**Postmortem**

El tiempo estimado para la realización del programa 2 utilizando el nivel 1.0 de PSP fue de 231 minutos, repartidos de la siguiente manera:

|  |  |
| --- | --- |
| Etapa | Tiempo estimado |
| Plan | 90 |
| Diseño | 30 |
| Codificación | 40 |
| Compilado | 1 |
| Pruebas | 30 |
| P.M | 40 |
| Total | 231 |

**Fase de planeación:** para la etapa de planeación lo que se hizo fue en primer lugar hacer una consulta de las librerías que servirían para realizar el programa y como se utilizan y que excepciones pueden arrogar. Luego, planee las clases que iba a necesitar, los atributos y los métodos de estas para posteriormente realizar el diseño conceptual. Finalmente se rellenó el Size Estimated Template en la columna de plan, sin embargo aquí se cometió un error pues al desconocer cómo es que se llena correctamente el número de proxys no se correspondían con el del diseño. El método Probe es el D pues solo se tiene un punto histórico. De forma real se utilizaron 103 minutos lo que supone una diferencia de 13 minutos o un 14% más del tiempo planeado. Esta delta del tiempo se debió principalmente en que se gastó mucho tiempo para realizar el diseño y luego por tratar de cumplir con el tiempo no se llenó bien el Size Estimated Template.

**Fase de diseño:** Una vez se cuenta con el diseño conceptual el diseño detallado se facilita bastante por lo que para hacerlo no se empleó mucho más tiempo de lo planeado, además de hacer esto en esta fase se documentaron los Test Report, la primera es la prueba unitaria con el programa uno que se hizo en el nivel 0, pero luego de implementarle el código estándar. La otra prueba unitaria era la del segundo programa, es decir, este mismo. El tiempo real fue de 33 minutos, 3 minutos más de lo planeado lo que supone un 10% más de tiempo. A pesar de que el desfase fue poco lo mejor sería emplear más tiempo ya que se cometieron 2 errores con el diseño detallado.

**Fase de codificación:** Se tenían que hacer 5 clases: DatosClase, que es la que tiene la información de una clase; DatosProyecto, contiene la información de todas la clases del proyecto, para esta clase tuvo un contratiempo y fue que no planee el atributo numeroLocsTotales por lo que tuve que agregarlo y también hacer su método Setter y Getter; BuscarClases, la cual busca los archivos .java para esta no hubo inconveniente; Contador, es en la que se cuentan los LOCs de cada clase es en la que me tome más tiempo ya que se puede decir que es la más importante pues es en la que se obtienen todos los datos; Principal, es en donde se inserta la ruta y luego se muestran los resultados. El tiempo real fue de 66 minutos, 26 más de lo planeado o un 65% de más. Esto se debió a lo que ya comente del atributo numeroLocsTotales, documentar el error y luego corregirlo en el código. Luego empleé mucho tiempo en la clase Contador. Tampoco tuve en cuenta que como en el código estándar indique que se debían poner comentarios instructivos y hacer esto también consume bastante tiempo.

**Fase de compilación**: se llevó a cabo sin ningún problema, no tuve ningún error por lo que tarde 1 minuto justo lo que había planeado. Sin embargo, no realice ninguna prueba pequeña para garantizar que en las pruebas unitarias los resultados sean los esperados y como se verá a continuación si hubieron defectos que se pudieron haber evitado si se hacían pruebas en esta fase.

**Fase de pruebas unitarias:** AL realizar las pruebas unitarias surgieron algunos errores: el primero, fue que los comentarios si se estaban contando y esto se debía a que primero había que eliminar los espacios de las líneas y luego si comprobar que empiece con algún comentario. Sin embargo el error siguió debido a que estaba usando el operador lógico ***“or”*** y era ***“and”*** entonces por ejemplo se compruebe que la línea no empiece con ***“//”*** y sale true y luego sigue que si empieza con ***“\*”*** y es false, entonces no se debería contar pues es un comentario múltiple, pero al tener el operador “or” si lo cuenta pues con ***“or”*** solo basta que uno sea verdadero. Luego, surgió un error al contar los métodos ya que como estaba lo que se comprobaba era que si la línea contenía public o private o protected y que si contenía ***“(“***y ***“)”*** y ***“{“***es porque era el inicio de un método. Lo que sucedió es que al contar el programa a sí mismo la línea que hacia el chequeo contaba con todos estos elementos y entonces se contaba como método cuando no lo es. Entonces para corregir esto cambie la lógica y el patrón que debe seguir es que la línea debe empezar con public o private o protected y debe llevar ***“(“***y ***“)”***, y se descartó la llave ***“{“***debido a que un método de una interface no lleva entonces no se contaría. El tiempo que gaste en esta fase fueron 45 minutos, lo que supone un 50% más de tiempo o 15 minutos.