Índice

Un caso real: crisis en Amazon Web Services[1]

El ingeniero jefe Chris Newcombe se dirigió a su compañero y a la vez subordinado, Tim Rath, con estas palabras:

—Tim, tenemos que hacer algo. ¿Te puedes creer que, después de todos los tests que hicimos a S3, todavía siguen apareciendo errores?

S3 era el nombre comercial de un sistema de almacenamiento en la nube que el grupo había desplegado pocos años antes y que usaban millones de usuarios en todo el mundo. En Amazon Web Services (AWS, para sus empleados), las cosas se hacían a lo grande.

—No es posible —respondió Tim—, yo mismo me encargué de corregir el último error y pasé de nuevo todos los tests. ¿Qué ha sido esta vez?

—El sistema de replicación de las tablas —dijo Chris—. Al parecer, se produjo un bloqueo temporal en la red y una de las

[1] (*Nota del autor*: Amazon Web Services es un departamento de ingeniería de la empresa Amazon, que suministra servicios de computación en la nube para esta y otras compañías tales como Dropbox o Netflix. Surgió en 2006 para dar soporte informático a la tienda en línea de Amazon. Los servicios que provee son extremadamente complejos y acceden a ellos millones de usuarios en todo el mundo.
El relato que comienza en este capítulo y continúa al comienzo de los que siguen es una recreación ficcionada por el autor de hechos y personajes reales tomados del artículo: Chris Newcombe, Tim Rath, Fan Zhang, Bogdan Munteanu, Marc Brooker, Michael Deardeuff, «How Amazon Web Services Uses Formal Methods», *Communications of the ACM 58*(4), pp. 66-73, 2015.)

copias ha quedado inconsistente. Hemos tenido que reiniciar el servidor y volver a cargar los datos. Como sabes, cada pérdida de datos nos penaliza y los superjefes no se toman estas cosas con buen humor.

—Me imagino —asintió Tim—, pero no se me ocurre cómo mejorar nuestros protocolos de trabajo. Hacemos test exhaustivos, inyectamos fallos a propósito para ver cómo reacciona el sistema, hacemos revisiones de código ¿qué más podemos hacer? Tú eres el jefe aquí. ¿Por qué no te das una vuelta por algún congreso a ver si das con un investigador que nos ayude?

Chris era un tanto reacio a esta idea. Nunca había confiado mucho en los científicos que, en su opinión, se pasaban la vida ideando teorías muy complejas, que luego plasmaban en lenguajes de juguete para resolver problemas de juguete. La vida real era otra cosa. Ellos, los ingenieros de AWS, tenían que bregar con centenares de miles de líneas de código y muchos miles de procesos ejecutándose simultáneamente en cientos de máquinas. El propio sistema S3 estaba procesando en esos momentos 1,1 millones de transacciones por segundo ¿Qué científico se ha enfrentado alguna vez a algo semejante?

A pesar de que, antes de desplegarlos, sometían a sus sistemas a pruebas exhaustivas, revisiones de código y ataques maliciosos intencionados, no eran infrecuentes los casos en que algún servidor caía, o se producían conflictos entre servidores en el acceso a los datos, dando como resultado la pérdida o la corrupción de una parte de los mismos. Estudiando la causa de estos fallos, observaban que se producían situaciones que nunca hubieran imaginado que pudieran llegar a materializarse. Al no estar prevista una reacción adecuada a las mismas, el software reaccionaba de manera incorrecta y se producía la catástrofe.

La situación no era, pues, satisfactoria. Cada vez que ponían un nuevo sistema en explotación, Chris se quedaba con la incómoda sensación de que tal vez se les había pasado alguna combinación de eventos por alto y de que cualquier día se produciría la combinación fatídica y aparecería un nuevo error, tal como

había sucedido en esta ocasión. Así que decidió seguir parcialmente el consejo de Tim. En lugar de buscar un congreso, se dedicó a bucear durante unos días en la literatura científica dedicada a los sistemas distribuidos.

A pesar de su escepticismo, leyó una buena cantidad de artículos sobre sistemas y algoritmos distribuidos, hasta que la suerte le sonrió y le hizo dar con uno en el que una investigadora —Pamela Zave, del Instituto de Tecnología de Massachusetts— afirmaba haber utilizado una herramienta llamada Alloy para la verificación de un sistema distribuido real. Con ella, había encontrado varios errores importantes en un complejo protocolo de sincronización de dicho sistema. El sistema fue finalmente desplegado y alcanzó un éxito notable por su mínima tasa de fallos. Tanto, que el artículo que describía la experiencia obtuvo en 2011 un premio en la conferencia anual del *ACM Special Interest Group on Data Communication*.

—Tim, lo tengo —dijo esa mañana a su colega—. Esta mujer parece que sabe lo que se trae entre manos. Por fin, he dado con un artículo que habla de la vida real, de un sistema real y de un funcionamiento sin fallos real. Parece que la clave está en esa herramienta que dice haber usado. Me pongo a ello, ya te contaré.

—¡Que no te pase nada! ¿No será uno de esos juguetes que usan los científicos, verdad? —respondió escéptico Tim—.

Chris se interesó por Alloy. Al fin y al cabo, los conceptos que utilizaba —conjuntos, relaciones, y un repertorio limitado de fórmulas lógicas— estaban al alcance de su formación como ingeniero. En pocos días, pudo escribir y verificar con Alloy uno de sus algoritmos distribuidos. Se percató de que el enfoque de Alloy consistía en comprobar de forma implícita todas las trazas del sistema. Una traza es una secuencia temporal de eventos que pueden proceder de varios procesos. Al estudiar todas las trazas, en realidad se investigaban todos los entrelazados de eventos que podían darse en una ejecución real, cuando dichos procesos se ejecutaran en paralelo.

Era justo lo que necesitaba. Siguió trabajando con Alloy y pronto se hizo consciente de sus limitaciones: en cuanto el problema aumentaba de volumen, o se necesitaban tipos de datos algo más complejos que los números enteros, la herramienta no tenía la expresividad necesaria y además requería una enorme cantidad de tiempo para completar las demostraciones.

—Tim —informó a su compañero—, el enfoque de la herramienta es el correcto, pero necesitamos algo más potente. Sigo buscando.

—Estos científicos hacen herramientas de juguete para sus problemas de juguete. Ya te lo advertí.

Pero la semilla de la curiosidad ya estaba plantada. Chris comprendió que, a veces, lo más práctico era disponer de una buena teoría. Los problemas complejos a los que se enfrentaban —reflexionó— no se podían resolver a base de fuerza bruta, y eso era lo que él y sus ingenieros habían estado haciendo hasta ahora: a más líneas de código, más horas de revisiones y de tests, más horas de máquina y más ingenieros. Pero había comprobado con dolor que tanto esfuerzo era insuficiente.

Siguió investigando teorías y herramientas y cada vez se sentía más esperanzado en encontrar una solución. Un día, en el apéndice de un artículo sobre un conocido algoritmo distribuido, el protocolo Paxos, mediante el cual un conjunto de máquinas conectadas a una red de comunicación se sincronizan para alcanzar un consenso sobre una decisión, descubrió una extraña notación en la cual estaba recogida toda la complejidad del algoritmo, ¡en tan solo una página! Las descripciones del algoritmo que él conocía necesitaban varias páginas de explicaciones y algunos diagramas para mostrar todas las sutilezas del protocolo. De inmediato, le entraron deseos de estudiar esa notación y de poder experimentar con ella otros algoritmos.

Aumentaba su motivación el hecho de que el autor de la misma, Leslie Lamport, era también el inventor del protocolo, el cual había llegado a hacerse un hueco en el mundo de los algoritmos distribuidos. También supo que la técnica se había usado

en Digital Equipment Corporation, donde trabajaba Lamport, para especificar y verificar un endiablado protocolo de coherencia de memoria de uno de los multiprocesadores fabricados por esta compañía. Tenía dudas sobre si podría expresar en ella las situaciones de fallo, que eran la pesadilla de los sistemas construidos por AWS. El protocolo de Digital suponía que no habría fallos en el hardware.

Chris reescribió en TLA+ —así se llamaba la notación de Lamport— los mismos protocolos que había escrito en Alloy, en los que sí se contemplaban posibles fallos del hardware. Enseguida comprobó que el lenguaje era mucho más expresivo y, también, que la herramienta de verificación era mucho más rápida.

Cuando estuvo convencido, se lo comunicó a Tim. De todos sus ingenieros, era con quien tenía más confianza.

—Tim, creo que esta vez he acertado. Me gustaría que usaras TLA+ en el nuevo proyecto.

—Ni lo sueñes Chris, lo que tengo entre manos es demasiado complejo para dedicarme a jugar con tus herramientas matemáticas. No tengo mucho tiempo disponible, ¿sabes?

Lo que Tim tenía entre manos era una nueva base de datos distribuida, a la que más adelante bautizaron como DynamoDB, que AWS pretendía lanzar en enero del año siguiente. Precisamente, Tim era el responsable de la parte de replicación y coherencia entre copias de la futura base de datos.

—Te dejaré un tiempo para que completes los tests habituales pero, en cuanto empiecen a aparecer los errores, tendrás que probar la herramienta. No te lo pediré, Tim, te lo ordenaré como tu jefe. No sé si me explico.

—¡Vaya, cómo te pones! Esta vez sí que te ha dado fuerte. No te preocupes, te obedeceré con toda disciplina. Pero no me hago responsable de los retrasos.

Así quedaron las cosas. Tim siguió trabajando en su nuevo proyecto y Chris continuó experimentando con su recién descubierta herramienta. ¿Quién tenía razón? No lo sabemos, pero invitamos al lector a que siga leyendo.