U.A.N.L.

Facultad de Ciencias Físico Matemático

Materia: Diseño Orientado a Objetos

Maestra: Lic. Miguel Ángel Salazar Santillán

“Los antipatrones de diseño”

Alumno: José Israel Guerrero Ibarra

Matrícula: 1734152

San Nicolás de los Garza, Mty., Nuevo León

28/04/2017

Antipatrones de Diseño

Los Antipatrones son ejemplos bien documentados de malas soluciones para problemas. Se estudian a fin de poderlos evitar en el futuro y se origina como una contra parte al término patrón, originado en la arquitectura de software, para definir las buenas prácticas de programación, diseño o gestión de sistemas. De tal manera podríamos hablar de que un sistema “bien hecho” está lleno de patrones, y debería carecer de anti-patrones.

Al documentarse los antipatrones, además de los patrones de diseño, se dan argumentos a los diseñadores de sistemas para no escoger malos caminos, partiendo de documentación disponible en lugar de simplemente la intuición. Los antipatrones se consideran una parte importante de una buena práctica de programación. Es decir, un buen programador procurará evitar los antipatrones siempre que sea posible, lo que requiere su reconocimiento e identificación tan pronto como sea posible, dentro del ciclo de vida del software.

El concepto de antipatrón se puede aplicar a la ingeniería en general, e incluso a cualquier tarea realizada por el hombre. Aunque no se escucha con frecuencia fuera del campo ingenieril, la noción está ampliamente extendida.

A continuación se revisarán algunos ejemplos y tipos de antipatrones que existen.

Anti-patrones de Codificación

1. Lava Flow: Algo así como “programar al estilo volcán”. Es construir grandes cantidades de código de manera desordenada, con poca documentación y poca claridad de su función en el sistema. Conforme el sistema avanza en su desarrollo, y crece, se dice que estos flujos de lava se solidifican, es decir, se vuelve mucho más complicado corregir los problemas que originan, y el desorden va creciendo geométricamente.

Esto puede suceder cuando se declaran variables no justificadas, se construyen clases o bloques de código muy grandes y complejas sin documentar, o que no se relacionan claramente con la arquitectura, cuando en el sistema existen muchas áreas con código por terminar o reemplazar o cuando dejamos código sin uso abandonado; interfaces o componentes obsoletos en el cuerpo del sistema.

Conforme los flujos se endurecen y solidifican (se escribe código y pasa el tiempo), rápidamente se vuelve imposible documentar el código o entender su arquitectura lo suficientemente bien como para hacer mejoras.

2. The God: Un programa omnipresente y desconocido. Aquel sistema donde una sola clase ó modulo (la función main o equivalente) hace todo. Así que el programa es un solitario y único archivo de muchísimas líneas. En consecuencia, tenemos un código desorganizado y fuertemente interdependendiente.

Anti-patrones de Arquitectura

3. Reinventar la rueda: Se refiere a re-implementar componentes que se pueden conseguir prefabricados de antemano, y hacer poco reúso en el código. En breves palabras: querer hacer todo uno mismo.

Podríamos hablar de poco nivel de reúso en el código, reúso de un proyecto a otro, con lo cual cada proyecto está comenzando desde cero, o cuando constantemente se reescriben fragmentos de código con la misma funcionalidad, o el software se vuelve innecesariamente más denso.

4. Stovepipe: Cocinado en “caliente”. Es la forma breve de referirse a la creación de islas automatizadas dentro de la misma empresa (cada departamento crea su propio subdepartamento de sistemas). “Islas” independientes, y en conflicto unas con otras. Cada isla desarrolla la parte del sistema que necesita para satisfacer sus requerimientos, sin preocuparse por el resto (no existe un plan o eje guía), el escenario resultante implica una pobre o nula interoperabilidad, obviamente, no existe el reúso y, consecuentemente se incrementan costos.

Anti-patrones de Administración de Proyecto

5. The Mythical Month Man: Mejor conocido como en el entorno como el “súper equipo de programadores”. Consiste en la creencia de que asignar más personal a un proyecto, acotará el tiempo de entrega. Regularmente como una forma desesperada de intentar corregir retraso del proyecto. Llega un punto donde entre más personal se asigne, más se retrasa el proyecto.

6. Project Miss-management: La jefa o el jefe que no saben coordinar. El proyecto se descuida y no se monitorea de manera adecuada, es muy difícil de detectar en etapas iniciales, pero repentinamente emerge de golpe y suele voltear de cabeza la situación del proyecto. Se manifiesta con retrasos en las fechas de entrega y/o áreas incompletas.

El listado de anti-patrones es muy amplio, casi tanto como el de patrones. Algunos de estos patrones, son incluso irrisorios, errores tan comunes que se supondría nadie podría cometerlos, y sin embargo, reflexionemos, cuántos de nosotros, estudiantes, profesionistas, es la primera opción a la que recurrimos.