6.104 · Software Design · MIT · Fall 2025

introduction to concept design

Daniel Jackson

your goals for today's class

get the gist of what concepts are about
what they're for, what they are

understand key elements of a concept
purpose, operational principle, states & actions
more on this on Wednesday

understand how concepts are composed
externalizing connections with syncs

appreciate how subtle concept ideas impact innovation
especially the role of killer concepts

a language
for design

When you go to design a house you talk to an architect first, not an engineer. Why is this?

Because the criteria for what makes a good building fall outside the domain of engineering.

Similarly, in computer programs, the selection of the **various components** must be driven by the conditions of use.

How is this to be done? By software designers.

Mitchell Kapor, *A Software Design Manifesto* (1996)

who are the software designers?

UX designers

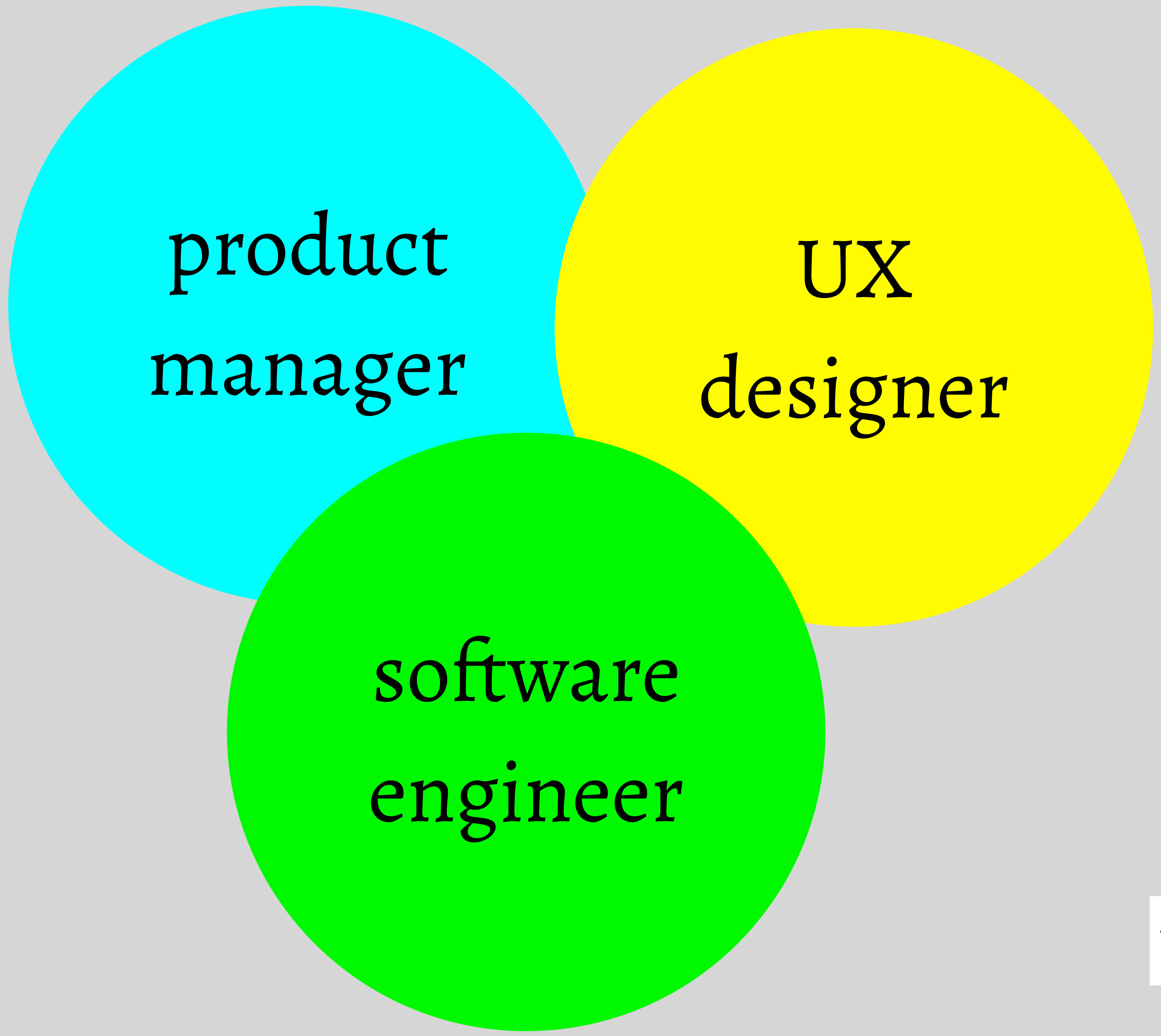
UX architects

product managers

information architects

software engineers

business analysts



"the product triad"

will AI change these roles?

what do designers need?

	modularity	patterns	design principles
for engineering	closures abstract types objects & classes algebraic datatypes microservices	hashtable factory publish/subscribe map/filter client-server	layering decoupling immutability rep independence Liskov substitution
for design	concepts syncs	Upvoting Karma Posting Commenting Bookmarking	completeness separation specificity genericity

what’s a concept?

key elements

Hacker News new | past | comments | ask | show | jobs | submit
login
posting
authentication
upvoting
favoriting
commenting
karma

▲ Jackson structured programming (wikipedia.org)

106 points by haakonhr 63 days ago | hide | past | favorite | 69 comme

▲ danielnicholas 63 days ago [–]

user: danielnicholas
created: 63 days ago
karma: 11

u might find helpful an annotated version [0] of Hoare's explanation of JSP that I edited for a Michael Jackson festschrift

, I'd point to these ideas as worth knowing:

ing problem that involves traversing ures can be solved very systematically. HTDP addresses this class, but bases ode structure only on input structure; JSP synthesized i

- The e archetypal problems that, however you code, can't be pushed under the rug—most notably structure clashes—and just recognizing them

- Coroutines (or code transformation) let you structure code more cleanly when you need to read or write more than one structure. It's why real iterators (with yield), which offer a limited form of this, are (in my view) better than Java-style iterators with a next method.

- The idea of viewing a system as a collection of asynchronous processes (Ch. 11 in the JSP book, which later became JSD) with a long-running process for each real-world entity. This was a notable contrast to OOP, and led to a strategy (seeing a resurgence with event storming for DDD) that began with events rather than objects.

[0] https://groups.csail.mit.edu/sdg/pubs/2009/hoare-jsp-3-29-09...

▲ ob-nix 63 days ago [–]

... this brings back memories! In the late eighties I, as a teenager, found a Jackson Struct. Pr. book at the town library. I remember I was amazed at the text and wondered why I hadn't heard about the method before.

If I remember correctly did the book clearly point out backtracking as a standard method, while mentioning that most languages lacked that, so it had to be implemented manually.

▲ CraigJPerry 63 days ago [–]

This is referenced(1) as a core inspiration in the preface to "How to Design Programs" but i never researched it further because i've found the "design recipes" approach in htdp to be pretty solid in real life problems

so where's the innovation?

hacker news is popular
> 10m page views/day
so evidently it was worth building

where's the innovation?
"combinational creativity" [Boden]
familiar elements combined in new ways

two kinds of innovation
a few tiny concept refinements: post = title + link
concept syncs: no downvote until your own posts upvoted

Y Hacker News new | past | comments | ask | show | jobs | submit
login

▲ Jackson structured progra
106 points by haakonhr 63 days ago

▲ danielnicholas 63 days ago [-

If you want an intro to JSP, … a Michael Jackson festschrift in 2009.

For those who don't know J

- There's a class of progran … ly. HTDP addresses this class, but bases code structure o

- There are some archetyp … ishes—and just recognizing them helps.

- Coroutines (or code trans … structure. It's why real iterators (with yield), which … od.

- The idea of viewing a sys …)) with a long-running process for each real-world entity. This was a notable contrast to OOP, and led to a strategy (seeing a resurgence with event storming for DDD) that began with events rather than objects.

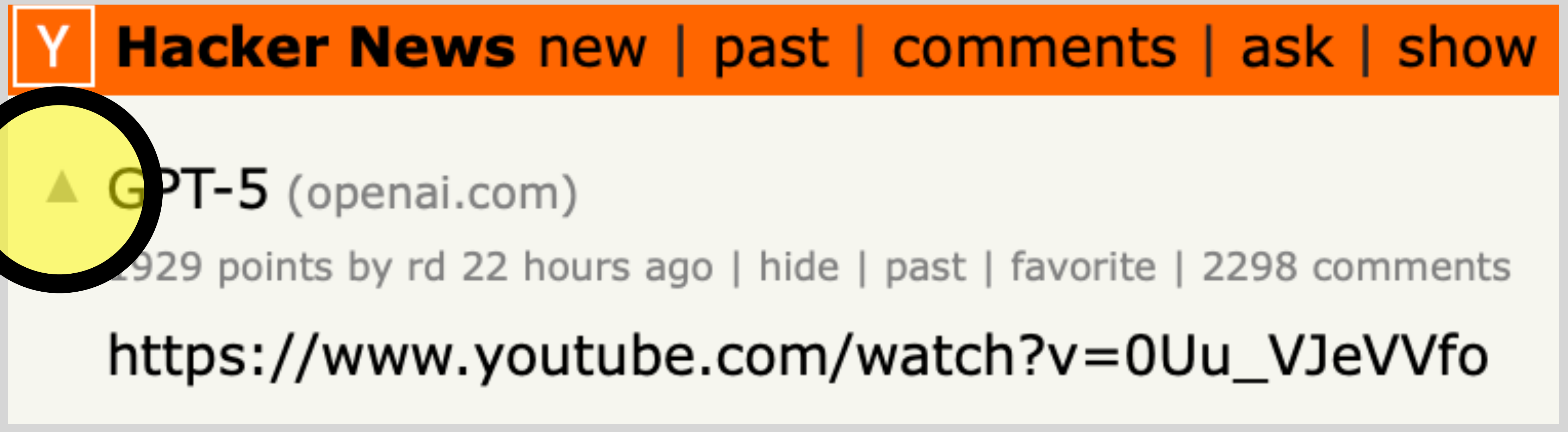
[0] https://groups.csail.mit.edu/sdg/pubs/2009/hoare-jsp-3-29-09...

▲ ob-nix 63 days ago [–]

... this brings back memories! In the late eighties I, as a teenager, found a Jackson Struct. Pr. book at the town library. I remember I was amazed at the text and wondered why I hadn't heard about the method before.

If I remember correctly did the book clearly point out backtracking as a standard method, while mentioning that most languages lacked that, so it had to be implemented manually.

a sample concept: Upvoting

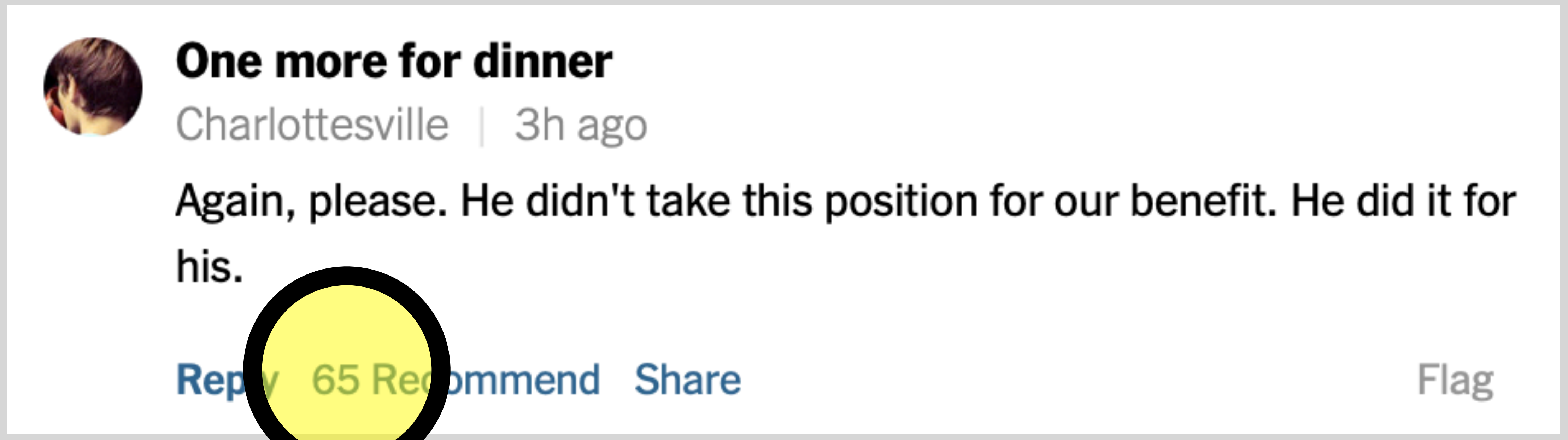


Y **Hacker News** new | past | comments | ask | show

▲ GPT-5 (openai.com)

2929 points by rd 22 hours ago | hide | past | favorite | 2298 comments

https://www.youtube.com/watch?v=0Uu_VJeVVfo

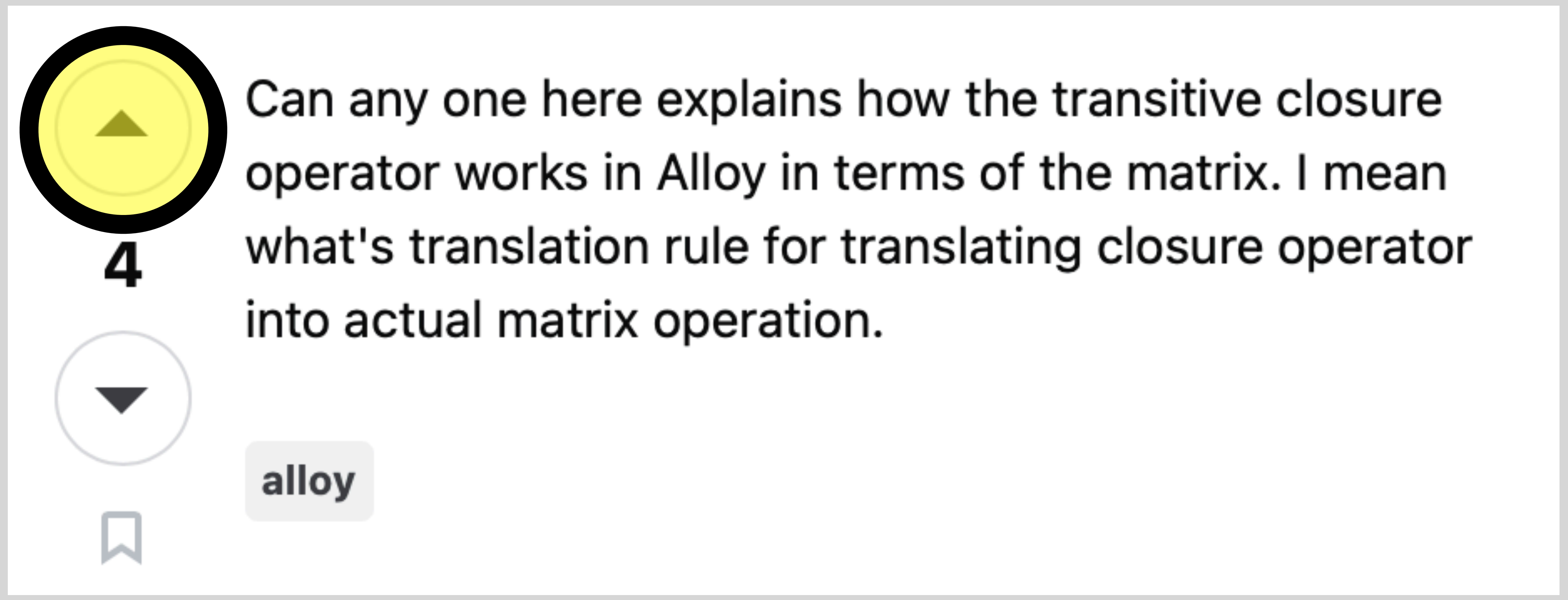


One more for dinner

Charlottesville | 3h ago

Again, please. He didn't take this position for our benefit. He did it for his.

Reply 65 Recommend Share Flag



Can any one here explains how the transitive closure operator works in Alloy in terms of the matrix. I mean what's translation rule for translating closure operator into actual matrix operation.

4

alloy

what's a concept?

a coherent **unit** of behavior

user-facing (a behavioral pattern)

a nano **service** (a backend API)

reusable & **familiar**

designed, coded and explained **independently**

defining a concept

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

state
  a set of Votes with
    a voter User
    a target Item

actions
  upvote (user: User, item: Item)
  unvote (user: User, item: Item)

new to concept design

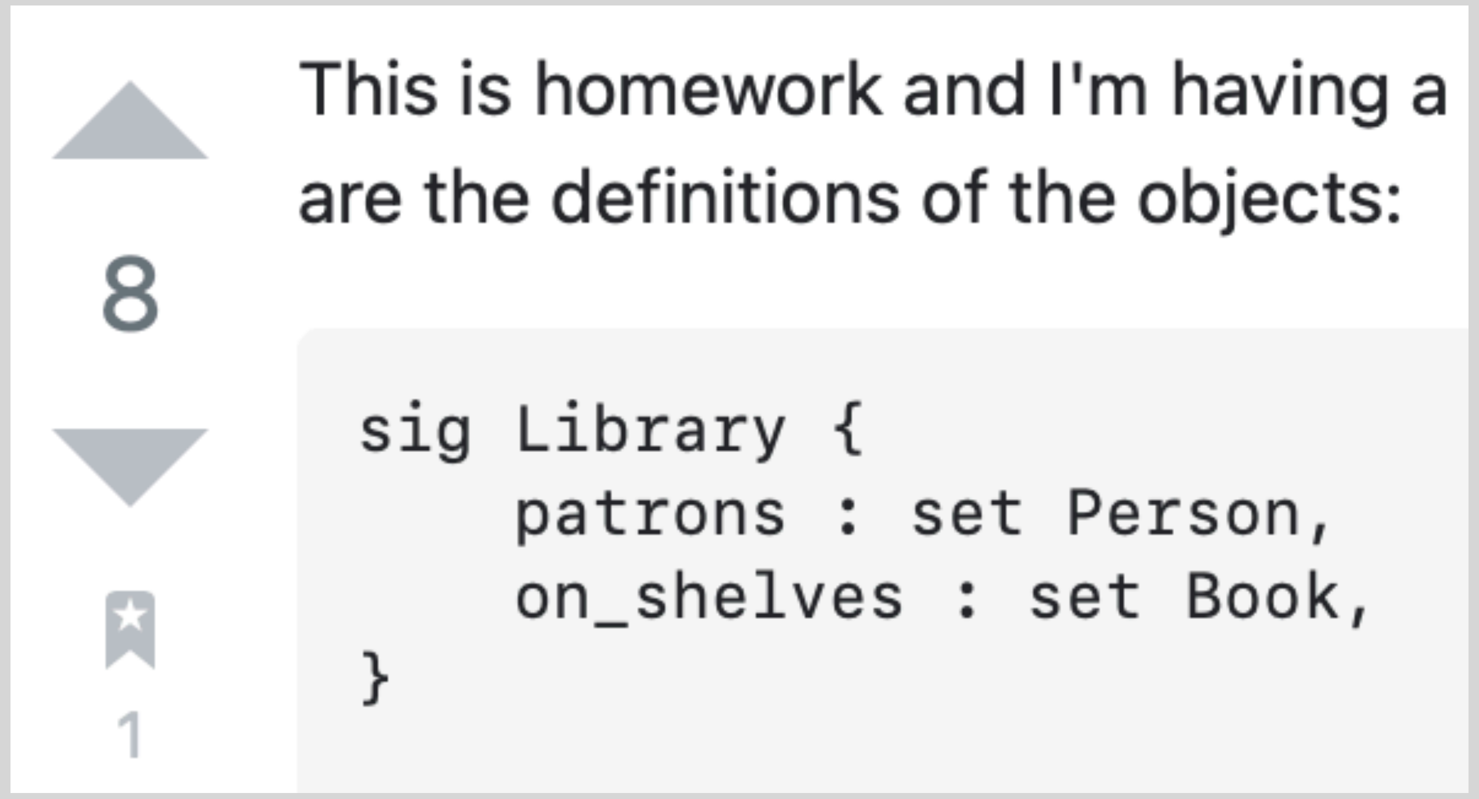
standard computer science

similar UIs, different concepts

concept Upvoting

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received



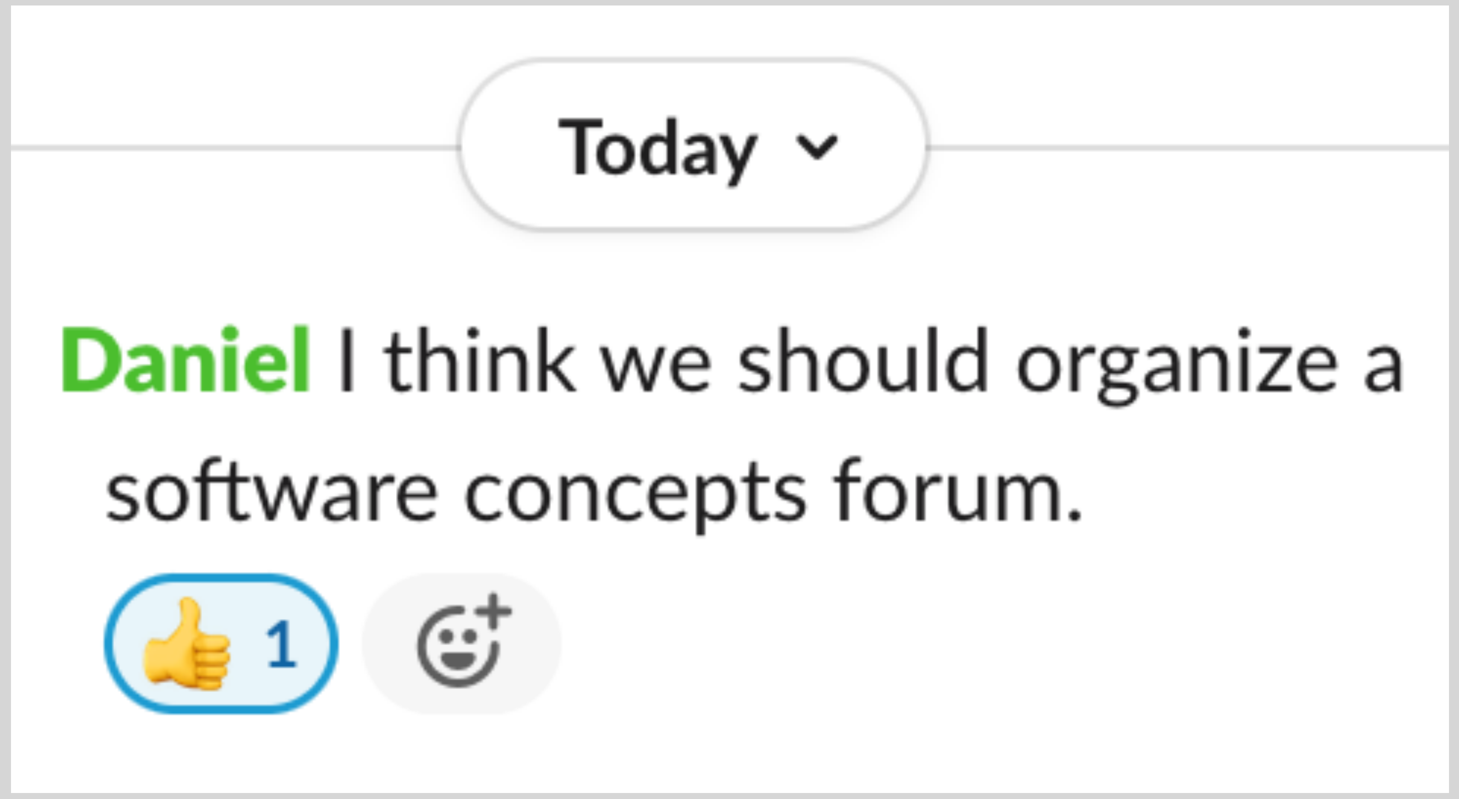
This is homework and I'm having a
are the definitions of the objects:

```
sig Library {
    patrons : set Person,
    on_shelves : set Book,
}
```

concept Reacting

purpose send reactions to author

principle when user selects reaction, it's shown to the author (often in aggregated form)

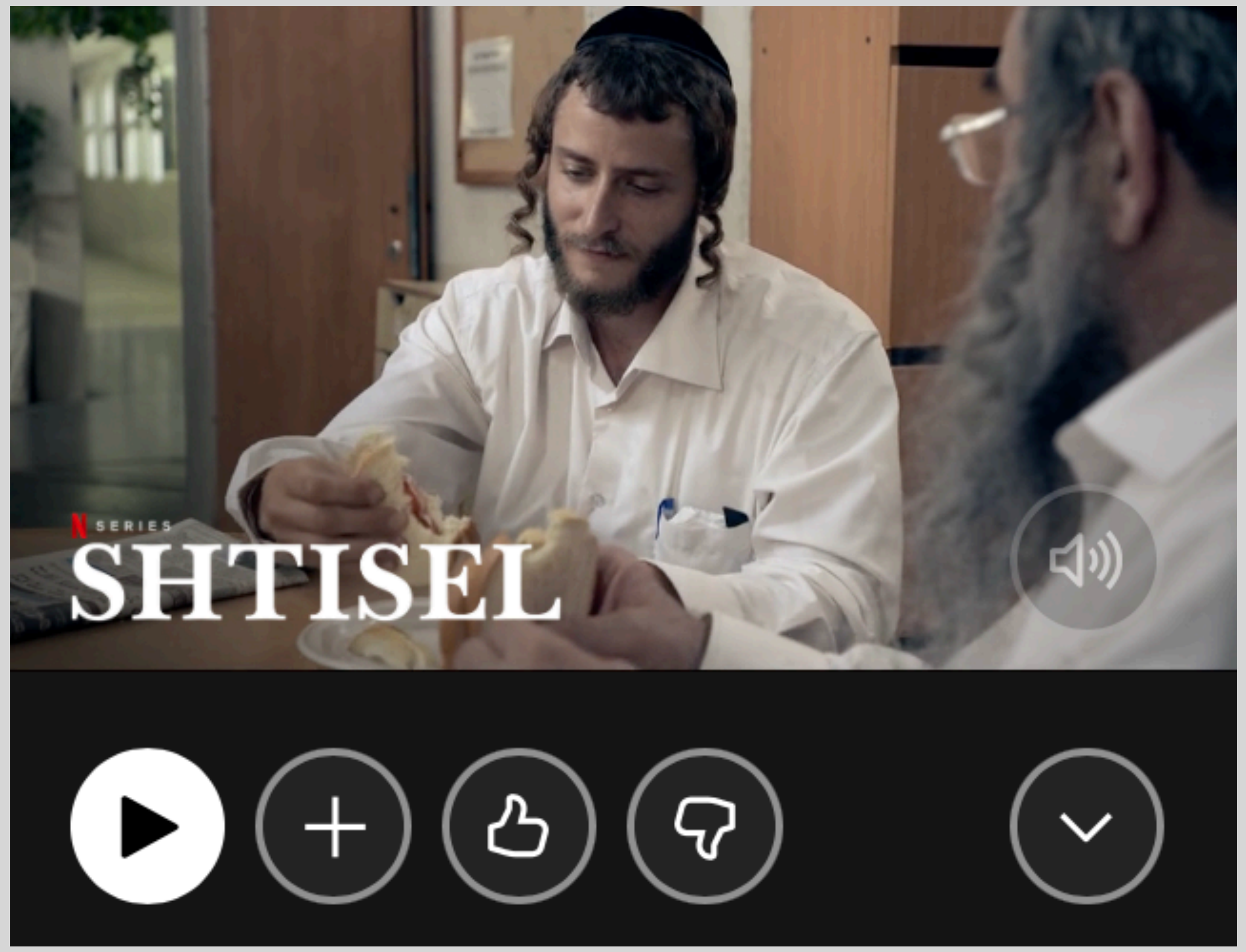


Daniel I think we should organize a software concepts forum.

concept Recommending

purpose use prior likes to recommend

principle user's likes lead to ranking of kinds of items, determining which items are recommended



concept name

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

state
  a set of Votes with
    a voter User
    a target Item

actions
  upvote (user: User, item: Item)
  unvote (user: User, item: Item)

name is very important!
becomes a shorthand for a design pattern
"let's use upvoting on comments"

type parameters
polymorphic types passed in and out of actions
concept assumes nothing about these
just opaque references to objects
so any kind of user, any kind of item

what are examples of items in familiar apps?

what might users correspond to?

purpose

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

```
state
  a set of Votes with
    a voter User
    a target Item
```

```
actions
  upvote (user: User, item: Item)
  unvote (user: User, item: Item)
```

purpose answers why
why use this concept? why invent this concept?
paradoxically, often the hardest part
but figuring this out can bring the most value

different purposes for different stakeholders
but there is usually a primary purpose

conflicting purposes
when more than one purpose, often conflict

what's the purpose for the platform?

operational principle

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

state
  a set of Votes with
    a voter User
    a target Item

actions
  upvote (user: User, item: Item)
  unvote (user: User, item: Item)

an archetypal scenario
a story about how the concept is used
the typical case, not an edge case
must illustrate how purpose is achieved

a bad operational principles
"after an upvote of an item, its count goes up"

what's wrong with the bad principle?

actions

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

state
  a set of Votes with
    a voter User
    a target Item

actions
  upvote (user: User, item: Item)
  unvote (user: User, item: Item)

actions are what users do
also system responses (eg, notify user)

user interface independent
not "click button" or "select item"
typically, one action for many micro steps in UI

actions aren't requests, so don't fail
if upvote would fail, action doesn't happen

actions implemented as functions
an API to the concept (along with state)

what other actions might upvoting have?

state

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

```
state
  a set of Votes with
    a voter User
    a target Item
```

```
actions
  upvote (user: User, item: Item)
  unvote (user: User, item: Item)
```

state is what the concept remembers
to determine whether actions are allowed
to generate outputs to actions
to show effect of actions to users

no observer actions needed
state is abstract, so no rep exposure worry
typically many kinds of queries
so don't want to specify each one
just assume state is visible and queryable

state implemented with persistent storage
eg, a relational or collection database
think of state as part of system's data schema

why does state store identities of voters?

what the state looks like: representation independence

concept Upvoting [User, Item]

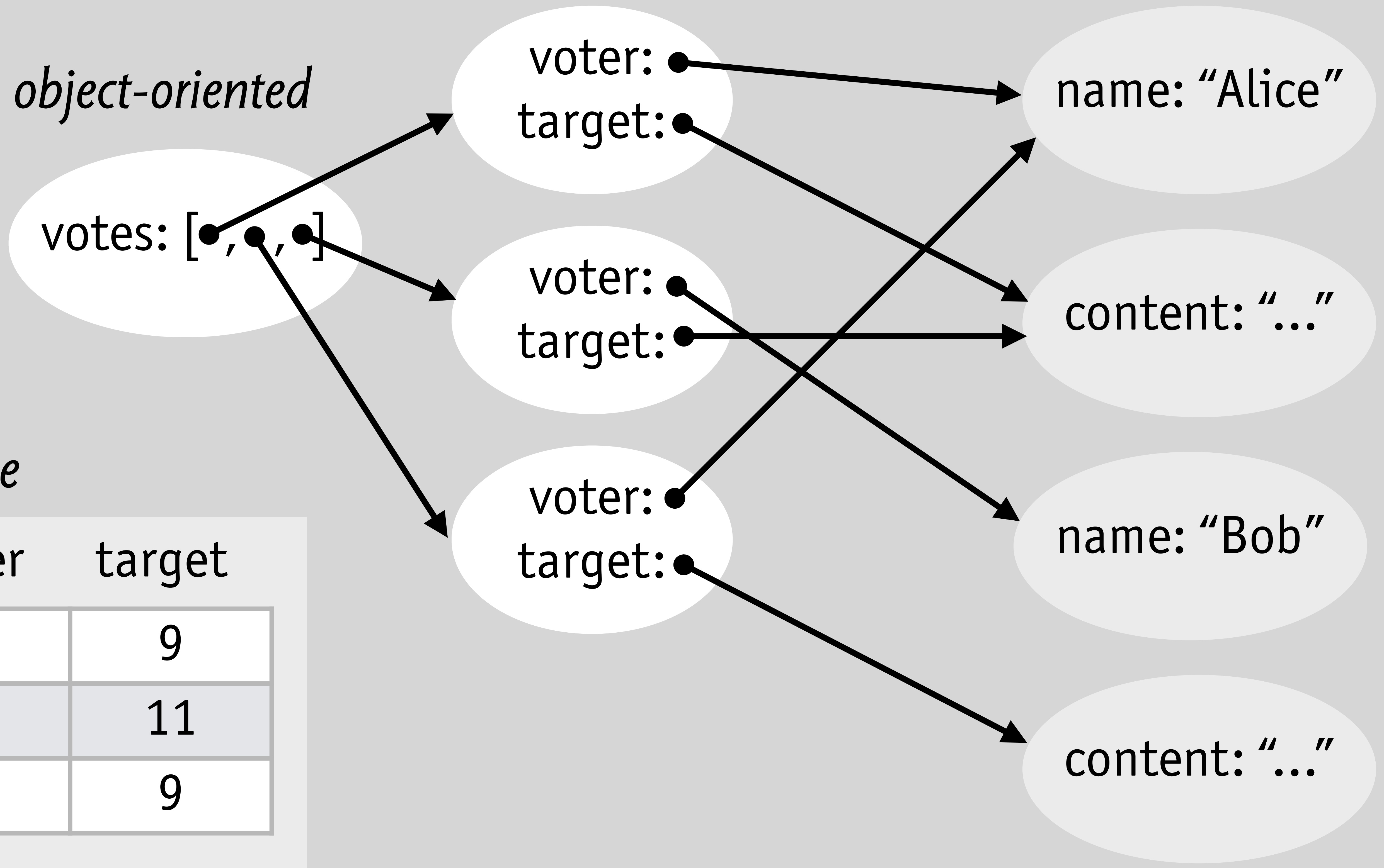
purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

state
a set of Votes with
a voter User
a target Item

actions
upvote (user: User, item: Item)
unvote (user: User, item: Item)

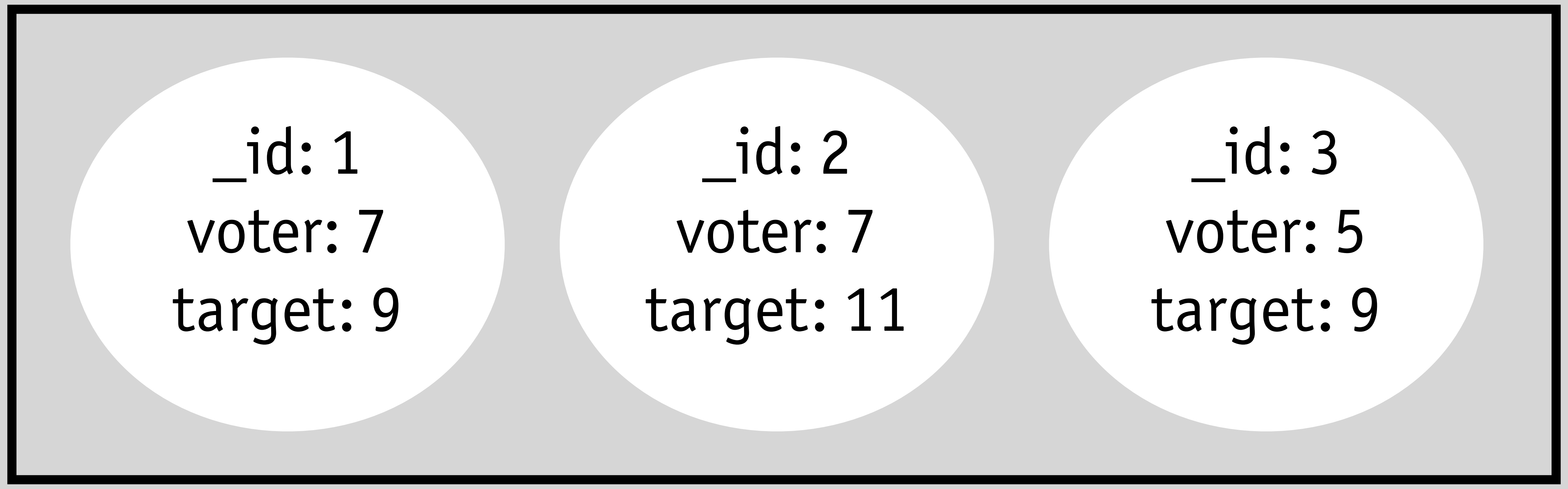
object-oriented



relational database

id	voter	target
1	7	9
2	7	11
3	5	9

document database



action specifications

concept Upvoting [User, Item]

purpose rank items by popularity

principle after a series of upvotes, can rank items by the number of votes they received

```
state
  a set of Votes with
    a voter User
    a target Item
```

```
actions
  upvote (user: User, item: Item)
    requires no vote by user for item
    effect add vote by user for item
```

can specify each action
standard pre/post specification
just like you've seen in 6.102

action specs determine behaviors
by induction, starting with initial states
by default, sets start empty

invariants (aka integrity constraints)
established at start and then preserved

what's an invariant of the state for upvoting?

concepts as carriers of design knowledge

concept: Upvoting

related concepts
Rating, Recommending, Reacting, ...

design variants
downvote as unvote
use age in ranking
weigh downvotes more
various identity tactics
freezing old posts

what are some known issues?

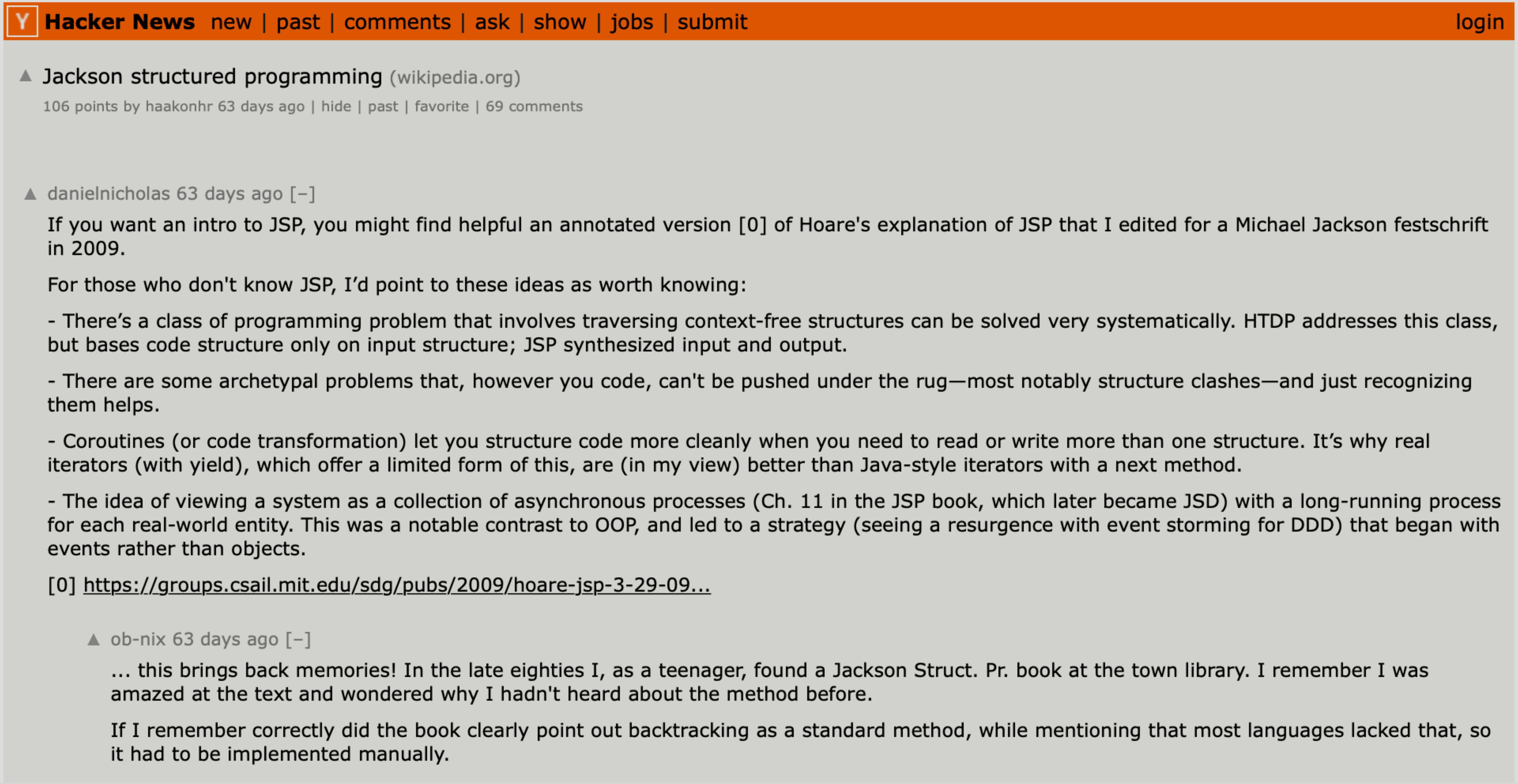
known issues
high votes can promote old content
feedback favors early upvotes
upvoting encourages echo chamber
preventing double votes

typical uses
social media posts
comments on articles
Q&A responses

often used with
Karma, Authentication ...

synchronization

composing concepts without coupling


Y Hacker News new | past | comments | ask | show | jobs | submit
login

▲ Jackson structured programming (wikipedia.org)
106 points by haakonhr 63 days ago | hide | past | favorite | 69 comments

▲ danielnicholas 63 days ago [–]

If you want an intro to JSP, you might find helpful an annotated version [0] of Hoare's explanation of JSP that I edited for a Michael Jackson festschrift in 2009.

For those who don't know JSP, I'd point to these ideas as worth knowing:

- There's a class of programming problem that involves traversing context-free structures can be solved very systematically. HTDP addresses this class, but bases code structure only on input structure; JSP synthesized input and output.

- There are some archetypal problems that, however you code, can't be pushed under the rug—most notably structure clashes—and just recognizing them helps.

- Coroutines (or code transformation) let you structure code more cleanly when you need to read or write more than one structure. It's why real iterators (with yield), which offer a limited form of this, are (in my view) better than Java-style iterators with a next method.

- The idea of viewing a system as a collection of asynchronous processes (Ch. 11 in the JSP book, which later became JSD) with a long-running process for each real-world entity. This was a notable contrast to OOP, and led to a strategy (seeing a resurgence with event storming for DDD) that began with events rather than objects.

[0] https://groups.csail.mit.edu/sdg/pubs/2009/hoare-jsp-3-29-09...

▲ ob-nix 63 days ago [–]

... this brings back memories! In the late eighties I, as a teenager, found a Jackson Struct. Pr. book at the town library. I remember I was amazed at the text and wondered why I hadn't heard about the method before.

If I remember correctly did the book clearly point out backtracking as a standard method, while mentioning that most languages lacked that, so it had to be implemented manually.

adding application-specific functionality

concept Upvoting

purpose rank items by popularity

actions
upvote (user, item)
downvote (user, item)
unvote (user, item)

suppose I want this behavior:
you can't downvote an item
until you've received
an upvote on your own post

could just modify Upvote
why is this bad?

define a new concept!
a hint: not just used by Upvote

concept Karma

purpose privilege good users

state
a set of Users with
  a karma Number

actions
reward (user, reward)

concept Posting

purpose share content

state
a set of Posts with
  a body String
  an author User

actions
create (user, body): post
delete (post)
edit (post, body)

a first synchronization

concept Upvoting

actions
upvote (user, item)
downvote (user, item)
unvote (user, item)

when Upvoting.upvote (post)
where Posting: author of post is user
then Karma.reward (user, 10)

concept Posting

state
a set of Posts with
  a body String
  an author User

actions
create (user, body): post
delete (post)
edit (post, body)

concept Karma

state
a set of Users with
  a karma Number

actions
reward (user, reward)

a second synchronization

concept Upvoting

actions
upvote (user, item)
downvote (user, item)
unvote (user, item)

concept Requesting

actions
upvote (user, item)
downvote (user, item)
...
edit (post, body)

when Requesting.downvote (user, post)
where Karma: user has karma >= 20
then Upvoting.downvote (user, post)

concept Posting

state
a set of Posts with
  a body String
  an author User

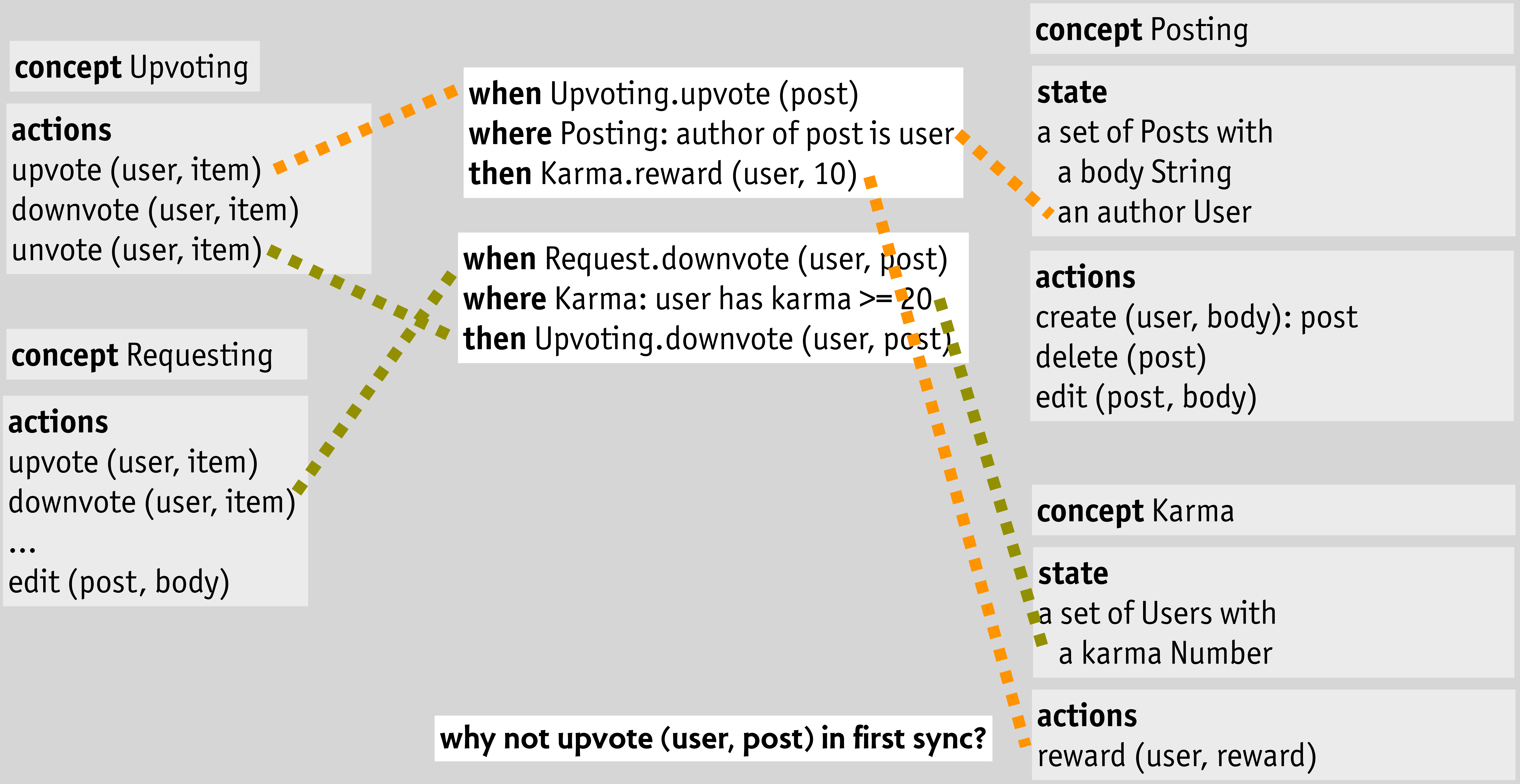
actions
create (user, body): post
delete (post)
edit (post, body)

concept Karma

state
a set of Users with
  a karma Number

actions
reward (user, reward)

controlling downvoting in two syncs



concept Upvoting

actions
upvote (user, item)
downvote (user, item)
unvote (user, item)

concept Requesting

actions
upvote (user, item)
downvote (user, item)
...
edit (post, body)

when Upvoting.upvote (post)
where Posting: author of post is user
then Karma.reward (user, 10)

when Request.downvote (user, post)
where Karma: user has karma >= 20
then Upvoting.downvote (user, post)

concept Posting

state
a set of Posts with
  a body String
  an author User

actions
create (user, body): post
delete (post)
edit (post, body)

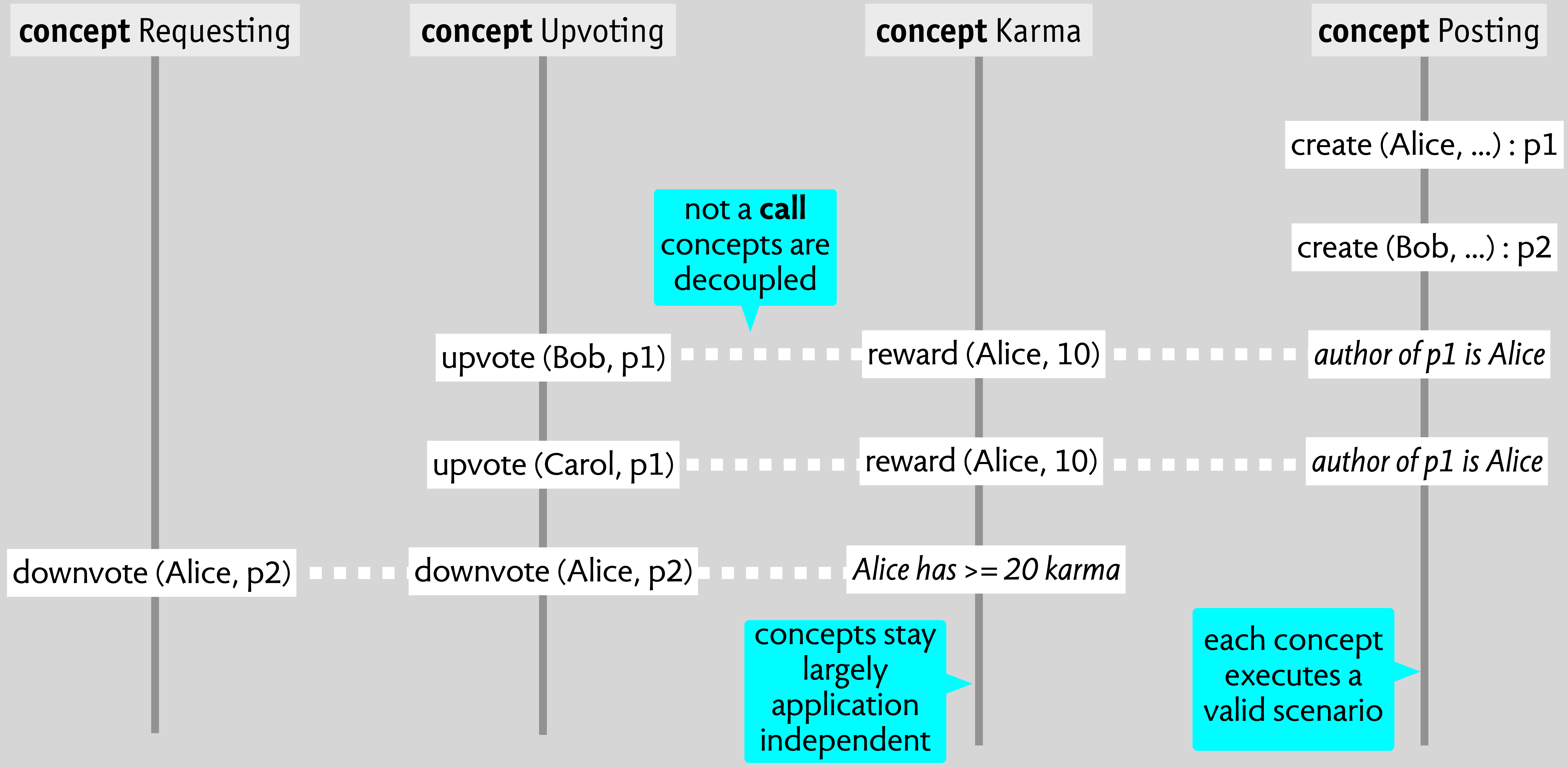
concept Karma

state
a set of Users with
  a karma Number

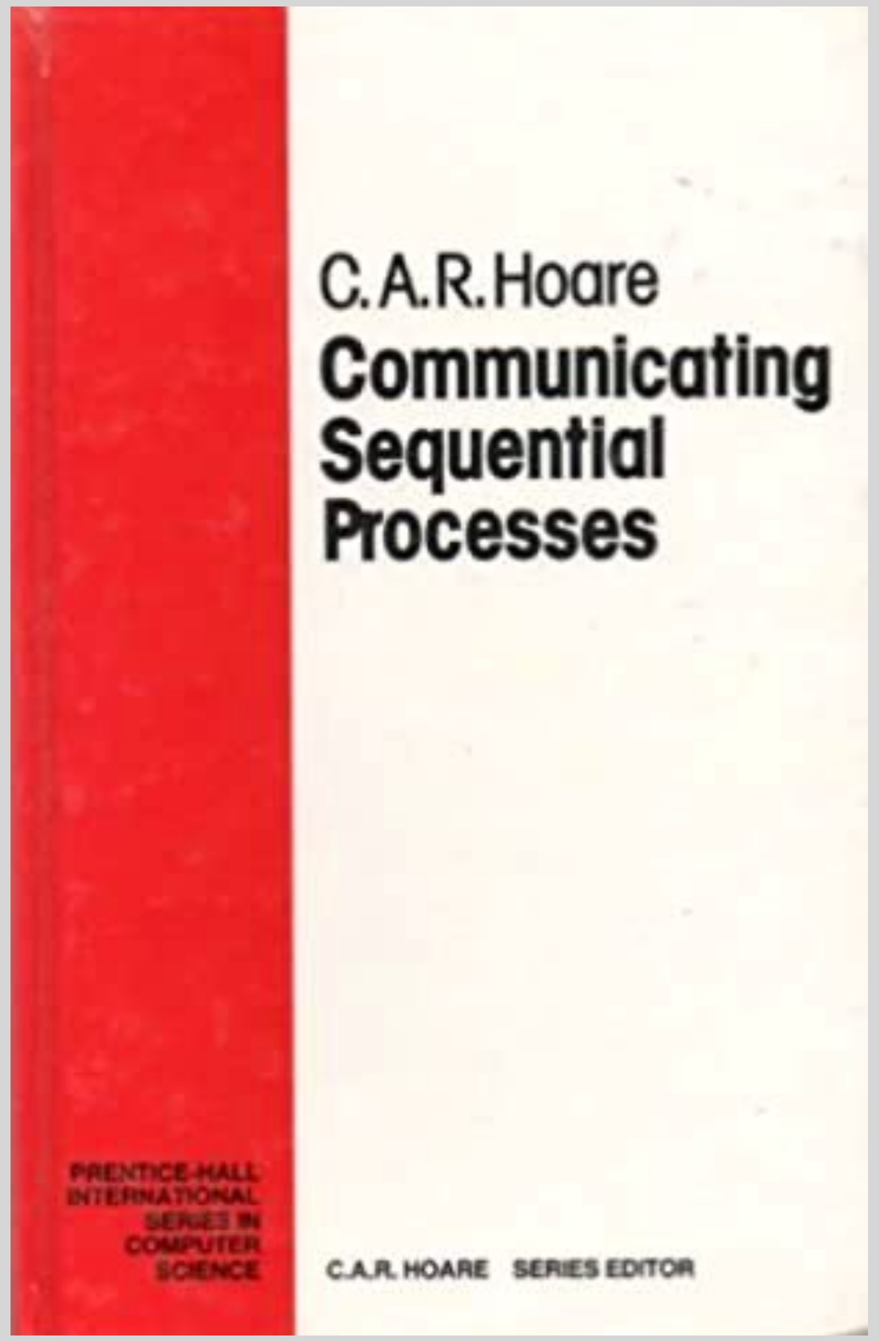
actions
reward (user, reward)

why not upvote (user, post) in first sync?

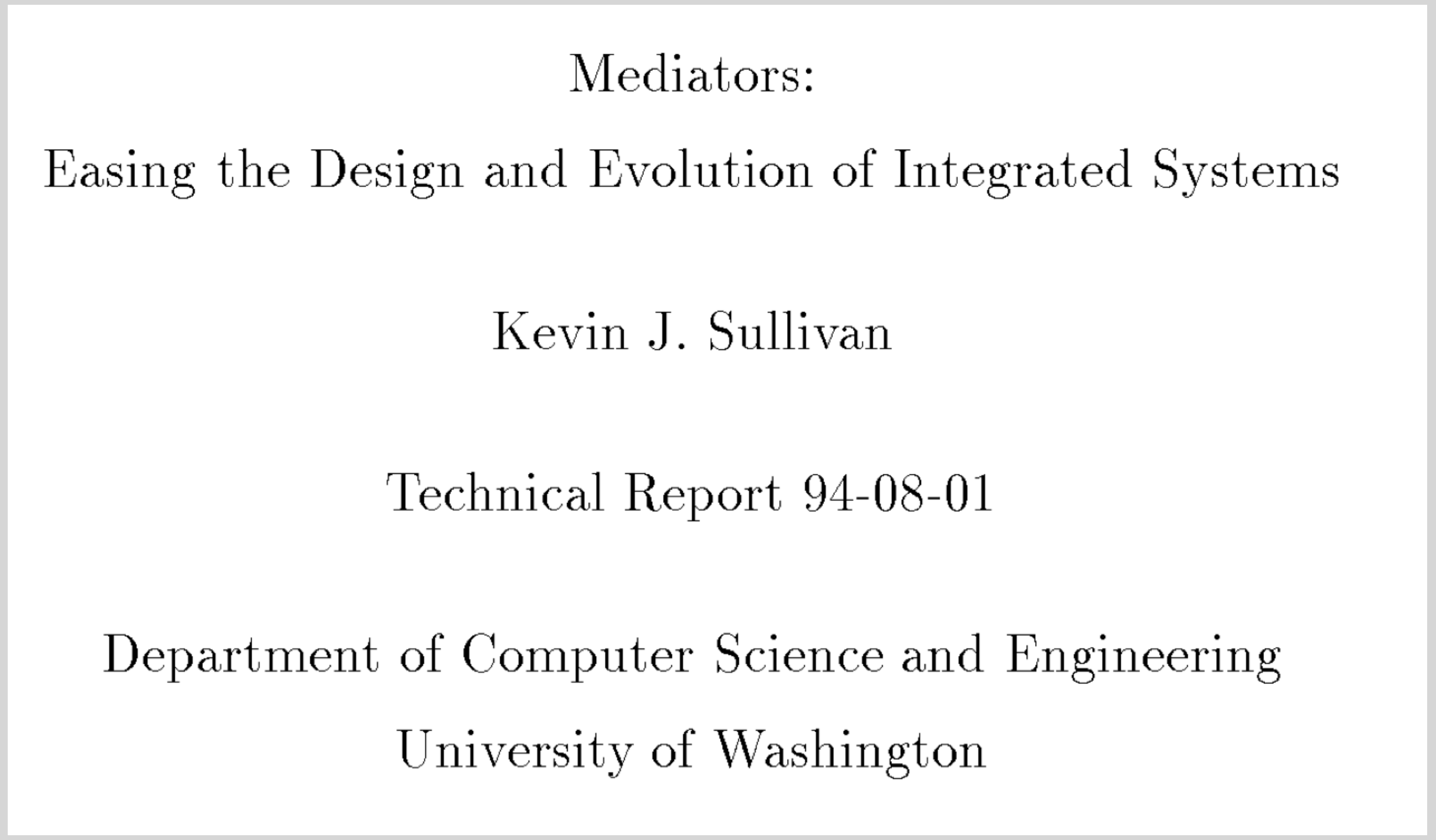
synchronization viewed over scenarios (traces)



not a new idea


C.A.R.Hoare
Communicating Sequential Processes
PRENTICE-HALL INTERNATIONAL SERIES IN COMPUTER SCIENCE
C.A.R. HOARE SERIES EDITOR

composition uses event sync from Hoare's CSP



Mediators:

Easing the Design and Evolution of Integrated Systems

Kevin J. Sullivan

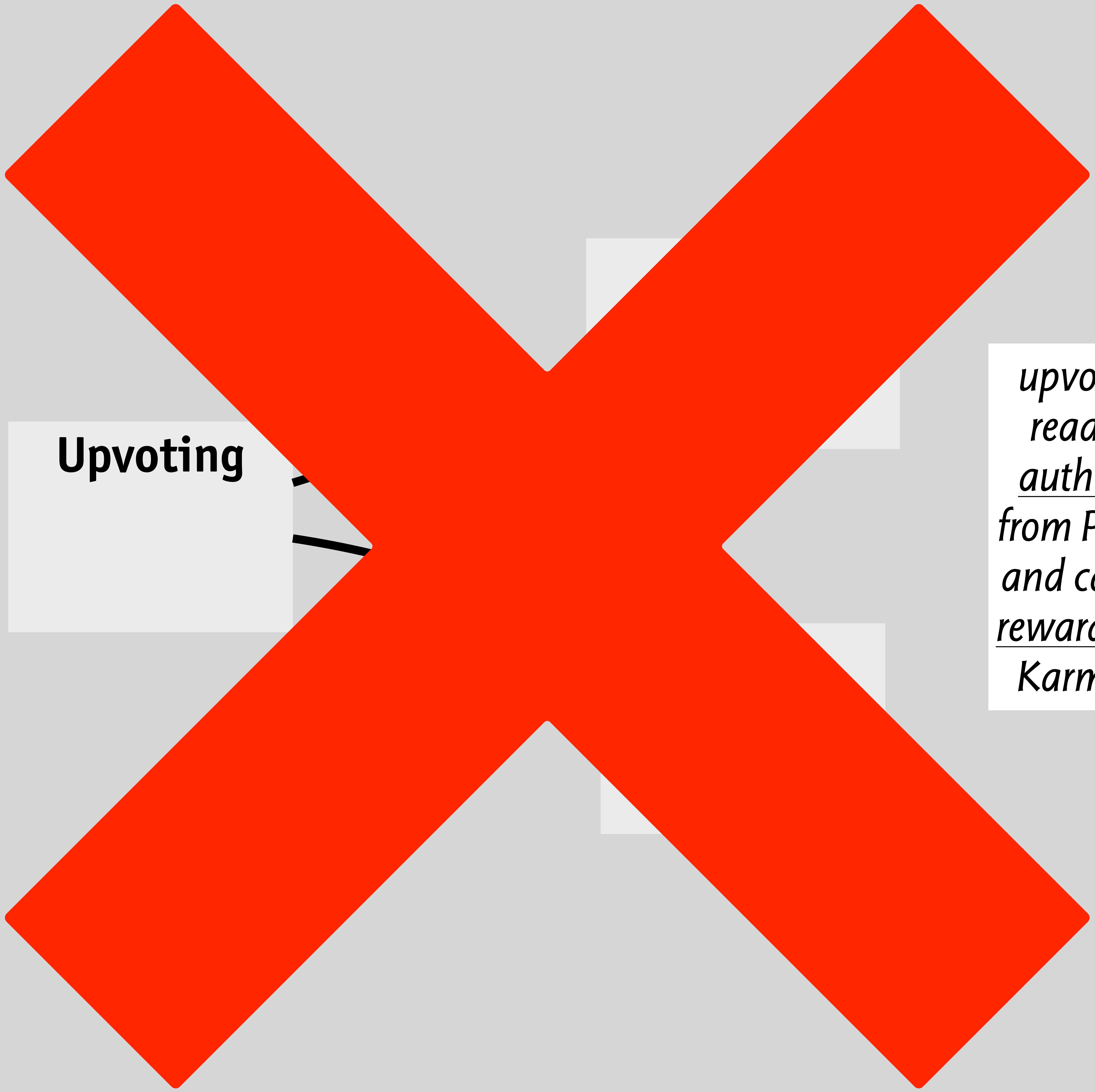
Technical Report 94-08-01

Department of Computer Science and Engineering

University of Washington

mediator pattern subject of Sullivan's thesis

a reminder: how we didn't do it



upvote reads <u>author</u> from Post and calls <u>reward</u> in Karma

concepts never
call each other's actions
read or write each other's state
share mutable composite objects

some more synchronization examples

```
when
  Requesting.createPost (body)
  Authenticate.authenticate (): user
then Posting.create (user, body)
```

authenticating users

```
when Commenting.create (post, body)
where Posting: author of post is user
then Notifying.notify (user, “Comment “ + body)
```

notifying post author when someone comments

```
when Posting.delete (post)
where Commenting: post is target of comment
then Commenting.delete (comment)
```

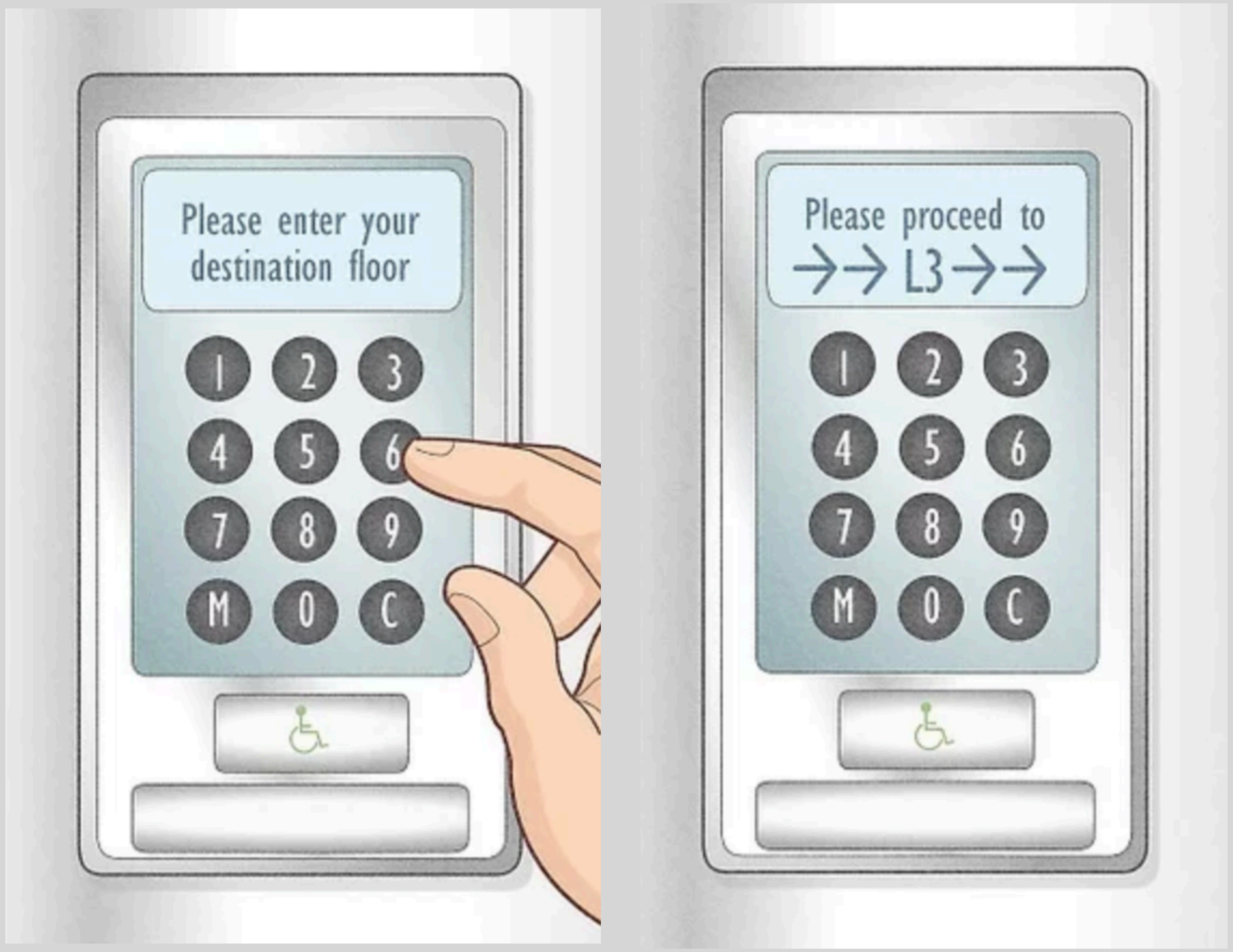
```
when Requesting.deletePost (post)
where Commenting: no comments on post
then Posting.delete (post)
```

two ways to handle post deletion and comments

some influential concepts & syncs

a concept that
changes our behavior

“destination dispatch elevator”



https://www.wikihow.life/Ride-a-Destination-Dispatch-Elevator / Creative Commons

the concept

concept DestinationDispatch

purpose make elevator scheduling more efficient

principle you request a floor to go to, and an elevator is assigned; you wait for that elevator and enter it when it arrives; it will then leave and eventually arrive at the requested floor

actions
  request (on, to: Floor): (assigned: Elevator)
  system arrive (e: Elevator, at: Floor)
  system leave (e: Elevator, at: Floor)

composing with other concepts

A boost to traffic performance

Schindler PORT groups passengers by destination to provide the shortest possible trip for every rider, avoiding chaotic elevator runs and random, multiple stops.

User-friendly operation

With Schindler PORT's universal card reader, a rider simply swipes a personal access card at the Schindler PORT terminal for an elevator car to be immediately assigned with an approved destination.

Excellent personalized service

Schindler PORT can customize your journey, whether it be a special VIP trip, a streamlining patient transport or allowing for more approach time, longer door opening time or additional space for passengers with special needs.

Enhanced building security

Floor access is defined by the building management through access cards or smartphones. Schindler PORT can also be integrated with turnstiles to further enhance building security.

CardAuthentication
Directory
DestinationDispatch
card C issued for user U
user U registered at floor F
card C authenticates user U
get floor for user U returns F
request floor F assigns elev E
elevator E arrives
elevator E arrives at floor F
1 5 9 13
-1 4 8 12
MAIN FLOOR

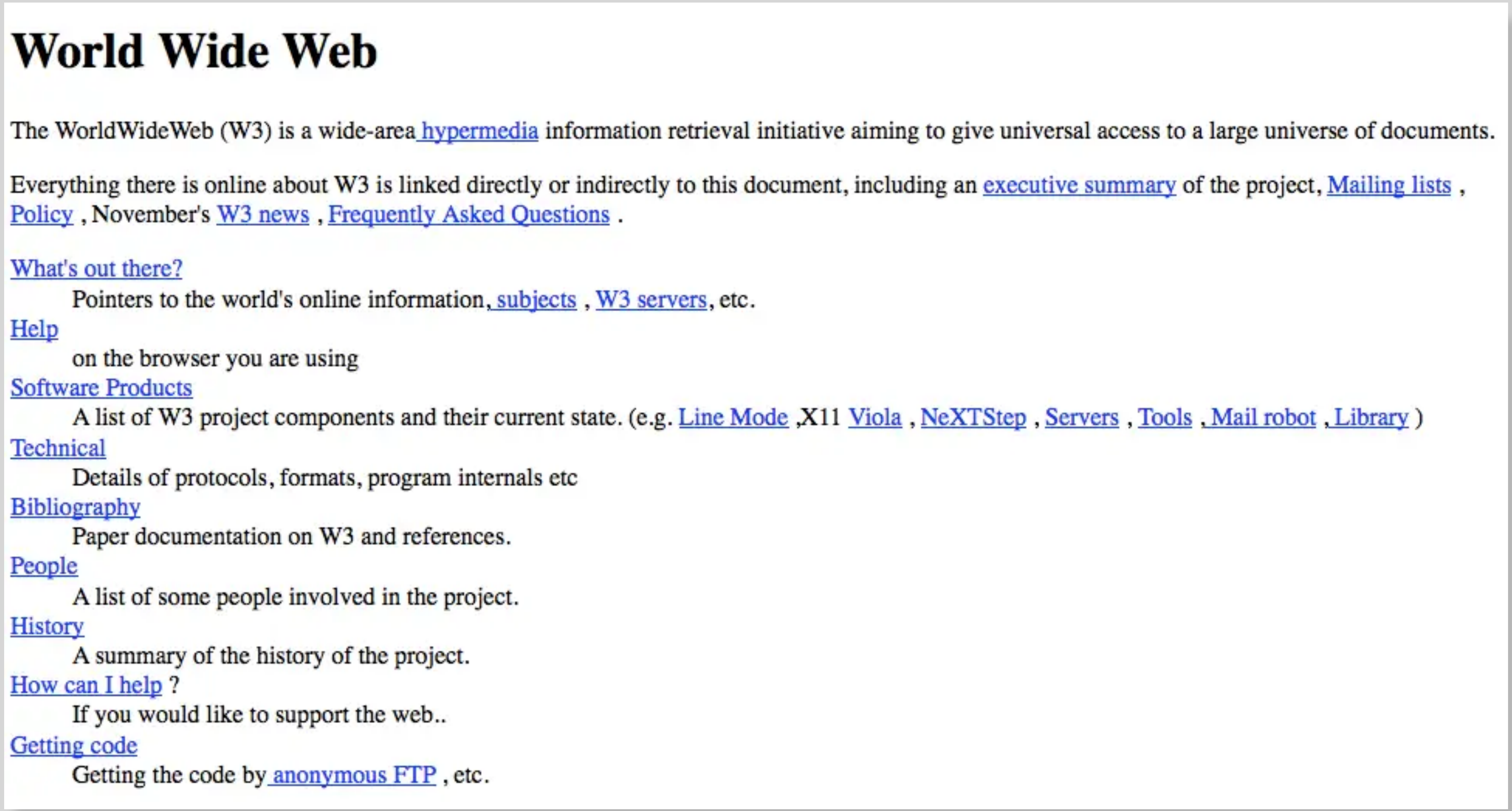
Schindler Miconic 10 (1992)
first commercial implementation

Leo Port, electrical engineer in Sydney
patents destination dispatch (1961)
but lets it expire (1977)

Schindler introduces PORT (2009)
“Personal Occupant Requirement Terminal”

a concept that
created a new world

what was novel about the web?



World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
: Pointers to the world's online information, subjects , W3 servers, etc.

Help
: on the browser you are using

Software Products
: A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)

Technical
: Details of protocols, formats, program internals etc

Bibliography
: Paper documentation on W3 and references.

People
: A list of some people involved in the project.

History
: A summary of the history of the project.

How can I help ?
: If you would like to support the web..

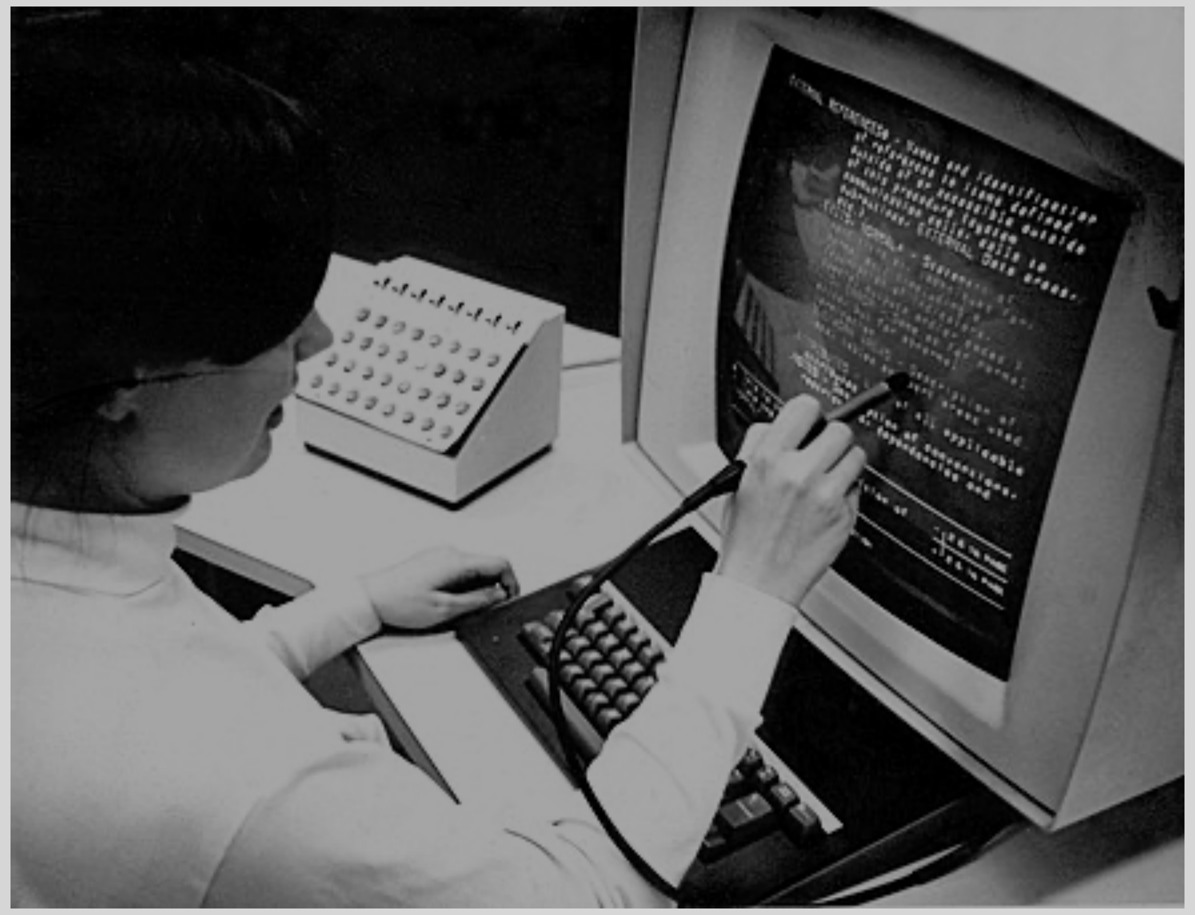
Getting code
: Getting the code by anonymous FTP , etc.

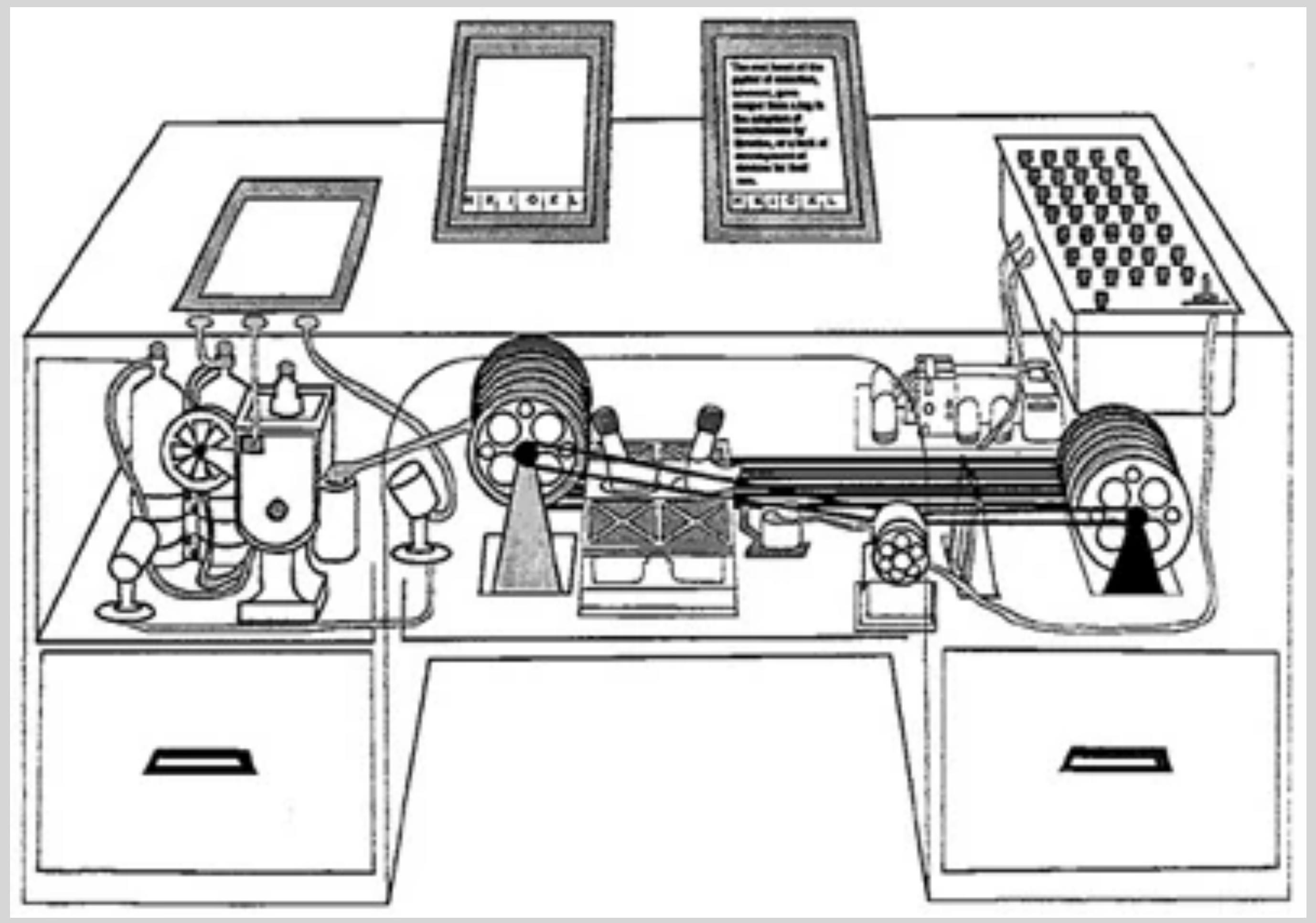
http://info.cern.ch/hypertext/WWW/TheProject.html

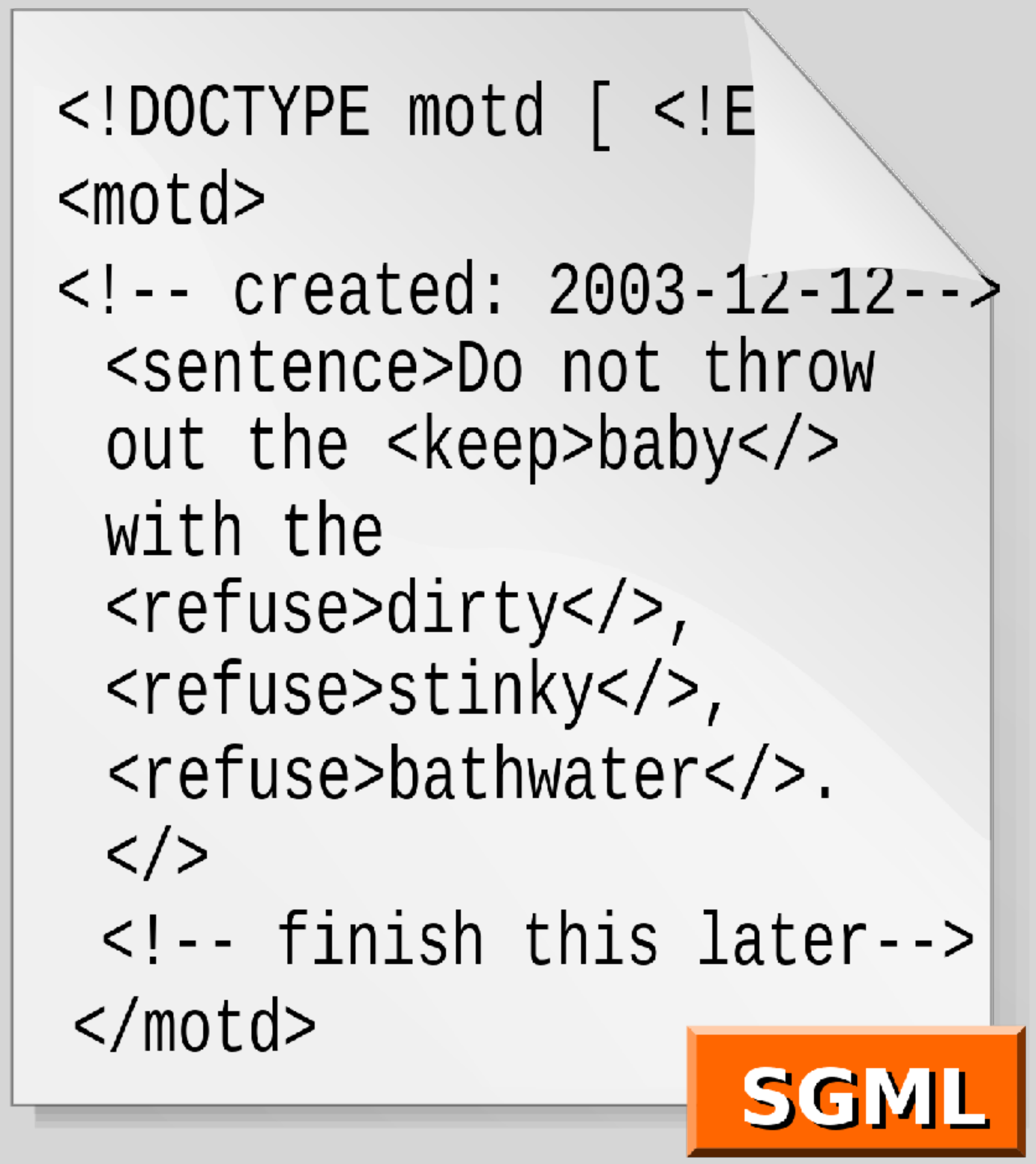
the technology?

NeXT computer
660MB hard disk
Motorola 68030, 25MHz
17" monitor with Display Postscript
built in ethernet connectivity

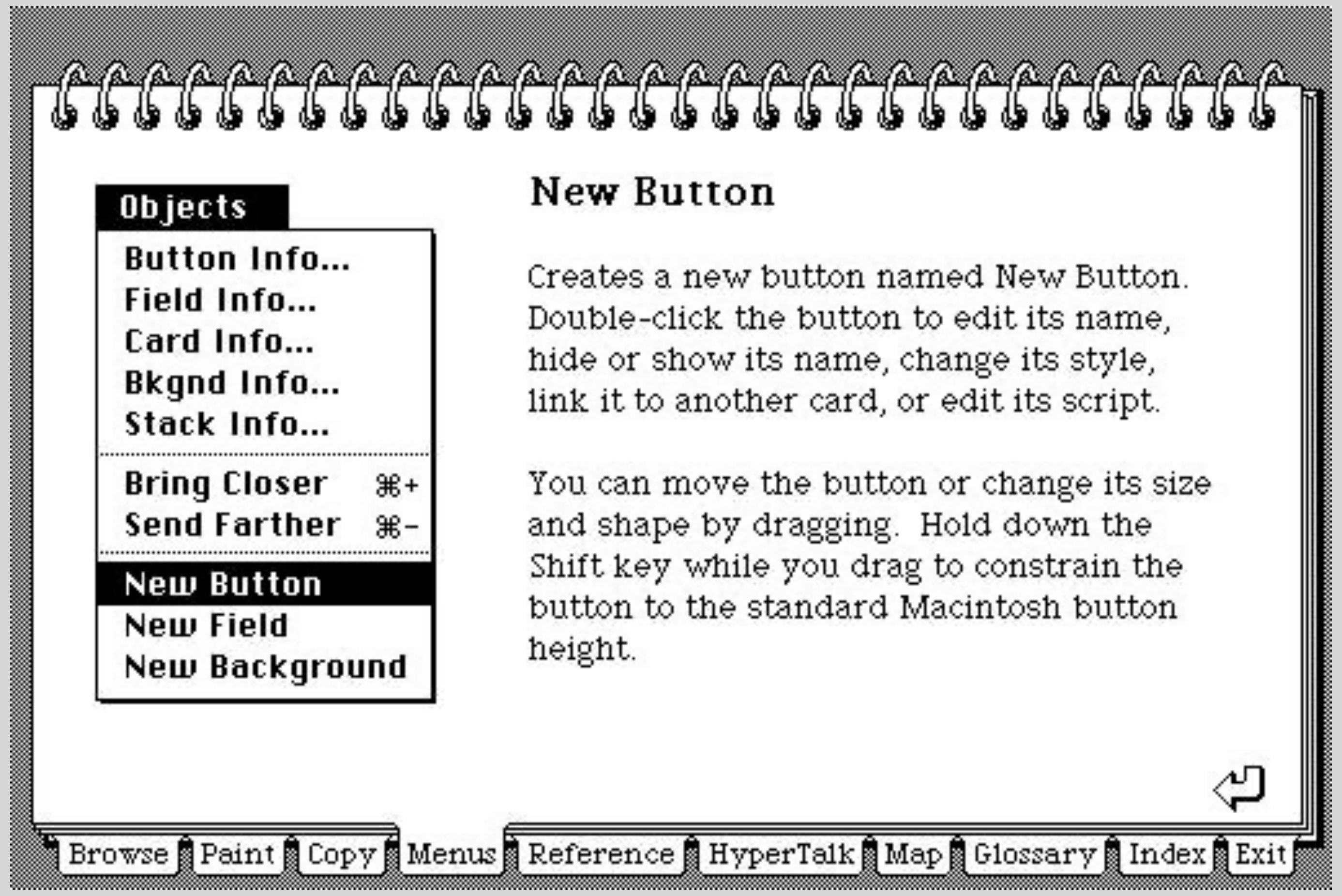
hypertext?


Hypertext Editing System
(Nelson & van Dam, 1967)


Memex
(Vannevar Bush, 1945)


```
<!DOCTYPE motd [ <!E
<motd>
<!-- created: 2003-12-12-->
 <sentence>Do not throw
 out the <keep>baby</>
 with the
 <refuse>dirty</>,
 <refuse>stinky</>,
 <refuse>bathwater</>.
 </>
 <!-- finish this later-->
</motd>
```

SGML
(Charles Goldfarb, 1986)


Objects
Button Info...
Field Info...
Card Info...
Bkgnd Info...
Stack Info...
Bring Closer ⌘+
Send Farther ⌘-
New Button
New Field
New Background

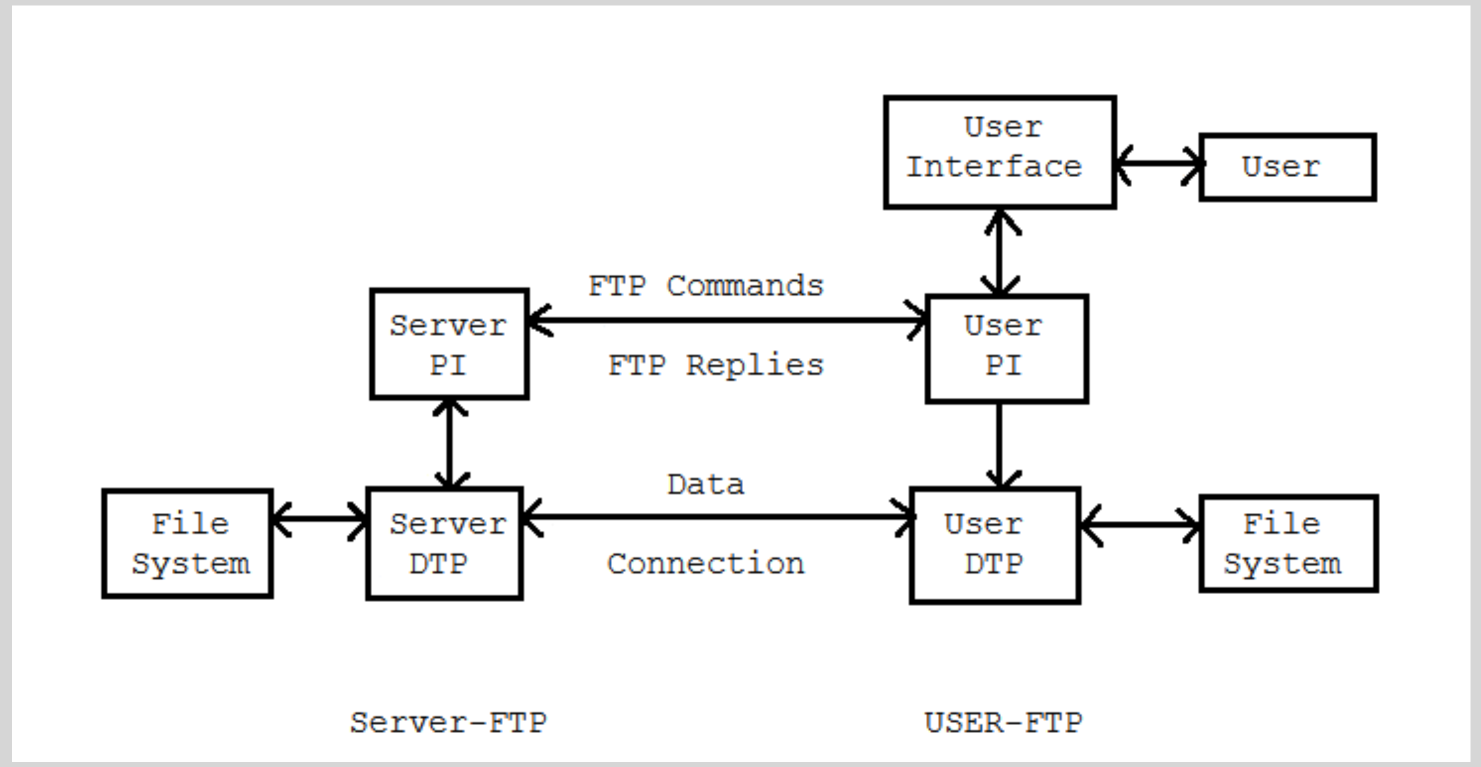
New Button

Creates a new button named New Button. Double-click the button to edit its name, hide or show its name, change its style, link it to another card, or edit its script.

You can move the button or change its size and shape by dragging. Hold down the Shift key while you drag to constrain the button to the standard Macintosh button height.

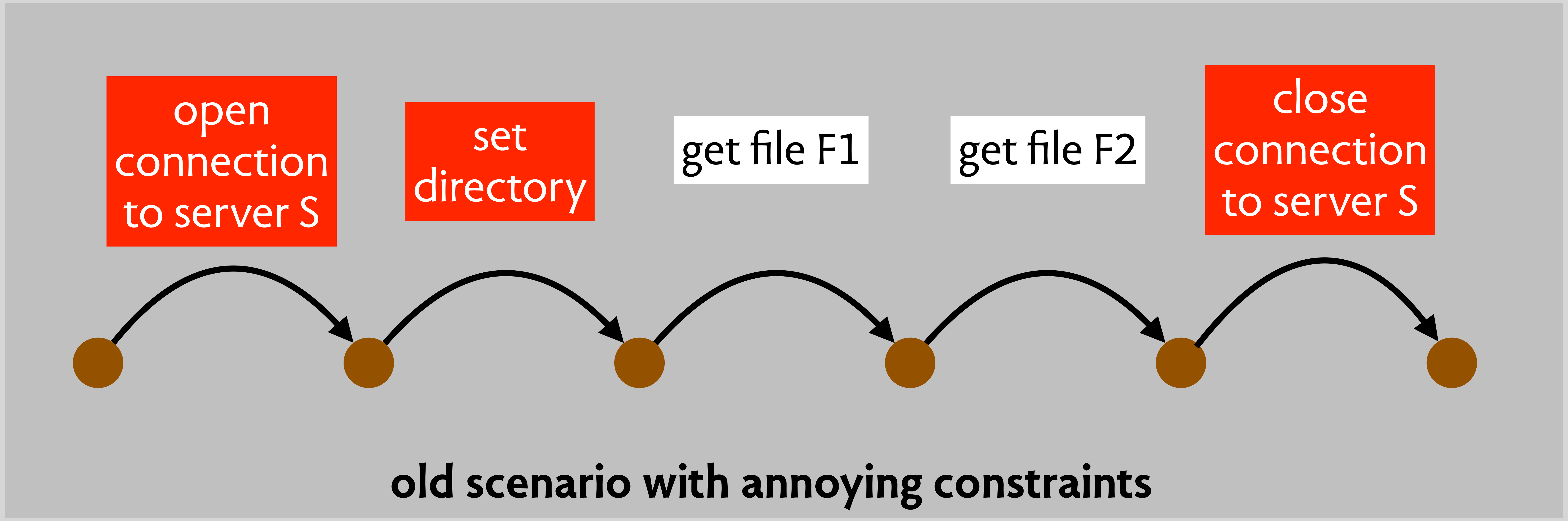
Browse | Paint | Copy | Menus | Reference | HyperTalk | Map | Glossary | Index | Exit

Apple HyperCard
(Bill Atkinson, 1987)

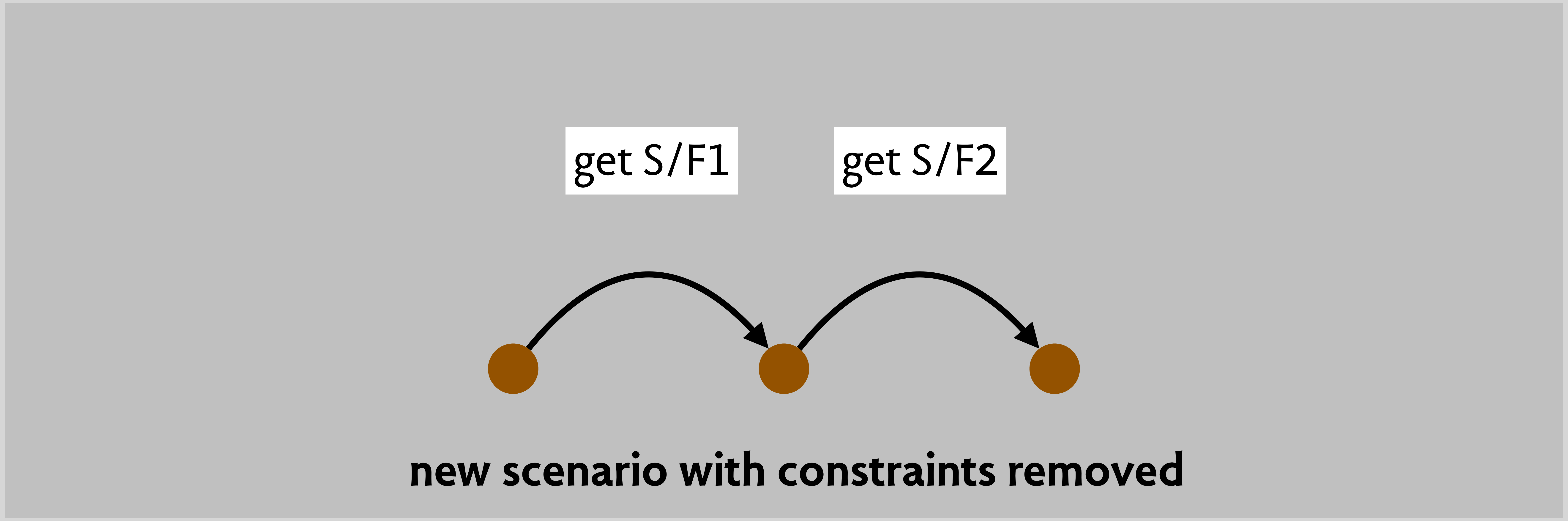


File Transfer Protocol
(Abhay Bhushan, 1971)

a new way to get content



old scenario with annoying constraints



new scenario with constraints removed

getting to the essence of URLs

concept DistributedNaming [Domain, Name]

purpose stable, global naming of resources

principle after publishing a resource at a domain and a path, a get at that domain and path will return the published resource

state

```
a set of Domains each with
  a set of NamedResources
a set of NamedResources each with
  a Name
  a Resource
```

actions

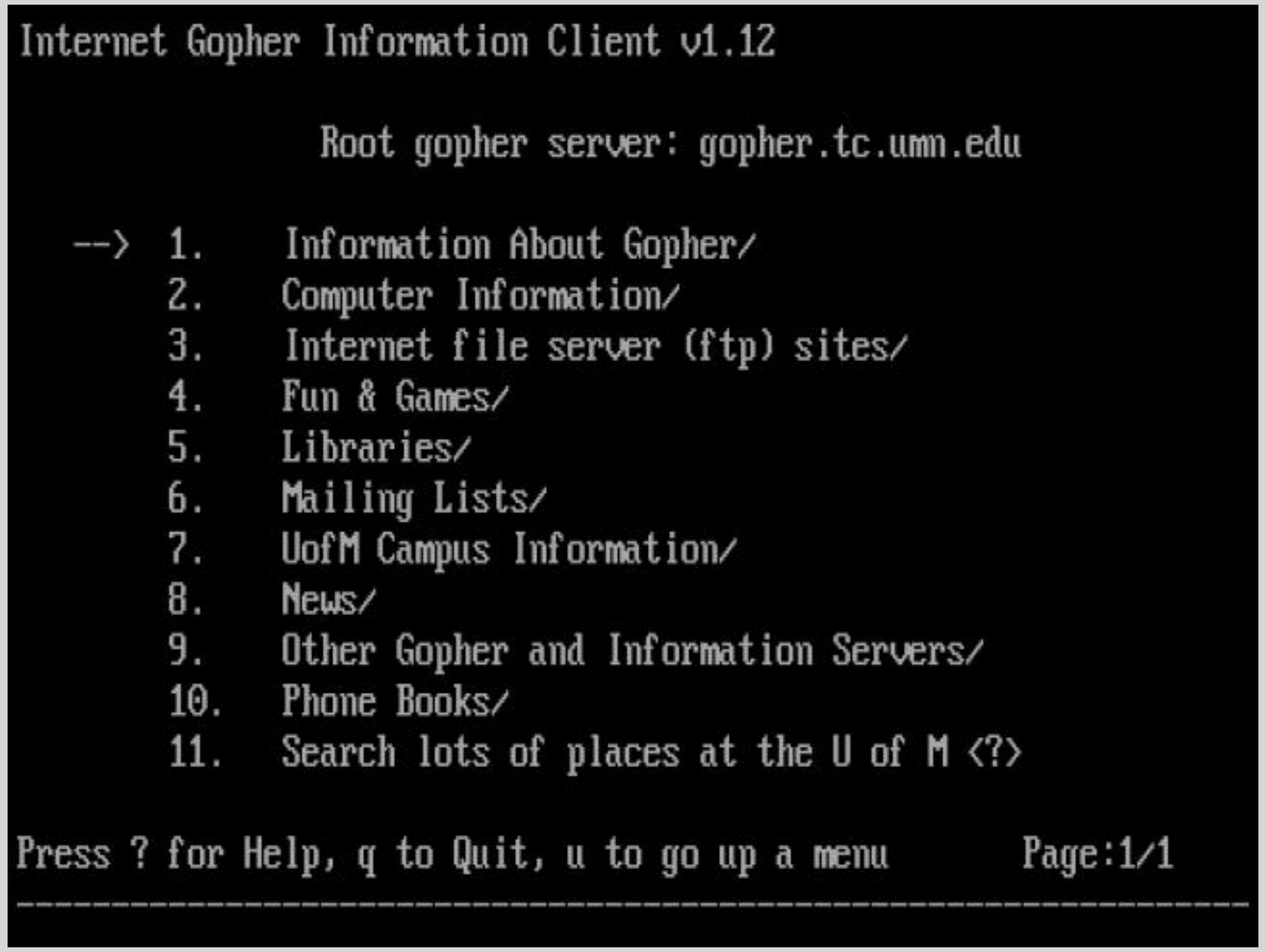
```
publish (d: Domain, n: Name, r: Resource)
unpublish (d: Domain, n: Name, r: Resource)
get (d: Domain, n: Name): (r: Resource)
```

why aren't domain and name strings?
not essential to this concept
(domains especially will be structured)

why separate domain and name?
they play very different roles!
domains allow separate control of naming
domain owner gets to choose name
domain are defined by DNS

an example

publish (*essenceofsoftware.com*, *post/ai-coding*, *blog post*)



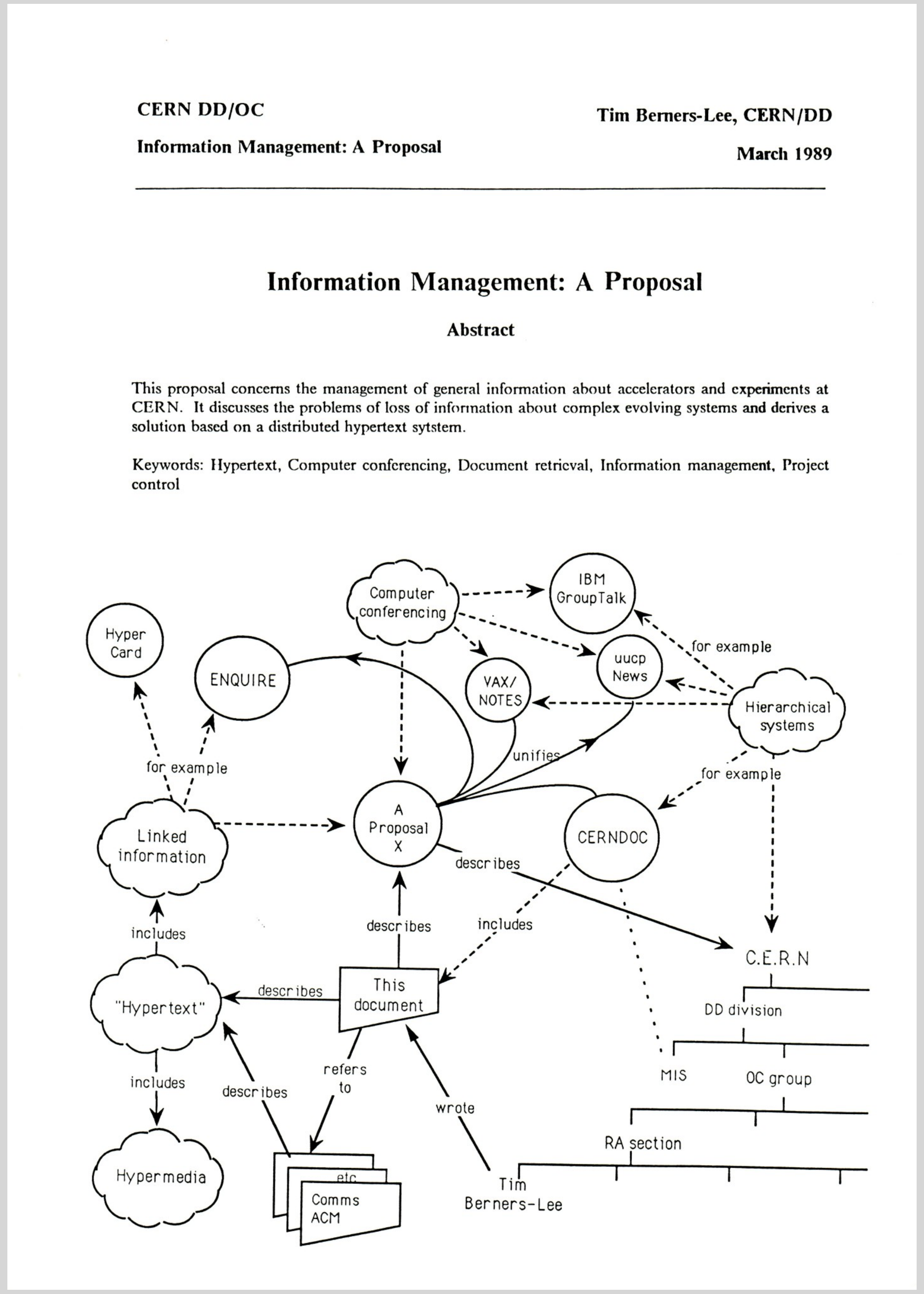
```
Internet Gopher Information Client v1.12

                Root gopher server: gopher.tc.umn.edu

    --> 1.     Information About Gopher/
        2.     Computer Information/
        3.     Internet file server (ftp) sites/
        4.     Fun & Games/
        5.     Libraries/
        6.     Mailing Lists/
        7.     UofM Campus Information/
        8.     News/
        9.     Other Gopher and Information Servers/
        10.    Phone Books/
        11.    Search lots of places at the U of M <?>

Press ? for Help, q to Quit, u to go up a menu          Page:1/1
```

Gopher: a hierarchical catalog (1991)



CERN DD/OC — Tim Berners-Lee, CERN/DD

Information Management: A Proposal — March 1989

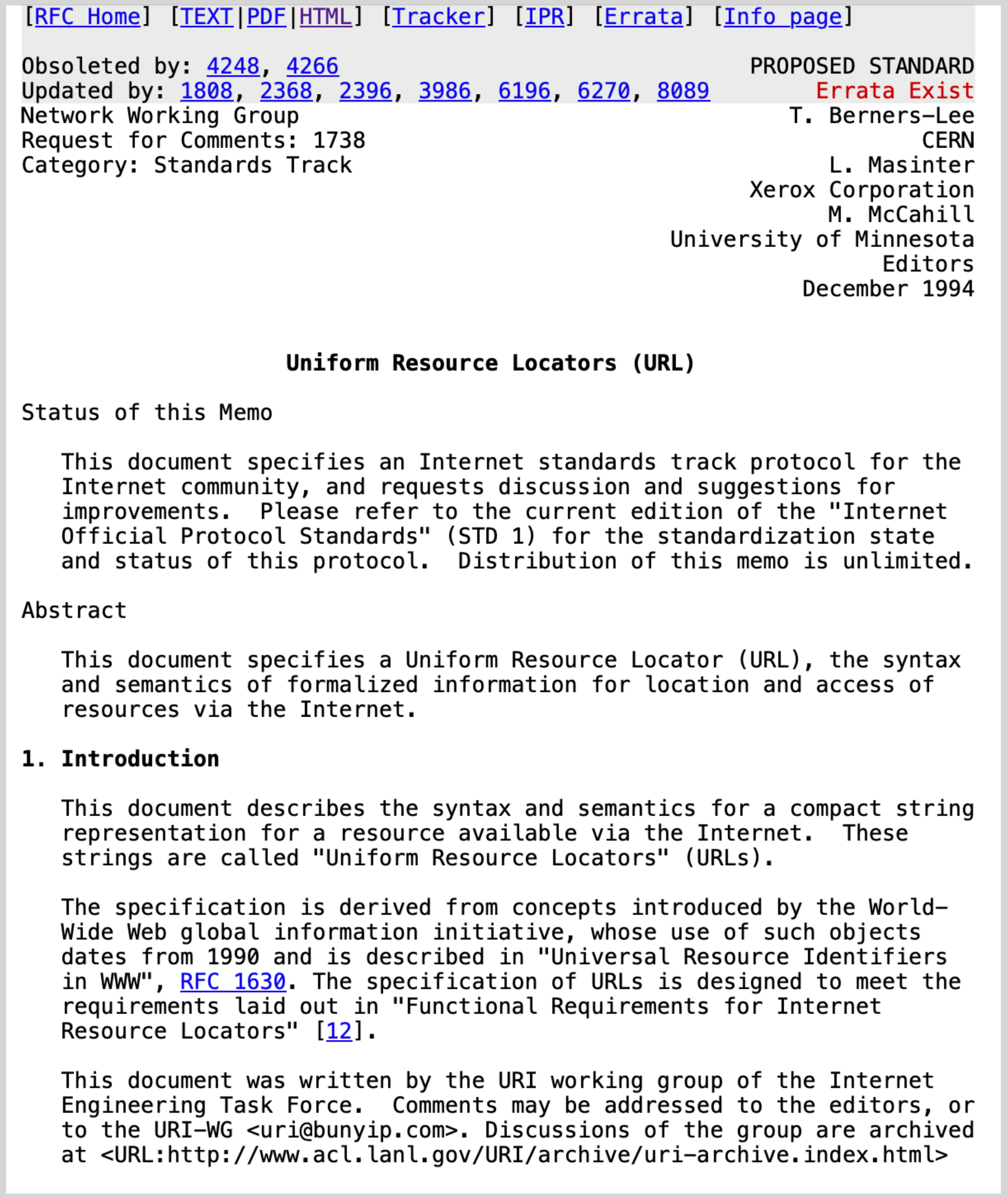
Information Management: A Proposal

Abstract

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext sytstem.

Keywords: Hypertext, Computer conferencing, Document retrieval, Information management, Project control

Tim Berners Lee outlines the elements of the web (1989): HTML, HTTP, URL



```
[RFC Home] [TEXT|PDF|HTML] [Tracker] [IPR] [Errata] [Info page]

Obsoleted by: 4248, 4266                              PROPOSED STANDARD
Updated by: 1808, 2368, 2396, 3986, 6196, 6270, 8089       Errata Exist
Network Working Group                                   T. Berners-Lee
Request for Comments: 1738                                        CERN
Category: Standards Track                                  L. Masinter
                                                     Xerox Corporation
                                                           M. McCahill
                                               University of Minnesota
                                                               Editors
                                                         December 1994


                    Uniform Resource Locators (URL)

Status of this Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements.  Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol.  Distribution of this memo is unlimited.

Abstract

   This document specifies a Uniform Resource Locator (URL), the syntax
   and semantics of formalized information for location and access of
   resources via the Internet.

1. Introduction

   This document describes the syntax and semantics for a compact string
   representation for a resource available via the Internet.  These
   strings are called "Uniform Resource Locators" (URLs).

   The specification is derived from concepts introduced by the World-
   Wide Web global information initiative, whose use of such objects
   dates from 1990 and is described in "Universal Resource Identifiers
   in WWW", RFC 1630. The specification of URLs is designed to meet the
   requirements laid out in "Functional Requirements for Internet
   Resource Locators" [12].

   This document was written by the URI working group of the Internet
   Engineering Task Force.  Comments may be addressed to the editors, or
   to the URI-WG <uri@bunyip.com>. Discussions of the group are archived
   at <URL:http://www.acl.lanl.gov/URI/archive/uri-archive.index.html>
```

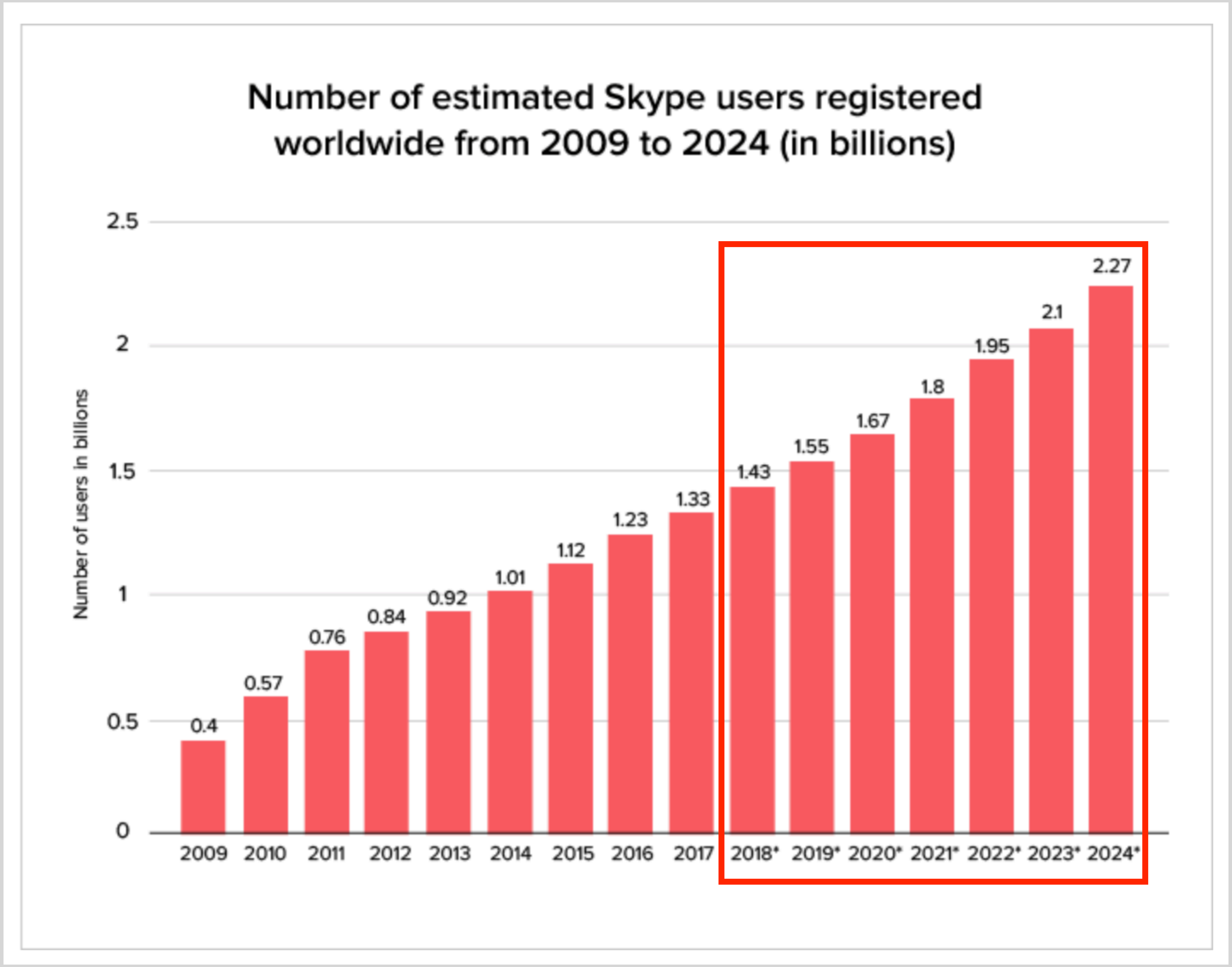
URLs defined in RFC 1738 (1994)

not a solved problem

50% of the URLs in United States Supreme Court opinions are broken

Jonathan Zittrain, Kendra Albert and Lawrence Lessig (2014)

Perma: Scoping and Addressing the Problem of Link and Reference Rot in Legal Citations a killer concept

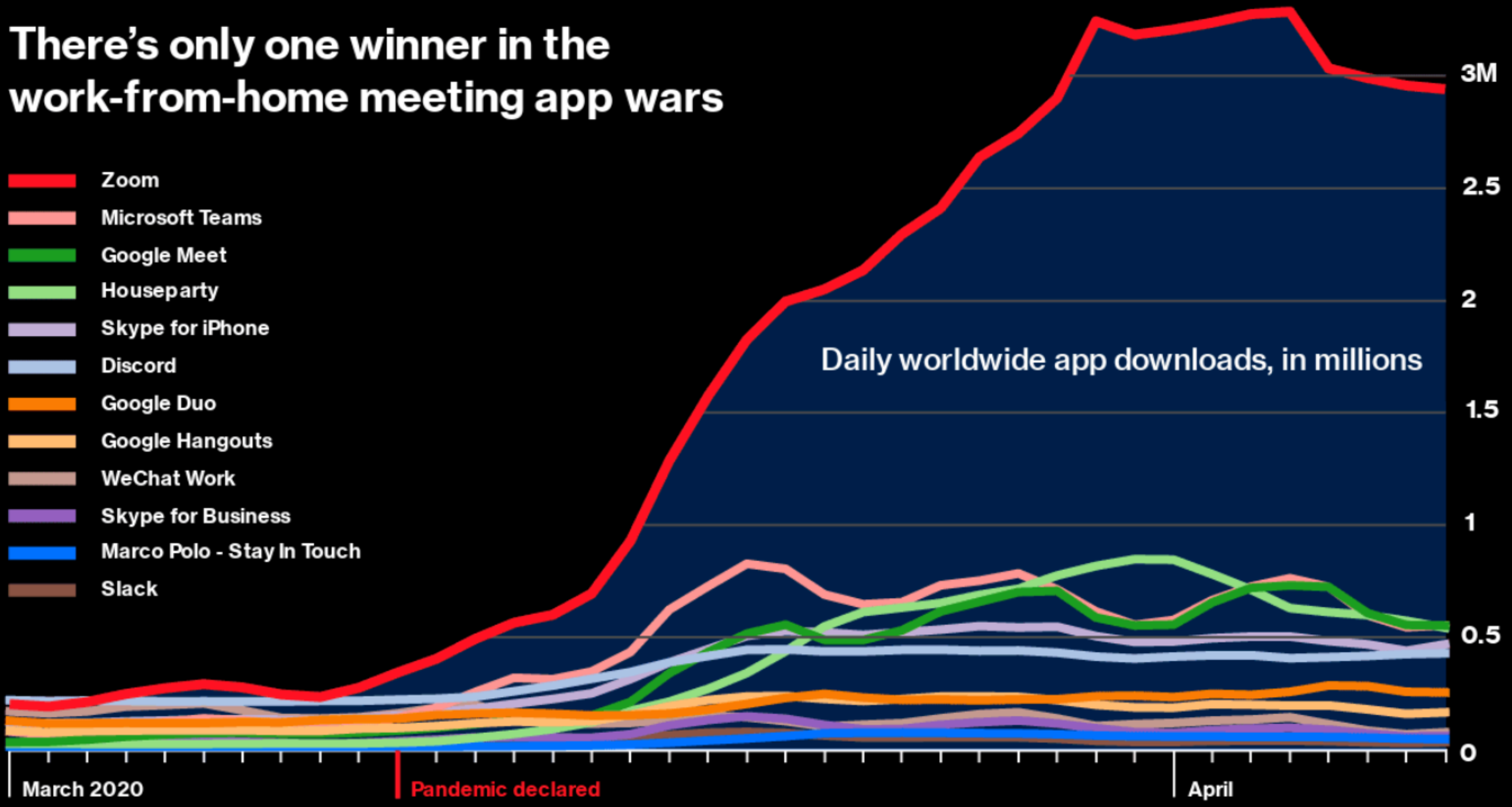


from Statistica: 2018-2024 estimated

Covid-19 / Zoom-a-Zoom-Zoom

BBOViz
Covid-19

There's only one winner in the work-from-home meeting app wars



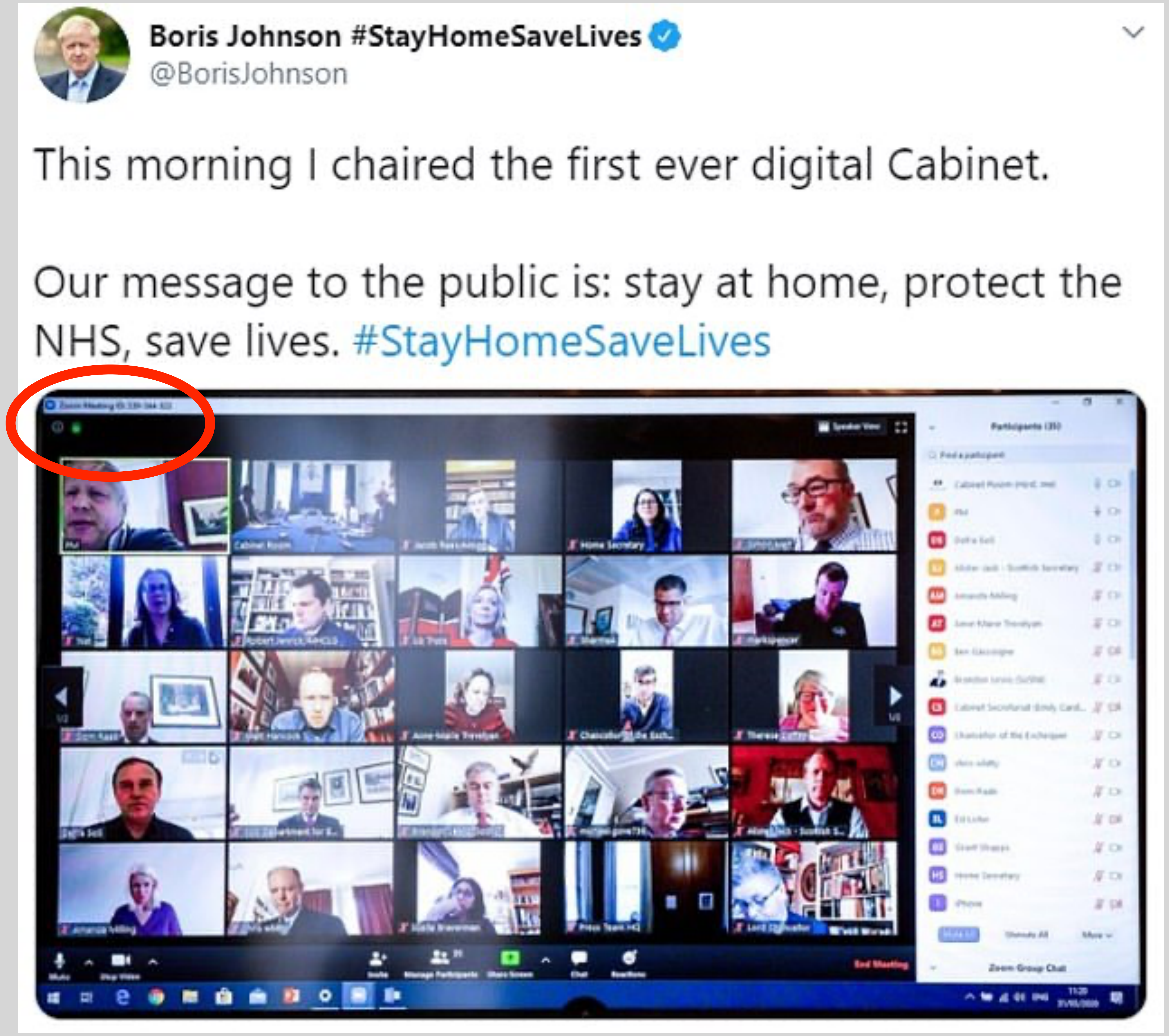
Source: Apptopia

BEN SCHOTT

BloombergOpinion

what was novel about Zoom?

shares meeting id!


Boris Johnson #StayHomeSaveLives
@BorisJohnson

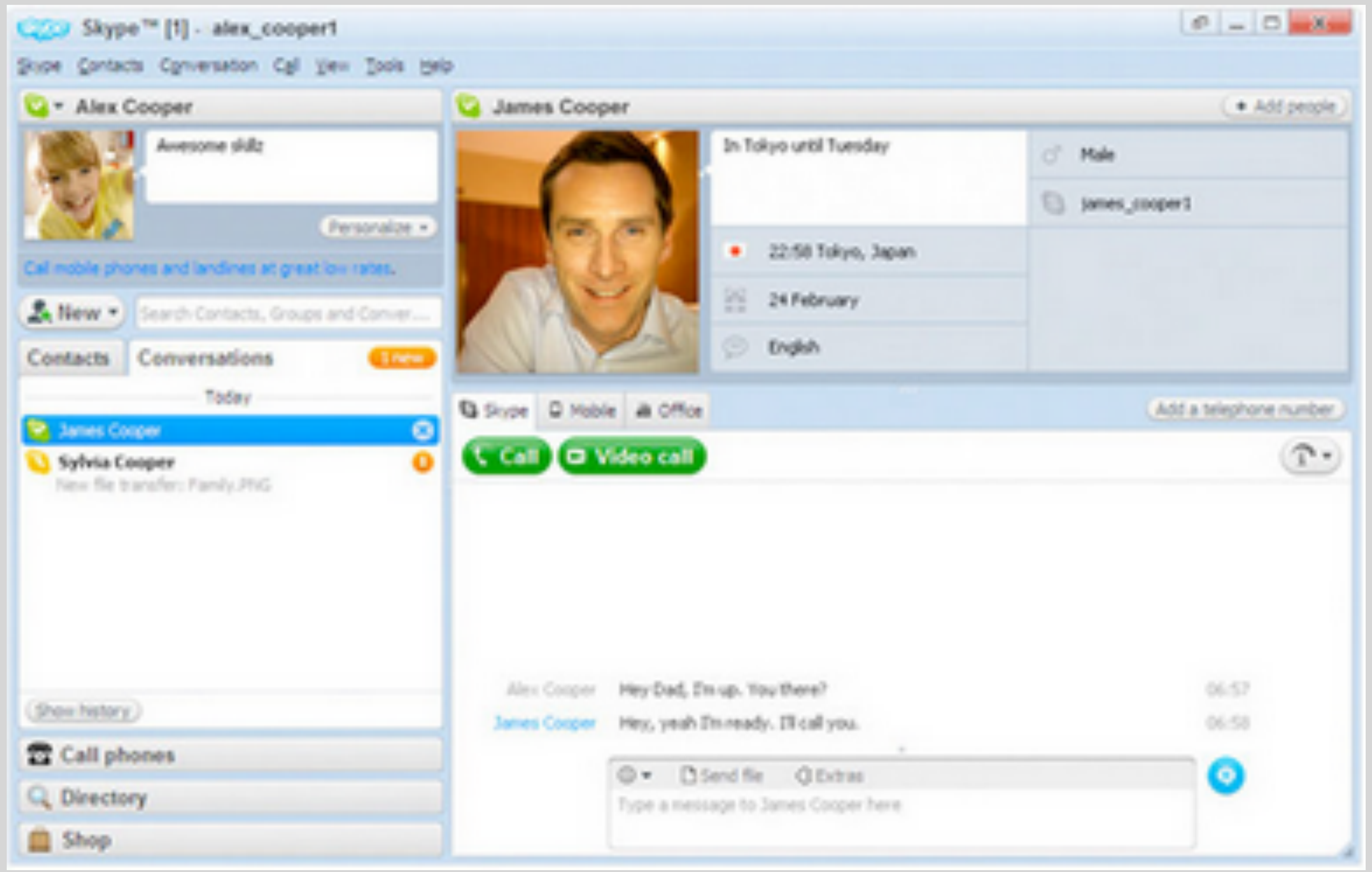
This morning I chaired the first ever digital Cabinet.

Our message to the public is: stay at home, protect the NHS, save lives. #StayHomeSaveLives

March 31, 2020

were video calls new?

Picturephone (1964)



Skype™ [1] - alex_cooper1

Alex Cooper: Awesome skillz

James Cooper: In Tokyo until Tuesday

Male

james_cooper1

22:58 Tokyo, Japan

24 February

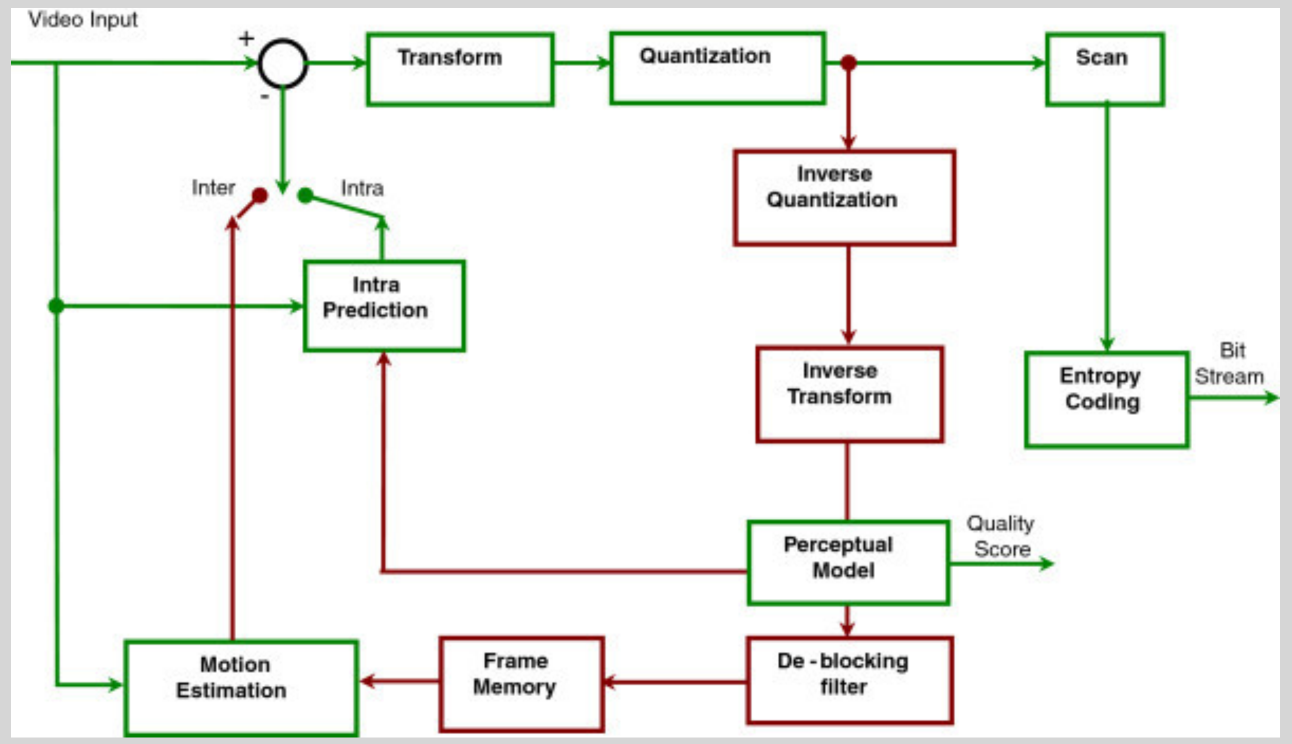
English

Sylvia Cooper — New file transfer: Family.PNG

Alex Cooper	Hey Dad, I'm up. You there?	06:57
James Cooper	Hey, yeah I'm ready. I'll call you.	06:58

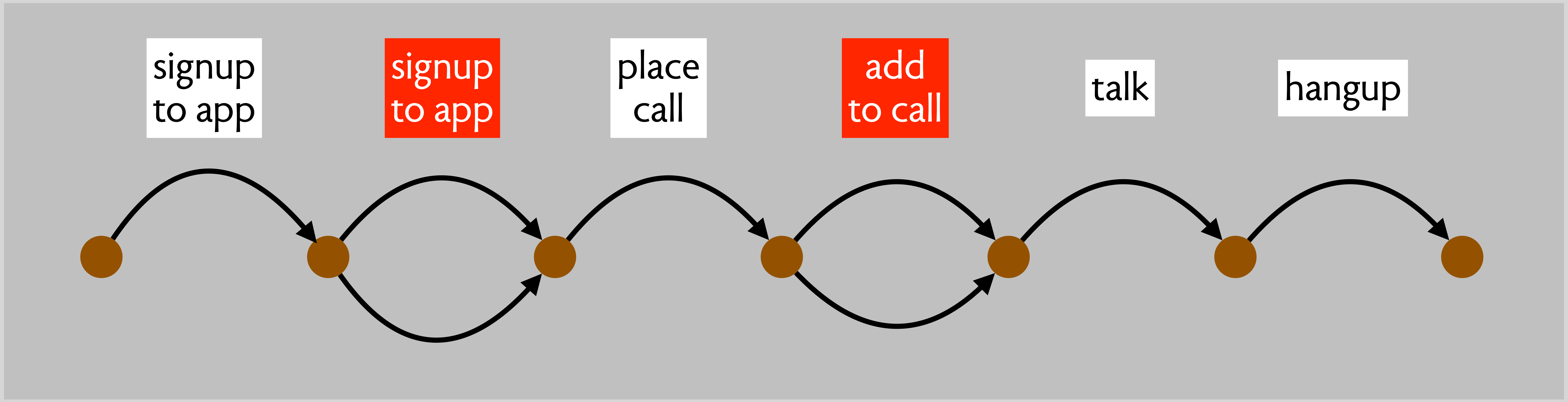
Skype (2009)

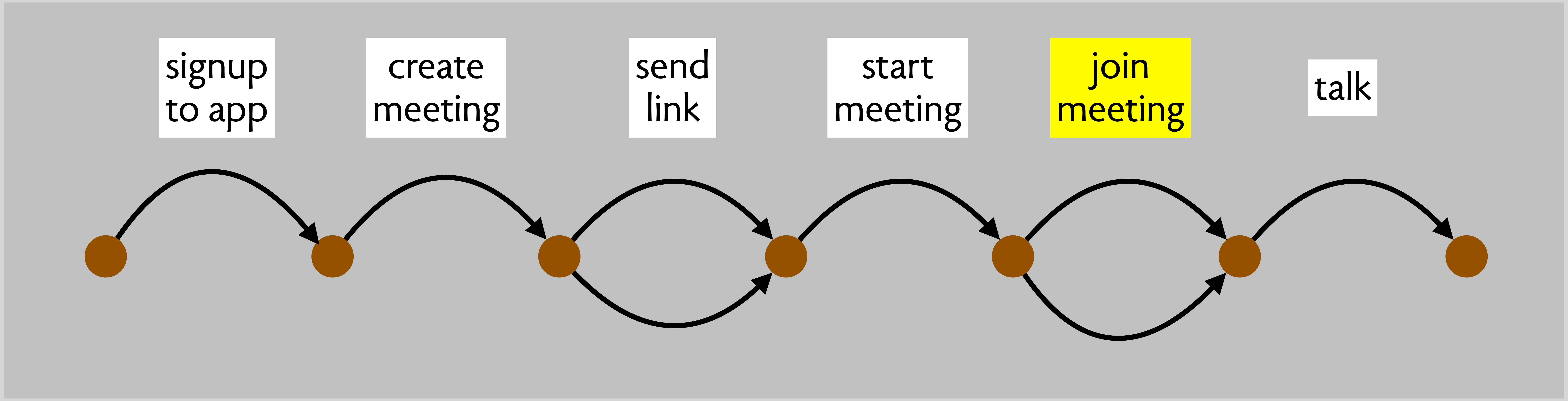
QuickCam (1994)
first commercial webcam



H.264 Video Codec (2003)

the meeting scenario





Zoom's meeting link concept

concept MeetingLink [Link, User]

purpose let users join meeting independently

principle meeting host creates a link and shares with participants, who can then join the meeting using the link

state

```
  a set of Meetings each with
    a Link
    a host User
    an active set of Users
```

actions

```
  create (host: User): (link: Link)
  join (user: User, link: Link): (meeting: Meeting)
  leave (user: User, meeting: Meeting)
```

three major design questions

what invariants? host vs. active users?

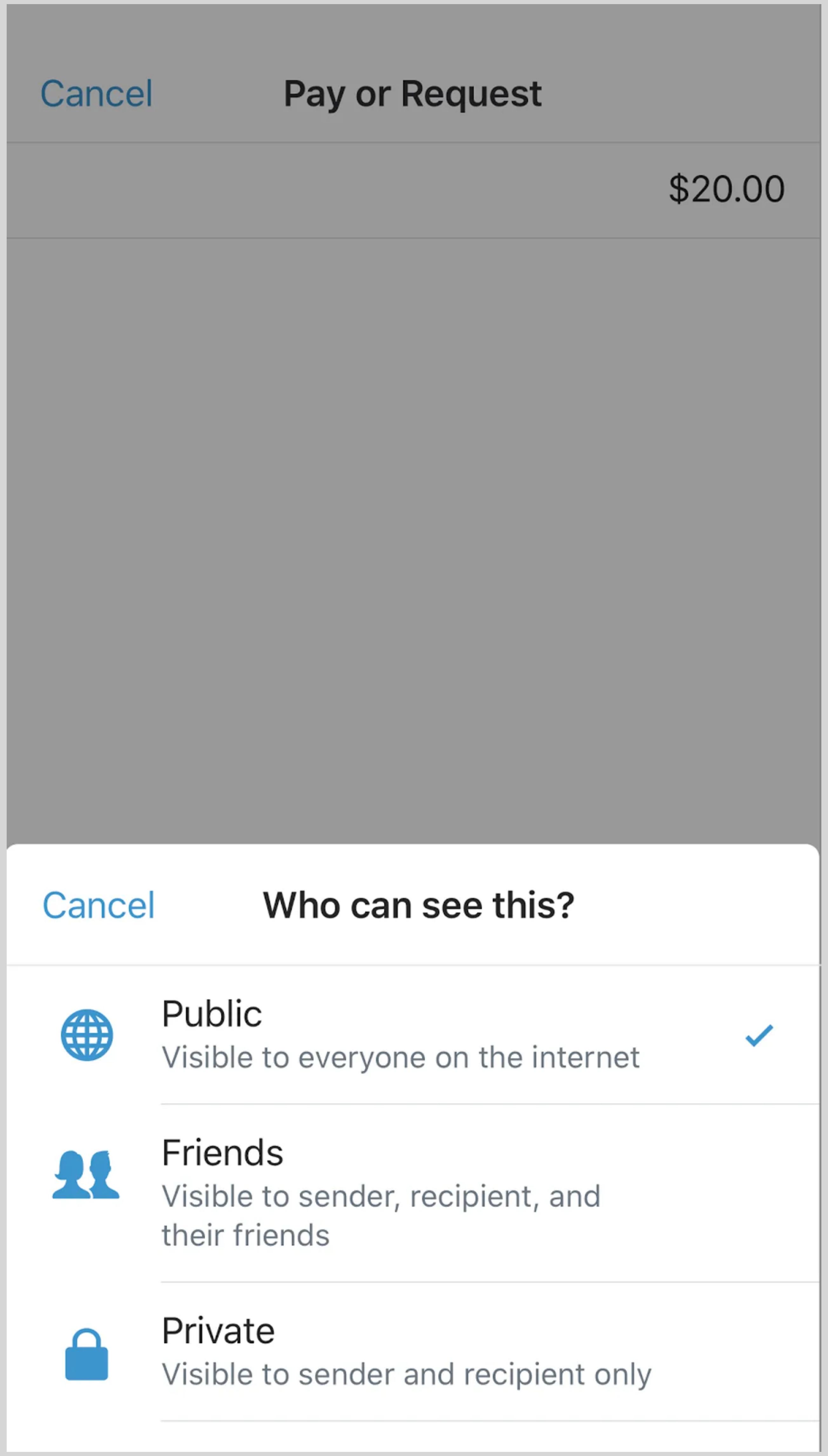
can meeting be restarted?

what will User be bound to?

tracing zoom's meeting concept

Skype
initially P2P (2003)
Microsoft (2013)
FaceTime
Apple (2010)
Google Hangouts
in Google+ (2011)
own product (2013)
Duo replaces (2016)
Zoom
Eric Yuan (2013)
Google Meet
launched (2017)
absorbs Duo (2022)
Microsoft Teams
launched (2017)
meeting concept
added to Skype
April 2020
join me
Join.me
LogMeIn (2010)
meeting scenario
meeting scenario
added to Teams
June 2022?

a concept sync and an FTC settlement


Cancel Pay or Request

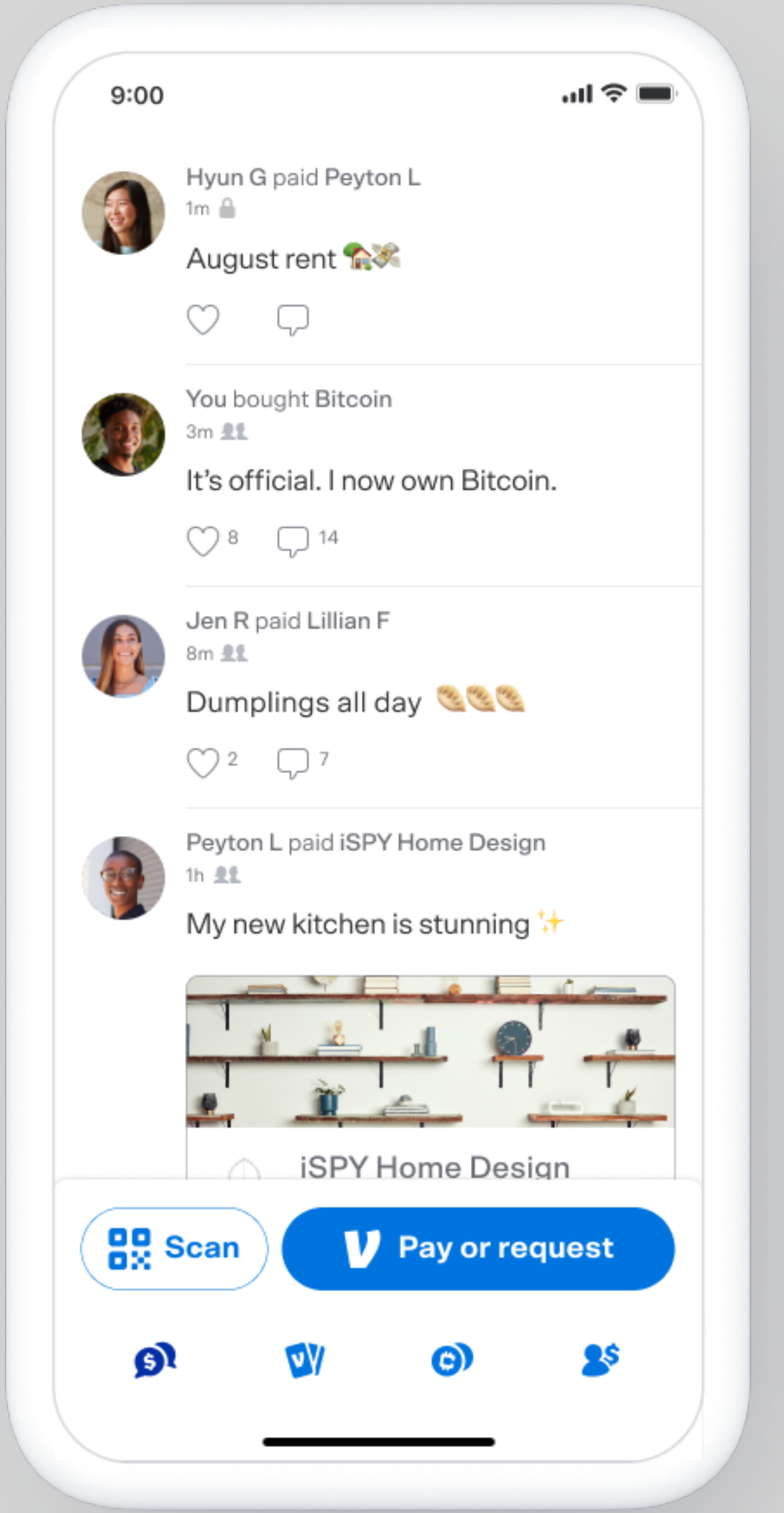
$20.00

Cancel Who can see this?

Public
Visible to everyone on the internet ✓

Friends
Visible to sender, recipient, and their friends

Private
Visible to sender and recipient only


9:00

Hyun G paid Peyton L
1m
August rent

You bought Bitcoin
3m
It's official. I now own Bitcoin.
8 14

Jen R paid Lillian F
8m
Dumplings all day
2 7

Peyton L paid iSPY Home Design
1h
My new kitchen is stunning

iSPY Home Design

The FTC settles with Venmo over a series of privacy and security violations

TechCrunch (Feb 2018)
"autofriending" and public posts by default

Venmo removes its global, public feed as part of a major redesign

TechCrunch (July 2021)

concept PeerToPeerPayment

purpose transfer funds between peers

```
actions
  transfer (from, to: User,
      amount: Dollar,
      memo: String)
  …
```

concept SocialFeed

purpose share short content publicly

```
state
  a set of Posts each with
    an author User
    a content String
```

```
actions
  post (author: User, content: String)
  …
```

when PeerToPeerPayment.transfer (from, to, memo)
then SocialFeed.post (from, memo)

a default synchronization

takeaways

the two aspects of a concept

users' perspective
a behavioral protocol

software perspective
a "nanoservice" or API

concept elements
name, purpose, principle, state+actions

externalizing connections
syncs on concept actions, no direct refs

concepts change our behavior
how to use an elevator, eg

concepts can eliminate friction
meeting links vs explicit groups, eg

many social problems from syncs
Venmo's public transaction feed, eg