ifsp – INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO

SABRINA FONTES CARVALHO

1462253

Personal Software Process for Engineers: Part I

PSP Interim Report

Trabalho apresentado ao Curso de Análise e Desenvolvimento de Sistemas, IFSP, como requisito parcial para a matéria de Metodologia de Análise de Sistemas Orientada a Objetos em 2016, sob orientação dos professores:

MSc Carlos Henrique Veríssimo Pereira

MSc Francisco Supino Marcondes e

MSc João Vianei Tamanini.

SÃO PAULO

2016

Sumário

LISTA DE ILUSTRAÇÕES 4

1 RELATÓRIO TÉCNICO-CIENTÍFICO 5

1.1 INTRODUÇÃO 5

1.2 DESENVOLVIMENTO 5

1.2.1 OBJETIVOS 5

1.2.1.1 GERAL 5

1.2.1.2 ESPECÍFICO 6

1.2.2 METODOLOGIA 6

1.2.2.1 PSP INTERIM REPORT 6

1.2.2.2 PROCEDIMENTOS EXPERIMENTAIS 7

1.2.2.2.1 REQUISITOS DO PSP INTERIM REPORT 7

1.2.2.2.2 PLANNING 8

1.2.2.2.3 DEVELOPMENT 8

1.2.2.2.3.1 ANÁLISE DA PRECISÃO DAS ESTIMATIVAS DE TAMANHO 9

1.2.2.2.3.2 ANÁLISE DA PRECISÃO DO TEMPO ESTIMADO E DA PRODUTIVIDADE 10

1.2.2.2.3.3 ANÁLISE DA DISTRIBUIÇÃO DE TEMPO 12

1.2.2.2.3.4 ANÁLISE DO TEMPO DE CORREÇÃO DOS DEFEITOS 14

1.2.2.2.3.5 ANÁLISE DOS DEFEITOS 14

1.2.2.2.4 POSTMORTEM 15

1.2.2.2.5 PLAN SUMMARY 15

1.2.2.2.6 TIME RECORDING LOG 16

1.2.2.3 RESULTADOS 16

1.2.2.3.1 PLANNING 16

1.2.2.3.2 DEVELOPMENT 16

1.2.2.3.3 POSTMORTEM 17

1.3 CONCLUSÕES 17

1.4 ELEMENTOS PÓS-TEXTUAIS 17

1.4.1 REFERÊNCIAS 17

1.4.2 APÊNDICES 18

1.4.2.1 PSP PROJECT PLAN SUMMARY – PROG1 18

1.4.2.2 TIME RECORDING LOG – PROG1 19

1.4.2.3 DEFECT RECORDING LOG – PROG1 19

1.4.2.4 PSP PROJECT PLAN SUMMARY – PROG2 20

1.4.2.5 TIME RECORDING LOG – PROG2 21

1.4.2.6 DEFECT RECORDING LOG – PROG 21

1.4.2.7 PSP PROJECT PLAN SUMMARY – PROG3 22

1.4.2.8 TIME RECORDING LOG – PROG3 23

1.4.2.9 DEFECT RECORDING LOG – PROG3 24

1.4.2.10 PSP PROJECT PLAN SUMMARY – PROG4 25

1.4.2.11 TIME RECORDING LOG – PROG4 26

1.4.2.12 DEFECT RECORDING LOG – PROG4 27

LISTA DE ILUSTRAÇÕES

FIGURA 1 - modelo de visão 4+1 da arquitetura 5

Quadro 1 - PSP Interim Report Process Script 7

Quadro 2 - tamanho em loc 9

Quadro 3 - tempo em minutos 10

Quadro 4 - produtividade 11

Quadro 5 - tempo médio fasto por fase 12

Quadro 6 - defeitos por fase 14

Quadro 7 - PSP Interim Report plan summary 15

Quadro 8 - PSP Interim Report TIME RECORDING LOG 16

FIGURA 2 - psp0 project plan summary 18

FIGURA 3 - psp TIME RECORDING LOG PROG1 19

FIGURA 4 - psp DEFECT RECORDING LOG PROG1 19

FIGURA 5 - psp0.1 project plan summary 20

FIGURA 6 - psp TIME RECORDING LOG PROG2 21

FIGURA 7 - psp DEFECT RECORDING LOG PROG2 21

FIGURA 8 - psp1 project plan summary – PARTE 1 22

FIGURA 9 - psp1 project plan summary – PARTE 1 23

FIGURA 10 - psp TIME RECORDING LOG PROG3 23

FIGURA 11 - psp DEFECT RECORDING LOG PROG3 24

FIGURA 12 - psp1.1 project plan summary – PARTE 1 25

FIGURA 13 - psp1.1 project plan summary – PARTE 2 26

FIGURA 14 - psp TIME RECORDING LOG PROG4 26

FIGURA 15 - psp DEFECT RECORDING LOG PROG4 27

# RELATÓRIO TÉCNICO-CIENTÍFICO

## INTRODUÇÃO

Este relatório tem o objetivo de apresentar resultados do dos programas um a quatro do *Personal Process for Engineers: Part I* utilizando a metodologia *Personal Software Process* (PSP), que é um processo de desenvolvimento de software que visa melhorar as estimativas de tempo e esforço, através de planejamento detalhado e acompanhamento de dados obtidos em programas anteriores.

Além dessa técnica, a arquitetura 4+1 está sendo utilizada para modelagem, conforme pode ser visto na Figura 1 e será desenvolvida na fase de *Design*.

1. modelo de visão 4+1 da arquitetura

Designn

Componente

Processo

Deploy

CDU

## DESENVOLVIMENTO

No decorrer deste relatório será desenvolvido o *Interim Report* utilizando o *Sample PSP Interim Report Process Script*, conforme a se encontra no ASGKIT Interim Report e Quadro 1 desse documento.

### OBJETIVOS

#### GERAL

O objetivo principal do PSP for Engineers I é mostrar o modo como é feito para medir, estimar e planejar com precisão um trabalho pessoal.

O objetivo geral desse relatório é estabelecer uma linha de base do processo de desenvolvimento de programas para que, no futuro, se possa medir o processo de melhoria pessoal.

Além disso, o *Interim Report* tem a finalidade de relatar através de demonstrações, os resultados obtidos durante todas as fases de planejamento, desenvolvimento e post-mortem dos programas anteriores, a fim de demonstrar como a metodologia PSP impacta de maneira positiva no desenvolvimento de projetos.

#### ESPECÍFICO

O objetivo específico desse relatório é avaliar o desenvolvimento dos programas anteriores, de um a quatro. Para isso, alguns pontos específicos serão tratados ao longo desse relatório, são eles:

* análise do tamanho, tempo gasto e defeitos produzidos;
* compreensão da precisão das estimativas e planejamento feitos;
* compreensão de onde o tempo de desenvolvimento foi gasto;
* compreensão de como os defeitos afetam o desenvolvimento;
* definição de um processo para uso pessoal com base nos dados coletados até o momento.

### METODOLOGIA

#### PSP INTERIM REPORT

Conforme pode ser observado na Quadro 1, o *PSP Interim Report*, possui questões que devem ser respondidas usando os dados coletados nos quatro programas anteriores.

Através desses dados, será realizada uma análise dos dados do PSP que serão reportados nesse relatório e que será feita em três fases: *Planning*, *Development* e *Postmortem*.

1. PSP Interim Report Process Script

|  |  |  |
| --- | --- | --- |
| **Fase #** | **Propósito** | **Guiar a análise e confecção do PSP *Interim Report*** |
|  | Critérios de Entrada | * Programas 1 a 4 completos e corrigidos pelo instrutor; * Cópia dos relatórios a serem preenchidos; * Os formulários *Time log* e *PSP Interim Report Summary.* |
| 1 | Planning | * Estimar o tamanho desse relatório:   + Número de parágrafos de análise;   + Número de tabelas e gráficos. * Estimar o esforço baseado no tamanho do relatório; * Reportar as estimativas *Plan Summary form;* * Reportar o tempo estimado no *Time log.* |
| 2 | Development | * Para cada questão da análise: * Gerar uma tabela ou gráfico; * Analisar a tabela ou gráfico e demais dados; * Escrever um parágrafo de análise. * Reportar o tempo gasto nessa fase no *Time log.* |
| 3 | Postmortem | * Medir o tamanho do relatório: * Número de tabelas e gráficos; * Número de parágrafos de análise. * Completar o *Plan Summary form;* * Reportar o tempo gasto nessa fase no *Time log.* |
|  | Critérios de Saída | * *PSP Interim Report* completo; * *Plan Summary* completo; * *Time log* completo. |

#### PROCEDIMENTOS EXPERIMENTAIS

##### REQUISITOS DO PSP INTERIM REPORT

Os requisitos encontrados para o *Interim Report* são:

* Desenvolver um processo para analizar os dados dos programas anteriores e, com eles, produzir um relatório;
* Incluir nesse processo as fases de planejamento e post-mortem;
* Usar o processo desenvolvido para:
  + Planejar o desenvolvimento do relatório;
  + Analisar os dados do processo;
  + Produzir o relatório;
  + Completar o Postmortem.
* Entregar o relatório, o processo e os dados analizados.

Através dos requisitos citados acima, fica mais claro o que se é esperado no começo, no desenvolvimento e na entrega desse.

##### PLANNING

Na fase de *Planning*, foi estimado o tempo gasto em cada objeto e em cada fase do *PSP Interim Report* e reportado no *Plan Summary* no Quadro 2 deste documento.

Através da análise feita, três objetos foram identificados, conforme solicita o PSP *Interim Report Process Script,* são eles os parágrafos de análise, tabelas e gráficos.

Nos ASGKIT *Interim Report,* é solicitado que seja feito um parágrafo de análise e pelo menos um gráfico ou tabela com os dados da análise, para cada pergunta a ser respondida e um parágrafo de conclusão, ao final do relatório. Por esse motivo, a estimativa foi feita contando que será realizado quinze parágrafos e catorze gráficos e tabelas.

Quanto ao esforço, foi estimado cinco minutos por parágrafo, 5 minutos por tabela e dois minutos por gráfico.

O tempo por parágrafo foi definido levando-se em conta que os dados para análise foram produzidos sempre levando em conta o programa anterior. Com isso, uma análise parcial já foi feita, não exigindo que muito esforço fosse gasto.

Para as tabelas, também foi levado em conta a facilidade de acesso aos dados e a necessidade de apenas compará-los.

Para os gráficos, foi pensado em gera-los a partir dos dados existentes na tabela, diminuindo o esforço para confeccioná-los.

##### DEVELOPMENT

Nessa fase, os dados dos programas anteriores serão analisados e utilizados para responder catorze questões a respeito das estimativas feitas ao longo do desenvolvimento.

###### ANÁLISE DA PRECISÃO DAS ESTIMATIVAS DE TAMANHO

1. tamanho em loc

**Pergunta 1:** Quais são os tamanhos reais de LOC médio, máximo e mínimo de seus programas até agora?

Como pode ser constatado no Quadro 2, com base nos dados de LOC (lines of code) dos programas de 1 a 4, pode-se constatar que o maior programa até o momento é o 4 e o menor é 1, esse resultado se deu por conta da complexidade de cada programa. Em média, foram desenvolvidas 87 linhas de código por programa.

Apesar do programa 4 ser o maior em linhas de código, é importante ressaltar que ele é o que mais possui reutilização de código.

**Pergunta 2:** Excluindo o programa 1, qual a percentagem acima ou abaixo do tamanho real foi o tamanho estimado para cada programa? Quais são os seus valores médio, máximo e mínimo para estes?

Para o programa 2, como foi a primeira estimativa, a porcentagem de erro é imensa, foi estimado 248% a mais do que foi realizada. Essa foi a maior taxa de erro na estimativa dos programas.

Para o programa 3, a porcentagem ficou 90% acima do que foi realizado. Esse resultado aconteceu, porque foi superestimado o esforço com base na complexidade do programa 3.

Para o programa 4, a porcentagem foi de 15% a mais do realizado. Essa foi a melhor estimativa com a menor margem de diferença entre o estimado e o realizado.

**Pergunta 3:** Excluindo o programa 1, qual é o total de erro acumulado para as estimativas de tamanho?

O total de erros acumalados é de 281 linhas de diferença entre o estimado e realizado, uma porcentagem de 80% a mais de tempo estimado do que foi realmente realizado. Isso ocorreu devido a falta de maturidade no processo de estimativas, assim como a falta de base para se calcular as estimativas.

Vale ressaltar que, com o passar do desenvolvimento dos programas, as estimativas foram fcando mais próximas do real.

###### ANÁLISE DA PRECISÃO DO TEMPO ESTIMADO E DA PRODUTIVIDADE

1. tempo em minutos

**Pergunta 1:** Quais são os tempos médios, máximos e mínimos das suas atribuições até hoje?

Baseado nos dados coletados e demonstrados no Quadro 4, o máximo de tempo gasto até o momento, foi no desenvolvimento do programa 2. Isso ocorreu, por causa do salto de complexidade que se deu entre o primeiro e o segundo programa. Apesar dos demais programas serem de maior complexidade, houve considerável reutilização de código, fazendo com que o esforço fosse empregado somente nos objetos novos que precisaram ser desenvolvidos.

O menor tempo gasto foi no empenhado no programa 1, por causa de sua simplicidade, e o tempo médio gasto no desenvolvimento dos programas foi de 310,75 minutos.

1. produtividade

**Pergunta 2:** Quais são os valores médio, máximo e mínimo de produtividade por programa até hoje em LOC/hora?

A produtividade maior ocrreu no Programa 4, mas isso ocorreu por conta das linhas de código que foram reaproveitadas dos outros programas.

O menor índice de produtividade ocorreu no programa 1, porque foi um. Isso ocorreu por causa da adaptação a linguagem, como fazia tempo que não se tinha contato com ela, foi necessário relembrar sintaxe e bilbliotecas.

Por isso, para uma análise mais efetiva, deve-se considerar a produtividade média como mais acertiva, que é de 16,44 LOC/hora.

**Pergunta 3:** Qual a porcentagem acima ou abaixo do tempo real foi o tempo estimado para cada programa? Quais são os seus valores médio, máximo e mínimo?

Todos os programas foram superestimados, mas assim como ocorreu com as linhas de código, a medida que os programas foram desenvolvidos, a estimativa melhorou, chegando perto do tempo real.

A maior discrepância foi encontrada no programa 2, que foi estimado 31,42% a mais do tempo real, a menor se deu no programa 4 em que a diferença foi de 23,85%.

**Pergunta 4:** Qual é o total de erro acumulado para as estimativas de tempo?

A diferença entre o tempo estimado e o real é de 300 minutos, ou seja, o estimado extrapolou 29% a mais do tempo real gasto.

###### ANÁLISE DA DISTRIBUIÇÃO DE TEMPO

1. tempo médio fasto por fase

**Pergunta 1:** Qual porcentagem do tempo foi gasta, em média, em cada fase do processo? Em *Design* e *Code* juntos? Em *Compile* e *Test* juntos?

Conforme pode ser observado no Quadro 5, a fase de *Design* e *Code* juntas são responsáveis por 68% do desenvolvimento dos programas. Já a fase de *Compile* e *Test,* são responsáveis por apenas 8% do processo.

Esse resultado ocorreu por causa da fase *Design*, quanto maior o tempo gasta no planejamento do programa, menor será o tempo gasto nas demais fases e de maior qualidade será o programa.

**Pergunta 2:** Se você gasta 1000 horas desenvolvendo programas semelhantes a esses, quanto tempo você gostaria de gastar na fase *Design*? E na fase *Code*? E em testes unitários? Quantas linhas de código você provavelmente escreverá nesse tempo? Quantos defeitos você provavelmente encontrará no teste unitário?

Em 1000 horas desenvolvendo, ao menos 500 horas serão destinadas a fase *Design.* Após os quatro programas, foi identificada a importância dessa fase e anecessidade de se executá-lo com a maior precisão possível, pois dela depende o sucesso de todo o projeto.

Na fase de *Code*, seriam empenhados 250 horas, ou seja, 25% do tempo, mas a efetividade dessa estimativa depende da qualidade fase *Design*. Os testes unitários seriam incluídos nos 25% gastos na fase de *Code*, uma vez que é entendido que esse tipo de teste faz parte da codificação do programa.

A partir das estimativas anteriores, são esperadas 400 linhas de códigos e cerca de 10 defeitos encontrados.

###### ANÁLISE DO TEMPO DE CORREÇÃO DOS DEFEITOS

1. defeitos por fase

**Pergunta 1:** Qual categoria teve o maior tempo médio de correção de defeitos?

A categoria com o maior tempo gasto em correção foi a fase *Code,* porque foi também a fase que apresentou o maior número de erros.

**Pergunta 2:** Qual categoria teve o maior tempo total de correção de defeitos?

A categoria que demandou o maior tempo total de correção também foi a fase de *Code*, pelo mesmo motivo anterior.

###### ANÁLISE DOS DEFEITOS

**Pergunta 1:** Quais são os valores médio, máximo e mínimo dos defeitos totais por programa até hoje?

O valor médio de defeitos é de 1,75 defeitos por programa, o menor número de defeitos encontrados ocorreu nos programas 1, 3 e 4, cada programa desses teve a ocorrência de apenas um erro, restando para o programa 2, a maioria dos defeitos, com um total de 4 defeitos.

**Pergunta 2:** Quais são os valores médio, máximo e mínimo de densidade de defeitos na compilação e teste unitário por programa até hoje?

Nas fase de *Test* não foram encontrados erros, pois foram corrigidos nas fases anteriores, porém na fase *Compile,* um erro ocorreu ao longo do programa 2, o erro ocorrido era de sintaxe e foi corrigido no mesmo momento.

**Pergunta 3:** Que tipo de defeito representa a maior parte do tempo gasto na compilação? Em teste? Qual a fase teve cada tipo de defeito injetado com mais freqüência?

Na fase de *Compile*, o erro encontrado no desenvolvimento do programa 2 foi um erro de sintaxe. Já na fase *Code,* que é a fase que provém a maior parte dos defeitos, mais da metade dos erros foram causados por problemas de sintaxe, mas foram encontrados também erros de documentação.

Os erros de sintaxe ocorreram, em sua maioria, por se implementar objetos novos e bibliotecas novas que não eram antes conhecidas, mas houve também falta de atenção em alguns casos, como no defeito que ocorreu pela falta de um ponto e vírgula.

Os erros de documentação foram descobertas na fase de desenvolvimento, onde se fez necessário alterar um diagrama.

##### POSTMORTEM

Na fase de *Postmortem,* o preenchimento do *Plan Summary* foi completado e o *Time Recording Log* foi realizado.

Nessa fase, constatou-se que o número de parágrafos quase dobrou em relação ao estimado, uma vez que se fez necessário comentar vários pontos importantes durante a análise.

As tabelas foram utilizadas para as montagens dos gráficos, porém elas fazem parte de uma ferramenta de criação de gráficos do Microsoft Word, que as mantém ocultas, de forma que sua função é apenas alimentar os gráficos.

Quanto aos gráficos, não se viu a necessidade de criar um para cada questão, uma vez que as questões são separadas por temas, permitindo que as questões compartilhem das mesmas informações dispostas no gráfico.

##### PLAN SUMMARY

1. PSP Interim Report plan summary

**PSP Interim Report****Plan Summary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Student | Sabrina Fontes Carvalho |  | Date | 04/12/2016 |
| Instructor | Carlos / Francisco / João |  |  |  |

**Size Data Effort Estimate**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Object** |  | **Plan Number** |  | **Actual Number** |  | **Est. Effort per Object** |  | **Estimated Effort** |
| Parágrafos |  | 15 |  | 27 |  | 5 |  | 75 |
| Tabelas |  | 14 |  | 0 |  | 5 |  | 70 |
| Gráficos |  | 14 |  | 5 |  | 2 |  | 28 |
|  |  |  |  |  |  | **Total** | | 173 |

Effort Data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase** |  | **Plan Time** |  | **Actual Time** |
| Planning |  | 40 |  | 32 |
| Development |  | 200 |  | 178 |
| Postmortem |  | 30 |  | 27 |
|  |  |  |  |  |

##### TIME RECORDING LOG

1. PSP Interim Report TIME RECORDING LOG

**PSP Interim Report****Time Recording Log**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Student: | | Sabrina Fontes Carvalho | | | Date: | | | | 04/12/2016 | |
|  | | |  | | | |  | | |  |
| Date | Start | | Stop | Interruption  Time | Delta  Time | Phase | | Comments | | |
| 04/11 | 11h43 | | 12h15 | - | 32 | Planning | | Estimativas | | |
| 04/11 | 13h20 | | 17h18 | - | 178 | Development | | Análise e desenvolvimento | | |
| 04/11 | 19h39 | | 20h06 | - | 27 | Postmortem | | Conferência e fechamento | | |

#### RESULTADOS

##### PLANNING

A fase de *planning,* a execução aconteceu abaixo do tempo estimado,

Porque o processo utilizado para realizar a análise, foi o mesmo sugerido no ASGKIT *Interim Report.* A opção de manter o processo sugerido ocorreu por falta de experiência no processo de análise.

Mesmo assim, após o término desse relatório, foi constatado que alguns outros processos poderiam ser incluídos, como a análise estática de código, para avaliar a qualidade do código produzido, e a análise de cobertura de código, para que se tenha um programa que alcance o objetivo proposto.

##### DEVELOPMENT

Nessa fase, catorze perguntas foram respondidas utilizando os formulários PSP *Project Summary*, *Time Recording Logs* e o *Defect Recording Logs* dos programas 1 a 4.

Através dela, inúmeras análise foram feitas e gráficos foram confeccionados para ajudar na visualização dos problemas e métricas dos programas.

##### POSTMORTEM

Na fase de *Postmortem,* foi feita a revisão de todos os documentos preenchidos nas fases anteriores e foram reportadas as horas gastas nas atividades no PSP *Interim Report Time Recording Log*.

## CONCLUSÕES

Através dos resultados que foram apresentados ao longo desse relatório, foi provado que uma aplicação bem planejada resulta em maior qualidade e assertividade na codificação e menor tempo de desenvolvimento.

Foi possível concluir também que, conforme o desenvolvimento de programas utilizando o PSP, melhor vai ficando as estimativas e a qualidade do código e conquentemente a diminuição dos defeitos.

## ELEMENTOS PÓS-TEXTUAIS

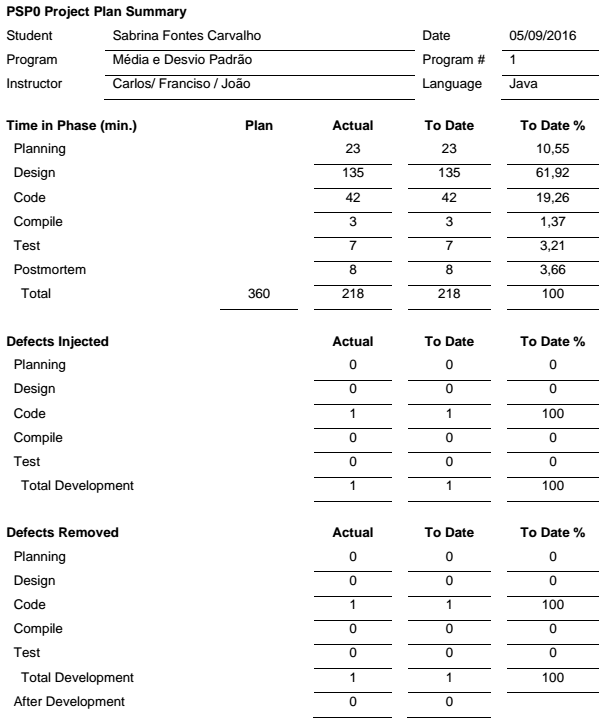
### REFERÊNCIAS

WEBGOAL. **Cronômetro Online**. Disponível em: <http://cronometronline.com.br/>.

### APÊNDICES

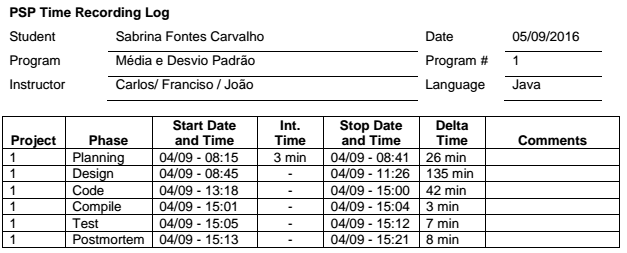
#### PSP PROJECT PLAN SUMMARY – PROG1

1. psp0 project plan summary



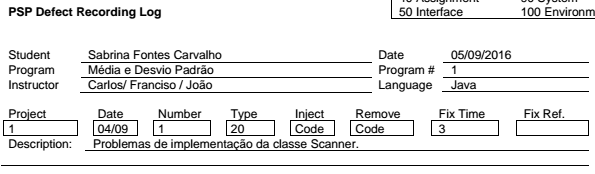
#### TIME RECORDING LOG – PROG1

1. psp TIME RECORDING LOG PROG1



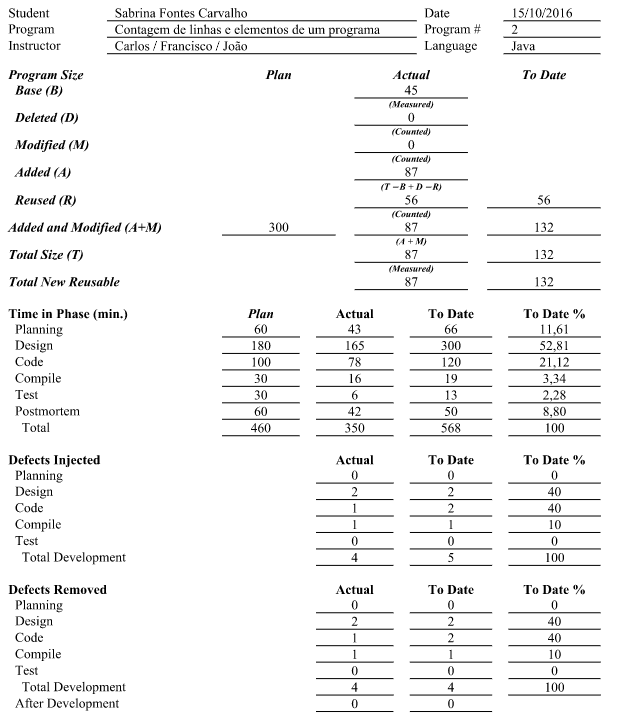
#### DEFECT RECORDING LOG – PROG1

1. psp DEFECT RECORDING LOG PROG1



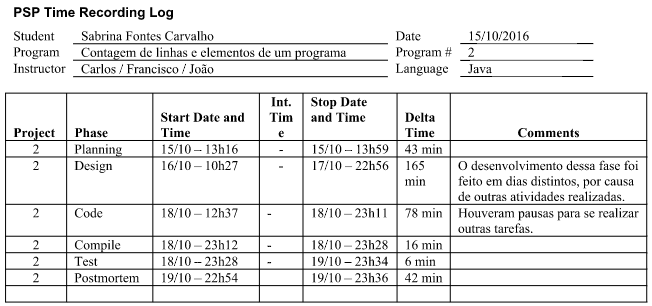
#### PSP PROJECT PLAN SUMMARY – PROG2

1. psp0.1 project plan summary



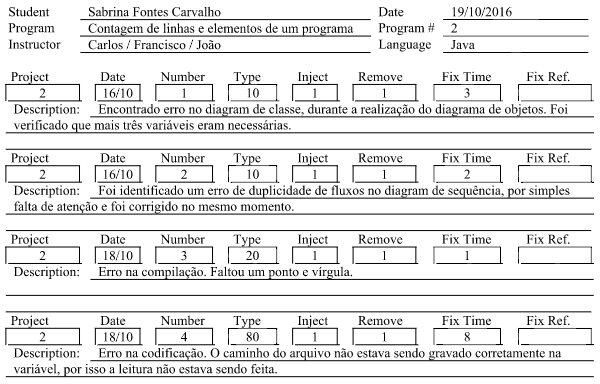
#### TIME RECORDING LOG – PROG2

1. psp TIME RECORDING LOG PROG2



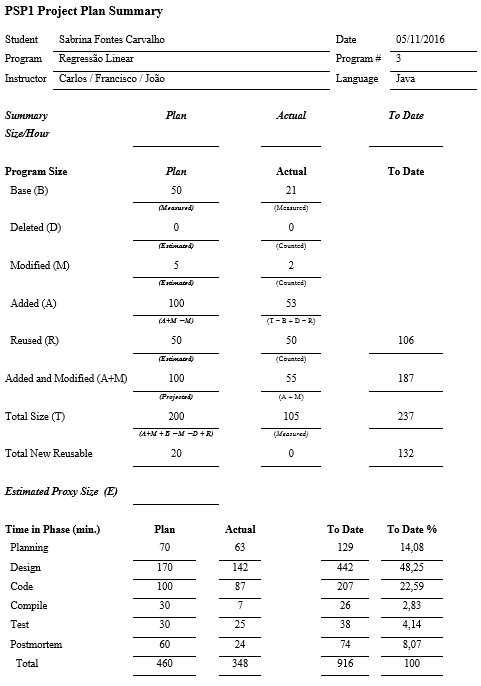
#### DEFECT RECORDING LOG – PROG

1. psp DEFECT RECORDING LOG PROG2

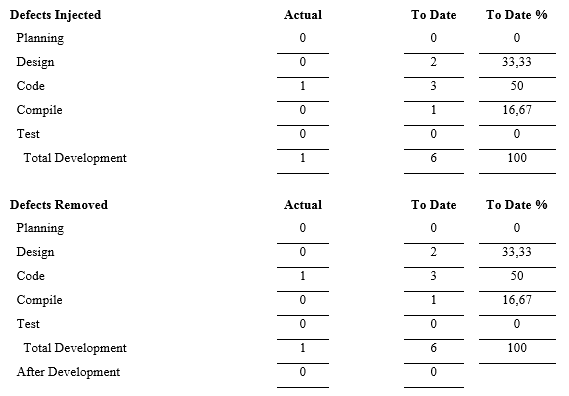


#### PSP PROJECT PLAN SUMMARY – PROG3

1. psp1 project plan summary – PARTE 1

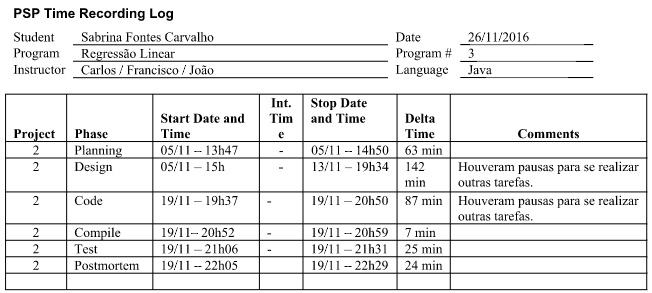


1. psp1 project plan summary – PARTE 1



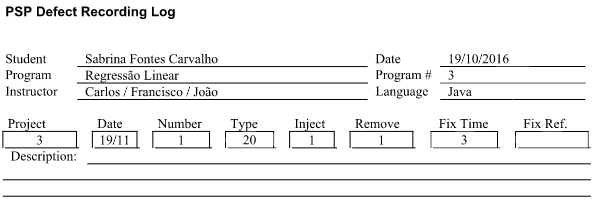
#### TIME RECORDING LOG – PROG3

1. psp TIME RECORDING LOG PROG3



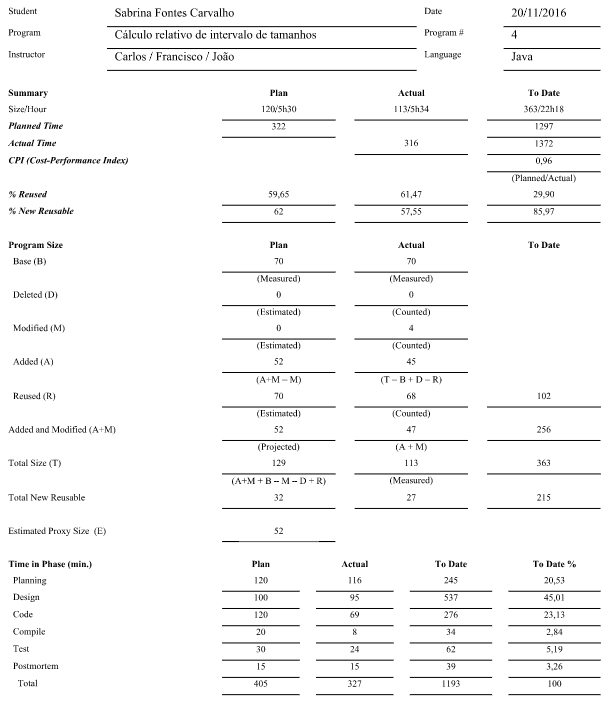
#### DEFECT RECORDING LOG – PROG3

1. psp DEFECT RECORDING LOG PROG3

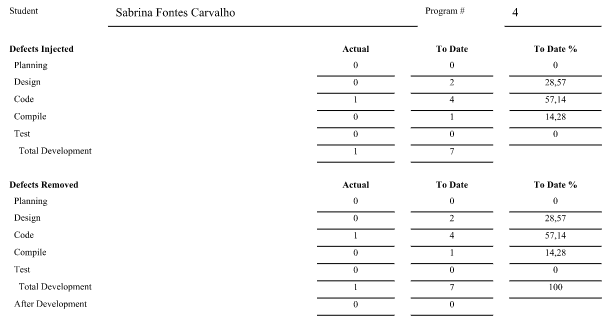


#### PSP PROJECT PLAN SUMMARY – PROG4

1. psp1.1 project plan summary – PARTE 1

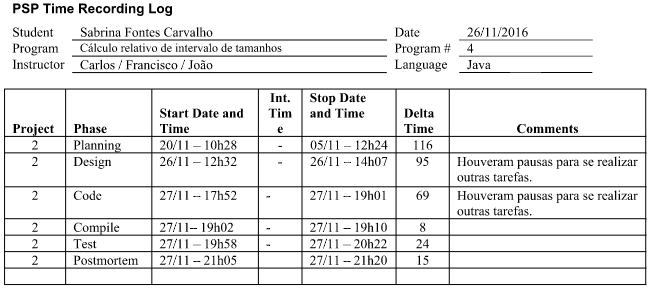


1. psp1.1 project plan summary – PARTE 2



#### TIME RECORDING LOG – PROG4

1. psp TIME RECORDING LOG PROG4



#### DEFECT RECORDING LOG – PROG4

1. psp DEFECT RECORDING LOG PROG4

