МЕТАДАННЫЕ ДЛЯ ПОИСКА В RAG-СИСТЕМЕ

Файл: PSP-Review-Script-Checklist.pdf

Тип: руководство

Темы: Personal Software Process for Engineers: Part I, Design and Code Review Checklists, Assignment Kit for Design and Co

Уровень: средний

Ключевые слова для поиска:

- Software Engineering Institute
- SEI
- Carnegie Mellon University
- Design review
- Code review

Краткое описание:

Руководство по личному процессу программирования для инженеров, включающее чеклисты для обзора дизайна и кода.

Assignment Kit for Design and Code Review Checklists

Personal Software Process for Engineers: Part I

Personal Software Process for Engineers: Part I

Assignment Kit for Design and Code Review Checklists

Overview

Overview

This assignment kit covers the following topics.

Prerequisites

Reading
- Chapter 9

Design review and code review checklists requirements

Design and code review checklists requirements

Produce, document, and submit design review and code review checklists.

Also submit your defect logs for programs 1-4.

The checklists should be tailored for the
- design notation and programming language that you use
- types of defects that you inject

Using review checklists

The goal of PSP reviews

The goal of PSP reviews is consistently high yield.

- Review yield is the percentage of defects in the product at review time found by the review.
- Process yield is the percentage of defects found before the first compile.

With practice, experienced engineers can achieve consistently high process and review yields of about 70% to 80%.

How checklists are used

Checklists are used to

- guide the review process
- establish a consistent set of criteria for judging that the design or code is correct and complete
- promote early defect removal
- tailor the review process to support personal defect-injection issues

Conducting effective reviews

To conduct effective PSP personal reviews

- follow the process - always
- use a personal checklist that is designed to find the defects that you make
- devise a review strategy and use it
- review one product component at a time
- check for one topic at a time
- treat each check as a personal certification that the product is free of this defect

PSP code review script

The code review process

A PSP code review process is described in the following script.

Your checklists should be designed to support your review process.

Code Review Script

Purpose	To guide you in reviewing programs
Entry Criteria	- A completed and reviewed program design - Source program listing - Code Review checklist - Coding standard - Defect Type standard - Time and Defect Recording logs
General	Do the code review with a source-code listing; do not review on the screen!

Step	**Activities**	**Description**
1	Review	- Follow the Code Review checklist. - Review the entire program for each checklist category; do not try to review for more than one category at a time! - Check off each item as it is completed. - For multiple procedures or programs, complete a separate checklist for each.
2	Correct	- Correct all defects. - If the correction cannot be completed, abort the review and return to the prior process phase. - To facilitate defect analysis, record all of the data specified in the Defect Recording log instructions for every defect.
3	Check	- Check each defect fix for correctness. - Re-review all design changes. - Record any fix defects as new defects and, where you know the number of the defect with the incorrect fix, enter it in the fix defect space.

Exit Criteria	- A fully reviewed source program - One or more Code Review checklists for every program reviewed - All identified defects fixed - Completed Time and Defect Recording logs

PSP design review script

The design review process

A PSP design review process is described in the following script.

Your checklists should be designed to support your review process.

PSP2 Design Review Script

Purpose	To guide you in reviewing detailed designs
Entry Criteria	- Completed program design - Design Review checklist - Design standard - Defect Type standard - Time and Defect Recording logs
General	Where the design was previously verified, check that the analyses - covered all of the design - were updated for all design changes - are correct - are clear and complete

Step	**Activities**	**Description**
1	Preparation	Examine the program and checklist and decide on a review strategy.
2	Review	- Follow the Design Review checklist. - Review the entire program for each checklist category; do not try to review for more than one category at a time! - Check off each item as you complete it. - Complete a separate checklist for each product or product segment reviewed.
3	Fix Check	- Check each defect fix for correctness. - Re-review all changes. - Record any fix defects as new defects and, where you know the defective defect number, enter it in the fix defect space.

Exit Criteria	- A fully reviewed detailed design - One or more Design Review checklists for every design reviewed - All identified defects fixed and all fixes checked - Completed Time and Defect Recording logs

Building review checklists

Overview

When constructing review checklists, your objective is to identify and devise checks for the specific defect types that you make.

To build and improve your review checklists, you will need
- a documented review process
- detailed defect data on the errors that you make
- data on the review process

These data can be derived from PSP defect logs and project plan summaries.

The steps in constructing a review checklist are as follows:
1. Analyze defect data to identify opportunities and priorities.
2. Devise checks for the defects that you make.
3. Organize checks to support the review process.

Analyzing defect data

To ensure that your reviews provide the most benefit, your checklists should focus on defects that
- give you the most trouble
- provide the most leverage

A Pareto distribution will help you to establish priorities. Create distributions for
- defect frequency by category
- defect cost by category

Figure 9.8, page 179, shows an example Pareto distribution.

As you gather more data on defects, it may be necessary to create defect sub categories. An example sub-categorization is shown in Table 9.5, page 178.

The interim and final reports contain other defect analyses that are useful in building review checklists.

Defects injected and removed by phase	- In what phase do I inject the most defects? - In what phase are they removed? - Could they be removed earlier?
Defect types found by the compiler	- What defect types will the compiler find? - How effective is the compiler at finding defects? - What defect types does the compiler miss
Defect fix time analysis	- What does it cost to fix a defect? - What could be saved by removing defects earlier?
Defect removal rates for reviews	- How efficient are my review processes? - How do they compare?
Review yield	- How effective are my review processes?

Continued on next page

Devising checks

The next step in building checklists is to devise checks for the highest priority defect types.

Examine your defect logs to identify the specific defects that you make and devise checks to find them.

Checks should be devised *to specifically* address the defects that you make. For example, if you often forget to include semicolons at the end of a statement, then devise a checklist item for this defect.

All statements end with a semicolon	√

While too much detail can be overwhelming, reviews are generally more efficient when each checklist item has a small, easily tested scope that can be quickly and accurately checked.

Organizing checklist items

The final step in building checklists is to organize your checklist items into categories.

When selecting categories consider the following.
- Merge similar items into one category.
- Order categories to support your review strategy and process.

The example design review and code review checklists illustrate one possible organization that you might use.

The design review checklist organization is explained in the following table.

Category	**Includes checks to ensure that**
Complete	The design is complete with respect to requirements, etc.
Logic	The program logic is correct.
Special Cases	Special and extreme cases are properly handled.
Functional use	Functional aspects and interfaces of the design are correct.
Names	Names, variables, parameters, objects, and so on are properly used and scoped.
Standards	The design conforms to design standards.

Example Checklists

Design and code review checklists The following design and code review checklists can be used as a starting point for designing your own checklists.

PSP2 Design Review Checklist

Student		Date	
Program		Program #	
Instructor		Language	

Purpose	To guide you in conducting an effective design review
General	- Review the entire program for each checklist category; do not attempt to review for more than one category at a time! - As you complete each review step, check off that item in the box at the right. - Complete the checklist for one program or program unit before reviewing the next.

Complete	Verify that the design covers all of the applicable requirements. - All specified outputs are produced. - All needed inputs are furnished. - All required includes are stated.				
External Limits	Where the design assumes or relies upon external limits, determine if behavior is correct at nominal values, at limits, and beyond limits.				
Logic	- Verify that program sequencing is proper. Stacks, lists, and so on are in the proper order. Recursion unwinds properly. - Verify that all loops are properly initiated, incremented, and terminated. - Examine each conditional statement and verify all cases.				
Internal Limits	Where the design assumes or relies upon internal limits, determine if behavior is correct at nominal values, at limits, and beyond limits.				
Special Cases	- Check all special cases. - Ensure proper operation with empty, full, minimum, maximum, negative, and ero values for all variables. - Protect against out-of-limits, overflow, and underflow conditions. - Ensure "impossible" conditions are absolutely impossible. - Handle all possible incorrect or error conditions.				
Functional Use	- Verify that all functions, procedures, or methods are fully understood and properly used. - Verify that all externally referenced abstractions are precisely defined.				
System Considerations	- Verify that the program does not cause system limits to be exceeded. - Verify that all security-sensitive data are from trusted sources. - Verify that all safety conditions conform to the safety specifications.				
Names	Verify that - all special names are clear, defined, and authenticated - the scopes of all variables and parameters are self-evident or defined - all named items are used within their declared scopes				
Standards	Ensure that the design conforms to all applicable design standards.				

Continued on next page

Code Review Checklist

Student		Date	
Program		Program #	
Instructor		Language	C++

Purpose	To guide you in conducting an effective code review
General	- Review the entire program for each checklist category; do not attempt to review for more than one category at a time! - As you complete each review step, check off that item in the box at the right. - Complete the checklist for one program or program unit before reviewing the next.

Complete	Verify that the code covers all of the design.				
Includes	Verify that the includes are complete.				
Initialization	Check variable and parameter initialization. - at program initiation - at start of every loop - at class/function/procedure entry				
Calls	Check function call formats. - pointers - parameters - use of '&'				
Names	Check name spelling and use. - Is it consistent? - Is it within the declared scope? - Do all structures and classes use '.' reference?				
Strings	Check that all strings are - identified by pointers - terminated by NULL				
Pointers	Check that all - pointers are initialized NULL - pointers are deleted only after new - new pointers are always deleted after use				
Output Format	Check the output format. - Line stepping is proper. - Spacing is proper.				
() Pairs	Ensure that () are proper and matched.				
Logic Operators	- Verify the proper use of ==, =, ||, and so on. - Check every logic function for ().				
Line-by-line check	Check every line of code for - instruction syntax - proper punctuation				
Standards	Ensure that the code conforms to the coding standards.				
File Open and Close	Verify that all files are - properly declared - opened - closed				

Evaluation criteria

Evaluation criteria

Keep your checklists simple and short.

Your checklists must be
- complete
- legible
- tailored to the design method and notation that you use
- tailored to the programming language that you use
- designed to address the defects that you inject

Design Review Checklist Template

Student ______________________ Date __________
Program ______________________ Program # __________
Instructor ______________________ Language __________

Purpose	To guide you in conducting an effective design review
General	- Review the entire program for each checklist category; do not attempt to review for more than one category at a time! - As you complete each review step, check off that item in the box at the right. - Complete the checklist for one program or program unit before reviewing the next.

Code Review Checklist Template

Student	______________________	Date	__________
Program	______________________	Program #	__________
Instructor	______________________	Language	__________

Purpose	To guide you in conducting an effective code review
General	- Review the entire program for each checklist category; do not attempt to review for more than one category at a time! - As you complete each review step, check off that item in the box at the right. - Complete the checklist for one program or program unit before reviewing the next.
