Software Architect

How are they different than Software Developers?

Why become one?

CNN / Money
(Best Jobs in America 2015)

#1 Software Architect

- **Median pay**: $124,000 / **Top pay**: $169,000
- **10-year job growth**: 23%
- **In the same way an architect designs a house, software architects lay out a design plan for new programs.** That usually means leading a team of developers and engineers, and making sure all the pieces come together to make fully-functioning software.
- **What's great:** New problems come up all the time and new technologies arise, making each day different, and keeping professionals in demand.

http://money.cnn.com/gallery/pf/2015/01/27/best-jobs-2015/

Software Architect
(Wikipedia Definition)

A software architect is a software expert who makes high-level design choices and dictates technical standards, including software coding standards, tools, and platforms. The main responsibilities include:

- Limit choices available during development by:
  - » choosing a standard way of pursuing application development
  - » creating, defining, or choosing an application framework
- **Recognize potential reuse in the application by:**
  - » observing and understanding the broader system environment
  - » **creating the component design**
  - » having knowledge of other applications in the organization
- **Subdivide a complex application, during the design phase, into smaller, more manageable pieces**
- ~~Grasp~~ **Define the functions of each component within the application**
- **Understand the interactions and dependencies among components**
- **Communicate these concepts to developers**

Building Phases

Customer	Planning Documents	Physical System Decisions	Physical Installation

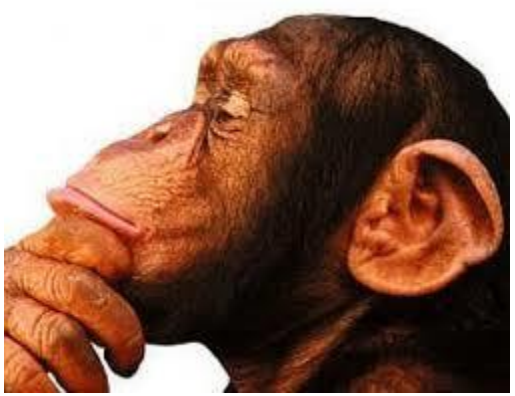
Building a New Home

Buyer	Architect (Blueprints)	General Contractor (Piping, Wiring)	Skilled Worker (Plumber, Electrician)

Building a New Software Application

Domain Expert	Software Architect (UML Artifacts)	Software Architect/Designer (Data Store, Security)	Software Developer/Programmer (Code)

But what does a Software Architect actually DO when developing a new software application?



Software Architect Responsibilities

- Read, understand and clarify Functional Specification

➔ *Requires direct interaction with the "Domain Expert" to identify and document the problem (requirements & constraints) that must be solved by the new system.*

FS: Enrollment System at De Anza College

De Anza courses are offered by its departments (Ex: CIS) and available Quarterly (Ex: Fall and Winter). Each course has an identifying number (ex: 28), a Name (Ex: Object Oriented Analysis and Design), a description, a number of credits and an optional set of required prerequisites.

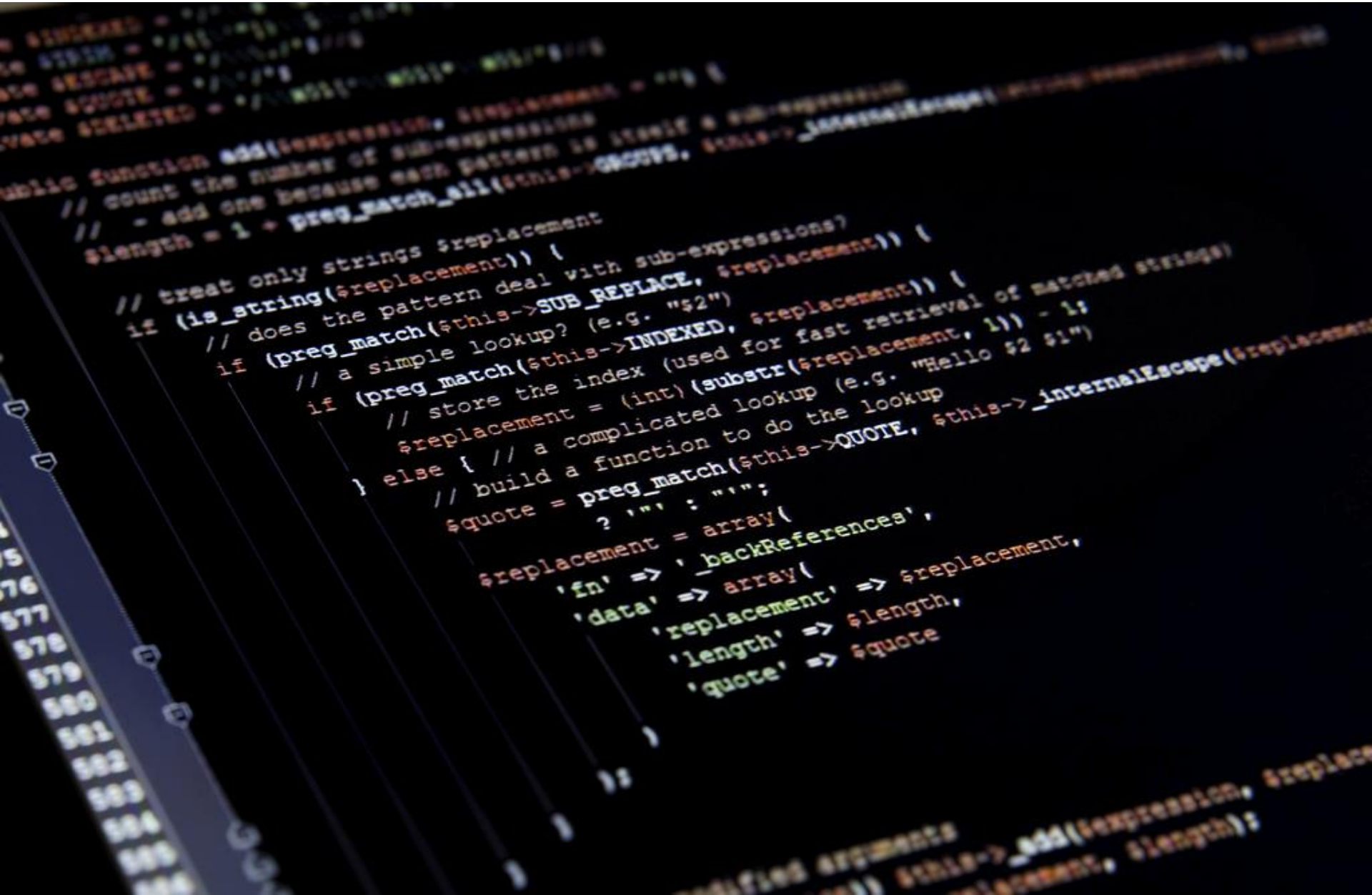
Each course is assigned a set of times during the week when it meets. It is also assigned a teacher who is both qualified and willing to teach that course, and who is free during the assigned times the course meets. It is also assigned a room, which also must be free during those times.

Students may then attempt to enroll in a course if they are paid up, have taken all its required prerequisites, are not already enrolled in the course, and if they are not already taking another course which overlaps the assigned times for this one.

Depending upon the size of the course and the number of students already enrolled, the student's enrollment request may either be accepted, or the student may be wait listed, or the request may be denied. If the student is accepted, her attendance will be tracked and at the end of the academic quarter she will receive a final grade.

You must produce a working solution. Where do you begin?

WRONG!!!


// treat only strings $replacement
if (is_string($replacement)) {
// does the pattern deal with sub-expressions?
if (preg_match($this->SUB_REPLACE, $replacement)) {
// a simple lookup? (e.g. "$2")
if (preg_match($this->INDEXED, $replacement)) {
// store the index (used for fast retrieval of matched strings)
$replacement = (int)(substr($replacement, 1)) - 1;
} else { // a complicated lookup (e.g. "Hello $2 $1")
// build a function to do the lookup
$quote = preg_match($this->QUOTE, $this->_internalEscape($replacement))
? '"' : "'";
$replacement = array(
'fn' => '_backReferences',
'data' => array(
'replacement' => $replacement,
'length' => $length,
'quote' => $quote

Software Architect Responsibilities

- Read, understand and clarify Functional Specification

OOA:

- **Identify basic “abstractions” referenced in the FS**

➔ *What are the words in the FS which must be further refined to determine exactly what the proposed system is actually required to do?*

FS: Enrollment System at De Anza College

De Anza **College Courses** are offered by its **Departments** (Ex: CIS) and offered Quarterly (Ex: Fall and Winter). Each course has an identifying number (ex: 28), a Name (Ex: Object Oriented Analysis and Design), a description, a number of credits and an optional set of required **Prerequisites.**

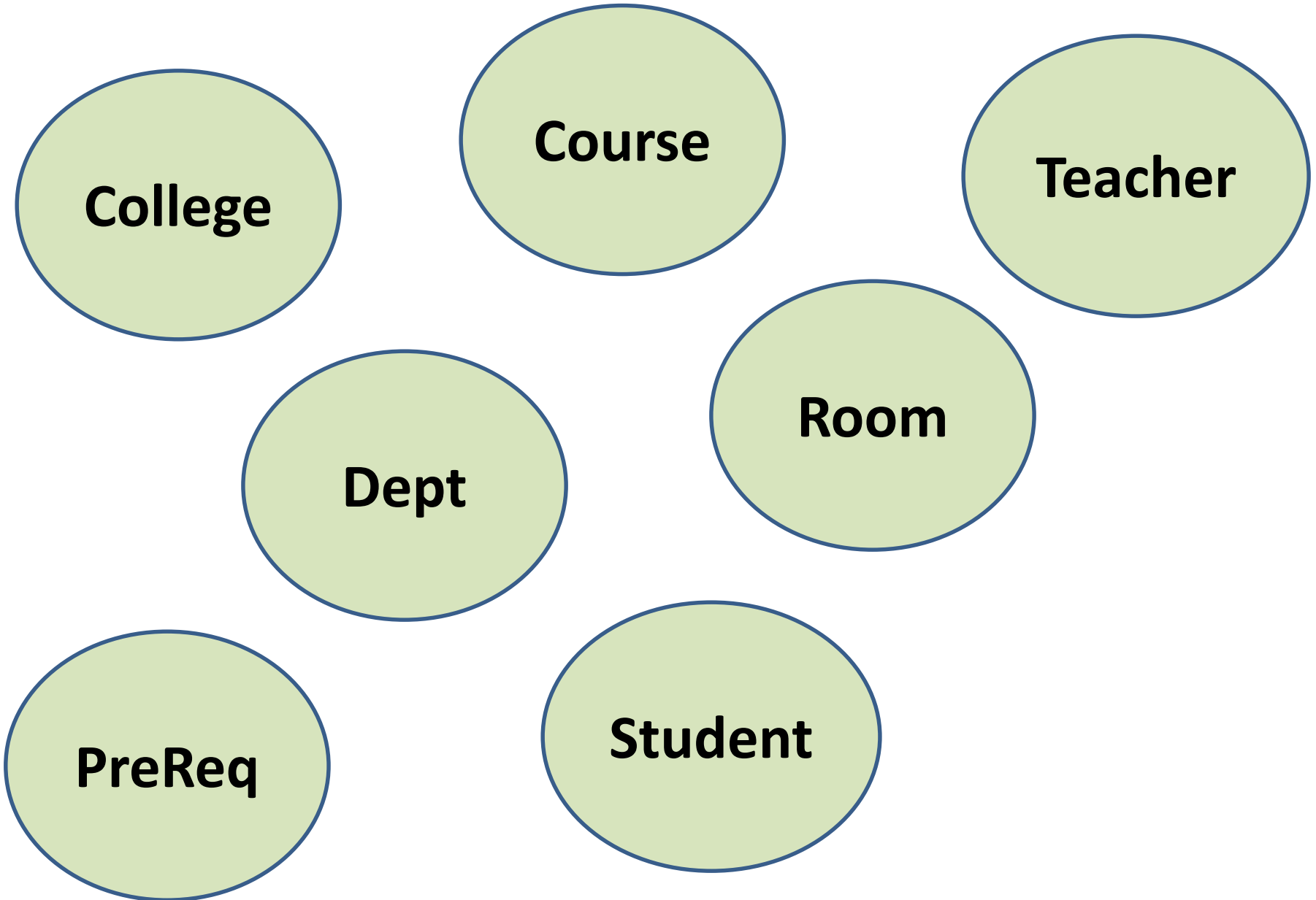
Each course is assigned a set of times during the week when it meets. It is also assigned a **Teacher** who is both qualified and willing to teach that course, and who is free during the assigned times the course meets. It is also assigned a **Room**, which also must be free during those times.

Students may then attempt to enroll in a course if they are paid up, have taken all its required prerequisites, are not already enrolled in the course, and if they are not already taking another course which overlaps the assigned times for this one.

Depending upon the size of the course and the number of students already enrolled, the student's enrollment request may either be accepted, or the student may be wait listed, or the request may be denied. If the student is accepted, her attendance will be tracked and at the end of the academic quarter she will receive a final grade for the course.

You must produce a working solution. Where do you begin?

Basic Abstractions in Enrollment System



Refining the Abstractions

- **Can a Room / Teacher be assigned to a Course?**

Refining the Abstractions

- **Can a Room / Teacher be assigned to a Course?**
  - Course might be taught in several rooms
  - Course might be taught by several teachers
- **Can a Student “enroll” in a Course?**

Refining the Abstractions

- **Can a Room / Teacher be assigned to a Course?**
  - Course might be taught in several rooms
  - Course might be taught by several teachers
- **Can a Student “enroll” in a Course?**
  - No. Need a new abstraction

Refining the Abstractions

- **Can a Room / Teacher be assigned to a Course?**
  - Course might be taught in several rooms
  - Course might be taught by several teachers
- **Can a Student "enroll" in a Course?**
  - No. Need a new abstraction: **Section**
  - Section has Room, Teacher, Students, Hours to meet
  - Course has description, Name, # Credits, textbook, …
- **What exactly is a "Prerequisite"?**

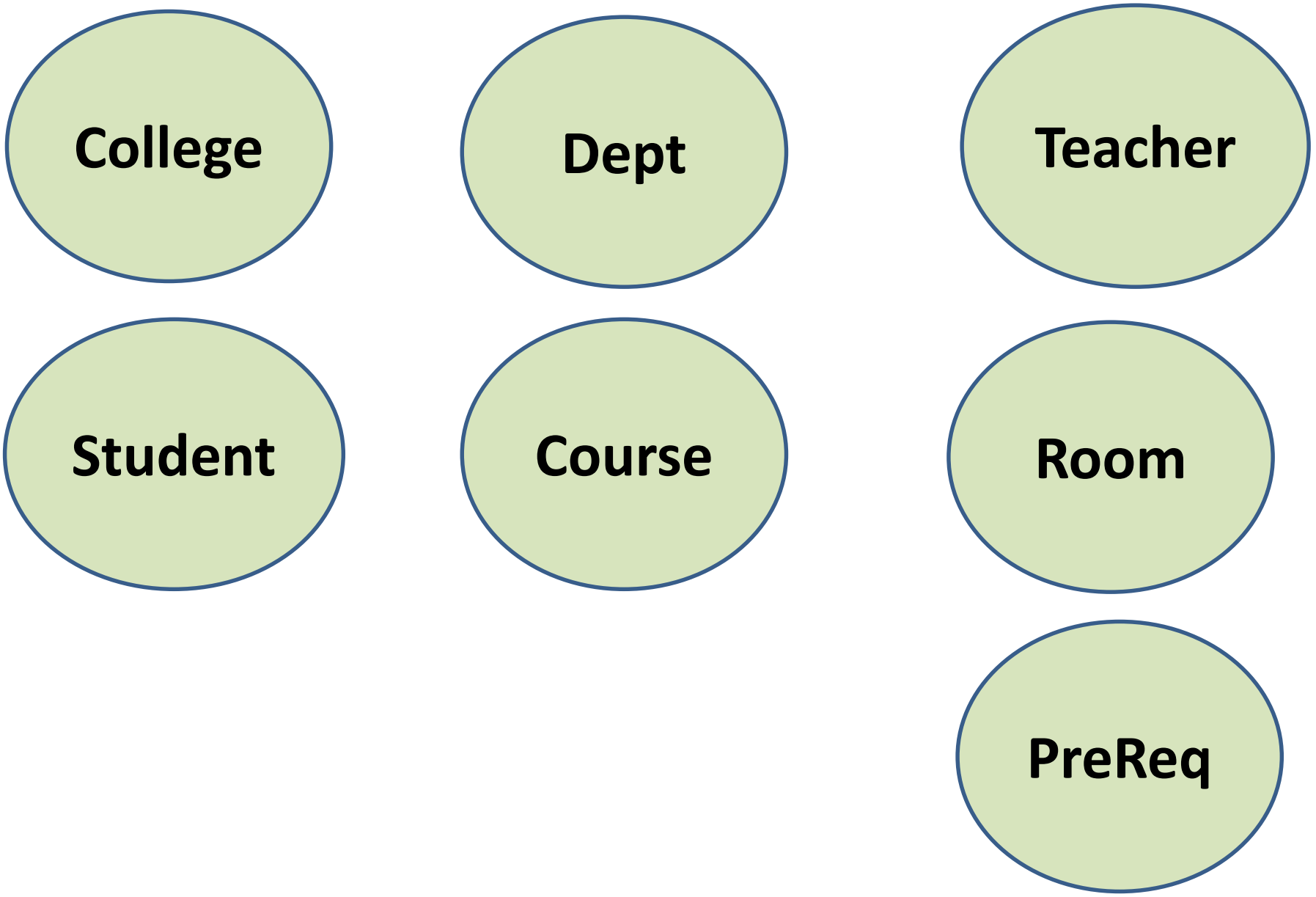
Refining the Abstractions

- **Can a Room / Teacher be assigned to a Course?**
  - Course might be taught in several rooms
  - Course might be taught by several teachers
- **Can a Student "enroll" in a Course?**
  - No. Need a new abstraction: **Section**
  - Section has Room, Teacher, Students, Hours to meet
  - Course has description, Name, # Credits, textbook, …
- **What exactly is a "Prerequisite"?**
  - A Prerequisite is a Course
- **How do Teachers, Rooms, Students, detect scheduling conflicts?**

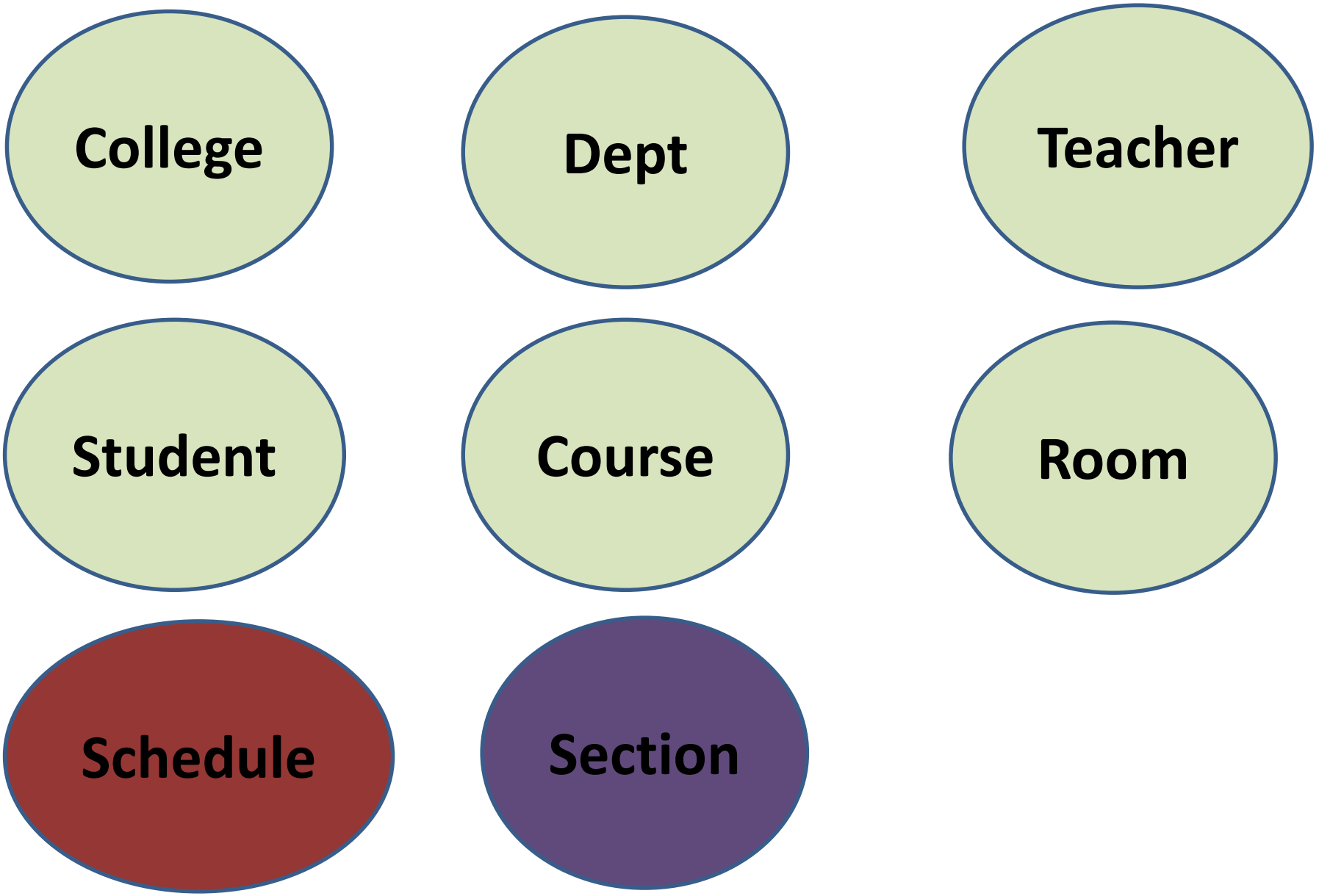
Refining the Abstractions

- **Can a Room / Teacher be assigned to a Course?**
  - Course might be taught in several rooms
  - Course might be taught by several teachers
- **Can a Student "enroll" in a Course?**
  - No. Need a new abstraction: **Section**
  - Section has Room, Teacher, Students, Hours to meet
  - Course has description, Name, # Credits, textbook, …
- **What exactly is a "Prerequisite"?**
  - A Prerequisite is a Course
- **How do Teachers, Rooms, Students, detect scheduling conflicts?**
  - They are each assigned their own Schedule

Basic Abstractions in Enrollment System

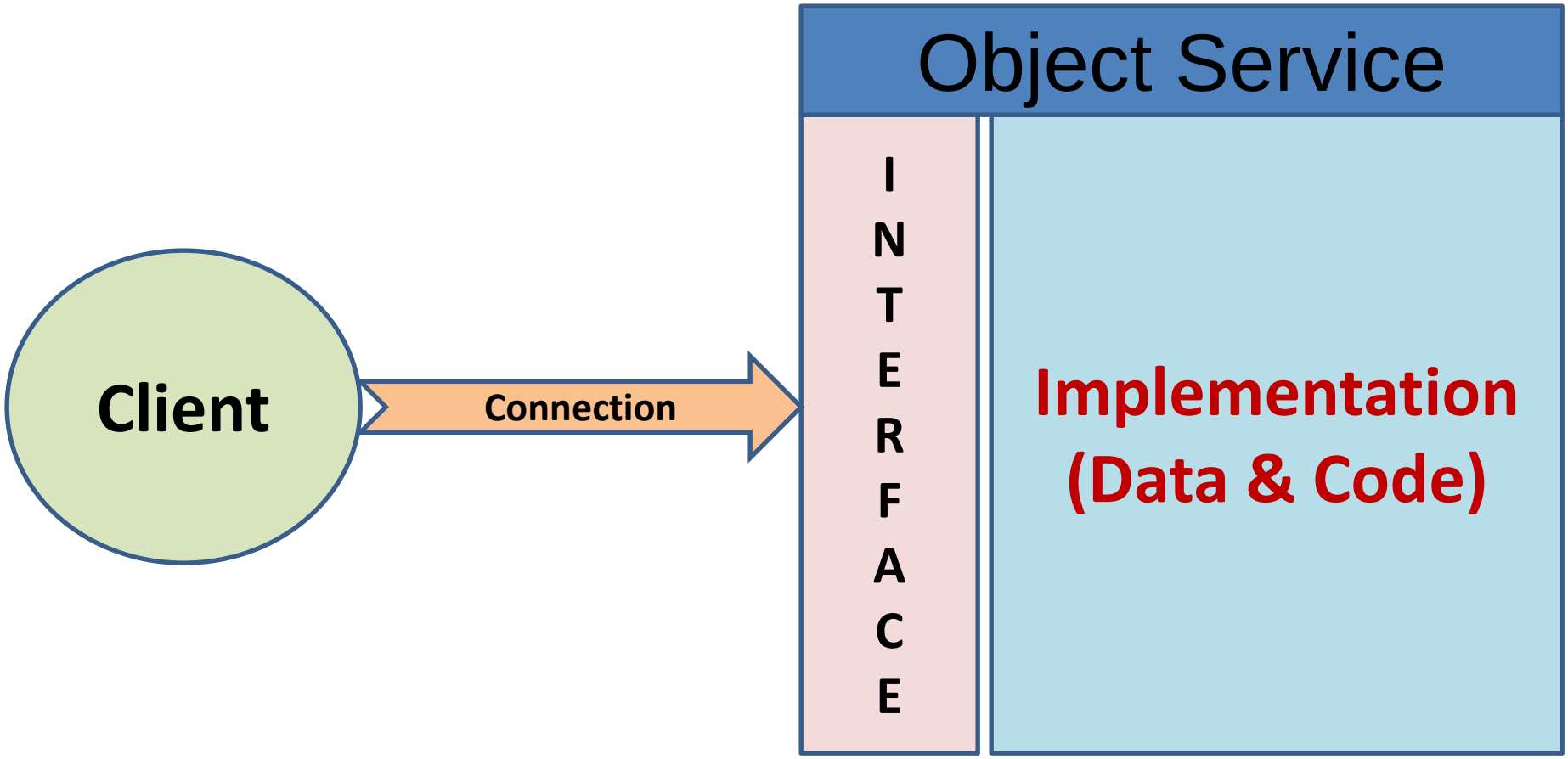


Refined Abstractions in Enrollment System



Language	Year	Construct	Components	Advance
Basic	1965	Program	Code, Global Data variables	Higher level Language
Fortran	1970	Subroutine	Independent Code & “local” Data variables	Reusable Code Modules
C	1975	Struct	Abstraction as collection of Data Types	Treat Data Types as new “entity” (ex: “Student”)
C++	1985	Object	Data Type Collection + Associated Functions	“Encapsulate” Data Types & the code that uses them

Key OO Concept: Encapsulation




```
                                    float cum;
float getCum ()   { return (cum); }
```

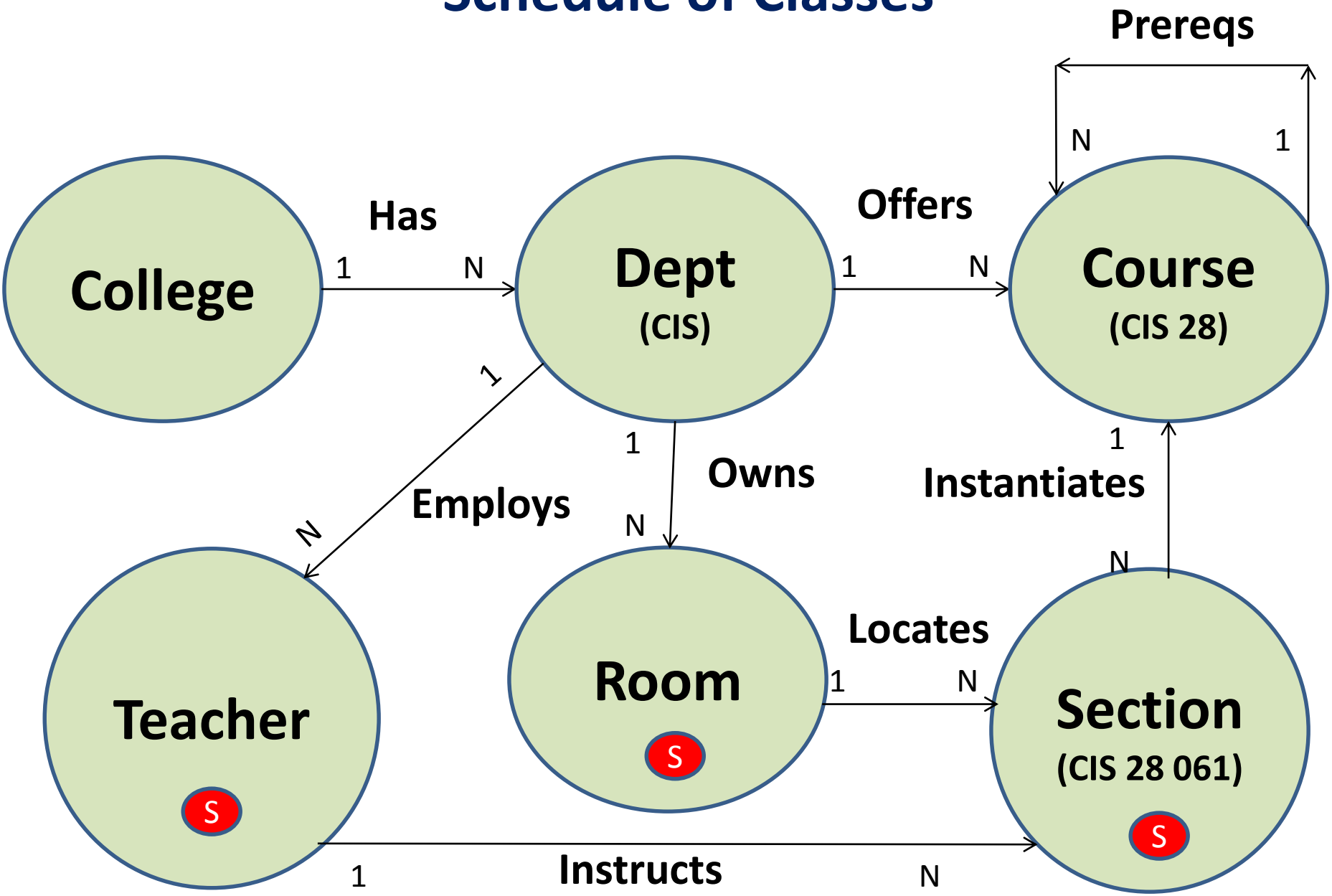
Software Architect Responsibilities

- Read, understand and clarify Functional Specification

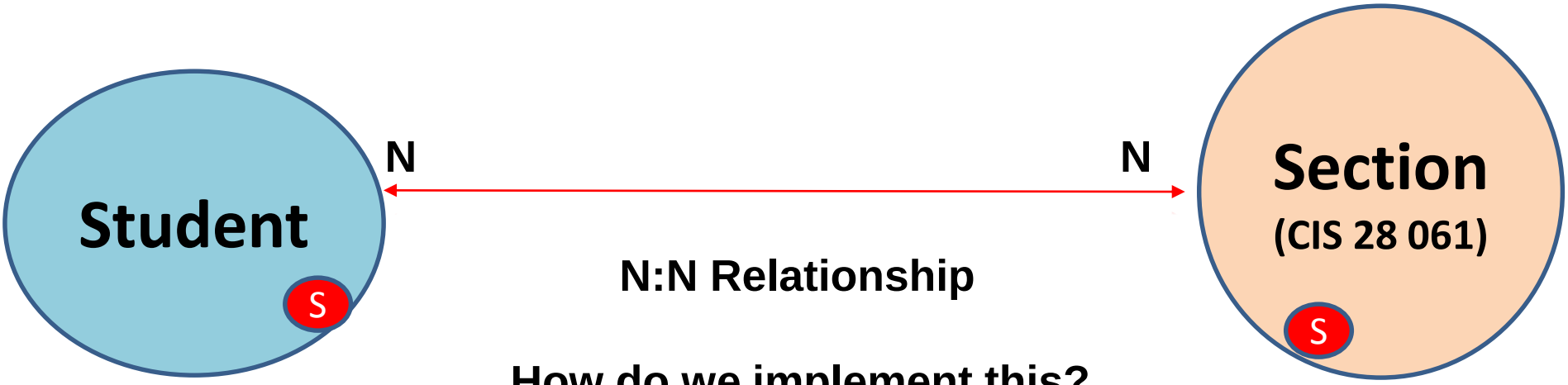
OOA:

- Identify basic “abstractions” referenced in the FS
- **Determine “relationship” between abstractions**
  - **(*Often involves selecting & incorporating “Design Patterns” into the architecture*)**
  - *http://www.uml.org.cn/c++/pdf/DesignPatterns.pdf*

Conceptual Class Diagram:
Schedule of Classes



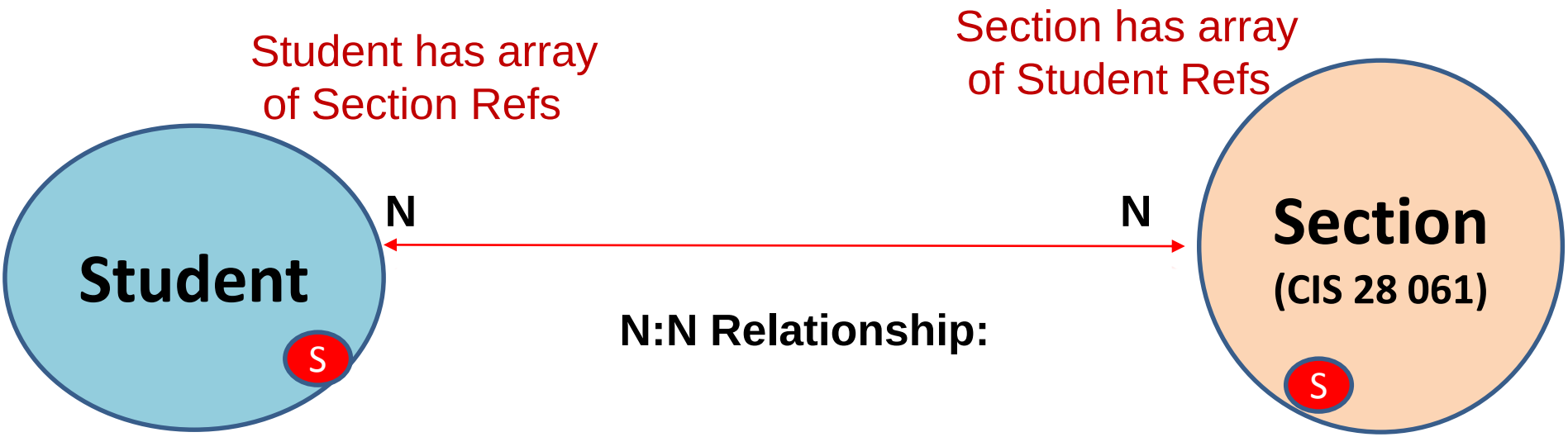
Conceptual Class Diagram: Enrollment



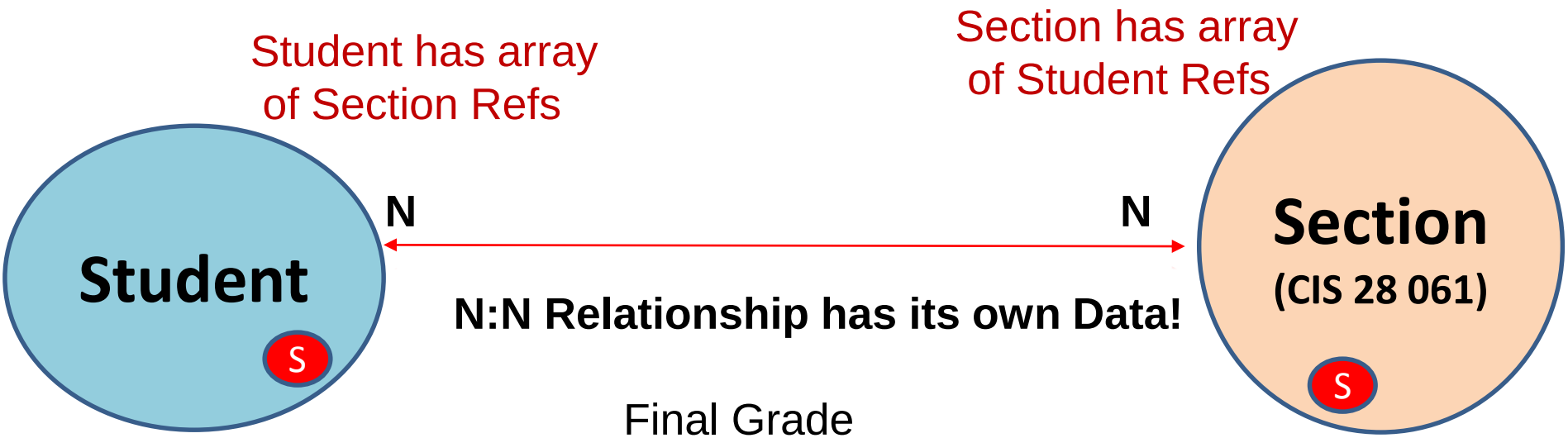
N:N Relationship

How do we implement this?

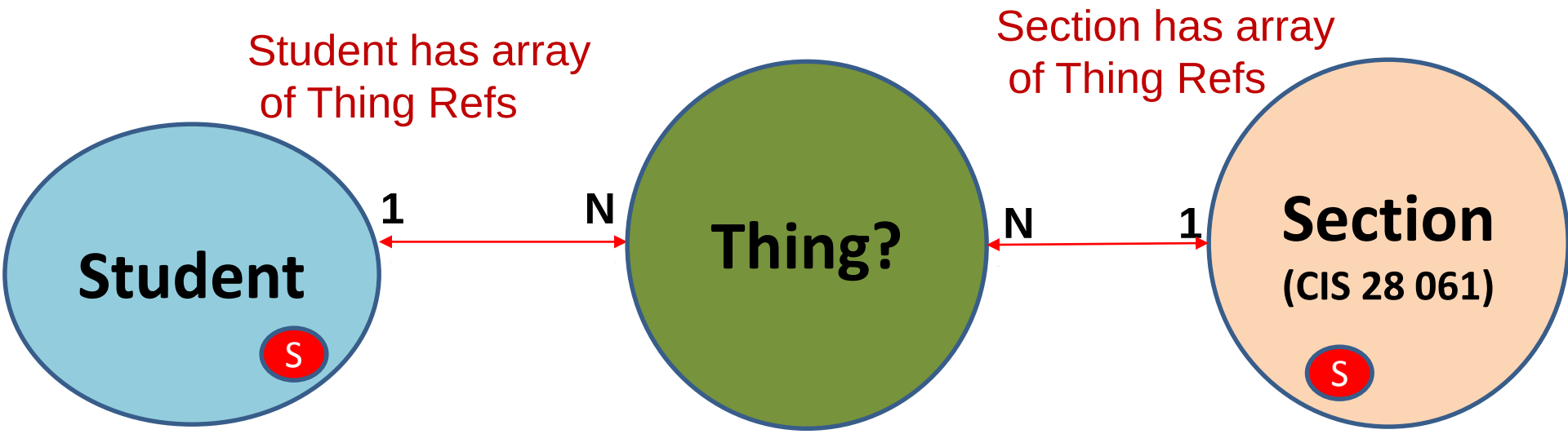
Conceptual Class Diagram: Enrollment



Conceptual Class Diagram: Enrollment



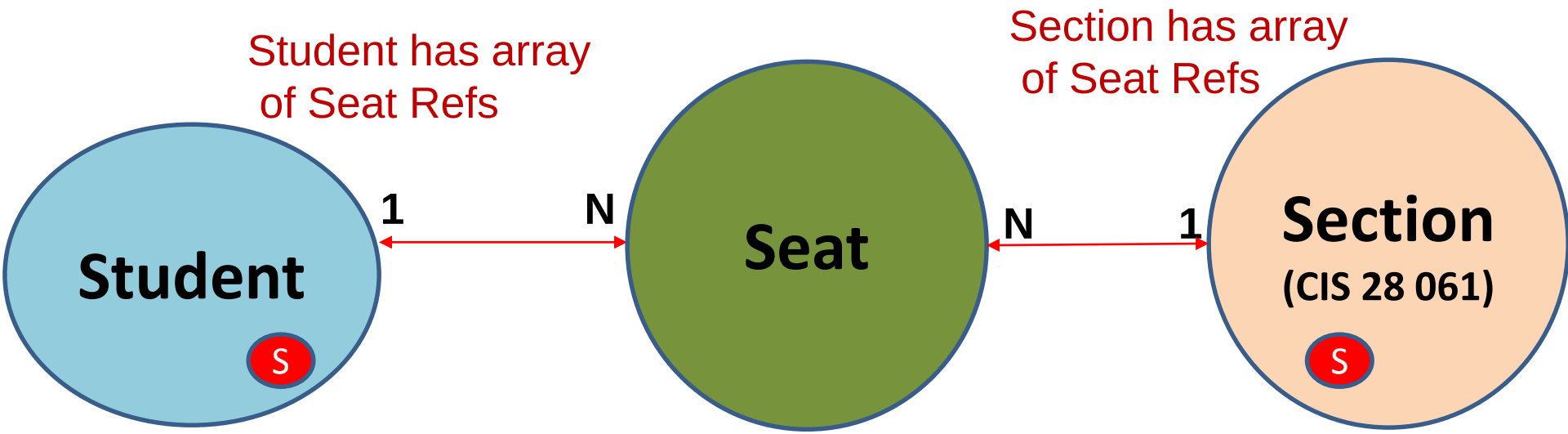
Conceptual Class Diagram: Enrollment



Design Pattern: "Junction Class"

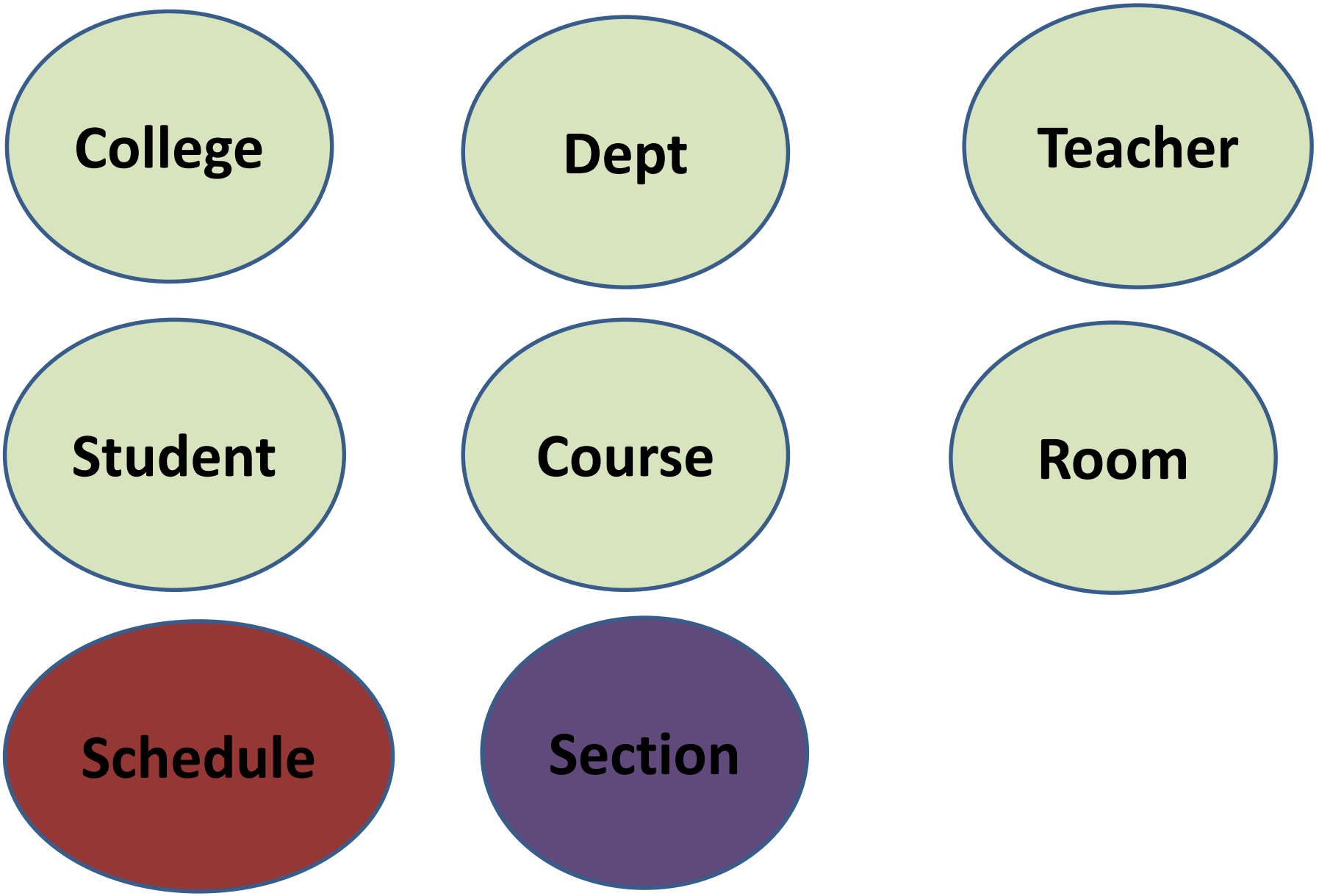
Student Reference
Section Reference
Final Grade
Attendance Records
Exam Grades
Teacher's Comments

Conceptual Class Diagram: Enrollment

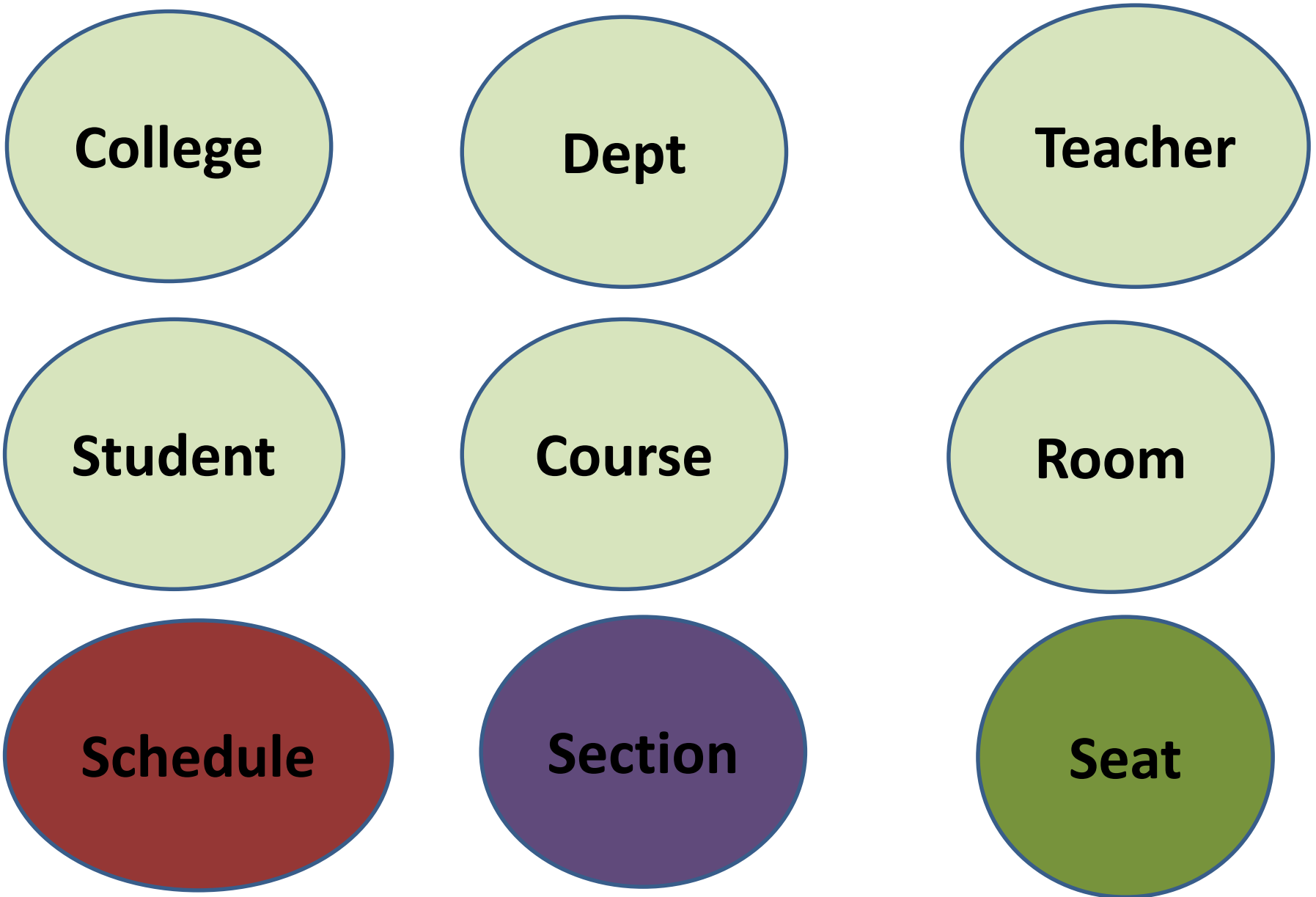


Student Reference
Section Reference
Final Grade
Attendance Records
Exam Grades
Teacher's Comments

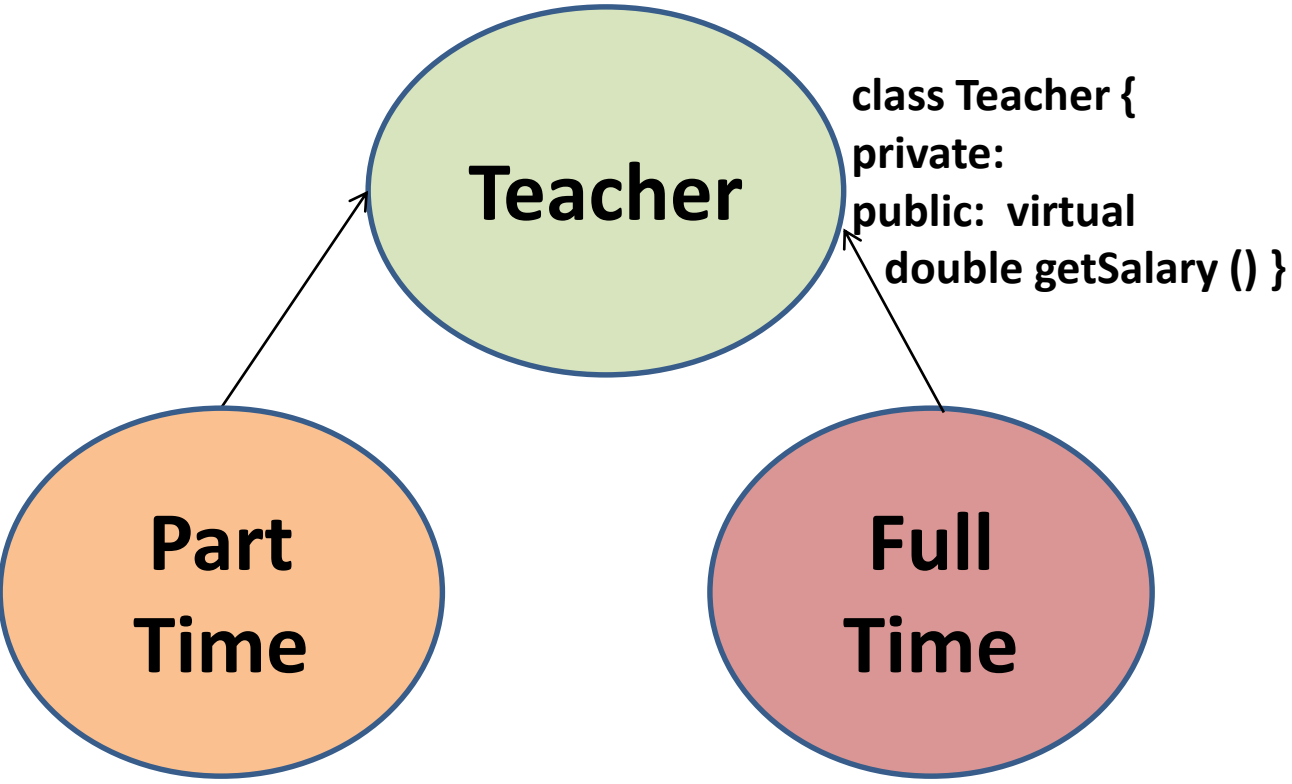
Refined Abstractions in Enrollment System



Refined Abstractions in Enrollment System



Other Possible Objects: Inheritance




```
class Teacher {
private:
public:  virtual
    double getSalary () }
```

```
double PartTime::getSalary ()
{
   code A
 }
```

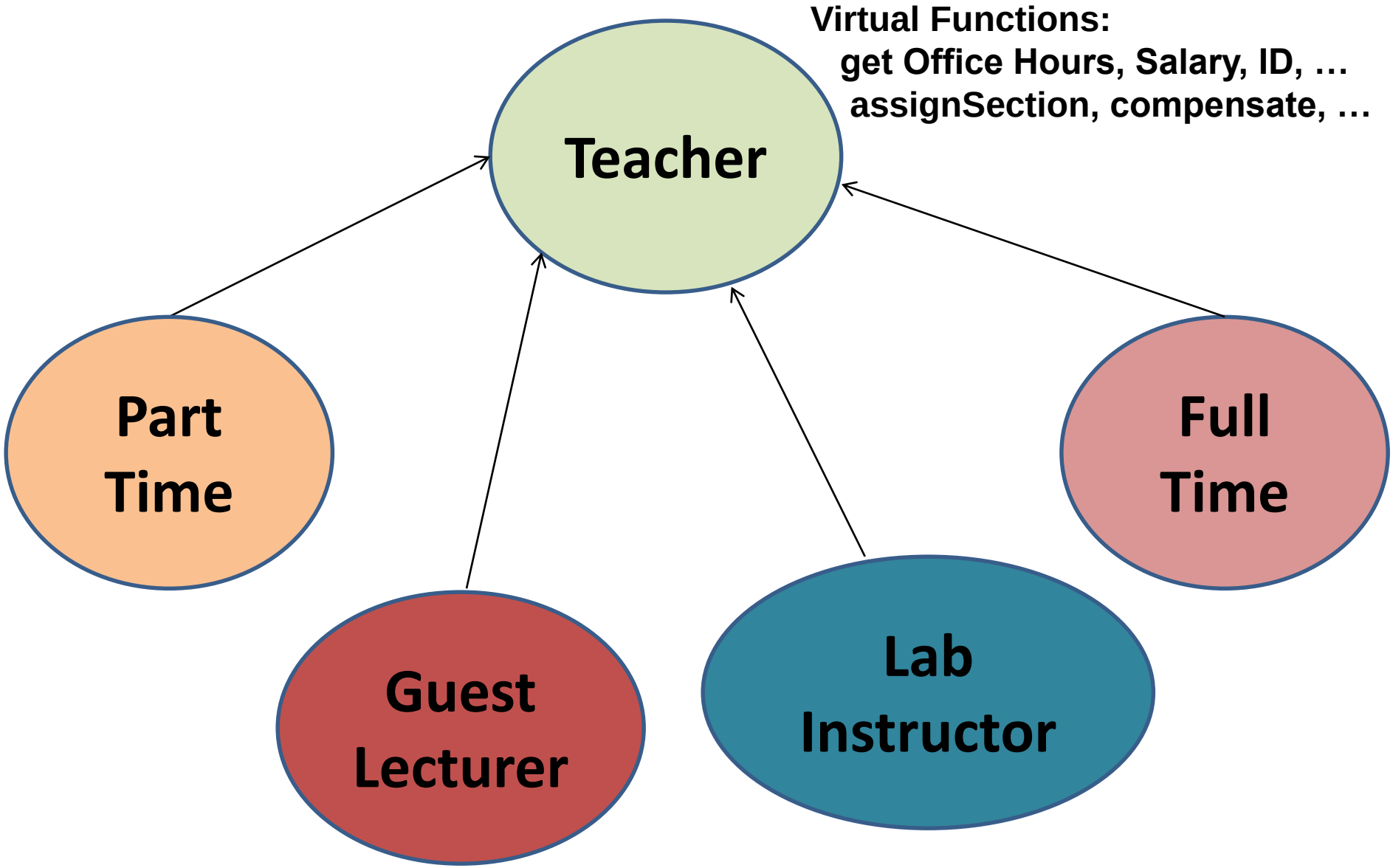
```
double FullTime::getSalary ()
{
   code B
 }
```

Teacher
(Type = PT, FT)

```
double Teacher::getSalary () {
   switch (type)  {
       case PT:
           code A
       case FT:
           code B
   }}
```

Criteria:
- # of known Types
- # of methods affected
- # of "unknown" Types

Parent & Child Classes



No code changes to add "Contractor"!!

Software Architect Responsibilities

- Read, understand and clarify Functional Specification **OOA:**
- Identify basic “abstractions” referenced in the FS
- Determine “relationship” between abstractions (N:1:N)
- **Flesh out abstractions with “attributes” ➔ Class**

➔ *What are the specific data elements that comprise any given object of this class (and are not found in any other class)?*

Ex: What exactly is a “Room”?

“Room” Attributes

- class Room
- { private: // Private Data
- Schedule sched; // Schedule of Room usage (1:1)
- Location *location; // Building or offsite facility (n:1) – *De Anza AT*
- int num; // Room Number
- Department *dept; // Owning Department (n:1) - *CIS*
- int capacity; // Max # of students in the room
- EquipmentList equips; // List of Equipment in Room (1:n)

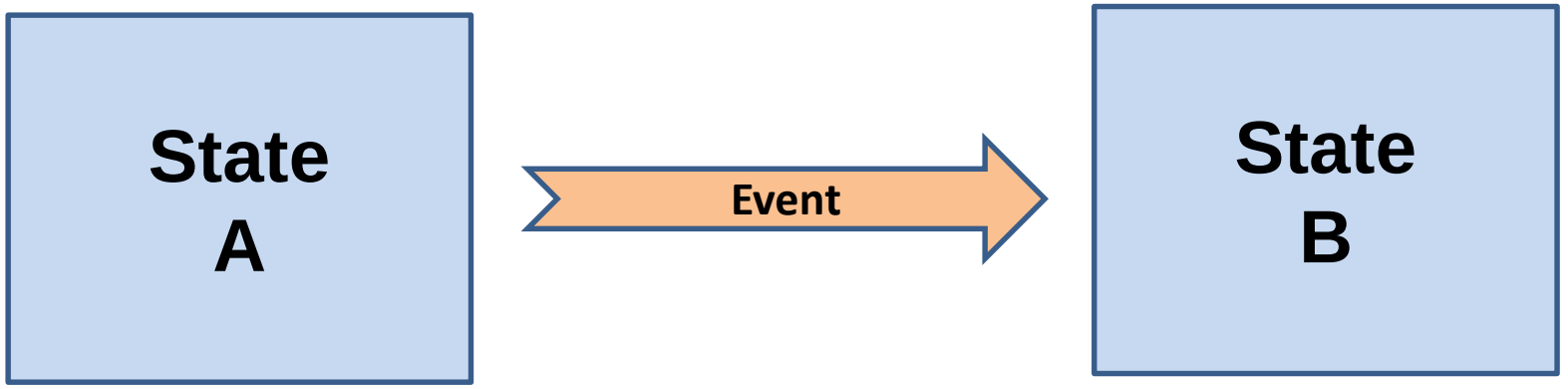
- public: // Public Functions …
- }

```
struct Equipment
{  // This will be used only within Room,
   enum type { OverheadProjector, Whiteboard,  Computer, Dais, InternetAccess };
   int amount;                          // Number of units of equipment;
}
```

Software Architect Responsibilities

- Read, understand and clarify Functional Specification **OOA:**
- Identify basic “abstractions” referenced in the FS
- Determine “relationship” between abstractions (N:1:N)
- Flesh out each abstraction with “attributes” ➔ Class
- **Define the “behavior” of all stateful classes**

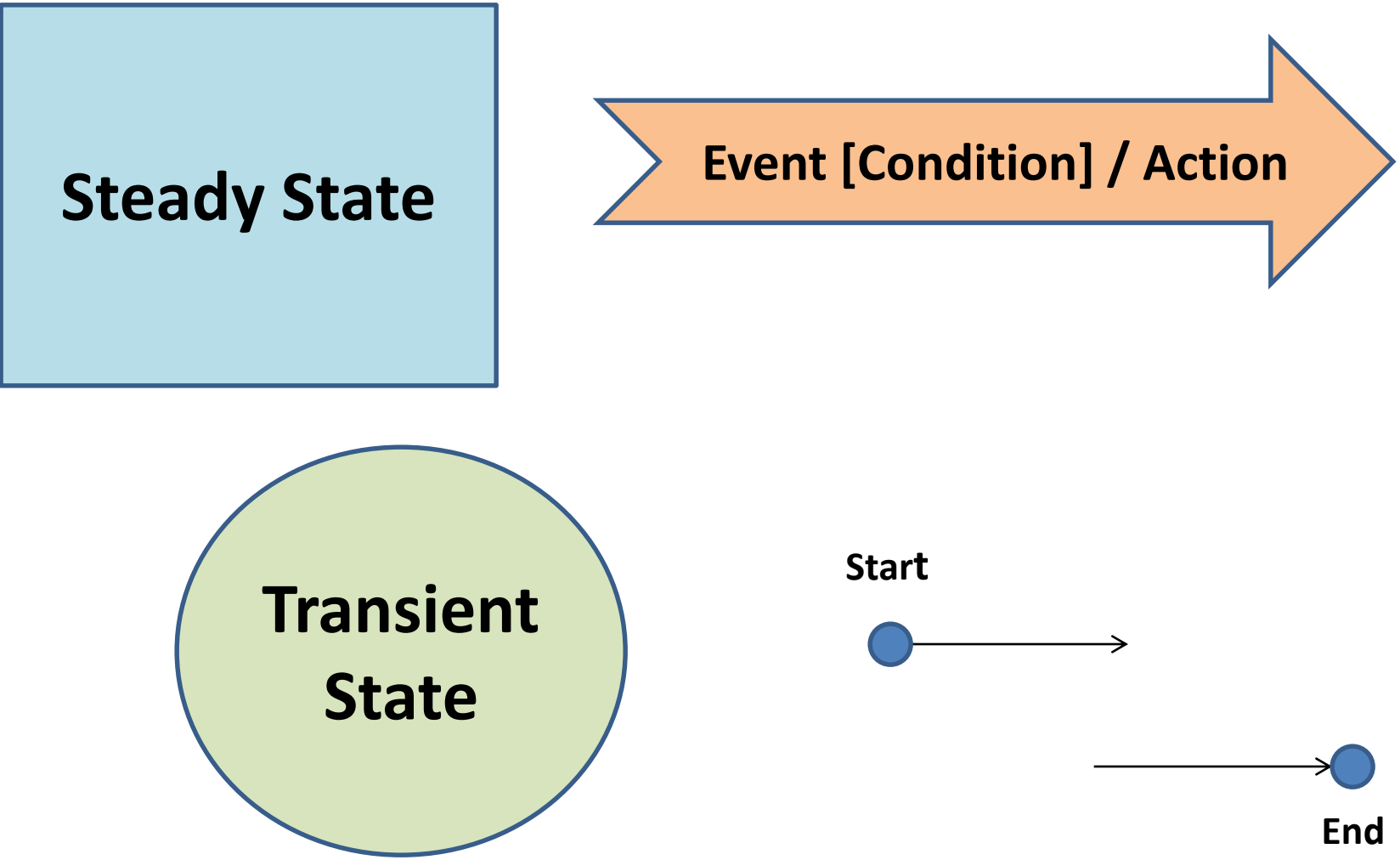
Object Behavior



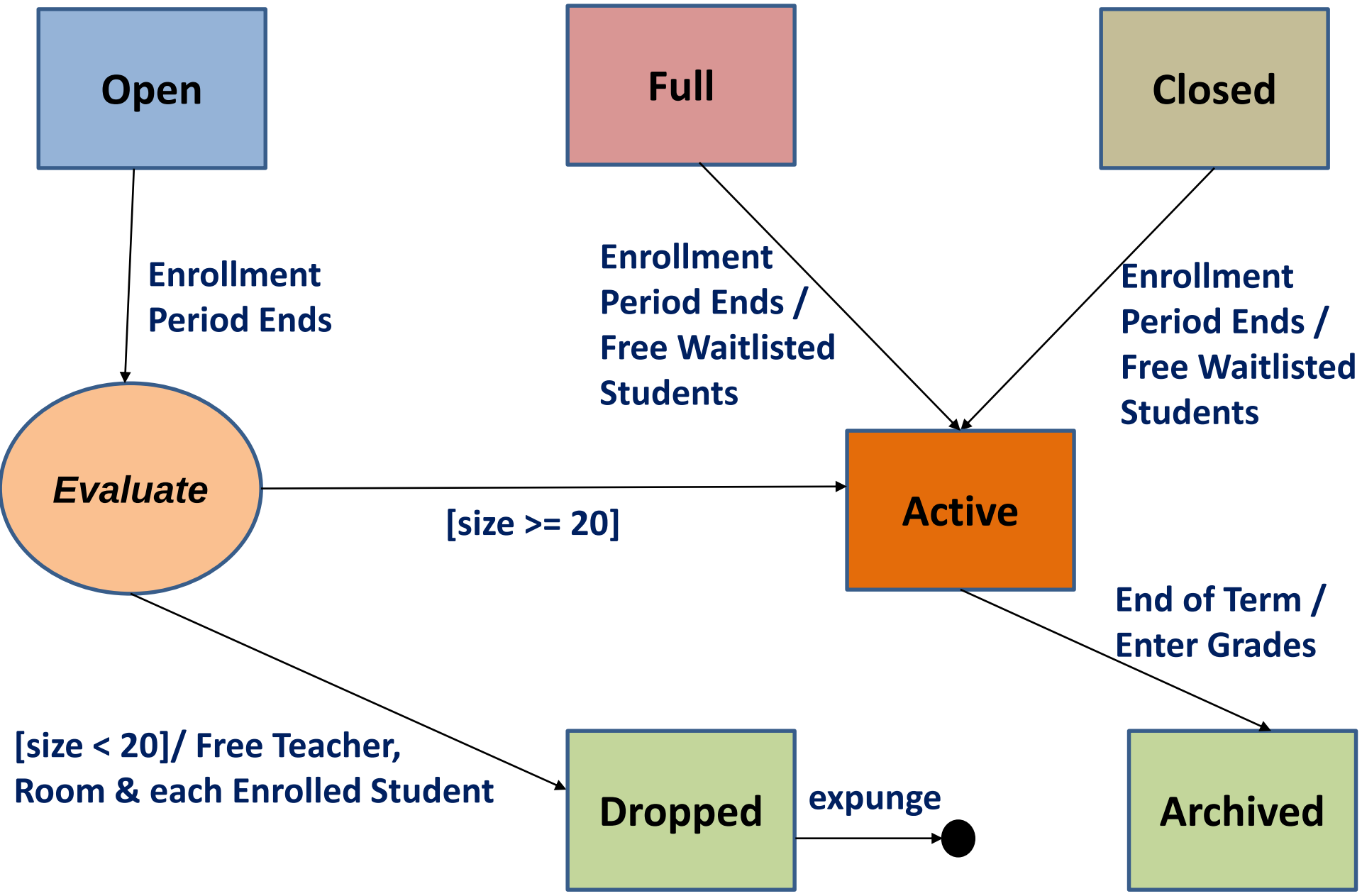
Object "Behavior" in response to a given Event (method invocation) is often dependent upon previous Events, and what "State" they have placed the object into.

- ***"Add Student"* Event for a Section whose "state" is:**
  - **Open**
  - **Full**
  - **Closed**
  - **Cancelled**

State / Event Diagram Components



Section *Behavior* (Partial)



Software Architect Responsibilities

- Read, understand and clarify Functional Specification

OOA:

- Identify basic “abstractions” referenced in the FS
- Determine “relationship” between abstractions (N:1:N)
- Flesh out each abstraction with “attributes” ➔ Class
- Define the “behavior” of all stateful classes

OOD:

- **Document “Use Cases” from the Functional Specification utilizing the set of defined classes**

FS: Enrollment System at De Anza College

De Anza **College Courses** are offered by its **Departments** (Ex: CIS) and available Quarterly (Ex: Fall and Winter). Each Course has an identifying number (ex: 28), a Name (Ex: Object Oriented Analysis and Design), a description, a number of credits and an optional set of required prerequisites.

Each offered course is "instantiated" by one or more "**Sections**". Each Section is assigned a Section number (ex: 061), and a **Schedule** of times during the week when it meets. It is also assigned a **Teacher** who is both qualified and willing to teach that course, and who has no Schedule conflicts with it. It is also assigned a **Room**, which also must be free during those times the Section meets.

Students may then attempt to enroll in a Section if they are paid up, have taken all its required prerequisites, are not currently enrolled in any Section of the course, and if they are not already enrolled in another Section with a conflicting Schedule .

Depending upon the size of the Section and the number of Students already enrolled, the Student's enrollment request may either be accepted, or the Student may be wait listed, or the request may be denied. If the Student is accepted, her attendance will be tracked and at the end of the academic quarter she will receive a final grade for the Section.

Use Cases:
What the Enrollment Application must do

- Create new Section Y (#, days-time) of Course C
- Assign Teacher T to Section Y
- Assign Room R to Section Y
- **Enroll Student X in Section Y**
- End of Enrollment Period (includes Section Drop)
- End of Term

Use Case: Student X Enrolls in Section Y

- **Restrictions:**
  - Section Y must have open slots
  - Student X must be paid up (tuition)
  - Section Y's Schedule must not conflict with the Schedule of any other Sections that Student X is enrolled in
  - Student X must have successfully completed all prerequisite Courses
- **Effects (on success):**
  - Seat Z is created which connects Student X and Section Y
  - Section Y's Schedule is "*added*" to Student X's Schedule

Enroll Student in Section

Web Server

Application Server

Browser

Tablet

Registrar Desk

Smart Phone

User Facing Servlet Thread

User Facing Servlet Thread

User Facing Servlet Thread

User Facing Servlet Thread

Enrollment Application Back End Interface

CLIENT

OBJECT APIS

Enrollment Application Service Object Library

(Implementations)

(*User ID, Password, Student ID, Section ID*)

(Enroll: *Student ID, Section ID*)

College::getStudent (ID)
College::getSection (ID)
Section::enroll (Student&)

Open (Internet)

Protected (Data Center/Cloud)

Software Architect Responsibilities

- Read, understand and clarify Functional Specification

OOA:

- Identify basic “abstractions” referenced in the FS
- Determine “relationship” between abstractions (N:1:N)
- Flesh out each abstraction with “attributes” ➔ Class
- Define the “behavior” of all stateful classes

OOD:

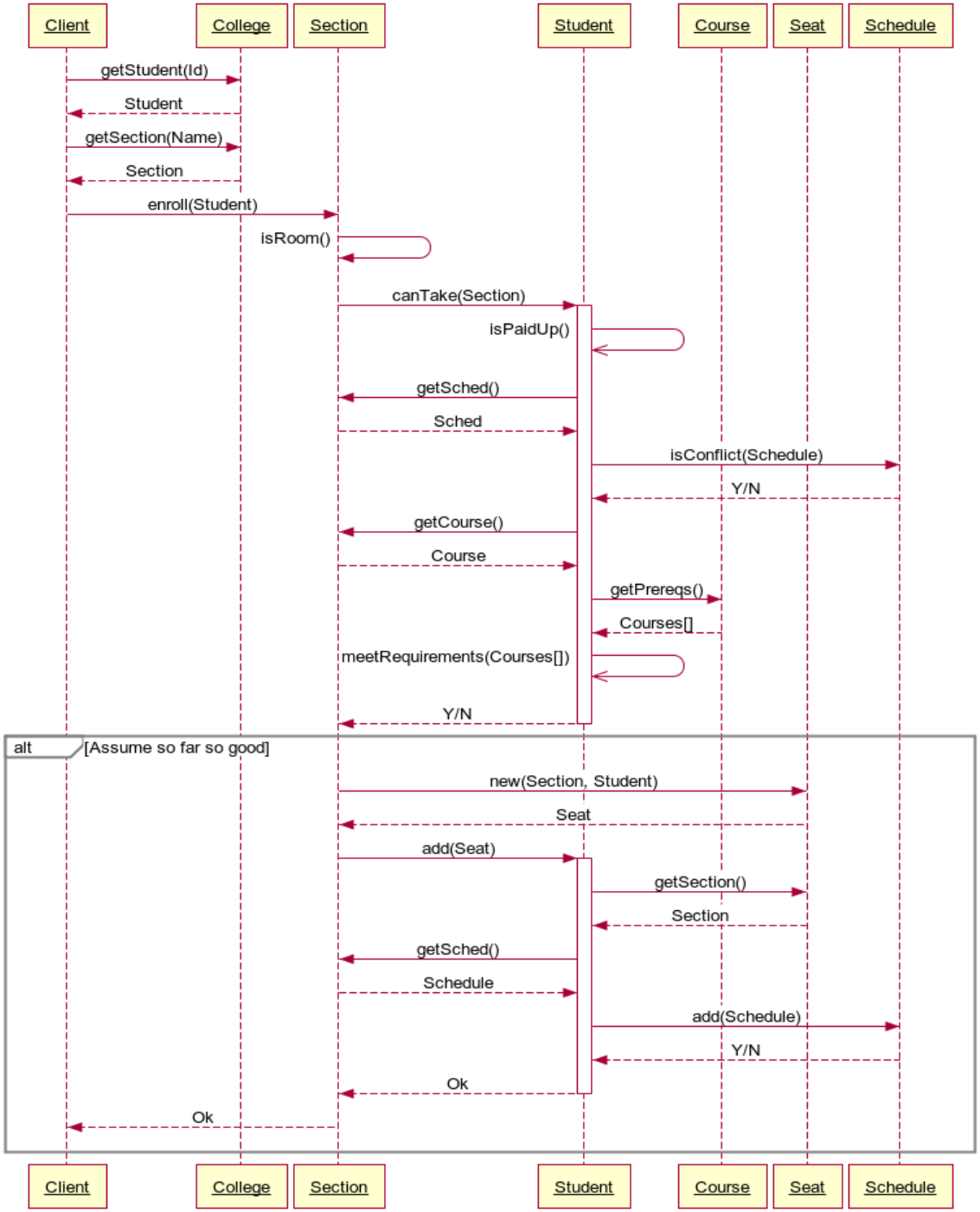
- Document “Use Cases” from the FS utilizing the set of defined classes
- **“Solve” each Use Case via a Sequence Flow of Process Orchestration among defined objects**

UML Artifact

Process Orchestration

Enroll Student X in Section Y

(Only Student ID and Section # Strings are known)



www.websequencediagrams.com

Software Architect Responsibilities

- Read, understand and clarify Functional Specification (FS)

OOA:

- Identify basic “abstractions” referenced in the FS
- Determine “relationship” between abstractions (N:1:N)
- Flesh out each abstraction with “attributes” ➔ Class
- Define the “behavior” of all stateful classes

OOD:

- Document “Use Cases” from the FS utilizing the set of defined classes
- “Solve” each Use Case via a Sequence Flow of Process Orchestration among defined objects
- **Generate the responsibilities and collaborations required of each Class (CRC Object Choreography)**

Class Responsibility Collaboration (CRC)

- Choreography of what one class must do to support **ALL** sequence flows mapping out **ALL** use case process orchestrations.
- Defines all required Class public methods (*the complete object interface!*) and dependencies on other objects (*how to implement them*).
- Every Collaboration must be a Responsibility of some other indicated Class
- Vital for conveying design results to object developers / programmers
- The following is a partial Student CRC which covers only responsibilities in "*Enroll Student in Section*".

Class: Section

Responsibilities

- enroll (Student)
- isRoom ()
- Sched& getSched ()
- Course& getCourse ()

Collaborations

- Section::isRoom ()
- Student::canTake (Section&)
- Seat::new (Student, Section)
- Student::addSeat (Seat)

Class: Student

Responsibilities	Collaborations
• canTake(Section)	• Student:: isPaidUp () • Section::getSchedule () • Section::getCourse () • Schedule::isConflict (Schedule &) • Student::meetRequirements (Courses[])
• meetReqs (Courses[]) • isPaidUp ()	• -- (Checks if all Courses were taken) • --
• addSeat (Seat)	• Seat::getSection () • Section:getSched () • Schedule::add (Schedule &)

Software Architect Responsibilities

- Read, understand and clarify Functional Specification (FS)

OOA:

- Identify basic “abstractions” referenced in the FS
- Determine “relationship” between abstractions (N:1:N)
- Flesh out each abstraction with “attributes” ➔ Class
- Define the “behavior” of all stateful classes

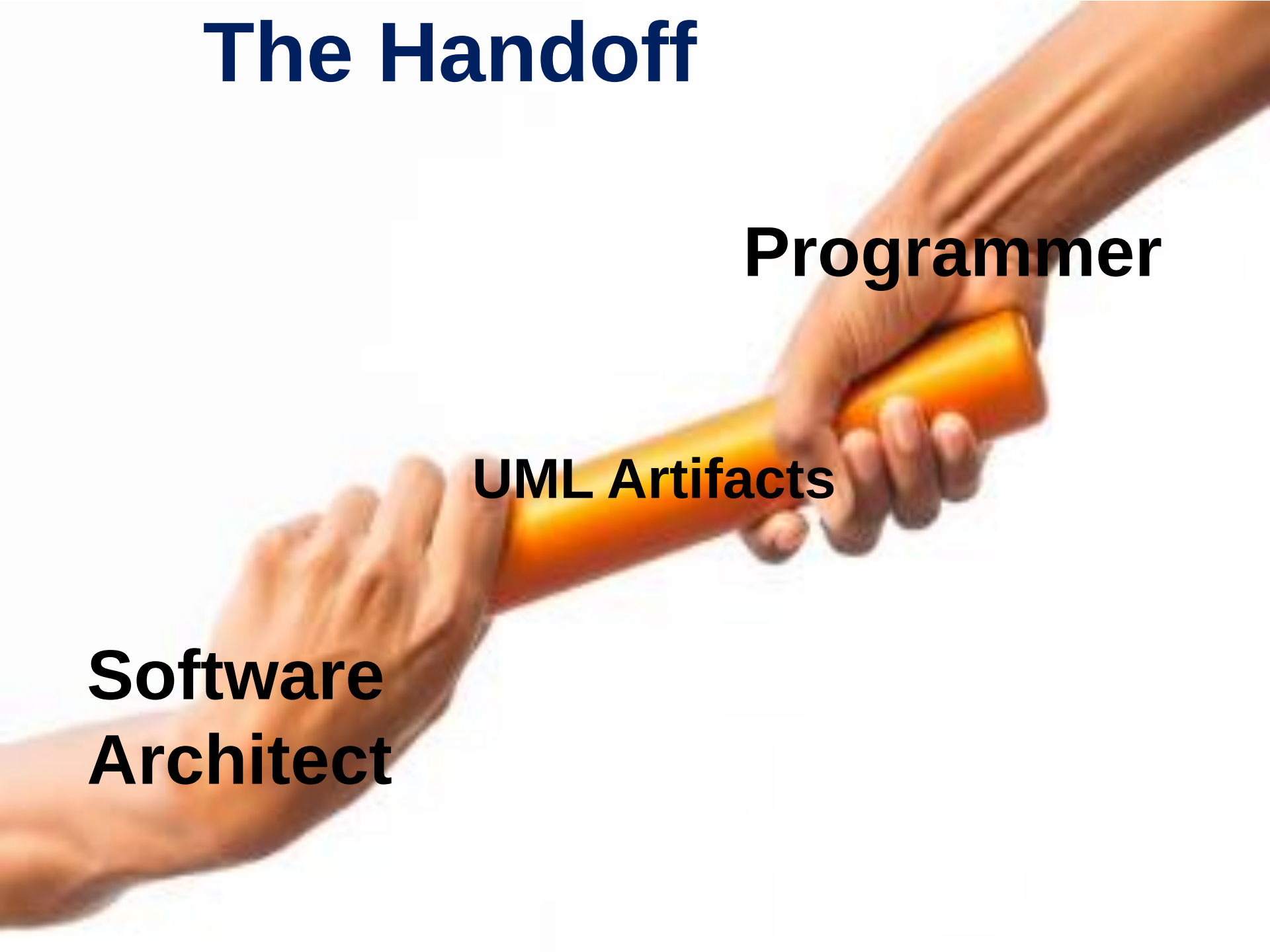
OOD:

- Document “Use Cases” from the FS utilizing the set of defined classes
- “Solve” each Use Case via a Sequence Flow of Process Orchestration among defined objects
- Generate the responsibilities and collaborations required of each Class (CRC Choreography)

OOP:

- **Bring in the Software Developers and Programmers**

The Handoff



Handing off to the Programmers

- Conceptual Class Diagram referencing clearly defined “abstractions” mapping to C++/Java Classes

Handing off to the Programmers

- Conceptual Class Diagram referencing clearly defined “abstractions” mapping to C++/Java Classes
- List of attributes for each class
  - Good start at defining private data elements

Handing off to the Programmers

- Conceptual Class Diagram referencing clearly defined “abstractions” mapping to C++/Java Classes
- List of attributes for each class
  - Good start at defining private data elements
- State / Event Diagrams defining object “behavior”

Handing off to the Programmers

- Conceptual Class Diagram referencing clearly defined “abstractions” mapping to C++/Java Classes
- List of attributes for each class
  - Good start at defining private data elements
- State / Event Diagrams defining object “behavior”
- Use Case description for every supported “process”

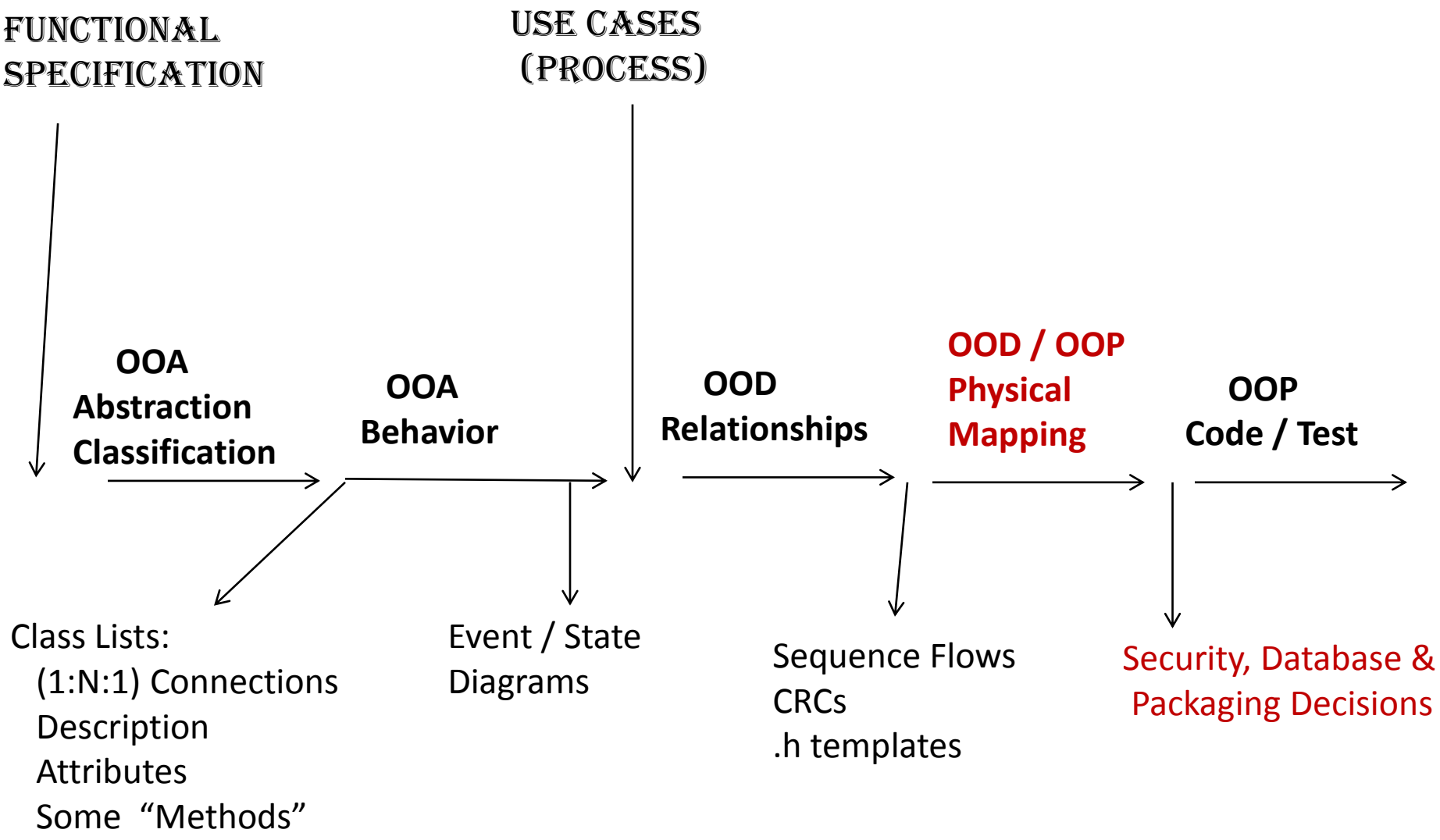
Handing off to the Programmers

- Conceptual Class Diagram referencing clearly defined “abstractions” mapping to C++/Java Classes
- List of attributes for each class
  - Good start at defining private data elements
- State / Event Diagrams defining object “behavior”
- Use Case description for every supported “process”
- Sequence Flow Orchestration for each such process

Handing off to the Programmers

- Conceptual Class Diagram referencing clearly defined “abstractions” mapping to C++/Java Classes
- List of attributes for each class
  - Good start at defining private data elements
- State / Event Diagrams defining object “behavior”
- Use Case description for every supported “process”
- Sequence Flow Orchestration for each such process
- Class Responsibility / Collaboration Choreographies
  - Responsibilities map 1:1 with required public methods
  - Collaborations indicate object methods that need to be invoked to implement these responsibilities

OO Phases


FUNCTIONAL SPECIFICATION
USE CASES (PROCESS)
OOA Abstraction Classification
OOA Behavior
OOD Relationships
OOD / OOP Physical Mapping
OOP Code / Test
Class Lists:
(1:N:1) Connections
Description
Attributes
Some "Methods"
Event / State Diagrams
Sequence Flows
CRCs
.h templates
Security, Database & Packaging Decisions

So what's left to do?

- **The application architecture is completely defined.**
  - Exactly what each object must do to support use cases

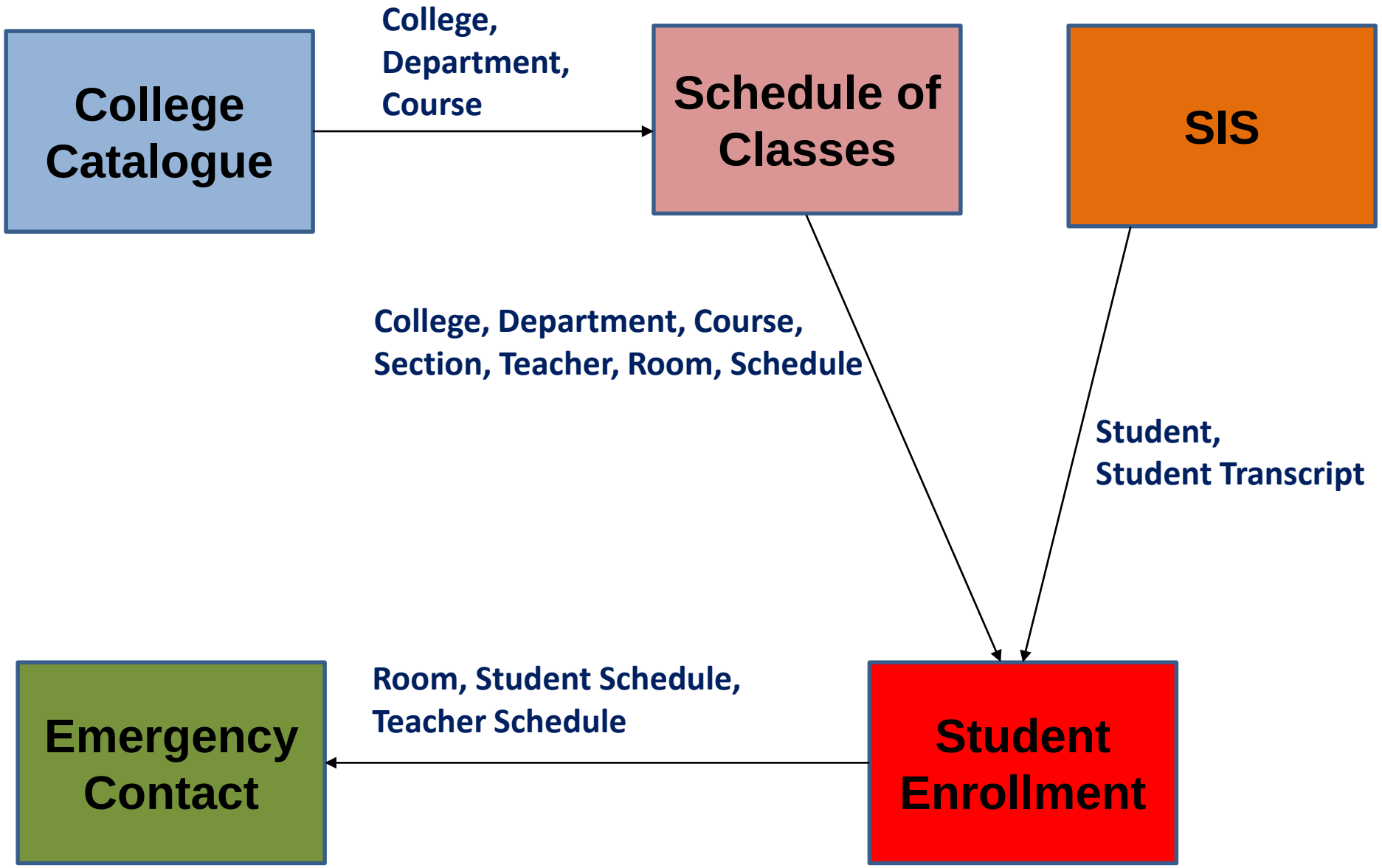
So what's left to do?

- **The application architecture is completely defined.**
  - Exactly what each object must do to support use cases
- **Next come the "Application Framework" decisions**
  - What API to access the Data Store (Oracle, Hadoop, SQL)?
  - What OS will be used (Linux, Windows, X, …)?
  - What Security strategy will be adopted
    - Authentication, Authorization, Encryption?
  - Distributed deployment (Data Center, Cloud)?

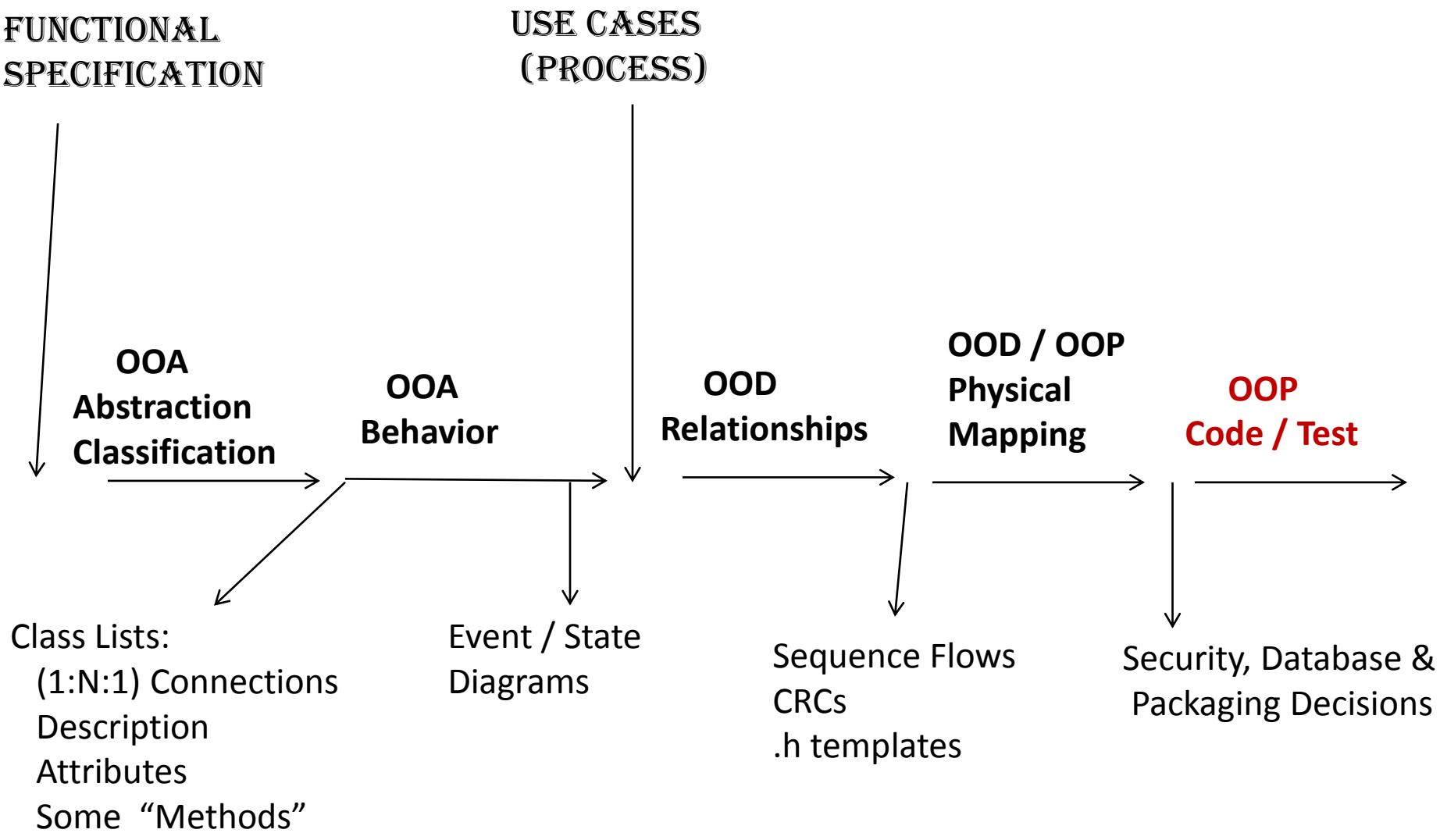
So what's left to do?

- **The application architecture is completely defined.**
  - Exactly what each object must do to support use cases
- **Make the "Application Framework" decisions**
  - What API to access the Data Store (Oracle, Hadoop, SQL)?
  - What OS will be used (Linux, Windows, X, …)?
  - What Security strategy will be adopted
    - Authentication, Authorization, Encryption?
  - Distributed deployment (Data Center, Cloud)?
- **Determine the "Packaging"**
  - Define relationship to other apps in the Organization

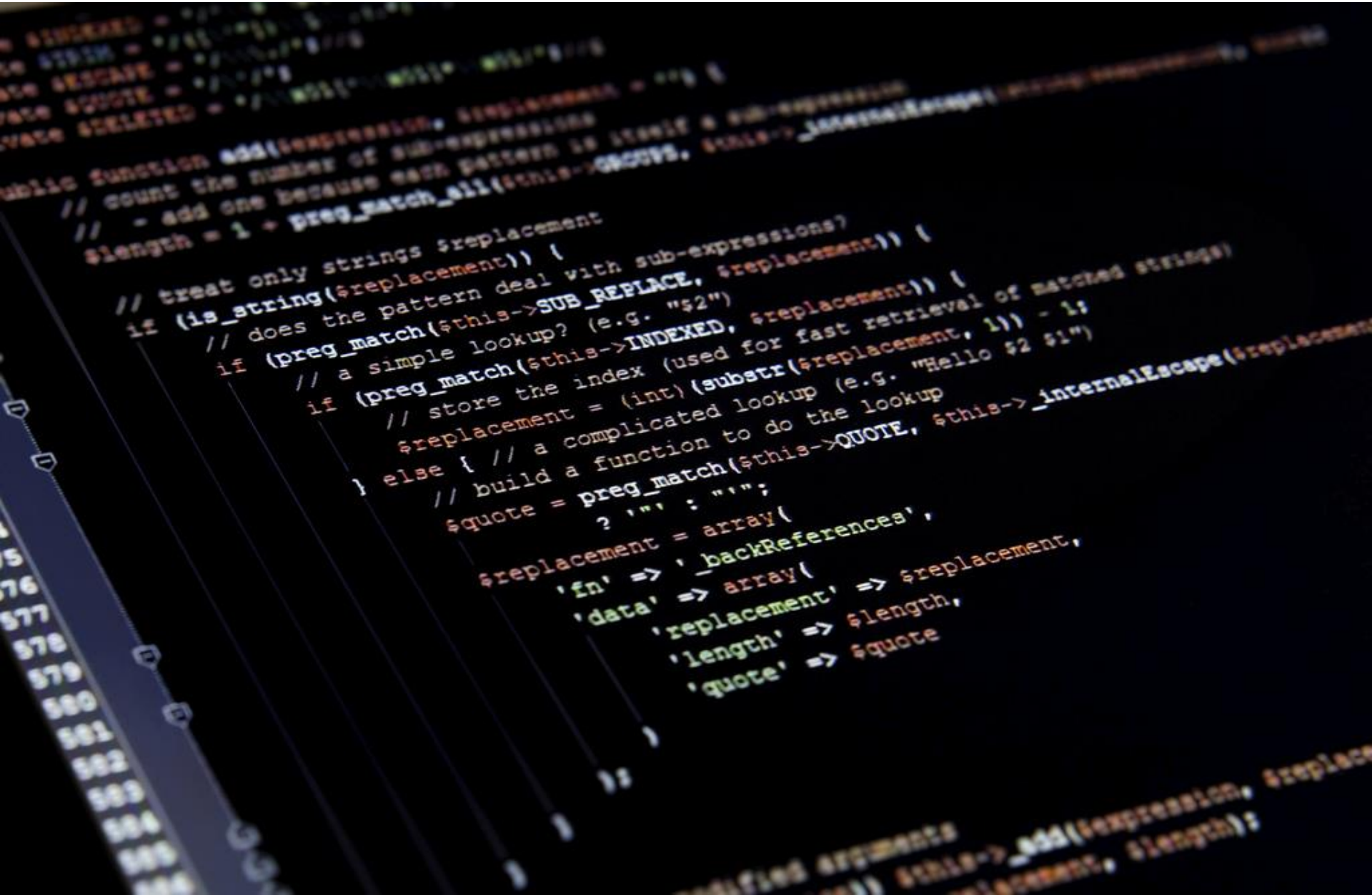
De Anza Software Applications



OO Phases


FUNCTIONAL SPECIFICATION
USE CASES (PROCESS)
OOA Abstraction Classification
OOA Behavior
OOD Relationships
OOD / OOP Physical Mapping
OOP Code / Test
Class Lists:
(1:N:1) Connections
Description
Attributes
Some "Methods"
Event / State Diagrams
Sequence Flows
CRCs
.h templates
Security, Database & Packaging Decisions

Now!!!


// treat only strings $replacement
if (is_string($replacement)) {
// does the pattern deal with sub-expressions?
if (preg_match($this->SUB_REPLACE, $replacement)) {
// a simple lookup? (e.g. "$2")
if (preg_match($this->INDEXED, $replacement)) {
// store the index (used for fast retrieval of matched strings)
$replacement = (int)(substr($replacement, 1)) - 1;
} else { // a complicated lookup (e.g. "Hello $2 $1")
// build a function to do the lookup
$quote = preg_match($this->QUOTE, $this->_internalEscape($replacement)
? '"' : "'";
$replacement = array(
'fn' => '_backReferences',
'data' => array(
'replacement' => $replacement,
'length' => $length,
'quote' => $quote

How to become a Software Architect

- **Software Developer**
  - Start with OOP programming skills
- **Software Designer**
  - Add in OOD and knowledge of Design Patterns
- **Software Architect**
  - Add in OOA and specific “Domain” knowledge

http://www.softwarearchitectures.com/career.html

We've covered a lot of ground in a very short time

CIS 28 (OOA & OOD) has further details

Questions?