Model-Based Software for Embedded Systems

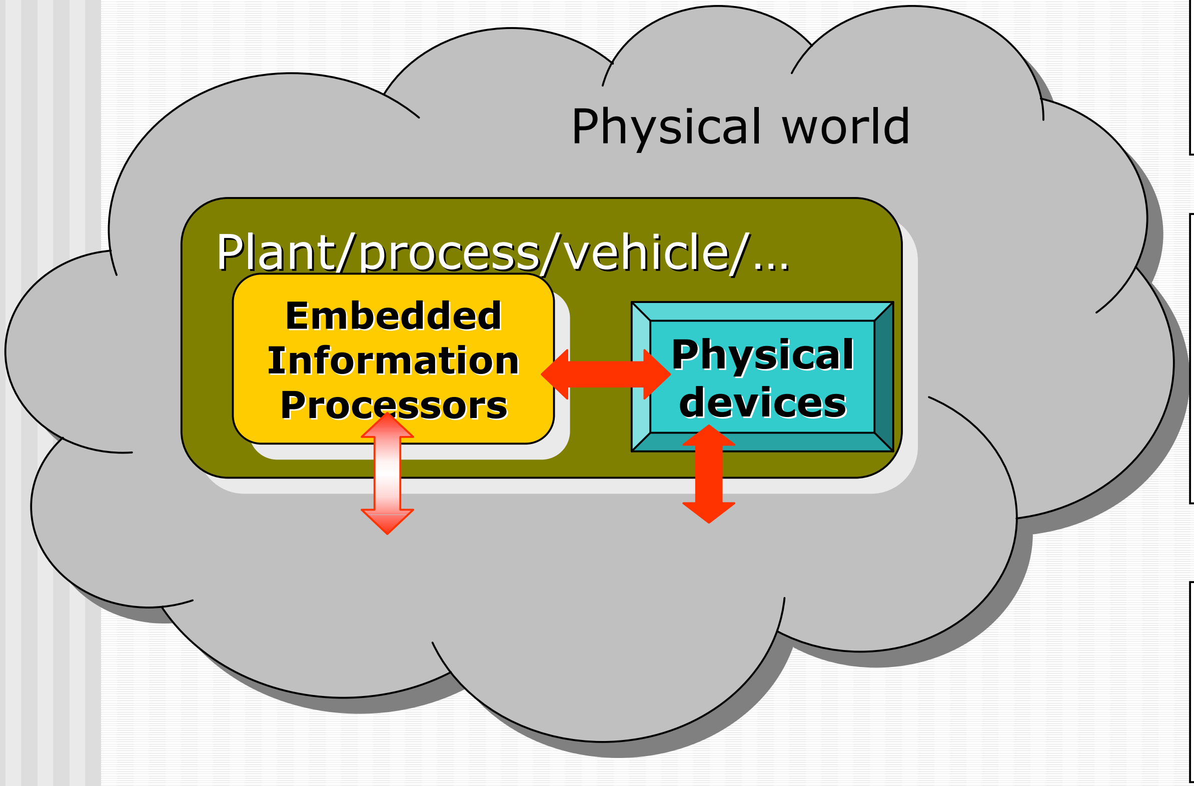
Gabor Karsai

ISIS

Vanderbilt University

VANDERBILT UNIVERSITY
INSTITUTE FOR SOFTWARE INTEGRATED SYSTEMS

Embedded Systems are ***Physical*** Systems



Physical world requirements, like timing, safety, reliability have a profound impact on the software design

❝ An *integrated view* is needed

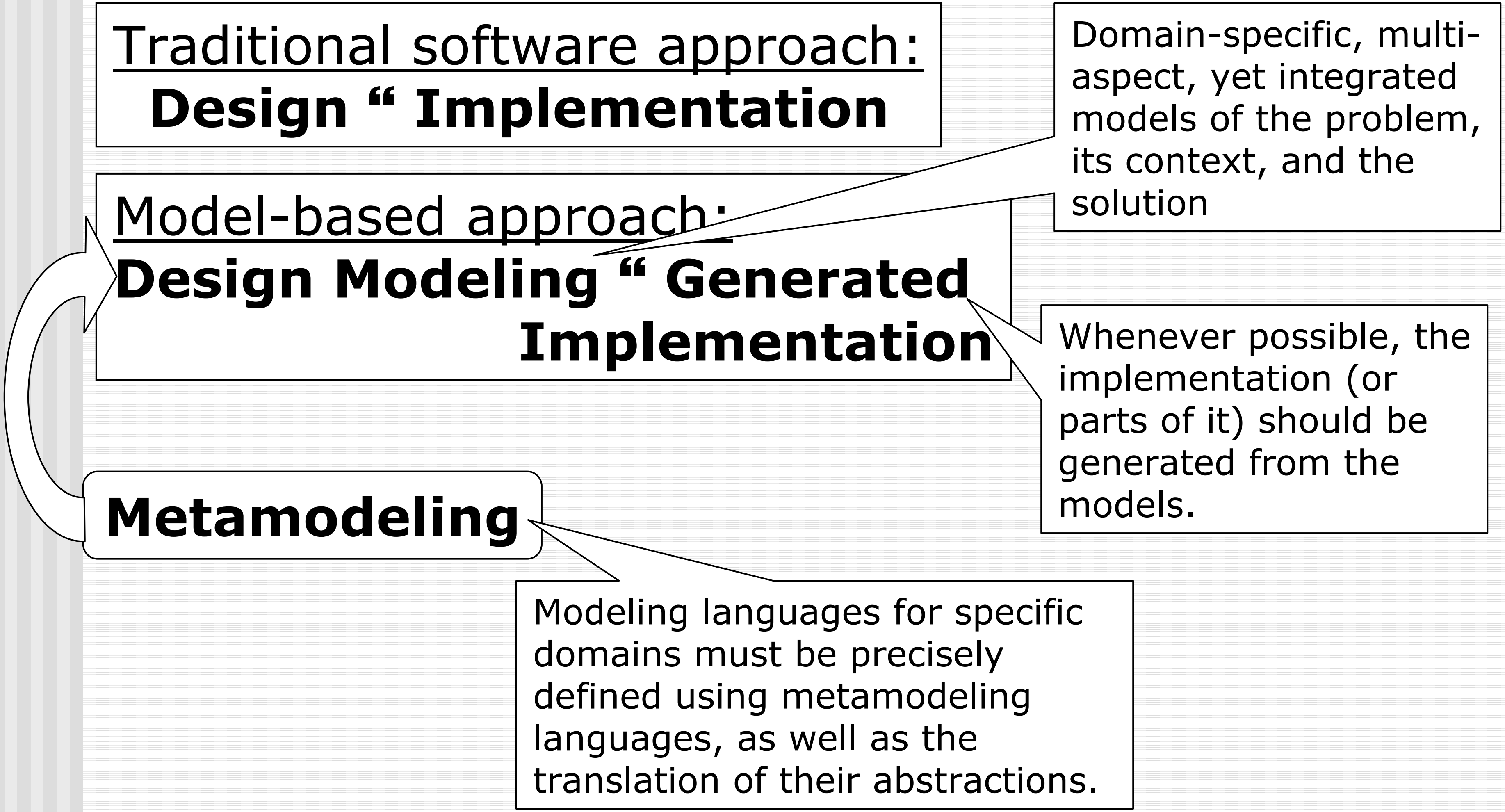
Abstraction layers don’t work well in embedded software: implementation details often “shine through”

❝ A *crosscutting approach* is needed

Centralized processing is being replaced by local, networked processing

❝ A *distributed view* is needed

Towards a ***Model-based*** Approach



Traditional software approach:

Design “ Implementation

Model-based approach:

Design Modeling “ Generated Implementation

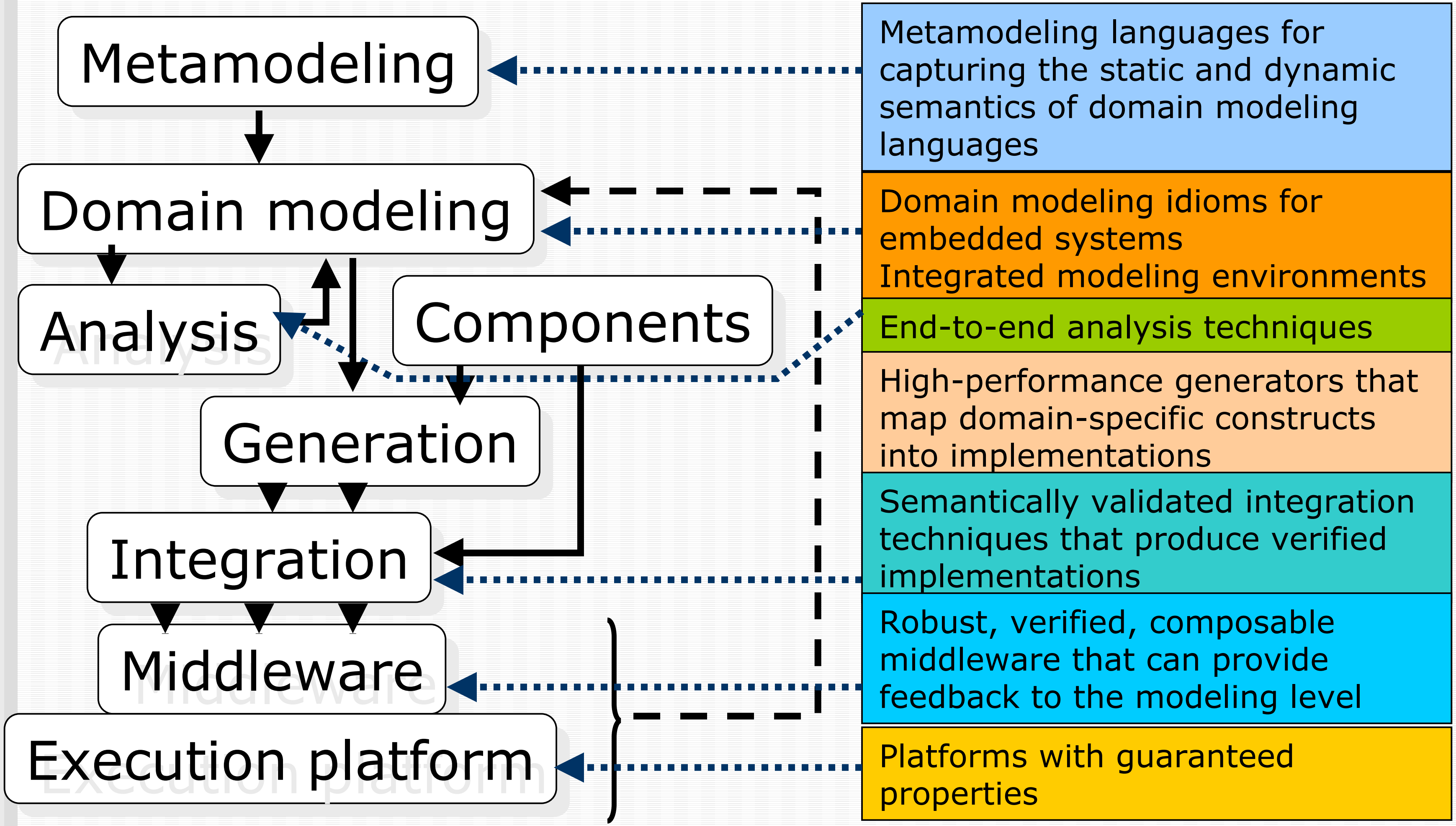
Domain-specific, multi-aspect, yet integrated models of the problem, its context, and the solution

Whenever possible, the implementation (or parts of it) should be generated from the models.

Metamodeling

Modeling languages for specific domains must be precisely defined using metamodeling languages, as well as the translation of their abstractions.

WANTED: ***Theory*** and ***Tools***



Metamodeling

Domain modeling

Analysis

Components

Generation

Integration

Middleware

Execution platform

Metamodeling languages for capturing the static and dynamic semantics of domain modeling languages

Domain modeling idioms for embedded systems

Integrated modeling environments

End-to-end analysis techniques

High-performance generators that map domain-specific constructs into implementations

Semantically validated integration techniques that produce verified implementations

Robust, verified, composable middleware that can provide feedback to the modeling level

Platforms with guaranteed properties