Welcome to SENG 480B / CSC 485B / CSC 586B Self-Adaptive and Self-Managing Systems

Dr. Hausi A. Müller
Professor
Department of Computer Science
University of Victoria

http://courses.seng.uvic.ca/courses/2013/summer/seng/480b
http://courses.seng.uvic.ca/courses/2013/summer/csc/485b
http://courses.seng.uvic.ca/courses/2013/summer/csc/586b

Reading Assignments

- ULS Book Section 1-3 on-line at
  - http://www.sei.cmu.edu/uls/the_report.html
- Murray (Ed.): Control in an Information Rich World Report of the Panel on Future Directions in Control, Dynamics, and Systems, SIAM (2003)
  - Chapters 1 & 2
  - http://www.cds.caltech.edu/~murray/cdspanel/report/cdspanel-15aug02.pdf

Ultra-Large-Scale (ULS) Systems

- Premise
  - ULS systems will place an unprecedented demand on software acquisition, production, deployment, management, documentation, usage, and evolution
- Needed
  - A new perspective on how to characterize the problem
  - Breakthrough research in concepts, methods, and tools beyond current hot topics such as SOA (service-oriented architecture) or MDA (model-driven architecture)
- Proposal
  - New solutions involving the intersections of traditional software engineering and other disciplines including fields concerned with people—microeconomics, biology, city planning, anthropology

Evolution of Software Systems

- Legacy systems
- Systems of Systems

Ultra-Large-Scale (ULS) Systems
Socio-Technical Ecosystems

Definitions

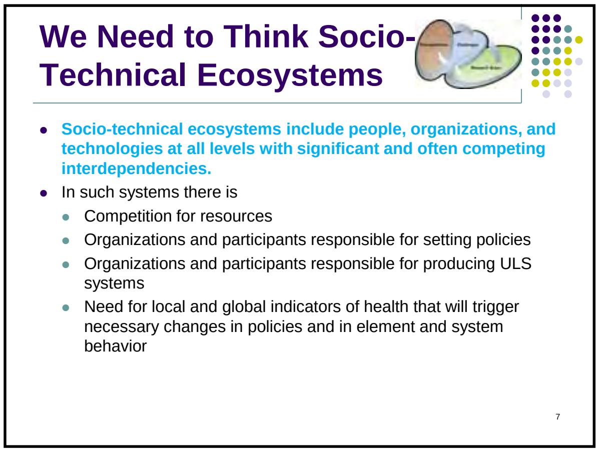
- ***Ecosystem***
  - In biology, an ecosystem is a community of plants, animals, and microorganisms that are linked by energy and nutrient flows interacting with each other and with the physical environment.
  - Rain forests, deserts, coral reefs, grasslands, and a rotting log are all examples of ecosystems
- ***Socio-technical ecosystem***
  - An ecosystem whose elements are groups of people together with their computational and physical environments
  - ULS systems can be characterized as *socio-technical ecosystems*
- ***ULS system***
  - A system whose dimensions are of such a scale that constructing the system using development processes and techniques prevailing at the start of the 21st century is problematic.
  - ULS system characteristics
    - Decentralization
    - Conflicting, unknowable, and diverse requirements
    - Continuous evolution and deployment
    - Heterogeneous and changing element
    - Erosion of the people/system boundary
    - Normal failures of parts of the system

cf. Glossary in ULS Book

From Systems of Systems to Ecosystems

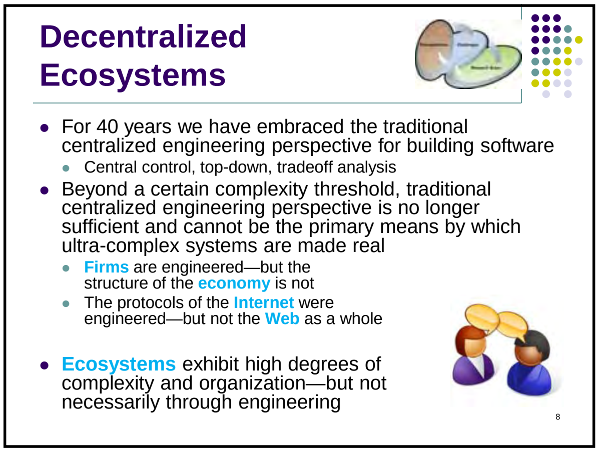
- A ULS system comprises a dynamic community of interdependent and competing organisms in a complex and changing environment
- The concept of an ecosystem connotes complexity, decentralized control, hard-to-predict reactions to disruptions, difficulty of monitoring and assessment

In many ways, legacy systems are already participating in socio-technical ecosystems



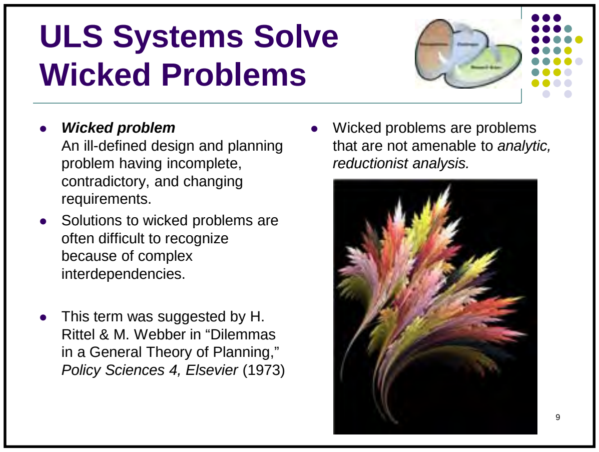
We Need to Think Socio-Technical Ecosystems

- **Socio-technical ecosystems include people, organizations, and technologies at all levels with significant and often competing interdependencies.**
- In such systems there is
  - Competition for resources
  - Organizations and participants responsible for setting policies
  - Organizations and participants responsible for producing ULS systems
  - Need for local and global indicators of health that will trigger necessary changes in policies and in element and system behavior



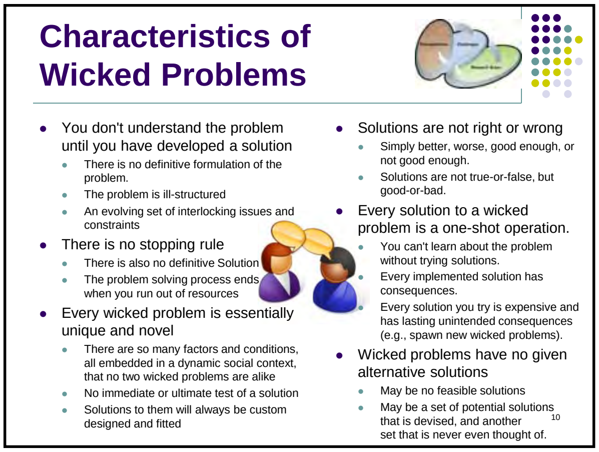
Decentralized Ecosystems

- For 40 years we have embraced the traditional centralized engineering perspective for building software
  - Central control, top-down, tradeoff analysis
- Beyond a certain complexity threshold, traditional centralized engineering perspective is no longer sufficient and cannot be the primary means by which ultra-complex systems are made real
  - **Firms** are engineered—but the structure of the **economy** is not
  - The protocols of the **Internet** were engineered—but not the **Web** as a whole
- **Ecosystems** exhibit high degrees of complexity and organization—but not necessarily through engineering



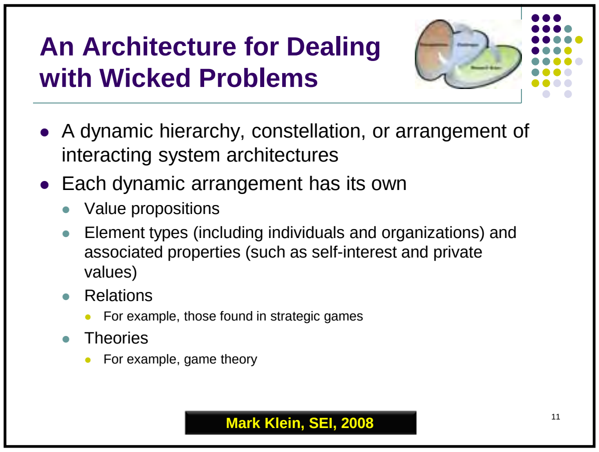
ULS Systems Solve Wicked Problems

- ***Wicked problem***
  An ill-defined design and planning problem having incomplete, contradictory, and changing requirements.
- Solutions to wicked problems are often difficult to recognize because of complex interdependencies.
- This term was suggested by H. Rittel & M. Webber in “Dilemmas in a General Theory of Planning,” *Policy Sciences 4, Elsevier* (1973)
- Wicked problems are problems that are not amenable to *analytic, reductionist analysis.*



Characteristics of Wicked Problems

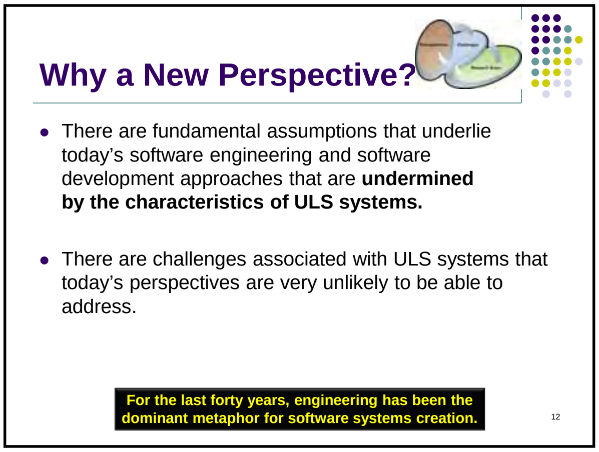
- You don't understand the problem until you have developed a solution
  - There is no definitive formulation of the problem.
  - The problem is ill-structured
  - An evolving set of interlocking issues and constraints
- There is no stopping rule
  - There is also no definitive Solution
  - The problem solving process ends when you run out of resources
- Every wicked problem is essentially unique and novel
  - There are so many factors and conditions, all embedded in a dynamic social context, that no two wicked problems are alike
  - No immediate or ultimate test of a solution
  - Solutions to them will always be custom designed and fitted
- Solutions are not right or wrong
  - Simply better, worse, good enough, or not good enough.
  - Solutions are not true-or-false, but good-or-bad.
- Every solution to a wicked problem is a one-shot operation.
  - You can't learn about the problem without trying solutions.
  - Every implemented solution has consequences.
  - Every solution you try is expensive and has lasting unintended consequences (e.g., spawn new wicked problems).
- Wicked problems have no given alternative solutions
  - May be no feasible solutions
  - May be a set of potential solutions that is devised, and another set that is never even thought of.



An Architecture for Dealing with Wicked Problems

- A dynamic hierarchy, constellation, or arrangement of interacting system architectures
- Each dynamic arrangement has its own
  - Value propositions
  - Element types (including individuals and organizations) and associated properties (such as self-interest and private values)
  - Relations
    - For example, those found in strategic games
  - Theories
    - For example, game theory

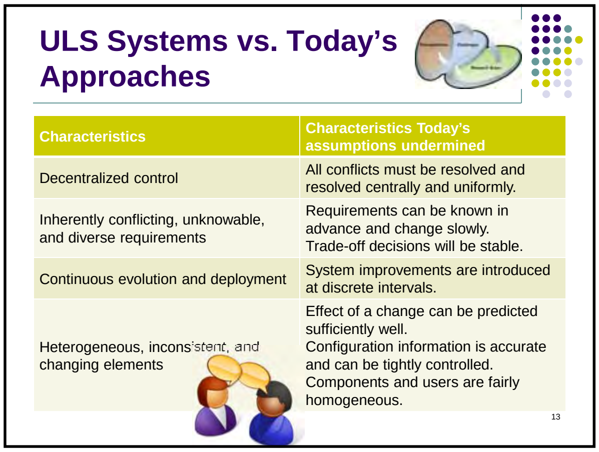
Mark Klein, SEI, 2008



Why a New Perspective?

- There are fundamental assumptions that underlie today’s software engineering and software development approaches that are **undermined by the characteristics of ULS systems.**
- There are challenges associated with ULS systems that today’s perspectives are very unlikely to be able to address.

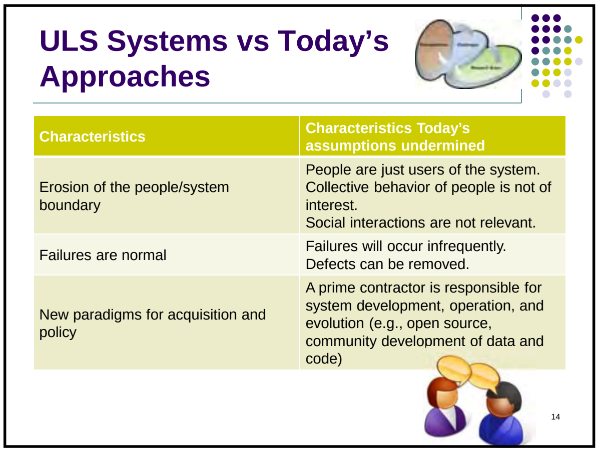
For the last forty years, engineering has been the dominant metaphor for software systems creation.



ULS Systems vs. Today's Approaches

Characteristics	Characteristics Today's assumptions undermined
Decentralized control	All conflicts must be resolved and resolved centrally and uniformly.
Inherently conflicting, unknowable, and diverse requirements	Requirements can be known in advance and change slowly. Trade-off decisions will be stable.
Continuous evolution and deployment	System improvements are introduced at discrete intervals.
Heterogeneous, inconsistent, and changing elements	Effect of a change can be predicted sufficiently well. Configuration information is accurate and can be tightly controlled. Components and users are fairly homogeneous.

13



ULS Systems vs Today's Approaches

Characteristics	Characteristics Today's assumptions undermined
Erosion of the people/system boundary	People are just users of the system. Collective behavior of people is not of interest. Social interactions are not relevant.
Failures are normal	Failures will occur infrequently. Defects can be removed.
New paradigms for acquisition and policy	A prime contractor is responsible for system development, operation, and evolution (e.g., open source, community development of data and code)

14