Commonality Faults in Super Scaled Systems

Andrew Loebl
Oak Ridge National Laboratory

Dean Hartley
Hartley Consulting

Situation

To achieve knowledge management… IT applications…must **work together**

$\overbrace{\text{[be designed and developed to]}}$

[be designed <u>and</u> developed to]

- **Lt. Gen. Boutelle (the U.S. Army's Chief Information Officer) has established a strategic partnering division of civilian, military, and contractor personnel to develop systems that interoperate. (9/27/04 issue of *Government Computer News*)**
- **To achieve interoperability "…breakdown… stovepipes … [for a] higher level of jointness," says Senator John Warner (9/27/04 issue of *Government Computer News*)**
- **"…biggest challenges… now is still horizontal integration of … systems," says Maj. Gen. Paul Nielsen (Chief Technology Office of the US Air Force, who retired on 6/25/04. (9/20/04 issue of *Government Computer News*)**

- Important, Chronic, Challenges of both Vertical and Horizontal Integration Persist
- Challenges span all agencies of the Executive Branch of the US government
- The entire information infrastructure struggles to move into the knowledge era
- Concept and means determination are essential to methodologically, programmatically and pragmatically accomplishing fundamental improvements

Modularization and Commonality Have Not Fully Contributed to Modern Systems Design and Development Needs

- Modularization in design/engineering is commonplace, today:
  - **Modules are less complicated thus easier to understand**
  - **Identical functions in various parts of the system can be identified and organized for development and coding efficiency through reuse, etc**
  - **Detected errors can be categorically corrected**
  - **Design is more effectively and efficiently completed**
- Commonality determination in systems design and engineering is also commonplace. Levels include:
  - **conceptual functional commonality, where the same** *conceptual* **function exists in at least two places (could be a problem)**
  - **where the functions have the same interface, and can therefore be used interchangeably**
  - **third level where the functions are equivalent, so that the same results are obtainable using either formulation**
  - **highest level of commonality is functions are actually identical, and the benefits of modularization are powerful, as a result**
- System Engineering is, primarily, a design discipline, focused on requirements determination/satisfaction without addressing methods to accomplish development for system integration.
- System Engineering has not richly contributed to closing the design/development gap as hardware capacities and capabilities expand.

Superscaled Software Systems

- **Enabled by modern computational technology improvements, i.e.:**
  - power of serial processors
  - capacity of serial processors
  - performance of serial processors
  - software systems; design, development and connection between[1]
  - May not be limited to serial computational environments or hardware designs available[2]
- **Posses Complicated Objectives and Expectations, e.g.**
  - Reconfigurable, distributed environment both computationally and in communications among nodes (reconfiguration does not negatively affect performance, fidelity, accuracy).[3]
  - Usually engages a non-trivial number of users whose primary responsibilities are operationally distinct and interdependent
  - Could be a single software system or a system of systems, and some components may not be limited to software
  - Envisioned functionality reflects
    - Problem parsing at a sophisticated and detailed level
    - Requirements for horizontal and vertical integration[5]
    - Interoperable, connected and distributed nodes of operation, data collection, and computation power
    - Support of system and operational functions bound by laws of physics
    - Support of rapid and flexible speed of service and accuracy of results (often for decision making or human abstraction)
    - System and operational functions decomposable to elemental levels
    - Complete understanding of desired properties, confirmed by decomposition, compose complete solution matrix

Commonality Concepts

Tasks and Functions

- **Standards and SME's identify and define tasks**
  - **tasks are common to many activities, but need to be defined only once**
  - **needed tasks for any single 'mission' are called out in some acceptable manner**
  - **Tasks could be broken down further, e.g.,**

    --dig ditch --land airplane

    --rub nose --control airspace
- **Software system has analogous elements**
  - **functions are like "rub nose"**
    - **find nose**
    - **move hand to nose**
    - **move hand back and forth**
  - **patterns are like "dig ditch"**
    - **common elements (use implement, remove dirt to make hole, etc.)**
    - **variable elements (type implement, dimensions of hole, etc.)**

Functional Commonality definition

- Commonality ***is the same or equivalent instantiation wherever that functionality is needed.***

- **To effectively advance our work, definitions of other terms and concepts were formulated.**

Ideal commonality: use of the identical instantiation for each occurrence of that function.

Practical commonality: use of instantiations of the function that yield essentially the same output for nearly the same input[6]; Commonality is not possible without interoperability and connectivity.

____

-Tight Coupling	*-Function*
-Interoperability	***-****Scenario*
-Connectivity	***-****Thread*
-System Integration	*-Interface*
-*Equivalence*	*-Mission*

-Network-centric Operational System ***or*** *System of Systems*

Problem

Non-Interoperability of S/MIME from Lack of Commonality

- S/MIME (Secure/Multipurpose Internet Mail Extensions) protocol for secure electronic mail:
  - digital signature, - non-repudiation
  - encryption, - message integrity
  - authentication, - message privacy
- **Standards specification:**
  - **allows multiple implementations**
  - **key interoperability features lack definition.**
- **Several commercial "S/MIME Enabled" products not interoperable**[10]
- **Solution: commonality means that the same instantiation MUST be used wherever needed.**

Short Definitions & Concepts
(selected)

- **Interface**: function to make non-interoperable functions interoperable
- **Function**: well-defined process that assigns output to each input
- **Interoperability**: output from one function may be input to another
- **Connectivity**: output from one function actually is input to another
- **Integration**: combines interoperable functions for higher level task.
- **Commonality**: is re-use of same function, wherever needed in the system or system of systems.
- **Tight coupling**: systems integration without needing a human in the loop.
- **Equivalence Principle:** same functionality between M&S and operational systems for training, rehearsal, analysis, etc.
- **Pattern:** a description of the core of the solution to a problem that re-occurs such that the solution can re-occur. identifies a set of elements used to accomplish or support a common need, capability or purpose.

Attention to Commonality Throughout Design & Development Cycle Yields Positive Outcomes

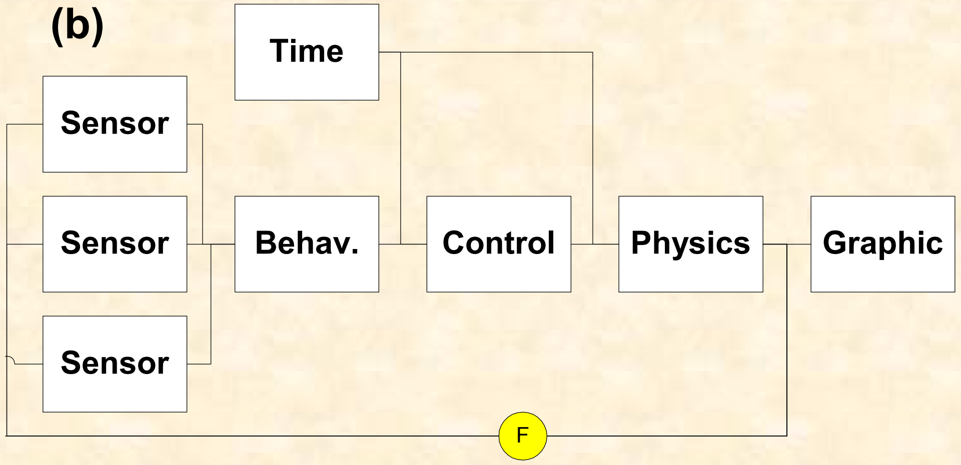
- Functionalities and operational characteristics arise from system not individual platforms
  - **System behaviors result from interactions of platforms and desired capabilities**
  - **Undesirable operational dynamics are explicitly determined and minimized**
- Operational concepts and software simulation/performance developed early
  - **Modern plug-and-play[7] concept speeds understanding and development**
- Control of undesirable properties that might emerge in system execution
- Enhancement of desired properties that might emerge in system execution
- Enables T&E to proceed without full development of final system
  - **Validation of models not just operation**
    - **need to validate system performance**
  - **Understanding of emergence effects**
  - **System concepts can evolve**
  - **Validation can possibly be mathematically confirmed**
- Operational system is computational engine for multiple domains[4], in real time or faster
- Leads to federation approaches to mod/sim development
  - **Modules or copies of modules are linked-in, not recreated to ensure commonality**
  - **Multiple SMEs represented**

Illustration

Commonality Use Across Real and Simulated Systems

(a)

(b)



Robot Swarm Experiment for Coverage and Automated Control:
4 Emergence Examples:

- Inadequate number of robots could not cover the entire building with the radio communications
- Too many robots overwhelmed the communications network.
- Initial placement of the robots strongly affected the final network topology, which was also sensitive to the building geometry via the line-of-sight radio performance.
- The robot jittered while negotiating a corner (<u>unexpected</u> but found in simulation **<u>and</u>** experiment)
    - control algorithms developed for real and simulation systems were commonly held
    - simulation correctly modeled system-of-systems interactions of robot swarm

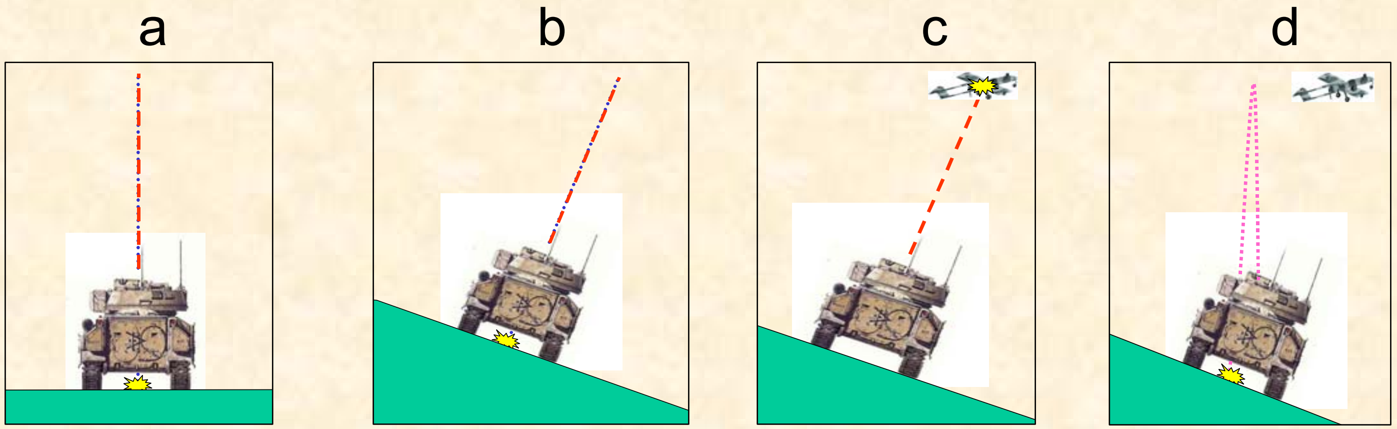
Commonality Has Another Dimension

- **Elements of operational and other domain use must be similar**
  - Discrete elements in physical systems must be represented as individual elements in software system
  - Behaviors of elements among domains of use must be similar (identical?) to behaviors of elements in physical systems
  - Interactions among elements in virtual system must be similar (identical) to interactions among elements in physical system
- **The architecture of the virtual system must reflect architecture of physical system[7]**

Problem

Lack of Commonality Between the Physical World and the BFV Simulation.

SIMNET problem involved improper trajectories training simulations for the Bradley Fighting Vehicle.



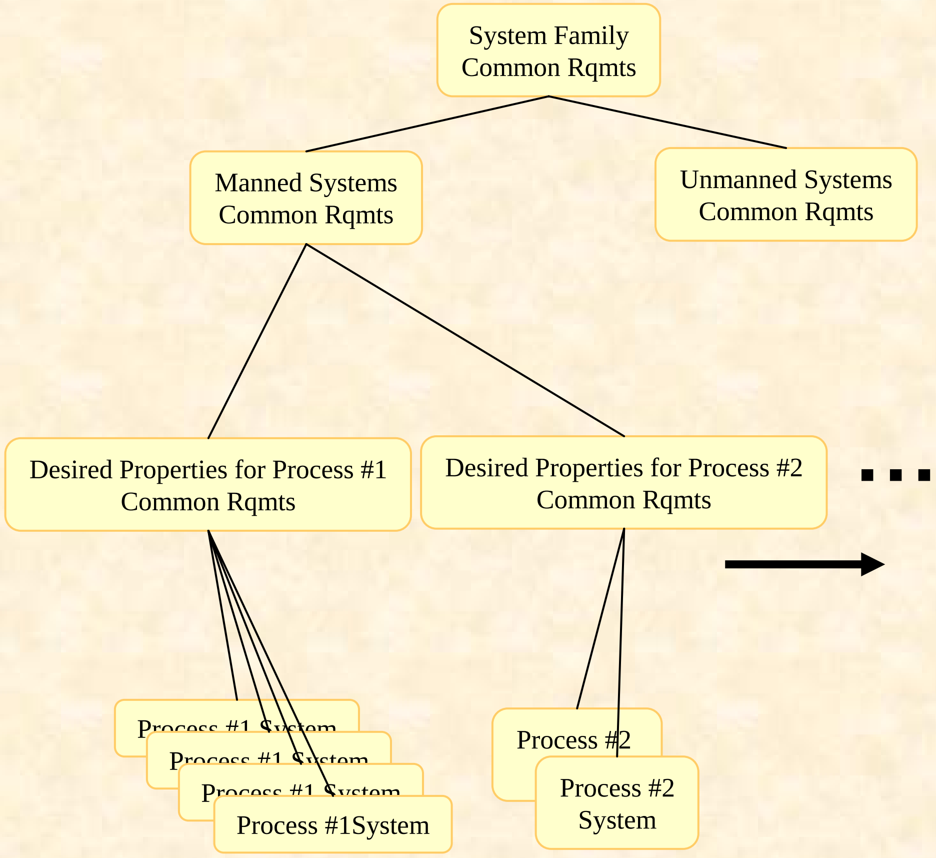
Firing directly away from the viewer

(a) BFV firing a round on level ground, (b) incorrect simulation of a BFV firing a round on a slope, (c) UAV hit in incorrect BFV simulation in (b), (d) correct simulation of BFV firing on a slope with turret turned uphill to miss UAV and hit the intended target.

Commonality Determination and Concomitant Requirements Are Pervasive in Hardware Procurements

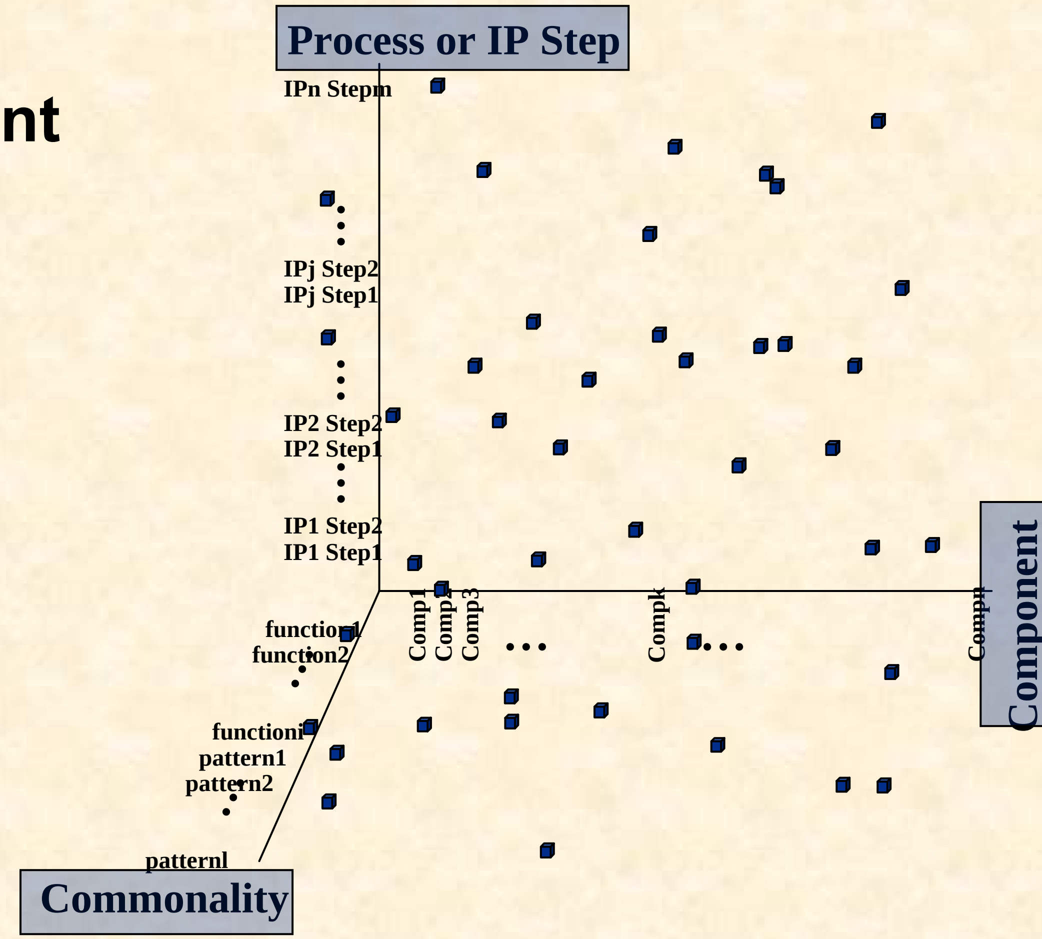
- **Contract, performance, form and fit requirements for hardware are expected.**
  - **Common over all systems**
  - **Common by group**
  - **Common by subgroup**
  - **Particular for system**

- **Why shouldn't Software have commonality specifications?**


System Family Common Rqmts
Manned Systems Common Rqmts
Unmanned Systems Common Rqmts
Desired Properties for Process #1 Common Rqmts
Desired Properties for Process #2 Common Rqmts
. . .
Process #1 System
Process #1 System
Process #1 System
Process #1System
Process #2
Process #2 System

Software Commonality, Also

- **More Complex**
- **By System Component**
- **By Process Step**
- **By Domain**
  - Operations,
  - M&S for T&E,
  - M&S for Analysis,
  - M&S for Training
  - M&S for Acquisition
- **By other?**



Design and Development Includes Several Disciplines

- Conscientious software design & development ubiquitous;
  - **Systems Engineering alone is inadequate for SW development**
  - **Transaction definition alone is inadequate for SW development**
  - **Interface reconciliation to resolve non-interoperability is inadequate**
- Commonality seems pervasive within/among processes of given systems
  - **Commonality determination and adherence in development needed for**
    - assured performance
    - lower life cycle cost
    - higher maintainability
  - **Commonality requirement stronger than "code re-use" requirement**
  - **Commonality is critical**
    - Specifications requirements for hardware domain: chassis, fuel, tools, chip set, power etc.
    - **Commonality is just as important** within **software domain**
  - **Tight-coupling**
    - Human decision making imposes time constraints for system performance
      - For machine speed support, there must be no human in the loop
      - For analysis/decision making "machine intelligence" is an oxymoron

VehicleTransmission

Analogy for “tightly coupled”

- **Manual transmission vs automatic transmission**
  - when highest human function is control of speed, e.g., racing

  choose manual control of gear selection

  - when most important human function subordinates speed, e.g., finding enemy

  choose automatic gear selection
- **Human in the loop vs computer control**
  - when most important human function is decision of action

  choose human in the loop

  - where speed of action is required

  choose tightly coupled computer control

New Paradigm Rules

For complicated/complex systems development

- **Covers software, domains, dimensions, e.g.**
  - Operations
  - M&S
  - Training
  - T&E
- **Direct and greatly simplified design & development with exhaustive and comprehensive commonality determination**
- **Single author**
  - for each common function
  - for each common pattern
  - requirement must cover all software in operational system
- **No code implemented in operational system until it has been implemented in simulations**

Problem

Failure from Lack of Commonality: Inconsistent Models

- **At the Korean Battle Simulation Center[8] three military models exercised:**
  - CBS simulates Army land warfare of units, combat activities, including Blackhawk helicopters. CBS probability of kill = (probability of a hit) x (probability of a kill, given a hit).
  - RESA simulates Navy warfare, including Seahawk helicopters. RESA models loss as one-step process with probability of kill is an input variable.
  - AWSIM simulates Air Force warfare, also including Pavehawk helicopters. AWSIM models loss: probability of kill as an input variable.
- **Each model was fully validated.**
- **The Aggregate Level Simulation Protocol (ALSP)**
  - combined these models
  - shared information among models
  - All three helicopters had equivalent capabilities thus equally vulnerable
- **ALSP simulation showed that Army Blackhawk helicopters were less vulnerable to all weapons.**
  - **Flaw arose from lack of commonality in the probability of a kill.**

Challenges

- **Modern software systems cannot be built piecemeal**
  - Technology allows for expectations to be fulfilled
  - Integrated design & development must be planned and continuous by SME's
    - just as expected, now, for platform design and development
  - High expectations by users of system performance and the methods used to build that performance
  - Parallelization, plug and play is part of the method; but must avoid:
    - Distributed & Uncoordinated work
    - Stovepipes (and stovepipes within the stovepipes)
- **Major development milestones must be integrated to assure understanding of desired performance**
  - Functional decomposition → Enormous data resources
  - Other efforts expended but often unutilized
  - Requirements analysis not directly related to functional analysis
- **Expectations for and performance of modern software/hardware systems is revolutionary**
  - Traditional M&S built to satisfy historic needs within historic technology limitations are now insufficient
  - Federation technology (HLA/TENA*) are now insufficient
  - Representations across body of knowledge inconsistent

Sadly, Most Systems Development Is <u>Directed</u> by Users

- **Focus of system development is operational requirements of hardware.**

  Requirements creep inevitable

- **System design must account for functional requirements and plan for desired properties:**

  connectivity ≠ design ≠ interoperability
  connectivity ≠ integration ≠ interoperability
  connectivity ≠ performance ≠ interoperability

- **Design does not end with development**

  For expectations of modern systems, design/development gap must be eliminated

- **For modern systems, an incomplete design will fail - incomprehensibly**

New Approach

- Assures performance
  - Design
  - Development
  - Operational
  - Life Cycle
- Reduces cost, e.g.
  - Development cost
  - Maintenance cost - faster response to changing requirements
- Enables test and evaluation
  - System behaviors arise from element interactions
  - Testing of components within an evolving system development context
    - Enables assessment of modern methods and technologies, e.g.: network-centric systems concepts
    - Evolutionary development thru plug and play
    - Helps turn an infinite problem into a finite one
- **Enables operational systems objectives**
  - Enables early training and human effectiveness evaluation
  - Enables algorithm development and technology insertion
- **Directly supports fulfillment of user vision and expectations as well as system performance**

Different Paradigm Exists

- **Culture change that abandoned "testing in" quality, and embraced "making quality certain."**
- ***Quality* defined as *conformance to requirements*.**
- **Proactive concept applied:**
  - **System design(er)**
    - must make sure design captures commonality among requirements, defined by function,
    - understand capabilities expected
    - assure performance of processes
    - must ensure commonality is enforced in that system's development.
  - **Development(er)**
    - must make sure design captures commonality among requirements, defined by function,
    - understand capabilities expected
    - assure performance of processes
  - **Customer/User** (as SME)
    - must make sure design captures commonality
    - enunciate and explain the full set of capabilities expected
    - define known processes and tasks
    - manage and direct design and development implementing new paradigm
    - assure life cycle interaction
- **Concept Proven Successful[9]**

Background

Implications

Management changes are needed

Technical changes are needed

Some Management Implications

- **“We have to do everything differently”**
  - New mindset: plug and play SoS, not just electronic interface
  - Holistic understanding (Common Frame of Reference)
  - No code implemented in operational system until it has been implemented in simulations
  - Contract for Commonality

- **Strengthen roles, responsibilities, and accountability of Chief Engineer, M&S Director**
  - Provide adequate resources up front; increase responsibility for tight coupling and integration.
  - Ensure integration of functions within processes and across IPs by comprehensive, informed commonality determination and M&S
  - User focus must shift to commonality determination rather than just requirements development requirements process is a means to an end, not an end in itself

Some Technical Implications

- Holistic understanding (Common Frame of Reference)
  - Requirements analysis for a purpose, not just for progress metrics
  - Close connections between SE and development
  - Commonality supports SE and design integration
  - Commonality supports system(s) integration and advances closure
  - Decomposition as a means to an end, not the end
  - Knowledge management fusion enhancements via commonality
  - Fundamentals of design and integration concepts, definitions and methods must flow to all levels of design/development/user/SME team; evangelize this paradigm, explain the shift
  - Commonality determination helps directly address the gap between design and development, with more front-end investment, savings accrue throughout the life cycle

- Develop and train to rigorously identified SE process and methods for studies, M&S, and integration of design.
  - Leverage/coordinate work among designers/developers
  - Constantly seek commonality at all levels
  - SE too much of an art

Emergent Topics
(selected)

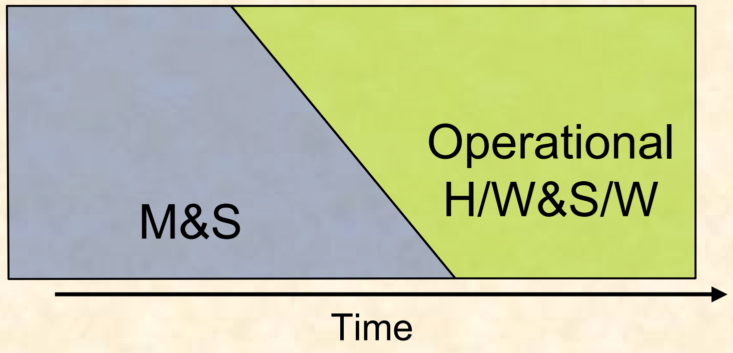
<u>Some Emergent Topics Addressed to Date</u>

•Control of Chaotic Dynamics in Network Traffic	March 2004
•Robustness against Complex Failures in Net-Centric…	April 2004
•Determination of detail in IERs & value to M&S	May 2004
•White paper on systems development	May 2004
•Machine vs human commonality	June 2004
•Identification of additional FD for Communications/DSS	July 2004
•White Paper on DOORS/UML for exhaustive FD	July2004
•White paper on Tightly Coupled Systems and Commonality	September 2004
•White paper on Commonality Possibilities Among Simulations	Sept. 2004

Emergent Topics (Con't.)
(Selected)

- **Lack of Commonality white papers:**
  - +Scope and Class of potential commonality problems July 2004
  - +Induced Problems in Simulations Analysis of Lessons Learned Field Artillery Commonality Problems September 2004
  - +Commonality and Levels of Aggregation in Models August 2004
  - +Commonality in System of Systems in M&S; Lessons October 2004
  - +SIMNET and Commonality September 2004
- **Mission Means Framework relative to SoS Integration October 2004**
- **Design-based Software Development of large and complex systems November 04**
- **OOP Encapsulation Supports Connectivity Not Integration Nov 04**
- **Costs of Lack of Commonality Initial Findings from the Commonality Pathfinder Project Dec 2004**
- **Definitions of terms and concepts, mathematics Dec 2004**
- **Successful SoS Design, Development, & Integration Dec 2004**

System(s) Integration Lab Use



•As the operational hardware is specified, that specification then development, spirals into a M&S representation that can evaluate the components' interaction with the remainder of the operational system.

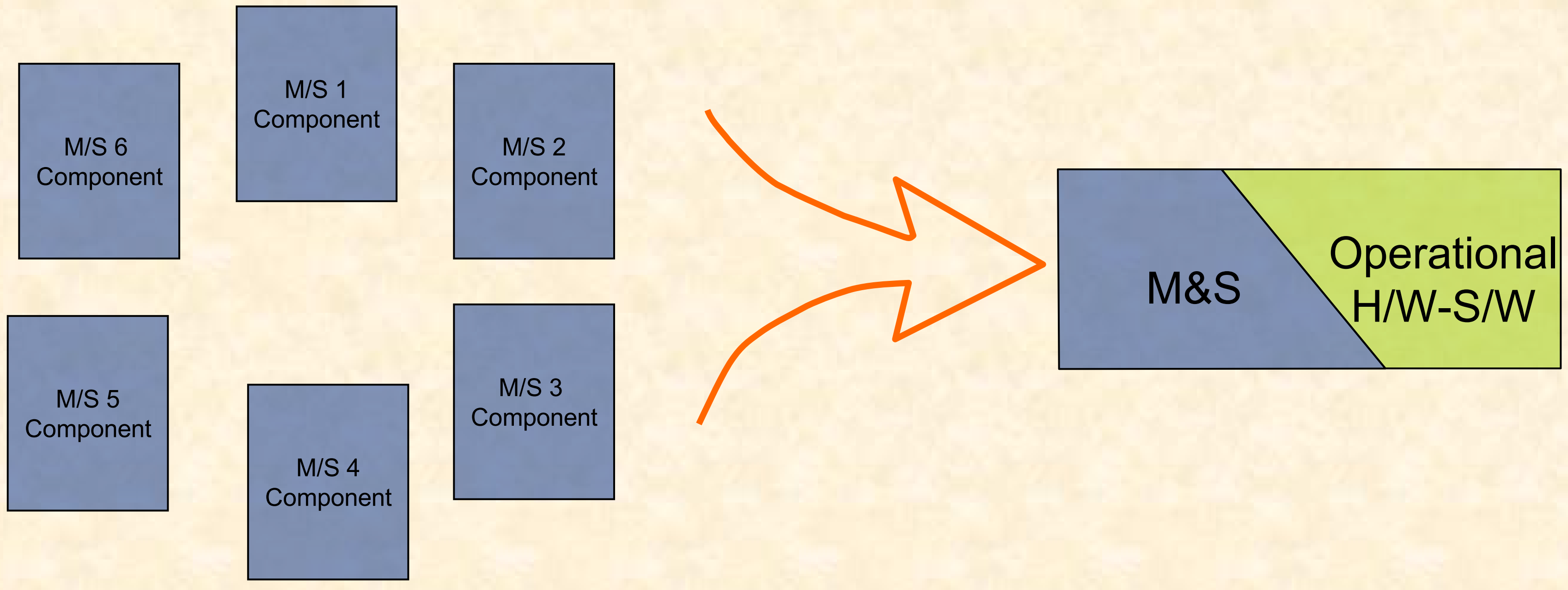
• This can only be accomplished if M&S representation is architecturally similar to the operational system such that subsets of the M&S can be substituted with operational components as they are developed. (plug and play)

•Ditto for upgrades throughout the life cycle

Modern Modeling and Simulation Has New Purpose

- **Present M&S community is unable to provide an appropriate, tightly-coupled, high fidelity environment in which to model the interactions necessary for complex systems representation (e.g. communications).**
- **Until the middle of the last decade, technology and methods were not far enough advanced to envision the possibility that this was possible.**
- **Thus M/S has had a different purpose: In a disaggregated sense, M/S uses various forms of abstraction to focus on disjoint areas of interest. For this discipline, there was never an intent that M/S be comprehensive nor consistent with respect to domain and inter-domain functionality, modern 'plug and play' fidelity/reliability.**
- **Given experience and intent to date, a properly integrated, tightly coupled and internally consistent, high fidelity, comprehensive M/S activity for integration is non-traditional.** ("A high-fidelity Communications model is not needed".)
- **The fidelity issues are far more complicated for domains tempered by complex, large, safety critical, and decision support system(s). E.g. network and communications domains, information and infrastructure security and assurance, C2.**

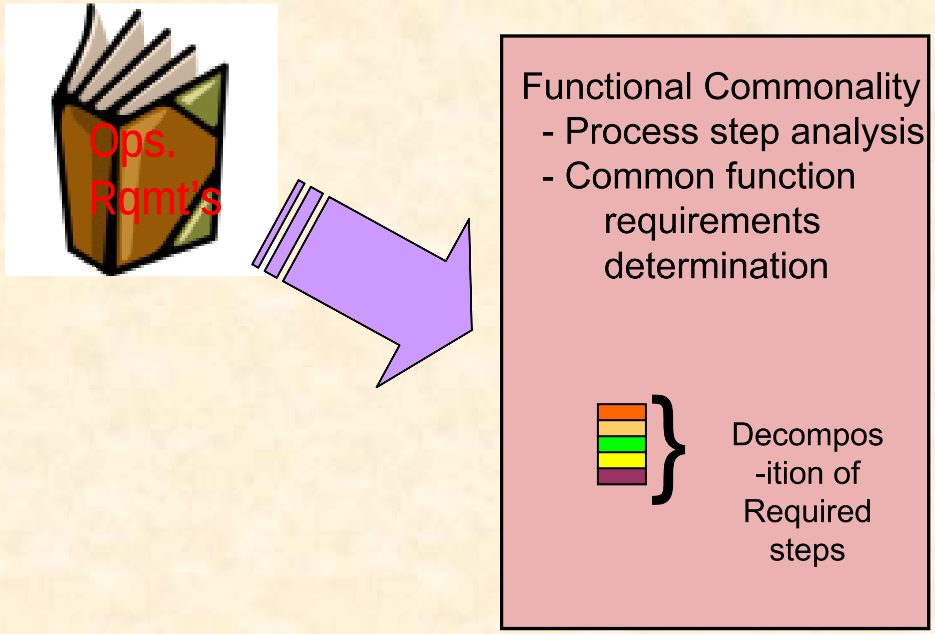
Another Challenge to Common Practice



From Here…

To Here?

Functional Decomposition for a Purpose

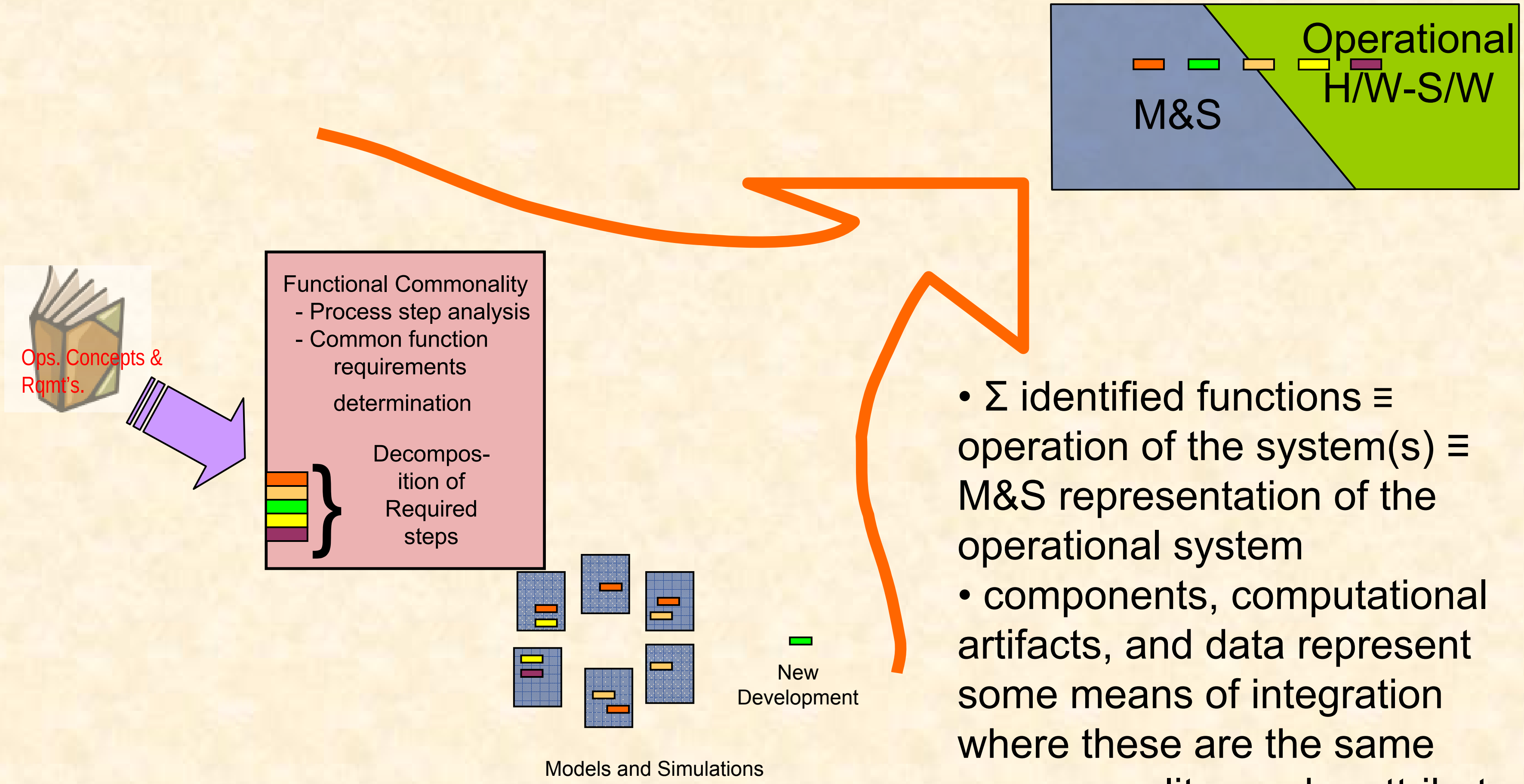

Ops. Rqmt's
Functional Commonality
- Process step analysis
- Common function requirements determination
Decompos -ition of Required steps

•Operations concepts and requirements are formally analyzed and documented to understand:

- functionality desired and needed

-which parameters or algorithms are to be used to determine if commonalities are indicated.

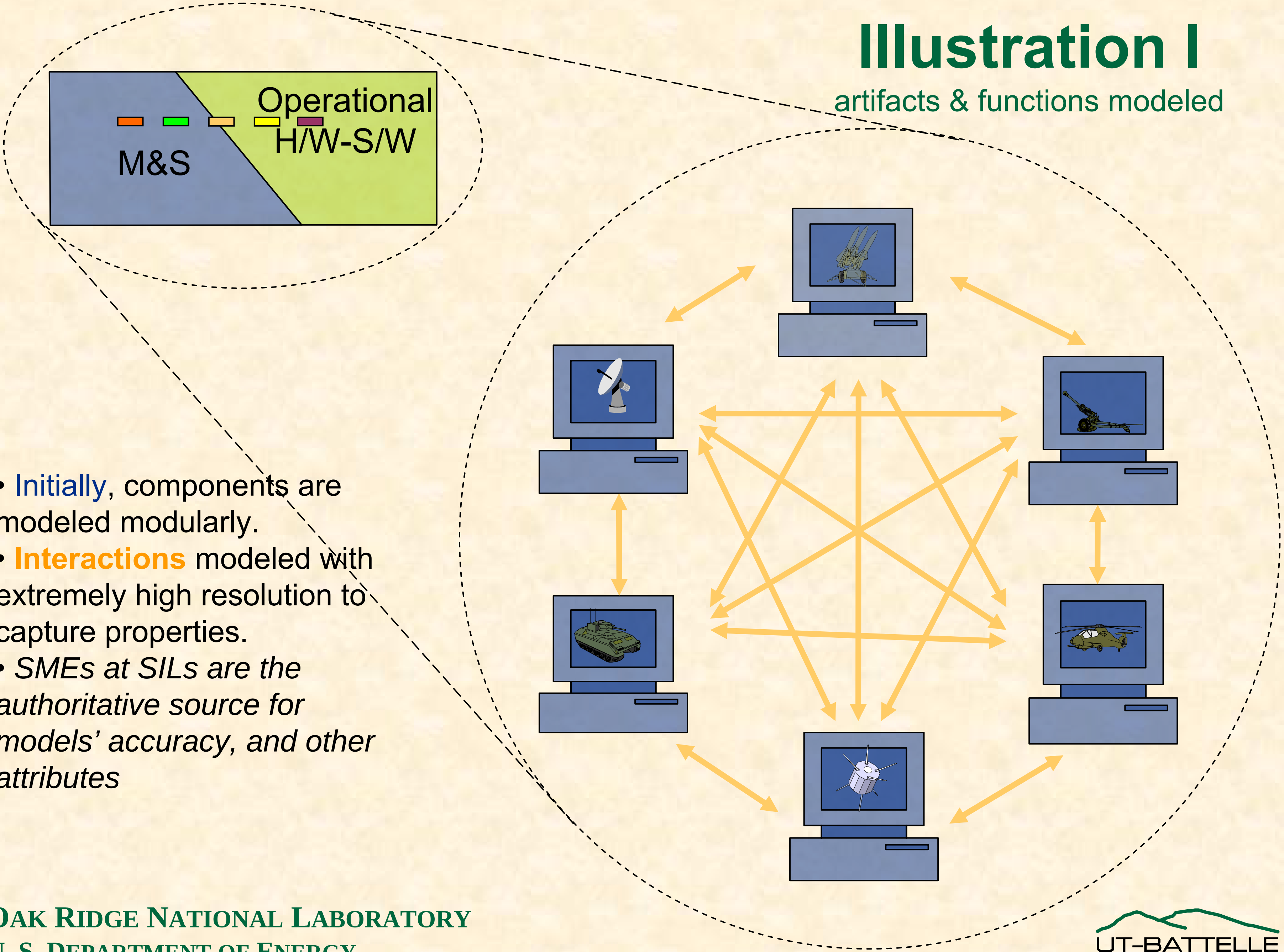
Some Outcomes



- Σ identified functions ≡ operation of the system(s) ≡ M&S representation of the operational system
- components, computational artifacts, and data represent some means of integration where these are the same
- commonality can be attributed and (e.g.) code development is reduced in useful ways

Illustration I

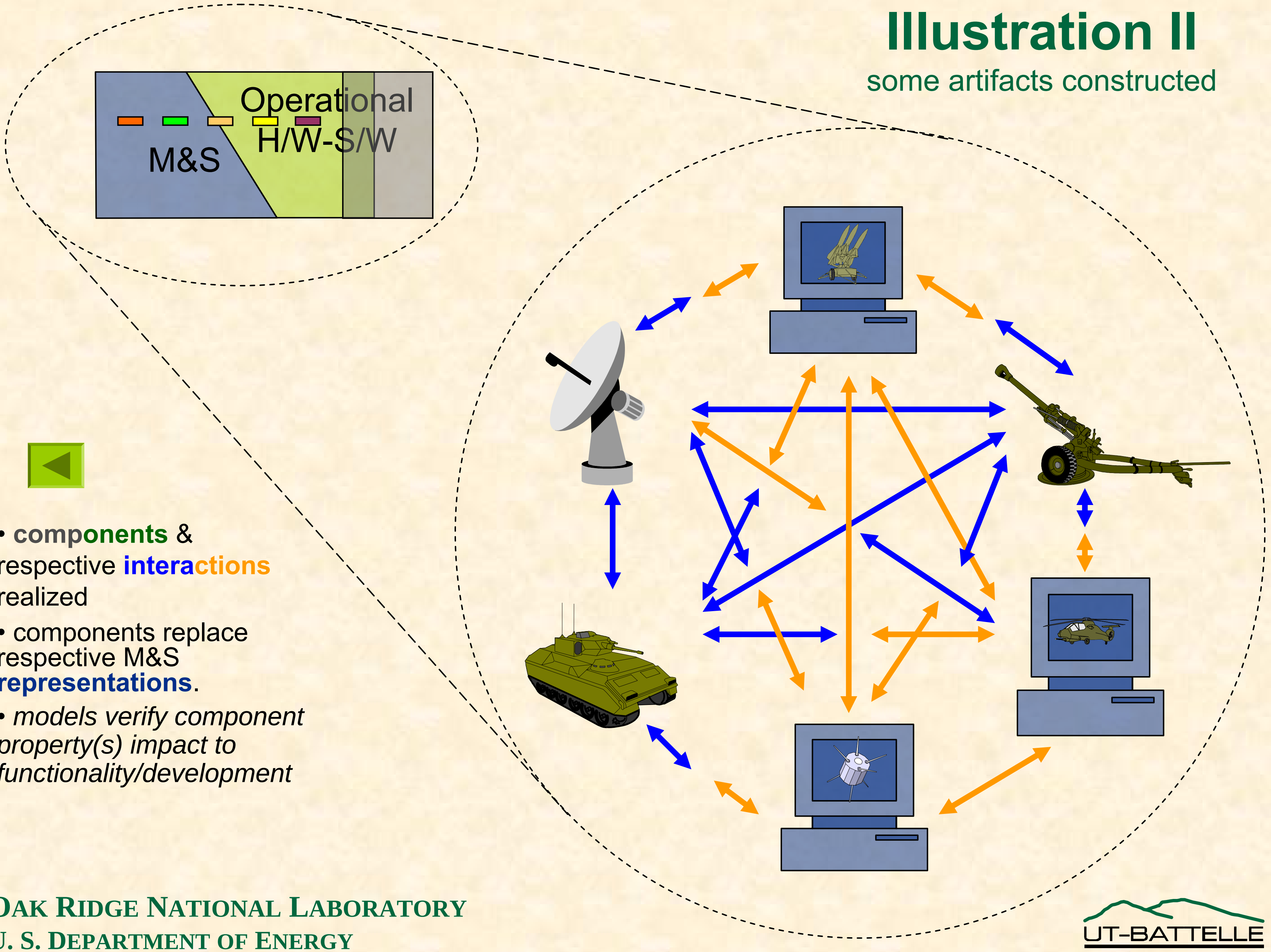
artifacts & functions modeled



- *Initially*, components are modeled modularly.
- **Interactions** modeled with extremely high resolution to capture properties.
- *SMEs at SILs are the authoritative source for models' accuracy, and other attributes*

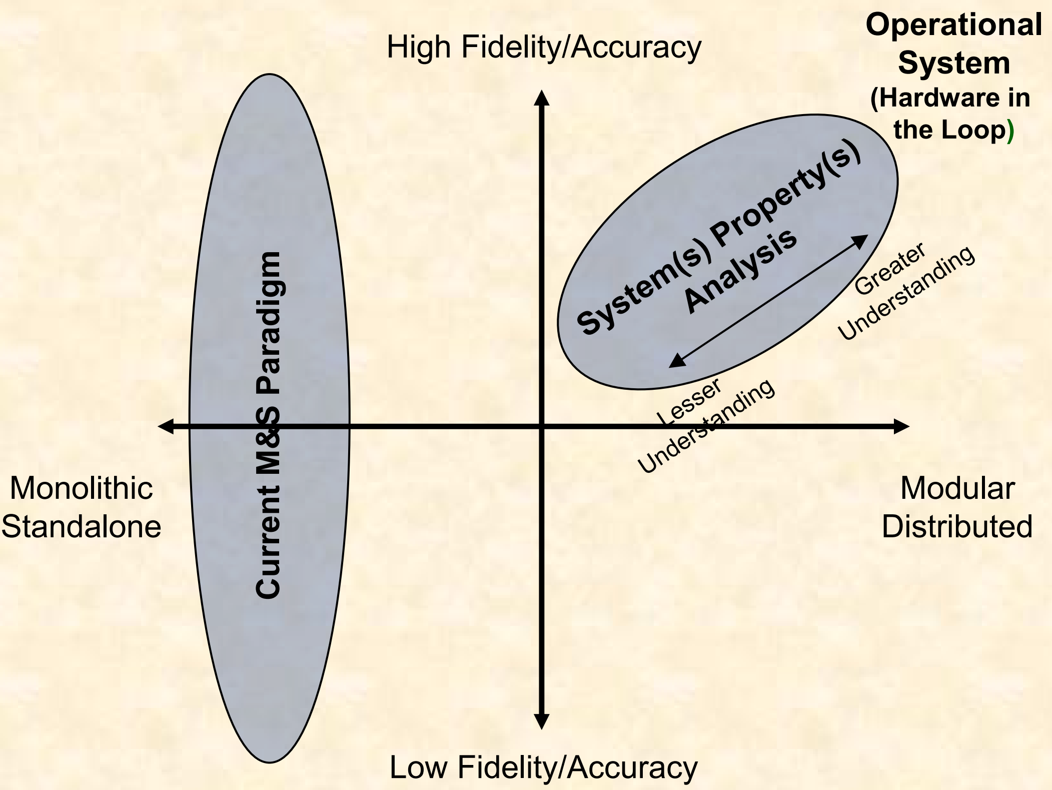
Illustration II

some artifacts constructed



- **components** & respective **interactions** realized
- components replace respective M&S **representations**.
- *models verify component property(s) impact to functionality/development*

Pathfinder Project & Simulation Philosophies



- System(s) Property(s) Analysis was not a formal part of this project.
- Methods and materials for understanding the predicates upon which integration is achieved was a part of this project
- The basis for determination of integration; to actually perform analysis on systems under development is a matter left to the SIL.
- The predicates are artifacts of system operations, e.g. data, computations, measurements, sensor data management and manipulation, equations and terms, etc.
- This project was intended to find fast and effective means for determining methods and materials that reduce cost in software development.
- Concept development for superscaled systems cross-functional integration was discovered to be a powerful means of accomplishment.

Field Artillery (FA) Lessons Learned from Operation Iraqi Freedom

Commonality problem.			
	Accountability		
		software: operational computer system vs admin computer system	FA: Personnel accountability and dynamic task organizations
		software: unclass vs classified	FA: Standard Installation Division Personnel System (SIDPERS) Transactions
	Crashes		
		software: ADOCS vs AFATDS	FA: AFATDS software issues - geometry
		software: LAN vs AFATDS	FA: Internet Protocol
	Inconsistency		
		physical: AFATDS vs sensors	FA: AFATDS software issues - time drift
		software: C2PC vs ADOCS vs BFT vs MCS-L	FA: Multiple C2 Automated Systems
	Morale		
		admin: hardcopy vs database	FA: Mail System
		admin: hardcopy vs e-copy	FA: Promotions
		software: admin vs admin	FA: Finance transactions
		software: external vs internal	FA: Red Cross Messages

	Tight coupling		
		doctrine: COMSEC vs operations	FA: Communications Security (COMSEC)
		doctrine: COMSEC vs operations	FA: Each division had its own transmission security key (TSK) and there were two load sets.
		personnel: skills	FA: Continuity Operations
		physical: PLS vs MLRS	FA: HEMTT/HEMAT as a re-supply vehicle for MLRS untis
		physical: Army issue vs commercial	FA: GPS limitations
		physical: M1068 vs M270A1	FA: Command and Control Vehicle (C2V)
		software: ADOCS vs AFATDS	FA: Coordination Handshakes
		software: BCS vs AFATDS	FA: Computing Maximum Ordinate
		software: FA radars vs AFATDS	FA: FA Radars and AFATDS Incompatibilities
		software: structured communications vs free text	FA: Free Text Messages
Integration problem.			
	Tight coupling		
		software: intelligence vs JDAMs	FA: High fidelity of intelligence products stressed ACE
		software: JMUL procedures	FA: JMUL Update Process
Training problem.			
	Mental workload		
		personnel: skills	FA: C4ISR capabilities at the battalion level
		personnel: skills	FA: Enabling network centric operations across the force
	Tight coupling		
		software: AFATDS vs ADOCS vs Client	FA: Automated Fires and Effects Coordination