design by concept

Daniel Jackson · CSAIL, MIT

9th Summer School on Formal Techniques · Menlo College, CA · May 19-24, 2019
lecture
one
how this project started
Field Trip Permission Form

Dear Parents:

Ms. Frizzle will again be taking her second grade class on an exciting field trip. Please sign and return the permission slip below.

Thank you!

Yes, I give permission for my child to go on the second grade “Touch and Feel” trip on Friday February 13th to the NastyCo Nuclear Dump. I understood that my child may encounter the normal risks of childhood play, including grazed knees, hurt feelings and exposure to toxic waste.

Count Olaf

Parents signature

Date
acrobat to the rescue?

Ms. Frizzle will again be taking her fifth-grade class on an exciting field trip. Please sign and return the permission slip below.

Thank you!

how to add a signature in acrobat
-- open document in acrobat
-- Tools → Advanced Editing → Touchup Object Tool
-- right click at desired point | Place Image...
then select jpg

how to add date
-- Tools → Typewriter
any idea what’s going on here?
adobe lightroom: easy cropping
i’m not alone

from http://amplicate.com
what’s the essence of the problem?

not the user interface!
polished and organized

task-oriented design
no unifying concepts

complex concepts
cropping vs. resizing
Adobe Fixes Acrobat

Version 9 (2008)
- Task-oriented design
- No unifying concepts

Version 10 (2010)
- Improved interface
- But still no concepts

Version 11 (2012)
- Unifying concepts
- Text/image object
a research & teaching program

designing software with concepts

- **a design theory**
  - concept structure & design rules
    - [Perez De Rosso, Onward 15]
- **design case studies**
  - Gitless
    - [Perez De Rosso, Onward 13, OOPSLA 16]
- **design patterns**
  - about 30 so far
- **code platform**
  - Deja Vu
    - [Perez De Rosso]
how bugs led us astray
the software problem

need
the motivation for building the system

implementation
the mechanisms of the system
separating concerns

- need: the motivation for building the system
- specification: the planned behavior of the system
- implementation: the mechanisms of the system

- pleasantness
- correctness

what we devoted ourselves to
what mattered more?

more?
correct ⇒ useful?

Any Gmail users who can explain?
correct $\Rightarrow$ safe?

- airborne $\Leftrightarrow \neg$ WheelPulse
- $\neg$ WheelPulse $\Leftrightarrow$ disabled
- airborne $\Leftrightarrow$ disabled

environment $\wedge$ specification $\Rightarrow$? requirement
We recently ran a password checker to evaluate passwords of all CSAIL users, and your password was readily broken. Please choose a new password ASAP…

my password:

sergeantpepper1967

8 char limit: passwd utility silently truncated rest

Aydal [2009]
Analyzed Tokeneer for security
Found 9 anomalous scenarios
eg, new configuration file silently ignored if one exists on disk
what's a concept?
What’s the difference between a text editor and a word processor?
app classes characterized by concepts too

**text editor**
- line buffer

**word processor**
- paragraph
- format
- style

**desktop publishing app**
- stylesheet
- text flow
- page template

---

**jamonh**  
Oct 22, 2013 7:19 PM  

Just upgraded to the new Pages and can't find a way to link text boxes anymore like

http://www.macobserver.com/tmo/article/pages-linking-text-boxes

Am I missing something, or is it really not possible anymore?
A mental model captures ideas in a problem domain, while a conceptual model represents 'concepts' (entities) and relationships between them.

A conceptual model in the field of computer science is a special case of a general conceptual model. To distinguish from other types of models, it is also known as a domain model. Conceptual modeling should not be confused with other modeling disciplines such as data modelling, logical modelling and
concepts are invented, not just out there

Tim Mott visits Ginn in 1974 brings idea of styles to PARC

Charles Simonyi’s team implements style in Bravo text editor

Simonyi brings style to Microsoft in 1983
the rewards of inventing a good concept

who is this and what is he doing?
concepts have purpose

purpose of style: enable consistent formatting

Apple Keynote adds style concept (2017?)
There is no problem in computer science that cannot be solved by introducing another level of indirection.  

*David Wheeler*
concepts are reusable

Powerpoint color schemes

Indesign swatches

Keynote image styles
not an instance of style

what crucial action is missing?
explaining concepts

how to explain the style concept?

If you assign Heading to two paragraphs and then you change the style from bold to italic, both paragraphs will be changed in concert.

If you create a style and assign to two elements, then when you modify the style, both elements will change...

not ontological: “a style is a mapping…”

not redundant: unlike full spec, shows how concept meets purpose

Johnson-Laird: constructive semantics

Michael Polanyi: operational principle
when concepts don’t fulfill purpose

Alexander’s misfits:
not bugs but bad specs
concepts: modules of behavior

- inventive
- purposeful
- behavioral
- self-contained
- reusable
concept models
**name** reservation

**purpose** make access to shared resource reliable

**structure**
- slots: Owner -> Slot
- holds: User -> Slot

**behavior**
- create (o: Owner, s: Slot)
  ```
  no slots.s => slots += o -> s
  ```
- reserve (u: User, o: Owner, s: Slot)
  ```
  no holds.s and o -> s in slots => holds += u -> s
  ```
- cancel (u: User, s: Slot)
  ```
  u -> s in holds => holds -= u -> s
  ```
- use (u: User, o: Owner, s: Slot)
  ```
  u -> s in holds and o -> s in slots =>
  ```

**tactic**
```plaintext
if create(o,s); reserve(u,o.s); ... no cancel(u,s) ... then can use(u,o,s)
```
A relational diagram:

- **Owner**
- **Slot**
- **User**
- **Taken**

- **Box represents set of atomic things**
- **Arrow represents binary relation**
- **Taken is a subset of Slot**
- **Multiplicty: each Taken has one user that holds it**

- **Owner** has ! slots to **Slot**
- **User** holds ! to **Taken**
- **Slot** holds ! to **Taken**
**alloy expressions in one slide**

- **u**: User
- **s**: Slot
- **holds**: User -> Slot
- **Taken**: set Slot

**a relation is a table of rows**
holds = {((u1,s1), (u1,s2))}
holds’ = {((u1,s1), (u1,s2), (u2,s3))}

**a set is a relation with one column**
Slot = {(s1), (s2), (s3), (s4)}
Taken = {((s1), (s2), (s3))}

**a scalar is a set with one row**
u = {((u2)}
s = {((s3))}

**set operators**
+ union, - difference, & intersection, in subset

Slot - Taken = {{(s4)}}
holds’ - holds = {{(u2,s3)}}

**relation operators**
-> product
  . join

**product examples**
u -> s = {{(u2,s3)}}
u -> Taken = {{(u2,s1), (u2,s2), (u2,s3)}}

**join examples**
u.holds’ = {{(s3)}}
holds’.s = {{(u2)}}
holds.Slot = {{(u1)}}

**formula examples**
holds’ = holds + u -> s
(also written holds += u -> s
User.holds = Taken
holds in User -> Slot

```
a -> b = { (a_0,..., a_n, b_0,..., b_m) | (a_0,..., a_n) ∈ a ∧ (b_0,..., b_m) ∈ b}
a . b = { (a_0,..., a_n-1, b_1,..., b_m) | (a_0,..., a_n) ∈ a ∧ (a_n, b_1,..., b_m) ∈ b}
```
## a reservation concept

<table>
<thead>
<tr>
<th>name</th>
<th>reservation</th>
</tr>
</thead>
<tbody>
<tr>
<td>purpose</td>
<td>make access to shared resource reliable</td>
</tr>
</tbody>
</table>
| structure | slots: Owner -> Slot  
           holds: User -> Slot |

### behavior

- **create** (o: Owner, s: Slot)
  
  ```
  no slots.s => slots += o -> s
  ```

- **reserve** (u: User, o: Owner, s: Slot)
  
  ```
  no holds.s and o -> s in slots => holds += u -> s
  ```

- **cancel** (u: User, s: Slot)
  
  ```
  u -> s in holds => holds -= u -> s
  ```

- **use** (u: User, o: Owner, s: Slot)
  
  ```
  u -> s in holds and o -> s in slots =>
  ```

### tactic

```c
if create(o,s); reserve(u,o.s); ... no cancel(u,s) ... then can use(u,o,s)
```
checking a tactic with electrum

sig Slot {}
sig Owner {var slots: set Slot}
sig User {var holds: set Slot}

pred create [o: Owner, s: Slot] {
  no slots.s
  slots' = slots + o -> s
  holds' = holds
}

pred reserve [u: User, o: Owner, s: Slot] {
  no holds.s
  o -> s in slots
  holds' = holds + u -> s
  slots' = slots
}

pred cancel [u: User, s: Slot] {
  u -> s in holds
  holds' = holds - u -> s
  slots' = slots
}

pred can_use [u: User, o: Owner, s: Slot] {
  u -> s in holds and o -> s in slots
}

pred skip {slots' = slots and holds' = holds}

fact {
  no holds and no slots -- initially
  always (skip or some u: User, s: Slot, o: Owner |
    create[o,s] or reserve [u,o,s] or cancel[u,s])
}

check {
  -- can always use after reserve: not true
  all u: User, s: Slot, o: Owner |
    always (create[o,s] and after reserve[u,o,s]
      implies after after always can_use[u,o,s])

see: https://github.com/haslab/Electrum
Design by concept is a new approach to creating software. A software product—whether an app, a service or a system—is viewed as a collection of interacting concepts, each with its own purpose, structure and behavior. Concepts can be invented afresh, but they can also be reused, exploiting the knowledge embodied in previous successful designs.

This book explains what concepts are and why they are central to software design; shows examples of concepts (from the most effective and ingenious to the most flawed and frustrating) taken from well-known applications; and presents design principles that can identify and eliminate flaws in existing and new designs.

Daniel Jackson is Professor of Computer Science, a MacVicar fellow, and Associate Director of the Computer Science and Artificial Intelligence Lab at MIT. His past research focused on software modeling and analysis; he is the creator of the Alloy language, and author of Software Abstractions: Logic, Language, and Analysis (MIT Press; second ed. 2012). His current interests include software design for improved usability, security and safety, and new programming paradigms. He was a recipient of the 2016 ACM SIGSOFT Impact Award, the 2017 ACM SIGSOFT Outstanding Research Award, and is an ACM Fellow.
a rather different book

Portraits of Resilience

Daniel Jackson

foreword by David A. Karp

http://portraits-of-resilience.com
studio 1
identifying concepts: resy

here’s a typical reservation app. what concepts can you identify?
identifying concepts: stack exchange

another example: a typical Q&A app

Philosophy

All Questions

1,695 questions

74 votes
19 answers
32k views

Could 'cogito ergo sum' possibly be false?
I've heard it postulated by some people that "we can't truly know anything". While that does seem to apply to the vast majority of things, I can't see how 'cogito ergo sum' can possibly be false. ...

186 votes
29 answers
26k views

Was mathematics invented or discovered?
What would it mean to say that mathematics was invented and how would this be different from saying mathematics was discovered? Is this even a serious philosophical question or just a meaningless/...
gmail
surprises
organizing messages

Google

label:hacking

Gmail

COMPOSE

Inbox
Starred
Sent Mail
Drafts
Trash

Categories
Social
Promotions
Updates
Forums

hacking
meetups
todo
More

0 GB (0%) of 15 GB used
Manage

Terms - Privacy

Last account activity: 14 hours ago
Details

9:14 pm

Alyssa P. Hacker

javascript - Reminds you of the old da

1–1 of 1
automating filtering
slightly surprising behavior #1

in:sent

javascript  Inbox  hacking

Alyssa P. Hacker  Reminds you of the old days, eh?  9:14 PM (33 minutes ago)

Ben Bitdiddle  <benito.bitdiddle@gmail.com>  9:40 PM (7 minutes ago)

to Alyssa  ▼

Yes, it does.

Click here to Reply or Forward

0 GB (0%) of 15 GB used  Terms - Privacy  Last account activity: 26 minutes ago
slightly surprising behavior #2
slightly surprising behavior #3
slightly surprising behavior #5

<table>
<thead>
<tr>
<th>Label: Social</th>
<th>Social</th>
<th>Promotions</th>
<th>Buy this! - My new JS book is out</th>
<th>10:33 am</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label: Promotions</td>
<td>Promotions</td>
<td>Social</td>
<td>Buy this! - My new JS book is out</td>
<td>10:33 am</td>
</tr>
<tr>
<td>Label: Social Label: Promotions</td>
<td>Promotions</td>
<td>Social</td>
<td>Buy this! - My new JS book is out</td>
<td>10:33 am</td>
</tr>
</tbody>
</table>

No messages matched your search. Try using search options such as sender, date, size and more.
find a partner so you can work in a pair
pick one of the Gmail surprises
all slides at https://tinyurl.com/ssft9a
analyze it in term of concepts
what are the key concepts involved?
which concept(s) is responsible for the surprise?
is the surprise a bug, a conceptual flaw or a user misunderstanding?
can you explain precisely what’s going wrong?
can you generalize your observation?

design a fix
propose a modification that eliminates the surprise
lecture two
three
design
problems
<table>
<thead>
<tr>
<th>Primary</th>
<th>Social</th>
<th>Promotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>New sign-in from Chrome on Mac</td>
<td>23 new</td>
</tr>
<tr>
<td>Keith Muhammad at DeMont.</td>
<td>DeMontrond Auto Group - 14101 North Freeway Houston</td>
<td>12:19 pm</td>
</tr>
<tr>
<td>AT&amp;T High Speed Internet.</td>
<td>AT&amp;T High Speed Internet Service Activation - Your A1</td>
<td>10:37 am</td>
</tr>
<tr>
<td>Keith Muhammad at DeMont.</td>
<td>DeMontrond Auto Group - 14101 North Freeway Houston</td>
<td>Aug 26</td>
</tr>
<tr>
<td>betterbatonrougejobs.com</td>
<td>Job Update -- 2015-08-26 - Looking For An Advantage</td>
<td>Aug 26</td>
</tr>
</tbody>
</table>
Choose which message categories to show as inbox tabs. Other messages will appear in the Primary tab.
Google gets it terribly, terribly wrong with Gmail tabs and makes me angry

By Mark Wilson | Published 2 years ago | Follow

71 Comments

Like many Gmail users, I greeted the news

Quick Tech Tip: Disabling Gmail's Category Tabs

Mon, 07/29/2013 - 12:17 | Chuck Gray
in LibraryPoint Blog Tech Tutorials Teen Blog Tech Answers Science and Technology Self-Help and Instructional

Are you a Gmail user? Did you wake up a week or two ago to find that your new messages were now being automatically organized by Gmail into tabs of different, pre-determined categories? And, did you think, like me, that they were really ugly, stupid, and unnecessary? Here's a quick tip on how to rid yourself of them!

see that tabs were now available to me. Excitement was short-lived, however; it quickly became apparent that this new feature is a disaster.
Using labels

Labels help you organize your messages into categories – work, family, to do, read later, jokes, recipes, any category you want. Labels do all the work that folders do, but with an added bonus: you can add more than one to a message.
what you can’t do

associate tabs with labels
feature available only for categories

create new categories
only new labels

use tabs outside inbox
tabs disappear when you filter on a label
fuji aspect ratio
my camera fuji x100s
image quality setting
aspect ratio
image size setting
non-standard ratio + raw?
what you can’t do

non-standard aspect ratio + raw
even though raw images get nice nondestructive crop!
indesign
styles
what’s a font?
what you can’t do

define a style that italicizes
Arno Regular to Arno Italic
Futura Book to Futura Book Oblique
Magma Light to Magma Light Italic
introducing a concept

Keynote ’09: has subfamilies

Keynote 6: gone again!
what’s going on?

gmail
one purpose :: two concepts
organizing messages :: label + category

camera
two purposes :: one concept
aspect ratio + image resolution :: image size

style
one purpose :: no concept
specify a font-independent styling :: ?
the singularity rule
Mitchell and Webb: “Unity of Purpose”
one-to-one mapping

purposes

P1

C1

P2

C2

concepts

Nam Suh: Axiomatic Design
four ways to fail

unfulfilled purpose

overloaded concept

unmotivated concept

redundant concepts
kinds of overloading
overloaded concepts

No one can serve two masters. Either you will hate the one and love the other, or you will be devoted to the one and despise the other. [Matthew 6:24]

4 forms of overloading:
- **piggybacking** new purpose hacked onto old concept
- **false convergence** two purposes looked the same
- **emergent purpose** second purpose emerged with use
- **denial** designer believes second purpose unnecessary
piggybacking fuji camera

new purpose hacked onto old concept

image size
aspect ratio piggybacked on JPEG dimensions
piggybacking epson driver

result: can’t create custom size for front loading
also, page size presets in Lightroom hold feed setting
false convergence facebook friend

two purposes looked the same

```
filter incoming posts
control access to my posts
distinct purposes

2011: Facebook added subscribe/follow
```
users find second purpose for concept

initial purpose: summarize content

emergent purpose: show sender
if you bcc a list, subject reveals to-address

thanks to Shriram Krishnamurthi

emergent purpose: group by conversation

can’t label reservations from Expedia by trip

thanks to Eunsuk Kang
designer believes second purpose unnecessary
the uniformity rule
what makes a usable concept?

operational principle is uniform
always the same actions, irrespective of context

concept: **Group** (Keynote)
purpose: treat set as one
OP: ... select(objs); group(); mutate()...

unless objs contains a text body object
concept: Range (Numbers)
purpose: define formula over adjustable group of cells
OP: ... define formula over range... select(c) in range... add(direction)... formula updated

unless range cell c is at top of range and dir is above or....
non-uniformity conversation

action applied to every message in conversation **unless** message in other folder or action is reply ...
kinds of non-uniformity

- **varies over type**: Keynote grouping *unless* `objs` contains a text body object
- **varies over mode**: Fuji aspect ratio setting *unless* set to raw only mode
- **varies over state**: Dropbox share folder *unless* folder is ancestor or descendant of shared folder
- **varies over state**: Git branch *unless* working directory contains uncommitted file or...
- **varies over arg**: Twitter mention *unless* mention includes first character of tweet
the
genericity rule
how concepts get applied

upvote
notification
related

NY Times
comments
breaking news
articles

StackExchange
answers
replies
questions

Amazon
reviews
when shipped
items
why reuse a concept?

- familiarity
  - users will get it
- save work
  - design options known
- no surprises
  - misfits anticipated

- options for upvote?
- misfit of notification?
the genericity rule
reusing a well-known generic concept is usually preferable to inventing one.
concept composition
### example: reservation

<table>
<thead>
<tr>
<th>Name</th>
<th>reservation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purpose</td>
<td>make access to shared resource reliable</td>
</tr>
<tr>
<td>Structure</td>
<td>reserved: bool = false</td>
</tr>
</tbody>
</table>
| Behavior | reserve()  
|           |     reserved := true  
|           | use ()  
|           |     reserved => reserved := false  
|           | cancel ()  
|           |     reserved => reserved := false |
| Tactic  | if reserve() and no cancel then can use() |

**Structure** defines the state space:

- `<>`
- `<reserve>`
- `<reserve, cancel>`
- `<reserve, use>`
- `<reserve, use, cancel>`
- ...

**Actions** give a labeled transition relation defining a trace set:

```plaintext
{<>,  
  <reserve>,  
  <reserve, cancel>,  
  <reserve, use>,  
  <reserve, use, cancel>,  
  ...
}
```
example: authentication

<table>
<thead>
<tr>
<th>name</th>
<th>authentication</th>
</tr>
</thead>
<tbody>
<tr>
<td>purpose</td>
<td>identify participant in interaction</td>
</tr>
<tr>
<td>structure</td>
<td>ok: bool = false</td>
</tr>
<tr>
<td>behavior</td>
<td>login() ok := true logout() ok =&gt; ok := false auth() ok =&gt;</td>
</tr>
<tr>
<td>tactic</td>
<td>if login() and no logout() then can auth()</td>
</tr>
</tbody>
</table>

what are the traces?

```{<>,
<login>,
<login, auth>,
<login, auth, auth>,
<login, logout>,
<login, auth, logout>,
...}
```
composing concepts

application

MyReservationApp

includes

reservation, authentication

behavior

login
  authentication.login
logout
  authentication.logout
reserve:
  reservation.reserve
  authentication.auth
cancel:
  reservation.cancel
  authentication.auth
use:
  reservation.use

concepts used

action of app is binding of concept actions

binding: Action -> Concept -> Action

{(login, authentication, login),
 (logout, authentication, logout),
 (reserve, reservation, reserve),
 (reserve, authentication, auth),
 (cancel, reservation, cancel),
 (cancel, authentication, auth),
 (use, reservation, use)}
semantics of composition

traces of **reservation**

\{
\langle >, \\
\langle reserve >, \\
\langle reserve, cancel >, \\
\langle reserve, use >, \\
\langle reserve, use, cancel >, \\
\ldots
\}

traces of **authentication**

\{
\langle >, \\
\langle login >, \\
\langle login, auth >, \\
\langle login, auth, auth >, \\
\langle login, logout >, \\
\langle login, auth, logout >, \\
\langle login, auth, logout >, \\
\ldots
\}

binding: Action \rightarrow Concept \rightarrow Action

\{
\langle login, authentication, login \rangle, \\
\langle logout, authentication, logout \rangle, \\
\langle reserve, reservation, reserve \rangle, \\
\langle reserve, authentication, auth \rangle, \\
\langle cancel, reservation, cancel \rangle, \\
\langle cancel, authentication, auth \rangle, \\
\langle use, reservation, use \rangle
\}

map trace \( t \) onto concept \( C \) with binding \( B \)

\[
\begin{align*}
\text{let } & \text{map}(t, C, B) = \\
& \text{map} \left( \langle > \right), C, B) = \langle > \\
& \text{map} \left( \langle \text{append}(t, a) \rangle \right), C, B) = \\
& \quad \text{if no } C.(a.B) \text{ then } \text{map}(t, C, B) \\
& \quad \text{else append} \left( \text{map}(t, C, B), C.(a.B) \right)
\end{align*}
\]

traces are all those consistent with concept traces

\[
\text{traces} = \{ t \in \text{action}^* \mid \forall C: \text{includes I map}(t, C, B) \in \text{traces}(C) \}
\]
reservation (again)

name reservation

purpose make access to shared resource reliable

structure slots: Owner -> Slot
holds: User -> Slot

behavior
create (o: Owner, s: Slot)
  no slots.s => slots += o -> s
reserve (u: User, o: Owner, s: Slot)
  no holds.s and o -> s in slots => holds += u -> s
cancel (u: User, s: Slot)
  u -> s in holds => holds -= u -> s
use (u: User, o: Owner, s: Slot)
  u -> s in holds and o -> s in slots =>

tactic if create(o,s); reserve(u,o.s); ... no cancel(u,s) ... then can use(u,o,s)
authentication (again)

name: authentication

purpose: identify participant in interaction

structure:
- password: User -> Password
- sessions: set User

behavior:
- register (u: User, p: Password)
  - no u.password => password += u -> p
- login (u: User, p: Password)
  - u.password = p => sessions += u
- logout (u: User)
  - u in sessions => sessions -= u
- auth (u: User)
  - u in sessions =>

tactic: if register(u,p), login(u,p), no logout(u) then can auth(u)
**name**

rating

**purpose**
identify participant in interaction

**structure**

used: User -> Item  
rated: User -> Item -> Int  
rating: Item -> Int = \{i: Item, r: Int \mid \text{avg (User, rated)}\}

**behavior**

use (u: User, i: Item)  
\text{used += u -> i}

rate (u: User, i: Item, r: Int)  
\text{u -> i in used => u.rated += i -> r}

show (i: Item): Int  
\text{result = i.rating}

**tactic**
if user(u,i), rate(u,i,r)… for multiple u… and show(i):r then r is avg of user’s ratings
reservation app (again)

application
MyReservationApp

includes
reservation, authentication, rating

behavior
register(u,p)
authentication.register(u,p)
login(u,p)
authentication.login(u,p)
logout(u)
authentication.logout(u)
reserve(u,o,s)
reservation.reserve(u,o,s)
authentication.auth(u)
use(u,o,s)
reservation.use(u,o,s)
rating.use(u,o)
cancel(u,s)
reservation.cancel(u,s)
authentication.auth(u)
rate(u,o,r)
authentication.auth(u)
rating.rate(u,o,r)
showRating(o)
rating.show(o)

ratings are authenticated

can’t rate until you’ve used reservation
the integrity rule
looking at sent messages in gmail

can’t see which messages were sent
interpreting composite behavior

each action in composite system interpreted as zero or more actions in each concept
the integrity rule

when concepts are combined, each concept’s behavior and OP should still apply

a simple criterion

projected behavior must satisfy concept spec:

∀ c: concept | ∀ t: traces(sys) | R_c(t) ∈ traces(c)
the label concept

name: label

purpose: organize items for easy retrieval

structure: labels: X -> Label

behavior:
mark (x: X, p: Label)
  labels += x -> p
unmark (x: X, p: Label)
  p in x.labels => labels -= x -> p
find (ps: set Label): set X
  result = {x | ps in x.labels}

tactic:
if mark(x, p); find(p):xs then x in xs
if no mark(x, P); find(p):xs then !in xs
conversation breaks label

when message \( m \) is sent
\[
\text{Label.mark}(m, \text{`sent'})
\]
occurs implicitly

when Sent link is clicked
\[
\text{Label.find(`sent')}:ms
\]
occurs

but \( ms \) includes
messages never marked
interaction of Trash and Volume (Apple Finder)
unmount of Volume removes files from Trash
not expressible in terms of Trash actions
a solution: one trash/volume?

what happens when you unmount a drive?
deja vu
reversing the process

the same concepts, again & again
post, comment, upvote, notification, ...

hard work to build
libraries often just client- or server-side
easy in a CMS, but structure hard-wired

idea: concept cliches
full stack implementation

app-specific assembly
in HTML, no JS or backend code

action synchronization
build app action by joining cliche actions
architecture of deja vu

client

post

comment

upvote

app

action

client-side

library

gateway

post

comment

upvote

server
a sample app

```
{
  "name": "hackernews",
  "usedCliches": {
    "authentication": {},
    "comment": {},
    "property": {...},
    "scoringposts": {"name": "scoring"}
  }
}

"routes": [
  { "route": "", "action": "home" },
  { "route": "news", "action": "home" },
  { "route": "post", "action": "post-detail" },
  { "route": "login", "action": "login" },
  { "route": "submit", "action": "submit-post" }
]
```
<table>
<thead>
<tr>
<th>Title</th>
<th>Score</th>
<th>Author</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPLASH 2018 (2018.splashcon.org)</td>
<td>3</td>
<td>eva</td>
<td>comments</td>
</tr>
<tr>
<td>Software Design Group  (sdg.csail.mit.edu)</td>
<td>2</td>
<td>eva</td>
<td>comments</td>
</tr>
<tr>
<td>Concept design         (sdg.csail.mit.edu)</td>
<td>1</td>
<td>alyssa</td>
<td>comments</td>
</tr>
<tr>
<td>Déjà Vu                (sdg.csail.mit.edu)</td>
<td>1</td>
<td>ben</td>
<td>comments</td>
</tr>
</tbody>
</table>

```html
<dv.action name="home">
<hackernews.navbar />
<div class="main">
  <scoringposts.show-targets-by-score
    noTargetsText="No posts yet"
    showAscending=false
    showScores=false
    showTarget=<hackernews.show-post post=$target id=$id />
  </scoringposts.show-targets-by-score>
</div>
</dv.action>
```
submit post action

```html
dv .action name="submit-post">
<hackernews.navbar />
div class="main">
<dv tx>
<dv.gen-id />
<property.create-object
  id=dv.gen-id.id
  initialValue={ author: hackernews.navbar.user.username }
  showExclude=["author"]
  buttonLabel="submit"
  newObjectSavedText="Post submitted" />
<scoringposts.create-score
  targetId=dv.gen-id.id
  value=0
  hidden=true />
<authentication.authenticate id=hackernews.navbar.user hidden=true />
dv.link href="/item" params={ id: dv.gen-id.id } />
</dv.tx> </div> </dv.action>
```
rebuilding class projects

<table>
<thead>
<tr>
<th></th>
<th>App</th>
<th>Authentication</th>
<th>Authorization</th>
<th>Passkey</th>
<th>Transfer</th>
<th>Rating</th>
<th>Follow</th>
<th>Geolocation</th>
<th>Comment</th>
<th>Event</th>
<th>Label</th>
<th>Group</th>
<th>Task</th>
<th>Scoring</th>
<th>Property</th>
<th>Chat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accord</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ChoreStar</td>
<td></td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MapMIT</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rendezvous</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SweetSpots</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Potluck</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GroceryShip</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lingua</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>LiveScorecard</td>
<td></td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

cliqué
other aspects of deja vu

WYSIWYG designer (Barry McNamara)

cliche library for social apps (Maryam Archie)

security: stop request forgeries

gateway

cliche support

make it easier to author cliches (Czarina Lao)
closing
thoughts
user-centered design: conceptual model should be designed
formal methods: software defined by its behavior
both originating around 1974
https://tinyurl.com/dbctouch
to keep in touch and be notified about publication of book

https://tinyurl.com/postcard-get
to sign up for monthly resilience postcards
studio 2
construct concept models in this order
post, friend, comment, upvote, tag

specify application binding

for each concept, give
purpose: informally stated
structure: text or diagram
behavior: actions specified formally
tactic: informal scenario

hints: make each concept
minimal: only essential functionality
free-standing: makes sense alone
orthogonal: avoid overlap

what issues came up?
reminder: a reservation concept

name: \textit{reservation}

purpose: make access to shared resource reliable

structure:
- slots: Owner -> Slot
- holds: User -> Slot

behavior:
- create (o: Owner, s: Slot)
  \textbf{no} slots.s => slots += o -> s
- reserve (u: User, o: Owner, s: Slot)
  \textbf{no} holds.s \textbf{and} o -> s \textbf{in} slots => holds += u -> s
- cancel (u: User, s: Slot)
  u -> s \textbf{in} holds => holds -= u -> s
- use (u: User, o: Owner, s: Slot)
  u -> s \textbf{in} holds \textbf{and} o -> s \textbf{in} slots =>

tactic:
\textbf{if} create(o,s); reserve(u,o.s); ... \textbf{no} cancel(u,s) ... \textbf{then can} use(u,o,s)
reminder: alloy expressions in one slide

**set operators**
- union, - difference, & intersection, in subset

Slot - Taken = {((s4)}
holds' - holds = {((u2,s3)}

**relation operators**
- product
  - join

**product examples**
u -> s = {((u2,s3)}
u -> Taken = {((u2,s1), (u2,s2), (u2,s3)}

**join examples**
u.holds' = {((s3)}
holds'.s = {((u2)}
holds.Slot = {((u1)}

**formula examples**
holds' = holds + u -> s
(also written holds += u -> s
User.holds = Taken
holds in User -> Slot

| a -> b = { (a0,..., an, b0,..., bm) | (a0,..., an) ∈ a ∧ (b0,..., bm) ∈ b} |
| a.b = { (a0,..., an-1, b1,..., bm) | (a0,..., an) ∈ a ∧ (an, b1,..., bm) ∈ b} |