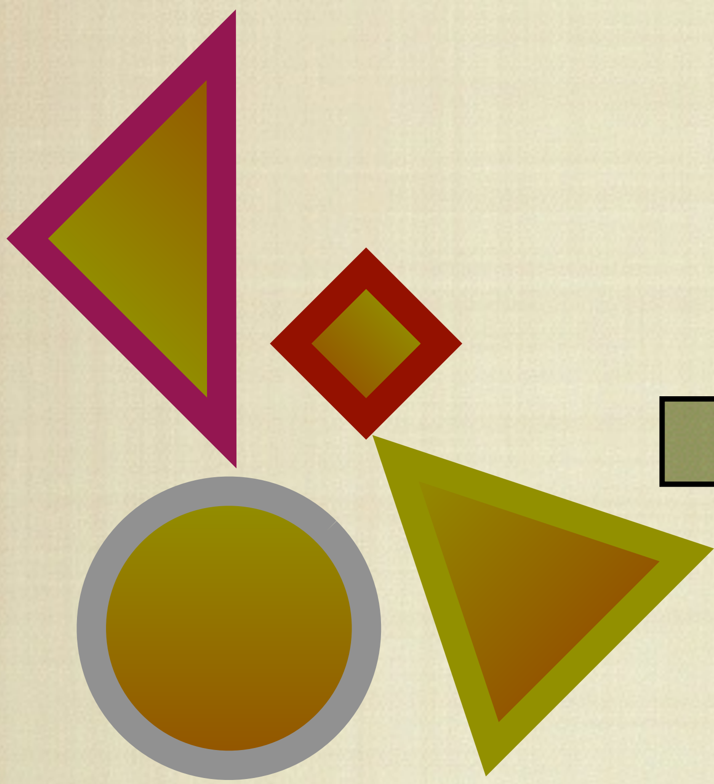
OO modeling choices For Useful models

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

Bentley University
Waltham Massachusetts

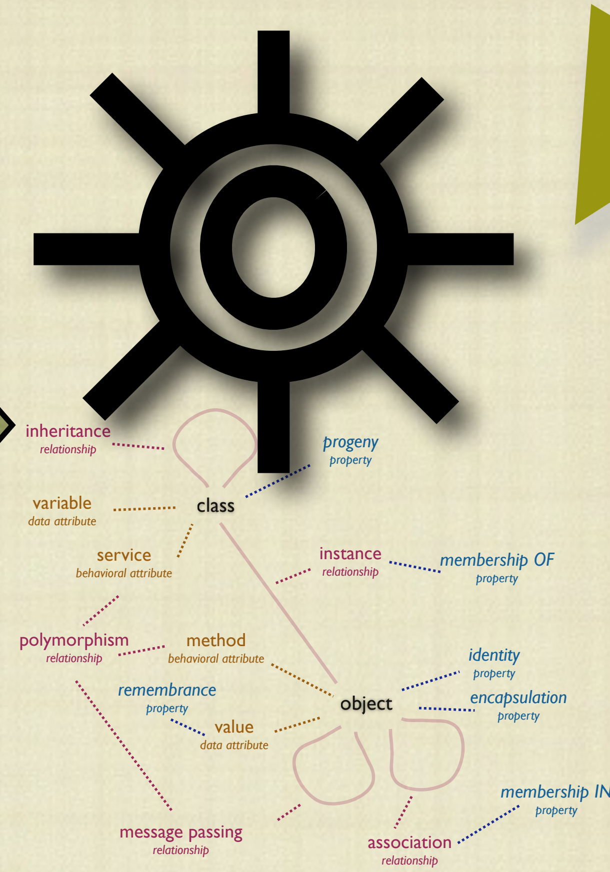
2014-4-23

The Design Process

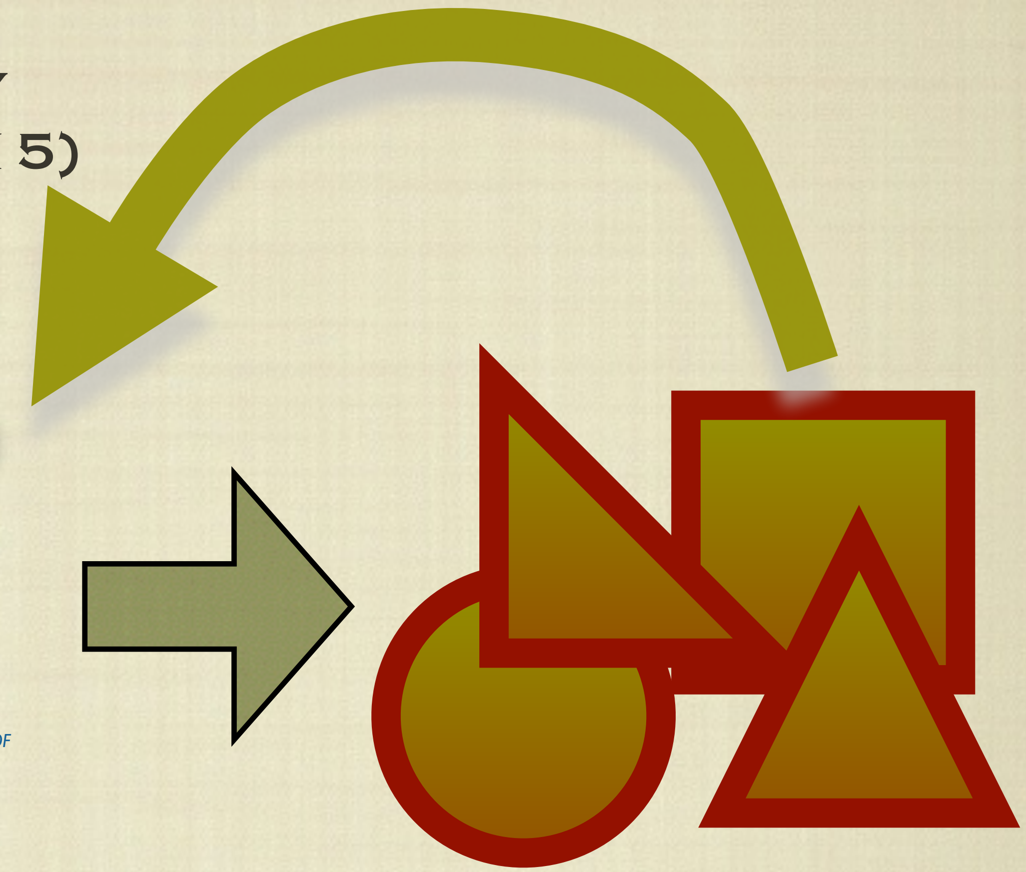


Stakeholder intensions
requirement elements
model elements
design elements

Choice Property
transformations(15)

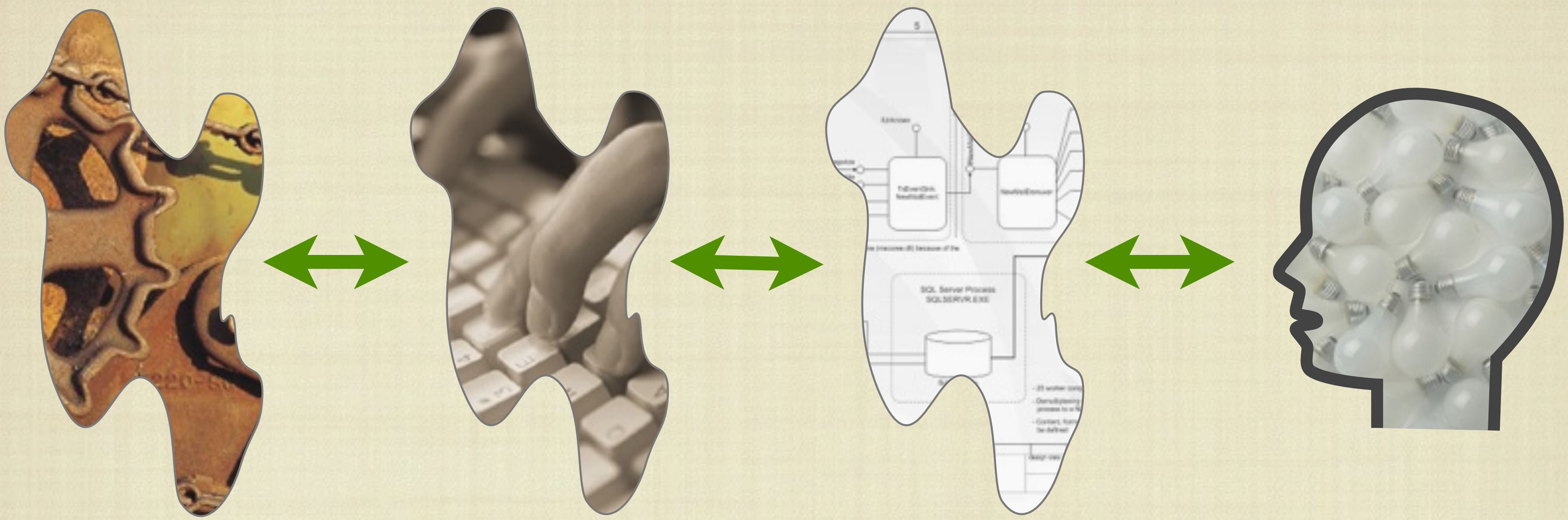


Representational
paradigm or Ontology



Resulting
Design
Elements

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

Deciphering Satisfaction

objective |əbˈjektiv| adjective

(of a person or their judgment) not influenced by personal feelings or opinions in considering and representing facts: historians try to be objective and impartial. Contrasted with subjective.
• not dependent on the mind for existence; actual: a matter of objective fact.

subjective |səbˈjektiv| adjective

based on or influenced by personal feelings, tastes, or opinions: his views are highly subjective | there is always the danger of making a subjective judgment. Contrasted with objective.
• dependent on the mind or on an individual's perception for its existence.

aesthetic |esˈθetik| (also **esthetic**) adjective

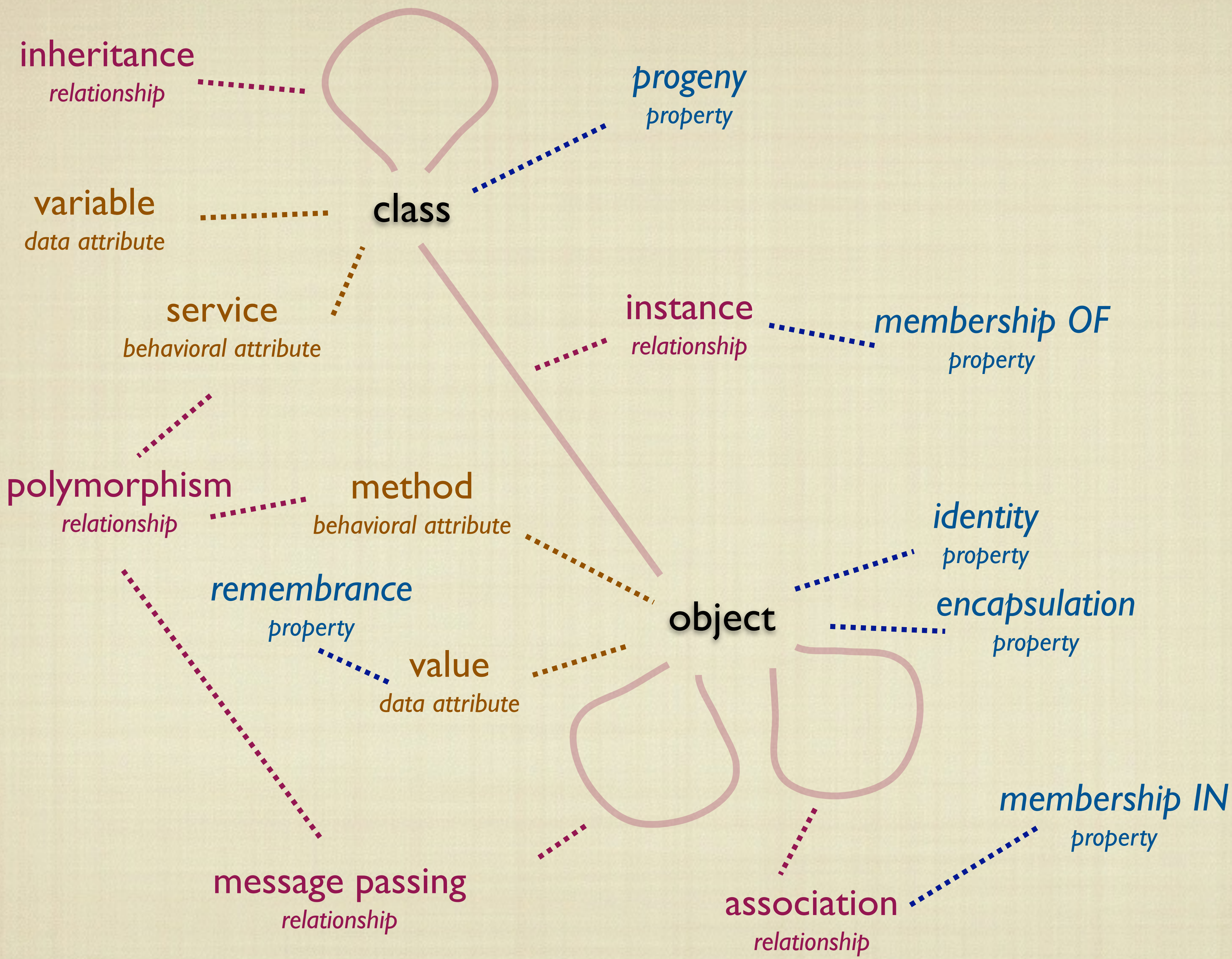
concerned with beauty or the appreciation of beauty : the pictures give great aesthetic pleasure.
• giving or designed to give pleasure through beauty; of pleasing appearance.

What do we "see" in useful design?

Design Choice Properties

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

Object-Oriented Ontology

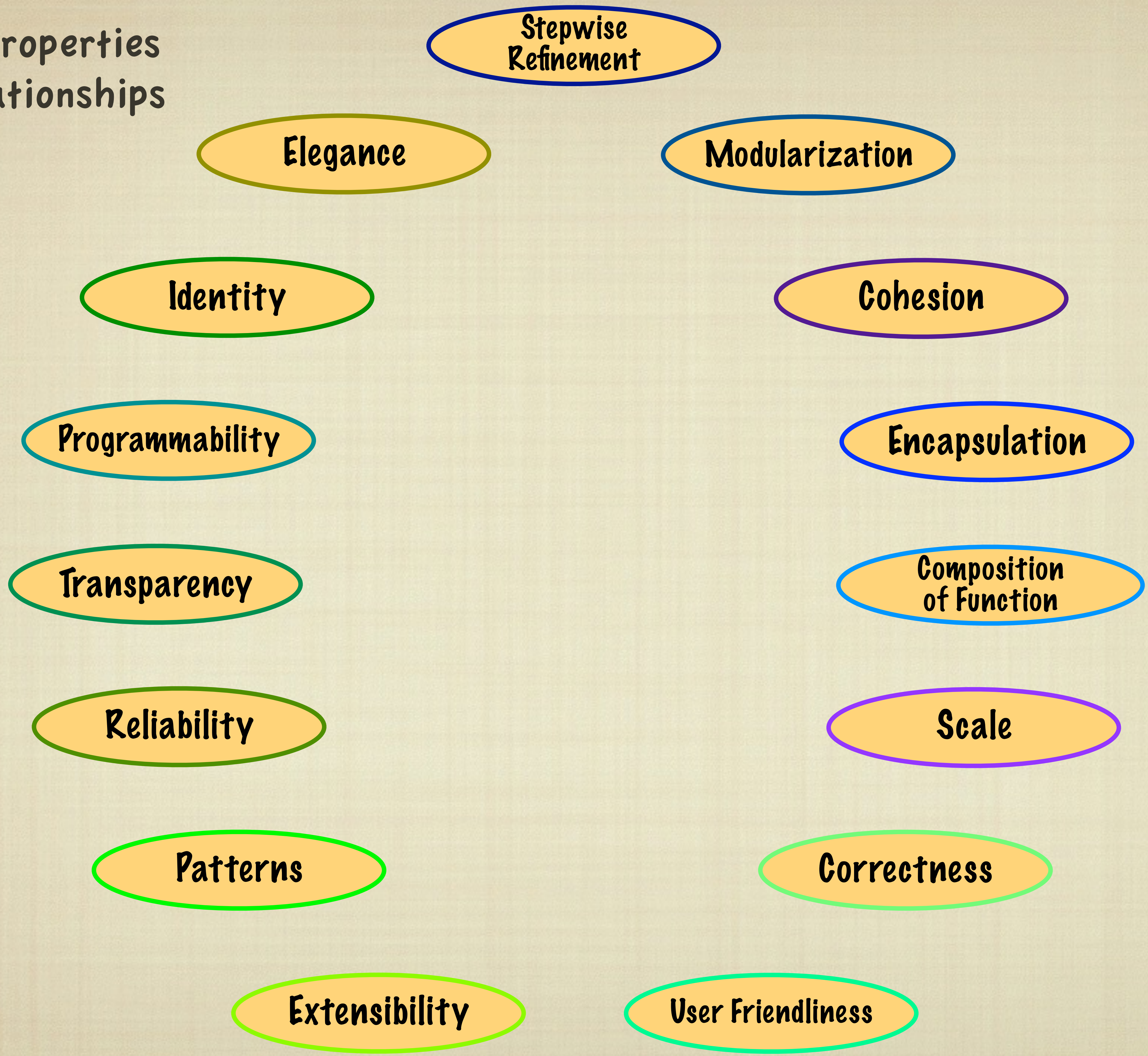


	Choice Property	Modeling Action	Action Rendition
1	Stepwise Refinement	elaborate	develop or present (a theory, policy, or system) in detail
2	Cohesion	factor	express as a product of factors
3	Encapsulation	encapsulate	enclose the essential features of something succinctly by a protective coating or membrane
4	Extensibility	extend	render something capable of expansion in scope, effect, or meaning
5	Modularization	modularize	employing or involving a module or modules as the basis of design or construction
6	Correctness	align	put (things) into correct or appropriate relative positions
7	Transparency	expose	reveal the presence of (a quality or feeling)
8	Composition of Function	assemble	fit together the separate component parts of (a machine or other object)
9	Identity	identify	establish or indicate who or what (someone or something) is
10	Scale	focus	(of a person or their eyes) adapt to the prevailing level of light [abstraction] and become able to see clearly
11	User Friendliness	accommodate	fit in with the wishes or needs of
12	Patterns	pattern	give a regular or intelligible form to
13	Programmability	generalize	make or become more widely or generally applicable
14	Reliability	normalize	make something more normal, which typically means conforming to some regularity or rule
15	Elegance	coordinate	

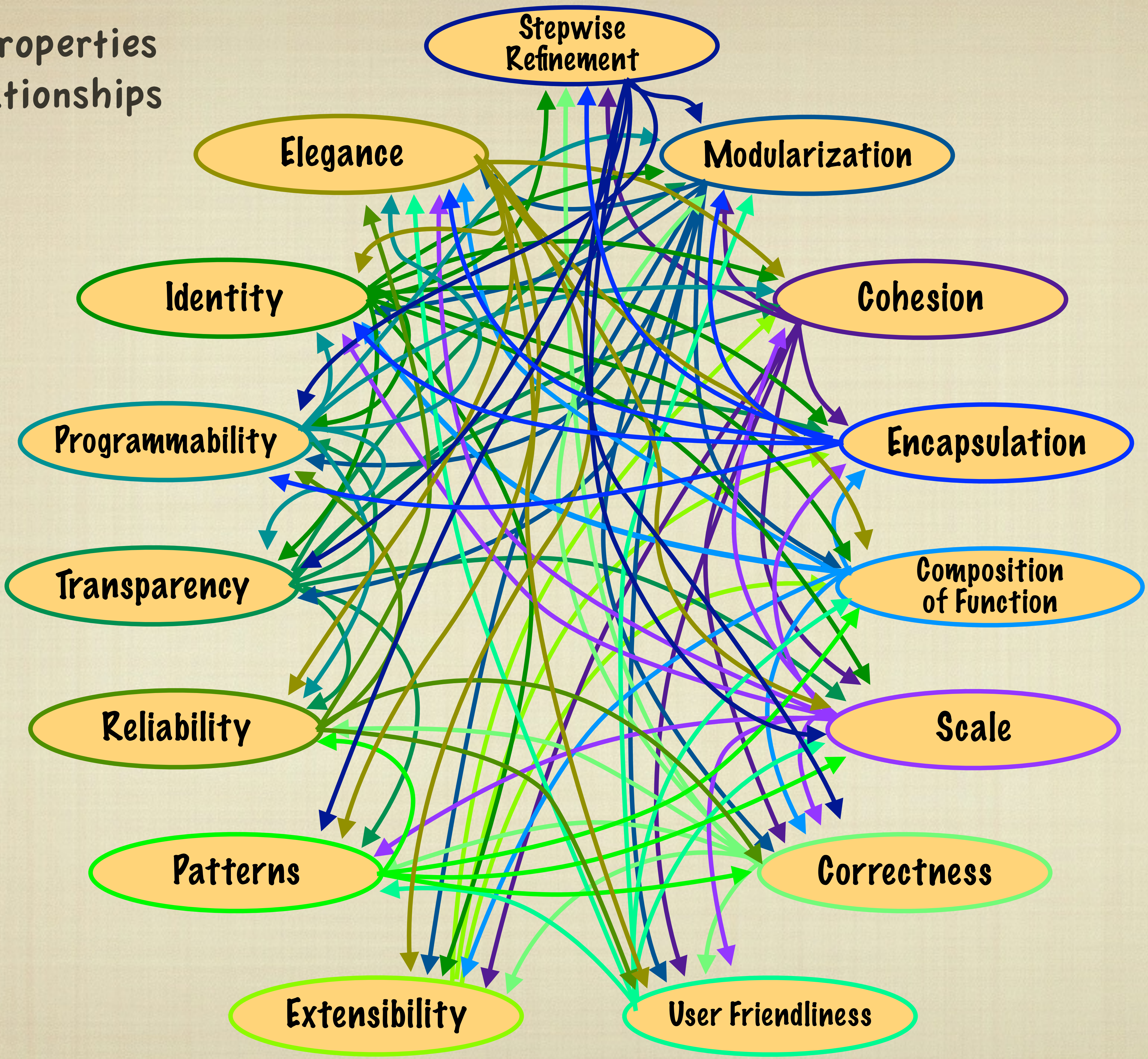
	Choice Property	Modeling Action	Action Rendition
1	Stepwise Refinement	elaborate	develop or present (a theory, policy, or system) in detail
2	Cohesion	factor	express as a product of factors
3	Encapsulation	encapsulate	enclose the essential features of something succinctly by a protective coating or membrane
4	Extensibility	extend	render something capable of expansion in scope, effect, or meaning
5	Modularization	modularize	employing or involving a module or modules as the basis of design or construction
6	Correctness	align	put (things) into correct or appropriate relative positions
7	Transparency	expose	reveal the presence of (a quality or feeling)
8	Composition of Function	assemble	fit together the separate component parts of (a machine or other object)
9	Identity	identify	establish or indicate who or what (someone or something) is
10	Scale	focus	(of a person or their eyes) adapt to the prevailing level of light [abstraction] and become able to see clearly
11	User Friendliness	accommodate	fit in with the wishes or needs of
12	Patterns	pattern	give a regular or intelligible form to
13	Programmability	generalize	make or become more widely or generally applicable
14	Reliability	normalize	make something more normal, which typically means conforming to some regularity or rule
15	Elegance	coordinate	bring the different elements of (a complex activity or organization) into a relationship that will ensure efficiency or harmony

	Choice Property	Modeling Action	Action Rendition Through Object Modeling
1	Stepwise Refinement	elaborate	employing class inheritance to expose and elaborate responsibilities and information management through levels of abstraction
2	Cohesion	factor	locating both service and data attributes for independent sufficiency
3	Encapsulation	encapsulate	delineate the responsibilities, knowledge, and interface of objects
4	Extensibility	extend	service abstractions that enable and control polymorphic extension
5	Modularization	modularize	individual and successive applications of encapsulation to compartmentalize design decisions and abstract system structure
6	Correctness	align	self-validating object interfaces that implement verification behavior
7	Transparency	expose	structural & behavioral relationships the show “fit” and “cooperation”
8	Composition of Function	assemble	design favoring simple parts combined for sophisticated function
9	Identity	identify	stakeholder visible constructs reflected in classes and relationships
10	Scale	focus	grouping objects & relationships in simplifying wrappers and facades
11	User Friendliness	accommodate	using user’s terminology and visible topology to maintain a familiarity that invites users into validation and verification
12	Patterns	pattern	nurturing familiarity and empowering evolution through polymorphism
13	Programmability	generalize	predicting and enabling adaptation of behavior without construction
14	Reliability	normalize	clearly distinguishing essential elements derived from business rules from artifacts necessary for technological compatibility or platform
15	Elegance	coordinate	satisfaction from an intuitively obvious design based not on having nothing else to add, but rather nothing else that can be left out

Choice Properties Interrelationships



Choice Properties Interrelationships

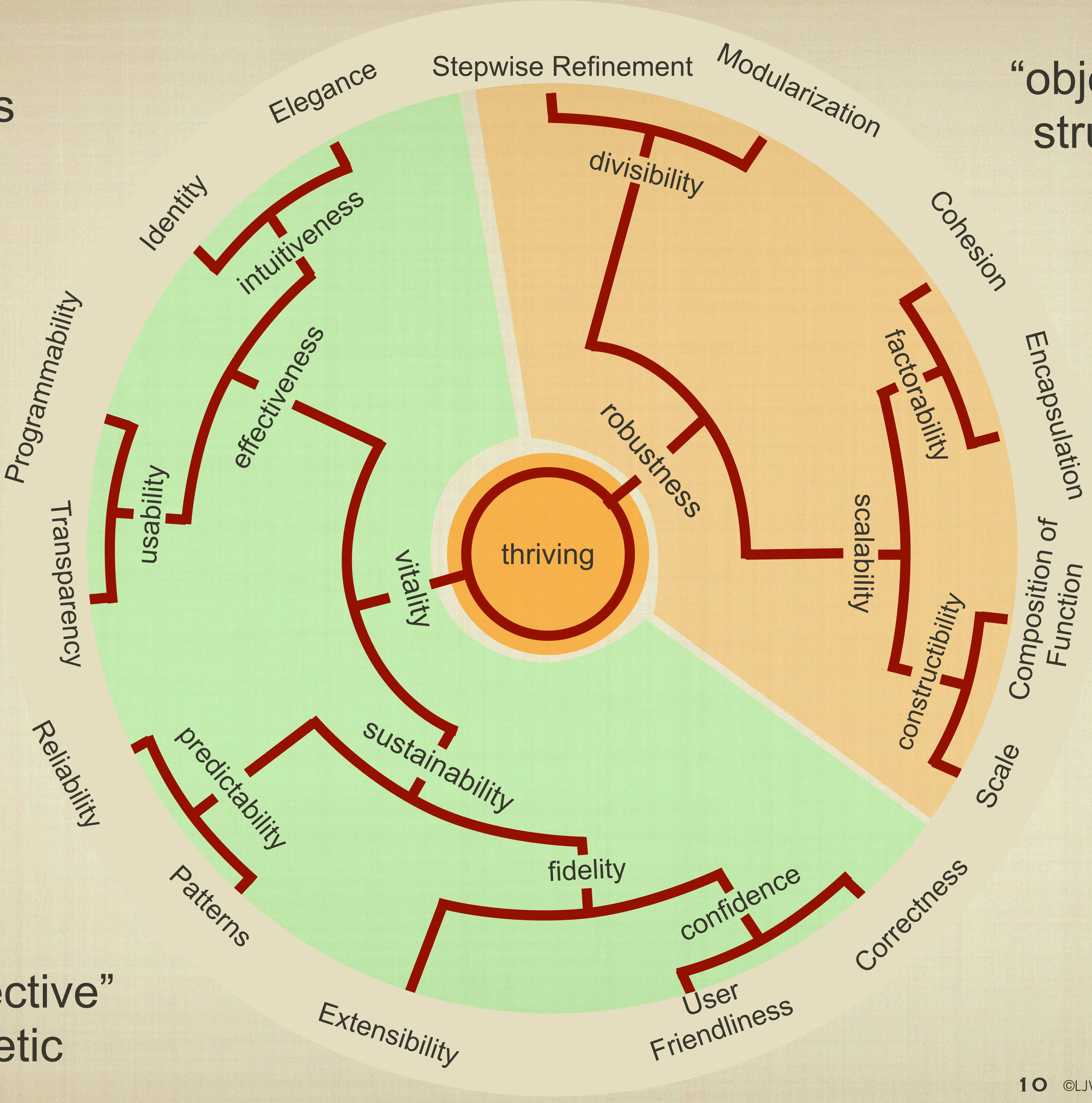


Thriving Systems Theory

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

Design Choice Properties in Confluence Experienced as Design Quality

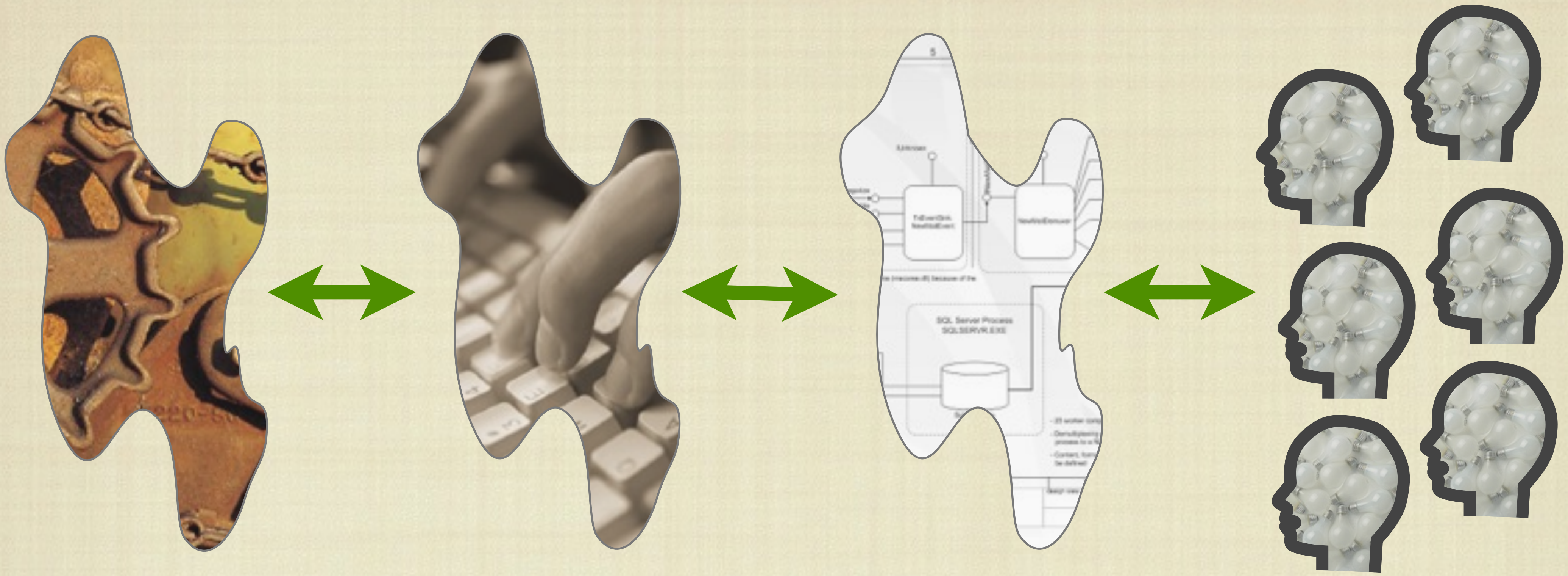
Thriving Systems Theory



A Sense of Great Design

- "Perceive the wholeness and the impact of individual design decisions on the system as a whole – not only in the static present but, in the dynamic unfolding of the stakeholders' perspectives of life; in the system they will live in." (Waguespack)
- Realign the modeling focus. Focus on why to use the tools – not on the tools themselves. Redirect decision-making energy to the question, "How does each decision increase the life in the system by fulfilling the stakeholders' evolving concerns?" and "What does life mean to these stakeholders?"

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


OCTOBER 10, 2010
Boston Sunday Globe
NEW ENGLAND'S LARGEST NEWSPAPER
DILBERT®/ by Scott Adams
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.
DilbertCartoonist@gmail.com
© 2010 Scott Adams, Inc. /Dist. by UFS, Inc.
www.dilbert.com
10-10-10

Waguespack

Thriving Systems Theory and Metaphor-Driven Modeling

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.

ISBN 978-1-84996-301-5

9 781849 963015

❯ springer.com

Waguespack

Thriving Systems Theory and Metaphor-Driven Modeling

Leslie J. Waguespack

Thriving Systems Theory and Metaphor-Driven Modeling

Springer