Model-Driven Engineering

F. Mallet
Frederic.Mallet@unice.fr

Université Nice Sophia Antipolis
Introduction

Outline

- MDE & Meta-Model
- UML Profiling mechanism
- Meta-Modeling & EMF

Application

- BPEL & BPMN
  - Domain Model
  - UML Profile
  - EMF Metamodel
MODEL-DRIVEN ENGINEERING
Model-Driven Engineering

- Software Engineering
  - Build software useful to end-users to solve a particular problem

- Model-Driven Engineering
  - Build models to help software engineer to build faster, better software able to handle more complex problems
  - Generative approaches
Model-Driven Engineering

- Coping with complexity
  - Reduce *accidental complexity*
  - Help build tools to build software

- A **model** is a representation of a thing that highlights some of its properties
  - Focuses on a **specific** viewpoint/aspects
  - Serves a particular purpose
  - Evolves when the system evolves!
  - Is used to derive the code **automatically**
Model: an example

- **Genealogy:** *model of a family*
  - Focus: family relationships
  - Purpose: keep track of ancestors and siblings

*A Model of the Thing*
Meta-Model: A model of a Model

Metamodel of ANY family

- Family
  - Set of persons
- Person
  - Name
  - Set of children
  - 2 parents (father, mother)
- Man
  - Is-a person
- Women
  - Is-a person

A model of ONE family
Model-Driven Engineering and UML Profiles

Meta-Model: A model of a Model

M2

A Model of the Thing: A metamodel

M1

A Thing: A model

F. Mallet
What are the models in Computer Science?

- A program is a model of a system (staff of a company, accounts in a bank, engine controller)
- To reason about the programs, we need to build models of the programs
  - Models about the data structure: classes, fields and methods
  - Models about the behavior/algorithm: state machines, scenarios, data and control flows
abstract class AbstractBeast {
    protected int x, y;  // position on the screen
    int speed;           // speed [pix/s]
    double direction;   // radians [0 - 2 PI]
    protected Color color;  // Filling color
    protected BeastField field;  // the field
    static final int SIZE = 10;

    protected AbstractBeast(BeastField field, int x, int y, Color color) {
        this.field = field;
        this.x = x;
        this.y = y;
        this.color = color;
    }

    public abstract void act();

    public boolean see(AbstractBeast b) {
        double angle = Math.atan2(b.getY()-y, b.getX()-x);
        double diff = Math.abs(angle-direction)%2*Math.PI;
        if (diff>Math.PI) diff=2*Math.PI-diff;
        return diff<champDeVue/2;
    }

    public double getDistance(IBeast b) {
        return distanceFromAPoint(b.getX(), b.getY());
    }

    double distanceFromAPoint(double x1, double y1) {
        // @returns distance between the beast and a point
        return Math.sqrt((x1 - x)*(x1-x) + (y1 - y)*(y1 - y));
    }

    protected IView view = null;
    void drawYourself(Graphics g) {
        if (this.view != null)
            this.view.draw(g, this);
    }

    final public void translate(double dx, double dy) {
        this.x += dx;
        this.y += dy;
    }

    public boolean see(Aesthetic sustainable) {
        return see(sustainable);
    }

    public double getDistance(Aesthetic sustainable) {
        return getDistance(sustainable);
    }
}

F. Mallet
M1 - A Model of the Program

BeastField

field

* beasts

« abstract »
AbstractBeast

position : Point
speed : Integer
direction : Real
color : Color

Beast

view

« interface »
IView

draw(Graphics g, AbstractBeast b)

behavior

« interface »
IBehavior

act(AbstractBeast b)

NominalBehavior

F. Mallet
M2 - A Model of ANY Program

A MetaModel of Java Classes
Models/Metamodels/Languages

M3: Meta-Metamodel
- Conforms to BNF grammar
- Belongs to Meta-language
  - E.g., MOF

M2: Metamodel
- Conforms to BNF grammar
- Belongs to Language
  - E.g., UML

M1: Model
- A description of the Thing
- Belongs to Thing
  - E.g., classes, SM

M0: Implementation

F. Mallet
Summary

Model-Driven Engineering

- Attempts to replace programs by models
- Useful when the code is generated from the models (~80%)