ARTEMISIA ASSOCIATION

The association for R&D actors in the field of ARTEMIS

INDustrial EXploitation of the indexys genesYS cross-domain architecture

ARTEMIS

Project n° 100021

Semantics-Based Integration of Embedded Systems Models

András Balogh, OptixWare Research & Development Ltd.

Advanced Research & Technology for EMbedded Intelligence and Systems

Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- Semantics-based integration
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- Conclusions

Outline

- **Embedded systems overview**
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- Semantics-based integration
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- Conclusions

Embedded systems

- Typically
  - Networked
  - Heterogeneous
  - Real-time
- Diverse application domains
  - Industrial
  - Transportation
  - Healthcare
  - Entertainment/mobile
- Common trends
  - Increasing complexity
    - Functionality
    - Connectivity
  - Decreasing time-to-market requirements

Development methodologies for ES

- Main trend
  - Adoption of model-based development techniques
    - High-level design
    - (semi-) automated synthesis
    - High-level simulation and analysis
  - Status depends on the domain
- MDD promises
  - Reduced development cycle length
  - Improving performance
  - Better reusability
  - Better response to changing requirements
    - Functionality
    - Implementation platform
    - Communication/integration protocols

Current Industrial State-of-the Art

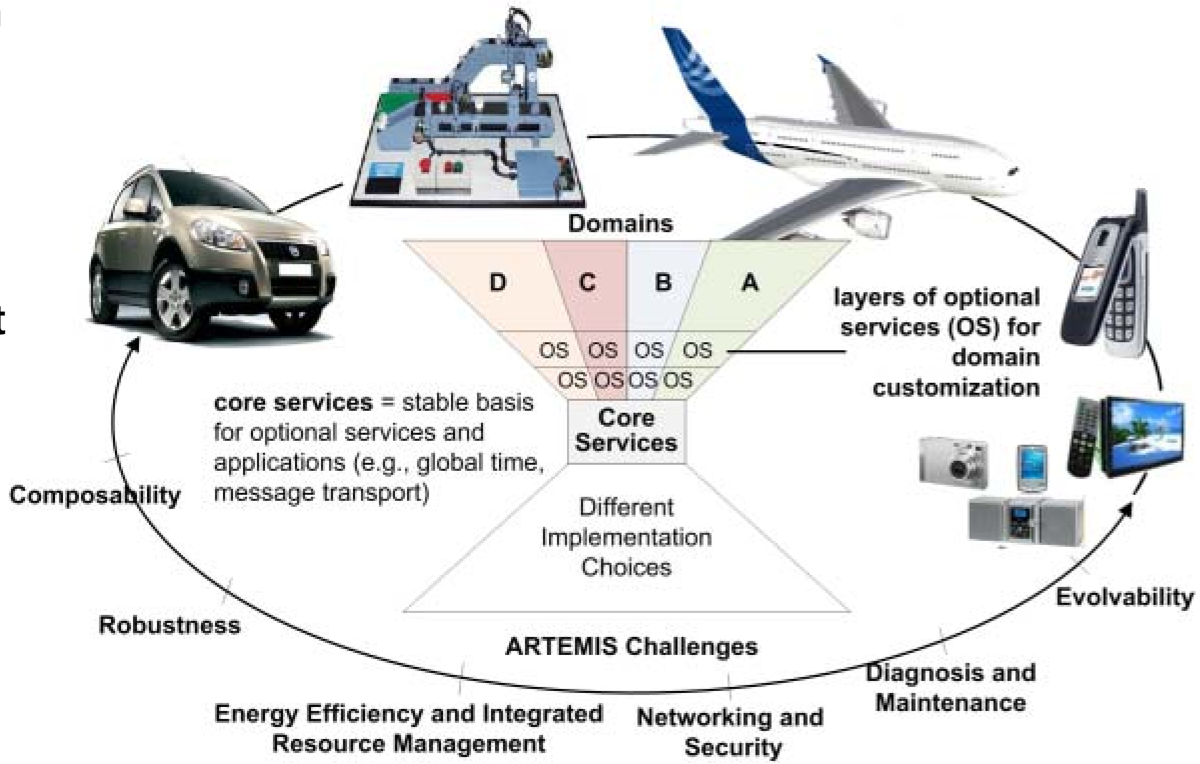
- Modeling
  - Using UML and its derivatives
    - Only for modeling
    - Limited analysis/synthesis possibilities
  - Using low-level descriptions (like FIBEX)
    - For synthesis
    - Often cannot be derived from high level models
  - All-in-one solutions
    - Example: AutoSAR (+EAST-ADL2)
    - From high level to configuration modeling
    - Lack several features (like Behavior modeling)
- Main problems
  - Diverse languages
    - Within an application domain
    - Between domains
    - Hinders reuse and knowledge transfer
  - Language integration is unsolved
  - Large abstraction gap between modeling and implementation

Outline

- Embedded systems overview
- **Overview of the GENESYS-INDEXYS approach**
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- Semantics-based integration
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- Conclusions

GENESYS
Generic Embedded Systems Platform

- EU Framework 7 Project 2008-2009
  - Aims at the establishment of an embedded systems platform and development method that meets the requirements of ARTEMIS
- Main goals
  - Definition of a **cross-domain architectural style** for embedded system design
  - Provide a **reference architecture template** for platform services
  - Establish a **model- and quality-driven development** methodology
  - **Prototypical evaluation** of results



INDEXYS - Industrial Exploitation of the Genesys Cross-domain Architecture

- EU Artemis Project 04/2009-09/2011
- Partners: Audi, EADS-IW, NXP, OptXware, Thales RSS, TTTech, TU Delft, TU Darmstadt, TU Kaiserslautern, TU Vienna
- Main objective
  - Realize industrial implementations of the GENESYS architectural concepts in different domains
    - Automotive
    - Aerospace
    - Railway
  - Implement a cross-domain tool chain implementing the GENESYS development approach
    - In an open, modular, extensible way
    - Integrating and extending legacy methods and tools

The GENESYS Development Approach

- Model-driven
  - Primary design artifacts: high-level models
  - Strong emphasis on end-to-end modeling
    - From requirements
    - To deployment-ready state
- Quality driven
  - Integrated analysis tools and methods
  - Continuous quality assurance via Verification/validation
- Relies on standards
  - Uses standard modeling languages (primary: UML MARTE)
  - Uses open, extensible platform for the design environment

The GENESYS Development Approach

- Open Questions
  - The GENESYS approach is strongly component-oriented
    - Interfaces play crucial role
      - » Only the syntax can be defined
      - » Temporal, behavioral, etc. description is not supported
  - Legacy integration is a key issue
    - No implementation of legacy model import/export
      - » Semantics-based integration?
  - Lack of precise modeling notation
    - UML MARTE
      - » High level
      - » Inconsistent
      - » Too permissive (to serve as basis for implementation)
    - Review note
      - » A GENESYS-specific DSL should be defined

INDEXYS Methodology and Tools

- INDEXYS relies on the main ideas of GENESYS, but
  - It should be integrated with industrial languages and tools
    - Integration of heterogeneous modeling notations
  - It should scale up to realistic project sizes
    - In-memory model management is insufficient
    - Team collaboration support required
    - Access control
    - Version control
  - It should provide end-to-end traceability (certification support)
    - Models should be handled in an integrated framework
  - It should support legacy communication and execution platforms
    - Implementing GENESYS applications on legacy platforms

INDEXYS Methodology and Tools

- INDEXYS relies on the main ideas of GENESYS, but
  - It should be integrated with industrial languages and tools
    - **Integration of heterogeneous modeling notations**
  - It should scale up to realistic project sizes
    - In-memory model management is insufficient
    - Team collaboration support required
    - Access control
    - Version control
  - It should provide end-to-end traceability (certification support)
    - Models should be handled in an integrated framework
  - It should support legacy communication and execution platforms
    - **Implementing GENESYS applications on legacy platforms**

Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- **Current modeling languages for embedded systems**
- Integration concept for modeling notations
- Semantics-based integration
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- Conclusions

Current Modeling Notations (selection)

- Generic standard notations
  - Used in several different embedded system application domains
  - SysML
  - UML MARTE
- Domain specific languages
  - Automotive
    - AutoSAR
    - EAST-ADL2
    - FIBEX
  - Aerospace
    - AADL
  - Safety critical distributed
    - DECOS languages
- Several other languages present

Elements of modeling

- Key modeling stages
  - Architectural modeling
    - Requirements
    - Software components and interfaces
    - Hardware resource modeling
    - System topology modeling
    - System allocation and scheduling
    - Detailed hw/sw configuration
  - Behavioral modeling
    - Behavior of the system
      - User viewpoint
      - High level scenario description
    - Behavior of a single component
      - High-level
      - Detailed (implementation-ready)
- Modeling aspects
  - Functional
  - QoS (timeliness, dependability, power consumption, etc.)

Analysis of the selected languages

- Varying abstraction level
  - From high-level to implementation-ready modeling
  - *All levels are needed in a typical design process*
    - Interconnection of levels?
    - Consistency of levels?
- Modeling aspects only partially covered
  - Non-functional (QoS) aspects typically missing
  - Most important ones
    - Timeliness (as we are dealing with real-time systems)
    - Dependability (for critical systems)
    - Resource consumption (memory, CPU, power)
- Modeling stages
  - No single language is able to describe all areas from requirements to deployment

Analysis of the selected languages (2)

- Software component descriptions similar
  - On syntactical level, models can be easily mapped
  - Semantic description is missing
- Inconsistent approaches to modeling component families
  - Example: hardware element descriptions
    - AutoSAR 3.0: detailed definition of several types (uP, RAM, …)
    - AutoSAR 4.0: more generic, user
    - SysML: a generic *component* type (can be specialized)
  - No universal *taxonomy* of concept available
- Hardware-software integration is different
  - From high level (SysML *allocate* relationship)
  - To detailed (AutoSAR *System mapping*)

Analysis of the selected languages (3)

- Communication configuration
  - Different layer count
    - GENESYS – single layer (Message)
    - AutoSAR – three layers (Signal – PDU – Frame)
    - FIBEX 2.x – two layers (Signal – Frame)
  - Different abstraction levels
    - For synthesis, detailed information is necessary
- Platform (middleware) configuration
  - Not supported in several languages

Emerging requirements for new modeling aspects

- Requirement management integration
  - For traceability purposes
    - EAST-ADL2
- Product line and variability support
  - AutoSAR 4.0
  - EAST-ADL
- These should also be handled
  - Additional complexity on
    - Model language
    - Model management
    - and tool implementation levels

Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- **Integration concept for modeling notations**
- Semantics-based integration
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- Conclusions

The INDEXYS modeling notation

- Based on the analysis, a new notation has been defined
  - Covers all modeling stages
  - Covers all modeling aspects
  - Has precise semantics for the key areas
  - All corresponding notations can be mapped to it
- Usage scenarios
  - Modeling backend for modeling tools
    - The user works with the legacy syntax
    - The tools use the common notation (tool interoperability and porting)
  - Common basis for inter-language transformations
    - Used as intermediate model
    - Given n languages, the n^2 transformations reduced to 2*n

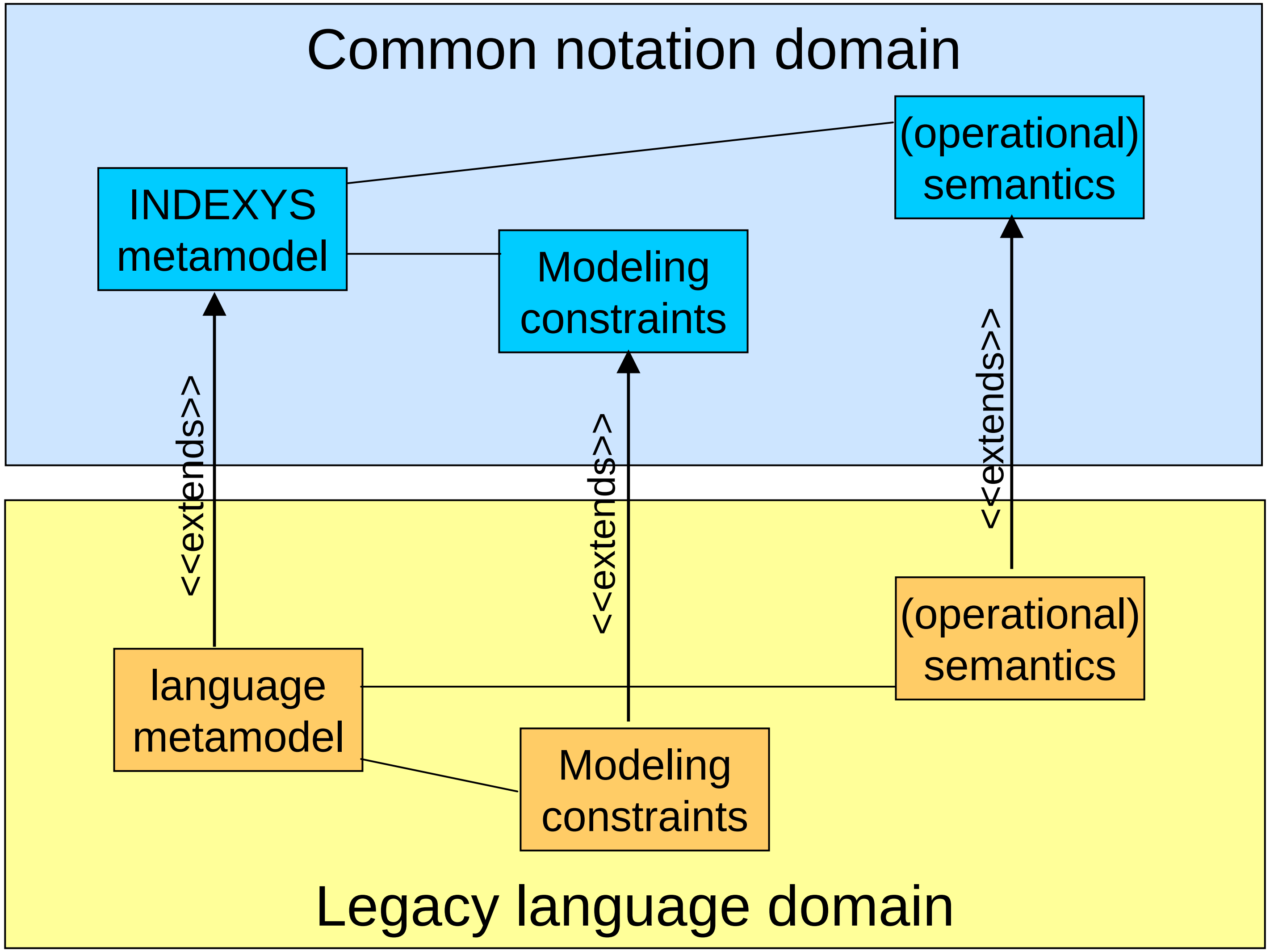
Why yet another notation?

- UML and profiles
  - UML is too complex, too generic, too imprecise
  - Profile mechanism proved to be inefficient in many projects
  - Profiles usually are imprecise – due to the limitations of the mechanism
- Existing languages
  - None of them covers all aspects
- Important note
  - Our proposal *does not aim at being a user-level* formalism
  - The key goal is the tight integration of different languages
    - The target audience is *tools* and *tool vendors*

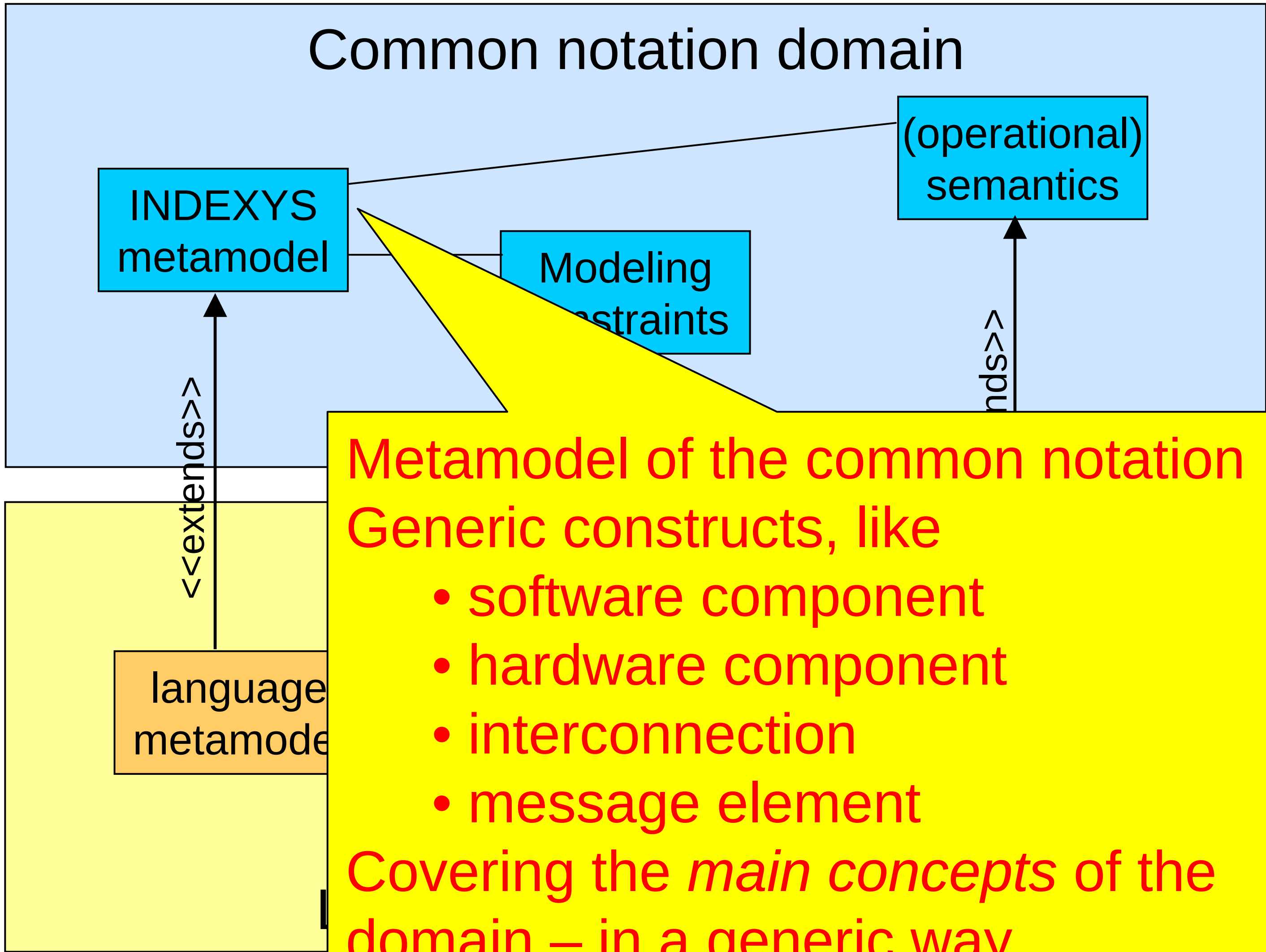
Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- **Semantics-based integration**
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- Conclusions

High-level structure of the INDEXYS notation



High-level structure of the INDEXYS notation



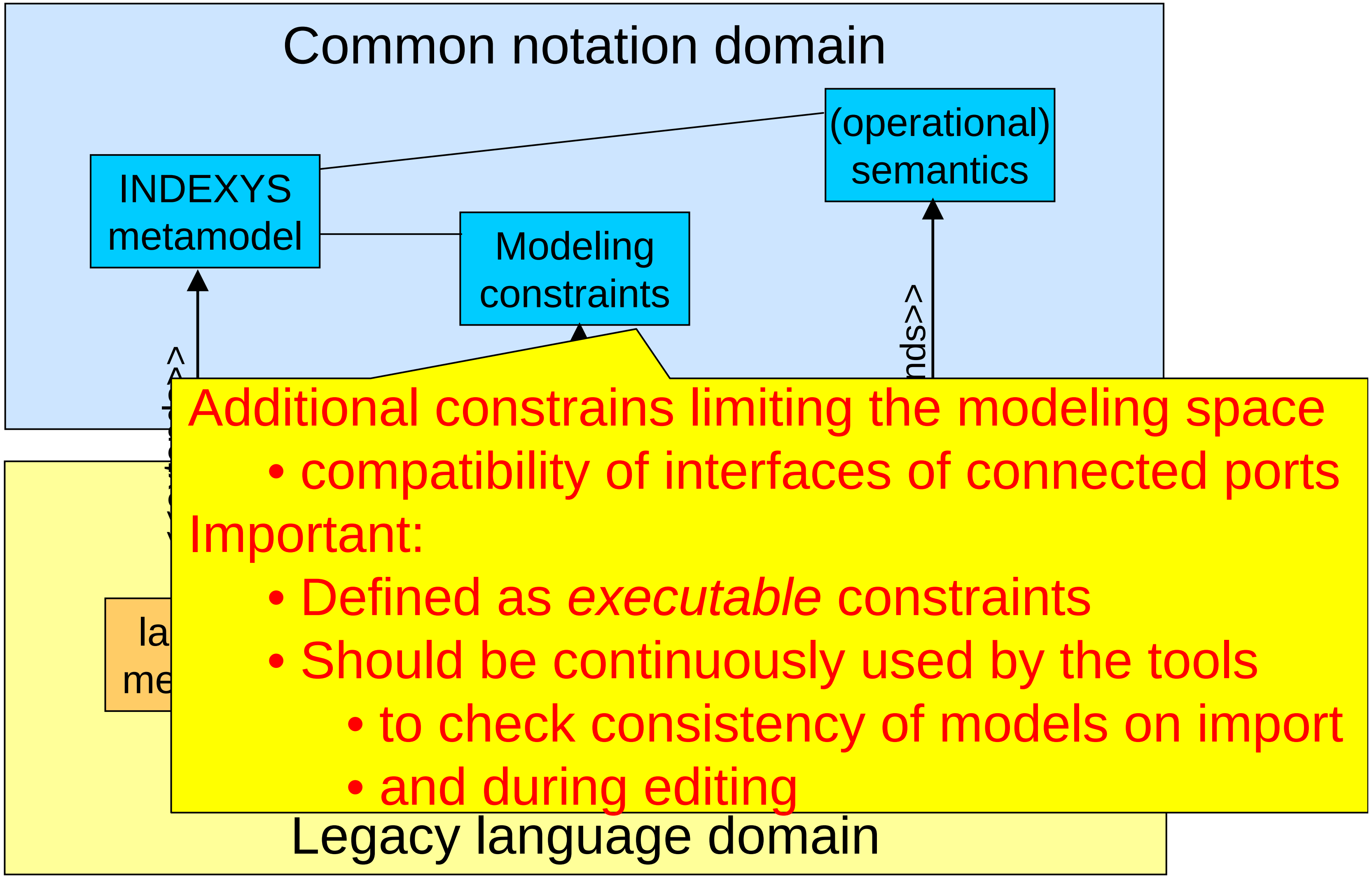
Metamodel of the common notation

Generic constructs, like

- software component
- hardware component
- interconnection
- message element

Covering the *main concepts* of the domain – in a generic way

High-level structure of the INDEXYS notation



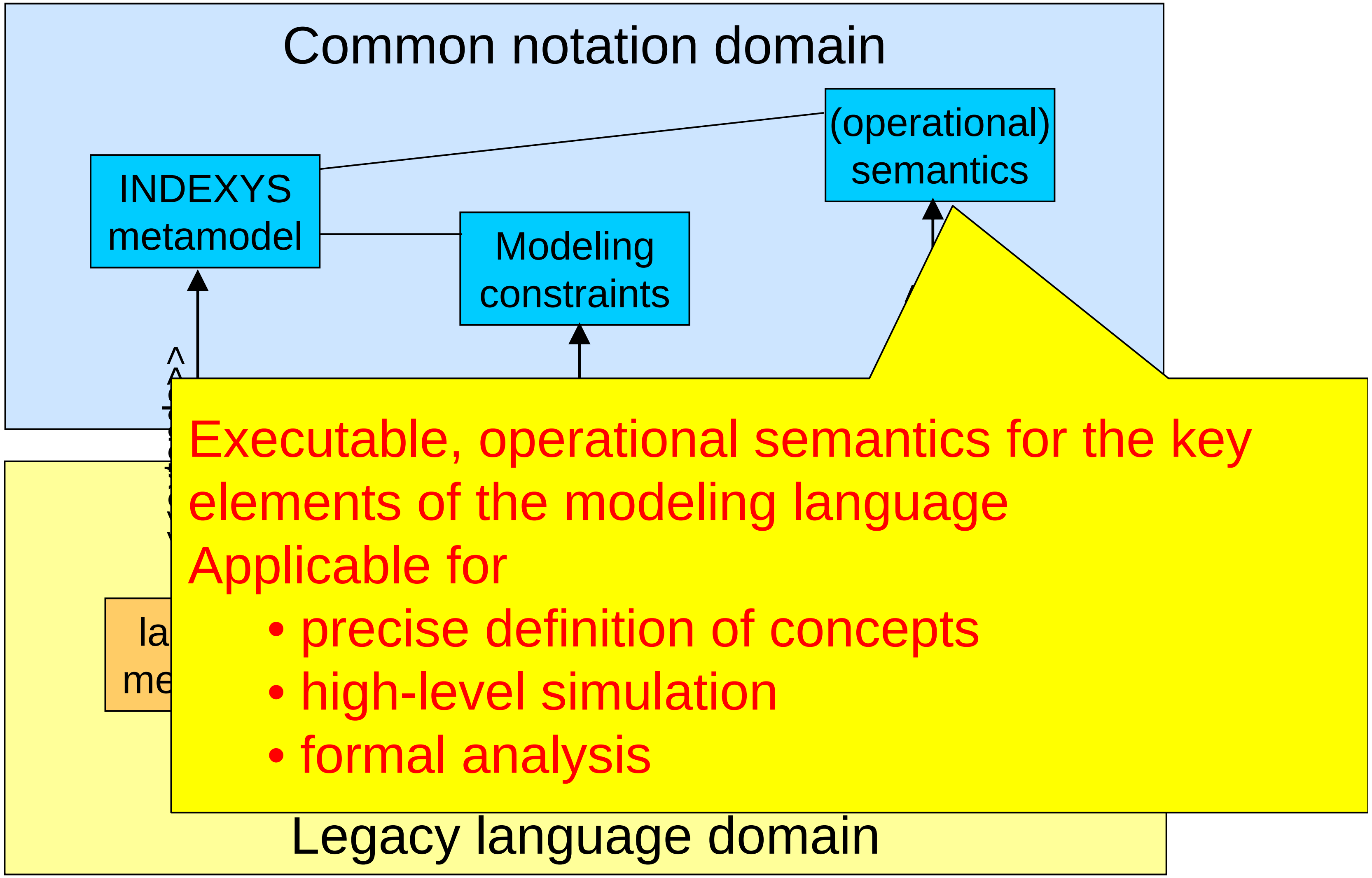
Additional constrains limiting the modeling space

- compatibility of interfaces of connected ports

Important:

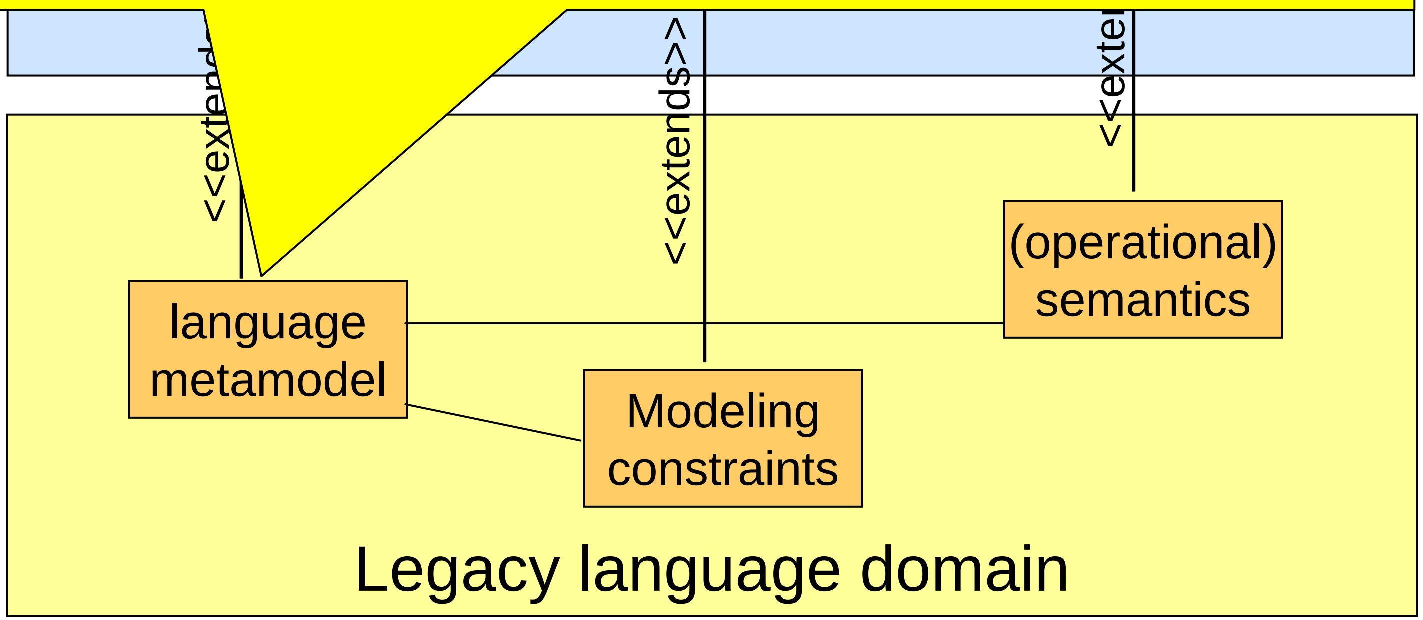
- Defined as *executable* constraints
- Should be continuously used by the tools
  - to check consistency of models on import
  - and during editing

High-level structure of the INDEXYS notation



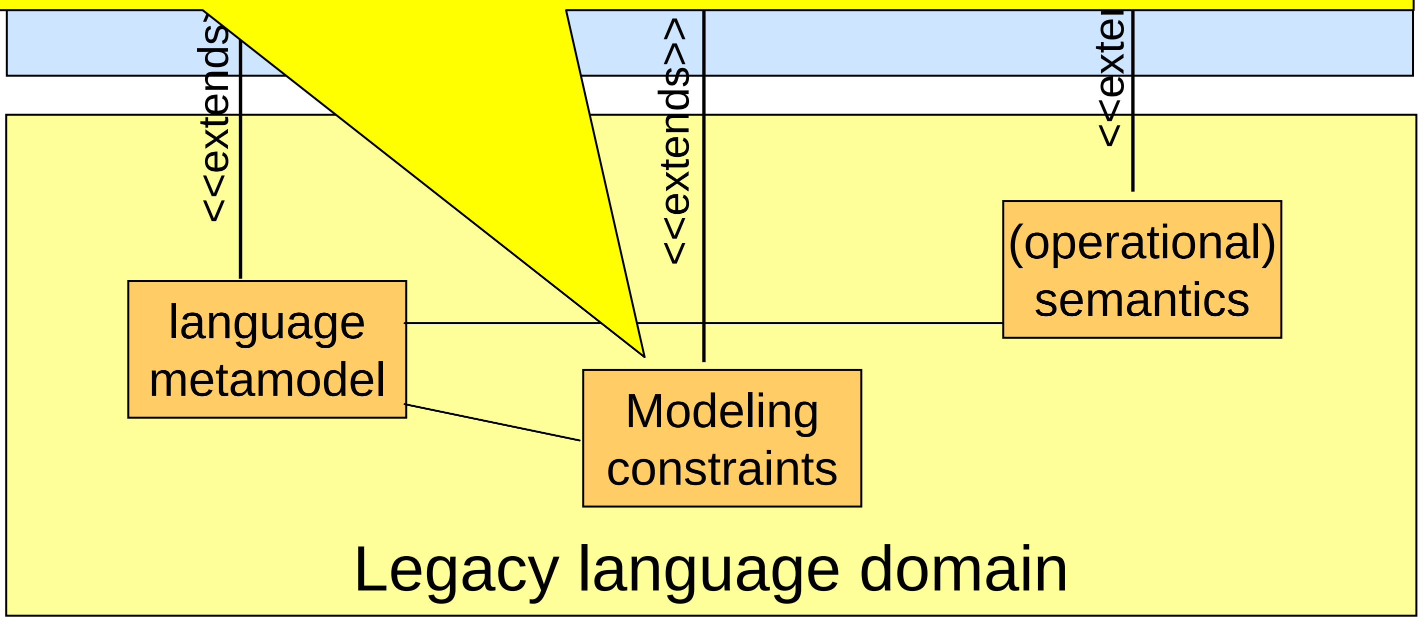
High-level structure of the INDEXYS notation

- Metamodel of a concrete, legacy language
- Connected to the common metamodel
  - via inheritence/extends relationships
- Acts as a *specialization* of the common model
- Can contain new concepts
  - new modeling aspects, etc.



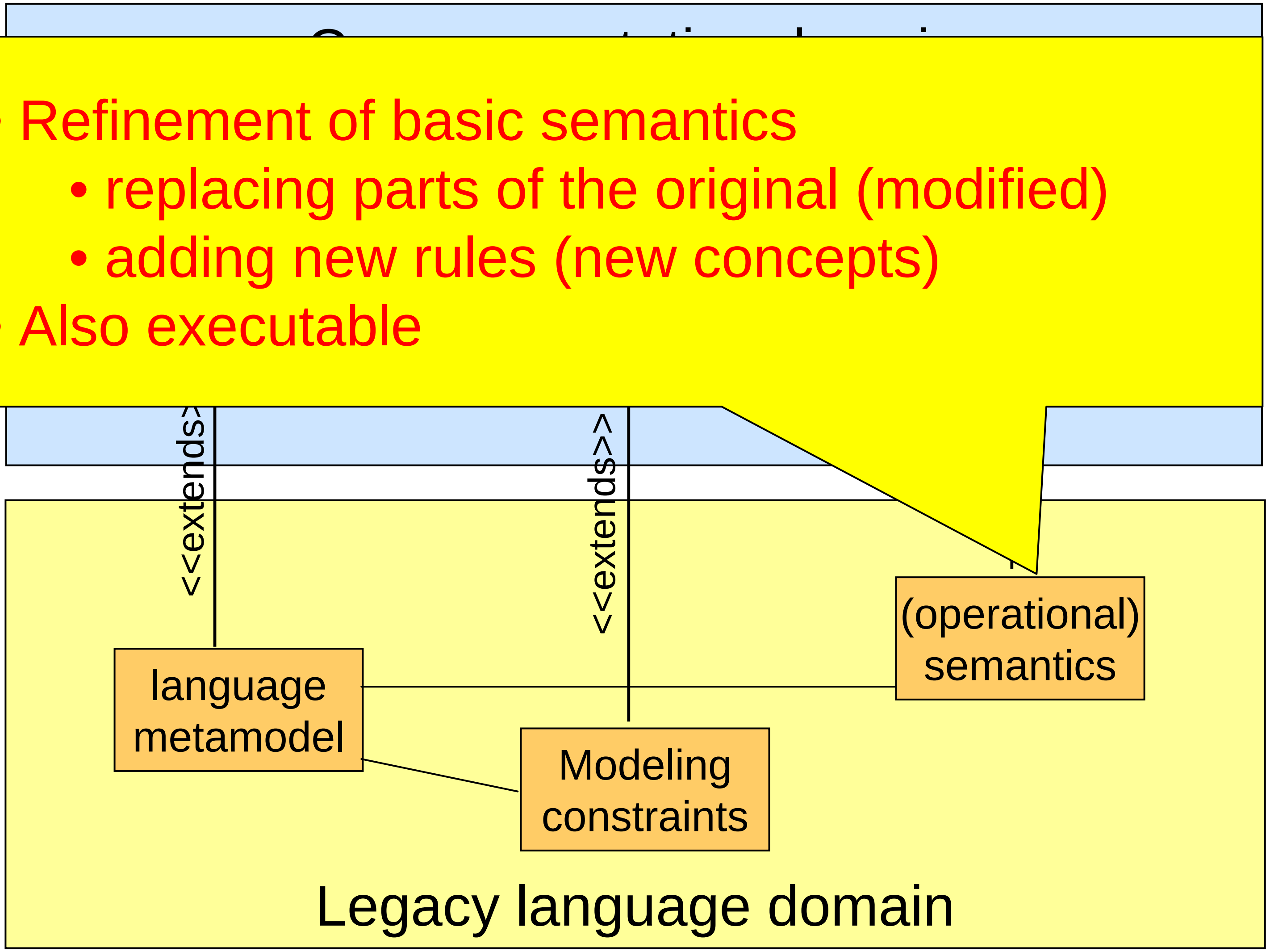
High-level structure of the INDEXYS notation

- Language specific constraints
- Can extend the basic set with
  - new constraints
  - more strict constraints
- Responsible for all constraints on novel modeling concepts introduced



High-level structure of the INDEXYS notation

- Refinement of basic semantics
    - replacing parts of the original (modified)
    - adding new rules (new concepts)
- Also executable



Benefits of the approach

- A common syntactical and semantic basis for all languages
  - Enables tool reuse
  - Enables *high-level, interactive simulation*
  - Enables *formal verification*
  - Eases inter-language mapping
    - Explicit constraints on both sides
    - Precise semantics on both sides
    - → detailed analysis of information loss possible
- By integrating several languages
  - End-to-end modeling is possible
  - End-to-end traceability can be implemented

Main partitions of the common ^notation

- Requirements modeling
- Feature modeling
  - high-level, user visible features
- Software component modeling
  - Components, interfaces, internal behavior
- Hardware resource modeling
  - HW component, peripherals
- System modeling
  - Topology definition
  - HW/SW mapping
  - Communication synthesis
- ECU configuration
  - Detailed configuration of HW and SW modules
- Non-functional properties
  - An extensible framework (based on MARTE) for QoS modeling and analysis

Non-functional property handling

- NFPs can be defined independently of the main metamodel
  - Allows extensibility for new aspects
- A core, consistent framework available
  - Based on the concept of UML MARTE
  - Allowing the uniform treatment of properties
- NFP calculus can be defined as
  - Set of constraints
  - And operational semantics rules
  - Or by mapping the corresponding entities to an analysis language
- Rationale
  - Existing solutions are inflexible as they constraint the usable NFPs
    - Like DECOS PIM
  - It is not possible to define all aspects centrally in all details
- A common taxonomy is needed for NFPs!

Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- Semantics-based integration
- **Ontology-based component catalogues**
- Towards cross-domain integrated modeling
- Conclusions

Need for common taxonomies

- Several element hierarchies in modeling
  - HW part types
  - NFPs
  - Physical units
- There is a natural *inheritance hierarchy* between the instances
  - Example: microcontroller → 32bit MCU → ARM core → Cortex M3 → LPC17xx series → LPC1768
  - Properties and capabilities inherited from the hierarchy
  - Tools (configuration, etc.) applicable to sub-trees in the hierarchy
    - An RTOS is applicable for all Cortex M3 processors
    - Currently: long list of types included
- No uniform taxonomy exists
  - Would enrich the semantics of models
  - And the interoperability of tools and design environments

Ontology - overview

- *“The term ontology has its origin in philosophy, and has been applied in many different ways. The core meaning within computer science is a model for describing the world that consists of a set of types, properties, and relationship types.”* Wikipedia
- Similar concept to meta-modeling, but
  - Assumes an *open* (incompletely modeled) world
  - Has its own calculus
- Utilization
  - Build up ontology for all modeling aspects that need taxonomies
  - Use a single, coherent ontology
    - Like the system of IP addresses, MAC addresses, etc.
    - Requires coordinated development
  - Elements become uniquely identifiable and *classifiable*

Ontology - benefits

- A single uniform classification of entities
- Systematic definition of properties and relationships
- Direct benefit
  - For tool vendors
    - Tool capabilities can be precisely specified
      - Target platforms
      - Supported protocols, etc.
    - Tool interoperability increases
  - For users (system designers)
    - No need for manual remodeling of elements in each project
    - Easier to find appropriate tools
    - Easier to migrate from a component/platform/etc. to an other
      - The relation of the current and future item can be analyzed

Ontology – open questions

- Maintaining the *central ontology*
  - Legal problems
  - Organizational problems
  - Technical problems
- Tool support
  - For ontology maintenance
  - For (remote) access to it
  - For designers – should be built in into design tools
- Current status
  - Prototypical experiments

Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- Semantics-based integration
- Ontology-based component catalogues
- **Towards cross-domain integrated modeling**
- Conclusions

The integrated modeling vision

- A single (invisible) model management system
  - Relies on state-of-the art platform solutions
  - Supports the INDEXYS common meta-model
  - Supports constraint checking and semantics execution
- Users can work with their domain-specific languages
  - Several languages
  - For different design stages
- Migration of models eases
  - Between projects (common ontology)
  - Between application domains (common meta-model)
  - Between tools (common basis)

Implementation Technology

- Modeling
  - Using the de-facto Eclipse Modeling Framework as front-end for meta-modeling
  - Using VPM (Visual Precise Meta-modeling) as backend
    - Formal definition of languages
    - Integrated constraint checking support
    - Integrated model execution (via transformations support)
- Model storage – OptXware ModelServer
  - Distributed, client-server solution
  - Supports team collaboration by real-time model sharing
  - Supports the use of several modeling languages simultaneously
  - Integrates VIATRA2 – one of the most robust model transformation engines
- Tool frontend
  - Relies on the Eclipse platform
    - Offers form and diagram based model editing
  - Integrates legacy tools via tool adapters

The INDEXYS WP1 roadmap

- Current status
  - Common meta-model work-in-progress
    - First versions available
    - Semantics definition and verification in progress
    - First prototypical tools available
- Progress
  - By the end of 2010
    - Final common meta-model available
      - » With mappings to at least GENESYS and AutoSAR
    - An *open, free* tool suite available
      - » Demonstrating the capabilities of the concept
      - » Extensible
  - By 2Q/2011
    - The Embedded Architect tool suite will be migrated to the new concept
      - » Primary product line of OptXware for embedded system design

Outline

- Embedded systems overview
- Overview of the GENESYS-INDEXYS approach
- Current modeling languages for embedded systems
- Integration concept for modeling notations
- Semantics-based integration
- Ontology-based component catalogues
- Towards cross-domain integrated modeling
- **Conclusions**

Conclusion

- Current problems in MDD for ES
  - Heterogeneous languages and tools
  - Imprecise semantics of component types, and language elements
  - Inefficient migration between projects, tools, and domains
- The INDEXYS solution
  - Common, precise modeling backend
  - Common ontology for element taxonomies
  - Integrated treatment of models of diverse languages
- Technology
  - Team-collaboration support
  - Robust modeling and model transformation support
  - Using the OptXware ModelServer technology

ARTEMISIA ASSOCIATION

The association for R&D actors in the field of ARTEMIS

INDEXYS: http://www.indexys.eu

INDustrial EXploitation of the indexys genesYS cross-domain architecture

ARTEMIS

Thank you for your attention

András BALOGH,

INDEXYS WP1 Leader

OptiXware Research & Development Ltd.

Tel: +36 1 81 49 057

Mail-to: balogh@optxware.com