What Forest?
All I See are these SE Trees!

Tutorial H05, INCOSE IS 2011

Denver, 21 June 2011

Mark Powell

Attwater Consulting

Introduction

- **Easy as an SE to get really Focused on THE Job at Hand**
  - **May do the Same SE Activity for Long Time, Years Perhaps**
  - **May work the Same Project for Years**
- **Short Tutorial Today based on Semester Class**
  - **Intended to Revisit Big Picture of SE**
  - **Intended to Reground SE's**
  - **Intended to Reinvigorate SE's**

To Help You See the SE Forest!

Some Generalities

A Definition of Systems Engineering

- **From INCOSE:**

  An interdisciplinary approach and means to enable the realization of successful systems.

- **Not Bad, *but Pretty Meaningless***

Interdisciplinary?

- **Andy Sage, INCOSE Fellow, Coined a better Term in 2006:**

  Transdisciplinary

- **SE finds the Balance Amongst and Across the Traditional Engineering Disciplines and Engineering Specialties**

The Fundamental Purpose and Justification for Systems Engineering is the Assurance of Communications despite Technical Complexity.

Some Important Systems Engineering Concepts

- ***Wholistic* - Always Looking at the Big Picture**
- **Fractal and Holographic**
  - **Iterative, Repetitive, and Reentrant**
  - **The Same Process at Any Level**
- **Top Down Development, Bottoms Up Integration and Verification**

> ***The concept that a Complex Problem can be successively broken down into simpler and simpler problems that can then be easily solved, and then successively grouping the solutions together into more and more complex units with verification until the Complex Problem is solved.***

- ***Disciplined***
- **Honest Brokerage**

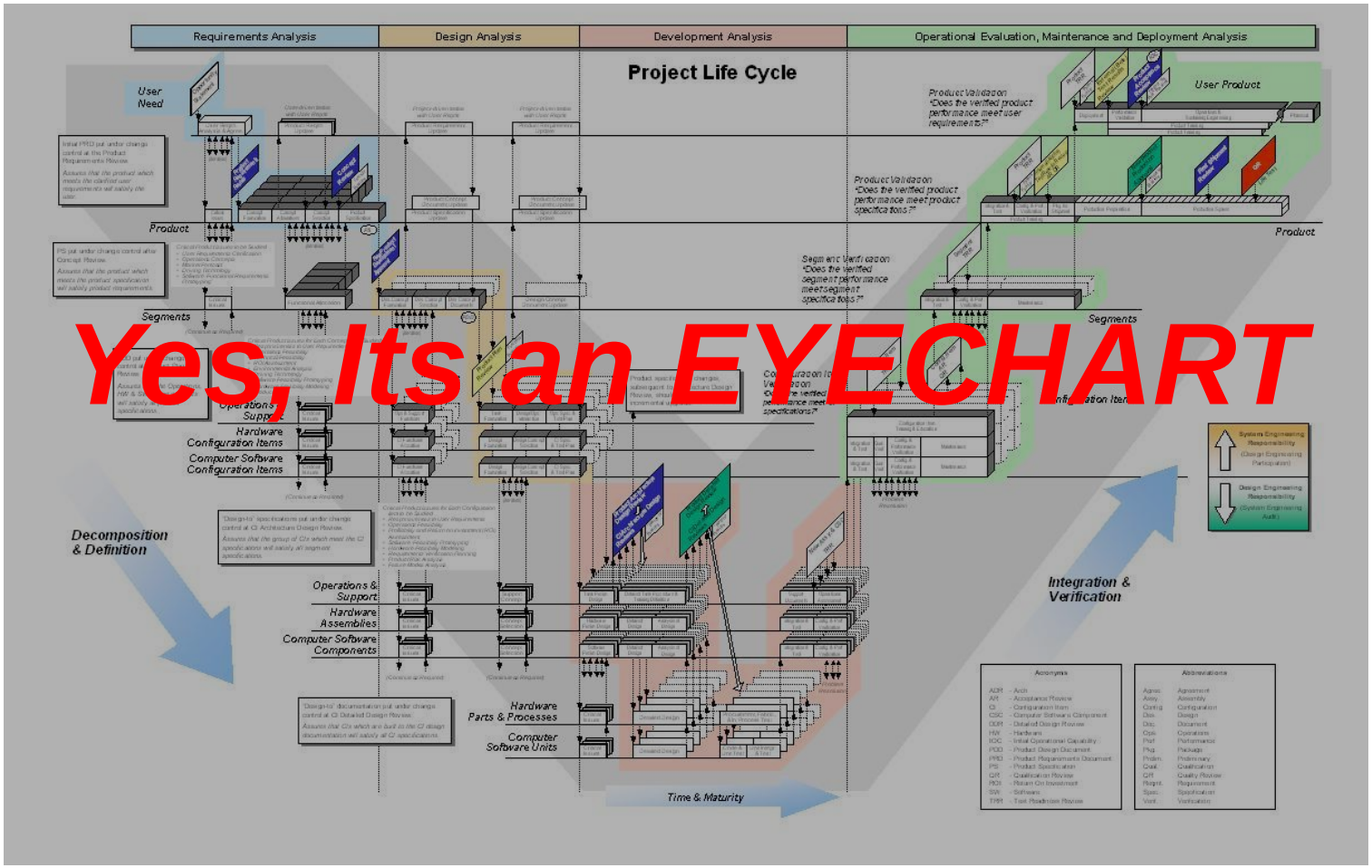
So, What is SE, Really?

- ***The Discipline* for How Technical Business is Conducted**
  - **The Infrastructure and Framework for Business**
  - **The Procedures and Processes for Business**
  - **The Logical Thought Processes for *Effective and Efficient Problem Solving***
- **Ultimately, a Collection of *Hard Lessons Learned***
  - **Based on What has Worked, and What has *NOT* Worked**
    - **Mostly, Ways to *Avoid* the Failures of the Past, from What was Done Wrong, What was Changed and Worked, and What could be Added to Make it Work**
    - **Lessons Learned being Collected Today are almost exclusively *Failures to Apply Systems Engineering***
  - **Formulated into a *Supra-Discipline* of Engineering**

To Be a Systems Engineer

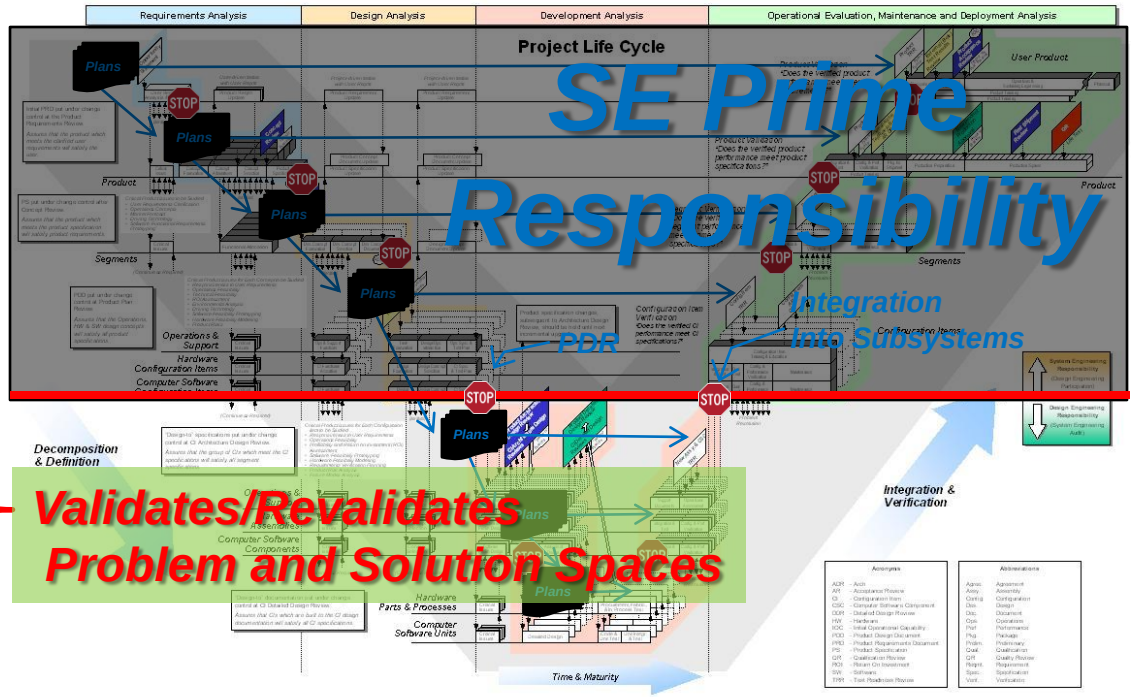
- **There are Three Types of Systems Engineers**
  - **Chief Systems Engineer (10-15+ years experience)**
    - **Top Technical Expert on the Project**
    - **Responsible for and Manages all Technical Work**
    - **Experienced in Most if not all of the Required Traditional Engineering Disciplines and Specialties**
    - **Experienced in SE and PM throughout the Lifecycle**
  - **SE Practitioner (typically 8+ years experience)**
    - **Experienced in Multiple Traditional Engineering Disciplines and Specialties**
    - **Experienced in SE in Parts of the Lifecycle**
  - **SE Clerk (typically 0-10 years experience)**
    - **Trained in SE, but not broadly Experienced**
    - **Performs Mostly Clerical and Bookkeeping Tasks**

SE in Context



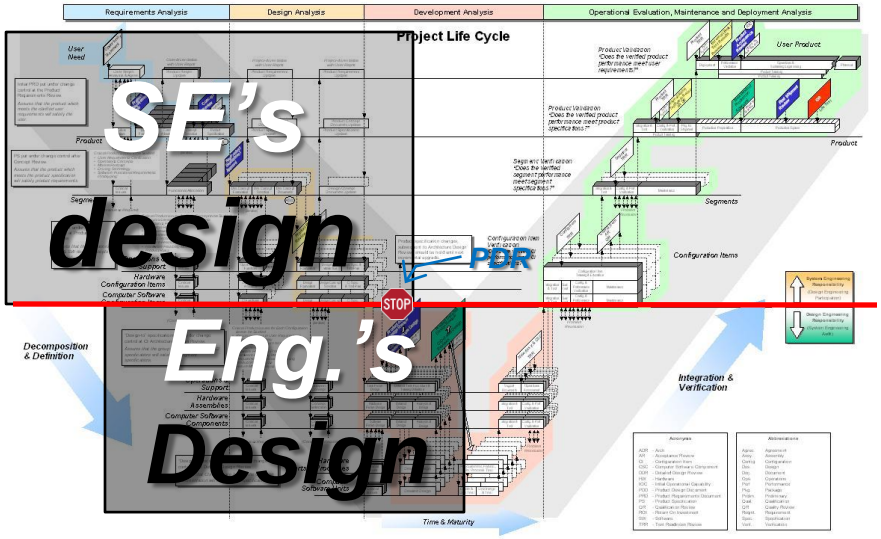
Important Aspects for SE

- **Plans**
  - **All Developed on Left Leg**
  - **Executed Down and Across**
- **Control Gates**
  - **Learning**
  - **Concurrence**
  - **Resets**
- **SE Prime Responsibility**



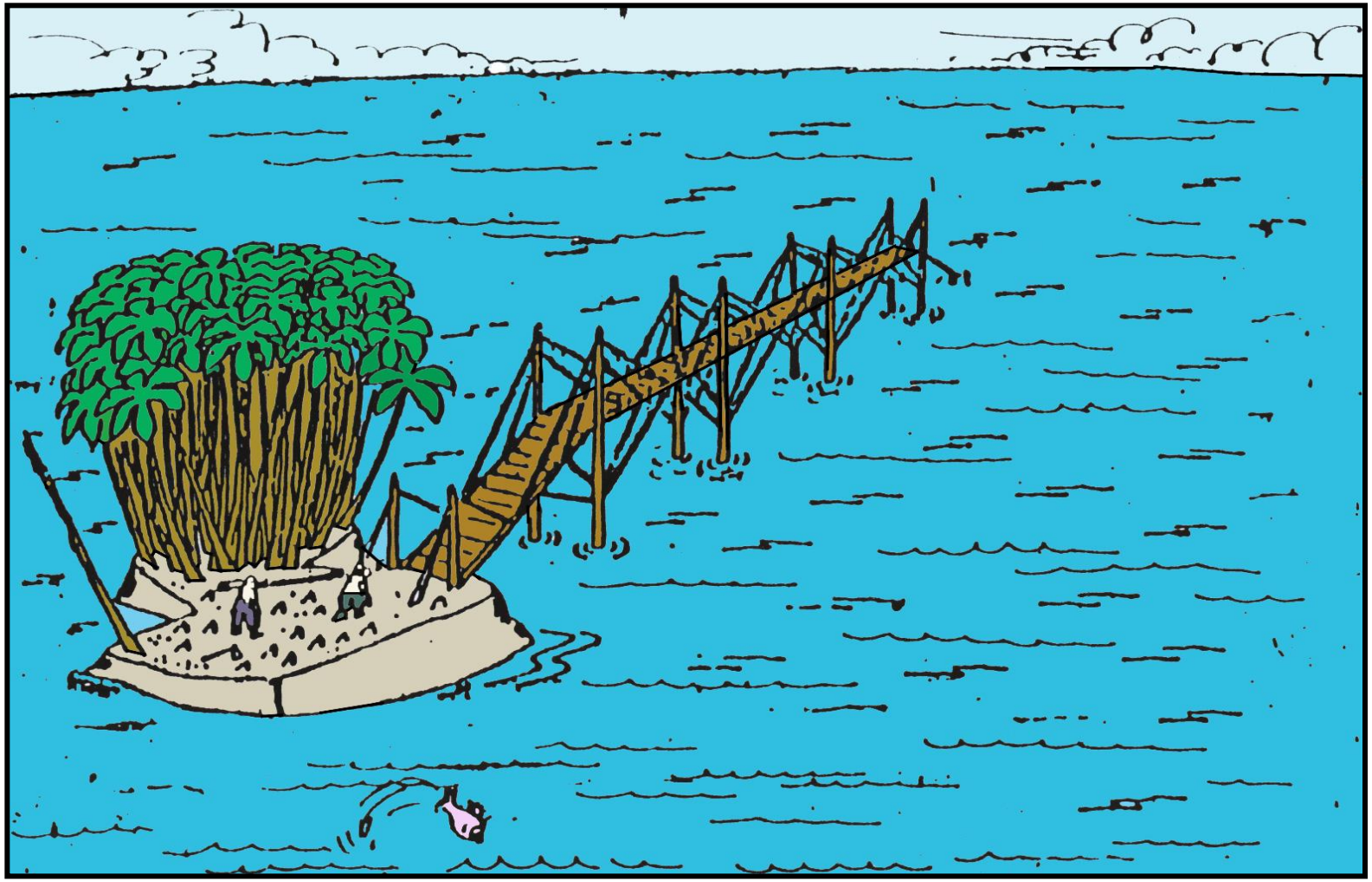
Big "D" Design vs. Little "d" design

- **Universities Train Traditional Discipline Engineers in *Design***
  - **Taught to Solve Problems *Within* The *Discipline***
  - **Taught *Point Solutions***
- **Systems Engineers Must Learn to *design***
  - **Must Solve Problems *Across* Disciplines**
  - **Optimal Sub-Solutions within Disciplines *Never* Combine (Integrate) into The Big Picture Optimal Solution**
  - **Must Always *Actively Avoid Point Solutions***



Proven Over and Over

Pitfall of the Point Solution



Customers and Stakeholders

- ***Stakeholders* – Manufacturing, Operations, Maintenance, Supportability, Users, Developing Agencies, Builders, Acquiring Agencies, Systems Engineers, Discipline Engineers, Project Management, Customers, Configuration Management, Specialty Engineering, Software Engineers, Integration and Test Engineers, Solution Providers, Victims**
- ***Anyone who will be Affected by the Problem and Its Solution* – Externally, and Internally**
- ***Fundamental* to the SE Process is Continual Consideration of All Stakeholders**

A Few Words of Caution

- ***SE's Absolutely Must use a Large Dose of Common Sense***
- **SE's *Never* have *Enough Resources* (including time and money) to:**
  - **Fully Investigate the Entire Universe of *Problem Space***
  - **Fully Investigate the Entire Universe of *Solution Space***
- **SE's Will *Never* be Such a *Good Communicators* that:**
  - **You will fully Understand All of the Customer's and other Stakeholder's Needs**
  - **You will be able to Fully Explain Everything to All the Customers and Stakeholders**
- ***Never Forget:*** **You are Part of the System that you are Trying to Optimize and Realize to solve the Problem**

Use Sound Judgment

Parallelism in design and Development

In *Top-Down* Design and Development

- **Apply SE Process to *Multiple Segments* Separately and Simultaneously with Interface Control Working Groups before Segment SRR (perhaps 5-9)**
- **Apply SE Process to *Multiple Elements* Separately and Simultaneously with Interface Control Working Groups before Segment SRR (perhaps 25-81, 5-9 per Segment)**
- **Apply SE Process to *Multiple Subsystems* Separately and Simultaneously with Interface Control Working Groups before SDR (perhaps 125-729 total, 5-9 per Element)**

Rule of Thumb:
Each Level of the Hierarchy should be Decomposed to NO MORE THAN 9 Entities in the Next Lower Level

9 Sub-entities Yield 36 Interfaces of Each Type! – Too Complex!

Parallelism in Integration and Verification

In *Bottoms-Up* Integration and Verification

- **Apply SE process to Verify *Subsystems* Separately and Simultaneously, and Integrate into Elements (hundreds of Verifications, 25-81 Integrations)**
- **Apply SE process to Verify and Integrate *Elements* into *Segments* Separately and Simultaneously (25-81 Verifications, 5-9 Integrations)**
- **Apply SE process to Verify and Integrate *Segments* into the *System* (5-9 Verifications, one Integration and Verification)**

Integration Poses the Single Largest Source of Risk in SE because of the Difficulty in Controlling Interfaces

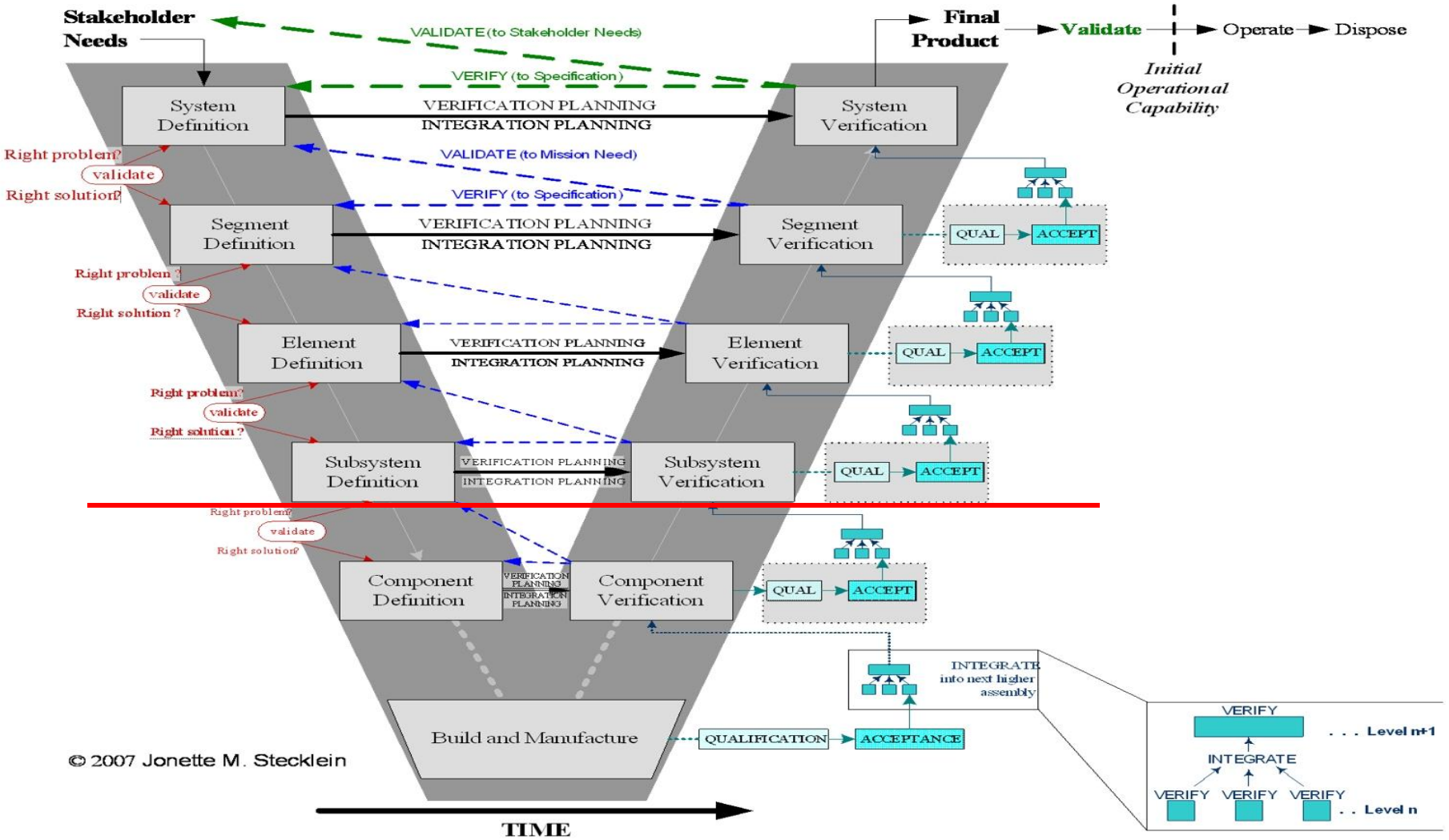
Integration, Validation & Verification

- **Integration**
  - **Interface Identification and Management**
  - **System Builds**
- **Validation Answers Two Fundamental Questions (with a *YES* of Course)**
  - ***Are we Addressing the Right Problem?***
  - ***Is our Solution Solving that Problem?***
- **Verification Assures that we Actually Built what we *designed***

Verification ensures you built it right

Validation ensures you built the right thing

A Good View of IV&V



Some Specifics

What is a Requirement?

- **A Legal *Contractually Binding* Statement**
- **Documentation of *Problem Space***
- **Documentation of *Solution Space***
- **The *Means* We Use to Communicate**

Shall Contractual Language

- ***Shall* Seems to be THE Universal Contract Word**
- **Requirements Always Part of Contract (at Minimum, by Reference)**
- ***Shall* Evokes *Enforceability***
- **Hard Lesson Learned – Contractual *Shall* Language in SE Dramatically Improves Chances of Getting What you Want**

SE's vs. Lawyers

Lawyers

- **Use Contractual *Shall* Language**
- **Use Evidence and Reasoning**
- **Not Necessarily an Honest Broker**
- **As Vague as Possible, Intentionally Open to Interpretation**
- **Thrive on Confusion**
- **Charge by the Hour**
- **Subject of Degrading Jokes**

SE's

- **Use Contractual *Shall* Language**
- **Use Evidence and Reasoning**
- **Honest Broker**
- **Clear, Concise, NOT Open to Interpretation**
- **Defeated by Confusion**
- **Paid by the Hour**
- **Not very Funny**

Requirements and Decisions

- **Where do we Find Our Requirements?**
- **We SE's Use Requirements to *Document Our Decisions***
  - **SRR – to Baseline Customer/SE Decisions as to What the Mission is**
  - **design – Our Decisions as to What the System design is are Documented as Requirements**
  - **Verification Requirements Document Decisions for Criteria for Acceptance of System by Customer**

SE Owns These Decisions!

Traceability – A Key Concept

- **We *Never* Make Up Requirements**
  - **They Always come *from Somewhere***
  - **They Always have to be *Justified* with Evidence**
- **Every Requirement Must be Traceable to a *Set* of Sources**
  - **Contracts, Memos, Meeting Minutes, RFP's, Specifications, Analyses, Requirements Documents, Plans (Program, SEMP, MIVP, CMP, Ops Plans, etc.), Interface Control Documents, Interface Requirements Documents, etc.**
  - **Justification Requires Traceability, and May Require *Rationale***
- **Every Requirement *Must* Have One or More Associated Traces to *Parents***
- ***Traceability Shall be Documented***

If you Write it Down, You Remember it!

Rationale – Another Key Concept

- **Most Requirements are Justified by Traces to *Multiple Sources***
  - **Usually to Higher Level Requirements Document(s)**
  - **Usually Also to Documented Analyses that Justify Decisions**
- **Often, *Reasoning* needed to Explain Why these Sources Justify the Requirement**
- **The Answer is *Rationale* (captures this *Reasoning*)**
  - **If it is not Completely Unmistakable to a Non-Native English Speaker that the Requirement is Justified by the Traces (or Parents), You must Document the Reasoning that Does So**
  - **Rationale is not Generally Required to be Associated with Every Single Requirement, But, *Not a Bad Idea***

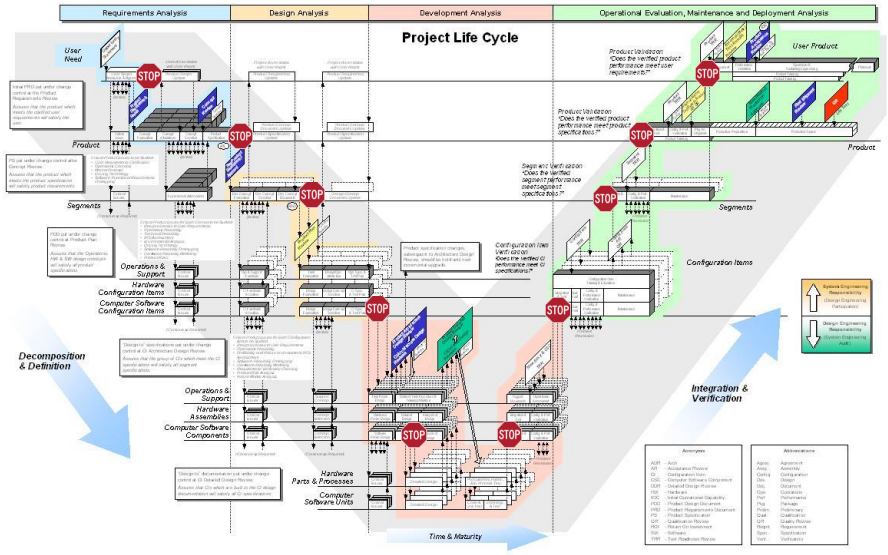
Proof Consists of Evidence and Reasoning

Requirements Have Cousins

- **Every Performance (and usually Functional also) Requirement has a lot of *Cousin Requirements***
  - **Operations, Training, Verification, Logistics, Reliability, Availability, Maintenance, Manufacturability, Operability/Human Factors, Safety, Environmental, Etc.**
  - ***Cousins* because they are *Related*, but Generally have Some Different Parents, or Parents who are Siblings**
  - ***Cousins* must be Linked to their Performance Requirements and Each Other**
- **These Cousins must be *Consistent***
  - **With the Performance Requirement**
  - **With the Rest of the Requirements and Each Other**
- **Remember the *Wholistic* Approach by SE**

Consistency and Completeness

- **Recall:**
  - **Top Down**
  - **Control Gates**
  - **SE Optimality in the Whole**
- **The Set of Requirements at Each Control Gate Must be**
  - **Consistent in the Whole – all at the *same level*, all describing the same problem and solution spaces**
  - **Complete in the Whole – all *cousins* are included**
  - **All Linked *Correctly*!**



Requirements Management

- **What a *Mess*!**
  - **Parent Requirements beget Child Requirements**
  - **Problem Space Requirements beget Solution Space Requirements**
  - **Functional and Performance Requirements have lots and lots of *Cousins***
  - **Traceability and Linkages and Rationale Must be Documented and Maintained**
  - **Baseline Sets of Requirements for Each Control Gate**
- **Management of it All**
  - **Configuration Management, Customer Level and Project Internal**
  - ***Requirements Analyses* and *Audits* Assure Consistency and Completeness**
  - **A Huge and Extremely Important *Bookkeeping* Job – *Can* be Made Easier today by DB Tools**

Control Gates and Hierarchy Levels

- **Reviewed at Each Control Gate**
  - **Set of Requirements Documents for that Hierarchy Level**
  - **ICD's**
- **Provided for Support at Each Control Gate**
  - **Analyses and Decision Documents**
  - ***Preliminary Versions* of Next Lower Level Requirements Documents and ICD's**

Control Gate	Hierarchy
System SRR	System
Segment SRRs	Segments
SDRs	Elements
PDRs	Subsystems
CDRs	Components and Parts

One of those SE Secrets:

You have to design One Hierarchy Level Deeper to Understand the Level You are Baselining

Reviews, Requirements, and design Maturity

Control Gate	Requirement Document Contents			Solution (design) Maturity
	3.1	3.2	3.7	
System SRR	Description of Mission and System	Mission and System	Segments	1%
Segment SRR	Description of System, Segments, and Elements,	Segments	Elements	5%
SDR	Description of Elements and Subsystems	Elements	Subsystems	10%
PDR	Description of Subsystems and Assemblies and Subassemblies	Subsystems	Assemblies and Subassemblies	30%
CDR	Description of Assemblies and Subassemblies, Components and Parts	Assemblies and Subassemblies, Components and Parts	N/A	90%

Some Perspective

- **Easy to See that an Element or Segment is a System**
  - **Often, Elements Have *Virtually Independent Reviews* (SRR, SDR, perhaps even PDR)**
  - **Usually Prior to Next System Review at that Level**
- **Easy to See that a Subsystem is a System**
  - **Often, Subsystems Have *Virtually Independent* Reviews**
  - **Subsystem SRR and SDR Usually after Element SRR and System SDR and before System PDR**
- **Project or Program may be Organized into Separately Funded Projects**
- **Recall Fractal Nature of Systems Engineering**
- **Definition of System depends on Perspective**
  - **New *Buzzword* – System of Systems**
  - **Nothing New, Used by FCS, Considered for CxP**

The Fundamental Concept of Verification

- **Verification is a *Risk Mitigation* Process**
  - **If we could be *Absolutely Certain* that the Contractor would *Build Exactly What we Intend* in our Specifications, there would be No Risk, No Need for Verification**
    - ***Not Always Sure* Our Requirements will Be *Understood and Interpreted* Properly**
    - ***Not Always Sure* Our Requirements are *Complete and Consistent***
  - **Verification *Reduces* this Risk, but Cannot Completely Eliminate it**
- ***How Well* Verification Requirements are Written Determines *How Well* Execution of Verification Mitigates the Risk**

Dirty Little Verification Secrets

- **Contractors *Always* Design and Build to the Verification Requirements, *Not Section 3!***

They want to get Paid!

- **Verification Requirements *Establish the Level of Risk* the Customer is to Accept for Satisfaction of a Section 3 Requirement – *Whether Stated or Not!***
- **Verification Requirements *Establish the Level of Risk* the Contractor will Face of Failing the Test When the Design Meets the Requirement, – *Whether Stated or Not!***

Some Contractors Will Overdesign and Charge the Customer to Reduce this Risk!

What are Engineering Specialties?

- **Ancillary, but Important Parts of Every Traditional Engineering Discipline**
  - **Often called the *ilities***
  - **Common to Every Traditional Engineering Discipline**
  - **Each is Similar if Not Identical in Every Discipline**
  - **Similar if Not Identical Methods Employed in Each *ility***
- **Examples: Reliability, Supportability, Quality, Compatibility, Survivability, Vulnerability, Commonality, Dependability, Accessibility, Availability, Sustainability, Maintainability, Seviceability, Safety, Verifiability, Logistics, Manufacturability, Producibility, Radcon, Controllability, Operability/Human Factors, Risk Management, Etc.**
- **Professional Societies and Certifications Exist for Most Engineering Specialties**
- ***The Source* for Most of our *Cousin* Requirements**

Why are Engineering Specialties Important?

- **Often, an Engineering Specialty Requirement may be THE *Driving Requirement* for a System**
  - **DoD: Availability of System for Defense**
  - **Commercial: Cost Effective Manufacturability and Producibility to Make Money**
  - **NASA Constellation: Safety, Risk of Loss of Crew and Risk of Loss of Mission**
- **Engineering Specialty Requirements Often form Basis for Many *Constraints***

The Throw it Over the Fence Phenomenon

- **Very Common and *Costly* Mistake**
- **Due to Failure to Fully Integrate the Engineering Specialties into a Project**
  - **Prime Responsibility of Chief Systems Engineer**
  - ***ilities* Integration Process Should be Fully Described in Systems Engineering Management Plan**
- **The Phenomenon**
  - ***ilities* Requirements (recall cousin requirements) always (or always should) Exist in a Baseline**
  - **SE's and Traditional Discipline Engineers to Derive Some *ilities* Requirements**
  - **The First Time Engineering Specialties Experts See these is at a Formal Review (SRR, SDR, PDR, CDR, etc.)**
- **The Result: Because the *ilities* Experts were not Supporting SE's from the Start, Serious Problems are Uncovered, Usually requiring *a Redesign***

The Common Theme Amongst the ilities

- **This is Critical:** ***At the Core, all ilities are Probabilistic***
  - **They are all Statements of some Required Probability for Performance, *by Definition***
  - **E. g., Reliability is defined as the probability that a product will survive (not fail) before some specified Service Life**
  - **All Establish Required *Maximum Acceptable Risks***
- **Some *ilities* Requirements may be Reduced to Best Practices (hard lessons learned) resulting from heuristics, laws, standards, etc. about How to Satisfy the Established Maximum Acceptable Risks**
- **Most SE's and Even Many Specialty Engineers are *Unaware* of this Probabilistic Theme**
  - **Practice May be reduced to Some Standard Process**
  - **Failure to Recognize Theme Makes Requirements Derivation *Extremely Difficult to Do Well***

Some ility Definitions

- ***Reliability*** **– the Probability that the item will Survive to (not fail before) a specified Service Life**
- ***Availability*** **– the Probability that the item will be in a condition and state ready to perform its intended function when called upon**
- ***Maintainability*** **– the Probability that a failed item can be repaired and returned to service within some period of time**
- ***Logistics*** **– the Probability that a part needed to repair a failed item can be provided within some period of time**
- ***Safety*** **– the Probability that no harm or injury will occur during some period of time**
- ***Quality*** **– the Probability that an item satisfies its requirements**
- ***Human Factors*** **– the Probability that any human accepted to use the item will be able to operate it as intended**
- **The Rest are all similar *Statements of Probabilities***

The Probabilistic Impact

- ***ilities* Requirements are all *Probabilistic* Requirements**
  - **Stated in Terms of a Required Probability of Performance**
  - **Establish an *Acceptable Risk* of Performance not being Achieved**
  - **Exceptions Only Apply to Established Practices (NEC, UL, OSHA, etc.) Derived from Probabilistic Requirements**
- **Verification of *ilities* Must Use Either Test or Analysis Methods**
  - ***Statistical Processing* of Actual or Simulated (Monte Carlo) Data**
  - **End up with Probabilities of Probabilities – *Drives Engineers Mad***
- **Allocation to Sub-entities can be Quite Complex**
  - **Probabilistic Allocations**
  - **Allocated Requirement May not Look Like Parent**

Many ilities are Directly Related to Each Other

- **Many Engineering Specialty Requirements are *Interdependent***
- **RAM and Logistics Example**
  - ***Driving Requirement is Availability*, the Probability that the item is in a state ready to perform its intended function when called upon (at random)**
  - **Drives Reliability Requirement: Failed Items are *not Available***
  - **Drives Maintainability Requirement: During Repair or Preventative Maintenance, Item *not Available***
  - **Drives Logistics Requirement: While Waiting for Parts for Repair, Item *not Available***
- **The Probability Statement can get Quite Complicated**
- ***Allocation* across from Availability Requirement to Reliability, Preventative Maintenance Schedules, Repair Maintenance, and Logistics for Part Supply can get Quite Complicated**

How to Write ility Requirements

- **Engineering Specialty Requirements should *Always* be Stated in terms of Some *Probability Level* that the Performance Level will be Achieved or Met**
  - ***Never* State in Terms of a Probability Model**
  - ***Never* State in Terms of Moments (Means and Variances)**
  - ***Never* State in Terms of "3 Sigma" (3*sqrt(Variance))**
  - ***Never* State with "Confidence" Levels**
    - **Intent Belongs in the Verification Requirements**
    - **"Confidence" Term means something different to a Statistician – *NOT what you want!***
- **These Rules also Apply to all Probabilistic Requirements**

Verification for ility Requirements

- **Verification Requirements for Engineering Specialty Requirements are *Always* Statistically Based**
  - **Recall, Verification Requirements establish the *Acceptable Risk* that the Requirement is not Satisfied with a Success**
  - **Must State a *Required Probability* for Performance Requirement being Satisfied**
  - **Must State *How* that Verification Required Probability is to be Obtained**
  - ***NEVER* use Confidence Intervals or the word *Confidence***
- **For *ility* Requirements, You must Verify by either**
  - **Test: Requires Data and a Statistical Inference, or**
  - **Analysis: Requires Monte Carlo Simulations (or PRA) and Statistical Inferences**

Practical Engineering Specialties Integration

- **Assure SEMP Identifies System *ilities* that are Needed for Project**
- **Assure SEMP assigns *ilities* Personnel to all Project Teams**
- **Freshen up on Probability and Statistics – Stevens Web Course SYS 601 is Good Start**
- **Assure *ilities* Requirements and Verifications Follow Rules as Presented**

What is a Risk?

- **For this Tutorial:**

  Simply An Uncertain Future Consequence

  - **Whether or Not the Consequence will be Realized is Uncertain**
  - **The Level of the Consequence that *might* be Realized is Uncertain**
- **The *Measure* of Risk:**

  The Probability that the Consequence will be Realized Above or Below a Specified Level Before the End of the Project!

What is Risk Management?

- **A Decision Process**
  - **For What to Do About Identified Future Consequences *Unanticipated* in Project Plans**
    - ***Expend Resources* to Change the Uncertainty**
    - ***Expend Resources* to Plan What to Do if the Uncertainty Changes in an Undesirable Way**
    - ***Commit the Resources* as if the Consequences have been Realized**
- **All Decisions on a Project (including RM Decisions) are Based on a *Risk Assessment***

Two Types of General Risks

- ***Anticipated* at the Start of a Project**
  - **Possible Slips/Surges in Schedule**
  - **Possible Under/Overruns in Budget**
  - **Possible Variations in Required Performance**
- ***Unanticipated* at the Start of a Project**
  - **The Problem to be Solved always Evolves**
  - **Almost Always Performance Based Risks**
  - **And *Note*: Performance Based Risks Always Produce Uncertainties in Schedule and Budget**

Handling Initially Anticipated Risks

- ***Project Management* Processes are *Risk Mitigation* Processes**
  - **Time-Proven *Lessons Learned* on What to do and What to NOT do to Make a Project or Enterprise a Success – *Reducing Risk!***
  - ***Project Management Plans* are *Risk Mitigation Plans* for Initially Anticipated Risks**
- ***Systems Engineering* Processes are *Risk Mitigation* Processes**
  - **Time-Proven *Lessons Learned* on What to do and What to NOT do to assure Required Technical Performance is Achieved**
  - ***Systems Engineering Management Plans* (and associated Plans) are *Risk Mitigation Plans* for Initially Anticipated Technical Performance Risks**
  - **Example: Verification Reduces the Risk that Performance Requirements were not Satisfied in the As-built System**

The Reason for a Risk Management Program

- **To Address those Initially *Unanticipated* Risks as they are Identified and Recognized**
- **Because Most Unanticipated Risks are Performance Related or Technical, *Systems Engineering Gets the Job of Risk Management***
- **The Risk Management Program Operates *in Parallel* with Program Management and Systems Engineering to Make the Enterprise Successful, *Despite* Unanticipated Risks**

Risk Management addresses those Uncertain Future Consequences that Nobody Anticipated at the Start of the Project

Risk Management Processes

- ***Risk Planning*** **– Establish Procedures for Conducting Risk Management on the Project**
- ***Risk Identification*** **– Discovery of Unanticipated Uncertain Future Consequences during the Project Life**
- ***Risk Analysis*** **– Establish Root Causes and Sensitivities**
- ***Risk Assessment*** **– Statistically Process Data to Determine Assurance of Risk Level**
- ***Risk Mitigation*** **– Plan and Execute a Project to Reduce or Eliminate Risk Level**
- ***Risk Tracking and Control*** **– Monitor and Measure Risk Management on the Project**
- ***Risk Communication – Explaining How Project Success is Being Assured by Managing Risk***

How to Use Risk Management in a Project

- ***All* Project Personnel and Teams should be Actively Identifying Risks as *Normal Part of Job***
- **Risk Manager and RM Team**
  - ***Review* Risks Identified by Project Personnel**
  - ***Assign* Risk Analysis Tasks to Engineering and Project Teams as Needed**
  - ***Perform* Risk Assessments as Needed (including Monitoring)**
  - ***Propose* Risk Mitigation Plans for Project Team to Execute**
  - **Track and Control All Risks**
  - **Prepare Risk Metrics and Risk Communications**
- **Project Manager and Chief Systems Engineer**
  - ***Communicate* Risk Metrics and Overall Project Risk Posture**
  - ***Decide* Upon Risk Mitigations, Assign Tasks to Project Teams**
  - ***Manage* Risk Margins, Release Resources *Only* when Project Risk Posture Diminishes with Time and Project Maturity**

Summary: Risk Management Program

- **Risk Management Program is for Risks *Unanticipated* at Start of Program (PMP and SEMP should address All Anticipated Risks)**
- **Operates *in Parallel* with Project Management and Systems Engineering**
- **PM and CSE Must Make Some *Tough Decisions* on Risk Margins, Risk Strategies, and Mitigations**
- **Risk Identification *Culture* Must be Innate**
- **Management Message: *Risk Management Saves Enterprises from Failure***

What is Technical Management?

- **Easier to List What it is *NOT***
  - ***NOT* Project Controls**
  - ***NOT* Administrative**
  - ***NOT* Project Management**
- **Very Much Like *ISO 9000***
  - **Plan and Document What and How Technical Effort is Going to be Done on the Project**
  - **Execute in Accordance with the Plan**
  - **Produce as Part of Project Records Evidence of Execution in Accordance with the Plan**
- **Performed by Chief Systems Engineer and Appointed SE Practitioners**

Technical Management Plans

- **All Projects Have *Three Primary Technical Plans***
  - **Systems Engineering Management Plan**
  - **Configuration Management Plan (subordinate to SEMP)**
  - **Risk Management Plan (subordinate to SEMP)**
- **Very Small Projects: All may be just Sections of Project Management Plan**
- **Large Projects: SEMP Usually Calls out Additional Plans (all subordinate to SEMP)**
  - **Technical Performance Measurement Plan**
  - **Technology Insertion Plan**
  - **Pre-Planned Product Improvement (P^3I) Plan**
  - **Master Test and Evaluation Plan**
  - **Interface Management Plan**
  - **Systems Integration Plan**
  - **Requirements Management Plan**
  - **Software Engineering Plan**
  - **Data Management Plan**
  - **Various Engineering Specialty Plans**

Additional Planning Products and Tools

- **Work Breakdown Structure (WBS)**
- **Systems Engineering Management Schedule (SEMS)**
- **Systems Engineering Detailed Schedule (SEDS)**
- **On Smaller Projects, These May be included in SEMP, or if No SEMP, in the PMP**

Temporal Aspects of Technical Management Plans

- **Plans Must Address *Entire Project Lifecycle***
- ***Ideally*, Plans Developed, Documented, and Under Configuration Control *at Project Start***
- ***Nominally*, Plans Delivered to Customer as Risk Mitigation *within 30 Days of Project Start***
- **Plans are *Continuously Living and Evolving* Under Configuration Control for the Life of the Project – All Get Updated at Each Control Gate**

How Can you Start Work on a Project if You Do Not Know What is to Be Done and How to Do It?

Authorship of Technical Management Plans

- **The *Person* in Control and *Responsible* for Execution of Work Should Plan the Work**
- **Based on Project Size**
  - **Very Small Projects: Project Manager is CSE, so PM is Author of All Plans**
  - **Larger Projects**
    - **CSE is Author of SEMP**
    - **CSE Appointees (SE Practitioners) are Authors of Plans for Work for which They are Responsible, in Accordance with the SEMP**

Important Heuristic: Any Plan of More than about 50 pages will Probably never be Used!

Remember What I Said about the CSE Job being Fun?

- **I had *Forgotten* that the CSE was Responsible for All these Technical Management Plans at the Start of a Project**
  - ***Organizing* and Planning Multiyear Technical Efforts for the Entire Lifecycle is *NOT Fun***
  - ***Deciding On*, Writing Down, Documenting, and Submitting to CM How You want All the Work to be Performed is *NOT Fun***
  - ***Finding* Qualified SE Practitioners Who Can Write their Plans and Then Execute is *NOT Fun***
  - **Getting this All Done and Made *Consistent* (Applying the SE Process) in less than 30 Days is *NOT Fun***
- **This is Why SE's and Specifically CSE's Get the *Big Bucks!***
- **BUT, if Technical Management Planning is Done Well, the CSE Job becomes the *Absolutely Most Fun and Rewarding* Job You will Ever Have**

More Fun for The CSE

- **CSE Must Assure That All Plans Subordinate to SEMP are Consistent**
  - **Schedule Consistency**
  - **Milestone and Control Gate Consistency**
- **SEMP and All Subordinate Plans Justify Allocation of Resources for Technical Management**

Practical Advice

- **For the CSE**
  - **Plan Until it Hurts**
  - **Execute According to the Plan**
  - **CYA Using Project Records**
- **For the SE Practitioner**
  - **Learn the SEMP, and Help the CSE with it**
  - **Follow the SEMP**
  - **Develop Good Subordinate Plans**
- **For the SE Admin or Clerk**
  - **Learn the SEMP**
  - **Follow the SEMP**

Summary

SE is not a Panacea, Just a Time Proven Set of Processes and Tools to Enable Success on Complex Technical Projects

A Set of Hard Lessons Learned

The Fundamental Purpose and Justification for Systems Engineering is the Assurance of Communications despite Technical Complexity.

Contact Information

- **Numerous Published Papers related to Execution of SE, I would be happy to Share**
- **Always looking for new, exciting, and challenging problems to solve so I can write more papers**
- **Teaching/Training Always Focuses on Practical Aspects of SE**
- **Contact Me**
  - **e-mail: attwater@aol.com**
  - **Telephone: 208-521-2941**
- ***Link with me*: http://www.linkedin.com/in/attwatermarkpowell**

Your Tutorial Instructor: Mark A. Powell

- **Over 39 years Experience in Systems Engineering and Project Management**
- **Former Chair, INCOSE Risk Management Working Group; INCOSE Technical Leadership Team, Former Assistant Director for Systems Processes**
- **Professor, Systems Engineering**
  - **Stevens Institute of Technology**
  - **University of Houston Clear Lake**
  - **University of Idaho**
- **Consultant at Attwater Consulting**
  - **Project Management and Chief Systems Engineer Coaching/Mentoring**
  - **Systems Engineering**
  - **Risk Assessments for Impossible Problems**
  - **Industry and Company Specific Education and Training**
  - **New Business Acquisition and Support**