RequirementsExperts

Training and Services for Project Success

Information Based Requirement Development

Product Development for the 21st Century

Lou Wheatcraft

louw@reqexperts.com

www.reqexperts.com/blog

Seilevel

Lou Wheatcraft

- Senior Product Manager for Seilevel/Requirements Experts (RE)
- Has taught over 190 requirement seminars over the last 18 years.
- 22 years in the US Air Force
- Heavy involvement in space systems (DoD launch vehicles and spacecraft, NASA Space Shuttle, International Space Station)
- Worked in the Astronaut Office at Johnson Space Center for 6 years.
- Works with both government and industry clients.
- **Chair of the INCOSE Requirements Working Group**
- Member of PMI, the Software Engineering Institute (SEI), the World Futures Society, International Institute of Business Analysis (IIBA), and the National Honor Society of Pi Alpha Alpha.
- Has a BS degree in Electrical Engineering, MA degree in Computer Information Systems, MS degree in Environmental Management, and has completed the course work for an MS degree in Studies of the Future
- Author of numerous papers and presentations concerning requirement development and management
- Is the primary contributor to RE's blog on requirements best practices. The blog can be assessed at: http://www.reqexperts.com/blog .

INCOSE

Integrated Data as a Foundation of Systems Engineering

September 2018

Whitepaper by the Requirements Working Group

Background

We are 18 years into the 21st century – why are we still using 20th century methods and processes?

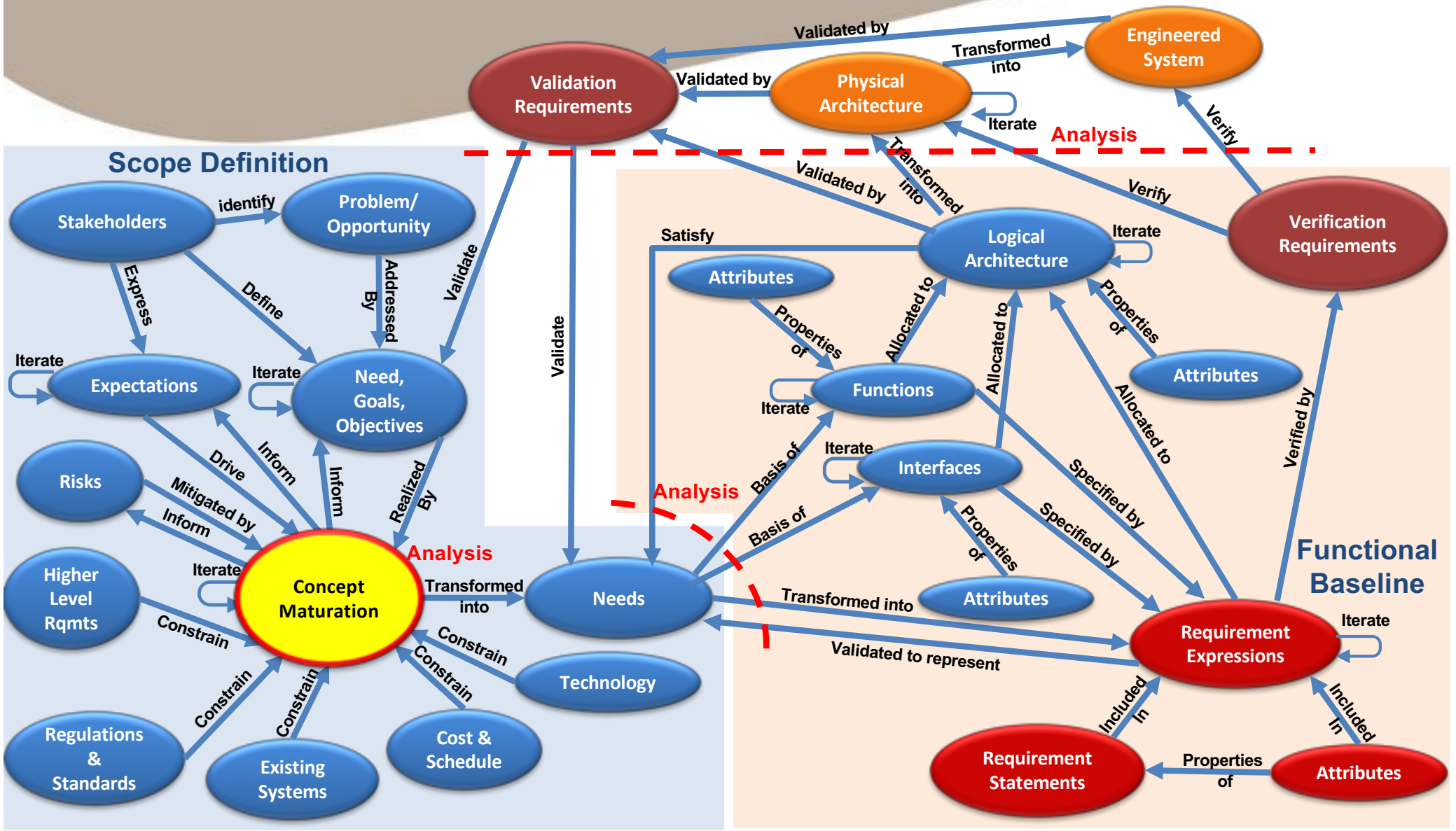
Background

- **Today's product development environment presents key challenges:**
  - Increasing complexity
  - Increasing role software has in the system architecture
  - Increasing dependencies between key parts of the system
  - Decreasing time to market expectations
  - Increasing risks

Approaches to address these challenges

- **Incorporate systems thinking into all phases of product development**
  - Focus on interdependencies of the parts that make up the system
  - Top down definition: system, element, subsystem, etc.
  - May have to sub-optimize the subsystems to optimize the system
- **Move software up in the system's architecture hierarchy**
  - Move from a hardware-centric view to a software-centric view
- **Communicate requirements at the proper level**
- **Recognize the importance of well-formed and managed requirements to the success of a project**
- **Define scope and stakeholder needs before developing requirements**
- **Address the feasibility of a concept before developing requirements**
  - Increased use of modeling to help ensure completeness, consistency, and correctness of stakeholder needs and resulting requirements
- **Transform stakeholder needs into well-formed requirements**
- **Adopt an information based approach to requirement development and management**

Information-based Requirement Development & Management



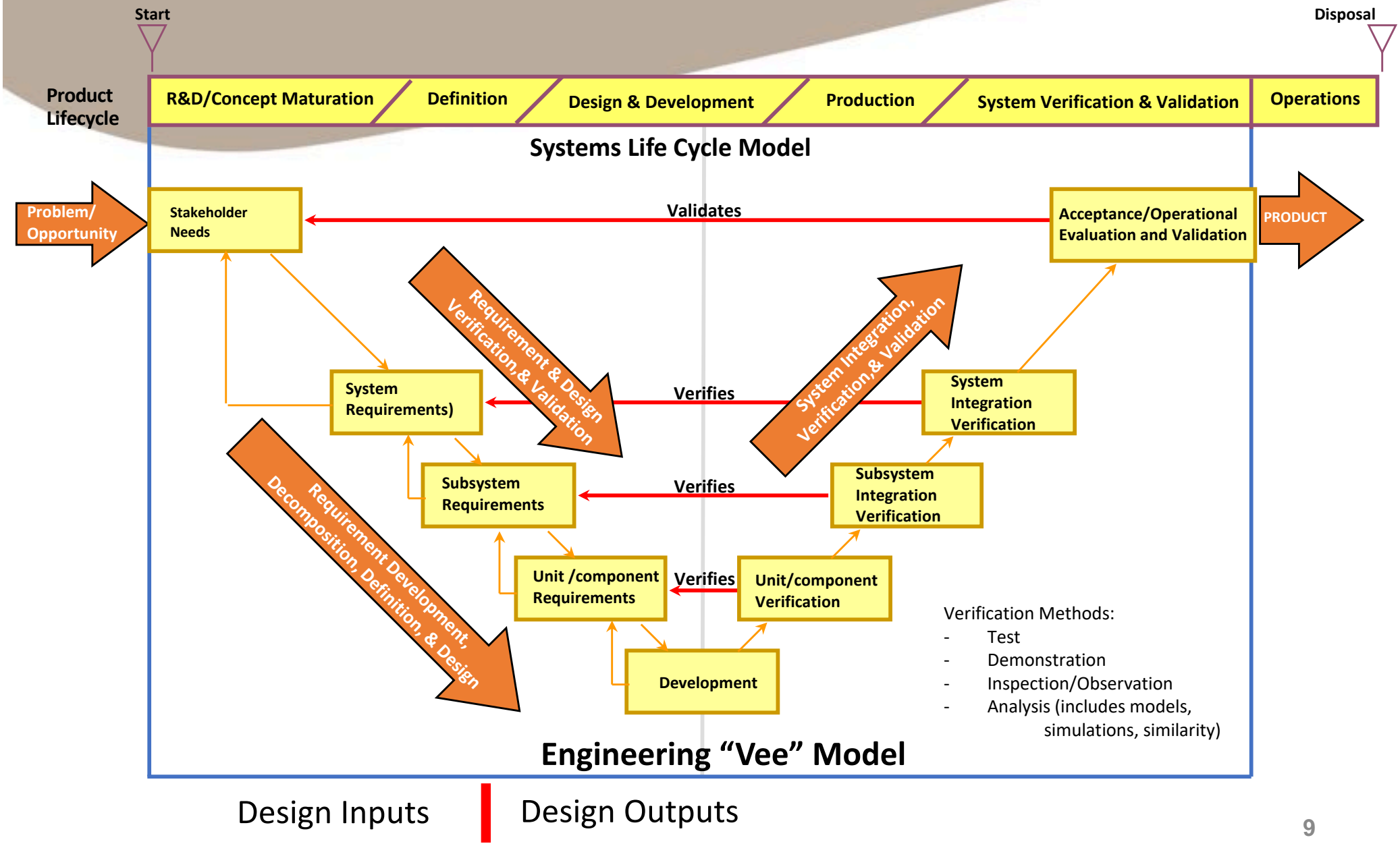
Traceability is critical to document relationships

RequirementsExperts

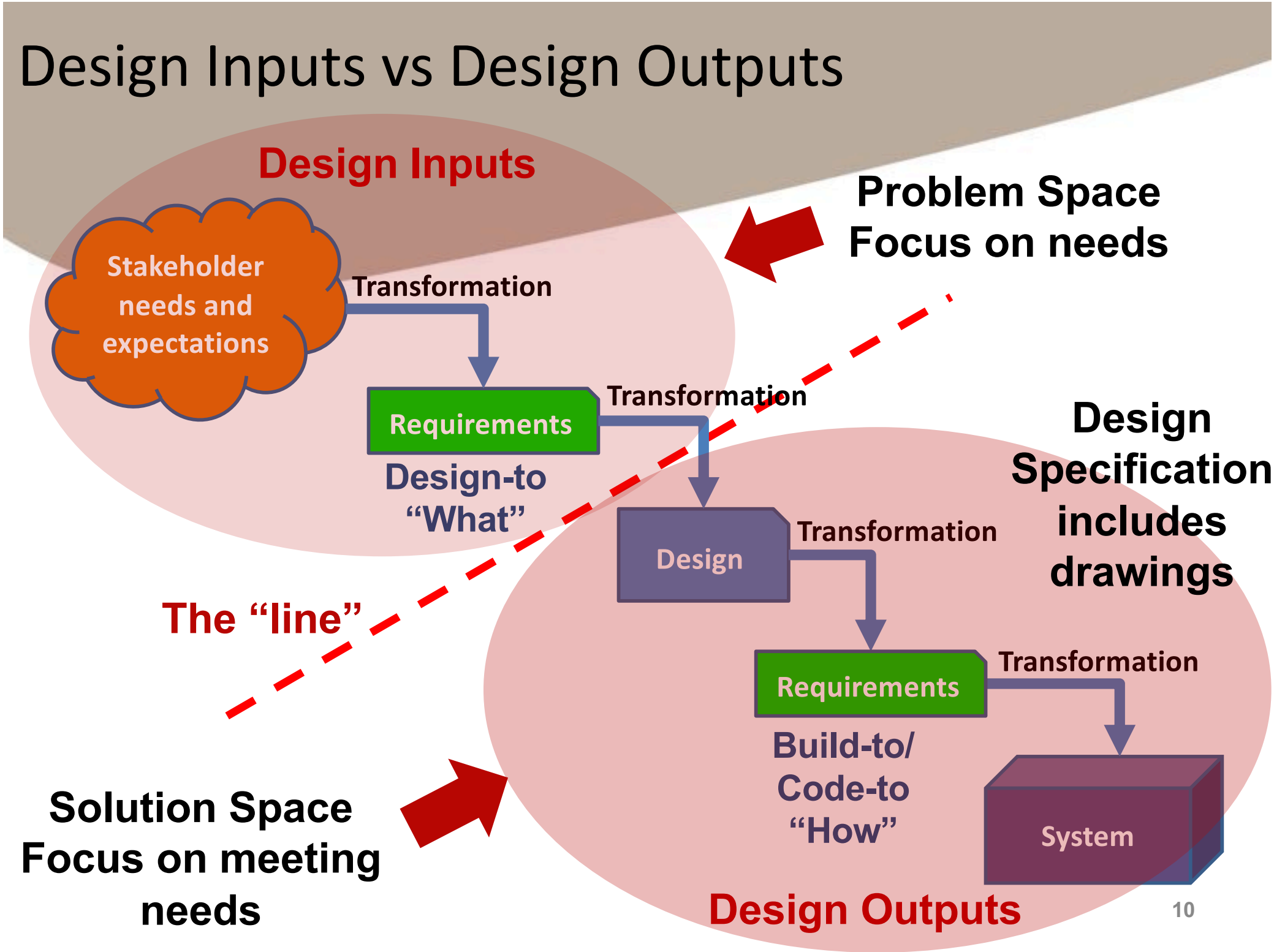
Training and Services for Project Success

Communicating Requirements at the Appropriate Level

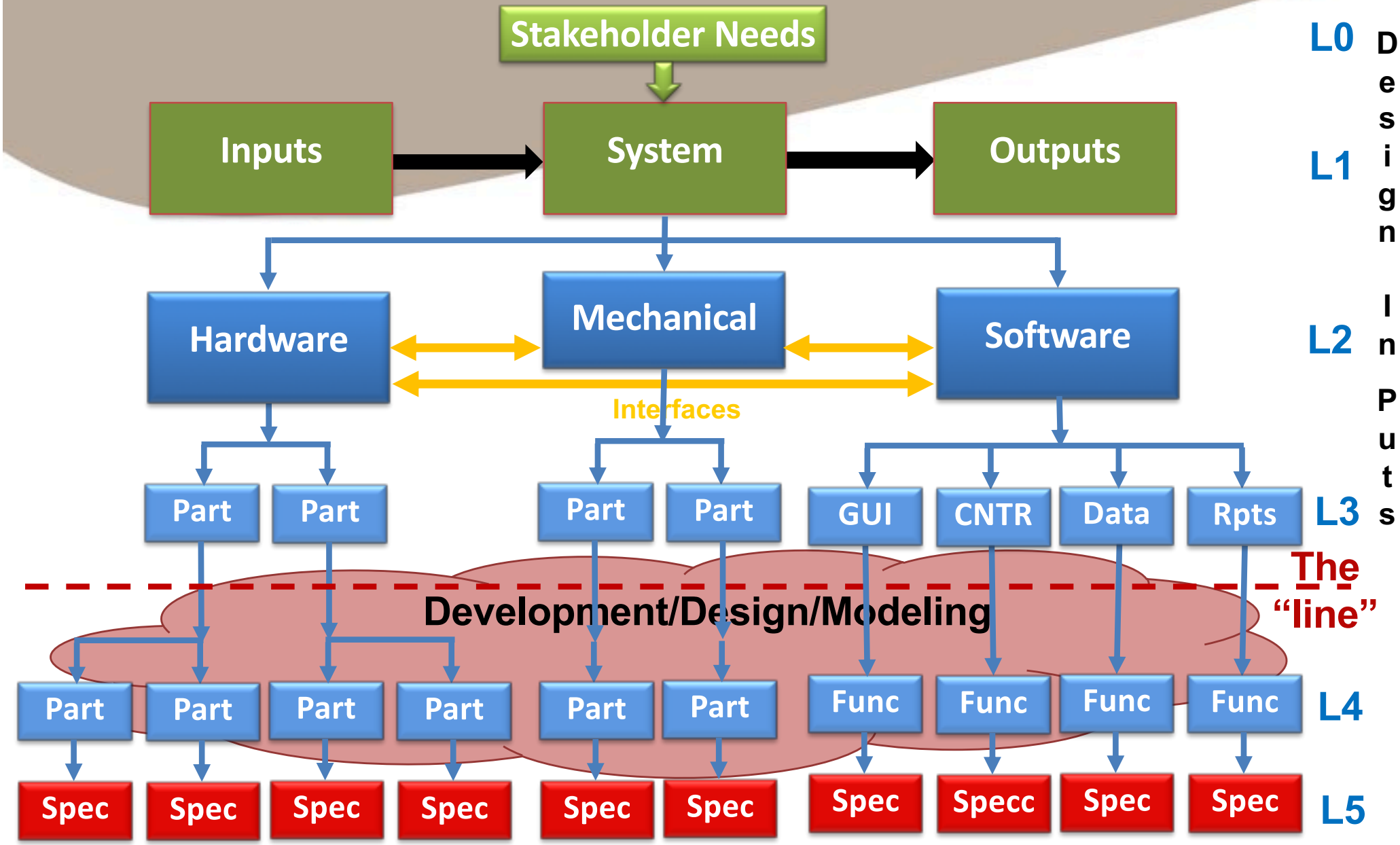
Systems Engineering "V" Model



Design Inputs vs Design Outputs



Requirements Architecture - Software



Design Outputs

Common Level Problems

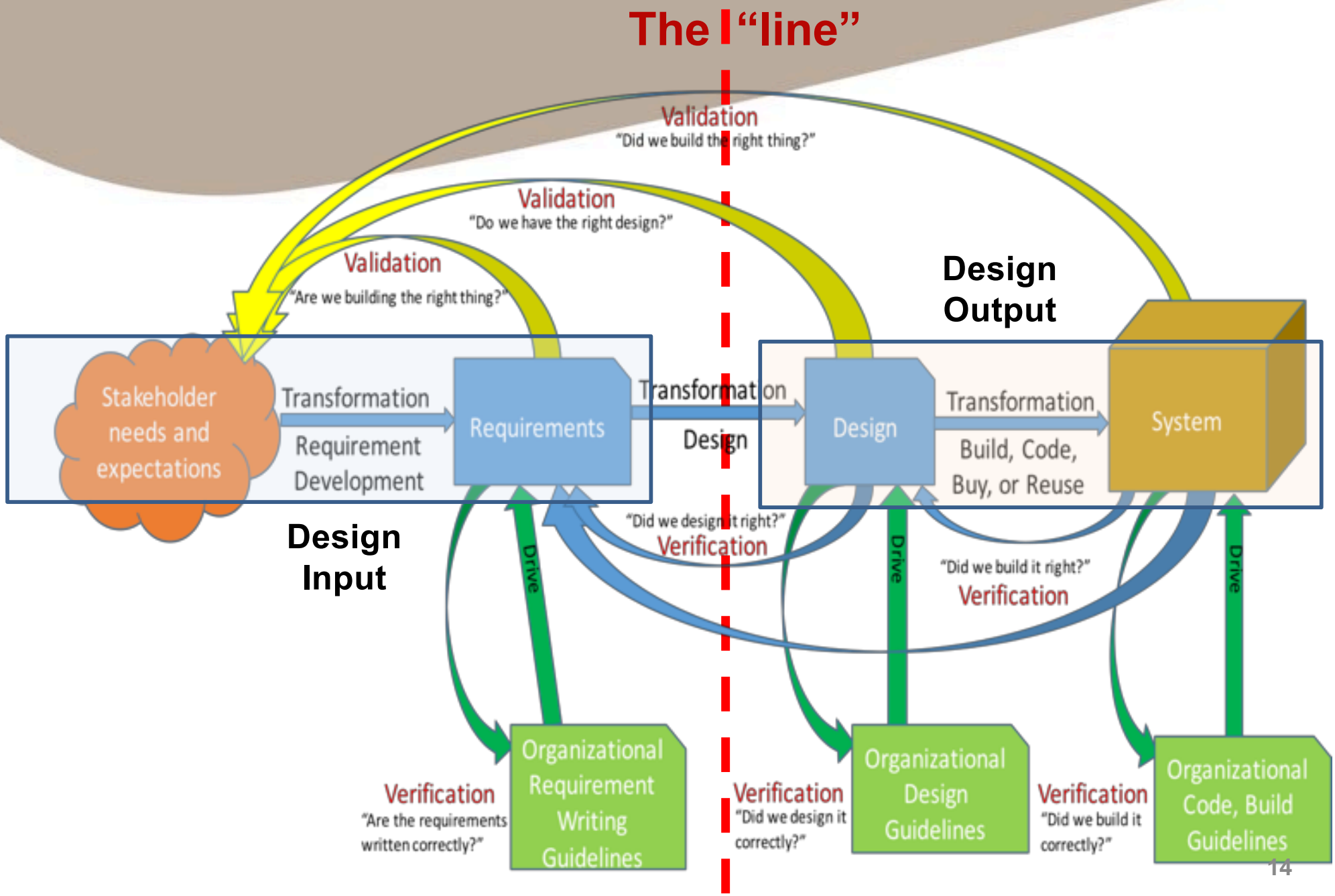
- Requirements at the wrong level
  - Below “the line” build-to, design output requirements communicated above “the line” in the design-to, design input requirement set
- Higher-level requirements not implemented at lower levels
  - parents without children
- Lower-level requirements that cannot be justified by higher-level requirements
  - gold plating or parentless children
- Inadequate impact assessment of changes to requirements
- Allocation & Traceability help address these problems

RequirementsExperts

Training and Services for Project Success

Recognizing the Importance of Well-formed and Managed Requirements

Verification and Validation



Importance of Requirements

"Requirements are the common thread that ties all the product development lifecycle phases together." Lou Wheatcraft

"Developing requirements is not an exercise in writing, but is an exercise in engineering. Every requirement represents an engineering decision as to what the system needs do or a quality the system needs to have in order to meet stakeholder needs." Lou Wheatcraft

Benefits of Well-Written Requirements

- Clearly communicate the stakeholder needs to the design team
- Establish the basis for agreement between the stakeholders and the developers on what the product is to do
- Reduce the impacts on cost & schedule due to rework due to omissions, misunderstandings, and inconsistencies
- Provide a basis for estimating costs & schedules
- Provide a baseline for design
- Provide a baseline for system verification
- **Result in satisfied customers**
- **Result in increased profits**

RequirementsExperts

Training and Services for Project Success

Defining Scope and Stakeholder Needs Before Developing Requirements

Scope is:

The set of information that provides a clear vision and common understanding of stakeholder expectations and needs for those who will write, review, and manage system requirements or have a significant interest in the system across its lifecycle.

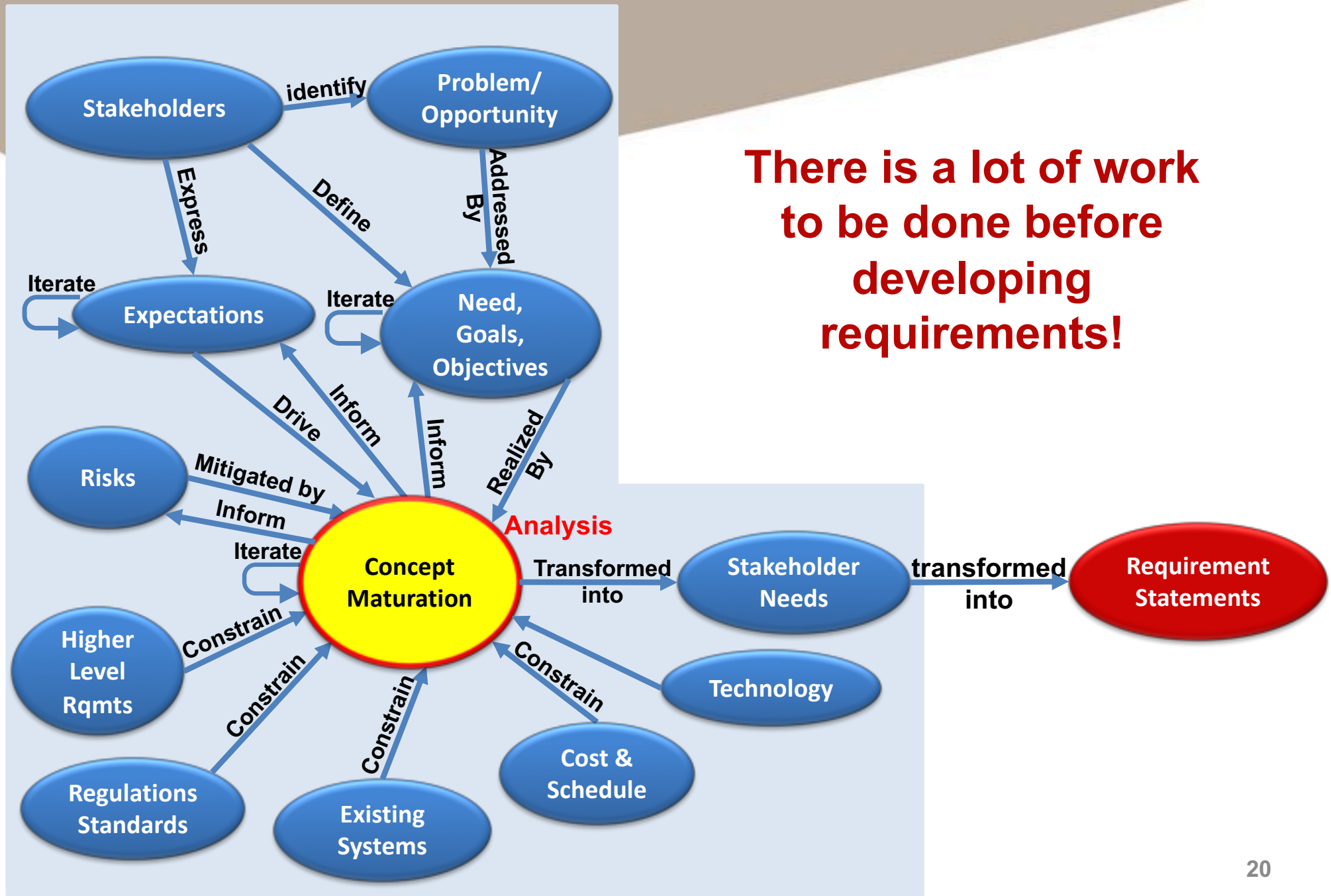
Scope definition activities include concept definition and maturation

Gathering the information needed to minimize risk and build a firm foundation for developing requirements

Scope Includes:

- Identifying stakeholders
- Defining the problem/opportunity
- Defining need, goals, objectives
- Eliciting stakeholder expectations
- Documenting risks, drivers, & constraints
- Developing a feasible concept
- Documenting an *integrated set of Stakeholder Needs*
- Baselining scope **before developing requirements**

Components of Scope Definition



There is a lot of work to be done before developing requirements!

RequirementsExperts

Training and Services for Project Success

Addressing the Feasibility of a System Concept Before Baselining Stakeholder Needs and Transforming Those Needs into Requirements

Feasible System Concept

- A concept that is proven to address the need, goals, objectives and meets stakeholder expectations within the defined drivers and constraints with acceptable risk
- Source of stakeholder needs that will be transformed into requirement statements for the entity (system) under development

Ensure that the technical team has defined a feasible concept that has been agreed to by the stakeholders before writing requirements.

Focus areas:

- **Form**
  - the shape, size, dimensions, mass, weight and other visual parameters that uniquely distinguish a system
- **Fit**
  - the ability of the system to physically interface with, connect to, or become an integral part of the macro system it is a part
  - Includes human system interactions and user interfaces
- **Function**
  - the action or actions that a part is designed to perform
  - Includes functionality and associated performance
- **Quality**
  - ”-ilities” – reliability, availability, operability, supportability, manufacturability, maintainability, interoperability, safety, security
- **Compliance**
  - With standards and regulations

Methods to help define and mature a feasible system concept

- **Functional Analysis**
  - Design Reference Missions
  - Operational Scenarios
  - Models and Diagrams (MBSE)
  - Movies/pictures
- **Risk Assessment**
  - Fault Tree Analysis
  - Failure Modes & Effects Analysis (FMEA)
  - Technology Risk Assessment (TRA)
- **Risk Mitigation**
  - Technology Readiness Levels (TRLs)
  - Margins & Reserves
  - Concept Readiness Levels (CRLs)
- **Selection of Alternatives**
  - Trade Studies
- **Demonstrations**
  - Animations
  - Simulations
  - Prototypes

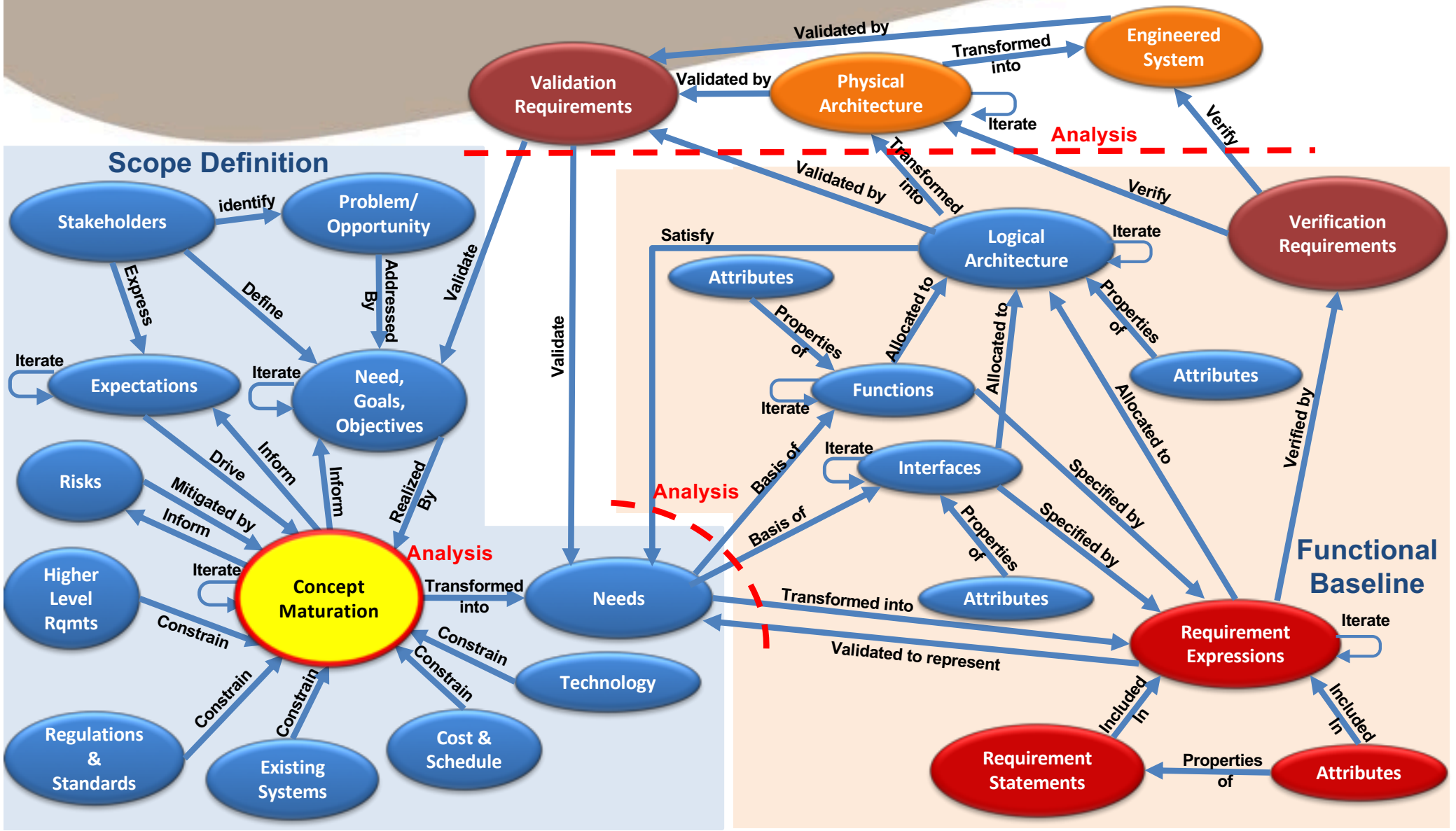
Using modeling techniques as part of concept definition and maturation

- **Functional diagrams**
  - Write functional requirements for each function
    - Link the functional requirements to the function
  - Write performance requirements for each function
    - Link the performance requirements to the associated functional requirement
- **Interface block diagrams**
  - Write interface requirements for each entity in the diagram
    - Written in pairs
    - Link interface requirements to the entity the requirement pertains to
  - Need to define the characteristics of the thing crossing the interface (ICD)
- **Non-functional requirements**
  - From the needs, identify all the non-functional requirements
    - Operational (if not covered above), quality (-ilities), standards, regulations, physical characteristics
  - Write specific requirements addressing the needs
  - Link them to the system of interest to which they apply

Models and Diagrams - Benefits

- Models and diagrams are excellent methods for defining and maturing a feasible concept
- Provide context for requirements
  - Helps ensure correctness, completeness, and consistency
- Make complex systems and processes easier to understand
- Facilitate communication
- Identify interdependencies
- Help to mature concepts
- The resulting data and information model can be used and matured in later SE development lifecycles.

Information-based Requirement Development & Management



Traceability is critical to document relationships

Technology Readiness Levels (TRLs)

	TRLs (generic)
1	Basic principles observed and reported
2	Technology concept and/or application formulated
3	Analytical and experimental critical function and/or characteristic proof-of-concept
4	Component and/or breadboard validation in laboratory environment
5	Component and/or breadboard validation in relevant environment
6	System/subsystem model or prototype demonstration in a relevant end-to-end environment
7	System prototype demonstration in an operational environment
8	Actual system completed and qualified through test and demonstration
9	Actual system proven through successful operations

TRLs

- Used to manage project and development risk
  - The lower the TRL the higher the risk
  - The higher the TRL the lower the risk
- Can assign desired TRL for each product development stage
  - Best practice is to not incorporate a given technology into a product unless the TRL is at least 3 at scope baseline and have a plan to advance the TRL to 6 by preliminary design approval
- Product launch dates should be based on the TRL of the critical technologies being included in a given product.
- TRLs can be combined with Concept Maturity Levels (CMLs) to determine the maturity of a concept

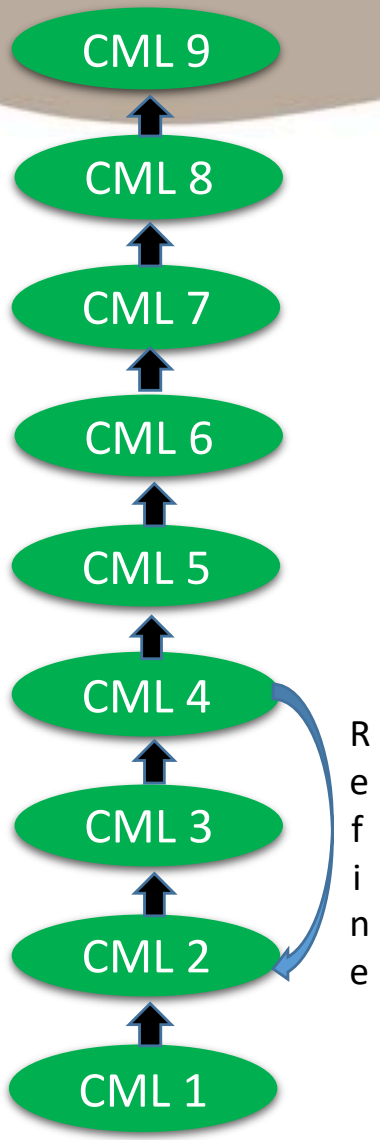
Assessing the Maturity of System Concepts

- **A dilemma faced by many organizations when deciding whether a project is mature enough to fund or proceed to the next life cycle is how to:**
  - Evaluate the feasibility (cost, schedule, technology) of a concept (project, product, system, mission) and its fulfillment of the project's Need, Goals, and Objectives (NGOs) and stakeholder expectations within the defined drivers and constraints
  - Assess whether or not the project in on track to deliver an acceptable ROI with acceptable risk
  - Determine if the maturity of the system concept, critical technologies, available resources, and associated planning are sufficient to:
    - approve additional funding, or
    - conduct the gate review, baseline the deliverables associated with the review (scope, requirements, design, project and technical plans), and proceed with the next lifecycle phase of product development

The answer: Concept Maturity Levels (CMLs)

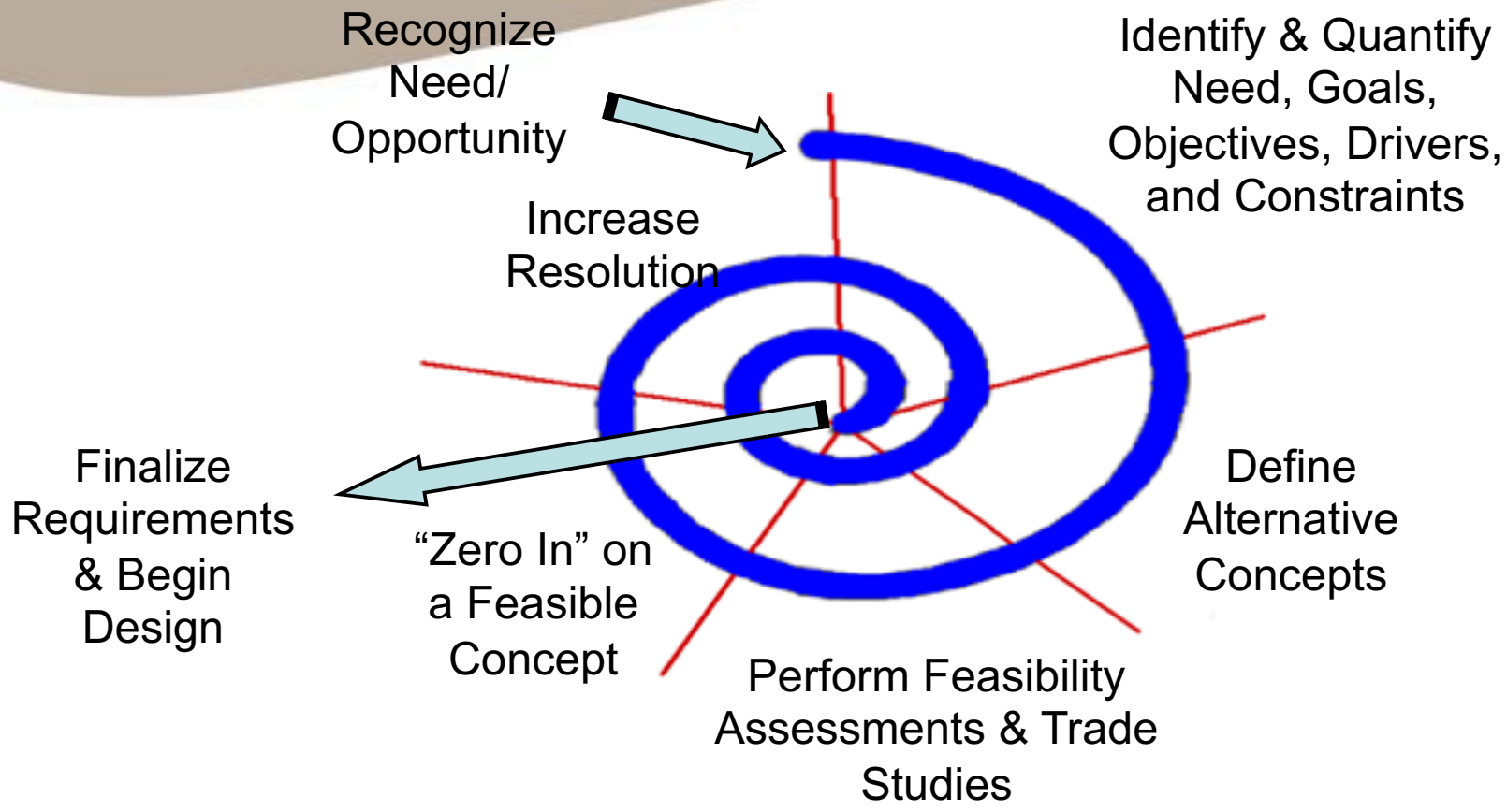
Based on JPL paper: "Space Mission Concept Development Using Concept Maturity Levels (CMLs)" (Wessen, R. R. et al 2013)

CMLs Defined



- **Critical Design Review (CDR), TRL 7,** "build-to" requirements and drawings are 80%-90% complete. ICDs are complete
- **Preliminary Design review (PDR), TRL 6,** "build-to" requirements and drawings are 10%-20% complete, interfaces defined, final integrated cost-schedule-design is baselined
- **System Design Review (SDR), TRL 5**: trades completed, feasible design identified
- **System Requirements Review (SRR), TRL 4:** stakeholder needs transformed into technical requirements.
- **Scope/Concept Review Baseline, TRL 3:** system concept baselined, stakeholder needs baselined
- **Point Design:** Candidate system physical architectures are identified TRLs defined, prototyping
- **Trade Space:** Functional architecture defined, candidate physical architectures evaluated for feasibility, TRA
- **Initial Feasibility:** initial concepts, risks, external interfaces, key measures, stakeholders engaged
- **"Cocktail Napkin":** Overview and Advocacy; problem; Need, goals, objectives; drivers & constraints defined

Doctrine of Successive Refinement



NASA Systems Engineering Handbook (NASA 1995)

CMLs

- The CML structure corresponds to an increasing level of maturity as the system concept, planning (project and technical), design, architecture, and risks are analyzed and evolve
- The CMLs apply to the left side of the SE "Vee" Model
- Using CMLs go a long way in reducing development risk by minimizing problems and cost over runs that often occur on the right side of the SE Vee Model during system integration, verification, and validation
- CML(s) provide the ability to measure a system concept's maturity guided by an incremental set of maturity criteria
  - CML Matrix
  - CML Checklists
- This defined maturity criteria can be tailored to correspond to the processes specific to a particular organization, domain, and project within that domain

Example CML Matrix

Life Cycle Phase	Pre-Phase A					Phase A	
	Advanced Studies			Concept Development		Early Formulation	
CML	1	2	3	4	5	6	7
Name	Cocktail Napkin	Initial Feasibility	Trade Space	Point Design	Baseline Concept	Integrated Concept	Preliminary Implementation Baseline
Life Cycle Gate	—	—	—	Concept Gate (Draft AO Out / Mission Study Report)	Baseline Commitment Gate / MCR	Step 2 Submittal	PMSR / MDR
Science							
Attribute							
Science Objectives & System Requirements	Science objectives described in one sentence	Objectives described to levels that allow comparison with previous investigations and NASA science community documents	Objectives linked to investigations and measurements Science return as a function of cost, risk and programmatics quantified	Produce draft Science Traceability Matrix Initial Level 1 requirements considered Specifying one Baseline and one Threshold Science investigation Key Performance Parameters listed	Science Traceability Matrix (or equivalent) produced Preliminary PLRA produced (assigned projects)	Proposed Level 1 requirements documented Level 2 & 3 driving requirements listed Full and minimum success criteria defined Baseline PLRA submitted @ SRR (assigned projects)	Update PLRA if necessary Preliminary Level 2 & 3 requirements listed
Science Data System	—	Identify science data drivers	Science data rates and volume included in trade space analysis	Science data system sizing	Science data processing architecture, release and archive approach defined	Science data management approach (includes Level 0, 1, 2 data products) defined	Same as for CML 6

Complete example matrix is included in JPL paper: "Space Mission Concept Development Using Concept Maturity Levels (CMLs)" (Wessen, R. R. et al 2013)

CML Matrix

- The intent of the CML matrix is to serve as a high-level guide for study/design and project teams through the stages of system concept maturation, architecture selection, and design.
- The matrix can be used by management and core project team in several ways to:
  1. Determine the maturity of a system concept at the time of a particular gate review.
     - As an example, by looking at the contents of the cells in the CML 5 column, a system architect can quickly see the material that is needed for a study/design team to pass their Mission Concept or Scope Review.
  2. Understand the deliverables and their maturity required as a function of time (life cycle stage).
  3. Use the contents of each column to generate a CML checklist for a specific CML.

Example CML checklist

CML 4 Checklist Sheet

2013 April 11

Functional Area	Criteria	Status (RYG)
	SCIENCE	
Science Objectives & Driving Requirements	o Draft Science Traceability Matrix produced	G
	o Initial Level 1 requirements considered	G
	o One Baseline and one Threshold Science investigation specified	G
	o Key Performance Parameters listed	G
Science Data System	o Science data system sized	R
	TECHNICAL	
Mission Development	o Driving requirements documented	G
	o Initital high-level scenarios, timelines and operational modes documented	G
	o Propellant load and delta-V requirements determined	G
	o Power generation and distribution approach defined	G
	o Telecommunication approach defined	G
	o Data processing approach documented	Y
	o Descope and backup options identified as needed	G
	o Launch period is 20 days long	G
Spacecraft or Instrument System Design	o System architecture & instrument designs (Earth Science & Astrophysics missions only) described by mechanical configuration drawings	G
	o System architecture & instrument designs (Earth Science & Astrophysics missions only) described by block diagrams	Y
	o Descope options compiled	G
	o Instrument performance requirements traced to level 1 requirements	Y
Ground System & Mission Operations System Design	o MOS / GDS architecture based on ops scenarios described	
Technical Risk Assessment & Management	o Risk drivers listed	
	o Top risks documented in 5 x 5 matrix (includes selected mitigation options)	
Technology	o Technology options characterized and baseline options selections and justified	
	o TRL for new technologies explained	
	o Fallback options for all new technologies identified	
Inheritance	o Major inherited assembly items tentatively selected	
Master Equipment Lists	o Assembly level (e.g., antenna, propellant tank, star tracker, etc.) MEL documented	

Complete example checklist is included in JPL paper: "Space Mission Concept Development Using Concept Maturity Levels (CMLs)" (Wessen, R. R. et al 2013)

CML Checklists

The CML Checklists:

- Allow management and the study/design team to quickly measure the system concept's maturity for a specific CML,
- Includes all technical and project management artifacts and work products defined in the product development process documentation
- Are reusable, i.e., the checklists can be applied to any project that is maturing their concepts, providing the same level of maturity score for concepts with the same level of maturity and,
- Identify deficiencies and provide clear information as to what areas of the concept need additional work to get to the overall mission concept to the desired level of maturity.

CML Summary

CMLs provide a standardized method to allow management to:

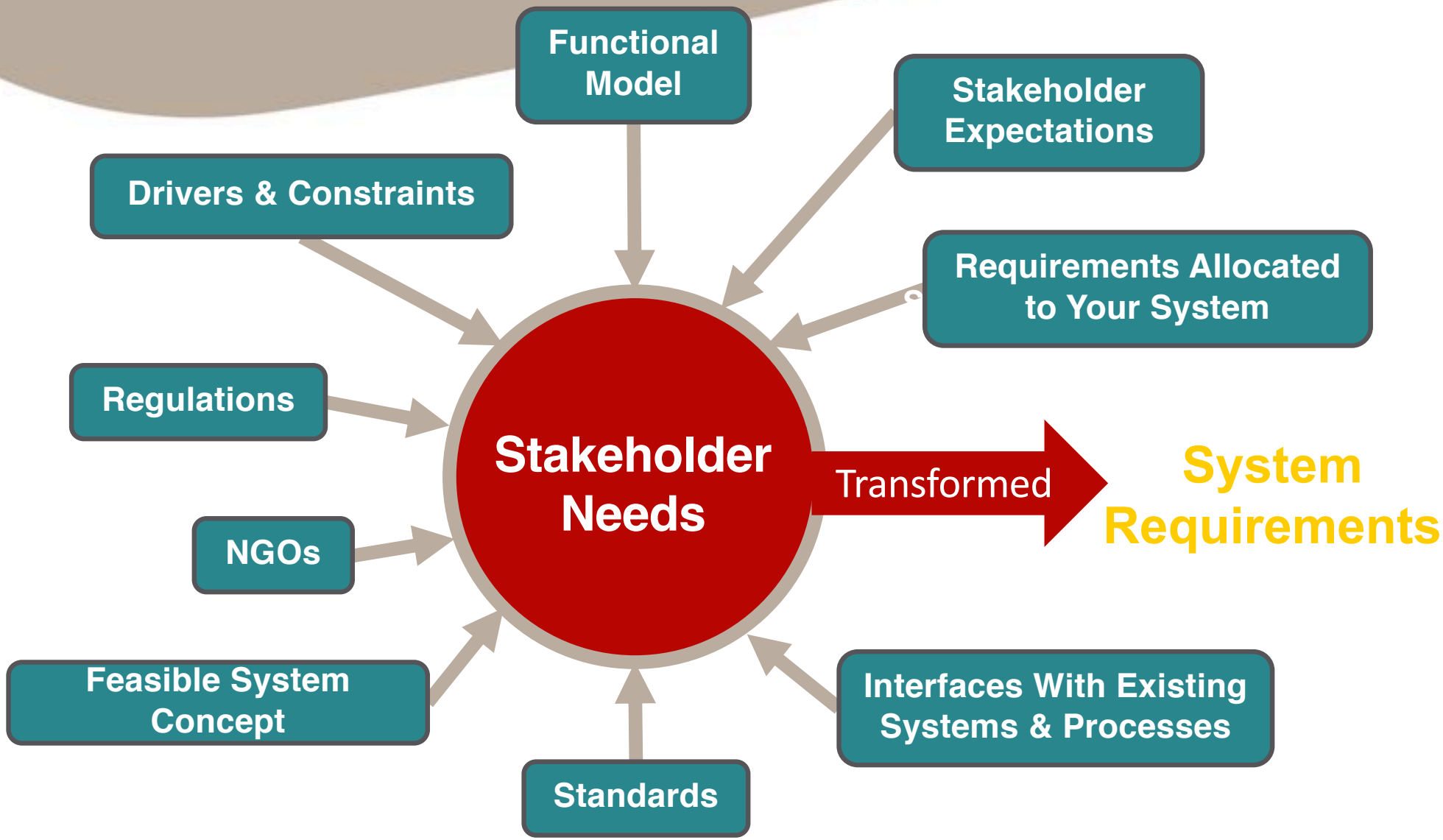
- Determine how much effort (resources and funding) has been placed into the definition and maturation of a system concept;
- Compare competing project system concepts in terms of relevance to meeting the organization's strategic goals, objectives, and ROI with acceptable risk;
- Determine which system concepts have had the same level of effort and can be compared on the same terms;
- Understand the maturity of critical technologies needed to meet the project's goals and objectives,
- Understand how much future effort will be required to mature the system concept;
- Have the information needed to determine when a proposed project's system concept is mature enough to proceed to the next system development lifecycle stage.

RequirementsExperts

Training and Services for Project Success

Transforming Stakeholder Needs into Requirements

Stakeholder Needs are the Result of the Scope Definition Activities



Definition of “needs”

Needs are the result of a formal transformation of one or more **concepts** for an entity into agreed-to expectation for that entity to perform some function(or possess some quality (within specified constraints).

- Developing a feasible concept results in an agreed-to set of needs
- It is the set of needs that must be proven to have been met (System Validation)
  - Does the delivered and verified system meet its intended purpose in its operational environment?
  - Was the right thing built?

Reference: INCOSE Guide for Writing Requirements – 2017

Characteristics of a "Set of Needs"

***Formal Transformation*.**

- If **a set of needs** results from the **formal transformation** of a **feasible concept** that implements the set of **stakeholder expectations** for an entity, the resulting set must have the following characteristics**:**
  - ***Complete*** - set of needs stands alone such that it sufficiently describes the necessary capabilities, features, functionality, performance, characteristics, constraints, and/or quality factors to meet the entity needs without needing other information
  - ***Consistent*** - set of needs contains individual need statements that are unique and do not conflict with other need statements. Uniform terminology is used for the same intent throughout the set of needs

Reference: INCOSE Guide for Writing Requirements – 2017

Characteristics of a "Set of Needs"

Agreed-to Obligation

- If the **set of needs** is to be a result of a **fair agreement** to meet an **obligation,** the set will have the following characteristics**:**
  - ***Comprehensible*** -- the set of needs must be expressed such that the reader can understand what is expected of the entity and its relation to the macro system of which it is a part
  - ***Feasible*** – the set of needs can be realized within entity constraints (e.g., cost, schedule, technical, legal, regulatory, ethical) with acceptable risk
  - ***Able to be validated*** -- It must be able to be proven, with acceptable risk, that when realized, the set of needs results in the achievement of the concept, stakeholder expectations, and agreed-to Need, goals, and objectives for the entity within defined drivers and constraints

Reference: INCOSE Guide for Writing Requirements – 2017

Transforming Stakeholder Needs into Technical Requirements

- **Stakeholder Needs are not requirements**
  - Focus is on stakeholder perspective – vs system perspective
- The SE's job is to develop technical requirements from the stakeholder needs
  - Decomposition
  - Derivation
- For each stakeholder need, ask: "What does the system have to do to implement the need?"
  - The answer is the technical requirements for the system

RequirementsExperts

Training and Services for Project Success

Well-formed Requirement Statements

Definition of an "Requirement Statement"

- A ***requirement statement*** is the result of a formal transformation of one or more **needs** into an agreed-to obligation for an entity to perform some function or possess some quality (within specified constraints).

Reference: INCOSE Guide for Writing Requirements – 2017

Characteristics of a Well-formed Requirement Statement

Formal Transformation

- For each "need" ask: what does the system have to do in order for the need to be realized? The resulting engineering analysis results in one or more requirements having the following characteristics:
  - ***Necessary*** –defines an essential capability, characteristic, constraint, and/or quality factor. If it is not included in the set of requirements, a deficiency in capability will exist, which cannot be fulfilled by other requirements.
  - ***Appropriate*** – specific intent and amount of detail is appropriate to the level of the entity to which it refers.
  - ***Singular*** –states a single capability, characteristic, constraint, or quality factor
  - ***Conforming*** – conforms to an approved standard template and style for writing requirements.
  - ***Correct*** - an accurate representation of the entity need

Reference: INCOSE Guide for Writing Requirements – 2017

Characteristics of a well-formed Requirement Statement

Agreed-to Obligation

- A requirement is not valid if it not agreed to by both the customer and provider
- If the requirement is to be a part of a fair agreement to meet an obligation, the following characteristics of a requirement can be derived:
  - ***Unambiguous*** - requirement is stated in such a way so that it can be interpreted in only one way
  - ***Complete*** - requirement sufficiently describes the necessary capability, characteristic, constraint, or quality factor to meet the entity need without needing other information to understand the requirement.
  - ***Feasible*** - can be realized within entity constraints (e.g., cost, schedule, technical, legal, regulatory, ethically) with acceptable risk
  - ***Verifiable*** - structured and worded such that its realization can be proven (verified) to the customer's satisfaction at the level the requirements exists

Reference: INCOSE Guide for Writing Requirements – 2017

Definition of a “Set of Requirements”

A **set of requirements** is a structured set of agreed-to requirement expressions for the entity and its external interfaces documented in an Entity (Enterprise/Business Unit/System/System Element/Process) Requirements Specification (Document).

- The set of requirements is what is included in a contract
- The set of requirements is legally binding
- It is the set of requirements the provider must show evidence that the delivered system meets (System Verification)
  - Was the thing built correctly?

Reference: INCOSE Guide for Writing Requirements – 2017

Characteristics of a "Set of Requirements"

***Formal Transformation*.**

- If a set of requirements results from the formal transformation of the set of needs for an entity, the resulting set will have the following characteristics:
  - ***Complete*** - requirement set stands alone such that it sufficiently describes the necessary capabilities, characteristics, constraints, and/or quality factors to meet the entity needs without needing other information
  - ***Consistent*** - set of requirements contains individual requirements that are unique and do not conflict with other requirements. Uniform terminology is used for the same intent throughout the set of requirements

Reference: INCOSE Guide for Writing Requirements – 2017

Characteristics of a "Set of Requirements"

Agreed-to Obligation

- **If the set of requirements is to be a result of a fair agreement to meet an obligation, the set will have the following characteristics:**
  - ***Comprehensible*** -- the set of requirements must be written such that the reader can understand what is expected of the entity and its relation to the macro system of which it is a part
  - ***Feasible*** – the set of requirements can be realized within entity constraints (e.g., cost, schedule, technical, legal, regulatory, ethically) with acceptable risk
  - ***Able to be validated*** -- It must be able to be proven, with acceptable risk, that when realized, the requirement set results in the achievement of the entity needs within the operational environment

Reference: INCOSE Guide for Writing Requirements – 2017

SMART requirements

- **Specific** - singular, concise, simple, clear, consistent (use of terms), unambiguous, understood one way
- **Measurable** – testable, verifiable, correct, unambiguous
- **Appropriate** – necessary, appropriate to level
- **Realistic** – feasible/achievable within constraints (cost, schedule, technology, ethics, legal, regulatory, resources, risk)
- **Traceable** – identifiable, linked, consistent (with other related requirements).

VAN Requirements

- **Verifiable** – singular, precise, concise, clear, testable, measureable, correct, unambiguous, understood one way, consistent
- **Achievable** – attainable, realizable, realistic, feasible within constraints (cost, schedule, technology, ethics, legal, regulatory, resources, risk)
- **Necessary** – needed, relevant, appropriate to level, traceable

Parting Thoughts

- **This presentation proposed an Information-based approach to Requirement Development and Management (RDM) to develop & manage requirements from the perspective that the requirements should not be developed and managed separate from other system data and information model development and management activities.**
  - Instead, requirements should be developed & managed concurrently from the beginning of the project as an integral part of the data and information modeling activities.
  - If done correctly, the Systems Engineering (SE) tools used can share data and the result is an integrated/federated data and information model of the system of interest that includes all artifacts generated during all SE life cycle phases.
  - The design modeling team will not have to import an often defective set of requirements and then analyze the requirements, correct defects, and then develop their design model.
  - The design modeling team would *work concurrently* with the RDM team such that the start of detailed design would begin with a logical system model which includes a high-quality set of requirements and a feasible concept from which those requirements were transformed rather than a set of requirements whose quality is questionable with no underlying data and information model.