Option 1: Design your own whiteboarding protocols

Online collaborative whiteboard applications, such as Miro, InVision Freehand, and IPEVO Annotator, have been widely used for teamwork, especially during the pandemic. The existing whiteboard applications often use proprietary protocols, except a few using standard messaging protocols like XMPP (extensible messaging and presence protocol).

Requirements: You are supposed to design your own protocols for a real-time interactive whiteboard app. For evaluation, you need to implement a prototype of the whiteboard app using the protocols, and test its performance.

The whiteboard application is supposed to

- allow any user to create a new session and invite others to join the session. The user who has created the session would serve as the host and can accept or decline the requests for joining the session. The session would end when the host leaves or ends the session.
- allow for freehand drawing and support the Erase and Undo functions.
- allow multiple users (more than 2 users) to draw on the same board at the same time.
- provide a consistent (shared) view for all the users in the same session.
- allow users to add, edit, or remove sticky notes.
- allow users to upload images and to comment on the images. Comments can be texts attached to the images, or drawing on the images.
- allow users to save the whiteboard as a JPEG or PNG file before ending the session

The protocols are expected to be light-weight and secured.

Guideline:

In the first meeting, it would be good to first go through the description of the assignment including the evaluation criteria. In case you have any questions regarding the requirements, please do not hesitate to contact our course assistants. We also recommend that you review the ten rules of design together. From project management perspective, it would be better to schedule regular meetings and to make sure the responsibilities and tasks are distributed fairly.

Ten rules of design:

*Make sure that the **problem is well-defined**. All design criteria, requirements and constraints, should be enumerated before a design is started.*

*Define the service to be performed at every level of abstraction before deciding which structures should be used to realize these services (**what comes before how**).*

***Design external functionality before internal functionality**. First consider the solution as a black-box and decide how it should interact with its environment. Then decide how the black-box can internally be organized. Likely it consists of smaller black-boxes that can be refined in a similar fashion.*

Keep it simple**. Fancy protocols are buggier than simple ones; they are harder to implement, harder to verify, and often less efficient. There are few truly complex problems in protocol design. Problems that appear complex are often just simple problems huddled together. **Our job as designers is to identify the simpler problems, separate them, and then solve them individually.

***Do not connect what is independent**. Separate orthogonal concerns.*

***Do not introduce what is immaterial**. Do not restrict what is irrelevant. A good design is ''open-ended,'' i.e., easily extendible. A good design solves a class of problems rather than a single instance.*

Before implementing a design, build a high-level prototype and verify that the design criteria are met.

Implement the design, measure its performance, and if necessary, optimize it.

Check that the final optimized implementation is equivalent to the high-level design that was verified.

***Don't skip Rules 1 to** 7.*

Step 1: Discuss together the use cases. Note that use cases define how users would interact with the app. You can use the following template to describe use cases. You should be able to clearly define entities and their roles in the system after this step.

Use case name:
Actors involved: (e.g. client A, client B, server and etc.)
Preconditions:
Steps:
The client xxxx
xxx
xxx
Error:
e.g. If xxx does not xxx, xxx will be xxx

Post-conditions:

Step 2: Go through the use cases and summarize the functional requirements. Note that use cases do not typically describe non-functional requirements and constraints. You can identify non-functional requirements by analyzing the operational environment (e.g. network environment, potentially large amount of users, and etc.), the expectations from user experience perspective (e.g. low latency, high resolution), security (e.g. data encryption, join approval), and any other relevant concerns.

Step 3: Architecture design. At this step, there are a couple of questions to answer. 1) Would you apply the layering technique? If so, how would you define the functionality of each layer and the interfaces between adjacent layers? You can find an example of a layered protocol architecture

from [1]. 2) Do you want to adopt a distributed or centralized architecture for data sharing? Regarding the 5 elements of a protocol, at this step, you can define the service and environment of each protocol.

Checkpoint 1 (15.2.2023): Submit a summary of functional/non-functional requirements, a draft of the architecture design, and a brief description about the next steps. Submit a PowerPoint presentation by 15.2.2023 and present it in a peer review session. Feedback will be given to each group.

Step 4: At this step, you can try to define messages to be exchanged between entities. One way is to draw the message sequence chart for each use case, and define the message format as well. It is also recommended to draw finite state machines for each entity. For example, if a protocol is about sending a file from A to B, then you can draw a state machine for the sender and another one for the receiver. Protocols must be prepared to deal appropriately with every feasible action and with every possible sequence of actions under all possible conditions. With the state machines, it would be easier to check if deadlock, livelock, or improper termination would happen.

When making your own designs at Step 3 and Step 4, it would be good to study how previous whiteboard protocols work, and think how to make a better design. You can choose to first make your own designs, and then compare with previous ones. Alternatively, you can first study the existing solutions, identify their limitations, and try to design a better one. Note that it is not acceptable if you simply copy the design of any existing whiteboarding protocol/software.

Step 5: Implement and test the protocols. It is possible to implement and test network protocols without a graphical user interface (GUI). For example, you can create a message that represents a new sticky note added to the board without a GUI. You need to check if the state machines have been implemented correctly, and test the performance, efficiency and scalability (e.g. latency and traffic size when the number of clients increases).

Step 6: Implement the whiteboard application using the designed protocols. The GUI can be kept as simple as possible. It is OK to support only a single font size and color.

You may want to implement Step 5 and Step 6 in parallel, or to switch the order. It is all up to you. Just keep in mind that the user interfaces and the protocols should belong to different modules.

Checkpoint 2 (4.3.2023): The main functions of the protocols and the app should be ready for testing. No need to submit any report at this point. This checkpoint is to help you manage the progress of your assignment.

Step 7: Test the application. Create a demo video if you prefer not to show a live demo to our course assistants. Remember to book a slot beforehand for demonstrating your work to our course assistants.

Step 8: Write the final report.

References:

[1] Marten van Sinderen, Phil Chimento, Luis Ferreira Pires. Design of a shared whiteboard component for multimedia conferencing. Proceedings of the 3rd International workship on protocols for multimedia systems. 1996. https://research.utwente.nl/files/5408604/proms96.pdf

[2] Lutz Gericke, Christoph Meinel. Evaluating an instant messaging protocol for digital whiteboard applications. International Conference on Internet Computing. 2011. https://hpi.de/fileadmin/user_upload/fachgebiete/meinel/papers/Design_Thinking/2011_Gericke_ICOMP.pdf
[3] XEP-0113: Simple Whiteboarding. https://xmpp.org/extensions/xep-0113.html#sect-idm45133283222496
[4] NetMeeting Whiteboard Protocol. https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-mnpr/0d384d5e-6d4a-4a55-ba67-6f7de6909a24
[5] 2D drawing application with real-time collaboration. https://github.com/fwcd/whiteboard

Deliverables:

1) Present your **demo** to our course assistant and submit its source code on MyCourses by **17.3.2023**. Instead of a live demo, you can also choose to create a demo video and just show the demo video.
2) **A group report** (deadline: **21.3.2023**). You can find the template of the report below.

Template of the final report (6-10 pages):

List each group member's name and student number

Chapter 1: System architecture (1 page)

Draw a figure that illustrates the high level architecture of the system. Add a brief description of the figure.
List the functional requirements. If the architecture includes several layers of protocol, you need to explain the functionality of each layer and the interfaces between adjacent layers.
List non-functional requirements.

Chapter 2: Design (2 – 4 pages)
State machines
Messages and their formats
Message sequence charts
Comparison with previous whiteboarding protocols (focusing on design)

Chapter 3: Implementation and Evaluation (2 – 4 pages)
Briefly explain how you implement the app (e.g. SDKs, reusing source code of any existing software)
Experimental setup
Test cases (#client, different use cases, different network conditions)
Result analysis (e.g. latency, data size)

Chapter 4: Team work (0.5 pages)
Describe how you have worked as a team (e.g. regular meetings, workshops, and etc.)
State clearly the responsibilities of each team member. (e.g. literature survey, programming tasks, network measurement, report writing, and etc.)

3) **An individual essay** (max 1 page, deadline: 21.3.2023) on what you have learnt from this project work. For example, what do you think are the most important things or criteria when designing communications mechanisms for an application? In this process, what do you think are the most challenging parts? What are the good things and bad things about teamwork? Feel free to discuss other topics as well.

Assessment Matrix (max 50 points + max 2 bonus points):

I. Interim report submitted at checkpoint 1 (max 5 points):

Topic (weight)	Unacceptable (0)	Marginal (1)	Acceptable (2)	Exceptional (3)
Requirement analysis (1)	Little or no grasp of the problem.	Some understanding of the problem.	Overall sound understanding of the problem. Use cases are clearly defined.	A list of functional and non-functional requirements is provided.
Project management (1)	Next steps are not planned.	Next steps are clearly defined.	Next steps are clearly defined. The schedule and task distribution are well planned.	

II. Final demo (max 23 points)

Topic (weight)	Marginal (1)	Acceptable (2)	Exceptional (3)
Joining a whiteboard session (1)	The client app can join a whiteboard session.	The newly associated participants are informed of the current view.	Each session has a host. The host can approve the requests for joining the session.
Freehand drawing (2)	A user can draw freely on the board.	A user can draw freely on the board. The erase function is supported.	More than one user can draw at the same time. The participants have a consistent (shared) view. Undo is supported.
Sticky note (1)	A user can add a sticky note to the board and can edit it.	Sticky notes can be added, deleted.	Sticky notes can be added, deleted or moved around on the board. The participants have a consistent (shared) view.
Image (2)	Images can be uploaded and displayed on the board.	Images can be uploaded and displayed on the board. Textboxes can be attached to an image. Texts in the textboxes can be edited.	Authors of comments in the textboxes are displayed. It is allowed to draw on top of images and undo is supported. The participants have a consistent (shared) view.
End a whiteboard session (1)	Any participant except the host can leave a whiteboard session whenever they want.	Any participant except the host can leave a whiteboard session whenever they want. The host can end a whiteboard session. In that case, all the other participants will have to leave as well.	Additionally, the board can be saved as a JPEG or PNG file. Any participant can save the board.
Data encryption (1)		Messages are encrypted.	

III. Final report (max 20 points)

Topic (weight)	Unacceptable (0)	Marginal (1)	Acceptable (2)	Exceptional (3)
Architecture design (1)	Little or no grasp of the problem.	Entities and their roles are clearly defined.	The service provided by each protocol involved is properly defined. If the architecture consists of several layers of protocols, the interfaces between adjacent layers are defined.	Explain why the architecture is designed in this way (e.g. why are there N layers and how you divide the functions between layers?) Explain your choice about the data sharing model (either distributed or centralised)
Protocol Design (2.5)	Not capable of achieving the objectives	The design can fulfill most of the functional and non-functional requirements.	The design can fulfill the functional and non-functional requirements.	Additionally, state machines, message sequence charts and/or class diagrams are used for explaining the design.
Evaluation (2)	The design is not evaluated.	Some tests have been done, but the test cases are not sufficient.	Performance in terms of latency, efficiency in terms of traffic size, and scalability are evaluated.	Illustrate how the design fulfills the non-functional requirements using the experimental results.
Discussion (1)	It is not compared with previous works.	The own design is compared with at least one other whiteboarding protocol. Differences in the design are identified. It is not required to compare experimental results.	Additionally, the limitations of own design are discussed.	
Academic Writing (0.5)	The report is difficult to follow. Many grammar errors in the text. A lot of text is irrelevant to the topic.	The report is otherwise easy to follow, but some important details are missing.	The report is easy to follow and the ideas are well expressed. The paper is well written, except for a few places which require clarification.	It is well written and concise. Ideas are well expressed. The report is well organized and easy to follow. Plots and diagrams are readily understandable and they support the text.

IV. Individual Essay (max 2 points)

Topic (weight)	Marginal (1)	Acceptable (2)
Reflection (1)	Cover at least one of the following perspectives: - What do you think are the most important things or criteria when designing a network protocol? - What are the most challenging parts in this process? - What I think are the good and bad things about teamwork	Cover all the following perspectives: - What do you think are the most important things or criteria when designing a new protocol? - What are the most challenging parts in this process? - What I think are the good and bad things about teamwork

V. Team work (max 2 bonus points)

Topic (weight)	Unacceptable (0)	Acceptable (1)	Exceptional (2)
Team work (1)	Tasks are not assigned to each member. The group does not manage to complete the tasks together.	Responsibility and workload is equally distributed between group members. It is fair to everybody.	Everybody shares his/her knowledge, eagerly working for achievement of the learning goal. Sharing the tasks is fair, and that helps achievement of learning goals.