INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

Fundamentals of Agile Systems Engineering – Part 1 (Agile-Systems Engineering)

Rick Dove
Paradigm Shift International, and
Stevens Institute of technology

Ralph LaBarge
Johns Hopkins University/APL

INCOSE IS14
Las Vegas, NV, USA
30 June – 3 July, 2014

In a Nutshell…

Agile systems-engineering and agile-systems engineering are two different concepts that share the word agile.

In the first case the system of interest is an engineering process, and in the second case the system of interest is what is produced by an engineering process.

The word agile refers to the adaptability and the sustainment of adaptability in both types of systems.

Sustained adaptability is enabled by an architectural pattern and a set of system design principles that are fundamental and common to both types of systems.

Research that identified this architectural pattern and design principles is reported, updated, and applied here in two Parts.

Part 1 focuses on agile-systems engineering, reviewing the origins, values, and core concepts that define and enable domain independent agility in any type of system.

Part 2 focuses on agile systems-engineering, identifying core agility-enabling concepts in the software-development domain-specific practice known as Scrum, reviewing an agile hardware/software satellite-development systems-engineering case for its source of agility,

and then suggesting the development of an agile systems-engineering life cycle model as a natural next step.

XXXXXXXX

The value proposition of an agile system is rooted in risk management, providing options when system mission or system survival is threatened.

Most natural systems have evolved sufficient agility to sustain existence in the inherently risky environments that surround them. But nature doesn't care. Agility is a byproduct of natural selection, an algorithm without an objective.

We can also learn from man-made systems that exhibit the ability to survive, even thrive, in uncertain and unpredictable environments, and analyze these systems for common and replicable patterns that provide this capability.

Intensively in the nineties, and continuously thereafter, well over 100 man-made systems exhibiting agile characteristics have been studied in workshops conducted at a wide variety of host sites, which examined systems in many domains including manufacturing processes, enterprise processes, hardware systems, software systems, and development systems (processes).

This article summarizes the findings of those empirical studies, with the purpose of presenting in one document what appear to be necessary and sufficient fundamental architecture and design guidance for the systems engineering practitioner. The engineering usefulness of the architecture and supporting design principles have been confirmed in twenty five years of evolution and deployed employment

Today's Agility Interest – Origin

1991 – US SecDef funded project at Lehigh University to identify next manufacturing competitive focus beyond Lean

- **13 companies participated full-time in 3-month workshop**
- **2 vol report: 21st Century Manufacturing Enterprise Strategy**
- **Problem/opportunity defined (for manufacturing enterprises)**

1992 – Agile Manufacturing Enterprise Forum founded at Lehigh, funded by Texas Instruments and General Motors

- **Purpose: Identify nature of Agile solution**
- **Method: Industry collaborative workshop groups**

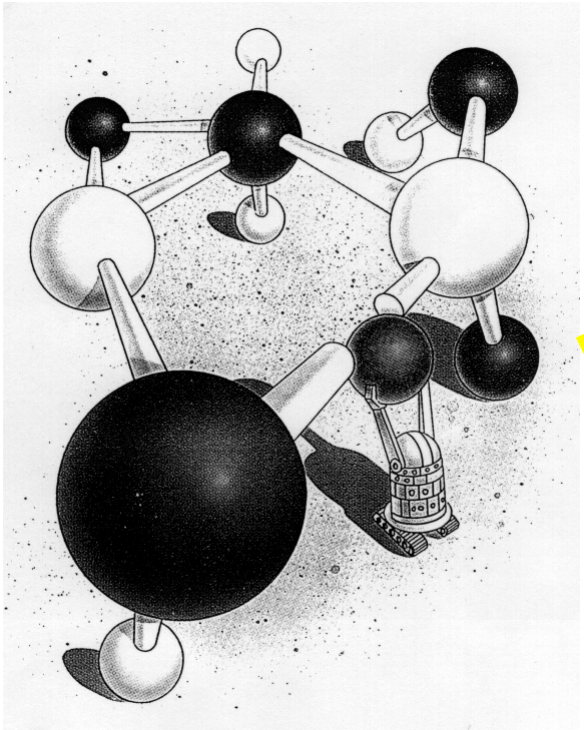
1994 – DARPA/NSF establish $5 Million x 5 year funding

- **Name changed to Agility Forum (any kind of enterprise/system)**
- **Research steering group and agenda established**
- **250+ orgs, 1000+ participants in focused workshop groups**
- **Conferences, papers, reference base, tools, reference model**

1998 – Mission accomplished, Agility Forum dissolved

- **Agility pursuit by industry and IT vendors entrenched**

AGILITY DEFINED



Circa 1991

The Ability to Thrive

in a

Continuously Changing,

Unpredictable

Environment.

RECONFIGURABLE EVERYTHING

Agile-Systems Research Focus

Problem:

- Technology and markets are changing faster than the ability to employ/accommodate
- System-needs are uncertain and unpredictable
- Flexible system approaches inadequate when requirements change
- New approach needed that could extend usefulness/life of systems

Solution Search:

- Examined 100s of systems of various types
- Looked for systems that responded *effectively*
- Looked for metrics that defined *effectively*
- Looked for categories of response types
- Looked for principles that enabled response

Note: This research took place at the Agility Forum 1992-1996, and in subsequent independent research 1997-1999

Essays chronicle knowledge development at www.parshift.com/library.htm

Defining Agility

Agility is *effective response* to opportunity and problem, within mission ... always.

**Not fast,
…just fast enough**

An *effective response* is one that is:

- **timely (fast enough to deliver value),**
- **affordable (at a cost that leaves room for an ROI),**
- **predictable (can be counted on to meet expectations),**
- **comprehensive (anything/everything within mission boundary).**

An ineffective response is failure - there is zero tolerance for failure today.

You can think of Agility as Requisite Variety.

You can think of Agility as proactive Risk Management.

The trick is understanding the nature of agile-enabling concepts, and how they can be applied to any type of system.

Domain Independent

Significant Touch Points

DoD's Command and Control Research Program began an exploration of agile command and control (Alberts 1996) that continues today.

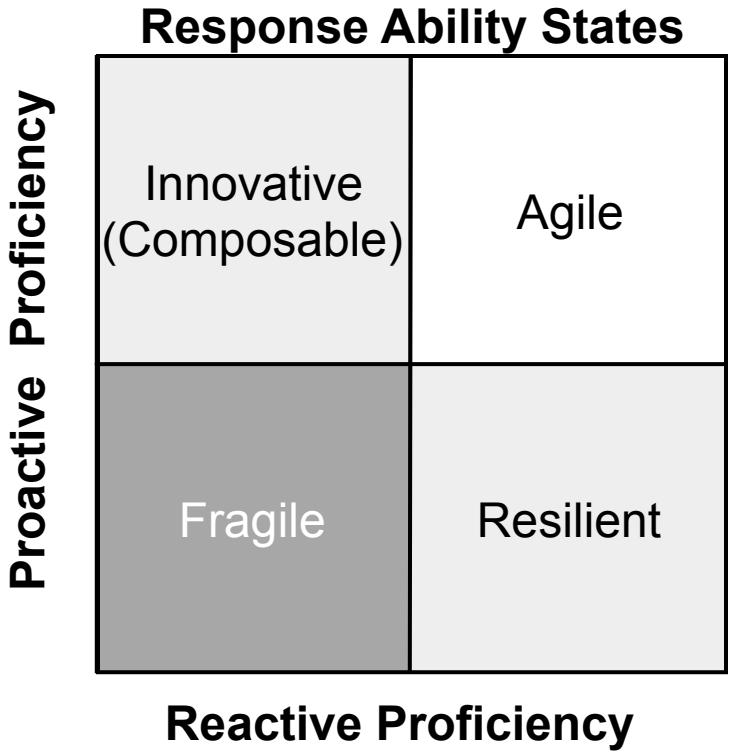
Alberts, David S. 1996 revised 2002. *Information Age Transformation – Getting to a 21st Century Military*. DoD Command and Control Research Program (CCRP). www.dodccrp.org/html4/books_downloads.html.

Alberts, David S. 2011. *The Agility Advantage: A Survival Guide for Complex Enterprises and Endeavors*. DoD Command and Control Research Program (CCRP). www.dodccrp.org/html4/books_downloads.html.

The Agile Manifesto for Software Development (Fowler and Highsmith 2001) adopted the agile label as appropriately descriptive and fundamentally consistent with their concepts. … Though they focus more on good project management practices rather than on what makes the system fundamentally agile. More on this in Part 2.

Two dimensions of response proficiency (Response Ability – RA)

- **Resilient: RA state marked by good reactive change competency, at least sufficient to be generally viable.**
- **Innovative: RA state marked by good proactive change competency, at least sufficient to be a market influencer.**
- **Agile: RA state marked by high competence at both proactive and reactive change.**
- **Fragile: RA state marked by small competency at change. Insufficiently reactive to shrug off adversity. Insufficiently proactive to influence the market and mission.**



Agile Systems, Fundamentally…

Are designed for change.

They can be augmented with new functional capability.

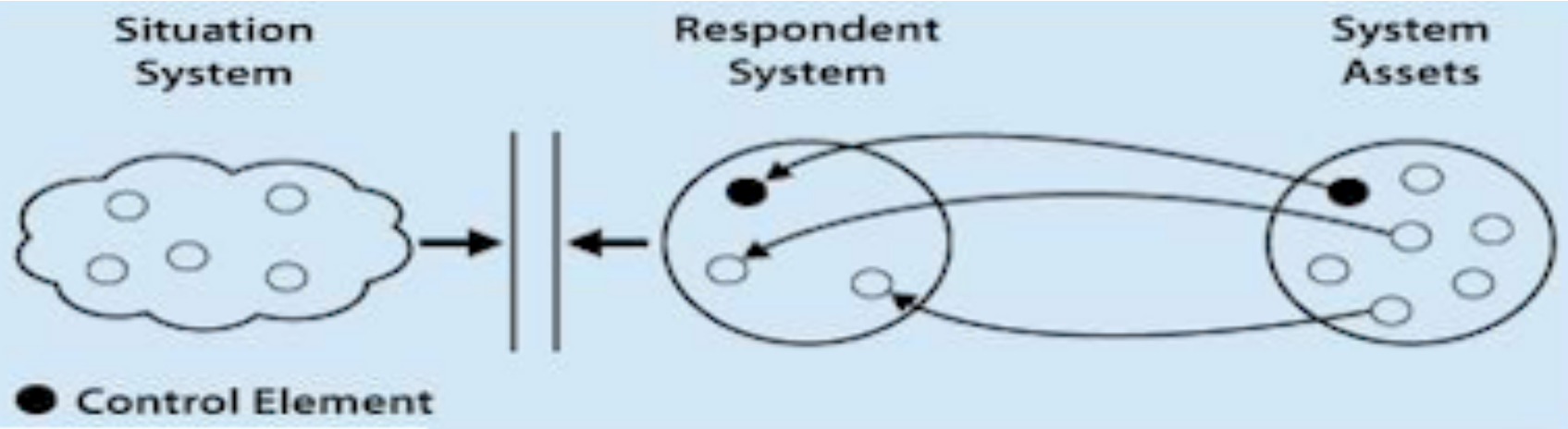
They can be restructured with different internal relationships among their subsystems.

They can be scaled up or down for economic delivery of functional capability.

They can be reshaped to regain compatibility or synergy with an environment that has changed shape.

These types of changes are structural in nature, and require an architecture that accommodates structural change.

System-Coupling Diagram



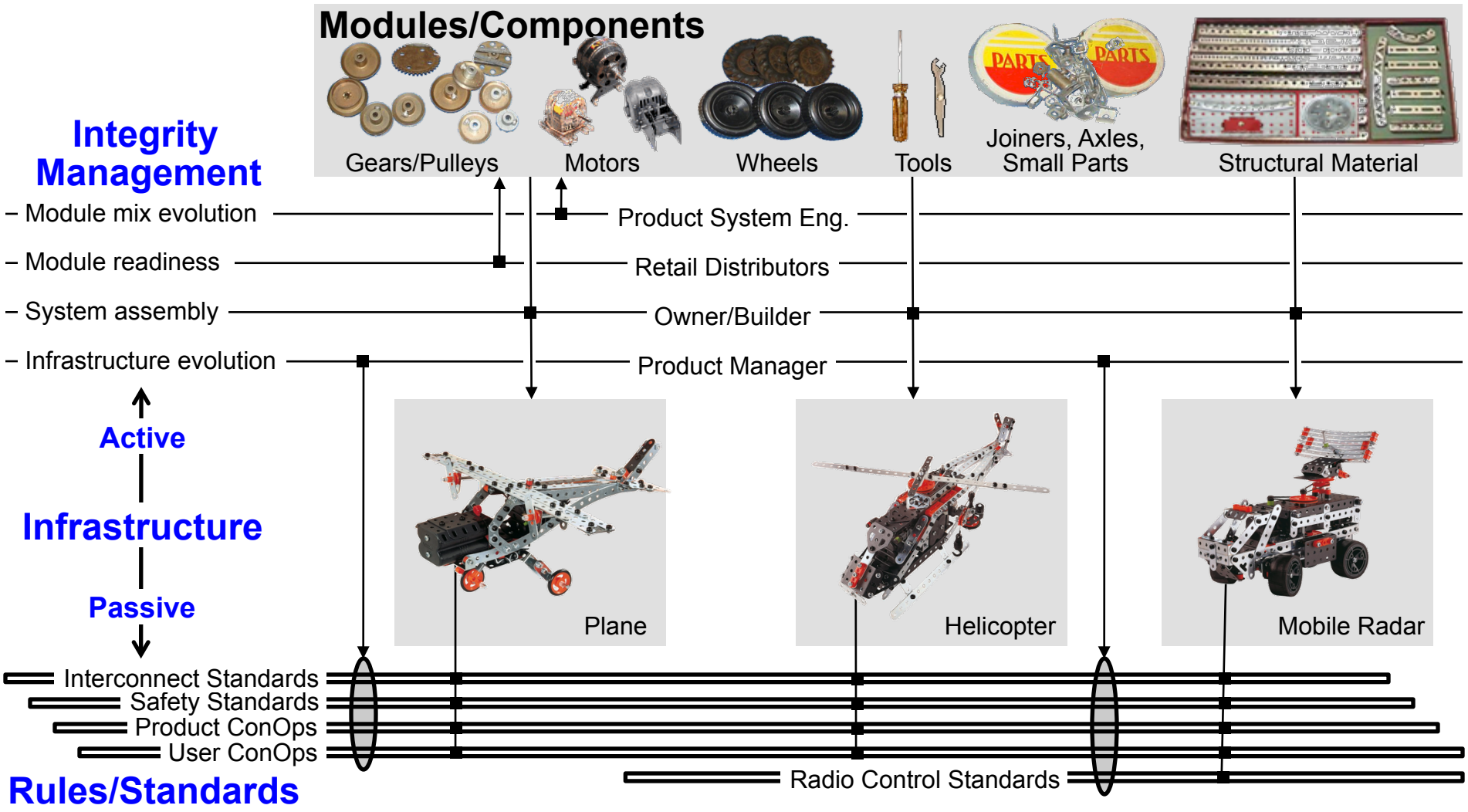
System-Coupling Diagram (Lawson 2010: 23) illustrating composability of a response system appropriate to a situation.

Lawson, Harold 'Bud'. 2010. *A Journey Through the Systems Landscape*.


http://www.vintagetoys.com/toys/classified/962
Sensational New
ERECTOR
DEVELOPED AT THE GILBERT HALL OF SCIENCE
GILBERT
ERECTOR
No 10052
THE ROCKET LAUNCHER SET
Build this
Exciting Action
Rocket Launcher!
POWERFUL ELECTRIC MOTOR
PARTS
PARTS
www.girdersandgears.com/erector-1958-10052.html
ERECTOR=MECCANO

System Construction Kit

Agile architecture pattern depicting an Erector-set construction-kit example.



Agile Architecture Fundamentals

There are three critical elements in the agile architectural pattern:

a roster of drag-and-drop encapsulated modules,

a passive infrastructure of minimal but sufficient rules and standards that enable and constrain plug-and-play interconnection, and

an active infrastructure that designates four specific responsibilities for sustaining agile operational capability.

Modules

Modules are self-contained encapsulated units complete with well-defined interfaces which conform to the plug-and-play passive infrastructure.

They can be dragged-and-dropped into a system of response capability with relationship to other modules determined by the passive infrastructure.

Modules are encapsulated so that their methods of functionality are not dependent on the functional methods of other modules, except perhaps as the passive infrastructure may dictate.

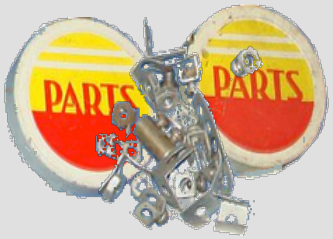
Modules/Components


Gears/Pulleys

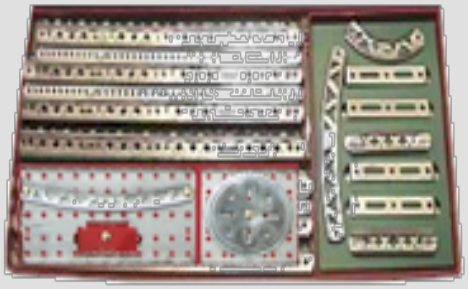

Motors


Wheels


Tools

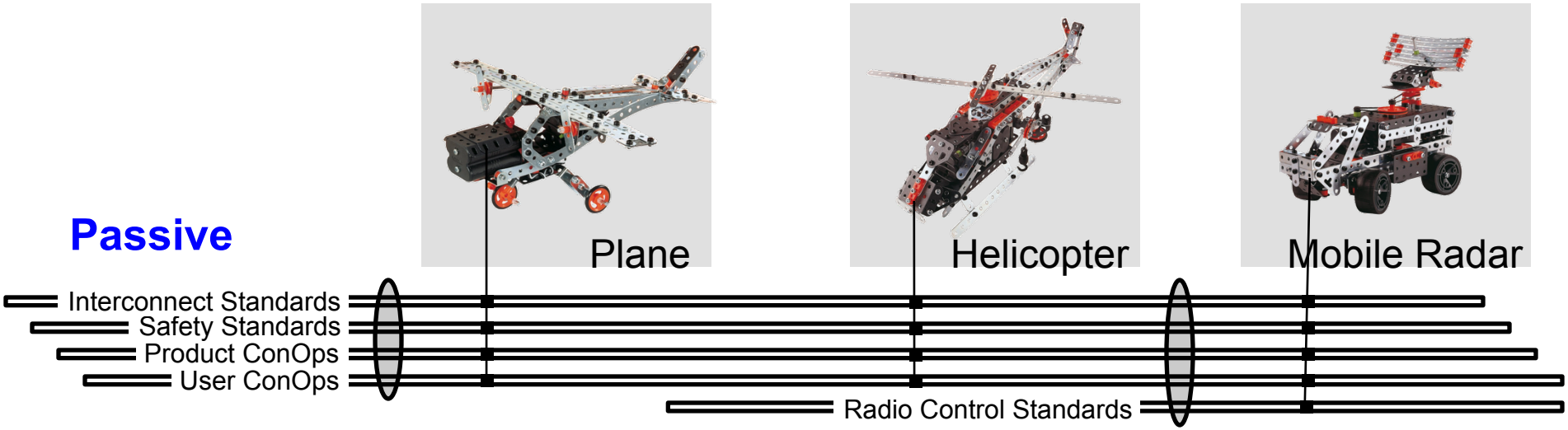


Joiners, Axles, Small Parts


Structural Material

Passive Infrastructure

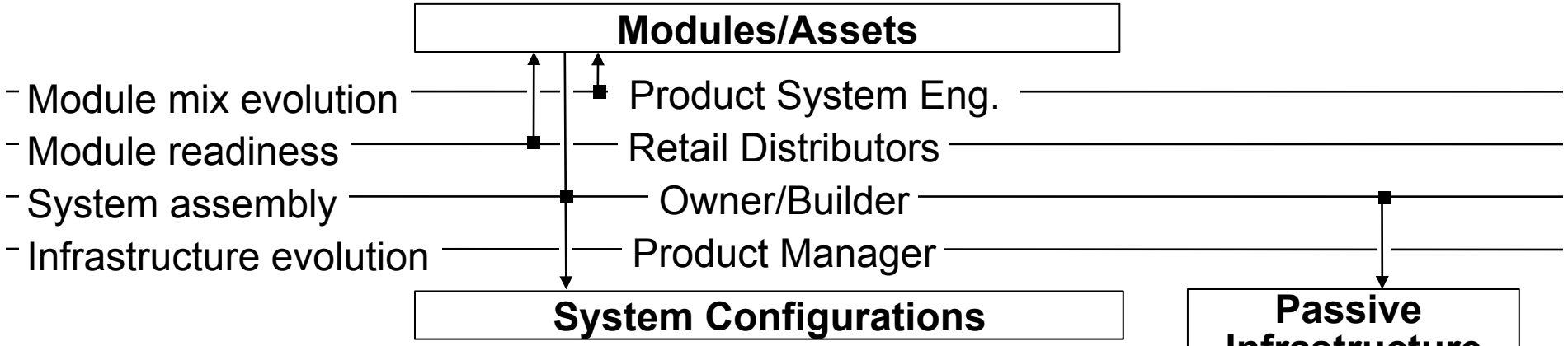
The passive infrastructure provides drag-and-drop connectivity between modules. Its value is in isolating the encapsulated modules so that unexpected side effects are minimized and new operational functionality is rapid. Selecting passive infrastructure elements is a critical balance between requisite variety and parsimony – just enough in standards and rules to facilitate module connectivity, but not so much to overly constrain useful innovative system configurations. At least five categories of standards and rules should be considered: sockets (physical interconnect), signals (data interconnect), security (trust interconnect), safety (of user, system, and environment), and service (system assembly ConOps and evolutionary agility sustainment).



Active Infrastructure

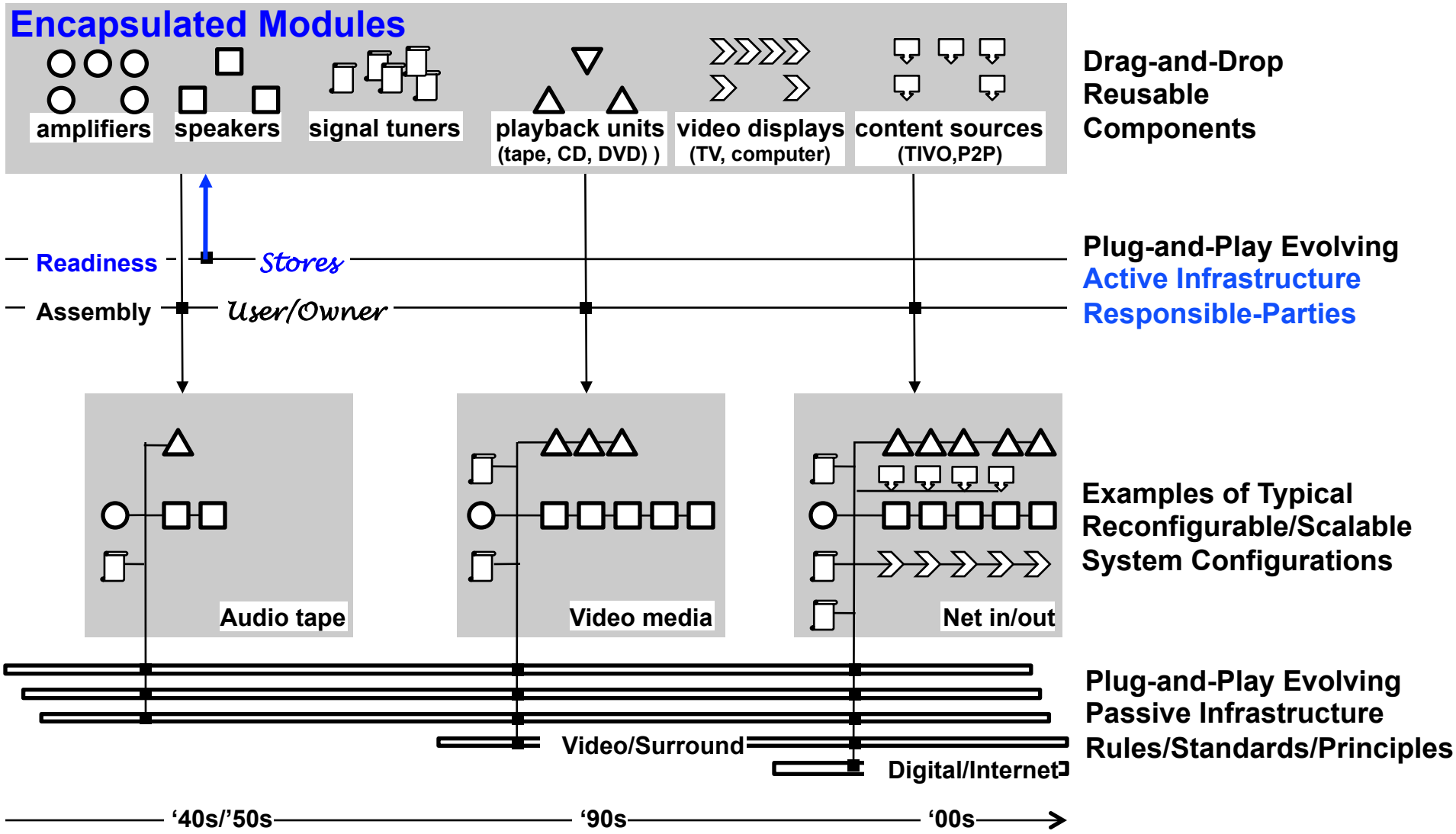
Four responsibilities are required and must be designated and embedded within the system to ensure that effective response capability is possible at unpredictable times. The "how" processes of dispatching responsibility should be articulated in the service element of the passive infrastructure.

- **Module Mix Evolution—Who (or what process) is responsible for ensuring that existing modules are upgraded, new modules are added, and inadequate modules are removed, in time to satisfy response needs?**
- **Module Readiness—Who (or what process) is responsible for ensuring that sufficient modules are ready for deployment at unpredictable times?**
- **System Assembly—Who (or what process) assembles new system configurations when new situations require something different in capability?**
- **Infrastructure Evolution—Who (or what process) is responsible for evolving the passive and active infrastructures as new rules and standards become appropriate to enable next generation capability.**



Case: Home Entertainment Technology Migration

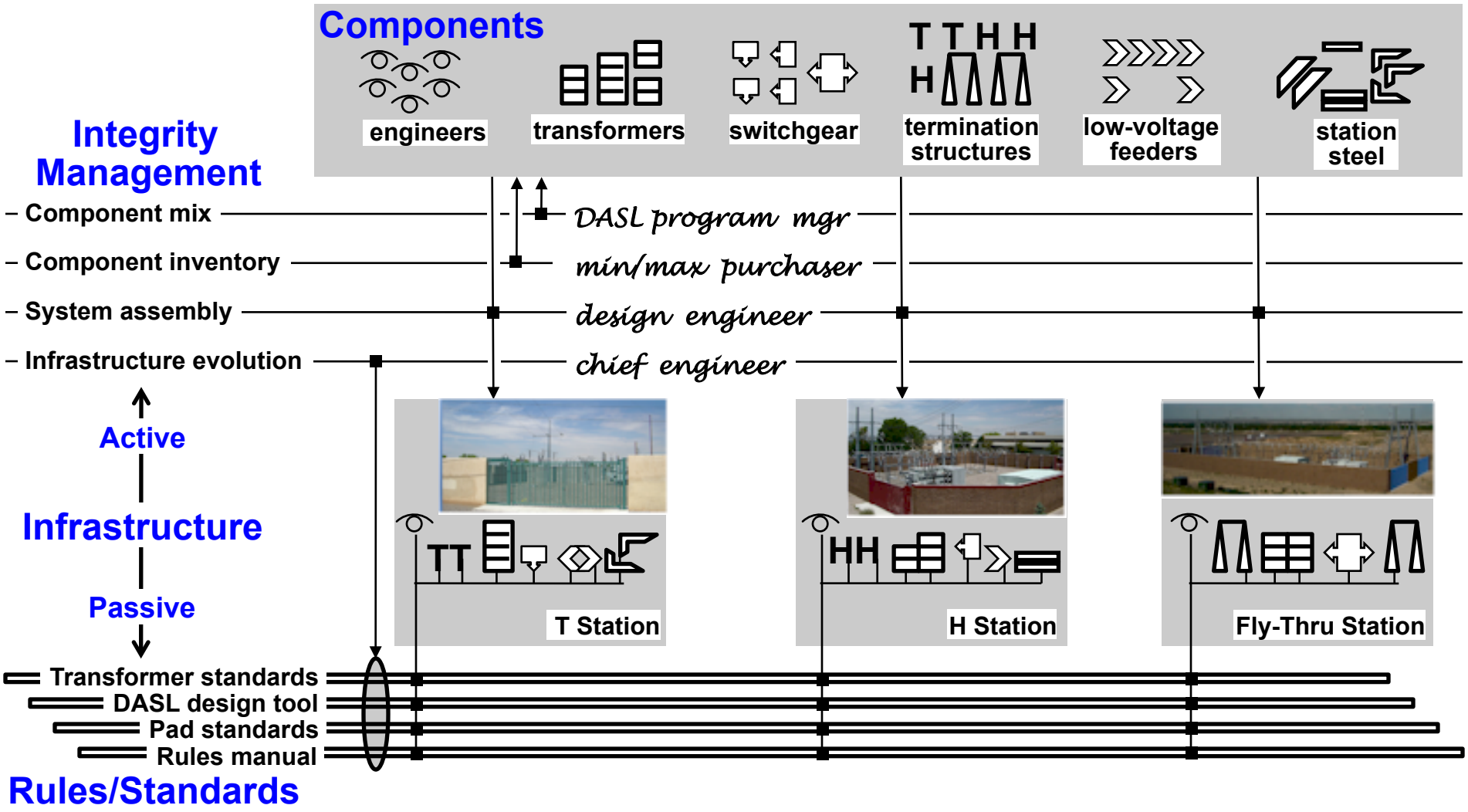
agile architecture pattern: drag-and-drop, plug-and-play



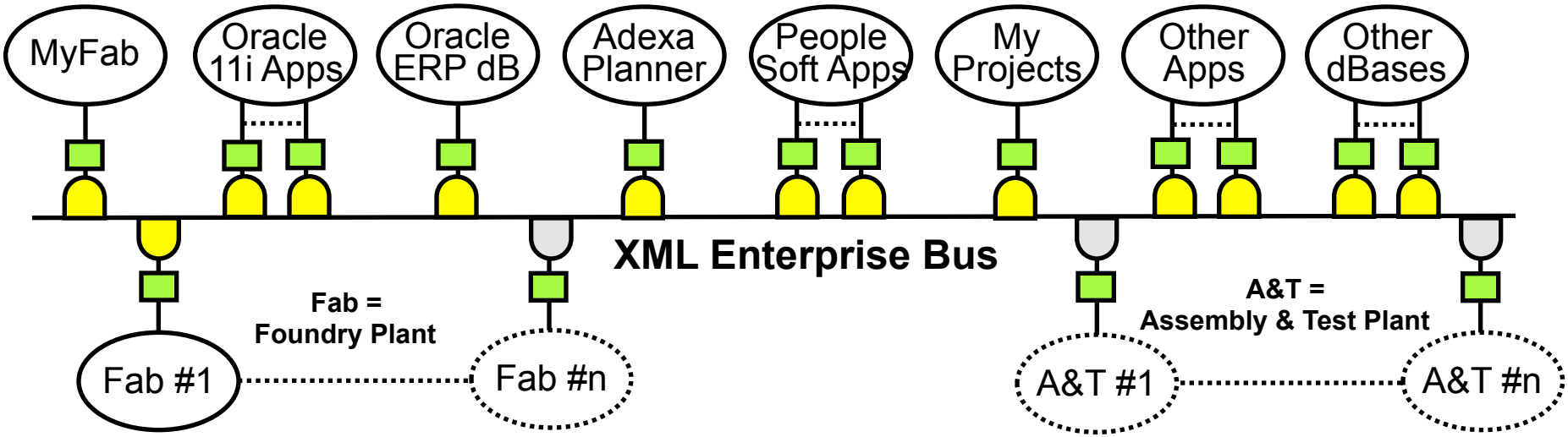
PNM Agile Substation System

www.parshift.com/Files/PsiDocs/Pap080404Cser2008DevOpsMigration.pdf

Architectural Concept Diagram



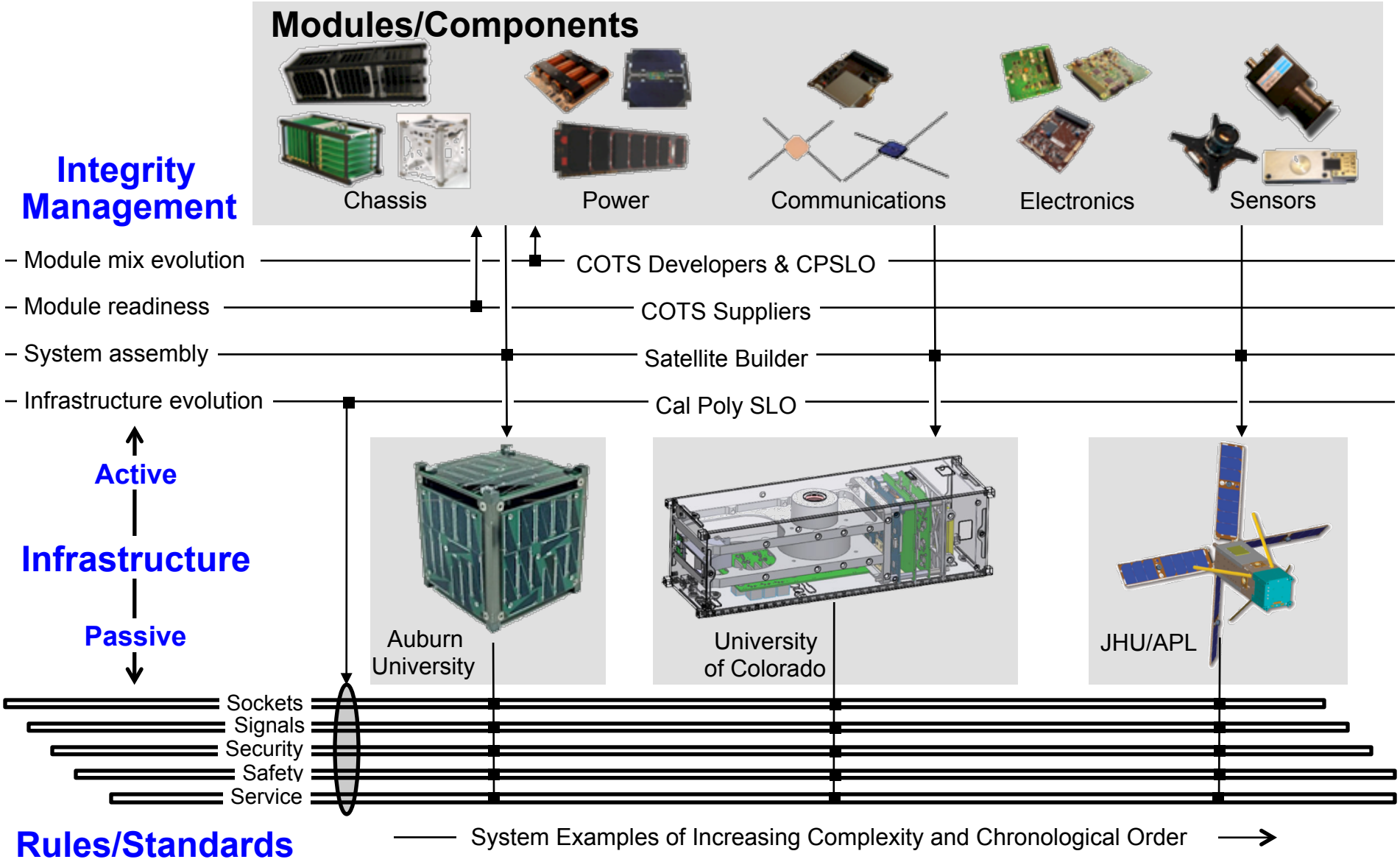
Silterra IT Infrastructure Design



- = Bus Interface Module (BIM)
- = ETL Interface Modules
- **MyProjects = Web-accessible strategic-project portfolio manager**
- **MyFab = Web-accessible operations transparency**

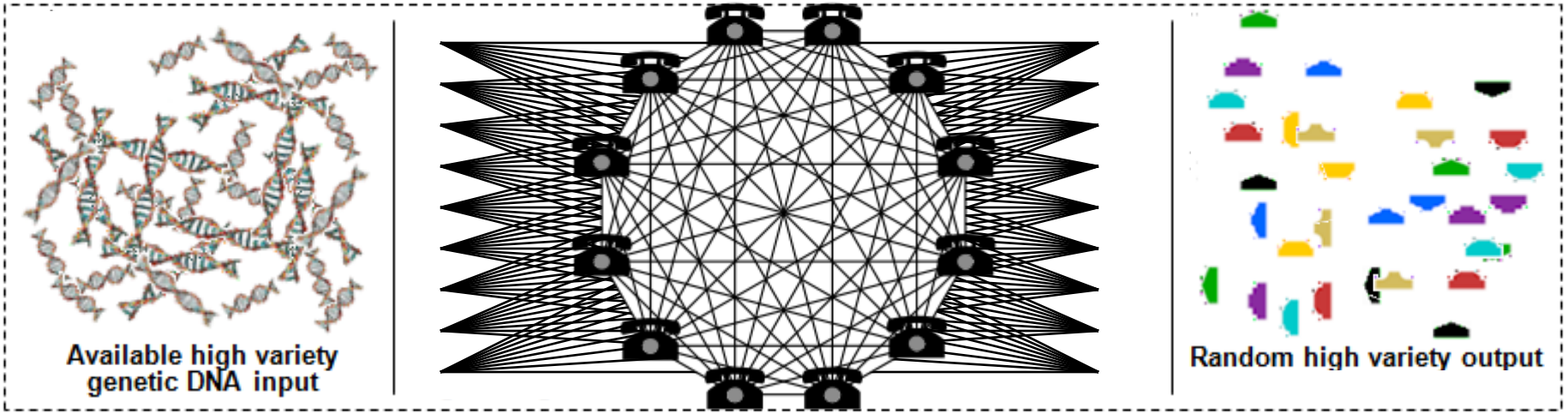
www.parshift.com/Files/PsiDocs/Rkd050324CserPaper.pdf

CubeSat Agile Architectural Pattern

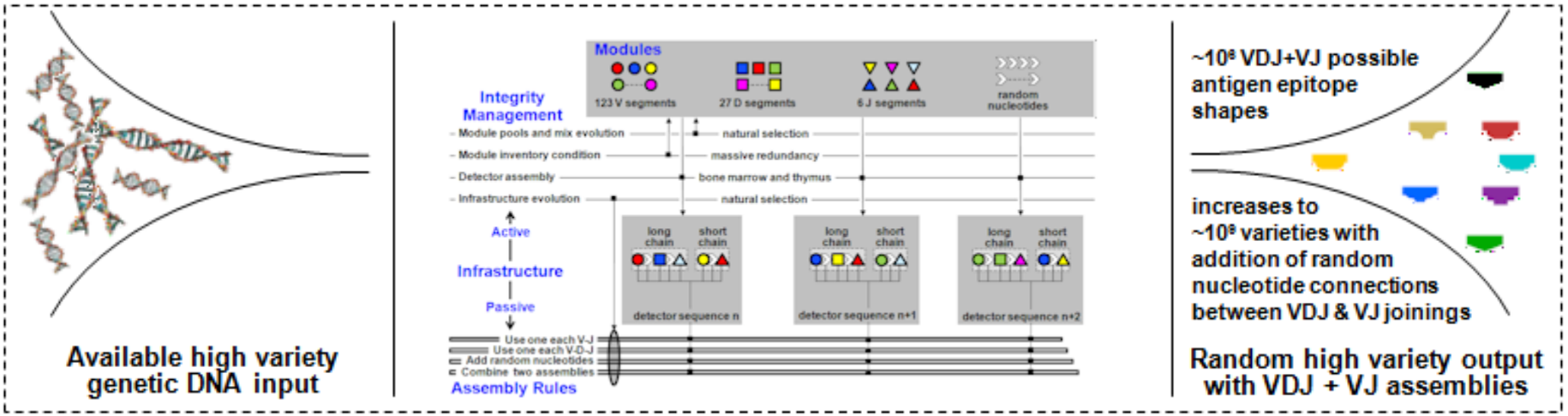


CP SLO: *Cal Poly San Luis Obispo*

Order Out of Chaos



Above: Point-to-Point interfaces needed between all resources and all needs
Below: Simple protocols minimize custom interfaces for satisfying needs – minimal failure points



Metcalf's Law: Value is the square of <u>compatibly communicating</u> modules

Dove, Rick, 2011, Self-Organizing Resilient Network Sensing (SornS) with Very Large Scale Anomaly Detection, IEEE International Conference on Technologies for Homeland Security, Waltham, MA, Nov. 15-17.

Response Able Design Principles – RRS

Reusable

Self-Contained Units (Modules)
Modules are encapsulated independent units loosely coupled through the passive infrastructure.

Plug Compatibility (Facilitated Interfacing)
Modules & infrastructure have features facilitating easy module insertion/removal.

Facilitated Reuse
Modules are reusable and/or replicable; with supporting facilitation for finding and employing appropriate modules.

Scalable

Evolving Standards (Infrastructure) - Module interface and interaction standards and rules that are monitored/updated to accommodate old, current, and new modules; defines module compatibility.

Redundancy and Diversity
Duplicate modules provide fail-soft & capacity options; diversity provides functional options.

Elastic Capacity
Module populations & functional capacity may be increased and decreased widely within the existing infrastructure.

Reconfigurable

Flat Interaction
Modules communicate directly on a peer-to-peer relationship; parallel rather than sequential relationships are favored.

Deferred Commitment
Module relationships are transient when possible; decisions & fixed bindings are postponed until necessary.

Distributed Control and Information
Decisions made at point of maximum knowledge; information accessible globally but kept locally.

Self-Organization
Module relationships are self-determined; and component interaction is self-adjusting or negotiated.

Reusable Principles

Encapsulated Modules (Modularity)—Need: System assemblers want effective module replacement and internal change without side effects. Intent: Modules physically encompass a complete capability, and have no dependencies on how other modules deliver their capabilities.

Facilitated Interfacing (Plug Compatibility)—Need: System assemblers want effective interfacing that facilitates integration and replacement of modules. Intent: Modules share minimal interface standards, and are readily inserted and removed.

Facilitated Reuse—Need: System assemblers want effective module selection and acquisition that facilitates reuse. Intent: Available modules are identified by capability and requirements, and can be readily discovered and acquired for deployment.

Reconfigurable Principles

Peer-Peer Interaction—Need: System assemblers want effective communication among modules. Intent: Modules communicate directly on a peer-to-peer basis to avoid intermediary relay failure, content filtering, and time delay.

Distributed Control and Information—Need: System assemblers want effective information-based operational decisions. Intent: Decisions are made where maximal situational knowledge exists, and relevant information is maintained local to decision making modules while accessible globally.

Deferred Commitment—Need: System assemblers want to maintain effective response ability. Intent: Conserve the commitment and consumption of limited resources to the last responsible moment, in anticipation of future unpredictable events and uncertain response needs.

Self-Organization—Need: Systems assemblers want effective adaptation of interacting modules. Intent: Module relationships are self-determined where possible, and module interactions are self-adjusting or self-negotiated.

Scalable Principles

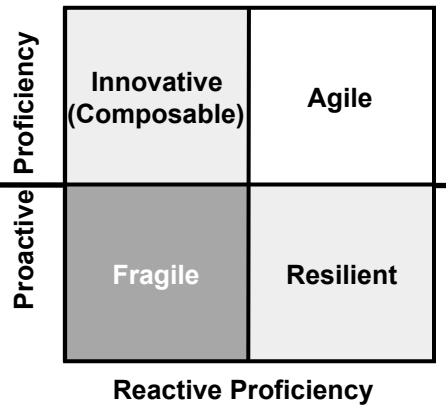
Evolving Standards—Need: System assemblers want effective acquisition and deployment of new module capabilities. Intent: Passive infrastructure standards and rules are monitored for current relevance, and evolve to accommodate new and beneficial module types in anticipation of need.

Redundancy and Diversity—Need: System assemblers want effective resilience under quantitative (need more of something) and qualitative (need something different) situational variance. Intent: Duplicate or replicable modules provides quantitative capacity options and fault tolerance options; diversity among similar modules provides situational fit options.

Elastic Capacity—Need: System assemblers want to incrementally match committed system resources to situational capacity needs of unpredictable or uncertain range. Intent: Modules may be combined in unbounded quantities, where possible, to increase or decrease deliverable functional capacity within the current architecture.

Change/Response Domains

	Change Domain	General Characteristic
Proactive	Creation (and Elimination)	Proactive Innovative/Composable Creates Opportunity Takes Preemptive Initiative
Proactive	Improvement	
Proactive	Migration	
Proactive	Modification (of Capability)	
Reactive	Correction	Reactive Resilient Seizes Opportunity Copes with Adverse Events
Reactive	Variation	
Reactive	Expansion (of Capacity)	
Reactive	Reconfiguration	



Change/Response Domains

Change Domain		
Proactive	Creation (and Elimination)	Proactive responses are generally triggered internally by the application of new knowledge to generate new value. They are still proactive responses even if the values generated are not positive and even if the knowledge applied is not new – self initiation is the distinguishing feature here. A proactive change is usually one that has effect rather than mere potential; thus, it is an application of knowledge rather than the invention or possession of unapplied knowledge. Proactive change proficiency is the wellspring of leadership and innovation in system capability.
	Improvement	
	Migration	
	Modification (of Capability)	
Reactive	Correction	Reactive responses are generally triggered by events which demand a response: problems that must be attended to or fixed, opportunities that must be addressed. The distinguishing feature is little choice in the matter – a reaction is required. Reactive responses often address threatening competitive or environmental dynamics, new customer demands, agility deterioration/failure, legal and regulatory disasters, product failures, market restructuring, and other non-competitor generated events. Reactive change proficiency is the foundation of resilience and sustainability in system capability.
	Variation	
	Expansion (of Capacity)	
	Reconfiguration	

Change/Response Domains

Change Domain		
Proactive	Creation (and Elimination)	**Proactive responses** are generally triggered internally by the application of new knowledge to generate new value. They are still proactive responses even if the values generated are not positive and even if the knowledge applied is not new – self initiation is the distinguishing feature here. A proactive change is usually one that has effect rather than mere potential; thus, it is an application of knowledge rather than the invention or possession of unapplied knowledge. Proactive change proficiency is the wellspring of leadership and innovation in system capability.
	Improvement	
	Migration	
	Modification (of Capability)	
Reactive	Correction	**Reactive responses** are generally triggered by events which demand a response: problems that must be attended to or fixed, opportunities that must be addressed. The distinguishing feature is little choice in the matter – a reaction is required. Reactive responses often address threatening competitive or environmental dynamics, new customer demands, equipment malfunctions, legal and regulatory disasters, product failures, market restructuring, and other non-competitor generated events. Reactive change proficiency is the foundation of resilience and sustainability in system capability.
	Variation	
	Expansion (of Capacity)	
	Reconfiguration	

Proactive Domains

Creation/Elimination—What range of opportunistic situations will need modules assembled into responsive system configurations; what elements must the system create during operation that can be facilitated by modules and module pools; what situational evolution will cause obsolesce of modules which should be removed? The distinguishing feature is the creation of something new or reincarnated that is not currently present. To note, this is not about the situation that calls for the original creation of an agile system, but rather about the evolution of the agile system during its operational period. Situations to identify are those that require system configuration assemblies during operation, and those that require new modules for employment in those assemblies.

Improvement—What improvements in system response performance will be expected over the system's operational life? The distinguishing feature is performance of existing response capability, not the addition of new capability. Situations to identify are generally those involving competencies and performance factors, and are often the focus of continual, open-ended campaigns.

Migration—What evolving technologies and opportunities might require future changes to the infrastructure? The distinguishing feature is a need to change the nature of the plug-and-play infrastructure, not the addition of new modules. Situations to identify are generally those that enable the transition to possible and potential next generation capabilities.

Modification (of capability)—What evolving technologies and opportunities might require modification of the available modules and roster of module pools? The distinguishing feature is a necessary change in available module capabilities. Situations are generally those that require something unlike anything already present, or the upgrade or change to something that does exist.

Reactive Domains

Correction—What types of response activities might fail in operation and need correction? The distinguishing feature is a dysfunction or inadequacy during attempted response. Situations to identify are those that require a recovery from response malfunction, recovery from unacceptable side effects of a response, and inability to assemble an effective response.

Variation—What aspects of operational conditions and resources vary over what range when response capabilities must be assembled? The distinguishing feature is predictable but uncertain variance. Situations to identify are those that manifest as variances in module availability, module performance, and module interactions.

Expansion/Contraction (of capacity)—What are the upper and lower bounds of response capacity needs? The distinguishing feature is capacity scalability. Situations to identify are those that can be satisfied with planned capacity bounds, as well as those that have indeterminate and unbounded capacity needs.

Reconfiguration—What types of situations will require system reconfiguration in order to respond effectively? The distinguishing feature is the configuration and employment of available modules for new or reincarnated response needs. Situations to identify are those that are within the system mission boundaries, and that may require a reconfiguration of an existing system assembly, perhaps augment with removal of modules or addition of available modules.

WRAP

This article is Part 1 of a two part article on agile systems engineering. This part deals with agile-systems engineering, a necessary precursor for understanding agility in agile systems-engineering, as an agile systems-engineering process is itself an agile system.

Unique to this article is:

- **the historical review of agile system definition, research, and concept development;**
- **and the recognition of David Albert's extensive work in Agile C4I and military enterprise as compatible.**

Also unique, but intended as the practitioner's take-away, is the model of agile-systems engineering as the engineering of a system construction kit;

- **the introduction of the UURV framework;**
- **the updated and augmented articulation of the agile architectural pattern,**
- **the ten agile system design principles, and**
- **the eight response situation analysis domains.**

The introduction of the CubeSat agile system example in this article will play a role in Part2, when the agile systems-engineering process at John Hopkins University Applied Physics Laboratory (JHU/APL) for developing CubeSats is examined.

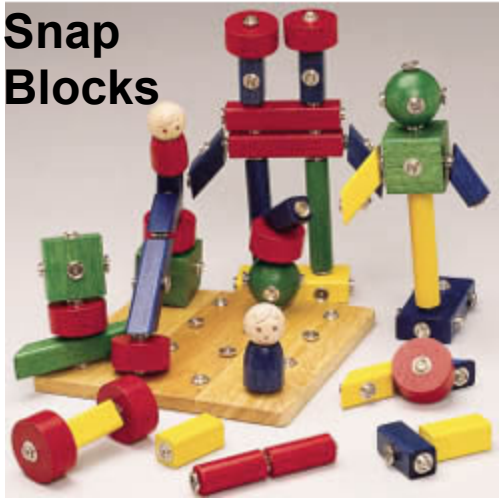
Design the Elements of Your Construction Set

Bendables

Straws and Connectors

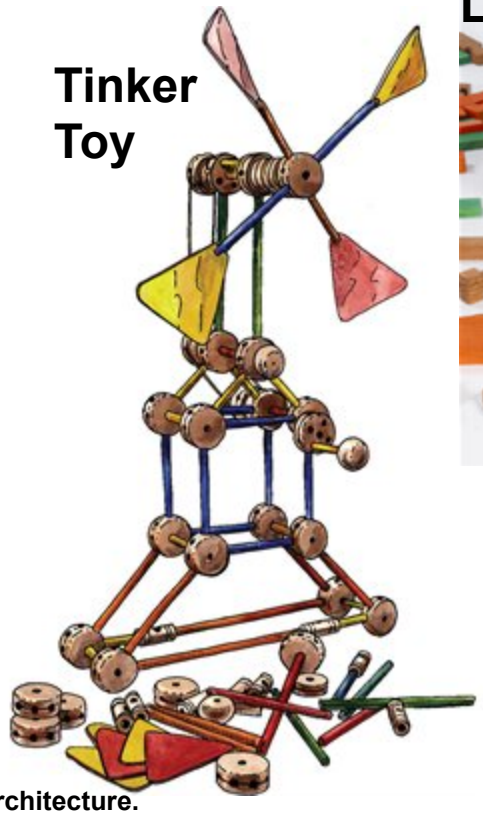
Marble Run

Snap Blocks



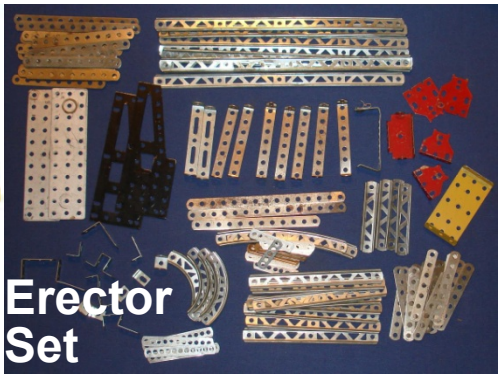
Woodbuilders

Tinker Toy



Log Builder

Lego



Bristle Blocks

Construction (response) architecture different from system functional architecture.
Response architecture is a domain-focused engineering architecture