Time Stamped Anti-Entropy protocol

Practice 1

Universitat Oberta de Catalunya

Relevant information:

- Deadline for delivery: 14/04/2025
- PR grade percentage: 40%

Authors: Joan Manuel Marquès, Antonio González, David Mor España, Montserrat Rovira Marco, Elohim Cabezas

Distributed Systems course

Spring 2025

Content

The use of artificial intelligence tools is not permitted in this activity.

In the teaching plan and on the UOC's website on academic integrity and plagiarism you will find information on what is considered irregular conduct in assessment and the consequences it may have.

Teaching details

Assignment Overview

The aim of this practical assignment is to implement and evaluate a weak-consistency protocol for data dissemination. The project involves:

- Implementing the Time Stamped Anti-Entropy (TSAE) protocol [1] into an application that stores cooking recipes across a set of replicated servers.
- Adding a remove operation to the recipes application.
- Evaluating how TSAE behaves under different conditions.

Time Stamped Anti-Entropy (TSAE) protocol

The **Time Stamped Anti-Entropy (TSAE)** [1] protocol is a weak-consistency protocol that ensures reliable and eventual delivery of issued operations.

To gain an overview and the main ideas about TSAE, read from [2]:

- Section 1. Introduction (excluding subsection 1.1).
- Section 2.2. Timestamped anti-entropy (also interesting to read: 2.1. Kinds of consistency).

For detailed information about the protocol, read **Chapter 5. Weak-consistency communication** from [1]. Specifically, focus on:

- 5.1.1. Data structures for timestamped anti-entropy.
- 5.1.2. The timestamped anti-entropy protocol.
- (If you implement purge) **5.3. Purging the message log**. Instead of using vector acks (loosely-synchronized clocks) as described in this section, you should use unsynchronized clocks, as outlined in section 5.4.4. **Anti-entropy with unsynchronized clocks.**

Practice statement

Practice 1 includes three theoretical exercises and a practical component, which involves implementing the methods for the Log and TimestampVector data structures.

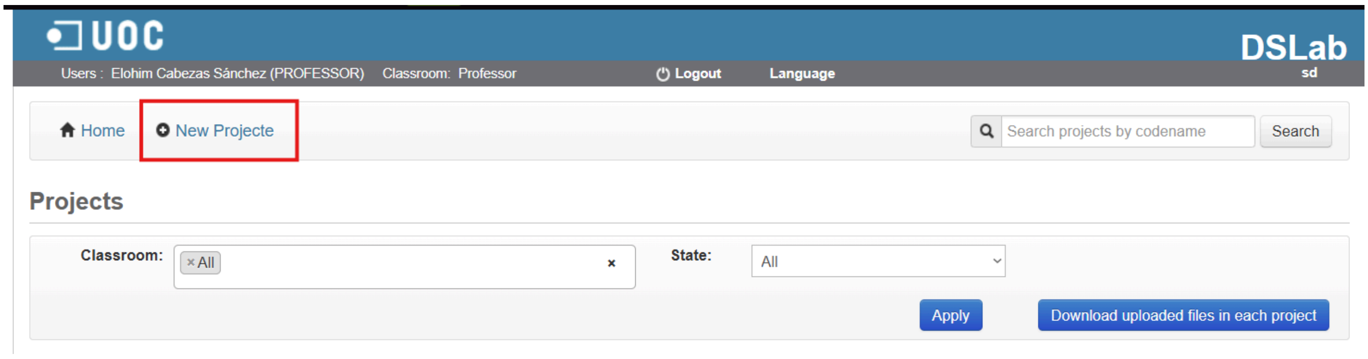
Since the activities are interconnected (i.e., completing one requires an understanding of the previous), **it is strongly recommended to complete the tasks and exercises in the order presented in this document.**

1 - Theoretical exercise of TSAE protocol

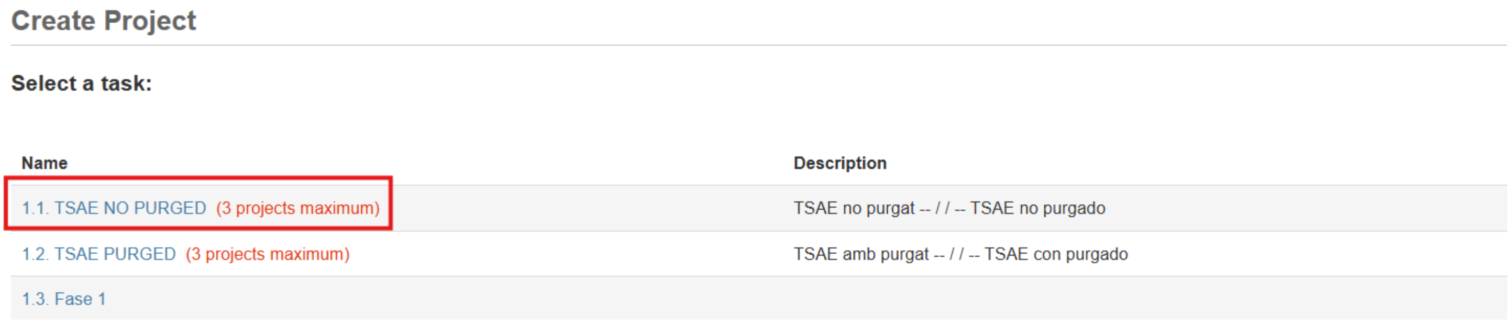
Exercice 1 - TSAE protocol exercise (no purged log)

Visit https://sd.uoc.edu/dslab/ and create a new project to access the exercise statement. Each student will receive a unique and distinct exercise. Follow these steps to generate the statement:

- Click the "New Project" button:


UOC
DSLab
sd
Users : Elohim Cabezas Sánchez (PROFESSOR) Classroom: Professor
Logout
Language
Home
New Projecte
Search projects by codename
Search
Projects
Classroom:
× All
State:
All
Apply
Download uploaded files in each project

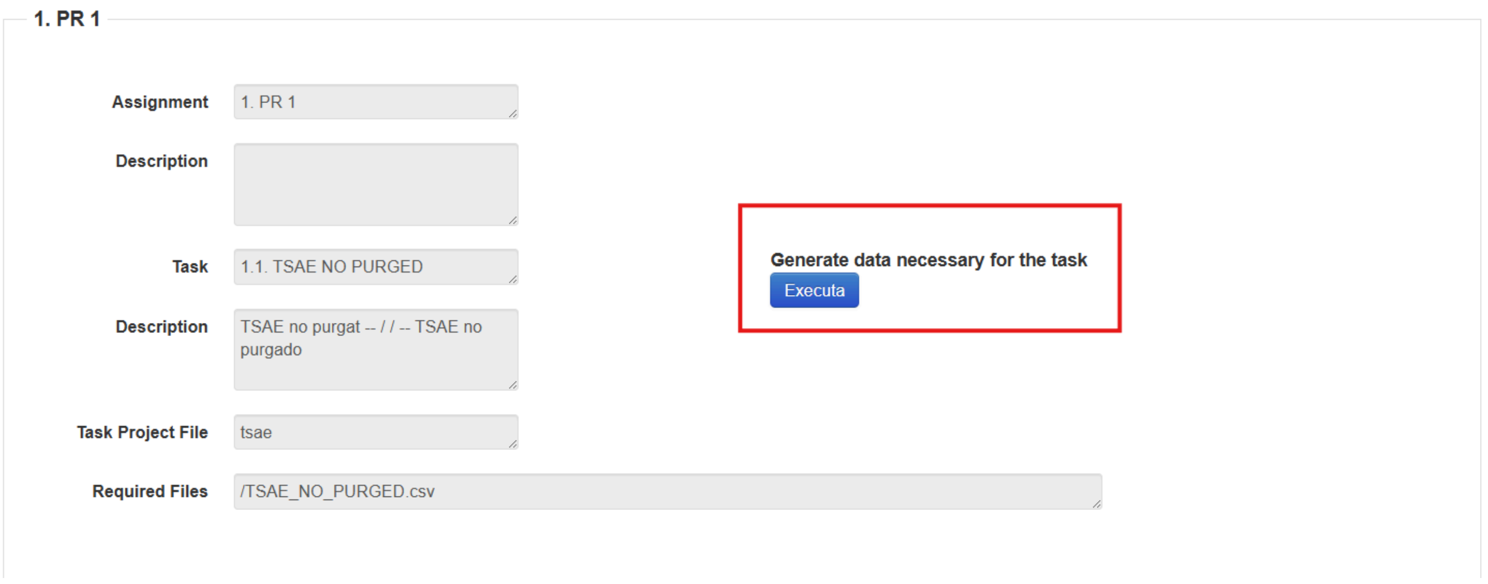
- Select 1.1 TSAE NO PURGED project:


Create Project

Select a task:

Name	Description
1.1. TSAE NO PURGED (3 projects maximum)	TSAE no purgat -- / / -- TSAE no purgado
1.2. TSAE PURGED (3 projects maximum)	TSAE amb purgat -- / / -- TSAE con purgado
1.3. Fase 1	

- Click the button below:

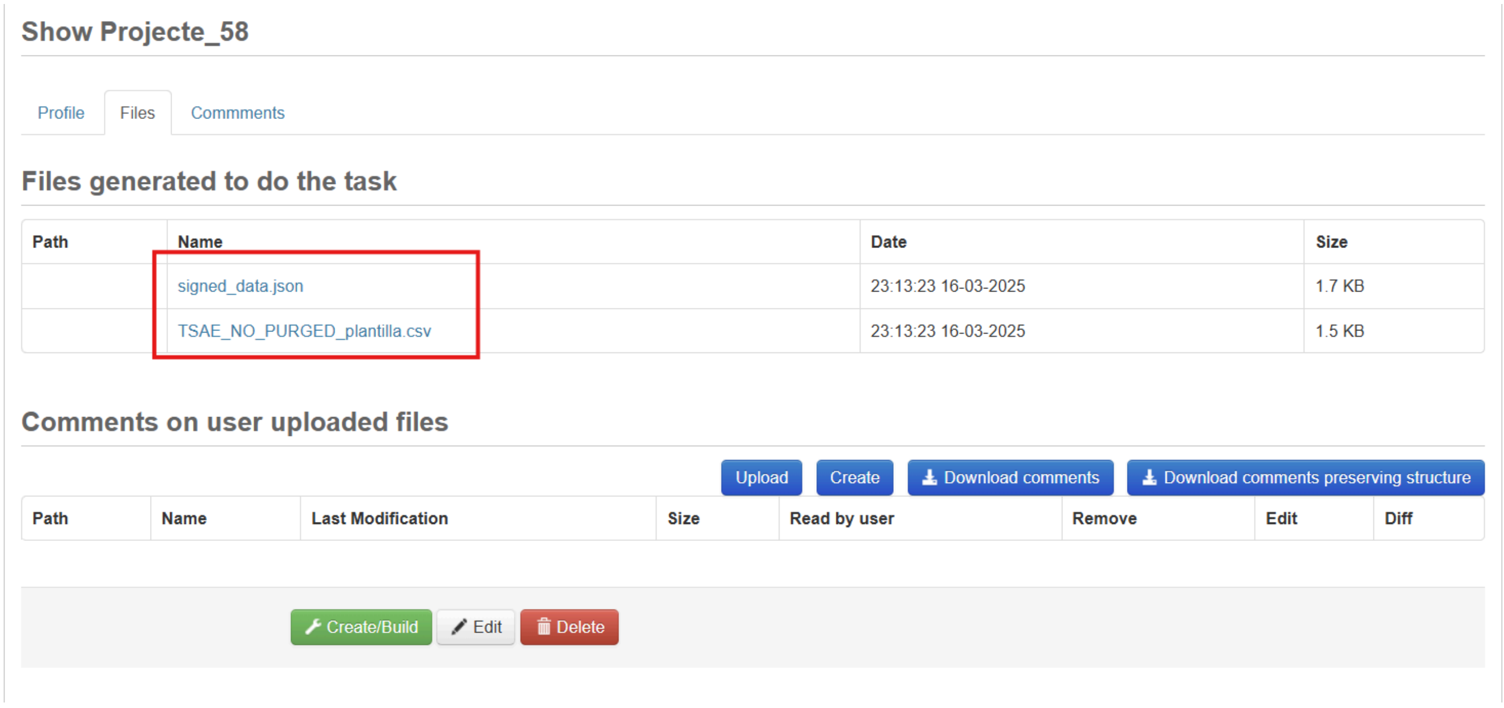


1. PR 1

Assignment	1. PR 1
Description	
Task	1.1. TSAE NO PURGED
Description	TSAE no purgat -- / / -- TSAE no purgado
Task Project File	tsae
Required Files	/TSAE_NO_PURGED.csv

Generate data necessary for the task

- The action above will generate two files. The exercise statement can be found in the file named "signed_data.json," and the template you need to work with is located in "TSAE_NO_PURGED_template.csv."



Show Projecte_58

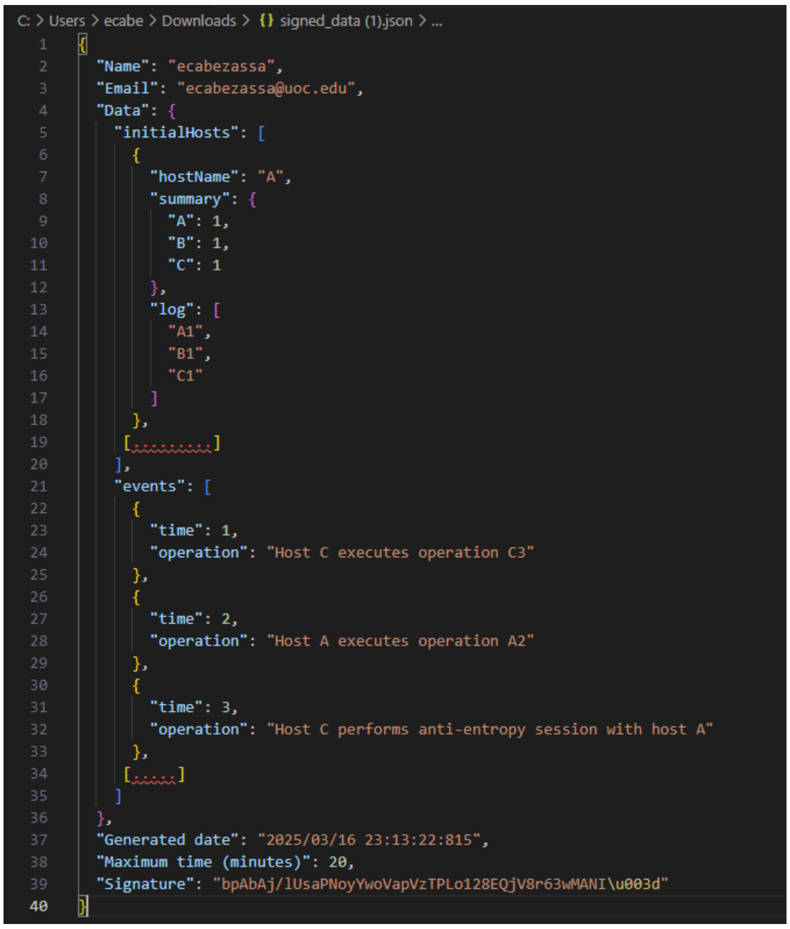
Files generated to do the task

Path	Name	Date	Size
	signed_data.json	23:13:23 16-03-2025	1.7 KB
	TSAE_NO_PURGED_plantilla.csv	23:13:23 16-03-2025	1.5 KB

Comments on user uploaded files

Path	Name	Last Modification	Size	Read by user	Remove	Edit	Diff

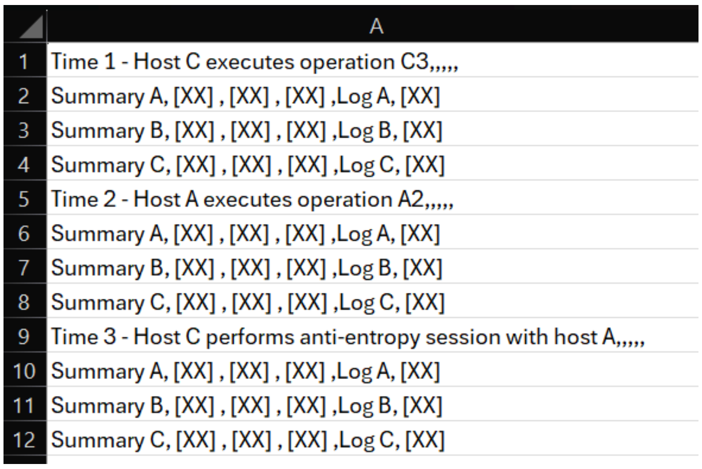
- In the "signed_data.json" file, your initial data can be found in the "Data" field, and the operations to be performed are listed in the "events" field.



C: > Users > ecabe > Downloads > {} signed_data (1).json > ...

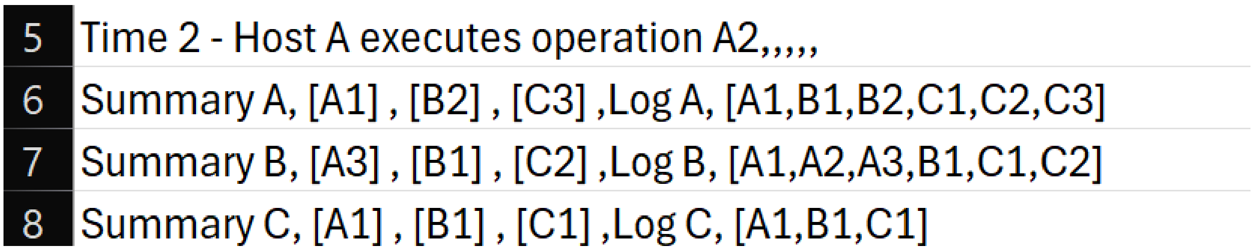
```
{
  "Name": "ecabezassa",
  "Email": "ecabezassa@uoc.edu",
  "Data": {
    "initialHosts": [
      {
        "hostName": "A",
        "summary": {
          "A": 1,
          "B": 1,
          "C": 1
        },
        "log": [
          "A1",
          "B1",
          "C1"
        ]
      },
      [..........]
    ],
    "events": [
      {
        "time": 1,
        "operation": "Host C executes operation C3"
      },
      {
        "time": 2,
        "operation": "Host A executes operation A2"
      },
      {
        "time": 3,
        "operation": "Host C performs anti-entropy session with host A"
      },
      [.....]
    ]
  },
  "Generated date": "2025/03/16 23:13:22:815",
  "Maximum time (minutes)": 20,
  "Signature": "bpAbAj/lUsaPNoyYwoVapVzTPLo128EQjV8r63wMANI\u003d"
}
```

- The solution to this exercise must be completed in the file "TSAE_NO_PURGED_plantilla.csv":



	A
1	Time 1 - Host C executes operation C3,,,,,
2	Summary A, [XX] , [XX] , [XX] ,Log A, [XX]
3	Summary B, [XX] , [XX] , [XX] ,Log B, [XX]
4	Summary C, [XX] , [XX] , [XX] ,Log C, [XX]
5	Time 2 - Host A executes operation A2,,,,,
6	Summary A, [XX] , [XX] , [XX] ,Log A, [XX]
7	Summary B, [XX] , [XX] , [XX] ,Log B, [XX]
8	Summary C, [XX] , [XX] , [XX] ,Log C, [XX]
9	Time 3 - Host C performs anti-entropy session with host A,,,,,
10	Summary A, [XX] , [XX] , [XX] ,Log A, [XX]
11	Summary B, [XX] , [XX] , [XX] ,Log B, [XX]
12	Summary C, [XX] , [XX] , [XX] ,Log C, [XX]

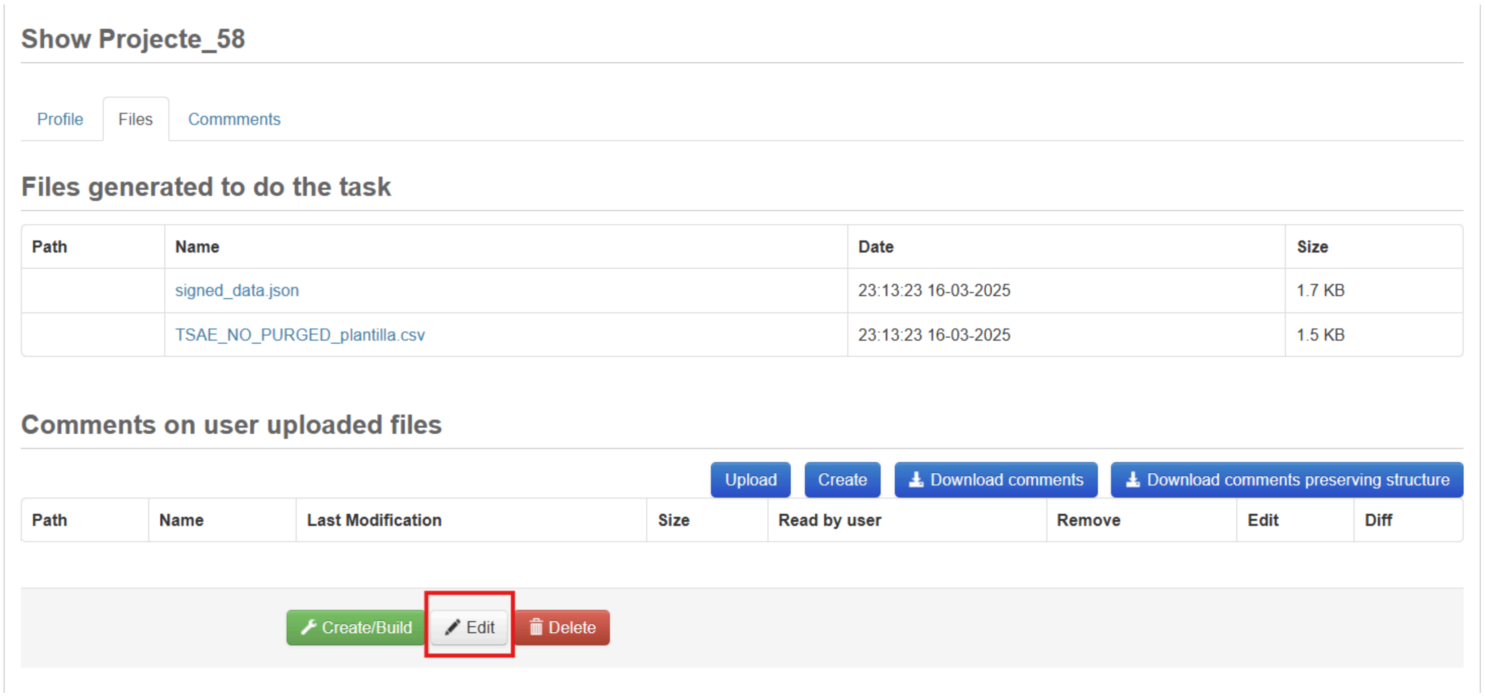
- You must enter your answers in the fields labeled [XX]. An example is provided below:



5	Time 2 - Host A executes operation A2,,,,,
6	Summary A, [A1] , [B2] , [C3] ,Log A, [A1,B1,B2,C1,C2,C3]
7	Summary B, [A3] , [B1] , [C2] ,Log B, [A1,A2,A3,B1,C1,C2]
8	Summary C, [A1] , [B1] , [C1] ,Log C, [A1,B1,C1]

Do not leave any blank spaces between operations, as the system will mark it as incorrect.
This is correct: [A1,A2,A3]
This is incorrect: [A1, A2, A3]

- Once you have completed the exercise, click the "Edit" button and upload the template file named "TSAE_NO_PURGED.csv":



Show Projecte_58

Profile | Files | Commments

Files generated to do the task

Path	Name	Date	Size
	signed_data.json	23:13:23 16-03-2025	1.7 KB
	TSAE_NO_PURGED_plantilla.csv	23:13:23 16-03-2025	1.5 KB

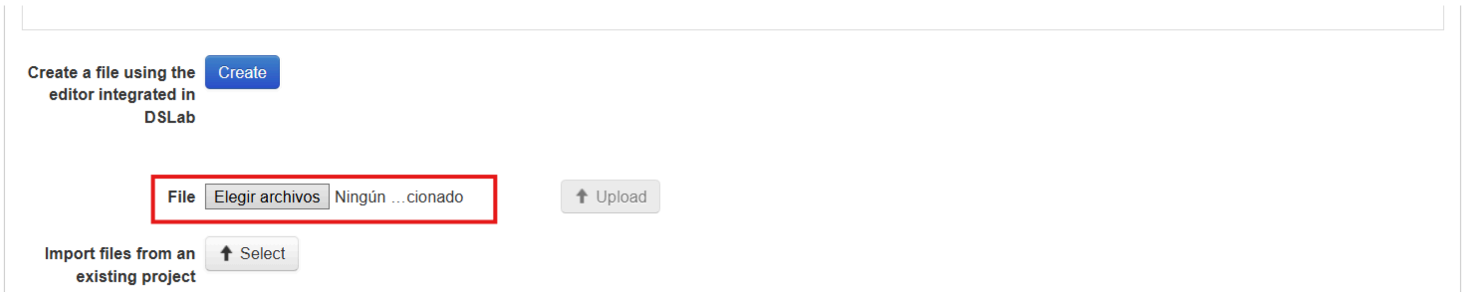
Comments on user uploaded files

Upload | Create | Download comments | Download comments preserving structure

Path	Name	Last Modification	Size	Read by user	Remove	Edit	Diff

Create/Build | Edit | Delete

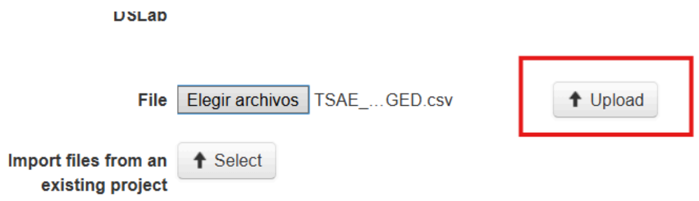
- Upload your file:


Create a file using the editor integrated in DSLab

File

Import files from an existing project

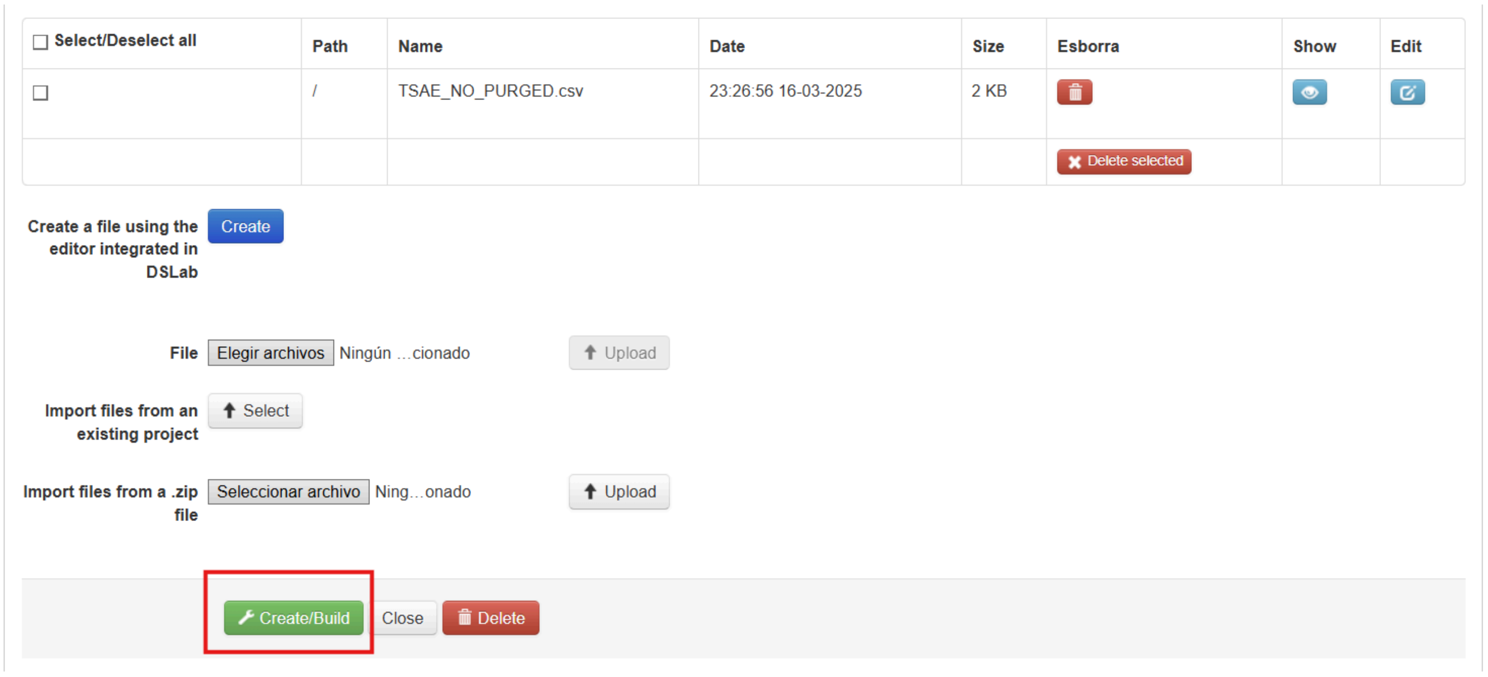
- Important: click the “Upload” button:


DSLab

File

Import files from an existing project

- And once uploaded, click the "Create/Build" button:



☐ Select/Deselect all	Path	Name	Date	Size	Esborra	Show	Edit
☐	/	TSAE_NO_PURGED.csv	23:26:56 16-03-2025	2 KB	🗑	👁	✎
					Delete selected		

Create a file using the editor integrated in DSLab

File

Elegir archivos Ningún ...cionado
Upload

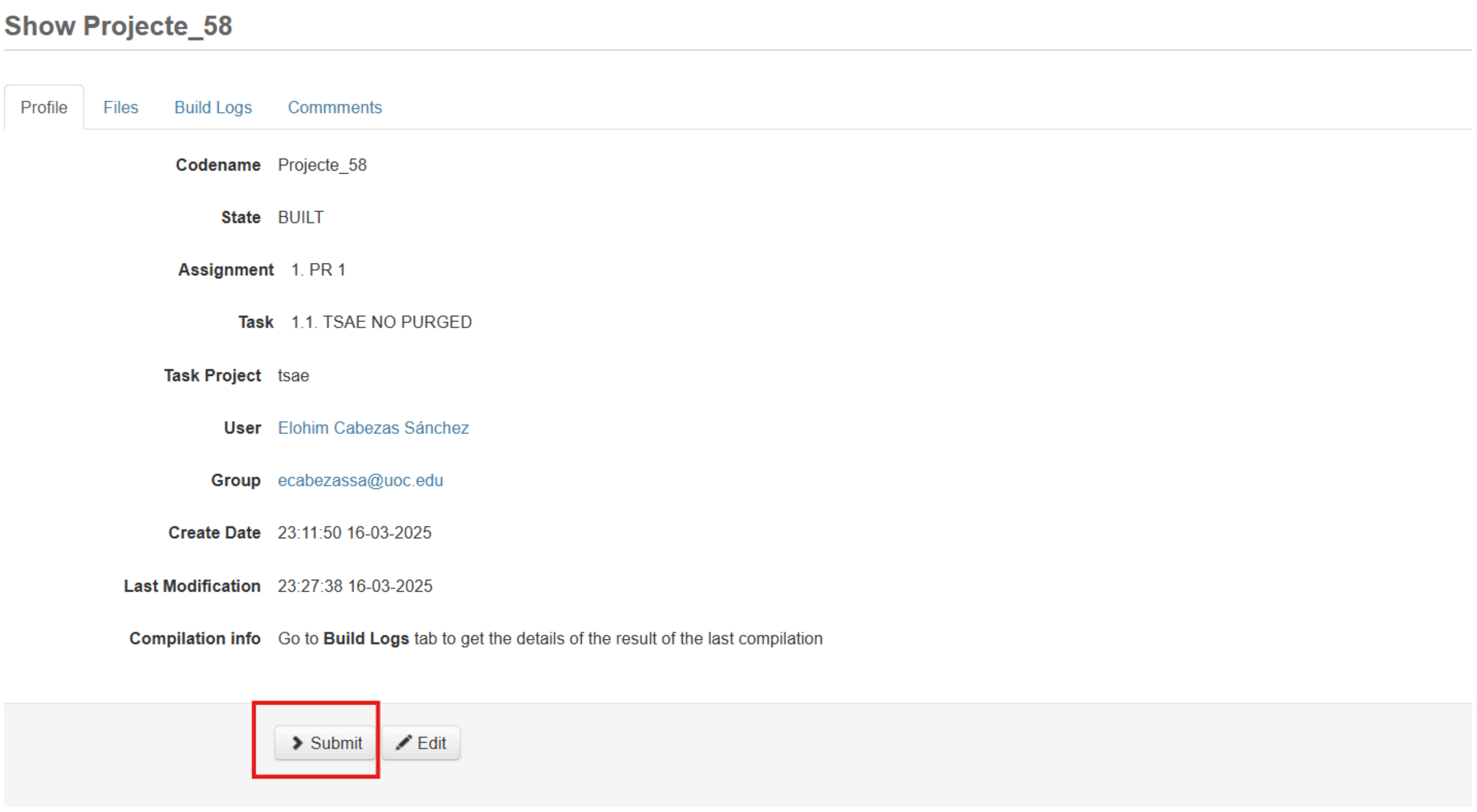
Import files from an existing project

Import files from a .zip file

Seleccionar archivo Ning...onado
Upload

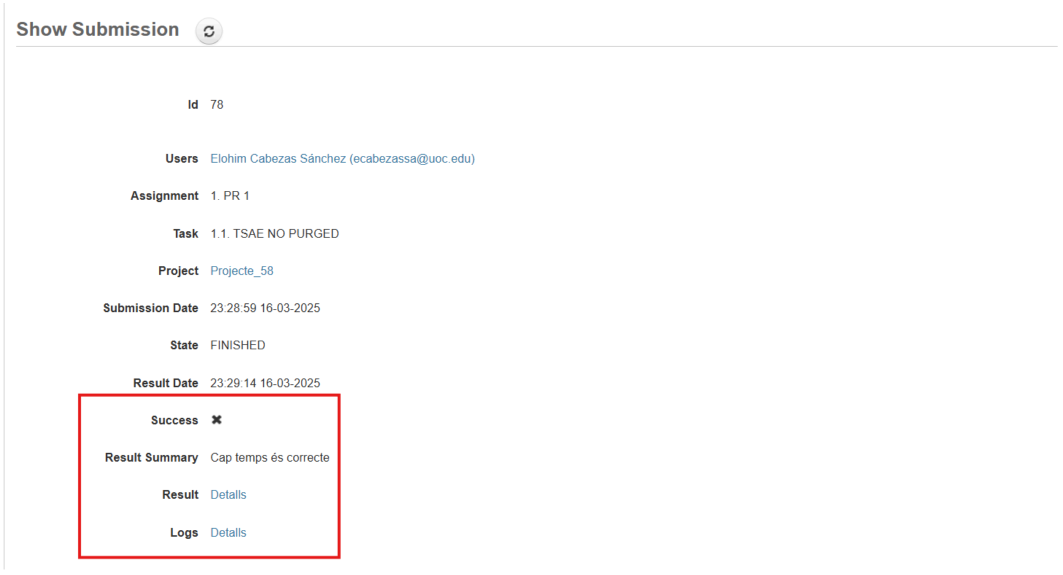
Create/Build
Close
Delete

- You can edit the project again if necessary. When you're ready, click the "Submit" button.



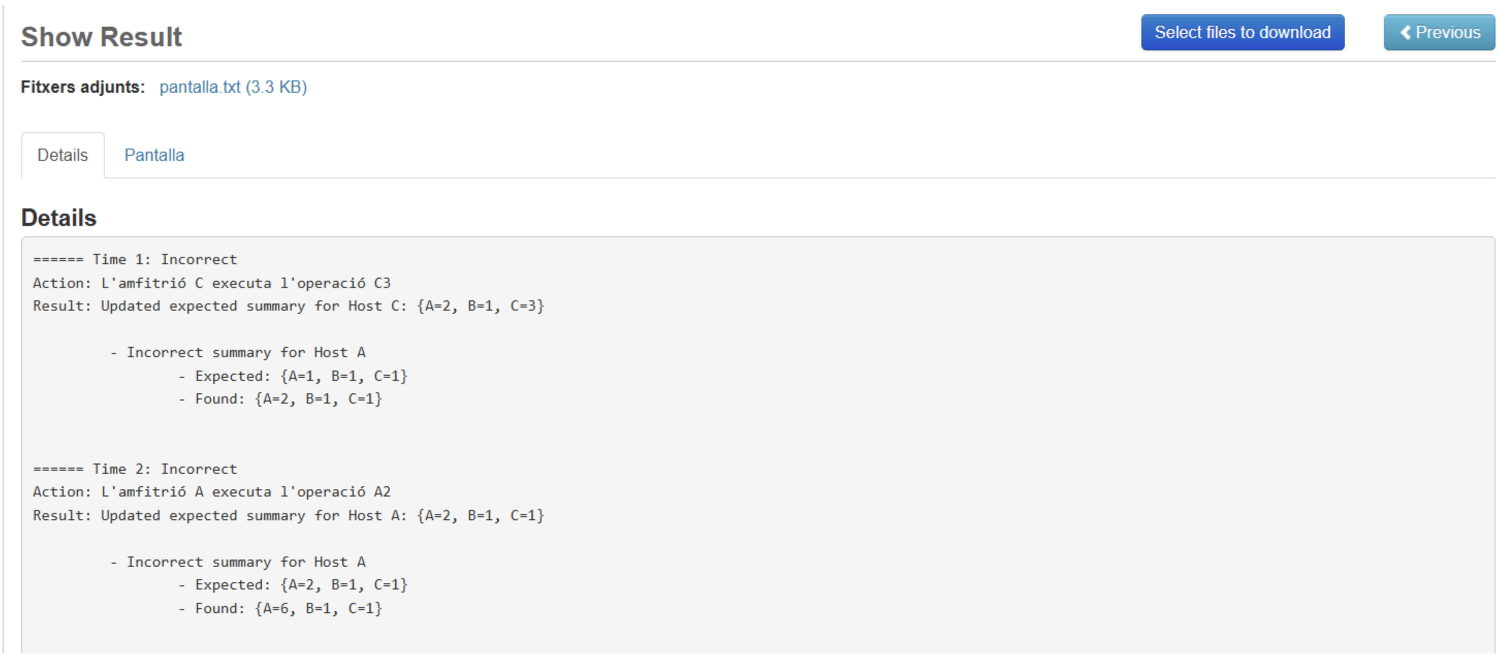
Codename	Projecte_58
State	BUILT
Assignment	1. PR 1
Task	1.1. TSAE NO PURGED
Task Project	tsae
User	Elohim Cabezas Sánchez
Group	ecabezassa@uoc.edu
Create Date	23:11:50 16-03-2025
Last Modification	23:27:38 16-03-2025
Compilation info	Go to **Build Logs** tab to get the details of the result of the last compilation

- The exercise is being evaluated, and the result will be available in a few seconds. You can view the result in the following options:



Id	78
Users	Elohim Cabezas Sánchez (ecabezassa@uoc.edu)
Assignment	1. PR 1
Task	1.1. TSAE NO PURGED
Project	Projecte_58
Submission Date	23:28:59 16-03-2025
State	FINISHED
Result Date	23:29:14 16-03-2025
Success	✖
Result Summary	Cap temps és correcte
Result	Details
Logs	Details

- If you click on the "Details" link under the "Result" option, you can view the outcome of your execution:

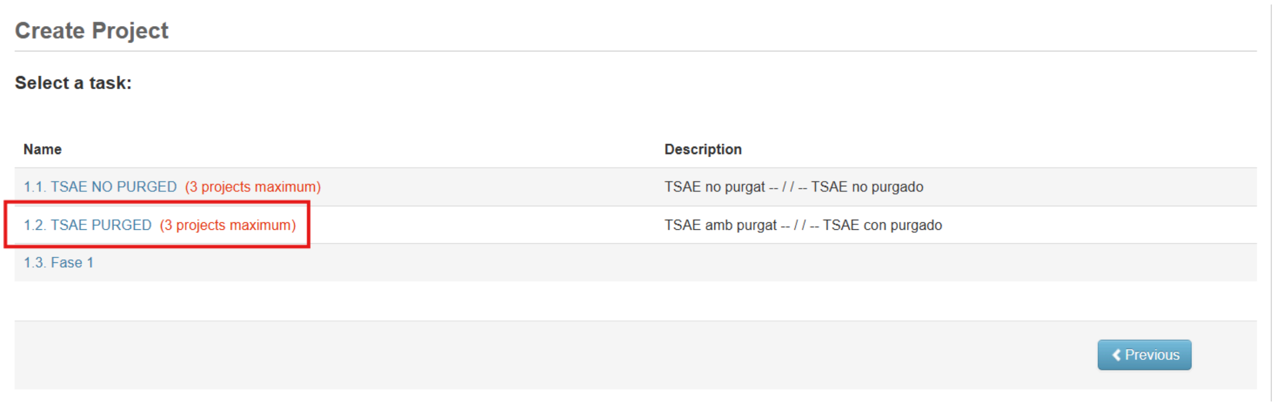


```
====== Time 1: Incorrect
Action: L'amfitrió C executa l'operació C3
Result: Updated expected summary for Host C: {A=2, B=1, C=3}

         - Incorrect summary for Host A
                 - Expected: {A=1, B=1, C=1}
                 - Found: {A=2, B=1, C=1}


====== Time 2: Incorrect
Action: L'amfitrió A executa l'operació A2
Result: Updated expected summary for Host A: {A=2, B=1, C=1}

         - Incorrect summary for Host A
                 - Expected: {A=2, B=1, C=1}
                 - Found: {A=6, B=1, C=1}
```

- The exercise will only be assessed if it is completed in DSLAB.
- You have a maximum of 3 attempts per exercise.
- Each time a new project is created, the data will be different.
- Each exercise must be completed within a maximum time of 20 minutes. After this time, the system will not grade the exercise and will mark it as incorrect.

Exercice 2 - TSAE protocol exercise (purged log)

Follow the same instructions as the previous exercise, except this time, select option 1.2. TSAE PURGED:



Create Project

Select a task:

Name	Description
1.1. TSAE NO PURGED (3 projects maximum)	TSAE no purgat -- / / -- TSAE no purgado
1.2. TSAE PURGED (3 projects maximum)	TSAE amb purgat -- / / -- TSAE con purgado
1.3. Fase 1	

Exercice 3 - TSAE protocol exercise

1. What data structures are used in the TSAE protocol, and how do they help in maintaining the consistency of replicas in a distributed system?

2. How does the TSAE protocol work to ensure reliable communication in a weak-consistency system, and what role does the exchange of vector clocks play in this process?

3. Why is purging the message log important in the TSAE protocol, and how does it affect the system's performance and consistency?

4. What are the key advantages of using weak-consistency replication protocols, and how do they help address challenges in large-scale distributed systems?

5. How do weak-consistency replication protocols handle network failures and replica disconnections, and what impact does this have on data consistency?

2 - Practical exercise of TSAE protocol

Phase 1: Implementation and testing of Log and TimestampVector data structures

Phase 1 will involve implementing the methods for the Log and TimestampVector data structures.

In this phase only add operations are issued. Don't implement the functionality to purge the log.

Annex A provides details about the timestamps used in this practical assignment.

Be cautious with concurrent access to data structures. Actions issued by different threads may interleave. Use a synchronization mechanism to prevent interference.

Implement the Log and TimestampVector data structures according to the TSAE protocol outlined in the "Time Stamped Anti-Entropy (TSAE)" section.

2.1 Environment

Requires Java 17.

We recommend using Eclipse as your IDE. We will provide an Eclipse project that includes the implementation of the cooking recipes application, except for the parts related to the TSAE protocol.

All scripts for running local tests are prepared for Ubuntu Linux, but other operating systems can be used. In that case, you will be responsible for adapting the scripts to your OS.

2.2 Test locally

To test the Log and TimestampVector data structures locally, run:

```
java -cp ../bin:../lib/* recipes_service.Server --phase1
(run it from the scripts folder)
```

Introduce a recipe and verify if the Log and TimestampVector data structures are correct.

```
./start.sh 20004 --phase1
(run it from the scripts folder)
```

$1: Listening port of Phase1TestServer, the server that tests the correctness of your solution. If port 20004 is already in use by another application, you can change it to any other unused port.

This command executes Log and TimestampVector with a predefined set of users and operations, and compares it with a Log and TimestampVector that was previously calculated.

2.3 Formal evaluation of phase 1

It is compulsory to use DSLab to assess Phase 1 of the practical assignment. Therefore, once your application is running locally, upload the Log and TimestampVector classes to DSLab (https://sd.uoc.edu/dslab/) and assess them:

1. Create a Project and upload Log and TimestampVector classes into this project.
2. Build and run the project created in step 1.
3. Check the result of the execution of step 2.

2.4 Deliverables

A file following the naming convention:

PR1-2025p-Surname-Name.zip

The zip file should contain a single directory named after the zip file, with the following structure and folder/file names:

- **/doc**:
  - **PR1_theoretical_exercises.pdf**: Solution to the three theoretical exercises.
  - **phase1.pdf**: A short report detailing the reasoning behind all decisions made. Additionally, you should highlight portions of your source code that you consider significant for this purpose.
- **/src**:
  - Source code: Log and TimestampVector classes (**Log.java** i **TimestampVector.java** que es troben a /src/recipes_service/tsae/data_structures/).

Annex A. Source code and documentation

The Papers folder contains the referenced papers and thesis that explain the TSAE protocol.

The TSAE folder contains all the packages and classes required for the practical assignment.

Important: Do not implement new classes. Modify only the classes specified in each phase (except if you modify the basic modeling of parameters in Phase 4).

Classes that you should modify

1. package recipes_service.tsae.datastructures:

- Log: A class that logs operations.
- TimestampVector: A class that maintains the summary.
- TimestampMatrix: A class that maintains the acknowledgment matrix of timestamps.

2. package recipes_service.tsae.sessions:

- TSAESessionOriginatorSide: The originator protocol for TSAE.
- TSAESessionPartnerSide: The partner protocol for TSAE.

3. package recipes_service:

- ServerData: Contains the server's data structures required by the TSAE protocol (log, summary, ack) and the application (recipes). You can add any necessary methods to allow TSAESessionOriginatorSide and TSAESessionPartnerSide to manipulate these data structures.

  ■ addRecipe method: adds a new recipe.
  ■ removeRecipe method: removes a recipe.

Classes that you should use but NOT modify

4. package recipes_service.tsae.data_structures:

- Timestamp: A timestamp allows the ordering of operations issued from the same host. It is a tuple <hostId, sequenceNumber>. The sequence number grows monotonically, starting with 0 for the first valid timestamp issued by a host. A negative sequence number indicates that the host has not issued any operation yet. Timestamps cannot be used to order operations issued from different hosts. The next timestamp can be obtained by calling the method nextTimestamp() from the ServerData class (package recipes_service).

5. package recipes_service.data:

- AddOperation: An add operation (operations are logged in the Log and exchanged with other partners).
- RemoveOperation: A remove operation (operations are logged in the Log and exchanged with other partners).
- Recipe: A recipe.
- Recipes: A class that contains all recipes.

6. package recipes_service.communication:

- MessageAErequest: A message sent to request an anti-entropy session.
- MessageOperation: A message sent for each operation exchanged during an anti-entropy session.
- MessageEndTSAE: A message sent to finish an anti-entropy session.

7. package communication:

- ObjectInputStream_DS: A class that implements a modification of the ObjectInputStream to simulate failures. Use the method readObject().
- ObjectOutputStream_DS: A class that implements a modification of the ObjectOutputStream to simulate failures. Use the method writeObject().

8. package recipes_service.activitysimulation:

Only for Phase 4 if you want to better understand or modify the modeling of activity and dynamicity.

Annex B. Activity simulation and dynamicity

Simulation of communication failures

The ActivitySimulation class (package recipes_service.activitysimulation) determines when to simulate the disconnection (or network failure) of a host and when to reconnect it.

To simulate communication failures, a disconnected host abruptly closes all its Input and Output streams, resulting in an exception for the two partners using those streams.

References

[1] Richard A. Golding (1992, December). Weak-consistency group communication and membership. Ph.D. thesis, published as technical report UCSC-CRL-92-52. Computer and Information Sciences Board, University of California. Santa Cruz. (chapter 5)

[2] R. Golding; D. D. E. Long. The performance of weak-consistency replication protocols, UCSC Technical Report UCSC-CRL-92-30, July 1992.