Industry Council PTAB
- System Level Design Committee
  Requirements Capture
  Working Group
  Steve Grout
  Jean Mermet
  Ron Waxman
  April 9-10, 1997

Requirements and Requirements Capture (R&RC)

- Who is the consumer?
  - System Engineer (Large complex projects)

Type of Target Systems for R&RC

- Real Time Systems
- Imbedded systems
- Systems on a chip
- Large complex systems
- Mixed systems: A/D, MEMS, Sensors, Actuators, DSP
- Leading edge complex products
- SLDL
- Reactive systems

R&RC - What we have today

- Tools
  - SPW, Statecharts, Simulation, Synthesis, Layout, RDD-100, EXPRESS
- Languages
  - Ada, VHDL, C/C++, SDL, Esterel, Mathlab, SpecCharts, JAVA, VDL, EXPRESS
- Libraries
  - ALF, “IP” Reuse, VITAL, Component, ECIX, OMI Libraries
- Methodologies and approaches
  - IDEF, Entity relationship diagram, SADT, OOXX, RDD-100, SDRTS or RTSA, JSD, SREM, OOA, SDL, ECS, VDM, LOTOS, MCSE, ISPM

R&RC - Current Capability

- How good/bad is it?
  - Back of envelope
  - WEB document w/hypertext
  - no/few metrics to measure adequacy of requirements
  - Not integrated, loosely coupled
  - Narrow focus
  - Lack of interfaces between subset requirements
  - No guarantee all requirements were captured/accurate
  - No/little reuse of requirements
  - Requirements discovered as design proceeds
  - Many tools and methods - but no complete integrated system/methodology

R&RC - What is needed

- Process to capture all requirements
  - domain dependent/independent
  - Electronic, machine readable, processable, unambiguous capture of each aspect of the requirements
- Process to validate all requirements
- Executable requirements (models)
- A friendly interface to requirements
  - providers, brokers, domain experts
- Process to decompose/track requirements

R&RC - What is needed (continued)

- A knowledge base for captured requirements
- Process to validate the use or application of requirements
- Process to validate the use of the requirements against previous experience
• An integrated system to capture, use, apply, track, validate requirements
• Tools to assess requirements against regulations, standards, and lower level designs

R&RC - What is needed (continued)

Process to capture static, dynamic, temporal, deterministic, nondeterministic, consistent, coupled, context dependent requirements
• A means to identify business impact (e.g. cost) of captured requirements
• Templates to help capture temporal, space, cost, power, weight, etc. requirements
• A set of template formulas to use to derive and/or apply and/or validate requirements
• Automation to create domain sentences from captured requirements for domain expert validation
• Human interfaces between tools/requirements data base, for validation

R&RC - Action items to develop language structure and content
• Define all types (classes) of requirements: Requirements Dictionary (and glossary)
  – Minimum/main (RD top) categories, 80%+ import existing RD - orthogonal groupings
  – Define actions and attributes
• Define meaning of “friendly interface”
• Define alternatives for knowledge base mechanisms
• Define alternatives for validation mechanisms
• Determine constraints for language: impact of regulations and other standards
• Define business impacts expected

Develop first pass list of static, dynamic, temporal, deterministic, nondeterministic, consistent, coupled, context dependent requirements

R&RC - Action items to develop language structure and content
• Capture categories (exhaustive) from domain experts
  – Identify experts on requirements content and on requirements process
  – Electrical, electrotechnical, electronic systems domain
  – Sample some different domains
  – Extract requirements from examples and interviews
  – Review specification documents from existing international standards
  – Scan reports of DoD, EU programs

Define Requirement Types/Classes
• Composition Elements - cells, designs, design reuse, macros, cell libs, circuits, elements
• Hierarchy - definition, instance, occurrence, configuration,
• Design management - identification, versioning, alternatives, configuration, definitions, interface,
• Test - test plan, test requirements, test H/S, vectors (observe/control), test bench/stimulus
• Topology - logical, structural, syntax
• Physical - area utilization
• Cost - material, labor, design, mfg, support, redesign, schedule
• Domains - digital, analog, mechanical, acoustic, light, mixed
• Decision - (don't) care
• General - definition, internal composition, interface (semantics, syntax)
• Deterministic - non..., (un)bound,
• Environment
• Signals - information, energy, material flow, token, protocol, channel/band,
• life cycle - cost, schedule, ...
• Power - control & consumption, distribution, flow
• Process - activation (when), guard, transformation, constraints, interaction, decision, flow control,
• Architecture - Control, dataflow, memory
• Reliability, quality, quality assurance, availability, design intent
• temporal - sequence/sequential, clock, state machine, process, state (dynamic),
  coherent, (a)synch, concurrent,
• functional organization - hierarchy
• Allocation, partitioning, decomposition
• Support for simulation, analysis, regression,
• implementation approach
• architecture approach
Define Associated Requirements Actions / Attributes

- Static, dynamic, temporal, (non)deterministic, consistent, coupled, context-sensitive
- Semantic consistency
- Granularity, templates, macros, models, views, policy, category, set/collection,
  bound, intent, association, interpretation, assignment, map
- continuous, contiguous, (re)allocation, incremental, parallel, distributed, heterogeneous,
  tracking
- Attribute - property, condition, constraints, specification, requirement
  Aggregation - queue, table, index, taxonomy, composite, parts, primitive, encapsulation, scope, refinement,
  inheritance, formal, executable, orthogonality, reusability
- cost, schedule, plan, milestone, declarative, decomposition
- Expression, generality, simplicity, compilable, synthesis, verifiable, model of
  computation
- prototyping - conception, virtual, HW, SW, H/S, modeling
- decision making
- constraint generation
- Requirements capture
- View to integrated definition
- How to use models without knowing how they work
- primary, derived requirements
- User interface - GUI / graphics,
  Multilevel, multidomain
- validation - (by) construction, verification, simulation, testing, intuition, assertion
- heterogeneous - homogenous