

# ΕΠΛ 603 – Topics in Software Engineering

A Problem that cries out for flexible code

---

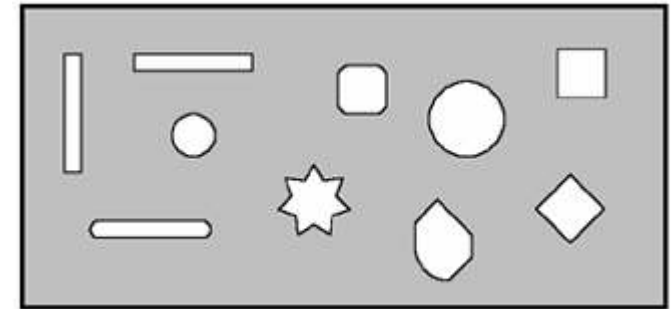
# Overview

- A problem we want to solve:

*Extract information from a large CAD/CAM database to feed a complex and expensive analysis program. Because the CAD/CAM system continues to evolve, the problem cries out for flexible code.*

# Problem

- You are supporting a design center in which engineers used a CAD/CAM system to make **drawings** of sheet-metal parts.



- Task: write a computer tool to **extract information** from the CAD/CAM system so that an expert system could use it in a particular way.

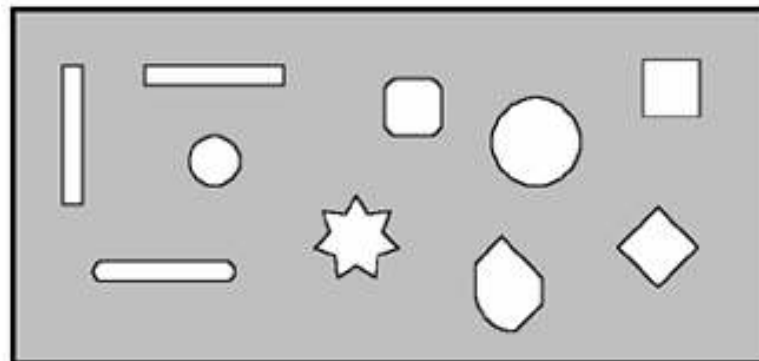
# What is an expert system?

- An **expert system** is a special computer system that uses the rules of a human expert(s) to make automated decisions. Building expert systems involves two steps.
  - Acquire and model the set of rules that experts use to make decisions and accomplish the task.
  - Implement this set of rules in the computer system; this step often uses some sort of commercially available expert system tool. The first step is by far the more difficult assignment for the analyst.

# Shape Classification

- The most important terms used are those that describe the *dimensions* and *geometry* of the cuts made in the sheet metal.
- A piece of sheet metal is cut to a particular size and has shapes cut out inside it. Experts call these cutouts by the general name **feature**. A piece of sheet metal can be fully specified by its external dimensions and the features contained in it.

Shape	Description
Slot	Straight cuts in the metal of constant width that terminate with either squared or rounded edges. Slots may be oriented to any angle. They are usually cut with a router bit. <a href="#">Figure 3-1</a> has three slots on the left side; one is oriented vertically, whereas the others are oriented horizontally.
Hole	Circles cut into the sheet metal. Typically they are cut with drill bits of varying width. <a href="#">Figure 3-1</a> has a hole toward the left surrounded by the three slots and has a larger hole toward the right of the sheet metal.
Cutout	Squares with either squared or rounded edges. These are cut by a high-powered punch hitting the metal with great impact. <a href="#">Figure 3-1</a> has three cutouts; the lower-right one is oriented at 45 degrees.
Special	Preformed shapes that are not slots, holes, or cutouts. In these cases, a special punch has been made to create these quickly. Electrical outlets are a common "special" case. The star shape in <a href="#">Figure 3-1</a> is a special shape.
Irregular	Anything else. They are formed by using a combination of tools. The five-sided object toward the bottom right of <a href="#">Figure 3-1</a> is an irregular shape.



# Additional CAD/CAM terminology

- CAD/CAM experts use the following terminology

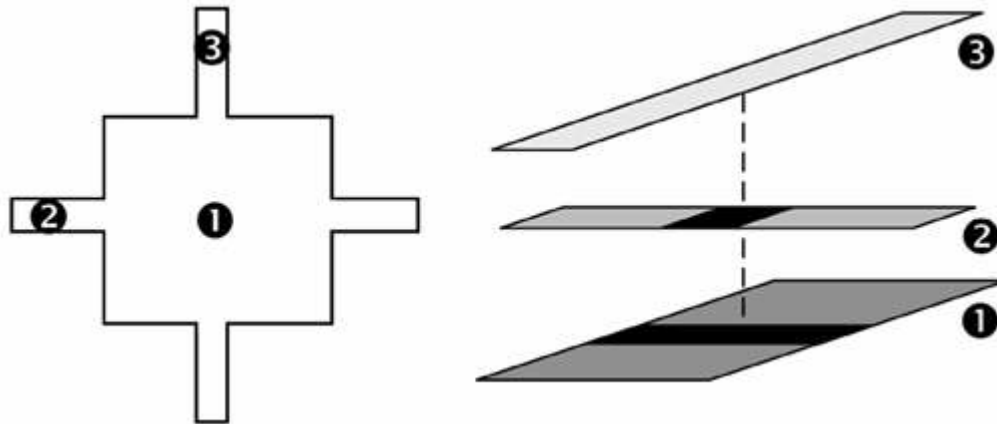
Term	Description
Geometry	The description of how a piece of sheet metal looks: the location of each of the features and their dimensions and the external shape of the sheet metal.
Part	The piece of sheet metal itself. I need to be able to store the geometry of each of the parts.
Dataset or model	The set of records in the CAD/CAM database that stores the geometry of a part.
NC machine and NC set	Numerically controlled (NC) machine. A special manufacturing tool that cuts metal using a variety of cutting heads that are controlled by a computer program. Usually the computer program is fed the geometry of the part. This computer program is composed of commands called the NC set.

# Problem description

- We need to design a program that will allow the expert system to open and read a **model** containing the **geometry** of a **part** that we want to analyze. The expert system will then generate the commands for the numerically controlled (NC) machine to build the piece of sheet metal.
- We are concerned only about sheet-metal parts in this example. However, the CAD/CAM system can handle many other kinds of parts.
- At a high level, we want the system to perform the following steps:
  - Analyze pieces of sheet metal.
  - See how they should be made, based on the features they contain.
  - Generate a set of instructions that are readable by manufacturing equipment. This set of instructions is called an NC set.
  - Give these instructions to manufacturing equipment when we want to make any of these parts

# The expert system's task is not trivial...

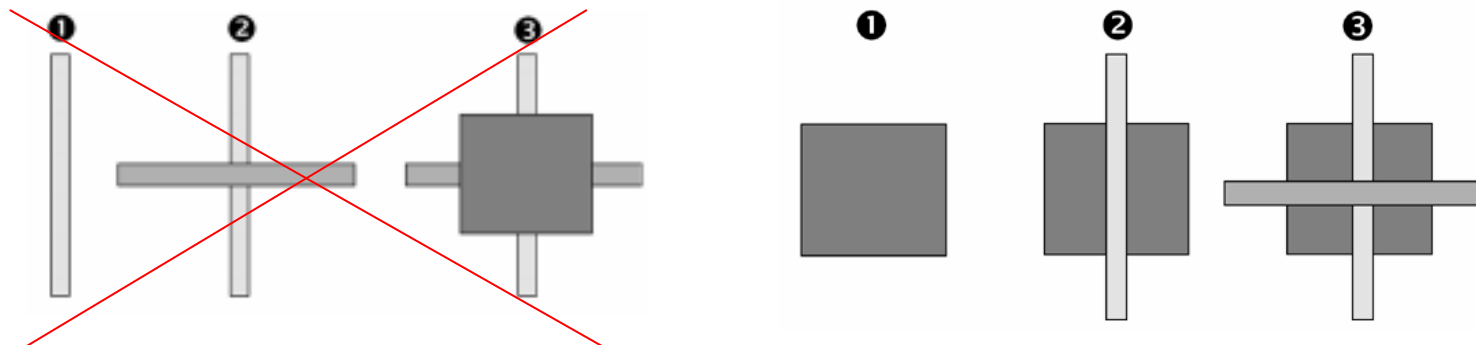
- The difficulty with the programming task is that we cannot just extract the features from the dataset and generate NC set commands. The type of commands to use and the order of these commands depend on the features and their relation to other features.
- Suppose, for example, that a shape is made up of several features: a cutout with two slots. One of the slots runs vertically through the cutout, whereas the other runs horizontally through it, as shown:



A cutout with two slots. Left: How the part looks when finished.  
Right: It is really composed of three features

# ...because it must determine the order of the features!

- It is important to realize that we are actually given the three features on the right to make up the shape on the left. That is because the engineers using the CAD/CAM system typically think in terms of the features to make up more complex shapes, because they know that doing so will enable quicker manufacturing of the parts.
- The problem is, we cannot just generate the NC set commands for the three features independently of one another and hope the part comes out properly often a particular order must be used. In the example, if we do the slots first and then the cutout, as shown, when the cutout is made (remember a cutout is created by using a high-impact punch), the sheet metal will bend because the slots will have weakened the metal.



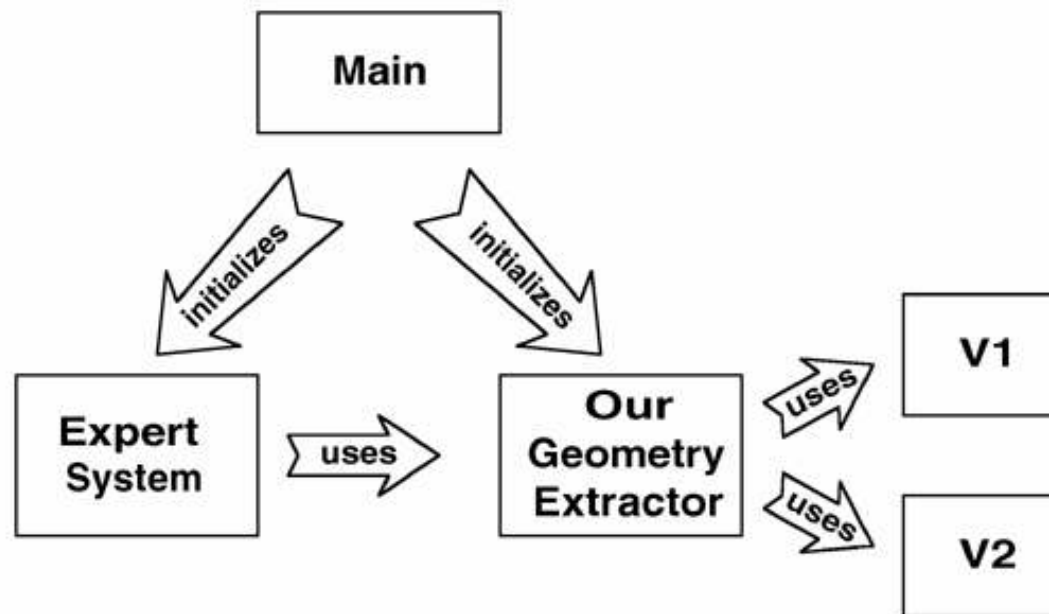
We must create the shape by punching out the cutout first, and then cutting the slots. This works because the slots are created using a router, which applies sideways pressure.

# Challenge:

- Allow the expert system to work with a constantly changing CAD/CAM system
- The CAD/CAM system is constantly evolving, changing. The real problem was to make it possible for the company to continue to use its expensive expert system while the CAD/CAM system changed. This is often the role of architecture in software: to isolate one part of the system, protecting it from another part that has a high risk of change.
- In our situation, they were currently using one version of the CAD/CAM system, Version 1 (V1), and a new version, Version 2 (V2), was coming out shortly. Although one vendor made both versions, the two versions were not compatible.
- For a variety of technical and administrative reasons, it was not possible to translate the models from one version to the next. Therefore the expert system needed to be able to support both versions of the CAD/CAM system.
- In fact, the situation was worse than just having to accommodate two different versions of the CAD/CAM system. We knew a third version was coming out before long, but did not know when that would happen, or what its nature would be. To preserve the investment in the company's expert system, we wanted a system architecture such as:

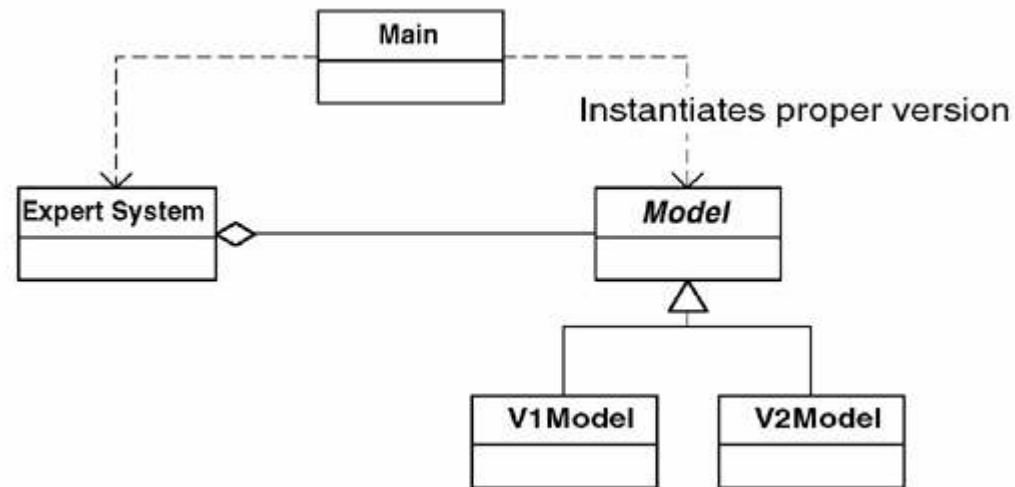
# High-level view of solution

- The application can initialize everything so that the expert system uses the appropriate CAD/CAM system. However, the expert system has to be able to use either version. Hence, we need to make V1 and V2 look the same to the expert system.



# High-level class diagram

- The expert system relates to the CAD/CAM systems through the Model class. The Main class takes care of instantiating the correct version of the Model (that is, V1Model or V2Model).



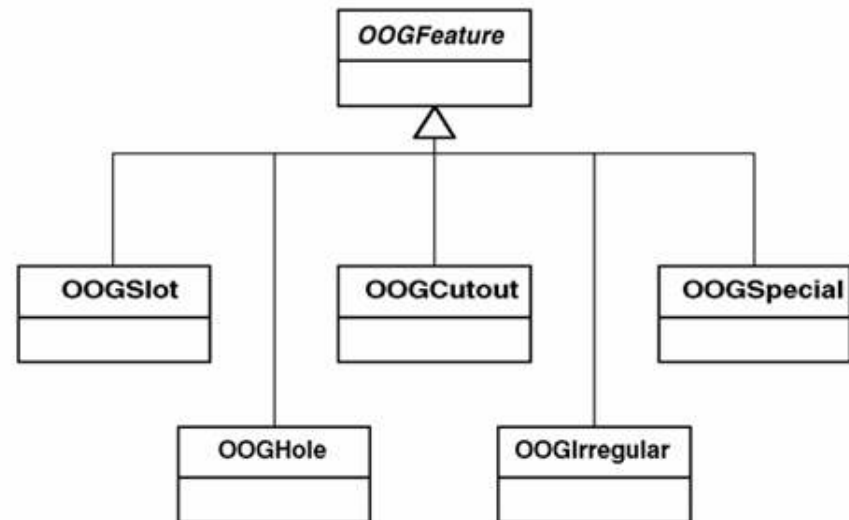
# CAD/CAM V1 is clearly not object-oriented

- Version 1 is essentially a collection of subroutine libraries. To get information from a model, a series of calls must be made. A typical set of queries in CAD/CAM Version 1 would be as follows:
  1. Open model XYZ and return a handle to it.
  2. Store this handle as H.
  3. For the model referred to by H, tell me how many features are present; store as N.
  4. For each feature in the model referred to by H (from 1 to N)
    - a. For the model referred to by H, tell me the ID of the ith element and store as ID.
    - b. For the model referred to by H, tell me the feature type of ID and store as T.
    - c. For the model referred to by H, tell me the X coordinate of ID and store as X.  
(Use T to determine the proper routine to call, based on type.)

# CAD/CAM V2 is object-oriented

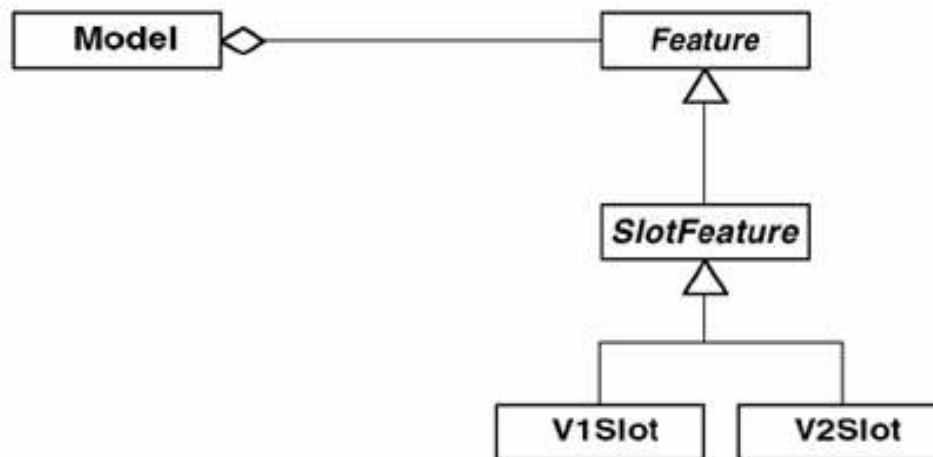
- The CAD/CAM vendor realized the inherent **limitations** of this type of system. The primary motivation for building V2 was to make it **object-oriented**. The geometry in V2 is therefore stored as objects. When a system requests a model, it gets back an **object that represents the model**. This model object contains a set of other objects, each representing a feature. Because the problem domain is based on features, it is not surprising that the classes V2 uses to represent these features correspond exactly to the ones previously mentioned: slots, holes, cutouts, specials, and irregulars.

- Therefore, in V2, we can get a set of objects that corresponds to the features that exist in the sheet metal. The UML diagram shows the classes for the features.

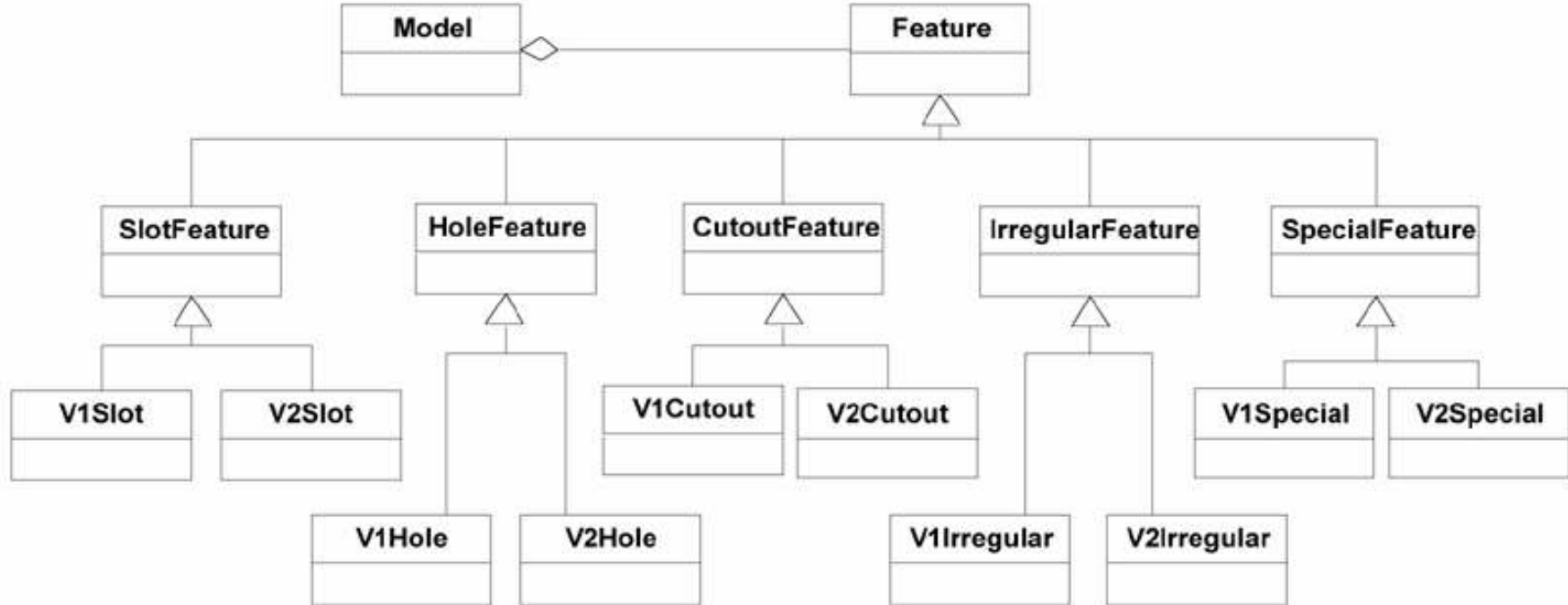


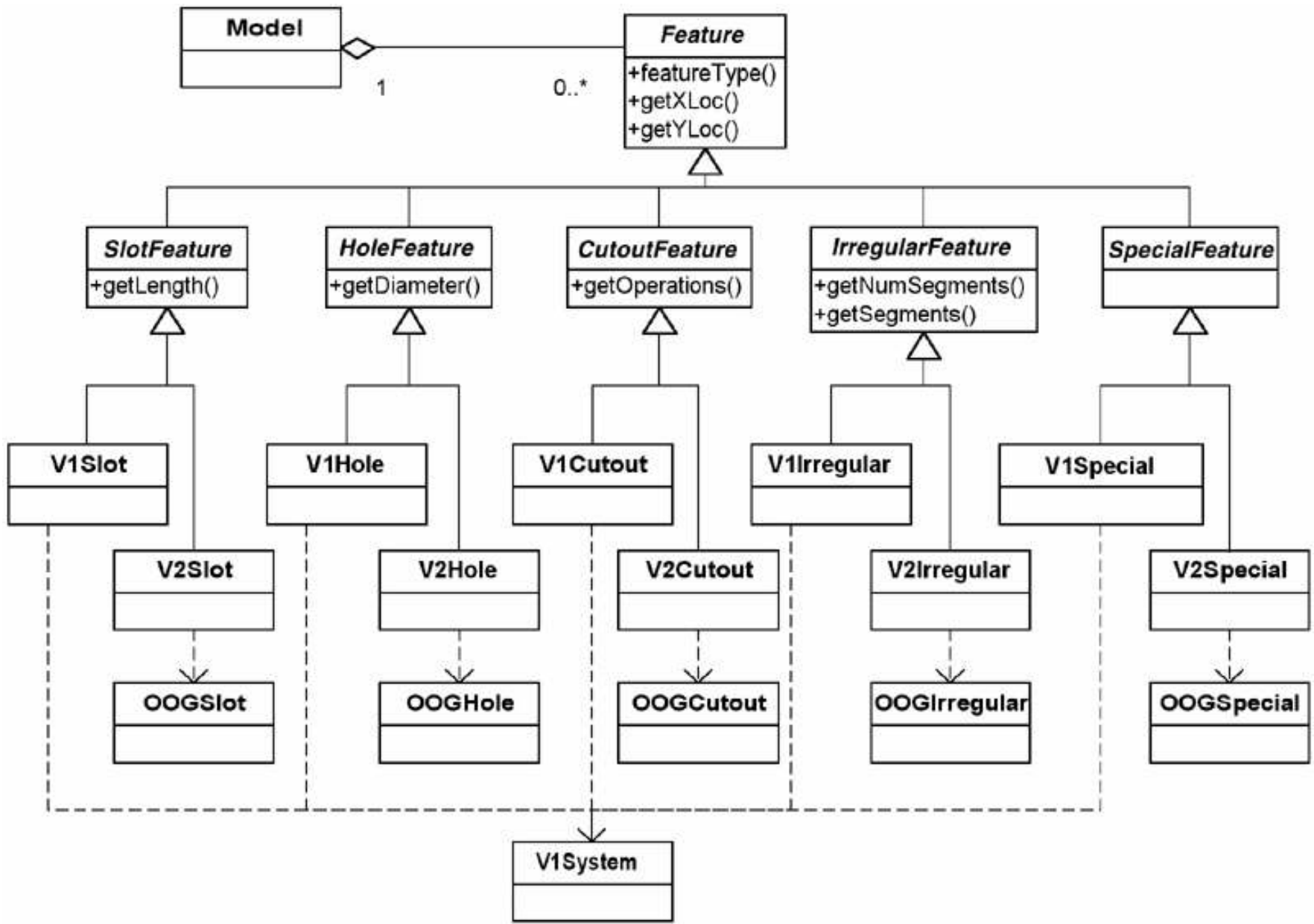
# A standard OO solution

- First attempt based on OO, gets the job done quickly
- In thinking how to solve this problem, we reasoned that if we can solve it for slots, we can use that same solution for cutouts, holes, and so on. In thinking about slots, we saw that we could easily specialize for each case. That is, we would have a SlotFeature class and make a derivation from SlotFeature when we had the V1 system and another derivation when we had a V2 system.



# Original solution to the problem of extracting information





# Solution Evaluation

- This solution satisfies one goal: A common API
- ...but....
- **..Messy.** This is not always a good predictor (and it is a very subjective measurement), but it is another factor that reinforces discomfort with the solution.
- **..Tight coupling.** This solution has tight coupling because the features are related to each other indirectly. These relationships manifest themselves as the likely need to modify all of the features if an existing CAD/CAM system is modified.
- **However, the greatest concern comes from looking into the future. Imagine what will happen when the third version of the CAD/CAM system arrives!**

# Design Patterns Arose from Architecture and Anthropology

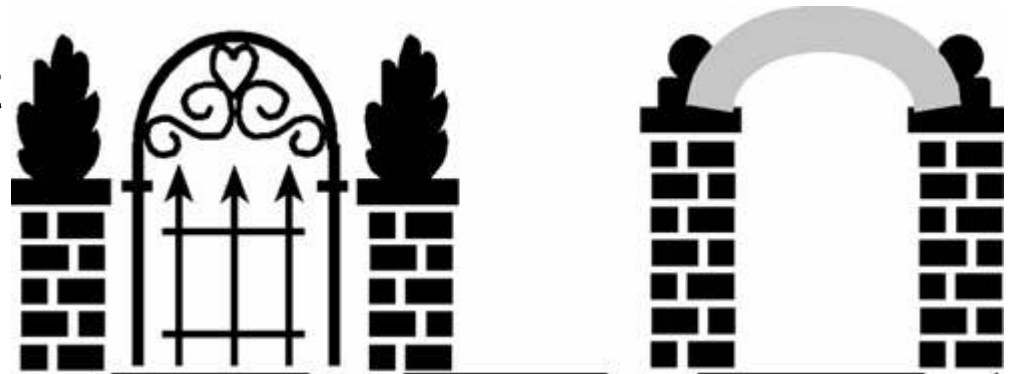
- Years ago, an architect named Christopher Alexander asked himself, "Is quality objective?" Is beauty truly in the eye of the beholder, or would people agree that some things are beautiful and some are not?
- The discipline of cultural anthropology discovered the same thing. That body of work suggests that within a culture, individuals will agree to a large extent on what is considered to be a good design, what is beautiful.

# How do you get good quality repeatedly?

- If you accept the idea that it is possible to recognize and describe a good quality design, how do you go about creating one? Imagine Alexander asking himself:
  - What is present in a good quality design that is not present in a poor quality design?
- ..and..
  - What is present in a poor quality design that is not present in a good quality design?
- Look for the commonalities... especially commonality in the **features** of the problem to be solved

# ..this led to the concept of a pattern

- Alexander discovered that by narrowing his focus in this way by looking at structures that solve similar problem he could discern similarities between designs that were high quality. He called these **similarities patterns**.
- *He defined a pattern as "a solution to a problem in a context":*
- Each pattern describes a problem which occurs over and over again in our environment and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice



# The four parts of every pattern description


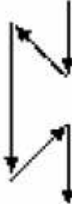
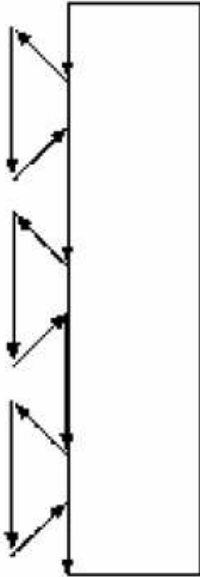
- To review, Alexander says that a description of a pattern involves four items:
  - The name of the pattern
  - The purpose of the pattern, the problem it solves
  - How we could accomplish this
  - The constraints and forces we have to consider in order to accomplish it
- Patterns exist for almost any design problem
- ...and may be combined to solve complex problems

# Adapting Alexander for software

- Are there problems in software that occur **over and over** again that could be solved in somewhat the **same manner**?
- Is it possible to design software in terms of **patterns**, creating **specific solutions** based on these patterns only **after** the patterns had been identified?

# Carpenter's design

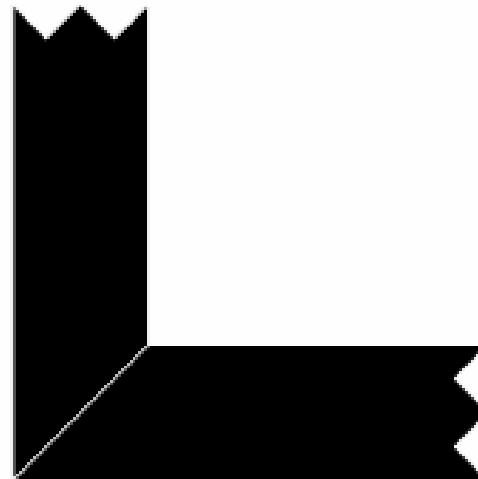
- Consider two carpenters discussing how to build the drawers for some cabinets.
  - **Carpenter 1:** How do you think we should build these drawers?
  - **Carpenter 2:** Well, I think we should make the joint by cutting straight down into the wood, and then cut back up 45 degrees, and then going straight back down, and then back up the other way 45 degrees, and then going straight back down, and then...

Carpenter Says...	Which Looks Like...
"Well, I think we should make the joint by cutting straight down into the wood, and then cut back up 45 degrees..."	
"...then going straight back down, and then back up the other way 45 degrees, and then going straight back down, and then..."	
"...until you end up with a <i>dovetail joint</i> . That is what I was describing!"	

# ...carpenters do not really talk at that level of detail...

- **Carpenter 1**: Should we use a **dovetail** joint or a **miter** joint?
- Already we see a qualitative difference. The carpenters are discussing differences in the quality of solutions to a problem; their discussion is at a higher level, a more abstract level. They avoid getting bogged down in the details of a particular solution.

- A miter joint:
  - is a simpler solution
  - is lightweight
  - is inconspicuous



...carpenters do not really talk at that level of detail...

- A dovetail joint:
  - It is a more complex solution, more involved, more expensive.
  - It is impervious to humidity, temperature.
  - It is independent of the fastening system
  - It is a more aesthetically pleasing

# Who is more efficient? Who would you rather work with?

- In other words, the dovetail joint is a strong, dependable, beautiful joint that is complex (and therefore expensive) to make.
- So, when Carpenter 1 asked..
  - Should we use a dovetail joint or a miter joint?
- The real question that was being asked was..
  - Should we use a joint that is expensive to make but is both beautiful and durable, or should we just make a quick and dirty joint that will last at least until the check clears?