

Systems Development

Preliminary Design

<http://lbgeeks.com/gitc/systemsPrel.php>

April 16, 2007

- **What is Design?**
- **Design Sub Phases**
- **Object Oriented Concepts**
- **Implementing OO Concepts**
- **Documenting OO Designs**
- **Schedule Reminders**

What is Design?

- **Answers a different development question:**
 - Requirements ask “What do we build?”
 - Design answers “How do we build it?”
- **Application of requirements to a specific solution set**
- **Integrates elements of this set to perform functions within performance limits**
- **Framework for system growth:**
 - Error detection and correction
 - Addition of new functions

Design Sub Phases

- **Preliminary design:**
 - Captures general organization
 - External representation of design elements
 - Transaction focus
- **Detailed design:**
 - Sufficient details for coding
 - Internal representation of design elements
 - Function focus
- **Splitting design into phases reduces errors:**
 - High level view spotlights interface errors
 - Verify on paper no missing elements

Object Oriented Concepts

- **Encapsulation**
- **Inheritance**
- **Data Abstraction**
- **Polymorphism**

Encapsulation

- **Data + function = object**
- **Grouping of the inputs, activities, and outputs necessary to perform task**
- **Class is a compile-time static collection**
- **Object is a run-time instantiation**
- **Must maintain state:**
 - **Object controls its own destiny**
 - **Responds properly to erroneous input**
 - **Never performs a “dangerous” transaction**

- **Coding the problem intersection, not union**
- **Specify deltas between source and target**
 - **Stack is “Last In-First Out”**
 - **Queue is “First In-First Out”**
 - **Deltas in “push” method, but not “pop”**
- **Supports reuse of tested code**
- **Improper correlation results in misuse**

Data Abstraction

- **Separation of responsibility between client and server**
- **Public interface for using the object**
- **Private interface for development**
- **General principles:**
 - **No public access to member data**
 - **Provide uniform interface to different data types**

Polymorphism

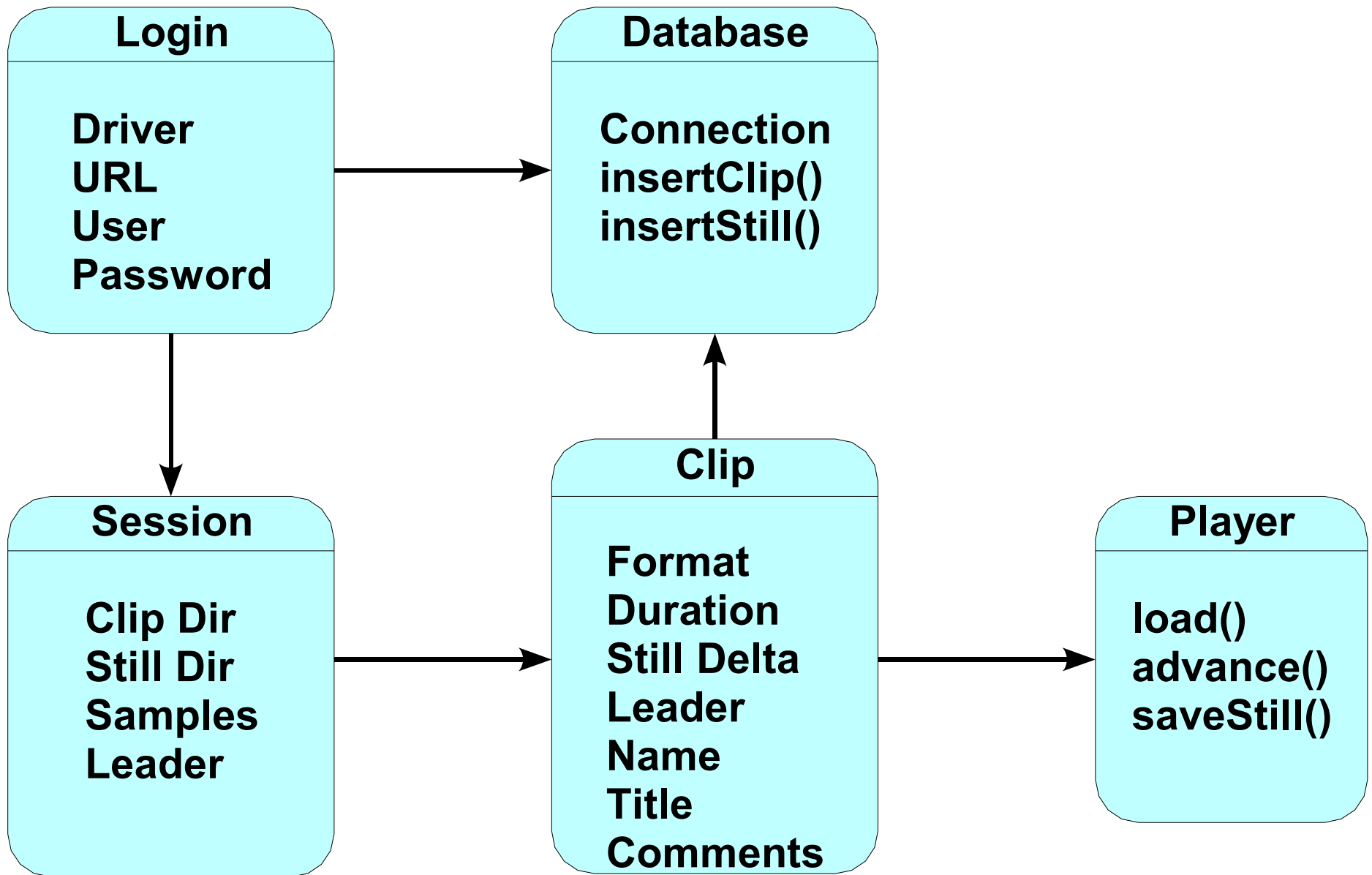
- **Same form exhibits different behaviour**
- **Example of text vs. password field**
- **Base object instantiated from different derived classes**
- **Run time system invokes appropriate code**

Documenting An OO Design

- **Object Models**
- **Sequence Diagrams**
- **Object Interaction Diagrams**
- **Trace Matrix**
- **Pseudo Code**

- **Diagram showing classes and associations**
 - **Rounded rectangles are classes**
 - **Solid lines are associations**
- **Name of class and attributes, methods recorded on diagram**
- **Words describe nature of association**
- **Two primary association types:**
 - **“Has-A” specifying containment**
 - **“Is-A” specifying inheritance**

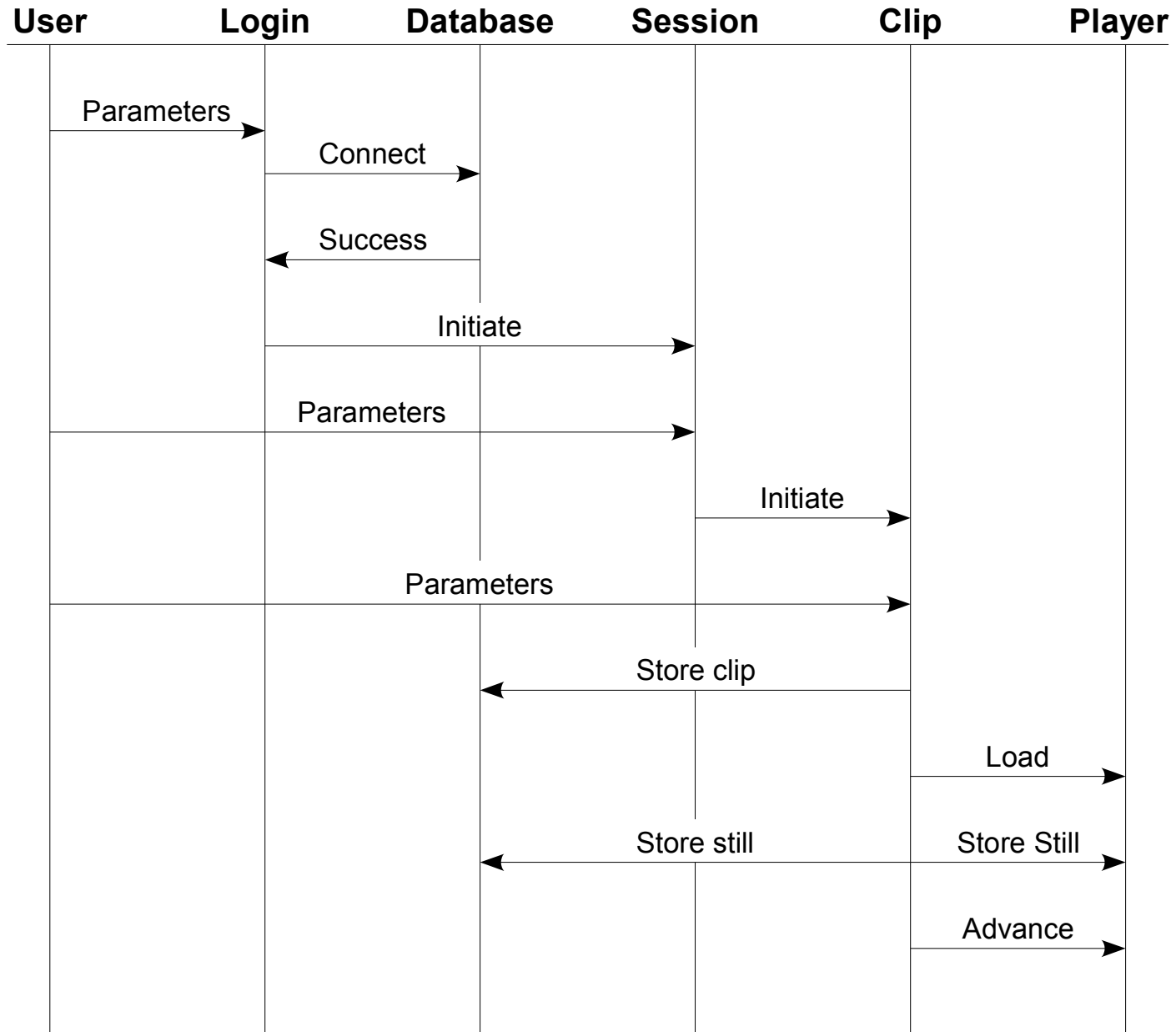
Object Model Example



Sequence Diagrams

- **Transaction execution flow among**
- **Data exchange among objects**
- **Processing steps within object**
- **Time ordered top to bottom**
- **Shows co-operation and dependency**

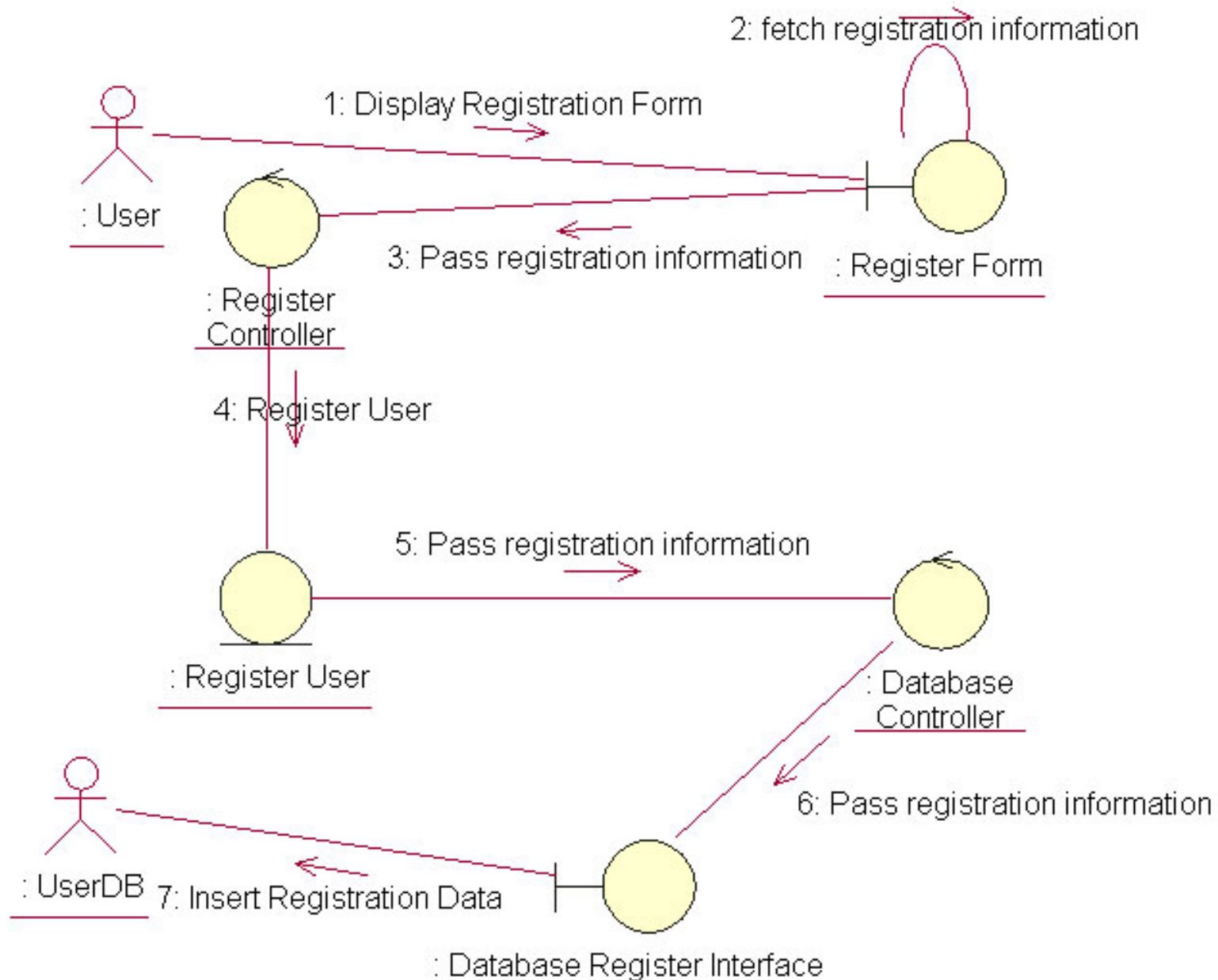
Sequence Diagram Example



Object Interaction Diagrams

- **Next level of detail under object diagrams**
- **Incorporates information from sequence diagram as well**
- **Shows order of method invocations among object in performing a transaction**
- **Complete specification for executives (methods that call other methods)**

Object Interaction Diagram Example



- **Requirements have unique identifier**
- **One requirement inspires multiple items during design**
- **Document “what to how” mapping:**
- **Spreadsheet or table with several columns:**
 - **Requirement ID**
 - **Design item type**
 - **Design item name**
 - **Purpose or reason**

Pseudo Code

- **Final step before implementation**
- **Used to organize algorithms, expressions**
- **Natural language expression of code**
- **Free form syntax may be imprecise**
- **Several steps may be represented by a single sentence**
- **Enough detail that experienced coder can generate computer language statements**

Implementing OO Concepts

- **Permanent and Temporary Storage**
- **Encapsulation**
- **Data Abstraction**
- **Inheritance and Polymorphism**

Permanent and Temporary Storage

- **Access front end and Oracle back end**
- **Permanent data stored in Oracle**
- **Temporary data stored in form controls or Visual Basic (VB) variables**
- **Exceptions:**
 - **Scratch tables created in Jet table space**
 - **Variables in PL/SQL code**

- **Methods as Access macros or VB code**
- **Run time attributes in VB**
- **Permanent attributes in Oracle tables**

- **Variant data type and test functions:**
 - **IsDate()**
 - **IsNull()**
 - **IsNumeric()**
 - **IsObject()**
- **Hidden controls in forms: visible property set to false**

Inheritance and Polymorphism

- **Base data in common table**
- **Link inherited data via foreign key**
- **Populate control or open form at run time using VB If statement**

Class Schedule Reminders

- **Sessions at 19:00:**
 - **Wednesday, April 18**
 - **Monday, April 23**
 - **Wednesday, April 25**
- **Next Unit Starts April 30**
 - **Quiz at 16:45**
 - **Assignment due at start of class**