



HAL
open science

Keynote: From groupware to large-scale trustworthy distributed collaborative systems

Claudia-Lavinia Ignat

► To cite this version:

Claudia-Lavinia Ignat. Keynote: From groupware to large-scale trustworthy distributed collaborative systems. CRIWG 2018 - 24th International Conference on Collaboration and Technology, Sep 2018, Costa de Caparica, Portugal. hal-01875534

HAL Id: hal-01875534

<https://hal.science/hal-01875534v1>

Submitted on 17 Sep 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

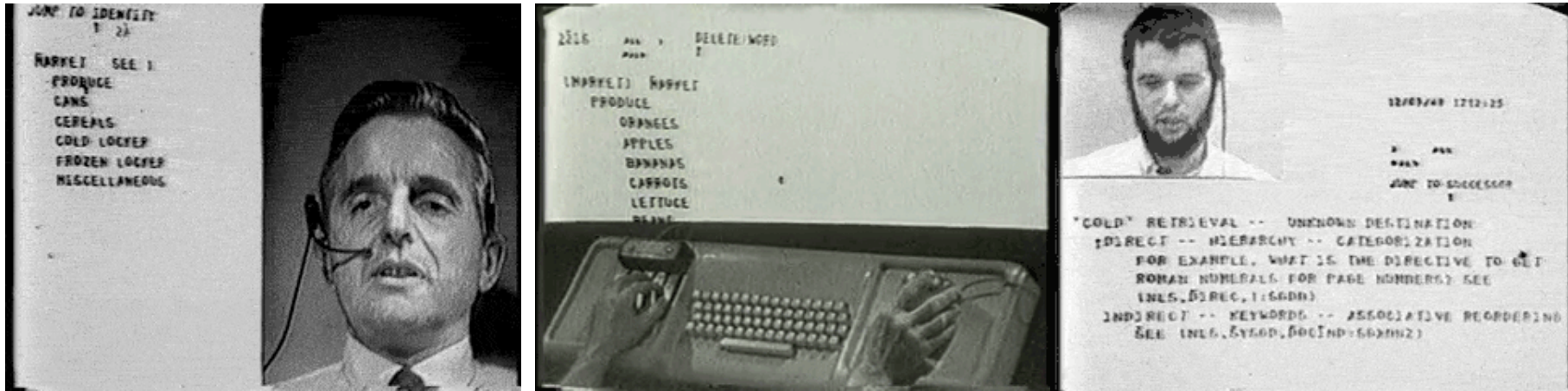


From groupware to large-scale trustworthy distributed collaborative systems

Claudia-Lavinia Ignat, Inria, France

CRIWG 2018
September 5, 2018
claudia.ignat@inria.fr

Douglas Engelbart: Augmenting Human Intellect



The Mother of all Demos, December 9, 1968



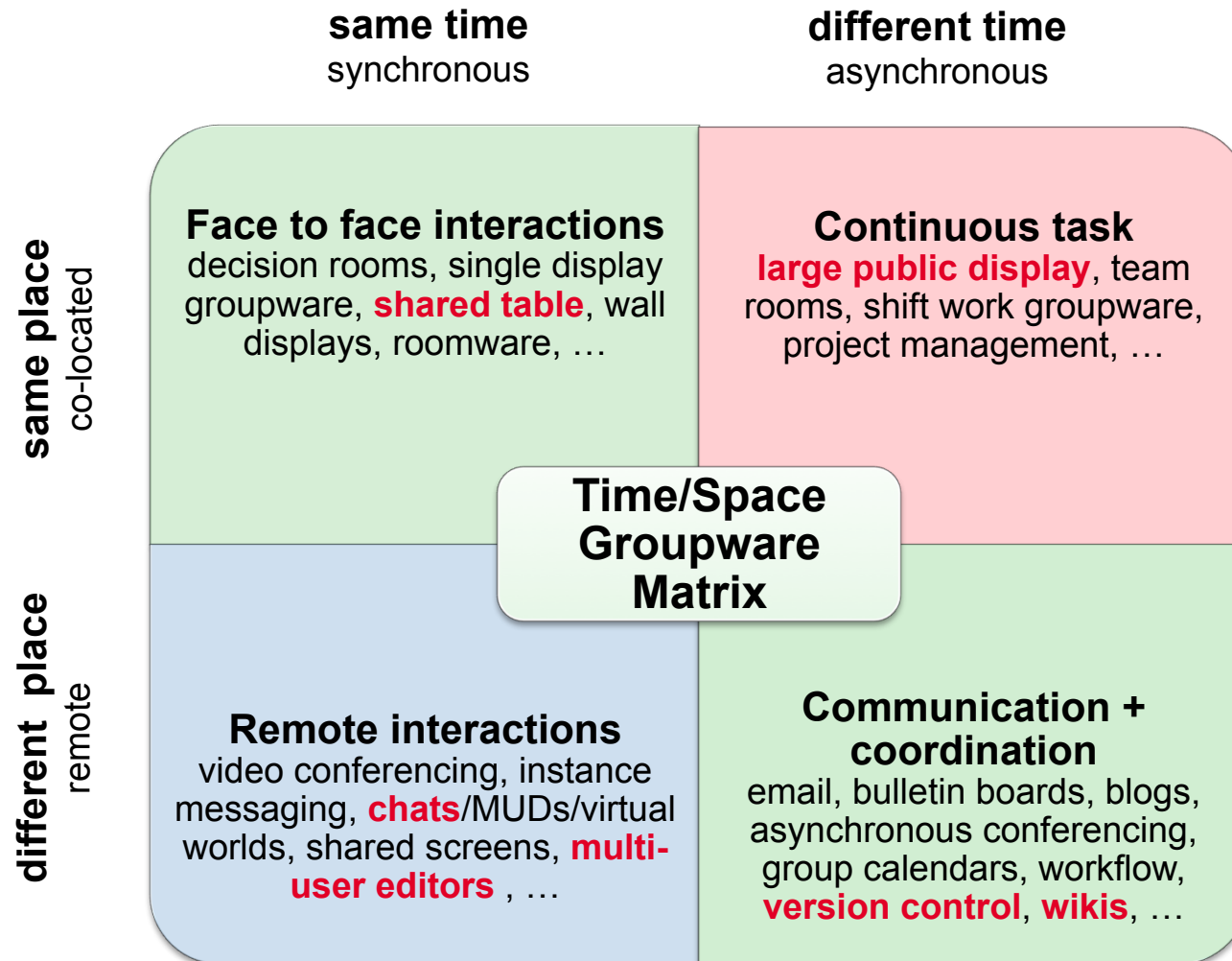
NLS: Online System

<https://archive.org/details/doungengelbartarchives>

Groupware, early 1990s

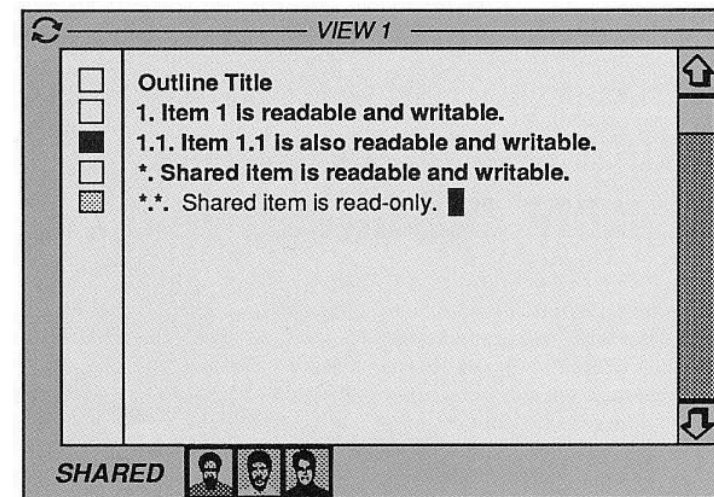
- « Computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment. » [EGR91]
- Lotus Notes, one of the first commercial groupware allowing remote group collaboration

Groupware Time Space Matrix [J88]



Groupware: supported solutions

- *Turn taking*: allow only one active participant at a time
 - e.g. *RTCAL* [SG88], *SHARE* [G90]
- *Locking*: concurrent editing allowed only if users lock and edit different objects
 - e.g. *Colab* [SFBKLS88]
- *Operational transformation*
 - e.g. *GROVE* [EG89]



Google Drive

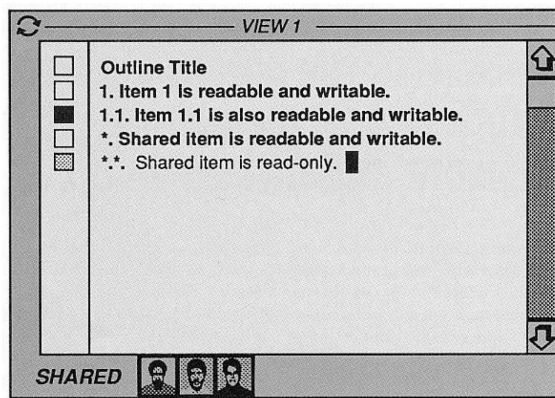


Collaborative Systems: from users to community of users



“Isn’t it chaotic to all edit in the same document, even the same paragraph, at the same time?”

“Why would a group ever want to edit in the same line of text at the same time?” [EGR91]



GROVE, 1989

Collaborative Systems: from users to community of users



- 2013: MOOC “Fundamentals of Online Education: Planning and Applications” with 40.000 participants
- 2016: Nuit debout, more than 70 people edit a pad
- 2018: online CSCW PC meeting with 120 members

Collaborative Systems: from users to community of users

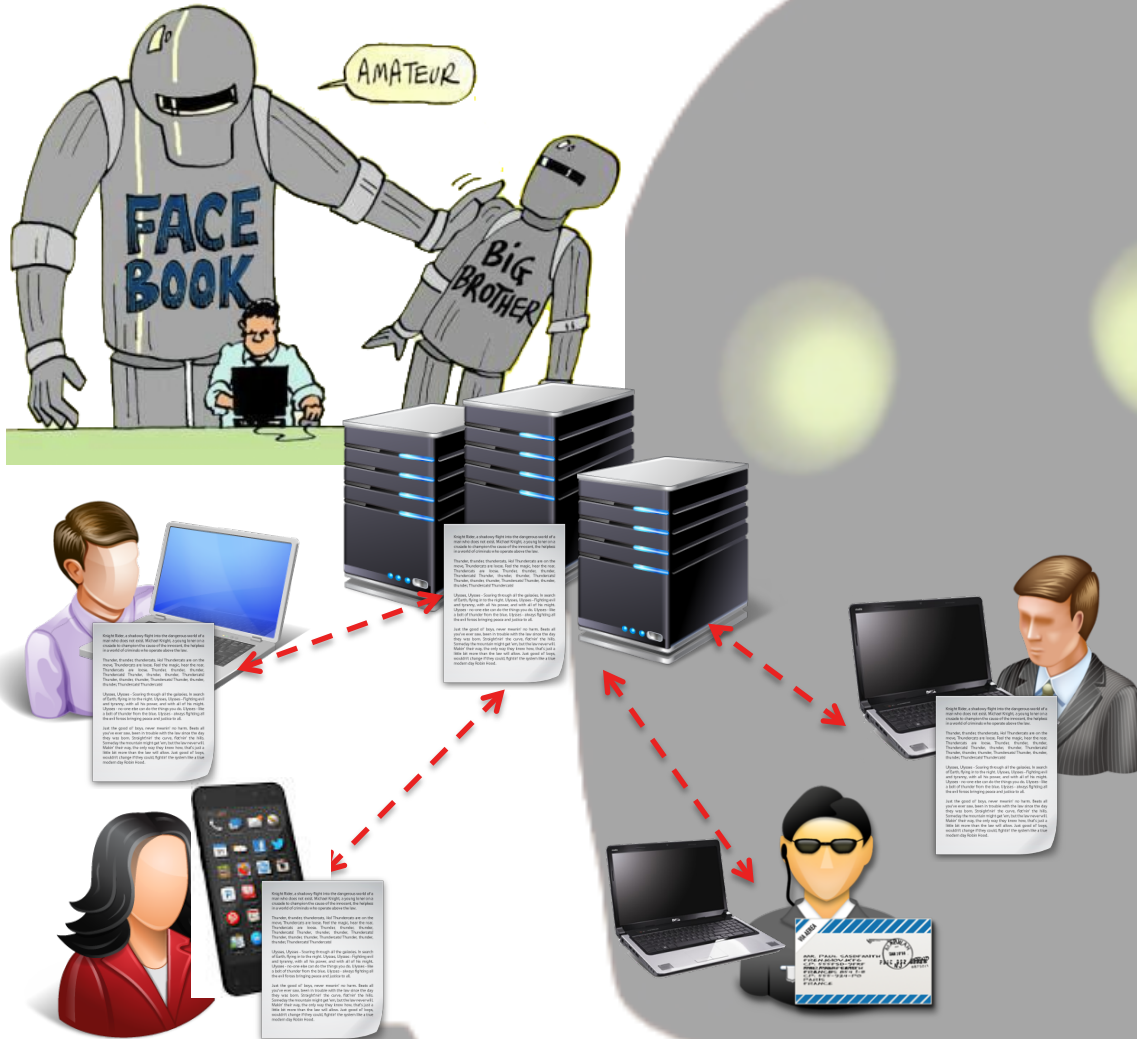
Real-time
Wikipedia

The screenshot shows the Wikipedia revision history page for 'Ponte Morandi'. The page title is 'Ponte Morandi: Revision history'. The URL is 'https://en.wikipedia.org/w/index.php?title=Ponte_Morandi&offset=20180814122400&limit=24&action=history'. The page includes a search bar for revisions, a list of 24 revisions with details like date, time, user, and byte changes, and navigation links for 'newest', 'oldest', and 'View (newer 24 | older 24)'. The left sidebar contains various Wikipedia navigation options like 'Main page', 'Contents', and 'Tools'.

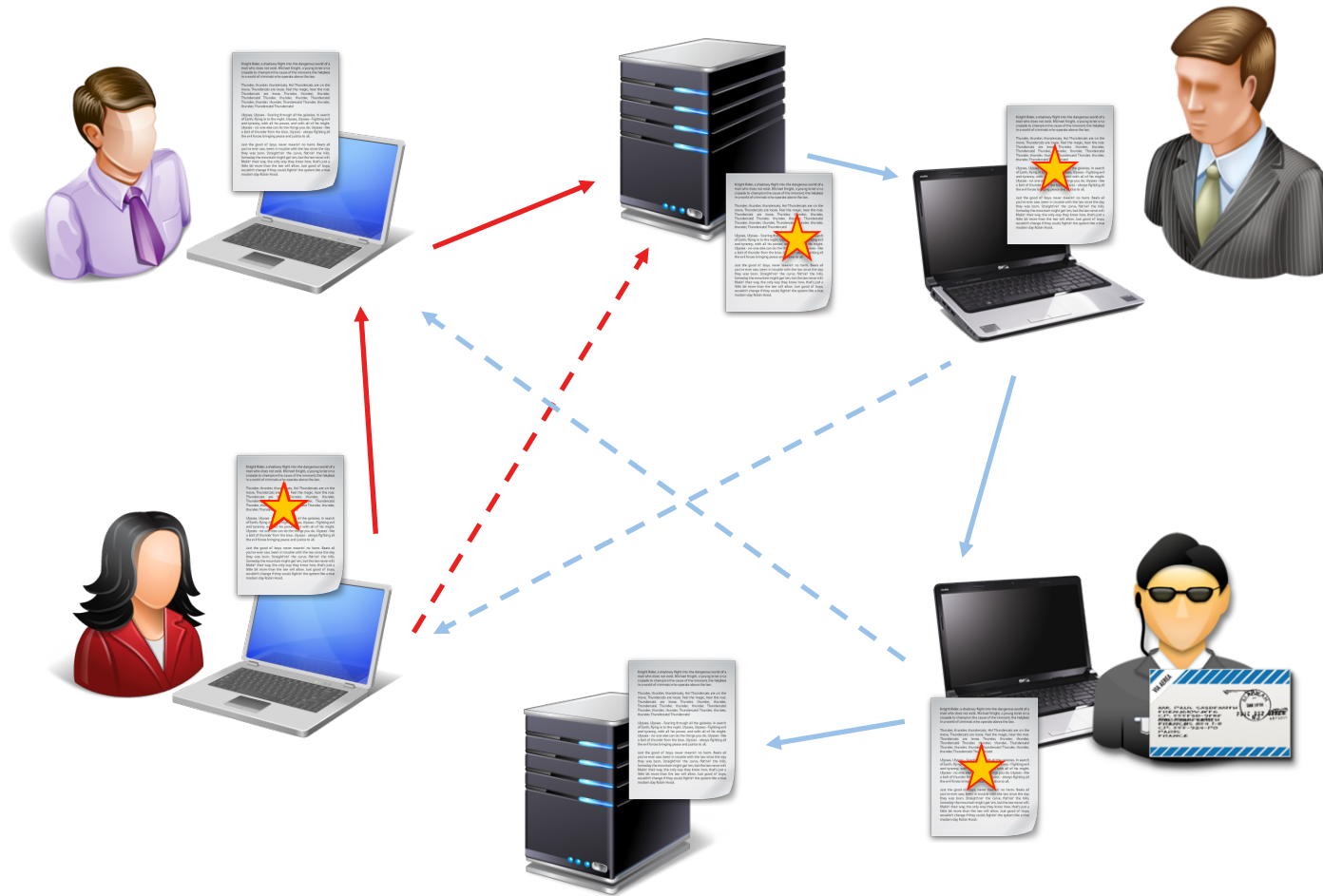
Limitations of Central Authority Systems

SCALABILITY

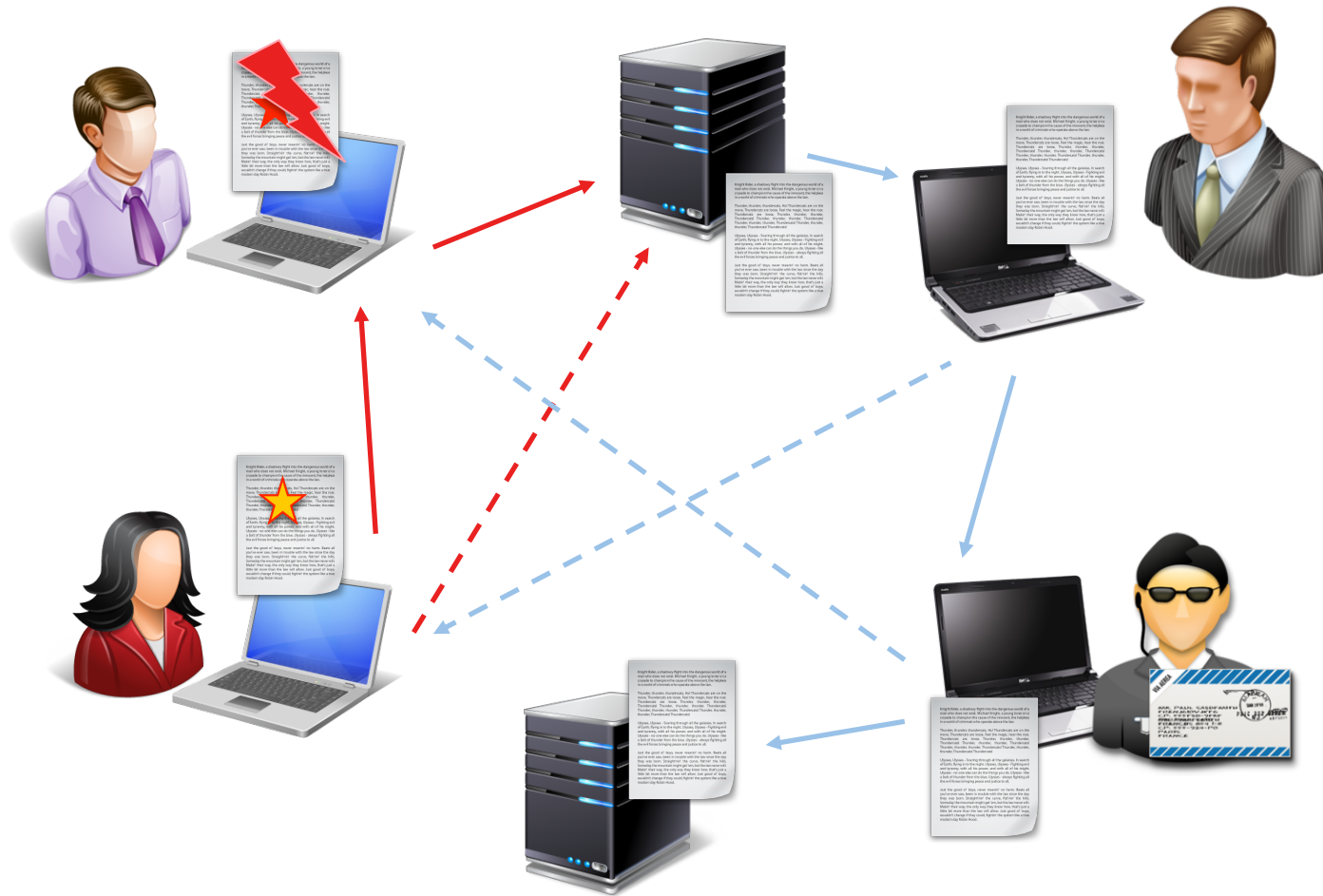
PRIVACY



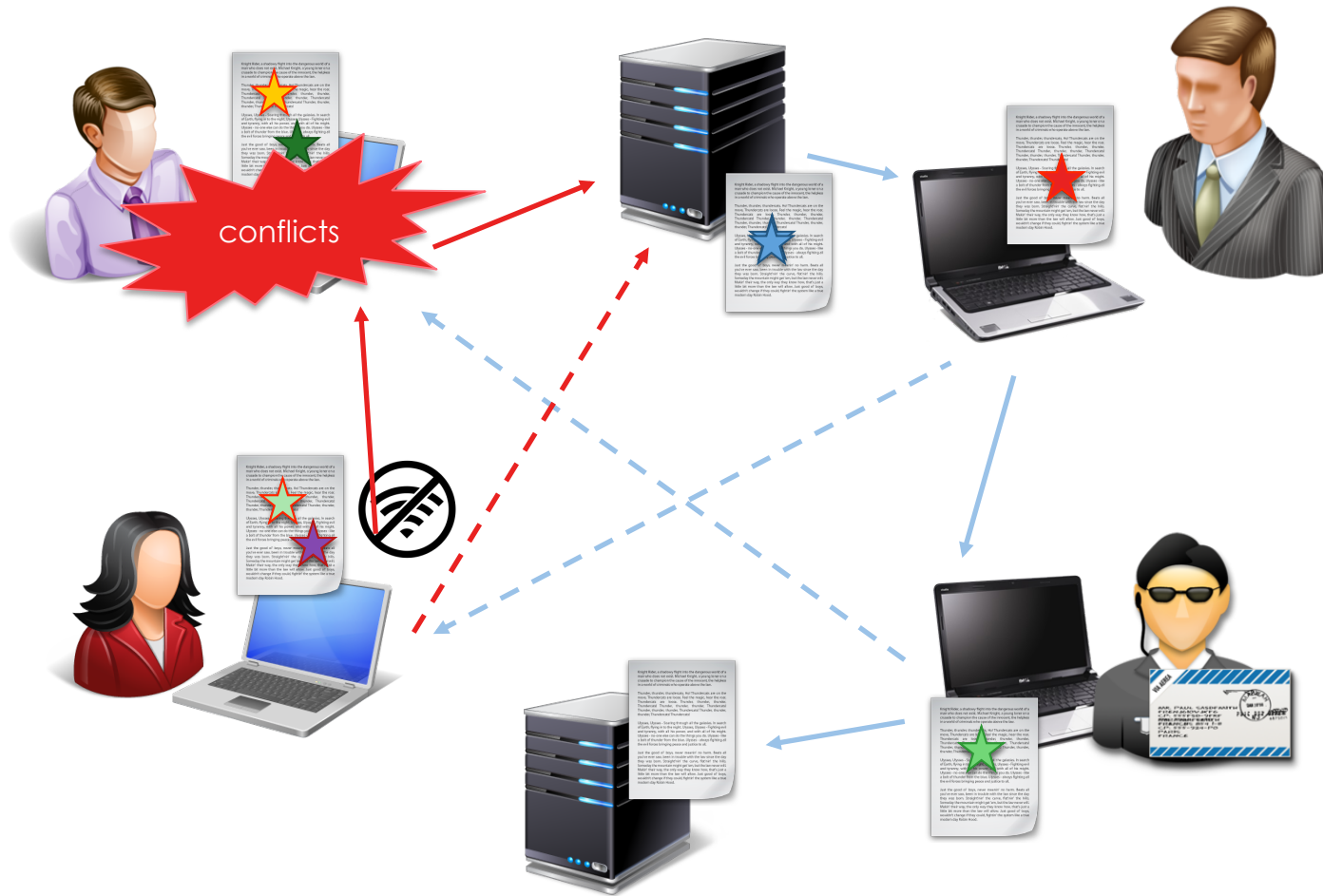
Peer-to-Peer Collaborative Systems



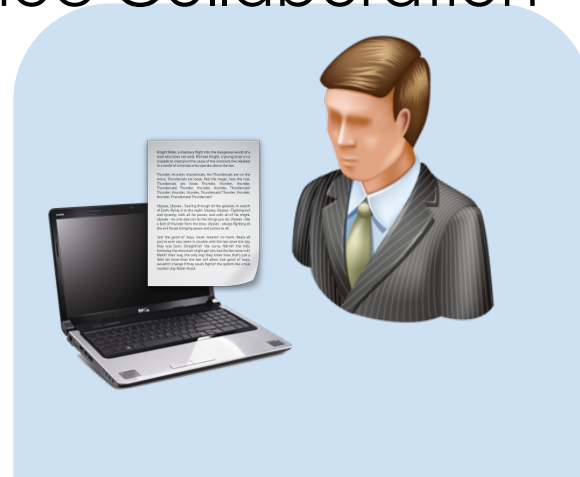
Collaboration Modes – Concurrent Changes



Collaboration Modes – Offline Work



Collaboration Modes – Ad-hoc Collaboration



Research issues

- 1 How to **maintain consistency of different copies** in the face of concurrent modifications?
- 2 How to **evaluate the design of collaborative systems** and approaches?
- 3 How to **secure collaboration data**?

Research issues

- 1 How to **maintain consistency of different copies** in the face of concurrent modifications?
- 2 How to **evaluate the design of collaborative systems** and approaches?
- 3 How to **secure collaboration data**?

Optimistic Replication [SS05]

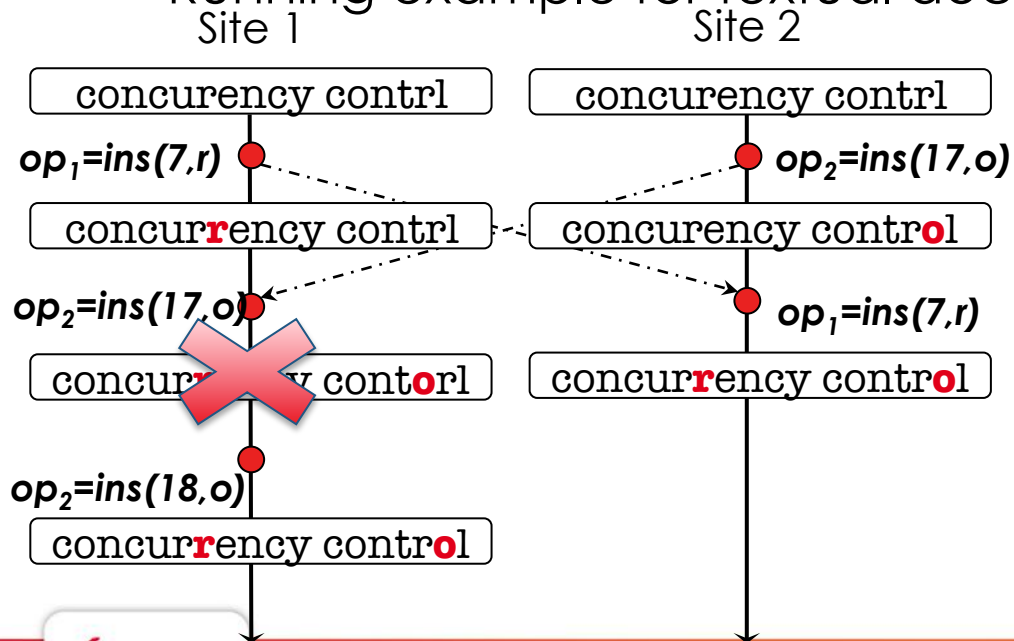
- Trade-off between consistency and availability
 - Optimistic replication : allows replicas to diverge
- Strong Eventual Consistency
 - Eventual delivery: An update executed at some correct replica eventually executes at all correct replicas
 - Strong convergence: Correct replicas that have executed the same updates have equivalent states
 - No consensus in background, no need to rollback
- Intention preservation
 - « *Effect of each operation should be observed on all copies* »

Operational transformation (OT) [EG89]

- n copies of an object hosted at n sites
- An object is modified by applying operations
- Each operation is
 - generated at a site (local execution),
and applied immediately on the local copy
 - broadcasted to other sites
 - integrated at those sites (remote execution)
- System is correct if when it is idle all copies are identical (SEC)

Operational transformation (OT)

- General architecture with two main components:
 - An integration algorithm (diffusion, integration)
 - A set of transformation functions (conflict resolution)
- Running example for textual document = sequence of characters



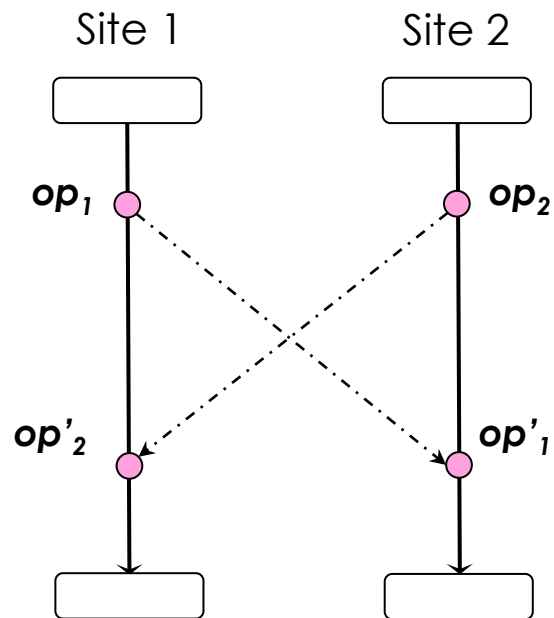
- Operations:
 - $ins(p, c)$
 - $del(p)$

$T(ins(p1, c1), ins(p2, c2)) :-$
 if $(p1 < p2)$ return $ins(p1, c1)$
 else return $ins(p1+1, c1)$
 endif

Operational transformation

Correctness [EG89]

$$(TP1) \quad op_1 \circ T(op_2, op_1) \equiv op_2 \circ T(op_1, op_2)$$



$T(op_2: \text{operation}, op_1: \text{operation}) = op'_2$

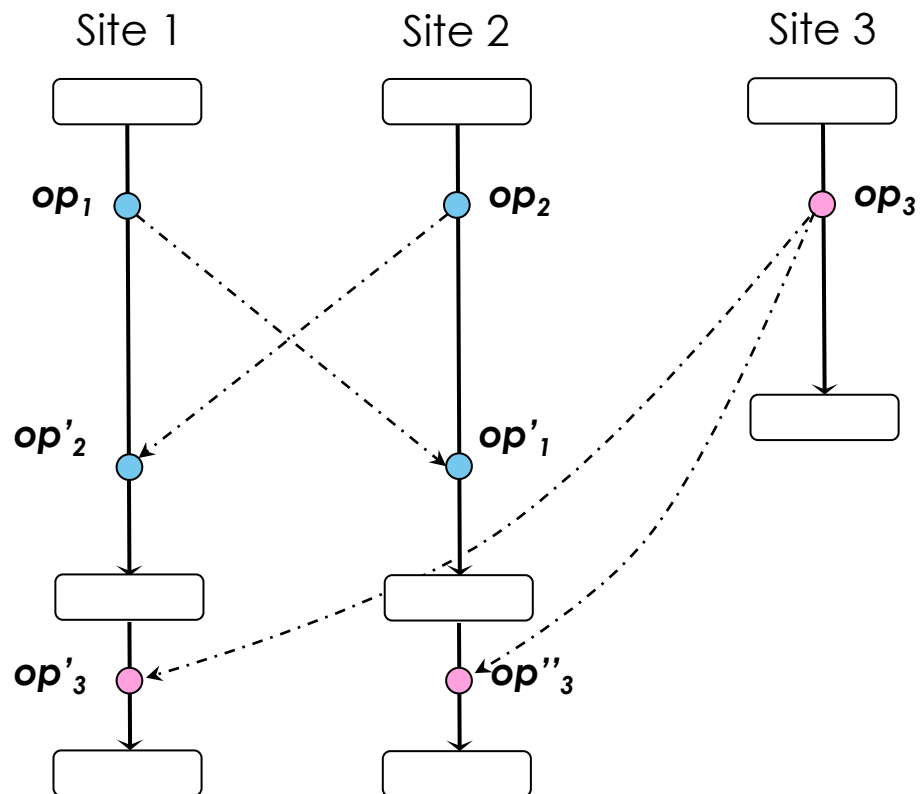
- op_1 and op_2 concurrent, defined on a state S
- op'_2 same effects as op_2 , defined on $S.op_1$



Operational transformation

Correctness [RNG96]

$$(TP2) \quad T(op_3, op_1 \circ T(op_2, op_1)) = T(op_3, op_2 \circ T(op_1, op_2))$$



Operational transformation (OT)

Existing approaches

- Two main families:
 - Transformation functions satisfying both TP1 and TP2: SOCT2 [SCF97] + TTF [OUMI06]
 - Control algorithms avoiding (needs of) TP2: SOCT4 [VCFS00], Jupiter [NCDL95]

Operational transformation (OT)

Summary

- Transforms non commuting operations to make them commute
- Genericity
- Time complexity
 - Average: $O(H c)$ H : #ops
 - Worst case: $O(H^2)$ c : avg. #conc. ops
- Difficult to write correct transformation functions
- State vectors used for detecting concurrency \Rightarrow scalability limitations
- **Not very suitable for large scale peer-to-peer**

collaboration

Inria

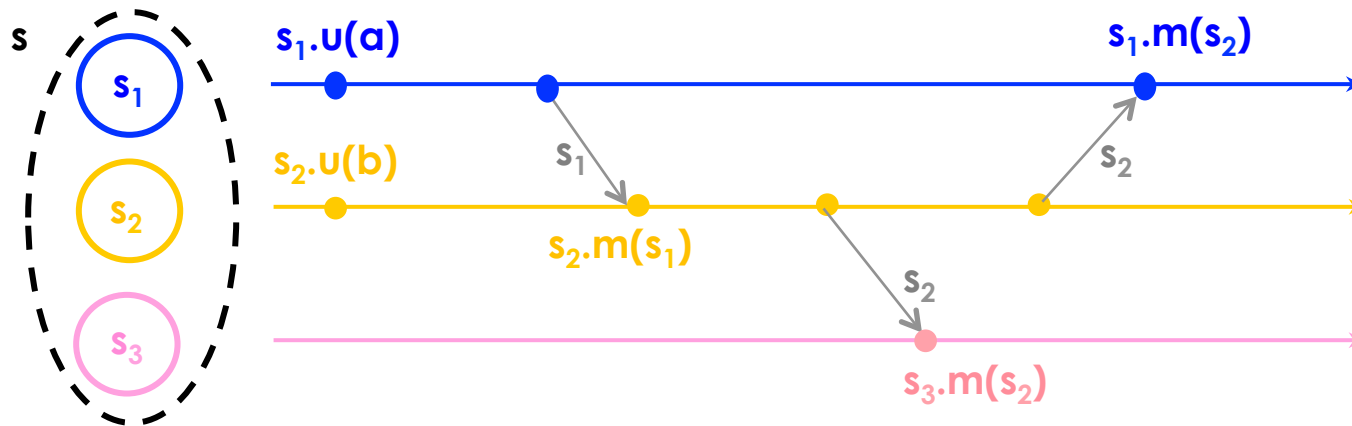
Conflict-free Replicated Data Types (CRDT)

[SPBZ11]

- Design operations to be commutative by construction
- Abstract data types
 - Designed to be replicated at multiple sites
 - Any replica can be modified without coordination
 - State convergence is guaranteed
- State-based and operation-based approaches

Conflict-free Replicated Data Types (CRDT)

State-based Replication



- Algorithm
 - Periodically, replica at p_i sends its current state to p_j
 - Replica p_j merges received state into its local state by executing m
- After receiving all updates (irrespective of order), each replica will have same state

Conflict-free Replicated Data Types (CRDT)

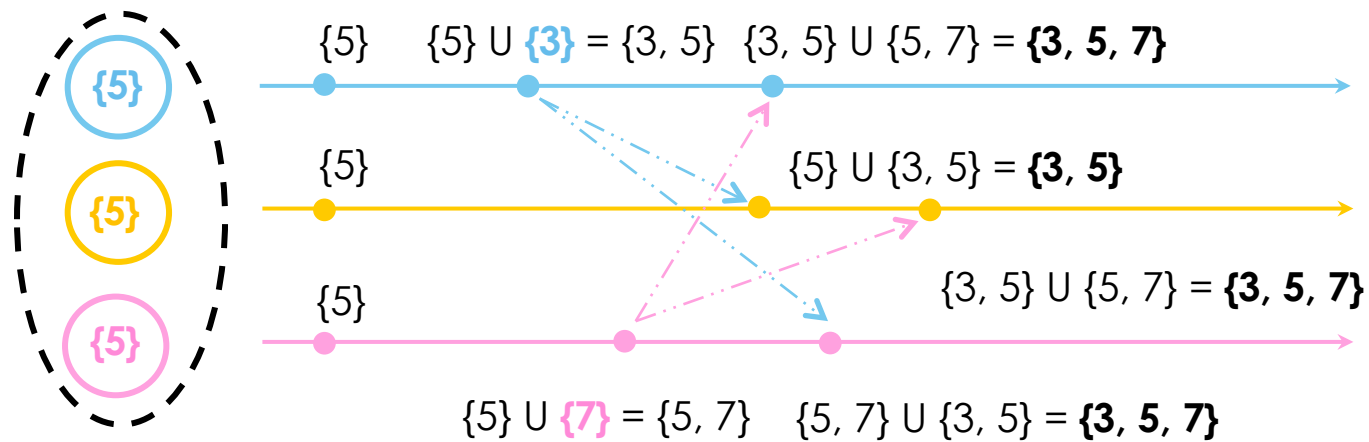
State-based Replication

- Merge operator:
 - **Commutative:** $x \bullet y = y \bullet x$
 - **Associative:** $(x \bullet y) \bullet z = x \bullet (y \bullet z)$
 - **Idempotent :** $x \bullet x = x$
- A semi-lattice is a Partial order \leq set S with a least upper bound (LUB), denoted \sqcup
 - $m = x \sqcup y$ is a LUB of $\{x, y\}$ under \leq if and only if $\forall m', x \leq m' \wedge y \leq m' \Rightarrow x \leq m \wedge y \leq m \wedge m \leq m'$
 - It follows that \sqcup is commutative, associative and idempotent

Conflict-free Replicated Data Types (CRDT)

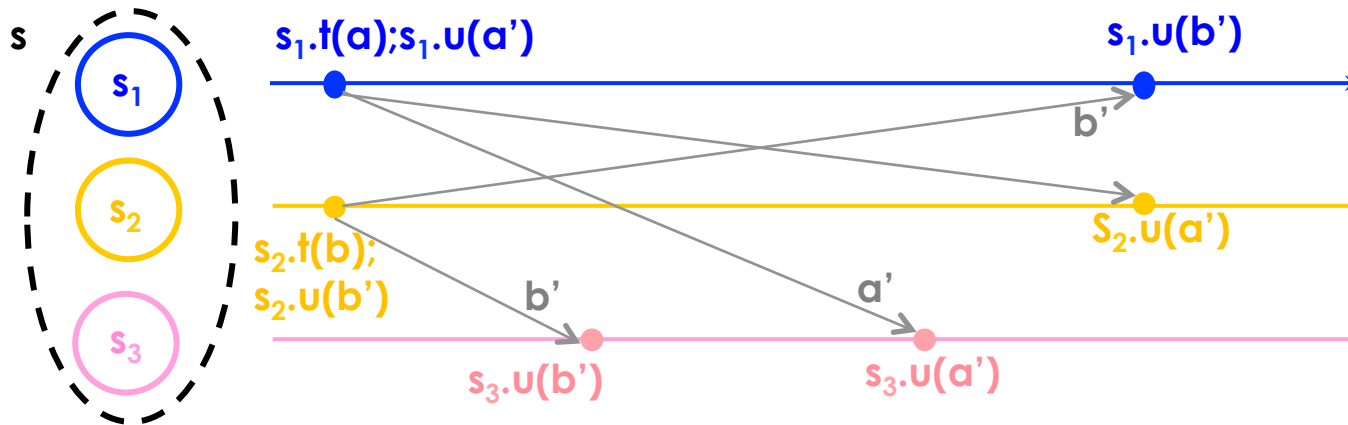
Convergent Replicated Data Type (CvRDT)

- Example



Conflict-free Replicated Data Types (CRDT)

Operation-based Replication

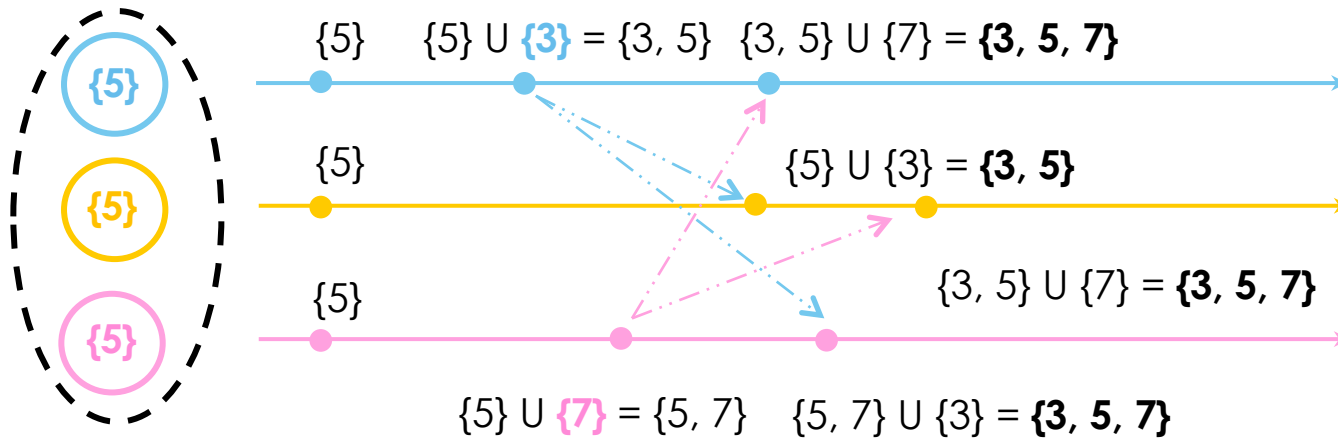


- An update split into (t,u) : t is a side-effect-free prepare-update method and u is an effect-update method
- Algorithm
 - Updates delivered to all replicas
 - Causally-ordered broadcast, every message delivered to every node exactly once w.r.t. happen-before order
- Commutativity holds for concurrent updates

Conflict-free Replicated Data Types (CRDT)

Commutative Replicated Data Type (CmRDT)

- Example



Conflict-free Replicated Data Types (CRDT)

CvRDT vs. CmRDT

- Both approaches are equivalent
 - A state-based object can emulate an operation-based object, and vice-versa
- Operation-based:
 - More efficient since you only ship small updates
 - But require exactly once causally-ordered broadcast
- State-based:
 - Only require reliable broadcast
 - Communication overhead of shipping the whole state
- Delta State-based [ASB18]:
 - Small messages
 - Dissemination over unreliable communication channels

Consistency Maintenance

Conflict-free Replicated Data Types (CRDT)

- Register
 - Last-Writer Wins
 - Multi-Value
- Set
 - Grow-Only
 - 2-Phase
 - Observed-Remove
 - Observed-Update-Remove

- Map
- Counter
- Graph
 - Directed
 - Monotonic DAG
 - Edit graph
- **Sequence**



Conflict-free Replicated Data Types (CRDT) (Text) Sequence [PMSL09] [WUM09]

- Document = linear sequence of elements
 - Each element has a unique identifier
 - Identifier constant for the lifetime of the document
 - Dense total order of identifiers consistent with element order:
 - $\forall id_x, id_y: id_x < id_y \Rightarrow \exists id_z: id_x < id_z < id_y$
- Different approaches for generating identifiers:
 - TreeDoc, Logoot, LogootSplit, ...

Conflict-free Replicated Data Types (CRDT)

Logoot [WUM09]

- Logoot identifiers: $\langle p_1, s_1, h_1 \rangle \langle p_2, s_2, h_2 \rangle \dots \langle p_k, s_k, h_k \rangle$

p_i integer

s_i site identifier

h_i logical clock at site s_i

$\langle 1, 2, 1 \rangle$	c
$\langle 1, 2, 2 \rangle$	o
$\langle 2, 1, 2 \rangle$	n
$\langle 3, 1, 3 \rangle$	c
$\langle 3, 1, 3 \rangle \langle 8, 4, 5 \rangle$	u
$\langle 3, 2, 5 \rangle$	r
$\langle 4, 1, 7 \rangle$	e
$\langle 4, 1, 7 \rangle \langle 9, 2, 6 \rangle$	n
$\langle 7, 2, 8 \rangle$	c
$\langle 9, 1, 7 \rangle$	y
$\langle 10, 2, 8 \rangle$	
$\langle 12, 3, 1 \rangle$	c
$\langle 12, 3, 1 \rangle \langle 6, 5, 1 \rangle$	o
$\langle 12, 3, 1 \rangle \langle 7, 8, 2 \rangle$	n
$\langle 12, 3, 1 \rangle \langle 7, 8, 2 \rangle \langle 12, 3, 5 \rangle$	t
$\langle 12, 3, 1 \rangle \langle 7, 8, 2 \rangle \langle 13, 3, 6 \rangle$	r
$\langle 12, 3, 1 \rangle \langle 7, 8, 2 \rangle \langle 14, 3, 7 \rangle$	l

ins($\langle 3, 2, 5 \rangle \langle 13, 1, 7 \rangle$, r)

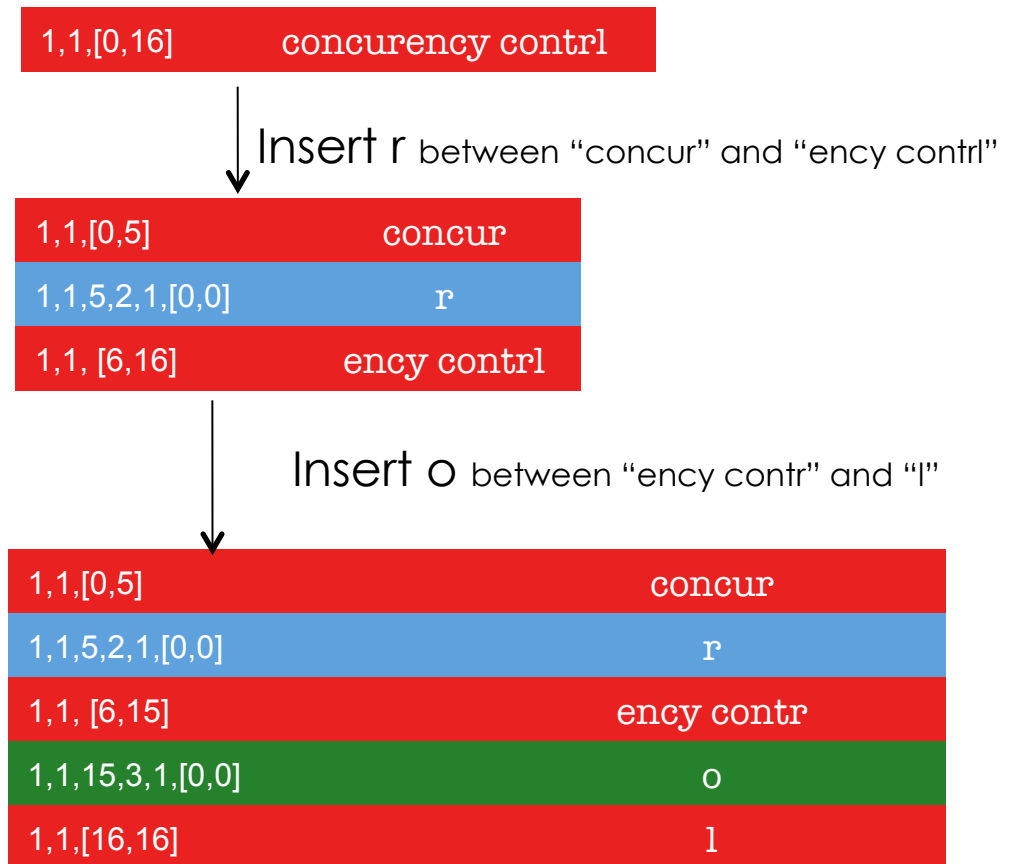
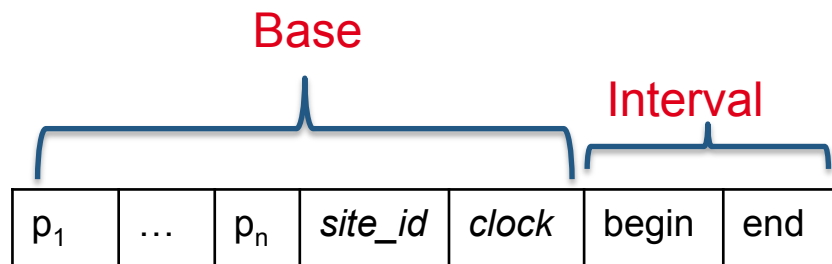
ins($\langle 12, 3, 1 \rangle \langle 7, 8, 2 \rangle \langle 13, 3, 6 \rangle \langle 7, 2, 9 \rangle$, o)

- Time complexity
 - Average: $O(k \log(n))$
 - Worst case: $O(H * \log(H))$
 - H: #ops
 - n: doc. size (non deleted chars.)
 - k: avg. size of Logoot identifier
- No need for concurrency detection
- Identifiers storage cost
- New design for each data type
- Suitable for large-scale collaboration**

Conflict-free Replicated Data Types (CRDT)

LogootSplit [AMO113]

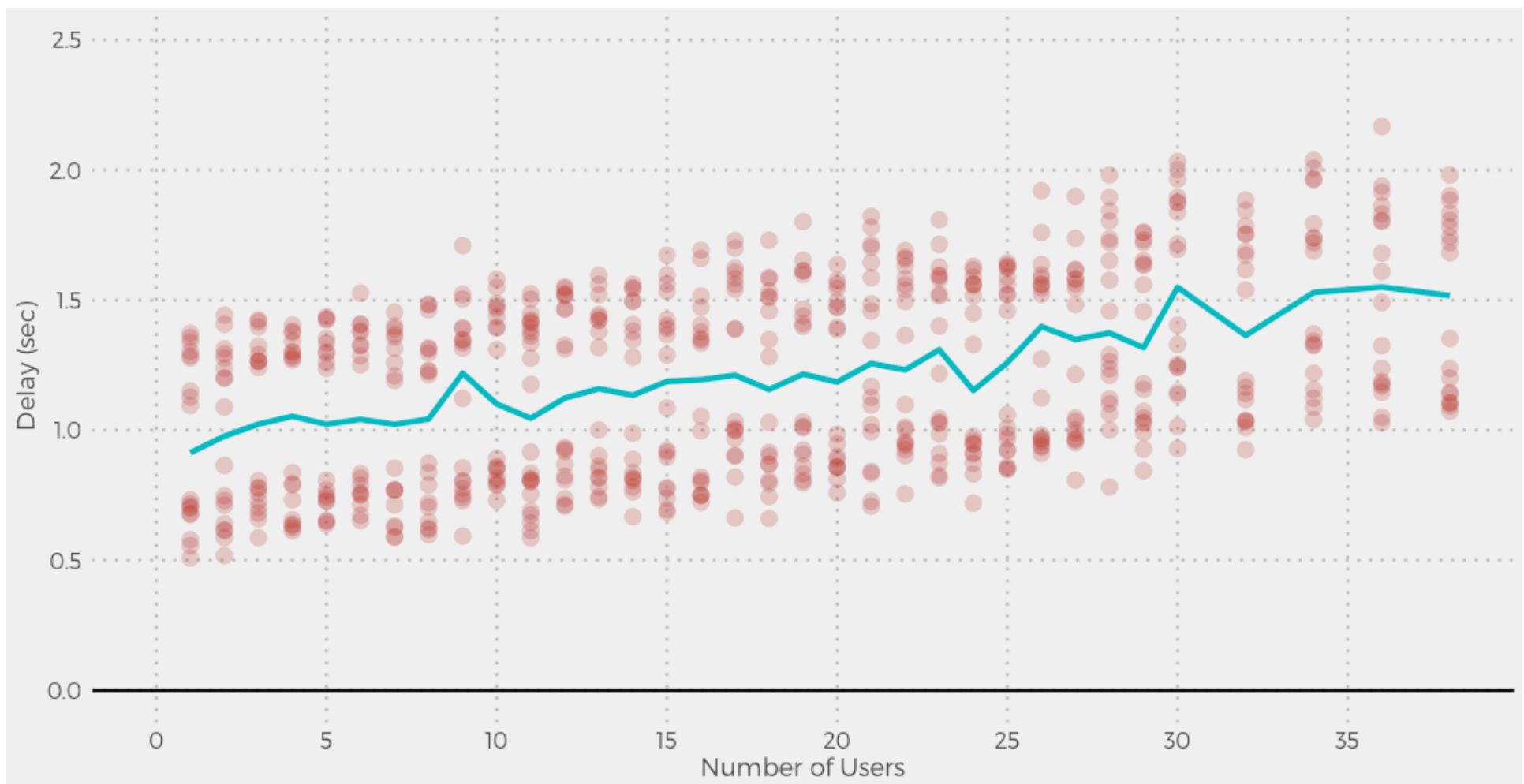
LogootSplit identifiers



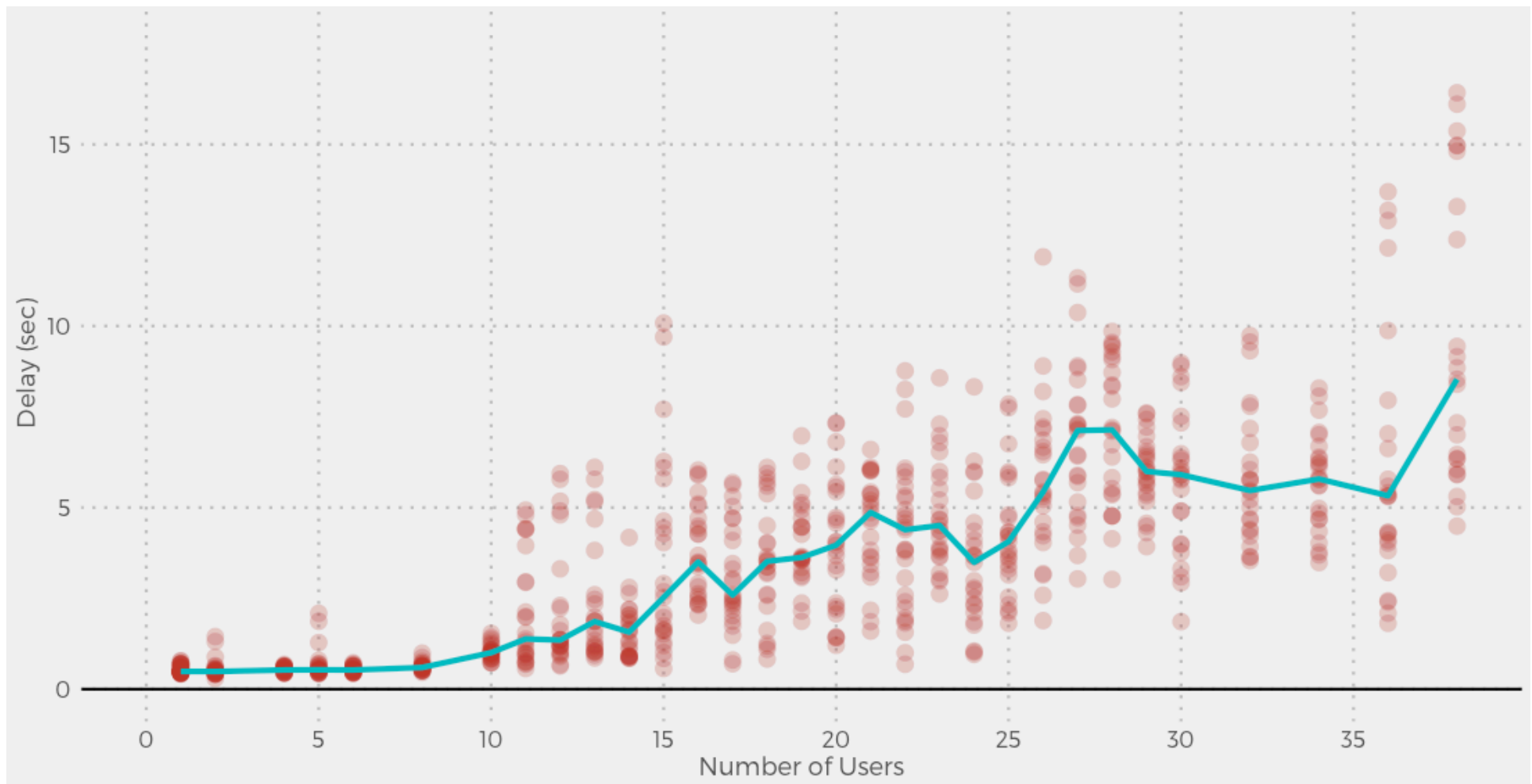
OT vs. operation-based CRDT

- CRDT: more formalised approach
- OT: more generic and guided
 - Generic concurrency control algorithm
 - Operation transformations specific to application domain
- CRDT: different solutions for concurrency handling for different data types
- CRDT: Metadata overhead

Delays in MUTE [NEOIC17] <https://coedit.re/>



Delays in GoogleDocs [DI16]

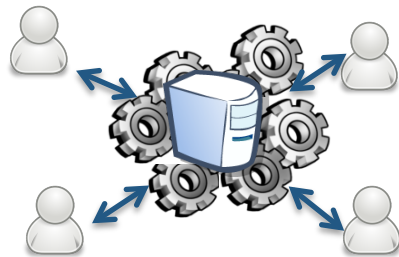


Research issues

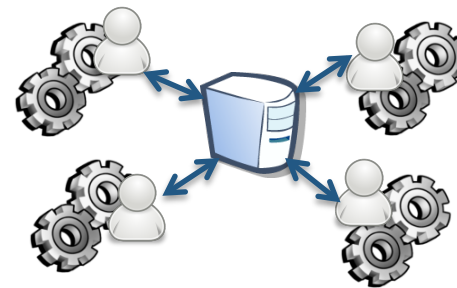
- 1 How to **maintain consistency of different copies** in the face of concurrent modifications?
- 2 How to **evaluate the design of collaborative systems** and approaches?
- 3 How to **secure collaboration data**?

User Study: The effect of delay on users

- Delays in seeing modifications of other users
 - Network delay
 - Time complexity of consistency maintenance algorithms
 - Types of architecture



Thin client architecture



Thick client architecture

- How does **delay influence group performance?**

Experiment design

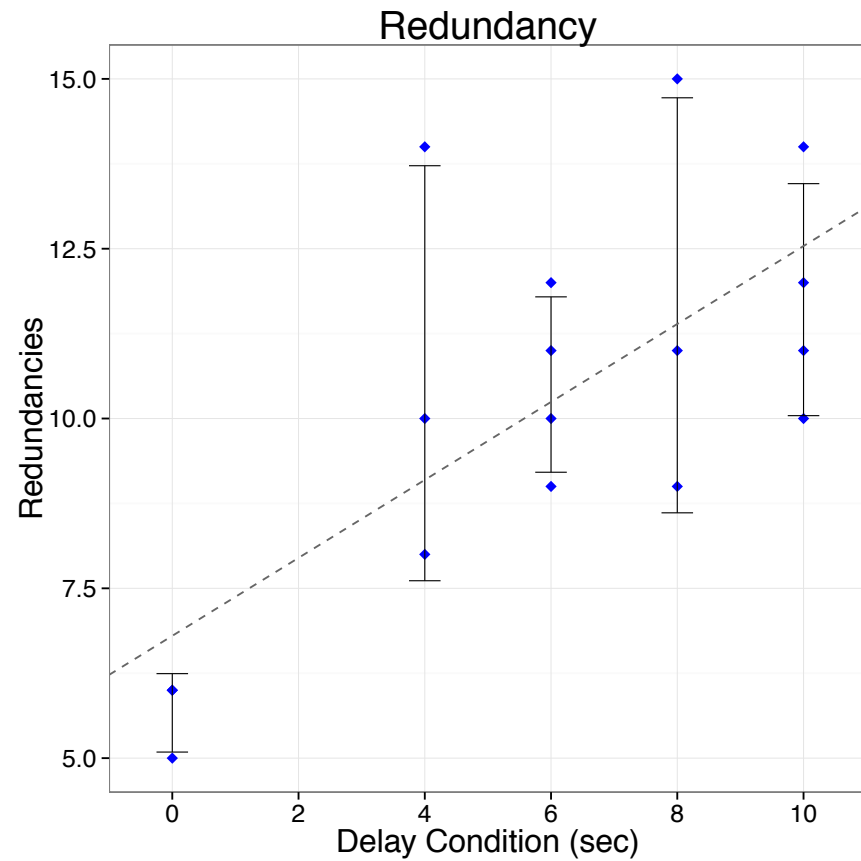
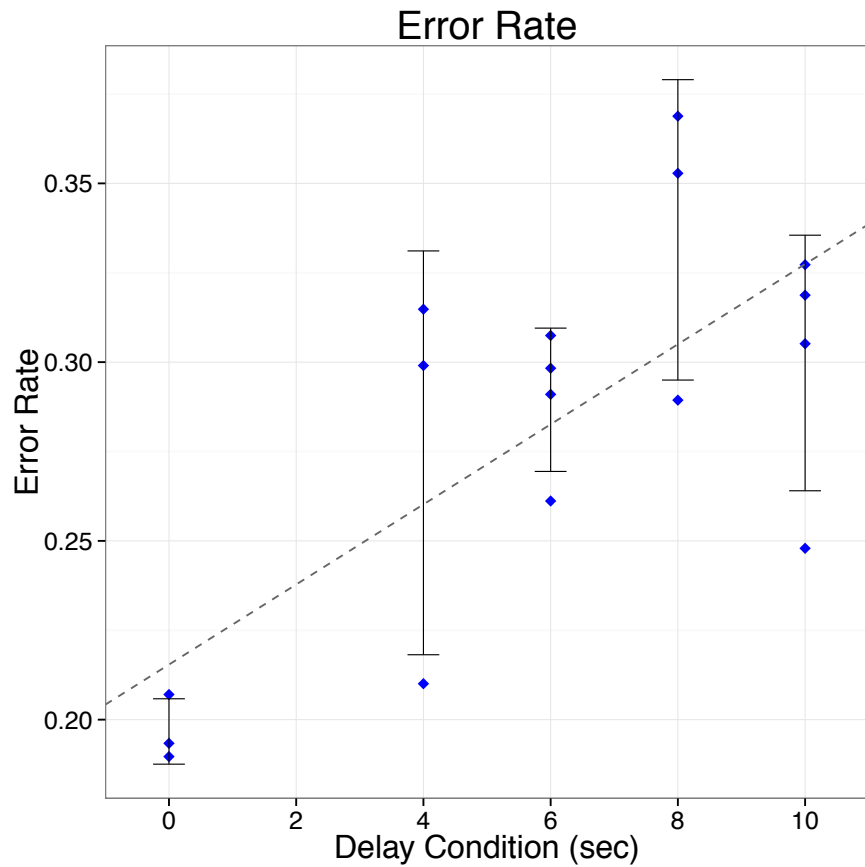
- 20 groups of 4 students
 - Perform several collaborative editing tasks
 - A proofreading task
 - A sorting task
 - **A note taking task**
 - Use the provided collaborative editor (Etherpad) + chat
 - Each group experienced a **certain delay** (0, 4, 6, 8, 10 s)
- Registration of user keyboard inputs
- Video recording of user activities on desktop

Note-taking [IOFSC15]

The image shows a screenshot of a web browser displaying a note-taking application. The browser's address bar shows the URL `ec2-184-72-75-76.compute-1.amazonaws.com/p/notes005`. The note content is a list of points about cloud computing, with several lines highlighted in purple, blue, and yellow. A black box with the text "Editing zone" is overlaid on the first few lines of the note. Three circular callouts are drawn around specific words: "matérialisé", "demateriali", and "matérialisé". A black box with the text "Chat dialogue" is overlaid on the right side of the screen, pointing to a chat window. The chat window shows a list of messages:

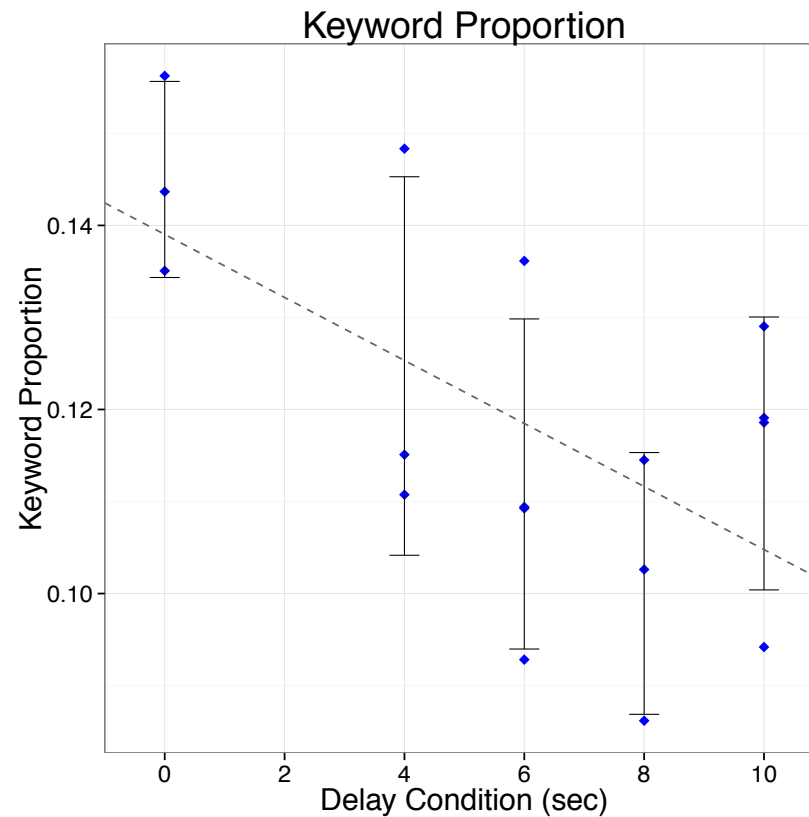
Chat	
user4: test	16:15
user2: test user 2	16:15
user2: great	16:15
user3: test	16:15

Delay reduces Group Performance



- Delay increases error rate and redundancy

Delay reduces Group Performance



- Delay decreases proportion of keywords

Design implications

- Reduce the delay by the choice of the architecture and synchronisation algorithms
- Make users aware of existing delays such that they can compensate for the delay by coordination strategies
- Analyse real collaboration traces to understand collaboration patterns and behavior [NI18]

Research issues

- 1 How to **maintain consistency of different copies** in the face of concurrent modifications?
- 2 How to **evaluate the design of collaborative systems** and approaches?
- 3 How to **secure collaboration data**?

Security in peer-to-peer collaboration



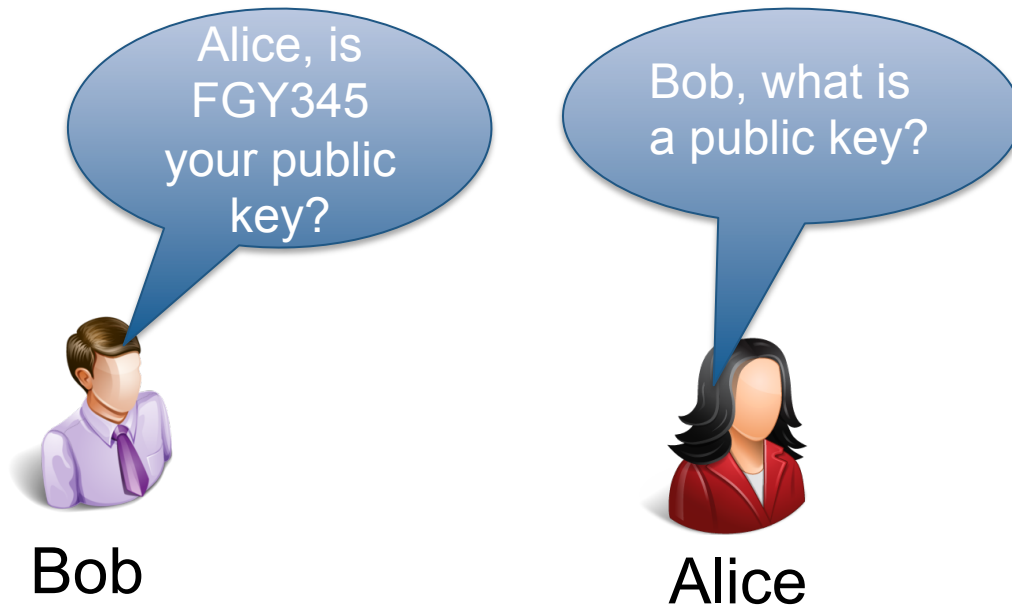
- How to learn and verify the other party's key ?
- Trust-based access control

Trust establishment

- How to learn and verify the other party's key before establish a secure communication channel ?
 - Out of band trust establishment
 - Trust establishment by the provider

Out of band trust establishment

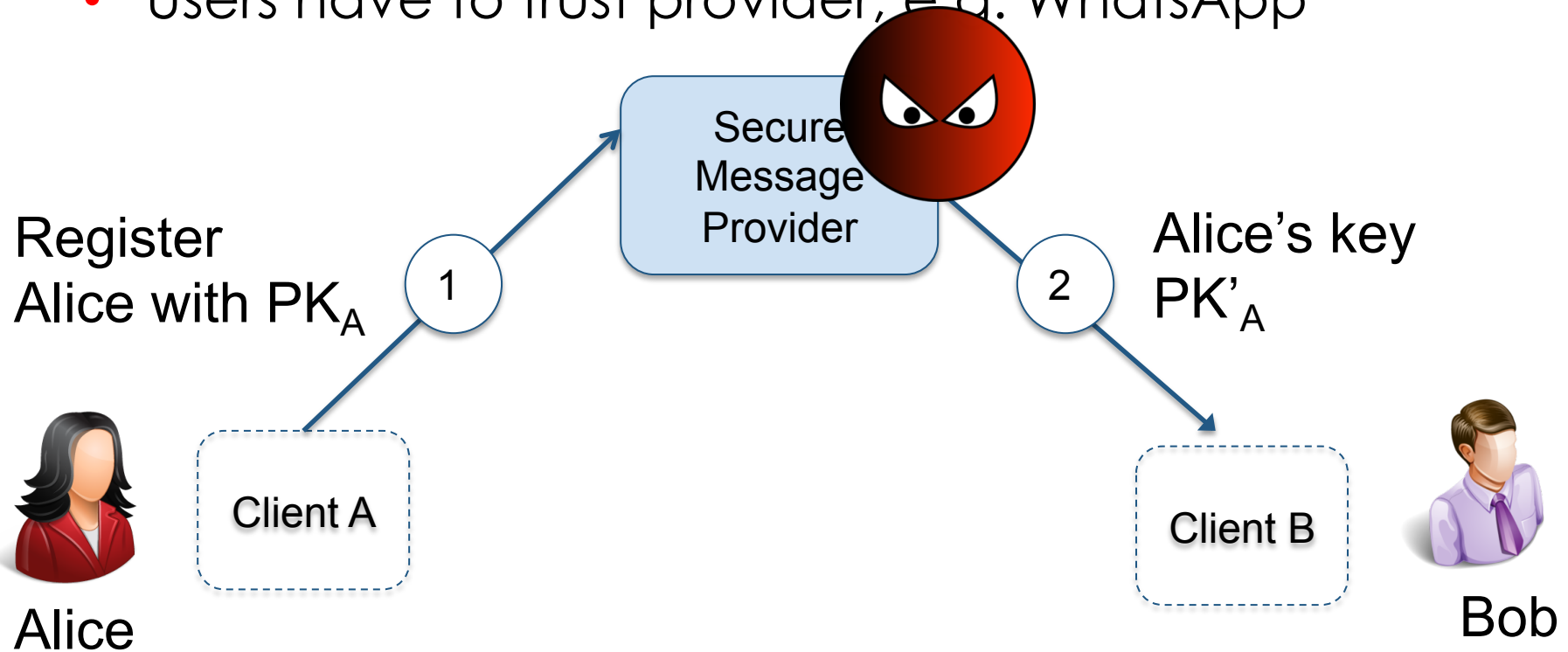
- Unintuitive, error-prone



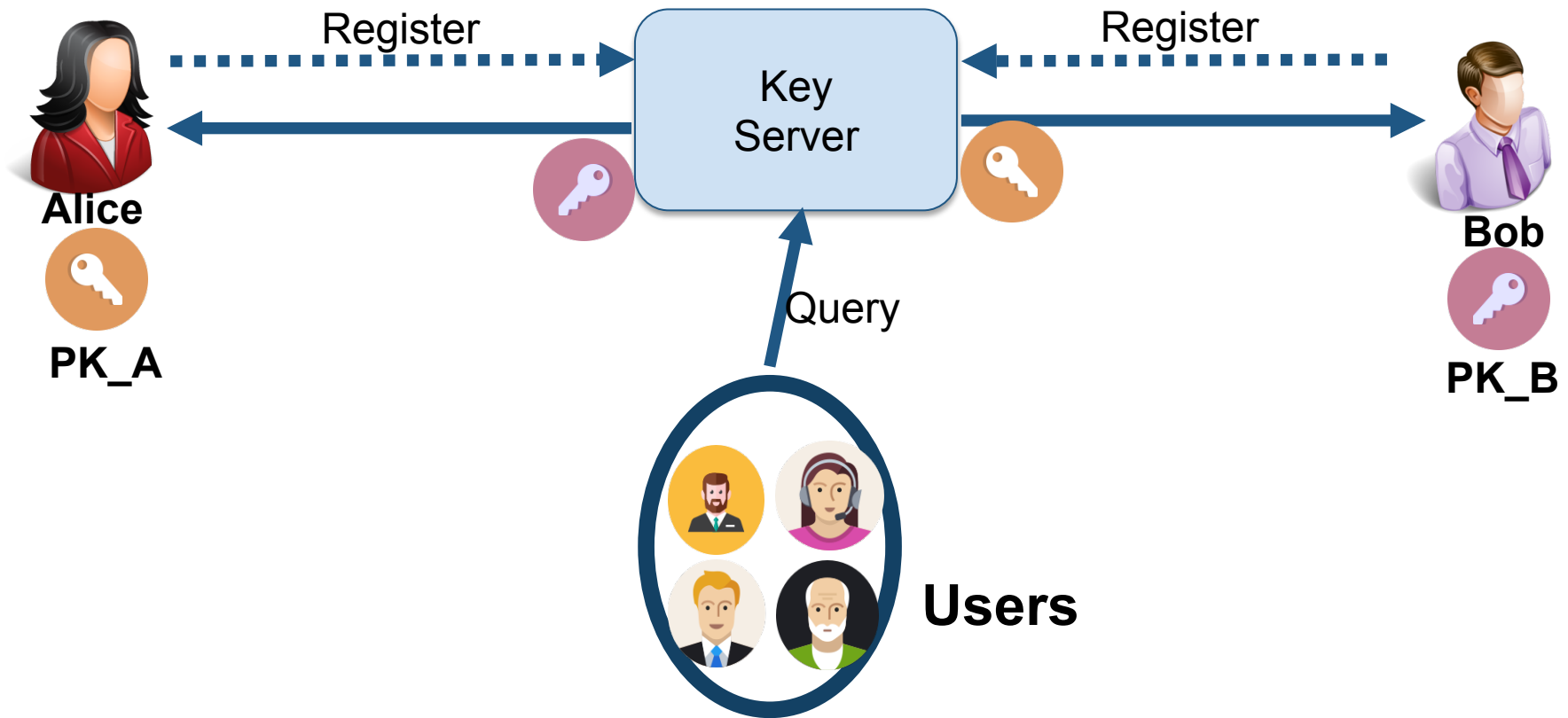
Trust establishment by the provider

Centralized key server

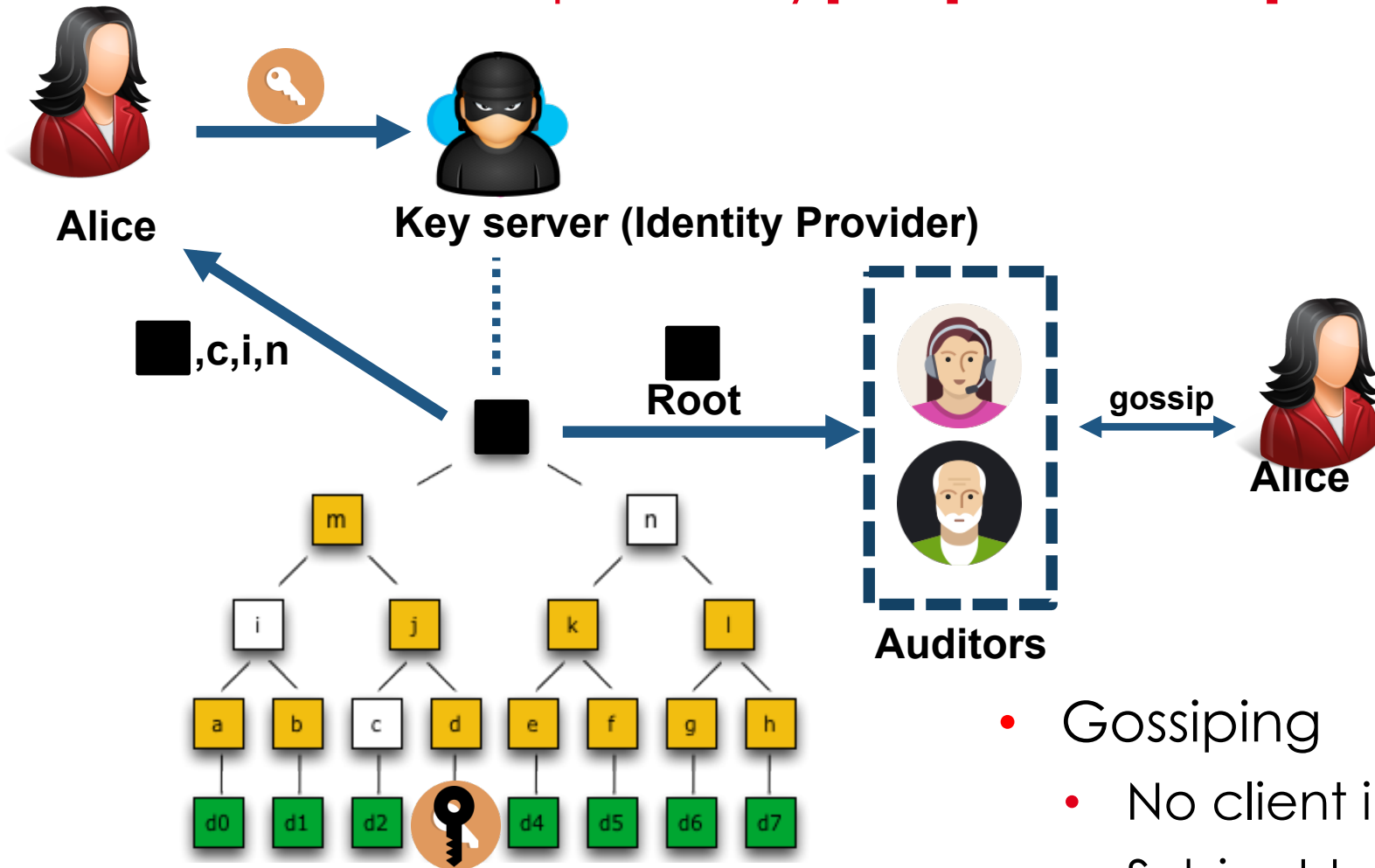
- Clients query providers for keys of other users
- Users have to trust provider, e.g. WhatsApp



Transparent log

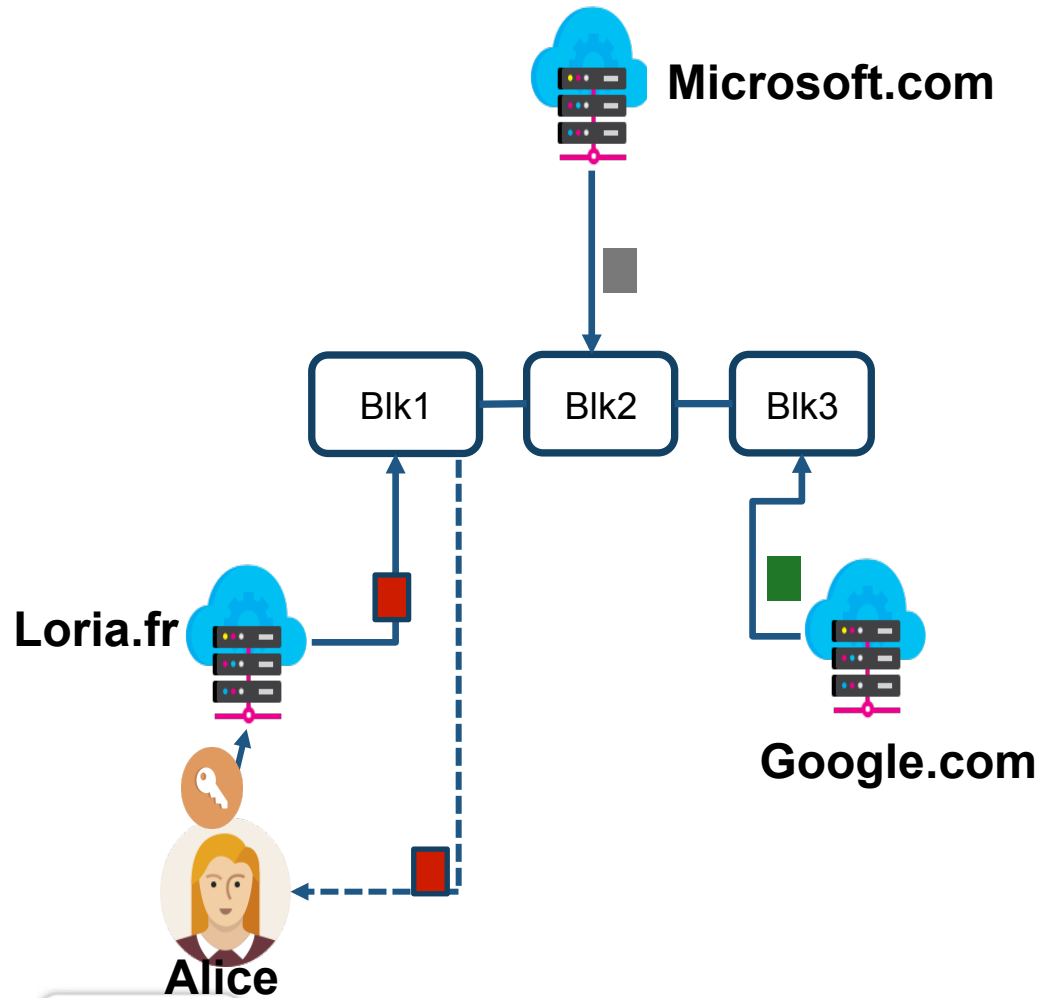


Certificate transparency[L14]/CONIKS [MBBFF15]



- Gossiping
 - No client incentive
 - Subject to Sybil and Eclipse attacks

Trusternity: Blockchain-based Auditing of Transparent Log Servers [NEIP18]

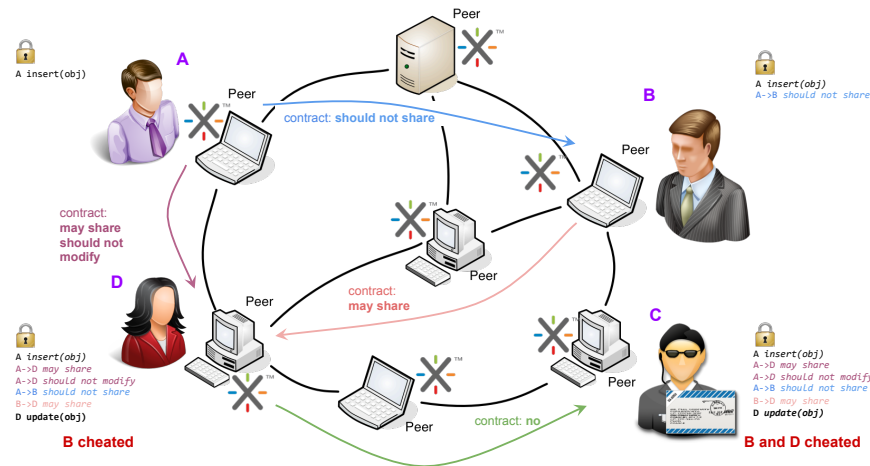


Trust-based access control

- Dynamic trust values among users
- How to **define an access control based on trust** and how to **compute trust based on collaborative experience?**

Trust computation

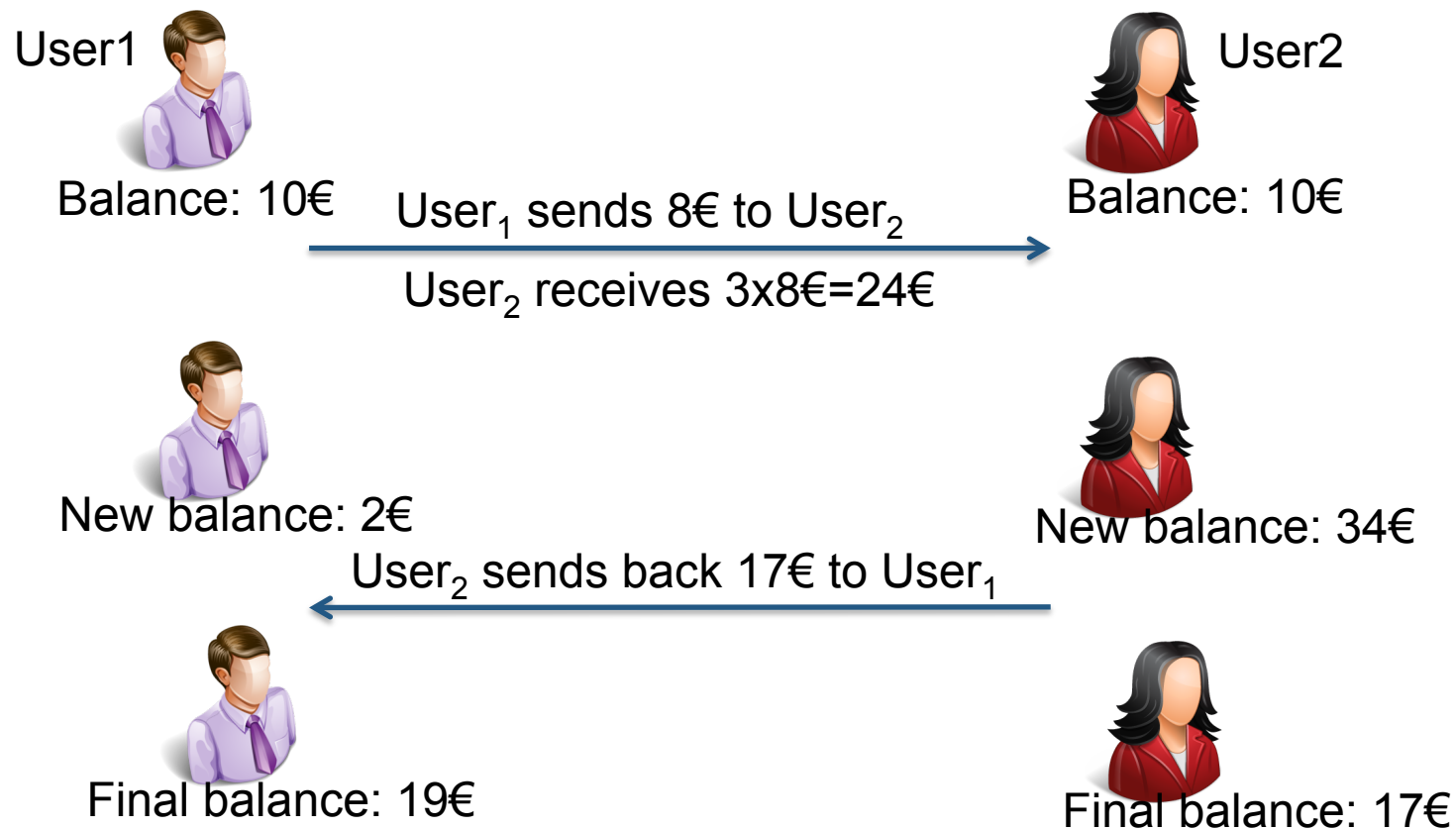
- Respect/Violation of contracts
 - Contracts in collaborative editing (share, edit)



- Reporting of fake news in Facebook
- Quality of user contributions

Validation of trust-based collaboration

- Using game theory (trust game) [BDM95]



Validation of trust-based collaboration

- Proposal of a trust metric reflecting user behavior [DI16]
- User studies on various trust game variations
 - Trust can replace knowing the identity of collaborators
 - People take into account the trust value of the partner in their future collaboration

Large-scale trustworthy distributed collaborative systems

- New uses and new practices due to large scale adoption
- New challenges
 - Consistency of replicated data
 - User studies
 - Trust and Security

References

- **[EGR91]** Groupware: Some issues and experiences, C.A. Ellis, S. J. Gibbs and G. Rein, Communications of the ACM, 1991
- **[J88]** GroupWare: Computer Support for Business Teams. R. Johansen. The Free Press, New York, NY, USA, 1988.
- **[SG88]** Computer-based real-time conferencing systems. Sunil Sarin and Irene Greif. Morgan Kaufmann Publishers Inc., 1988, page 397–422.
- **[G90]** Sharing views and interactions with single-user applications. Saul Greenberg. ACM SIGOIS and IEEE CS TC-OA conference on Office information systems, 1990.
- **[SFBKLS88]** Beyond the chalkboard: computer support for collaboration and problem solving in meetings. Mark Stefik, Gregg Foster, Daniel G. Bobrow, Kenneth Kahn, Stan Lanning, and Lucy Suchman. Morgan Kaufmann Publishers Inc., 1988, page 335–366

References

- **[EG89]** Concurrency Control in GroupWare Systems, C.A. Ellis and S.J. Gibbs. ACM SIGMOD Record 18(2), 1989.
- **[SS05]** Optimistic replication. Y. Saito and M. Shapiro. ACM *Computing Surveys*. 37(1), 2005.
- **[RNG96]** An integrating, transformation-oriented approach to concurrency control and undo in group editors, M. Ressel, D. Nitsche-Ruhland, and R. Gunzenhäuser. CSCW 1996.
- **[OUMI06]** Tombstone transformation functions for ensuring consistency in collaborative editing systems, G. Oster, P. Urso, P. Molli, and A. Imine. CollaborateCom 2006.
- **[SCF97]** Serialization of concurrent operations in a distributed collaborative environment, M. Suleiman, M. Cart, and J. Ferrié. GROUP 1997.

References

- **[VCFS00]** Copies convergence in a distributed real-time collaborative environment. N. Vidot, M. Cart, J. Ferrié, and M. Suleiman. CSCW 2000.
- **[NCDL95]** High-latency, low-bandwidth windowing in the Jupiter collaboration system, D. A. Nichols, P. Curtis, M. Dixon, and J. Lamping. UIST 1995.
- **[SPBZ11]** Conflict-free Replicated Data Types, M. Shapiro, N. Preguica, C. Baquero, M. Zawirski. Research Report, RR-7687, INRIA, 2011
- **[PMSL09]** A commutative replicated data type for cooperative editing, N. Preguica, M. Joan, M. Shapiro, and M. Letia. ICDCS 2009.
- **[WUM09]** Logoot : a Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks, S. Weiss, P. Urso and P. Molli. ICDCS 2009.

References

- **[AMOI13]** Supporting Adaptable Granularity of Changes for Massive Scale Collaborative Editing, L. André, S. Martin and G. Oster C.-L. Ignat. CollaborateCom 2013.
- **[ASB18]** Delta state replicated data types. Paulo Sérgio Almeida, Ali Shoker, Carlos Baquero. Journal of Parallel and Distributed Computing, 2018
- **[NEOIC17]** MUTE: A Peer-to-Peer Web-based Real-time Collaborative Editor. Matthieu Nicolas, Victorien Elvinger, Gérald Oster, Claudia-Lavinia Ignat, François Charoy. ECSCW, 2017
- **[DI16]** Performance of real-time collaborative editors at large scale: User perspective. Quang Vinh Dang, Claudia-Lavinia Ignat. Networking 2016

References

- **[IOFSC15]** How Do User Groups Cope with Delay in Real-Time Collaborative Note Taking. Claudia-Lavinia Ignat, Gérald Oster, Olivia Fox, Valerie L. Shalin, François Charoy. ECSCW 2015
- **[NI18]** An Analysis of Merge Conflicts and Resolutions in Git-Based Open Source Projects. Hoai Le Nguyen, Claudia-Lavinia Ignat. Journal of Computer Supported Cooperative Work, 2018
- **[MBBFF15]** CONIKS: Bringing key transparency to end users. M. Melara, A. Blankstein, J. Bonneau, E. W. Felten, M. J. Freedman. *USENIX Security*. 2015.
- **[L14]** Certificate transparency. B. Laurie. Queue 12(8), 2014
- **[NEIP18]** Blockchain-Based Auditing of Transparent Log Servers. Hoang-Long Nguyen, Jean-Philippe Eisenbarth, Claudia-Lavinia Ignat, Olivier Perrin. DBSec 2018

References

- **[BDM95]** Trust, social history, and reciprocity. J. Berg, J. Dickhaut, and K. McCabe, Games and Econ. Behav. 1995.
- **[DI16]** Computational Trust Model for Repeated Trust Games. Quang Vinh Dang, Claudia-Lavinia Ignat. Trustcom 2016

modifications operation-based

documents
management
modification
friend-to-friend
operation-based
work
granularity
conflicts

web-based
push-pull-time
compro
mising
increasing
round-free
securing
development

networks
assessment
peer-to-peer
awareness
environments
peer-to-peer
collaborative
xml

undo
framework
analysis
development

studying
multi-mode
web
cdrts
adaptable
wiki
push-pull
text
history

changes
analysis
xml

repositories
structures

data
contract
tree-based
editor
model
tree
graphical
distributed

real-time
data
user
resolution
grouping
multi-level

algorithms
processes
annotation
concurrent
trust

merging
drawing
editing
algorithm
hi story
multi-part

organisations
auditing
information
reconciliation
recovery
drawing
partitioning
drawing
editing
algorithm
hi story
multi-part

writing
information
log
multi-pair
model
authoring
communication

consistency
changes
software
note
delay
content
evaluating
optimistic
authenticating

wiki
information
log
multi-pair
model
authoring
communication

framework
state-based
privacy
massive-scale
hierarchical
draw-together

documents
systems
multi-level

performance
inter-document
co-authoring

Thank you

COAST Team

<http://team.inria.fr/coast/>