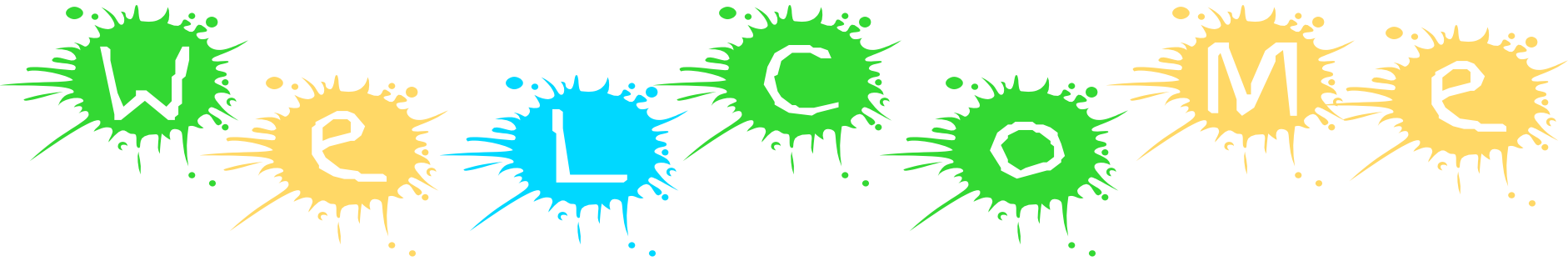




# Plex for 2E Developers





- Introduction of Plex to 2E Developers

# Connection Info

---

- CAPlex2E1 wireless password
  - plex2e2011
- Remote Desktop
- Administrator
- Cmtech2010

# 2E/Plex Comparative Anatomy

---

- CA 2E developers will feel at home with many of CA Plex's core concepts
- CA Plex expands and enhances CA 2E's essential concepts
- CA 2E skills translate well into CA Plex skills

- **Data modeling** - both 2E and Plex implement a data-driven approach in exactly the same way.
- **Modeling language** - Plex allows developers to describe their data models using exactly the same grammar and verbs as 2E (refers to, owned by, known by and has). Plex builds on and extends this simple and effective language.
- **Design objects** - Plex uses most of the same design objects as 2E - entities, functions, fields, views and so on. Plex has additional object types to support functionality not provided by 2E.
- **Action diagrams** - Plex uses action diagrams to describe procedural logic in a manner very similar to 2E. “

CA 2E	CA Plex
Field Attribute	Field
File Attribute	Entity and Table
Access Path Type	View
Function Type	Function
Application	Subject Area
File	Entity
Access Path	View
Field	Field
Field Condition	Value(s) and State
Constant	Value and State
Arrays	Entity and View
Function	Function and Source Code
Device Design (Screens)	Panel
Message	Function, Message and Second Level Text Topic
Narrative	Narrative

**ce**

# Model Based Development

# What Is A Group Model?

---

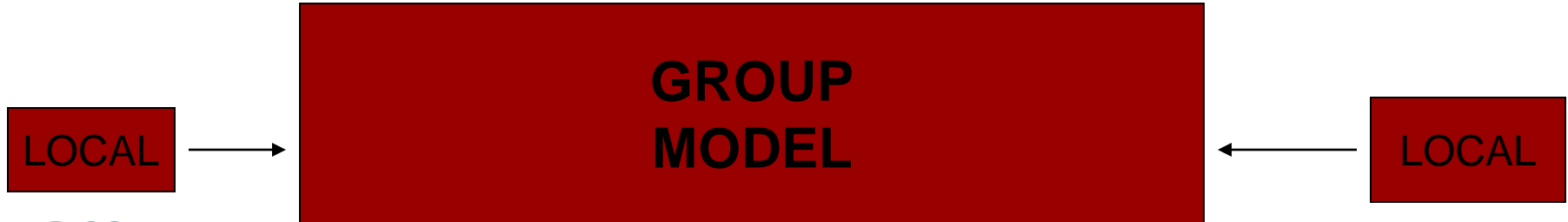
- Group Model – master repository for all information about a Plex application

**GROUP MODEL**

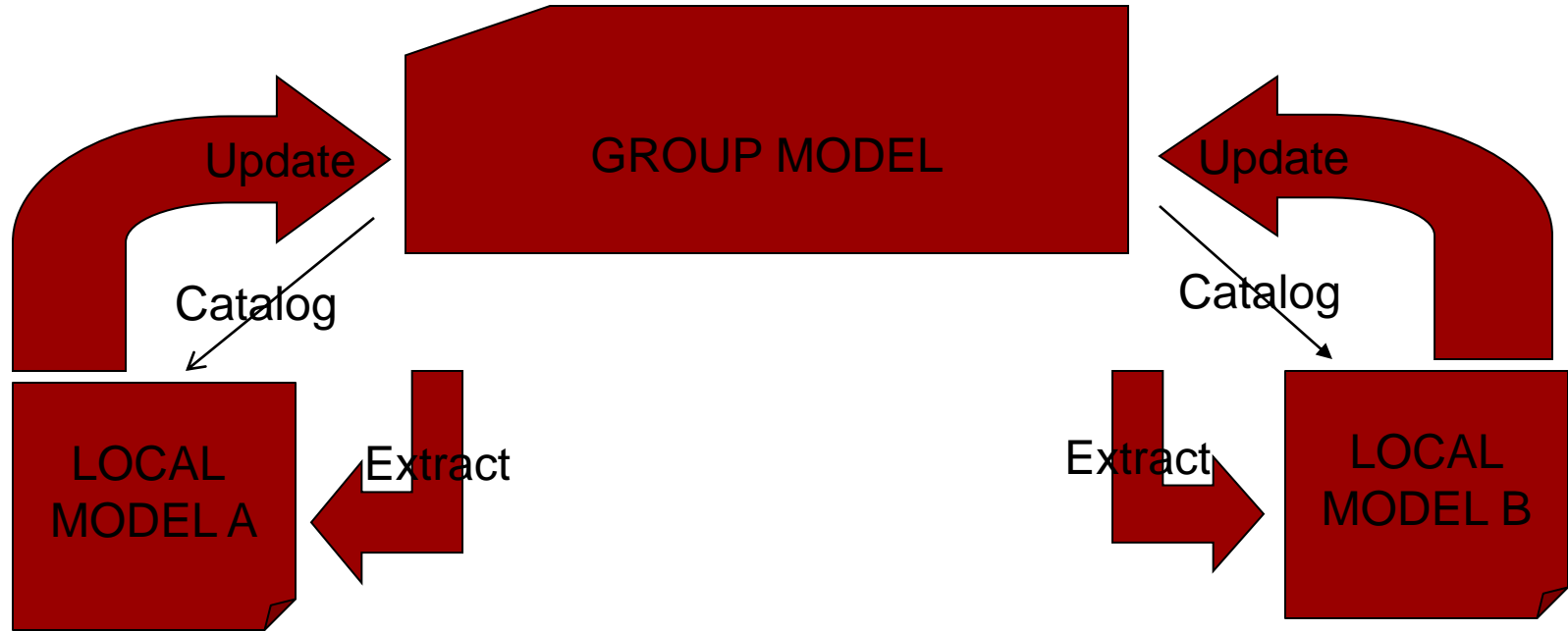


# What Is A Local Model?

- Local model: working copy of the information in a host group model
- A group model can have unlimited local models associated with it
- A local model belongs to only one host group model



# Exchanging Data Between Group and Local Models



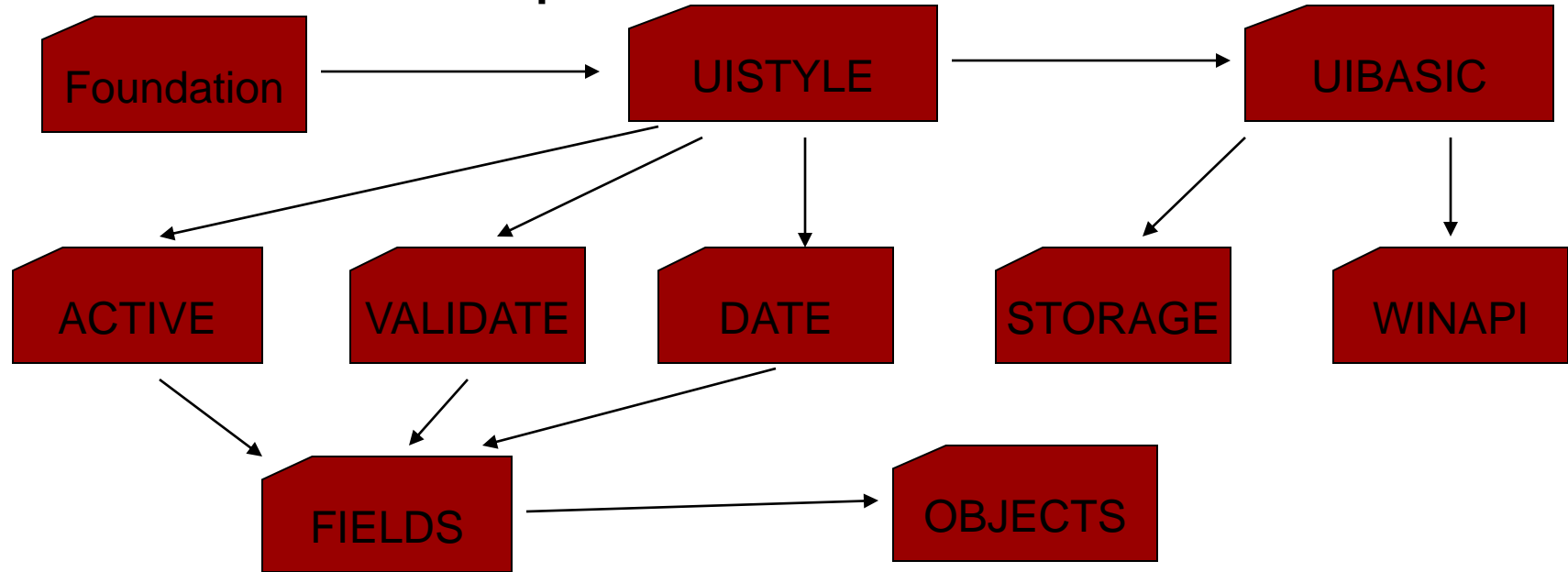
# What Is A Library Model?

---

Library Model: a group model that is being referenced by another group model

In 2e you had to export and Import objects from one model to another

## Pattern libraries provided with Plex

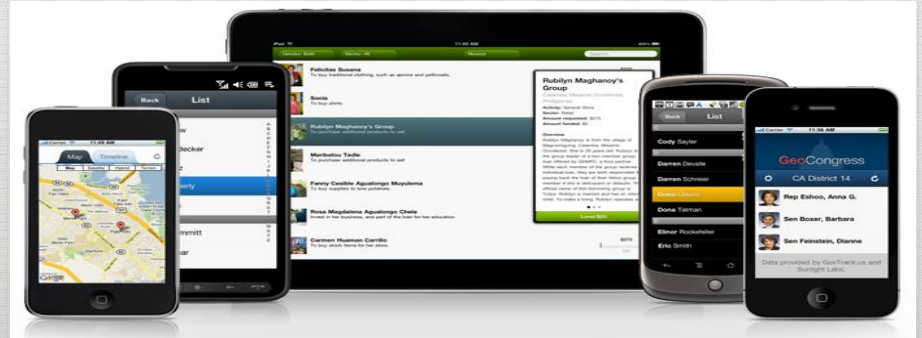




## Exercise 1

# Create Models Set up Environment Page 1-40

---



# Navigating in Plex

# Recognizing Objects

---

- Objects are the basic building blocks
- Defined by assigning name and type
- Accessed using the Object Browser

# Object Type determines Verb use

---

- Hundreds of Plex verbs
- Specifies relationship between objects of a particular type
- Identified by:
  - Source object      verb      target object
  - Example:
    - ENT                      known by                      FLD



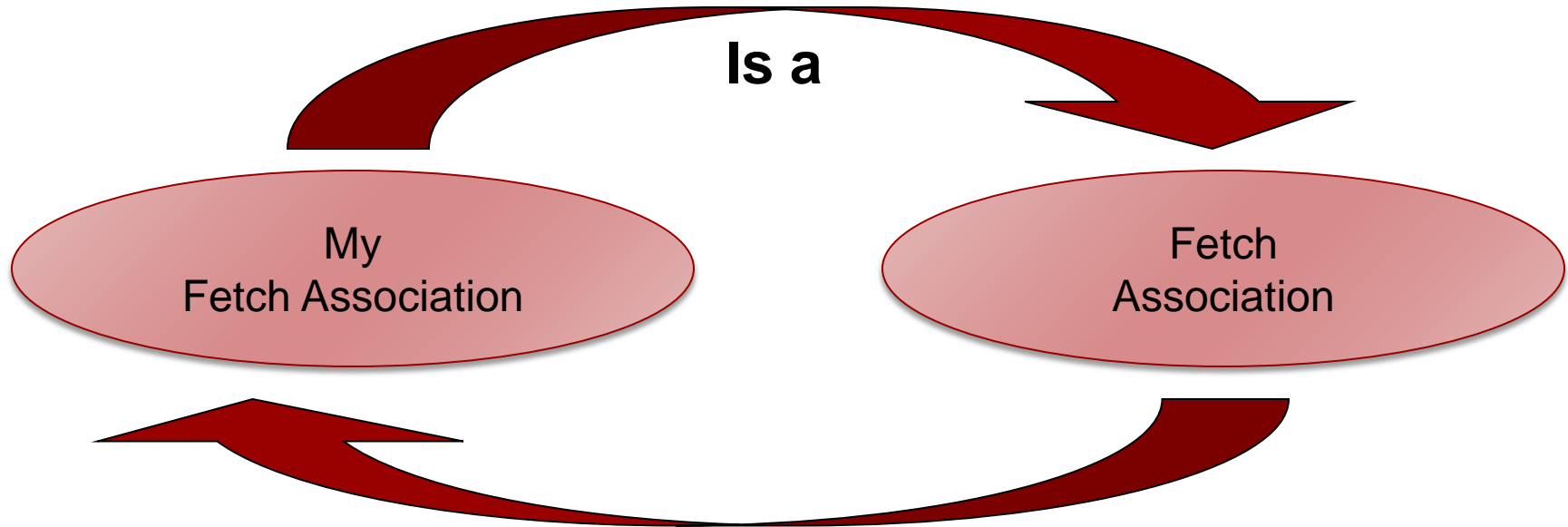
# Object Inheritance

---

- Inheritance:
  - The mechanism that allows an object to include the properties of another object or pattern

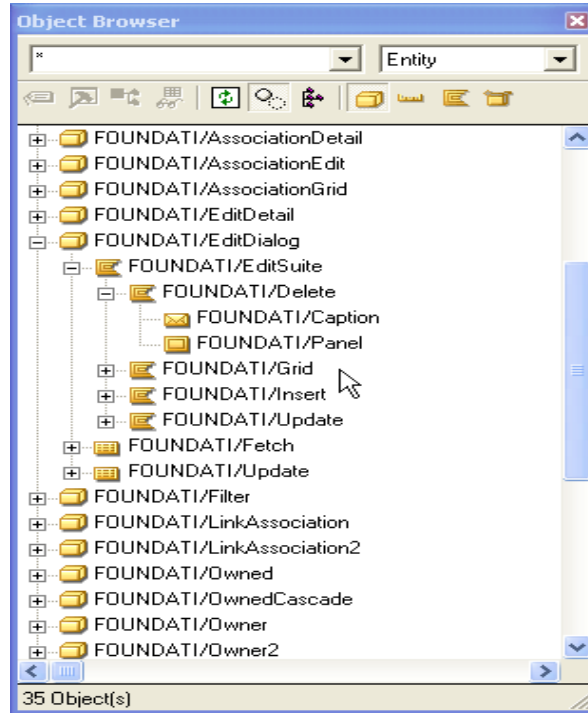
# Defining Inheritance

- Inheritance works through the “is a” verb
- Source object inherits all the properties of the target object

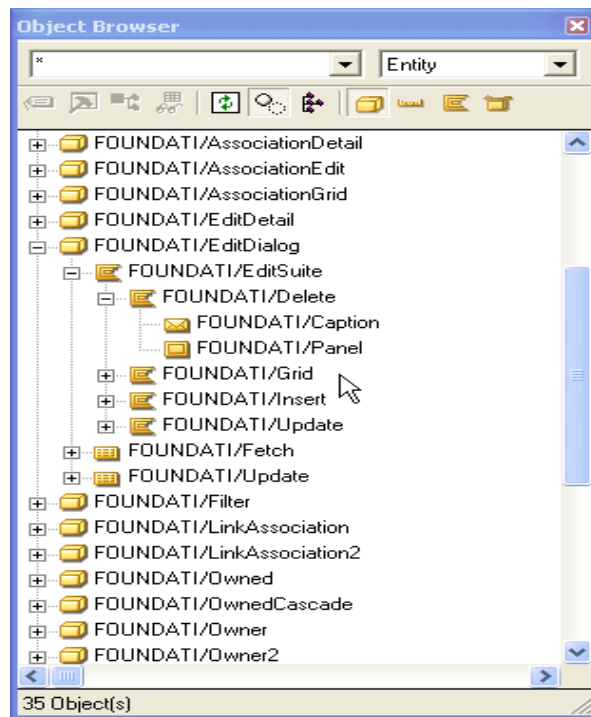


- **Patterns:**
  - Describe a solution to a common problem
  - Abstract
  - Reusable
- **Pattern Libraries**
  - Set of models that contain patterns
  - Can be combined
  - CA Supplied or third party eg Websyidian Patterns

# Work with the Object Browser

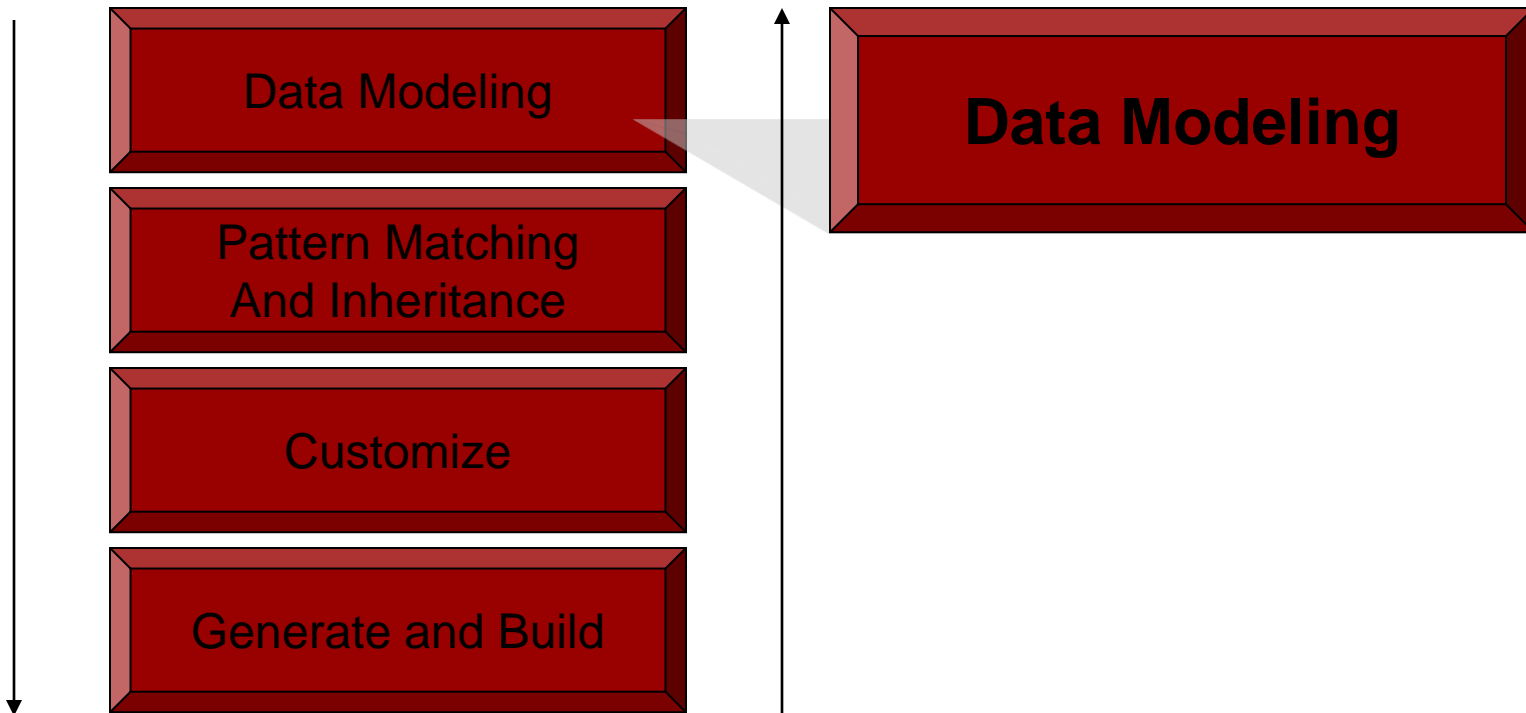


# Scoping



# Data Modeling

# CAT LEX Application Development Approach



# Categories of Entity Relationships

---

- Entity-to-Field

- Define the attributes
  - “known by”: primary keys
  - “has”: non-key attributes

- Entity-to-Entity

- Define the relationships between entities
  - “owned by”: primary keys of the owning file become primary keys of the owned file
  - “refers to”: primary keys of target entity become non-key attributes of source entity



Relationship	Inheritance	Attribute type	Attribute field	Replaced
owned by COURSE		Primary Key	Course Code	
known by Race Date		Primary Key	Race Date	
known by Race Time		Primary Key	Race Time	
has Race Name		Data	Race Name	
has Going Condition		Data	Going Conditions	
has Distance		Data	Distance	
has Prize Money		Data	Prize Money	

This column shows the ENT is a ENT triple by which the relation is inherited (if any)

The has relation defines the non-key (data) attributes of an entity.

A known by relationship defines the lower order primary key of an entity

A refers to relationship results in the primary keys of the target entity becoming foreign key attributes of the source

An owned by relationship results in the primary key of the owning entity becoming primary keys of the source entity

# Assigning Values

- Use the triple FLD value VAL
- A value is scoped by its field
- One field can have many values

	Name	Literal
Employee Status	Full Time	F
	Part Time	P
	Retired	R

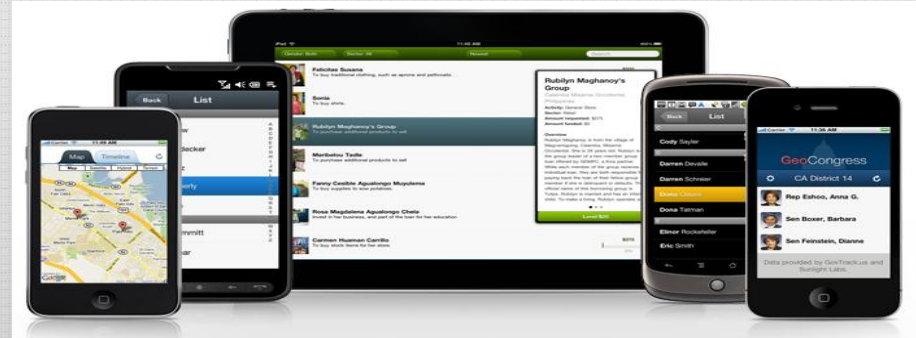
Employee Status	value	Full time
Employee Status	value	Part time
Employee Status	value	Retired



## Exercise 1

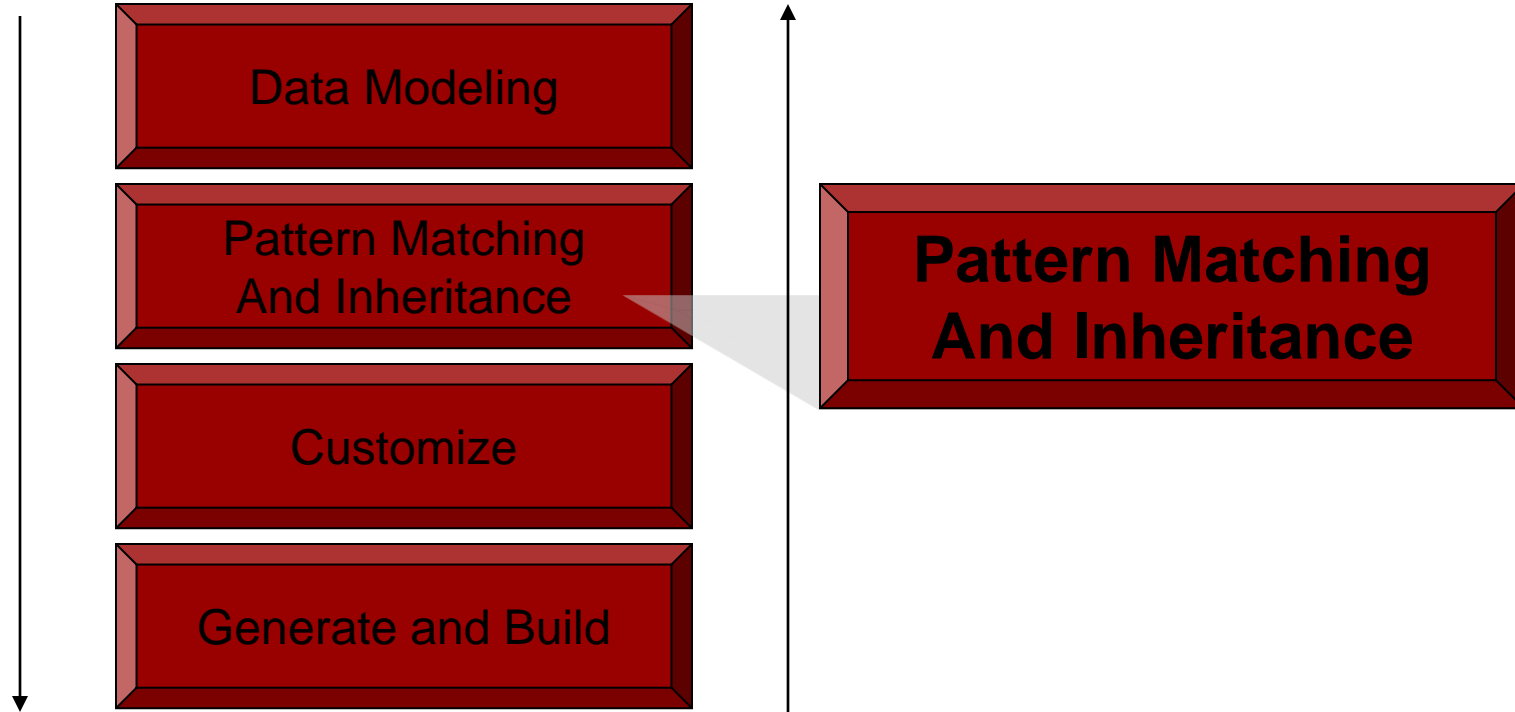
# Navigation Data Modeling Page 21-52

---

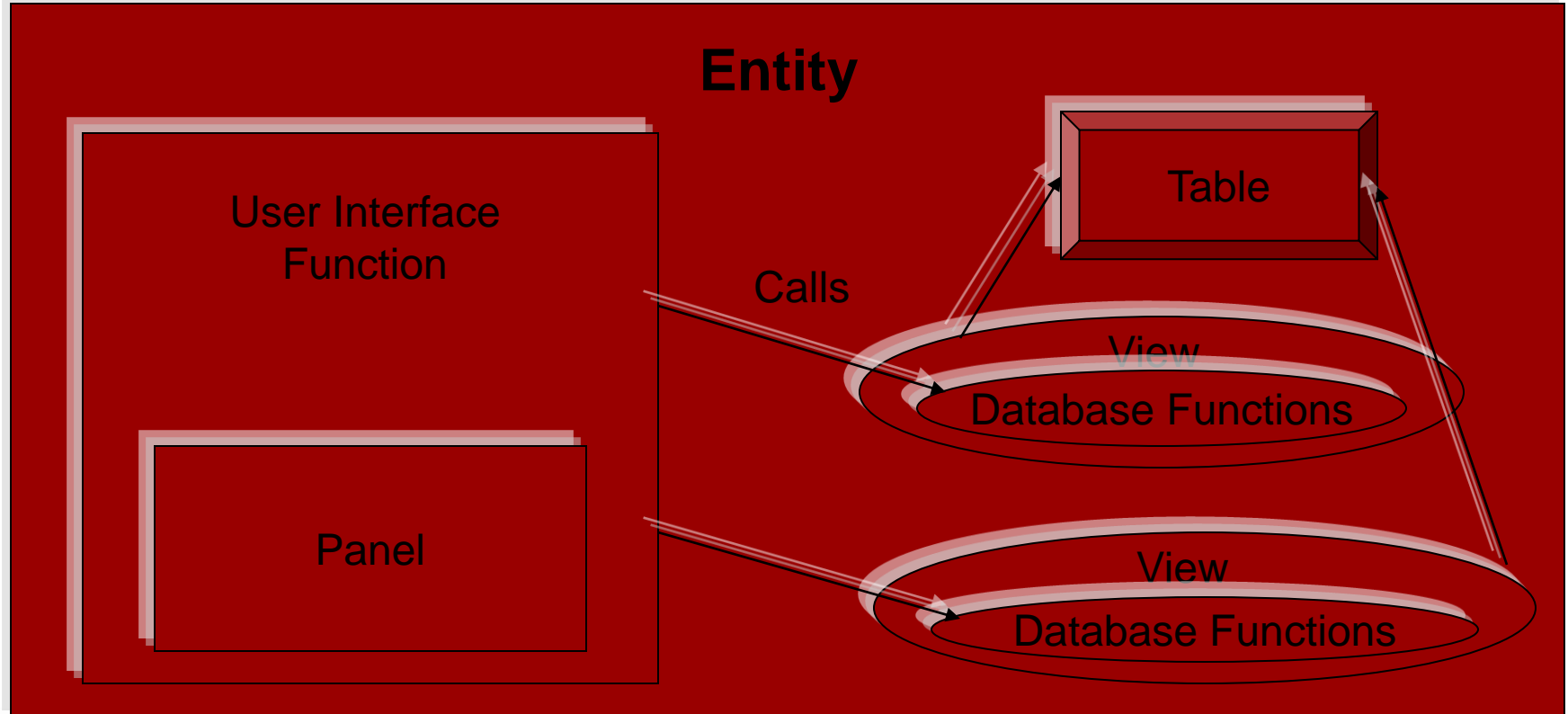


# Pattern Matching and Inheritance

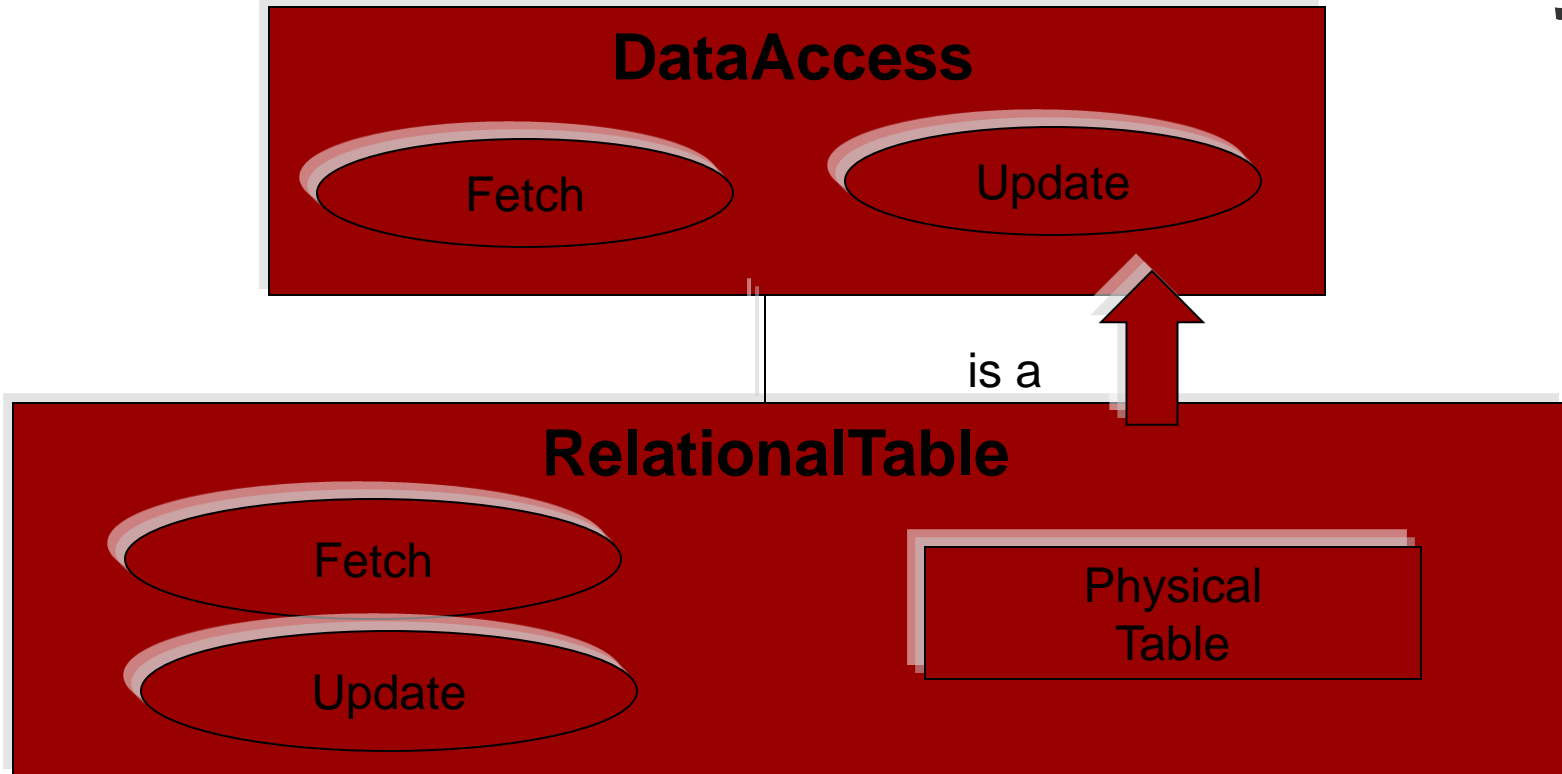
# Plex Application Development Approach



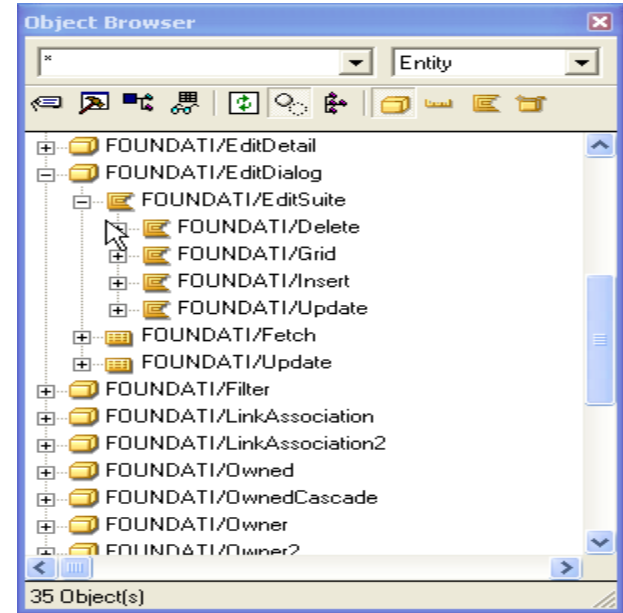
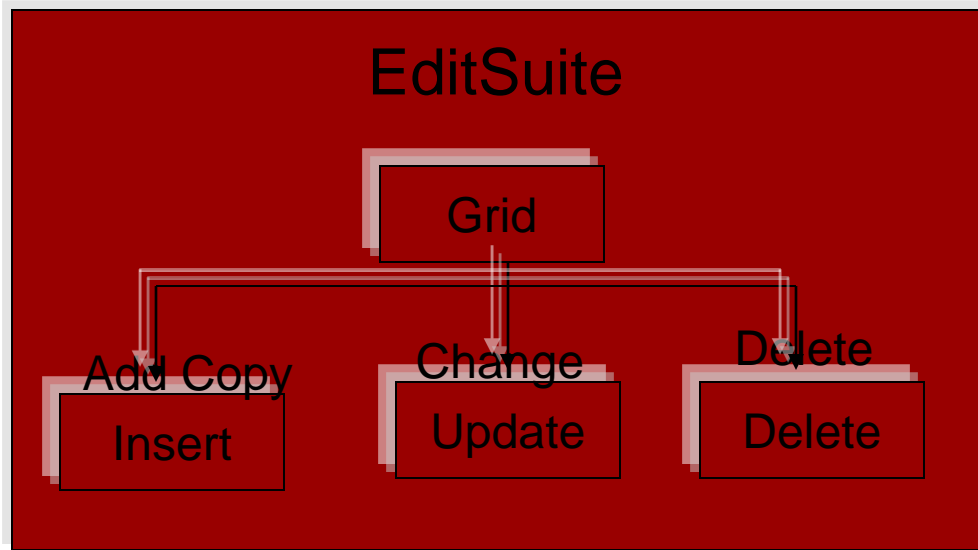
# Executing an Entity



# STORAGE/RelationalTable Entity

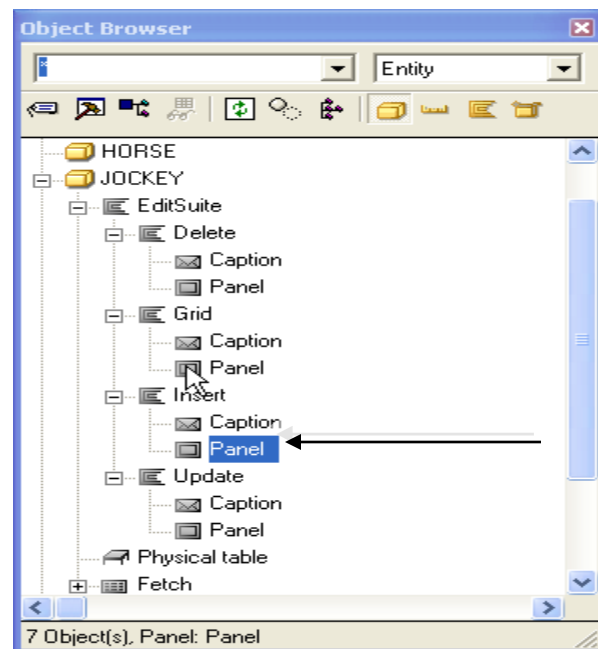
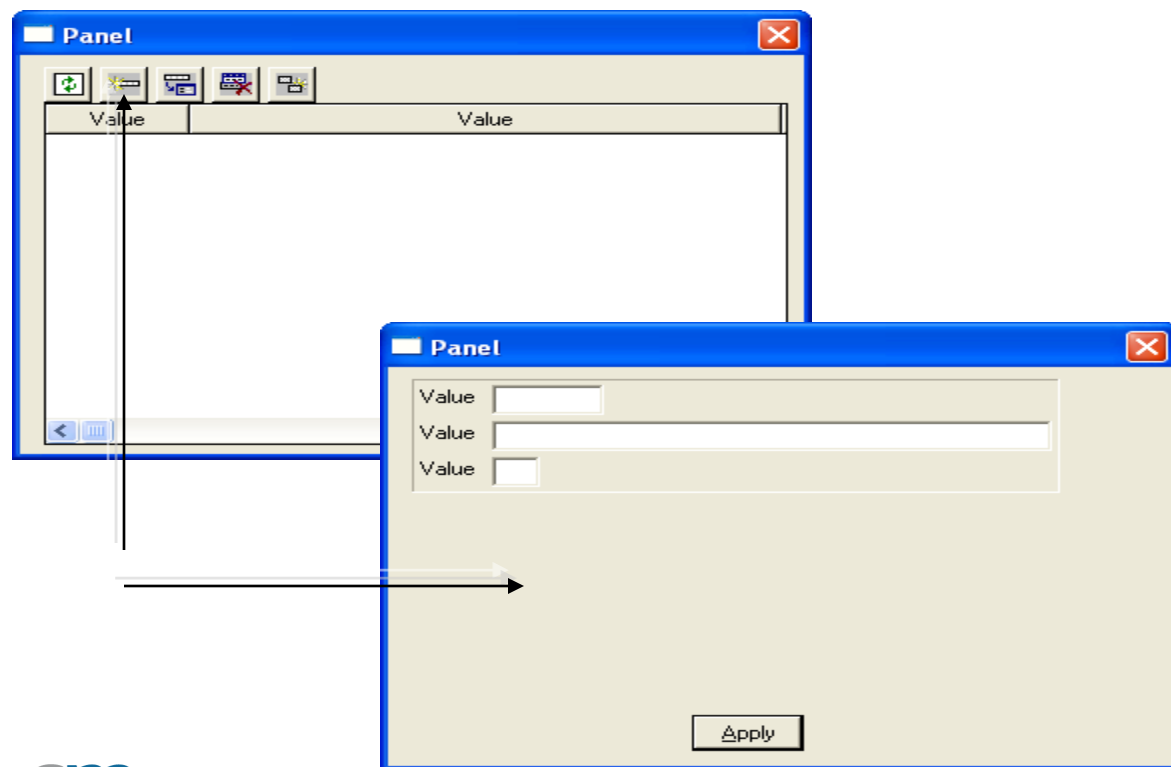


# FOUNDATION/EditDialog Entity





# Edit Suite



# Explicit vs. Implicit Inheritance

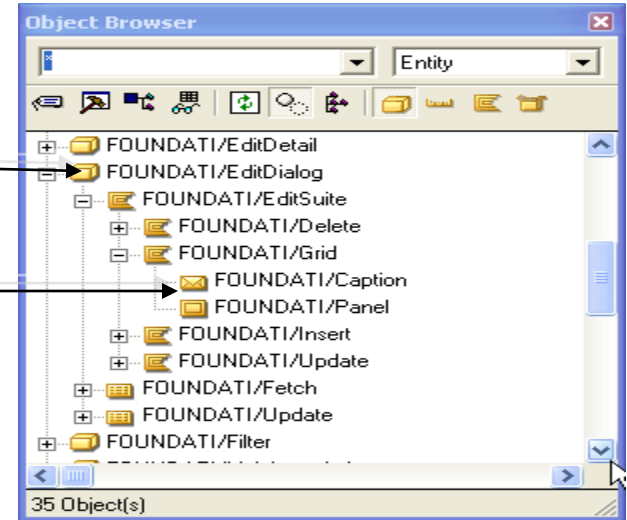
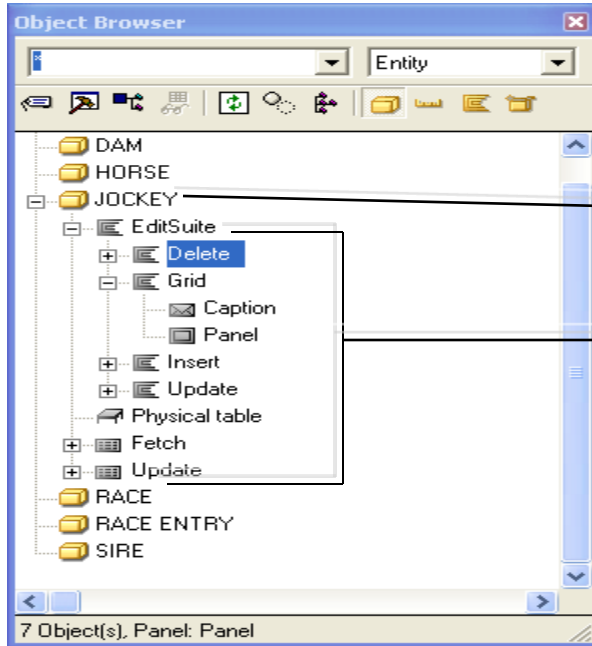
ENT is a ENT

Is a

explicit inheritance

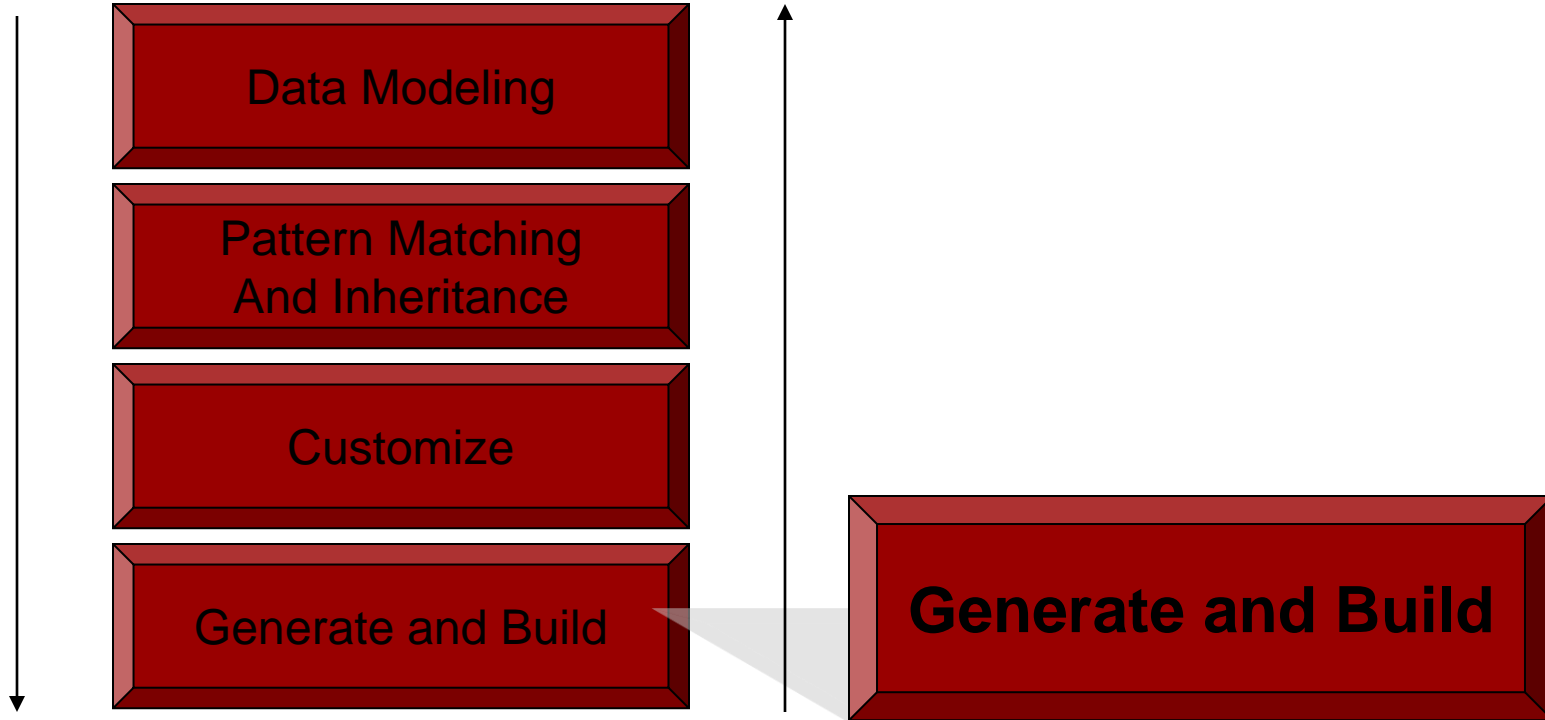
Implicit inheritance

Implicit objects



# Generate and Build

# Approach



# Generate and Build

---

- **Generate:**
  - constructs the source code necessary to execute the objects in the model
  - Interactive on your computer or remote on another computer
- **Build:**
  - Process that constructs executable objects from generated source code
  - Compiling source using a compiler

# Name Allocation Routine

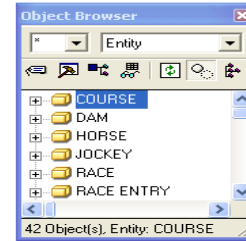
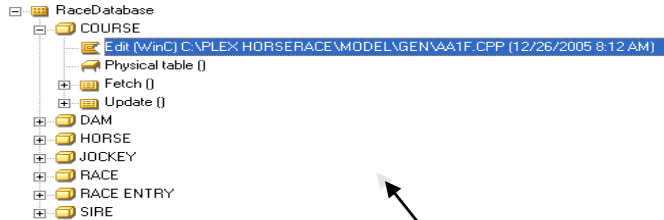
---

- Controlled by Name Allocation Routine in Generate and Build options
- Assign all names manually:
  - Turn off routine
- Assign some names manually:
  - Leave Name Allocation Routine on
  - Add triples manually before generating for first time
- Do not assign names manually:
  - Leave Name Allocation Routine on
    - Default prefix is AA
      - AAxxxxT – tables
      - AAxxxxV – views
      - AAxxxxF - functions

- Set the Generate and Build options
- Generate objects
- Build generated objects
- Construct an executable program from a compiled function
- View generated source code

# the Generate and Build Window

- CTRL+G or Generate and Build from the Tools menu



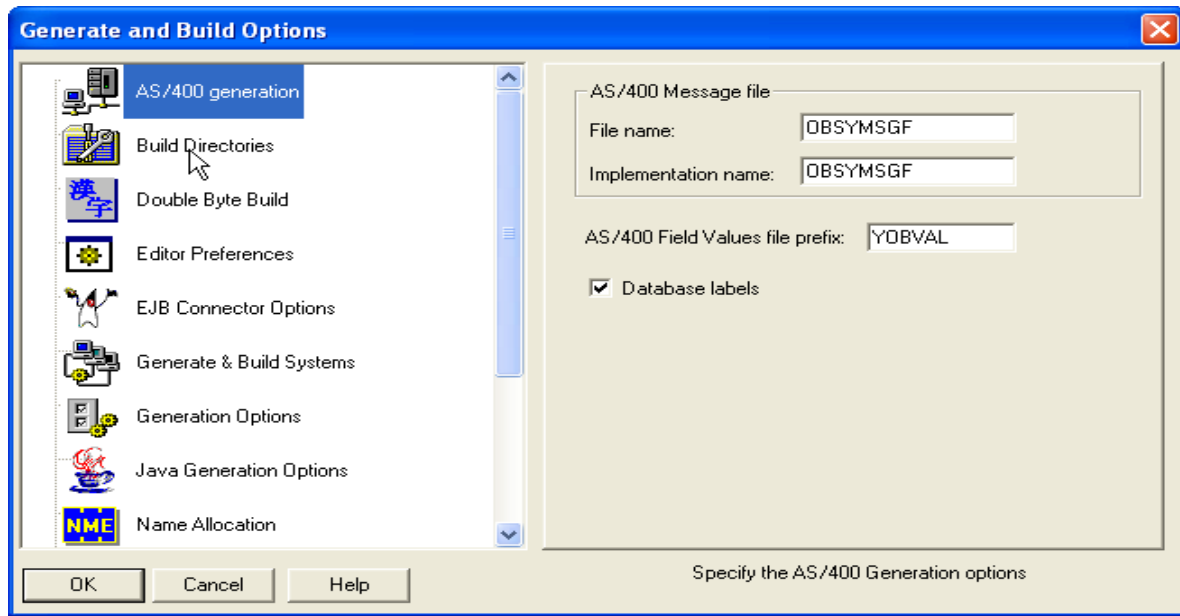
Displays path of primary source file and the time it was generated (optional)



- Errors construct a .LST file
- Use View Source option to view file
- Generation Status dialog reports number of warnings and errors
- Warnings and errors appear in the Message Log

# Setting Generate and Build Options

- Options are set at the model level
- Categories:
  - AS/400 generation
  - Double Byte Build
  - Generate and Build Systems
  - Name Allocation
  - System Definitions
  - Build Directories
  - Editor Preferences
  - Generation Options
  - Java Generation Options
  - Name Allocation Parameter
  - Topic Types



# Designing Panels

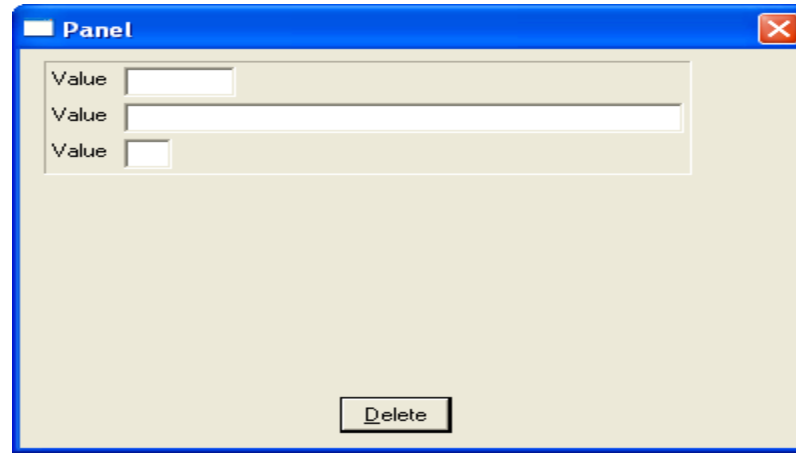
# Panel Designer

---

- Panel Designer
  - Design window
    - Displays layout of panel
  - Panel Palette
    - Represents the structure
    - Contains elements not visible in window
  - Property Sheet
    - Modify properties of any element on panel
- Drag and Drop Editing

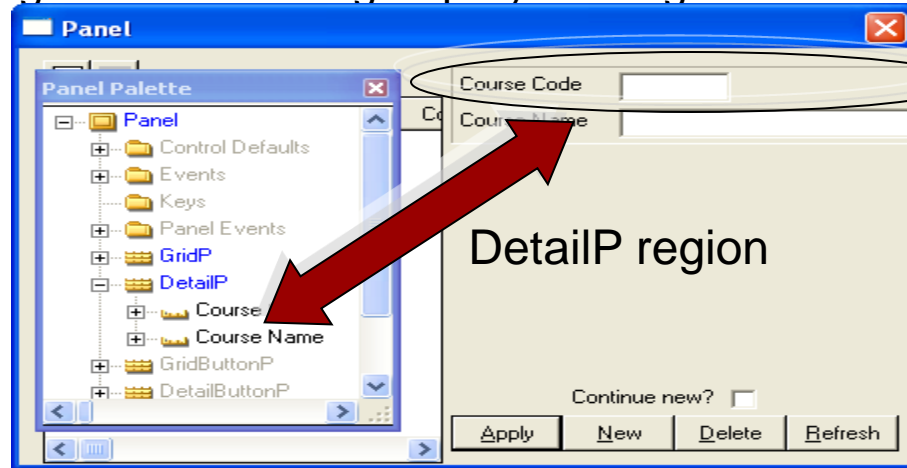
# Design Window

- Displays panel you are designing
- Closely resembles executed panel



# Field Groups on Design Panels

- Each field on a panel has multiple elements attached
  - Data control
  - Static text controls
  - Push buttons
- Selecting a field selects all elements in that field
- Select a single element in a group by holding down Ctrl while it is clicked

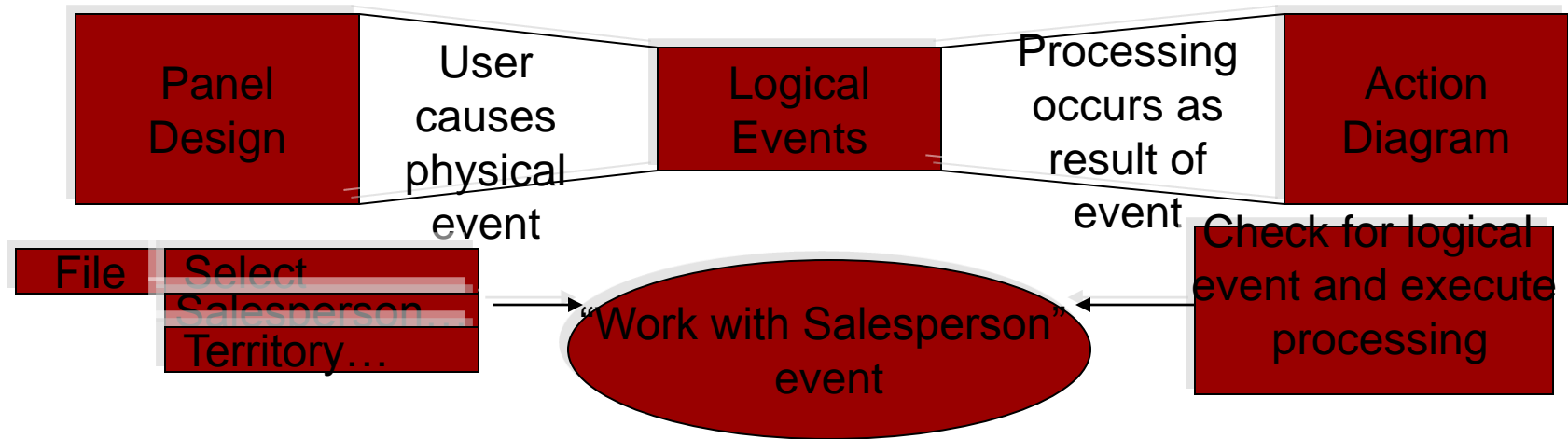


Field group

# Working with Panels

# Event Driven Design

- User starts some action by making a physical event process
- Physical events:
  - Pressing a function key
  - Pressing a button (windows)
  - Choosing a menu command (windows)
  - Entering a subfile option and pressing a key (DDS)





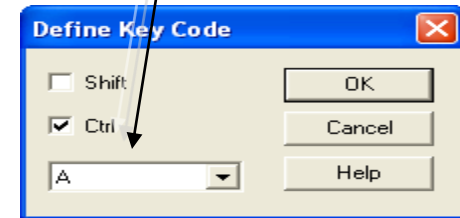
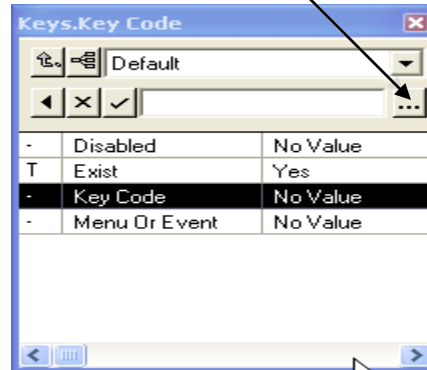
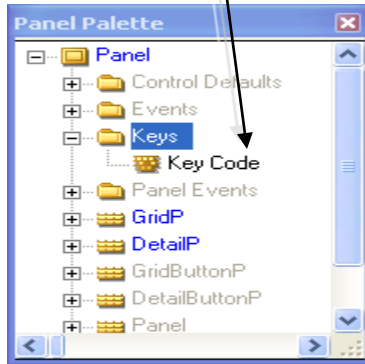
# Behavior of Elements

---

- GUI panels have built-in behaviors for elements
- To enhance behaviors:
  - Associate a physical and logical event
  - Code processing to the logical event
- Examples of physical events:
  - Query Close
  - Clock Tick
  - Notified
  - Drag and drop events
  - Mouse Move, Mouse Down, Mouse Up
  - Activated and Deactivated

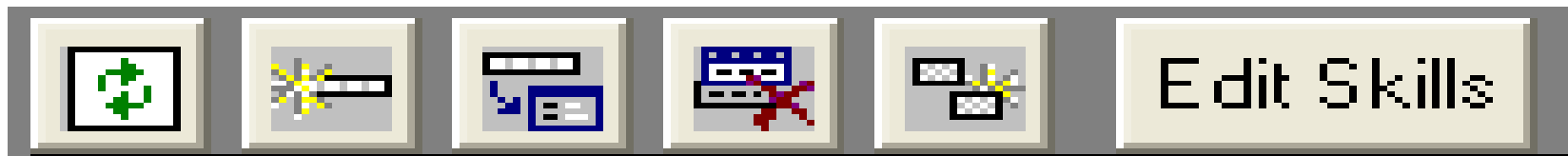
# Key Codes

- Key Code: a key (or combination) used to start processing
- To add a Key Code:
  - Add a Key Code to the Keys folder
    - Change name using Properties Sheet
      - Focus on Key Code
      - Click ellipses button to select a valid key combination
      - Events can be attached to function keys and keys using Ctrl or Shift



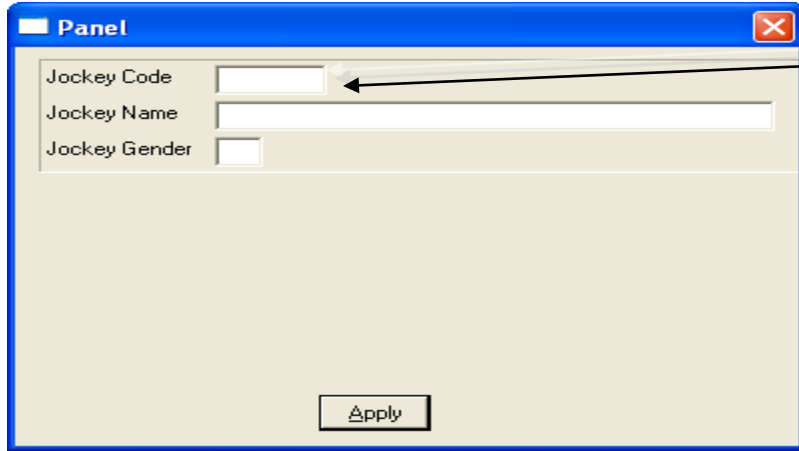
# Push Buttons

- Push button: a control used to start some action
  - Click with mouse
  - ALT + shortcut key
  - If it has focus – ENTER



# ReferredTo

# Working with the Selector(F4)



A screenshot of a software window titled "Panel". It contains three input fields: "Jockey Code", "Jockey Name", and "Jockey Gender". An arrow points from the "Jockey Code" field to the text "Double-click on Jockey Code to call Selector". At the bottom right of the window is an "Apply" button.

Double-click on Jockey Code to call Selector



A screenshot of the same "Panel" window after the "Jockey Code" field has been double-clicked. The window now displays a table with the following columns: "Jockey Code", "Jockey Name", and "Jockey Gender". The table is currently empty. To the right of the table is a "Position" button. At the bottom right of the window is a "Select" button. An arrow points from the "Jockey Code" field to the "Select" button.

# Views

- Views are scoped to entities
- Views do not need to be generated
  - Define a group of fields for a particular purpose
- Views do not contain data
- Properties:
  - Attributes included
  - Processing order of rows
  - Selection criteria

# Functions

---



# Tools for the Action Diagram

Type in the input line

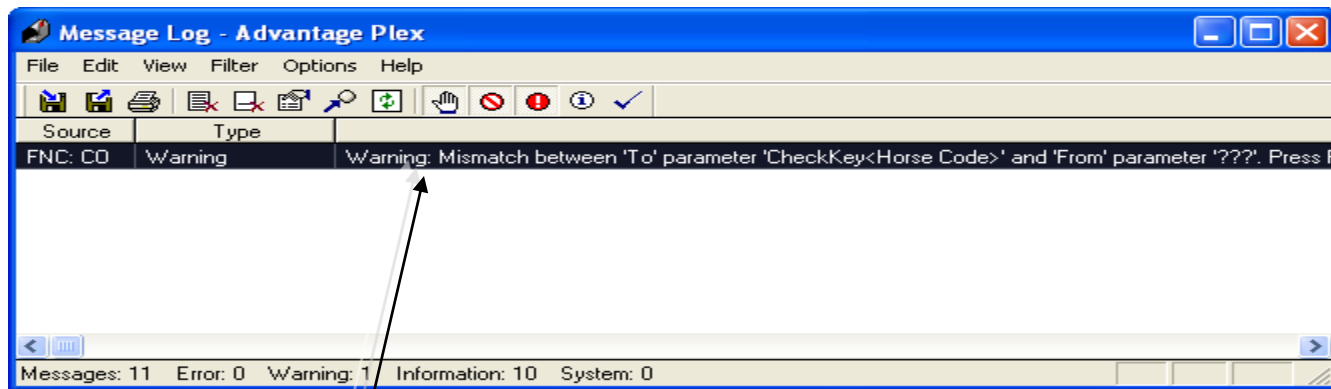
The screenshot displays the software interface for creating an action diagram. At the top, the input line shows the text "call Function: COURSE.Edit". Below this, the "Object Browser" window is visible, showing a tree structure with "COURSE" selected, containing "Edit", "Physical table", "Fetch", "Update", and "DAM". The "Action Diagram Palette" window is also open, showing a list of actions such as "Call <Function>", "Comment <Comment>", "Conditions...", "Edit Point <Name>", "Loops...", "Operators...", "Return", "Seq <Name>", "Set <Assignment>", "Subroutine...", "Terminate", "Component operations...", and "Events...". Arrows indicate the workflow: one arrow points from the "Return" action in the palette to the input line, and another arrow points from the "Edit" action in the Object Browser to the input line.

Function: COURSE.Edit  
Copyright 2004 Computer Associates International, Inc. All Rights Reserved.  
Seq Description  
Go Sub Initialize  
Pre Point  
Edit Point Execute  
Events Handler  
Event  
Event Event: Close  
Pre Point  
Edit Point Events  
Post Point  
Return  
Post Point Execute  
Sub Initialize  
Sub Terminate  
Sub Send message  
Pre Point  
Edit Point Subroutines  
Post Point

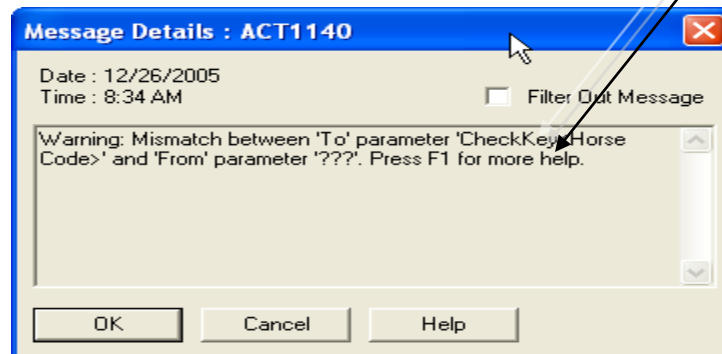
Drag and drop from the Object Browser to the input line

Drag and drop from the Palette to the input line or directly into the action diagram

# Message Log



Double click to open function and find the error



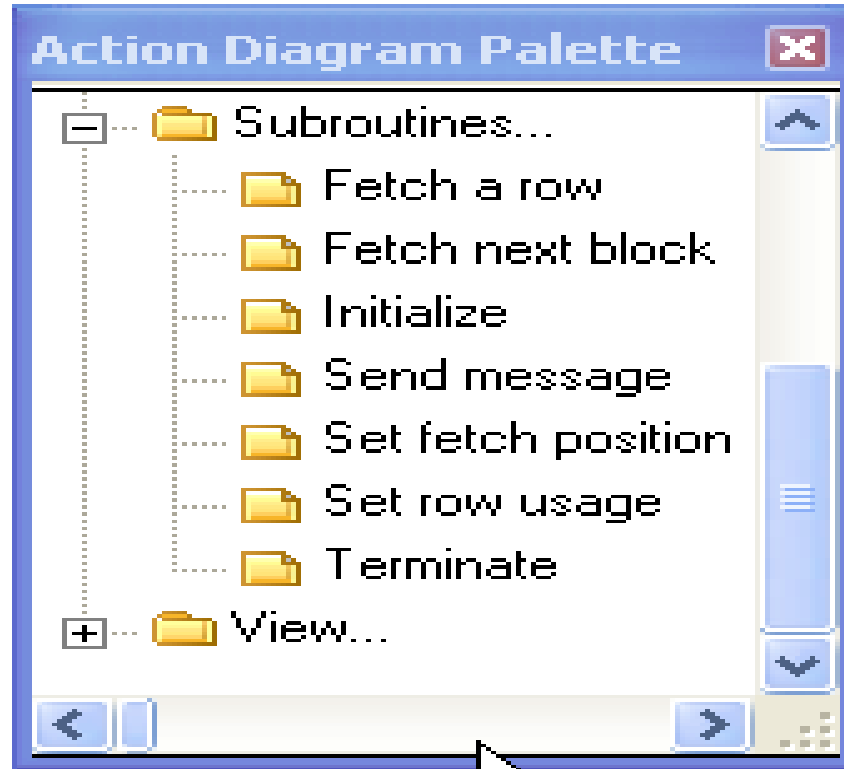
Right click for message details

# Recognizing Subroutines

---

- Reusable code
- Can be called from anywhere in the Action Diagram it resides
- The preferred method of coding
  - Easily understood
  - Easily maintained

# Subroutines Folder

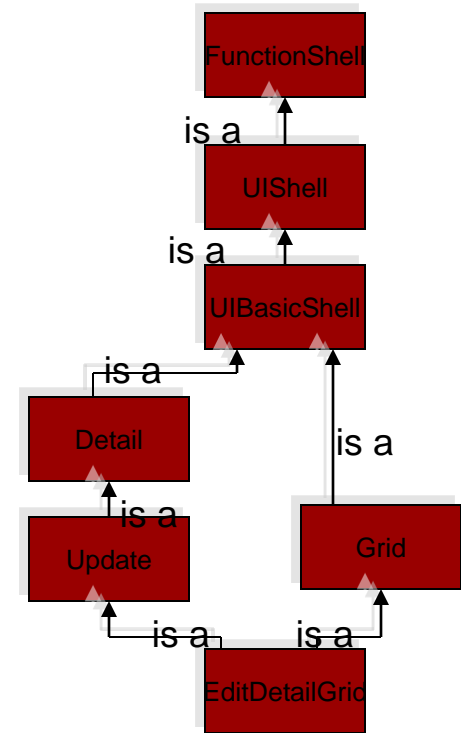


# Multiple Inheritance

---

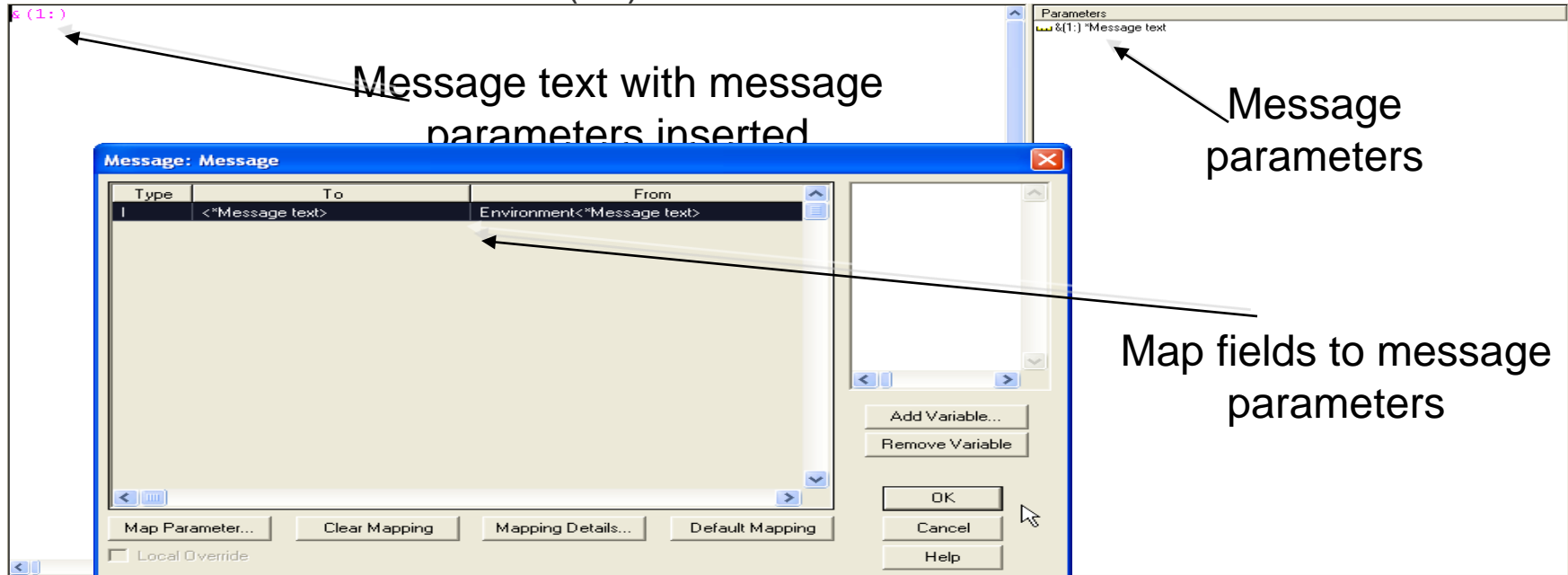
- One source function can inherit from many target functions
  - All must have common ancestor
  - All code enhancements inserted in Pre and Post points

- **UISTYLE/EditDetailGrid**
  - EditDetailGrid =  
UIBASIC/Update and  
UIBASIC/Grid
    - Read-only grid region (from Grid)
    - Editable single instance region (from Update)
    - Code is blended in Action Diagram



# Parameters

- ~~Add parameters to a message;~~
  - Triples:
    - MSG parameter FLD
    - ... QLF
  - Select and call Editor (F9)



# Message Types: Dialog

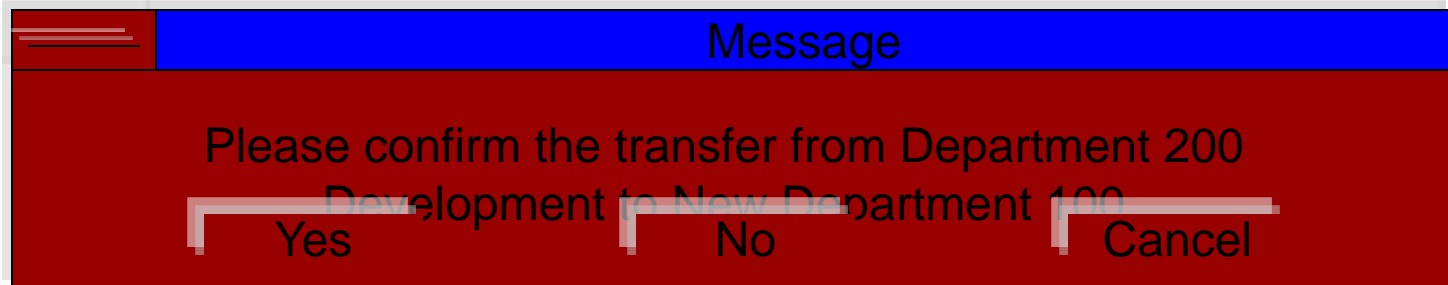
- Dialog
  - Uses modal dialog box with OK button
  - User must respond





- Enquiry
  - Uses modal dialog box with Yes, No, Cancel buttons
  - User must respond

Enquiry message Message: Employee.Transfer Msg, Work<Enquiry Answer>



- Status
  - Shown in status bar
  - No response required from user

Status Message Message: Calc 1.Status msg

Add Input 1 to Input 2

- 
- For a complete line of CA Plex education courses, contact:

**CM First**

7000 North Mopac Expy  
Plaza 7000 Second Floor

Austin, TX 78731

[cmfirstgroup.com](http://cmfirstgroup.com)