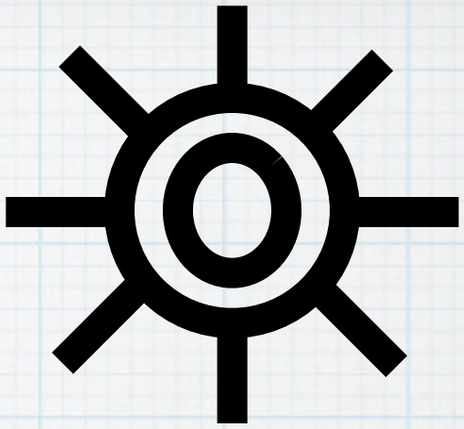
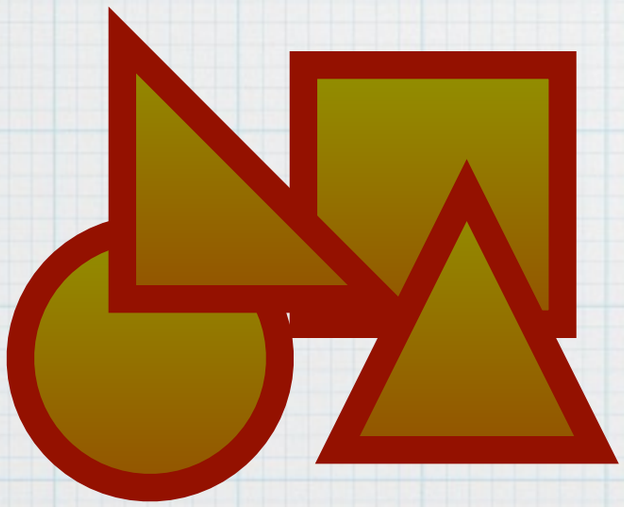


Requirements Engineering
Fundamentals

Les Waguespack, Ph.D.



Slides Two

Copyright and References

* Requirements Engineering, Kotonya & Sommerville, Wiley, Chichester, West Sussex, England, ISBN 0-471-97208-8
* Software Requirements Engineering, Second Edition, Richard H. Thayer and Merlin Dorfman, eds., pp. 7-22. Los Alamitos, Calif.: IEEE Computer Society Press, 1997.
* Use Case Modeling, Bittner & Spence, Addison-Wesley / Pearson Education, Inc., Boston, MA, ISBN 0-201-70913-9
* Writing Effective Use Cases, Cockburn, Addison-Wesley, Boston, MA, ISBN 0-201-70225-8
* UML and the Unified Process - Practical Object-Oriented Analysis and Design, Arlow & Neustadt, Addison-Wesley / Pearson Education, Inc., Boston, MA, ISBN 0-201-77060-1
* Business Modeling With UML, Eriksson & Penker, Wiley, Indianapolis, IN, ISBN 0-471-29551-5
* UML 2 Toolkit, Eriksson, Penker, Lyons & Fado, Wiley, Indianapolis, IN, ISBN 0-471-46361-2
* Enterprise Modeling With UML Designing Successful Software Through Business Analysis, Addison-Wesley, Reading, MA, ISBN 0-201-43313-3
* Object Oriented Systems Engineering, Waguespack, course notes CS390, CS460, CS630, CS771, Computer Information Systems Department, Bentley College, Waltham, MA.

Outline

1. Elicitation
2. Elicitation Practices

1. Elicitation

* e·lic·it vt
* 1. to cause or produce something as a reaction or response to a stimulus of some kind
* 2. to bring to light, or cause somebody to disclose, something hidden or not immediately obvious, especially by a process of questioning or research

Purposeful Elicitation

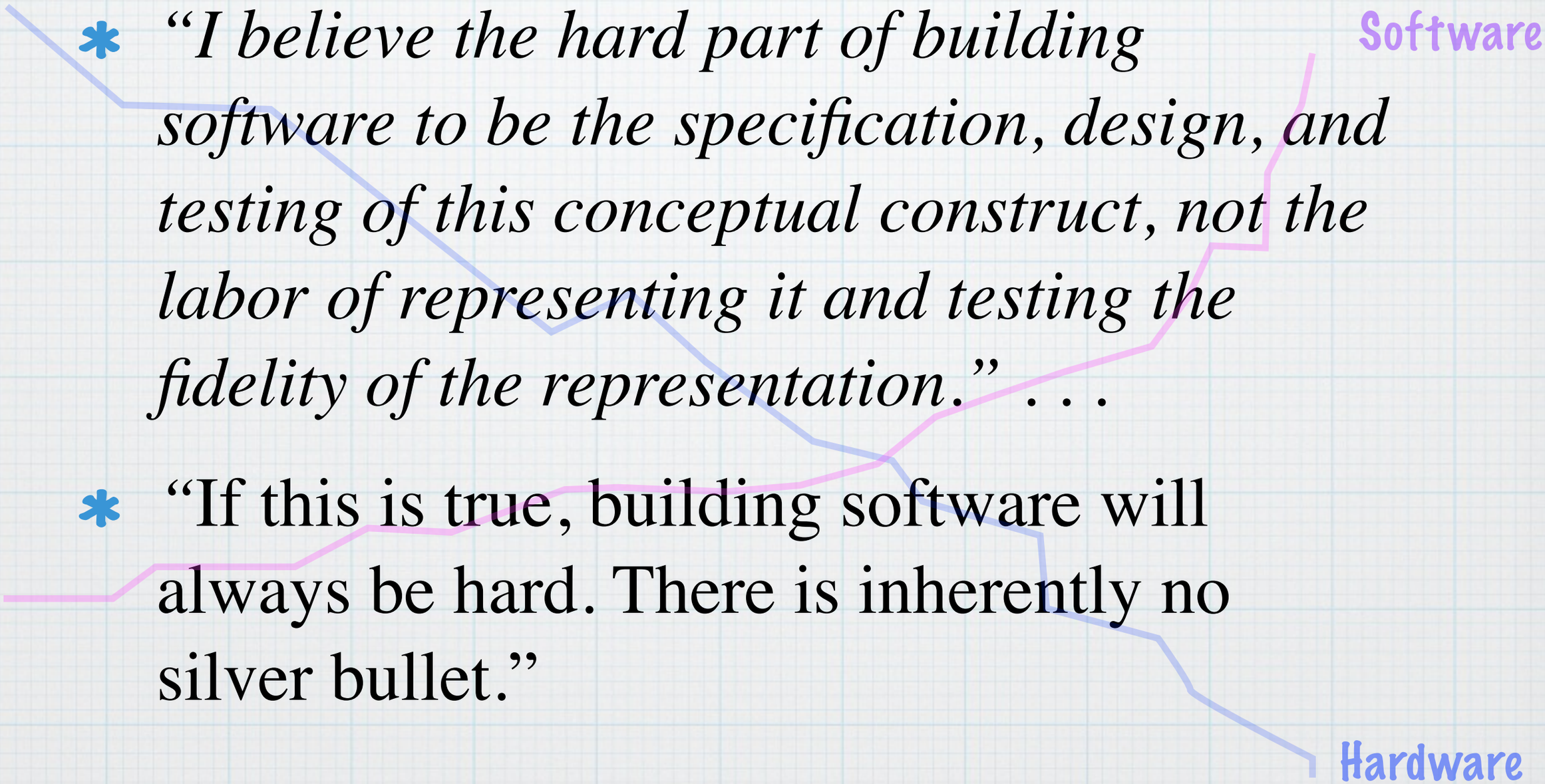
* systematic information gathering
  * set objectives
  * identify background and context
  * organize: gather, attribute, catalog
  * identify requirement origins - application domain, business policy, "accident of implementation," quality improvement goal
  * authenticate - necessity, consistency, completeness, feasibility
  * build consensus - challenge dreams, triangulate on perspectives, prioritize, converge on group objectives

Elicitation Challenges

* the prime stakeholder sources may be the least accessible - adjust the timeline
* the sleuth needs to know enough about the problem to ask meaningful questions
* some key stakeholders may not have "bought into" the new system "quest"

Fred Brooks 1987

* *"I believe the hard part of building software to be the specification, design, and testing of this conceptual construct, not the labor of representing it and testing the fidelity of the representation." . . .*



* "If this is true, building software will always be hard. There is inherently no silver bullet."

Brooks' "No Silver Bullet"

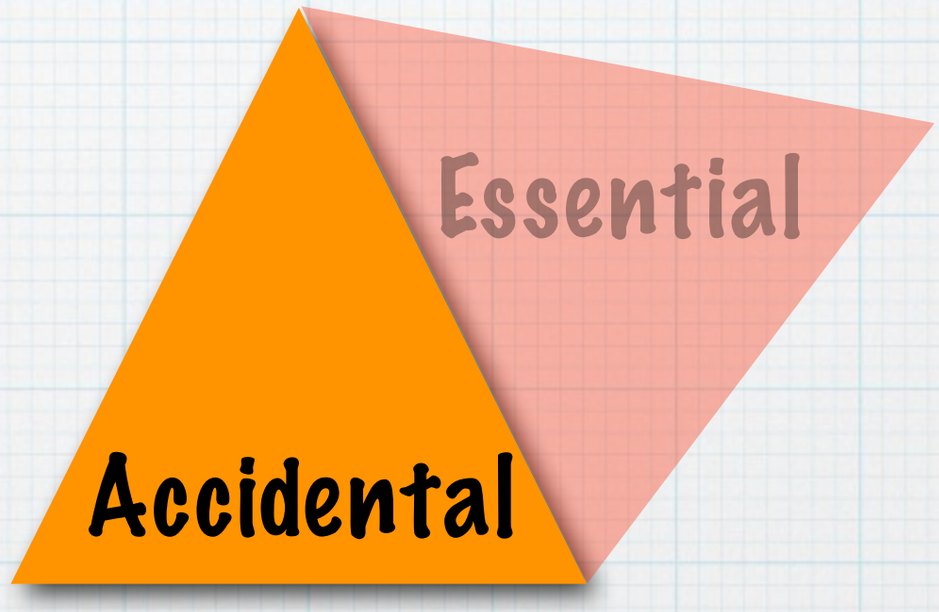
* Software development is difficult because of:
  * essential aspects of the requirement itself
  * accidental aspects due to development process choices

Essential Difficulties

* complexity - requirement specific
* conformity - adjusting to human situation
* changeability - evolution absorbent
* invisibility - modeled abstraction

Accidental Difficulties

* Development Process Choices
  * specification language(s)
  * technology combinations
  * development tools
  * methodologies
  * ...

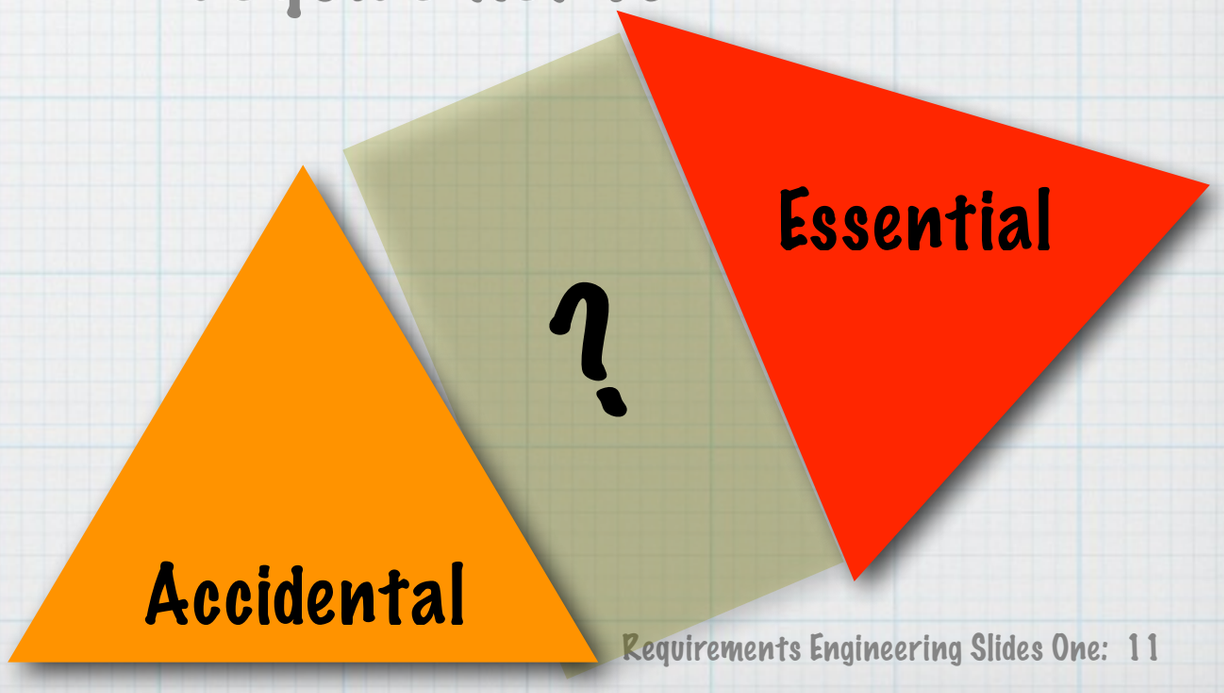


Software Engineering's Success

* Accidental difficulties are well understood and there are numerous "remedies" in place

* Essential difficulties are indeed "fixed" as long as the requirement is "fixed"

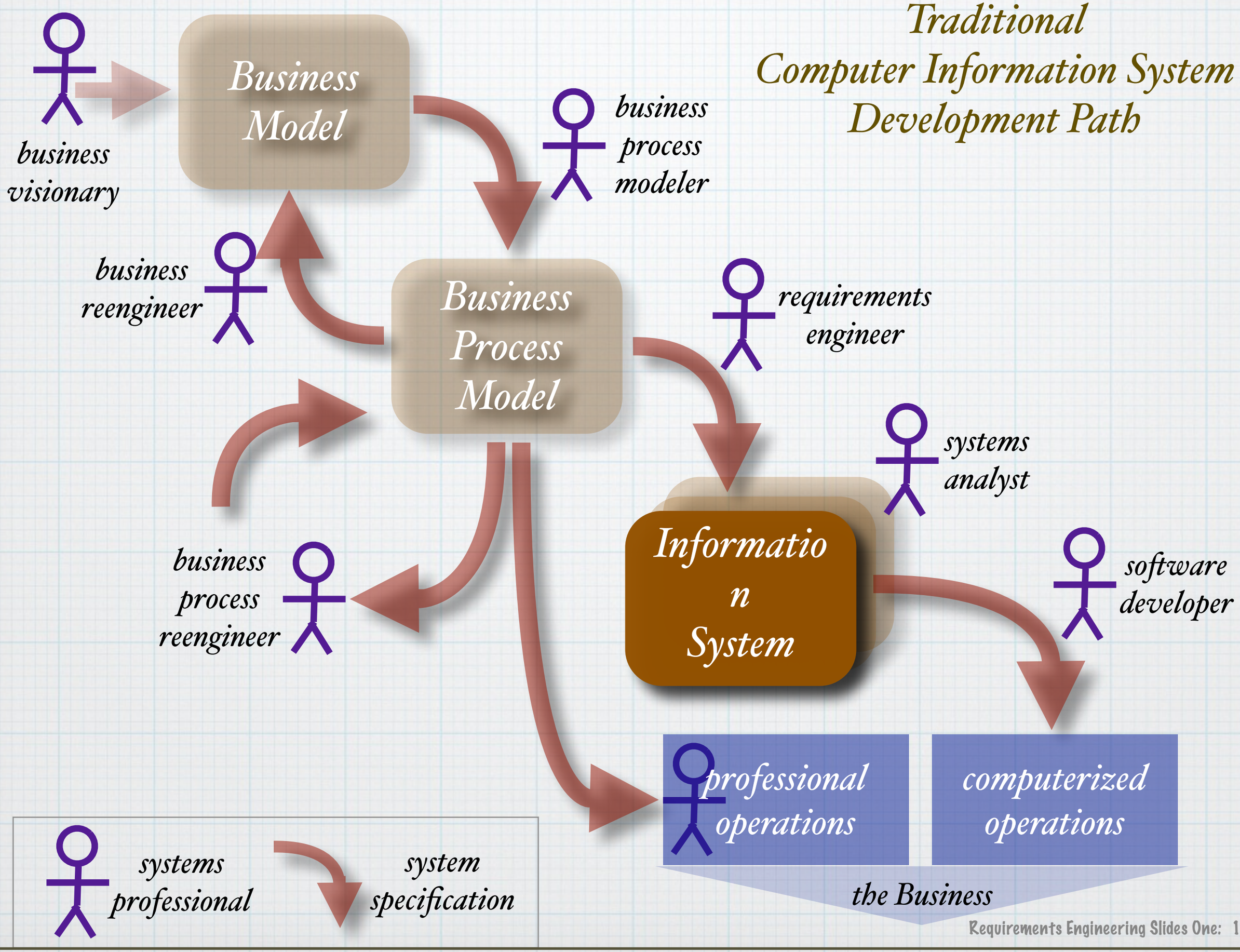
* Much cost of ownership lies in "revisiting" the essential difficulties in the form of the underlying "business requirements"



Accidental vs. Essential?

* Essential aspects map to the Business Model - aka "factors of effectiveness"
* Accidental aspects map to the Business Process Model and Implementation Choices - aka "factors of efficiency"

Traditional Computer Information System Development Path



TRUST

* trust n
  * 1. confidence in and reliance on good qualities, especially fairness, truth, honor, or ability
  * 2. responsibility for taking good care of somebody or something
  * 3. the position of somebody who is expected by others to behave responsibly or honorably
  * 4. somebody who or something that people place confidence or faith in (archaic or literary)
  * 5. hopeful reliance on what will happen in the future
  * 6. something entrusted to somebody to be responsible for

2. Elicitation Practices

* strategies
* interviewing
* scenarios
* prototyping
* requirement reuse

Elicitation Strategies

* "divide and conquer" - partition the "problem" (record the decomposition tree)
* attempt to abstract categories / classes of function or behavior
* cast the same requirement from multiple stakeholder perspectives to enrich "corporate understanding" (projection)

Interviewing

* the purpose is to learn and validate information from client practitioners
  * "open ended" interviews assume the interviewee will guide the discussion (open ended learning)
  * "closed" interviews presume a set of questions that need answers (validating existing assumptions / info)
* mutual openness
  * "seeking benefit for the organization" is a key premise
  * skepticism raises "presumption" and "guarded answers"

Group vs. Individual Interviews

* group sessions tend toward goals, policies, and "architecture"
  * exposes client information to "peer review"
  * separates "folk" business models from reality
* individual interviews tend toward detail
  * focuses on work product production
  * precisely identifies current practice issues
* consistency is the convergence of both

Scenarios

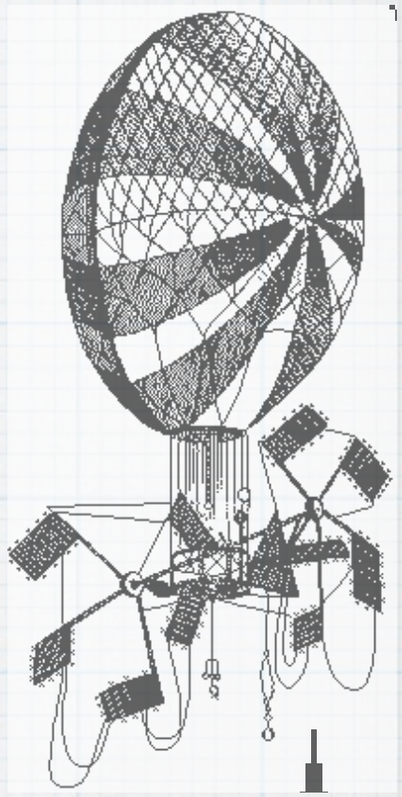
* A scenario is a "cogent" story of what happens in the system domain
* Tends to reveal the "normal" behavior
* Formal scenarios provide structure
  * USE CASE: pre-conditions, initiator, normal flow, exception flow, scenario dependencies, post-conditions

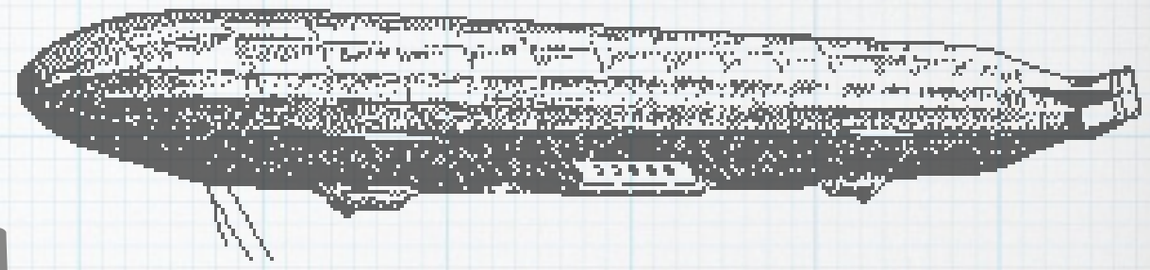
Prototyping

* "One of a kind" problems are seldom well understood
  * ("We do things differently here!")
* Requirements are more likely "discovered" than elicited (even by user!)
  * ("I didn't know that's how we did that!")
* "Test driving" features validates whether they are "client-rational"
  * ("I know that's what I asked for but I don't want that!")

Prototyping

- construction of a model of the final system limiting realism in specific areas of detail
  - performance
  - capacity
  - error recovery
  - external system integration
- to allow users and requirements analysts to explore system potential with concept models
- to test interfaces and controls for user feasibility
- to eliminate dreams from requirements





- Types of Prototyping
  - Exploratory
    - exploring problem design options (in case of user uncertainty)
  - Experimental
    - analysis of alternative designs (e.g. interfaces)
  - Evolutionary
    - prototype as the first of series of continuing releases of the product

Requirement Reuse

* many requirements are "déja vu"
  * application domain
  * organizational standards
  * organizational policies
* careful "requirement asset" mgmt can save 50% or more on these issues
* reused requirements still need to be re-validated

Wrap Up

1. Elicitation

* purpose driven
* challenge rich

2. Elicitation Practices

* strategies
* interview, scenario, prototyping, reuse