracy Improvement: Hour-based estimation offers a finer-grained perspective on effort, potentially enhancing accuracy for tasks.

- Detailed Planning: Estimating in hours facilitates precise scheduling and resource allocation, enabling detailed planning.

strengths

- Accuracy Improvement: Hour-based estimation offers a finer-grained perspective on effort, potentially enhancing accuracy for tasks.

- Detailed Planning: Estimating in hours facilitates precise scheduling and resource allocation, enabling detailed planning.

- Concrete Metrics: Hour-based estimates offer tangible metrics for monitoring progress and comparing planned versus actual time spent.

strengths

- Accuracy Improvement: Hour-based estimation offers a finer-grained perspective on effort, potentially enhancing accuracy for tasks.
- Detailed Planning: Estimating in hours facilitates precise scheduling and resource allocation, enabling detailed planning.
- Concrete Metrics: Hour-based estimates offer tangible metrics for monitoring progress and comparing planned versus actual time spent.
- Clear Accountability: Assigning hours to tasks clarifies responsibilities and promotes accountability among team members.
- Ensures better predictability

opportunities

- Process Improvement: Enhance estimation by refining techniques, leveraging historical data, and integrating feedback.

opportunities

- Process Improvement: Enhance estimation by refining techniques, leveraging historical data, and integrating feedback.

- Training and Education: Boost team skills and confidence in hour-based estimation through dedicated training programs.

opportunities

- Process Improvement: Enhance estimation by refining techniques, leveraging historical data, and integrating feedback.

- Training and Education: Boost team skills and confidence in hour-based estimation through dedicated training programs.

- Tool Development: Develop or adopt specialized tools to streamline hour-based estimation and enhance accuracy.

opportunities

- Process Improvement: Enhance estimation by refining techniques, leveraging historical data, and integrating feedback.
- Training and Education: Boost team skills and confidence in hour-based estimation through dedicated training programs.
- Tool Development: Develop or adopt specialized tools to streamline hour-based estimation and enhance accuracy.
- Continuous Improvement: Foster a culture of experimentation and learning to refine estimation approaches iteratively.

weaknesses

- Inherent Uncertainty: Hour-based estimation faces challenges due to inherent uncertainty in software development tasks, leading to potential inaccuracies.

weaknesses

- Inherent Uncertainty: Hour-based estimation faces challenges due to inherent uncertainty in software development tasks, leading to potential inaccuracies.

- Time Consuming: Estimating tasks in hours demands more time and effort than other techniques, potentially delaying the planning process.

weaknesses

- Inherent Uncertainty: Hour-based estimation faces challenges due to inherent uncertainty in software development tasks, leading to potential inaccuracies.
- Time Consuming: Estimating tasks in hours demands more time and effort than other techniques, potentially delaying the planning process.
- Rigid Planning: Hour-based estimates may foster rigid planning, hindering adaptation to changes or uncertainties during the project.

weaknesses

- Inherent Uncertainty: Hour-based estimation faces challenges due to inherent uncertainty in software development tasks, leading to potential inaccuracies.
- Time Consuming: Estimating tasks in hours demands more time and effort than other techniques, potentially delaying the planning process.
- Rigid Planning: Hour-based estimates may foster rigid planning, hindering adaptation to changes or uncertainties during the project.
- Limited Flexibility: Hour-based estimates may overlook non-linear complexities or unexpected dependencies, resulting in inflexible schedules.

threats

- Precision Overemphasis: Excessive focus on hour-based estimates can cause stress when actual outcomes differ.

threats

- Precision Overemphasis: Excessive focus on hour-based estimates can cause stress when actual outcomes differ.

- Estimation Bias: Biases like anchoring or optimism can distort hour-based estimates, leading to inaccuracies.

threats

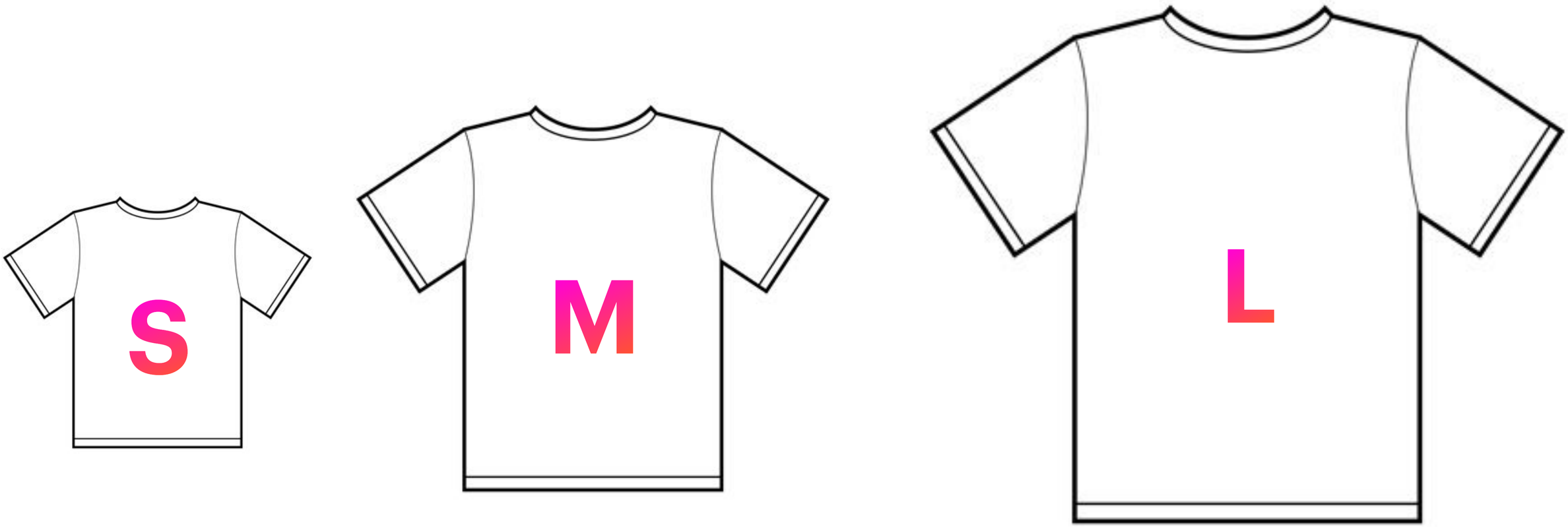
- Precision Overemphasis: Excessive focus on hour-based estimates can cause stress when actual outcomes differ.
- Estimation Bias: Biases like anchoring or optimism can distort hour-based estimates, leading to inaccuracies.
- Resource Constraints: Limited resources or expertise may hinder accurate hour-based estimation, especially for complex projects.

threats

- Precision Overemphasis: Excessive focus on hour-based estimates can cause stress when actual outcomes differ.
- Estimation Bias: Biases like anchoring or optimism can distort hour-based estimates, leading to inaccuracies.
- Resource Constraints: Limited resources or expertise may hinder accurate hour-based estimation, especially for complex projects.
- External Factors: Changes in project scope, market conditions, or technology may impact the accuracy of hour-based estimates.

relative estimation

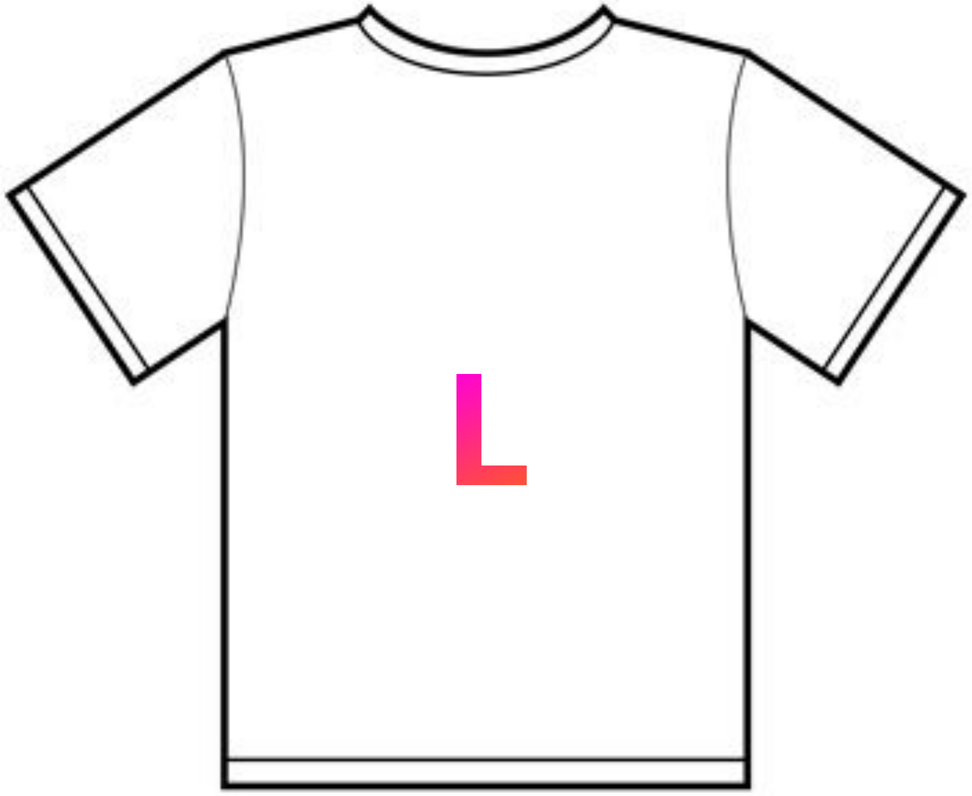
t-shirt size estimation



extended (affinity) t-shirt size estimation

small large

small large



small large

the bucket system

0	1	2	3	4	5	8	13	20	30	50	100	200
Splash Screen TODO	Login Using 4 Digit Password TODO	Sync Down TODO	Search Catalogue TODO		Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Admin Sign in with 4 Digit PIN TODO	Admin Sign Out TODO
	Log Out TODO	Select Option TODO			Add Text Here TODO		Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Admin Sign in using Username TODO	
	Auto Log Out TODO						Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO		
							Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO		
							Add Text Here TODO	Add Text Here TODO	Add Text Here TODO	Add Text Here TODO		

- In this technique different buckets are created with values ranging from: 0, 1, 2, 3, 4, 5, 8, 13, 20, 30, 50, 100, and 200. Here the estimator placed stories according to the suitability
- Bucket system is much quicker than playing planning poker

https://www.techagilist.com/agile/scrum/bucket-system-agile-estimation-method/

planning poker

https://www.youtube.com/watch?v=TxSzo3lwwWQ

DOT VOTING

A

B

C

D

E

F

- Dot voting is typically used for prioritization rather than directly for complexity estimation. However, you can adapt the dot voting technique to indirectly estimate complexity by assessing the perceived complexity of different items within a list. Here's how you can use dot voting for complexity estimation in Agile:

- Identify the Items to Be Estimated: Compile a list of user stories, tasks, or features that need to be estimated for complexity. These could be items from the product backlog or tasks for an upcoming sprint.

- Distribute Dot Voting Tokens: Provide each team member with a certain number of dot voting tokens. These tokens represent their votes for the complexity of the items.

- Understand the Items: Before voting, ensure that the team understands each item on the list. Discuss the details, requirements, and any known challenges associated with each item.

- Vote on Complexity: Ask team members to use their dot voting tokens to vote on the complexity of each item. They can distribute their votes across multiple items or concentrate them on a few items they perceive as more complex.

- Tally the Votes: Count the number of votes each item received. The number of votes indicates the perceived complexity of each item within the team.

- Discuss the Results: Facilitate a discussion to understand why certain items received more votes than others. Encourage team members to share their insights into the perceived complexity of each item.

- Refine Estimates: Use the insights gained from the dot voting exercise to refine complexity estimates for the items. Items with more votes may be considered more complex and may require additional time or resources during implementation.

- Iterate as Needed: Dot voting can be used iteratively to continually reassess the complexity of items as the project progresses. Regularly revisit complexity estimates to ensure they remain accurate and up to date.

another way of estimation

story points (SP) - relative units

etalon task

story points (SP) - relative units

the uncertainty paradox

fibonachi sequence

what do we estimate

dev, test, devops, documentation, ...

velocity chart

why it does make sense ?

convert SP to H

agile way of estimation

https://www.youtube.com/watch?v=Hwu438QSb_g

practical assignment

https://shorturl.at/yGNSO

t-shirt size estimation

extended t-shirt size estimation

planning poker