# Task Tracking	start time: 9:15

This activity explores ways to **track tasks** for a **project** with many people, and tasks that vary in importance, time, and skill required, and that may depend on each other. The activity focuses less on **events** that must occur at specific times (e.g. classes, meetings) and more on **tasks** that can be scheduled with some flexibility.

Once you understand concepts, it is easier to learn about specific tools in the future.

Before you start, complete the form below to assign a role to each member. If you have 3 people, combine Speaker & Reflector.

Team	**Date**
Lewis, Scott, Anastasia.S,Aidan (Team G)	**2017/09/19**
Team Roles	**Team Member**
Recorder: records all answers & questions, and provides copies to team & facilitator.	Lewis
Speaker: talks to facilitator and other teams.	Scott
Manager: keeps track of time and makes sure everyone contributes appropriately.	Anastasia
Reflector: considers how the team could work and learn more effectively.	Aidan

Notes

1. Write legibly & neatly so that everyone can read & understand your responses.

a. Note the time whenever your team starts a new section or question.

Task Tracking Worksheet

Do ***not*** *start to fill in the worksheet*. Read each question below, and for each **attribute**, rank each task tracking **method** (a, b, ...) from **1 (least)** to **5 (most)** (ties are OK).

	Question:	**A.2.**	**A.3.**	**A.4.**		**B.2.**	**B.3.**	**B.4.**
	Solo or Team:	**Solo**	**Solo**	**Solo**		**Team**	**Team**	**Team**
	Attribute:	**Access ible**	**Flex ible**	**Scal able**		**Access ible**	**Flex ible**	**Scal able**
a.	In your head.	5	5	1		1	1	1
b.	On many separate sticky notes.	2	5	3		2	4	3
c.	On a single sheet of paper.	3	4	3		2	2	1
d.	On a whiteboard.	1	4	3		1	1	2
e.	In a file or app on a laptop or desktop.	4	5	4		4	4	5
f.	In a file or app on a phone or tablet.	4	5	4		5	4	4
g.	In a file or app on a shared server.	5	5	4		5	4	5
h.	On a web server in the cloud.	4	5	4		5	4	4

i.	Meeting in Person	5	5	5		1	1	3

(12 min) A. Individual Task Lists	start time: 9:30

There are many situations where you need to track **tasks** you need to do.

For example:
- Teachers plan classes, meet with students, and create and grade homework.
- Chefs cook multiple dishes for many customers.
- Software developers work on many defects or features, each of which may involve many files, classes, methods, and related documentation.

You could use a variety of methods to manage your list of tasks (see the worksheet above). You may add 1-2 other approaches to the list if you want do.

1. (1 min) Review the methods in the worksheet:

a.	Which involve technology?	e,f,g,h
b.	Which are portable?	a,b,c,e,f,g,h

2. (2 min) A task list should be easy to access as you move around between locations. In the **worksheet**, **rank** each method from 1 (least) to 5 (most) **accessible**.

3. (2 min) With your task list, it should be easy to add, remove, and rearrange tasks. In the **worksheet**, **rank** each method from 1 (least) to 5 (most) **flexible**.

4. (2 min) Some methods might have problems if you have many (10 s? 100 s?) of tasks. In the **worksheet**, **rank** each method from 1 (least) to 5 (most) **scalable**.

5. (2 min) Describe any **other issues** that might be relevant for **individual tasks**.

If the tasks are too complex it could take too long. Some of the platforms are to complex for one person.

6. (3 min) In complete sentences, summarize the advice you would give to a friend considering how to track tasks for their own project.

You should use a platform that you can access anywhere, that has enough space for whatever you are doing. Make sure that if it is electronic that you know how to use it and it’s functions.

(12 min) B. Team Task Lists	start time: 9:45

Teams (2+ people) also need to track tasks, which are often interrelated. Thus:

- Chefs specialize in tasks or types of food, but a set of dishes should be ready to serve at the same time.
- Software development often involves coordination between developers, parts of the program, and activities (e.g. analysis, design, implementation, testing). As a result, teams may require or prefer different task tracking methods.

1. (1 min) If you can think of other task tracking methods, add them to the list.

- Meeting in person

2. (2 min) For a team, a task list should be easy to access for everyone in the team whether they are face-to-face, working at different times, or in different places. In the **worksheet**, **rank** each method from 1 (least) to 5 (most) **accessible**.

3. (2 min) For a team, it should be easy to add, remove, and rearrange tasks, reassign tickets among people, and track relationships between tasks, such as subtasks of a larger task, or tasks that must be done in a particular order. In the **worksheet**, **rank** each method from 1 (least) to 5 (most) **flexible**.

4. (2 min) A large team could easily have thousands (or tens of thousands) of tickets, and hundreds of people spread across many countries, time zones, and languages. In the **worksheet**, **rank** each method from 1 (least) to 5 (most) **scalable**.

5. (2 min) Describe any other issues that might be relevant for tracking team tasks

Team not cooperating, or on the same page with one another.

6. (3 min) In complete sentences, summarize the advice you would give to a team considering how to track tasks for their own project.

Make sure that everyone is on the same page. Make sure it's easy for all team members to access and that all team members know how to use the platform or show them how. Set rules and restrictions on how they should follow through with the tasks.

(20 min) C. Issues to Consider	start time: 10:00

1. (3 min) Some tasks have **multiple steps**. For example, a software developer might design, code, test, and document a software library.

Describe reason(s) to use:

a. One task for all of the steps. b. A separate task for each step.

a.) One task can get really confusing if there all together, where as a separate task for each is easier because people won't get as confused
b.) If there are multiple steps for one task you can modularize it, which will break down the more complicated steps.

2. (3 min) Some tasks involve **multiple people**. For example, a graphic designer, a developer, and a database guru might work together on a phone app.

Describe reason(s) to use:

a. One task for all of the work. b. A separate task for each person's work.

A.) For one task people can get confused as to what they have to do, or what step they are on.
B.) If there are separate tasks each person can focus on one job, which in the end would make the process simpler.

3. (3 min). A team leader or manager must be able to quickly summarize project status.

For a project with many tasks, how could we determined the project status from:

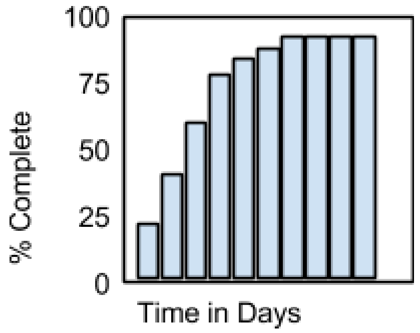
a. the **state** (e.g. *not started, started, finished*) of each task?

With state, you can only know what tasks are started, finished, and not started

b. the **percent complete** (e.g. *0%, 50%, 90%*) for each task?

How much of it is actually complete/done, can be a more accurate representation of how much of the work is done.

4. (3 min) The graph at right shows a task's percent complete over a sequence of days.



a. What happens to the % complete over time?

The percentage should go up

b. How might this happen with a real task?

The more of the task/project that you complete

c. If this occured with many tasks how could it distort a manager's view of the project?

It would show a manager that people were working on the task then moved onto a new task.

5. (3 min) In complete sentences, explain the advantages of tracking project status using a **large number of small tasks**.

It makes it easier to know which tasks are complete. As well as which have yet to be started. It is easier to assign different people to different tasks, so there is less overlapping. (e.g. when multiple people are trying to edit the same project at the sametime)

6. (3 min) Some tasks (e.g. defects or new features), may be discussed and debated before they are started or completed. Explain why a task tracking system could be a good place to save or summarize this discussion, and any associated documents (e.g. requirements, sketches, screenshots).

A.) A task tracking system help people's ideas get expressed to a large group of people in a simply way.

(20 min) D. Task Tracking Systems & Concepts	start time: 10:25

A **task tracking system (TTS)** is a tool designed to help multiple people track and coordinate tasks. TTSs are commonly used in software development, particularly for defects (bugs) and enhancements, but can be used for other types of projects.

Your team has 8 screenshots from 4 different TTSs (Bugzilla, GitHub, Trac, and TikiWiki). (They have been edited to reduce page size and remove some extraneous information.)

The first screen for each TTS shows a **list of tasks**; the second shows **details for one task**.

1. (6 min) **Manager**, give each person one pair of handouts (skip one or double up as needed).

Individually, study the handouts and try to determine:

a. What sorts of **data** are shown for each task, typically?

Circle each piece of data, not each data values (e.g. "date" not "2001-12-31"). b. What sorts of **actions** can be performed on the page? Put a **star** next to each action.

2. (4 min) As a team, compare your findings and answers to the previous question, and add the most common data and actions to the table below.

	List of Tasks	**Detail on one Task**
a. What sorts of <u>data</u> are shown for each task, typically?	N/A	N/A
b. What are <u>typical actions</u> that can be performed on the page?	N/A	N/A

3. (6 min) A task tracking system (TTS) is usually implemented with a **database**, so that every task has a consistent set of **attributes** or **fields**.
(Many TTSs also support **custom fields** , so they can be adapted to local situations.) Review the handouts to **identify** the most common task fields.
For each field, decide whether it should be **required**, and describe the **possible values**. For example, the possible values might be dates, integers, names of people, or a value from a enumerated set of options.

Field Name	**Required or Optional**	**Possible Values**	**Comments**
Identifier	R	usually numeric, assigned automatically	ensures that every task has a unique identifier
State or Status	R	enumerated set, e.g. new/open/closed; new/started/done	makes it easy to see which tasks have been started, finished, etc.
N/A	N/A	N/A	N/A
N/A	N/A	N/A	N/A
N/A	N/A	N/A	N/A
N/A	N/A	N/A	N/A

N/A	N/A	N/A	N/A
N/A	N/A	N/A	N/A

(20 min) E. Explore	start time: 10:30

NOTE: If you do not have time to complete this section during class, do it for homework. **Individually or with one partner**, connect to a live TTS.

Choose a favorite open source project, or one of the following examples:

Apache https://issues.apache.org/bugzilla/ Audacity
http://wiki.audacityteam.org/wiki/Bug_Lists jQuery
https://github.com/jquery/jquery LibGit2
https://github.com/libgit2/libgit2
Moodle https://tracker.moodle.org/secure/Dashboard.jspa
TikiWiki http://dev.tiki.org

1. (2 min) Choose a site, and enter the site name and URL here:
https://github.com/jquery/jquery

2. (6 min) Try to find each of the following, and specify the task by ID or name / summary.

a.	A task that has been started but not finished.	CSS: Detect more website styles
b.	A task that has not yet been started.	N/A
c.	A task that has been completed.	.babelrc
d.	The most recently modified task.	CSS: Detect more website styles
e.	The most recently created task.	.mailmap
f.	The oldest task.	.gitattributes

g.	A task that seems very simple to complete.	.travis.yml
h.	A task that seems too complex, or poorly defined.	package.json

}