

Especificación y Verificación Formal de Sistemas Distribuidos con TLA+

01. Introducción

Axel Suárez Polo

October 13, 2022

BUAP

¿Por qué aprender TLA+?

¿Qué es TLA+ y TLC?

¿Por qué no aprender TLA+?

¿Por qué aprender TLA+?

¿Por qué aprender TLA+?

- Escribir la *especificación* de un sistema ayuda a entenderlo
- Es buena idea entender un sistema antes de implementarlo.
- Por lo tanto, es buena idea *especificar* un sistema antes de implementarlo.

¿Por qué aprender TLA+?

- La especificación de un sistema puede ir desde simple prosa, hasta una especificación matemática, como ocurre en la ciencia y la ingeniería.
 - En la realidad, los planetas tienen montañas, océanos, olas, clima, etc
 - Pero los podemos modelar como puntos de masa con posición y momento

¿Por qué aprender TLA+?

- Este mismo método científico lo podemos aplicar a los programas:
 - En la realidad, hay procesos físicos ocurriendo en las computadoras, transistores cambiando de estado, diferencias de voltaje, etc.
Incluso a un nivel de abstracción más alto existen direcciones de memoria, estados de registros, etc.
 - Pero podemos describir lo que hace una computadora con modelos como las **máquinas de Turing**, el **cálculo lambda**, etc.

¿Por qué aprender TLA+?

- TLA+ nos da un modelo para razonar sobre casi cualquier sistema discreto [Lam99].
 - Diseño de hardware [LSTY01].
 - Sistemas concurrentes y distribuidos [NRZ⁺15].
 - Sistemas operativos [VBF⁺11].
 - Algoritmos como 2PC, Paxos, alojamiento de memoria, etc [tla16].

¿Qué es TLA+ y TLC?

¿Qué es TLA+?

- TLA+ un lenguaje de alto nivel para modelar sistemas digitales. [Lam21].
 - **sistemas digitales** abarca tanto algoritmos como sistemas de computadoras.
 - **alto nivel** hace referencia a que se está hablando a nivel de diseño y no de código ejecutable.




¿Qué es TLC?




- TLC es una herramienta para verificar estos modelos automáticamente.
 - Estas herramientas permiten encontrar y corregir errores de diseño. Hoy en día los errores de diseño ocupan los primeros lugares en vulnerabilidades de seguridad [Fou21].



¿Por qué no aprender TLA+?

¿Por qué no aprender TLA+?

- En el espectro de los métodos formales, TLA+ presenta una dificultad mayor a la de los sistemas de tipos de lenguajes como C o Java [FLR17].
- Existe software simple de escribir y software difícil de escribir.
- Existe software crítico y software que no lo es.
- **No se puede verificar que el software implementado efectivamente sigue la especificación.**

-  Kathleen Fisher, John Launchbury, and Raymond Richards, *The hacms program: using formal methods to eliminate exploitable bugs*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **375** (2017), no. 2104, 20150401.
-  OWASP Foundation, *A04:2021 - Insecure Design*, https://owasp.org/Top10/A04_2021-Insecure_Design/, 2021, [Online; accessed 13-Oct-2022].
-  Leslie Lamport, *Specifying concurrent systems with tla+*, Calculational System Design (1999), 183–247.

-  ———, *TLA+ Video Course*, <https://lamport.azurewebsites.net/video/videos.html>, 2021, [Online; accessed 13-Oct-2022].
-  Leslie Lamport, Madhu Sharma, Mark Tuttle, and Yuan Yu, *The wildfire challenge problem*.
-  Chris Newcombe, Tim Rath, Fan Zhang, Bogdan Munteanu, Marc Brooker, and Michael Deardeuff, *How amazon web services uses formal methods*, Communications of the ACM **58** (2015), no. 4, 66–73.

-  tlaplus, *tlapplus Examples*,
<https://github.com/tlaplus/Examples>, 2016,
[Online; accessed 13-Oct-2022].
-  Eric Verhulst, Raymond T Boute, José Miguel Sampaio Faria,
Bernhard HC Spath, and Vitaliy Mezhuyev, *Formal
development of a network-centric rtos: software
engineering for reliable embedded systems*, Springer
Science & Business Media, 2011.