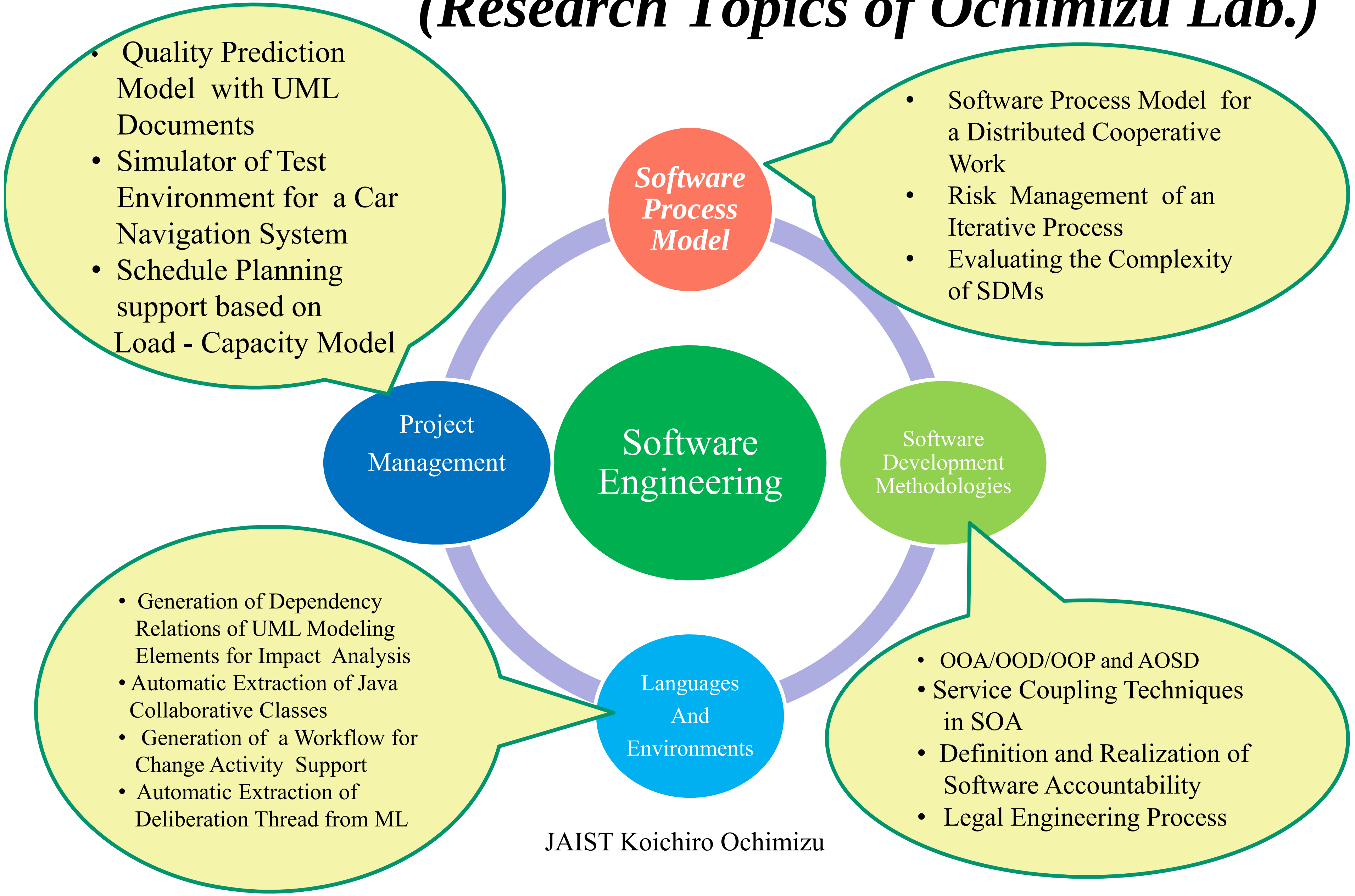
Software Design Methodologies Goal and Scope

Koichiro Ochimizu
Japan Advanced Institute of Science and Technologies
School of Information Science

Supporting Software Development and Evolution (Research Topics of Ochimizu Lab.)


Software Engineering
Software Process Model
Project Management
Software Development Methodologies
Languages And Environments

- Quality Prediction Model with UML Documents
- Simulator of Test Environment for a Car Navigation System
- Schedule Planning support based on Load - Capacity Model

- Software Process Model for a Distributed Cooperative Work
- Risk Management of an Iterative Process
- Evaluating the Complexity of SDMs

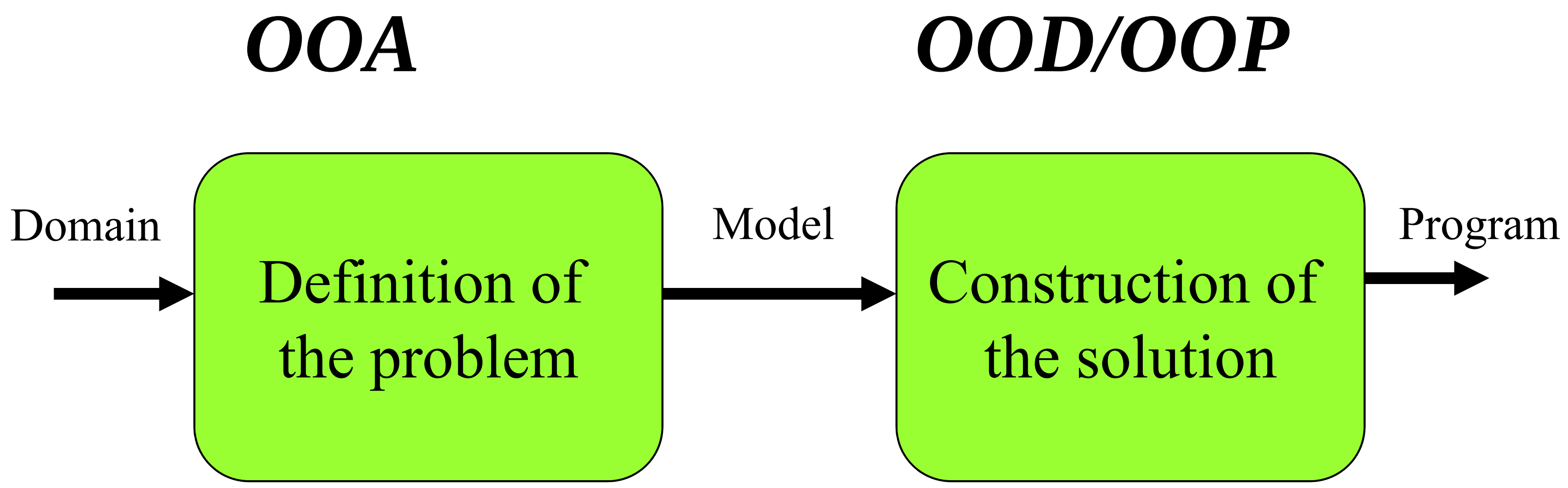
- Generation of Dependency Relations of UML Modeling Elements for Impact Analysis
- Automatic Extraction of Java Collaborative Classes
- Generation of a Workflow for Change Activity Support
- Automatic Extraction of Deliberation Thread from ML

- OOA/OOD/OOP and AOSD
- Service Coupling Techniques in SOA
- Definition and Realization of Software Accountability
- Legal Engineering Process

JAIST Koichiro Ochimizu

Object Oriented Analysis/Design/Programming

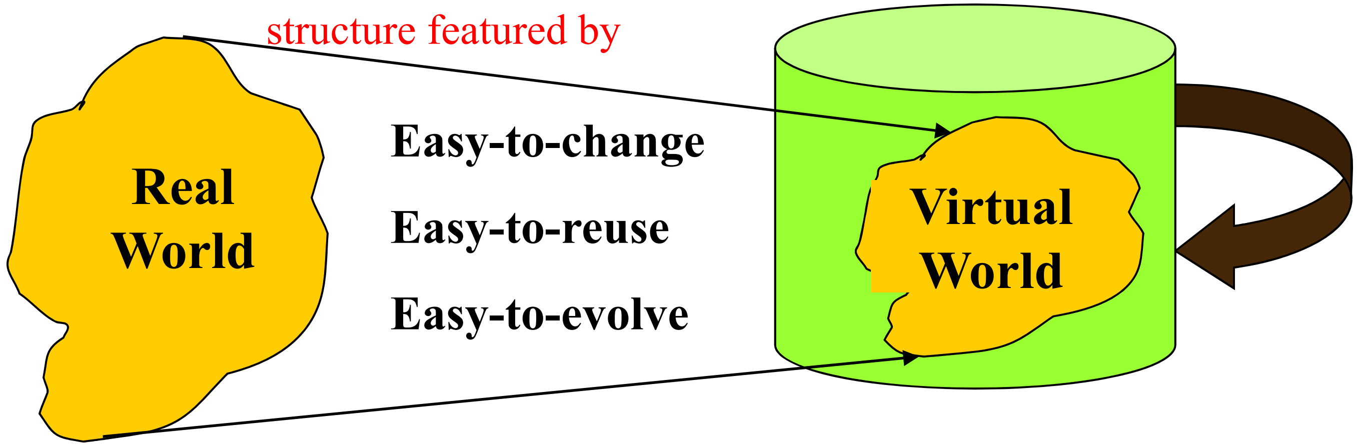
Iterative and Incremental



Three major advantages of OO Technology

- *Project the real world into the computer as you recognize and understand it.*
- *Maintain the virtual world constantly corresponding to mismatches between the real world and the virtual world and evolution of the real world.*

Enable us to realize the software structure featured by



Scope and Goal

- **Goal** enable students to understand of basic principles and concepts of OOT and their application to practical use.
- **Content**
  - Basic Principles and Concepts (object, class, association, message passing, inheritance)
  - Modeling Techniques(Static Modeling, Dynamic Modeling)
  - Modeling Language(UML) and Programming Languages(Java)
  - Object-oriented Software Development Method（Unified Process, COMET）
  - Aspect Oriented Software Design(AOSD)
- **Case Studies**
  - Real-time system Analysis and Design: Elevator Control System
  - Product Line Design: Microwave Oven
  - Aspect Oriented: Hotel Reservation System
- **Contribution of OOT in SE field**

Contents(1)

- **Goal and Scope**
- **Basic Concepts on OOT**
  – Basic Concepts to represent the world
  – Basic Concepts for Reuse
  – Information Hiding Principle and Java Program
  – Superiority of OOT
- **Modeling Techniques**
  – Static Model: Class and Association
  – Dynamic Model: State Machine
  – Dynamic Model: Interaction Diagram
  – Concurrency Description: Active Object and Multi-thread Programming
  – Outline of UML2.0

Content(2)

- **Object-oriented Software Development Methodology**
  - Outline of Unified Process and Use-case Driven Approach
  - Elevator Control System:
    Problem Description and Use-case Model
  - Elevator Control System:
    Finding of Problem Domain Objects
  - Elevator Control System:
    Sub-System Design and Task Design
  - Elevator Control System:
    Performance Evaluation
- **Product Line Technology**
  - Feature modeling
- **Aspect Oriented Software Design**
- **Contribution of OOT in Software Engineering**
  - History of SE Technologies and Contribution of OOT
    in SE field

Important Concepts to be studied

- Class and Instance（O.J. Dahl SIMULA67,1967)
  - Removal of redundant description
- Information Hiding Principle（D.L.Parnas)
  - Easiness of modifying a data structure
- Abstract Data Type
  - Both
- Inheritance
  - Reuse of classes by sub-classing
  - Easiness of extension of functions by sub-typing
- Polymorphism
  - Dynamic binding
- Use of the same concepts through analysis, design and programming
  - Simple correspondence among software artifacts
- Handling cross-cutting concerns by Aspect

Object-Oriented Programming

- 1967: Simula by O.J. Dahl Class and Instance
- 1972: Parnas Module by D.Parnas Information hiding
- 1972: Smalltalk72(Xerox PARC)
- 1977: CLU by B. Liskov abstract data type
- 1981: Smalltalk80 by Xerox class library
- 1986: Objective-C by Cox, C++ by Strusrup
- 1988: Eiffel by B. Meyer
- 1989: CLOS by Moon
- 1995: Java
- 1997: AOP by Gregor Kiczales

Object-Oriented Technologies (Object Oriented Analysis and Design)

- 1986: OOD by G. Booch
- 1988: Shlare/Mellor,
- 1991: Coad/Yordon,
- 1991: OMT by J.Rumbaugh
- 1992: OOSE by Ivar Jacobson
- 1993-1994: Design Patterns by GoF
- 1997: CBSE by Szyperski
- 1997: UML
- 2004: UML2 & MDA
- 2005: AOSD by I.Jacobson

References

•Walker Royce, “Software Project Management A Unified Framework”,

ADDISON-WESLY, 1998.

• Ivar Jacobson, James Rumbaugh, Grady Booch: The Unified Software Development Process, Addison-Wesley, (1999).

・ Hassan Gomaa, Designing Concurrent, Distributed And Real-Time Application with UML, Addison Wesley, (2000).

•Hassan Gomaa, Designing Software Product Line with, Addison Wesley, (2004).

•Ivar Jacobson, Pan-Wei Ng: Aspect-Oriented Software Development With Use Cases, Addison-Wesley, (2005).

•James Rumbaugh, Ivar Jacobson, Grady Booch: The Unified Modeling Language Reference Manual, Second Edition, Addison-Wesley, (2005).

References

- HansErik Eriksson, Magnus Penker, Brain Lyons, David Fado, "UML 2 Toolkit", Wiley Publishing, Inc. 2004.
- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, "Design Patterns", Addison-Wesley Publishing , 1995.
- Martin Fowler, " Analysis Patterns: Reusable Object Models" Addison-Wesley Publishing Company, 1997.
- Frank Bushmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, "Pattern-Oriented Software Architecture --- A System of Patterns" John Wiley & Sons, 1996
- Douglas Schmidt, Michael Stal, Hans Rohnert, Frank Bushmann, "Pattern-Oriented Software Architecture Vol.2 --- Patterns for Concurrent and Networked Objects" John Wiley & Sons, Ltd, 2000.