Development of Rigorous Service-driven Applications

Dr. Nasreddine Aoumeur
FIN, ITI, DB group
aoumeur@iti.cs.uni-magdeburg.de

Course Web-Site:
wwwiti.cs.uni-magdeburg.de/~aoumeur
wwwiti.cs.uni-magdeburg.de/iti_db/lehre/soa
Planned Agenda

1. **General Introduction** to Information Systems (IS) and their development
2. **Object paradigm** to IS semi-formal modelling:
   - UML-class and object-diagrams,
   - Use-cases and Sequence-diagrams.
3. **Service technology and Business Process: Introduction**
4. **(High-level) Petri nets** for Modelling service-driven applications
5. **Advanced techniques** to **Evolving** service-driven applications:
   - Architectural techniques
   - Business rules
   - Aspect-oriented concepts
Information Systems: Some definitions

- Set of interrelated *(software and human)* components that:
  - collect, process, store and distribute information
  - to (semi-automatically) support decision making in an *organization*.

- Information systems in *large organizations* are:

  **Reactive systems**
  - Provide a number of *services* on the basis of the organization *knowledge*
  - React upon *interactions with their environment*, and
  - Act according to *(business) rules/procedures* defining
    » the *organization* overall functioning and behaving, and
    » Relation to the external world (e.g. government regulations, market laws, cross-organization relations)
Information Systems: Working definition

Information systems are

- *reactive* systems (i.e. in continuous interaction with their environment), with

- *large amount* of *immutable and non-immutable data* (i.e. fixed and changing) and, with

- *processes and activities* for exhibiting behaviors on these (state-less and -full) data.
Information Systems: Graphical show

Environment

Potential Users
(e.g. Private / Corporate)

Organizations

Society

Information system

State-less and -full DATA

Processes and Rules

Managerial / organizational / human / Business concerns

messages

events

request

provide
### Information Systems: Their Categories

- **Process-aware IS:** when the focus is more on the **process side**
  - Business processes / workflows (causal activities) / service-driven applications
  - **Example:** Process to enroll at the university, to get ensured / refunded, etc …

- **Data-aware IS:** when the focus is more on the **data side**
  - Databases, data-warehouse,

- **Web-aware IS:** when the focus is more on the **(online) service side**
Information Systems: Illustrations

Environment

Students
Teachers
Readers (public/private)

Library system

State-less and -full DATA

Book:
- author(s)
- copie(s)
- year
- status
- ...

Student:
- name
- card-number
- status
- ...

Author:
- name
- ...
- status
- ...

Processes and Rules

Rule1: constraints for lending
Rule2: constraints for editing...
Rule3: ...

process1: subscribe ->
borrow -> return
process2: publish -> copyright,..
Rule3: penalties....
Information Systems: Illustrations

Environment

Users

Banks

Actions (public/private)

Banking system

State-less and -full DATA

Account:
- number
- balance
- limit
- status
- ...

Customer:
- name
- holder-account
- credit
- ...

Mortgage:
- name
- ...
- status
- ...

Processes and Rules

rule1: how to withdraw
Rule2: to use credit-card...
Rule3: ...

process1: mortgage...
Process2: insert \(\rightarrow\) transact\(\rightarrow\) ...
Process3: ...
Information Systems: Illustrations

Environment

Financial Organizations

Actions (public/private)

Staff-Payment System

State-less and full DATA

Admins:
- number
- balance
- limit
- status
- ...

Employee:
- name
- holder-account
- credit
- ...

Invoice:
- name
- ...
- status
- ...

Processes and Rules

rule1: how to advance
Rule2: how to be paid...
Rule3: ...

process1: probation → ...
Process2: ...
Process3: ...

Payroll
advance
vaccation
action
Web-based or Service-driven Informations: Illustrations

User

Look for a trip

Get a trip

Travel Agency

Look4trip
Find4trip
Confirm-trip

Banking Interface
Accept-card
Debit-account

AirLines Interface
Book-ticket
Confirm-flight

Hotels Interface
Book-room
Confirm-room

Composition-Level
(XML-based) BPEL / WS-CDL

(XML-based) SOAP---SOAP---SOAP---SOAP

Interfaces WSCD

I.S.1 : Banking
Hidden-Software

I.S.2 : AirLines
Hidden-Software

I.S.3 : Hotels
Hidden-Software
But, how do we know what provider does upon a request, what operations must be called by requestor in which order to achieve a certain goal?

→ Business processes describe what is done when and by whom
Many Many other IS ...

- Insurance information system
- (E-) government information system
- (E-) health information system
- (E-) Banking systems
- Military system
- Automotive/avionic systems
- ...
So what is at stake ????

Advanced pg. Languages
- JAVA / C++ / ...
- AOP, COM, ...

aspect activity1 {
  pointcut chg1 = call (* .. Within.)
  before : ...
  After : ...
  aspect activity2 {.....
  around : call(....) }

Public static main {
  activity1(...); activity2(....)
  If ...while switch (activity_j)
  "....."

Problems !!!!!
- No certification
- No communication
- NO EVOLUTION

Organization
- Goals
- Resources
- General functioning
- Interfaces with outside

Domain / UoD
Rigoros and Adaptive 
... Service Applications
IS Development Life Cycle Sketch

1. Business Modeling
   Initial Planning
2. Requirements
   Planning
3. Analysis & Design
   Config. & Change Management
4. Implementation
   Test
5. Deployment
6. Evaluation
Information systems life-cycle

Informal description of the Universe of discourse
- Questionnaires
- Interviews
- Gathering and analysis of existing manual doc.
- Understanding of rules and procedures

Precise description:
Conceptual Modeling
- Description of structural aspects
- Description of behavioural aspects
- Validation of the system
- Verification of the main system properties

Implementation
- Efficient implementation in the appropriate language
- Test and deployment
Quality criteria for IS development

• **Adequacy (completeness):** An information system should represent all the desired and needed aspects of the universe of discourse:
  – a faithful correct mirror of its target organization.
  – *Animations* and construction of *prototypes* and simulation help towards adequacy.

• **Correctness:** An information system (the implementation) is correct if it does not violate its conceptual modelling or specification.
  – An indispensable prerequisite here is the availability of a *mathematical* formal specification.
Quality criteria for IS development

- **Verifiability**: The main properties of the system should be easy to verify against the specification.
  - *Formal analysis techniques* are to be used with division of concerns, so that different system units could be independently verified.

- **Modifiability**: As requirements change over time, the system should be flexible to meet the new requirement.
  - Evolution with meta-reasoning is at stake
Quality criteria for IS development

- **Reusability**: Reusing similar parts of the system instead of developing a new specification from scratch.
  - This requires a formal specification with a full determination of the impact of modification of reused parts.
- **Integrity**: Conditions that must be guaranteed in every state of the system and/or in a sequence of states.
- **Safety**: It states what should be always be guaranteed by the system (in execution).
IS development conceptual principles

- **Rigor**: A conceptual modelling should be as formal as possible.
  - Formality induces preciseness and makes possible analysis of specification.
  - Formality, however, is hard for clients and systems managers. Graphical notations could help for enhancing understandability.

- **Modularity (composability)**: A complex information system consists usually of a number of subsystems or components. On the conceptual level, we must be able to:
  - (1) decompose a system into components;
  - (2) compose components to form a system; and understands components in isolation.
IS development conceptual principles

- **Abstraction**: Identify the important aspects of a system and to ignore irrelevant details. Abstract from implementation-details and from reality: decide which aspects are relevant.

- **Incrementality**: proceed in a stepwise way. That is, from an initial design (an increment) we must be able to expand it by additional properties and to refine it with more detail.
IS development conceptual principles

• **Understandability**: A specification must be understandable. That is, the specification formalism should enhance expressiveness by allowing to express systems’ aspects as natural as possible. The use of graphical symbols help a lot.

• **Modifiability**: As the conceptual modelling is the only vehicle for interactions between developers (system’s analysts) and clients and due to the gradual and volatile emergence of requirements, the specification should allow adaptation.
IS development conceptual principles

• **Structural and behavioural Modelling**: In a specification of a model of the problem domain, we must be able to specify *structural* and *behavioural aspects*. Entity-relationships, Z, NIAM, *abstract-data types* allow describing data and its transformations. CCS, CSP, B, temporal logic, *state-charts*, *Petri nets* allow describing behavioural aspects.

• **Declarative and operational specifications**: Behaviour can be specified in several styles.
  - The *declarative* style employs formulas to *constrain* the possible behaviour: valid states, valid changes, and admitted behaviour. It suffer from the *frame problem*: we have to explicitly exclude the indesired behaviour.
IS development conceptual principles

- The *operational* style employs explicit control through calls and triggers to execute operations that modify information. No frame problem but no abstract description of admissible behaviour.

- **Top-Down and Bottom-Up Modeling** : The usual way in requirements analysis is to employ a *top-down* approach. That is, global objectives are identified as a whole, then different units are singled out. *Bottom-up* modeling means to analyse the problem in order to identify required components, select them from library and adapt or create them and assemble them to form a system. In practice, important is to combine both approaches.