Model Driven Engineering and Real-Time Analysis of Embedded Systems: The UML MARTE Standard and its Challenges

Tool Platforms for ES Modeling, Analysis and Validation
ARTIST Workshop at CAV 2007
June 1st, 2007
Berlin, Germany

Huascar Espinoza, François Terrier and Sébastien Gérard
CEA LIST / DTSI / SOL / L-LSP
Commissariat à l'Energie Atomique, FRANCE
Agenda

- Background & Motivation
- UML MARTE
- MARTE for Quantitative Analysis
- Frontiers & Challenges
- Summary
Need for Design-Analysis Integration

Background & Motivation

Industry View (Trade-off Problem)
- Profit
- Time-to-Market
- Dependability
- Analysis

Industry Response (Designer ≠ Analyst Problem)
- Designers do analyses
- Analysis Experience
- Time for Analyses

Research Solution
- Design-Analysis Integration
- Analyzable Design Models

Model-Driven Engineering

Domain Specific Languages

Behavior coverage
- Formal Verific.
- Simulation

Modules, Subsystems, Systems
- Design Size

Delays
- Importance

Most Designers

Analyzable Design Models
- Behavior coverage
- Formal Verific.
- Simulation

Modules, Subsystems, Systems
- Design Size

Delays
- Importance

Most Designers

Analyzable Design Models
- Behavior coverage
- Formal Verific.
- Simulation

Modules, Subsystems, Systems
- Design Size

Delays
- Importance

Most Designers

Analyzable Design Models
- Behavior coverage
- Formal Verific.
- Simulation

Modules, Subsystems, Systems
- Design Size
Collaborative Environments

Background & Motivation

- CASE Tool
- Model Checking
- Code Generation
- Requirements Management
- Analyze Requirements
- Design
- Validate & Verify
- Build Code
- Reliability Analysis
- Schedulability Analysis
- WCET Analysis
- Performance Simulation

Common Repository
Integration of V&V Information
Model Driven Engineering & RTES

Integrate Models to...

DESIGN-BUILD-VALIDATE-VERIFY

- Evaluate architectural design alternatives
  - MoCCs, fault-tolerance configurations…
  - RTOS, SoC architectures, communication buses…

- Multiple points of view with non-functional requirements
  - memory usage, power consumption, real-time response…

- Best software engineering practices
  - refinement/abstraction, multi-perspective, layering…
  - metamodelling (metamodel = model repository = data structure)
    - Technology independent (mapping to XML, Java)
Domain Specific Modeling Languages

- **Use of domain concepts as language constructs**
  - E.g. concepts of “memory”, “task”, “bus”, instead of general ones “components”, “nodes”, “classes”...

- **Criteria for success of a DSML**
  - Technical validity and absence of ambiguities
  - Domain appropriateness: syntax close to RTES ontology
  - Interoperability with other technologies (verification, code generation)
  - Support: tools, documentation, capacity for evolution

But mainly… balance between *expressive power and simplicity*

- **Main approaches to defining a DSML**
  1. New language from scratch
  2. Extend an existing language (add new language constructs)
  3. Refining an existing language (specialize language constructs)
Profiling UML for a Domain

**Advantages of UML Profiles:**
- Reuse of language infrastructure (tools, specifications)
- Requires less language design skills
- Allows for new (graphical) notation of extending stereotypes
- A profile can define model viewpoints
  - E.g. UML activity diagram extended to specify multitask behavior

**Disadvantages:**
- Syntactical constructs and semantics constrained by UML metamodel

**How to face these limitations?**
- Profile semantics: on-going work: ‘Executable UML Foundation’
- Create rich profiles (expressiveness) and complement with specific methodologies (precise semantics)
Agenda

- Background & Motivation
- **UML MARTE**
- **MARTE for Quantitative Analysis**
- Frontiers & Challenges
- Summary
SPT was the first OMG’s UML profile for Real-Time Systems:

- Support for Schedulability analysis with RMA-type techniques
- Support for Performance analysis with Queuing theory and Petri nets
- A rich model for “metric” Time and time mechanisms

Despite its extensive use, several improvements were required:

- Modeling HW and SW platforms, logical time, MoCCs, CBSE…
- Alignment to UML2, QoS&FT, MDA,…
- SPT’s constructs were considered too abstract and hard to apply

A Request For Proposals for a new profile was issued
UML Profiles for RTES

**Industrial Specs. (OMG)**
- QoS & FT (2005-adopted)
- SysML (2006-adopted)

**Industrial Specs.**
- AADL (2006-draft)
- Autosar (2006-adopted)

**Academic Specs.**
- Accord (2000)
- Omega (2004)
- MAST (2002)
- Others

---

**Partners**

**Industrials:**
- Alcatel*
- Lockheed Martin
- Thales*

**Tool vendors:**
- ARTISAN Software Tools*
- IBM*
- Mentor Graphics Corporation*
- Softeam*
- Telelogic AB* (I-Logix)
- Tri-Pacific Software
- NoMagic
- MathWorks

**Academics:**
- Carleton University
- CEA
- ESEO
- ENSIETA
- INRIA
- INSA from Lyon
- SEI (CMU)
- Universidad de Cantabria

---

Enhanced SPT + MDA + CBSE + HW&SW

- **RFP** (Feb 2005)
- **Initial Subm.** (Dec. 2005)
- **Final Subm.** (June 2007)
MARTE High Level Architecture

- Property Qualifiers
- Extended Data Types
- Expressions language
- Causal/temporal
- Clocked/synch.
- Physical/real-time (multiple clocks, time constraints, observation…)
- Models of Computations

Common concepts for quantitative analysis
- Scheduling properties
- Predictability and resource usage
- Worst case timing metrics and constraints

RTEA (Real-Time & Embedded Analysis)
- Generic Quantitative Analysis
  - Schedulability
  - Performance
- RTEM (Real-Time and Embedded Design)
  - MoCC
  - HW Resources
  - SW Resources
  - Refined Platform Modeling Constructs (RTOS, FPGA, DSP, ECU, Power Supply…)

UML MARTE
Modeling Non-Functional Properties

Non-functional properties describe the “fitness” of systems behavior (e.g., performance, memory usage, power consumption…)

- **NPF’s nature:**
  - Quantitative: magnitude + unit (e.g., energy, data size, duration)
  - Qualitative (e.g., “periodic” or “sporadic” event arrival patterns)

- **NFP values are qualified by:**
  - source, statistical measure, precision,…

- **NFPs are parametric and derivable:**
  - variables: placeholders for unknown values
  - expressions: math and time expressions

- **NFP annotation mechanism is flexible:**
  - predefined NFPs (e.g., end-to-end latency, processor utilization)
  - user-specific NFPs (but still unambiguously interpreted)
**NFP Annotations in a Nutshell**

1) **NFP Type Declaration**
   - Define measurement units and conversion parameters
   - Define NFP types with qualifiers

2) **NFP Declaration**
   - Define NFPs in stereotype attributes using NFP Types
   - Modelers can also declare NFPs in UML Classifiers

3) **NFP Value Specification**
   - Three mechanisms: tagged values, constraints, and slots.
   - Values and expressions use VSL (Value Specification Language)
Value Specification Language (VSL)

- **UML lacks of value specification mechanisms for NFPs**
  - No syntax for structured values
  - No syntax for algebraic and time expressions
  - Data type system is not sufficient for non-functional modeling

- **Foundations of VSL...**
  - Reuse OCL constructs: grammar for values and expressions
  - Generic data type system: (based on ISO’s General-Purpose Datatypes)
  - Defined by abstract and concrete syntaxes (grammar).
  - Flexibility: expression operators in model-specific libraries
  - VSL extends UML Simple Time model (occurrence index, jitters,...)
Time Expressions with VSL

Duration expression between two successive occurrences

Constraint in an observation with condition expression

Specification example in Sequence diagrams...

Extended duration intervals with bound « [ ] » specification

Jitter constraint

Instant Interval Constraint
Software Resource Modeling Profile

- To describe API of software execution platforms
  - Constructs for Multitasking models
  - Real Time Operating Systems (RTOS)
    - standard RTOS API (e.g. POSIX, OSEK/VDX and ARINC 653)
  - Language libraries (e.g. ADA)

- Concurrent execution mechanisms
  - Task, Interrupt, Alarm & Counter...

- Synchronization mechanisms
  - Events, Mutual Exclusion Access Mechanisms...
Hardware Resource Modeling Profile

- To describe structure of Hardware platforms
  - Different abstraction levels for: processor simulation, power consumption calculation, WCET analysis, block diagrams
  - Two sub-views: logical (functionality) and physical (layouts)

- HW Logical constructs
  - RAMs, ASICs, I/O Devices, Clocks, Buses...

- HW Physical constructs
  - Cards, Chips, Power Supplies,...
Agenda

- Background & Motivation
- UML MARTE
- MARTE for Quantitative Analysis
- Frontiers & Challenges
- Summary
MARTE Features for Quantitative Analysis

Quantitative analysis calculate mathematically NFPs of interest based on other available NFPs and the system behavior

- **Improvements w.r.t. SPT**
  - Extend implementation and scheduling models
    - e.g. distributed systems, hierarchical scheduling
  - Extend the set of analysis techniques supported
    - e.g. offset-based techniques

- **New features w.r.t. SPT**
  - Support for sensitivity analysis
  - Improve modeling reuse and component-based design.
  - Support of the MDA approach
UML-Based Analysis Foundations

- **GQAM Profile factorizes common constructs and NFPs**
  - Stereotypes define “analysis” abstractions
    - workload events, scenarios,…
    - schedulable entities, shared resources, processing nodes, schedulers…
  - Stereotype attributes define pre-defined NFPs
    - e.g. event arrival patterns, end-to-end deadlines, wcet-bcet-acet,…

- **The analysis sub-profiles define model well-formedness rules**
  - It includes “constraints” to construct “analyzable” models, w.r.t…
  - ”Analysis Model Viewpoints” (e.g., schedulability analysis viewpoint)
  - Specialized constraints must be refined by technique-specific approaches

The MARTE analysis sub-profiles provide standard constructs to map UML models on well-established analysis techniques

- MARTE “Foundations” and “GQAM” allow for extending to further techniques
Analysis Modeling Structure

Analysis Context

Workload Behavior

Resources Platform

evaluate situation

evaluate capacity

uses

resource allocation

scenarios

load

exec.host

protected resources

comm.host

broker
Modeling Workloads for Scheduling Analysis

- **Workload Behavior**
  - « gaWorkloadBehavior » NormalMode
  - « workloadEvent » ControlTrigg
    - periodic (period= (5, ms))
  - « workloadEvent » ReportTrigg
    - periodic (period= (100, $pR, ms))
  - « workloadEvent » ReportTrigg
    - periodic (period= (1, s))
  - « gaScenario »
    - respTime= ($r1, ms), utilization= $u1, execTime= ($e1, ms)
      - Control
    - respTime= ($r2, ms), utilization= $u2, execTime= ($wcet1, max, ms)
      - Report
    - respTime= ($r3, ms), utilization= $u3, execTime= ($e3, ms)
      - Command

- **EndToEndFlows**
  - (end2end deadlines and predicted times)

- **Workload Events**
  - (arrival patterns)

- **Scenario**
  - (response times, hosts utilization...)

**Note:** The diagram represents a model for quantifying and analyzing workloads and scheduling scenarios, including events and their periodicities, along with the corresponding response times and utilization metrics.
Modeling Scenarios for Scheduling Analysis

Step (exec. time)

Shared Resource

Binding to Thread

"gaScenario" Report

ContrClock

"allocated" Reporter
allocatedTo:
Reported

"allocated" ServosData

"allocated" ControllerComm
allocatedTo:
MsgStatus

"allocated" StationComm
allocatedTo:
MsgStatus

"allocated" DisplayRefresh
allocatedTo:
DisplayRefreshTask

"SaSharedResource"
DisplayData

"allocated" task

"allocated" task

"allocated" task

Binding to Thread

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task

"allocated" task
MARTE for Quantitative Analysis

Modeling Analysis Contexts

Instance of a WorkloadBehavior model

Context under Analysis

Context-specific variables

Simple Schedulability Analysis context

Sensitivity Analysis context

TeleoperatedRobotSAM

\[\text{var} \{\text{dir}= \text{inout}\} \text{isSched\_System}: \text{NFP\_Boolean}= \text{isSchSys}\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{wcet\_Report}: \text{NFP\_Duration}= \text{wcet1}\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{procRate\_CAN}: \text{NFP\_Real}= \text{prCAN}\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{period\_Report}: \text{NFP\_Duration}= \text{pR}\]

\[\text{var} \{\text{dir}= \text{inout}\} \text{isSched}= (\text{true}\), \text{isSchSys}\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{wcet}\_Report= (5, \text{ms}, \text{determ})\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{procRate}\_CAN= (1, \text{determ})\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{period}\_Report= (30, \text{ms}, \text{determ})\]

\[\text{var} \{\text{dir}= \text{inout}\} \text{isSched}\_System= (\text{true}, \text{req})\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{wcet}\_Report= (50, \text{ms}, \text{max}, \text{calc})\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{procRate}\_CAN= (0.2, \text{ms}, \text{min}, \text{calc})\]
\[\text{var} \{\text{dir}= \text{inout}\} \text{period}\_Report= (10, \text{ms}, \text{min}, \text{calc})\]
Schedulability Analysis Modeling Profile

- Classic RMA
- Extended RMA
- Holistic Approach

- Timed Automata with Tasks
  - AEIOLTS

- Compositional Analysis

- Active Object Semantic
  - Event Priorities vs. Thread Priorities

- PIM, PSM, PDM

RMA-Style

Timed Automata

Modular Analysis

SAM (MARTE)

MDA

Object Oriented
Constraining SAM for Specific Techniques

- Conceptual Constructs
- Stereotypes and tags definition
- OCL Constraints for Specific Analysis Techniques

- « specification » SAM
- « domainModel » SAM_Domain
- « profile » SAM_UML
- « constraintsSet » Scheduling Analysis

- « import »
- « usageGuide » RMA-Based Techniques
- « constraintsSet » ClassicRMA
- « constraintsSet » Holistic
- « constraintsSet » Compositional

- « import »
- « usageGuide » State-Based Techniques
- « constraintsSet » Timed Automata with Tasks

- « import »
- « usageGuide » State-Based Techniques
- « constraintsSet » Holistic
MARTE Tooling

**Current Implementations supporting MARTE**
1. Full MARTE Profile & Libraries for Eclipse UML2
2. VSL edition assistant and type checker as a Eclipse plug-in for the UML Papyrus tool
3. Generator of OIL (OSEK Implementation Language) files for OSEK from UML Models annotated with the SRM profile

**On-going work:**
1. Eclipse plug-ins to transform UML models annotated with the SAM profile to input files of MAST and SymTA/S tools
2. Eclipse plug-ins to transform UML models annotated with the HRM profile to processor simulators data files
3. …

Open Source Implementation in UML Papyrus: www.papyrusuml.org
Agenda

- Background & Motivation
- UML MARTE
- MARTE for System Analysis
- Frontiers & Challenges
- Summary
MARTE Frontiers

- **MARTE define the language constructs only!**
  - Common patterns, base building blocks, standard NFP annotations
  - Generic constraints that do not force specific execution models, analysis techniques or implementation technologies

- **It does not cover methodologies aspects:**
  - Interface-Based Design, Design Space Exploration
  - Means to manage refinement of NFP measurement models
  - Concrete processes to storage, bind, and display NFP context models
  - Mapping to transform MoCCs into analysis models

MARTE is to the RTES domain as UML to the System & Software domain: a family of large and open specification formalisms!
Revisiting the A&D Integration Problem:
Trends: Dedicated & Monolithic Approaches

Design Artifacts

Applications

Platforms

Different Modeling Formalisms

Design-Oriented Modeling
(code generation, ...)

Analysis-Oriented Modeling
(formal specifications)

Semantic Mismatch

Data-flow, Components, Classes, State-Charts, ...

Petri Nets, Timed Automata, RMA-based, ...

Different Analysis Tools

Performance Simulation

Schedulability Analysis

Reliability Analysis
Revisiting the A&D Integration Problem:
Challenge: Standardized & Compositional Approach

Design Artifacts

Standard Modeling Formalisms

Different Analysis Tools

Applications

Platforms

Analyzable Design Models

Context-Based Analysis Models
(composition of analysis building blocks)

Extended UML Composite Structures

UML + non-functional annotations

Performance Simulation

Schedulability Analysis

Reliability Analysis
How Does MARTE Support it?

- Analyzable Design Model
  - Design-Analysis model links
  - « workload » EnvironmentView
    - Estimates
    - Configuration Data
    - Requirements (e.g., time)
  - « resources » ResourcesView
    - Measurements
    - Simulation Data
  - « behavior » BehaviorView
    - Simulation Data

- Context-Based Analysis Model
  - « AnalysisContext » AnalysisBuildingBlock
    - « Parameters »
      - p1...
    - « Constraints »
      - c1...
  - « execEngine »: CPU1
    - « scheduler »: EDF
  - « scenario »: scenario1
  - « scenario »: scenario2

- High-level Interaction view Annotated with NFPs
- Hardware & Software Platform View annotated With NFPs
- Scenarios or State models Annotated with NFPs
- UML MARTE Backbone
  - Technique-Specific Analysis Constructs (e.g., SAM Profile)
  - High-Level Analysis Constructs (GQAM UML Profile)
  - Non-Functional Properties Constructs and Expression Language (NFP & VSL UML extensions)
Needs for the Compositional Approach

- **Mechanisms to attach analysis annotations in design models**
  - Consistent analysis views over design views
    - e.g. fragment of activity model annotated with scenario NFPs to extract analysis model
  - Analysis annotations that not disturb design model
    - e.g. observer and observation mechanism

- **Mechanisms to manage multiple “NFP value views”**:
  - NFP already allow to define different versions of NFP!
    - Use NFP qualifiers to indicate the context of validity of NFP values (refinement, platform binding, analysis context)
  - NFP already support VSL variables!
    - Use VSL variables to define parametric NFP values
  - Provide support to manage transformation and consistency checking between different NFP value refinement levels

- **Different analysis techniques in (composite) analysis contexts**
  - To define adaptation interfaces to transform heterogeneous information
Integration of Multiple Analysis Information

- **Analyzable Models**
  - Design models annotated with Analysis information (usage, metrics, overheads)
  - Behavior and Platform models
  - Tools used for estimating Analysis specific inform.
  - Classifier -Instance relationship supported by “Parts” in Composite Struct. Diagrams
  - Connectors between analysis building blocks
    - (adaptation functions, objective functions,...)
  - Analysis Context Models based on Analysis Building Blocks
  - Sched. Analysis Knowledge
  - Perf. Analysis Knowledge
  - WCET Analysis Knowledge

- **Connections**
  - Processing entities model
  - Communication entities model
  - Analysis Techniques specific Models (declarative or detailed)

**Tools used for estimating Analysis specific inform.**

**Analysis Context Models**

**Reliability Analysis**

**Schedulability Analysis**

**WCET Analysis**

**Performance Simulation**
Analysis-Based Design Decision

To cohesively link the design and analysis processes in a well-structured, traceable and constrained decision-making process.

Conceptual Formulation of Design Problem

Phase 1: Formulate Design Space

Phase 2: Specify Design Parameters

Phase 3: Specify the Systems Goals and Design Responses

Phase 4: Specify Design and Behavior Constraints

Phase 5: Determine Objective Functions

Phase 6: Determine Quantities Associated with Analysis Model

Phase 7: Determine the Analysis Relationships

Phase 8: Establish Validity Space through Meta-level Restrictions

Phase 9: Encapsulate and Execute Analysis

Phase 10: Integrate Analysis Results into Design Decision

Analysis Support for Decision Making

Phase 6: Establish Validity Space through Meta-level Restrictions

Phase 9: Encapsulate and Execute Analysis

Phase 10: Integrate Analysis Results into Design Decision
Summary

- **Industrial Use of V&V can benefits from MDE**
  - Analysis task must be cohesively integrated with Design tasks
  - Application of individual analysis techniques should be regarded as an essential part of an integrated V&V methodology

- **MARTE: a rich MDE-oriented specification language**
  - Common patterns, base building blocks, standard NFP annotations
  - Generic constraints that doesn’t force specific execution models, analysis techniques or implementation technologies

- **Methodological support is still under way:**
  - Complex analysis scenarios for Interface-Based Design, Multiobjective Design Space Exploration...
  - Means to manage NFP measurement models
  - Methods to map/transform MoCCs into analysis models