Personal Software Process

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

- The cost, schedule, and quality of software is now a critical business concern.

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

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

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

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.

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 *individual* 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

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

What Does 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

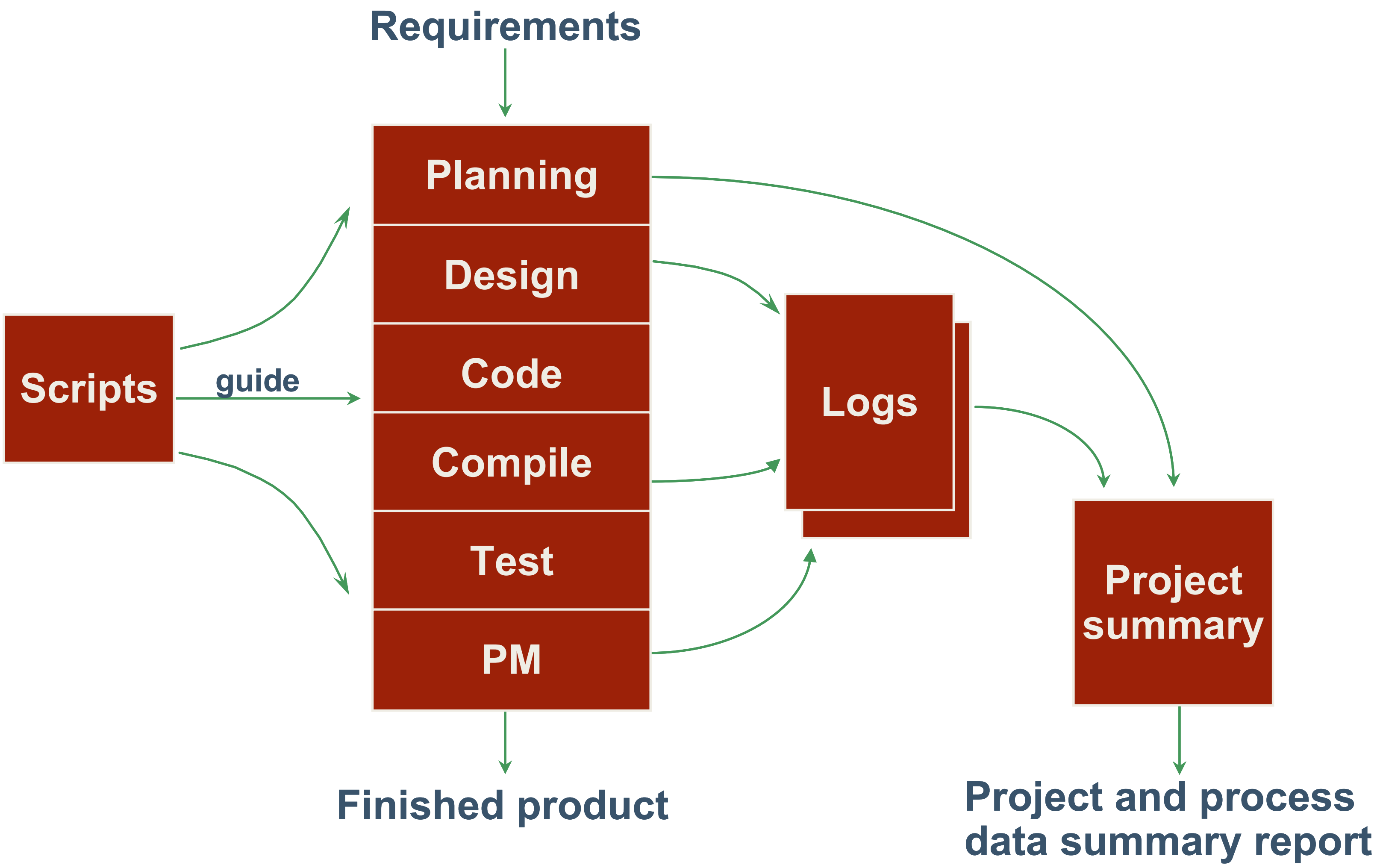
What Does PSP Provide?

- 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

What is the PSP?

- The PSP is a personal process for developing software or for doing any other defined activity. The PSP includes
  - 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.

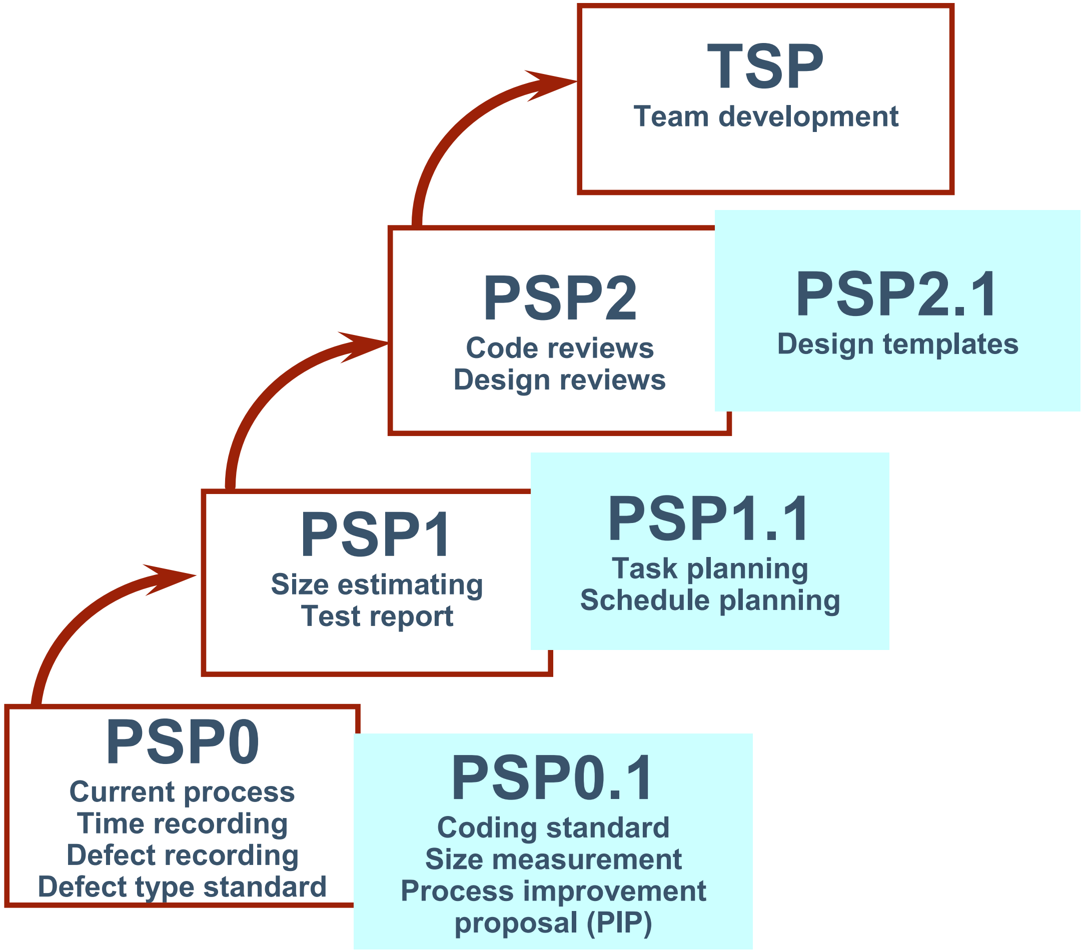
The PSP Process Flow



Learning the PSP - 1

- The PSP is introduced in six 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

Learning the PSP - 2



TSP
Team development

PSP2
Code reviews
Design reviews

PSP2.1
Design templates

PSP1
Size estimating
Test report

PSP1.1
Task planning
Schedule planning

PSP0
Current process
Time recording
Defect recording
Defect type standard

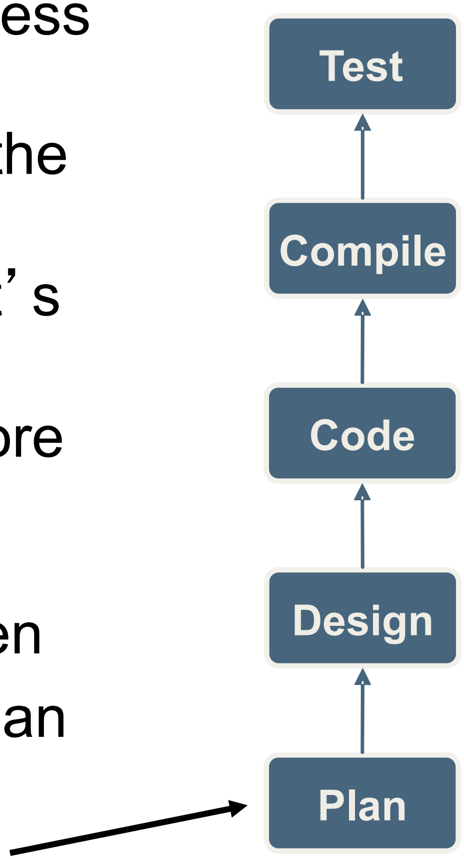
PSP0.1
Coding standard
Size measurement
Process improvement proposal (PIP)

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.

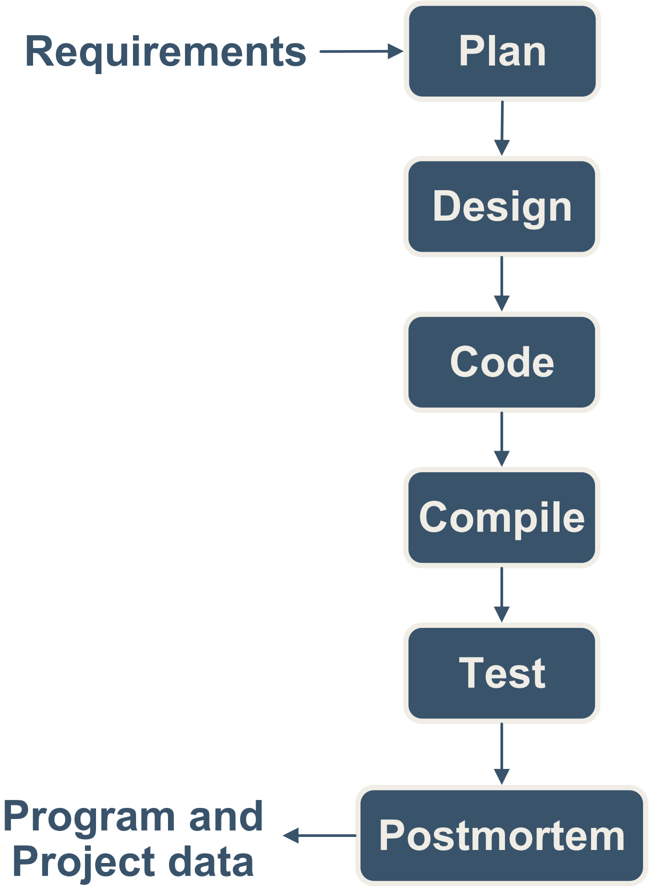
Phase Order

- The PSP looks like a waterfall process but it’s not
- The 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
- Conclusion…start here with a plan



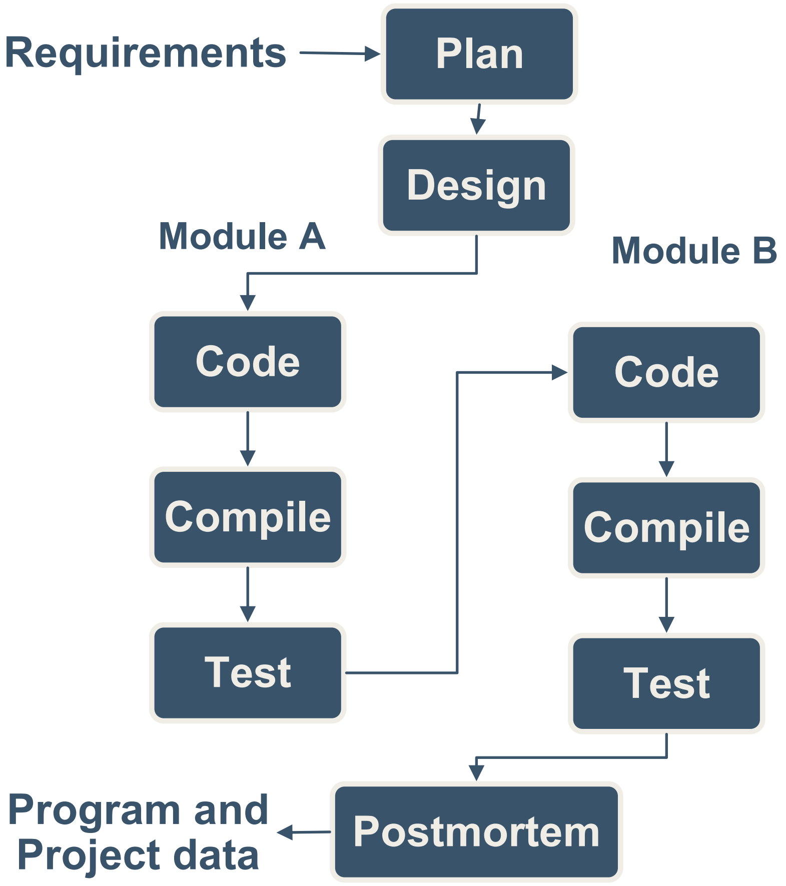
Process Flow

- When programs are small or well understood, you can execute the phases in order
- Produce a plan
- Design all modules
- Code all modules
- Compile the coded program
- Summarize the project data during the postmortem



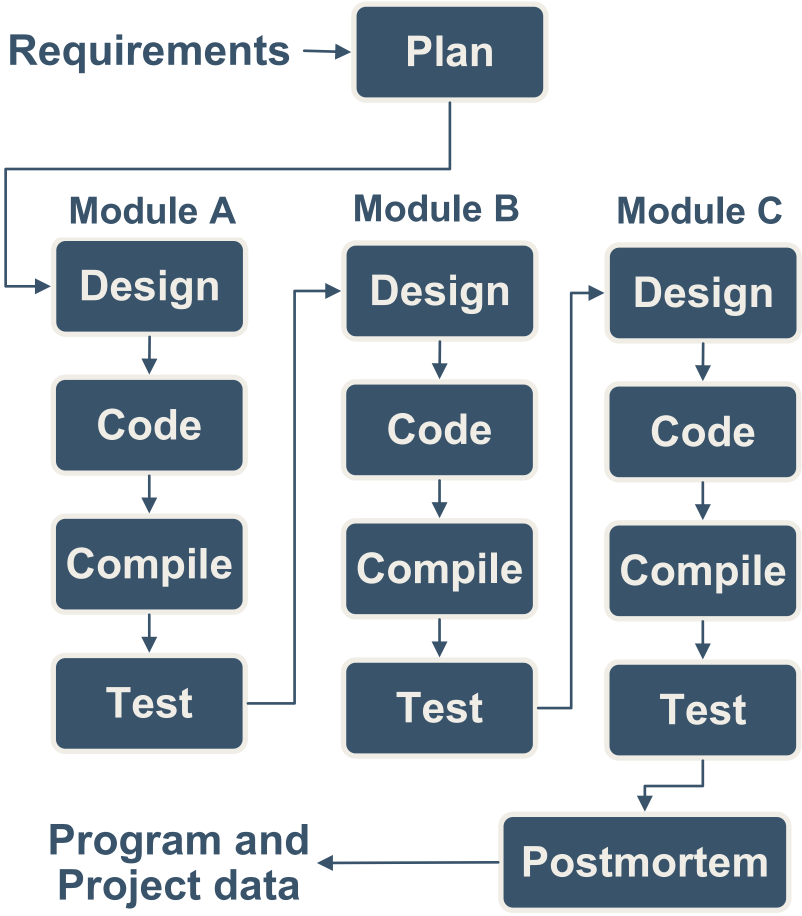
Cyclic Process Flow -1

- Large programs or those that are not well understood may require an iterative approach
- In this example the design is completed in one step
- Two modules are identified during the design, modules A and B
- Then each module is separately coded, compiled, and tested
- This example uses the PSP0 phases and two cycles of code-compile-test



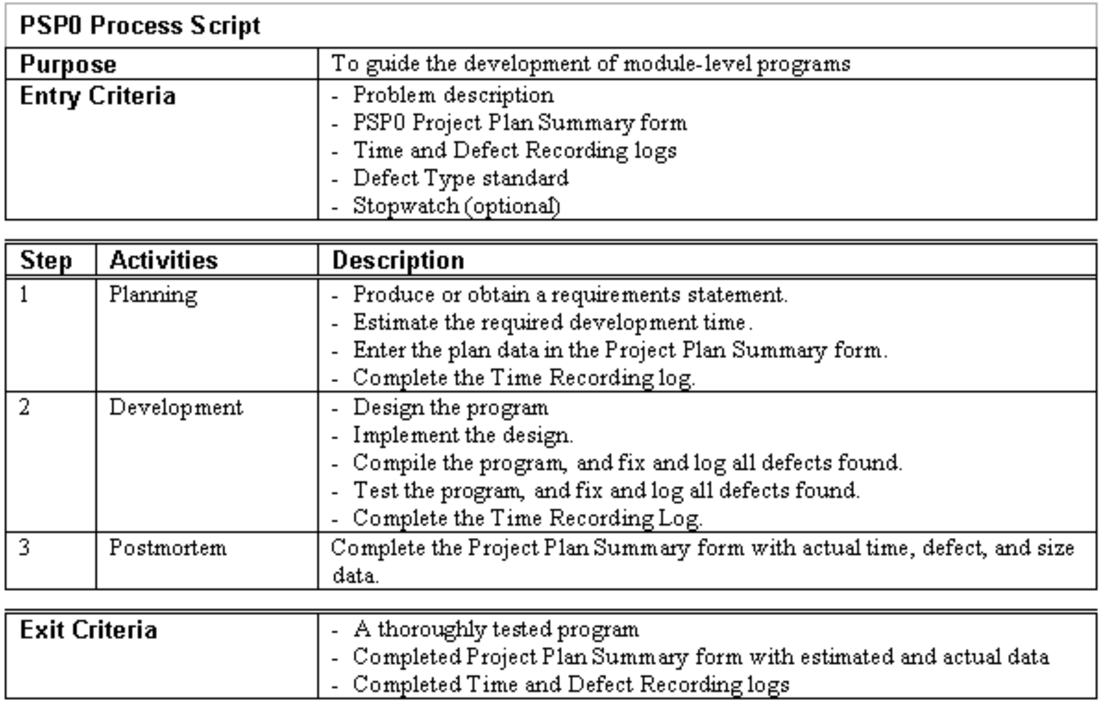
Cyclic Process Flow -2

- There can be more than two cycles and cycles can also include the design phase as in this example.
- Note that each cycle is focused on producing part of the program, e.g. Module A, Module B, Module C.
- Part size is a key factor for determining cycles.
  - a line of code is too small
  - a program may be too large
- One or more classes, methods, procedures, or functions are probably the right size.
- You need to determine what works for you.



Process Scripts

- Process scripts provide “expert-level” guidance on how to use the process
  - They are one or two pages long.
  - They describe the
    - Purpose
    - Entry criteria
    - General guidelines
    - Steps
    - Exit criteria



PSP0 Process Script

Purpose	To guide the development of module-level programs
Entry Criteria	- Problem description - PSP0 Project Plan Summary form - Time and Defect Recording logs - Defect Type standard - Stopwatch (optional)

Step	**Activities**	**Description**
1	Planning	- Produce or obtain a requirements statement. - Estimate the required development time. - Enter the plan data in the Project Plan Summary form. - Complete the Time Recording log.
2	Development	- Design the program - Implement the design. - Compile the program, and fix and log all defects found. - Test the program, and fix and log all defects found. - Complete the Time Recording Log.
3	Postmortem	Complete the Project Plan Summary form with actual time, defect, and size data.

Exit Criteria	- A thoroughly tested program - Completed Project Plan Summary form with estimated and actual data - Completed Time and Defect Recording logs

The PSP Scripts - 1

- Planning: Estimate the development time
- 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.

The PSP 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

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.

PSP Measures and Forms

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