

CS 350: Introduction to Software Engineering

---

Slide Set 1  
C. M. Overstreet  
Old Dominion University  
Spring 2006

---

---

---

---

---

---

---

---

Lecture overview

---

- Announcements
- Class expectations
  - Structure, grading, project, logistics, semester structure
- PSP: what's it about
- TSP: what's it about
- PSP0: details

Spring 2006 CS 350/ODU 2

---

---

---

---

---

---

---

---

Announcements

---

- CS Dept. Systems groups looking for people
  - Contact Ajay Gupta.
- Any interest in working closely with students in Berlin, Germany on term project?
  - German not required! Everything in English
  - Trip to Berlin in Spring Break
    - Partial support for tickets likely
    - Free stay (almost) in Berlin likely
    - Need at least 5 people to make this work

Spring 2006 CS 350/ODU 3

---

---

---

---

---

---

---

---

## CS 350: Intro to SE

Texts: 1) *PSP: A Self-Improvement Process for Software Engineers* Watts S. Humphrey, Addison-Wesley, 2005.  
2) *Introduction to the Team Software Process*, Watts S. Humphrey, Addison-Wesley, 2000 (not used till we finish PSP text).

---

---

---

---

---

---

---

---

## Miscellaneous Class Information

- Class material is available on web ([www.cs.odu.edu/~cmo](http://www.cs.odu.edu/~cmo) under cs350)
- Class is time consuming
  - But more time spent on process than coding
- Reading assignments:
  - PSP, chapters 1 & 2

---

---

---

---

---

---

---

---

## Recitations - 1

- Recitations meet on Fridays
  - But not this week!

---

---

---

---

---

---

---

---

## Recitations - 2

---

- Recitations:
  - You must register for one
  - You must attend
  - Will be used for:
    - Group meetings (later)
    - Project discussions
    - Covering some software tools
    - Reviews of class performance on programming
      - Your data compared to class averages
    - Answering questions

Spring 2006

CS 350/ODU

7

---

---

---

---

---

---

---

---

## Announcements

---

- You must have a CS dept account:
  - Go to [www.cs.odu.edu](http://www.cs.odu.edu), pick **Online Services**, then select **Account Creation**.

Spring 2006

CS 350/ODU

8

---

---

---

---

---

---

---

---

## General Information

---

- Prerequisites
  - CS 361 or CS 330
  - UNIX exposure
- Helpful background
  - Some simple statistics (but we'll cover in class as needed)

Spring 2006

CS 350/ODU

9

---

---

---

---

---

---

---

---

## Course Overview

---

- **Activities:**
  - 5 programming assignments, PSP based
  - 1 team project, TSP based
  - 2 in-class exams & comprehensive final
  - recitation/class assignments
- **Grading:**
  - Individual projects: 25%
  - Team project: 20%
  - In-class exams: 25%
  - Recitation/class assignments: 10%
  - Final exam: 20%

Spring 2006

CS 350/ODU

10

---

---

---

---

---

---

---

---

## From the Syllabus

---

- Honor code
- Lateness policy
- Special needs
- Read the syllabus!

Spring 2006

CS 350/ODU

11

---

---

---

---

---

---

---

---

## Slides

---

- Available before class on Web
- Slides are OUTLINE only.
  - For content, read textbooks, assigned readings
  - For content, come to class
  - If class is missed, get GOOD notes from class member

Spring 2006

CS 350/ODU

12

---

---

---

---

---

---

---

---

## Structure/purpose of class projects

- Not about programming
  - Focus is on the process used to develop software products.
  - Learning the process involves using it to development some software.
- Five individual projects involving coding
  - However you will spend more time on process steps than coding
    - Industry data: of total project time, 15% is coding
  - Emphasize your Personal Software Process
- One team project
  - Emphasizes the Team Software Process

Spring 2006

CS 350/ODU

13

---

---

---

---

---

---

---

---

## Course Objectives 1: PSP

- Introduce you to a process-based approach to developing software
- Show you how to measure and analyze your Personal Software Process (PSP)
- Improve your software development skills:
  - faster development
  - fewer errors (i.e. better software)
  - more predictable (more accurate estimates of time required to complete a project)
- Show you how to use data to improve your personal performance

Spring 2006

CS 350/ODU

14

---

---

---

---

---

---

---

---

## Course Objectives 2: TSP

- Assumes large software project
  - Many people
  - Maybe > 100,000 KLOC
- Working on teams requires specific skills
- TSP goals:
  - Understand how to build teams
  - Understand different team roles
  - Understand how to work on teams

Spring 2006

CS 350/ODU

15

---

---

---

---

---

---

---

---

## One problem with teaching SE principles

- Students often don't see need
  - if they haven't worked on large software projects
  - if they haven't worked on projects where quality is required
    - Society increasing reliance on software whose failure can do significant harm
  - General observation: currently accepted SE principles slow things down in order to speed things up!
  - These techniques work!
    - But programmers (generally) don't like them

Spring 2006

CS 350/ODU

16

---

---

---

---

---

---

---

---

## One definition of software engineering:

- Computer science with economics
  - What's the cheapest way to build a quality system?
- One definition of an engineer:
  - Someone who can for dime what any fool can build for a dollar.

Spring 2006

CS 350/ODU

17

---

---

---

---

---

---

---

---

## Quick Survey (Quiz 1!) Due Tues.!!

- What's best prog. language?
- What language do you prefer to use?
- What's the best OS?
- What's the best debugger!
- Your largest program:
  - a) < 500 loc
  - b) > 500 & < 1 kloc
  - c) > 1 & < 10 kloc
  - d) > 10 & < 100 kloc
  - e) > 100 kloc
- What's a kloc?
  - a) comments inc.?
  - b) declarations?
  - c) only exec. stmts?
  - d) number of CRs?

Mail to [cmo@cs.odu.edu](mailto:cmo@cs.odu.edu)  
by Tues. Jan. 17

Spring 2006

CS 350/ODU

18

---

---

---

---

---

---

---

---

## SE emphasis on metrics & data

- It's not engineering if you can't measure and predict.
- You can't predict if without data!

Spring 2006

CS 350/ODU

19

---

---

---

---

---

---

---

---

## PSP composed of **this** week's "best-known practices"

- May be different next year; likely very different in 10 years.
- Approaches similar to PSP are widely advocated, and often used in industry.
- Past perception: most software organizations use poor practices resulting in overly expensive, late, and unreliable software.
  - This must change otherwise more software jobs will move to India!
- Now required of DoD software contractors.

Spring 2006

CS 350/ODU

20

---

---

---

---

---

---

---

---

## Unpleasant Facts of life:

- Some PSP aspects I don't like. Some I don't believe.
- You may not either, but after this course, you should be knowledgeable.
  - You will have been exposed in detail to one highly regarded software process; there are many others.
- Future of software development?
  - India has bright, well-trained software developers, that earn \$15k rather than \$80k per year
- PSP is all about costs, predictability, and quality!
- In PSP:
  - If you can find a better way to reduce costs, improve quality and improve predictability – and you have data **shows** it works, you should use it.

Spring 2006

CS 350/ODU

21

---

---

---

---

---

---

---

---

## Process, Process, Process!

- Current industry belief: the process used to develop software has significant impact on quality and costs.
- Things like programming language or design notation mostly don't
  - C++, C#, Java, PHP, UML will be replaced with something else
    - Maybe even better but we're in a field too often driven by fads rather than facts!
- So get the **process** right!
  - **Warning:** process emphasis may be this week's fad; I'm not sure. You may be able to tell in 5 years

Spring 2006

CS 350/ODU

22

---

---

---

---

---

---

---

---

## The PSP Metaprocess

- A process consists of a defined sequence of a steps.
- For PSP, each step consists of:
  - A set of **entry criteria** - step cannot start until all entry conditions are satisfied
  - A sequence of carefully defined **activities** - this is the work to be done in this step
  - A set of **exit criteria** - you're not done with the step until these are satisfied

Spring 2006

CS 350/ODU

23

---

---

---

---

---

---

---

---

## Lecture Topics

- The need for change
- PSP<sup>SM</sup> and TSP<sup>SM</sup> principles and objectives
- What is the TSP?
- The need for management support
- What is the PSP and how does it help?
- Course results

Spring 2006

CS 350/ODU

24

---

---

---

---

---

---

---

---

## The Changing World of Software

- Software now controls most business, government, and military systems.
- Factories are managed by software.
- Most advanced products are controlled by software.
- Finance, administrative, and business operations are largely run by software.
- Typical new car has ~16 CPUs.
- The cost, schedule, and quality of software is now a critical business concern.

Spring 2006

CS 350/ODU

25

---

---

---

---

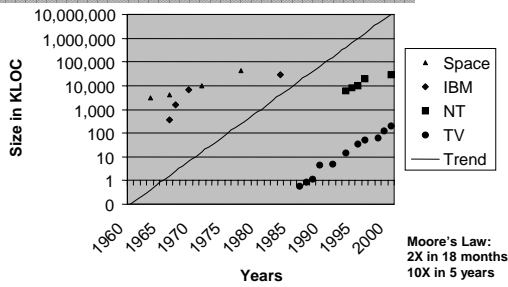
---

---

---

---

## Software Products are Bigger



Spring 2006

CS 350/ODU

26

---

---

---

---

---

---

---

---

## Big Software Projects Usually Fail

- With increased size, projects are more troubled.

Project Size	People	Time (Months)	Success Rate
Less than \$750K	6	6	55%
\$750K to \$1.5M	12	9	33%
\$1.5M to \$3M	25	12	25%
\$3M to \$6M	40	18	15%
\$6M to \$10M	+250	+24	8%
Over \$10M	+500	+36	0%

- This is a problem of scale: current software practices do not scale up.

Standish: Chaos Reports, 1999

Spring 2006

CS 350/ODU

27

---

---

---

---

---

---

---

---

## Why Projects Fail - 1

- Large and small software projects fail for four reasons.
- Project commitments are often unrealistic.
- The larger the project, the less influence we have.
- If we don't have anything to say, nobody will listen.
- Larger projects are harder to control.
- Today, few developers have personal plans.
- Without a plan, you cannot know job status.
- If you don't know where you are, management can't understand job status.
- If management doesn't understand job status, they can't manage projects.

Spring 2006

CS 350/ODU

28

---

---

---

---

---

---

---

---

## Why Projects Fail - 2

- Quality problems get worse with project size.
- In software systems, if any part has quality problems, the system will have quality problems.
- If the developers do not manage quality, their teams cannot manage quality.
- When unmanaged, quality will always be poor.
- To be effective, teams need leadership and coaching.
- Leaders build team motivation and commitment.
- Coaching develops team cohesion.
- Cohesive, motivated, and committed teams do the best work.

Spring 2006

CS 350/ODU

29

---

---

---

---

---

---

---

---

## The Need for Change

- Many lives and businesses now depend on software.
- We now need larger, more complex, and safer software systems on predictable schedules.
- Without different software practices, this will not happen.
- The Team Software Process (TSP) addresses this need.
- The PSP provides the knowledge and skill that developers need to work on TSP teams.

Spring 2006

CS 350/ODU

30

---

---

---

---

---

---

---

---

## Management Support - 1

- An initial TSP objective is to convince management to let your team be self directed.
- A self-directed team
  - sets its own goals
  - establishes its own roles
  - decides on its own development strategy
  - defines its own processes
  - develops its own plans
  - measures, manages, and controls its own work
- Self-directed teams do the best work.

Spring 2006

CS 350/ODU

31

---

---

---

---

---

---

---

---

## Management Support - 2

- Management will support you as long as you
  - strive to meet their needs
  - provide regular reports on your work
  - convince them that your plans are sound
  - do quality work
  - respond to changing needs
  - come to them for help when you have problems

Spring 2006

CS 350/ODU

32

---

---

---

---

---

---

---

---

## Management Support - 3

- Self-directed teams are a bargain.
- Management will agree to your managing your own work as long as they believe that you are doing a superior job.
- To convince them of this, you must
  - maintain precise and accurate plans
  - measure and track your work
  - regularly show management that you are doing superior work
- The PSP shows you how to do this.

Spring 2006

CS 350/ODU

33

---

---

---

---

---

---

---

---

## PSP Principles - 1

- The quality of a software system is determined by the quality of its worst components.
- The quality of a software component is governed by the individuals who developed it.
- The quality of a software component is governed by the quality of the process used to develop it.
- The key to quality is the individual developer's skill, commitment, and personal process discipline.

Spring 2006

CS 350/ODU

34

---

---

---

---

---

---

---

---

## PSP Principles - 2

- As a software professional, you are responsible for your personal process.
- You should measure, track, and analyze your work.
- You should learn from your performance variations
- You should incorporate lessons learned into your personal practices.

Spring 2006

CS 350/ODU

35

---

---

---

---

---

---

---

---

## What Does a PSP Provide?

- A stable, mature PSP allows you to
  - estimate and plan your work
  - meet your commitments
  - resist unreasonable commitment pressures
- You will also
  - understand your current performance
  - be better equipped to improve your capability

Spring 2006

CS 350/ODU

36

---

---

---

---

---

---

---

---

## What Does the PSP Provide?

- The PSP provides
  - a proven basis for developing and using an industrial-strength personal process
  - a discipline that shows you how to improve your personal process
  - the data to continually improve the productivity, quality, and predictability of your work

Spring 2006

CS 350/ODU

37

---

---

---

---

---

---

---

---

## Warning 1: PSP omits keys steps in software development. e.g.

- I think the most fun is deciding what software should do.
- Requires determining:
  - What's the real problem?
    - Is it a problem susceptible to a computerized solution?
  - What's feasible?
    - Technically?
    - Economically?
  - Of feasible solutions, does this one seem the best?

Spring 2006

CS 350/ODU

38

---

---

---

---

---

---

---

---

## Warning 2. PSP does not discuss support of software after deployment.

- 80% of software costs is maintenance

Spring 2006

CS 350/ODU

39

---

---

---

---

---

---

---

---

## What is the PSP?

- The PSP is a personal process for developing software or for doing any other defined activity.
  - defined steps
  - forms
  - standards
- It provides a measurement and analysis framework for characterizing and managing your personal work.
- It is also a defined procedure that helps you to improve your personal performance.

Spring 2006

CS 350/ODU

40

---

---

---

---

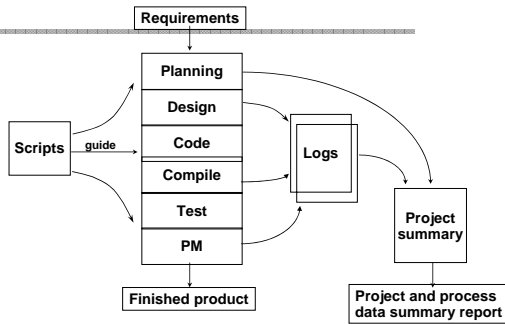
---

---

---

---

## The PSP Process Flow



Spring 2006

CS 350/ODU

41

---

---

---

---

---

---

---

---

## The Personal Software Process

- The PSP process is designed for individual use.
- It is based on scaled-down industrial software practice.
- The PSP course demonstrates the value of using a defined and measured process.
- It helps you and your organization meet the increasing demands for high quality and timely software.

Spring 2006

CS 350/ODU

42

---

---

---

---

---

---

---

---

## Learning the PSP - 1

- The PSP is introduced in several upward-compatible steps.
- You write one or more module-sized programs at each step.
- You gather and analyze data on your work.
- You use the results to improve your personal performance.

Spring 2006

CS 350/ODU

43

---

---

---

---

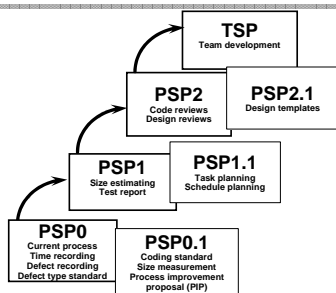
---

---

---

---

## Learning the PSP - 2



Spring 2006

CS 350/ODU

44

---

---

---

---

---

---

---

---

## Learning the PSP - 3

- PSP0: You establish a measured performance baseline.
- PSP1: You make size, resource, and schedule plans.
- PSP2: You practice defect and yield management.

Spring 2006

CS 350/ODU

45

---

---

---

---

---

---

---

---

## At Course Conclusion

- You can have practiced the key elements of an industrial-strength software process.
- You can understand which methods are most effective for you.
- You can do better work.
- You can have long-term improvement goals.

Spring 2006

CS 350/ODU

46

---

---

---

---

---

---

---

---

## Course Results

- SEI now has data on over 30,000 programs written using the PSP.
- The following charts show how others have improved during the PSP course.
  - size and effort estimating
  - compile and test time
  - productivity

Spring 2006

CS 350/ODU

47

---

---

---

---

---

---

---

---

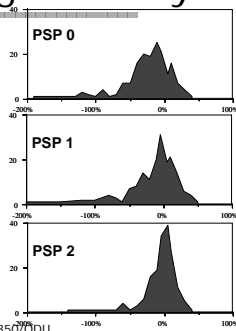
## PSP Estimating Accuracy

- Majority are underestimating
- With PSP, balance of over- and underestimates
- Much tighter balance around zero

Spring 2006

CS 350/ODU

48



---

---

---

---

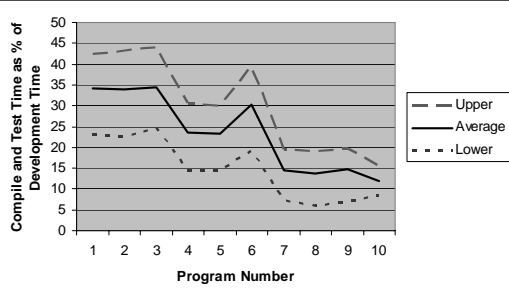
---

---

---

---

## Compile and Test Time – 810 Engineers



Spring 2006 CS 350/ODU 49

---

---

---

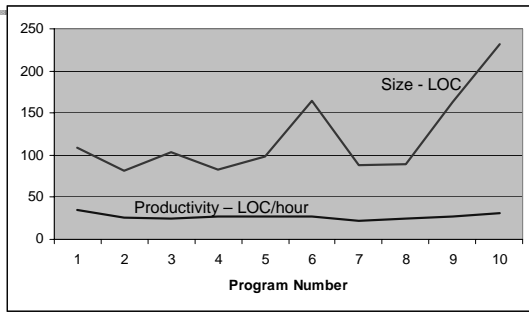
---

---

---

---

## Size and LOC/hour – 810 Engineers



Spring 2006 CS 350/ODU 50

---

---

---

---

---

---

---

## Messages to Remember

- The PSP is a defined process that can help you do better work.
- Once you have completed the course, you will know how to apply the PSP to your personal needs.
- You will have the knowledge and skill to be on a TSP team.
- With PSP0, the objective is to gather accurate and complete data on your work.

Spring 2006 CS 350/ODU 51

---

---

---

---

---

---

---

## Messages to Remember

- In using PSP0, your principal objective is to learn to gather and report accurate and complete data on your work.
- Once you have completed this course, you will know how to adjust and extend the PSP to meet your future needs.
- Until then, make your best effort to follow the PSP process scripts and instructions.

---

---

---

---

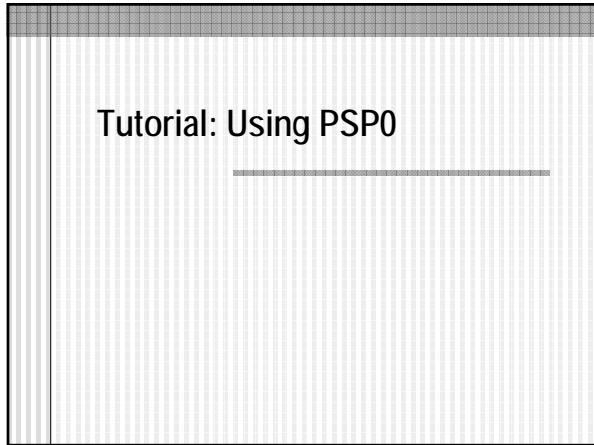
---

---

---

---

## Tutorial: Using PSP0



---

---

---

---

---

---

---

---

## Tutorial Objectives

- After this tutorial, you will
  - understand the PSP0 process
  - know how to use PSP0 process scripts and forms
  - be prepared to use PSP0 for program 1

---

---

---

---

---

---

---

---

## PSP0 Process

- PSP0 is a simple, defined, personal process.
  - Make a plan.
  - Use your current design and development methods to produce a small program.
  - Gather time and defect data on your work.
  - Prepare a summary report.

Spring 2006

CS 350/ODU

55

---

---

---

---

---

---

---

---

## PSP0 Objective

- The objective for PSP0 is to
  - demonstrate the use of a defined process in writing small programs
  - incorporate basic measurements in the software development process
  - require minimal changes to your personal practices

Spring 2006

CS 350/ODU

56

---

---

---

---

---

---

---

---

## PSP0 Process Phases - 1

- PSP0 has six phases.
- Planning – produces a plan for developing the program defined by the requirements.
- Design – produces a design specification for the program defined by the requirements.
- Coding – transforms the design specification into programming language statements.

Plan

Design

Code

Spring 2006

CS 350/ODU

57

---

---

---

---

---

---

---

---

## PSP0 Process Phases - 2

- Compile – translates the programming language statements into executable code.
- Test – verifies that the executable code satisfies the requirements.
- Postmortem – summarizes and analyzes the project data.

Compile

Test

Postmortem

Spring 2006

CS 350/ODU

58

---

---

---

---

---

---

---

---

## Phase Order

- Phase order is determined by the dependencies between phases.
  - You can't test the code before it's compiled.
  - You can't compile the code before it's written.
  - You can't use the design if it's produced after the code is written.
  - There's no reason to make a plan after you're done.
- You should start here →

Test

Compile

Code

Design

Plan

Spring 2006

CS 350/ODU

59

---

---

---

---

---

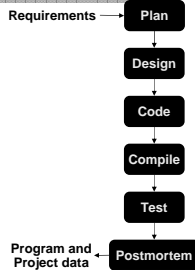
---

---

---

## Process Flow

- For programs that are small or well understood, execute the phases in order.
- A plan is produced.
- All modules are designed.
- All modules are then coded.
- The coded program is compiled and tested.
- The project data are summarized during the postmortem.



Spring 2006

CS 350/ODU

60

---

---

---

---

---

---

---

---



## The PSP0 Scripts - 1

- Planning: Use your previous experience to estimate:
  - the development time
  - the program size (in LOC)
- Development: Develop the product using your current methods.
- Postmortem: Complete the project plan summary with the time spent and defects found and injected in each phase.

Spring 2006

CS 350/ODU

64

---

---

---

---

---

---

---

---

## The PSP0 Scripts - 2

- Design: Design the program using your current design methods.
- Coding: Implement the program.
- Compile: Compile until defect-free.
- Test: Test the program and fix all defects.
- Record defects in the defect log and time per phase in the time log.

Spring 2006

CS 350/ODU

65

---

---

---

---

---

---

---

---

## Using Process Scripts

- Process scripts guide you through the process.
- You should
  - check the entry criteria before starting a phase
  - record the phase start time
  - perform the phase steps and instructions
  - record defects as they are found and corrected
  - check the exit criteria before ending a phase
  - record the phase end time
  - go to the next phase
- Force yourself to use this paradigm until it becomes a habit.

Spring 2006

CS 350/ODU

66

---

---

---

---

---

---

---

---

## PSP0 Measures and Forms

- PSP0 measures
  - Time – track time in phase
  - Defects – record defects as they are found and fixed
- PSP0 has four forms
  - PSP0 Project Plan Summary – summarizes planned and actual time and defects by phase
  - PSP0 Time Recording Log – used to record time
  - PSP0 Defect Recording Log – used to record defects
  - PSP0 Defect Type Standard – used to define standard defect types

Spring 2006

CS 350/ODU

67

---

---

---

---

---

---

---

---

## PSP Student Workbook

- The PSP Student Workbook provides support for the PSP.
  - scripts
  - forms
  - measures
  - calculations
  - planning
  - tracking
  - quality management
  - analysis
  - historical data
  - access to class materials
- It also provides support for post-course use of the PSP.

Spring 2006

CS 350/ODU

68

---

---

---

---

---

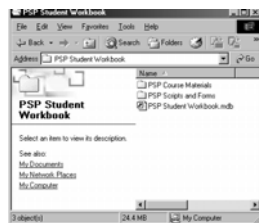
---

---

---

## Installing the PSP Student Workbook

- Create a folder to hold the contents from the class web site.
- Copy the contents of the web site to this folder.
- Contents
  - PSP Course Materials
  - PSP Scripts and Forms
  - PSP Student Workbook



Spring 2006

CS 350/ODU

69

---

---

---

---

---

---

---

---

## Open the PSP Student Workbook

- Open the file PSP Student Workbook.
- The welcome form will open followed by the student profile.



Spring 2006 CS 350/ODU 70

---

---

---

---

---

---

---

---

---

---

---

---

## Complete the Student Profile

- Enter the following
  - name
  - initials
  - date
  - name of your organization or company, if any
- Answer the questions under each tab.
  - employment status
  - software experience
  - programming experience
  - educational background
- Click Finish.



Spring 2006 CS 350/ODU 71

---

---

---

---

---

---

---

---

---

---

---

---

## Opening a PSP Project

- Select the first project, Assignment 1.
- Click Open Project.



Spring 2006 CS 350/ODU 72

---

---

---

---

---

---

---

---

---

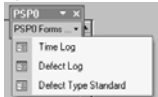
---

---

---

## PSP0 Forms - 1

- This is the PSP0 Project Plan Summary.
- To open the other PSP0 forms click PSP0 Forms... on the PSP0 menu.



- Select the form to open
  - Time Log
  - Defect Log
  - Defect Type Standard

Project Plan Summary

Phase	Plan	Actual	% Comp.	% Done
All	0	0	0%	0%
Analysis	0	0	0%	0%
Design	0	0	0%	0%
Code	0	0	0%	0%

CS 350/ODU

73

---

---

---

---

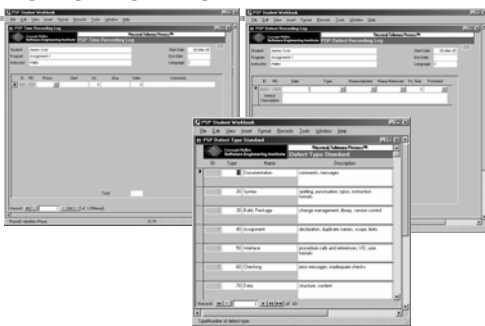
---

---

---

---

## PSP0 Forms - 2



Spring 2006

CS 350/ODU

74

---

---

---

---

---

---

---

---

## PSP0 Time Recording Log - 1

- Phase: Select the phase on which you were working.
- Start: Enter the date and time you started working. Double click to enter the current date and time.
- Int.: Enter any interruption time in minutes.

Student: James Over     Start Date: 05-Mar-05  
Program: Assignment 1     End Date: 05-Mar-05  
Instructor: Vlastis     Language: C

ID	PD	Phase	Start	Int.	Stop	Delta	Comments
				0			

Spring 2006

CS 350/ODU

75

---

---

---

---

---

---

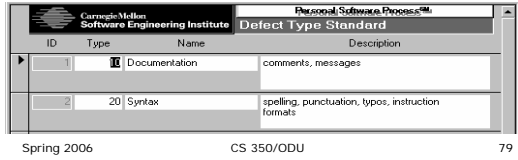
---

---



# Defect Type Standard

- The defect type standard provides a general set of defect categories.
- While you may add items or replace this standard with your own, it is generally wise to stick with these simple definitions until you have data to guide your changes.



---

---

---

---

---

---

---

---

# PSP0 Project Plan Summary

- Enter your best estimate of the total time that the development will take.  
Total
- The remaining items are calculated automatically.
- Time in phase
  - Actual time
  - To Date time
  - To Date % time
- Defects injected and removed in phase
  - Actual defects
  - To Date defects
  - To Date % defects



---

---

---

---

---

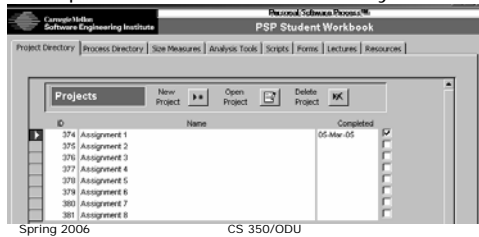
---

---

---

# Completing a PSP Project

- Select the project, e.g. Assignment 1.
- Enter a date in the completed fields or click the completed checkbox to enter today's date.



---

---

---

---

---

---

---

---

## Measurement Hints

- Gather and record data on your process as you work, not afterwards. If you forget, promptly make your best guess.
- Be precise and accurate.
  - time in minutes
  - count every defect
- You will be using your own data to manage your process; gather data that is worthy of your trust.

Spring 2006

CS 350/ODU

82

---

---

---

---

---

---

---

---

## Defect Fix Time

- Defect fix time is often misunderstood.
- It is the time taken both to find and fix the defect.
- Example
  - 8:05 run compiler on p1a.c, "line 23 - type mismatch"
  - 8:06 run editor on p1a.c
  - 8:15 change declaration on line 6 from integer to real
  - 8:16 run compiler on p1a.c, "no errors"
- Q: What is the defect fix time?
- A: 10 minutes

Spring 2006

CS 350/ODU

83

---

---

---

---

---

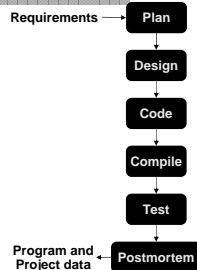
---

---

---

## Defect Phase Injected

- The injected phase for a defect depends on the phase the program is in.
- Example:
  - Tom finds a major logic error in his program during test. He has to redesign and code part of his program.
- Q: What phase is Tom's program in?
- A: Test
- Tom finds a defect in the new code he has written.
- Q: In what PSP phase was the defect injected?
- A: Test



Spring 2006

CS 350/ODU

84

---

---

---

---

---

---

---

---

# Measurement in the Cyclic Process

- Considerations

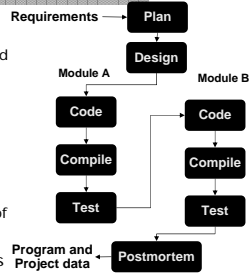
- Include a program part identifier in the notes field on time log entries.
- Add a similar annotation to defect log entries.
- Use "Test" as the phase removed when defects are found in a previously tested part.

- Example

- Tom finds and fixes an interface error in part A of his program while coding part B.

- Q: In what PSP phase was this defect removed?

- A: Test



Spring 2006

CS 350/ODU

85

---

---

---

---

---

---

---

---