a new way to think about specifications

Daniel Jackson · CSAIL, MIT

ABZ · June 7, 2018
a different project

Portraits of Resilience
Daniel Jackson
foreword by David A. Karp

http://portraitsofresilience.com
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

what we devoted ourselves to

what mattered more?

pleasantness

correctness
correct ⇒ useful?

label:hacking

label:meetups

label:hacking label:meetups

No messages matched your search. Try using search options such as sender, date, size and more.
correct ⇒ useful?

correctness & pleasantness are not independent

Empty Trash now (messages that have been in Trash more than 30 days will be automatically deleted)

- me, Alyssa (13)
- Andy from Google

There are no conversations with this label.

label:todo

label:todo label:trash

label:todo OR label:meetup

Some messages in Trash or Spam match your search. View messages.
correct ⇒ useful?
correct $\Rightarrow$ safe?

- $\text{airborne} \Leftrightarrow \neg \text{WheelPulse}$
- $\neg \text{WheelPulse} \Leftrightarrow \text{disabled}$
- $\text{airborne} \Leftrightarrow \text{disabled}$

$\text{environment} \land \text{specification} \Rightarrow ? \text{ requirement}$

Airbus A320, Warsaw 1993
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
a research program

explore software design
what makes a good spec

a design theory
design case studies
design patterns
code platform
how is design done?
simple design
complicated design

architecture
graphic design
music
software
core concepts: a universal design strategy

architecture

graphic design

identity

massing

music

motif

mit media lab
core concepts for software

- **Apple iPod**: core concepts: *buffer*
- **WhatsApp**: core concepts: *song, playlist*
- **Slack**: core concepts: *message, channel, mention*
- **Word (Microsoft)**: core concepts: *paragraph, styles*
- **InDesign (Adobe)**: core concepts: *paragraph, styles, linkedBox, page*
- **Photoshop (Adobe)**: core concepts: *pixel map, layer/mask*
concept basics
<table>
<thead>
<tr>
<th>Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inventive</td>
<td>so not just modeling</td>
</tr>
<tr>
<td>purposeful</td>
<td>so not an action</td>
</tr>
<tr>
<td>behavioral</td>
<td>so not an entity</td>
</tr>
<tr>
<td>self-contained</td>
<td>so not a feature or an abstract type</td>
</tr>
<tr>
<td>reusable</td>
<td>so not a feature</td>
</tr>
</tbody>
</table>
**trash**

- **inventive**: first in Apple Lisa (1982) not really about the GUI (despite Apple vs Msft, 1994)

- **purposeful**: purpose is: to undo deletions!

- **behavioral**: to delete a file, move to trash; can restore from there; to make space, empty trash

- **self-contained**: synergistic with file system but concept requires only set of objects

- **reusable**: messages (Gmail) photos (iPhoto) posts (WordPress) notes (Evernote)

**reservation**

- **inventive**: for restaurants, started in 19th century (tables and rooms)

- **purposeful**: purpose is: efficient allocation of resources

- **behavioral**: select resource to reserve; request reservation; make use of resource

- **self-contained**:

- **reusable**: restaurant tables airplane seats medical appointments …
The trash concept

**Purpose**
allow undo of deletion

**State**
all, deleted: set $X$

**Actions**
new (): $X \triangleq \text{result} \notin \text{all and all'} = \text{all} + \text{result}$
del (x: $X) \triangleq \text{deleted'} = \text{deleted} + x$
empty () $\triangleq \text{no deleted'} \text{ and all'} = \text{all} - \text{deleted}$

**Spec**
showAll () : set $X \triangleq \text{result} = \text{all}$

**Operational Principle**
del(x) ... not empty() ... showAll():xs $\Rightarrow x$ in xs
del(x) ... empty() ... showAll():xs $\Rightarrow x$ not in xs
The label concept

**Purpose**
Organize items for easy retrieval

**State**
labels: \( X \rightarrow \text{Label} \)

**Actions**
- mark \((x: X, p: \text{Label}) \triangleq \text{labels}' = \text{labels} + x \rightarrow p\)
- unmark \((x: X, p: \text{Label}) \triangleq \text{labels}' = \text{labels} - x \rightarrow p\)

**Spec**
- find \((ps: \text{set Label})\): \( \text{set} \ X \triangleq \text{result} = \{x \mid ps \in x.\text{labels}\} \)
- show \((x: X)\): \( \text{set} \ \text{Label} \triangleq \text{result} = x.\text{labels} \)

**Operational principle**
- \( \text{mark}(x,p) \ldots \text{not unmark}(x,p) \ldots \text{search}(p):xs \Rightarrow x \in xs \)
- \( \text{not mark}(x,p) \ldots \text{search}(p):xs \Rightarrow x \text{ not in } xs \)
the singularity rule
one-to-one mapping

purposes

P1

concepts

C1

P2

C2
four ways to fail

unfulfilled purpose

P1 — C1

unmotivated concept

P1 — C1

overloaded concept

P1 — C1

P2

redundant concepts

P1 — C1

P2

C2
my camera fuji x100s
image quality setting
aspect ratio
non-standard ratio + raw?
what you can’t do

non-standard aspect ratio + raw
even though raw images get nice nondestructive crop!
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
emergent purpose email subject

users find second purpose for concept

initial purpose: summarize content

To: Daniel Jackson <dnj@mit.edu>
Re: Catch me if you can in real life!

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

thanks to Shriram Krishnamurthi

To: csail-related@lists.csail.mit.edu
Re: [csail-related] turn off the lights?

emergent purpose: group by conversation

thanks to Eunsuk Kang

can’t label reservations from Expedia by trip
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(f, c, R) ... new(rc, dir): nc ... enter(nc,v) ... show(c)

unless range cell rc 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 integrity rule
interpreting composite behavior

each action in composite system interpreted as zero or more actions in each concept
action is \texttt{Label.show}(p):ms
where \( p \) is the label \textit{sent}
and \( ms \) is the set of two messages listed

prior sending of msg \( m \) was an instance of action \texttt{Label.mark}(m, p)
where \( p \) is the label \textit{sent}
a simple criterion
projected behavior must satisfy concept spec:
∀ c: concept | ∀ t: traces(sys) | R_c(t) ∈ traces(c)
example gmail

action is `Label.search(p):ms` where p is the label sent and ms is the set of two messages listed prior sending of msg m was an instance of action `Label.mark(m, p)` where p is the label sent

violates spec of `Label.search`
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
integrating concepts to make apps

Deja Vu (Santiago Perez De Rosso)

library of polymorphic concepts, each implementing **full stack** composed in HTML by linking actions

<table>
<thead>
<tr>
<th>app</th>
<th>Accord</th>
<th>ChoreStar</th>
<th>MapMIT</th>
<th>Rendezvous</th>
<th>SweetSpots</th>
<th>Potluck</th>
<th>GroceryShip</th>
<th>Lingua</th>
<th>LiveScorecard</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></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>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>cliché</th>
<th>Authentication</th>
<th>Authorization</th>
<th>Passkey</th>
<th>Market</th>
<th>Rating</th>
<th>Follow</th>
<th>Geolocation</th>
<th>Post</th>
<th>Comment</th>
<th>Event</th>
<th>Label</th>
<th>Group</th>
<th>Task</th>
<th>Checklist</th>
<th>Property</th>
<th>Scoring</th>
<th>Chat</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
closing thoughts
finding bugs in design: what kind?

mantras: to reduce costs, find bugs early

but what kinds of bugs?

corner cases easily found & fixed later

snags vs snafus

simulation > proof
models in design vs engineering

what engineers use models for?
calculating properties, checking against higher level specs
to find bugs

what designers actually use models for
simulating experience of finished product
to find unanticipatable misfits

shallow flaws

deep flaws
the essence of UX design
below surface of the UI, in the semantics

(conjunctive)

concepts: a structure for functionality
purpose-driven & free standing

(de)constructing apps
non-conflicting concepts are “conjunctive”
organizing messages

Google

Gmail

COMPOSE

Inbox
Starred
Sent Mail
Drafts
Trash

Categories
Social
Promotions
Updates
Forums

hacking
meetups
todo
More

label:hacking

Alyssa P. Hacker
Inbox
javascript - Reminds you of the old da 9:14 pm

0 GB (0%) of 15 GB used
Manage

Terms - Privacy

Last account activity: 14 hours ago Details
automating filtering
slightly surprising behavior #1

Alyssa P. Hacker  Reminds you of the old days, eh?

Ben Bitdiddle  <benito.bitdiddle@gmail.com>
   to Alyssa
   Yes, it does.

Click here to Reply or Forward
slightly surprising behavior #2

<table>
<thead>
<tr>
<th>Primary</th>
<th>Social</th>
<th>Promotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ □ me, Alyssa (12) hacking meetups javascript - Hello again Ben 11:48 am</td>
<td></td>
<td></td>
</tr>
<tr>
<td>□ □ me, Alyssa (12) Inbox meetups javascript - Hello again Ben 9:43 am</td>
<td></td>
<td></td>
</tr>
<tr>
<td>□ □ me, Alyssa (12) Inbox hacking javascript - Hello again Ben. 9:58 am</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

label:hacking

label:meetups

label:hacking label:meetups

No messages matched your search. Try using search options such as sender, date, size and more.
slightly surprising behavior #3
slightly surprising behavior #4

Empty Trash now (messages that have been in Trash more than 30 days will be automatically deleted)

- me, Alyssa (13) 11:48 am
- Andy from Google 9:01 am

label:todo

There are no conversations with this label.

label:todo label:trash

- me, Alyssa 10:11 am

label:todo OR label:meetup

Some messages in Trash or Spam match your search. View messages.
slightly surprising behavior #5
question
when you grow a design by adding a concept,
how can you ensure its integrity is preserved?

related question
given a design comprising some concepts,
how can you tell if the concepts are mutually consistent?

a really simple answer
1. interpret composite behavior in terms of concept actions
2. check that each behavior satisfies each concept’s OP and spec

notes
really saying: consistent if you can find an interpretation s.t. ...
can attribute blame when one concept breaks another
slightly surprising behavior #1

violates Label’s operational principle:

not mark(m,p) ... search(p):ms ⇒ m not in ms

attribute blame to Conversation
slightly surprising behavior #2

violates nothing: expected behavior of Label and Conversation
slightly surprising behavior #4

violates Label’s operational principle: mark(m,p) ... search(p):ms ⇒ m in ms

attribute blame to Trash
slightly surprising behavior #5

violates Label’s spec

attribute blame to Category
<table>
<thead>
<tr>
<th></th>
<th>Label</th>
<th>Category</th>
<th>Conversation</th>
<th>Trash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
<td>-</td>
<td>breaks spec</td>
<td>breaks OP</td>
<td>breaks OP</td>
</tr>
<tr>
<td>Category</td>
<td>breaks spec</td>
<td>-</td>
<td>breaks spec</td>
<td></td>
</tr>
<tr>
<td>Conversation</td>
<td></td>
<td>-</td>
<td></td>
<td>breaks OP</td>
</tr>
<tr>
<td>Trash</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
</tbody>
</table>
addendum: on specs
what is a specification?

A specification
distinct from code
partial, abstract & declarative

A

a dependence
module A uses module C

B

C
what specifications gave us

local reasoning
using only code of A and specs of A, B, C

coupling
can change code of C without affecting A so long as specs fixed

components & reuse
any B that meets spec can be substituted

major caveat
modules can be coupled without any uses
some specs matter more

top level specification determines the impact of the software in the world
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 mattered more?
what we devoted ourselves to