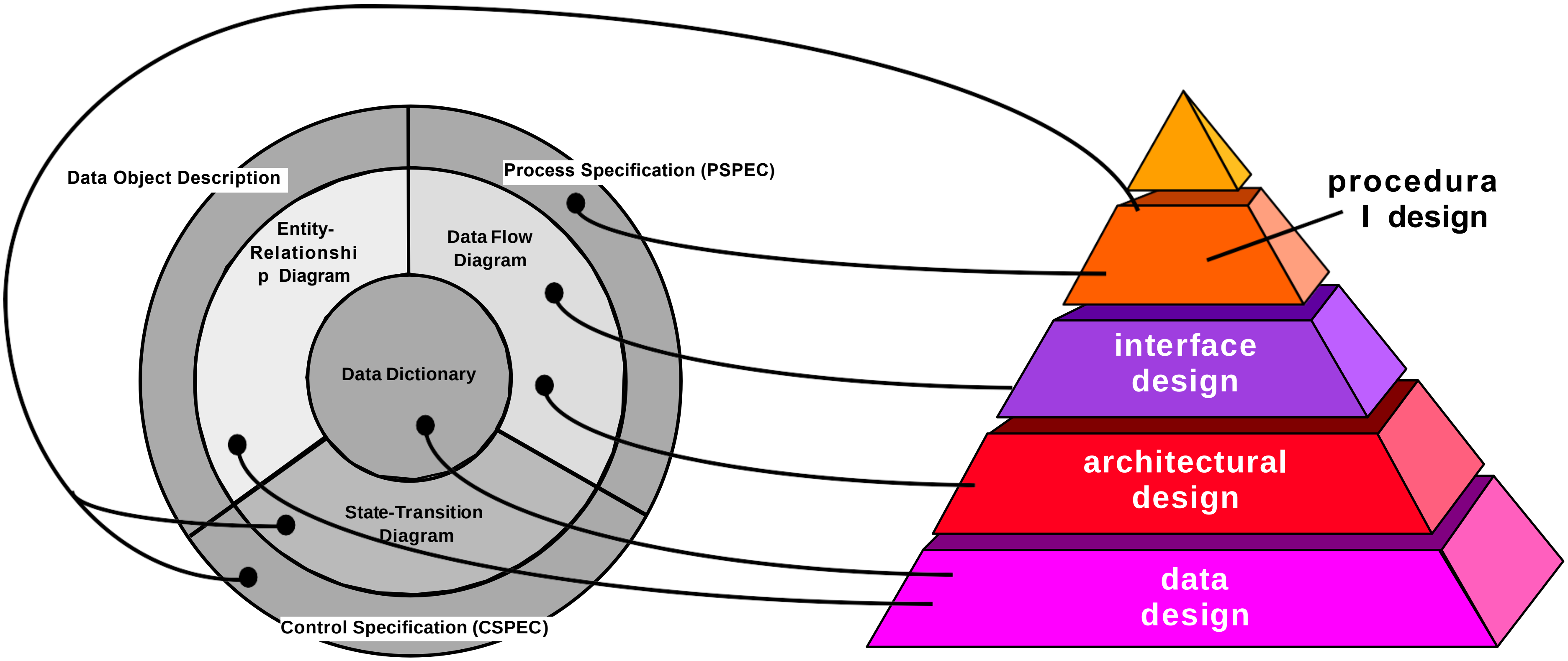
Software Design

SW Design

- Software design is an iterative process through which requirements are translated into a "blueprint" for constructing the software.
- Initially, the blueprint depicts a holistic view of software.

Process of Design Engineering

- During the design process the software specifications are transformed into design models
- Models describe the details of the data structures, system architecture, interface, and components.
- Each design product is reviewed for quality before moving to the next phase of software development.
- At the end of the design process a design model and specification document is produced.
- This document is composed of the design models that describe the data, architecture, interfaces and components.



THE ANALYSIS MODEL

THE DESIGN MODEL

Design Specification Models

- **Data design** – created by transforming the analysis information model (data dictionary and ERD) into data structures required to implement the software. Part of the data design may occur in conjunction with the design of software architecture. More detailed data design occurs as each software component is designed.
- **Architectural design** - defines the relationships among the major structural elements of the software, the "design patterns" than can be used to achieve the requirements that have been defined for the system, and the constraints that affect the way in which the architectural patterns can be applied. It is derived from the system specification, the analysis model, and the subsystem interactions defined in the analysis model (DFD).

Design Specification Models

- **Interface design** - describes how the software elements communicate with each other, with other systems, and with human users; the data flow and control flow diagrams provide much of the necessary information required.
- **Procedural / Component-level design** - created by transforming the structural elements defined by the software architecture into procedural descriptions of software components using information obtained from the process specification (PSPEC), control specification (CSPEC), and state transition diagram (STD).

Design - Fundamental Concepts

- Abstraction
- Architecture
- Patterns
- Modularity
- Information hiding
- Functional independence
- Refinement
- Refactoring

Abstraction

- Data Abstraction
- Procedural Abstraction

Architecture Design

Design can be represented as

- Structural Models
  - Defines the components of a system (e.g., modules, objects, filters) and
  - How the components are packaged and interact with one another.
- Framework Models
  - Increase level of abstraction
- Dynamic Models and Process Models
  - Predicts behavioral and reliability aspects
- Functional Models
  - Depicts functional Hierarchy.

Patterns

- a pattern is "a common solution to a common problem in a given context." While architectural styles can be viewed as patterns describing the high-level organization of software (their macroarchitecture), other design patterns can be used to describe details at a lower, more local level (their microarchitecture).

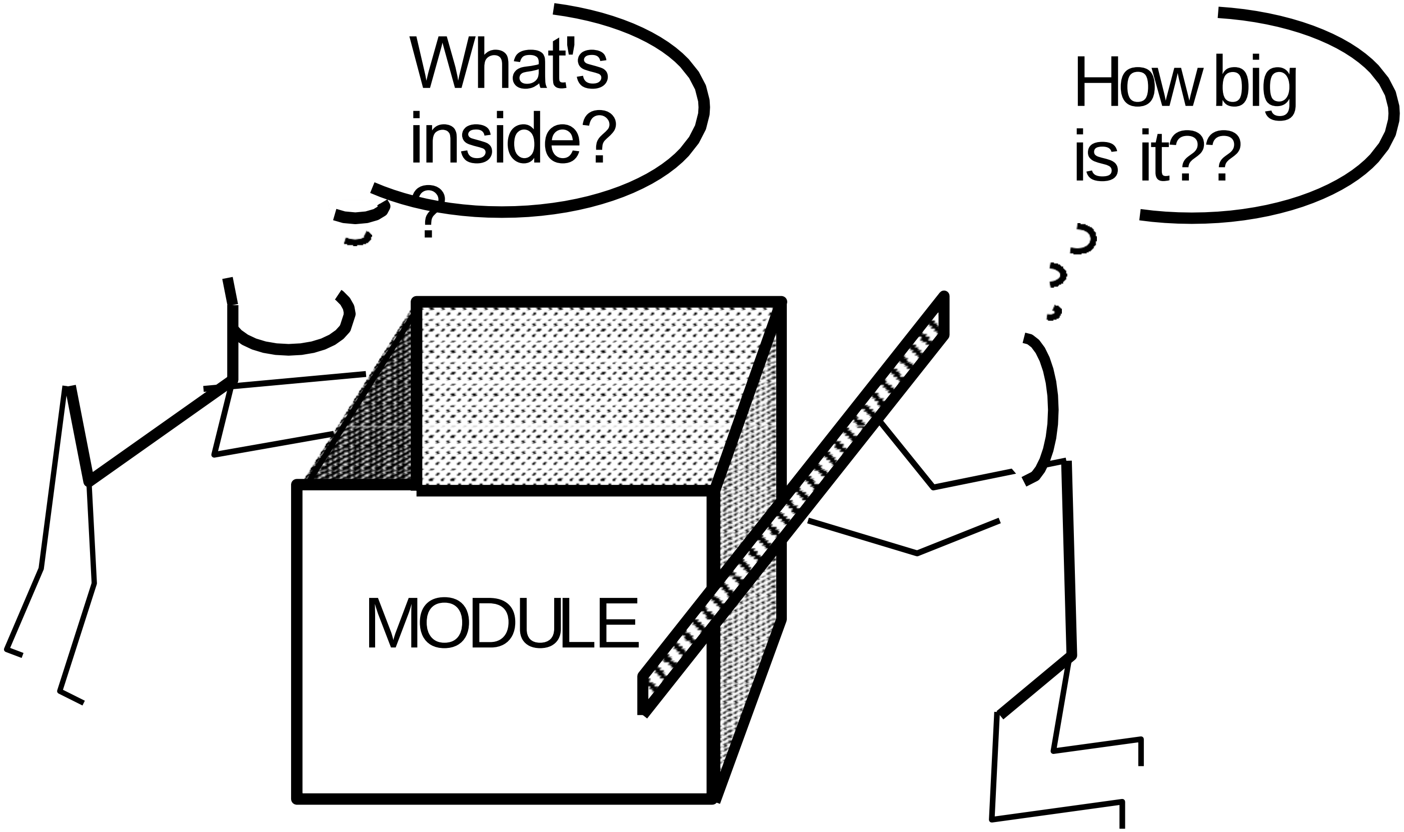
Design Pattern

- Design Pattern enables a designer to determine whether the pattern :
  - is applicable to the current work
  - can be reused
  - can serve as a guide for developing a similar, but functionally or structurally different pattern.

Modular Design

- Easier to change
- Easier to build
- Easier to maintain

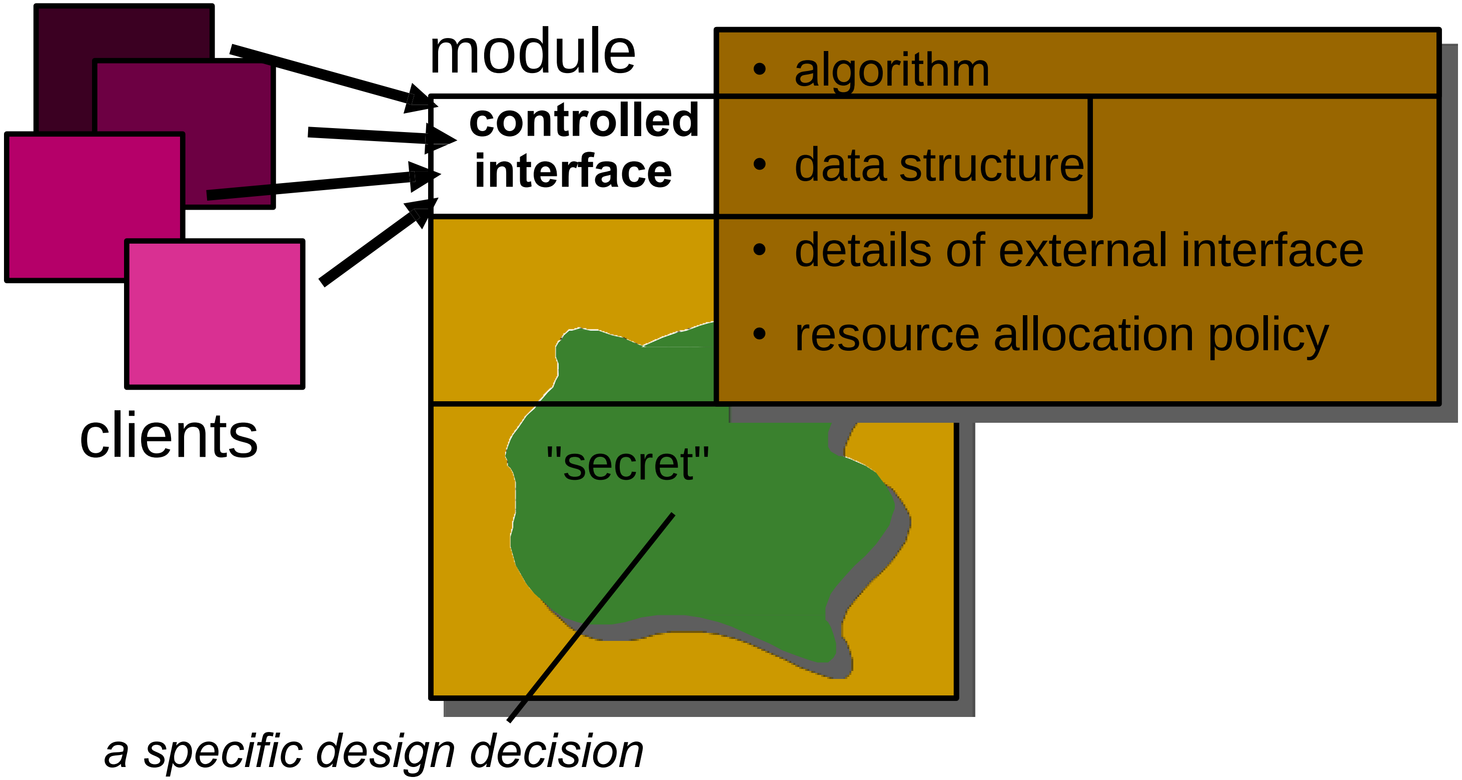
Sizing Modules: Two Views



Information Hiding

- Principle of information hiding says that a good split of modules is when modules communicate with one another with only the information necessary to achieve the s/w function.
- Data hiding is a CRITERION for modular design.

Information Hiding


module
controlled interface
clients
"secret"
a specific design decision
• algorithm
• data structure
• details of external interface
• resource allocation policy

Information Hiding (Benefits)

- reduces the likelihood of “side effects”
- limits the global impact of local design decisions
- emphasizes communication through controlled interfaces
- discourages the use of global data
- leads to encapsulation—an attribute of high quality design
- results in higher quality software

Functional Independence

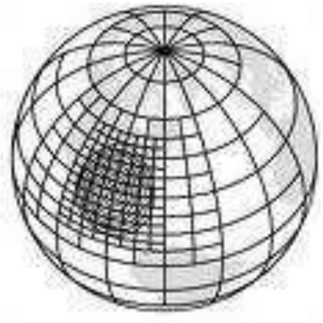
COHESION - the degree to which a module performs one and only one function.

COUPLING - the degree to which a module is "connected" to other modules in the system.

Summary

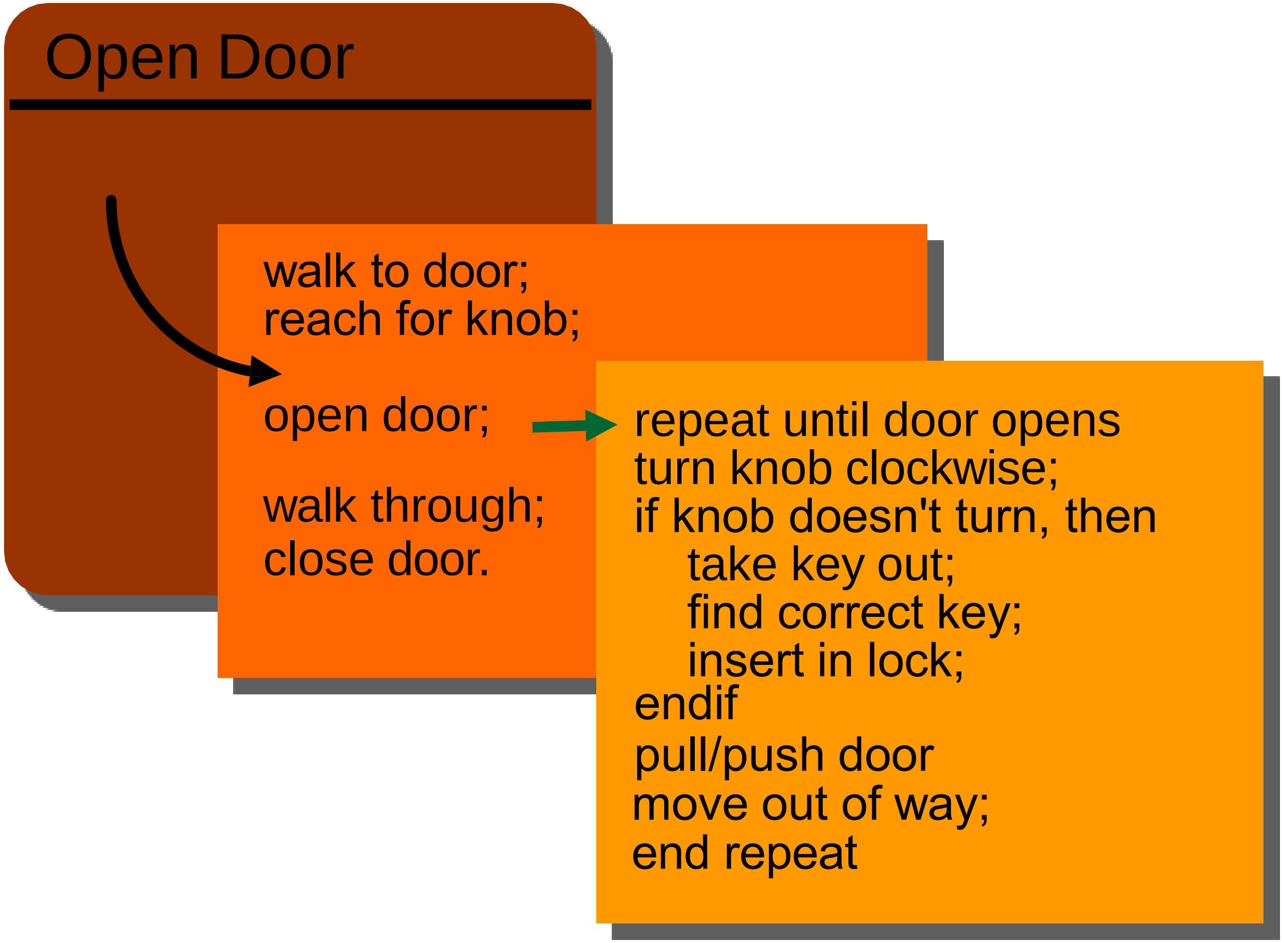
COHESION	COUPLING
The measure of strength of the association of elements within a module	The measure of interdependence of one module to another
It is the degree to which the responsibility of a single component form a meaningful unit	It describes the relationships between software components
It is a property or characteristics of an individual module	It is a property of a collection of modules

Refinement



- Refinement is a process of **elaboration**
- It is a top-down design strategy
- A program is developed by successfully refining levels of procedural details

Stepwise Refinement



Open Door

```
walk to door;
reach for knob;

open door;

walk through;
close door.
```

```
repeat until door opens
turn knob clockwise;
if knob doesn't turn, then
    take key out;
    find correct key;
    insert in lock;
endif
pull/push door
move out of way;
end repeat
```

Refactoring

- "Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code [design] yet improves its internal structure.“
- When software is refactored, the existing design is examined for
- redundancy
- unused design elements
- inefficient or unnecessary algorithms
- poorly constructed or inappropriate data structures
- or any other design failure that can be corrected to yield a better design.