Práctica 3. Tolerancia a Fallos

1. Objetivos y Requisitos

Esta práctica tiene como objetivo el diseño y la implementación de un sistema distribuido con técnicas de tolerancia a fallos.

1.1 Objetivo

- Familiarización con técnicas de tolerancia a fallos
- Diseño e implementación de tolerancia a fallos de tipo crash, omission, timing y response en una arquitectura *master worker*, tanto en el máster como en el *worker*

1.2 Requisitos

- Elixir y su entorno de desarrollo
- Seguir la guía de codificación de Elixir publicada aquí:

https://github.com/christopheradams/elixir_style_guide/blob/master/README.md

2. Descripción del Problema

Partiendo de un escenario parecido al escenario 3 de la práctica 1, **se pide** *diseñar un sistema distribuido* que incorpore estrategias de tolerancia a fallos de manera que se intente garantizar el QoS de cada una de las tareas. Esta vez, el QoS que mide el cliente se va a medir de forma más simple, cada tarea debe terminar en 2,5 segundos. Se proporciona el generador de la carga de trabajo que deberéis modificar para verificar que se cumple el QoS. También se proporciona el código del Worker que no se puede modificar.

Opcionalmente, se puede implementar el algoritmo del matón sobre el máster y simular errores en el máster.

Deberéis escribir una memoria[1] de hasta 5 páginas de extensión. En ella deberéis proporcionar:

1) Análisis y clasificación de los fallos que pueden aparecer
2) Estrategias de detección para cada uno de los fallos
3) Estrategias de corrección para cada uno de los fallos
4) Modificaciones arquitecturales respecto del escenario 3 de la práctica 1

Para obtener la calificación máxima A+ en la solución del problema, deberá implementarse el algoritmo del matón en el máster (en cuyo caso, el máster no puede tener estado). Y se deberá simular fallos en él para poder probarlo. En caso de no implementar el Algoritmo del máton, se podrá obtener una calificación hasta A en la solución del problema.

Respecto de la tolerancia a fallos, podéis utilizar las estrategias que deseéis, se valorará que el sistema minimice las violaciones del QoS y gestione los recursos de forma inteligente.

---

[1] Tenéis disponible una plantilla en latex para poder mejorar la presentación de la memoria. Podéis acceder a ella a través de este link: https://www.overleaf.com/read/bkfrktpwmvdr

4. Evaluación

La realización de las prácticas es por parejas, pero los dos componentes de la pareja deberán entregarla de forma individual. En general estos son los criterios de evaluación:

- Deben entregarse *todos* los programas, se valorará de forma negativa que falte algún programa.
- Todos los programas deben compilar correctamente, se valorará de forma muy negativa que no compile algún programa.
- Todos los programas deben funcionar correctamente como se especifica en el problema.
- Todos los programas tienen que seguir el manual de estilo de Elixir, disponible en[2] (un 20% de la nota estará en función de este requisito). Además de lo especificado en el manual de estilo, cada fichero fuente deberá comenzar con la siguiente cabecera:

```
# AUTORES: nombres y apellidos
# NIAs: números de identificaci'on de los alumnos
# FICHERO: nombre del fichero
# FECHA: fecha de realizaci’on
# TIEMPO: tiempo en horas de codificación
# DESCRIPCI'ON: breve descripci'on del contenido del fichero
```

4.1. Rúbrica

Con el objetivo de que, tanto los profesores como los estudiantes de esta asignatura por igual, puedan tener unos criterios de evaluación objetivos y justos, se propone la siguiente rubrica en la Tabla 1. Los valores de las celdas son los valores mínimos que hay que alcanzar para conseguir la calificación correspondiente y tienen el siguiente significado:

A+ (excelente). En el caso de software, conoce y utiliza de forma autónoma y correcta las herramientas, instrumentos y aplicativos software necesarios para el desarrollo de la práctica. Plantea correctamente el problema a partir del enunciado propuesto e identifica las opciones para su resolución. Aplica el método de resolución adecuado e identifica la corrección de la solución, sin errores. En el caso de la memoria, se valorará una estructura y una presentación adecuadas, la corrección del lenguaje, así como el contenido explica de forma precisa los conceptos involucrados en la práctica. En el caso del código, este se ajusta exactamente a las guías de estilo propuestas.

A (bueno). En el caso de software, conoce y utiliza de forma autónoma y correcta las herramientas, instrumentos y aplicativos software necesarios para el desarrollo de la práctica. Plantea correctamente el problema a partir del enunciado propuesto e identifica las opciones para su resolución. Aplica el método de resolución adecuado e identifica la corrección de la solución, con ciertos errores no graves. Por ejemplo, algunos pequeños casos (marginales) no se contemplan o no funcionan correctamente. En el caso del código, este se ajusta casi exactamente a las guías de estilo propuestas.

B (suficiente). En el caso de software, conoce y utiliza de forma autónoma y correcta las herramientas, instrumentos y aplicativos software necesarios para el desarrollo de la práctica. No plantea correctamente el problema a partir del enunciado propuesto y/o no identifica las opciones para su resolución. No aplica el método de resolución adecuado y / o identifica la corrección de la solución, pero con errores. En el caso de la memoria, bien la estructura y / o la presentación son mejorables, el lenguaje presenta deficiencias y

[2] https://github.com/christopheradams/elixir_style_guide/blob/master/README.md

/ o el contenido no explica de forma precisa los conceptos importantes involucrados en la práctica. En el caso del código, este se ajusta a las guías de estilo propuestas, pero es mejorable.

B- (suficiente, con deficiencias). En el caso de software, conoce y utiliza de forma autónoma y correcta las herramientas, instrumentos y aplicativos software necesarios para el desarrollo de la práctica. No plantea correctamente el problema a partir del enunciado propuesto y/o no identifica las opciones para su resolución. No se aplica el método de resolución adecuado y/o se identifica la corrección de la solución, pero con errores de cierta gravedad y/o sin proporcionar una solución completa. En el caso de la memoria, bien la estructura y / o la presentación son manifiestamente mejorables, el lenguaje presenta serias deficiencias y / o el contenido no explica de forma precisa los conceptos importantes involucrados en la práctica. En el caso del código, hay que mejorarlo para que se ajuste a las guías de estilo propuestas.

C (deficiente). El software no compila o presenta errores graves. La memoria no presenta una estructura coherente y/o el lenguaje utilizado es pobre y/o contiene errores gramaticales y/o ortográficos. En el caso del código, este no se ajusta exactamente a las guías de estilo propuestas.

Calificación	**Solución al problema**	**Código**	**Memoria**
10	A+ (TF y matón)	A+	A+
9	A+ (TF y matón)	A	A
8	A (sin matón)	A	A
7	A (sin matón)	B	B
6	B	B	B
5	B-	B	B-
suspenso	C	C	C

Tabla 1. Rúbrica

5. Entrega

Deberéis entregar un fichero zip que contenga: (i) los fuentes desarrollados y (ii) la memoria en pdf (5 páginas). La entrega se realizará a través de moodle2 en la actividad habilitada a tal efecto. La fecha de entrega será no más tarde del día anterior al comienzo de la siguiente práctica. Además, será necesaria la defensa presencial de la práctica durante la sesión de la práctica 4.