A Design Quality Learning Unit in OO Modeling Bridging Engineer and Artist

Les Waguespack, Ph.D.
Professor & Chairperson of Computer Information Systems

Bentley University
Waltham Massachusetts

ISECON - Wilmington, NC - 2011-11-3
“How is it that one system is more effective, appealing, satisfying and/or more beautiful than another to its stakeholder community?”

“How is it that one system is more effective, appealing, satisfying and/or more beautiful than another to its stakeholder community?”


How is it that one object model is better than another?
“How is it that one system is more effective, appealing, satisfying and/or more beautiful than another to its stakeholder community?”


How is it that one object model is better than another?

What determines design quality in an object model?
defining design quality
defining design quality

quality  |ˈkwælɪtɪ| noun

1 the standard of something as measured against other things of a similar kind; the degree of excellence of something: *an improvement in product quality* | *people today enjoy a better quality of life.*

• general excellence of standard or level: *a masterpiece for connoisseurs of quality* | [*as modifier*] : *a wide choice of quality beers.*

2 a distinctive attribute or characteristic possessed by someone or something: *he shows strong leadership qualities* | *the plant’s aphrodisiac qualities.*
defining design quality

quality |ˈkwɒləti| noun

1 the standard of something as measured against other things of a similar kind; the degree of excellence of something: an improvement in product quality | people today enjoy a better quality of life.

• general excellence of standard or level: a masterpiece for connoisseurs of quality | [ as modifier ] : a wide choice of quality beers.

2 a distinctive attribute or characteristic possessed by someone or something: he shows strong leadership qualities | the plant’s aphrodisiac qualities.

design |dəˈzīn| verb [ with obj. ]

decide upon the look and functioning of (a building, garment, or other object), typically by making a detailed drawing of it: a number of architectural students were designing a factory | [ as adj. with submodifier ] (designed) : specially designed buildings.

• do or plan (something) with a specific purpose or intention in mind: [ with obj. and infinitive ] : the tax changes were designed to stimulate economic growth.
the individual’s experience of design quality

implementation  threshold  expectation  mindset
the individual’s experience of design quality

implementation  threshold  expectation  mindset

the “mental picture” the observer brings to the experience within which they will “understand” the experience.
the individual’s experience of design quality

implementation  threshold  expectation  mindset

the subset of the observer’s mindset (conscious or unconscious) that is specifically relevant to the event

the “mental picture” the observer brings to the experience within which they will “understand” the experience
the individual’s experience of design quality

implementation

the point of encounter between the expectation and the system’s features

threshold

expectation

the subset of the observer’s mindset (conscious or unconscious) that is specifically relevant to the event

mindset

the “mental picture” the observer brings to the experience within which they will “understand” the experience
the individual’s experience of design quality

implementation
the assembled artifact’s realization that creates the opportunity for observation

threshold
the point of encounter between the expectation and the system’s features

expectation
the subset of the observer’s mindset (conscious or unconscious) that is specifically relevant to the event

mindset
the “mental picture” the observer brings to the experience within which they will “understand” the experience
the individual’s experience of design quality

implementation
the assembled artifact’s realization that creates the opportunity for observation

threshold
the point of encounter between the expectation and the system’s features

expectation
the subset of the observer’s mindset (conscious or unconscious) that is specifically relevant to the event

mindset
the “mental picture” the observer brings to the experience within which they will “understand” the experience
the community’s experience of design quality

implementation → threshold → expectation ← mindset
the community’s experience of design quality

implementation
the assembled artifact’s realization that creates the opportunity for observation

threshold
the point of encounter between the expectation and the system’s features

expectation
the subset of the observer’s mindset (conscious or unconscious) that is specifically relevant to the event

mindset
the “mental picture” the observer brings to the experience within which they will “understand” the experience
Quality Design: The application of quality Principles in the process of creating artifacts
“How is it that one system is more effective, appealing, satisfying and/or more beautiful than another to its stakeholder community?”


How is it that one object model is better than another?

What determines design quality in an object model?
The Design Process

Stakeholder intensions
requirement elements
model elements
design elements
The Design Process

“rules of thumb”
Traditional patterns

Stakeholder intensions
requirement elements
model elements
design elements

Representational paradigm or Ontology

©LJW, 2010 All Rights Reserved
The Design Process

“rules of thumb”
Traditional patterns

Stakeholder intensions
requirement elements
model elements
design elements

Representational paradigm or Ontology

Resulting Design Elements
The Design Process

“Rules of thumb”
Traditional patterns

Stakeholder intentions
Requirement elements
Model elements
Design elements

Representational paradigm or Ontology

Resulting Design Elements
The Design Process

Choice Property-Driven Design Principles

Stakeholder intentions
requirement elements
model elements
design elements

Representational paradigm or Ontology

Resulting Design Elements
Thriving Systems Theory
Thriving Systems Theory

Properties of Order in Nature
Thriving Systems Theory

Art & Architecture

Properties of order in nature
Architectural domains
Thriving Systems Theory

Art & Architecture

Properties of Order in Nature
Architectural Domains
Discipline Defined Design Characteristics
Thriving Systems Theory

- Levels of Scale
- Positive Space
- Strong Centers
- Boundaries
- Deep Interlock and Ambiguity
- Gradients
- Good Shape
- Roughness
- Alternating Repetition
- Echoes
- Simplicity and Inner Calm
- Local Symmetries
- The Void
- Contrast
- Not Separateness

Properties of Order in Nature
Architectural Domains
Discipline Defined Design Characteristics
Vocabulary of Design Properties
Thriving Systems Theory

- Levels of Scale
- Positive Space
- Strong Centers
- Boundaries
- Deep Interlock and Ambiguity
- Gradients
- Good Shape
- Roughness
- Alternating Repetition
- Echoes
- Simplicity and Inner Calm
- Local Symmetries
- The Void
- Contrast
- Not Separateness

Properties of Order in Nature
Architectural Domains
Discipline Defined Design Characteristics
Vocabulary of Design Properties
Thriving Systems Theory

- Levels of Scale
- Positive Space
- Strong Centers
- Boundaries
- Deep Interlock and Ambiguity
- Gradients
- Good Shape
- Roughness
- Alternating Repetition
- Echoes
- Simplicity and Inner Calm
- Local Symmetries
- The Void
- Contrast
- Not Separateness

Properties of Order in Nature
Architectural Domains
Discipline Defined Design Characteristics
Vocabulary of Design Properties
How is it that one system is more effective, appealing, satisfying and/or more beautiful than another to its stakeholder community? This question drove Christopher Alexander’s fifty-year quest to explain great physical architecture and give birth to pattern-languages for building that underpin much of modern systems engineering.

How is it that so many individual stakeholders consistently recognize the same quality, the same beauty in a system? This question led George Lakoff to research the role of conceptual metaphor in human understanding.

What is essential to stakeholders’ satisfaction with systems? Fred Brooks addressed this question in No Silver Bullet: Essence and Accidents of Software Engineering.

This monograph fuses these diverse streams of thought in proposing Thriving Systems Theory by translating Alexander’s properties of physical design quality into the abstract domain of information systems and modeling.

Metaphor-Driven Modeling incorporates the theory while examining its impact throughout the system life cycle: modeling, design and deployment. The result is holistic and innovative, a perspective on system quality invaluable to students, practitioners and researchers of software and systems engineering.

Les Waguespack is a computer science Ph.D., professor and chairperson of computer information systems at Bentley University, USA. Dr. Waguespack’s experience as programmer, software engineer, software architect, database architect, project manager and systems consultant underpins 35 years of teaching and research, the last 20+ years teaching object-oriented modeling and systems engineering to undergraduates, graduate students and practicing professionals.
These choice properties propose a coherent, descriptive language including:

- a vocabulary for describing and comparing aspects of system components and structures, and
- design actions to guide design choices leading to desirable system characteristics.
<table>
<thead>
<tr>
<th><strong>Choice</strong></th>
<th><strong>Property</strong></th>
<th><strong>Modeling Action</strong></th>
<th><strong>Action Rendition</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Stepwise Refinement</td>
<td>elaborate</td>
<td>develop or present (a theory, policy, or system) in detail</td>
</tr>
<tr>
<td>2</td>
<td>Cohesion</td>
<td>factor</td>
<td>express as a product of factors</td>
</tr>
<tr>
<td>3</td>
<td>Encapsulation</td>
<td>encapsulate</td>
<td>enclose the essential features of something succinctly by a protective coating or membrane</td>
</tr>
<tr>
<td>4</td>
<td>Extensibility</td>
<td>extend</td>
<td>render something capable of expansion in scope, effect, or meaning</td>
</tr>
<tr>
<td>5</td>
<td>Modularization</td>
<td>modularize</td>
<td>employing or involving a module or modules as the basis of design or construction</td>
</tr>
<tr>
<td>6</td>
<td>Correctness</td>
<td>align</td>
<td>put (things) into correct or appropriate relative positions</td>
</tr>
<tr>
<td>7</td>
<td>Transparency</td>
<td>expose</td>
<td>reveal the presence of (a quality or feeling)</td>
</tr>
<tr>
<td>8</td>
<td>Composition of Function</td>
<td>assemble</td>
<td>fit together the separate component parts of (a machine or other object)</td>
</tr>
<tr>
<td>9</td>
<td>Identity</td>
<td>identify</td>
<td>establish or indicate who or what (someone or something) is</td>
</tr>
<tr>
<td>10</td>
<td>Scale</td>
<td>focus</td>
<td>(of a person or their eyes) adapt to the prevailing level of light [abstraction] and become able to see clearly</td>
</tr>
<tr>
<td>11</td>
<td>User Friendliness</td>
<td>accommodate</td>
<td>fit in with the wishes or needs of</td>
</tr>
<tr>
<td>12</td>
<td>Patterns</td>
<td>pattern</td>
<td>give a regular or intelligible form to</td>
</tr>
<tr>
<td>13</td>
<td>Programmability</td>
<td>generalize</td>
<td>make or become more widely or generally applicable</td>
</tr>
<tr>
<td>14</td>
<td>Reliability</td>
<td>normalize</td>
<td>make something more normal, which typically means conforming to some regularity or rule</td>
</tr>
<tr>
<td>15</td>
<td>Elegance</td>
<td>coordinate</td>
<td>bring the different elements of (a complex activity or organization) into a relationship that will ensure efficiency or harmony</td>
</tr>
<tr>
<td>Choice Property</td>
<td>Modeling Action</td>
<td>Action Rendition Through Object Modeling</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>----------------------------------------</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Stepwise Refinement</td>
<td>elaborate</td>
<td>employing class inheritance to expose and elaborate responsibilities and information management through levels of abstraction</td>
</tr>
<tr>
<td>2</td>
<td>Cohesion</td>
<td>factor</td>
<td>locating both service and data attributes for independent sufficiency</td>
</tr>
<tr>
<td>3</td>
<td>Encapsulation</td>
<td>encapsulate</td>
<td>delineate the responsibilities, knowledge, and interface of objects</td>
</tr>
<tr>
<td>4</td>
<td>Extensibility</td>
<td>extend</td>
<td>service abstractions that enable and control polymorphic extension</td>
</tr>
<tr>
<td>5</td>
<td>Modularization</td>
<td>modularize</td>
<td>individual and successive applications of encapsulation to compartmentalize design decisions and abstract system structure</td>
</tr>
<tr>
<td>6</td>
<td>Correctness</td>
<td>align</td>
<td>self-validating object interfaces that implement verification behavior</td>
</tr>
<tr>
<td>7</td>
<td>Transparency</td>
<td>expose</td>
<td>structural &amp; behavioral relationships the show “fit” and “cooperation”</td>
</tr>
<tr>
<td>8</td>
<td>Composition of Function</td>
<td>assemble</td>
<td>design favoring simple parts combined for sophisticated function</td>
</tr>
<tr>
<td>9</td>
<td>Identity</td>
<td>identify</td>
<td>stakeholder visible constructs reflected in classes and relationships</td>
</tr>
<tr>
<td>10</td>
<td>Scale</td>
<td>focus</td>
<td>grouping objects &amp; relationships in simplifying wrappers and facades</td>
</tr>
<tr>
<td>11</td>
<td>User Friendliness</td>
<td>accommodate</td>
<td>using user’s terminology and visible topology to maintain a familiarity that invites users into validation and verification</td>
</tr>
<tr>
<td>12</td>
<td>Patterns</td>
<td>pattern</td>
<td>nurturing familiarity and empowering evolution through polymorphism</td>
</tr>
<tr>
<td>13</td>
<td>Programmability</td>
<td>generalize</td>
<td>predicting and enabling adaptation of behavior without construction</td>
</tr>
<tr>
<td>14</td>
<td>Reliability</td>
<td>normalize</td>
<td>clearly distinguishing essential elements derived from business rules from artifacts necessary for technological compatibility or platform</td>
</tr>
<tr>
<td>15</td>
<td>Elegance</td>
<td>coordinate</td>
<td>satisfaction from an intuitively obvious design based not on having nothing else to add, but rather having nothing else that can be left out</td>
</tr>
</tbody>
</table>
Design Quality Naming
Design Quality Naming

1. Stepwise Refinement
5. Modularization

goal-directed decomposition reflects stakeholder perception of relationships among concepts parts are essential to the distribution and tolerance of complexity
Design Quality Naming

Divisibility

1. Stepwise Refinement

5. Modularization
Design Quality Naming

5. Modularization
   1. Stepwise Refinement

Divisibility

2. Cohesion
   cohesion reflects choice
   self-sufficiency: well-formed with cogency

3. Encapsulation
   encapsulation bounds but also interfaces the choice to the surrounding collection
Design Quality Naming

Divisibility

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
5. Modularization

Factorability
Design Quality Naming

**Divisibility**

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. 8. Composition of Function
5. Modularization
6. 10. Scale
7. Factorability

Composable choices enable growth in capacity or complexity; new choices composed from existing ones.

Scale reflects a useful granularity of attention or focus.
Design Quality Naming

Divisibility

Constructibility

Factorability

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Composition of Function
5. Modularization
6. Scale

©LJW, 2010 All Rights Reserved
Design Quality Naming

Divisibility

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
8. Composition of Function
10. Scale
5. Modularization

Constructibility

soundness of individuals, internal stability and structural independence
opportunity of combining to build larger/more capable arrangements

Factorability
Design Quality Naming

Divisibility:
- 1. Stepwise Refinement
- 5. Modularization

Constructibility:
- 2. Cohesion
- 3. Encapsulation
- 8. Composition of Function

Scalability:
- 10. Scale

Factorability:
Design Quality Naming

Divisibility

1. Stepwise Refinement
5. Modularization
8. Composition of Function
10. Scale

Constructibility

2. Cohesion

3. Encapsulation

Scalability

divisible, sound, static structure: building blocks; iteratively decomposed and assembled, separating and insulating concerns

scalability enabling composition, fusing independent self-sufficiency to span the breadth and width of stakeholder intentions
Design Quality Naming

- Divisibility
  - 5. Modularization
    - 8. Composition of Function
    - 10. Scale
  - 1. Stepwise Refinement
- Constructibility
  - 2. Cohesion
    - 3. Encapsulation
- Scalability
- Robustness

Factorability
choice alignment with intentions (dynamically)
a perception of “what is natural,” “seeing” what you expect reinforces reliance and trust
Design Quality Naming

- 1. Stepwise Refinement
- 2. Cohesion
- 3. Encapsulation
- 5. Modularization
- 6. Correctness
- 8. Composition of Function
- 10. Scale
- 11. User Friendliness

Divisibility

Constructibility

Scalability

Robustness

Factorability

Confidence
proven formulae or techniques enable repeatable success
consistently safe practice continues success with compatible challenges
Design Quality Naming

Divisibility

Constructibility

Scalability

Robustness

Factorability

Predictability

Confidence

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Divisibility
5. Modularization
6. Correctness
7. 8. Composition of Function
8. 10. Scale
9. 2. Cohesion
10. 3. Encapsulation
11. 12. Patterns
12. 14. Reliability
13. 11. User Friendliness
14. Reliability

©LJW, 2010 All Rights Reserved
Design Quality Naming

- Divisibility
  - 1. Stepwise Refinement
  - 5. Modularization
- Constructibility
  - 6. Correctness
  - 8. Composition of Function
- Scalability
  - 7. Transparency
  - 10. Scale
- Predictability
  - 12. Patterns
  - 14. Reliability
- Robustness
  - 2. Cohesion
  - 3. Encapsulation
- Factorability
  - 11. User Friendliness
  - 13. Programmability

Keeping “technology” out of the way of intentions enabling users to recognize then apply choices to their needs
Design Quality Naming

- Divisibility
- Constructibility
- Scalability

Robustness
- Factorability

- Correctness
- Transparency
- Identity
- Composition of Function
- Modularization
- Stepwise Refinement
- Cohesion
- Encapsulation
- Scale

Usability
- Programmability
- Elegance
- Patterns
- Identity
- Reliability
- Patterns

Confidence
- User Friendliness
- Correctness
- Reliability

Fusing the conceptual with the linguistic (concise, precise, clear, distinct)
Realizing resonance between choice and intentions like the “ring” in a chord
Design Quality Naming

Divisibility

Constructibility

Scalability

Intuitiveness

Predictability

Robustness

Factorability

Usability

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Extensibility
5. Modularization
6. Correctness
7. Transparency
8. Composition of Function
9. Identity
10. Scale
11. User Friendliness
12. Patterns
13. Programmability
14. Reliability
15. Elegance

achieving confidence in the “now”

enabling alignment into the future to embrace change
Design Quality Naming

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Extensibility
5. Modularization
6. Correctness
7. Transparency
8. Composition of Function
9. Identity
10. Scale
11. User Friendliness
12. Patterns
13. Programmability
14. Reliability
15. Elegance

Divisibility
Constructibility
Scalability
Intuitiveness
Predictability
Robustness
Factorability
Usability
Confidence
Fidelity
usability is the ease with which the user can grasp and apply a choice

intuitiveness expresses the “naturalness” of the choice’s expression of the intention it is supposed to represent (or implement)
Design Quality Naming

- **Divisibility**
  - Modularization
  - Stepwise Refinement
  - Composition of Function

- **Constructibility**
  - Transparency
  - Identity

- **Scalability**
  - Correctness
  - Extensibility

- **Intuitiveness**
  - Usability
  - Predictability

- **Effectiveness**
  - Fidelity
  - Robustness

- **Fidelity**
  - Anchors choices in intentions
  - Predictability fosters choice evolution with anticipated intentions, enabling long-term viability
Design Quality Naming

- Divisibility
- Constructibility
- Scalability
- Intuitiveness
- Effectiveness
- Predictability
- Sustainability
- Robustness
- Factorability
- Usability
- Confidence
- Fidelity

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Extensibility
5. Modularization
6. Correctness
7. Transparency
8. Composition of Function
9. Identity
10. Scale
11. User Friendliness
12. Patterns
13. Programmability
14. Reliability
15. Elegance
Design Quality Naming

Divisibility
- 5. Modularization
- 8. Composition of Function
- 10. Scale

Constructibility
- 1. Stepwise Refinement
- 2. Cohesion
- 3. Encapsulation

Scalability
- 6. Correctness
- 7. Transparency
- 9. Identity

Intuitiveness
- 11. User Friendliness
- 14. Reliability

Effectiveness
- 12. Patterns
- 15. Elegance

Predictability
- 4. Extensibility
- 13. Programmability

Sustainability
- 12. Patterns
- 11. User Friendliness

Robustness
- 4. Extensibility
- 14. Reliability

Factorability
- 4. Extensibility
- 14. Reliability

Fidelity
- 4. Extensibility
- 14. Reliability

Effectively representing intentions: choices that are understandable and applicable.

Choices aligned in the current reality; poised to respond to change; to grow predictably and evolve.
Design Quality Naming

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Extensibility
5. Modularization
6. Correctness
7. Transparency
8. Composition of Function
9. Identity
10. Scale
11. User Friendliness
12. Patterns
13. Programmability
14. Reliability
15. Elegance

- Robustness: soundness of structure, integrity of form, capacity for stability
- Vitality: alignment with stakeholder intentions, capacity for useful growth and unfolding over time

Living Structure
Design Quality Naming

1. Stepwise Refinement
2. Cohesion
3. Encapsulation
4. Extensibility
5. Modularization
6. Correctness
7. Transparency
8. Composition of Function
9. Identity
10. Scale
11. User Friendliness
12. Patterns
13. Programmability
14. Reliability
15. Elegance

Robustness

Thriving

Vitality
Vitruvius
De architectura
78 BC

- 1. Stepwise Refinement
- 2. Cohesion
- 3. Encapsulation
- 4. Extensibility
- 5. Modularization
- 6. Correctness
- 7. Transparency
- 8. Composition of Function
- 9. Identity
- 10. Scale
- 11. User Friendliness
- 12. Patterns
- 13. Programmability
- 14. Reliability
- 15. Elegance

Robustness
Thriving
Vitality
design quality across the system models

implementation

threshold

expectation

mindset
design quality across the system models

implementation threshold expectation mindset

Choice Property-Driven Design Principles
design quality across the system models

Choice Property-Driven Design Principles

implementation
hardware & software architecture

threshold
interfaces

expectation
requirements engineering & specification

mindset
object oriented “systems think”
Pursuing a Universal Foundation of System Design Quality

from the beauty in nature
to the quality in systems

properties of order in nature
Pursuing a Universal Foundation of System Design Quality

from the beauty in nature to the quality in systems

properties of order in nature
Pursuing a Universal Foundation of System Design Quality

from the beauty in nature

properties of order in nature

elaborate
modularize
factor
encapsulate
assemble
focus
align
accommodate
extend
pattern
normalize
expose
generalize
identify
coordinate

Steps of Refinement
Modularization
Cohesion
Encapsulation
Composition of Function
Scale
Correctness
User Friendliness
Extensibility
Patterns
Reliability
Transparency
Programmability
Identity
Elegance

system design actions and choice properties

to the quality in systems
Pursuing a Universal Foundation of System Design Quality

from the beauty in nature

- elaborate
- modularize
- factor
- encapsulate
- assemble
- focus
- align
- accommodate
- extend
- pattern
- normalize
- expose
- generalize
- identify
- coordinate

system design actions and choice properties

to the quality in systems

- Stepwise Refinement
- Modularization
- Cohesion
- Encapsulation
- Composition of Function
- Scale
- Correctness
- User Friendliness
- Extensibility
- Patterns
- Reliability
- Transparency
- Programmability
- Identity
- Elegance

properties of order in nature
Pursuing a Universal Foundation of System Design Quality

from the beauty in nature

elaborate
modularize
factor
encapsulate
assemble
focus
align
accommodate
extend
pattern
normalize
expose
generalize
identify
coordinate

Composition of Function
Encapsulation
Cohesion
Modularization
Stepwise Refinement
Scale
Correctness
User Friendliness
Extensibility
Patterns
Reliability
Transparency
Programmability
Identity
Elegance

Object-oriented ontology

system design actions and choice properties

object-oriented models that thrive
to the quality in systems

properties of order in nature
Pursuing a Universal Foundation of System Design Quality

From the beauty in nature 

elaborate 
modularize 
factor 
encapsulate 
assemble 
focus 
align 
accommodate 
extend 
pattern 
normalize 
expose 
generalize 
identify 
coordinate 

Stepwise Refinement 
Modularization 
Cohesion 
Encapsulation 
Composition of Function 
Scale 
Correctness 
User Friendliness 
Extensibility 
Patterns 
Reliability 
Transparency 
Programmability 
Identity 
Elegance 

System design actions and choice properties 

Object-oriented models that thrive 

Relational data models that thrive 

Agile project management models that thrive 

System design properties of order in nature 

Relational data ontology 
Object-oriented ontology 
Agile vocabulary of SCRUM 

Thayer & Dorfman 1997
Pursuing a Universal Foundation of System Design Quality

from the beauty in nature

elaborate
modularize
factor
encapsulate
assemble
focus
align
accommodate
extend
pattern
normalize
expose
generalize
identify
coordinate

properties of order in nature

system design actions and choice properties

Cohesion
Modularization
Composition of Function
Encapsulation
Scale
Correctness
User Friendliness
Extensibility
Patterns
Reliability
Transparency
Programmability
Identity
Elegance

Object-oriented ontology

Relational data ontology

Agile vocabulary of SCRUM

relational data models that thrive

agile project management models that thrive

Business process modeling vocabulary or ontology?
WALLY, HAVE YOU MADE ANY PROGRESS CODING YOUR MODULE?

PROGRESS IS DIFFICULT TO MEASURE IN THE SOFTWARE REALM.

YOU COULD MEASURE THE LINES OF CODE I PRODUCE, BUT THAT WOULD REWARD INEFFICIENCY.

THE ART OF THIS JOB IS BINDING THE RARE MOMENTS OF INSPIRATION TO KNOWLEDGE AND MACHINES.

IN FACT, JUST A MINUTE AGO I COULD FEEL THE INSPIRATION WELLING UP INSIDE ME.

BUT THEN YOU INTERRUPTED ME WITH YOUR NAIVE QUESTION AND THE MOMENT WAS LOST.

MAYBE YOU SHOULD GO BACK TO YOUR OFFICE AND REFLECT ON THE DAMAGE YOU'VE DONE HERE TODAY.

THERE GOES THE ONE PERSON WHO HAS LESS OF A REAL JOB THAN I DO.
Dogbert is a creativity consultant.

We don't need any of your "intuition" mumbo jumbo. We need quantitative data!

The only way to make decisions is to pull numbers out of the air, call them "assumptions," and calculate the net present value.

Of course, you have to use the right discount rate, otherwise it's meaningless.

Go away.
How is it that one system is more effective, appealing, satisfying and/or more beautiful than another to its stakeholder community? This question drove Christopher Alexander’s fifty-year quest to explain great physical architecture and give birth to pattern languages for building that underpin much of modern systems engineering.

How is it that so many individual stakeholders consistently recognize the same quality, the same beauty in a system? This question led George Lakoff to research the role of conceptual metaphor in human understanding.

What is essential to stakeholders’ satisfaction with systems? Fred Brooks addressed this question in No Silver Bullet: Essence and Accidents of Software Engineering.

This monograph fuses these diverse streams of thought in proposing Thriving Systems Theory by translating Alexander’s properties of physical design quality into the abstract domain of information systems and modeling. Metaphor-Driven Modeling incorporates the theory while examining its impact throughout the system life cycle: modeling, design and deployment. The result is holistic and innovative, a perspective on system quality invaluable to students, practitioners and researchers of software and systems engineering.

Les Waguespack is a computer science Ph.D., professor and chairperson of computer information systems at Bentley University, USA. Dr. Waguespack’s experience as programmer, software engineer, software architect, database architect, project manager and systems consultant underpins 35 years of teaching and research, the last 20+ years teaching object-oriented modeling and systems engineering to undergraduates, graduate students and practicing professionals.