Object-Oriented Design for Data Warehouses
OOD – for Data Warehouses

- Giovinazzo, p. 35 “It has been well-documented that the majority of data warehouse projects fail”
- This may have been true in 2000 when the book was published, but it’s not true today.
Waterfall Design Approach versus Iterative Design Approach

- Unit Test + Acceptance Test in Giovinazzo
Spiral Design Approach
# What’s Right and Wrong with Waterfall Approach?

## Pros
- Not messy
- Works well when requirements are really set before design/development begins
- Works well when deliverable is not needed urgently

## Cons
- Static set of requirements is usually a faulty assumption, especially in the arena of business intelligence
- With system use, requirements will change
- Often, sponsors want something delivered NOW
What’s Right and Wrong with Spiral Development Approach?

Pros

- Flexible
- Must be able to meet diverse needs of users
- Short time to partial delivery
  - Start with a modest deliverable goal and “get it done” (e.g. start with a data mart)
  - Add new functionality with each “pass” through the spiral

Cons

- Scope creep
- Unstated requirements
Important Elements of Object Orientation in Design
Metadata

- More comprehensive than data
- Transcends data in the sense that it goes beyond the individual data item and provides the context in which the data exists
- Context gives data meaning
Where is Metadata Important?

- Definition phase – metadata created here
  - Metadata about context
- Analysis phase – metadata used and created here
- Design phase – metadata used and created here
  - Data elements
  - Database structure
  - Processing allowed on elements
Types of Metadata

Metadata
  - Static (Structural)
    - Name
    - Format
    - Relation
    - Domain
    - Business Rule
  - Dynamic (Operational)
    - Data Quality
      - Statistical
    - Status
    - Process
  - Data Type
    - Picture
Types of Metadata -- Static

- **Name** – by which the data element is known to the system
  - Ex: Employee_name, Customer_Name, Customer_ID
- **Description** – full text description of the data element
- **Format** – data presentation rules
- **Data Type** – data storage rules
  - Ex: integer, floating point, boolean, character
- **Relation** – relationships between objects represented in the system
  - Ex: Customers buy Products
- **Domain** – domain or range of valid values
- **Business Rules** – rules of the organization that govern the data element
Types of Metadata -- Dynamic

- Quality – describes the quality of the data in the system
  - Accuracy
  - Completeness
  - Consistency
  - Validity
- Statistical – usage and administrative characteristics of the system
  - Data access frequency
  - Types of analyses performed on the data
  - Which users access the data
- Status – general system health
  - Backup statistics
  - System performance
  - Error log
• Note that static metadata changes infrequently – whereas dynamic metadata is updated very frequently
Metadata Repository Accommodates Users of Metadata

- Data Warehouse Architect
- System Engineer
- Administrator
- User
Enterprise Data Model

- Central Data Repository contains an enterprise view of data
- A blueprint of the organization’s business
- Gets input from
  - Accounting
  - Manufacturing
  - Human Resources
  - Sales
How Does an Object Orientation Help?

- Helps structure the metadata
What is an Object?

- A concept
- An object has properties or attributes
- An object has behaviors
Terminology of an Object Orientation

- **Class**
  - When we start grouping objects according to like attributes we are forming classes

- **Superclass**
  - A set of classes with common attributes

- **Aggregation**
  - When objects combine to form a new object
Object Modeling Reveals Conceptual Metadata for an Organization and Its Data Warehouse
What Does It Mean to Be Object-Oriented?
Object

- Object: A thing presented to or capable of being presented to the senses
- Does not have to be a thing
- It is a concept
Properties of Object Orientation

- Encapsulation
- Abstraction
- Polymorphism
- Inheritance
Encapsulation

- Encapsulation: grouping related ideas into one unit, which can then be referred to by a single name
- Object-oriented encapsulation: the packaging of operations and attributes representing state into an object type so the state is accessible or modifiable only via the interface provided by the encapsulation
- Software encapsulation has been around since the beginning of programming. At first functionality was encapsulated in subroutines and/or functions
What Defines an Object and Enforces Encapsulation?

- Its operations: functionality normally visible to other objects
  - Other objects can use the functionality
  - Also referred to as a Method
- Its attributes: represent the information (data) that the object remembers
  - Normally accessed and updated by the object’s operations
  - Attributes are used much like variables and constants within the object
What Defines an Object and Enforces Encapsulation?

- **State**: The set of values that an object holds
  - State values are usually found in the attributes

- **Interface**: The formal way in which other objects gain access to the attribute values of an object
  - Normally done through the object’s operations

- The name and parameter list of a function is the formal interface used in traditional programming languages
Information Hiding

- Part of encapsulation
- Information/Implementation Hiding: the use of encapsulation to restrict from external visibility certain information or implementation characteristics (such as metadata)
- Benefit: localizes design decisions
  - Design decisions that affect only the object itself have no impact on the rest of the system
  - Facilitates interoperability and integration
- Benefit: decouples content from representation
  - Separates content and its structure from format
  - no user of an object’s information that is external to the object is tied to any particular internal information format
State Retention

- State Retention: An object retains its state.
Object Identity

- Object Identity: each object, regardless of class or current state, can be identified
- The same handle remains with the object for its life, regardless of what is done to the object by processing
- No two objects can have the same handle
Summary – Object Orientation Property
- Encapsulation

- Encapsulation
  - package attributes and operations together

- Information/Implementation Hiding
  - hide what information an object holds and how it is implemented

- State Retention
  - an object never forgets

- Object Identity
  - all objects are distinct for as long as they live
Object-Oriented Design for Data Warehouses