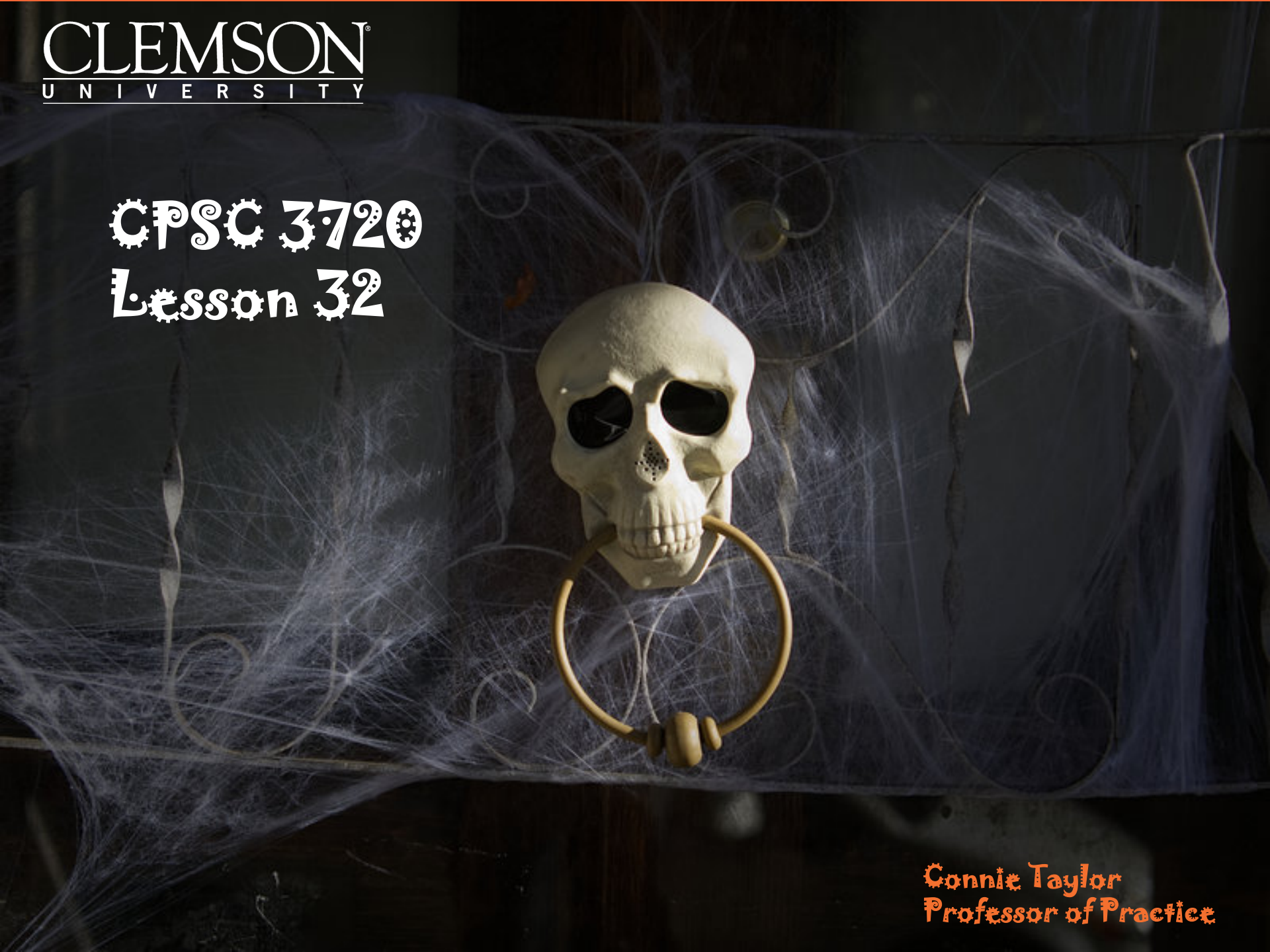
CLEMSON UNIVERSITY

CPSC 3720
Lesson 32



Connie Taylor
Professor of Practice

Announcements

- Read my latest announcement on Canvas if you are interested in registering for 4910 next semester
- Nov 4 class is project lab (work during class)
- Sprint 2 due on Nov 6 (you will need to submit info on the specific Collections and APIs I should review – see submission details and Rubric
- I still see issues with Team DFDs. See comments in Canvas

Planning Software Projects is EXTREMELY HARD

In The Mythical Man Month Fred Brooks gives five reasons this is so hard:

1. Our techniques of estimating are poorly developed. More seriously, they reflect an unvoiced assumption which is quite untrue, i.e., that all will go well.

2. Our estimating techniques fallaciously **confuse effort with progress**, hiding the assumption that people and months are interchangeable.

Planning Software Projects is EXTREMELY HARD

3. Because we are uncertain of our estimates, software managers often lack the courteous stubbornness of Antoine's chef

4. Schedule progress is poorly monitored. Techniques proven and routine in other engineering disciplines are considered radical innovations in software engineering.

5. When schedule slippage is recognized, the natural (and traditional) response is to add manpower. Like dousing a fire with gasoline, this makes matters worse, much worse.

Much of software development is sequential

"The bearing of a child takes nine months, no matter how many women are assigned."

Or "9 women in a month cannot make a baby"

Software development is a systems effort

"Since software construction is inherently a systems effort—an exercise in complex interrelationships—communication effort is great, and it quickly dominates the decrease in individual task time brought about by partitioning. Adding more [people] then lengthens, not shortens, the schedule."

Fred Brooks Planning Rule of Thumb

1/3 planning

1/6 coding

1/4 component test and early system test

1/4 system test, all components in hand

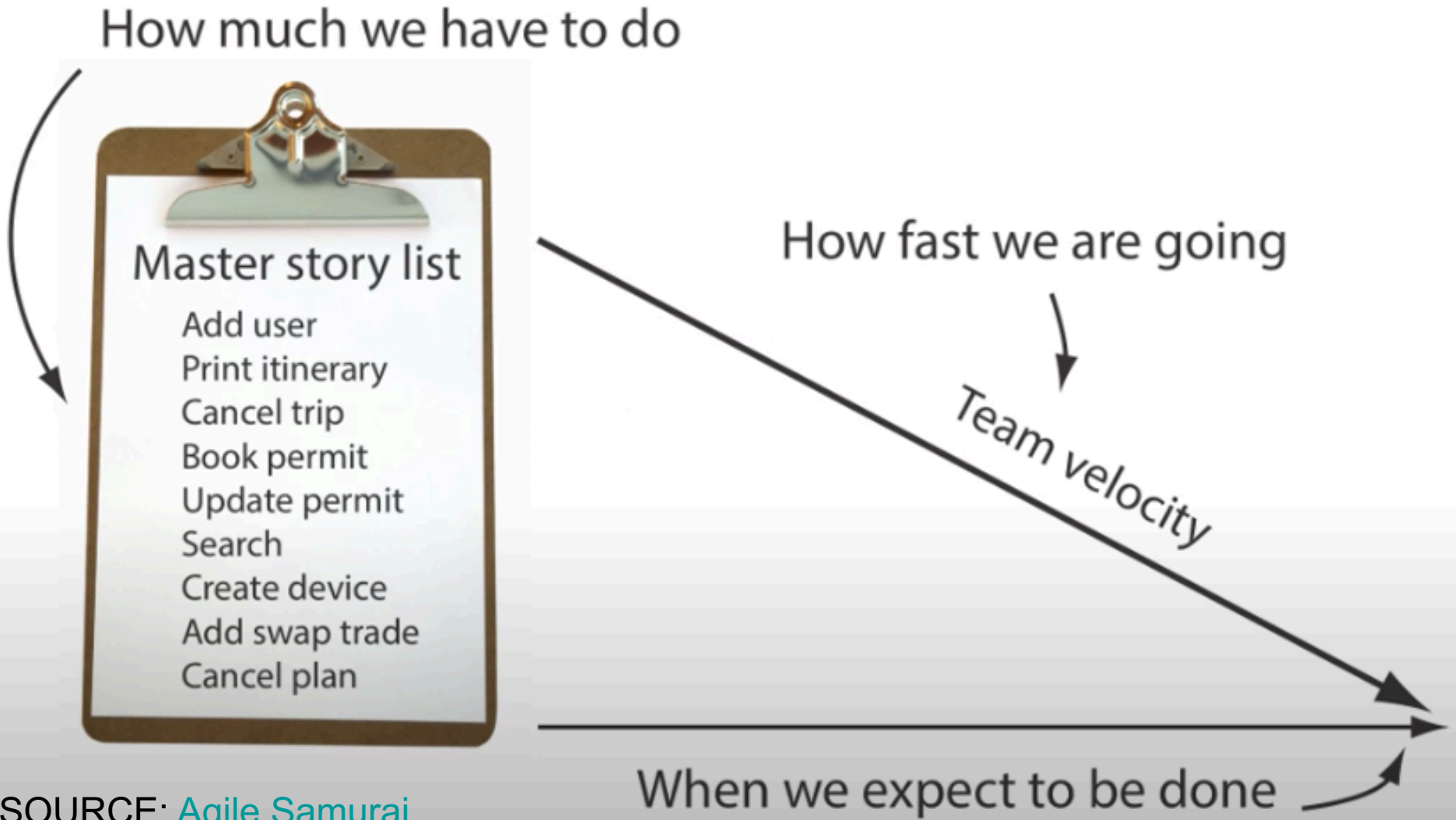
How does Agile help with planning?

How does Agile help with planning?

- Agile embraces the uncertainty in software development

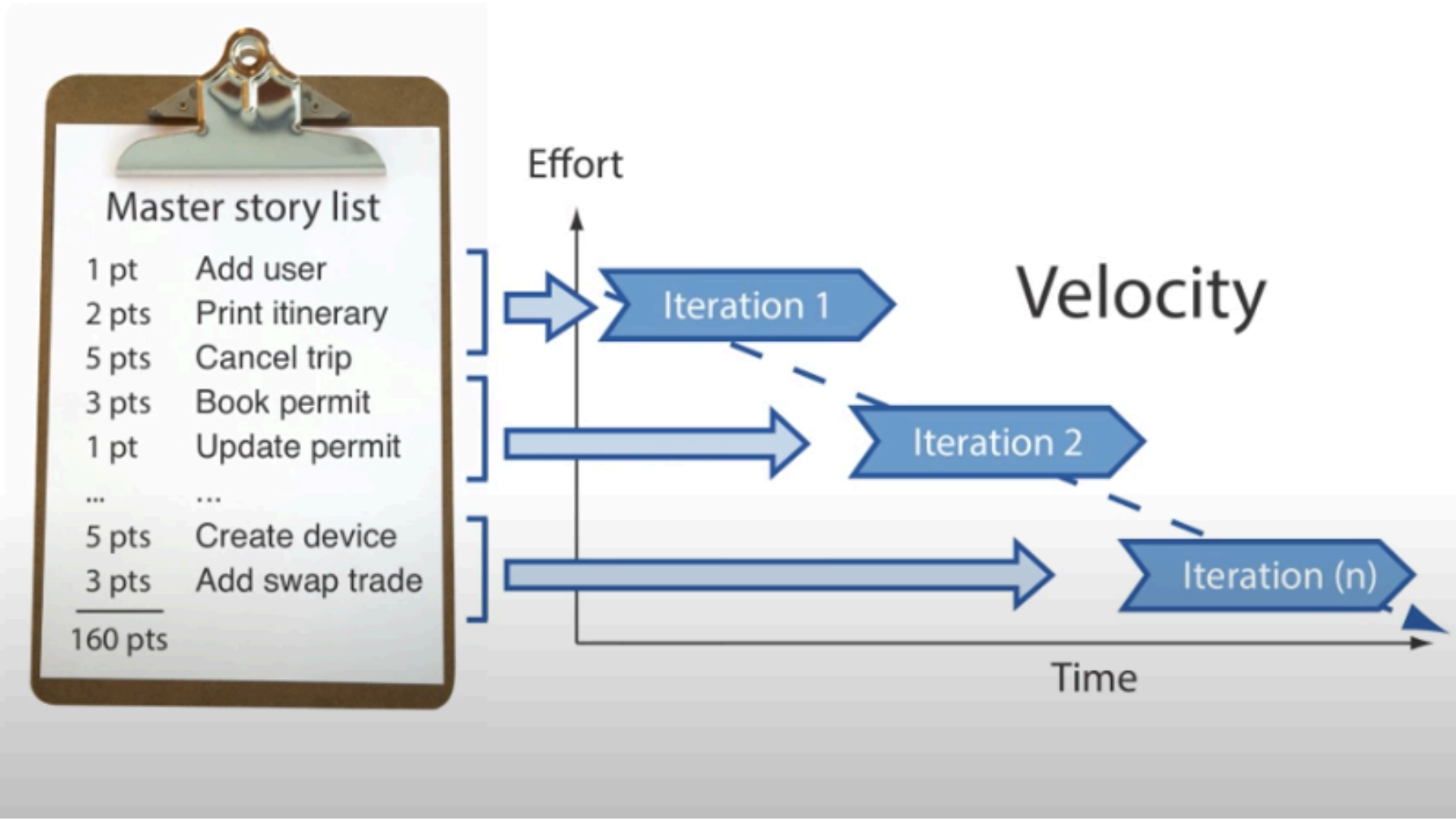
- Agile focuses on creating plans that:
  - Are highly honest and visible
  - Focus on customer value in the shortest amount of time
  - Allow you to change course
  - Enable frequent progress reviews with each sprint

The Agile Plan



SOURCE: Agile Samurai

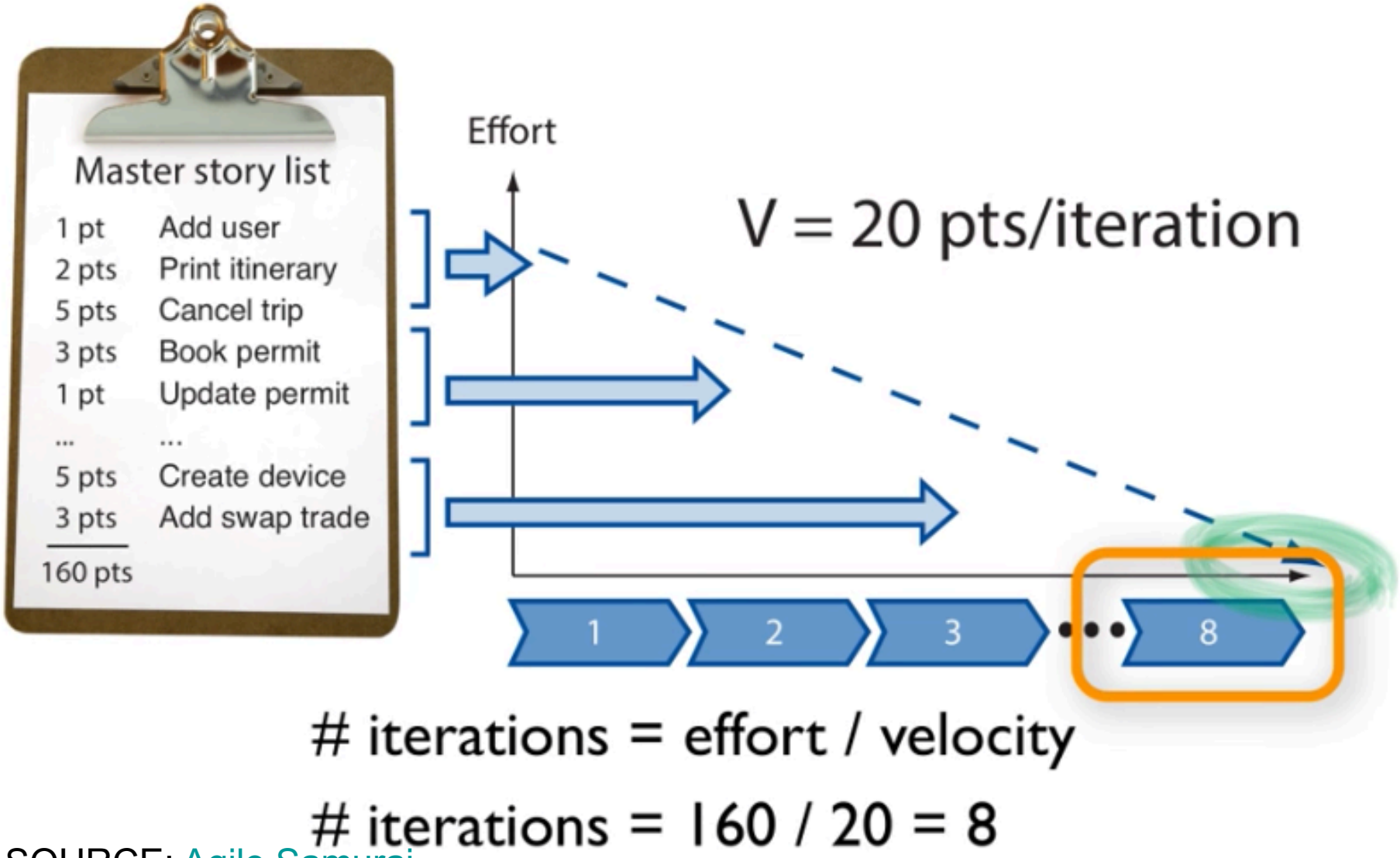
The Agile Plan



SOURCE: Agile Samurai

NOTE: Iteration = Sprint

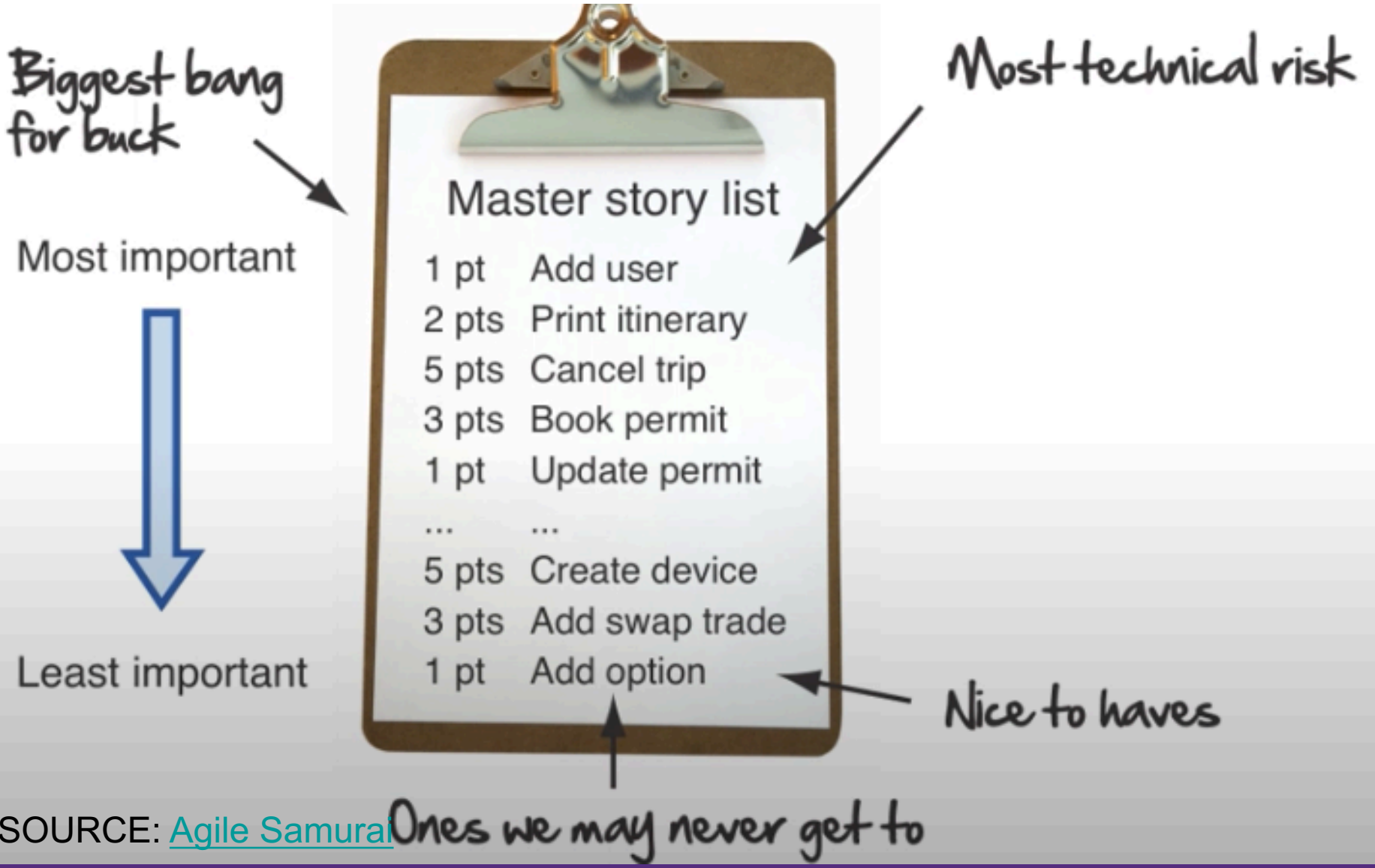
The Agile Plan – First "Guess"



SOURCE: Agile Samurai

Committing to the Initial Plan= The Death March

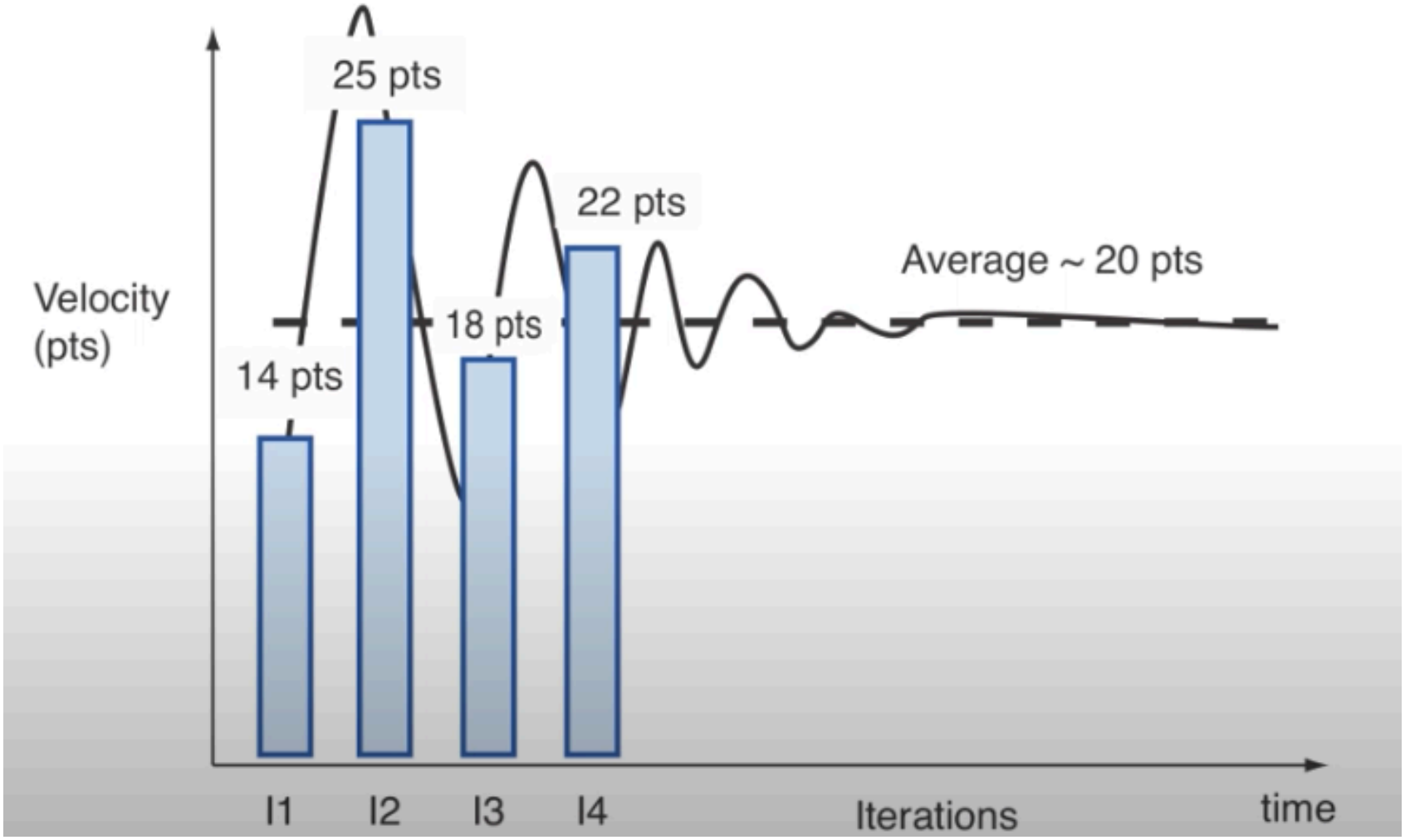
Prioritize!!! And this may Change!



SOURCE: Agile Samurai

Velocity fluctuates too!

What impacts velocity?



SOURCE: Agile Samurai

After Every Sprint

- Review Velocity and adjust
- Review Release Burndown and adjust
- Revisit backlog estimates and adjust
- Review Backlogs and adjust

What do we do if any of these change from our original plan?

Ways to Adjust

- Move the date
- Adjust features
- Add people

Which is best?

Ways to Adjust

- Move the date


Adjust features

- Add people

Nothing gets software done like a date!

Reading for Next Class

Two (short) chapters in The Dynamics of Software Development by Jim McCarthy (O'Reilly book)

Don't Go Dark

Beware of a Guy in a Room

Planning Poker

https://play.planningpoker.com/play/game/8PS0xUGHjvVFF0WXiueF6Ku9mUhe4nNZ

Reading for Friday

Ch. 2 of the Mythical Man Month (O'Reilly book)

"The Mythical Man Month"

Also, we will play Poker on Friday!