

Sherlock Holmes y el caso del programa inoperante

Adaptado y extractado de:

<http://www.extropia.com/tutorials/misc/sherlock.html>

Vamos a abordar una cuestión harto conocida: “He hecho un programa, y no funciona”. Sí, una de esas cosas que suelen pasar varias veces al día. La forma en que responderemos a esta cuestión es a través de una historia. Vamos a contar una historia de misterio.

Es una historia sobre cómo encontrar al culpable cuando no se está absolutamente seguro sobre qué estará pasando por los entresijos del programa. Al culpable de que el próximo fin de semana haya que quedarse pegaditos al ordenador tirándonos, desesperados, de los pelos por ese dichoso método que no acaba de funcionar.

Es un misterio que, como la mayor parte de los misterios, comienza con Sir Arthur Conan Doyle.

Veamos lo que Doyle tiene que decir sobre depuración de programas:

“Por las uñas de una persona, por su abrigo, por sus botas, por las rodilleras de sus pantalones, por las callosidades de su índice y de su pulgar, por su expresión, por los puños de su camisa... Por cada una de estas cosas la culpabilidad de un hombre se revela claramente. Que todo eso unido no sirva al investigador competente para resolver el misterio es casi inconcebible.”
De “Un estudio escarlata”.

Bien, esto puede no parecer un comentario sobre depuración de software, pero en realidad lo es. Lo que Doyle está tratando de decir es que todos los errores de software y de hardware DESEAN ser atrapados. De hecho lo desean tan desesperadamente, que nos dejan pistas a su alrededor. Quizá Doyle pretendía decir algo como lo siguiente:

“Por los mensajes de error al ejecutar, por la salida obtenida, por los mensajes del compilador, por la interacción de sus métodos, por sus llamadas a funciones de biblioteca y las respuestas que obtienen de éstas... Por cada una de estas cosas los errores del programa se revelan claramente. Que todo esto unido no sirva al depurador competente para hacer funcionar el programa es casi inconcebible.”
De “Un estudio software”.

Como depurador, es tu trabajo escuchar todas estas pistas, construir con ellas una teoría que se pueda probar y contrastar dicha teoría con el programa. En **todos** los casos, te acabarás dando una fuerte palmada en la frente y diciéndote a ti mismo: “¡Claro, era eso! ¿Cómo no me he dado cuenta antes?”. Porque, en el fondo, los

ordenadores son criaturas muy simples, y cuando se estropean siempre es por razones muy simples.

La virtud de la nada

Benjamin Hoff una vez contaba esta interesante historietita sobre el Taoísmo, que puede resultarte muy útil:

“Estoy aprendiendo”, dijo Yen Hui.

“¿Cómo?”, preguntó el Maestro.

“Olvidé las reglas de la Corrección y los niveles de la Benevolencia”, respondió.

“Bien, pero podrías hacerlo mejor”, dijo el Maestro.

Unos días después, Yen Hui comentó: “Estoy haciendo progresos”.

“¿Cómo?”, preguntó el Maestro.

“Olvidé los Rituales y la Música”, respondió.

“Mejor, pero aún no es perfecto”, dijo el Maestro.

Algún tiempo después, Yen Hui dijo al Maestro: “Ahora me senté y lo olvidé todo”.

El maestro lo miró, asombrado: “¿Qué significa que lo olvidaste todo?”.

“Olvidé mi cuerpo y mis sentidos, y dejé atrás toda apariencia e información”, respondió Yen Hui. “En medio de la Nada, me uno a la fuente de Todas las Cosas”.

El Maestro asintió: “Has trascendido las limitaciones del tiempo y del conocimiento. Estoy muy por detrás de ti. ¡Has encontrado el Camino!”.

Benjamin añadía: “Una mente vacía es valiosa a la hora de encontrar cosas porque puede ver lo que tiene delante. Una mente demasiado llena es incapaz”.

¿Qué tiene esto que ver con la depuración de programas, te estarás preguntando? Bien, tiene todo que ver con depuración de programas. La depuración no es una habilidad. No es algo que se aprenda en la escuela. Tampoco es algo que se pueda hacer a través de FAQs, libros, administradores del sistema o foros de discusión.

La depuración es un estado de la mente.

Yo cuando me paso más de una hora con un problema, paro. Muy pocos problemas necesitan más de una hora para resolverse, así que si he estado sentado tratando de resolverlo más de una hora, entonces puedo estar seguro de que lo más probable es que el problema que tengo no es el fallo del programa, sino yo mismo.

En ese momento, apago el monitor, enciendo una vela y algo de incienso (siempre tengo incienso en el cajón de mi escritorio), pongo un poco de música y me tumbo en el suelo tratando de aislar cada instrumento de la canción que esté sonando.

Nota: para depurar programas recomiendo “Technotronic: The Best of Trance”, cualquier cosa de Cocteau Twins o Enya, o también “Wish You Were Here” de Pink Floyd o “Kiss” de The Cure.

A veces incluso salgo y camino alrededor de la manzana si el día es cálido y soleado... además hay un buen árbol para escalar enfrente de mi oficina.

Unos veinte minutos después suelo estar listo para regresar al trabajo, tras haber conseguido varias cosas importantes:

- He disminuido las posibilidades de tener un ataque al corazón a los 35.
- No estoy ni enfadado ni frustrado.
- He vaciado mi mente de todas mis ideas preconcebidas sobre lo que creo que el fallo me está diciendo, y estoy preparado para “escuchar” al fallo y descubrir lo que tiene verdaderamente que decirme.
- No me siento intimidado por el programa. Programar es como montar a caballo: en cuanto el programa empieza a creer que él es el que manda, en ese momento te tira abajo.

La Metodología Newtoniana y el “kid” de la depuración

Tras retornar de la nada, lo primero que hago es dejar a un lado el programa y empezar codificando algo verdaderamente pequeño.

Ya ves, depurar es un ejercicio Newtoniano. Y en un universo Newtoniano lo mejor que se puede hacer es descomponer todo en las partes más pequeñas en que sea posible, porque en este caso el todo es la suma de las partes, y cuando se encuentre la parte que falla, se habrá encontrado el problema. (1)

Comenzando por el Hola Mundo

De esta forma, deberías empezar por minimizar todo lo que puedas el código del método, comentando todo lo demás.

Empezando por fragmentos de código pequeños podemos ir estando “relativamente” seguros de que lo que vamos haciendo “no tiene” errores.

A partir de ahí, debes ir añadiendo (“descomentando”) código poco a poco, y después empezar a ver como interactúan entre sí los módulos.

Cuidado, porque a partir de este momento la cosa se complicará rápidamente. Hay que saber siempre en dónde se está y no perderse...

En conclusión

Bien, eso es todo amigos. Si te sientes cómodo con las indicaciones dadas aquí y estás preparado par meterte en faena, entonces no deberías preocuparte. Piensa en la depuración como en un juego divertido. De hecho, para practicar, pásate por algún foro de discusión sobre Java e intenta ayudar a la gente a resolver sus problemas. De esa forma no solamente aumentarás tus habilidades, sino que además contribuirás a hacer de la comunidad de desarrolladores Java un grupo al que merezca más la pena pertenecer. Buena suerte.

Notas

1. Por cierto, la perspectiva Newtoniana es horrible para el proceso de “creación” del software, que es un proceso complejo y no un proceso Newtoniano. Si estás

familiarizado con los conceptos de sistemas complejos y de propiedades emergentes, olvídalas durante el proceso de depuración, pues no te serán más útiles que el viejo paradigma Newtoniano. Si no estás familiarizado con estos conceptos, te recomiendo que leas “Out of Control: The New Biology Machines, Social Systems and the Economic World”, de Kevin Kelly. Con toda seguridad hará de ti un mejor diseñador de software.