Center for LifeLong Learning & Design

University of Colorado at Boulder

Wisdom is not the product of schooling
but the lifelong attempt to acquire it.
- Albert Einstein

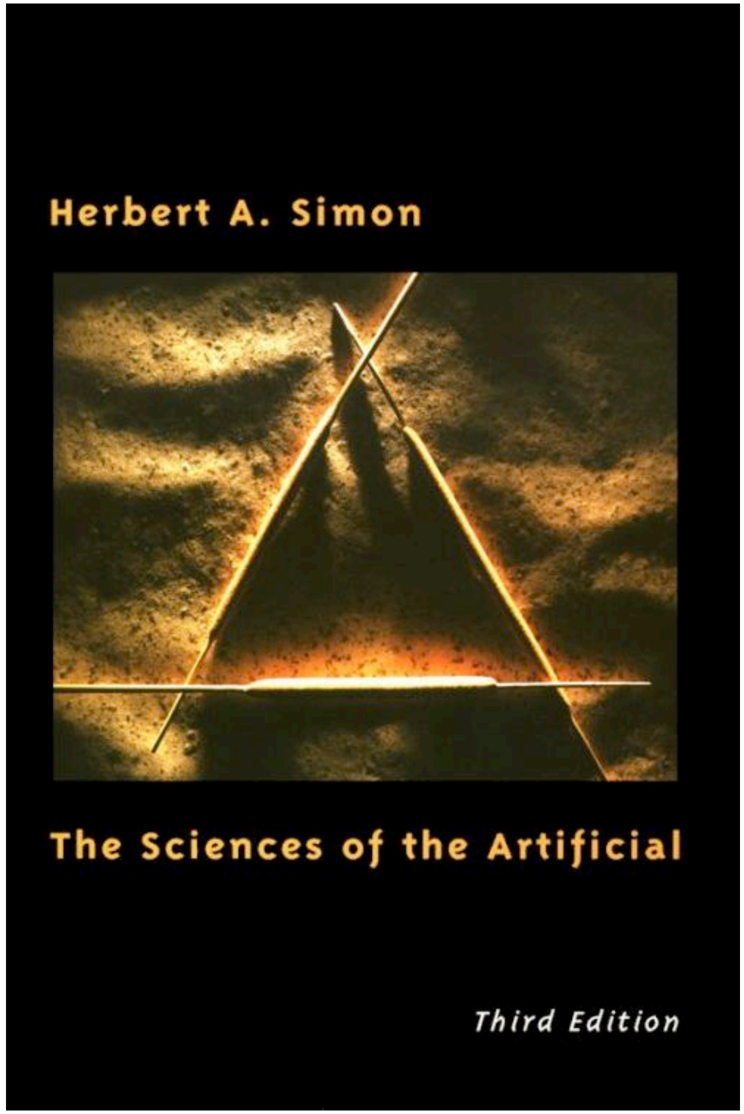
Design = The Sciences of the Artificial and The Architecture of Complexity

Gerhard Fischer and Hal Eden

Spring Semester 2007

January 31, 2007

The Sciences of the Artificial



The Sciences of the Artificial

- **"The Sciences of the Artificial" — a book by Herbert Simon**
  - 1st edition: 1969
  - 2nd edition: 1981
  - 3rd edition: 1996

- **who was Herbert Simon (1916-2001):**
  - a founder of Artificial Intelligence
  - a Nobel Prize Winner in Economics (1978) →
    - "for his pioneering research into the decision-making process within economic organizations"
    - his interest in simplifying and understanding complex decision-making situations led him at an early stage to the problem of breaking down complex equation systems
  - a major figure in Psychology and Cognitive Science
  - the "intellectual founder" of the current NSF program entitled "Science of Design"

Boundaries for the Sciences of the Artificial

- artificial things are **synthesized** (though not always or usually with full forethought) by humans
- artificial things may **imitate appearances** in natural things while lacking, in one or more respects, the reality of the latter
- artificial things can be characterized in terms of **functions, goals, adaptation, and flexibility**
- artificial things are often discussed, particularly when they are designed, in terms of **imperatives** as well as **descriptives**

The Unifying Themes in Simon's Work

- **bounded rationality** = there are limits on a human as a decision maker and a problem solver (especially limits in cognitive processing) → these limits are important for the behavior of humans

- **satisfycing** = accepting solutions which are "good enough"
  - "The best is often the enemy of the good"
  - the concept of "**satisfycing**" separated Artificial Intelligence from Operations Research

Some Famous Quotes from Simon

- **definition of design**: “Everyone designs who devise courses of action aimed at changing existing situations into preferred ones. The intellectual activity that produces material artifacts is no different fundamentally from the one that prescribes remedies for a sick patient or the one that devises a new sales plan for a company or a social welfare policy for a state”
  importance: a science of design

- “What information consumes is rather obvious: it consumes the attention of its recipients. Hence a wealth of information creates a **poverty of attention**, and a need to allocate efficiently among the overabundance of information sources that might consume it.”
  importance: contextualizing information (beyond current “push” technologies, making information relevant to the task at hand, critiquing)

- “The smartest people in the world do not generally look very intelligent when you give them a problem that is **outside** the domain of their vast experience.”
  importance: domain-orientation, human problem-domain interaction

Some Systems Developed by Simon

- **systems**
  - **Logic Theory Machine (LT)** → finding proofs of theorems in elementary symbolic logic
  - **General Problem Solver (GPS)** → a program that simulates human thought (objects, operators, differences)
  - **Bacon** → scientific discovery: to induce laws from data

- **prediction:** he predicted around 1965 that within 10 years there will be a computer program that will be the **best chess player in the world**

- **for more information:** Feigenbaum, E. A. & Feldman, J. (1963) Computers and Thought, McGraw-Hill Book Company, New York.

Some Concepts Developed by Simon

- **informational efficiency:** two representations are informationally equivalent if all of the information in the one is also inferable from the other, and vice versa. Each could be constructed from the information in the other.

- **computational efficiency:** two representations are computationally equivalent if they are informationally equivalent and, in addition, any inference that can be drawn easily and quickly from the information given explicitly in the one can also be drawn easily and quickly from the information given explicitly in the other, and vice versa

- ***informational equivalence* versus *computational equivalence* of representations** → *"even if two representations contain exactly the same information, it may be far cheaper, computationally, to make some of this information explicit using one representation than using the other"*

- **ill-defined problems**

The Importance of Representations

- **number scrabble:**
  - two person game
  - numbers from 1 to 9
  - players alternate and take one of the numbers
  - the player who can add ***exactly*** three numbers in her/his possession to equal 15 will win

- **critical importance of representations in design:**
  “Solving a problem simply means representing it so as to make the solution transparent”

Well-Defined versus Ill-Defined Problems

- **Well-Defined Problems:**
  - the essential conditions of the problem are stated
  - their solutions are the same for all problem solvers
  - examples: school problems, mutilated checker board, implementing given algorithms

- **Ill-Defined (or Wicked) Problems:** problem solvers take an active role what the problem is
  - fill gaps in the problem definition
  - jump into the problem
  - use information gained while trying to solve the problem
  - examples: architects, engineers, lawyers, legislators, software designers, writers, teachers, ...

Design Deals with Wicked or Ill-Defined Problems

Horst Rittel in Cross "Developments in Design Methodology"

- there is **no definitive formulation** of a wicked problem. For any given tame problem, an exhaustive formulation can be stated containing all the information the problem-solver needs for understanding and solving the problem.
- they have **no stopping rule**. In tame problems, problem solvers know when they have done the job. Problem solvers terminate work on a wicked problem, not for reasons inherent in the 'logic' of the problem.
- solutions to wicked problems are not **"true-or-false", but "good-or-bad"**
- every wicked problem is **essentially unique** ***("universe-of-one")***
- the aim of design is not to find the truth, but to **improve** some characteristics of the world where people live

Examples for Large-Scale Design

- **going to the moon** → a "complex" problem along one dimension; sources for success:
  - exceedingly cooperative environment
  - employing a single new organization
  - single, highly operational goal

- **the American Constitution:**
  "the founding fathers did not postulate a new man to be produced by new institutions but accepted as one of their design constraints the psychological characteristics of men and women as they knew them, their selfishness as well as their common sense"

- **"designed" cities:** Brasilia, Abudja, Canberra, …. (versus: evolving cities)

- in many large-scale designs → we need not so much a "correct" conceptualizatio as one that **could be understood by everyone**

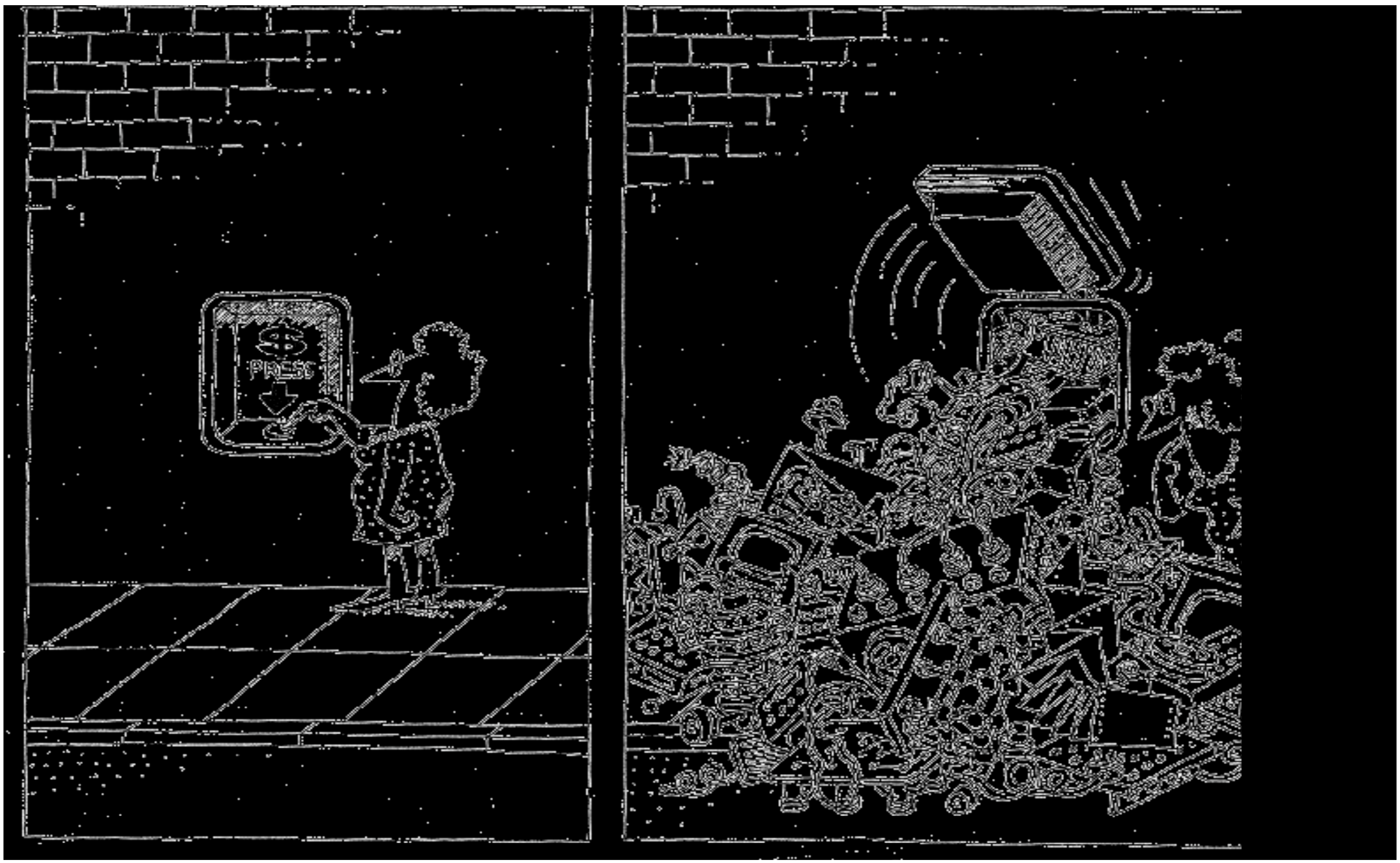
The Shape of the Design
—
Hierarchy and the Problem of Modularity

- **claims**:
  - to design a complex structure, one powerful technique is to discover viable ways of decomposing it into semi-independent components corresponding to its many functional parts
  - the design of each component can then be carried out with some degree of independence of the design of others, since each will affect the others largely through its function and independently of the details of the mechanisms that accomplish the function.

- **different approaches towards modularity in programming**:
  - nearly decomposable systems (objective of any program structured with subroutines and functions)
  - functional programming → LISP
  - object-oriented programming → Smalltalk, C++
  - rule-based systems → production systems, Agentsheets

Creating the illusion of simplicity (by Grady Booch; IBM Fellow)

<thanks to: Jonathan Dormody>



Examples of Hierarchies

organizations — e.g.: universities such as CU: system → campus → college → department

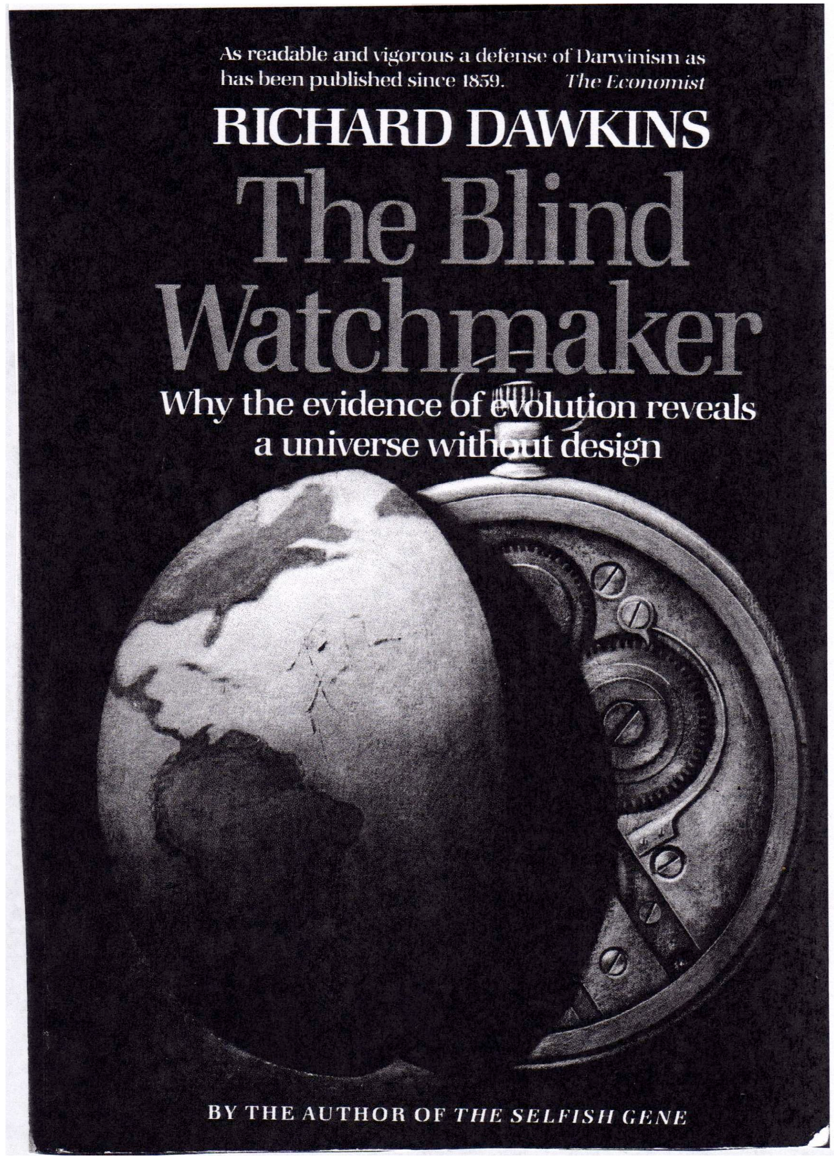
basic structure of matter — molecules → atoms → particles

books → chapters → sections → paragraphs → sentences

Tempus and Hora

- watches of a 1000 parts — interruptions by phone calls
- **Tempus:** interruptions lead to restart from scratch
- **Hora:** subassemblies of ten (at each level) → 111 subassemblies
- **hypotheses / axioms:**
  - "complex systems evolve faster if they can build on stable subsystems"
  - "the evolution of complex forms from simple elements depends critically on the numbers and distribution of potential stable intermediate forms"
  - "given the properties of the parts [of a complex system] and the laws of their interaction, it is not a trivial matter to infer the properties of the whole"

Dawkins, R. (1987) The Blind Watchmaker



Complexity of Designs
—
Dawkins, R. (1987) The Blind Watchmaker

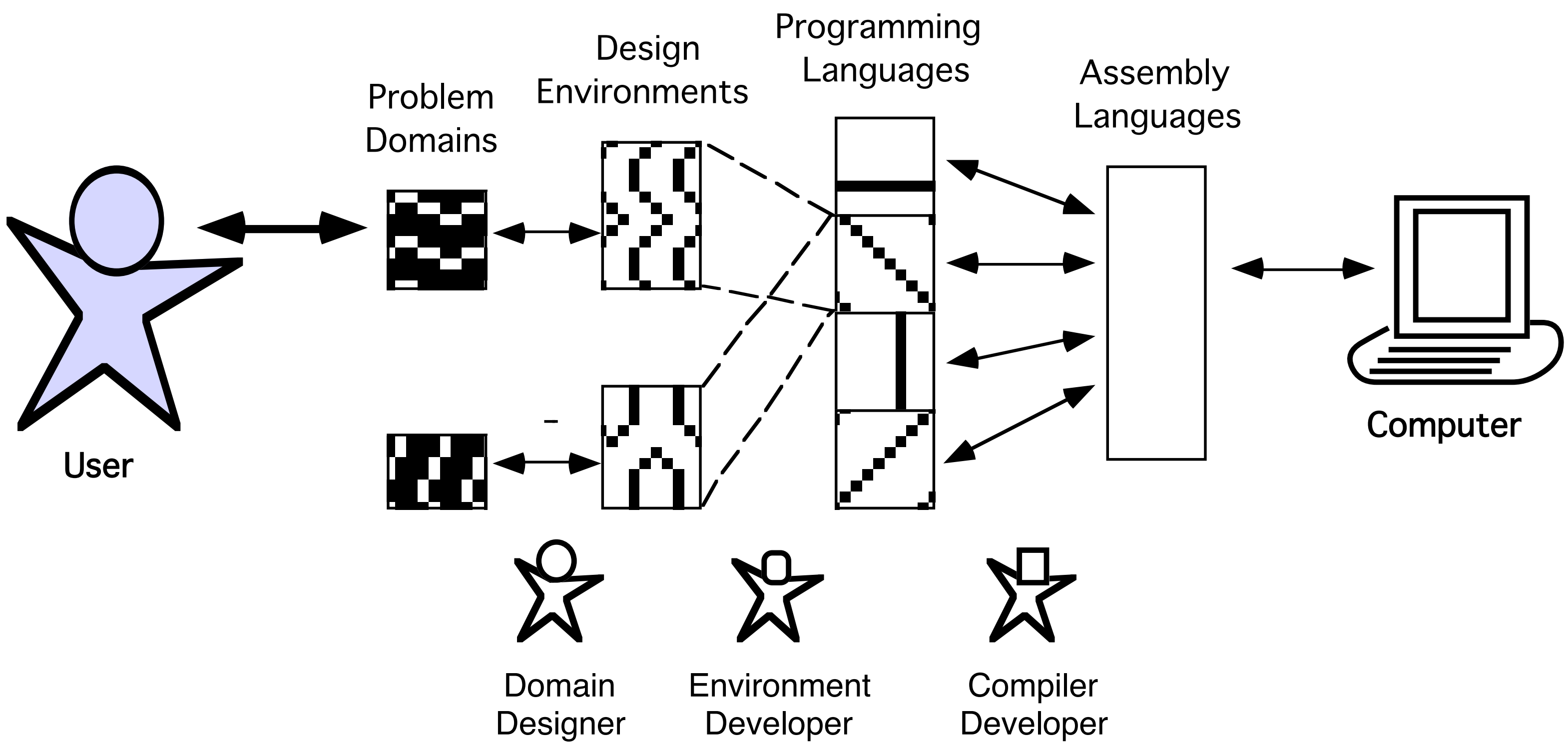
- **complexity of design:**
  - physics = the study of simple things (belong to the natural world)
  - biology = study of complicated things that give the appearance of having been designed for a purpose (belong the world of the artificial?)
  - human made artifacts (computers, cars, airplanes, cities): should be treated as "biological objects" (they are designed for a purpose)

- **claims by Dawkins:**
  - the behavior of physical, non-biological objects is so simple that it is feasible to use existing mathematical language to describe it, which is why physics books are full of mathematics

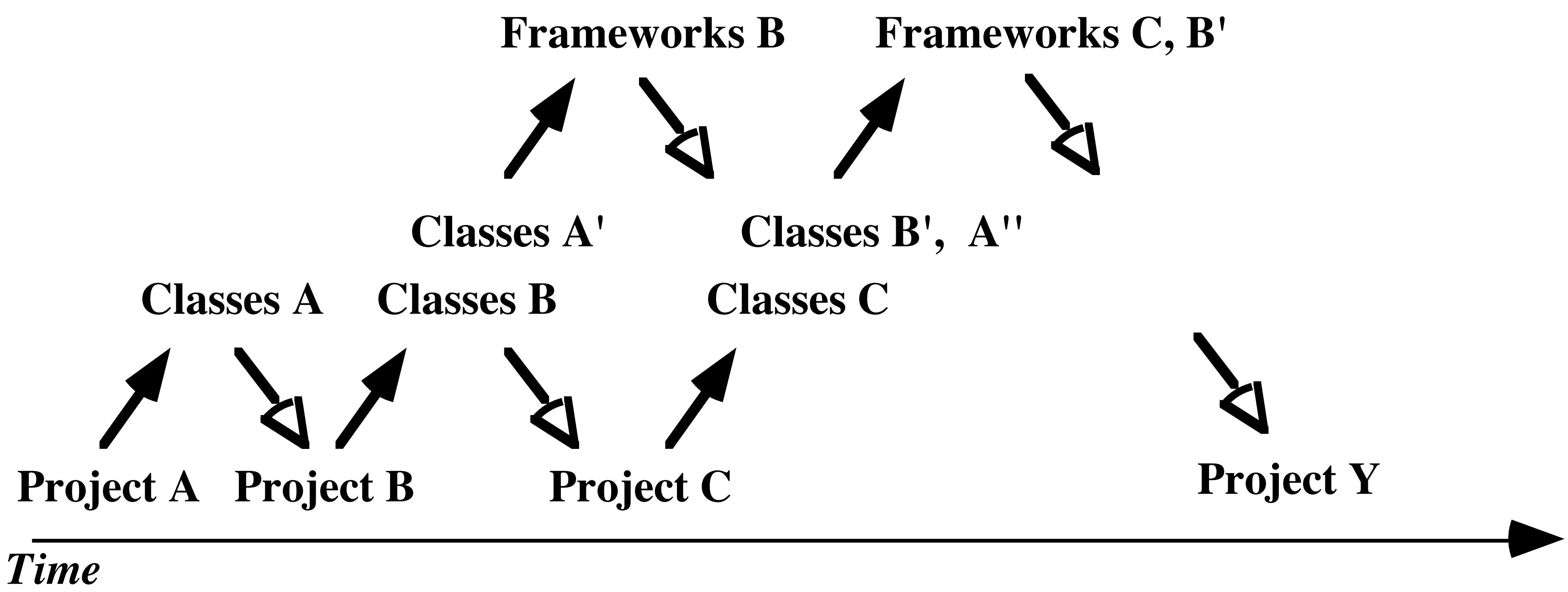
What is a Complex Object?

- has many **parts**
- these parts are of **more than one kind**
- a complex object is something whose constituent parts are arranged in a way that is **unlikely to have arisen by chance alone**
- if there is a complex thing that we do not yet understand, we can come to **understand it in terms of simpler parts** that we do already understand
- hierarchy of components ("**hierarchical reductionism**"): explain a complex entity at any particular level in the hierarchy of organizations in terms of entities only one level down the hierarchy
- **near decomposability** = which the interactions between subsystems are "weak but not negligible"

A Layered Architecture Supporting
Human Problem Domain Interaction



An Evolutionary Model of OO development



- **source:** Fischer, G., Redmiles, D., Williams, L., Puhr, G., Aoki, A., & Nakakoji, K. (1995) "Beyond Object-Oriented Development: Where Current Object-Oriented Approaches Fall Short," Human-Computer Interaction, Special Issue on Object-Oriented Design, 10(1), pp. 79-119.

Explanations for the Diagram

- A progression of software development projects delineate time and provide a long-term context in which the fundamental claims of OO technology, including domain-orientation and reuse, may be studied.
- The solid-tip arrows are primarily associated with evolution, driven by software developers creating new software objects to accommodate new projects.
- Hollow-tip arrows indicate software developers reusing components, although reuse at times leads to redesign. Stable structures of class libraries and frameworks emerge over multiple projects.
- Domain-orientation permeates the model as all of the creation, reuse, and redesign of components is driven by problems arising in the development of specific projects.

Desiderata for Design Processes

- to create a system / a world which offer as many alternatives as possible to future decision makers, avoiding irreversible commitment that they cannot undo → adaptable, end-user modifiable systems

- allow people to design → the act of envisioning possibilities and elaborating them is itself a pleasurable and valuable experience

- to leave the next generation of decision makers with a better body of knowledge and a greater capacity for experience → underdesign, meta-design