THE ARCHITECTURE OF COMPLEXITY REVISITED: DESIGN PRINCIPLES FOR ULTRA-LARGE-SCALE SYSTEMS

RICK KAZMAN

UNIVERSITY OF HAWAII AND

SOFTWARE ENGINEERING INSTITUTE/CMU

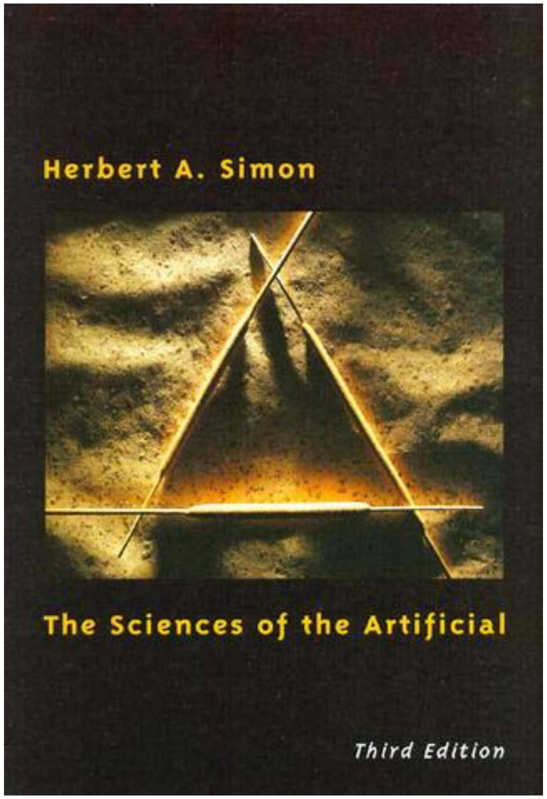
THE ARCHITECTURE OF COMPLEXITY

Simon, 1962: "complexity frequently takes the form of hierarchy"

- complex systems are organized as hierarchies with stable sub-assemblies

- sub-systems are "nearly decomposable"

- e.g. social systems, biological systems

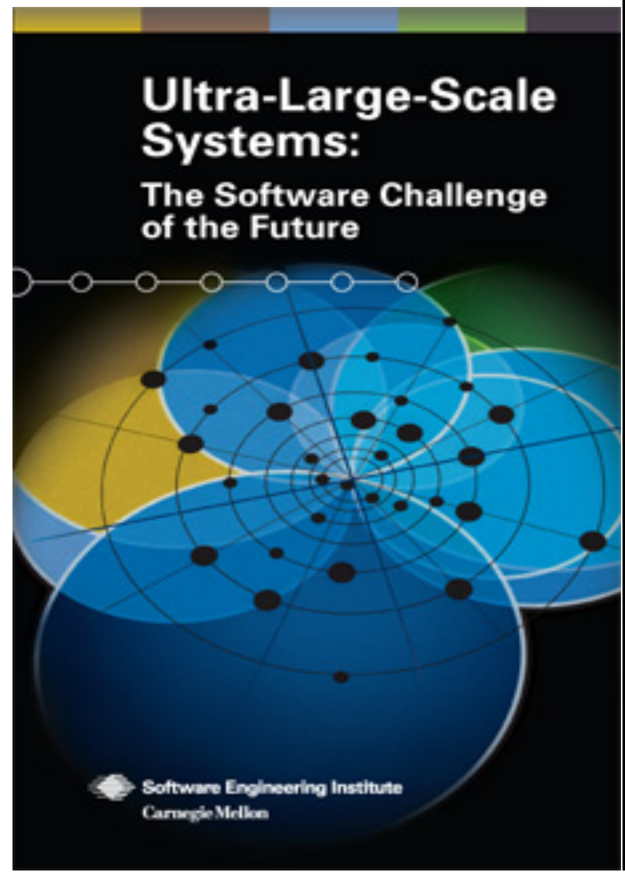


ULS SYSTEMS

ULS Report, Northrop et al, 2006:

ULS systems are "socio-technical ecosystems"

- **such systems are undoubtedly hierarchical and nearly decomposable**
- **but their properties do not come from hierarchy and near decomposability alone**



ULS SYSTEM CHARACTERISTICS

1. decentralization
2. inherently conflicting, unknowable, and diverse requirements
3. continuous evolution and deployment
4. heterogeneous, inconsistent, and changing elements
5. erosion of the people/system boundary
6. normal failures, and
7. new paradigms for acquisition and policy

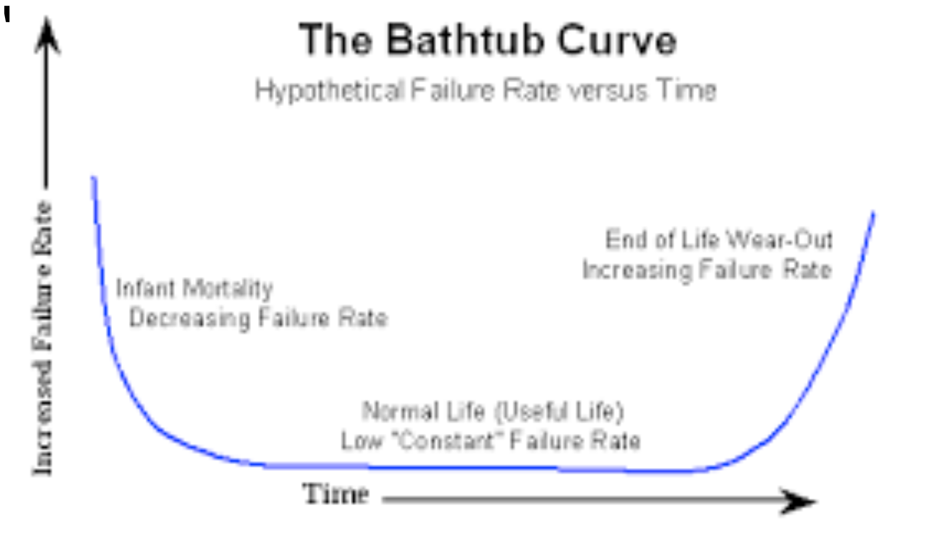
ULS SYSTEMS ARE DIFFERENT

ULS characteristics can be seen in systems of conventional scale, e.g. normal failures, heterogeneity, …

but in ULS systems, they *dominate*

Hence ULS systems can not be built in ways that have previously sufficed.

"Scale changes everything"



RESEARCH QUESTION

Given that ULS systems are different, what are the underlying principles for their construction?

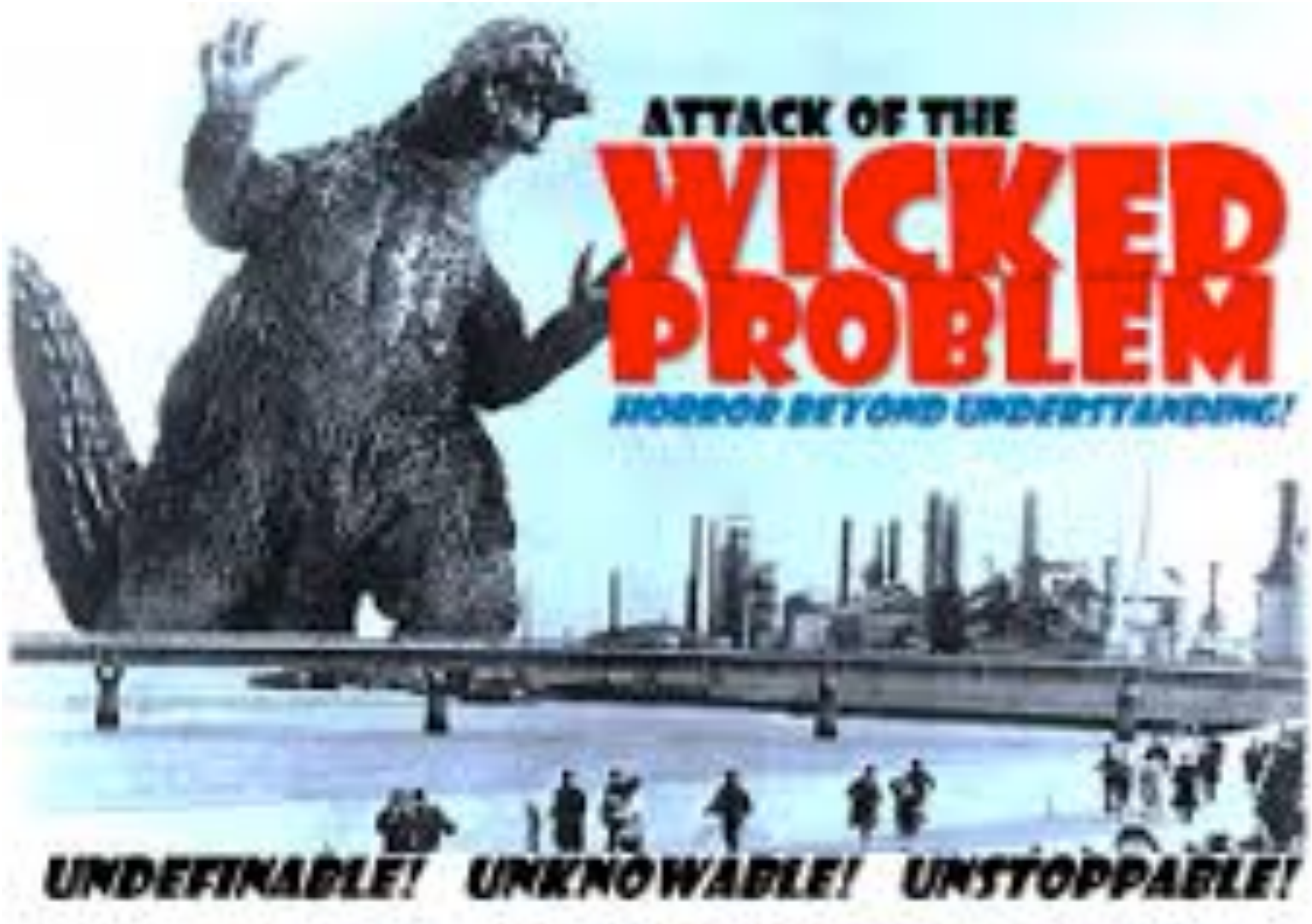
WICKED PROBLEMS

ULS systems are "wicked" problems.

Wicked problems possess several unique characteristics:

- stakeholders do not all agree on the problem to be solved—requirements are vague and unstable
- solutions are not right or wrong, they are better or worse
- enormous complexity, both among the subcomponents and between the "problem" and the world; and any solution may *change* the problem
- they have no single objective measure of success

WICKED PROBLEMS



RESEARCH METHODOLOGY

Examine systems that exhibit ULS properties

- **P2P systems (such as Skype and its predecessor KaZaA),**
- **the internet,**
- **the public switched telephone network (PSTN),**
- **biological systems,**
- **…**

and attempt to distill the *architectural principles* behind them.

FOLLOWING SIMON'S FOOTSTEPS

Architectures for ULS must embody three principles:

- *peer to peer structure*: dependence upon centralized resources must be avoided.

- *local assemblies of components*: complex systems contain local assemblies of a "small" number of components that interact weakly with other assemblies.

- *hierarchical structure*: Simon's key organizing principle of complex systems.

RAW DATA

Let us now examine the principles behind:

- **P2P systems**
- **biological systems**
- **the internet**
- **the PSTN**

PRINCIPLES FROM KAZAA

Distributed Design

Exploiting Heterogeneity

Load Balancing

Locality in Neighbor Selection

Connection shuffling

Efficient gossiping algorithms

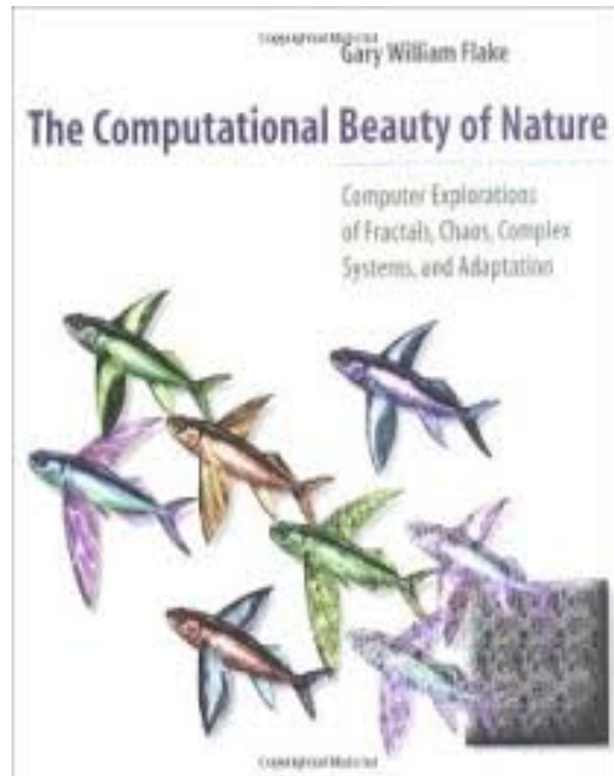
PRINCIPLES FROM BIOLOGICAL SYSTEMS

Flake's attributes of agents in complex systems:

- Collections, Multiplicity, Parallelism

- Iteration, Recursion, Feedback

- Adaptation, Learning, Evolution



PRINCIPLES FROM BIOLOGICAL SYSTEMS

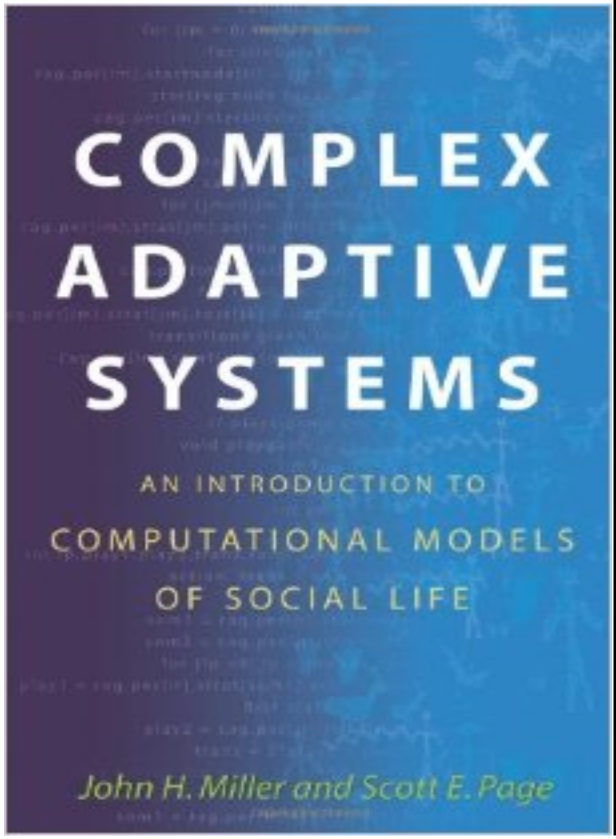
Holland defined characteristics and mechanisms of complex adaptive systems.

Characteristics:

- *Aggregation*
- *Non-linearity*
- *Flows*
- *Diversity*

Mechanisms:

- *Tagging*
- *Internal Models*
- *Building Blocks*



ANT COLONY OPTIMIZATION

A computational technique for searching—finding paths through graphs—inspired by the behavior of ant colonies.

ACO simulates this behavior, by not only following existing paths but also randomly trying new paths and leaving computational "pheromones" that decay over time.

PRINCIPLES FROM THE INTERNET

Arguably the most successful (man-made) ULS.

IETF design principles for the internet:

- One and only one protocol

- End-to-end functions realized by end-to-end protocols

- Heterogeneity

- Scale-free design

- Modularity

- Send/receive asymmetry

- Self-description

PRINCIPLES FROM THE PSTN

The PSTN successfully handles hundreds of millions of concurrent customers with switches that experience no more than 2 hours of failure over 40 year lifespans.

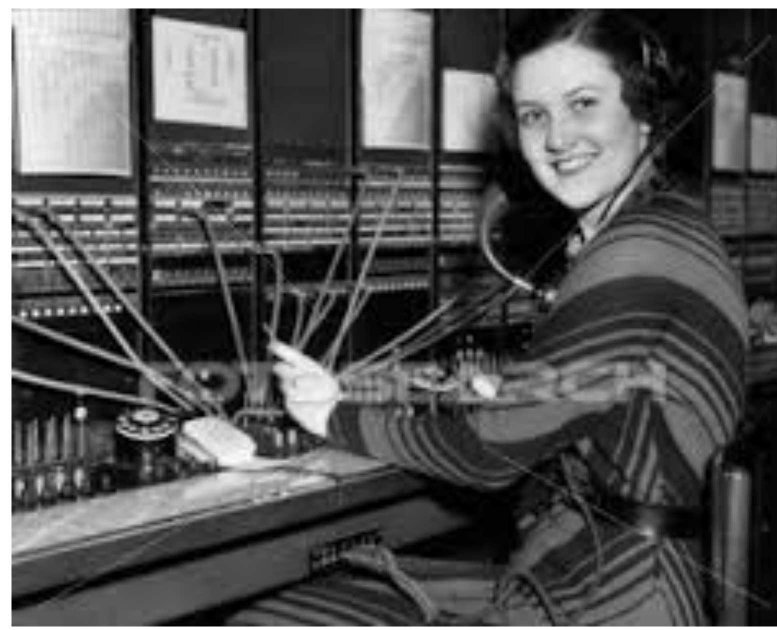
Properties that lead to its robustness:

- Reliable software

- Dynamic rerouting

- Loose coupling

- Human intervention



REFLECTION

From these principles we can now attempt to triangulate—determine their commonalities.

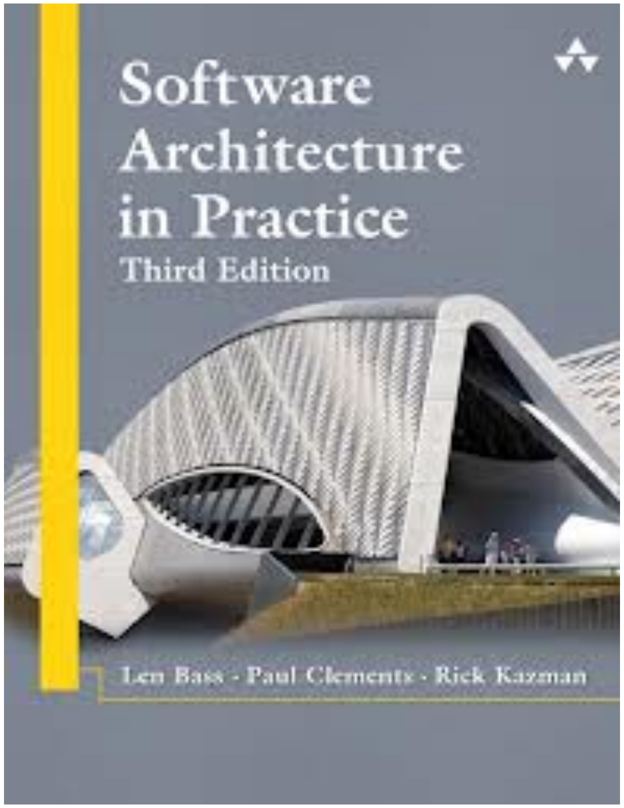
The goal: to generate a set of *tactics*.

TACTICS

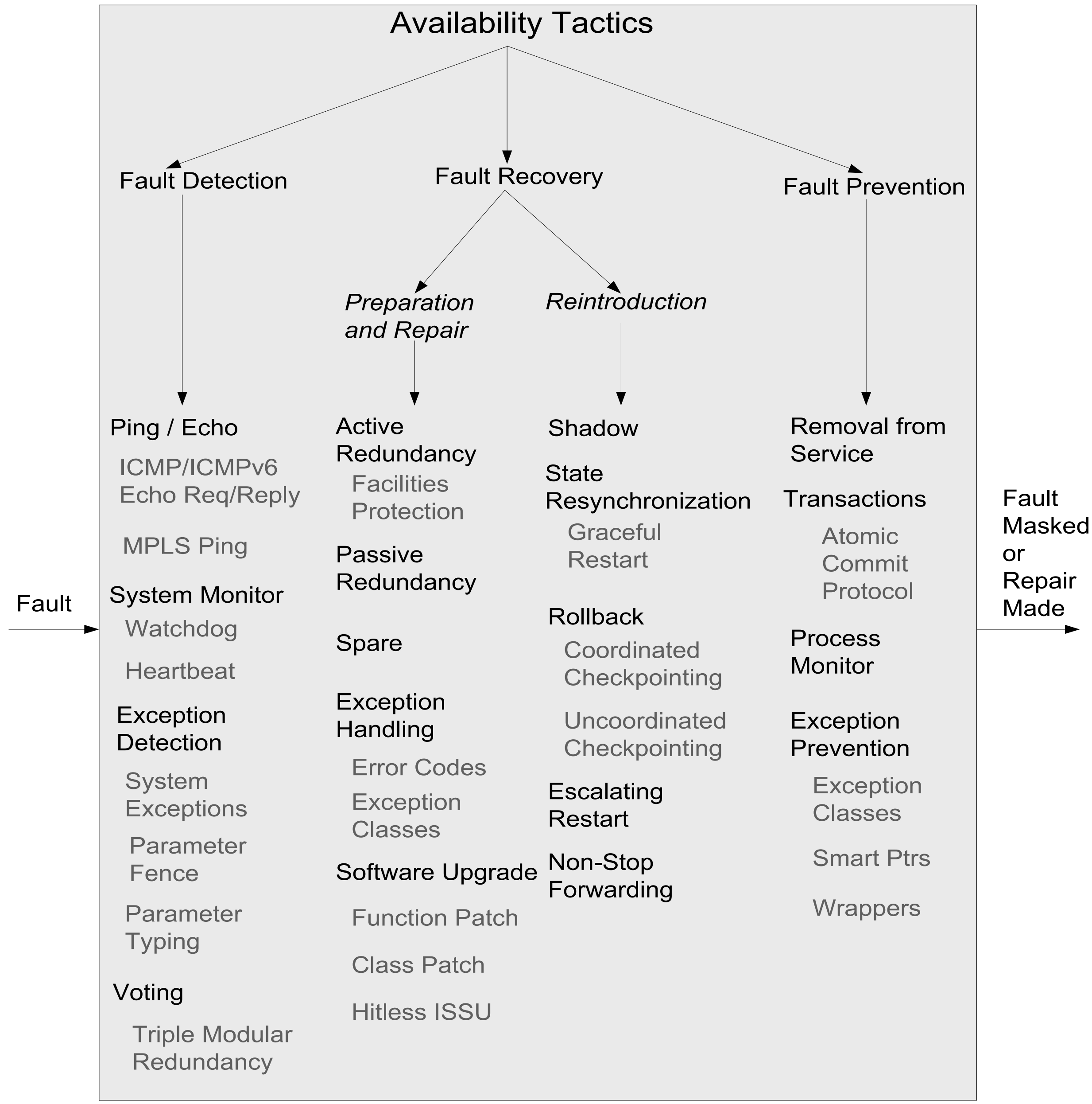
Tactics are the building-blocks of architecture.

A tactic is a design decision that is influential in the control of a quality attribute response.

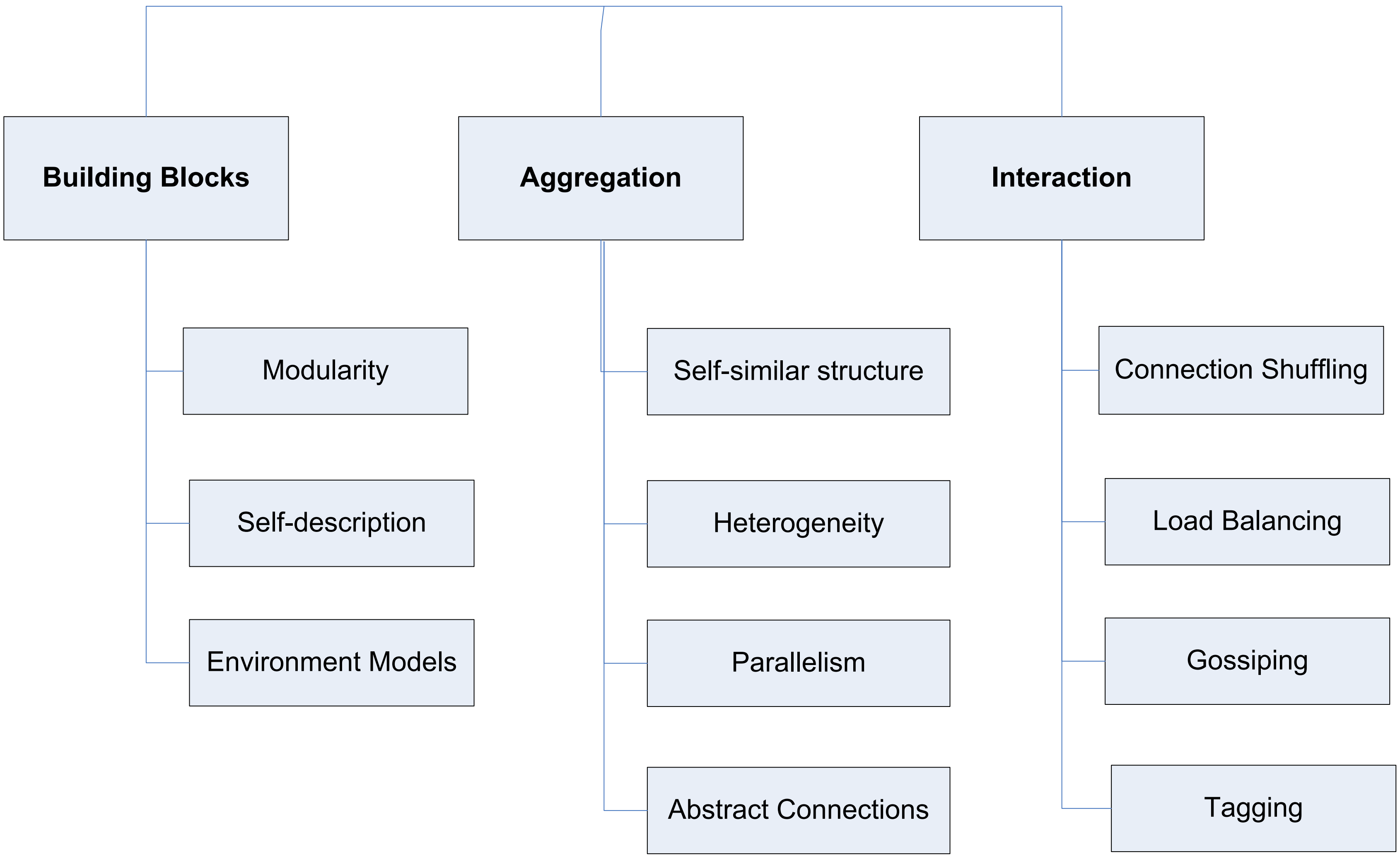
- **We previously defined tactics that address seven quality attributes: availability, interoperability, modifiability, performance, testability, security, and usability.**



EXAMPLE: AVAILABILITY TACTICS



TACTICS FOR ULS SYSTEMS



BUILDING BLOCKS

Modularity

Self-description

Environment Models

BUILDING BLOCKS: MODULARITY

Arguably the most common design principle in all of software engineering

- supports super-linear growth in software
- supports the loose coupling of the PSTN
- all biological systems are composed of independent agents

BUILDING BLOCKS

Modularity

Self-description

Environment Models

BUILDING BLOCKS: SELF-DESCRIPTION

Models that the peers maintain of themselves

- self-description permits a rudimentary form of self-awareness
- a core property of the internet: "objects should be self-describing"
- biological agents all have internal models, which they update over time

BUILDING BLOCKS

Modularity

Self-description

Environment Models

BUILDING BLOCKS: ENVIRONMENT MODELS

Models that peers maintain of their environment

- complex adaptive systems contain internal models of the environment which are constantly updated to reflect observed phenomena
- P2P systems use such models in their gossiping algorithms

AGGREGATION

Self-similar structure

Heterogeneity

Concurrency

Abstract Connections

AGGREGATION: SELF-SIMILAR STRUCTURE

Collections of entities (including collections of collections) are treated similarly to individuals

- **supernodes in P2P systems**
- **fractal structure of the internet**

AGGREGATION

Self-similar structure

Heterogeneity

Concurrency

Abstract Connections

AGGREGATION: HETEROGENEITY

Peers will have different properties; these must be abstracted and supported

- **internet nodes**
- **P2P nodes**
- **diversity in biological systems**

AGGREGATION

Self-similar structure

Heterogeneity

Concurrency

Abstract Connections

AGGREGATION: CONCURRENCY

Peers can run independently, in parallel, without centralized control

- true of all biological systems
- true of any system of systems

AGGREGATION

Self-similar structure

Heterogeneity

Concurrency

Abstract Connections

AGGREGATION: ABSTRACT CONNECTIONS

Connections must be abstract, and realized at run-time

- **internet nodes**
- **SOA services and SoS nodes are annotated with properties**
- **biological systems, where connections are all realized at run-time**

INTERACTION

Connection Shuffling

Load Balancing

Gossiping

Tagging

INTERACTION: CONNECTION SHUFFLING

Peers actively and continuously seek out their neighbors

- **core feature of the PSTN, P2P systems, the internet**
- **randomness in ant interactions**

INTERACTION

Connection Shuffling

Load Balancing

Gossiping

Tagging

INTERACTION: LOAD BALANCING

For efficient operation the work needs to be appropriately apportioned among the peers

- P2P systems
- web-server farms

However, in ULS systems balancing needs to account for power laws of relationships.

INTERACTION

Connection Shuffling

Load Balancing

Gossiping

Tagging

INTERACTION: GOSSIPING

The peers need to be constantly interacting: adapting to their ever-changing state and environment

- **P2P systems exchange state information**
- **ants leave pheromone trails**

INTERACTION

Connection Shuffling

Load Balancing

Gossiping

Tagging

INTERACTION: TAGGING

For groups to form, and to create boundaries, some form of tagging is required

- **DNS servers support hierarchical tagging**
- **super-nodes in P2P systems**
- **nest-mates in ant colonies**

CONSEQUENCES

Not only are these tactics characteristic of ULS systems but they are *necessary* characteristics.

It is difficult to imagine how a system could grow without bound without *all* of these tactics.

Different tactics may be present to different degrees in the PSTN, or KaZaA, or the internet, or in biological systems, but they are all present.

PROOF (?)

Clearly these claims can not be “proven”.

But what are their explanatory power?

We will consider additional two examples, one biological and one artificial:

1) MANETs (Mobile Ad hoc NETworks)
2) Slime molds

EVIDENCE: MANETS

Mobile Ad hoc NETworks:

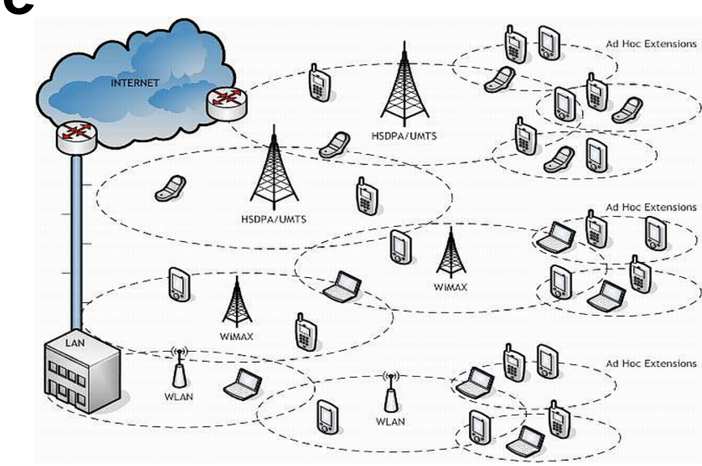
- each peer is *modular*, operating in *parallel*

- peers are typically *heterogeneous*, sharing only a communication protocol, which acts as an *abstract connection* mechanism

- peers are *self-describing*

- nodes exhibit *self-similar structure*

- nodes are continually *connection shuffling* and *gossiping*



EVIDENCE: SLIME MOLDS

- slime mold cells
- are peers (hence modular) with their own state, hiding internals, operating concurrently.
- exist as individuals, but can aggregate into groups of up to 10^5 to create “slugs” that can travel, or into spore-bearing fruiting structures—sporangiophore—for reproduction.
- have internal models
- have environmental models
- exhibit heterogeneity

PARTING THOUGHTS

Every tactic exists to serve an objective in the larger process of design, to control a systemic response.

The tactics presented here exist to manage (ultra-large) scalability concerns.

Taken together, these tactics represent an empirically grounded ontology of ULS design.


Thank You