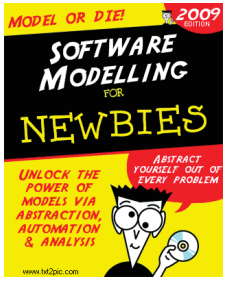
Beyond Code:
An Introduction to Model-Driven Software Development (CISC836)



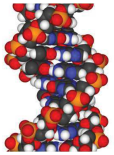
Topic 1: What is a model?

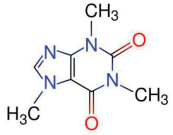
Juergen Dingel
Jan 2021

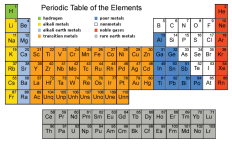
Today's Lecture

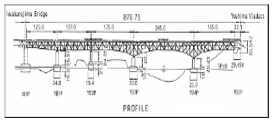
- **Topic 1: Observations about modeling in general**
  - Examples
  - What is a model?
  - What are they used for?
  - Activities on models
- **Topic 2: Intro to software modeling**
  - What is being modeled?
  - Why?
  - How? I.e., what is a good model?
  - Models as primary artifact
    - Model-Driven Development (MDD)
    - Examples: EGGG, IBM RoseRT, EMF

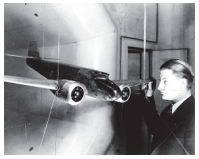
Some Examples

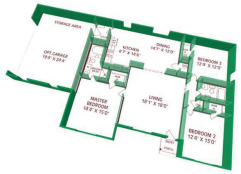








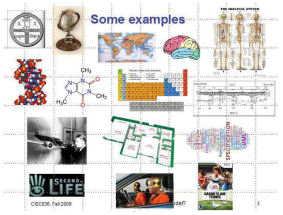




Definitions (Cont'd)

> *"Modeling, in the broadest sense, is the cost-effective use of something in place of something else for some cognitive purpose.*
>
> *It allows us to use something that is simpler, safer or cheaper than reality instead of reality for some purpose.*
>
> *A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality.*
>
> *This allows us to deal with the world in a simplified manner, avoiding the complexity, danger and irreversibility of reality."*

Which of these are models in the above sense?



[J. Rothenberg. The Nature of Modeling. In Artificial Intelligence, Simulation, and Modeling, L.E. William, K.A. Loparo, N.R. Nelson, eds. New York, John Wiley and Sons, Inc., 1989, pp. 75-92]

Models, Abstraction, and Interpretation

Interpretation

Add detail

i : Model x Reality

Not necessarily a function

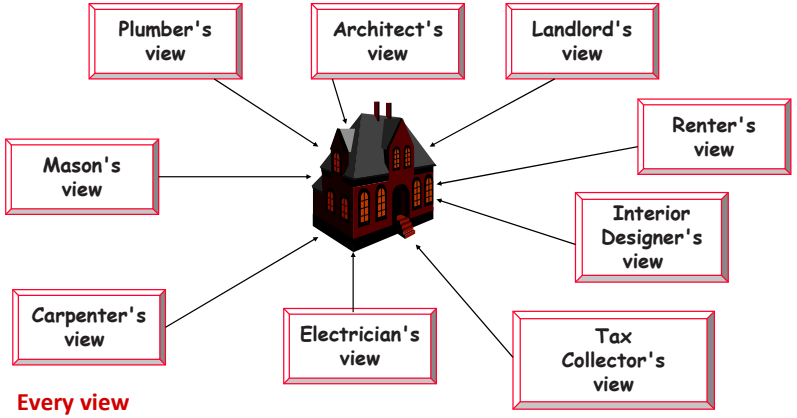
Abstraction

Remove detail

a : Reality → Model

Typically, there's more than one useful abstraction

Models as Views



Every view

- obtained via different abstraction
- may be expressed in different notation (modeling language)
- reflects different intent

[Example from J. Bezivin]

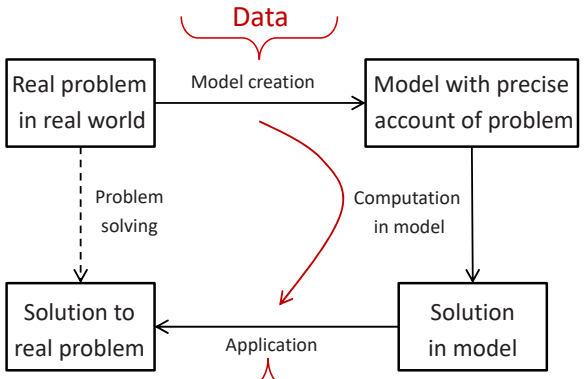
Modeling to Solve Problems

"Executable model of gene expression"
[Fisher et al, CAV'16]

"Russian Track and Field Team Barred From Rio Olympics"
[NY Times, 06/17/16]

"CDC Concludes Zika Causes Microcephaly"
[04/13/16]

"Programs can be analyzed productively with statistical models"
[Devanbu et al, CACM'16]



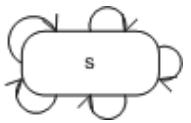
Management recommendation Ad placement

Scientific result Medical guideline

Policy decision Maintenance plan

But, Models Can be Useless, Harmful

Interpretation



Abstraction

- Still useful?
  - Depends on
- Must be careful not to remove too much detail

"Make everything as simple as possible, but no simpler" [A. Einstein]

Models Can be Useless, Harmful (Cont'd)

- Models may only be used for their purpose
- **Example 1:**

« Could I travel from Paris to Anchorage without using a boat? »

« What is the temperature at the bottom if I dig a 100 km deep hole at the surface of the earth ? »

[Example from J. Bezivin]

⇒ Purpose of model must be clear!

Models Can be Useless, Harmful (Cont'd)

If the model is incorrect, the action may be inappropriate

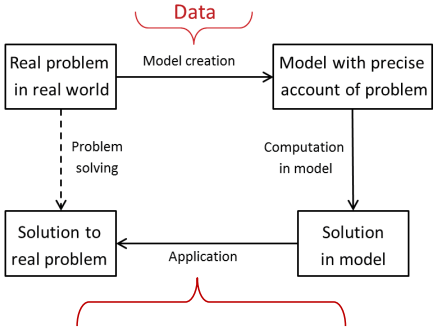
Example 2: The medical technique of bloodletting was based on an incorrect model of the body:

- Hippocrates and many others believed that
  - four crucial elements earth, air, water and fire were balanced within the human body as the four humors: blood, phlegm, and black and yellow bile.
  - disease was due to an imbalance in the four humors.
  - treatment involved restoring their balance through bloodletting.
- In 1799, George Washington died after heavy blood loss sustained in a bloodletting treatment for laryngitis.

[Example from J. Bezivin]

⇒ Model must be validated with respect to purpose

But Models can be Useless, Harmful (Cont'd)



"Debt is bad for growth"

Growth in a Time of Debt

By Carmen M. Reinhart and Kenneth S. Rogoff

American Economic Review: Papers & Proceedings 100 (May 2010): 573–578

https://en.wikipedia.org/wiki/Growth_in_a_Time_of_Debt

Methodology flawed! [2013]

But What Exactly is the Purpose of Models?

On the first glance

- Facilitate/enable analysis, prediction, understanding, decision making
- depending on intent

But need to make important distinction

- **Descriptive models**: model as description

  Reality ⇒ Model ⇒ Predictions & actions

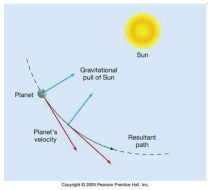
  used in natural sciences and "*backwards engineering*"
- **Prescriptive models**: model as specification

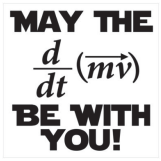
  Reality ⇒ Model ⇒ Actions ⇒ Product

  used in "*forwards engineering*"

Models in Natural Sciences

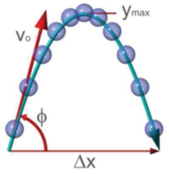
- Mostly descriptive
- Facilitate predictions
- Validation via experiments
- Examples:
  - Kepler's laws of planetary motion (ca. 1605)
  - Newton's laws of motion (1687)
    - together with law of gravitation explains Kepler's laws
    - *Law 2:*
      "the acceleration of an object is proportional to the force applied, and inversely proportional to the mass of the object": F = m*a

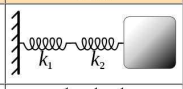




Models in Natural Sciences (Cont'd)

- Examples:
  - Classical mechanics
    - mathematically describe movement of bodies under the influence of forces
    - refinements:
      - quantum mechanics, relativistic mechanics
  - Physics: Coil springs
    - How do springs behave? In isolation? In combination?
    - Hooke's law:
      "the extension of a spring is in direct proportion with the load added to it as long as this load does not exceed the elastic limit": F = -k*x
      where F is force, x is displacement, and k is spring constant



Comparison	In Series	In Parallel
		
Equivalent spring constant	$\frac{1}{k_{eq}} = \frac{1}{k_1} + \frac{1}{k_2}$	$k_{eq} = k_1 + k_2$
Compressed distance	$\frac{x_1}{x_2} = \frac{k_2}{k_1}$	$x_1 = x_2$
Energy stored	$\frac{E_1}{E_2} = \frac{k_2}{k_1}$	$\frac{E_1}{E_2} = \frac{k_1}{k_2}$

Models in Economics

- Examples:
  - Financial models
    - MONIAC (Monetary National Income Analogue Computer) to model the national economic processes of the UK (1942)
      http://en.wikipedia.org/wiki/Moniac

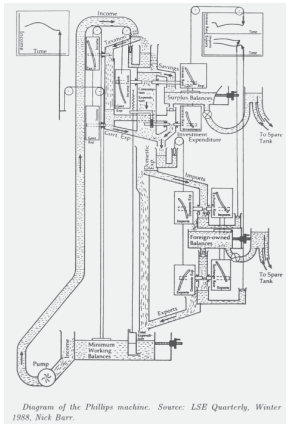
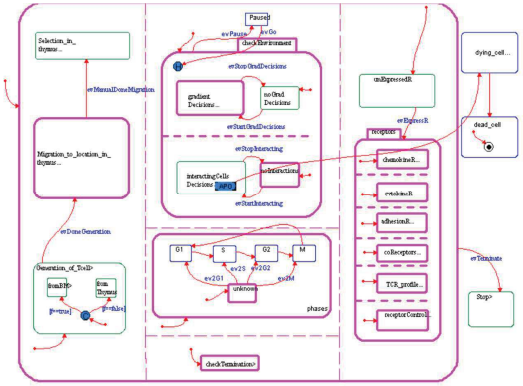


Diagram of the Phillips machine. Source: LSE Quarterly, Winter 1988, Nick Barr.

    - http://www.nytimes.com/interactive/2015/12/16/upshot/fed-interest-rates-rube-goldberg-machine.html

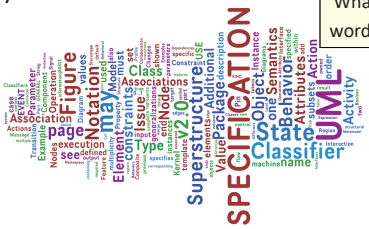
Models in Biology

- Examples:
  - Model of T cell development in the thymus gland [EHC03]



Models in Document Representation and Retrieval

- E.g., approaches based on word frequency (ignore: where in document a term occurs)
  - **"Word clouds"** created by **www.wordle.net**



What is this a word cloud of?

- **Vector Space Model**
  - represent each document by vector
  - similarity given by proximity b/w vectors

TERM VECTOR MODEL BASED ON $w_i = tf_i * IDF_i$

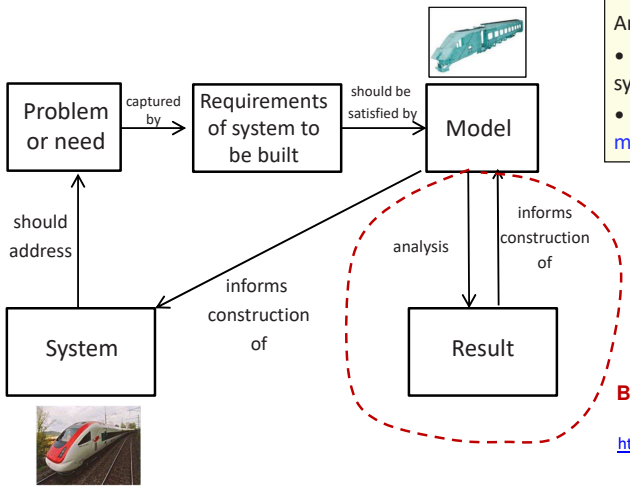
Query, Q: "gold silver truck"
D_1: "Shipment of gold damaged in a fire"
D_2: "Delivery of silver arrived in a silver truck"
D_3: "Shipment of gold arrived in a truck"
D = 3; IDF = log(D/df$_i$)

	Counts, tf_i						Weights, $w_i = tf_i * IDF_i$			
Terms	Q	D_1	D_2	D_3	df_i	D/df_i	IDF_i	Q	D_1	D_2	D_3
a	0	1	1	1	3	3/3 = 1	0	0	0	0	0
arrived	0	0	1	1	2	3/2 = 1.5	0.1761	0	0	0.1761	0.1761
damaged	0	1	0	0	1	3/1 = 3	0.4771	0	0.4771	0	0
delivery	0	0	1	0	1	3/1 = 3	0.4771	0	0	0.4771	0
fire	0	1	0	0	1	3/1 = 3	0.4771	0	0.4771	0	0
gold	1	1	0	1	2	3/2 = 1.5	0.1761	0.1761	0.1761	0	0.1761
in	0	1	1	1	3	3/3 = 1	0	0	0	0	0
of	0	1	1	1	3	3/3 = 1	0	0	0	0	0
silver	1	0	2	0	1	3/1 = 3	0.4771	0.4771	0	0.9542	0
shipment	0	1	0	1	2	3/2 = 1.5	0.1761	0	0.1761	0	0.1761
truck	1	0	1	1	2	3/2 = 1.5	0.1761	0.1761	0	0.1761	0.1761

http://www.miislita.com/term-vector/term-vector-3.html

Models in Engineering

Mostly prescriptive, facilitate construction

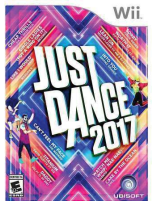


Benefits:
Analyses on model
- easier, cheaper than on system
- facilitate early decision making

typically codified as rules & techniques in textbooks & tools

E.g., PennDOT's **B**ridge **A**utomated **D**esign and **D**rafting Software
http://bradd.engrprograms.com

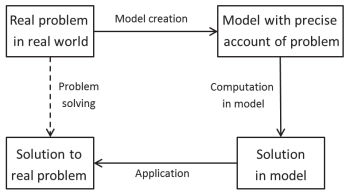
Models in Entertainment

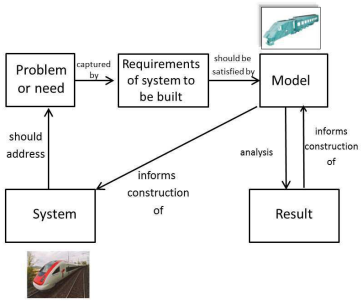


- Appeal comes from both
  - being like reality, and
  - not being like reality

Activities on Models

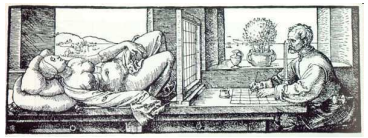
- Creation
- Validation
- Verification
- Analysis
- Integration





Activities on Models: Creation

- **Mostly manual**
  - can be very, very difficult
    - may require supporting tools, techniques, etc



A. Duerer
1471 - 1528

    - supporting techniques/methods
      - c.f., "*scientific method*" [1]
        - New knowledge is the result of sequences of observations, hypotheses, predictions, and experiments

Model of how scientific models are created

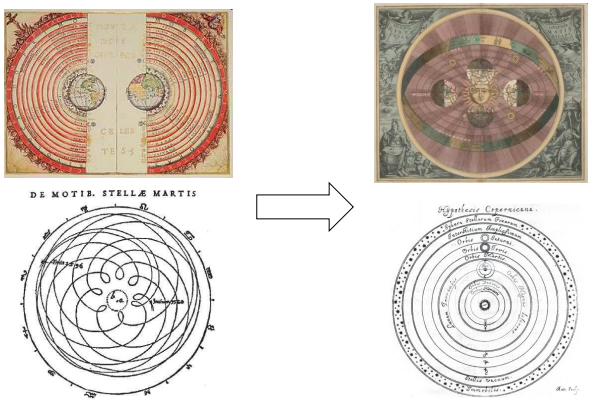
    - or even a "*revolution*" [Kuh62]
  - easier cases can be automatic (e.g., model extraction for reverse engineering)

[1] http://en.wikipedia.org/wiki/Scientific_method

[Kuh62]: Th. Kuhn. The Structure of Scientific Revolutions. 1962

Activities on Models: Creation (Cont'd)

- Can get you into prison
  - See, Galileo Gallilei (1564 – 1642)



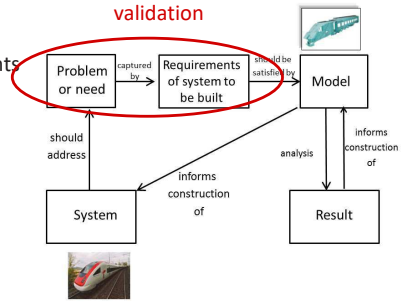
Activities on Models: Validation

- In forward engineering:

"Are we building the right product?"

relative to what the user wants

- E.g., in requirements engineering:
  - Validation checks that requirements are correct in the first place and system meets user's needs
  - Techniques used:
    - inspections,
    - testing,
    - prototyping



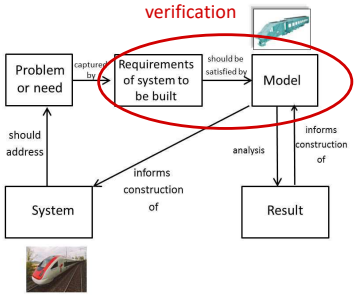
Activities on Models: Verification

- In forward engineering:

"Are we building the product right?"

relative to what, e.g., some specification, standard, or guideline says

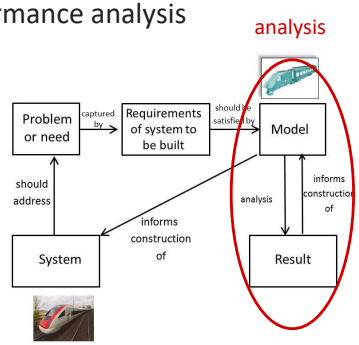
- E.g., in requirements engineering:
  - is system correct wrt to requirements?
  - techniques used:
    - safety analysis (check for system "hazards"/failures)
    - testing (if model executable, tractable)
    - formal verification



Activities on Models: Analysis

"Which properties does model have?"

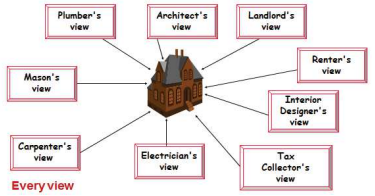
- E.g., structure, behaviour or performance analysis

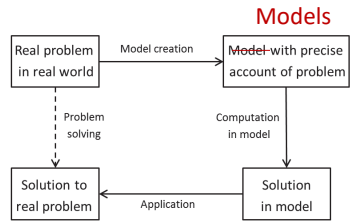


Activities on Models: Integration

Combine different views

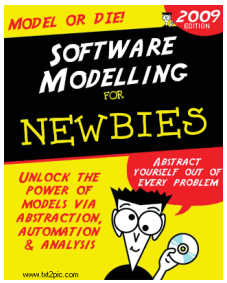
- May be hard, because
  - models to be integrated may
    - be expressed in different or conflicting notations or assumptions
  - Integrated model may be too large
- **Example:**
  - UML models
  - Coupled earth system model [EJ09]





[EJ09] S.M. Easterbrook, T.C. Johns. Engineering the Software for Understanding Climate Change. Submitted

CISC836: Models in Software Development: Methods, Techniques and Tools



Topic 2: Models in Software Development

Juergen Dingel
Jan, 2020

Overview

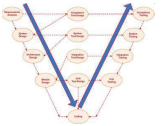
- What is being modeled?
- Why?
- How? I.e., what is a good model?
- Models as primary artifact
  - Model-Driven development (MDD)
  - Examples
    - EGGG
    - IBM RSARTE, Rhapsody
    - EMF
    - Unity

Observations

- Models in software engineering (SE) can be both **prescriptive** (e.g., requirements models) or **descriptive**:
  - e.g., models created via analyses performed on source code for program understanding, reverse engineering, impact analysis, test case generation, etc
- Creation, adoption, analysis of models
  - in SE not as well-understood as in other engineering disciplines
- MDD (and also this course) focuses more on prescriptive models
  - Following courses are more about automatic creation and use of descriptive models of code
    - ELEC 875: Design Recovery and Automated Evolution (Tom Dean)
    - ELEC 876: Software Reengineering (Jenny Zou)

What is being Modeled?

- Almost any artifact created/used during development:
  - Models of structure
    - data, execution state (object graphs), of languages (e.g., grammars and meta models), variability models
  - Models of software behaviour
    - on different levels: requirements, design, architecture, code
    - different parts: individual operations, methods, interactions (b/w processes, user and interface, or system and environment)
    - faults, performance, processes, workflows, environment, test models, variability models (for software product lines)
  - Models of software development processes

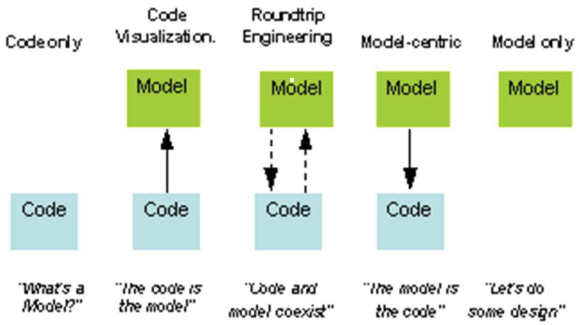


Purpose: Dealing with Complexity

- *"build models, because cannot comprehend system in its entirety"* [BRJ05, p6-7]
- **Essential complexity** [GS04,p36]
  - "inherent to problem being solved", "number of features, relationships, and dependencies that must be considered when decomposing problem"
  - cannot be reduced or eliminated
  - **Example**: online ordering process handled by single company vs online ordering process handled by multiple companies
- **Accidental complexity** [GS04,p36]
  - "an artifact of solution", "number of features, relationships, and dependencies that must be considered when composing solution"
  - can be reduced
  - **Examples**:
    - online commerce application written in assembly language vs one written in Java, C# etc
    - platform (e.g., J2EE, .NET, Corba) complexity [Sch06]

Purpose (Cont'd)

- Documentation (e.g., for design decisions), visualization, specification, design templates, understanding/analysis, round-tripping, code generation



Purpose (Cont'd)

- Execution and analysis
  - **checking static (i.e., structural) properties**
    - e.g., Alloy [Jac02]
  - **checking dynamic (i.e., behavirourl) properties**
    - e.g., model checking [BA10, Eme08]
      - component/process interaction [KMR02, MC01]
      - navigation in web applications [HEBB10]
    - performance analysis [Pet09, HLT+04]
  - **test case generation**
    - for embedded (automotive) systems [SYR08]
    - incrementally for software product lines [UGKB08]
  - **"what if" analyses [Sch06]**

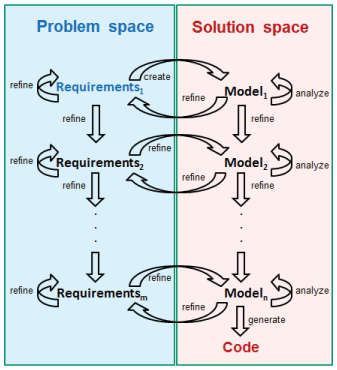
Characteristics of Good Models [Sel03]

1. **Abstraction**
   - irrelevant detail removed to deal with complexity
   - *"Make everything as simple as possible, but no simpler"*
2. **Understandability**
   - effort required for understanding (via, e.g., intuitive notation)
3. **Accuracy**
   - true-to-life representation of the modeled system's features of interest
4. **Predictiveness**
   - be able to use model to predict model's non-obvious properties
5. **Inexpensive**
   - model must be cheaper to construct than the modeled system

[Sel03] B. Selic. The Pragmatics of Model-Driven Development. IEEE Software 20(5):19-25, Sep/Oct '03.

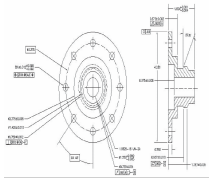
Model-Driven Development (MDD)

- **Core ideas [Sel03][KSLB03][Sch06]**
  - Abstraction and automation [Sel03]
    - Automation, e.g., for model transformation, code generation, analysis
    - performance of current code generators within 5-15% of manually implemented systems
  - Iterative development and successive refinement via automated model transformations
  - Analysis



Model-Driven Development (MDD) (Cont'd)

- **The potential power of these ideas**
  - **MDSD = CAD/CAM for software**


  - **[Sel03]: software engineering medium best positioned to benefit from modeling, b/c**
    - model can gradually evolve into final product w/o the risks that "discontinuities", e.g., in material, bring
    - conducive to incremental iterative development, b/c no discontinuities that preclude backtracking

How well does MDSD currently realize this vision?

Who is Practising MBSE?

IBM Rational

Sampling of Embedded Software Developed Using MDD

Automated doors, Base Station, Billing (In Telephone Switches), Broadband Access, Gateway, Camera, Car Audio, Convertible roof controller, Control Systems, DSL, Elevators, Embedded Control, GPS, Engine Monitoring, Entertainment, Fault Management, Military Data/Voice Communications, Missile Systems, Executable Architecture (Simulation), DNA Sequencing, Industrial Laser Control, Karaoke, Media Gateway, Modeling Of Software Architectures, Medical Devices, Military And Aerospace, Mobile Phone (GSM/3G), Modem, Automated Concrete Mixing Factory, Private Branch Exchange (PBX), Operations And Maintenance, Optical Switching, Industrial Robot, Phone, Radio Network Controller, Routing, Operational Logic, Security and fire monitoring systems, Surgical Robot, Surveillance Systems, Testing And Instrumentation Equipment, Train Control, Train to Signal box Communications, Voice Over IP, Wafer Processing, Wireless Phone

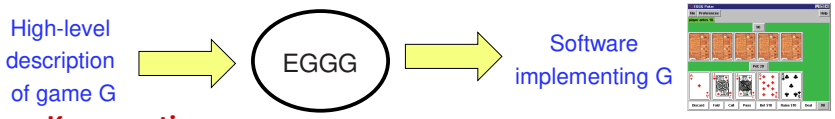
16 ON DEMAND BUSINESS

Model-Driven Software Development (MDSD): Examples

- **Industrial: E.g.,**
  - Motorola [BLW05], OCE [DS02], EADS (Scade), [ADSAA04], GM, avionics, automotive
- **Academic: E.g.,**
  - EGGG [Orw00]
  - Feature-oriented programming [BD07,BSR04,BG97]
  - Software Factories (MS) [GS04]
  - Intensional programming [Sim01, ADKdMRS98]
  - Language-oriented programming [MPS09, LOP09]
  - Language workbench [Fow09]
- **Supporting commercial tools: E.g.,**
  - IBM Rational: RSARTE (RoseRT), Rhapsody
  - The MathWorks: MATLAB StateFlow/Simulink
  - MetaCase: MetaEdit
  - No Magic: Cameo Systems Modeler
  - MS Visual Studio [GS04]

MDSD Example: EGGG

- **EGGG: Extensible Graphical Game Generator**
- Jon Orwant's PhD 1999
- Exploit commonalities among games to write software that automatically generates the code for a game

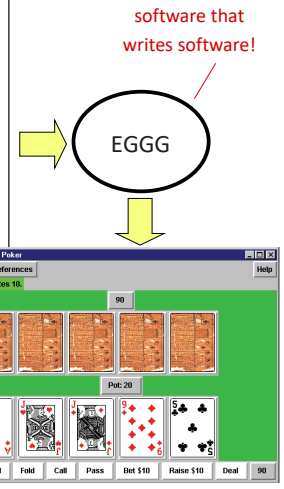


- **Key questions:**
  - Which games can be supported? How to best describe a game? How write the code generator?
- Domain analysis results in identification of concepts that allow description of certain class of games in terms of models represented in a domain-specific language (DSL)

MDSD Example: EGGG (Cont'd)

```
game is poker

turns alternate clockwise

Discard means player removes 0..3 cards or 4 cards if Ace
Fold means player loses

2..6 players

game is Shuffle(deck) and Deal(cards, 5) and (bet(money) or Fold)
        and Discard(hand, N) and Deal(cards, 5-N)
        and (bet(money) or Fold) and compare(cards)

StraightFlush is (R, S) and (R-1, S) and (R-2, S) and (R-3, S) and (R-4, S)
FourKind is (R, s) and (R, s) and (R, s) and (R, s)
FullHouse is (R, s) and (R, s) and (R, s) and (Q, s) and (Q, s)
Flush is (r, S) and (r, S) and (r, S) and (r, S) and (r, S)
Straight is (R, s) and (R-1, s) and (R-2, s) and (R-3, s) and (R-4, s)
ThreeKind is (R, s) and (R, s) and (R, s)
TwoPair is (R, s) and (R, s) and (Q, s) and (Q, s)
Pair is (R, s) and (R, s)
HighCard is (R, s)

hands are [StraightFlush, FourKind, FullHouse, Flush, Straight,
           ThreeKind, TwoPair, Pair, HighCard]

hand is five cards
goal is highest(hand)
```

Domain-specific language (DSL)



MDSD Example: EGGG (Cont'd)

- **What makes this work?**
  - Thorough domain analysis
    - identify communalities & differences, relevant concepts, parameters & their ranges, rules & principles, patterns & anti-patterns, classification, scope of application
      ⇒ develop suitable meta model
    - develop code generator
    - develop user-friendly way allowing users to describe an element within the scope (i.e., a model)
      ⇒ find suitable concrete syntax

abstraction

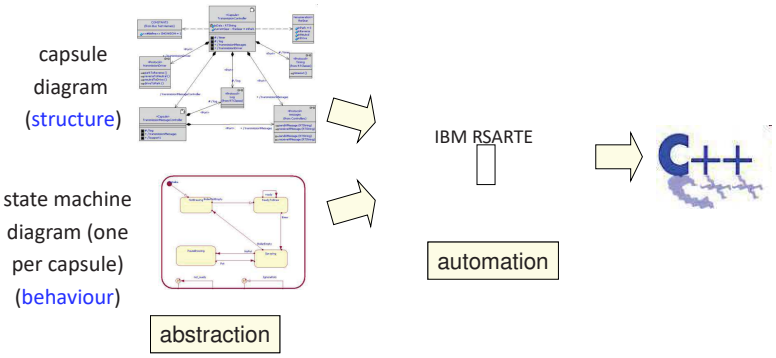
automation

- **Consequences**
  - Deep understanding required
  - Higher upfront cost, but once in place, development much simplified

> *"We need to know what we are doing before we can automate it. A DSM solution is implausible when building an application or a feature unlike anything developed earlier".*
>
> *"Early adopters of domain-specific modeling have been enjoying productivity increases of 500-1000% in production for over 10 years now"*
>
> [KT08, p18]

MDSD Example: IBM RSARTE

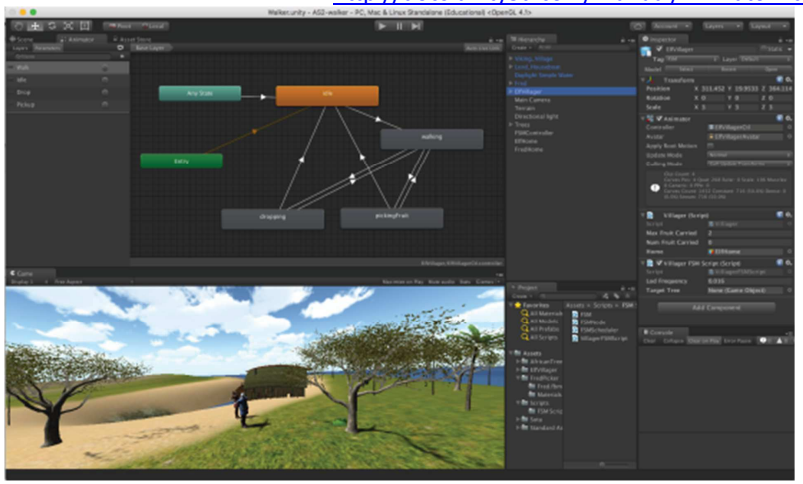
- IBM Rational Rhapsody, RoseRT



- UML-RT as DSL (UML variant) [Sel98]
  - for timed, reactive systems

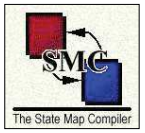
MDSD Example: Game Industry

http://docs.unity3d.com/Manual/Animator.html



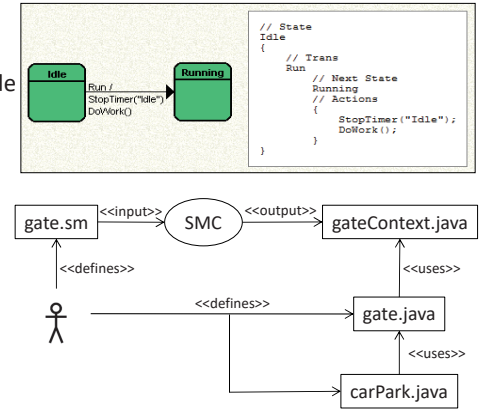
Screenshot courtesy Nick Graham

SMC: The State Machine Compiler



- SMC allows
  - textual definition of state machine

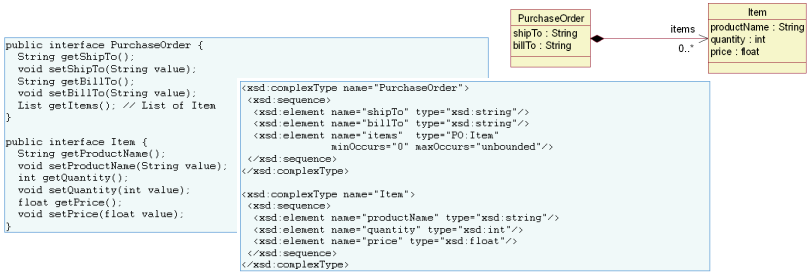
  automatic generation of code implementing that state machine
  - visualization of state machine
- Supports 15 languages
- Generated code uses State Design Pattern



http://smc.sourceforge.net/

MDSD Example: EMF [Bud05]

- **Every program manipulates data**
  - defined using e.g., Java, XML, database schema, or UML class diagram



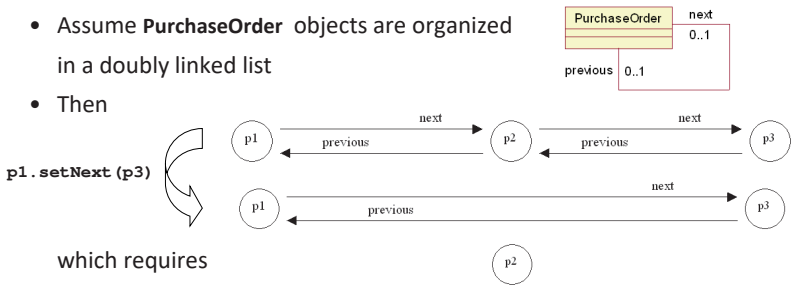
- **Advantages of identifying data model**
  - communicate design and share data with other applications
  - may be able to generate some, if not all, implementation code

MDSD Example: EMF [Bud05] (Cont'd)

- **Eclipse Modeling Framework (EMF)**
  - describe data model using
    1. Ecore (meta data description language similar to UML class diagrams)
    2. UML
    3. Java interfaces
    4. XML schema
  - use EMF to generate
    - data model in other formats
      - including Java code
      - code generator can be extended and modified
    - tree-based editor that allows you to input instances of the model

MDSD Example: EMF [Bud05] (Cont'd)

- Generated code may be quite sophisticated
- **Example:**
  - Assume **PurchaseOrder** objects are organized in a doubly linked list
  - Then



which requires

Remove *p1.next* pointer to *p2*.
Remove *p2.previous* pointer to *p1*.
Set *p1.next* pointer to *p3*.
Remove *p3.previous* pointer to *p2*.
Remove *p2.next* pointer to *p3*.
Set *p3.previous* pointer to *p1*.
Send notifications (changes to *p1, p2,* and *p3*).

```
public void setNext(PurchaseOrder newNext) {
  if (newNext != next) {
    NotificationChain msgs = null;
    if (next != null)
      msgs = ((InternalEObject)next).eInverseRemove(this,
        POPackage.PURCHASE_ORDER__PREVIOUS, PurchaseOrder.class, msgs);
    if (newNext != null)
      msgs = ((InternalEObject)newNext).eInverseAdd(this,
        POPackage.PURCHASE_ORDER__PREVIOUS, PurchaseOrder.class, msgs);
    msgs = basicSetNext(newNext, msgs);
    if (msgs != null) msgs.dispatch();
  }
  else if (eNotificationRequired())
    eNotify(new ENotificationImpl(this, Notification.SET,
      POPackage.PURCHASE_ORDER__NEXT, newNext, newNext)); // touch notification
}
```


XText

Xtext

- Eclipse-based open-source framework for development of programming languages and domain-specific languages
- Offers
  - Parser generator
  - Editor plugin generator supporting
    - Syntax highlighting
    - Well-formedness checking (validation) w/ error markers and quick fixes
    - Background parsing
    - Auto-completion with content assist
    - Hyperlinking connecting uses with declarations
    - Hovering
    - Folding and outline view
  - Support for
    - Code generation (using Xtend, a variant of Java)
    - Interpretation, translation to Java
  - Large user community, http://www.eclipse.org/Xtext/community.html

"A language is only as good as its supporting tooling"
[B. Selic]

Today's Lecture

- Topic 1: Observations about modeling in general
  - Examples
  - What is a model?
  - What are they used for?
  - Activities on models
- Topic 2: Intro to software modeling
  - What is being modeled?
  - Why?
  - How? I.e., what is a good model?
  - Models as primary artifact
    - Model-Driven Software Development (MDSD)
    - Examples: EGGG, IBM RoseRT, EMF

Next in CISC836

- Topic 3: Expressing software models
  - UML
  - UML-RT
- Topic 4: MDD with UML-RT
  - IBM RSA-RTE
- Topic 5: Using software models
  - Code generation with EMF
- Topic 6: DSLs