Software with the Quality that Has No Name

Federico Mena Quintero
federico@gnome.org

Desktop Summit, Berlin, Aug/2011
Our house in the middle of our street

FIXME: before/after pictures
A Pattern Language
Towns · Buildings · Construction

Christopher Alexander
Sara Ishikawa · Murray Silverstein
WITH
Max Jacobson · Ingrid Fiksdahl-King
Shlomo Angel

1977
Intimacy gradient

House

Office
Light on two sides of every room

Wrinkle the building's edge
Indoor sunlight

- Porch
- Workshop
- Family rooms
- Storage
- Kitchen
- Breakfast nook
- Bedrooms
- Garage
- Garden

Evening sun: Garden
Morning sun: South-facing rooms
Pattern name

- Super-patterns
- Statement of problem
- Discussion
- Summary of the solution
- Sub-patterns
Light on two sides of every room

- **Super-patterns:** wings of light, positive outdoor space, cascade of roofs
- **Statement of problem:** People gravitate to well-lit rooms.
- **Discussion:**
- **Summary of the solution:** Light on two sides; natural light through the windows
- **Sub-patterns:** Roof layout, windows overlooking life, window place, filtered light
Light on two sides of every room
Intimacy gradient
Alcoves
Window place
Filtered light
Positive outdoor space
Indoor sunlight
Sleeping to the East
Common areas at the heart
A room of one's own
Half-private office
Cascade of roofs
Reception welcomes you
A room of one's own
Light on two sides of every room
Alcoves
Patterns do not give you a final form
Patterns give you a vocabulary

Architecture
Alcove
Positive space
Cascade of roofs

Programming
Factory
Strategy
Listener
Pattern: Zooming

1.05^0
1.05^1
1.05^2
1.05^3
1.05^4

Zoom: 50%

Zoom: %

Zoom:

25%
50%
75%
100%
150%
200%

Red X

Blue Arrows

Green Check
The ticket booth
The Quality Without A Name
```
partition (array, left, right, pivot)
{
    pivot_value = array[pivot];
    swap (array, pivot, right);
    store = left;
    for (i = left; i < right; i++) {
        if (array[i] < pivot_value) {
            swap (array, i, store);
            store++;
        }
    }
    swap (array, store, right);
    return store;
}

quicksort (array, left, right)
{
    if (left < right) {
        pivot = (left + right) / 2;
        new_pivot = partition (array, left, right, pivot);
        quicksort (array, left, new_pivot - 1);
        quicksort (array, new_pivot + 1, right);
    }
}  
```
The Quality for Software

(According to Richard Gabriel)

- It was not written to unrealistic deadline
- Any bad parts were repaired during the maintenance or are being repaired now
- If it is small, it was written by an extraordinary person, someone I would like as a friend; if it is large, it was not designed by one person, but over time in a slow, careful, incremental way
- If I look at any small part of it, I can see what is going on
- If I look at any large part in overview, I can see what is going on
- It is like a fractal, in which every level of details is as locally coherent and as well thought as any other level
- Every part of the code is transparently clear - there are no sections that are obscure in order to gain efficiency
- Everything about it seems to be familiar
- I can imagine changing it, adding some functionality
- I am not afraid of it, I will remember it
# 15 Properties of Living Structure

<table>
<thead>
<tr>
<th>Levels of scale</th>
<th>Strong centers</th>
<th>Thick boundaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternating repetition</td>
<td>Positive space</td>
<td>Good shape</td>
</tr>
<tr>
<td>Local symmetries</td>
<td>Deep interlock and ambiguity</td>
<td>Contrast</td>
</tr>
<tr>
<td>Gradients</td>
<td>Roughness</td>
<td>Echoes</td>
</tr>
<tr>
<td>The void</td>
<td>Simplicity and inner calm</td>
<td>Non-separateness</td>
</tr>
</tbody>
</table>
The void
The void

Good shape
The void

Good shape

Echoes
The void
Good shape
Echoes
Positive space
Local symmetries
The void
Good shape
Echoes
Positive space
Local symmetries
Strong centers
Roughness
The void
Good shape
Echoes
Positive space
Local symmetries
Strong centers
Roughness
Alternating repetition
Negative space – amorphous leftovers
Weak centers
GIMP toolbar

Empty image

Negative space – amorphous leftovers
Weak centers

Layers dialog
Positive space
(convex, enclosed)

Boundary

Strong centers
Design as computation
Stepwise: one step at a time
Stepwise: one step at a time

Reversible: test using models, prototypes, trial and error
Stepwise: one step at a time

Reversible: test using models, prototypes, trial and error

Structure-preserving: each step builds on what is already there
Stepwise: one step at a time

Reversible: test using models, prototypes, trial and error

Structure-preserving: each step builds on what is already there

Design from weakness: each step improves coherence
Stepwise: one step at a time

Reversible: test using models, prototypes, trial and error

Structure-preserving: each step builds on what is already there

Design from weakness: each step improves coherence

New from existing: emergent structure combines what is already there
Structure-preserving transformations
A class

weak, latent center

PhoneCall
Pattern: Half-object + Protocol

Local symmetry, strong center, levels of scale
What joins to what?

Local symmetry, levels of scale, boundaries, deep interlock and ambiguity
Explicit boundary

Local symmetry, deep interlock, and this is composable
Composable elements

Multi-way calls, conference calls

Diagram: Call connected to two HalfCall elements.
Form languages
Form language (Japan/China)
Form language (Germany)
Pattern language
Pattern language

Form language
Pattern language

Form language
Behavior-preserving transformations
Move common code to function
Strong center
Boundary
Move common code to function
Strong center
Boundary

Add parameter to a function
Roughness
Non-separateness
Move common code to function
  Strong center
  Boundary

Add parameter to a function
  Roughness
  Non-separateness

Replace parameter with explicit methods
  Strong centers
  Simplicity
  Non-separateness
  Deep ambiguity and interlock
Delete a bunch of code
The void
Simplicity and inner calm
Credits

- Rob Hopkins – Picture of Christopher Alexander
- Amazon.com – Book covers
- Oxford University Press – diagrams from “A Pattern Language”
- Other pictures – Flickr Creative Commons
- Japan – Lennart Poettering
- Process diagrams – Nikos Salingaros