Software Architecture

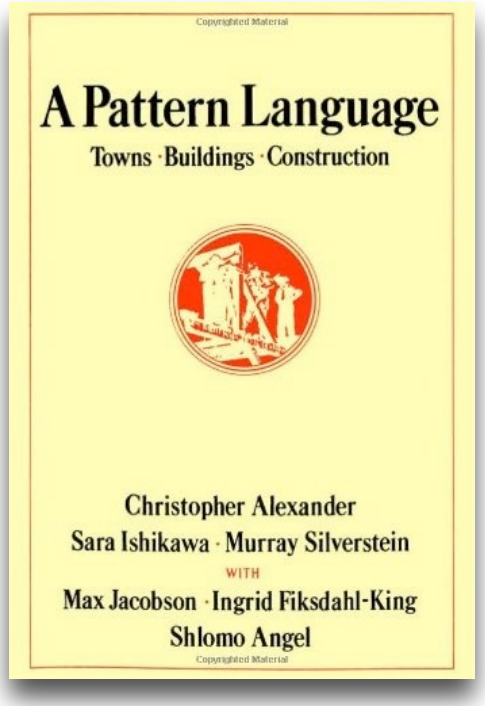
FEUP-M.EIC-ASSO-2024

Ademar Aguiar, Neil Harrison

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Patterns of Software

The Origins of Patterns, 1977



- Christopher Alexander in this book presents a pattern language, an ordered collection of 253 patterns.
- The goal was to enable non-experts to architect and design their own houses and communities with quality.
- Patterns describe current practice and provide vision for the future.
- Patterns integrate knowledge from diverse sources and link theory and practice.
- Patterns have a strong “bottom up” orientation.

http://c2.com/cgi/wiki?ChristopherAlexander

What is a pattern?

> The elements of this language are entities called patterns. Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.

from *A Pattern Language*, Alexander et al, 1977

253 patterns
from global to particular problems

- 1. Independent Regions
- 2. The Distribution of Towns
- 16. Web of Public Transportation
- 18. Network of Learning
- 20. Mini-buses
- 43. University as a Marketplace
- 83. Master and Apprentices
- 134. Zen View
- 253. Things from your life

title

context

problem

discussion

20 MINI-BUSES*

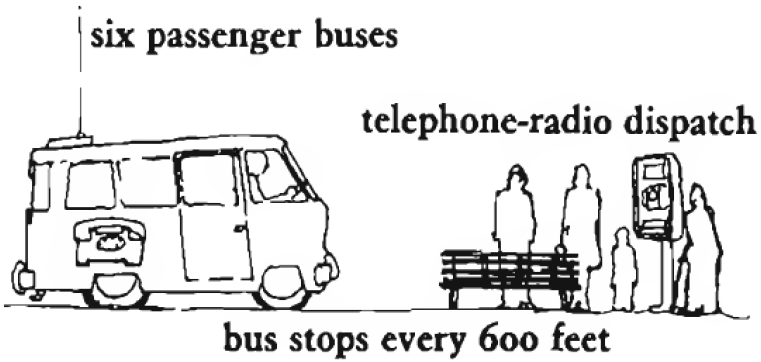
. . . this pattern helps complete the LOCAL TRANSPORT AREAS (11) and the WEB OF PUBLIC TRANSPORTATION (16). The local transport areas rely heavily on foot traffic, and on bikes and carts and horses. The web of public transportation relies on trains and planes and buses. Both of these patterns need a more flexible kind of public transportation to support them.

Public transportation must be able to take people from any point to any other point within the metropolitan area.

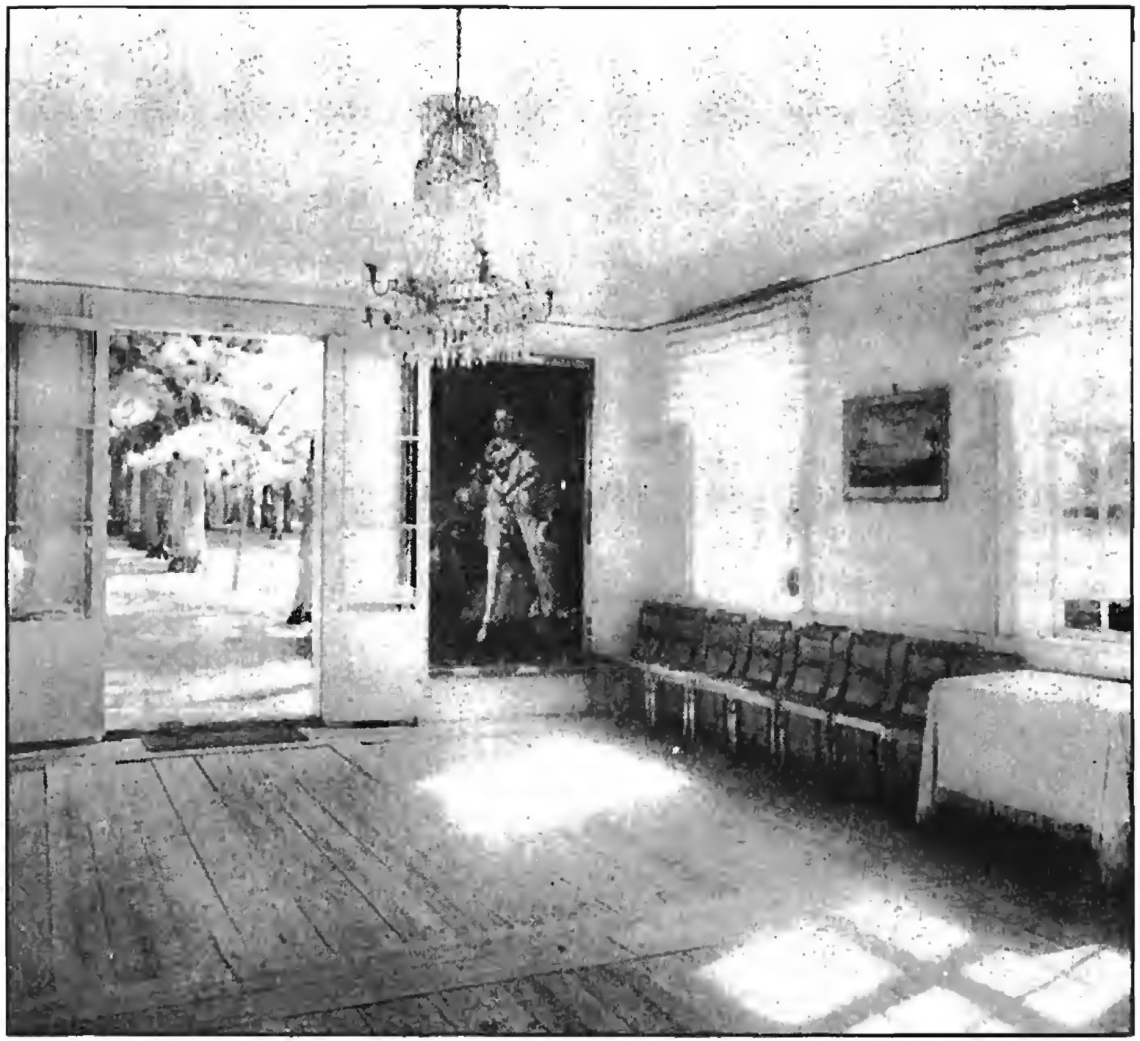
Buses and trains, which run along lines, are too far from most origins and destinations to be useful. Taxis, which can go from point to point, are too expensive.

solution

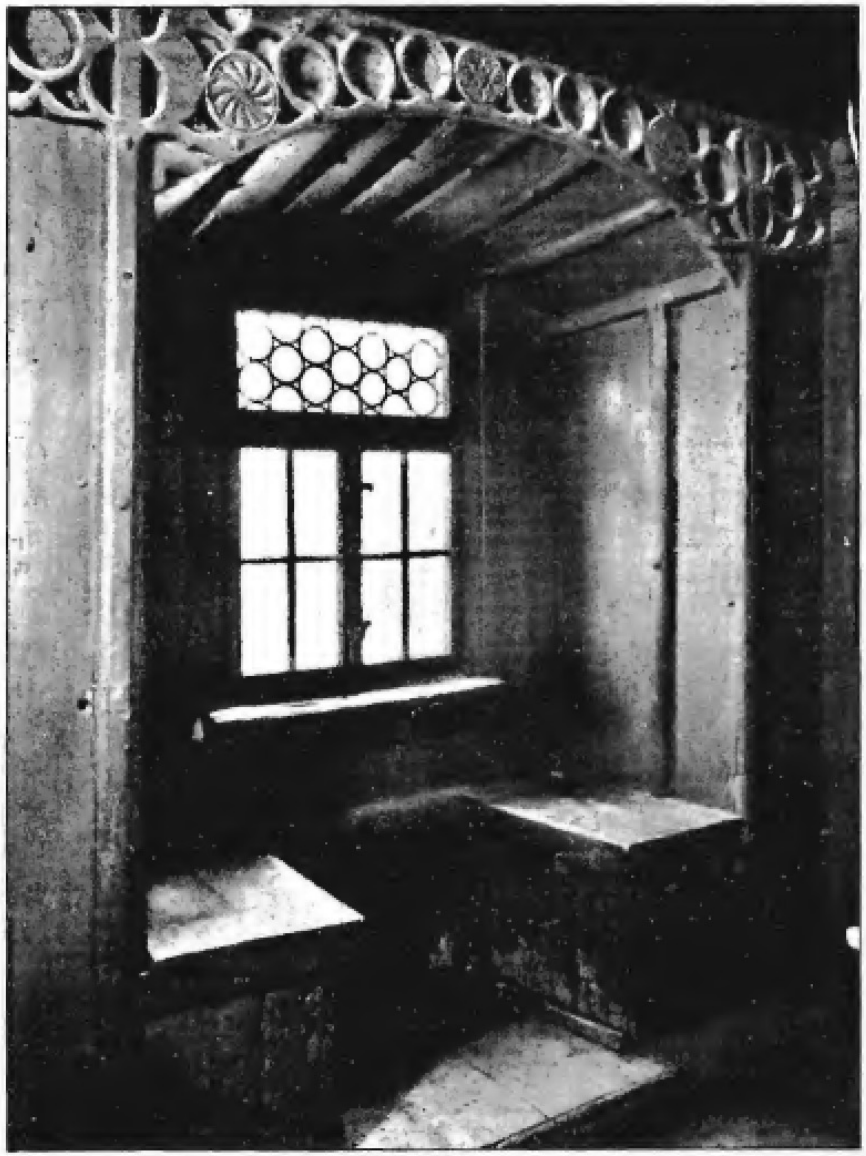
Establish a system of small taxi-like buses, carrying up to six people each, radio-controlled, on call by telephone, able to provide point-to-point service according to the passengers' needs, and supplemented by a computer system which guarantees minimum detours, and minimum waiting times. Make bus stops for the mini-buses every 600 feet in each direction, and equip these bus stops with a phone for dialing a bus.



159 LIGHT ON TWO SIDES OF EVERY ROOM**



180 WINDOW PLACE**



251 DIFFERENT CHAIRS

253 THINGS FROM YOUR LIFE*

QWAN

Quality Without A Name

Fifteen Properties

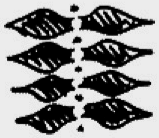
The degree of life which appears in a thing depends upon the life of its component centers and their density. Following are fifteen structural features which he has identified as appearing again and again in things which have life.
Together, these fifteen properties identify the character of living systems. They are the principal ways in which centers can be strengthened by other centers. They are not independent, but rather rely on and reinforce each other. Things which are more whole, which exhibit more life, will have these fifteen properties to a strong degree. Conversely, the things in this world which are most lifeless will have these properties to the least degree.

1. Level of Scale

2. Strong Centers

3. Boundaries

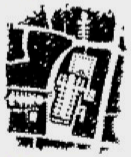
4. Alternating Repetition



5. Positive Space

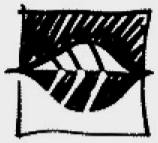
6. Good Shape

7. Local Symmetries



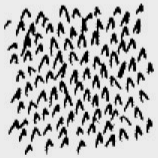
8. Deep Interlock

9. Contrast



10. Graded Variation

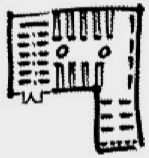
11. Roughness



12. Echoes

13. The Void

14. Inner Calm

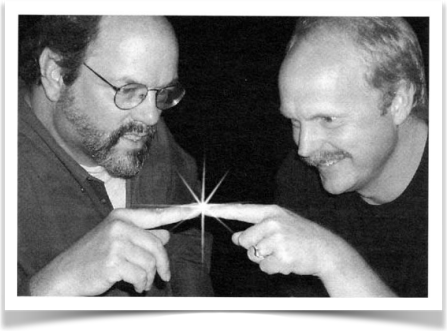


15. Not-Separateness

Quality attributes in Software Architecture

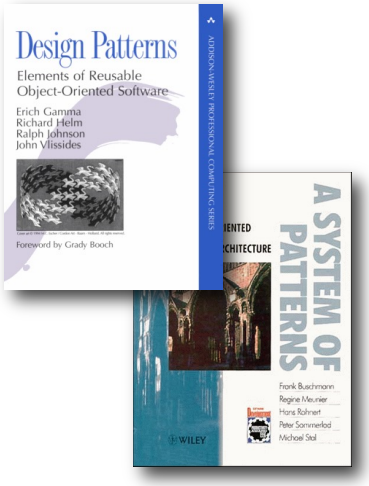
https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

Software Patterns, 1987..1994..



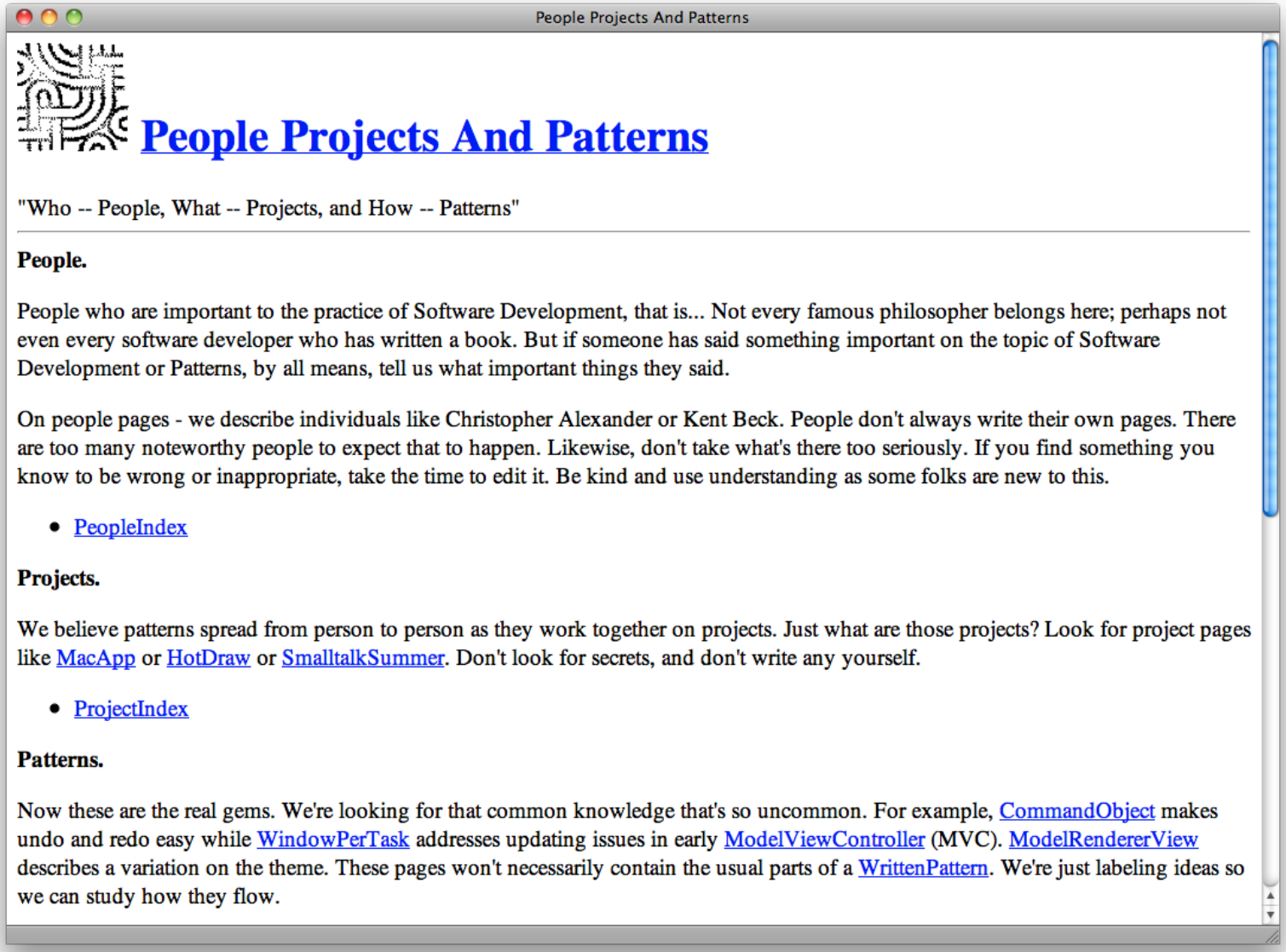
A new software patterns discipline started by a group later called “The Hillside Group”, 1993.

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (aka Gang of Four), published the first book on “Design Pattens”, the “bible”, 1994.

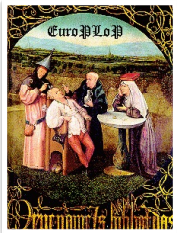


- Kent Beck & Ward Cunningham, “Using Pattern Languages for Object-Oriented Program”, OOPSLA '87, 1987.

http://wiki.c2.com/?HistoryOfPatterns



PLoP Conferences



www.hillside.net/conferences

Pattern Languages of Programs (PLoP) Conferences

Friday, Jul 07, 2017

Login Text Size

THE HILLSIDE GROUP

Site Search

SEARCH

Hilltop Books Contact Conferences Patterns Vision Wiki

MAIN MENU

- Hilltop
- Books
- Contact
- Conferences
  - GuruPLoP
  - PLoP
  - Chili PLoP
  - EuroPLoP
  - Meta PLoP
  - Asian PLoP
  - ParaPLoP
  - Scrum PLoP
  - Sugarloaf PLoP
  - Viking PLoP
  - Other Conferences
  - Shepherding
  - Useful Documents
  - PLoP Conference Proceedings
  - PLoP Paper Template
- Patterns
- Vision

You are here: Home Conferences

PATTERN LANGUAGES OF PROGRAMS (PLOP) CONFERENCES

We have compiled our collected PLoP experiences into a series named How to Run PLoP

The Hillside Group Sponsors many different conferences such as: PLoP, EuroPLoP, AsianPLoP, ScrumPLoP, VikingPLoP, SugarLoafPLoP, UP, and ChiliPLoP. These conferences focus on writing groups to better improve patterns through group exposure. Each conference offers advanced topics for the more adept pattern writers. Participants have the opportunity to refine and extend their patterns with help from knowledgeable and sympathetic patterns enthusiasts.

Conference Proceedings

PLOP

Pattern Languages of Programs (PLoP™) conference is a premier event for pattern authors and pattern enthusiasts to gather, discuss and learn more about patterns and software development... Learn More

CHILIPLOP

The 13th Annual ChiliPLoP features "hot topics" for experienced folks... Learn More

SUGARLOAF PLOP

SugarLoafPLoP brings together researchers and practitioners whose interests span a remarkably broad range of topics, who share an interest in exploring the power of the pattern form... Learn More

EURO PLOP

Patterns do and do not...

Patterns do...

- provide common vocabulary
- provide “shorthand” for effectively communicating complex principles
- help document software architecture
- capture essential parts of a design in compact form
- show more than one solution
- describe software abstractions

Patterns do not...

- provide an exact solution
- solve all design problems
- only apply for object-oriented design

Patterns can be...

- non-generative (*e.g.* Gamma patterns)
  - observed in a system
  - descriptive and passive
- generative (*e.g.* Alexander patterns)
  - generate systems or parts of systems
  - perspective and active

Thank you!