UMass Boston CS 410
Homework 3
Posted Monday, May 5, 2025
Due Wednesday, May 14, 2025 at 11:59 pm

0 General Instructions

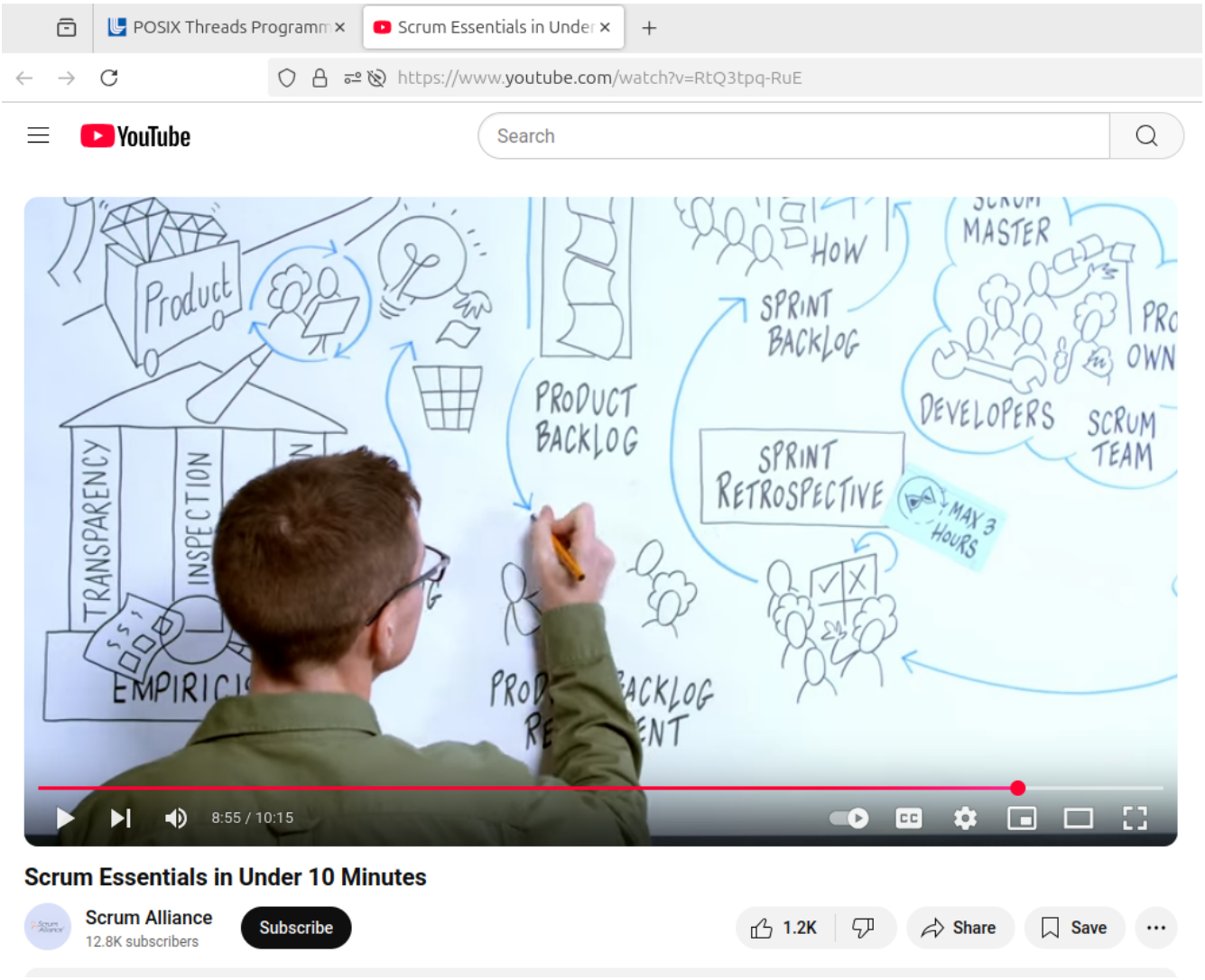
Homework must be typed and converted to Portable Document Format (PDF), see `https://en.wikipedia.org/wiki/PDF`. If you have a problem, request an extension before the work is due and explain how much you have done as well as your reason. We use the Linux servers of the Computer Science Department to collect homework. Homework submitted by email will be sent back.

To submit your homework, prepare one PDF file called `hw3.pdf` — the filename must be exactly `hw3.pdf`, otherwise it will not be collected. Upload the file to the `cs410` folder linked to your home directory on the CS Linux server. If you have trouble with uploading, email `operator@cs.umb.edu` for immediate help and copy the instructor at `jane.deblois@umb.edu`. The questions in this homework are based on the reading in Chapters 3 and 5-8 in Essential Scrum by Kenneth S. Rubin and the agile principles at `https://www.agilemanifesto.org`.

1 Q0 - already done in class, 20 points, separately graded

This task required viewing the youtube `https://www.youtube.com/watch?v=RtQ3tpq-RuE` in which Stewart teaches us how to scrum. You were asked to describe the changes made as a result of the sprint2 review and retrospective.

You submitted work in class, written on the following picture of Stewart at work. It has been graded separately.



2 Looking at Chapter 2 - Scrum Framework, pp16-28

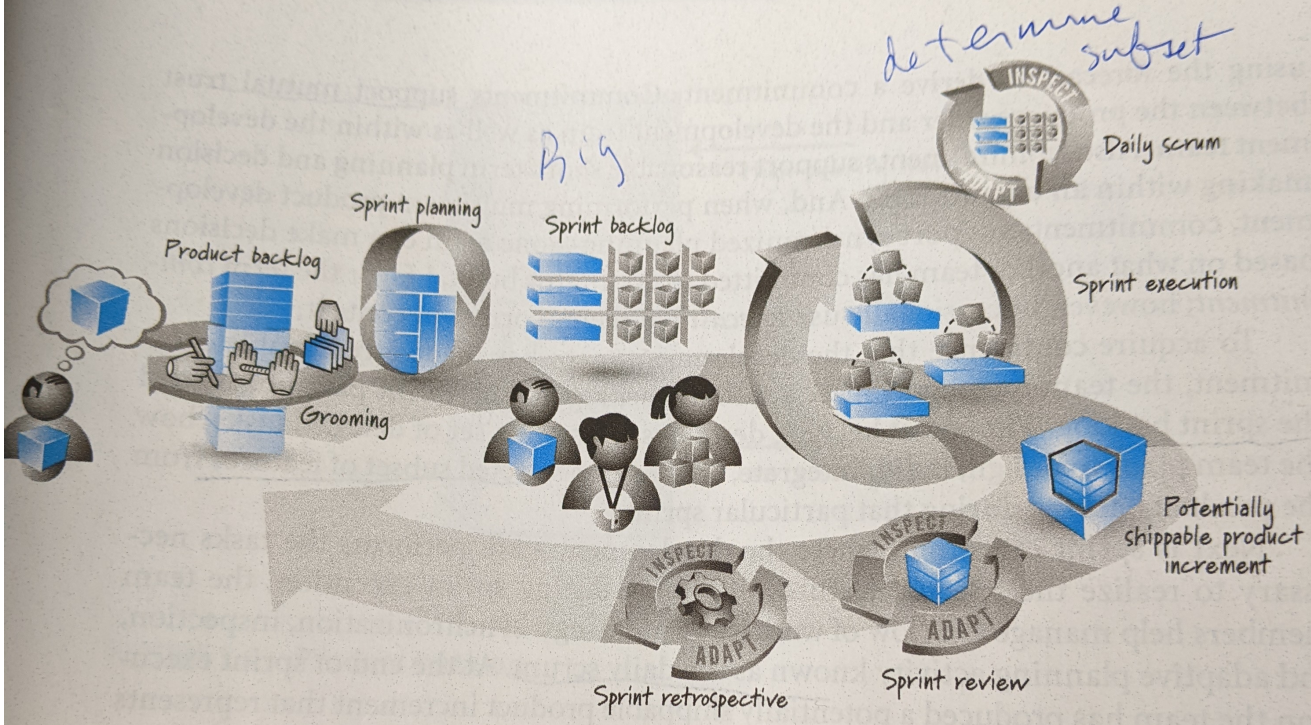
For this question, please reread pages 16-28 in Essential Scrum.

Please compare the two summary figures we have studied: Stewart's picture and Figure 2.3 Scrum framework in Rubin.

Stewart goes one step further – he addresses values. He talks about empiricismand lean thinking. He emphasizes that scrum is based on transparency, inspection and adaptation. This might mean some of your code is put aside and not included in the final product.

If you accept this as worthwhile, then you are willing to write code to build something in a simple form so everyone can run it. We all know that product ideas that are abstract may not turn out so well. It is critical for speed in development to try things to check on feasibility. This is a skill to list on your T-Shaped Skills.

a. (10 points) Write 5 sentences to explain to a new team member why the values mentioned above mean that you might build code that might not be part of the final product.



URE 2.3 Scrum framework

Let's summarize the diagram, starting on the left side of the figure and workin clockwise around the main looping arrow (the sprint).

The product owner has a vision of what he wants to create (the big cube). Becaus the cube can be large, through an activity called **grooming** it is broken down into set of features that are collected into a prioritized list called the product backlog.

A sprint starts with sprint planning, encompasses the development work durin the sprint (called sprint execution), and ends with the review and retrospective. Th sprint is represented by the large, looping arrow that dominates the center of the fi ure. The number of items in the product backlog is likely to be more than a develop ment team can complete in a short-duration sprint. For that reason, at the beginnir of each sprint, the development team must determine a subset of the product backlo items it believes it can complete—an activity called sprint planning, shown just the right of the large product backlog cube.

3 Looking at Chapter 6: Product Backlog

For this question, please reread pages 99-118 in Essential Scrum.

Review Figure 6-2. Product Backlog Items. Note the four types of items. When you adapt your work, you may well need to note a defect or perform a knowledge acquisition task to help correct a problem.

Please also review Figure 6-10. Definition of Ready and Table 6.2 Example Definition-of-Ready Checklist. Note how the discussion of whether a PBI is ready makes it possible to determine whether the item is done and can be added to the potentially shippable product increment.

a. (10 points) From your work (on any sprint), describe a feature that you were working on that turned out to have a defect. Then describe how you fixed it. Was knowledge acquisition necessary? Use 2-5 sentences.

b. (5 points) In scrum we often focus on increasing the business value of the product. Name a feature you have built and explain its business value. See definition of user story on p421. Give a description of the feature you chose and format it as a requirement that has the three parts: As a ¡user role¿ I want to achieve ¡goal¿ so that I get ¡benefit¿.

c. (5 points) Think of a feature that you built that had to later be changed or expanded. Write the three-part requirement for what you built first. Then write the three-part requirement for the revised feature.

As long as there is a product or system being built, enhanced, or supported, there is a product backlog.

Product Backlog Items

The product backlog is composed of backlog items, which I refer to as PBIs, backlog items, or simply items (see Figure 6.2).

Most PBIs are features, items of functionality that will have tangible value to the user or customer. These are often written as user stories (although Scrum does not specify the format of PBIs). Examples of features include something brand-new (a login screen for a new website), or a change to an existing feature (a more user-friendly login screen for an existing website). Other PBIs include defects needing repair, technical improvements, knowledge-acquisition work, and any other work the product owner deems valuable. See Table 6.1 for examples of the different types of PBIs.

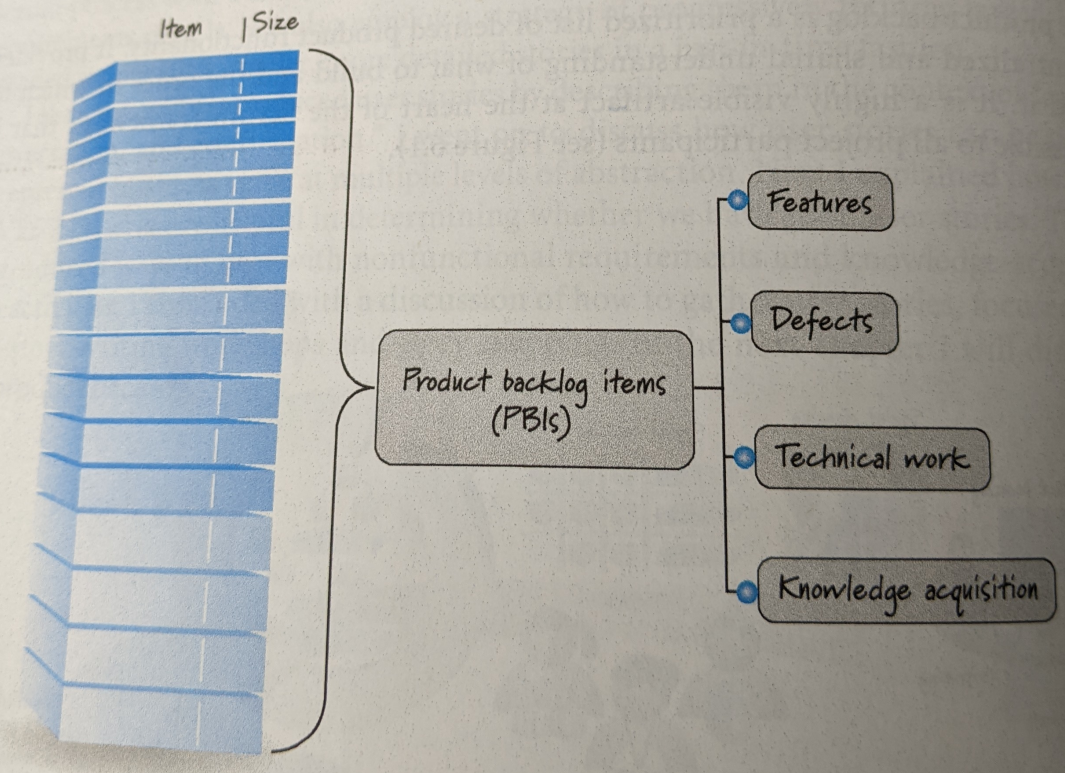


FIGURE 6.2 Product backlog items

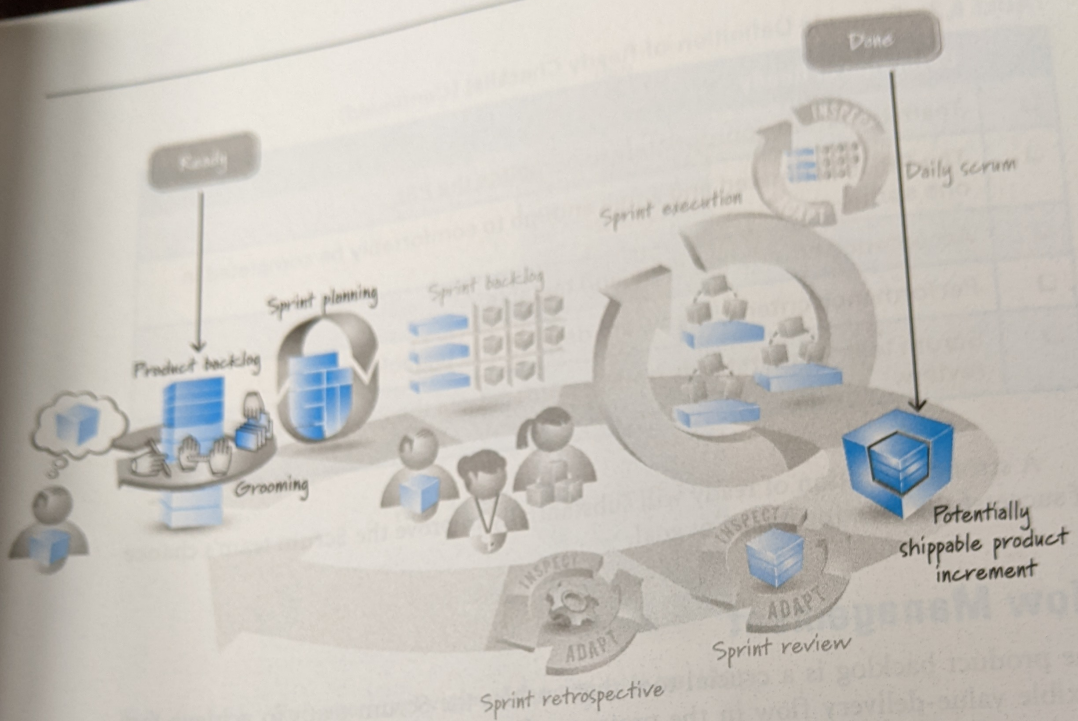


FIGURE 6.10 Definition of ready

Some Scrum teams formalize this idea by establishing a **definition of ready**. You can think of the definition of ready and the definition of done (see Chapter 4) as two states of product backlog items during a sprint cycle (see Figure 6.10).

Both the definition of done and the definition of ready are checklists of the work that must be completed before a product backlog item can be considered to be in the respective state. An example of a definition-of-ready checklist for product backlog items is given in Table 6.2.

TABLE 6.2 Example Definition-of-Ready Checklist

	Definition of Ready
❑	Business value is clearly articulated.
❑	Details are sufficiently understood by the development team so it can make an informed decision as to whether it can complete the PBI.
❑	Dependencies are identified and no external dependencies would block the PBI from being completed.

continues

4 Looking at Chapter 15 - Multilevel Planning

Please read pages 257-265 in Essential Scrum.

Look closely at Figure 15.7 Hierarchical Scrum Planning. Consider whether you could use this figure as system diagram of your team's long project.

a. (10 points) Use the diagram to write 5 sentences about your team's project. Sentence 1 should state the overview and mention the first product backlog list. Sentences 2, 3 and 4 should each describe later sprints. Sentence 5 can be your summary of what you believe the final product will look like.

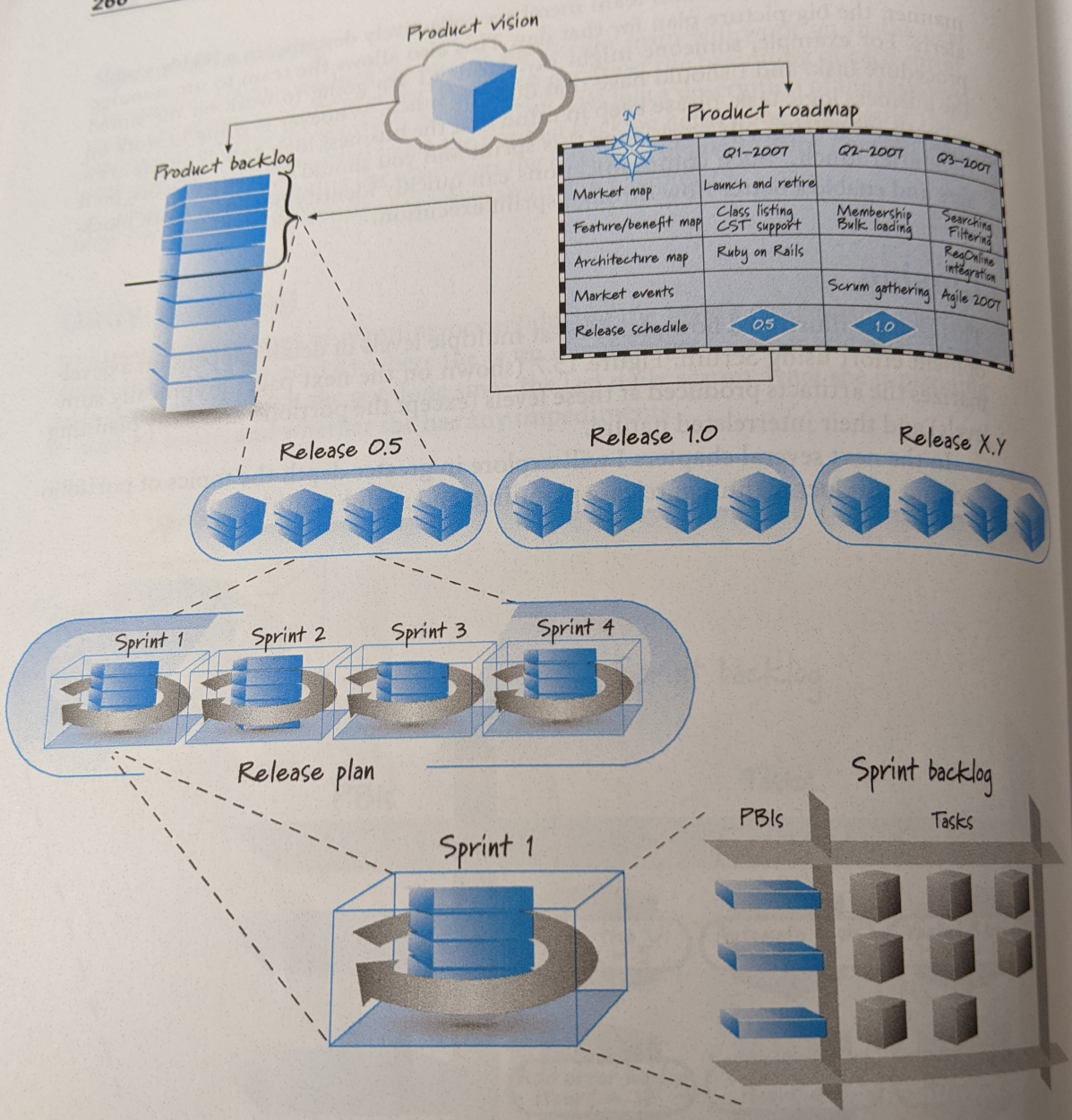


FIGURE 15.7 Hierarchical Scrum planning

5 Looking at Chapter 19 - Sprint Planning

For this question, please read pages 343-346 in Essential Scrum. Look at Figure 19.6 Sprint Backlog showing PBIs and task plan. Each feature from the product backlog that is being done in this sprint is broken down into a set of tasks that are placed in the chart left to right. Estimates of time required are given.

a. (5 points) Identify where the graders can look on the CS servers to find this chart for sprint3 of your team. It may not be finished yet. Select a location in GROUP2 directory and work on it together.

b. (10 points) Identify the lines of the chart that show all the subtasks for your work for sprint3. Make this subset of the chart into a spreadsheet and save it on the CS server in your course directory. Identify where it is by file name.

c. (5 points) Identify where and in what file you make notes about your progress. This is a personal artifact.

Code
Refactor
Test

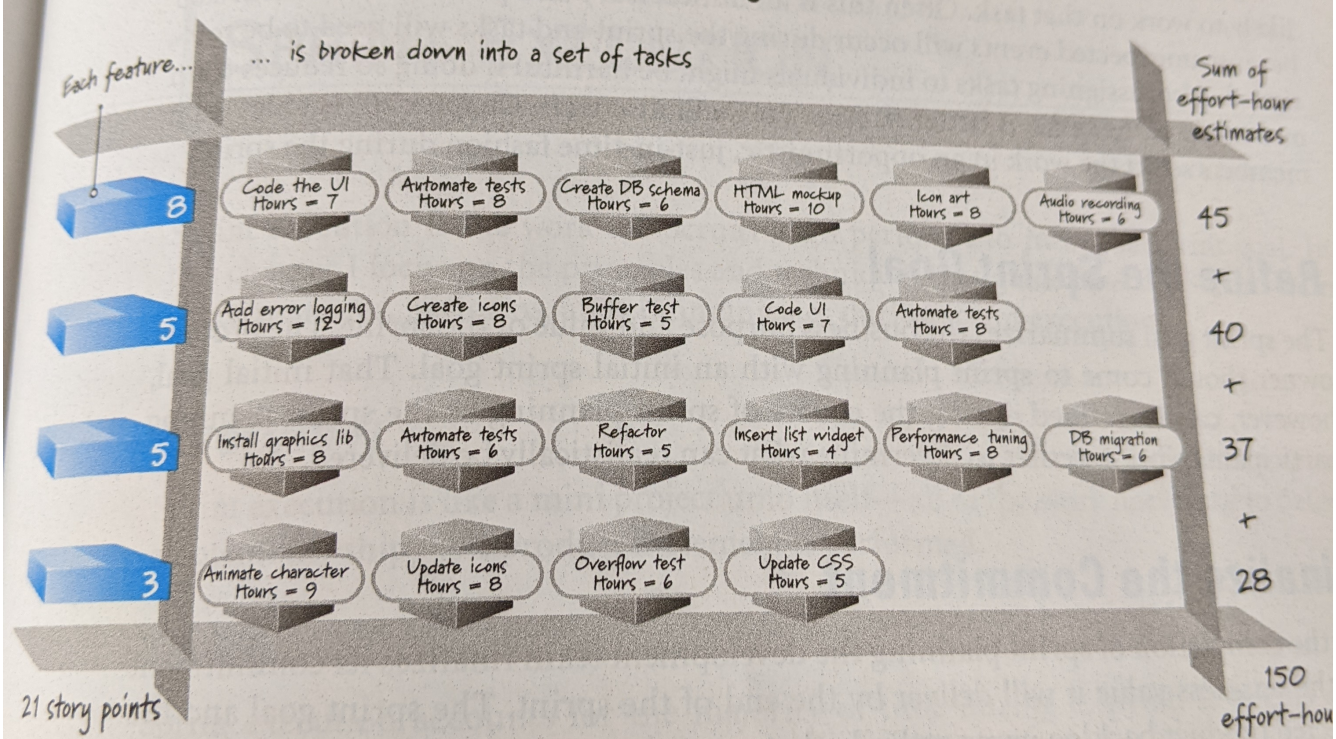


FIGURE 19.6 Sprint backlog showing PBIs and task plan

If the team's predicted velocity is 25 points, a commitment of 21 points see reasonable. But let's use the task-level details to see if the commitment still lo good. The sum of the tasks for all four product backlog items is 150 effort-ho Assume that the team for this sprint is the team identified in Table 19.2, which commitment of 150 effort-h

6 Looking at Chapter 20 - Sprint Execution

For this question, please read pages 347-361 in Essential Scrum. Look at Figure 20.5 Subset of Extreme Programming Technical Practices which summarizes some excellent technical practices that you have studied.

Then turn the page and look at Figure 20.6 Example Task Board to see how Rubin (and others including Tom Mullaly) recommend that we track our own daily progress (as a team or individually).

These task boards are sometimes called Kanban boards.

a. (10 points) How many of the technical practices have you used this semester? Please list them and describe in a sentence where you used it.

b. (10 points) Create your own task board for your start of sprint3, take a photo of it, include it and then provide one update of it a few days later. Be sure each of your tasks are represented as reasonably sized subtasks listed across the row. In the second picture, each subtask should appear and some should be listed as in progress or possibly as completed.

Task Performance—Technical Practices

Development team members are expected to be technically good at what they do. I'm not saying that you need a team of superstars to use Scrum. However, working in short, timeboxed iterations where there is an expectation of delivering potentially shippable product increments does exert pressure on teams to get the job done with good control over technical debt. If team members lack appropriate technical skills, they will likely fail to achieve the level of agility needed to deliver long-term, sustainable business value.

If you are using Scrum to develop software, team members need to be skilled in good technical practices for developing software. I'm not referring to esoteric skills but instead to skills that have been in use for decades and are essential to being successful with Scrum or arguably any software development approach—for example, continuous integration, automated testing, refactoring, test-driven development, and so on. Today the agile community refers to many of these technical practices as Extreme Programming (Beck and Andres 2004), but most are practices that predate that label (see Figure 20.5 for a subset of the Extreme Programming technical practices).

As an example, consider automated testing, which is necessary to support several of the practices in Figure 20.5. Development teams that don't focus on automating their tests will quickly start to slow down and take ever-increasing risks. At some point, it could take all of the sprint execution time just to manually rerun the regression tests for previously developed features. In such cases, the team might choose not

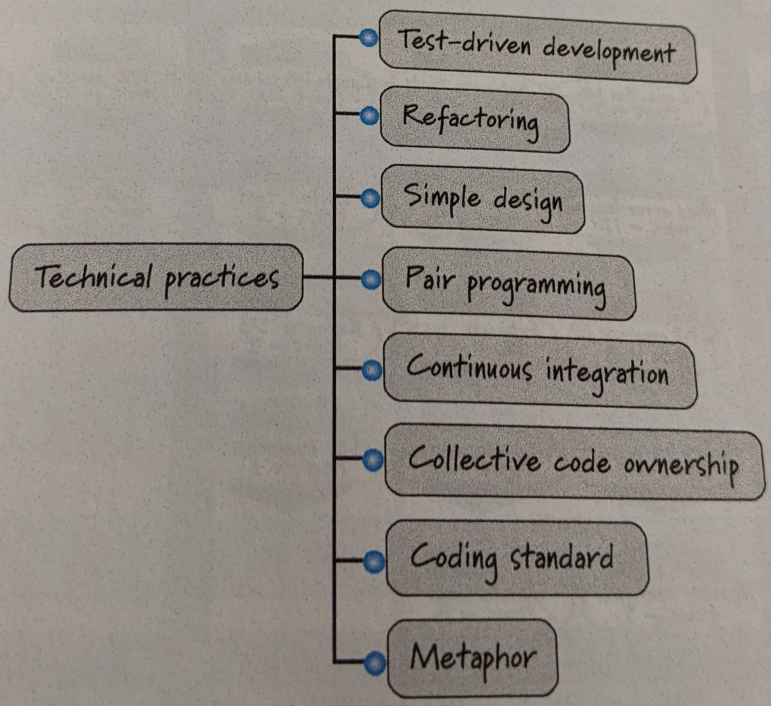


FIGURE 20.5 Subset of Extreme Programming technical practices

to rerun all of the manual tests each sprint, which could allow defects to propagate forward, adding to the system's technical debt (increased risk). You won't be agile for very long if you don't start automating your tests.

Similar arguments can be made for other core technical practices. Most teams achieve the long-term benefits of Scrum only if they also embrace strong technical practices when performing task-level work.

Communicating

One of the benefits of working in short timeboxes with small teams is that you don't need complex charts and reports to communicate progress! Although any highly visible way of communicating progress can be used, most teams use a combination of a task board and a burndown and/or burnup chart as their principal **information radiator**.

Task Board

The **task board** is a simple but powerful way to communicate sprint progress at a glance. Formally, the task board shows the evolving state of the sprint backlog over time (see Figure 20.6).

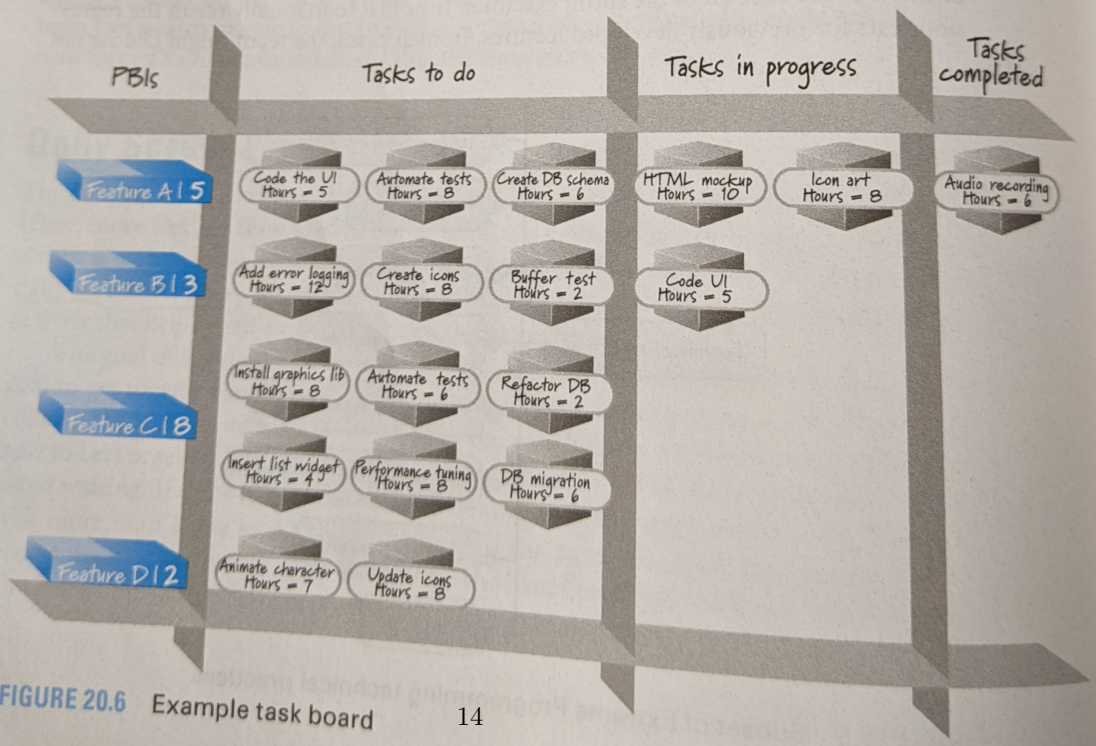


FIGURE 20.6 Example task board