Object-oriented system development: survey of structured methods

-A. G. Sutcliffe, 1991

T5
200611495 유상원 [PPT 작성]
200611496 유승현 [번역]
200611512 임재식 [PPT 작성]
200611524 허준호 [발표]

0. Introduction

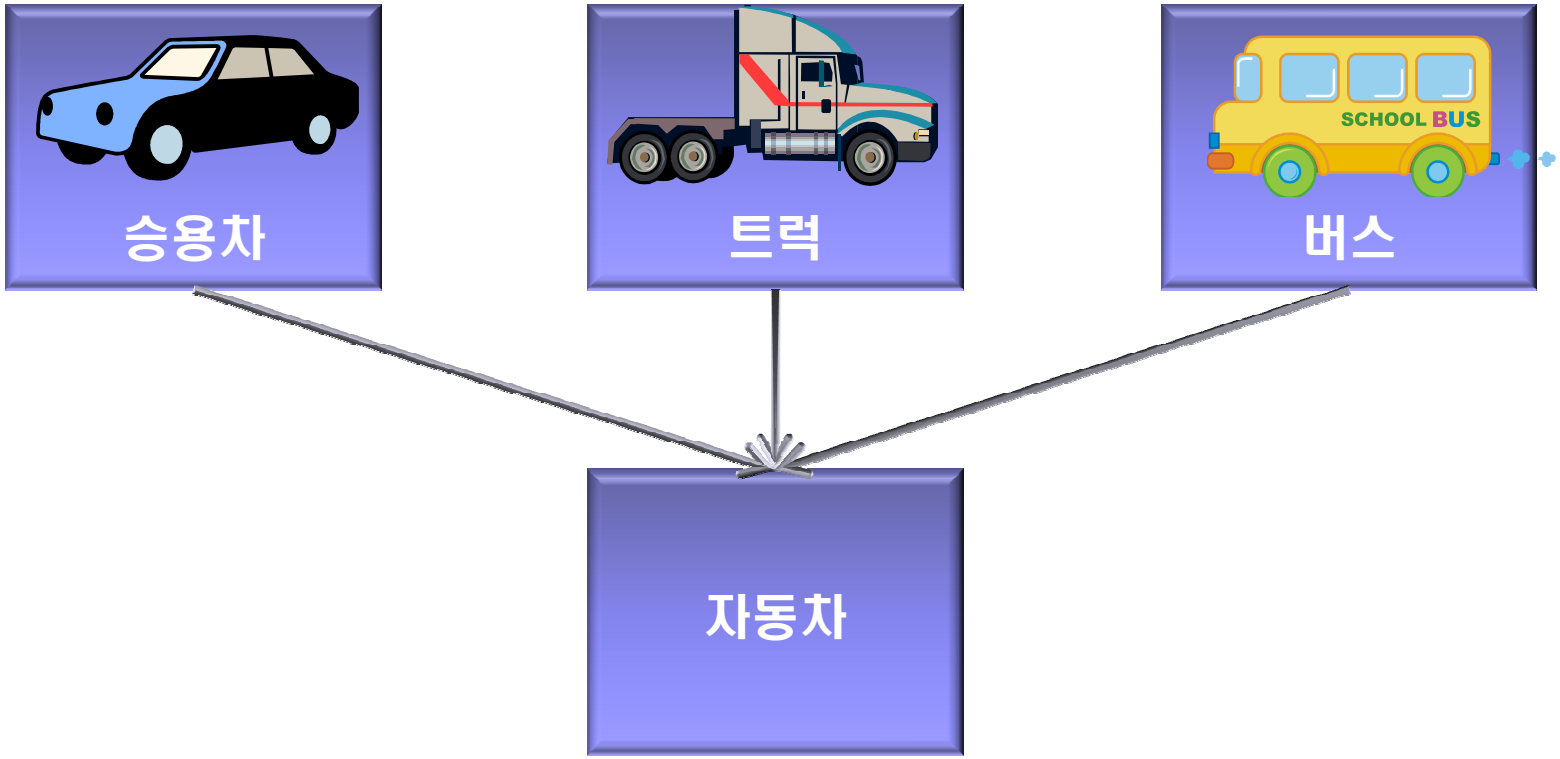
- 1. Object-Oriented Concepts
- 2. Evaluation of modeling components
- 3. Evaluation Procedure
- 4. Object-Oriented Methods
- 5. Structured Methods
- 6. Conclusions

1. Object-Oriented Concepts

- Three principles that make OOD to improve software design for reliability and maintenance
  - Abstraction
  - Encapsulation
  - Inheritance

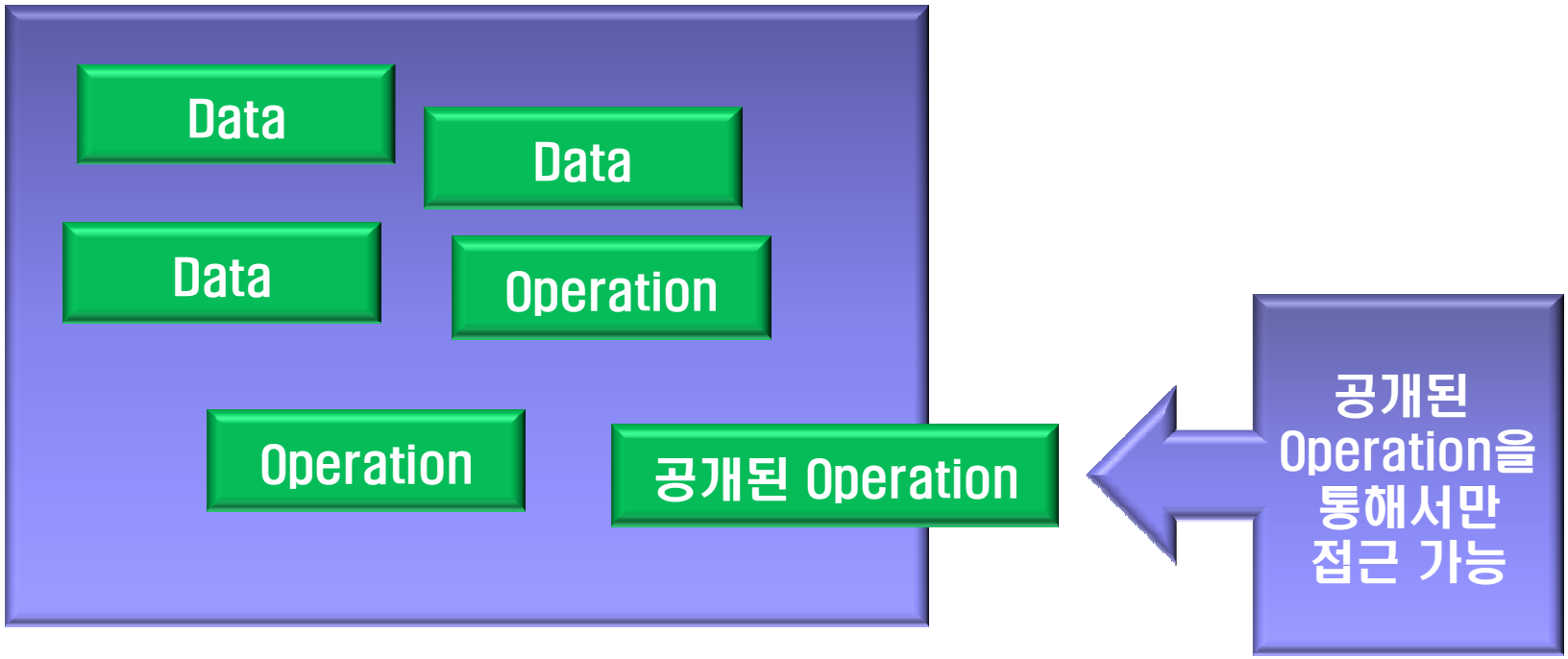
1. Object-Oriented Concepts

- Abstraction
  - Objects are an abstraction of parts of real-world. More maintainable and reusable



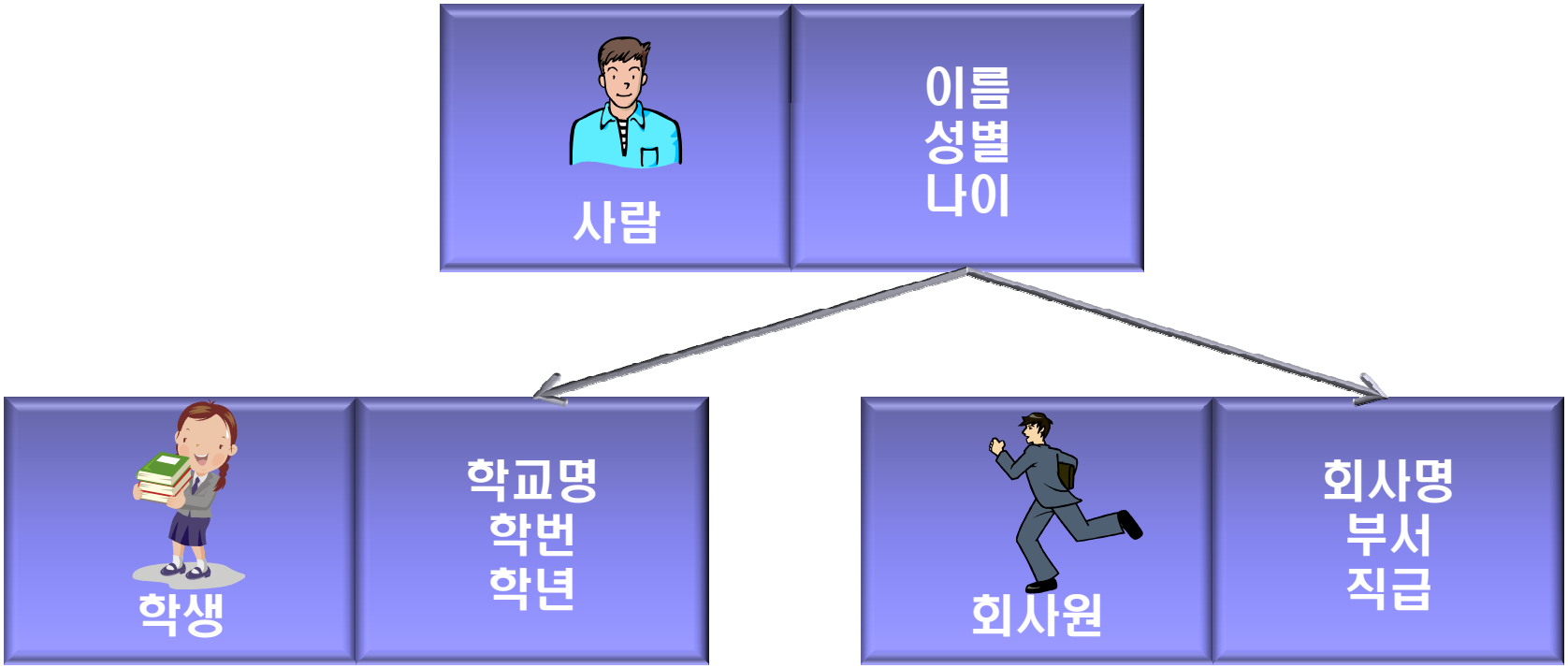
1. Object-Oriented Concepts

- Encapsulation
  - Objects hide their internal contents from other components to improve maintainability



1. Object-Oriented Concepts

- Inheritance
  - By organizing objects in class hierarchies to promote reuse



2. Evaluation of Modeling Components

- Object vs Traditional Concepts of Entities and Functions

	Traditional	Object-Oriented
Same	Entity	Object
	Instance, Type, Class (ISO TC97 report)	
	Event	Message
	Controlling their behavior	
Different	Separate Data and Function	Data + Activity

2. Evaluation of Modeling Components

- Objects may have more or less activity associated with them
  - Data-Oriented Objects
  - Task-Oriented Objects

2. Evaluation of Modeling Components

- Booch divides objects into actors, agents, and servers
  - Actors are object that perform actions which influence other objects in the system
  - Servers are the recipients of an actor’ s activity and related to the database entity concept
  - Agents are an amalgam of both characteristics

3. Evaluation Procedure

- Object-Oriented meta-model
  - Conceptual modeling
  - Procedure and Guidance
  - Transformations and products

3. Evaluation Procedure

- Conceptual modeling
  - The data and processing control parts of a system are modeled in one unit rather than separately
  - The method produces model of objects commuicating by messages
  - Classification of objects is supported with property inheritance

3. Evaluation Procedure

- Procedure and Guidance
  - The method should guide the analyst towards identifying and describing objects
  - Guidance should be available for analysis, specification and design phases

3. Evaluation Procedure

- Transformations and products
  - Design transformation should support change of OO specifications into designs implementable in OOP languages

4. Object-Oriented Methods

- HOOD(Hierarchical Object-Oriented Design)
- OOSD(Object-Oriented System Design)
- OOSA(Object-Oriented Systems Analysis)
- OOA(Object-Oriented Analysis)
- ObjectOry

4. Object-Oriented Methods

- HOOD(Hierarchical Object-Oriented Design)
  - Objects are modeled in a hierarchical manner.
  - Supports object classes, but inheritance and reuse are not made explicit.
  - Incorporates many OO properties.
  - Real time-method -> data specification and associated inheritance receive less attention.

4. Object-Oriented Methods

- OOSD(Object-Oriented System Design)

- Provides detailed notation for object classes and management of inheritance.
- Supplies Detailed notation for interface description and encapsulation.
- Inter-object communications (event/message types)
- No analysis advice is given.

4. Object-Oriented Methods

- OOSA(Object-Oriented Systems Analysis)

- Many heuristics for object identification and analysis
- Models an object relationship network with subclasses
- Produces a composite activity-data model
- Lack of support for inheritance and reuse is not explicitly supported

4. Object-Oriented Methods

- OOA(Object-Oriented Analysis)

- Covers all OO concepts, although analysis method only.
- Classification and inheritance are modeled and abstraction is helped by the structure layer
- Uses hierarchical inheritance.
- Specification of encapsulation and object interfaces is not as detailed as OOSD, or HOOD.

4. Object-Oriented Methods

- ObjectOry

- Supports OO concepts of classification, encapsulation and inheritance.
- Adds "use cases" to the OO approach.
- Reuse is supported by component libraries
- Guidance for analysis is less comprehensive.
- Target applications: like HOOD real-time systems and engineering systems.

4. Object–Oriented Methods

- Summary of OO methods

Method	Abstraction	Classification	Inheritance	Encapsulation	Coverage (R-A-S-D-I)
HOOD	Y	Y	Partial	Y	------
OOSD	Y	Y	Y	Y	------
OOSA	Y	Partial	-	-	-----
OOA	Y	Y	Y	-	------
ObjectOry	Y	Y	Y	Partial	-------

Key: Y = Yes.
R-A-S-D-I in coverage refers to Requirements Analysis. Analysis, Specification, Design, and Implementation. The measure of coverage is judged from the methods procedures and notations.

- The coverage of OO methods is Variable and not all methods meet the necessary range of criteria.
- No complete object oriented method exists.

5. Structured Methods

- IE
- ISAC
- SASD
- SSADM
- SADT
- JSD
- NIAM
- Mascot-3

5. Structured Methods

- IE(Information Engineering)

- Encourage data modeling
- Functional specification uses process dependency and action diagram, separated from data modeling

5. Structured Methods

- ISAC(Information System Activity and Change Analysis)
- ISAC advocates top-down functional decomposition of processing and data

5. Structured Methods

- SASD(Structure Analysis/Structured Design)
- SASD use top-down functional decomposition to analysis system in terms of a network of processes connected by dataflow messages

5. Structured Methods

- SSADM(Structrued Systems Analysis and Design Method)
- SSADM is a composite method derived from structured analysis, structured design and data analysis.

5. Structured Methods

- SADT(Structured Analysis and Design Technique)
- JSD(Jackson System Development)
- NIAM(Nijssen’ s Information Analysis Method)
- Mascot-3

5. Structured Methods

- Summary of OO methods

- Methods using functional decomposition encourage identification of goal related components in systems.
- OO approach promotes system components more compatible with data models.
- Functionally oriented analyst will identify different modules from OO analyst.
- Current structured methods using an entity-modeling and/or entity life history have potential to evolve towards OO.

5. Structured Methods

Summary of method specification models and approaches

Method	Functional process	Data relationship	Event sequence	Coverage (R-A-S-D-I)	Application
IE	Y	Y	Y	----------	IS
ISAC	Y	Y	N	------	IS
SASD	Y	N	Y	---------	IS
SSADM	Y	Y	Y	----------	IS
SADT	Y	Y	N	------	IS. RT
JSD	N	Y	Y	----------	IS. RT
NIAM	Y	Y	N	---------	IS (data intensive)
Mascot	Y	N	N	---------	RT

Key: Y = Yes, N = No.
Coverage of the life-cycle: Requirements (R), Analysis (A) Specification (S), Design (D), Implementation (I).
Application: IS = information systems, RT = real-time.

Summary of structured methods' object-oriented features

	Object model	Data + activity	Encapsulation	Types + instances	Classification
IE	Poss	N	N	Y	N
ISAC	Y	N	N	N	N
SASD	Y	N	N	N	N
SSADM	Y	N	N	Y	N
SADT	Y	N	N	N	N
JSD	Y	Y	Y	Y	N
NIAM	Poss	Poss	N	Y	Y
Mascot	Y	Y	Y	Y	N

Notes:
(1) For the object model, Poss means an object model could possibly be constructed from the data model in these methods.
(2) To score Y for the object model, methods have to specify a concurrent network of message-passing processes, however these processes may be functional or data-oriented. This can be cross-checked on column two, which records whether data and processing are modelled together in an object.

6. Conclusion

- Use of a particular system development method will bias implementation of OO systems, OO design may not derived from any specification
- Data model and OO specification show considerable convergence. It is feasible to migrate from structured method such as JSD, IE and SSADM to OO Method.
- Functionally based development methods are less well suited to development of OO system.
- OO methods have yet proven in practice, they have little CASE tool support, lack of modeling techniques for reuse system development.