Grasping the Concept of Decentralized Systems for Instant Messaging

Luisa Gebhardt
luisa.gebhardt9@kit.edu
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Marc Leinweber
marc.leinweber@kit.edu
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Florian Jacob
florian.jacob@kit.edu
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Hannes Hartenstein
hannes.hartenstein@kit.edu
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

ABSTRACT

Whether a centralized, distributed, or decentralized system approach is selected for Internet-based services affects sovereignty and responsibilities of users and providers alike. Therefore, computer science education can contribute to informed decision-making and citizenship education by teaching power structures of and responsibilities in digital infrastructures. In this practical report, we focus on the example of instant messaging. We analyze three different algorithms for instant messaging that vary in their degree of (de-) centralization. Based on the analysis, we propose a teaching activity called Klemmchat using the concept of computer science unplugged to educate students on the discovered key aspects and trade-offs. We report on results obtained by teaching Klemmchat in two classes in grades 11 and 12. The evaluation shows that the activity is suitable for conveying trade-offs and helping students to engage with the topic. The results, however, leave open whether the acquired understanding affects usage decisions.

CCS CONCEPTS

• **Security and privacy** → *Social aspects of security and privacy*; • **Applied computing** → **Education**; • **Computer systems organization** → *Distributed architectures.*

KEYWORDS

Citizenship Education, Distributed Computing Education, Computer Science Unplugged

ACM Reference Format:
Luisa Gebhardt, Marc Leinweber, Florian Jacob, and Hannes Hartenstein. 2022. Grasping the Concept of Decentralized Systems for Instant Messaging. In *Proceedings of the 17th Workshop in Primary and Secondary Computing Education (WiPSCE '22), October 31-November 2, 2022, Morschach, Switzerland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3556787.3556864

1 INTRODUCTION

Decentralization is a rather abstract notion, still, it strongly affects sovereignty and responsibilities of involved parties and promises favorable properties such as independence and freedom (see, e.g., [3]). The notion of decentralization can be transferred from a sociological and political context in form of governments or political systems to the governance of Internet-based services like instant messaging. Thus, understanding decentralization forms the basis for an informed usage decision of Internet-based services as well as for understanding power structures and responsibilities in organizations or institutions in the (digital) world in general. Therefore, we build on the thesis that gaining an understanding of the trade-offs and differences between centralized, distributed, and decentralized systems is important for high school students: This knowledge can enable students to become "responsible creators" [7, translated].

In a *centralized* instant messaging system, there is one single entity that controls the system, for example Meta (Facebook) in the case of the WhatsApp messaging service. For scalability, a service like WhatsApp is not provided by a single server but by a *distributed* system: many servers are used to provide the service, but still one entity controls the corresponding messaging system. In a *decentralized* (messaging) system, such a service is provided by various entities that cooperate but do not have unilateral control over the system as a whole (see, e.g., [5] for a clarification of these notions). While decentralized instant messaging services exist (see, e.g., the Element messenger based on [10]), their popularity compared to centralized/distributed messaging services is still rather low.

We observe that the concept of decentralization meets the criteria of a fundamental idea by Schwill [13]. As the idea of decentralization in power structures can be found recently in technological contexts, and for a long period of time in societal contexts, the idea meets the horizontal criterion and the criteria of time. As elaborated before, the idea of differently decentralized power structures is also present in every-day technology such as instant messengers and email. Therefore, it also meets the criterion of sense. We propose a teaching concept for the idea of decentralization in power structures, thereby showing that this idea meets the vertical criterion as well, as it can be taught on different levels of abstraction. The case of instant messaging and the different corresponding power structures, depending on the degree of decentralization, can therefore be used as a contribution of the field of computer science education to informed decision-making and citizenship education (see, e.g, [7]).

We investigate different messaging approaches by looking at their underlying algorithms: we select one centralized, one distributed, and one decentralized algorithm from [6]. We are interested in identifying the "essence" of these approaches: reliability and power structure. The centralized algorithm represents *i)* a single point of failure that trades reliability for less computational effort, and *ii)* a single point of control that can prevent malicious or discriminatory acts by individual users while affording the same

WiPSCE '22, October 31-November 2, 2022, Morschach, Switzerland

ACM ISBN 978-1-4503-9853-4/22/10…$15.00
https://doi.org/10.1145/3556787.3556864

powers to providers. In contrast, the distributed algorithm does not rely on a single point of failure. Moreover, the decentralized algorithm also does not rely on a single point of control, making it more difficult to take measures against individual discrimination, but preventing structural discrimination by giving users more responsibility and independence.

Consequently, the research question underlying this practical report is as follows: *How can the discovered socio-technical characteristics and trade-offs, in particular regarding reliability and power structures, of instant messaging be taught for informed decision-making?* To this end, we developed a teaching activity called `Klemmchat` that makes the previously introduced algorithms and their trade-offs and characteristics tangible. The activity is inspired by Sivilotti and Pike [15] and the criteria for Computer Science Unplugged by Bell and Vahrenhold [1]. In this teaching activity, students enact the different algorithms in simplified form and are encouraged to attack each algorithm or to play a faulty party during each simulation. Afterwards, students are encouraged to discuss their findings to discover trade-offs, responsibilities and power structures, as well as their implications, such as necessary trust. We report on a teaching experiment with 25 students with an observer and a questionnaire using the SOLO Taxonomy [2] to measure the effects on knowledge as well as on informed usage decisions regarding instant messaging. The evaluation shows that the majority of students reaches the multi-structural or relational stage of the SOLO levels.

The paper is structured as follows. We discuss the selected algorithms regarding their trade-offs and possible influence of misbehaving parties and propose a simplified version for each of the algorithms in Sec. 2; corresponding learning objectives are derived in Sec. 3. Using the discussed algorithms, we propose the `Klemmchat` activity in Sec. 4 to teach the socio-technical trade-offs and influence of misbehaving parties to students. We evaluate `Klemmchat` regarding the achievement of the learning objectives by using the SOLO Taxonomy in Sec. 5 and conclude in Sec. 6.

2 CORE ASPECTS

We are interested in instant messaging services offering a group chat functionality. Each chat group consists of a group of users, and messages written by different users forming the group chat's history. Each chat history is a totally ordered list of all messages sent by users. At the core of instant messaging functionality is an algorithm to determine the chat history based on sent messages. The algorithms are derived from [6] and simplified for easier understanding of their reliability and underlying power structure. The distributed algorithm is derived from the *Majority Voting Regular Register*, the decentralized algorithm from the *Authenticated-Data Byzantine Quorum*. "Byzantine" indicates that participants may behave arbitrarily. Instead of providing a single register value, we adjust the algorithms to provide a list of values to represent a chat history. In the teaching experiment, these adjustments intentionally make the algorithms less robust to attacks, for example by a Byzantine system of students (cf. Sec. 4). Additionally, we assume a secure communication channel between servers and users.

In centralized messaging, messages are sent by users directly to a single server hosted by a provider. The server verifies the identity of the sender and appends the message to the chat history.

In distributed messaging, messages are sent by users to a randomly chosen server s from a group of servers S that emulate a single server under the control of a single provider (logical centralization [5]). This randomly chosen server s distributes each message to all other servers. Each server in S maintains a local chat history to which received messages are appended. If the local copies of the chat history differ, the servers perform a majority vote in which more than $|S|/2$ servers have to agree on the same history.

In decentralized messaging, each user operates their own server. All messages are sent directly from a user's local server to servers of other users. Each server appends messages to its local chat history. Again, in case of diverging histories, a majority voting is performed.

We analyze all three chat group architectures regarding the consequences of the existence or non-existence of a single point of failure and a single point of control.

Reliability: A single point of failure is characterized by a single piece of technology on which the messaging service depends. If a single point of failure is faulty, then the entire service stops working. This single point of failure is apparent in the centralized algorithm as a single server provides the entire service, making the system as a whole vulnerable to attacks by clients and, thus, impeding the reliability of the service as a whole. If the server crashes or a client spams the server thus occupying its resources indefinitely, a user cannot rely on the service to work. To avoid a single point of failure, distributed and decentralized messaging distributes the load and chat histories across multiple servers. This distribution increases both fault-tolerance and the computational effort, making it more expensive to run such a messaging service. The computational effort increases as distributed messaging requires coordination of the servers to keep chat histories consistent. Users therefore need to rely on and trust either external providers or their own server.

Power Structures: A power structure defines who is capable of shaping and influencing a given instant messaging service. Power over the service is achieved by hosting and/or accessing a majority of servers participating in a messaging service. A user of such a messaging service needs to trust the controlling entity to act benevolently towards all users. A single point of control exists in the centralized and distributed algorithm in the form of a provider.

Centralized power can be used to the benefit of users, as a single point of control can shield the messaging service and users against malicious actors that try to harm the messaging service itself. Such malicious actors can be excluded by commanding servers to discard or not accept their messages. A single point of control can also protect messaging users from harmful message contents of other users. To this end, a single point of control might check the contents of a message and filter, e.g., *non-constitutional* or *discriminatory* messages. The social media service Twitter uses this power to tag potential fake news accordingly for its users [18].

The power of a single point of control can also be abused. A single point of control may *censor* a service or *oppress users and opinions* by either excluding non-malicious users or messages with a specific content. In analogy with email, an email provider can discard or reject emails according to arbitrarily defined rules. This would exclude customers of this provider from ever receiving emails from an example.org user, potentially non-transparent to any users.

For users who do not want to trust a provider, it is possible to use decentralized messaging. As every user provides their own

server, a messaging-controlling third party is no longer required. Nevertheless, a single point of control can still exist if a majority of servers is somehow controlled by the same entity. After all, every decision in the messaging service is made through a majority vote, *involving all participants*. Consequently, if a malicious client should be excluded from all messaging, a majority of participants must coordinate and exclude the same servers. Looking at the example from before in which a specific domain was ignored by an email provider, the exclusion of a domain by one provider only affects customers of this one provider. Thus, a mail sent from an excluded domain can still be received by users independent of this provider. For malicious message contents, the case is quite similar, as each receiving user has to classify malicious message contents on its own. We see that, due to the voting process, the difficulty of preventing malicious message contents and of excluding malicious participants has increased while, at the same time, structural discrimination and censorship becomes more difficult. Thus, decentralized messaging trades *freedom* and *independence* from a provider with *responsibility to operate and participate with a server* to protect oneself against discrimination. Centralized or distributed messaging is prone to attacks from providers, while decentralized messaging is prone to attacks from users inside chat groups.

3 LEARNING OBJECTIVES

The different messaging architectures directly imply trade-offs related to citizenship education: In decentralized messaging, a user has to take responsibility but also has the power to participate equally in a messaging service. In contrast, in distributed and centralized messaging, a user delegates its power and responsibility to a trusted provider to protect them. Independent of personal preferences or the fit of an algorithm for a certain use case, we recognize the importance of understanding the implications, differences, and trade-offs of these architectures for informed decision-making and citizenship education.

We teach these trade-offs and properties to students by looking directly at the different algorithms. In addition, students should gain the ability to apply these trade-offs to usage decisions regarding messengers. Thus, it is important for students to not only gain knowledge in the form of describing architectures and mechanics or simply listing or contrasting trade-offs for given scenarios. This knowledge can be applied to reflect and evaluate the usage of messaging services. Thus, we propose these learning objectives using the operators according to the SOLO Taxonomy:

Students are able to ...

- *LO 1:* *describe* the basic functionality of the different algorithms (centralized, distributed, decentralized) for instant messaging
- *LO 2:* *discuss* the trade-offs of the different algorithms
- *LO 3:* *contrast* which algorithm to use in a given scenario based on their characteristics
- *LO 4:* *evaluate* instant messaging services, based on their underlying algorithm, in different usage scenarios.

4 KLEMMCHAT ACTIVITY

To choose a suitable method for our teaching activity, we studied the existing research in the fields of distributed computing education and citizenship education. The existing research of teaching in the field of distributed computing education, such as [11], mainly focuses on teaching technical intricacies and trade-offs of distributed (i.e., architecturally decentralized) computing. In contrast to the existing research, our research solely focuses on the aspects of decentralization of power structure and reliability (cf. Sec. 2).

Distributed computing can be taught by using a simulation framework on a computer (see, e.g., [4, 14]), metaphors [12], or kinesthetic learning activities to simulate algorithms [11] which are nowadays known as Computer Science Unplugged [1].

To educate students on citizenship, the "Guardian of Democracy" [8] names six "proven practices" of civics learning. Hahn [9] analyses the research concerning two of them, namely simulation of democratic practices and discussion of recent events, showing that especially simulation but also discussion is an effective tool to educate students on citizenship. Stephens et al. [16] show that especially the simulation of democratic processes is of great aid when teaching citizenship.

To simulate democratic processes with kinesthetic learning activities [15], we propose an unplugged learning activity named `Klemmchat`. We choose the unplugged approach as it does not require any additional knowledge of students to execute the algorithms and allows abstracting from unnecessary technical intricacies, making the reliability measures and power structure directly observable for all students.

4.1 Unplugging Instant Messaging: Klemmchat

In `Klemmchat`, students simulate and exploit different algorithms in a messaging service with a single chatroom to learn about distributed and decentralized computing.[1] To simulate the algorithms unplugged, we use simple tools to translate them into graspable procedures. We form small groups of about four students each that impersonate a user/server, assign each group a unique color as identifier, and distribute them in the classroom. We use colored lego duplo bricks as traceable message packets that contain a message written on them using a whiteboard marker. Each group receives a set of bricks of their unique color. This color identifies the group and acts akin to a Public Key Infrastructure (PKI), so a server can verify the identity of each message's sender. The chat history (Fig. 1) is formed by appending a newly received message brick to the bottom of the existing stack (Fig. 2).

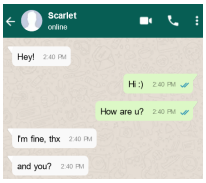

Figure 1: Chat in WhatsApp

Figure 2: Chat in `Klemmchat`

A write operation is signaled by a sender and corresponding message bricks are collected and verified regarding color and message written on them by one of the processing servers. This procedure eliminates a potential message queue and strengthens the view on

[1] Additional material can be found at https://www.dsn.kastel.kit.edu/software.php

messaging as a service. It also translates the communication effort of the algorithms into real physical effort.

To simulate the centralized algorithm, all groups play a user, whereas the central server is played by the teacher. To write a message, the corresponding message brick is collected, verified and appended by the teacher. To read a message, the teacher reads every lego brick from the top to the bottom of the stack in the form of: "*color of brick* writes *message*".

To simulate the distributed algorithm, a group of students is chosen that simulates the group of servers S instead of a chat user. This group is not allowed to write messages. Each of the students in this group represents a single server s, so each student has its own stack of message bricks. For a correct write, a group impersonating a user must provide $|S|$ identical bricks for the group S. Only one student s of the group S collects all message bricks corresponding to a single message. Student s is required to ensure that all collected bricks are the same and distributes these bricks the other students in S. Then, all students in S append the message to their stack. To read the chat history, the server group S performs a majority vote by agreeing on one of the $|S|$ message histories. From now on, concurrency is a major problem of the algorithms as multiple messages can be written at once. Therefore, a server might append one message m_1 before another message m_2, while another appends m_2 before m_1. In some cases, this makes agreeing on a chat history difficult. This issue should be briefly discussed once encountered. Students may directly target this issue by reordering the messages.

To simulate the decentralized algorithm, every group of students plays a server with its own chat history, but is allowed to write messages. To write a message, a sending group must provide an identical signed message brick to all groups including themselves. All other groups task a student of their group to collect a single message brick from the sending group. The groups then verify the identity of the sender and append the brick to their local stack. To read a message, a representative of each group presents their chat history and the representatives perform a majority vote by agreeing on one of the presented message histories. In the decentralized algorithm there exists no trusted third party that preserves the integrity of messages. The same problem exists in our simulation, as the lego bricks are now exchanged directly between users. Thus, the brick content may be manipulated. The decentralized algorithm uses digital signatures to verify the senders identity. To simulate digital signatures, each group receives a tape of a unique color, sticks the tape onto their bricks, and writes the message onto the tape. The tape makes it impossible to erase the message without removing the tape. To verify the identity, the receiver now compares the brick and tape color combination to a global look-up table, which matches tape and brick color to the correct sender.

4.2 Activity Structure

The activity consists of three successively simulated algorithms framed by an introduction and a final discussion. The introduction explains the general setup to students, whereas the discussion should secure the results of the simulations and should help students in transferring their newly acquired knowledge to their messaging service usage decisions. In the discussion, the teacher should map the simulated algorithms to various real messaging services.

Each algorithm is played by the students for multiple rounds. To make it easier for students to observe the simulation, each round has an explicit write and read phase. In the write phase, the students have time to write messages; in the read phase, a read operation will be performed. The agreement process in the read operation is displayed in front of all students, so that all students can observe the agreement process and its result. This transparency is necessary to recognize and uncover the influence of the power structure in each algorithm. The first round of each experiment serves as a warm-up in which the algorithms should be simulated according to its specification. After the first round, students are encouraged to attack the algorithm to discover flaws and reliability measures in the algorithm design. If the students struggle coming up with ideas, the teacher may assist.

After the last round of each experiment, a discussion on the discovered problems and an evaluation of trade-offs compared to previous algorithms is started with the students. Hence, the students comment on the improvements and deterioration in regards to the previous algorithm. Additionally, the students are asked for possible improvements to each algorithm. The students are expected to come up with trade-offs connected to the reached reliability and consequences of power structures as discussed in Sec. 2.

5 EVALUATION

To evaluate the proposed teaching activity, we conducted a teaching experiment with an observer and a questionnaire after the activity. The evaluation was performed independently in two different classes of grade 11 and 12 with 13 and 12 students respectively at a high school in Baden-Württemberg during regular Computer Science lessons. The students of grade 11 have chosen the main subject of IMP (Combination of Informatics, Maths and Physics) since grade 9, the students of grade 12 have chosen Computer Science as elective minor subject for grade 12 and 13 but no prior Computer Science education. As the teaching activity was planned for 90 to 115 minutes, the activity was split between the centralized and distributed algorithm. The two parts were taught in a 45 min lesson and a 90 min lesson a week apart. In grade 11, the teaching experiment was conducted in the morning, in grade 12 in the late afternoon. The questionnaire was conducted at the end of the second part and was time-boxed by the end of the lesson.

5.1 Methodology

5.1.1 Observation. During the teaching experiment, an observer was tasked with recording the activity neutrally by writing down all student and teacher inputs as well as all performed attacks. We analyzed the discovered trade-offs, performed attacks, as well as the teacher's assistance, and suggested improvements for the algorithm.

5.1.2 Questionnaire. A questionnaire was used to investigate the knowledge gain of students regarding the learning objectives. For this purpose, we selected six questions of varying difficulty (see Tab. 1). Each question is mapped to a specific level of the SOLO taxonomy. We use the five levels of the SOLO taxonomy as adapted by Thies and Vahrenhold [17]. To measure the results numerically, we assigned each SOLO level a number ranging from 0 (prestructural) to 4 (extended abstract). For each question, we assigned the

Table 1: Questions to evaluate the activity and knowledge of students ordered by difficulty. Each question is mapped to a SOLO level and corresponding learning objectives. If not obvious, the requirements for a correct answer are stated.

Level	LO	Question	Requirements
Uni-structural	1	Who owns the chat history in the respective alg.?	
Multi-structural	2	Name two advantages and disadvantages of each alg.	$\geq$ 6 correct mentions
Relational		Scenarios: Which messenger would you recommend and why?	coherently reasoned answer
	3/4	· two grandmas, private conversation	
	3/4	· university, good IT infrastructure	
Extended Abstract	3/4	· Scenario: Teachers exchange grades. Decentral. alg. suitable?	privacy mentioned coherently reasoned answer
	4	· Identify problems/risks regarding your messenger usage. How do you deal with these in the future?	

corresponding learning objectives (LO) and defined properties that a correct answer fulfills (if a correct answer is not obvious) and classified a student into the level of understanding according to the most difficult question that is correctly answered. If there are multiple questions for a level, the stage is reached if all questions of the level and the levels below were answered correctly. This means that a student reached a relational level of understanding if all but the tasks of extended abstract are answered correctly.

5.2 Results

5.2.1 Observation. The students in our experiment discovered multiple aspects of each algorithm: The explicit remarks made by the students are given in Tab. 2 (we summarized semantically equivalent contributions). The mentioned aspects correspond mostly with the reached reliability and consequences of power structures of instant messaging. Students in grade 11 quickly came up with different attacks and were less inhibited. The students in grade 12 took more time carefully planning attacks and did not perform certain attacks directly harming others as "that would not be nice". The contributed observation of the property "errors can be detected" by the students is – in our interpretation – a result of the transparency of the simulation. The properties which are only observable due to simulation need to be targeted explicitly to avoid misconceptions. As our observation shows, these simulation-only properties were discussed in each class if necessary. The absence of confidentiality and encryption in the whole simulation was never mentioned by students. Thus, confidentiality was not discussed in either class.

Both classes came up with various ideas on how to improve each algorithm, e.g., using cryptography or performing agreement processes during write operations instead of reads. In subsequent rounds of the distributed and decentralized algorithm, as concurrency was an increasing problem, with students even intuitively implementing new measures to synchronize message histories.

Table 2: Overview of trade-offs/improvements/deterioration. The aspects mentioned by the students during the discussion correspond with reliability and power structure.

Property	Grade 11	Grade 12
Centralized Algorithm: Trade-offs		
easily overloaded	x	x
message & sender integrity not secured	x	x
arbitrary messages can be added	x	
censorship	x	x
understandable	x	x
easy message ordering	x	x
identity checking faults		x
cannot read while writing	x	
errors can be detected	x	
Distributed Algorithm: Improvements/Deteriorations		
difficult to overload	x	x
exclusion possible	x	
message ordering & agreement difficult	x	x
errors can be detected		x
Decentralized Algorithm: Improvements/Deteriorations		
no censorship	x	
more complex	x	
minorities can be overruled	x	x
mal. party can make agreement impossible	x	x

Table 3: Number of students who reached a specific SOLO level. The overall mean level is $\mu = 2.04$. The students from grade 11 with two years in the subject "*I*nformatics, *M*athematics, *P*hysics" perform better ($\mu = 2.77$) than the students in grade 12 without the subject ($\mu = 1.25$).

SOLO Level	Prestr. (0)	Unistr. (1)	Multistr. (2)	Relat. (3)	Ex. Abstr. (4)
Grade 11 (IMP)	0	0	6	4	3
Grade 12 (no IMP)	5	2	2	3	0

5.2.2 Questionnaire. The results of our SOLO level classification of the students are depicted in Tab. 3. The mean of $\mu = 2.04$ shows that on average the students have reached a multistructural level of understanding of the trade-offs of different messaging architectures. More than a third of the students reaches a level of understanding above multistructural. Additionally, we observe that the students of grade 11 who already had two years of Computer Science lessons performed better overall.

If we take a closer look at all answers of the extended abstract stage, we can observe a quite fascinating combination: Some students fail to mention the importance of privacy and cryptography when it comes to the exchange of grades via a messaging services at all. Some students seem to assume the existence of tools to ensure confidentiality of message contents in the decentralized algorithm. In contrast to that, at the same time nearly all students identified missing confidentiality of message contents and a lack of encryption as the most important problem of their used messaging services, which they all correctly identified as distributed. However, the majority of messaging services they use encrypt their messages.

The large number of students still on a prestructural level of understanding is unexpected. Upon closer inspection, we see that

three of these students answered with "multiple providers" when asked about the ownership of the chat history in the distributed algorithm. This answer is quite ambiguous, as we do not know whether the students meant multiple independent providers or multiple servers of the same provider. Additionally, one student failed to give answers to the ownership questions at all. However, we assume that these students have actually achieved a higher level of understanding, as they answered the following questions correctly. If we classify those four students according to the remaining answers, we receive a mean of $\mu = 2.48$.

5.3 Discussion and Limitations

Based on the reached SOLO levels of the defined learning objectives and taking the notes of the observer into account, we conclude that the proposed teaching activity works reasonably well. The differences in the performance of the two classes indicate that prior Computer Science education or knowledge might influence the effectiveness of our concept. However, we do not have the necessary data to substantiate this indication.

As our observations vary for each class, we assume that the creativity to come up with different attacking ideas greatly influences the discovered properties. Due to two years of prior Computer Science education, the students of grade 11 have more experience and potentially a higher intrinsic motivation. Thus, they discovered more aspects with less assistance in the same time compared to the students of grade 12.

The deliberate choice of not introducing encryption might have led to confusion regarding the level of confidentiality or privacy one can assume. As cryptography is currently absent from the whole activity, the students assumed that some messaging services do not need or do not use encryption, even though the services encrypt their messages. The 'induced' misconception might interfere with the collection of results corresponding to informed decision-making. Thus, one should be cautious to interpret the results of our study regarding the ability of informed decision-making. However, this aspect of integrating encryption and/or privacy needs to be investigated further in future work. We definitely suggest to integrate aspects of encryption into the proposed activity to enable students to also link encryption with power structures.

We were not able to investigate ability *improvements* of students, as we did not make a test targeting the learning objectives before the activity. We used a questionnaire with multiple free text answers to investigate the ability to make informed decisions. These questions were, due to the limited time frame, only answered briefly by the students. Most students gave only a single reason for their decisions.

6 CONCLUSION

We investigated reliability and power structures of different instant messaging algorithms: Essentially, while architectural decentralization gains increased reliability by investing in computation and communication, the decentralization of power gains a higher degree of sovereignty and protection against structural discrimination, however by giving up some convenience and protection against malicious individual users. As shown in the evaluation of the proposed teaching activity `Klemmchat`, sovereignty, responsibility, and power structure in Internet-based services can be taught at the example of instant messaging using concepts of computer science unplugged and without deep-diving into technical specifications. As future work, a study based on individual interviews with more time for a pre and post test could evaluate the actual influence on informed decision-making. In addition, aspects of confidentiality and privacy need to be integrated into the `Klemmchat` teaching activity.

ACKNOWLEDGMENTS

We like to thank the reviewers for their insightful feedback.

REFERENCES

[1] Tim Bell and Jan Vahrenhold. 2018. CS Unplugged—How Is It Used, and Does It Work? In *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*. Springer International Publishing, Cham, 497–521. https://doi.org/10.1007/978-3-319-98355-4_29

[2] John B. Biggs and Kevin F. Collis. 1982. *Evaluating the Quality of Learning: The SOLO Taxonomy (Structure of the Observed Learning Outcome)*. Academic Press New York. https://doi.org/doi.org/10.1016/C2013-0-10375-3

[3] Balázs Bodó, Jaya Klara Brekke, and Jaap-Henk Hoepman. 2021. Decentralisation: A multidisciplinary perspective. *Internet Policy Review* 10, 2 (2021), 1–21. https://doi.org/10.14763/2021.2.1563

[4] Cora Burger and Kurt Rothermel. 2001. A Framework to Support Teaching in Distributed Systems. *J. Educ. Resour. Comput.* 1, 1es (mar 2001), 3–es. https://doi.org/10.1145/376697.376698

[5] Vitalik Buterin. 2017. *The meaning of decentralization.* Retrieved 2022-08-10 from https://medium.com/@VitalikButerin/the-meaning-of-decentralization-a0c92b76a274

[6] Christian Cachin, Rachid Guerraoui, and Luís Rodrigues. 2011. *Introduction to Reliable and Secure Distributed Programming*. Springer, Berlin Heidelberg. https://doi.org/doi.org/10.1007/978-3-642-15260-3

[7] Robert Feil. 2019. *Leitfaden Demokratiebildung*. Ministerium für Kultus, Jugend und Sport Baden-Württemberg, Stuttgart. https://km-bw.de/site/pbs-bw-km-root/get/documents_E-2008466037/KULTUS.Dachmandant/KULTUS/KM-Homepage/Publikationen%202019/2019_Leitfaden%20Demokratiebildung.pdf

[8] Jonathan Gould, Kathleen H. Jamieson, Peter Levine, Ted McConnell, and David B. Smith. 2011. *Guardian of Democracy: The Civic Mission of Schools.* The Leonore Annenberg Institute for Civics of the Annenberg Public Policy Center at the University of Pennsylvania. https://media.carnegie.org/filer_public/ab/dd/abdda62e-6e84-47a4-a043-348d2f2085ae/ccny_grantee_2011_guardian.pdf

[9] Carole L. Hahn. 2016. Pedagogy in citizenship education research: A comparative perspective. *Citizenship Teaching & Learning* 11, 2 (2016), 121–137. https://doi.org/10.1386/ctl.11.2.121_1

[10] Matrix.org Foundation C.I.C. 2022. *Matrix Specification v1.3.* Technical Report. https://spec.matrix.org/v1.3/

[11] Florin Pop and Valentin Cristea. 2019. Distributed Systems Education: From Traditional Models to New Paths of Learning. In *22nd International Conference on Control Systems and Computer Science (CSCS 2019)*. 383–386. https://doi.org/10.1109/CSCS.2019.00070

[12] Kiwamu Sato, Norio Shiratori, Hiroshi Nunokawa, Takashi Kusumi, and Syoichi Noguchi. 1997. A user interface metaphor for distributed systems. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)* 80, 11 (1997), 82–93. https://doi.org/10.1002/(SICI)1520-6440(199711)80:11<82::AID-ECJC9>3.0.CO;2-U

[13] Andreas Schwill. 1993. Fundamentale Ideen der Informatik. *Zentralblatt für Didaktik der Mathematik* 25, 1 (1993), 20–31.

[14] Alan Shaw. 2014. A framework for teaching centralized and decentralized peer-to-peer programming paradigms in introductory computer science courses. In *Proceedings of the 2014 ACM Southeast Regional Conference* (Kennesaw, GA, USA) *(ACM SE '14)*. ACM, 42:1–42:4. https://doi.org/10.1145/2638404.2638515

[15] Paolo A. G. Sivilotti and Scott M. Pike. 2007. A Collection of Kinesthetic Learning Activities for a Course on Distributed Computing: ACM SIGACT News Distributed Computing Column 26. *SIGACT News* 38, 2 (jun 2007), 56–74. https://doi.org/10.1145/1272729.1272741

[16] Jason Stephens, Joseph Feinberg, and John Zack. 2013. Those who do: Social studies teachers' use of role play and simulations and the making of 21st century citizens. In *The Status of Social Studies: Views from the Field.* Information Age Publishers, Charlotte, NC, USA, 259–279. http://hdl.handle.net/2292/30241

[17] Renate Thies and Jan Vahrenhold. 2016. Back to School: Computer Science Unplugged in the Wild. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education* (Arequipa, Peru) *(ITiCSE '16)*. ACM, New York, NY, USA, 118–123. https://doi.org/10.1145/2899415.2899442

[18] Twitter, Inc. 2022. *How we address misinformation on Twitter.* Retrieved 2022-08-10 from https://help.twitter.com/en/resources/addressing-misleading-info