澳 門 理 工 學 院
Instituto Politécnico de Macau
Macao Polytechnic Institute

Online Learning Materials

COMP223: Software Engineering

System Modeling

Dr. Kim, Song-Kyoo (Amang)
Associate Professor,

Computer Science Program
MACAO POLYTECHNIC INSTITUTE
Macau, SAR

Session (Chapter 5) Objectives

- Context models
- Interaction models
- Structural models
- Behavioral models
- Model-driven engineering

Online Learning Materials

澳 門 理 工 學 院
Instituto Politécnico de Macau
Macao Polytechnic Institute

Model Driven Engineering

Model-driven engineering (1/8)

- **Model-driven engineering (MDE)** is an approach to software development where models rather than programs are the principal outputs of the development process.
- The programs that execute on a hardware/software platform are then generated automatically from the models.
- Proponents of MDE argue that this raises the level of abstraction in software engineering so that engineers no longer have to be concerned with programming language details or the specifics of execution platforms.

Model-driven engineering (2/8)

- Usage of model-driven engineering
  - Model-driven engineering is still at an early stage of development, and it is unclear whether or not it will have a significant effect on software engineering practice.
  - **Pros**
    - Allows systems to be considered at higher levels of abstraction
    - Generating code automatically means that it is cheaper to adapt systems to new platforms.
  - **Cons**
    - Models for abstraction and not necessarily right for implementation.
    - Savings from generating code may be outweighed by the costs of developing translators for new platforms.

Model-driven engineering (3/8)

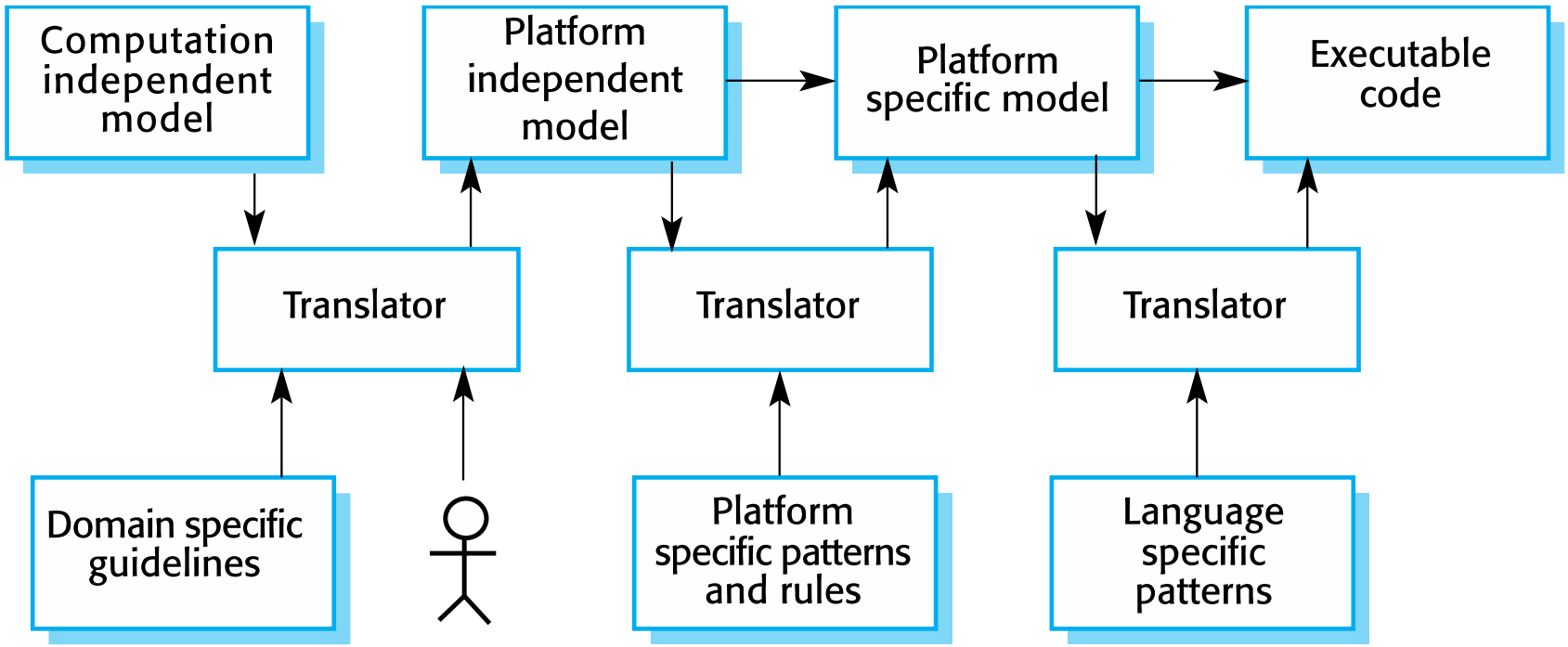
- **Model driven architecture (MDA)**
  - Model-driven architecture (MDA) was the precursor of more general model-driven engineering
  - MDA is a model-focused approach to software design and implementation that uses a subset of UML models to describe a system.
  - Models at different levels of abstraction are created.
  - From a high-level, platform independent model, it is possible, in principle, to generate a working program without manual intervention.

Model-driven engineering (4/8)

- Types of model
  - **A computation independent model (CIM):** It is the important domain abstractions used in a system. CIMs are sometimes called domain models.
  - **A platform independent model (PIM):** The operation of the system without reference to its implementation. The PIM is usually described using UML models that show the static system structure and how it responds to external and internal events.
  - **A Platform specific model (PSM):** It is a transformation of the platform-independent model with a separate PSM for each application platform. In principle, there may be layers of PSM, with each layer adding some platform-specific detail.

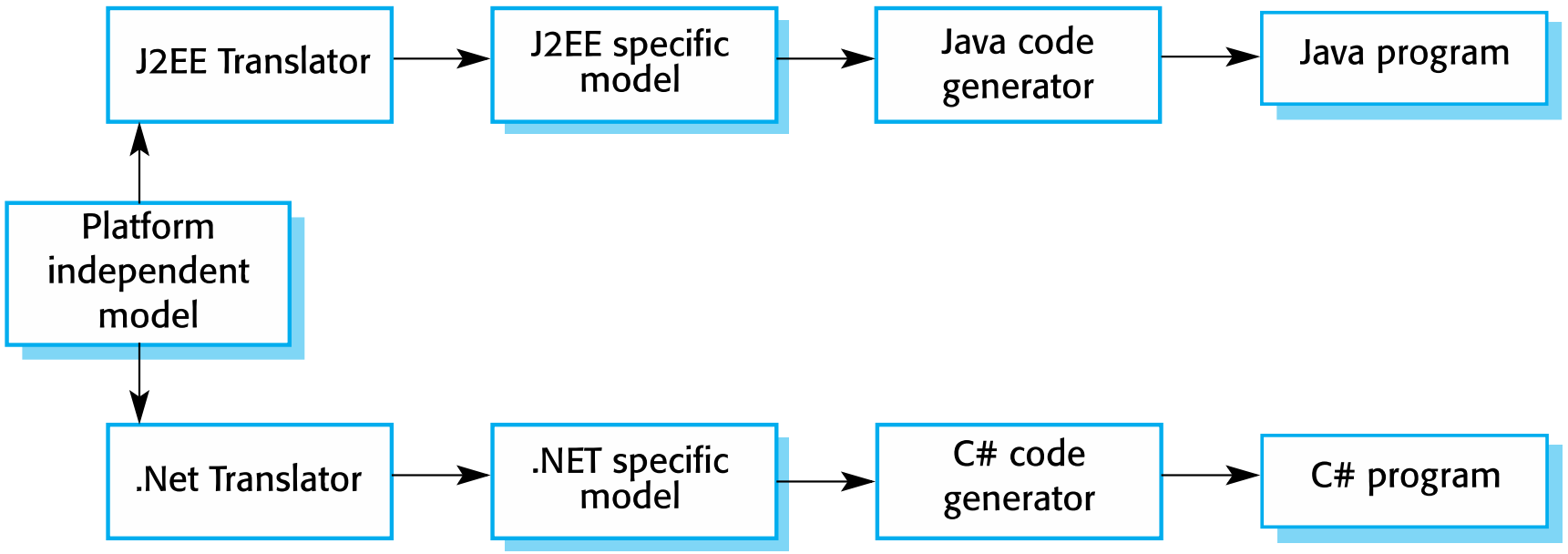
Model-driven engineering (5/8)

- MDA transformations



Model-driven engineering (6/8)

- Multiple platform-specific models



Model-driven engineering (7/8)

- Adoption of MDA
  - A range of factors has limited the adoption of MDE/MDA.
  - Specialized tool support is required to convert models from one level to another.
  - There is limited tool availability and organizations may require tool adaptation and customization to their environment.
  - The widespread adoption of agile methods over the same period that MDA was evolving has diverted attention away from model-driven approaches.
  - The arguments for platform-independence are only valid for large, long-lifetime systems.

Model-driven engineering (8/8)

- Adoption of MDA (cont.)
  - For software products and information systems, the savings from the use of MDA are likely to be outweighed by the costs of its introduction and tooling.
  - For the long-lifetime systems developed using MDA, companies are reluctant to develop their own tools or rely on small companies that may go out of business.
  - The abstractions that are useful for discussions may not be the right abstractions for implementation.
  - For most complex systems, implementation is not the major problem → requirements engineering, security and dependability, integration with legacy systems and testing are all more significant.

Session Summary (1/3)

- A model is an abstract view of a system that ignores system details.
- Complementary system models can be developed to show the system's context, interactions, structure and behavior.
- Context models show how a system that is being modeled is positioned in an environment with other systems and processes.
- State diagrams are used to model a system's behavior in response to internal or external events.

Session Summary (2/3)

- Use case diagrams and sequence diagrams are used to describe the interactions between users and systems in the system being designed.
- Use cases describe interactions between a system and external actors; sequence diagrams add more information to these by showing interactions between system objects.
- Structural models show the organization and architecture of a system.
- Class diagrams are used to define the static structure of classes in a system and their associations.

Session Summary (3/3)

- Behavioral models are used to describe the dynamic behavior of an executing system.
- This behavior can be modeled from the perspective of the data processed by the system, or by the events that stimulate responses from a system.
- Activity diagrams may be used to model the processing of data, where each activity represents one process step.
- Model-driven engineering is an approach to software development in which a system is represented as a set of models that can be automatically transformed to executable code.


澳門理工學院
INSTITUTO POLITÉCNICO DE MACAU
MACAO POLYTECHNIC INSTITUTE