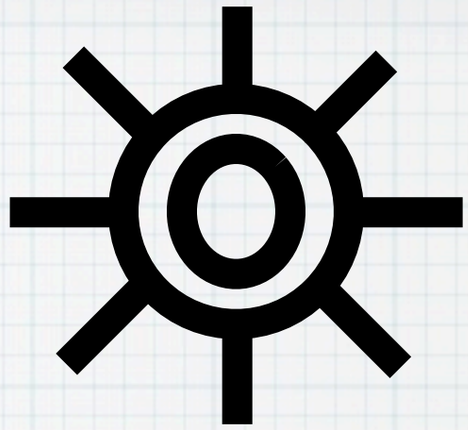
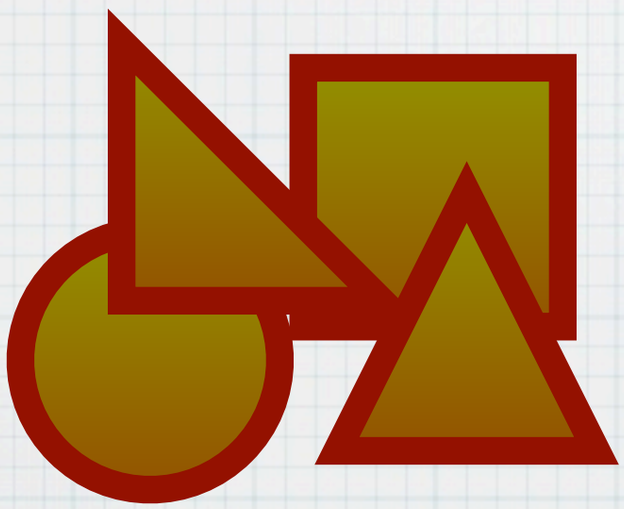


Object Modeling with UML 2 - Fundamentals

Les Waguespack, Ph.D.



Slides One

Copyright and References

* UML 2 and the Unified Process 2nd Ed - Practical Object-Oriented Analysis and Design, Arlow & Neustadt, Addison-Wesley / Pearson Education, Inc., Boston, MA, ISBN 0-321-32127-8
* UML 2 Toolkit, Eriksson, Penker, Lyons & Fado, Wiley, Indianapolis, IN, ISBN 0-471-46361-2
* UML 2.0 - Superstructure, Object Management Group, http://www.omg.org/cgi-bin/doc?formal/05-07-04
* Object Oriented Analysis, 2nd Ed, Peter Coad and Edward Yourdan, Prentice-Hall, 1991.ISBN 978-0136299813
* Business Modeling With UML, Eriksson & Penker, Wiley, Indianapolis, IN, ISBN 0-471-29551-5
* Enterprise Modeling With UML Designing Successful Software Through Business Analysis, Addison-Wesley, Reading, MA, ISBN 0-201-43313-3
* Use Case Modeling, Bittner & Spence, Addison-Wesley / Pearson Education, Inc., Boston, MA, ISBN 0-201-70913-9
* Writing Effective Use Cases, Cockburn, Addison-Wesley, Boston, MA, ISBN 0-201-70225-8
* Object Oriented Systems Engineering, Waguespack, course notes CS390, CS460, CS630, CS771, Computer Information Systems Department, Bentley College, Waltham, MA.

Agenda

* Overview
  * Course Description
  * UML 2 Certification
* Modeling
  * Identification, Description, Explanation, Communication
  * Scope and Focus
* Object Modeling
* Complexity Management

0. Overview

* Object-Oriented Modeling is based on a system of concepts that define the existence and relationships of facts within a defined system boundary.
* The system of concepts is called the Object-Oriented Paradigm.
* OOM is independent of UML or any other OO language: C ++, Java, Smalltalk, C#, ...
* The Object-Oriented Paradigm is stable, well understood and documented.
* UML is an evolving, growing tool attempting to address a growing and evolving industry of system development.

Course Description

This course introduces the object oriented paradigm as a basis for the standard documentation syntax of the Unified Modeling Language, UML, commensurate with the fundamental level of OMG UML certification. The course assumes no prior formal knowledge of the object oriented paradigm, UML or programming. Students will learn through course texts, lecture material, in-class and out-of-class object modeling exercises both individually and in groups. The primary focus is on the principles of object oriented modeling as documented in standard UML syntax. Students who successfully complete the course are skilled in the most commonly used elements of UML and able to develop basic object models. Student evaluation is accomplish through critical analysis of student modeling projects and continuous monitoring of learning in class. The course consists of five classroom sessions of four hours each.

OO Modeling with UML 2

- Objectives
  - Understanding the Object-Oriented Paradigm as a basis for documenting systems in UML 2
  - Knowledge of UML syntax commensurate with fundamental level of OMG UML 2 certification
  - Skill in developing basic object-oriented models using UML 2
- Pedagogy
  - Reading the course text: UML Distilled, 3rd, Fowler
  - Lectures on Theory and Practice of OOM and UML syntax
  - In class exercises in OOM with UML
  - **Post-Course re-reading and individual study for the UML 2 fundamental certification exam**

http://www.omg.org/uml-certification/exam_info.htm

OMG Certified  **UML Professional**

The Exams

There are three OCUP Exams - Fundamental, Intermediate and Advanced. Each Exam tests your knowledge of a different subset of the UML. Certification indicates the following abilities and qualifications.

Follow the links below for detailed information on each Exam.

Fundamental	You can work with the most commonly encountered UML elements You can create simple UML models You are qualified to be a member of a UML Development Team.
Intermediate	You can work with a broad range of UML elements You can create complex UML models You are qualified to be a senior member of a UML Development Team.
Advanced	You can work with the full range of UML elements You can create extremely large, complex UML models You are qualified to manage a UML Development Team.

OCUP Fundamental Exam

The Fundamental Exam covers a broad range of essential concepts and knowledge from the UML specification.

Exam Number:..... OM0-100

Duration:..... 90 minutes (80 questions)

Minimum Passing Score:..... 46

Exam Fee:..... US$200 (or equivalent in local currency)

Prerequisite:..... None

The Table of Contents of the UML 2.0 spec, color-coded to show which sections are on the Fundamental Exam - and more importantly, which are not - is here (PDF, 103 kb).

The following Coverage Table will help you to decide how much to study in various areas.

COVERAGE TABLE - FUNDAMENTAL

Class Diagrams (Basic)	30%
Activity diagrams (Basic)	20%
Interaction Diagrams (Basic)	20%
Use Case Diagrams (Basic)	20%
Miscellaneous basic notions	10%
Total-	100%

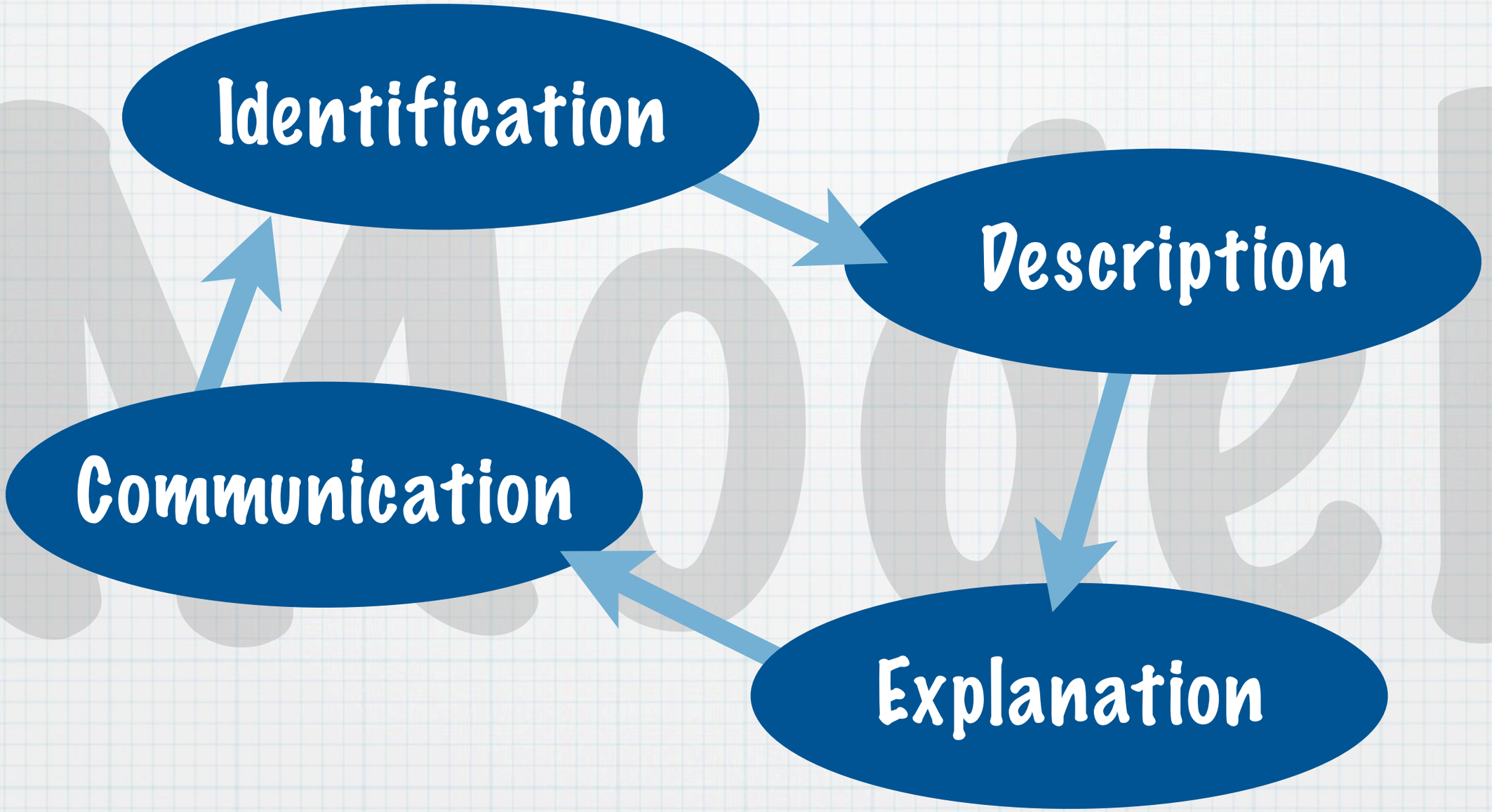
OnLine UML Resources

* OMG - UML 2.0 Superstructure
  * http://www.omg.org/cgi-bin/doc?formal/05-07-04
* Borland - tutorial
  * http://dn.codegear.com/article/31863
* Sparx - tutorials
  * http://www.sparxsystems.com.au/resources/uml2_tutorial/index.html

1. Modeling

* Modeling is the process of identifying facts and describing them by characterizing their identity and their (inter-) relationships to form an understanding; and then communicating that knowledge to others by explaining the relevance and workings of the system of concepts under study.

System Analysis and Modeling



overview

* Systems analysis is the process of understanding and acting upon that understanding
  * identification
  * description
  * explanation
  * communication
* Understanding is constructed from experience and analysis
  * abstraction
  * terminology
  * context
  * responsibility
  * clarity
  * fidelity
* The most effective tool for building understanding is the MODEL

*model

* n 1: a simplified description of a complex entity or process; "the computer program was based on a model of the circulatory and respiratory systems" [syn: theoretical account, framework] 2: a type of product; "his car was an old model" 3: a person who poses for a photographer or painter or sculptor; "the president didn't have time to be a model so the artist worked from photos" [syn: poser] 4: representation of something (sometimes on a smaller scale) [syn: simulation] 5: something to be imitated; "an exemplar of success"; "a model of clarity"; "he is the very model of a modern major general" [syn: exemplar, example, good example] 6: someone worthy of imitation; "every child needs a role model" [syn: role model] 7: a representative form or pattern; "I profited from his example" [syn: example] 8: a woman who wears clothes to display fashions; "she was too fat to be a mannequin" [syn: mannequin, manikin, mannikin, manakin, fashion model] 9: the act of representing something (usually on a smaller scale) [syn: modelling, modeling] v 1: plan or create according to a model or models [syn: pattern] 2: form in clay, wax, etc; "model a head with clay" [syn: mold, mould] 3: assume a posture as for artistic purposes; "We don't know the woman who posed for Leonardo so often" [syn: pose, sit, posture] 4: display (clothes) as a mannequin; "model the latest fashion" 5: create a representation or model of; "The pilots are trained in conditions simulating high-altitude flights" [syn: simulate] 6: construct a model of; "model an airplane" [syn: mock up]

exercises to explore modeling

* using a photograph of a tinker toy assembly
  * provide a poor quality visual image of the original assembly
  * have one student (guide) look at the picture and describe the assembly
  * have another student (builder) with back to the screen attempt to reconstruct the assembly with only written description
  * give the builder a complete set of tinker toy parts (more than required)
  * the builder may pause in the attempt by taking hands off
  * once the builder has put down the working pieces have another student (judge) indicate to the guide and builder whether the current attempted assembly is accurate or not by only saying "consistent" or "inconsistent"
  * repeat the sequence of attempts and reports until successful
* repeat the exercise with new assembly and new students and these changes
  * provide a high quality visual image of the original assembly with multiple perspectives
  * provide a visual glossary defining the names of the tinker toy parts
  * make the guide write down on paper the assembly's description
  * allow the guide to indicate consistency as frequently as desired

Problem Solving and Modeling

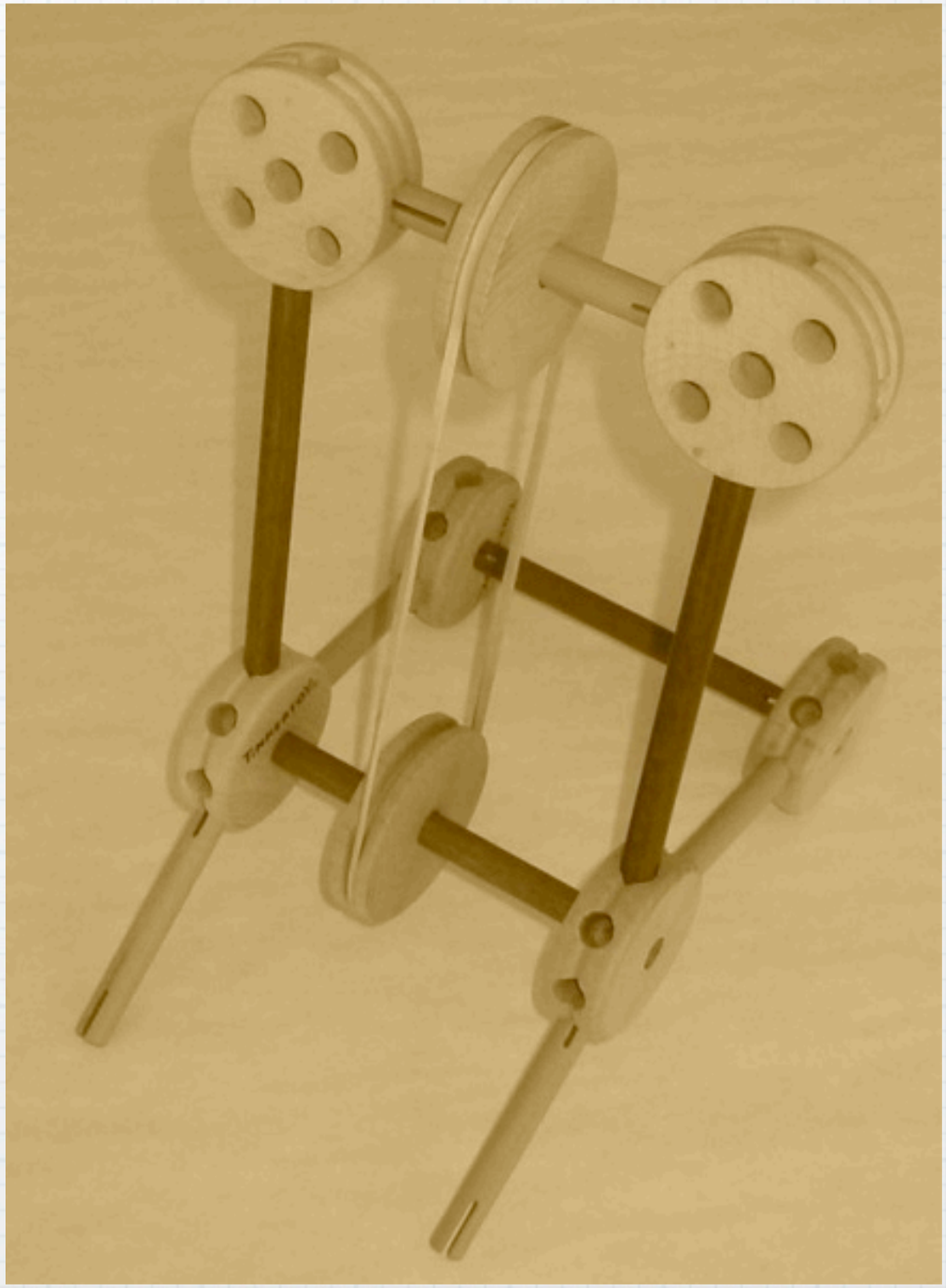
- Setting
  - Three volunteers for each exercise
    - Guide - provides a description of the assembly to be reconstructed
    - Builder - performs the construction of the new assembly
    - Judge - determines if the construction so far is consistent with the source
- Process
  - A pre-existing assembly is shown on the screen
  - The Builder sits with back to the screen unable to see the assembly
  - The Guide describes the assembly to the Builder
  - The Builder reconstructs the screen image with real parts
  - The Judge informs the Builder whether the work is consistent or not

What modelers try to do. . .

* identify as many characteristics of modeling as possible using the exercises with the tinker toys that follow . . .

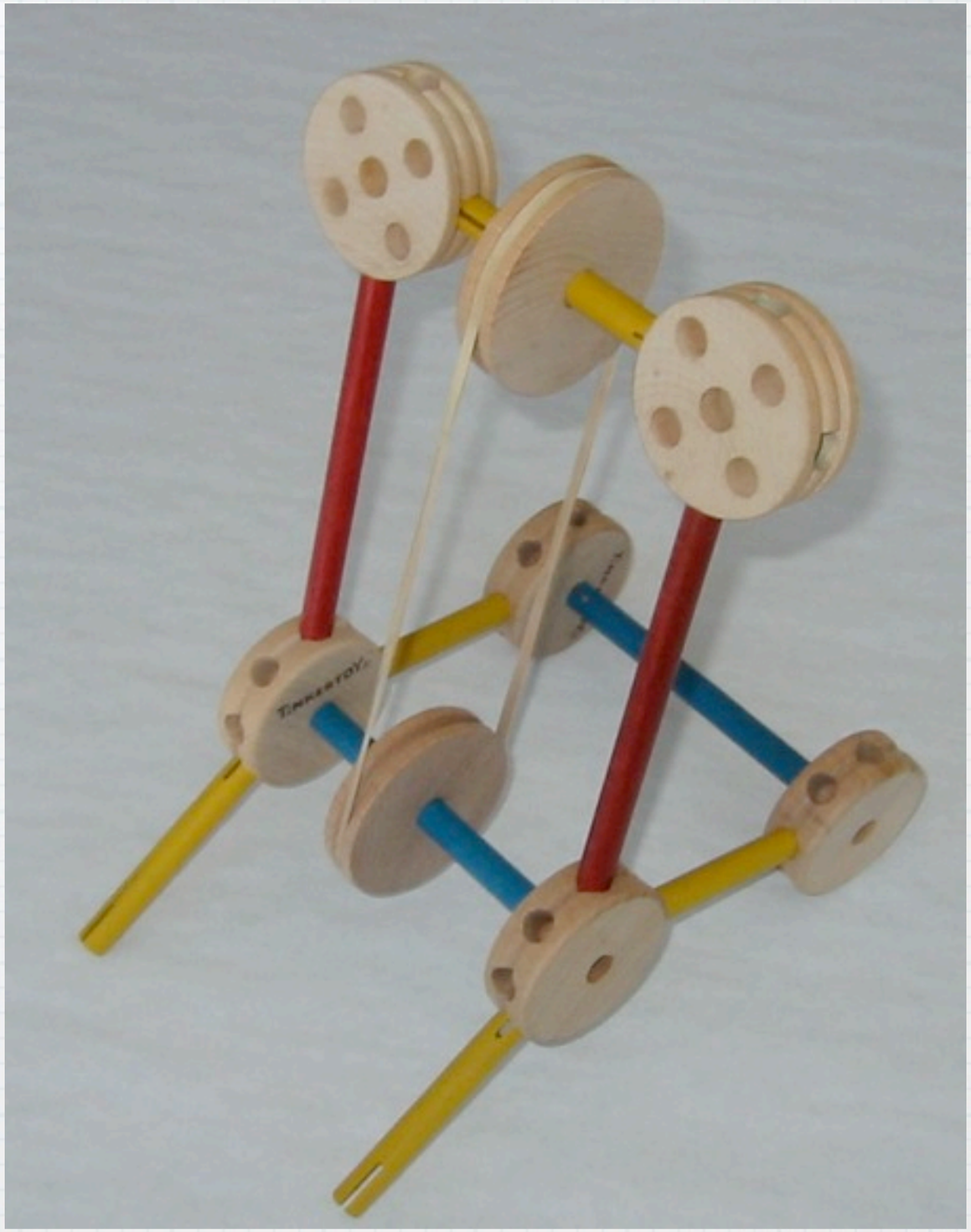
Experiment One

- Process Instructions
  - Guide writes description on the board (unseen by Builder)
  - Judge reads the instructions exactly
- Whenever Builder stops work putting down any parts the Judge may indicate either
  - Consistent
  - In-consistent
- Class times the attempt
- Observations . . .

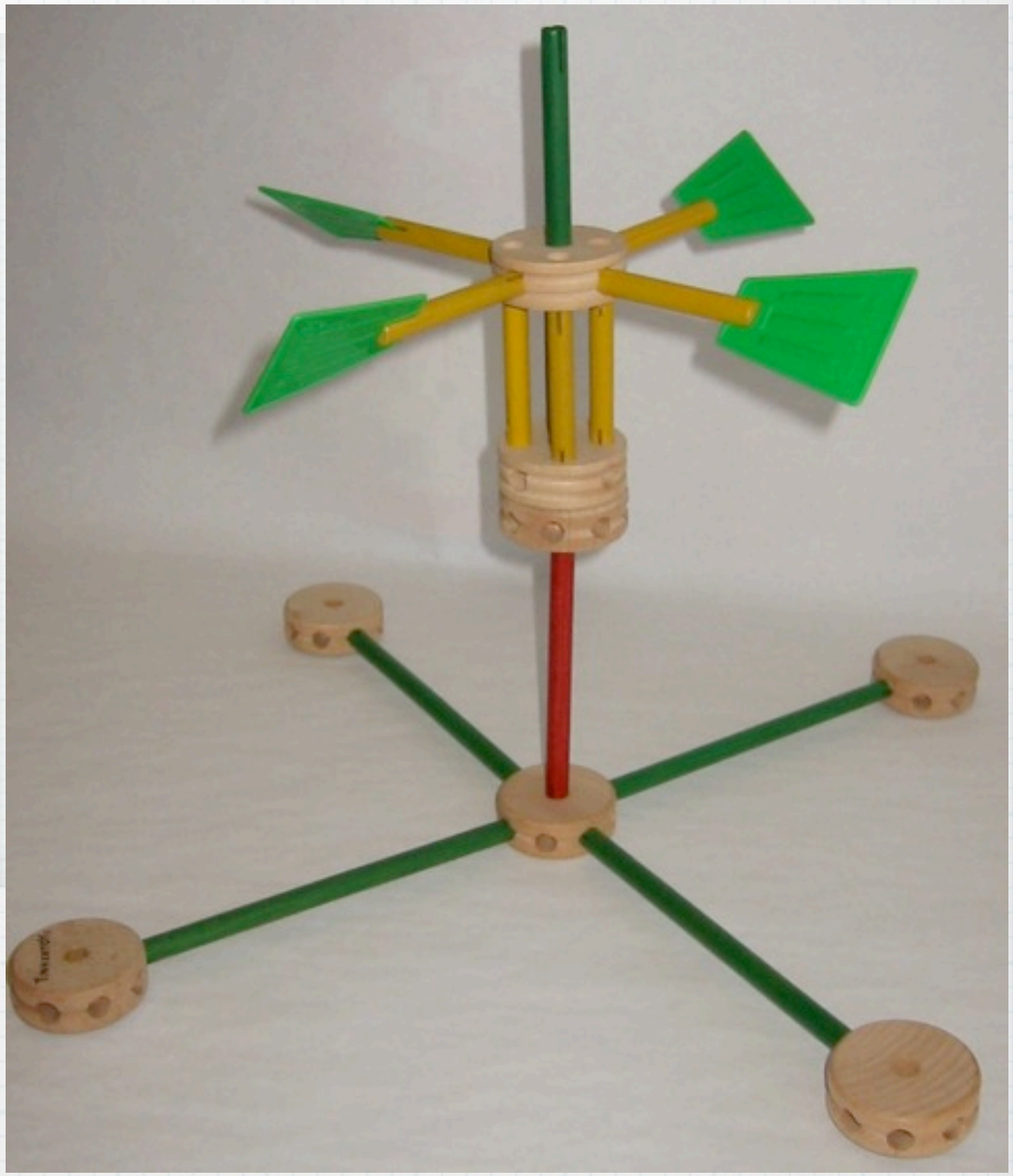


Experiment Two

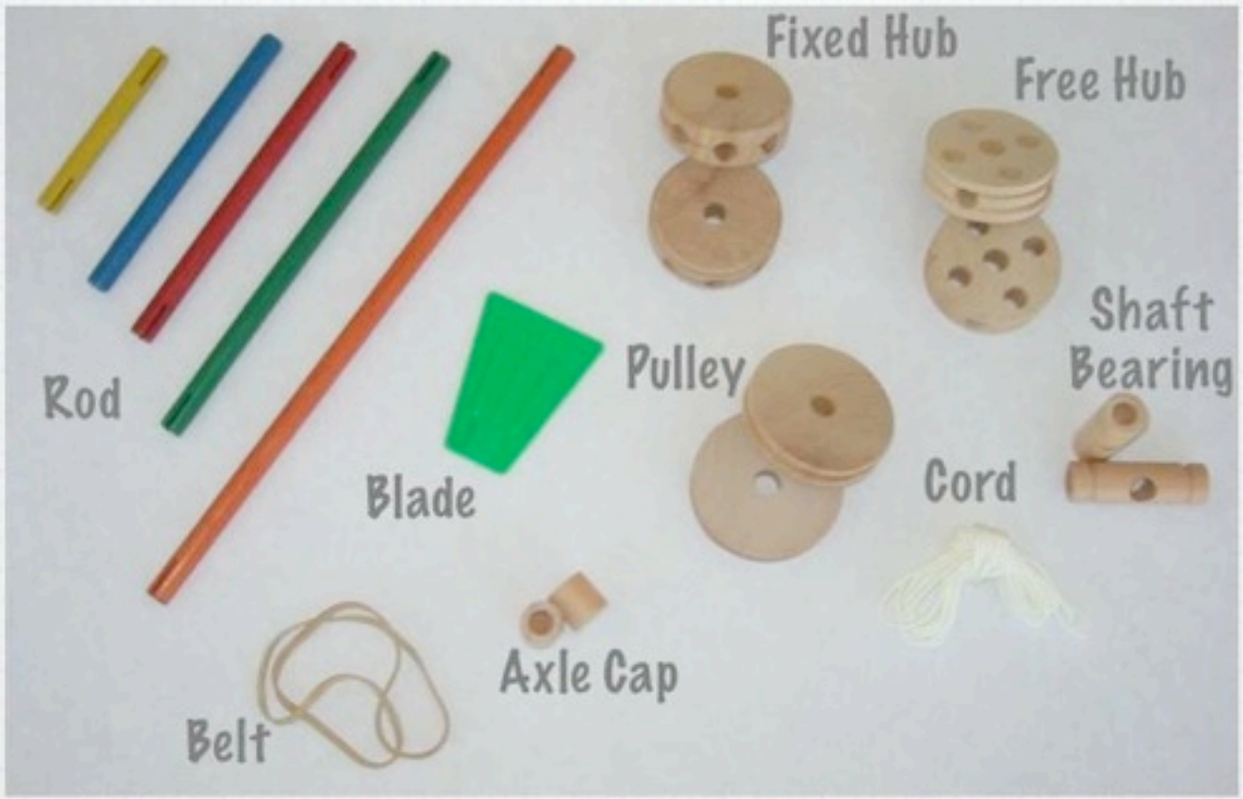
- Process Instructions
  - Guide describes the screen image verbally without watching the Builder's work
  - Whenever Builder stops work putting down any parts the Judge may indicate either
    - Consistent
    - In-consistent
- Class times the attempt
- Observations . . .

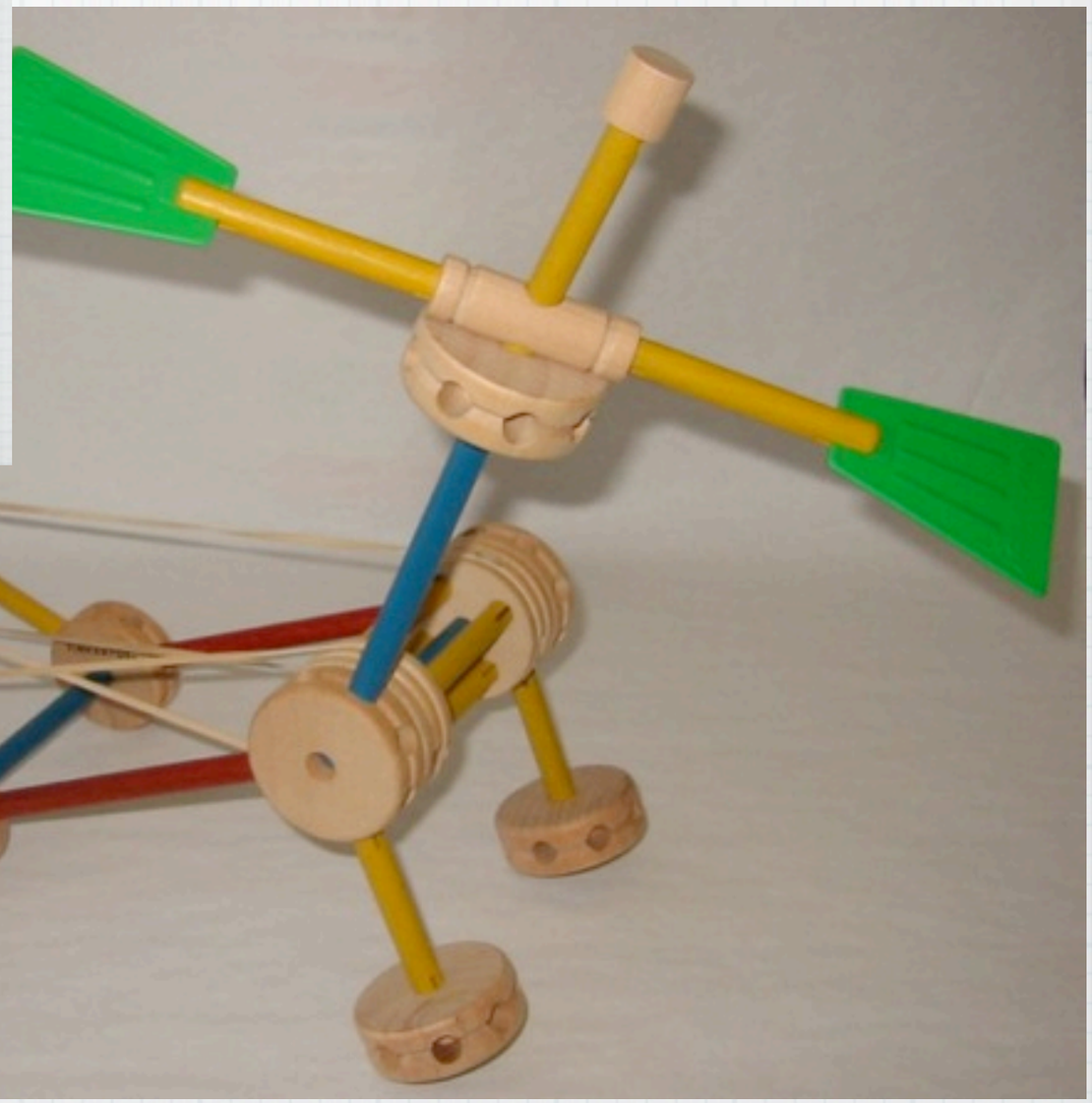


Experiment Three

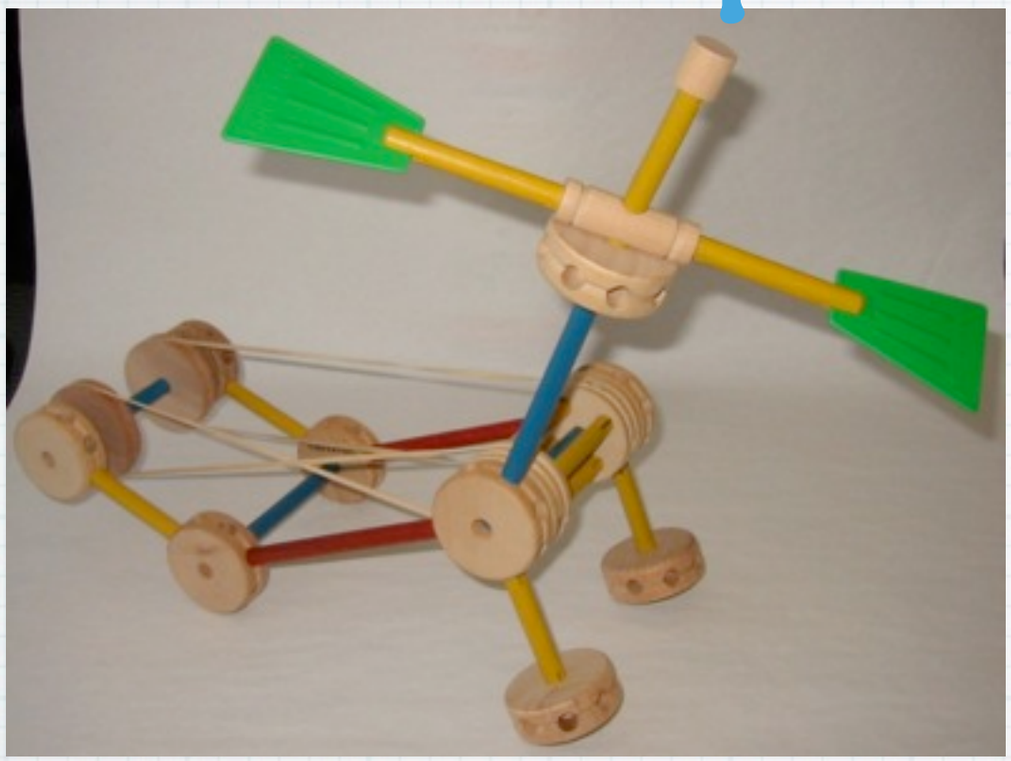


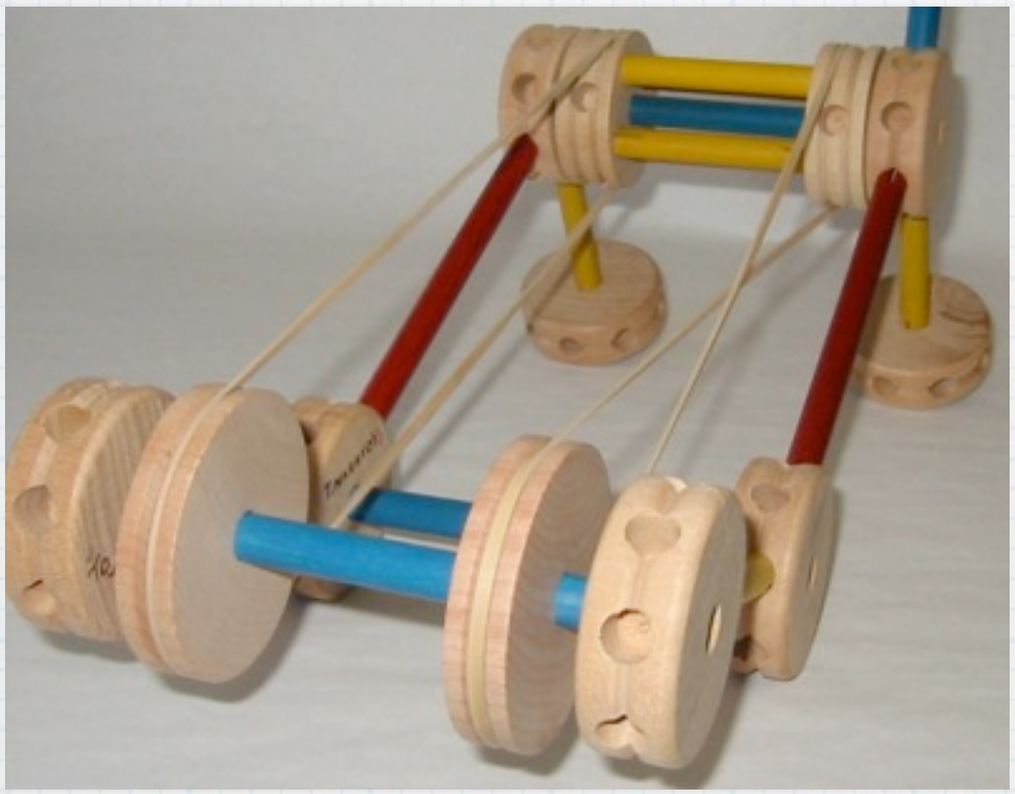
Experiment Four

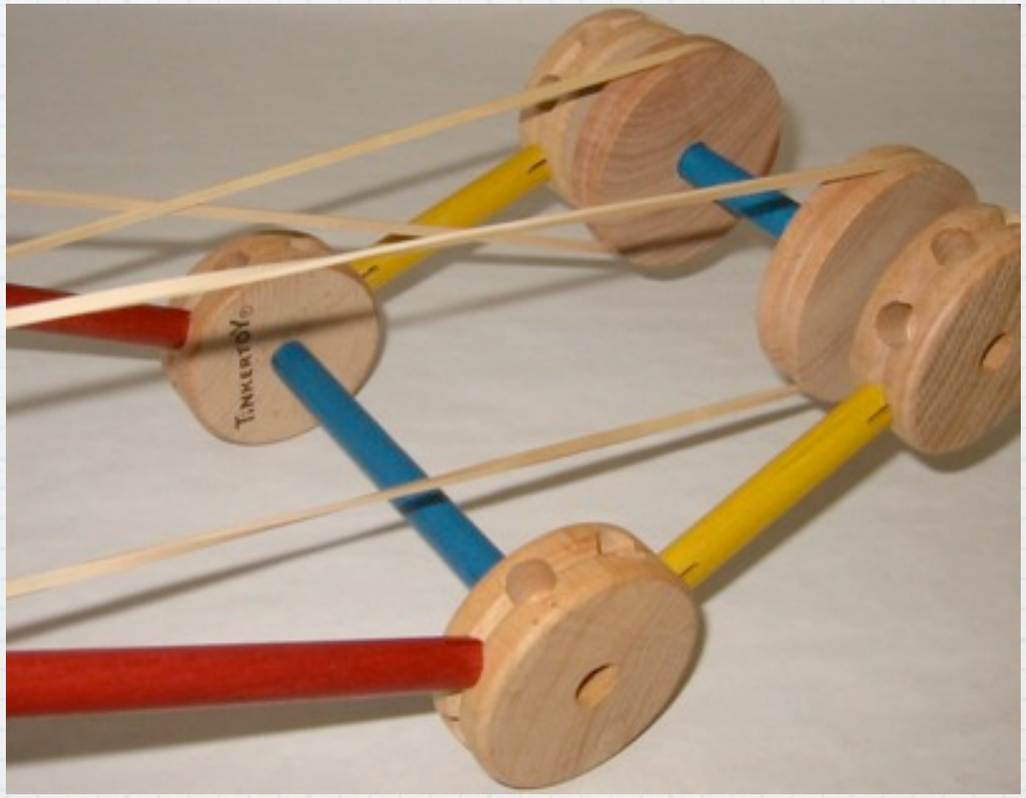




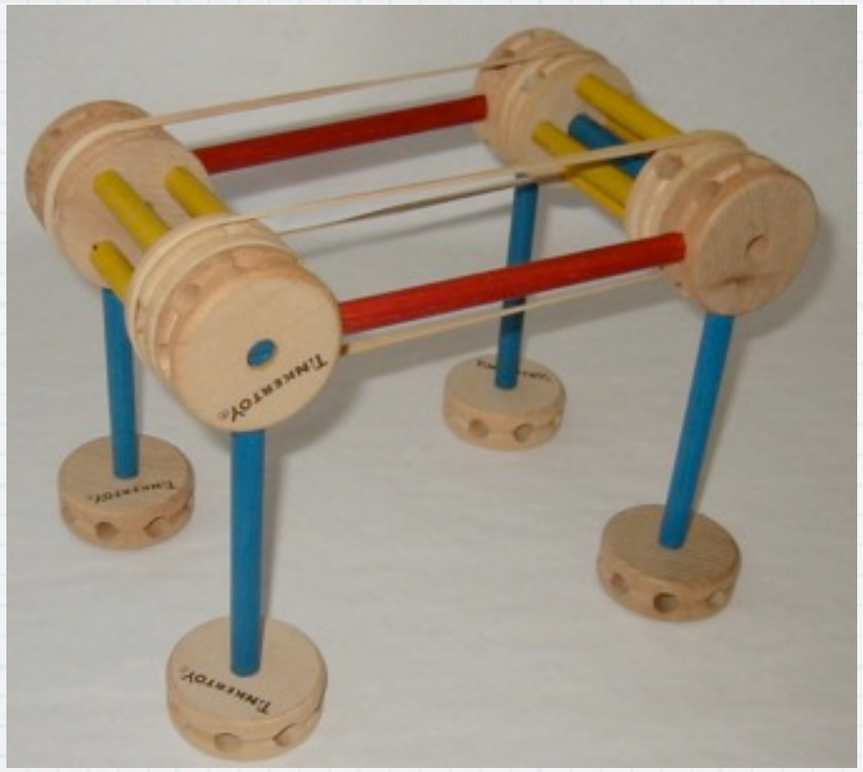
Experiment Four

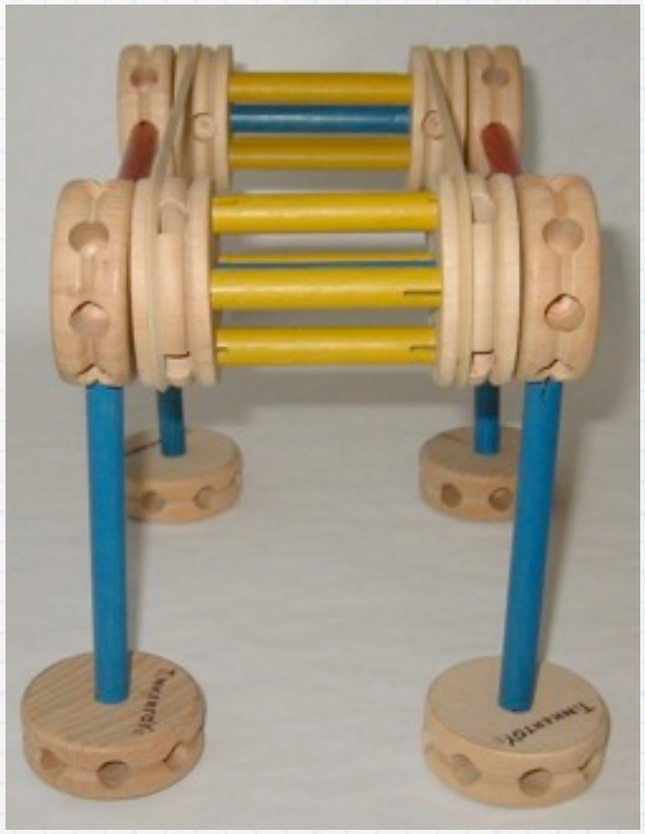


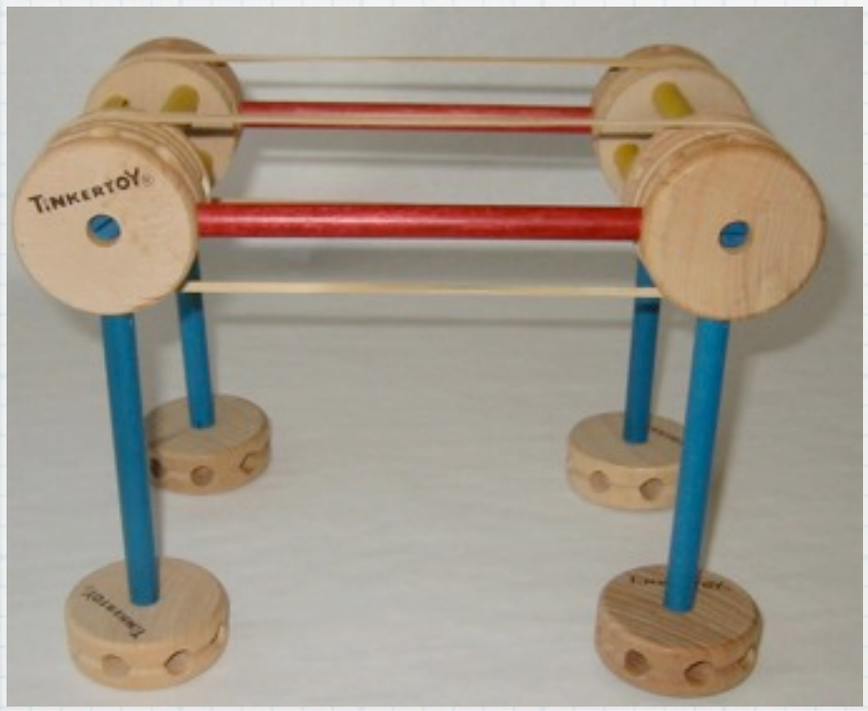




Experiment Five







What modelers try to do. . .

- identify as many characteristics of modeling as possible using the exercises with the tinker toys above . . .
  - domain knowledge
    - experience
      - history
      - exploration
    - vocabulary
    - quality control
  - communication
    - medium
      - written
      - oral
    - accuracy
      - perspective
      - quality control
    - clarity
      - terminology
      - consistency
      - relevance

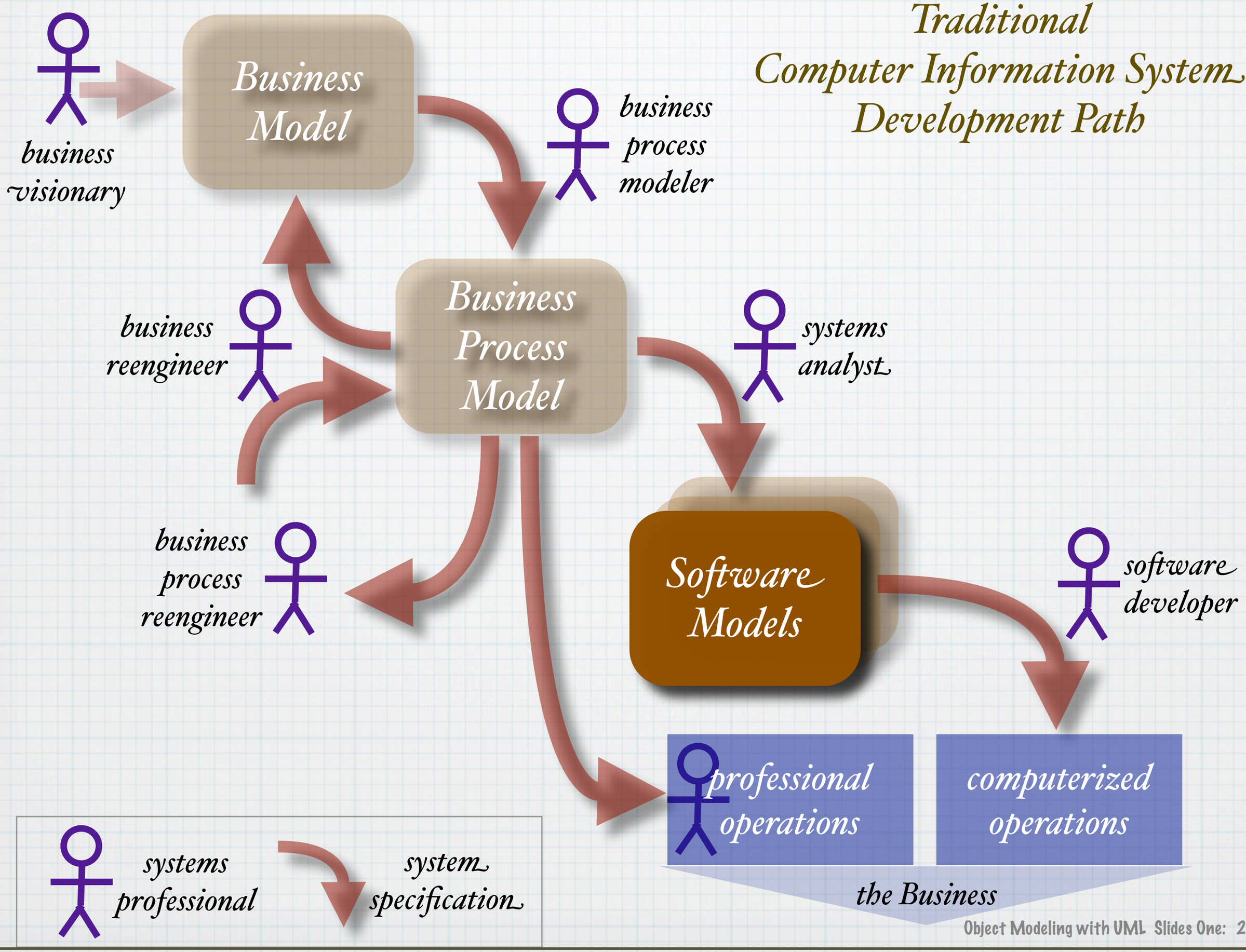
How modelers prepare to do it...

* domain knowledge
* experience
  * history - formal education, work experience
  * exploration - experimentation, research
  * vocabulary - terminology, paradigms, methodologies
  * quality control - testing, work reviews, client interaction
* communication
  * medium - analysts spend most of their time communicating
    * written - documentation becomes long term memory, resource, knowledge asset
    * oral - builds confidence and trust between analysts and with client
* accuracy
  * perspective - "who is right?" often depends upon the viewing angle
  * quality control - "Assuming only makes an A__ out of You and Me!"
* clarity
  * terminology - we all need to speak the same language, maybe a new one for us!
* consistency - the parts need to fit from every angle, over and over again
* relevance - the world is a big place, how much of it is needed in this problem?

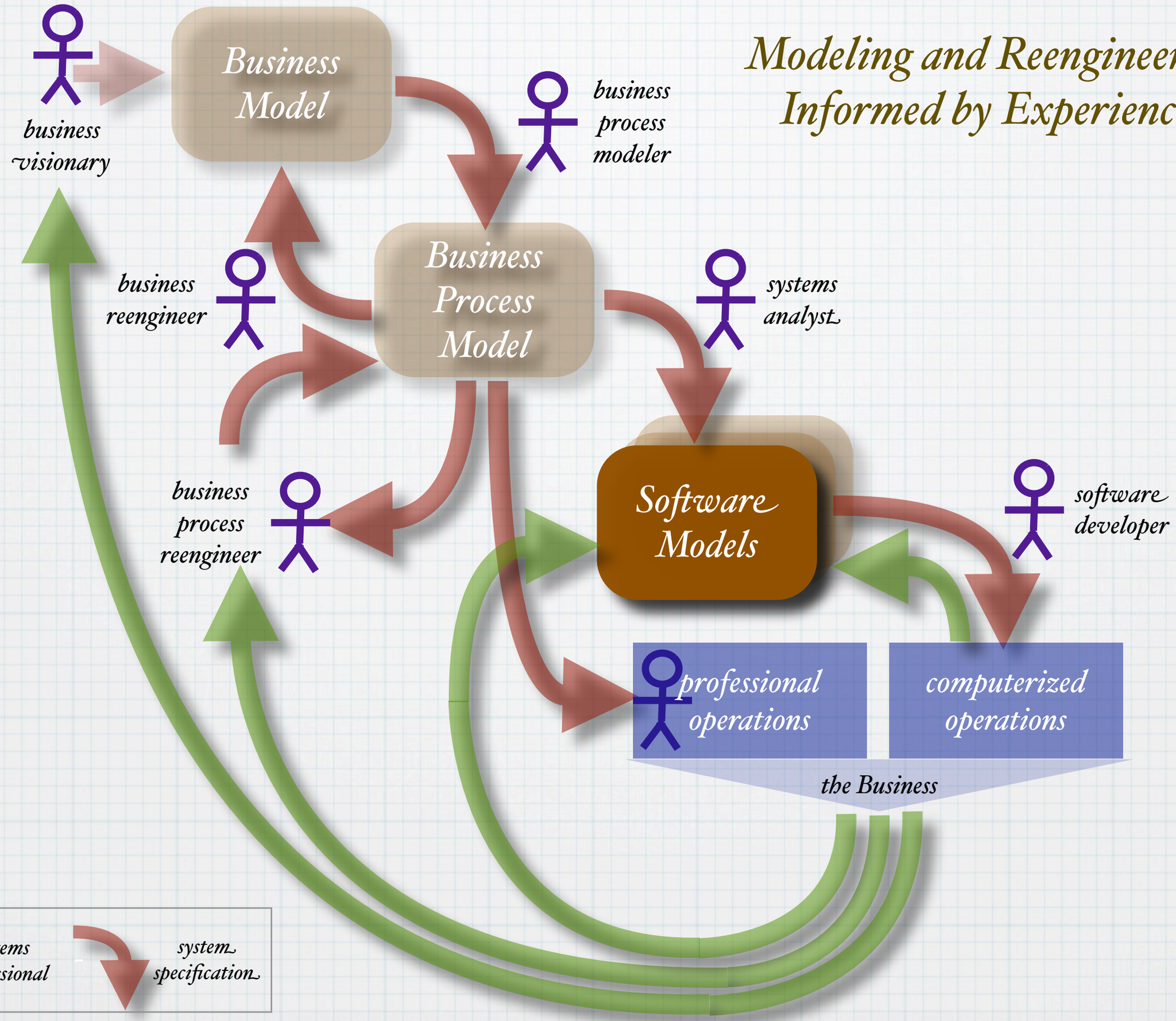
Scope & Focus

* "Seeing the forest for the Trees"
* "Showing the forest for the Trees"

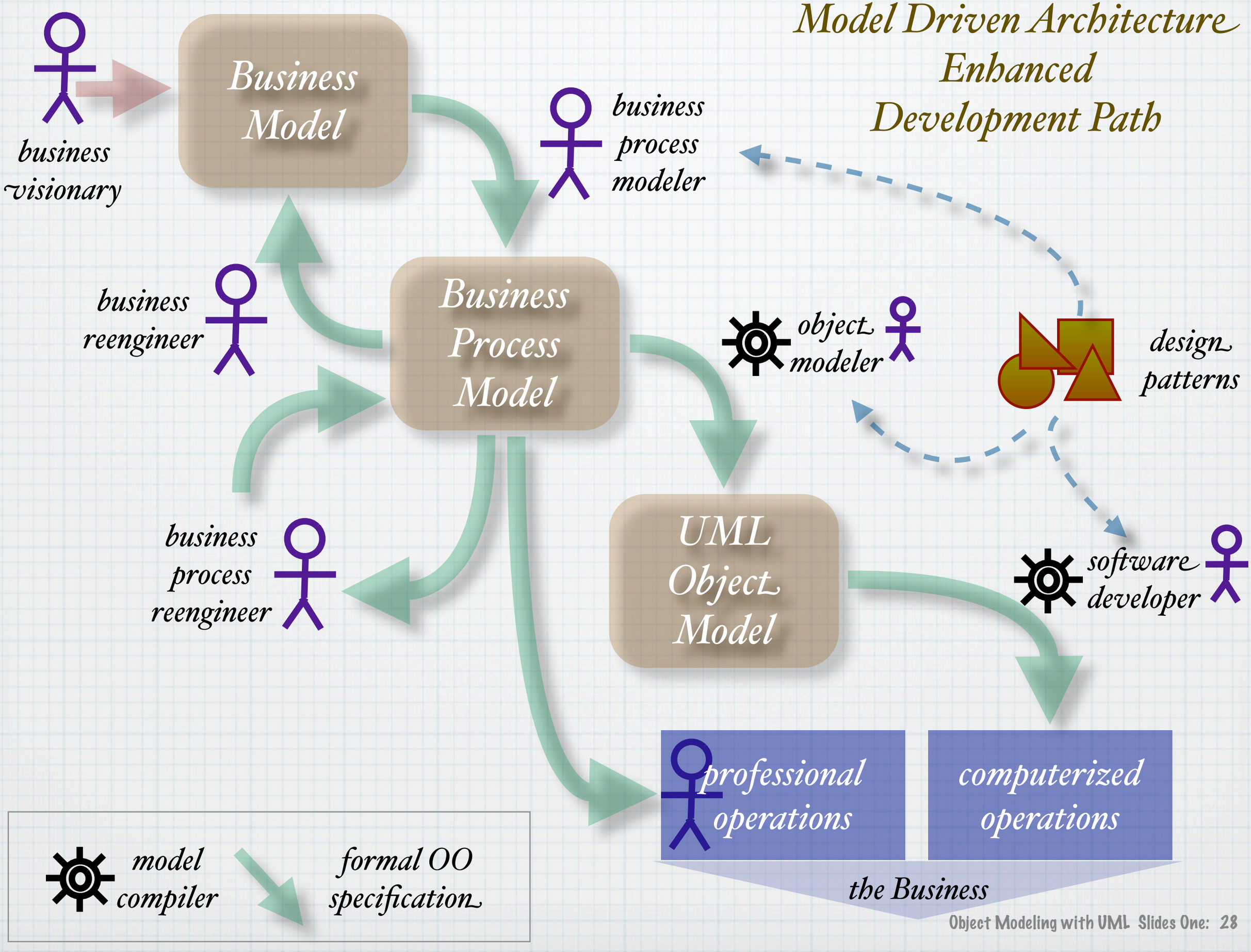
Traditional Computer Information System Development Path



Modeling and Reengineering Informed by Experience



Model Driven Architecture Enhanced Development Path



Keys to Clear Communication

* Shared vocabulary
* Shared objectives
* Shared tools for abstraction
* Consistent construction and maintenance of context
  * Shared SCOPE !!!!
  * Shared FOCUS !!!!

2. Object Modeling

* Object Modeling is a description and analysis tool based upon the object paradigm.
* Modeling at its core is Analysis, a detailed examination of the elements and structure of something.
* The object paradigm defines the characteristics and relationships that may exist and provides a network of conceptual primitives that permit a precise description of a system: components, composites and behavior.

Object Oriented Analysis

As described in Object-Oriented Analysis 2nd edition by Coad and Yourdan, Prentice-Hall, 1991

* Analysis is the precise documentation of the problem domain and the system responsibilities to satisfy user requirements
  * Problem: a question proposed for solution or consideration
  * Domain: sphere or field of activity or influence
* Problem Domain: field of endeavor under consideration
  * System: arrangement of things so related or connected as to form a unity or organic whole
  * Responsibility: condition of being responsible, answerable, accountable, or liable, as for person trust, office or debt
* System Responsibilities: An arrangement of things accountable for, related together as a whole.

Complexity

- Managing it with ...
  - Abstraction
    - procedural
    - data
  - Encapsulation
  - Inheritance
  - Association
  - Communication with Messages
  - Pervading methods of organization
    - objects and attributes
    - whole and parts
    - classes and members, and distinguishing between them
  - Scale
  - Categories of behavior
    - immediate causation
    - change over time
    - similarity of functions

Abstraction

* **Abstraction**: the principle of ignoring those aspects of a subject that are not relevant to the current purpose in order to concentrate more fully on those that are. (Oxford, 1986).
  * When we use abstraction, we admit that what we are considering is complex; rather than try to comprehend the entire thing, we select parts of it.
* **Procedural abstraction**: the principle that any operation that achieves a well-defined effect can be treated by its users as a single entity, despite the fact that the operation may actually be achieved by some sequence of lower-level operations (ibid).
* **Data abstraction**: the principle of defining a data type in terms of the operations that apply to objects of the type, with the constraint that the values of such objects can be modified and observed only by the use of the operations. (ibid)

Encapsulation

* **Encapsulation (information hiding)**: a principle, used when developing an overall program structure, that each component of a program should encapsulate or hide a single design decision.... The interface to each module is defined in such a way as to reveal as little as possible about its inner workings. (Oxford, 1986)
  * If an analyst encapsulates the parts of the analysis effort that are the most volatile, then the (inevitable) changing of requirements becomes less of a threat to the overall effort.
  * Localizing volatility is essential.

Inheritance

* **Inheritance**: a mechanism for expressing similarity among Classes, simplifying definition of Classes similar to one (or more) specialization, making common Attributes and Services explicit within the Class hierarchy or lattice.
  * This principle forms the basis for a significant technique of explicit expression of commonality.
  * Research indicates that this may be the core biological mechanism underlying memory and cognition.

Association

* **Association**: the union or connection of ideas
  * Used to tie together things or events that happen at the same time or under the same circumstances
  * Used to denote that one element of a problem is affected (or aware) of another

Communication with Messages

* **Message**: any communication, written or oral, sent between
* Message interaction corresponds to the imperative mood and present imperative tense of verbs in languages. "The imperative mood conveys commands or requests..." [Britannica, "Imperative mood," 1986]

Pervading methods of organization

* **Objects and Attributes**: the differentiation of experience into particular objects and their attributes (identity and description) –
  * e.g., when they distinguish between one tree and another, and between its size or spatial relationship to other objects.
* **Whole and parts**: the distinction between whole structures and their component parts –
  * e.g., when they contrast a tree with its component branches or distinguish a single tree within a forest.
* **Classes and Members**: the formation of and distinction between different classes of objects (likeness and difference) –
  * e.g., when they form the class of all trees and the classes of all stones and distinguish between members of each class.

Scale

* **Scale**: a principle that applies the whole-part principle to help an observer relate to something very large - without being overwhelmed.
* "When the proportions of architectural composition are applied to a particular building, the two-termed relationship of the parts to the whole must be harmonized with a third term - the observer. This three termed relationship is called scale" [Britannica, "Architecture, The Art of," 1986]
* With scale, analysis notation and strategy can include ways to guide a reader through a larger model revealing information in a gradual and orderly fashion.

Categories of Behavior

* What about the active side of objects – what about their behavior?
* There are three types of behavior classification used most commonly:
  * on the basis of immediate causation,
  * on similarity of evolutionary history [change over time], and
  * on the similarity of function.
    * [Britannica, "Animal Behavior," 1986]

Major Tools in OO Modeling

* In the overall approach, OO modeling achieves these ends:
  * Identifying business objects and determining their sameness and difference
  * Identifying class structures that explain/define the sameness and difference of objects
  * Identifying association structures that define awareness and accessibility
  * Defining attributes that describe individual objects
  * Defining services / behaviors that describe the actions of objects and responsibilities in a specific problem domain

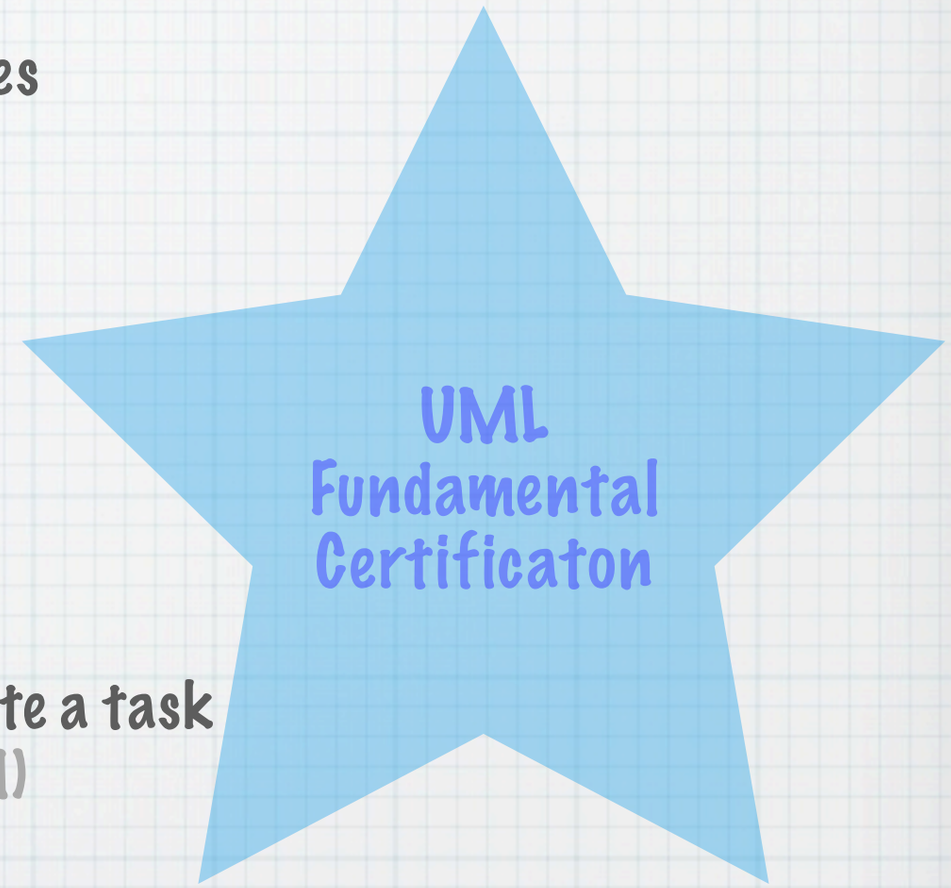
OO Modeling in UML 2

- UML is a collection of diagramming disciplines that define the static and dynamic characteristics of a problem or system
  - Structure diagrams (static in UML 1)
    - Class
    - Composite Structure 2
    - Component
    - Deployment
    - Object
    - Package
  - Dynamic diagrams
    - Activity
    - Interaction
      - Sequence
      - Communication (collaboration in UML 1)
      - Interaction Overview 2
      - Timing 2
    - Use Case
    - State Machine

2 new in UML2

OO Modeling in UML 2

- UML is a collection of diagramming disciplines that define the static and dynamic characteristics of a problem or system
  - Structure diagrams (static in UML 1)
    - **Class** - business objects and their structures
    - Composite Structure 2
    - Component
    - Deployment
    - Object
    - Package
  - Dynamic diagrams
    - **Activity**
    - Interaction
      - **Sequence** - system actions that complete a task
      - **Communication** (collaboration in UML 1)
      - Interaction Overview 2
      - Timing 2
    - **Use Case** - user / system interactions / interfaces
    - State Machine



2 new in UML2

Documentation vs. Diagrams

* The UML diagrams are very useful modeling tools
  * They can be a white board "mock-up"
  * They can be the back of a bar napkin "pipe dream"
  * They can be the back of an envelope "notion to be completed later"
  * They can be so disciplined/detailed that code can be auto-generated
* "Diagrams are complete documentation -- NOT!!"
  * Each element of a diagram requires a prose description
    * class - abstract or concrete
    * generalization / specialization - how the same; how different
    * attribute - valid values, range constraints
    * service - prose, pseudo-code, Java, Smalltalk, C#
    * association - composition (whole-part), aggregation, instance connection
    * cardinality - required versus optional relationships
  * The prose explains how the diagram element accurately reflects the " real world " business rule that being documented

You Need to be able to Explain:

* Domain
* Abstraction
  * procedural
  * data
* Encapsulation
* Organization
  * wholes and parts
  * classes and members
* Inheritance
* Association
* Communication with Messages
* Scale
* Behavior
  * immediate causation
  * change over time
  * similarity of function