Georgia Tech Research Institute

Problem. Solved.

Differences in Cognitive Skills Required for Systems Engineering Versus Software Engineering

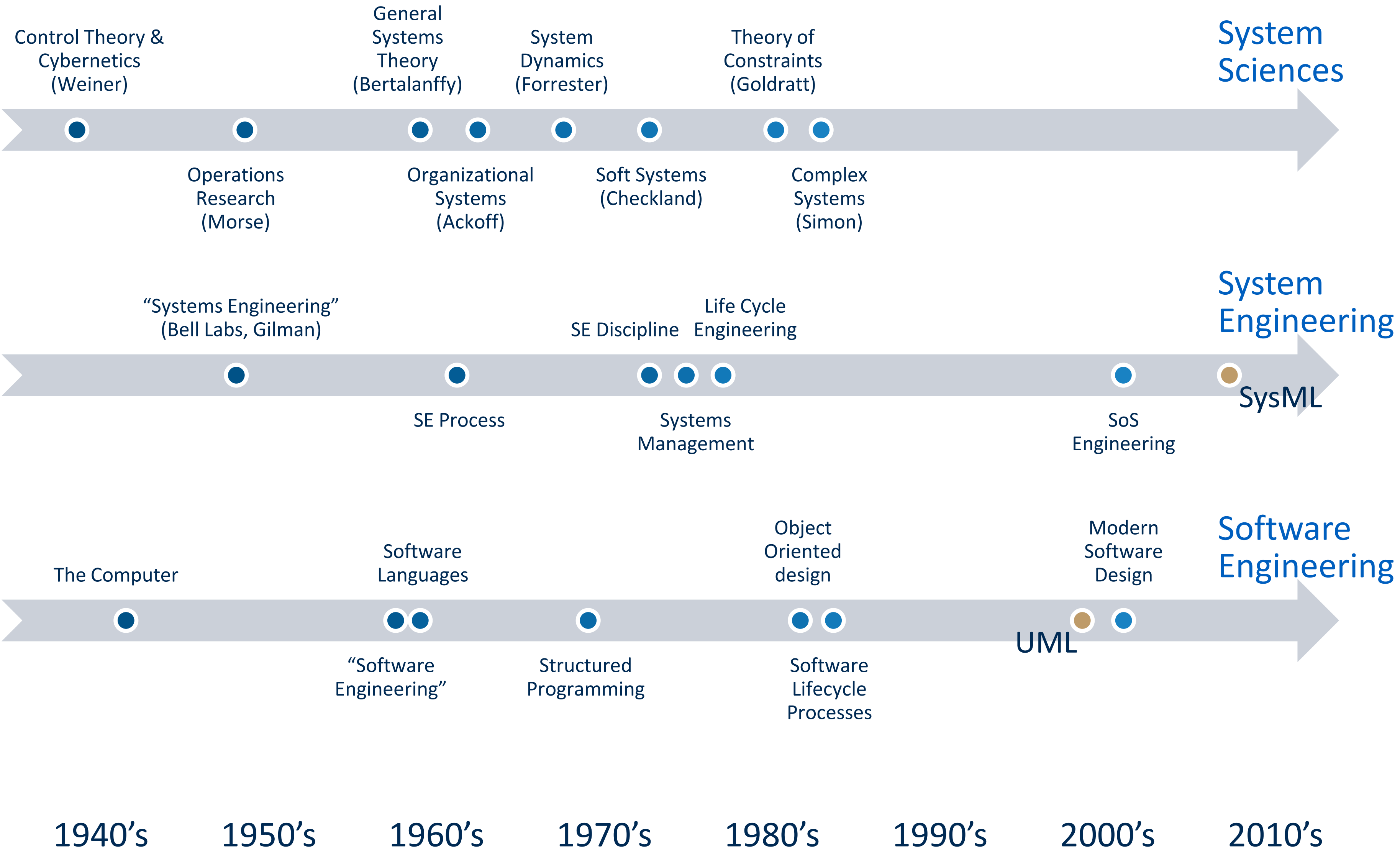
Tom McDermott, Dr. Dennis Folds
Georgia Institute of Technology

October 31, 2013

Premise: Are we adequately preparing systems engineers for today's complexity?

- Georgia Tech Systems Engineering Education Program
  - Several hundred students widely diversified across government, aerospace, and commercial industry
  - Generally all with >5 years experience in engineering fields, 5-30 year breadth of experience levels and roles
- Observation: when dealing with open-ended problems and complex architectures, software trained students are more adept at grasping SE concepts
- Premise: experience and training in SE fields does not emphasize abstract thought, software does better
- Is there a model we can promote, based on role?

General history of SE disciplines



System Sciences

Control Theory & Cybernetics (Weiner)

Operations Research (Morse)

General Systems Theory (Bertalanffy)

Organizational Systems (Ackoff)

System Dynamics (Forrester)

Soft Systems (Checkland)

Theory of Constraints (Goldratt)

Complex Systems (Simon)

System Engineering

“Systems Engineering” (Bell Labs, Gilman)

SE Process

SE Discipline

Life Cycle Engineering

Systems Management

SoS Engineering

SysML

Software Engineering

The Computer

Software Languages

“Software Engineering”

Structured Programming

Object Oriented design

Software Lifecycle Processes

UML

Modern Software Design

1940’s 1950’s 1960’s 1970’s 1980’s 1990’s 2000’s 2010’s

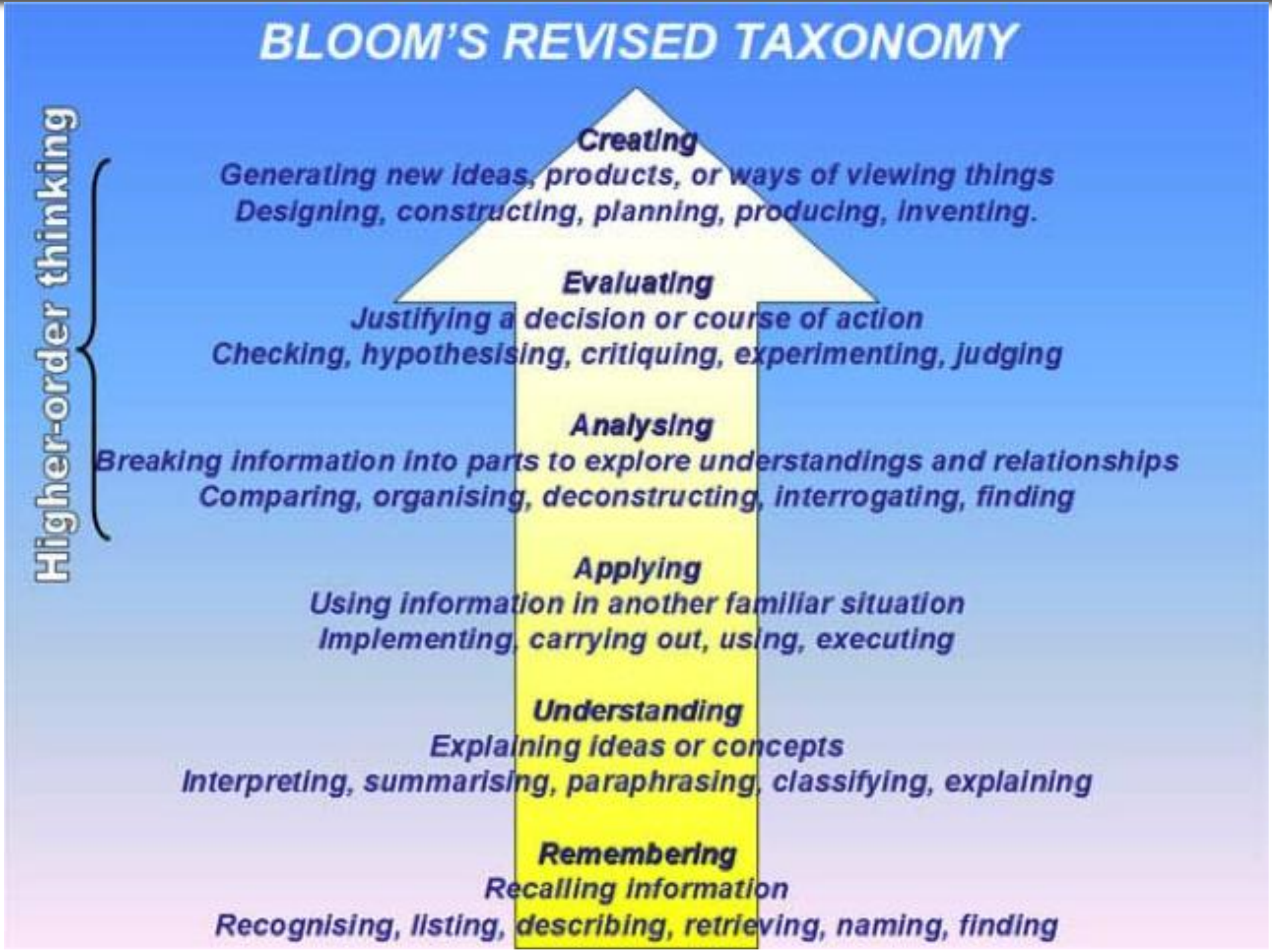
Concrete and Abstract Thinking

- Distinction between concrete and abstract thought predates Psychology as a science
- Many definitions of “abstraction” in the research literature
- Representative definition: the “*process of identifying a set of invariant central characteristics of a thing*” (Burgoon, Henderson, & Markman, 2013)
- Generally thought of as a useful process
  - Marker of cognitive development in children
  - Some disorders manifest in incorrect abstractions

Benefits of Abstraction

- Generalization of acquired knowledge to novel situations (e.g., wayfinding, categorization)
- Anticipation / prediction of future conditions or events
- Basis for creativity and innovation
- But possible errors due to inappropriate stereotypes, extrapolations, etc.

A Continuum, not a Dichotomy



BLOOM'S REVISED TAXONOMY

Higher-order thinking

Creating
Generating new ideas, products, or ways of viewing things
Designing, constructing, planning, producing, inventing.

Evaluating
Justifying a decision or course of action
Checking, hypothesising, critiquing, experimenting, judging

Analysing
Breaking information into parts to explore understandings and relationships
Comparing, organising, deconstructing, interrogating, finding

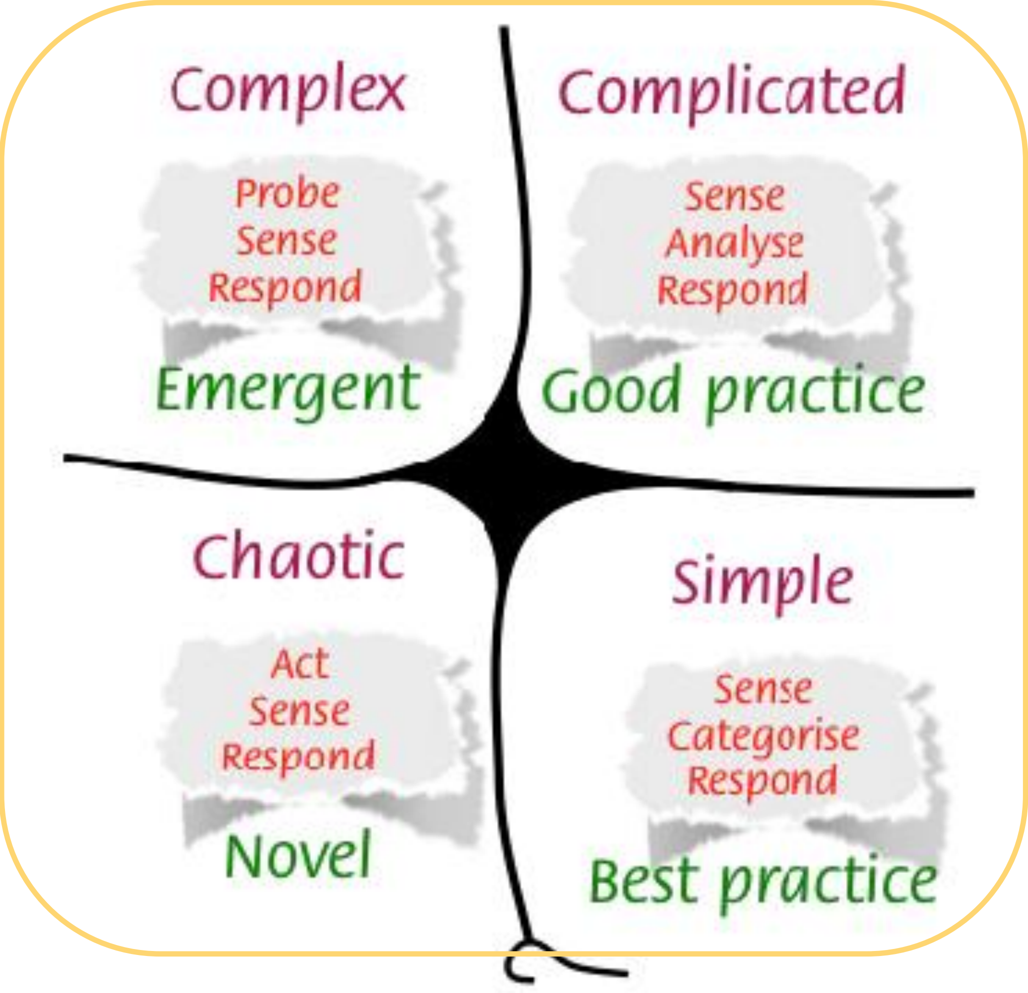
Applying
Using information in another familiar situation
Implementing, carrying out, using, executing

Understanding
Explaining ideas or concepts
Interpreting, summarising, paraphrasing, classifying, explaining

Remembering
Recalling information
Recognising, listing, describing, retrieving, naming, finding

Analysis Framework - Cynefin

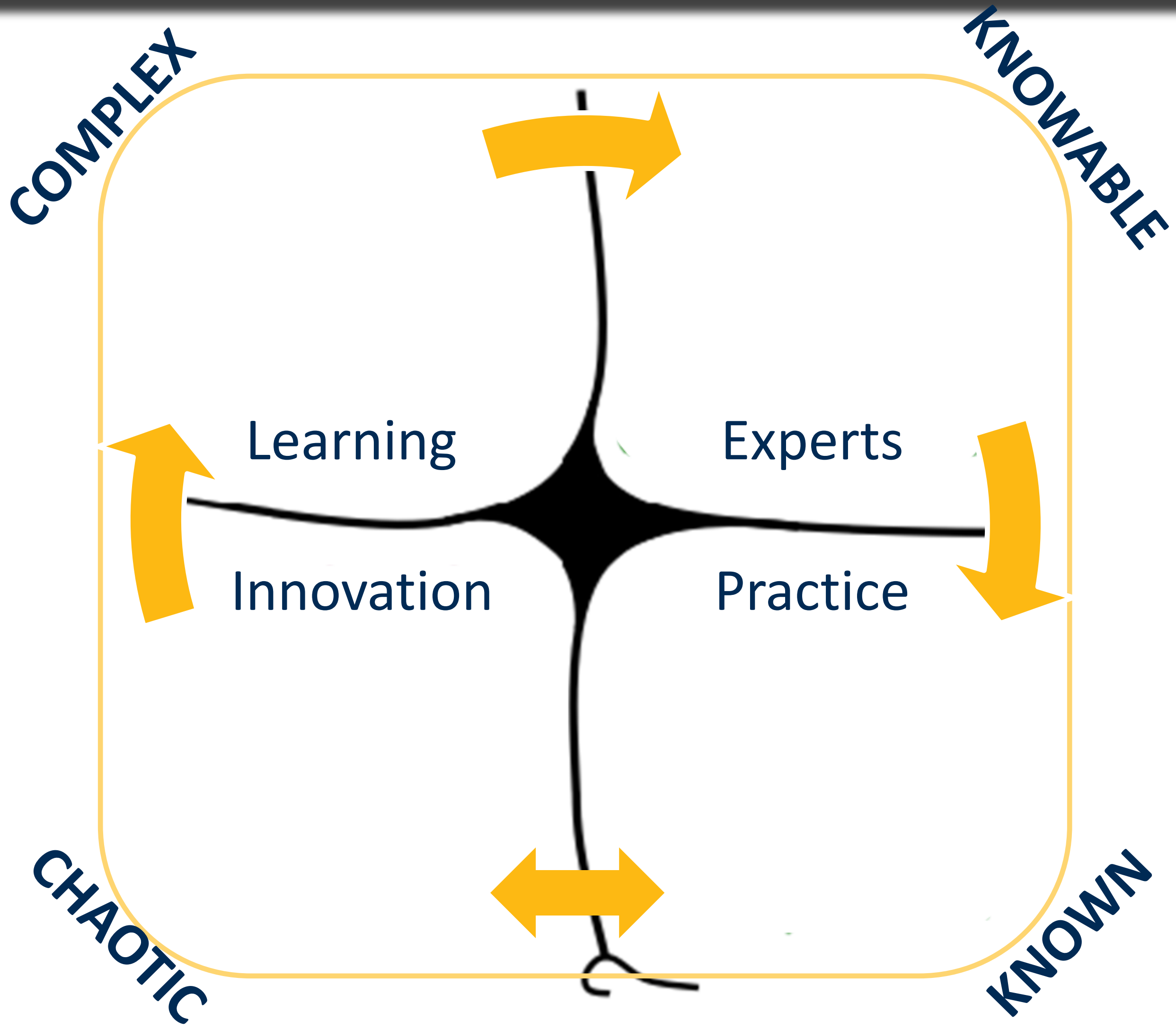
UNORDER



ORDER

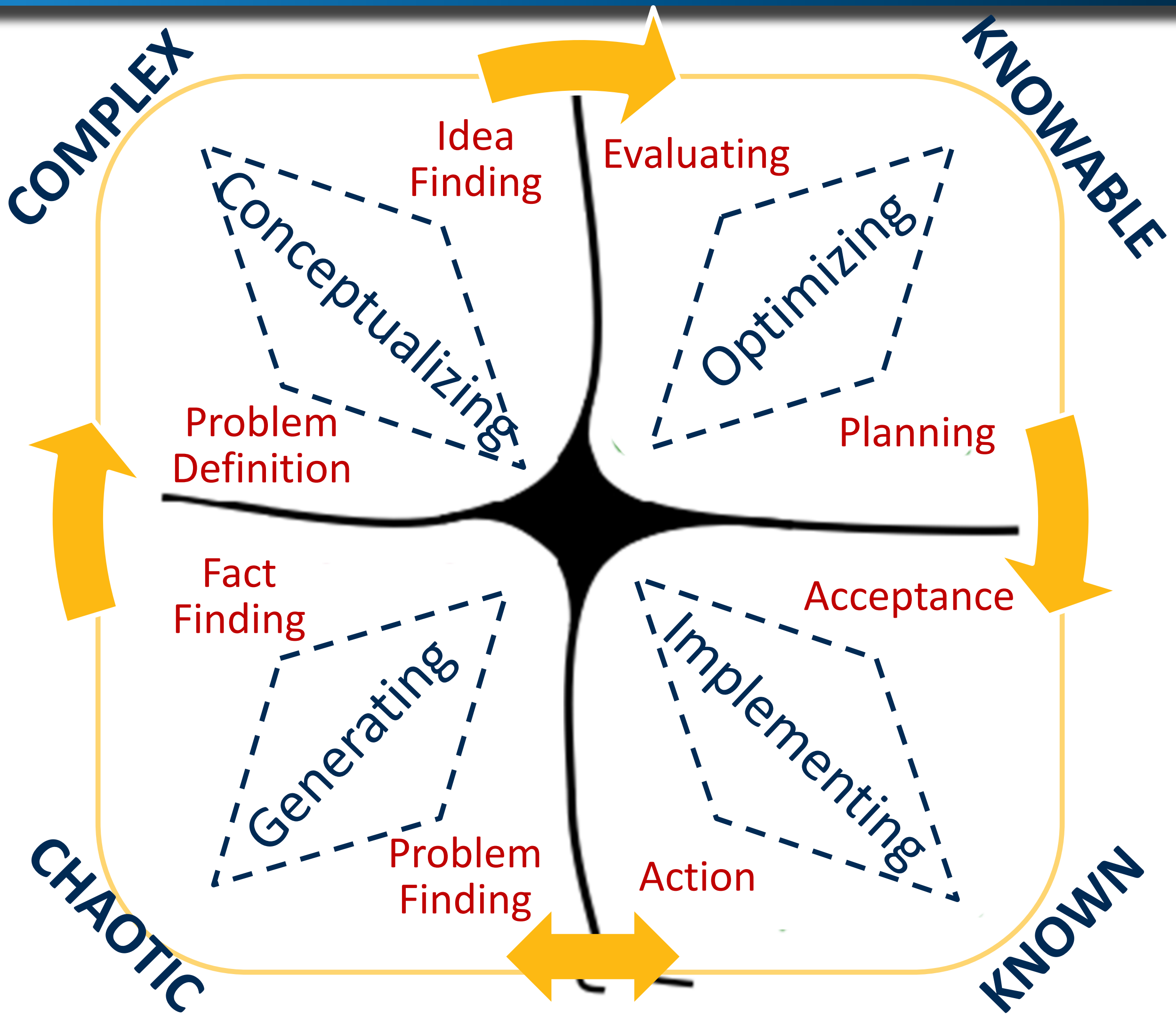
David Snowden, The Origins Of Cynefin, Copyright © 2007-10 Cognitive Edge Pte Ltd. All Rights Reserved.

Cynefin Dynamics



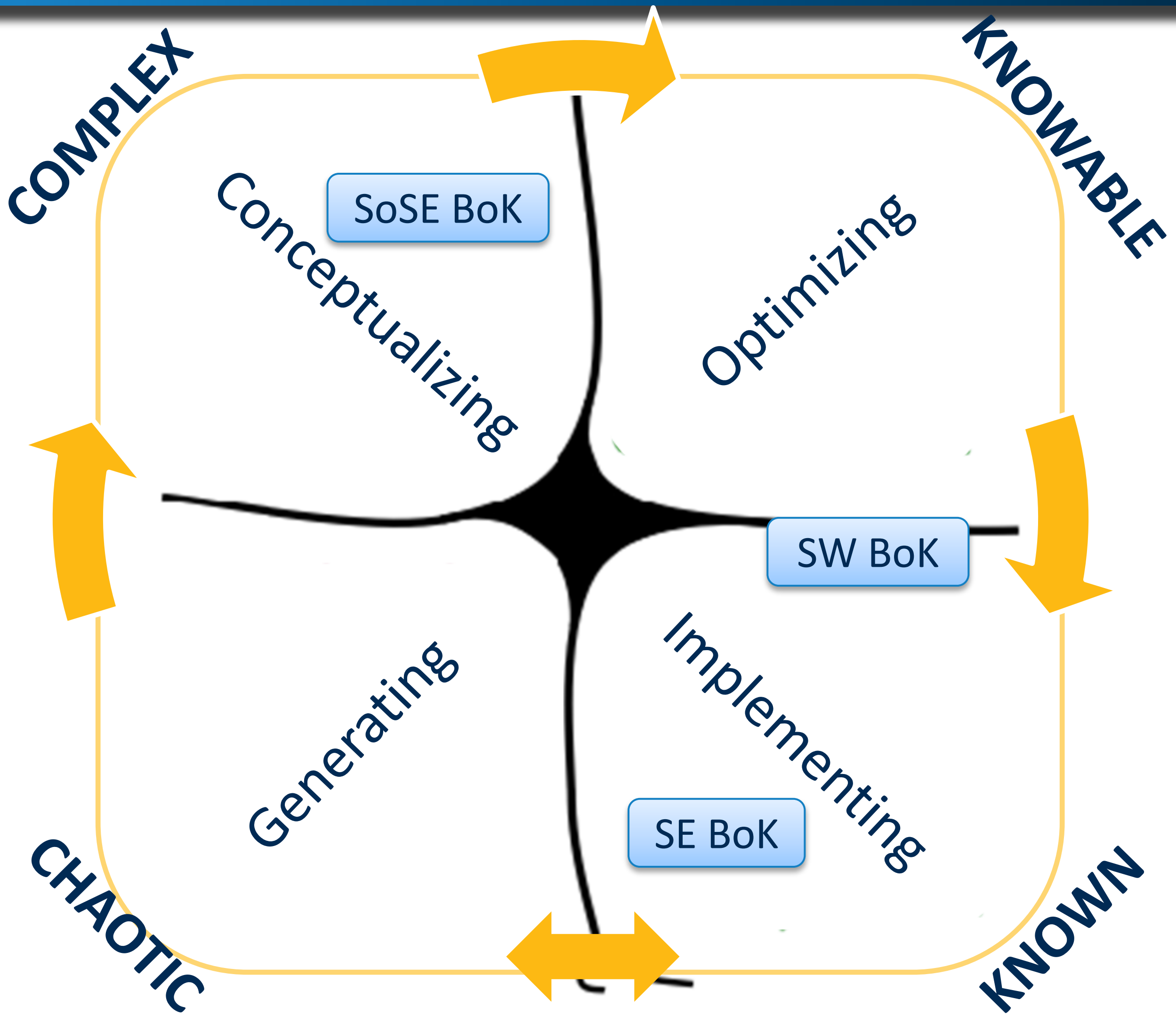
David Snowden and Cynthia Kurtz, The New Dynamics Of Strategy, IBM Systems Journal, Vol. 42, No. 3, 2003.

Divergent & Convergent Thinking



Basadur, M.S., Graen, G.B., and Green, S.G., "Training in Creative Problem Solving: Effects on Ideation and Problem Finding in an Applied Research Organization," Organizational Behavior and Human Performance, 30, 41-70., 1982

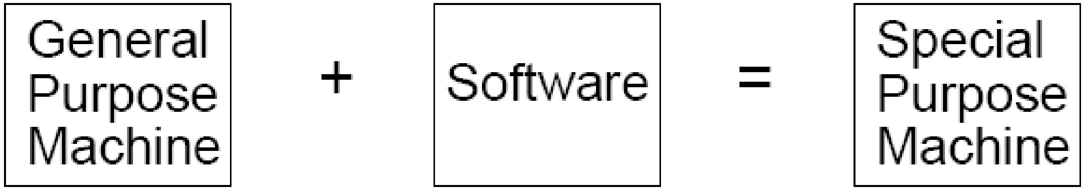
Where is SE?



Basadur, M.S., Graen, G.B., and Green, S.G., "Training in Creative Problem Solving: Effects on Ideation and Problem Finding in an Applied Research Organization," Organizational Behavior and Human Performance, 30, 41-70., 1882

The Software Revolution

- Impossible or impractical machines now feasible
- Design (or Requirements) can be changed without retooling or remanufacturing
- Design is separated from physical realization
  - Design becomes abstract concept
  - Process is separated from other disciplines



General Purpose Machine + Software = Special Purpose Machine

- Heuristics:
  - “Software is never finished”
  - “Software never costs less”

Software's Role in a System

- SCALE - Generation, storage, manipulation and interpretation of large volumes of information
- SIMPLIFICATION - Human interfaces that abstract away the underlying hardware scale
- POLICY - Control of complex, non-linear systems
- AUTOMATION - of operator provided functions
- AUTONOMY - Adaption of the system to the behavior of the environment and users
- ADAPTATION - Customized user capability and experience

Software has Unique Properties

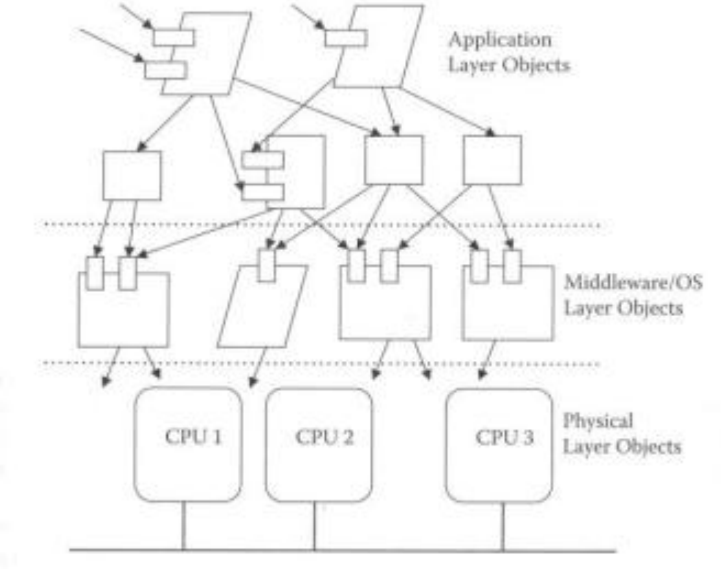
"*The essence of a software entity is a construct of interlocking constructs: data sets, relationships among data items, algorithms, and invocations of functions. This essence is abstract, in that the conceptual construct is the same under many representations. It is nonetheless highly precise and richly detailed*." (Fred Brooks)

- The essence of software, 4 differentiating properties:
  1. Complexity (# of states, lack of repetition)
  2. Conformity (to other man made constructs)
  3. Changeability (emergence, infinite life)
  4. Invisibility (intangible, unvisualizable)

Brooks, F.P., "No Silver Bullet – Essence and Accident in Software Engineering," *Computer*, 20, 4-10., 1987

Traditional Architecting and Hierarchy

- A central tenet of classical systems engineering is that all systems can be viewed in hierarchies
  - A system is composed of subsystems that are composed of smaller units
  - One persons component is another's system
- Object-oriented software construction observes two tenets:
  - Hierarchy – via hierarchical types or modules
  - Abstraction – via abstract types or classes



Case Study – Future Autonomous Systems

- The hardware and software design of such systems sits in the realm of best practices

- The system of systems design will require deep understanding of:
  - Complexity - swarm behaviors
  - Conformity - man-unmanned teaming
  - Changeability - emergence
  - Invisibility – behaviors observable only in usage

SEBOK and SWEBOK

- Software Engineering Body of Knowledge (SWEBOK V3):
  - Computing Foundations
    - Problem solving techniques
    - Algorithms and complexity
    - Abstraction
    - Data structure and representation
  - Software Design and Construction
    - Software structure and architecture
    - Software construction
  - Software Programming
    - Abstraction
    - Information hiding
    - Object-oriented programming

"Great designs come from great designers. Software construction is a creative process." (Fred Brooks)

SEBOK and SWEBOK

- System Engineering Body of Knowledge (SEBOK V1.1):
  - Systems Thinking
    - Problem solving techniques
    - Patterns and complexity
  - System Modeling
    - System modeling concepts =>

- Computing foundations
- Software design & construction
- Software programming

  - System Architecture
    - Logical and physical architecture =>
      - System of systems (complexity)
      - Conops & Scenarios (conformity)
      - Business models (changeability)
      - Views & viewpoints (invisibility)

- Architecture construction: an architecture represents a set of abstracted designs of the system

Systems Engineering Curricula

- Specific recommendations:
  - Systems thinking
    - Case studies, applied throughout the curriculum
    - Capstone projects
  - Systems modeling or Software systems
    - Hierarchy and abstraction
    - Object-oriented design
  - Programming languages
    - SysML, UML
  - Systems architecture
    - Fundamentals
    - Complexity & Systems-of-systems
    - Business and enterprise
    - Evaluation methods

Professional Master's Degree in Applied Systems Engineering

Today's Engineer... Tomorrow's Leader

» Earn your Systems Engineering Master's Degree at Georgia Tech

www.pmase.gatech.edu