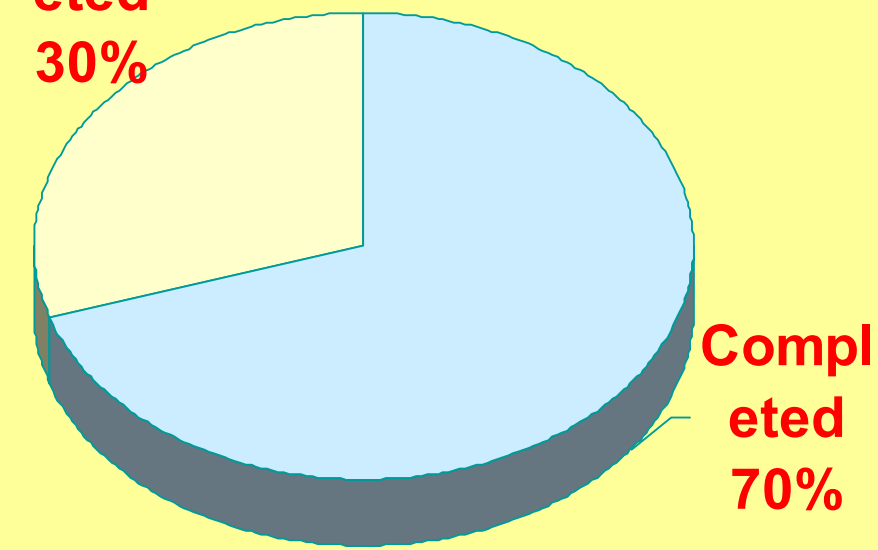
Software Economics: The Motivation for Object Technologies and Iterative Development

Software—A Risky Business

- **Business as usual is not working.**

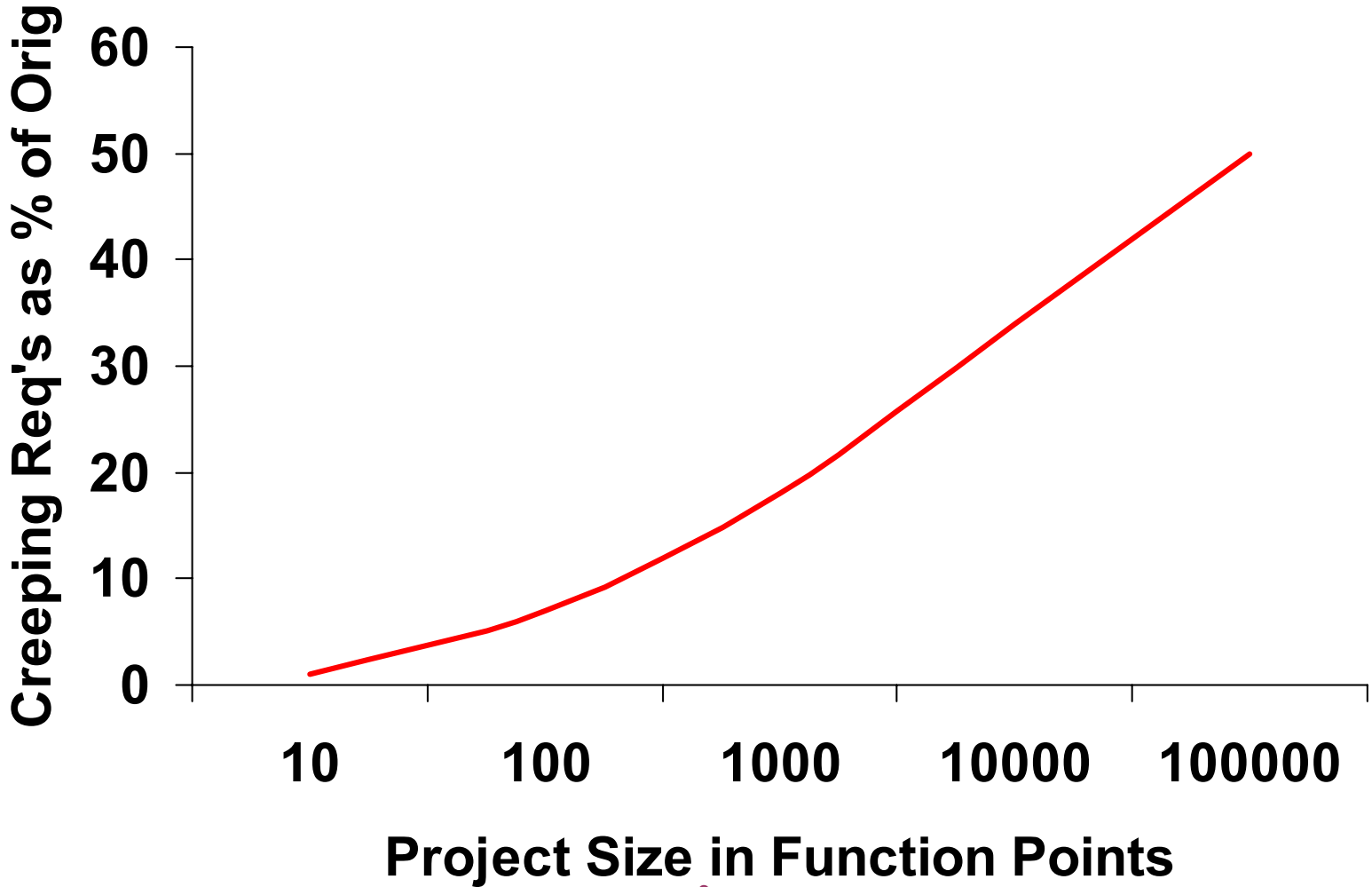


Source: Standish Report, 1994

- All based on waterfall lifecycle.
- 53% cost almost 200% of original estimate.
- Estimated $81 billion spent on failed U.S. projects in 1995.

Faulty Assumption 1: Requirements can be Fairly Accurate

- **Applied Software Measurement, Capers Jones, 1997. Based on 6,700 systems.**



Faulty Assumption 2: Requirements are Stable

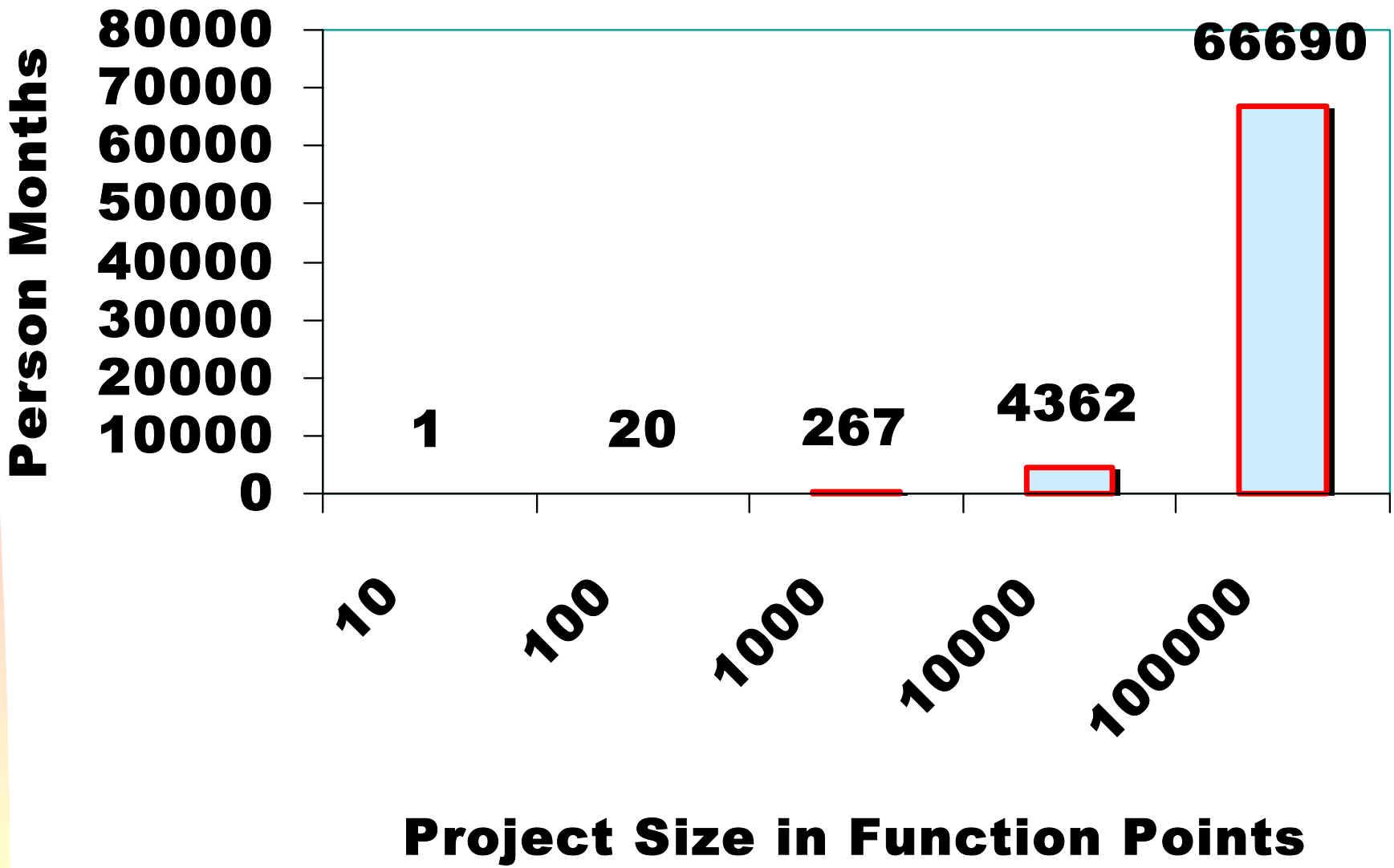
- The market changes—constantly.
- The technology changes.
- The goals of the stakeholders change.

Faulty Assumption 3: The Design can be Done, before Programming

- Ask a programmer.
- Requirements are incomplete and changing.
- Too many variables, unknowns, and novelties.
- A complete specification must be as detailed as code itself.
- Software is very “hard”.
  - Discover Magazine, 1999: Software characterized as the most complex “machine” humankind builds.

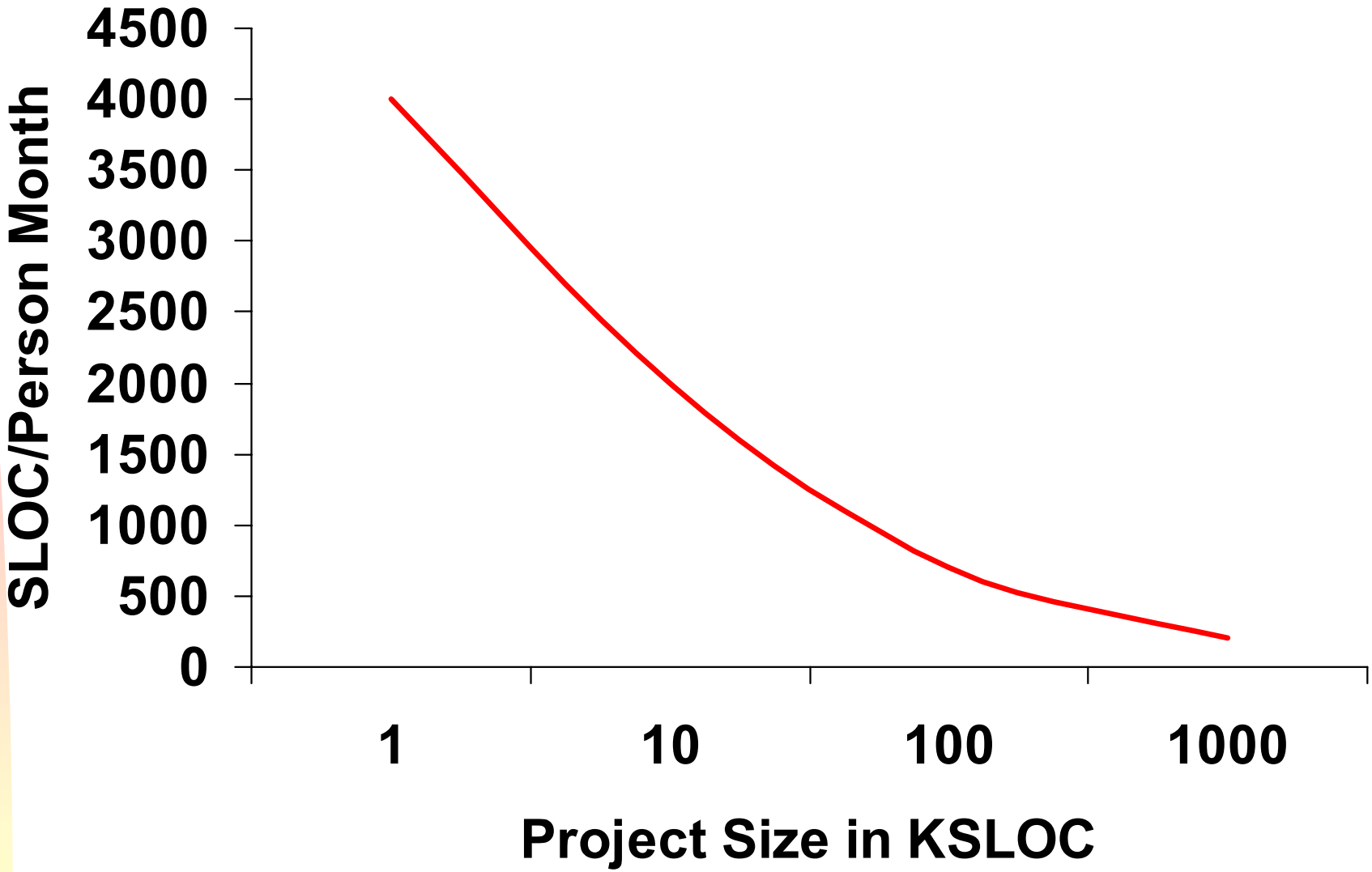
Effort—The Danger of Large Steps

Source: Applied Software Measurement, Capers Jones, 1997. Based on 6,700 systems.



Productivity—The Danger of Large Steps

Source: Measures For Excellence, Putnam, 1992. Based on 1,600 systems.



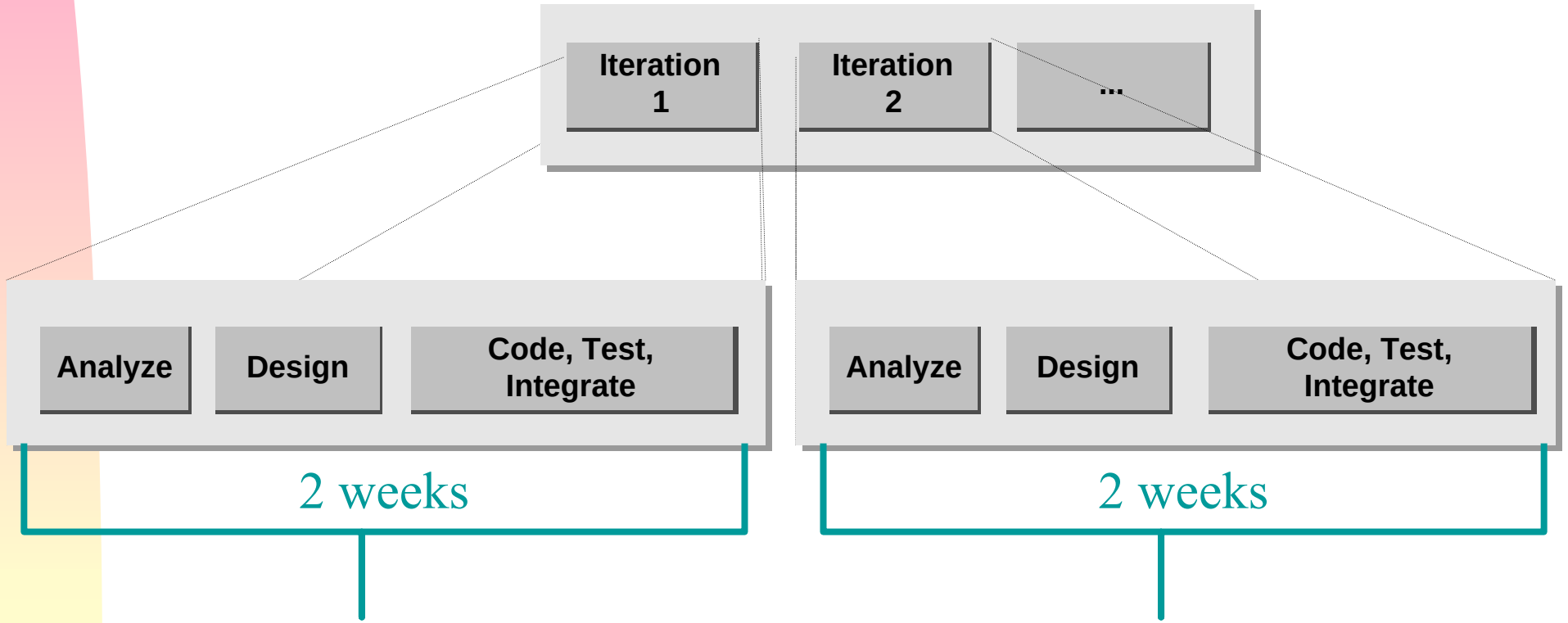
The Voice of Experience and Research

- **In 1994, the DOD dropped their waterfall 2167A specification, because of abysmal failure.**
  - **They now encourage an iterative process.**
- **Virtual every publication on software development process written in the last 5 years has advocated replacing a waterfall with an iterative lifecycle.**
  - ***The Unified Software Development Process***
  - ***Applying UML and Patterns***
  - ***Software Project Management***
  - ***Succeeding with Objects***
  - ***Object Solutions***
  - ***Surviving Object-oriented Projects***
  - **….**

Therefore...

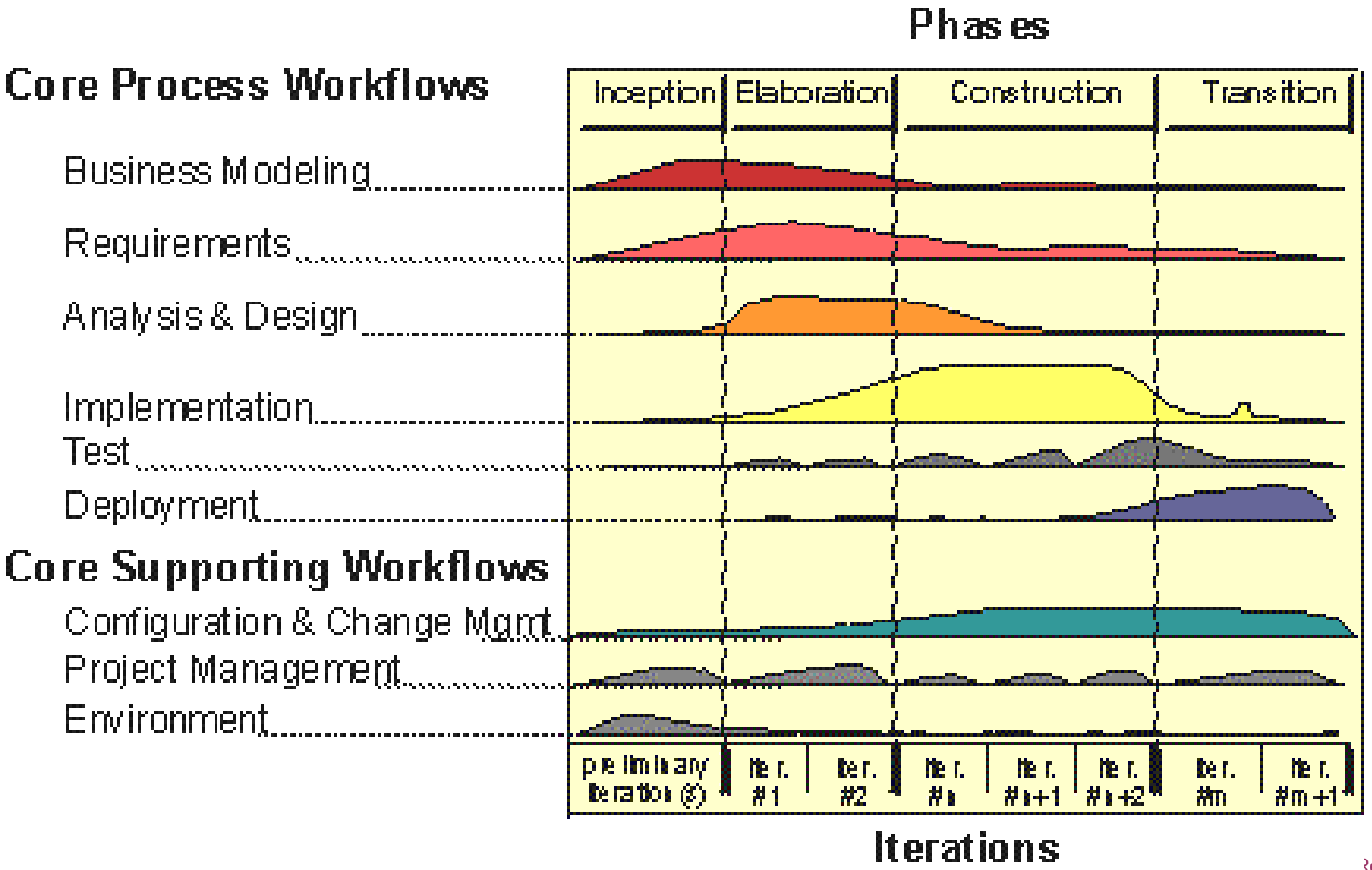
Iterative Development

- **Small steps, feedback and refinement.**
- **Iterative, incremental, time-boxed.**
- **AKA evolutionary, spiral, . . .**



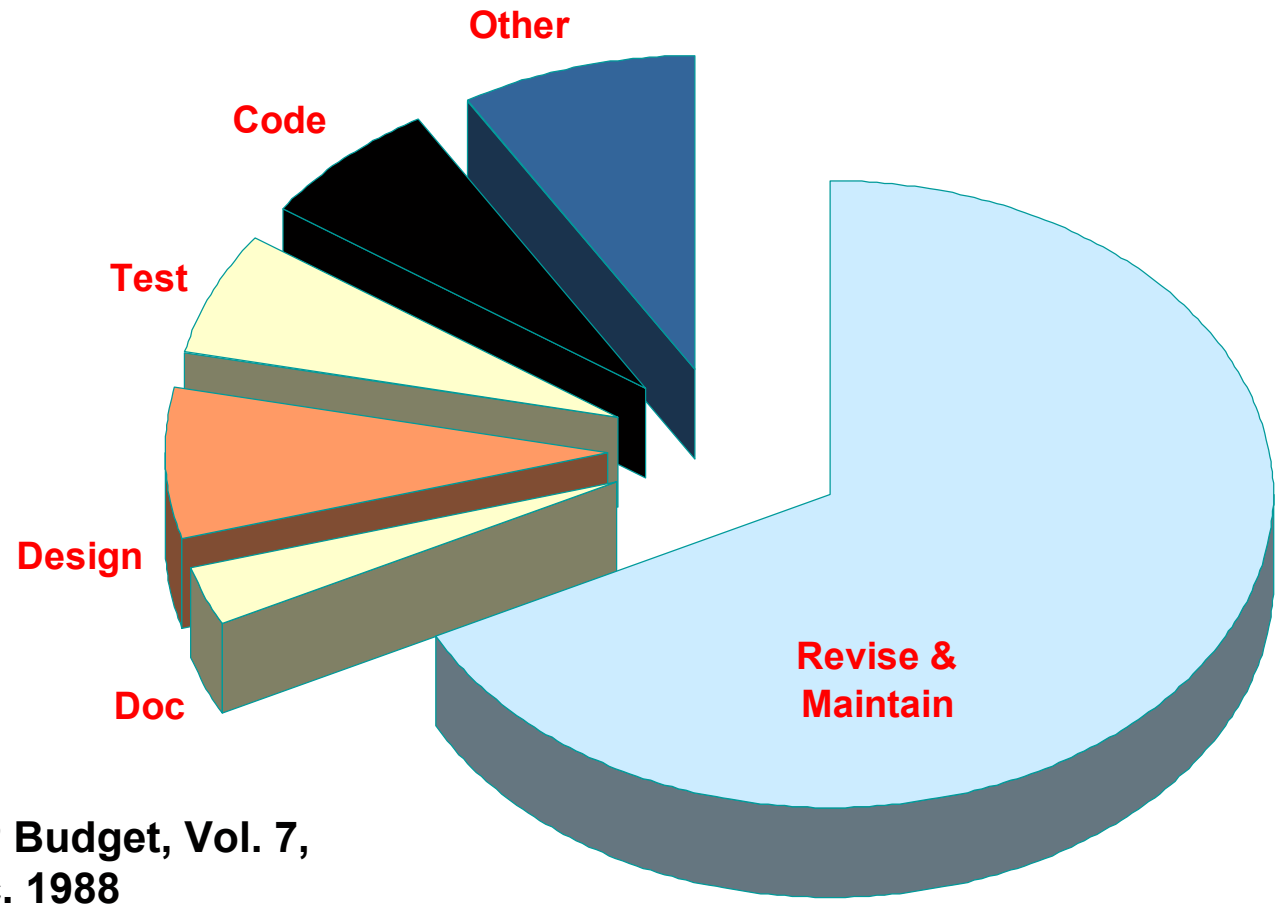
A Popular Iterative Process—The Unified Process

- **Warning: The phases are not iterations.**



The Cost of Change

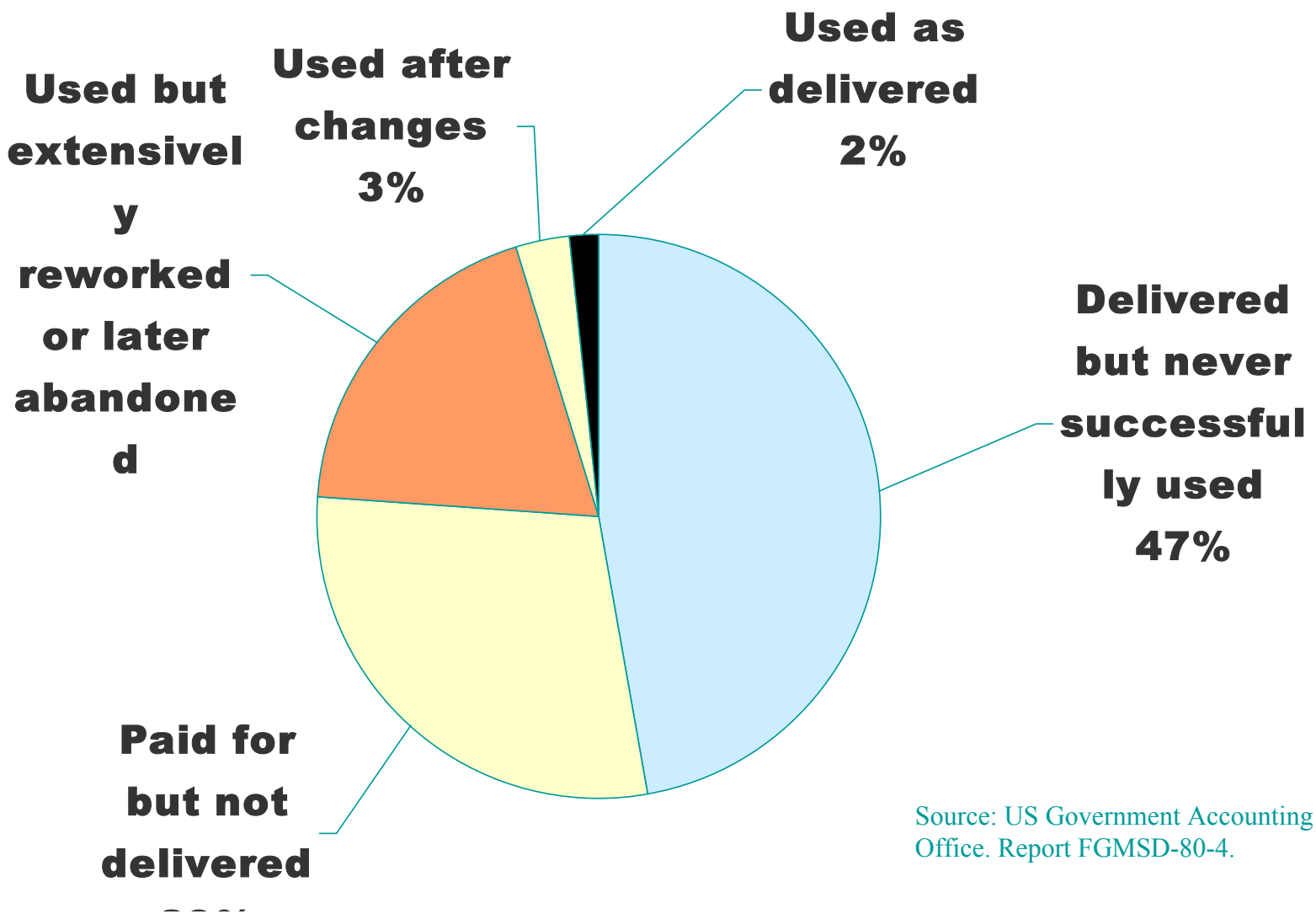
Strategic rational system development plans are based on the complete cost of a system, not solely on development costs.



Source: DP Budget, Vol. 7, No. 12, Dec. 1988

The Cost of Change

Randomly selected U.S. government software projects:



Source: US Government Accounting Office. Report FGMSD-80-4.

The Cost of Change

- **An AT&T study indicated that, on average,**

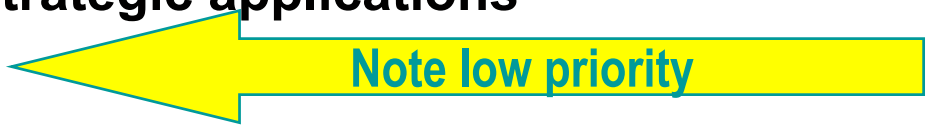
Business Rules change at the rate of 8% per month.

Why Object Technology?

- **Prime software problems are**
  - **lowering the cost and time of change**
  - **Increasing the ease of adaptability**
- **Fact: Object technology is especially good at**
  - **Reducing the time to adapt an existing system (quicker reaction to changes in the business environment).**
  - **Reducing the effort, complexity, and cost of change.**

Why Object Technology?

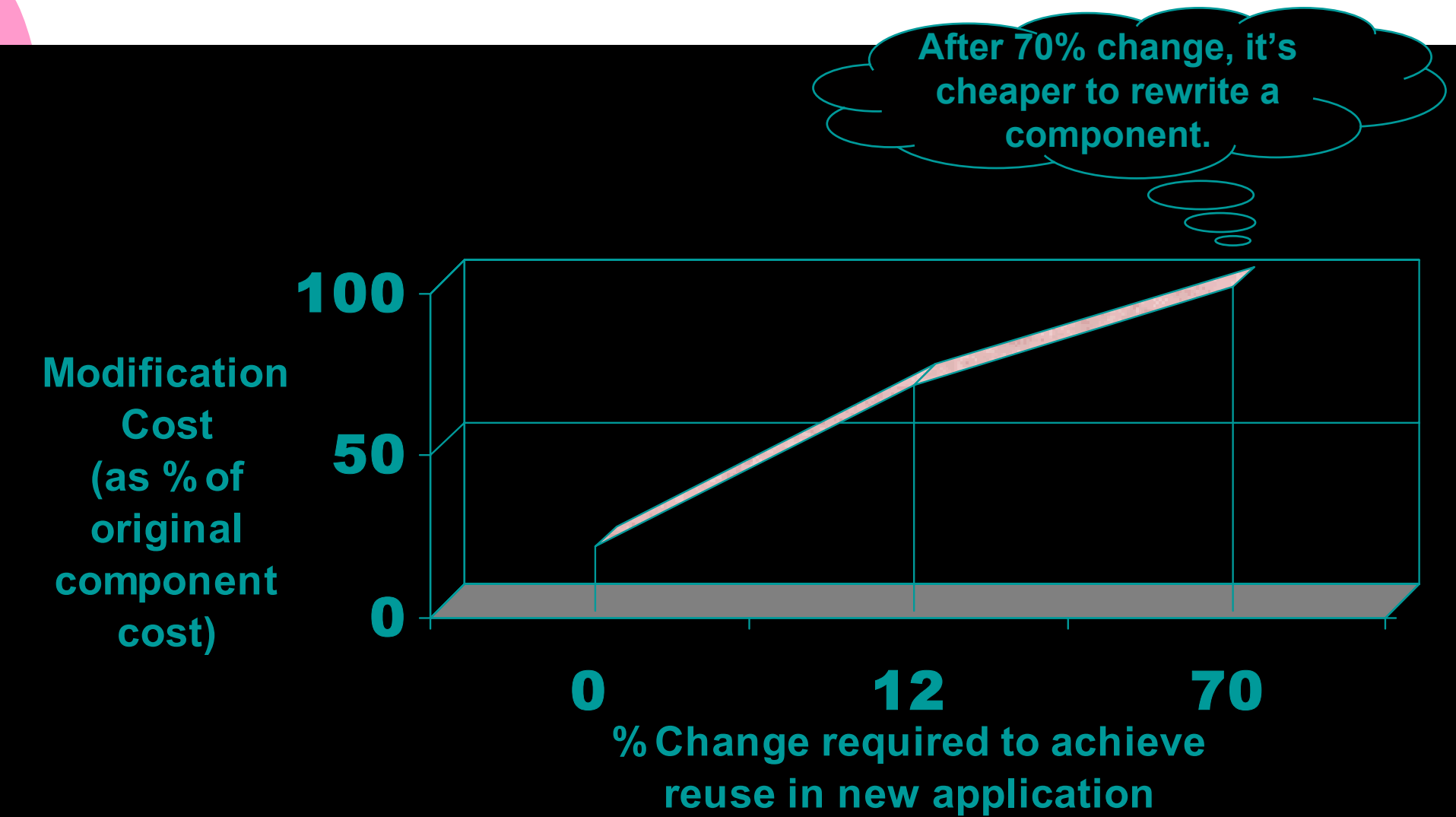
- What about reuse?
- *The Corporate Use of OT*, Dec 1997, Cutter Group.
  Prioritized reasons for adopting OT:
  1. Ability to take advantage of new operating systems and tools
  2. Application maintenance
  3. Cost savings
  4. Development of revenue-producing applications
  5. Encapsulation of existing applications
  6. Improved interfaces
  7. Increased productivity
  8. Participation in "the future of computing"
  9. Proof of ability to do OO development
  10. Quick development of strategic applications
  11. Software reuse



Why Object Technology

- **What about reuse?**
- **Reuse is a good and worthy goal, but you will discover that the real power and advantage of OT is its capacity to support easily adaptable systems.**
- **Unfortunately, reuse is hard, and expert OT organizations see little of it.**
  - **The issues are more *organizational* than technical.**

Reuse at What Level?



Source: W. Pree, OOPSLA '95

Object Reuse Versus Cohesive Framework Reuse

Reuse at What Level?

- **Research shows no relationship between increased reuse and collecting a library of reusable components from prior projects.**
  - **Communications of the ACM, pp 75-87 June, 1995.**
  - **http://ite.gmu.edu/~nnada/pap.html**
- **Reuse is not usually achieved or worthwhile at the object-level.**
- **Focus on:**
  - **The organizational factors to support reuse.**

    http://ite.gmu.edu/~nnada/pap.html

    Software Reuse (Griss)
  - **A culture of *framework* creation and use.**
  - **Reuse of architecture, analysis and design patterns.**

Reuse at the Framework Level

- **Where do expert OT organizations see some reuse?**
  - **With *frameworks*.**

- **However, effective reusable framework creation requires:**
  - **Very skilled object designers/programmers.**

Why Object Technology?

- **Another non-technical, but non-trivial reason is. . .**
  - **Ability to attract and retain talented software engineers.**
- **OT is the undisputed "hot technology" for the foreseeable future, and it is not easy to retain talent unless an IT organization is using it.**

Why Object Technology?

- **Organizations usually start the adoption thinking OT will lead to shorter development time, more reuse, better productivity.**
  - **Unfortunately, that does not usually happen.**
  - **However, they do discover something useful and powerful. . .**
- **"It takes complexity to manage complexity."**
  - **R. Martin, Editor of C++ Report, co-author of *Object-Oriented Analysis and Design*, with Grady Booch.**

Why Object Technology?

"The value of OT (OOA&D, OOP) primarily lies in its ability to handle complex problems and create comprehensible, manageable systems that can scale up to increasing complexity, and that are easily adaptable—*if* designed skillfully."

Craig Larman

1. **Elegantly tackle complexity & create easy adaptability.**
2. **. . .**
9. **Productivity**
10. **Reuse**

The productivity is realized in the maintenance or modification phases of a system—often with profoundly faster changes, if the system was designed skillfully.