rethinking software design by analyzing state

Daniel Jackson
Workshop Honoring Shmuel Katz · Technion · Dec 19, 2013
three puzzles
three puzzles

why are formal methods not widely used?
› great advances, successful application in specialized domains
› but still a niche, little impact on mainstream development
three puzzles

why are formal methods not widely used?
› great advances, successful application in specialized domains
› but still a niche, little impact on mainstream development

why is analysis often a second order effect?
› key rationale for formalization: mechanical analysis?
› but in many case studies, most errors found during formalization
three puzzles

why are formal methods not widely used?
› great advances, successful application in specialized domains
› but still a niche, little impact on mainstream development

why is analysis often a second order effect?
› key rationale for formalization: mechanical analysis?
› but in many case studies, most errors found during formalization

why is software so “reliable without proof”?
› better languages & more testing don’t explain it
› least usable features are the least reliable?
a hypothesis
a hypothesis

one underlying driver
› clarity of the underlying conceptual model
a hypothesis

one underlying driver
› clarity of the underlying conceptual model

bad concepts affect both
› user: can’t form mental model
› developer: can’t implement clean modules
a hypothesis

one underlying driver
› clarity of the underlying conceptual model

bad concepts affect both
› user: can’t form mental model
› developer: can’t implement clean modules

so:
› why don’t formal methods have more influence?
   with good conceptual model, informal reasoning goes far
› why does formalization alone find flaws so effectively?
   it forces you to clarify the concepts
› why do the least usable features have the most bugs?
   because the developers are confused too
interface
interface

user's model
research program

basic theory
- defining concepts
- concept dependence
- structural design criteria

conceptual redesigns
- git, gmail, dropbox, css

concept models
- concept idioms
- behavioral design criteria
WARNING

evolving research
as the thesis reader said: “There are new and good ideas here”
as the thesis reader said: “There are new and good ideas here”

“But what’s new isn’t good and what’s good isn’t new”
concept models
IT WILL DO YOU GOOD. Classify

Spread recycling!! To save limited natural resources for our children’s future.

© 1991 Produced by Super Planning Company Limited
classification syntax
atoms are
› distinguishable: have an identity
› immutable: don’t change
› indivisible: not structured

box
› set of atoms (empty, singleton, finite, infinite)
› italic: exhausted by subsets

fat arrow
› subset, not necessarily static
› shared arrow: disjoint subsets
classification syntax

atoms are
› distinguishable: have an identity
› immutable: don’t change
› indivisible: not structured

box
› set of atoms (empty, singleton, finite, infinite)
› italic: exhausted by subsets

fat arrow
› subset, not necessarily static
› shared arrow: disjoint subsets
**classification syntax**

**atoms are**
- **distinguishable**: have an identity
- **immutable**: don’t change
- **indivisible**: not structured

**box**
- **set** of atoms (empty, singleton, finite, infinite)
- italic: exhausted by subsets

**fat arrow**
- **subset**, not necessarily static
- shared arrow: disjoint subsets
kinds of relation
› property
› containment
› association
› naming
kinds of relation
› property
› containment
› association
› naming

relations syntax & semantics
kinds of relation
› property
› containment
› association
› naming

R maps \( m \) A’s to each B
R maps each A to \( n \) B’s

+ one or more
* zero or more
! exactly one
? at most one
omitted = *
example word styles
model word styles

Paragraph \(\rightarrow\) Style \(\rightarrow\) Rule

Style \(\rightarrow\) Property \& Value

Rule \(\rightarrow\) basedOn, next

prop \& value
instance word styles

P0 (Paragraph) style S0 (Style) basedOn S1 (Style)

S0 (Style) rules R0 (Rule)

R0 (Rule) prop P0 (Property) Face

P0 (Property) value V0 (Value) Caslon
semantics word styles
semantics word styles
semantics word styles
semantics word styles
semantics word styles
semantics word styles
semantics word styles
semantics word styles
semantics word styles
semantics word styles

P0 (Paragraph) → style → S0 (Style) 

S0 (Style) → basedOn → PO (Property) 

rules → prop → R0 (Rule) 

R0 (Rule) → prop → PO (Property) 

basedOn, next

Paragraph → style ! → Style 

rules ! → Rule 

prop ! → Property 

value ! → Value
semantics word styles
semantics word styles

P0 (Paragraph) \(\xrightarrow{\text{style}}\) S0 (Style) \(\xrightarrow{\text{basedOn}}\) P0 (Property)

R0 (Rule) \(\xrightarrow{\text{rules}}\) prop \(\xrightarrow{\text{basedOn, next}}\) ∈

Paragraph \(\xrightarrow{\text{style!}}\) Style

rules

Rule

prop !

Property

value !

Value
adding constraints word styles
adding constraints word styles
adding constraints word styles
adding constraints word styles

all s: Style is not in s.basedOn
adding constraints word styles

Paragraph → Style

style

Style → Rule

rules

Rule → Property

prop

Property → Value

value

Value
not just application state
all o: JSObject | o._proto = o._constructor.prototype
requests that are not embedded come from the client
all r: Request - Embedded | r.origin = r.from

embedded requests have the same origin as the response
all r: Response, e: r.embeds | e.origin = r.origin

request is only accepted if origin is server itself or sender
all s: Server, r: Request | r.to = s implies r.origin = r.to or r.origin = r.from
model degree rules

// plan must include one course from each group
all p: Plan, g: Group | some c: p.selects | c in g.courses

// plan cannot include conflicting courses
all p: Plan | no c1, c2: p.selects | c1 in c2.conflicts
original concept model for Word styles
**abstracted basic concept idiom**
variant idiom with basedOn
variant idiom with stylesheet

**style idiom**

- **Element**
  - **Style**
    - **Rule**
      - **Property**
      - **Value**
    - **basedOn**
  - **styles**
  - **rules**
  - **style**!
There is no problem in computer science that cannot be solved by introducing another level of indirection.
--David Wheeler
style other instantiations

Powerpoint schemes

Indesign swatches
Apple color picker

value relation must be `mutable`
idiom selection

slides in Keynote

photos in Adobe Lightroom

messages in Apple Mail
Document → Selection

Selection → Element

Document → Element

Selection → selected

elements
some variants
one or more selections per document? selected elements and active element? selection is 0/1 or 0..1?
some variants
one or more selections per document?
selected elements and active element?
selection is 0/1 or 0..1?
some variants
one or more selections per document?
selected elements and active element?
selection is 0/1 or 0..1?
can select groups too
Element shows Tag includes Filter
tags includes
some variants
filter has disjuncts/conjuncts
tags are key/value pairs
some tags are system tags
some tags inhibit display
some variants
filter has disjuncts/conjuncts
tags are key/value pairs
some tags are system tags
some tags inhibit display
Idiom tagging

Some variants:
- Filter has disjuncts/conjuncts
- Tags are key/value pairs
- Some tags are system tags
- Some tags inhibit display

Examples:
- Labels in Gmail
- Keywords in Lightroom
- File properties in OS X
idiom invariants
every style has a rule for every property

\[
\text{all } s: \text{Style}, p: \text{Property} \mid \text{some } r: s.\text{rules} \mid r.\text{prop} = p
\]
why it matters
› if a style must include all properties then:
› a style can’t inherit a rule from its parent

but unfortunately
› many designs don’t consider implications fully...
can you inherit a property?
can you inherit a property?

Word: property absent until entered; then remove only in Visual Basic!
can you inherit a property?

- **Word:** property absent until entered; then remove only in Visual Basic!
- **Indesign:** property absent until entered; then remove only with Reset (since 2007)
can you inherit a property?

Indesign: property absent until entered; then remove only with Reset (since 2007)

Word: property absent until entered; then remove only in Visual Basic!

Pages: aaah! properties are optional
selecting a group selects its elements too

all s: Selection, o: s.selected & Group | o.contents in s.selected
invariant variants selection
invariant variants selection

why it matters
invariant variants selection

why it matters

- if groups and their members can be selected separately, the design is more flexible for the user
why it matters
  › if groups and their members can be selected separately, the design is more flexible for the user

variants
why it matters
› if groups and their members can be selected separately, the design is more flexible for the user

variants
› drawing apps: until recently, grouping prevented separate selection now many apps allow elements of groups to be selected alone
**invariant variants**

**selection**

**why it matters**
- if groups and their members can be selected separately, the design is more flexible for the user

**variants**
- drawing apps: until recently, grouping prevented separate selection. Now many apps allow elements of groups to be selected alone.
- Apple Mail: selecting an element of a group and an element outside the group causes all elements of the group to be selected.
why it matters
› if groups and their members can be selected separately, the design is more flexible for the user

variants
› drawing apps: until recently, grouping prevented separate selection now many apps allow elements of groups to be selected alone
› Apple Mail: selecting an element of a group and an element outside the group causes all elements of the group to be selected
› git: eliminates notion of group by not syncing directories
invariant variants selection

why it matters
› if groups and their members can be selected separately, the design is more flexible for the user

variants
› drawing apps: until recently, grouping prevented separate selection now many apps allow elements of groups to be selected alone
› Apple Mail: selecting an element of a group and an element outside the group causes all elements of the group to be selected
› git: eliminates notion of group by not syncing directories
› CrashPlan: selection of directory has different meaning; sets default for files that will be added later
a filter shows elements with its included tags

```
all f: Filter | f.shows = f.includes.~tags
```
why it matters
› users get very confused if things they expect to be there are not variants
› Lightroom: deleted images are never shown
› Apple Finder: “include trash” separated out (but will create a smart folder that shows files marked as invisible!)
why it matters

› users get very confused if things they expect to be there are not

variants

› Lightroom: deleted images are never shown
› Apple Finder: “include trash” separated out
  (but will create a smart folder that shows files marked as invisible!)
A deleted message matches your search. View it or go to Trash to delete forever.

generally won’t show trashed messages
generally won’t show trashed messages

if you ask for them explicitly, you’ll see some
generally won’t show trashed messages

if you ask for them explicitly, you’ll see some

hmm...
analyzing concepts
refactoring concept models
refactoring concept models

suppose we have a bad concept model
› can we refactor into a better one?
› and show that the two are somehow equivalent?
refactoring concept models

suppose we have a bad concept model
› can we refactor into a better one?
› and show that the two are somehow equivalent?

an example from the “Area 2 web app”
› application that tracks degree requirements for MIT CS students
Circle four subject numbers in the table below. Of the 4 subjects, two subjects should be selected from a single group. The remaining two subjects must be selected from two other groups. If you have already received a grade in the subject, please enter the grade in the box. Please enter the term that you completed the subject or plan to take the subject as well (e.g. FT12 is the term starting September 2012 and ST13 is the term starting February 2013). Prior to Drop Date of the Spring term 2013, changes in your choices may be made by submitting a new version of this form; after that date, a petition to the Committee on Graduate Students is required.

<table>
<thead>
<tr>
<th>Group 1: Systems in CS</th>
<th>Group 2: Theoretical CS</th>
<th>Group 3: Artificial Intelligence</th>
</tr>
</thead>
</table>

|-----------------------------------------------|----------------------------------------|-----------------------------------------------|

<table>
<thead>
<tr>
<th>Group 7: Bioelectrical Engineering</th>
<th>Group 8: Electromagnetics</th>
<th>Group 9: Physical Science and Engineering</th>
</tr>
</thead>
</table>

Note: Students in Area II Computer Science select subjects from Group 1, 2, 3 only (shaded boxes)
- 6.840 or 6.854 are recommended for students who plan to take only one subject in Group 2.
- 6.839 can be used as the second AI subject, but not the only subject.
- 6.831 can be the second subject in Group 1 or 3, but not the only subject in either group.
Circle four subject numbers in the table below. Of the 4 subjects, two subjects should be selected from a single group. The remaining two subjects must be selected from two other groups. If you have already received a grade in the subject, please enter the grade in the box. Please enter the term that you completed the subject or plan to take the subject as well (e.g. FT12 is the term starting September 2012 and ST13 is the term starting February 2013). Prior to Drop Date of the Spring term 2013, changes in your choices may be made by submitting a new version of this form; after that date, a petition to the Committee on Graduate Students is required.

<table>
<thead>
<tr>
<th>Group 1: Systems in CS</th>
<th>Group 2: Theoretical CS</th>
<th>Group 3: Artificial Intelligence</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Any 1 or 2 subject allowed)</td>
<td>(Any 1 or 2 subject allowed)</td>
<td></td>
</tr>
<tr>
<td>(Any 1 or 2 subject allowed)</td>
<td>(Any 1 or 2 subject allowed)</td>
<td></td>
</tr>
<tr>
<td>Group 7: Bioelectrical Engineering</td>
<td>Group 8: Electromagnetics</td>
<td>Group 9: Physical Science and Engineering</td>
</tr>
<tr>
<td>(Any 1 or 2 subject allowed)</td>
<td>[6.641 xor 6.561 xor 6.685]</td>
<td>(Any 1 or 2 subject allowed)</td>
</tr>
</tbody>
</table>

Note: Students in Area II Computer Science select subjects from Group 1, 2, 3 only (shaded boxes)
- 6.840 or 6.854 are recommended for students who plan to take only one subject in Group 2.
- 6.839 can be used as the second AI subject, but not the only subject.
- 6.831 can be the second subject in Group 1 or 3, but not the only subject in either group.
Circle four subject numbers in the table below. Of the 4 subjects, two subjects should be selected from a single group. The remaining two subjects must be selected from two other groups. If you have already received a grade in the subject, please enter the grade in the box. Please enter the term that you completed the subject or plan to take the subject as well (e.g., FT12 is the term starting September 2012 and ST13 is the term starting February 2013). Prior to Drop Date of the Spring term 2013, changes in your choices may be made by submitting a new version of this form; after that date, a petition to the Committee on Graduate Students is required.

<table>
<thead>
<tr>
<th>Group 1: Systems in CS</th>
<th>Group 2: Theoretical CS</th>
<th>Group 3: Artificial Intelligence</th>
</tr>
</thead>
</table>

|-----------------------------------------------|----------------------------------------|---------------------------------|

<table>
<thead>
<tr>
<th>Group 7: Bioelectrical Engineering</th>
<th>Group 8: Electromagnetics</th>
<th>Group 9: Physical Science and Engineering</th>
</tr>
</thead>
</table>

Note: Students in Area II Computer Science select subjects from Group 1, 2, 3 only (shaded boxes)
- 6.840 or 6.854 are recommended for students who plan to take only one subject in Group 2.
- 6.839 can be used as the second AI subject, but not the only subject.
- 6.831 can be the second subject in Group 1 or 3, but not the only subject in either group.
Circle four subject numbers in the table below. Of the 4 subjects, two subjects should be selected from a single group. The remaining two subjects must be selected from two other groups. If you have already received a grade in the subject, please enter the grade in the box. Please enter the term that you completed the subject or plan to take the subject as well (e.g. FT12 is the term starting September 2012 and ST13 is the term starting February 2013). Prior to Drop Date of the Spring term 2013, changes in your choices may be made by submitting a new version of this form; after that date, a petition to the Committee on Graduate Students is required.

<table>
<thead>
<tr>
<th>Group 1: Systems in CS</th>
<th>Group 2: Theoretical CS</th>
<th>Group 3: Artificial Intelligence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 7: Bioelectrical Engineering</td>
<td>Group 8: Electromagnetics</td>
<td>Group 9: Physical Science and Engineering</td>
</tr>
</tbody>
</table>

Note: Students in Area II Computer Science select subjects from Group 1, 2, 3 only (shaded boxes)
- 6.840 or 6.854 are recommended for students who plan to take only one subject in Group 2.
- 6.839 can be used as the second AI subject, but not the only subject.
- 6.831 can be the second subject in Group 1 or 3, but not the only subject in either group.
implied conceptual model

- **Option**
  - options
  - conflicts
  - NotOnly Option
  - course
  - a. select 2 options from one group
  - b. select one option from other groups
  - c. NotOnly option is not only option in group
  - d. options may not conflict
Select one subject from each of the four groups. Note that the following subjects conflict; you may take at most one from each set:

- 6.345, 6.863, and 6.864
- 6.437, 6.438, and 6.867
- 6.831 and 6.839
- 6.840 and 6.841
- 6.866 and 6.869
- 6.874 and 6.878
simplified conceptual model

a. select one more course than groups
b. select at least one course per group
c. courses may not conflict
alloy model
alloy model

forward: check {
  all p: TQE_Plan | valid[p] implies simpler_valid[p]
} for 4 but 1 TQE_Plan
forward: check {
  all p: TQE_Plan | valid[p] implies simpler_valid[p]
} for 4 but 1 TQE_Plan

backward: check {
  all p: TQE_Plan | simpler_valid[p] implies valid[p]
} for 4 but 1 TQE_Plan
counterexample: new too strong
counterexample: new too strong
counterexample: new too strong

TQE_Plan
- courses: Course0, Course1, Course2
- map: Course0→Option0, Course1→Option2, Course2→Option3
counterexample: new too strong

TQE_Plan
courses: Course0, Course1, Course2
map: Course0→Option0, Course1→Option2, Course2→Option3

plan rejected by new rules but accepted by old ones because courses 0 and 2 only conflict for some options
counterexample: new too weak
counterexample: new too weak
counterexample: new too weak

TQE_Plan
courses: Course0, Course1, Course2
map: Course0->Option3, Course1->Option2, Course2->Option1
counterexample: new too weak

plan rejected by old rules but accepted by new ones because option was chosen for course 1 that leaves a ‘not only’ course in group 1
when is simplification valid?
when is simplification valid?

P1. When two options conflict, any other pair of options that corresponds to the same two courses also conflicts.
P1. When two options conflict, any other pair of options that corresponds to the same two courses also conflicts.

P2. If two options (in different groups) are for the same course, then those options are “not only” options
conclusions
conclusions

simple invariants expose subtle problems
use idioms to explore standard solutions
conclusions

simple invariants expose subtle problems
use idioms to explore standard solutions

formal methods might help
cost amortized when applied to idiom
conclusions

simple invariants expose subtle problems
use idioms to explore standard solutions

formal methods might help
cost amortized when applied to idiom

conceptual modeling: old idea with new challenges

Analysis Patterns (Fowler, 1997)
Data Model Patterns (Hay, 2011)
Conceptual Models (Henderson & Johnson, 2011)
conclusions

simple invariants expose subtle problems
use idioms to explore standard solutions

formal methods might help
cost amortized when applied to idiom

case conceptual modeling: old idea with new challenges

*Analysis Patterns* (Fowler, 1997)
*Data Model Patterns* (Hay, 2011)
*Conceptual Models* (Henderson & Johnson, 2011)