

Webinar

**Agile Systems and Processes:
Necessary and Sufficient Fundamental Architecture
(Agile 101)**

**16 October 2013
(last update)**

Rick Dove

**Taos County, New Mexico, dove@parshift.com, 575-586-1536
PI/CEO, Paradigm Shift International
Adjunct Professor, Stevens Institute of Technology**

**Download latest version: www.parshift.com/s/AgileSystems-101.pdf
(updated asynchronously from time-to-time)**

Agile Systems and Processes: Necessary and Sufficient Fundamental Architecture

Abstract: Agility is enabled and maintained by a fundamentally necessary and sufficient common architecture in systems of all kinds; from agile development and deployment, to the agile systems and products that are deployed. This webinar will focus on that common architecture, and how it enables both reactive resilience and proactive innovation. Examples will include quick reaction capability in aircraft major instrumentation refurb, project management in two different domains, and enterprise IT; and the processes known as lean and agile software development will be shown to employ the same enabling architecture, with the recent lean concept-overlays streamlining the delivery of agile development processes.

Bio: Rick Dove was co-PI on the original work which identified Agility as the next competitive differentiator, funded by the US Office of the Secretary of Defense through the Navy in 1991 at Lehigh University. He went on to organize and lead the US DARPA-funded industry collaborative research at Lehigh University's Agility Forum, developing fundamental understandings of what enables and characterizes systems agility. He authored *Response Ability – The Language, Structure, and Culture of Agile Enterprise* (Wiley, 2001). He has employed these agile concepts in both the architecture and program management for large enterprise IT systems, for rapid manufacturing systems and services, and for highly distributed resilient network anomaly detection. Through Stevens Institute of Technology he teaches two 40-hour graduate courses in basic and advanced agile-systems and agile systems-engineering, at client sites. Through Paradigm Shift he provides training workshops and strategy development services. He chairs the INCOSE working groups on Agile Systems and Systems engineering, and on Systems Security Engineering.

Objective: System X-Ray Vision



<http://awespendo.us/animemangacomics/kermit-at-the-doctor/>

**Please reserve questions until after the presentation.
Slides are numbered if you want to reference them in your questions.**

Content

Background: Origins of Agile System and Process Thinking

Case: Home Entertainment System

Case: QRC Aircraft Refurb

Case: Last Planner Project Management

Case: Enterprise IT with Agile ERP Developed System

Case: Enterprise IT with Agile ERP Development System

Case: Agile Software Development, with new Lean Thinking

Precedence: echoes in biology and complex systems

References

This seminar generally opens a workshop, with learning exercises:

Full-Day Workshop with Exercises (3-4 Hr + collaborative group brief out)

Half-Day Workshop with Exercises (1-2 Hr + collaborative group brief out)

Today's Agility Interest – Origin & Continuation

- 1991** – 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)
 - 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
- Since then** – Confirmation & employment in various projects
 - Many graduate SE student term and masters projects
 - Refinement of architectural concepts, no basic changes

Agile-Systems Research Focus

Problem:

- Technology and markets are changing faster than the ability to employ/accommodate
- Life cycle requirements 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

Agility - Fundamentally

The Ability to Thrive in a Continuously Changing, Unpredictable Environment.

Agility is *effective response* to opportunity and problem,
within mission ... always ... no matter what.

An *effective response* is one that is:

- | | |
|--|---------------|
| ■ timely (fast enough to deliver value), | <u>Metric</u> |
| ■ affordable (at a cost that leaves room for an ROI), | time |
| ■ predictable (can be counted on to meet expectations), | cost |
| ■ comprehensive (anything/everything within mission boundary). | quality |
| | scope |

You can think of Agility as Requisite Variety.

You can think of Agility as proactive Risk Management.

You can think of Agility as Innovative Response in unpredictable situations.

You can think of Agility as Life Cycle Extension.

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

Domain Independent

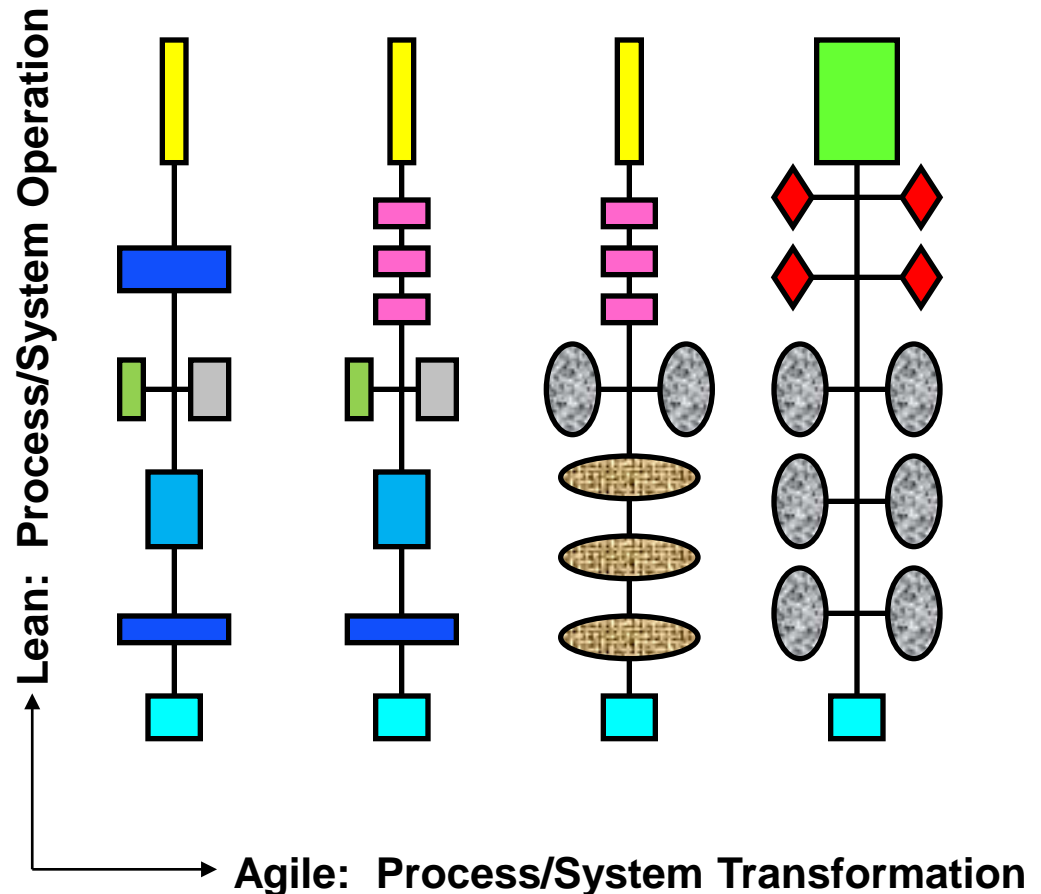
Lean & Agile: Orthogonal Focus

Agility deals with
“design-for-transformation”.

In a very general
interpretation,

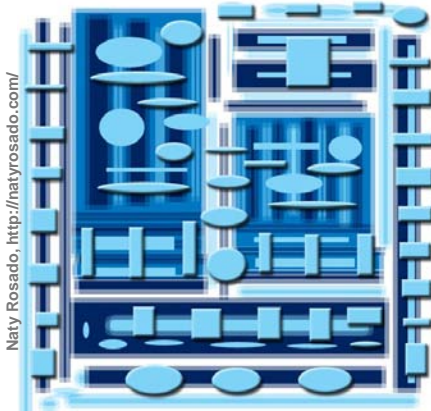
Lean values efficiency of
operation and achieves this
mainly through operational
principles;

Agile values effective
response ability and achieves
this mainly through
architectural principles.



Both are concerned with operational effectiveness. Since the two have a different means for achieving different ends they are not necessarily in one-or-the-other conflict – but can be.

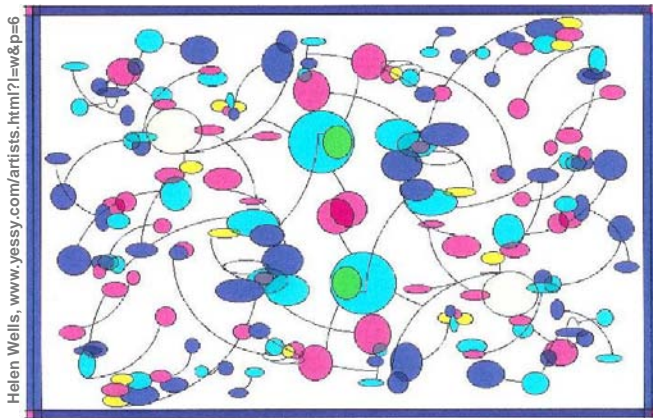
Class 1 Agile Systems are Reconfigurable



Useful Metaphors:

**Plug-and-Play – Drag-and-Drop
(focus of this presentation)**

Class 2 Agile Systems are Reconfiguring



Useful Metaphors:

**Ecologies and Evolution
(not in this presentation)**

Both employ the same fundamental agile architecture pattern

RAP* – 7 Thought-Guiding Frameworks

Response requirements categories (4 reactive and 4 proactive elements):

Reactive: correction, variation, expansion, reconfiguration

Proactive: creation, improvement, migration, modification

Response performance metrics (4 elements):

Response: cost, time, quality, scope

Response-enabling design principles (10 elements):

Encapsulation, Compatibility, Reusability, Redundancy/Diversity, Scalability, Distributed, Loose, Deferred Commitment, Self-Organizing, Evolving Standards

Design quality principles (3 elements):

Requisite Variety, Parsimony, Harmony

An overarching architectural philosophy (3 elements):

Reusable modules Reconfigurable in a Scalable architecture (RRS)

Sustainable agility responsibilities (4 elements):

Module Inventory, System Re-configuration

Module Evolution, Infrastructure Evolution

An agile architecture pattern:

Drag-and drop modules in a plug-and-play infrastructure

***RAP: Response Ability Principles**

RAP – 7 Thought-Guiding Frameworks

Response requirements categories (4 reactive and 4 proactive elements):

Reactive: correction, variation, expansion, reconfiguration

Proactive: creation, improvement, migration, modification

Response performance metrics (4 elements):

Response: cost, time, quality, scope

Response-enabling design principles (10 elements):

Encapsulation, Compatibility, Reusability, Redundancy/Diversity, Scalability, Distributed, Loose, Deferred Commitment, Self-Organizing, **Evolving Standards**

Design quality principles (3 elements):

Requisite Variety, Parsimony, Harmony

fundamentals focus

An overarching architectural philosophy (3 elements):

Reusable modules Reconfigurable in a Scalable architecture (RRS)

Agility-sustaining responsibilities (4 elements):

Module Readiness,

System Re-configuration

Module Mix Evolution,

Infrastructure Evolution

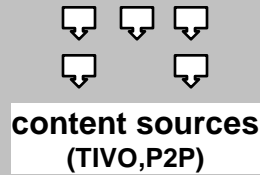
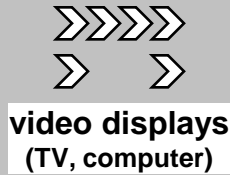
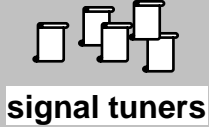
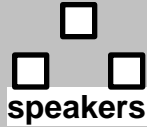
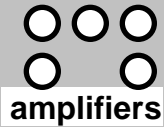
An agile architecture pattern:

Drag-and drop modules in a plug-and-play infrastructure

Case: Home Entertainment Technology Migration

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

Encapsulated Modules



Drag-and-Drop
Reusable
Components

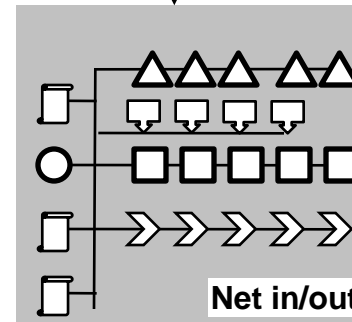
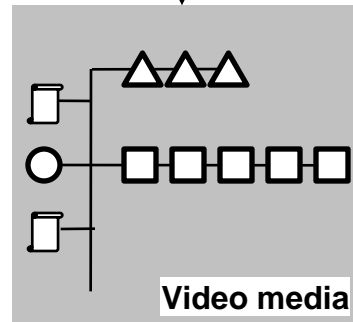
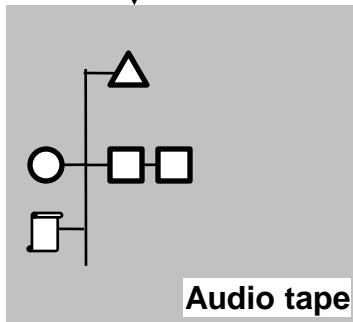
Case: Home Entertainment Technology Migration

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

Encapsulated Modules



Drag-and-Drop
Reusable
Components



Examples of Typical
Reconfigurable/Scalable
System Configurations

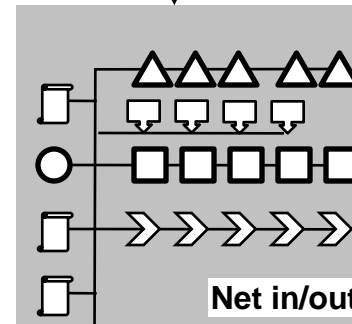
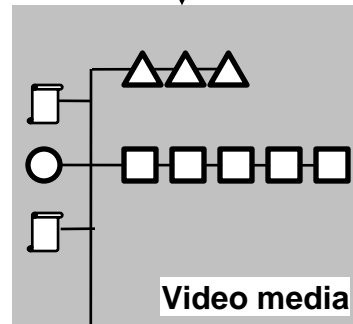
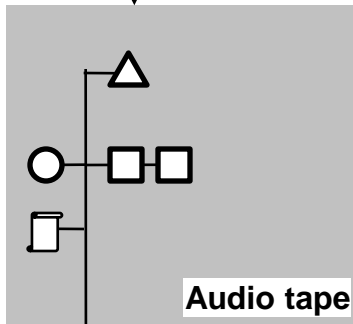
Case: Home Entertainment Technology Migration

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

Encapsulated Modules



Drag-and-Drop
Reusable
Components



Examples of Typical
Reconfigurable/Scalable
System Configurations

Plug-and-Play Evolving
Passive Infrastructure
Rules/Standards/Principles

'40s/'50s

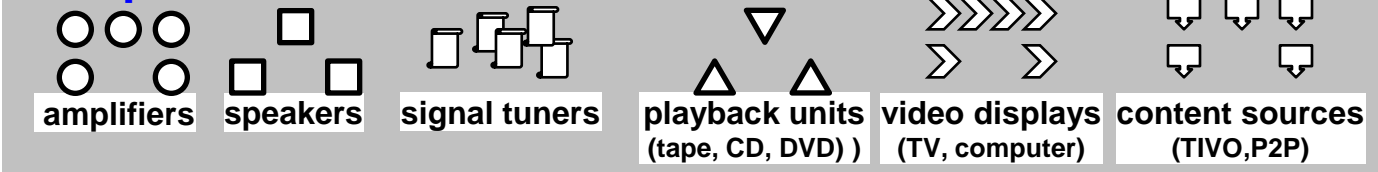
'90s

'00s

Case: Home Entertainment Technology Migration

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

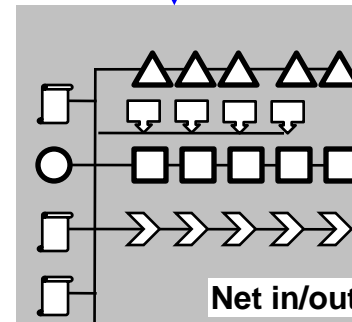
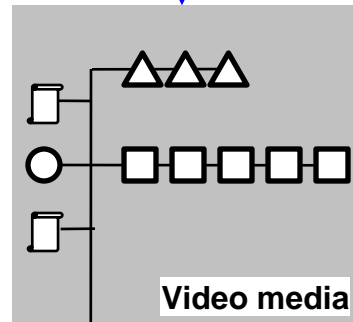
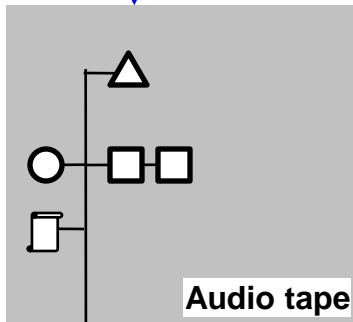
Encapsulated Modules



Drag-and-Drop
Reusable
Components

Plug-and-Play Evolving
Active Infrastructure
Responsible-Parties

Assembly User/Owner



Examples of Typical
Reconfigurable/Scalable
System Configurations

Plug-and-Play Evolving
Passive Infrastructure
Rules/Standards/Principles

Video/Surround

Digital/Internet

'40s/'50s

'90s

'00s

Case: Home Entertainment Technology Migration

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

Encapsulated Modules



Drag-and-Drop
Reusable
Components

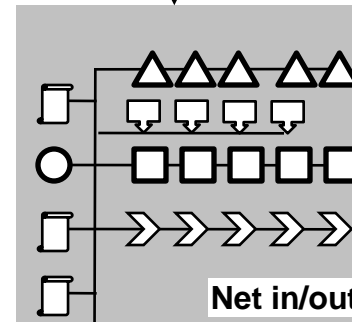
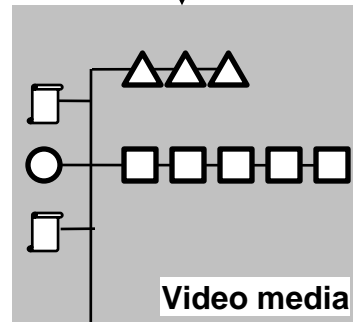
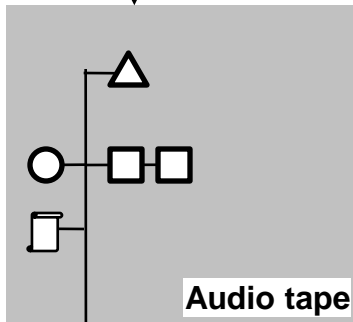
Readiness

Stores

Assembly

User/Owner

Plug-and-Play Evolving
Active Infrastructure
Responsible-Parties



Examples of Typical
Reconfigurable/Scalable
System Configurations

Video/Surround

Digital/Internet

Plug-and-Play Evolving
Passive Infrastructure
Rules/Standards/Principles

'40s/'50s

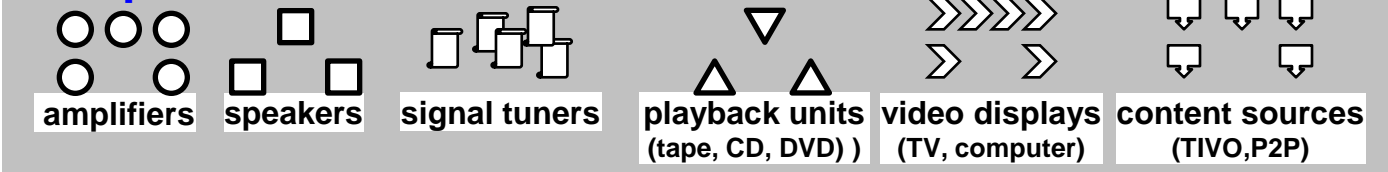
'90s

'00s

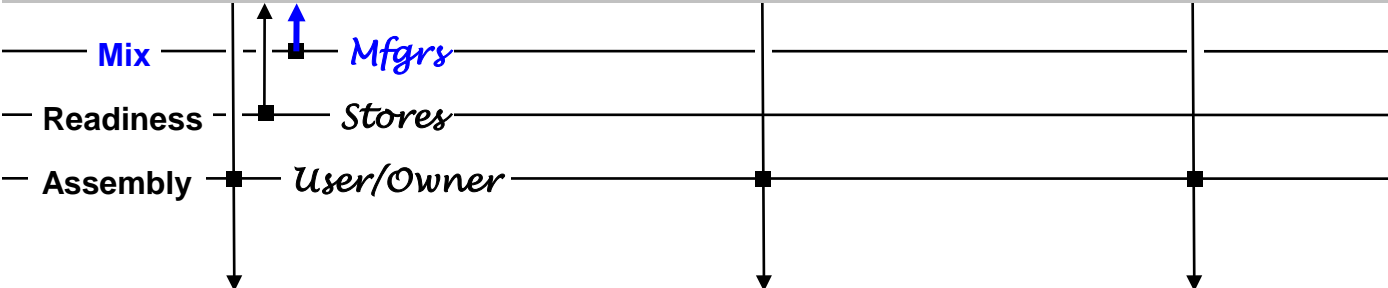
Case: Home Entertainment Technology Migration

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

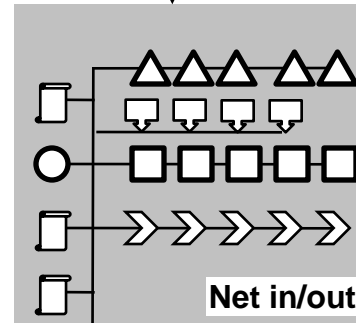
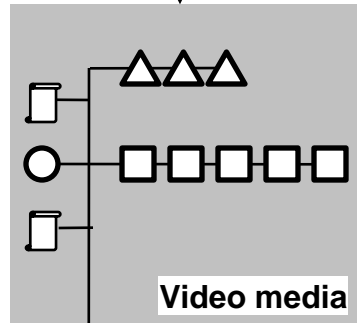
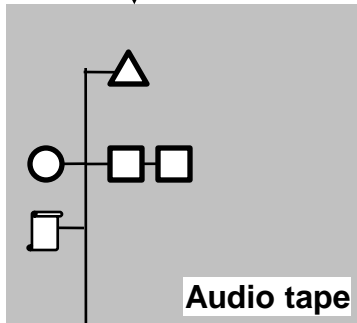
Encapsulated Modules



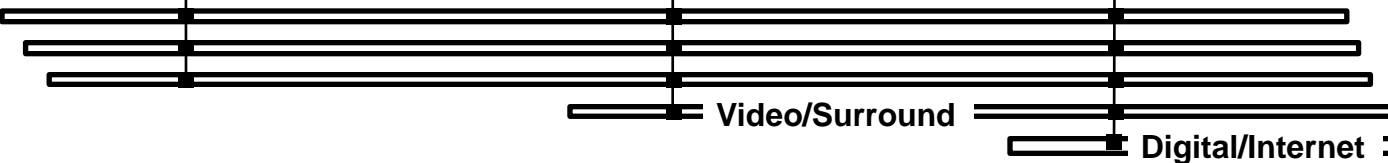
Drag-and-Drop
 Reusable
 Components



Plug-and-Play Evolving
 Active Infrastructure
 Responsible Parties



Examples of Typical
 Reconfigurable/Scalable
 System Configurations



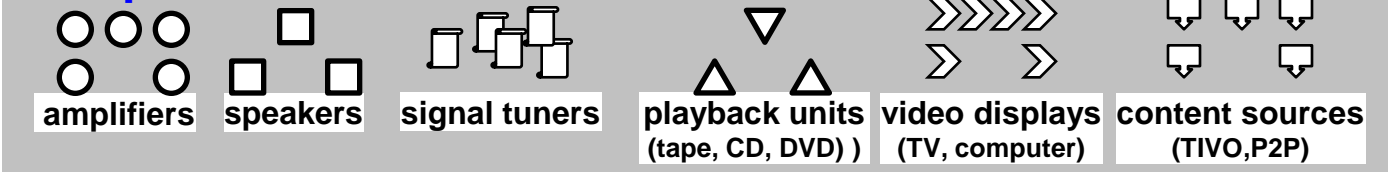
Plug-and-Play Evolving
 Passive Infrastructure
 Rules/Standards/Principles

'40s/'50s '90s '00s

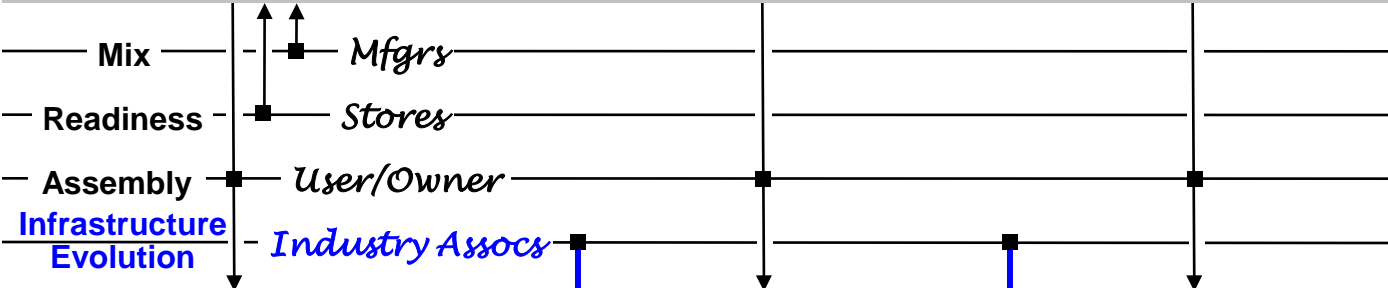
Case: Home Entertainment Technology Migration

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

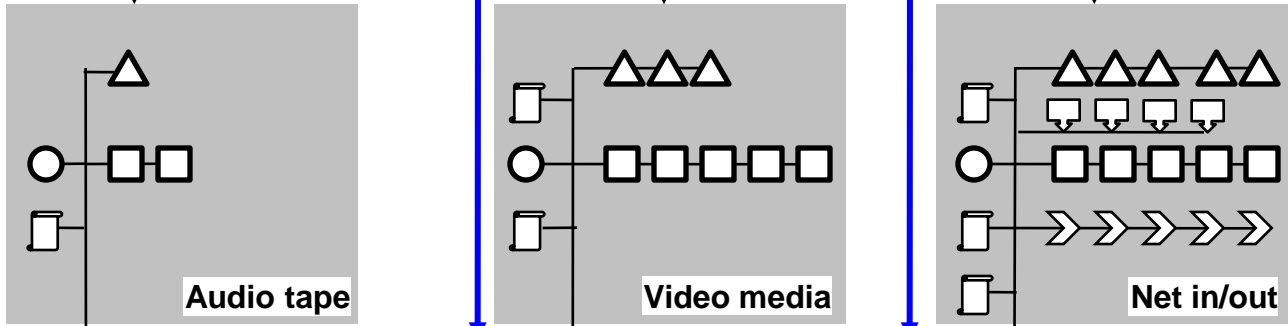
Encapsulated Modules



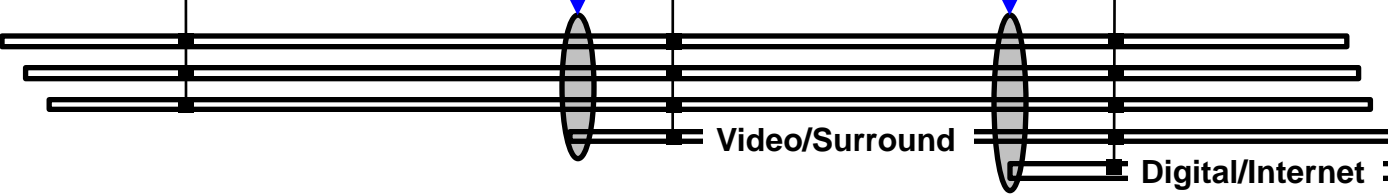
Drag-and-Drop
Reusable
Components



Plug-and-Play Evolving
Active Infrastructure
Responsible Parties



Examples of Typical
Reconfigurable/Scalable
System Configurations



Plug-and-Play Evolving
Passive Infrastructure
Rules/Standards/Principles

'40s/'50s '90s '00s

Fundamental Concept

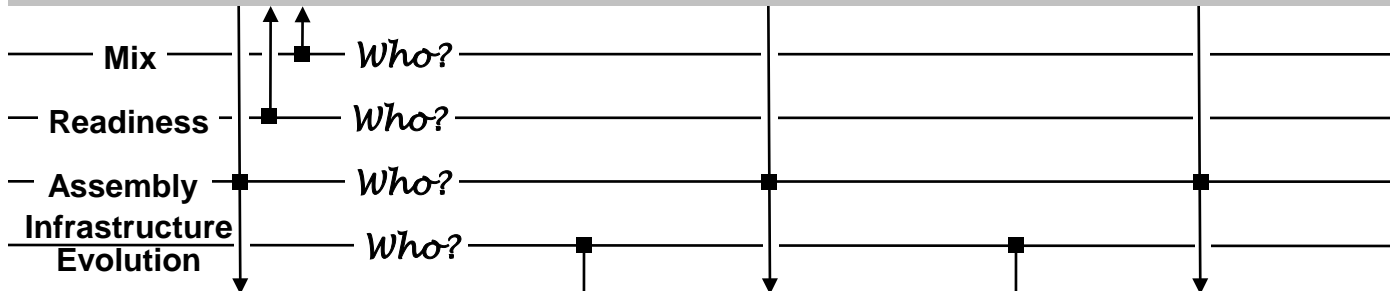
Reusable modules Reconfigurable in a Scalable architecture (RRS)

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

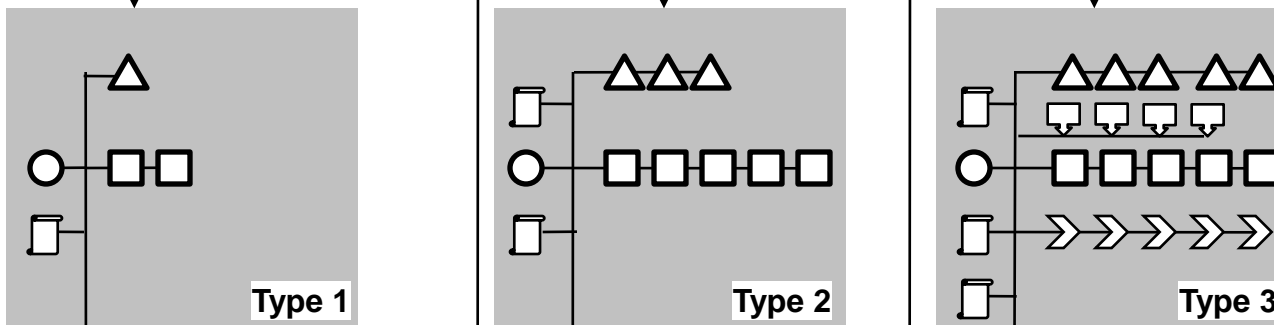
Encapsulated Modules



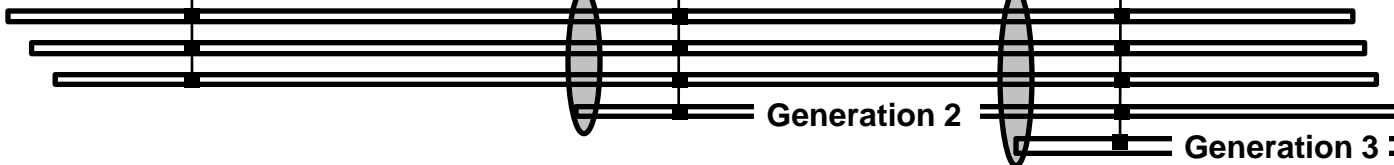
Drag-and-Drop
Reusable
Components



Plug-and-Play Evolving
Active Infrastructure
Responsible-Parties



Examples of Typical
Reconfigurable/Scalable
System Configurations



Plug-and-Play Evolving
Passive Infrastructure
Rules/Standards/Principles

Variety/Time/Maturity/Range/Increments/Migrations/Evolutions/etc →

Case: Aircraft Refurb QRC

Jason Boss masters project, Agile Aircraft Installation Architecture In a Quick Reaction Capability Environment, INCOSE IS10, Chicago, July 12-15.
www.parshift.com/Files/PsiDocs/Pap100712IS10-AgileAircraftInstallationArchitecture.pdf

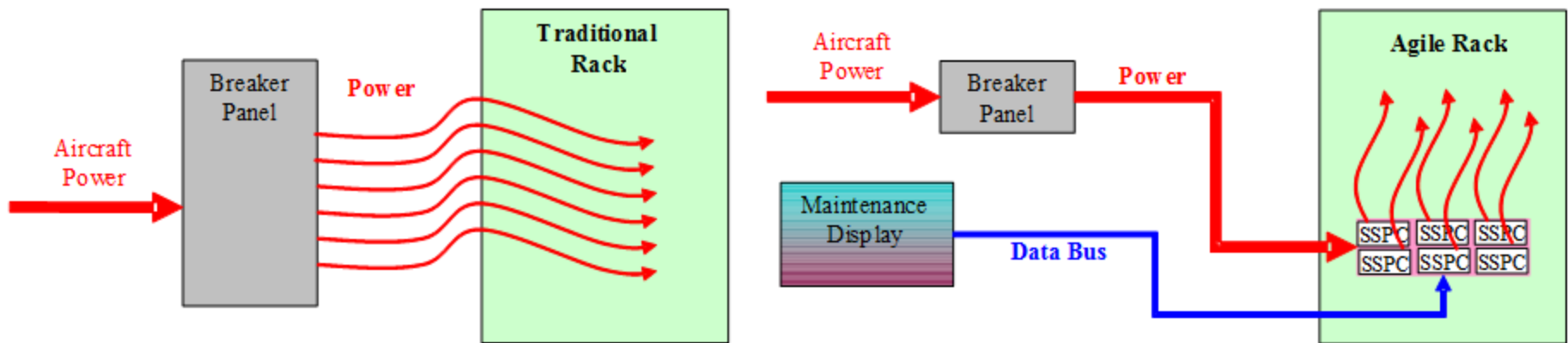
- ❑ **Mission system installation in military acquisition context.**
- ❑ **Customer's need for the latest technology.**
- ❑ **Technology advances are creating new mission systems at an increasing rate, driving the demand for QRC.**
- ❑ **Goal is to shorten the completion time without compromising quality.**
- ❑ **Mission requirements and “boxes” often change late.**
- ❑ **Army wants QRC for intelligence surveillance reconnaissance (ISR) to be robust, scalable, tailorable.**
- ❑ **Air Force wants QRC challenges continually met, success is measured in rapidly adapted Electronic Warfare.**

Example: Agile vs. Traditional Power Distribution

Traditionally a breaker centralized panel distributes power to each box, creating an interface for every box and many wire routing paths. Some aircraft contain over 1000 boxes, and wire routing becomes a large modification effort.

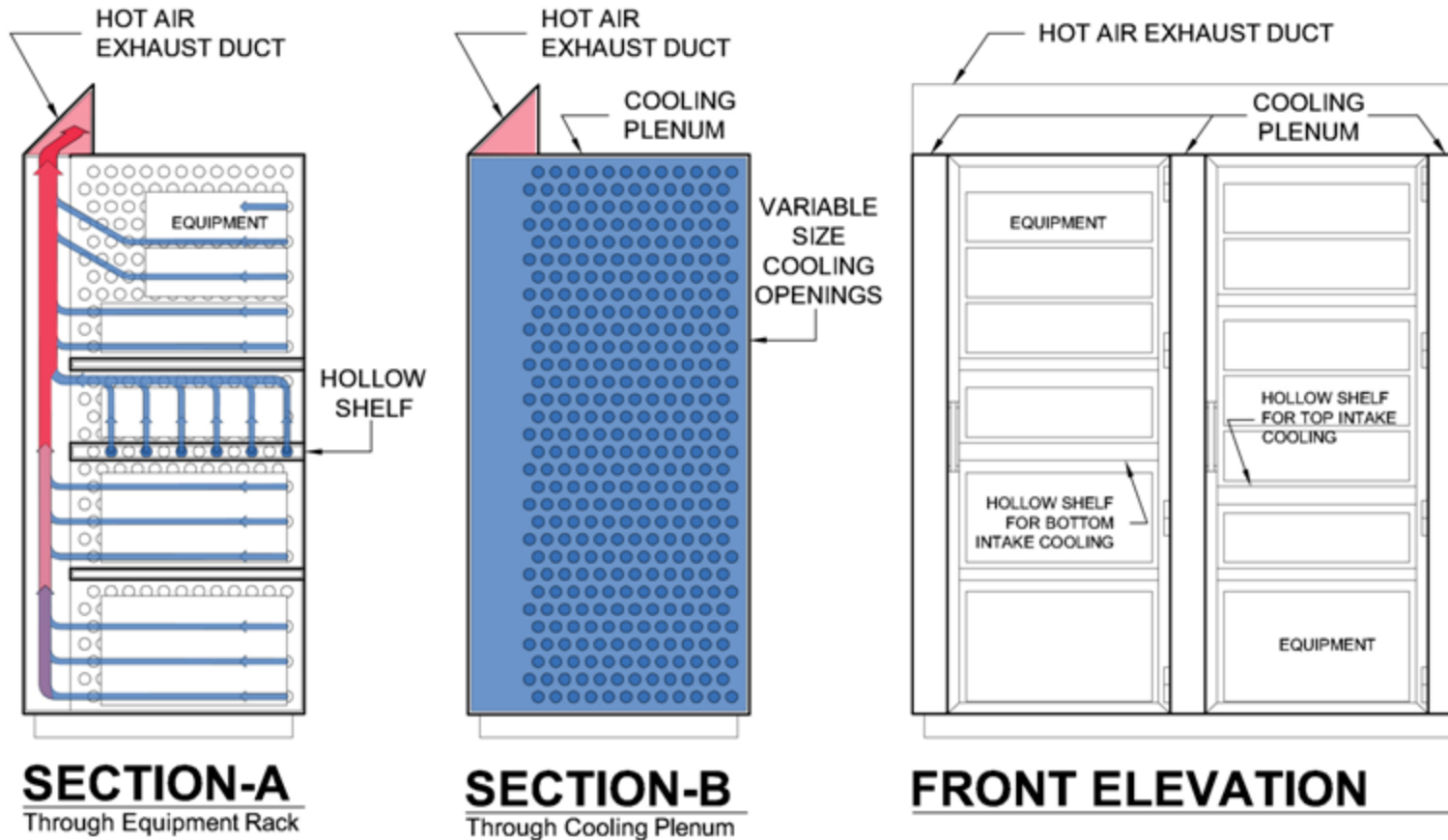
To reduce the number of interfaces, decrease wire routing effort, and allow rack modularity, the power distribution can be moved from the aircraft to within the rack itself.

A single breaker then provides power to the rack, and a secondary breaker panel within the rack would distribute power to each box. Remote controlled solid state power controllers (SSPCs) allows re-programming an SSPC instead of changing a breaker out and routing a new wire between the breaker box and the rack.



Rack becomes an encapsulated module. Power infrastructure is minimal.

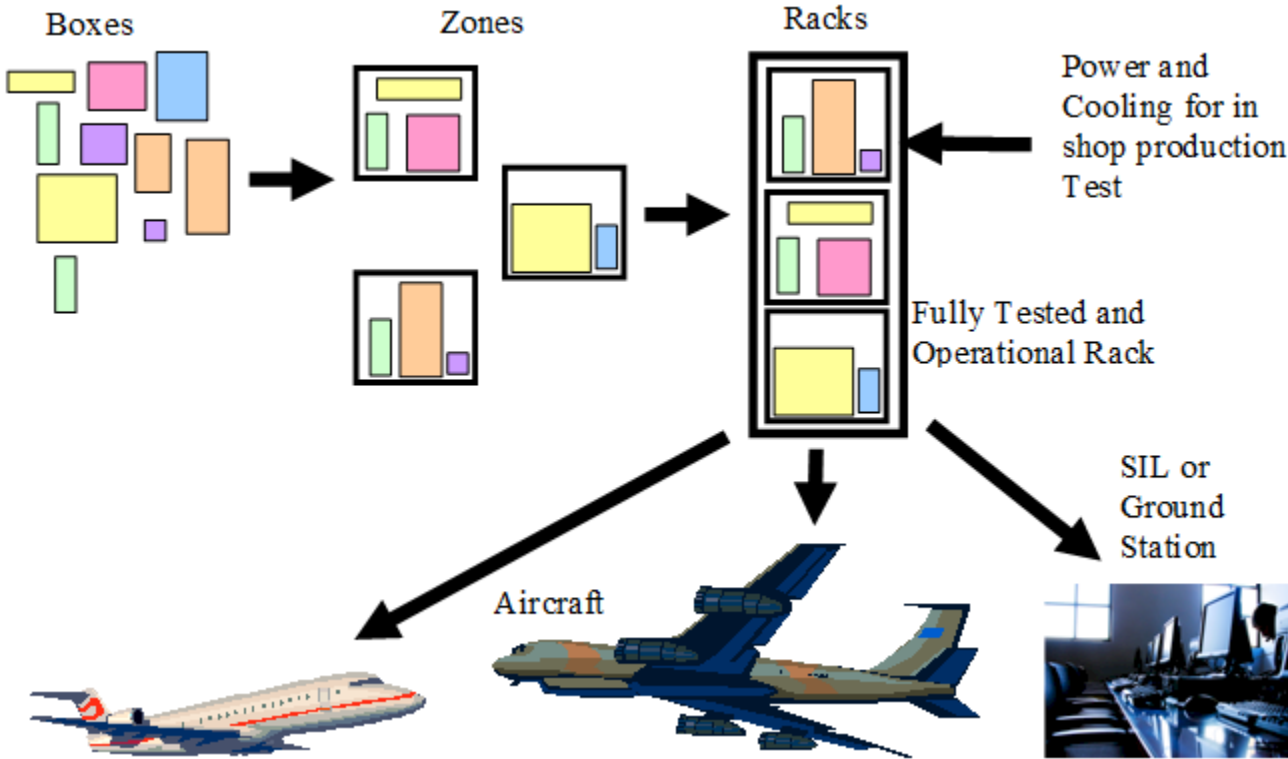
Example: Modular Rack Cooling



Solution mitigates the rerouting effort of existing aircraft ductwork. The proposed cooling architecture is really a combination of a cold air distribution subsystem that gets cold air from the aircraft source to the boxes, and a hot air exhaust subsystem that must dispose of the waste air.

Rack becomes an encapsulated module. Cooling infrastructure is minimal.

Encapsulated Modules, Minimal Infrastructure



Aircraft installation infrastructure is modified... once.

The SIL* has a duplicate infrastructure.

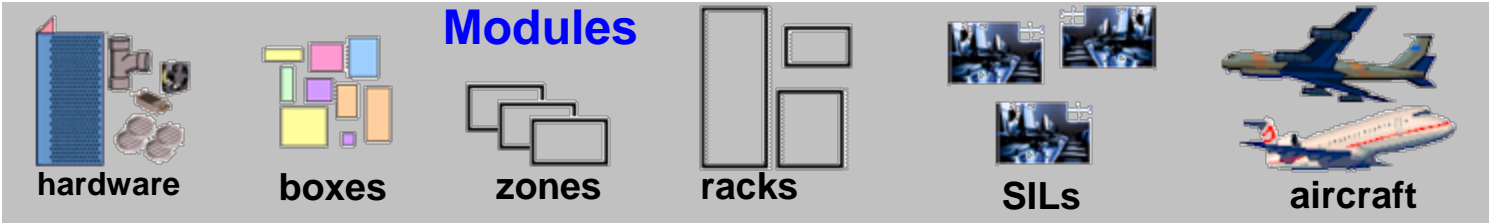
“Everything” is fully integrated and tested in the SIL ... before the aircraft arrives.

Aircraft installation is a simple relocation of pluggable modules.

Minimizes aircraft downtime and eliminates custom installation work.

Parameter	Nature of Standard	*SIL: System Integration Lab
Space	Racks shall be designed in preset widths, depths and heights.	
Power	Each rack shall have a maximum kW equipment load rating. Racks with multiple power types (e.g. 115 VAC 400 Hz and 28 VDC) limits should be set on each type.	
Weight	Each rack shall have a maximum equipment weight rating.	
Cooling	Each rack shall rate the kW cooling capacity at a specified exhaust temperature.	
Physical Interfaces	Rack mounting provisions, cooling connections, and electrical connection interfaces shall have standard locations and configurations.	

QRC Aircraft Installation – Agile Architecture

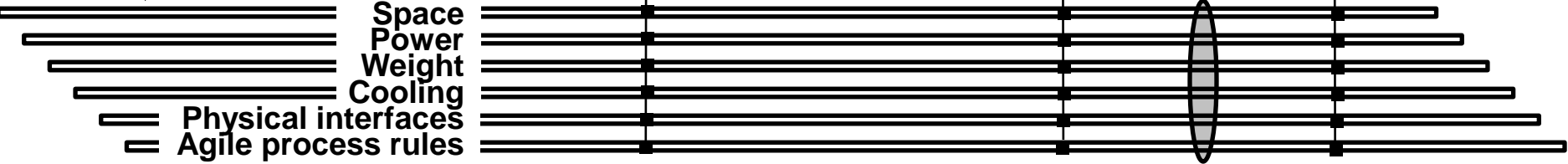
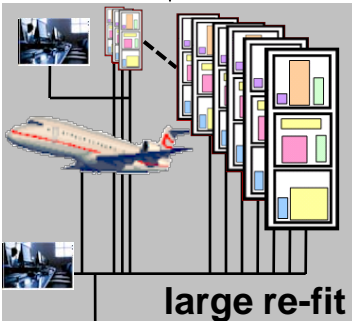
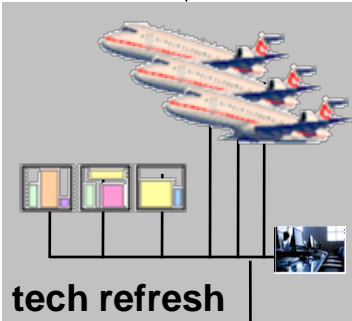
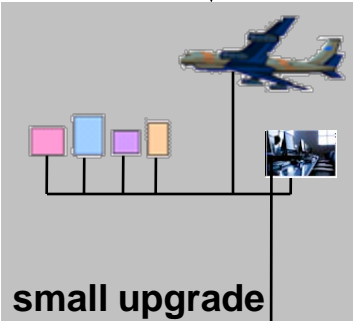


Integrity Management

- Module pool & mix evolution
 - Module inventory condition
 - Assembly in SIL
 - Infrastructure evolution
- system engineer
 - material manager
 - production
 - process engineer

Active Infrastructure

Passive



Rules/Standards

A Construction Project Case Study Based on the “Last Planner System” by Glenn Ballard

Lean and Agile Project Management
[www.parshift.com/AgileSysAndEnt/Cases/Case Last Planner System.pdf](http://www.parshift.com/AgileSysAndEnt/Cases/Case%20Last%20Planner%20System.pdf)

**Creating Options
Reconfigurable Task Schedules
Deferred Assignment Commitments
Proactive Expediting**

“When environments are dynamic and the production system is uncertain and variable, reliable planning cannot be performed in detail much before the events being planned.

“Consequently, deciding what and how much work is to be done next by a design squad or a construction crew is rarely a matter of simply following a master schedule established at the beginning of the project. [pages 3-15 and 3-16 of Ballard Thesis]

**Herman Glenn Ballard
Director of Research, Lean Construction Institute, and Lecturer, Construction
Engineering and Management Program, Dept. of Civil and Environmental Engineering,
University of California at Berkeley, 4536 Fieldbrook Road, Oakland, CA 94619,
510/530-8656, FAX 510/530-2048, ballard@ce.berkeley.edu**

Traditional Task Selection from Master Schedule

A key early finding was that only about half of the assignments made to construction crews at the beginning of a week were completed when planned.

Experiments were performed to test the hypothesis that failures were in large part a result of lack of adequate work selection rules (these might also be called work release rules).

Task Selection Method Addressing Schedule Uncertainty

Quality criteria were proposed for assignments regarding definition, sequence, soundness, and size.

In addition, the percentage of assignments completed was tracked (PPC: percent plan complete) and reasons for noncompletion were identified, which amounted to a requirement that learning be incorporated in the control process.

Quality Criteria for Work Assignment

Definition: Are assignments specific enough that the right type and amount of materials can be collected, work can be coordinated with other trades, and it is possible to tell at the end of the week if the assignment was completed?

Soundness: Are all assignments sound, that is: Are all materials on hand? Is design complete? Is prerequisite work complete? Note: During the plan week, the foreman will have additional tasks to perform in order to make assignments ready to be executed, e.g., coordination with trades working in the same area, movement of materials to the point of installation, etc. However, **the intent is to do whatever can be done to get the work ready before the week in which it is to be done.**

Sequence: Are assignments selected from those that are sound in the constructability order needed by the production unit itself and in the order needed by customer processes? Are additional, lower priority assignments identified as workable backlog, i.e., additional quality tasks available in case assignments fail or productivity exceeds expectations?

Size: Are assignments sized to the productive capability of each crew or subcrew, while still being achievable within the plan period? Does the assignment produce work for the next production unit in the size and format required?

Learning: Are assignments that are not completed within the week tracked and reasons identified?

As a result of applying these criteria, plan reliability (the percentage of assignments completed) increased, and with it, crew productivity also increased (Ballard and Howell, 1997)¹⁶.

¹⁶ On the whole, improvements tended to be from PPC (percent plan complete) levels around 50% to the 65-70% level, with a corresponding **increase of 30% in productivity**. Productivity improvement has ranged from 10% to 40%+.

Rules and Objectives Established

A set of rules was proposed for allowing scheduled activities to remain or enter into each of the three primary hierarchical levels of the scheduling system:

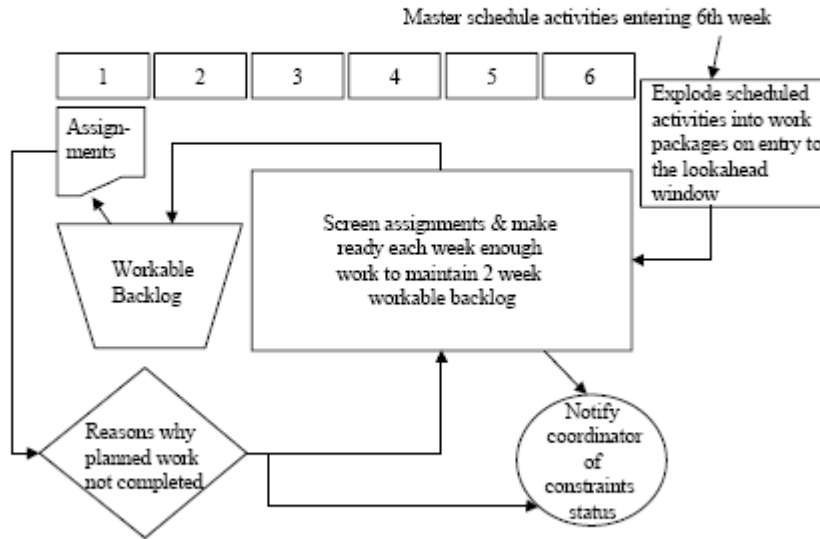
- Rule 1: Allow scheduled activities to remain in the **master schedule** unless positive knowledge exists that the activity should not or cannot be executed when scheduled.
- Rule 2: Allow scheduled activities to remain in the **lookahead window** only if the planner is confident that the activity can be made ready for execution when scheduled.
- Rule 3: Allow scheduled activities to be released for selection into **weekly work plans** only if all constraints have been removed; i.e., only if the activity has in fact been made ready.

In addition, a set of objectives was proposed for the lookahead process:

- ❑ Shape work flow sequence and rate
- ❑ Match work flow and capacity
- ❑ Decompose master schedule activities into work packages and operations
- ❑ Develop detailed methods for executing work
- ❑ Maintain a backlog of ready work

Figure 3.3

The Lookahead Process: Make Ready by Screening & Pulling



Make Ready by Screening and Pulling

Figure 3.3 is a schematic of the lookahead process, showing work flowing through time, right to left.

Potential assignments enter the lookahead window 6 weeks ahead of scheduled execution, then move forward a week each week until they are allowed to enter into workable backlog, indicating that all constraints have been removed and that they are in the proper sequence for execution.

If the planner were to discover a constraint ... that could not be removed in time, the assignment would not move forward.

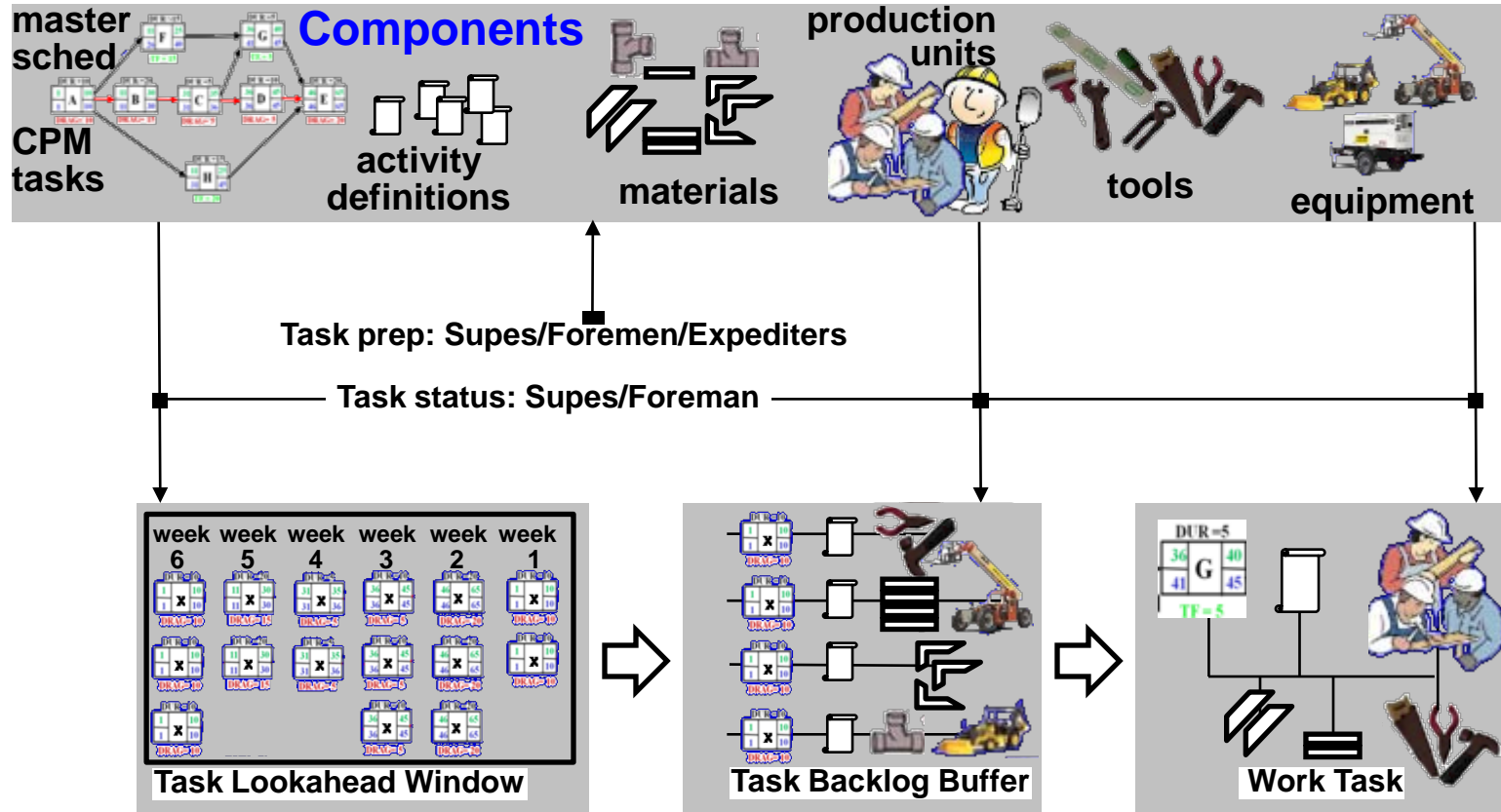
The objective is to maintain a backlog of sound work, ready to be performed, with assurance that everything in workable backlog is indeed workable.¹³ **Weekly work plans are then formed from workable backlog**, thus improving the productivity of those who receive the assignments and increasing the reliability of work flow to the next production unit.

¹³ **Deliberately building inventories, inventories of ready work in this case, may seem contradictory to the goals of just-in-time. To clarify, inventories of all sort are to be minimized, but as long as there is variability in the flow of materials and information, buffers will be needed to absorb that variability. Reducing variability allows reduction of buffer inventories.**

Last Planner Work Flow Management

www.parshift.com/s/130624Last Planner.pdf

Active management of the anticipated schedule and work flow to ensure there is always a buffer of “quality” jobs ready to work on and matched with resources.



Tasks enter lookahead window 6 weeks in advance of execution schedule, advancing according to readiness, with action on prep for execution.

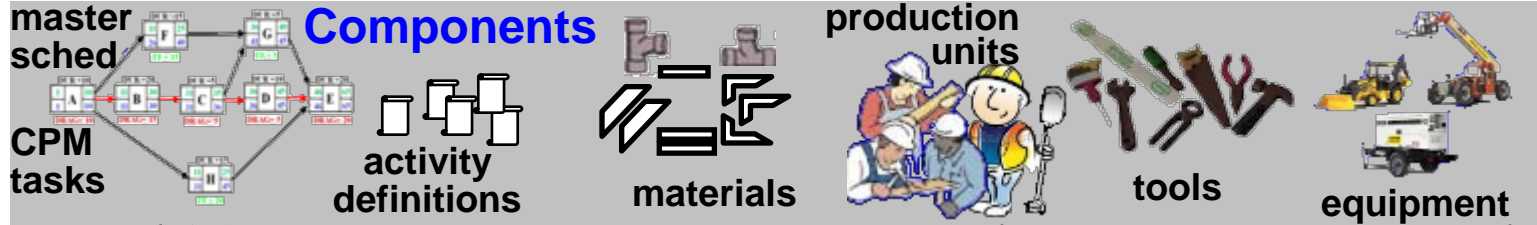
Tasks enter backlog whenever all necessary elements are ready for execution.

Weekly work tasks are drawn from readiness backlog, keeping crews fully employed.

Last Planner Agile Project Management

www.parshift.com/s/130624Last Planner.pdf

Active management of the anticipated schedule and work flow to ensure there is always a buffer of “quality” jobs ready to work on and matched with resources.

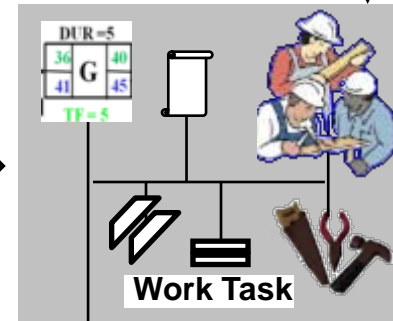
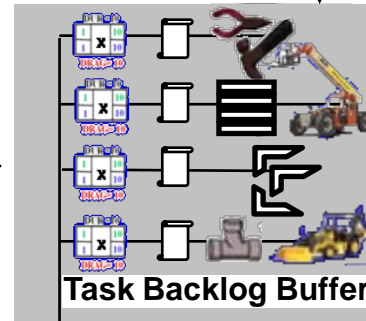
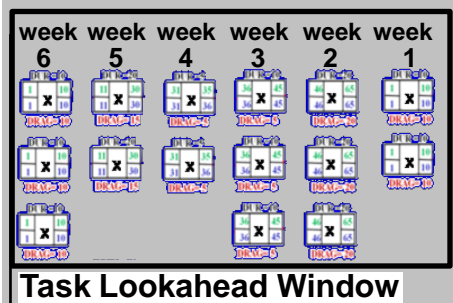


Integrity Management

- Task elements: Project Manager
- Task readiness: Supes/Foremen/Expediter
- Task assembly: Supes/Foreman
- Infrastructure evolution: Last Planner Process Manager

Key Practices:
 Rules 1-2-3 and
 • Lookahead
 • Make ready
 • Learn & Correct

Active
 ↑
Infrastructure
 ↓
 Passive



- QR1: Definition
 - QR2: Soundness
 - QR3: Sequence
 - QR4: Size
- QR = Task Quality Requirement

Standards

Agile architecture Pattern based on:
 (Ballard 1997) Lookahead Planning: the Missing Link in Production Control
 (Ballard 1998) Shielding Production: an Essential Step in Production Control
 (Ballard 1999) Improving Work Flow Reliability
 (Ballard 2000) The Last Planner System of Production Control-PhD Thesis

RRS Principles – two are necessary the other eight are amplifiers

Encapsulated Modules

- 1:1 physical/functional packaging
- Black box to other modules
- Functional methods can change, but interface protocols cannot

Evolving Minimal Standards (Infrastructure)

- Defines module-interface protocols/standards (and operating rules)
- Enables and constrains agility
- Delicate balance of requisite variety and parsimony

Encapsulated Modules	Reusable	Scalable	Evolving Minimal Standards
Plug Compatibility (Facilitated Interfacing)			Redundancy and Diversity
Facilitated Reuse			Elastic Capacity
Reconfigurable			
Flat Interaction	Distributed Control and Information		
Deferred Commitment	Self-Organization		

Case: Silterra: Malaysian Semiconductor Foundry

Rick Dove. 2005. Fundamental Principles for Agile Systems Engineering. Conference on Systems Engineering Research (CSER), Stevens Institute of Technology, Hoboken, NJ, March. www.parshift.com/Files/PsiDocs/Rkd05032.pdf

October 1999 (dot.com bubbling, semiconductor slump ending).

Silterra is a start-up semiconductor foundry in Malaysia, with interim USA top management and ex-pat process experts.

Funded mainly by government designated sources.

Mixed Cultures: 60% Malay, 30% Chinese, 10% Indian.

Few employees have built or run such a company, and have little idea about what they will need or want in business processes.

CEO has a vision for a preemptive modern-day competitor...

Goal: Build a uniquely superior foundry business.

Strategy: Best practices + Agile IT infrastructure.

CIO (interim exec) is writing book on systems agility...

Goal: Meet CEO's goals with Agile Systems design principles.

Strategy: Design a differentiation strategy and apply principles.

Opportunity

New company:

No operating culture, performance metrics, or infrastructure legacy.

+

New technology:

Internet. Broadband. PDAs. XML. Enterprise IT. eBusiness.

+

New environment:

More uncertain, connected, knowledgeable. Faster. Always changing.

+

New customer expectations:

Personal attention. Immediate response. Self service.
Lots of information.

= New Opportunity

to design a company IT support system
fit to the new and changing environment,
and focused on new values

Objectives

**Supporting strategy with best-fit tools
is enabled rather than inhibited**

**Switching/upgrading to new technology and applications
is enabled rather than inhibited.**

**Accommodating custom electronic "partner" relationships
is enabled rather than inhibited.**

**Integrating new plants, facilities, mergers, and acquisitions
is enabled rather than inhibited.**

**All information is accessible electronically
to those authorized to see it.**

**Electronic "dashboards" will provide real-time vision and monitoring
of operational and strategic activities.**

**Provide competitive advantage through
enterprise visibility, adaptability, and latest technology**

General Strategy

Business System Analyst (BSA) Group:

- Assigned to IT-assist dept managers (cross dept responsibilities)
- Business Process IT application configuration/evolution
- IT tool selection/acquisition

Strategic System Analyst (SSA) Group:

- Evolution of infrastructure framework
- Enforcing infrastructure usage rules

User Collaboration:

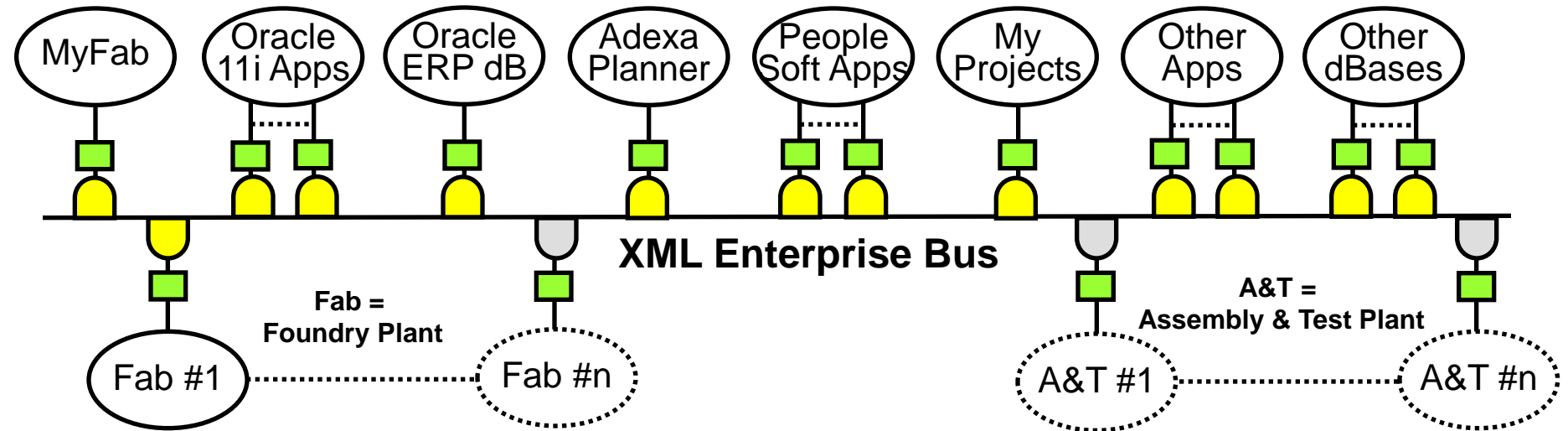
- Mandatory Response Situation Analysis (agility-tool)



COTS Applications: No customization of purchased software

IT Internal Responsibilities – not to be outsourced:

- Infrastructure architecture design and evolution
- Management of installation/integration projects
- Configuration of applications

Enterprise IT-Infrastructure Design

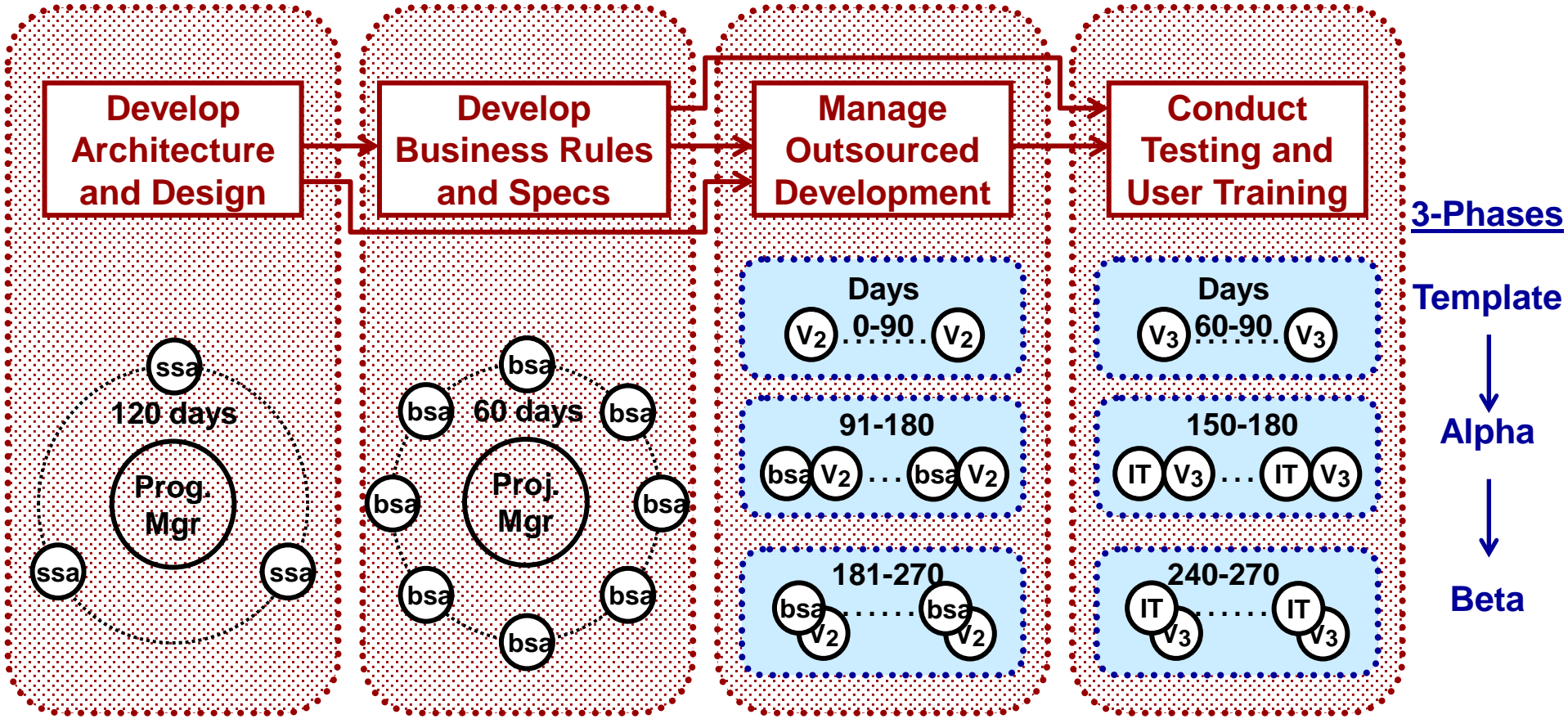


-  = Bus Interface Module (BIM)
-  = Extract/Transfer/Load (ETL) Interface Modules
- MyProjects = Web-accessible strategic-project portfolio manager
- MyFab = Web-accessible operations transparency

Project Development Process – Strategy/Rules

- Vendor is responsible for total solution: HW and SW
- Requirements will not change during implementation
- No expedient customization allowed
- Three Phase Implementation Sequence:
 - P1: Out-of-box best-practice from vendor – supporting the company
Vendors configure the applications
 - P2: BSA-developed business process rules
Vendors + BSAs configure the applications
 - P3: Refined business processes
BSAs configure the applications
- No violation of infrastructure rules (repeatedly invoked)
- Don't say it can't be done, tell what is needed to do it (repeatedly invoked)

Encapsulated Development Process



- Designed to Accommodate Requirements Evolution -

Effective Predictability

ERP on time, below budget, on spec

- **3 months functional ERP "best practice" (Phase 1)**
- **3 months later preferred business processes (Phase 2)**
- **3 months later refined business processes (Phase 3)**

**HRM modularized and
added below time, on budget, on spec**

**Adexa planner
added on time/budget/spec**

**Existing Time and Attendance system
modularized and integrated on time/budget/spec**

<u>Wish</u>	<u>Typical Imp</u>	<u>Actual Imp</u>
ERP in 12 mos total	24-36 mos	12 ^{1,2}
75% of license budget	200-300%	75%
\$10 Million (5 + 5)	\$15-25 Million	\$9 Million
HRM in 6 mos	12-18 mos	5 mos

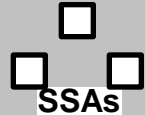
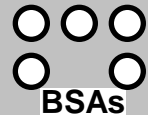
HOW??

- Principle-based installation/integration methodology and management
- Adherence to methodology (ie, effective management)
- BSAs utilizing MBW tool to develop and capture business processes
- BSAs taking responsibility for integrating ERP with users
- Bus architecture connecting ERP with HRM
- Experienced outsource to help integrate ERP/CIM^{2,3} (did it before)
- Expertise in agile system design and implementation

Notes: 1) 12 months = 3 mo concept design and vendor selection + 9 mo implementation, time included infrastructure bus/ETL/BMI implementation, but not shop floor (CIM) integration (+6)
2) New Oracle 11i ERP with typical bugs and lack of documentation of new systems
3) Additional 6 mos due to independent CIM system shake out

Silterra Agile ERP – Development System

Components/Modules



Integrity Management

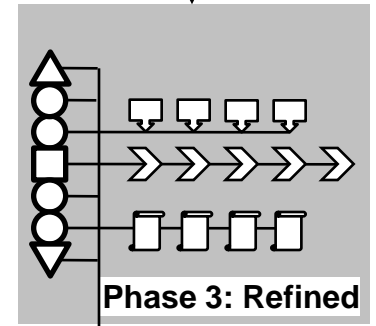
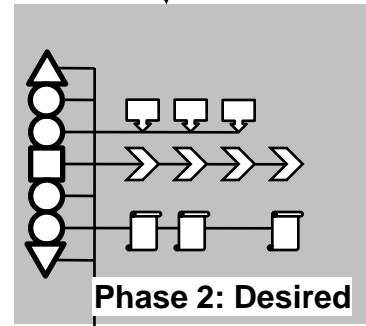
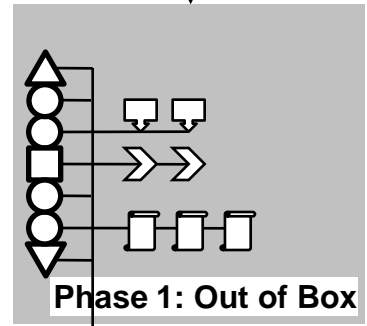
- Module mix
- Module readiness
- System assembly
- Infrastructure evolution

- BSAs
- Proj Mgr
- Dept User
- Prog Mgr

Infrastructure

Active

Passive



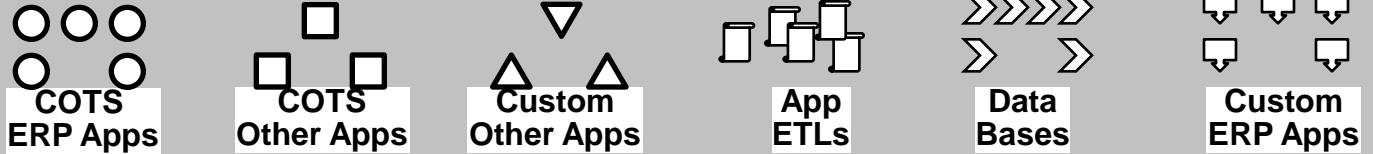
- Fixed reqs during phases
- No change to COTS
- Bus XML comm only
- Internal integ. mgt.
- Contractor peers

ETL Template

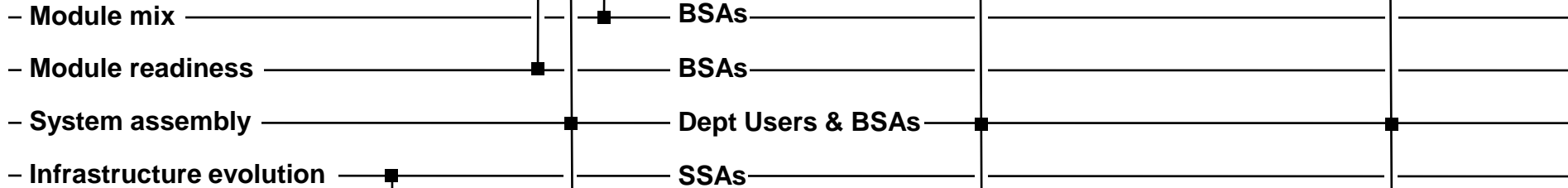
Rules/Standards

Silterra Agile ERP – Developed System

Components/Modules



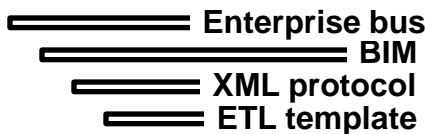
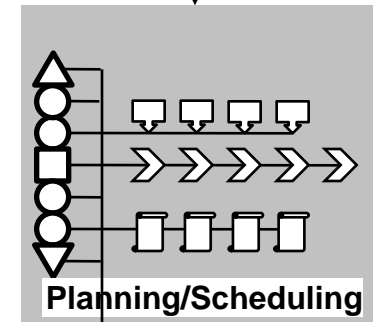
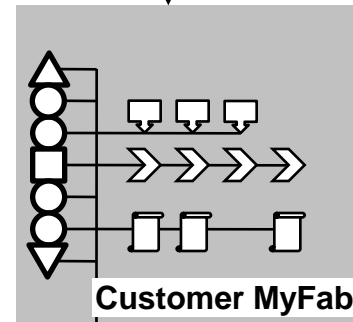
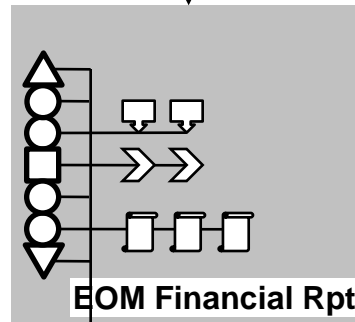
Integrity Management



Infrastructure

Active

Passive



System examples are SOA-like instances of departmental needs

Agile Software Development and New Lean Thinking

Be aware of the difference between:

- ❑ Agile (a branded software development process) and**
- ❑ agile (a dictionary defined capability/property)**

“Classic” Scrum

Ken Schwaber, Jeff Sutherland. 2013. The Scrum Guide. www.scrum.org/

Jeff Sutherland, Ken Schwaber. 2007. The Scrum Papers: Nuts, Bolts and Origins of an Agile Process. Scrum Foundation. <http://scrumfoundation.com>

Inputs from
Customers, Team,
Managers, Execs



Product Owner

Development
Team



Scrum
Master



Daily Standup
Meeting



1-4 Week
Sprint

Sprint Review



1	
2	
3	Prioritized list of what is required: features, bugs to fix...
4	
5	
6	
7	
8	

Product
Backlog

Team selects
starting at top
as much as it
can commit
to deliver by
end of Sprint

Sprint
Planning
Meeting

Task
Breakout

Sprint
Backlog

Sprint end date and
team deliverable
do not change

Finished Work

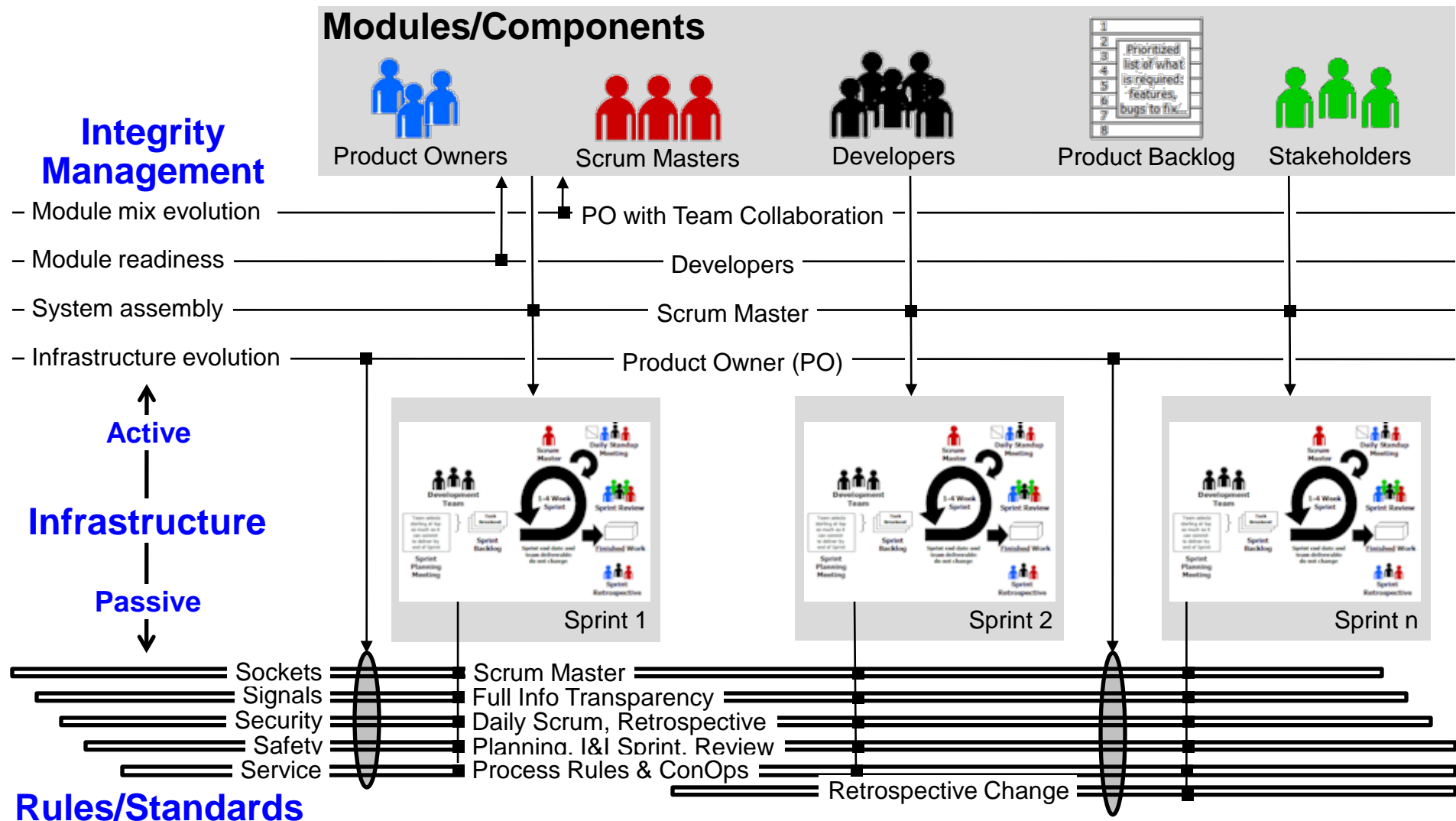


Sprint
Retrospective

“Scrum’s roles, artifacts, events, and rules are immutable, and although implementing only parts of Scrum is possible, the result is not Scrum.

Scrum exists only in its entirety, and functions well as a container for other techniques, methodologies, and practices.” (Schwaber and Sutherland 2013)

Scrum has an Agile Architecture Pattern (AAP) Structure suitable for agile SW development, but not for agile systems-engineering ...



... because the RSA is different for an agile systems-engineering process, and the Scrum AAP strategy is inadequate for systems engineering

Joshua Kerievsky on Lean Startup: Why It Rocks Far More Than Agile Development

Text and video at: www.infoq.com/presentations/Lean-Startup



Agile has reached middle-age, which means it now has love handles and other unsightly blemishes (like outdated ideas, an infestation of easy-as-pie certifications, training classes that buy you PDUs, planning tools that inhibit process improvement, cookie-cutter Agile transition approaches that ignore technical debt, etc.).

It's a big lump o' stuff and so many folks are practicing it just like we did over a decade ago.

Meanwhile, better ideas came along that fundamentally changed our software processes and how we improve them.

Lean Software Development and Lean Startup are two such big ideas.

They have **helped us find slowness and waste in places we didn't even think to look** (like iterations, backlogs, velocity calculations) and have made us far more successful in business (via rigorous instrumentation, Getting Out of the Building, Validated Learning, MVP, the Pivot catalog).

Our practice of Agile today doesn't look remotely like it did ten years ago, and that is thanks to the kick-ass ideas from Lean/Lean Startup.

Lean Startup is furiously frugal and focused on building the right thing. It is the engine, the driver, while Agile is merely along for the ride. And as I said, the Agile that is along for the ride now looks far different than it did a decade ago. We don't do TDD when we are inexpensively validating an idea and we don't wait two weeks to ship something - we ship many times per day.

Agile Software Development 2.0?

From: Guest Speaker: Joshua Kerievsky, Lean Startup: Why It Rocks Far More Than Agile Development, slides: 70-72
www.infoq.com/presentations/Lean-Startup

Agile

Product Roadmap

Product Vision

Release Planning

Sprint

On Site Customer

User Story

Backlog

Customer Feedback

Acceptance Test

Continuous Integration

Velocity/Burn Down/etc.

Led by Successful Consultants

Lean Startup

Business Model Canvas

Product Market Fit

Minimal Viable Product

Learn/Measure/Build

“Get Out of the Building” (developer directed)

Hypothesis

To-Learn List

Customer Validation

Split Test

Continuous Deployment

AARRR

Led by Successful Entrepreneurs

*AARRR: Acquisition, Activation, Retention, Referral, Revenue
<http://500hats.typepad.com/500blogs/2007/09/startup-metrics.html>

Agile 2.0? - Lean and Agile Confusion and Complementarity

Posted by Dove to *Linked in Agile Systems Engineering Forum* - 23Apr2012

We see the phrase "Lean and Agile" appearing together as what might rightfully be an Agile 2.0 complimenting emergence. We also see a lot of confusion of the fundamental meanings of lean and of agile, and the diffusion away from why those two words were chosen meaningfully to designate two different types of fundamental focus. Those with lean leanings too often think that agile concepts are a subset of an overarching lean rubric. Those firmly in the agile camp sometimes think the reverse. These two concepts are based on valuable process objectives: lean is a focus on removing waste, of all kinds; agile is a focus on dealing with uncertainty by creating optional response capability. But as both age in practice the dogmatic detail grows under each - and starts to claim ownership of the other.

Joshua Kerievsky has what I consider to be a thought-starting presentation on "Lean Startup" at www.infoq.com/presentations/Lean-Startup. He says he was an agile practitioner that is now embracing lean concepts as a more useful approach - but in reality he is leaning out the agile approach that he says "has reached middle-age, which means it now has love handles". In many cases, I believe, those who claim to have converted from agile to lean were practicing a dogmatic approach to agile rather than an insightful approach – based on an appreciation for the fundamental concepts – which are still timeless.

With respect I see in Kerievsky's presentation the emergence of an Agile 2.0 understanding, but with chagrin I see his choice of rhetoric polarizing this emergence as a lean dominated lead. Fundamentally he is showing how a more mature understanding of agile software development (specifically) is seeing and removing the wasted resource commitments in the first generation of agile concepts - a timely rethinking and a good start.

There is no turning back on the choice of words headlining the brand-specific "Lean Software Development" and "Lean Startup"; though each is built on a foundation of Agile Software Development and rethinking of first generation agile brand-specific (XP, Scrum, etc) detail dogma with second generation unproductive-waste removal.

It's a good start - but needs to appreciate how agile and lean basics compliment each other rather than how one should dominate the other.

I think the fundamental roots of agile and lean should be kept in mind, as many adaptations of agile software development in different organizational cultures are necessarily cherry picking comfortable dogma from different brand approaches and losing the fundamental value focus.

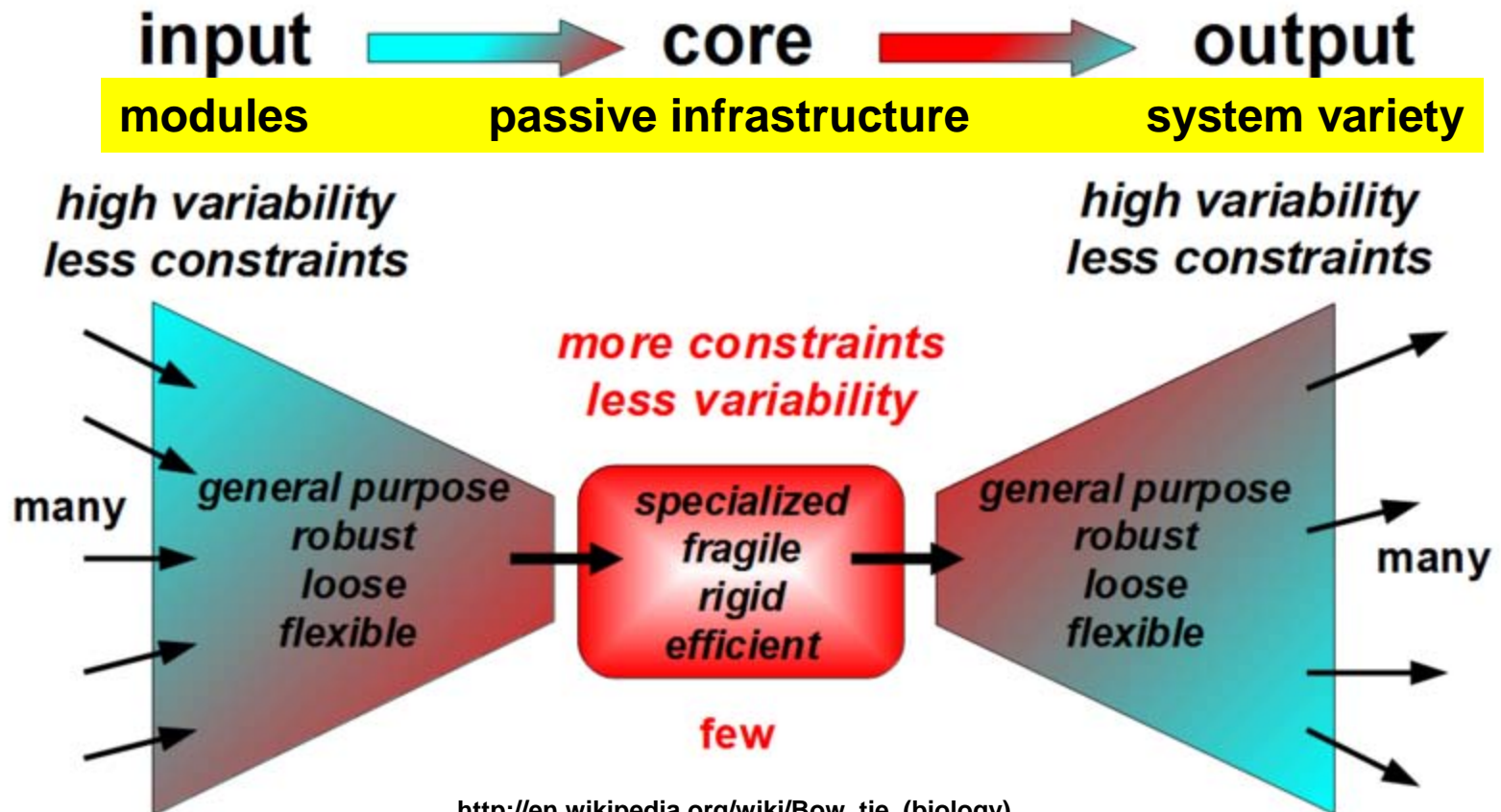
Echoes from Biology and Complex Systems

Bow Tie Process Echo from Science

Our work was based on observation of many real systems that exhibited agile characteristics in a large variety of enterprise domains.

Since then, we have discovered a body of science behind the architecture, with work carried out by collaborators:

- John Doyle, John G Braun Professor of Control & Dynamical Systems, Electrical Engineering, and BioEngineering at Caltech
- Jean Carlson, Department of Physics, UC Santa Barbara
- Marie Csete, (now) Staff Physician at UC San Diego Anesthesiology

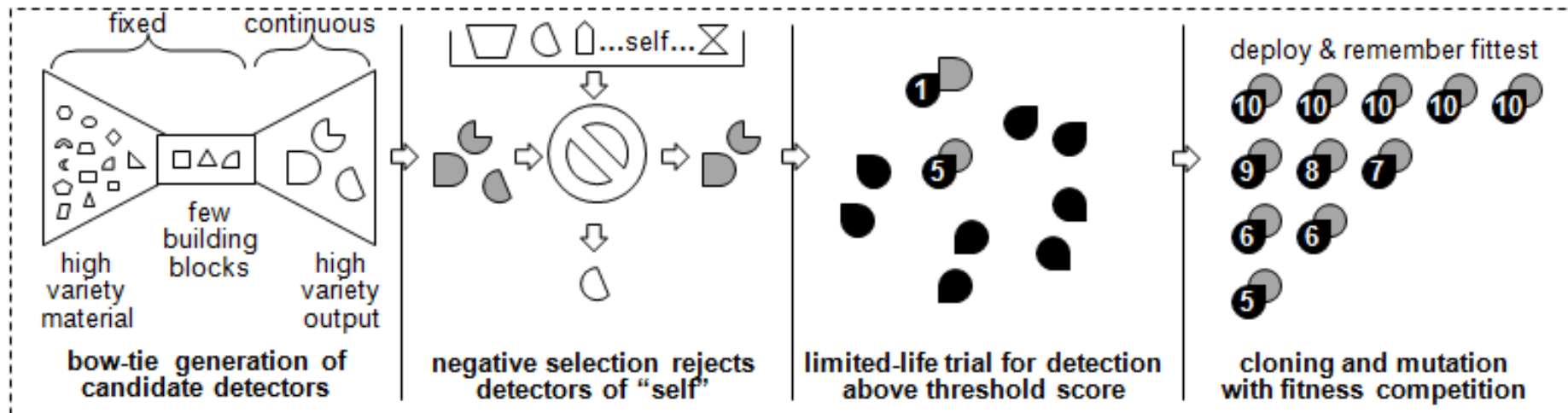
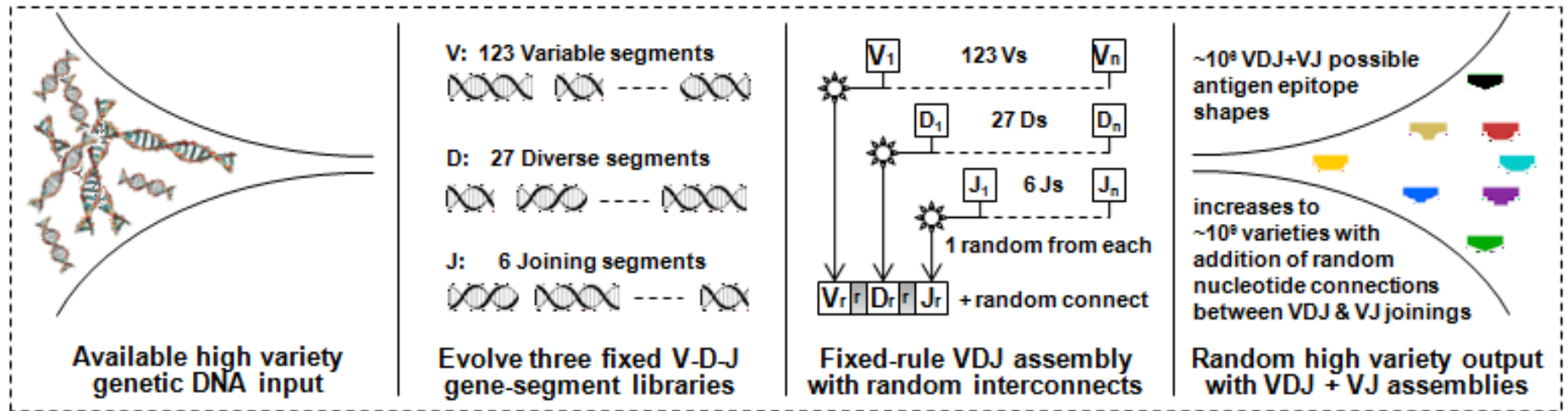


[http://en.wikipedia.org/wiki/Bow_tie_\(biology\)](http://en.wikipedia.org/wiki/Bow_tie_(biology))

rick.dove@parshift.com, attributed copies permitted

Bow Tie Pattern in the Immune System

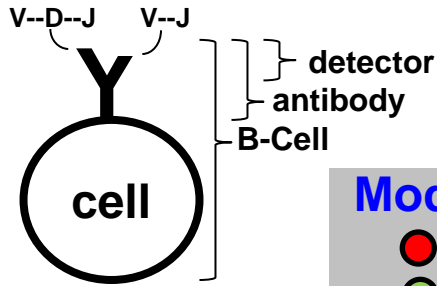
Millions of random infection detectors generated continuously by fixed rules and modules in the “knot”
 (Dove 2010). Pattern Qualifications and Examples of Next-Generation Agile System-Security Strategies.



Speculative generation and mutation of detectors recognizes new attacks like a biological immune system

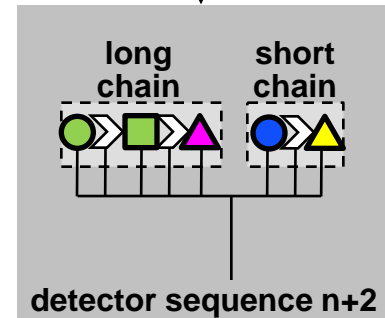
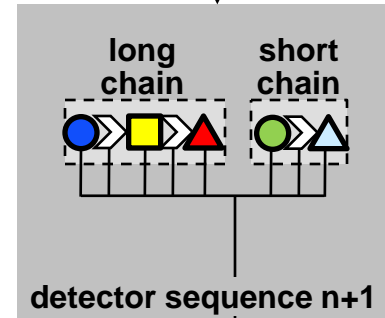
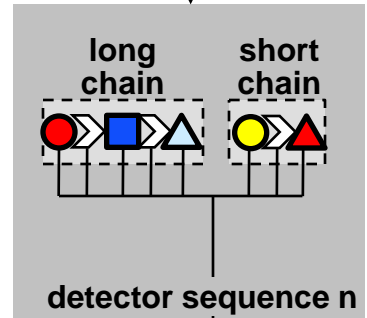
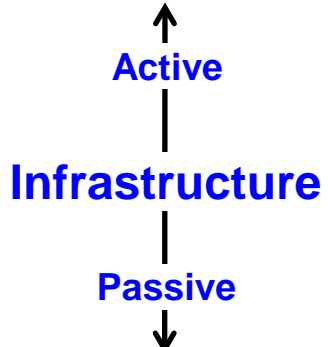
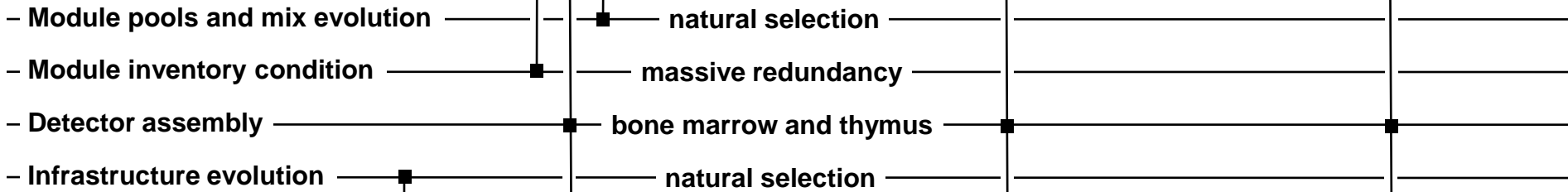
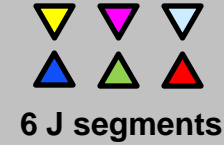
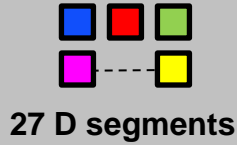
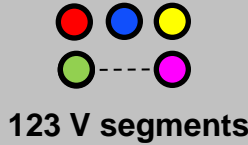
(Dove 2011a) Patterns of Self-Organizing Agile Security for Resilient Network Situational Awareness and Sensemaking

Adaptable Immune System Bow-Tie Antigen-Detector Generator



Integrity Management

Modules



Assembly Rules

On Passive infrastructure

...protocols (infrastructure) are far more important ... than are modules

Marie E. Csete and John C. Doyle. 2002. Reverse Engineering of Biological Complexity. Vol 295 SCIENCE, 1 March.
www.cds.caltech.edu/~doyle/CmplxNets/CseteDoyle.pdf

Consider the ubiquitous Lego toy system. *The signature feature of Lego is the patented snap connection for easy but stable assembly of components. The snap is the basic Lego protocol, and Lego bricks are its basic modules.*

We claim that protocols are far more important to biologic complexity than are modules. They are complementary and intertwined but are important to distinguish. In everyday usage, **protocols are rules designed to manage relationships and processes smoothly and effectively.**

If modules are ingredients, parts, components, subsystems, and players, then protocols describe the corresponding recipes, architectures, rules, interfaces, etiquettes, and codes of conduct.

Protocols here are rules that prescribe allowed interfaces between modules, permitting system functions that could not be achieved by isolated modules.

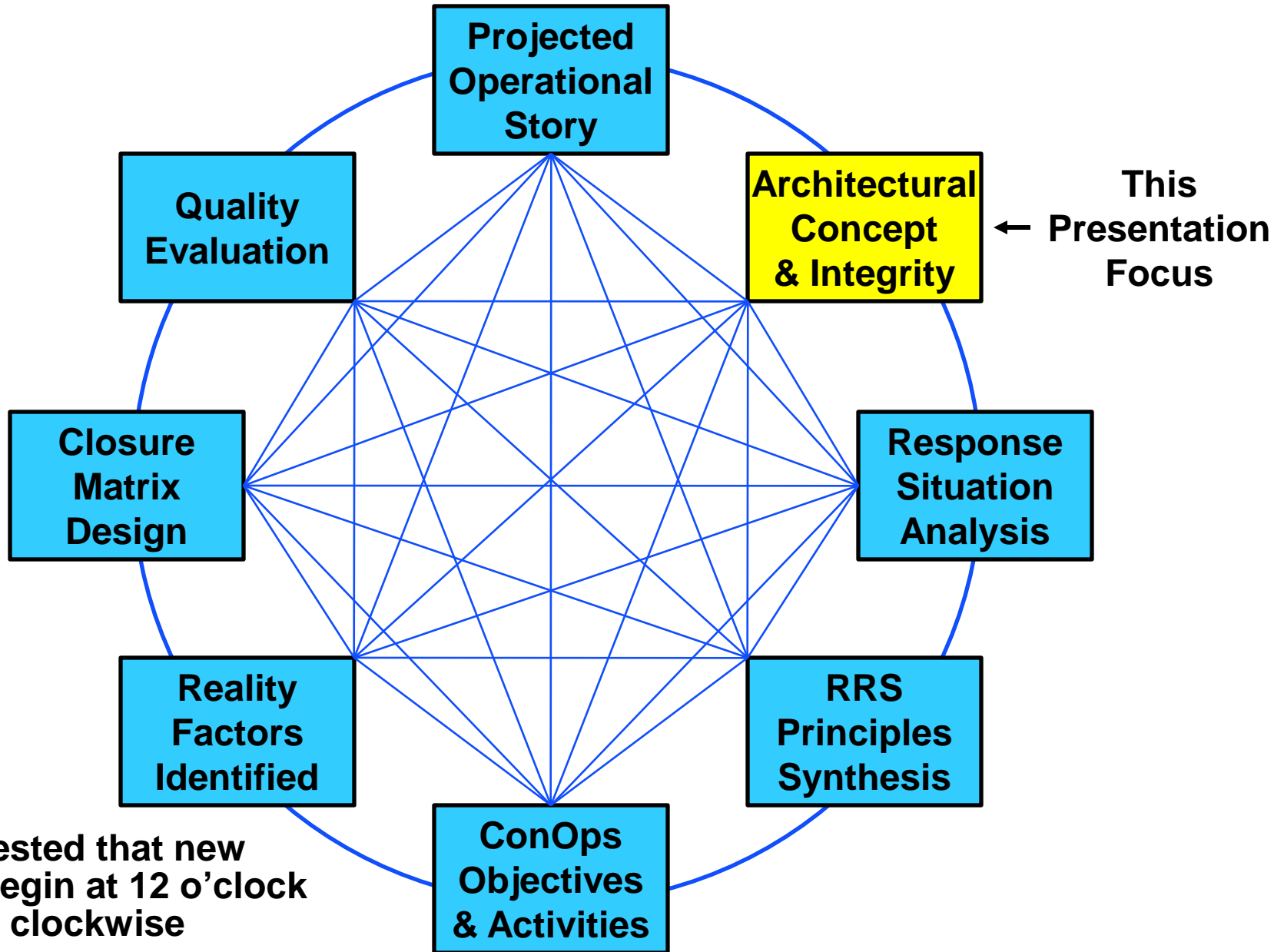
Protocols also facilitate the addition of new protocols and organization into collections of mutually supportive protocol suites.

Like modules, they simplify modeling and abstraction, and as such may often be largely “in the eye of the beholder.”

A good protocol is one that supplies both robustness and evolvability.

Wrapping it Up

Eight principle tools to employ when designing or analyzing a system for agility



Agility - Fundamentally

The Ability to Thrive in a Continuously Changing, Unpredictable Environment.

Agility is *effective response* to opportunity and problem,
within mission ... always ... no matter what.

An *effective response* is one that is:

- | | |
|--|---------------|
| ■ timely (fast enough to deliver value), | <u>Metric</u> |
| ■ affordable (at a cost that leaves room for an ROI), | time |
| ■ predictable (can be counted on to meet expectations), | cost |
| ■ comprehensive (anything/everything within mission boundary). | quality |
| | scope |

You can think of Agility as Requisite Variety.

You can think of Agility as proactive Risk Management.

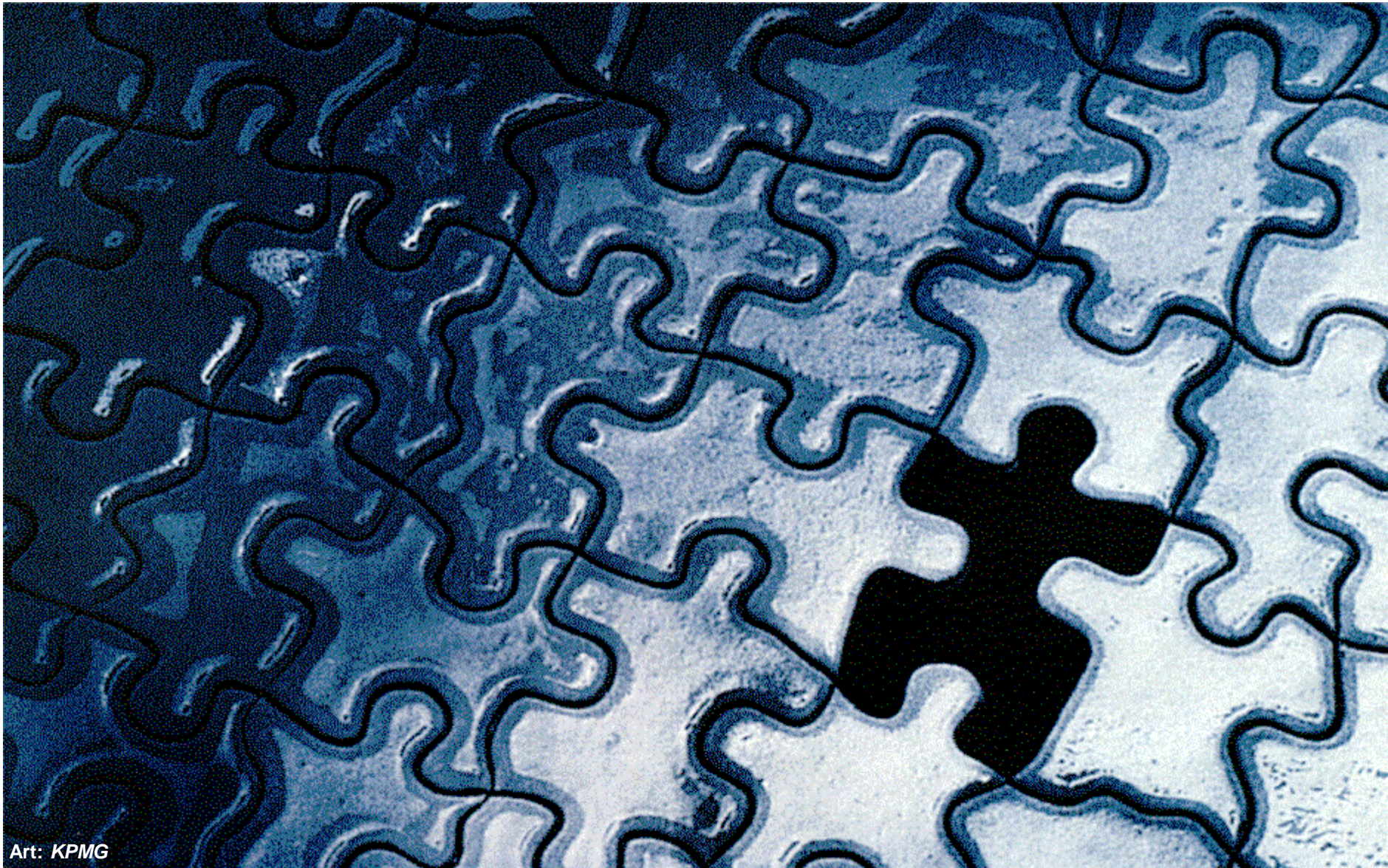
You can think of Agility as Innovative Response in unpredictable situations.

You can think of Agility as Life Cycle Extension.

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

Domain Independent

Modular – But Not Agile



Art: KPMG

Agile System and Project Management by Design

Risk and Uncertainty Management Through:

- ❑ **Creation of drag-and-drop response options**
- ❑ **Enabling effective plug-and-play use of options**
- ❑ **Agility management through active infrastructure responsibility**
that constantly evolves the system and keeps it currently effective

System X-Ray Vision

The bone structure is depicted in the Agile Architecture Pattern. All truly agile systems have the same basic structure and strategy. Knowing this will change the way you “see” and evaluate a system.



<http://awespendo.us/animemangacomics/kermit-at-the-doctor/>

Agile Systems and Systems Engineering (AS&SE) Working Group

**A Working Group of INCOSE
(International Council on Systems Engineering)**

On Request to rick.dove@parshift.com:

- 1. Get on mail list for general announcements.**
- 2. Participate in WG remote-collaboration projects.**
- 3. Get working group charter.**

Chair: Rick Dove

Co-Chair: Ron Lyells, Honeywell

Co-Chair: Mike Coughenour, Lockheed Martin

References and Supportive Readings

- (Bohem 2004) B. Boehm and R. Turner, R., *Balancing Agility and Discipline – A Guide for the Perplexed*, Addison-Wesley, 2004.
- (Boss 2010) Jason Boss and Rick Dove. Agile Aircraft Installation Architecture In a Quick Reaction Capability Environment. INCOSE International Symposium 14Jul2010, Chicago. www.parshift.com/Files/PsiDocs/Pap100712IS10-AgileAircraftInstallationArchitecture.pdf
- (Ballard 2000) Herman Ballard. The Last Planner System of Production Control. PhD Thesis at Birmingham University. www.leanconstruction.org/pdf/ballard2000-dissertation.pdf
- (Csete 2002) Marie E. Csete and John C. Doyle. Reverse Engineering of Biological Complexity. Vol 295 SCIENCE, 1 March. www.cds.caltech.edu/~doyle2/wiki/images/7/7a/Science1664-2002.pdf
- (Csete 2004) Marie Csete and John Doyle. Bow Ties, Metabolism and Disease. TRENDS in Biotechnology 22(9), September. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.173.3019&rep=rep1&type=pdf>
- (Dove 1996) Rick Dove, Sue Hartman and Steve Benson. An Agile Enterprise Reference Model – with a case study of Remmele Engineering. Agility Forum, Report AR96-04. <http://www.parshift.com/Files/PsiDocs/AerModAll.pdf>
- (Dove 2001a) Rick Dove. Response Ability – The Language, Structure and Culture of the Agile Enterprise. Wiley.
- (Dove 2001b) Rick Dove. Design Principles for Highly Adaptable Business Systems, With Tangible Manufacturing Examples. Book chapter in Maynard's Industrial Handbook, McGraw Hill. <http://www.parshift.com/Files/PsiDocs/Rkd8Art3.pdf>
- (Dove 2005) Rick Dove. Fundamental Principles for Agile Systems Engineering. Conference on Systems Engineering Research (CSER), Stevens Institute of Technology, Hoboken, NJ, March. <http://www.parshift.com/Files/PsiDocs/Rkd05032.pdf>
- (Dove 2006) Rick Dove. Engineering Agile Systems: Creative-Guidance Frameworks for Requirements and Design. 4th Annual Conference on Systems Engineering Research (CSER), Los Angeles, CA, Apr 7-8. <http://www.parshift.com/Files/PsiDocs/Rkd060407CserEngineeringAgileSystems.pdf>
- (Dove 2008a) Rick Dove and Garry Turkington. Relating Agile Development to Agile Operations. Conference on Systems Engineering Research (CSER), Redondo Beach, CA, April. www.parshift.com/Files/PsiDocs/Pap080404Cser2008DevOpsMigration.pdf
- (Dove 2008b). Rick Dove. Embedding Agile Security in Systems Architecture. INSIGHT 12(2):14-17, INCOSE. www.parshift.com/Files/PsiDocs/Pap090701Incose-EmbeddingAgileSecurityInSystemArchitecture.pdf
- (Dove 2009) Rick Dove and Garry Turkington. On How Agile Systems Gracefully Migrate Across Next-Generation Life Cycle Boundaries. Global Journal of Flexible Systems Management, Vol 10, No 1, pp 17-26, 2009. www.parshift.com/Files/PsiDocs/Pap080614GloGift08-LifeCycleMigration.pdf
- (Dove 2010) Rick Dove. Pattern Qualifications and Examples of Next-Generation Agile System-Security Strategies. IEEE International Carnahan Conference on Security Technology (ICCST), San Jose, CA, 5-8 Oct. www.parshift.com/Files/PsiDocs/PatternQualificationsForAgileSecurity.pdf
- (Dove 2011a) Rick Dove. Patterns of Self-Organizing Agile Security for Resilient Network Situational Awareness and Sensemaking. 2011 Eighth International Conference on Information Technology: New Generations. www.parshift.com/s/110411PatternsForSORNS.pdf
- (Dove 2011b) Rick Dove. Self-Organizing Resilient Network Sensing (SornS) with Very Large Scale Anomaly Detection. IEEE International Conference on Technologies for Homeland Security, Waltham, MA, 15-17Nov. www.parshift.com/s/111115VeryLargeScaleAnomalyDetection.pdf
- (Papke 2013) Barry Papke, and Rick Dove. Combating Uncertainty in the Workflow of Systems Engineering Projects. Paper submitted for INCOSE IS13 review. www.parshift.com/s/130624LastPlanner.pdf
- (Schumacher 2011) Col. Ludwig J. Schumacher. Dual Status Command for No Notice Events Integrating Military Response to Domestic Disasters. Homeland Security Affairs, Vol 7, Feb. www.hsaj.org/?download&mode=dl&h&w&drm=resources/volume7/issue1/pdfs/&f=7.1.4.pdf