

Distributed Systems

Name : Seila Moro Herrero

Group : N/A

1. Introduction

La estructura de la solución no se ha modificado y es la siguiente:

```
src → communication
    lsim → element → recipes_service
    recipes_service → activity_simulation
                     communication
                     data
                     test
    tsae → data_structures
          sessions
util
```

Se han implementado, dentro de “src\recipes_service”, los métodos de las siguientes clases:

- ServerData
 - removeRecipe(), messageOperationProcess()

Dentro de “src\recipes_service\data_structures” los métodos de las siguientes clases:

- Log
 - listNewer(), purgeLog()
- TimestampVector
 - TimestampVector(), updateMax(), getLast(), mergeMin(), clone(),
- TimestampMatrix
 - TimestampMatrix(), getTimestampVector(), updateMax(), update(), minTimestampVector(), clone(), equals()

Y dentro de “src\recipes_service\sessions” los métodos de las siguientes clases:

- TSAESessionOriginatorSide
 - sessionTSAE()
- TSAESessionPartnerSide
 - run()

2. Implementation

La función “sessionTSAE” de la clase “TSAESessionOriginatorSide” se encarga de iniciar las sesiones TSAE. Siguiendo el esquema indicado en el enunciado primero se obtiene el ACK y el SUMMARY del ServerData y se manda el mensaje de inicio de la sesión TSAE y a continuación recibe las operación del otro servidor de forma que se recibe un mensaje por cada operación. Por cada mensaje de este tipo se llama a la función “messageOperationProcess” del “ServerData”. Cuando se recibe el mensaje de final de sesión TSAE se envían al otro servidor las operaciones que aún no tiene. Se envía un mensaje por operación y al final del envío se le manda el mensaje de fin de sesión TSAE. Por último se actualiza el ACK y el SUMMARY, y se llama al purgar los Logs.

La función “run” de la clase “TSAESessionPartnerSide” cuando le llega un mensaje de inicio de sesión TSAE, siguiendo el esquema indicado en el enunciado primero e envían al otro servidor las operaciones que aún no tiene. Se envía un mensaje por operación y luego se obtiene el ACK y el

SUMMARY del ServerData para luego contestar con otro mensaje de inicio de sesión TSAE. A continuación recibe las operaciones del otro servidor, una operación por cada mensaje y en cada mensaje de este tipo llama a la función “messageOperationProcess” del “ServerData”. Cuando se recibe el mensaje de final de sesión TSAE contesta con otro mensaje de este tipo y por último se actualiza el ACK y el SUMMARY, y se llama al purgar los Logs.

La función “messageOperationProcess” de la clase “ServerData” se encarga de ejecutar y añadir al log las operaciones que llegan desde una sesión TSAE, tanto de añadir como de eliminar.

La función “removeRecipe” de la clase “ServerData” se encarga de eliminar el elemento indicado, para ello crea una operación de tipo “Remove” y añade la operación al log. Por último quita el elemento indicado de la lista.

La función “listNewer” de la clase “Log” recorre el registro para obtener las operaciones que no han sido vistas por el propietario del resumen. Para ello recorre el registro, y por cada clave compara la última marca de tiempo con la marca de tiempo de todas las operaciones. Si una operación tiene una marca de tiempo más reciente se añade a la lista de operaciones pendientes, que se devuelve como salida de la función.

La función “purgeLog” de la clase “Log” se encarga de borrar los log anteriores de un servidor y que ya no son necesarios. Para ello primero obtiene las marcas de tiempo conocidas por todos los servidores y a continuación recorre la lista de log de los servidores. Por cada servidor obtiene su mínima marca de tiempo y su lista de operaciones. Compara cada una de esas operaciones y si su marca de tiempo es menor o igual que la mínima marca de tiempo se añade a la lista de operaciones para eliminar. Por último se recorre la lista de operaciones a eliminar de forma inversa, para no perder la referencia de los índices de la lista, y se van eliminando las operaciones indicadas.

La función “TimestampMatrix” de la clase “TimestampMatrix” se utiliza para poder crear una instancia vacía de esta clase.

La función “updateMax” de la clase “TimestampMatrix” mezcla la matriz de dos marcas de tiempo de forma que queda el elemento máximo en cada posición de la matriz. Para ello, primero recorre la matriz recibida, en cada iteración busca el mismo servidor en la matriz de la clase y llama a la función “updateMax” de la clase “TimestampVector”.

La función “minTimestampVector” de la clase “TimestampMatrix” devuelve un vector de marca de tiempo que contenga, para cada servidor, la marca de tiempo conocida por todos los demás servidores. Para ello, recorre los servidores y para cada uno llama a la función “mergeMin” de la clase “TimestampVector”.

La función “updateMax” de la clase “TimestampVector” fusiona dos vectores tomando el máximo de cada elemento. Para ello, recorre todos los servidores del vector local y por cada iteración busca la última marca de tiempo del servidor en el vector a merguear y si esta es mayor que la última marca de tiempo actual del servidor se reemplaza su valor en el vector local.

La función “mergeMin” de la clase “TimestampVector” fusiona dos vectores tomando la marca de tiempo más pequeña por cada elemento. Para ello, recorre todos los servidores que recibe como parámetro, por cada iteración obtiene la última marca de tiempo del servidor. Si la marca de tiempo es nula se añade el nuevo servidor con la marca de tiempo recibida. Si no, se compararan las dos marcas de tiempo y si la recibida es menor, se reemplaza en el vector local.

Phase 4

4.1.1

El problema al implementar la funcionalidad de “remove recipe” es que, al intercambiar este tipo de operaciones entre dos servidores mediante una sesión TSAE, es posible que llegue la operación de eliminar antes de la operación de añadir. Lo cual provocaría que uno de los servidores tenga un recipiente más, que debería haber sido eliminado y no estén correctamente sincronizados.

3. Tests

Test N1

- Descripción: se prueba la “phase 2” de forma local.
- Objetivo: probar la solución del ejercicio de forma local, con operaciones de tipo “add” introducidas manualmente.
- Ejecución: se prueba usando el siguiente comando, desde la carpeta scripts:
“./start.sh 20004 3 --menu --nopurge”
Añadiendo dos recipientes en cada servidor de la prueba.
- Resultado:

```
Archivo Editar Ver Buscar Terminal Ayuda
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

groupXX@127.0.1.1:35001: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

#### [groupXX@127.0.1.1:35001] Result:
Group id: groupXX
Node id: groupXX@127.0.1.1:35001
Recipes: {a=[a, a, groupXX], b=[b, b, groupXX], c=[c, c, groupXX], d=[d, d, groupXX], e=[e, e, groupXX], f=[f, f, groupXX]}
Log: AddOperation [recipe=[e, e, groupXX], timestamp=groupXX@127.0.1.1:35002: 0]
AddOperation [recipe=[f, f, groupXX], timestamp=groupXX@127.0.1.1:35002: 1]
AddOperation [recipe=[a, a, groupXX], timestamp=groupXX@127.0.1.1:35000: 0]
AddOperation [recipe=[b, b, groupXX], timestamp=groupXX@127.0.1.1:35000: 1]
AddOperation [recipe=[c, c, groupXX], timestamp=groupXX@127.0.1.1:35001: 0]
AddOperation [recipe=[d, d, groupXX], timestamp=groupXX@127.0.1.1:35001: 1]

Summary: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

Ack: groupXX@127.0.1.1:35002: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

groupXX@127.0.1.1:35000: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

groupXX@127.0.1.1:35001: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

Results are equal      Nodes converged at the last iteration

=====

***** num received results: 2
***** % received results: 66
***** minimal required number of results: 2
```

- Conclusión: el resultado de la prueba es satisfactorio.

Test N2

- Descripción: se prueba la “phase 2” de forma local.
- Objetivo: probar la solución del ejercicio de forma local, con operaciones de tipo “add” predeterminadas, introducidas de forma automática.
- Ejecución: se prueba usando el siguiente comando, desde la carpeta scripts:
“./start.sh 20004 15 --logResults --nopurge --noremove”
- Resultado:

```
Archivo Editar Ver Buscar Terminal Ayuda
groupXX@127.0.1.1:135007: 1
groupXX@127.0.1.1:135005: 2

ServerResult --- equals: summaries are not equals
ServerResult --- ! equals -- summary: groupXX@127.0.1.1:135012: -1000
groupXX@127.0.1.1:135002: 0
groupXX@127.0.1.1:135014: 4
groupXX@127.0.1.1:135011: 1
groupXX@127.0.1.1:135008: 1
groupXX@127.0.1.1:135003: 4
groupXX@127.0.1.1:135000: 3
groupXX@127.0.1.1:135010: 4
groupXX@127.0.1.1:135013: 4
groupXX@127.0.1.1:135006: 2
groupXX@127.0.1.1:135001: 3
groupXX@127.0.1.1:135009: 2
groupXX@127.0.1.1:135004: 2
groupXX@127.0.1.1:135007: 2
groupXX@127.0.1.1:135005: 2

ServerResult --- ! equals -- summary2: groupXX@127.0.1.1:135012: -1000
groupXX@127.0.1.1:135002: 0
groupXX@127.0.1.1:135014: 4
groupXX@127.0.1.1:135011: 1
groupXX@127.0.1.1:135008: 1
groupXX@127.0.1.1:135003: 4
groupXX@127.0.1.1:135000: 3
groupXX@127.0.1.1:135010: 3
groupXX@127.0.1.1:135013: 4
groupXX@127.0.1.1:135006: 2
groupXX@127.0.1.1:135001: 3
groupXX@127.0.1.1:135009: 2
groupXX@127.0.1.1:135004: 2
groupXX@127.0.1.1:135007: 2
groupXX@127.0.1.1:135005: 2

Results are equal      Nodes converged at the iteration 1

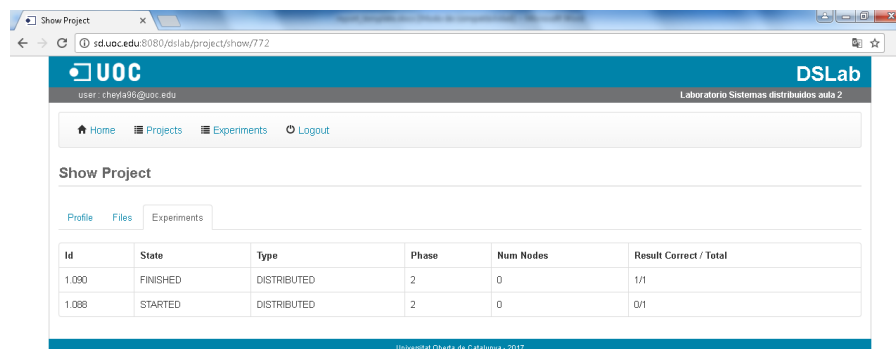
*****
***** num received results: 8
***** % received results: 53
***** minimal required number of results: 8
```

Los resultados se pueden ver en los ficheros “groupXX_phase2” y “groupXX_phase2.data”.

- Conclusión: el resultado de la prueba es satisfactorio.

Test N3

- Descripción: se prueba la “phase 2” mediante el DSLab.
- Objetivo: probar la solución del ejercicio en un entorno distribuido real.
- Ejecución: se crea el proyecto “SD_PHASE2_TEST_FINAL”, se añaden los ficheros correspondientes, se compila y ejecuta la prueba.
- Resultado:



The screenshot shows the DSLab web interface. At the top, there's a header with 'UOC' and 'DSLAb' logos, and a user profile 'chevaf96@uoc.edu'. Below the header, there's a navigation bar with 'Home', 'Projects', 'Experiments', and 'Logout'. The main content area is titled 'Show Project' and has tabs for 'Profile', 'Files', and 'Experiments'. The 'Experiments' tab is active, displaying a table with project details.

Id	State	Type	Phase	Num Nodes	Result Correct / Total
1.090	FINISHED	DISTRIBUTED	2	0	1/1
1.098	STARTED	DISTRIBUTED	2	0	0/1

Como se puede ver en la tabla, una de las pruebas se quedó bloqueada en estado “STARTED”.

- Conclusión: el resultado de la prueba es satisfactorio.

Test N4

- Descripción: se prueba la “phase 3” de forma local.
- Objetivo: probar la solución del ejercicio de forma local, con operaciones de tipo “add” introducidas manualmente.
- Ejecución: se prueba usando el siguiente comando, desde la carpeta scripts:
“./start.sh 20004 3 --logResults -path ../results --menu”
Añadiendo dos recipientes en cada servidor de la prueba.
- Resultado:

```
Archivo Editar Ver Buscar Terminal Ayuda
groupXX@127.0.1.1:35000: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

groupXX@127.0.1.1:35001: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

#### [groupXX@127.0.1.1:35001] Result:
Group id: groupXX
Node id: groupXX@127.0.1.1:35001
Recipes: {a=[a, a, groupXX], b=[b, b, groupXX], c=[c, c, groupXX], d=[d, d, groupXX], e=[e, e, groupXX], f=[f, f, groupXX]}
Log: AddOperation [recipe=[e, e, groupXX], timestamp=groupXX@127.0.1.1:35002: 0]
AddOperation [recipe=[f, f, groupXX], timestamp=groupXX@127.0.1.1:35002: 1]
AddOperation [recipe=[a, a, groupXX], timestamp=groupXX@127.0.1.1:35000: 0]
AddOperation [recipe=[b, b, groupXX], timestamp=groupXX@127.0.1.1:35000: 1]
AddOperation [recipe=[c, c, groupXX], timestamp=groupXX@127.0.1.1:35001: 0]
AddOperation [recipe=[d, d, groupXX], timestamp=groupXX@127.0.1.1:35001: 1]

Summary: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

Ack: groupXX@127.0.1.1:35002: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

groupXX@127.0.1.1:35000: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

groupXX@127.0.1.1:35001: groupXX@127.0.1.1:35002: 1
groupXX@127.0.1.1:35000: 1
groupXX@127.0.1.1:35001: 1

Results are equal      Nodes converged at the last iteration

*****

***** num received results: 2
***** % received results: 66
***** minimal required number of results: 2
```

Los resultados se pueden ver en los ficheros “groupXX_phase3_t1” y “groupXX_phase3_t1.data”.

- Conclusión: el resultado de la prueba es satisfactorio.

Test N5

- Descripción: se prueba la “phase 3” de forma local.
- Objetivo: probar la solución del ejercicio de forma local, con operaciones de tipo “add” predeterminadas, introducidas de forma automática.
- Ejecución: se prueba usando el siguiente comando, desde la carpeta scripts:
“./start.sh 20004 15 --logResults -path ./results --noremove”
Añadiendo dos recipientes en cada servidor de la prueba.
- Resultado:

```
Archivo Editar Ver Buscar Terminal Ayuda
groupXX@127.0.1.1:35005: 1

ServerResult --- equals: summaries are not equals
ServerResult --- ! equals -- summary: groupXX@127.0.1.1:35012: 2
groupXX@127.0.1.1:35002: 0
groupXX@127.0.1.1:35014: 2
groupXX@127.0.1.1:35011: 3
groupXX@127.0.1.1:35008: 5
groupXX@127.0.1.1:35003: 1
groupXX@127.0.1.1:35000: 4
groupXX@127.0.1.1:35010: 4
groupXX@127.0.1.1:35013: -1000
groupXX@127.0.1.1:35006: 3
groupXX@127.0.1.1:35001: 0
groupXX@127.0.1.1:35009: 3
groupXX@127.0.1.1:35004: 0
groupXX@127.0.1.1:35007: 3
groupXX@127.0.1.1:35005: 1

ServerResult --- ! equals -- summary2: groupXX@127.0.1.1:35012: 2
groupXX@127.0.1.1:35002: 0
groupXX@127.0.1.1:35014: 2
groupXX@127.0.1.1:35011: 3
groupXX@127.0.1.1:35008: 5
groupXX@127.0.1.1:35003: 1
groupXX@127.0.1.1:35000: 4
groupXX@127.0.1.1:35010: 4
groupXX@127.0.1.1:35013: 0
groupXX@127.0.1.1:35006: 3
groupXX@127.0.1.1:35001: 0
groupXX@127.0.1.1:35009: 3
groupXX@127.0.1.1:35004: 0
groupXX@127.0.1.1:35007: 3
groupXX@127.0.1.1:35005: 1

Results are equal      Nodes converged at the iteration 1

=====

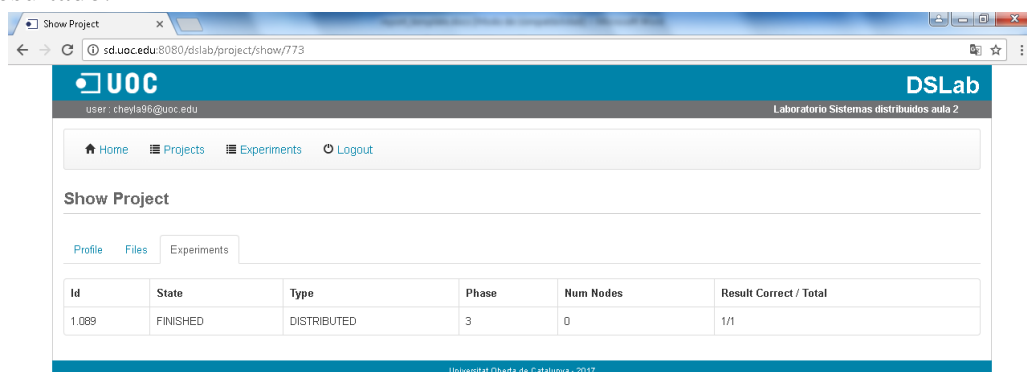
***** num received results: 8
***** % received results: 53
***** minimal required number of results: 8
```

Los resultados se pueden ver en los ficheros “groupXX_phase3_t2” y “groupXX_phase3_t2.data”.

- Conclusión: el resultado de la prueba es satisfactorio.

Test N5

- Descripción: se prueba la “phase 3” mediante el DSLab.
- Objetivo: probar la solución del ejercicio en un entorno distribuido real.
- Ejecución: se crea el proyecto “SD_PHASE3_TEST_FINAL”, se añaden los ficheros correspondientes, se compila y ejecuta la prueba.
- Resultado:



- Conclusión: el resultado de la prueba es satisfactorio.

Test N6

- Descripción: se prueba la “phase 4.1” de forma local.
- Objetivo: probar la solución del ejercicio de forma local, con operaciones de tipo “add” predeterminadas, introducidas de forma automática.
- Ejecución: se prueba usando el siguiente comando, desde la carpeta scripts:
“./start.sh 20004 15 --logResults -path ../results”
- Resultado:

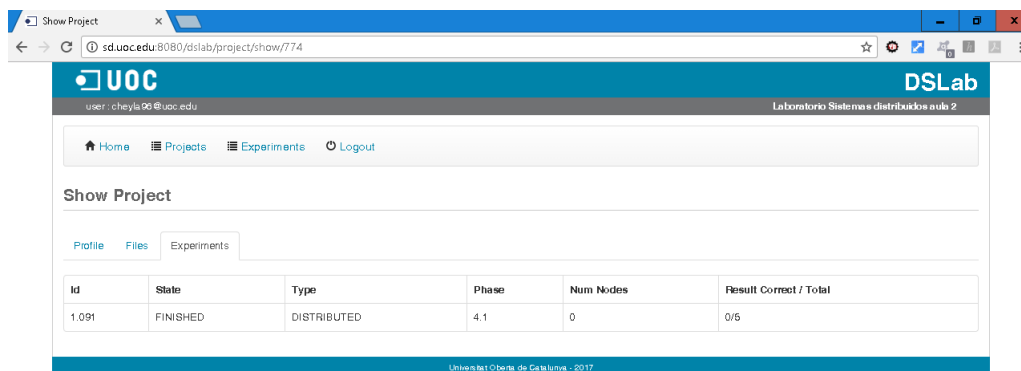
[illegible]

Los resultados se pueden ver en los ficheros “groupXX phase4 1” y “groupXX_phase4_1.data”.

- **Conclusión:** el resultado de la prueba es satisfactorio.

Test N7

- Descripción: se prueba la “phase 4.1” mediante el DSLab.
- Objetivo: probar la solución del ejercicio en un entorno distribuido real.
- Ejecución: se crea el proyecto “SD_PHASE4_TEST_FINAL”, se añaden los ficheros correspondientes, se compila y ejecuta la prueba.
- Resultado:



- **Conclusión:** el resultado de la prueba es erróneo. No purga bien los log cuando existe una operación de tipo “remove”.

4. Conclusions

El protocolo TSAE es muy complejo tanto de entender como de implementar, es necesario muchos métodos, clases y variables para poder implementarlo correctamente. La mayor dificultad en la comparación de las marcas de tiempo para purgar los log que ya no son necesario.

La operación de eliminar recipiente, que parecía muy sencilla, resultó dar muchos problemas. Ya que hay que tener muchos casos en cuenta. Hasta el punto que no fui capaz de purgar los log cuando llega una operación de este tipo.

Para realizar unas pruebas realistas de este protocolo es necesario disponer de varios equipos en red. Ya que, de forma local no da problemas con las marcas de tiempo.