An Initial Ontology for System Qualities

Barry Boehm, USC

Universitat Politècnica de Catalunya Talk
April 29, 2015

SYSTEMS ENGINEERING Research Center

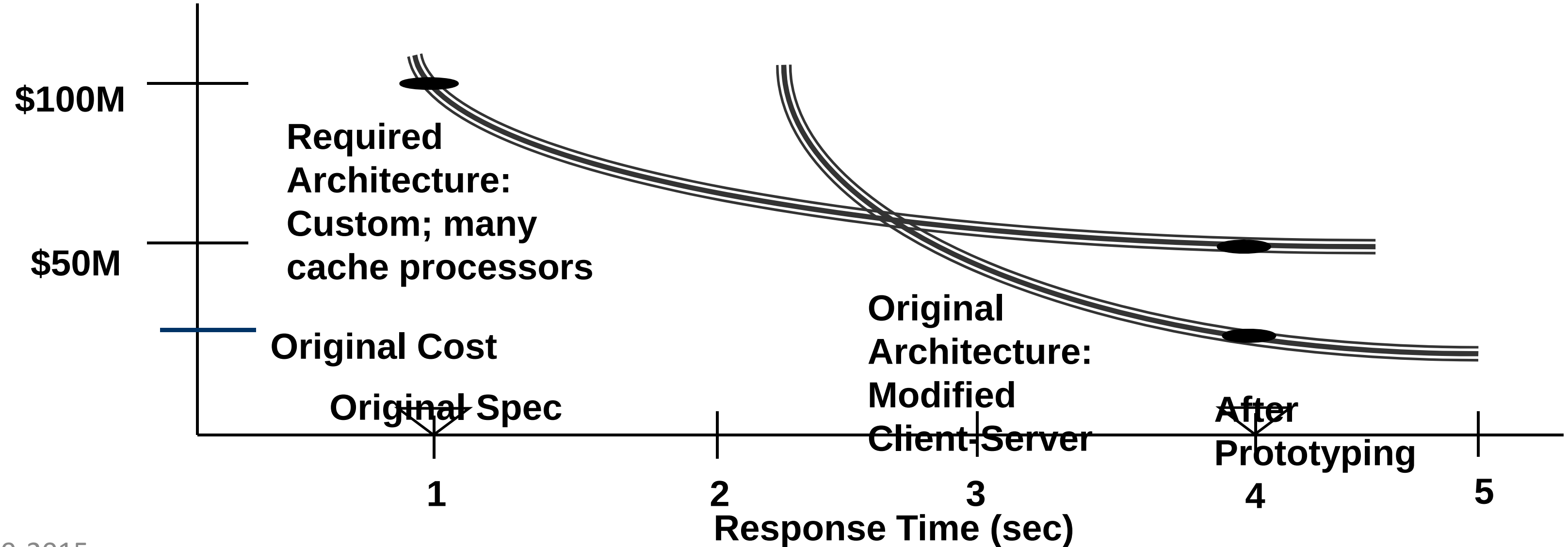
Outline

- Critical nature of system qualities (SQs)
  - Or non-functional requirements; ilities
  - Major source of project overruns, failures
  - Significant source of stakeholder value conflicts
  - Poorly defined, understood
  - Underemphasized in project management
- Need for SQs ontology
  - Nature of an ontology; choice of IDEF5 structure
  - Stakeholder value-based, means-ends hierarchy
  - Synergies and Conflicts matrix and expansions
    - Example means-ends hierarchy: Affordability

Importance of SQ Tradeoffs

Major source of DoD system overruns

- **SQs have systemwide impact**
  - System elements generally just have local impact
- **SQs often exhibit asymptotic behavior**
  - Watch out for the knee of the curve
- **Best architecture is a discontinuous function of SQ level**
  - "Build it quickly, tune or fix it later" highly risky
  - Large system example below



SYSTEMS ENGINEERING Research Center

Example of SQ Value Conflicts: Security IPT

- **Single-agent key distribution; single data copy**
  - **Reliability: single points of failure**
- **Elaborate multilayer defense**
  - **Performance: 50% overhead; real-time deadline problems**
- **Elaborate authentication**
  - **Usability: delays, delegation problems; GUI complexity**
- **Everything at highest level**
  - **Modifiability: overly complex changes, recertification**

Proliferation of Definitions: Resilience

- **Wikipedia Resilience variants: Climate, Ecology, Energy Development, Engineering and Construction, Network, Organizational, Psychological, Soil**

- **Ecology and Society Organization Resilience variants: Original-ecological, Extended-ecological, Walker et al. list, Folke et al. list; Systemic-heuristic, Operational, Sociological, Ecological-economic, Social-ecological system, Metaphoric, Sustainabilty-related**

- **Variants in resilience outcomes**
  - **Returning to original state; Restoring or improving original state; Maintaining same relationships among state variables; Maintaining desired services; Maintaining an acceptable level of service; Retaining essentially the same function, structure, and feedbacks; Absorbing disturbances; Coping with disturbances; Self-organizing; Learning and adaptation; Creating lasting value**
  - **Source of serious cross-discipline collaboration problems**

Example of Current Practice

- "The system shall have a Mean Time Between Failures of 10,000 hours"
- What is a "failure?"
  - 10,000 hours on liveness
  - But several dropped or garbled messages per hour?
- What is the operational context?
  - Base operations? Field operations? Conflict operations?
- Most management practices focused on functions
  - Requirements, design reviews; traceability matrices; work breakdown structures; data item descriptions; earned value management
- What are the effects on other SQs?
  - Cost, schedule, performance, maintainability?

Outline

- **Critical nature of system qualities (SQs)**
  - **Or non-functional requirements; ilities**
  - **Major source of project overruns, failures**
  - **Significant source of stakeholder value conflicts**
  - **Poorly defined, understood**
  - **Underemphasized in project management**

➡ **Need for system SQs ontology**
  - **Nature of an ontology; choice of IDEF5 structure**
  - **Stakeholder value-based, means-ends hierarchy**
  - **Synergies and Conflicts matrix and expansions**
    - **Example means-ends hierarchy: Affordability**

SYSTEMS ENGINEERING Research Center

Need for SQs Ontology

- Oversimplified one-size-fits all definitions
  - ISO/IEC 25010, Reliability: the degree to which a system , product, or component performs specified functions under specified conditions for a specified period of time
  - OK if specifications are precise, but increasingly “specified conditions” are informal, sunny-day user stories.
    - Satisfying just these will pass “ISO/IEC Reliability,” even if system fails on rainy-day user stories
  - Need to reflect that different stakeholders rely on different capabilities (functions, performance, flexibility, etc.) at different times and in different environments
- Proliferation of definitions, as with Resilience
- Weak understanding of inter-SQ relationships
  - Reliability Synergies and Conflicts with other qualities

Nature of an ontology; choice of IDEF5 structure

- An ontology for a collection of elements is a definition of what it means to be a member of the collection

- For “system qualities,” this means that an SQ identifies an aspect of “how well” the system performs
  - The ontology also identifies the sources of variability in the value of “how well” the system performs

- After investigating several ontology frameworks, the IDEF5 framework appeared to best address the nature and sources of variability of system SQs
  - Good fit so far

Initial SERC SQs Ontology

- **Modified version of IDEF5 ontology framework**
  - **Classes, Subclasses, and Individuals**
  - **Referents, States, Processes, and Relations**
- **Top classes cover stakeholder value propositions**
  - **Mission Effectiveness, Resource Utilization, Dependability, Flexibiity**
- **Subclasses identify means for achieving higher-class ends**
  - **Means-ends one-to-many for top classes**
  - **Ideally mutually exclusive and exhaustive, but some exceptions**
  - **Many-to-many for lower-level subclasses**
- **Referents, States, Processes, Relations cover SQ variation**
  - **Referents: Sources of variation by context: Product Q.; Q. In Use**
  - **States: Internal (beta-test); External (rural, temperate, sunny)**
  - **Processes: Operational scenarios (normal vs. crisis; experts vs. novices)**
  - **Relations: Impact of other SQs (security as above, synergies & conflicts)**

Referents: Product Quality; Quality in Use

- Product Quality: Anticipate future usage, build in added capabilities
  - Versatility: car with GPS, Bluetooth for mobile phone
  - Endurability: Extra-strong tires for off-road use
- Quality in Use: Usage profile stimulates need for changes
  - Modifiability: easy to add GPS, Bluetooth
  - Resilience: easy to adapt car for reliable off-road use
  - Or have a car with built-in Versatility, Endurability
- Both often called Changeability
  - Even though Versatile, Endurable product doesn't change

- MIT change-oriented semantic framework clarifies variations in causes and effects of changes

MIT 14-D Semantic Basis

Prescriptive Semantic Basis for Change-type Ilities

In response to "cause" in "context", desire "agent" to make some "change" in "system" that is "valuable"

Cause	Context	Phase	Agent	Impetus Change	System	Outcome Change	System	Valuable

In response to "perturbation" in "context" during "phase" desire "agent" to make some "nature" impetus to the design "parameter" with "destination(s)" in the "aspect" to have an "effect" to the outcome "parameter" with "destination(s)" in the "aspect" of the "abstraction" that are valuable with respect to thresholds in "reaction", "span", "cost" and "benefits"

Perturbation	Context	Phase	Agent	Impetus: Nature	Impetus: Parameter	Impetus: Destination	Impetus: Aspect	Outcome: Effect	Outcome: Parameter	Outcome: Destination	Outcome: Aspect	Abstraction	Reaction	Span	Cost	Benefit
					"parameter"	"state"			"parameter"	"state"			"threshold"	"threshold"	"threshold"	"threshold"
disturbance	circumstantia	pre-ops	internal	increase	level	one	form	increase	level	one	form	architecture	sooner	shorter	less	more
shift	general	ops	external	decrease	set	few	function	decrease	set	few	function	design	later	longer	more	less
none	any	inter-LC	either	not-same	any	many	operations	not-same	any	many	operations	system	always	same	same	same
any		any	none	same		any	any	same		any	any	any	any	any	any	any
			any	any				any								

Perturbation	Context	Phase	Agent	Impetus: Nature	Impetus: Parameter	Impetus: Destination	Impetus: Aspect	Outcome: Effect	Outcome: Parameter	Outcome: Destination	Outcome: Aspect	Abstraction	Reaction	Span	Cost	Benefit	Ility Label
shift		ops						same	"Value"	few							Value Robustness
disturbance		ops						same	"Value"	few							Value Survivability
shift		ops						same		few							Robustness
shift		ops		not-same				same		few							Active Robustness
shift		ops		same		few		same		few							Passive Robustness
shift		ops	none	same		few		same	level	few	form	system					Classical Passive Robustness
disturbance		ops						same		few							Survivability
			either	not-same				not-same									Changeability
shift	general	inter-LC		not-same				not-same				architecture					Evolvability
			internal	not-same				not-same									Adaptability
			external	not-same				not-same									Flexibility
				not-same				not-same	level								Scalability
				not-same				not-same	set								Modifiability
		ops	either	not-same				increase	set								Extensibility
				not-same				not-same	any					shorter			Agility
				not-same				not-same	any				sooner				Reactivity
		ops		same	"Element set"	one	form	not-same	"Link set"		form						Form Reconfigurability
		ops		same	"Element set"	one	operations	not-same	"Order set"		operations						Operational Reconfigurability
		ops		same		one	form/ops	not-same	set	few/many							Versatility
		ops		same		one	form/ops	not-same	set	few/many	function						Functional Versatility
		ops		same		one	form/fnct	not-same	set	few/many	operations						Operational Versatility
		ops		same		one	fnct/ops	not-same	set	few/many	form						Substitutability

SYSTEMS ENGINEERING
Research Center

Example: Reliability Revisited

- Reliability is the probability that the system will deliver stakeholder-satisfactory results for a given time period (generally an hour), given specified ranges of:
  - Stakeholders: desired and acceptable ranges of liveness, accuracy, response time, speed, capabilities, etc.
  - System internal and external states: integration test, acceptance test, field test, etc.; weather, terrain, DEFCON, takeoff/flight/landing, etc.
  - System internal and external processes: security thresholds, types of payload/cargo; workload volume, diversity
  - Effects of other SQs: synergies, conflicts

Stakeholder value-based, means-ends hierarchy

- **Mission operators and managers want improved Mission Effectiveness**
  - **Involves Physical Capability, Cyber Capability, Human Usability, Speed, Accuracy, Impact, Endurability, Maneuverability, Scalability, Versatility, Interoperability**
- **Mission investors and system owners want Mission Cost-Effectiveness**
  - **Involves Cost, Duration, Personnel, Scarce Quantities (capacity, weight, energy, …); Manufacturability, Sustainability**
- **All want system Dependability: cost-effective defect-freedom, availability, and safety and security for the communities that they serve**
  - **Involves Reliability, Availablilty, Maintainability, Survivability, Safety, Security, Robustness**
- **In an increasingly dynamic world, all want system Flexibility: to be rapidly and cost-effectively changeable**
  - **Involves Modifiability, Tailorability, Adaptability**

SYSTEMS ENGINEERING Research Center

U. Virginia: Coq Formal Reasoning Structure

- Inductive Dependable (s: System): Prop :=
- mk_dependability: Security s -> Safety s -> Reliability s ->
- Maintainability s -> Availability s -> Survivability s ->
- Robustness s -> Dependable s.

- Example aSystemisDependable: Dependable aSystem.
- apply mk_dependability.
- exact (is_secure aSystem).
- exact (is_safe aSystem).
- exact (is_reliable aSystem).
- exact (is_maintainable aSystem).
- exact (is_avaliable aSystem).
- exact (is_survivable aSystem).
- exact (is_robust aSystem).
- Qed.

Outline

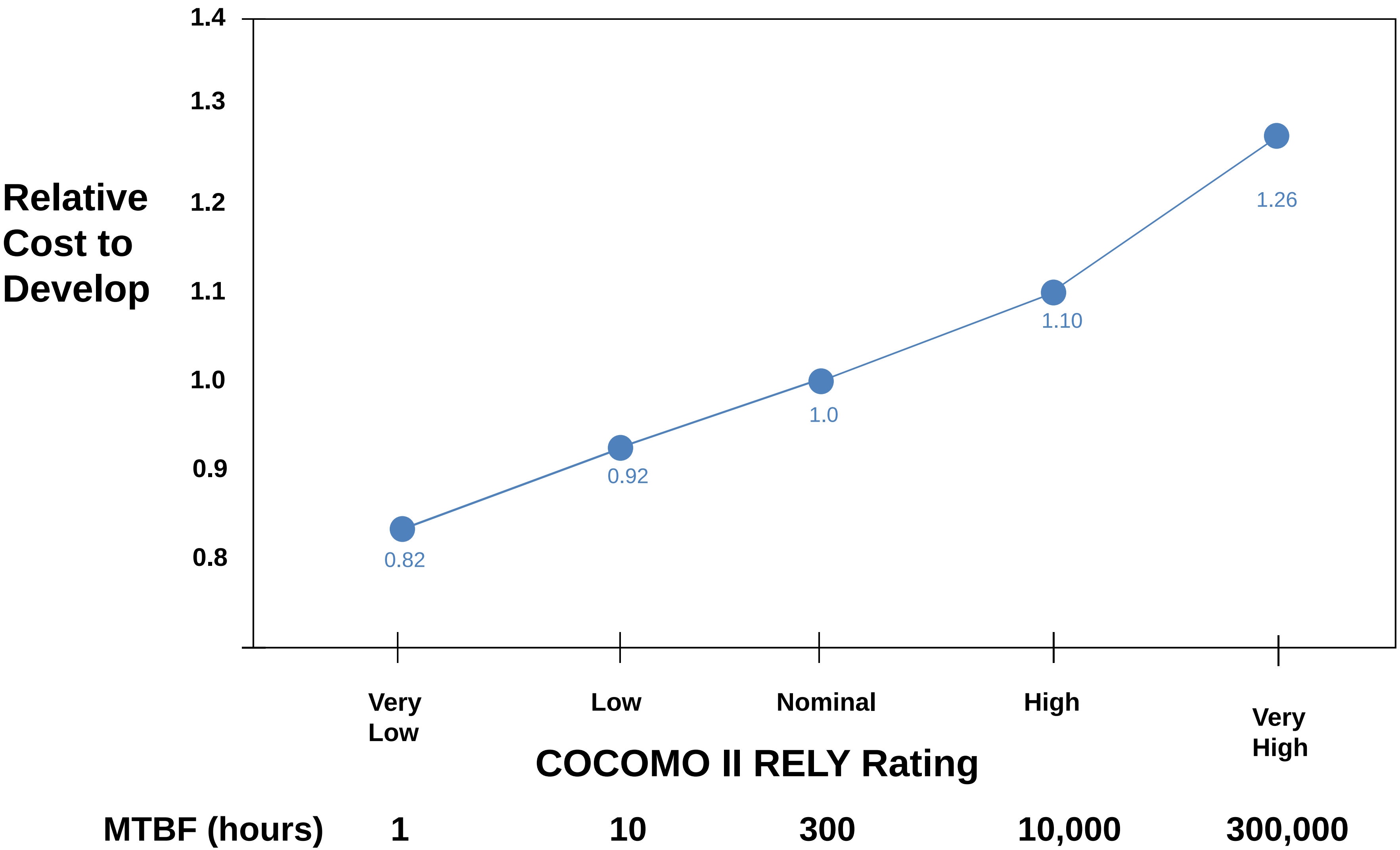
- Critical nature of system qualities (SQs)
  - Or non-functional requirements; ilities
  - Major source of project overruns, failures
  - Significant source of stakeholder value conflicts
  - Poorly defined, understood
  - Underemphasized in project management
- Need for SQs ontology
  - Nature of an ontology; choice of IDEF5 structure
  - Stakeholder value-based, means-ends hierarchy
  - ➡ Synergies and Conflicts matrix and expansions
    - Example means-ends hierarchy: Affordability

7x7 Synergies and Conflicts Matrix

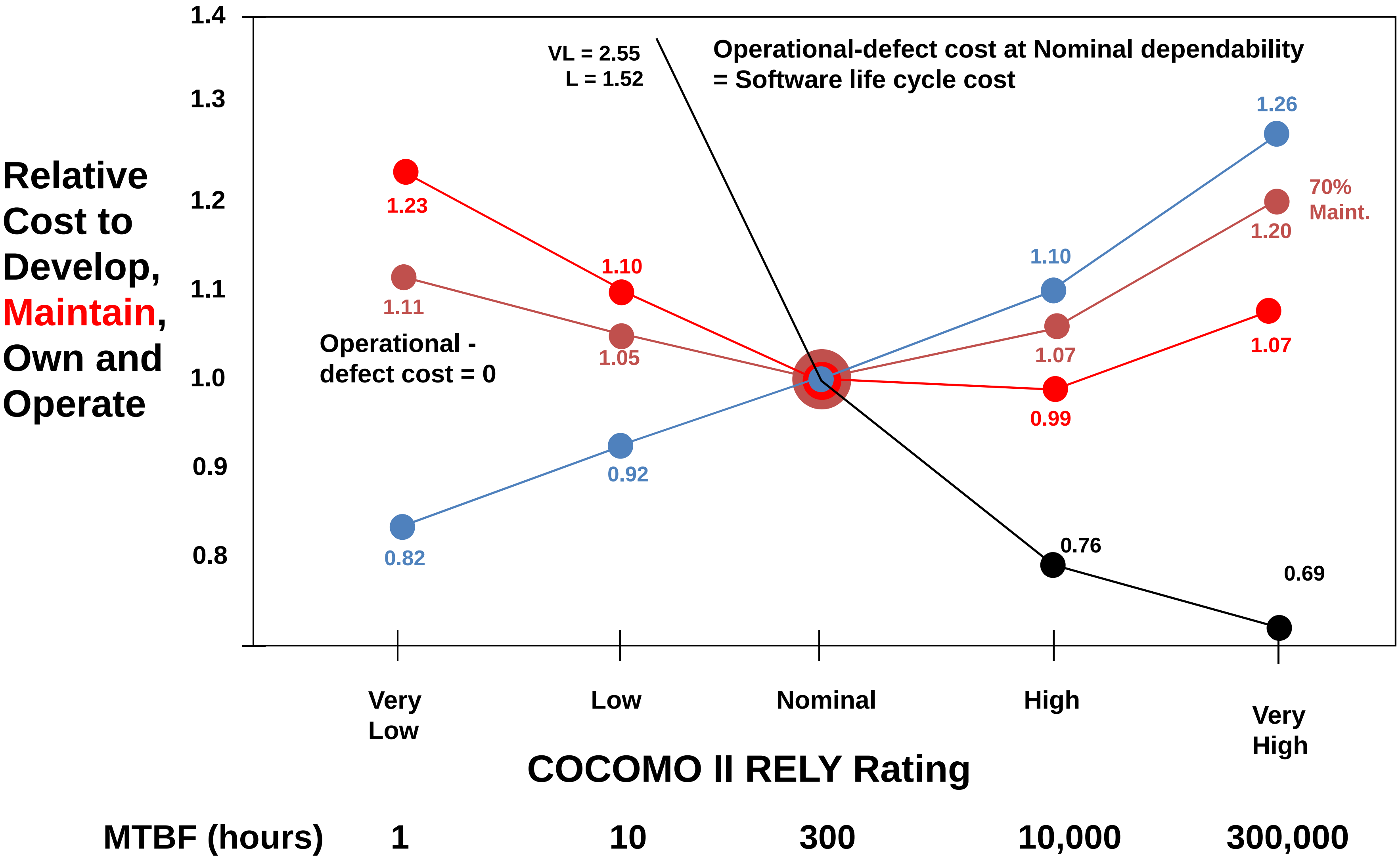
- **Mission Effectiveness expanded to 4 elements**
  - **Physical Capability, Cyber Capability, Interoperability, Other Mission Effectiveness (including Usability as Human Capability)**
- **Synergies and Conflicts among the 7 resulting elements identified in 7x7 matrix**
  - **Synergies above main diagonal, Conflicts below**
- **Work-in-progress tool will enable clicking on an entry and obtaining details about the synergy or conflict**
  - **Ideally quantitative; some examples next**
- **Still need synergies and conflicts within elements**
  - **Example 3x3 Dependability subset provided**

	Flexibility	Dependability	Mission Effectivenss	Resource Utilization	Physical Capability	Cyber Capability	Interoperability
Flexibility		Domain architecting within domain Modularity Self Adaptive Smart monitoring Spare Capacity Use software vs. hardware	Adaptability Many options Service oriented Spare capacity User programmability Versatility	Adaptability Agile methods Automated I/O validation Loose coupling for sustainability Product line architectures Staffing, Empowering	Adaptability Spare capacity	Adaptability Spare capacity	Adaptability Loose coupling Modularity Product line architectures Service-oriented connectors Use software vs. Hardware User programmability
Dependability	Accreditation Agile methods assurance Encryption Many options Multi-domain modifiability Multi-level security Self Adaptive defects User programmability		Accreditation FMEA Multi-level security Survivability Spare capacity	Automated aids Automated I/O validation Domain architecting within domain Product line architectures Staffing, Empowering **Total Ownership Cost** Value prioritizing	Fallbacks Lightweight agility Redundancy Spare capacity Value prioritizing	Fallbacks Redundancy Value prioritizing	Assertion Checking Domain architecting within domain Service oriented
Mission Effectivenss	Autonomy vs. Usability Modularity slowdowns Multi-domain architecture interoperability conflicts Versatility vs. Usability	Anti-tamper Armor vs. Weight Easiest-first development Redundancy Scalability Spare Capacity Usability vs. Security		Automated aids Domain architecting within domain Staffing, Empowering Value prioritizing	Automated aids Domain architecting within domain Staffing, Empowering Value prioritizing	Automated aids Domain architecting within domain Staffing, Empowering Value prioritizing	Automated aids Domain architecting within domain Staffing, Empowering
Resource Utilization	Agile Methods scalability Assertion checking overhead Fixed cost contracts Modularity Multi-domain architecture interoperability conflicts Spare capacity Tight coupling Use software vs. hardware	Accreditation **Acquisition Cost** Certification Easiest-first development Fallbacks Multi-domain architecture interoperability conflicts Redundancy Spare Capacity, tools costs Usability vs. Cost savings	Agile methods scalability Cost of automated aids Many options Multi-domain architecture interoperability conflicts Spare capacity Usability vs. Cost savings Versatility		Automated aids Domain architecting within domain Staffing, Empowering Value prioritizing	Automated aids Domain architecting within domain Staffing, Empowering Value prioritizing	Automated aids Domain architecting within domain Rework cost savings Staffing, Empowering
Physical Capability	Multi-domain architecture interoperability conflicts Over-optimizing Tight coupling Use software vs. hardware	Lightweight agility Multi-domain architecture interoperability conflicts Over-optimizing	Multi-domain architecture interoperability conflicts Over-optimizing	Cost of automated aids Multi-domain architecture interoperability conflicts Over-optimizing		Automated aids Staffing, Empowering Value prioritizing	Automated aids Domain architecting within domain
Cyber Capability	Agile Methods scalability Multi-domain architecture interoperability conflicts Over-optimizing Tight coupling Use software vs. hardware	Multi-domain architecture interoperability conflicts Over-optimizing	Multi-domain architecture interoperability conflicts Over-optimizing	Cost of automated aids Multi-domain architecture interoperability conflicts Over-optimizing	Over-optimizing Physical architecture or cyber architecture		Automated aids Domain architecting within domain
Interoperability	Multi-domain architecture interoperability conflicts User-programmed interoperability	Encryption interoperability Multi-domain architecture interoperability conflicts	Multi-domain architecture interoperability conflicts	Assertion checking Cost, duration of added connectors	Over-optimizing Tight vs. Loose coupling	Reduced speed of Assertion checking Reduced speed of connectors, standards compliance Tight vs. Loose coupling	

Software Development Cost vs. Reliability



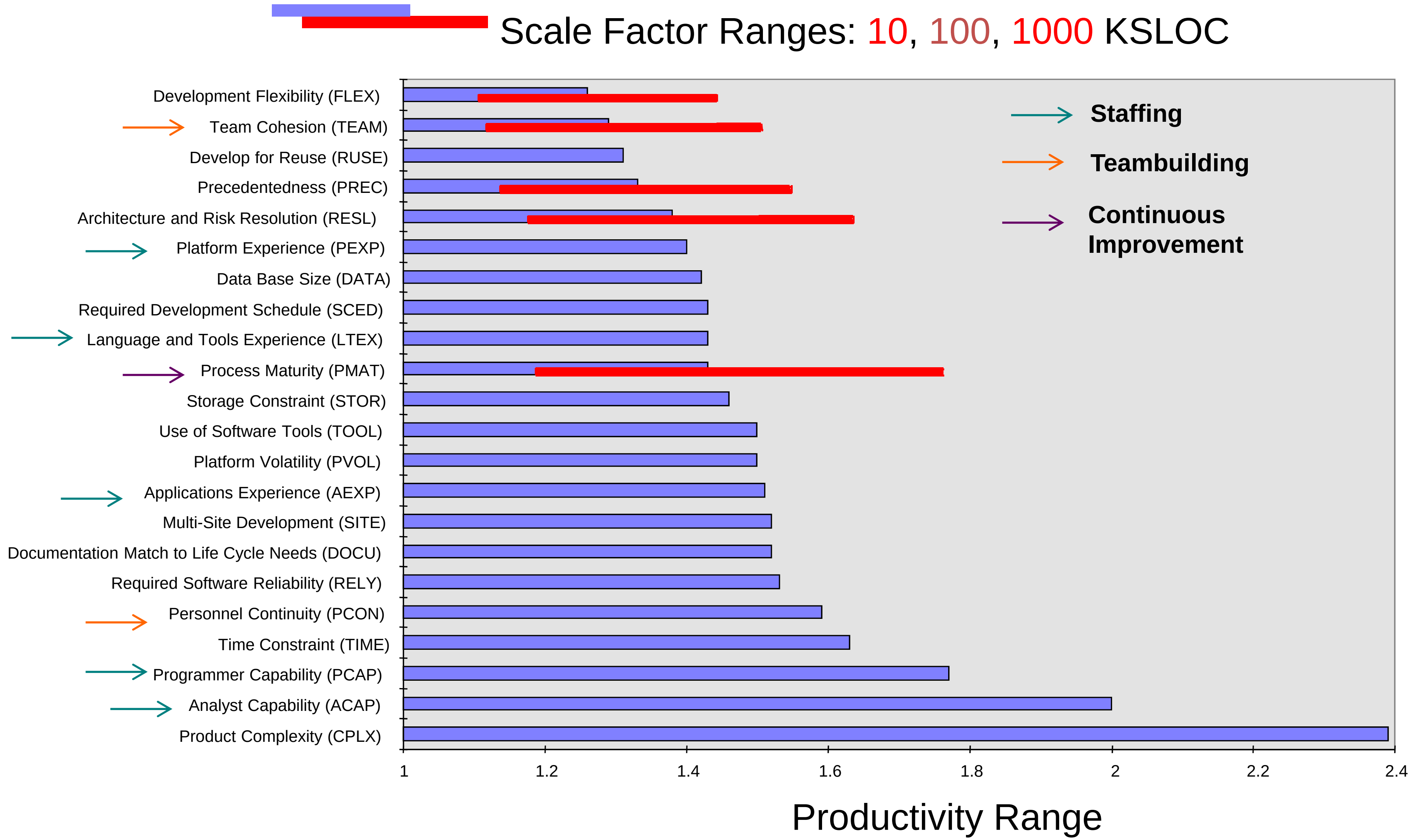
Software Ownership Cost vs. Reliability



Affordability and Tradespace Framework

- **Cost Improvements and Tradeoffs**
  - **Get the Best from People**
    - Staffing, Incentivizing, Teambuilding
    - Facilities, Support Services
    - Kaizen (continuous improvement)
  - **Make Tasks More Efficient**
    - Tools and Automation
    - Work and Oversight Streamlining
    - Collaboration Technology
  - **Eliminate Tasks**
    - Lean and Agile Methods
    - Task Automation
    - Model-Based Product Generation
  - **Eliminate Scrap, Rework**
    - Early Risk and Defect Elimination
    - Evidence-Based Decision Gates
    - Modularity Around Sources of Change
    - Incremental, Evolutionary Development
    - Value-Based, Agile Process Maturity
    - Risk-Based Prototyping
  - **Simplify Products (KISS)**
    - Value-Based Capability Prioritization
    - Satisficing vs. Optimizing Performance
  - **Reuse Components**
    - Domain Engineering and Architecture
    - Composable Components,Services, COTS
    - Legacy System Repurposing
  - **Reduce Operations, Support Costs**
    - Automate Operations Elements
    - Design for Maintainability, Evolvability
    - Streamline Supply Chain
    - Anticipate, Prepare for Change
  - **Value- and Architecture-Based Tradeoffs and Balancing**

Costing Insights: COCOMO II Productivity Ranges



SYSTEMS ENGINEERING
Research Center

Conclusions

- System qualities (SQs) are success-critical
  - Major source of project overruns, failures
  - Significant source of stakeholder value conflicts
  - Poorly defined, understood
  - Underemphasized in project management

- SQs ontology clarifies nature of system qualities
  - Using value-based, means-ends hierarchy
  - Identifies sources of variation: states, processes, relations
  - Relations enable SQ synergies and conflicts identification

- Continuing SERC research creating tools, formal definitions