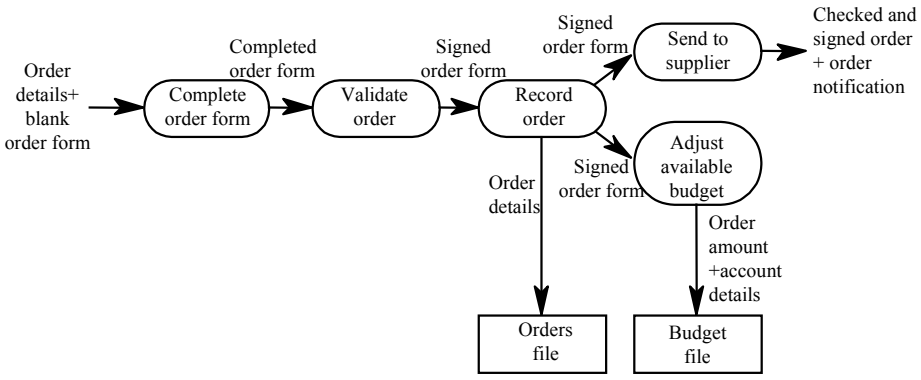
System models

- Abstract presentations of systems whose requirements are being analysed

Data-flow models （データフローモデル）

- Show the processing steps as data flows through a system
- Intrinsic part of many analysis methods
- Simple and intuitive notation that customers can understand
- Show end-to-end processing of data

Order processing DFD

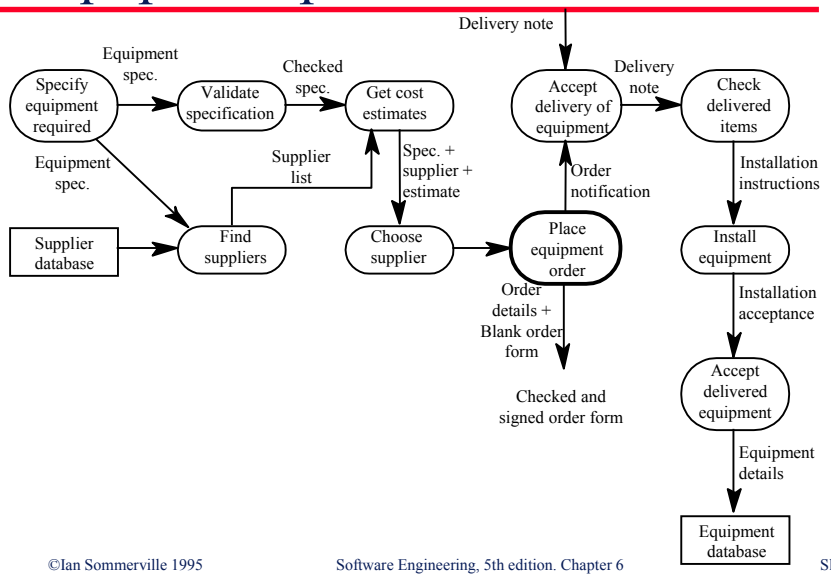


DFD: データフローダイアグラム

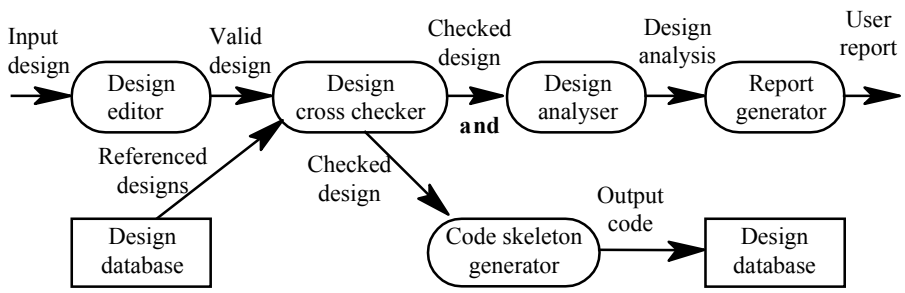
Data-flow diagrams

- may be used to show processing at different levels of abstraction from fairly abstract to fairly detailed
- May also be used for architectural description showing data interchange between the sub-systems making up the system
- Not a good way to describe system interfaces

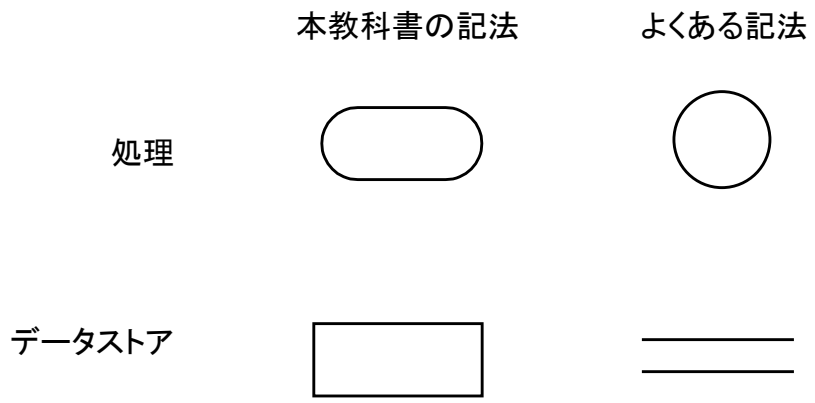
Equipment procurement DFD



CASE toolset DFD



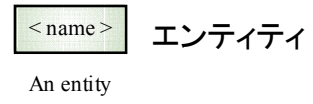
注：



Semantic data models（意味データモデル）

- Used to describe the logical structure of data processed by the system
- Entity-relation (ER) model sets out the entities in the system, the relationships between these entities and the entity attributes
- Widely used in database design. Can readily be implemented using relational databases

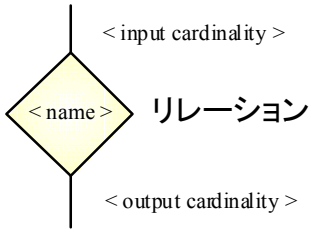
Notation for semantic data models



エンティティ

An entity

An entity or relation attribute

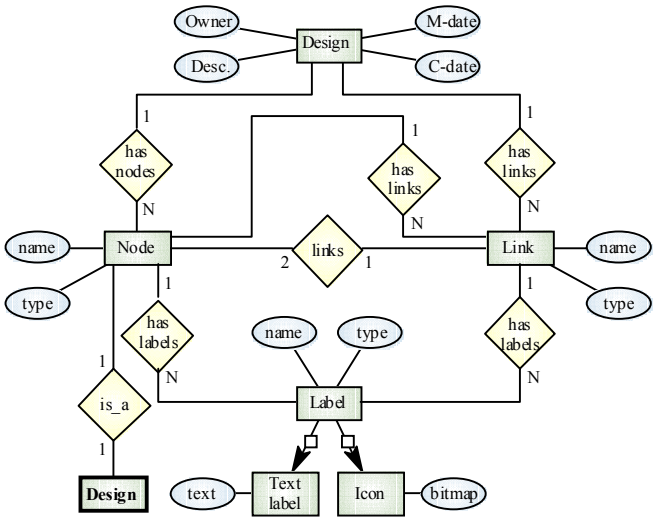


リレーション

A relation between entities. The number of input entity instances is the input cardinality. The number of output instances is the output cardinality.

An inheritance relation. An entity inherits the attributes of its related entity. The sub-type is referenced by the arrow.

Software design semantic model



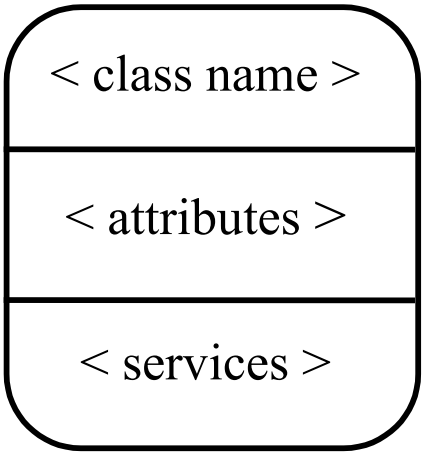
Object models

- Object models describe the system in terms of object classes
- An object class is an abstraction over a set of objects with common attributes and the services (operations) provided by each object
- Various object models may be produced
  - Inheritance models
  - Aggregation models
  - Service models

Object models

- Natural ways of reflecting the real-world entities manipulated by the system
- More abstract entities are more difficult to model using this approach
- Object class identification is recognised as a difficult process requiring a deep understanding of the application domain
- Object classes reflecting domain entities are reusable across systems

Object class notation



Inheritance models

- Organise the domain object classes into a hierarchy
- Classes at the top of the hierarchy reflect the common features of all classes
- Object classes inherit their attributes and services from one or more super-classes. These may then be specialised as necessary
- Class hierarchy design is a difficult process if duplication in different branches is to be avoided

Library class hierarchy

- Replace with portrait slide

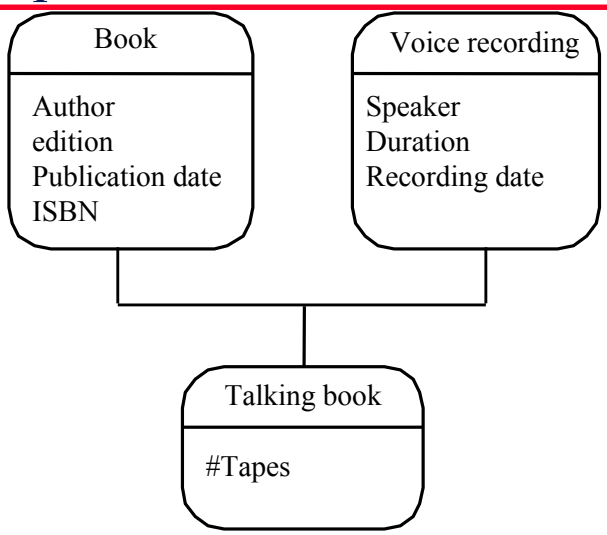
User class hierarchy

- Replace with portrait slide

Multiple inheritance（多重継承）

- Rather than inheriting the attributes and services from a single parent class, a system which supports multiple inheritance allows object classes to inherit from several super-classes
- Can lead to semantic conflicts where attributes/services with the same name in different super-classes have different semantics
- Makes class hierarchy reorganisation more complex

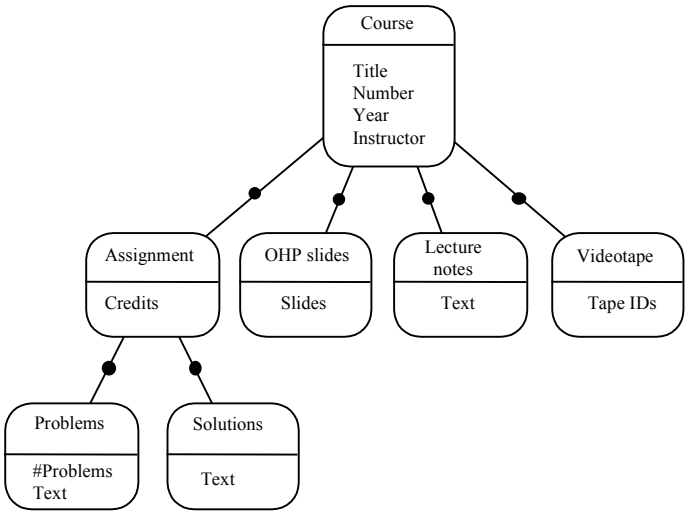
Multiple inheritance



Object aggregation

- Aggregation model shows how classes which are collections are composed of other classes
- Similar to the part-of relationship in semantic data models

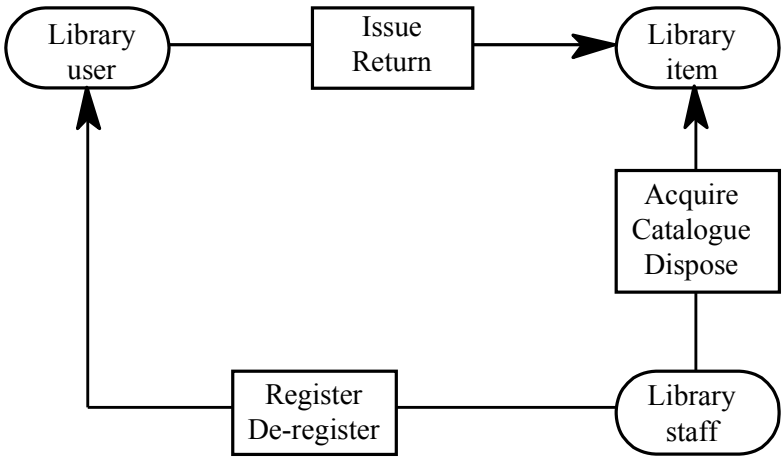
Object aggregation



Service-usage models

- These models show how services provided by one object are used by other objects
- Must be used sparingly as, obviously, some objects provide common services which are used by many other objects in the system

Service usage



Data dictionaries（データ辞書）

- A data dictionary is a list of names and associated descriptions of entities used in the system
- It represents a shared repository of system information
- It serves as
  - A mechanism for name management. As a system model may be developed by different people, there is potential for name clashes
  - A link from analysis to design and implementation

Data dictionary semantic model

- Replace with portrait slide

Data dictionary entries

- All names used in the system model, design and implementation should be entered in the data dictionary
- Support software should be used to create, maintain and query the dictionary
- The data dictionary may be integrated with CASE tools so that its construction and maintenance may be partially automated

Data dictionary entries

Name	Description	Type	Date
has_labels	1:N relation between entities of type Node or Link and entities of type Label.	Relation	5.10.93
Label	Holds structured or unstructured information about nodes or links. Labels can be text or can be an icon.	Entity	8.12.93
Link	Represents a relation between design entities represented as nodes. Links are typed and may be named.	Relation	8.12.93
name (label)	Each label has a name which identifies the type of label. The name must be unique within the set of label types used in a design.	Attribute	8.12.93
name (node)	Each node has a name which must be unique within a design. The name maybe up to 64 characters long.	Attribute	15.11.93