Software, Process, Business Process and Software Process

Tao Xie

Research work was done while being a summer intern at Software Technology Research Department
Avaya Labs Research
Outline

• Process Conceptual Framework
• Software & Process
• Business Process & Software Process
• Process Program Design
• PASTA Process Modeling & Automation
• Conclusions
Background

• Leon J. Osterweil, Software Processes are Software Too, ICSE 9 (Awarded as Most Influential Paper 10 years later)
  
  Software ↔ Software Process

• Our approach:
  
  → Business Process
  
  Software ↔ Process | → Software Process

Using PASTA (Process and Artifact State Transition Abstraction) to model and automate

• FAST (Family-Oriented Abstraction, Specification, and Translation)
• SARB (Software Architecture Review Board)
• Business Processes (workflow)
Process Conceptual Framework

Role

Work On

(Manual) Activity

Control flow

(Automated) Activity

Data flow

Artifact

Data flow

Process Abstraction
Software and Process

Role

I/O

Work On

Method/Component

(Manual) Activity

Control flow

Data flow

Method Call/Connector

(Automated) Activity

Method/Component

Artifact

Data variable / Data structure

Inter-procedural dataflow

Control flow in the end of method; Continuation
Concurrency
Data flow and control flow are relatively independent

Process Abstraction
Process Program vs. Conventional Program

- High flexibility and customizability
- Process flow (control flow) is explicitly defined and enforced.
- Process flow instance (artifact instances) are explicitly distinguished and maintained automatically

Process Program:
- Process Modeling
- Process Flow Engine
- Build time
- Runtime

Conventional Program:
- Process Modeling Tools
- Model
- Process Enactment

Process-Centered Environment

Process Automation
Business Process vs. Software Process

- More iterations
- Higher organization flexibility
- More non-intrusive support
- More complex artifacts
- Different process instances need modification of process
- Few concurrent multiple process instances
- Artifact instance version control
Process Correctness

• “Are we building the right process?”
• “Are we building the process right?”
• “Are we building the process the participants are willing to follow?”

• Process Definition
• Process Enactment
Modeling Language Design Principles

• Expressiveness vs. usability
• Activity decomposition
  • Process vs. control abstraction
• Control flow
  • Event-based vs. state-based
• Data flow
  • Param passing vs. Artifact association
Process Design Principles

- Guidance vs. non-intrusiveness
  - Control constraints among activities
  - Process abstraction level
  - Artifact representation option
- Incremental top-down vs. bottom-up
- Information hiding- artifact abstraction
- Process evaluation
  - Process language & process
More on Process Design

• Modeling is Not Enough
  • Model checking/analysis
  • Process enactment or simulation

• Support What Can Be Supported
  • Preparation meeting
  • Different forms of software artifacts - Integrator
PASTA (Process and Artifact State Transition Abstraction)
PASTA Conceptual Framework

Role in hierarchy

Form Artifact

Role list for P-state

GUI

Artifact List

for P-state

Entrance
Cond

Exit Cond

P-state (fill)

State-of(A) = empty

P-state (check)

State-of(A) = filled

Referenced Artifact List

for P-state

Artifact A

A-state machine:

empty

fill

filled

check

checked

State-of(A) = filled

State-of(A) = checked
P-State Machine -> Petri Net

- **Operation as transition**
  - **Role as triggering of transition**
  - **state-of(Economic_Model)=Reviewed**

  State of place as place

  **Domain Manager**
  - **Analyze_Data**
  - **Reject**
  - **Accept**

  **Domain Manager**
  - **state-of(Family_Artifact:1)=Domain_Rejected**
  - **state-of(Family_Artifact:1)=Domain_Qualified**

  **Domain Engineer**
  - **Gather_Data**

  **state-of(Economic_Model)=Reviewed**
SARB PASTA Artifact Hierarchy

- SARB_Artifacts
  - Input_To_The_Review
    - Problem_Statement
    - Functional_Requirements
    - System_Requirements
    - Architecture_Specification
    - Informational_Artifacts
  - Supporting_Artifacts
    - Project_Preparation_Information
    - Review_Preparation_Information
  - Outputs_Of_The_Review
    - Snow_Card
    - Recommendation_Card
    - SARB_Written_Report
    - Management_ALERT_Letter

- Snow_Card
  - Login_Section
    - Issues_Format_Select
      - Issues_Format_Select
        - Internal_Document
          - Internal_Document_Section
            - Text_Section
              - Issues_Text
            - External_Document
              - External_Document_Section
                - Issues_Document
                  - File_Location
                  - File_Type
  - Analysis_Section
SARB_PASTA Artifact State Machine

SARB_Preparation_Report

[Meeting_Review_Choprson_And_Project_Management]

ready

delivered

[Send_Preparation_Report]

sent

Artifact Definition Form - SARB_Preparation_Report

Name: SARB_Preparation_Report
Description:
Complexity: Composite
Data Type: Form
State List: delivered
            ready
            sent
Sub-Artifact List:
Report_Format_Section
Relation List:
Attributes:

PASTA Editor Version 2.0 - Artifact Definition Form - SARB_Preparation_Report
SARB PASTA P-state Hierarchy
Create_Preparation_Report

Send_Preparation_Report

Meeting_Review_Chiperson_And_Project_Management

Exit Condition: state::(SARB_Preparation_Report:255)=sent
Ready

Artifact List:
SARB_Preparation_Report

Referenced Artifact List:
SARB_Preparation_Report

Sub-Operation List:
Meeting_Review_Chiperson_And_Project_Management
Send_Preparation_Report

Analysis List:

Conditional Action List:

Reference Parameter (Artifact or Constant):

Role Name:
Review_Chiperson
Project_Manager

Entrance Condition:
state::(SARB_Preparation_Report:255)=ready

SARB PASTA P-state Machine
## Analogy between PASTA and Software

<table>
<thead>
<tr>
<th>PASTA Process</th>
<th>Traditional Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-state</td>
<td>Component/Subsystem/Method</td>
</tr>
<tr>
<td>P-state hierarchy</td>
<td>System decomposition hierarchy</td>
</tr>
<tr>
<td>Artifact</td>
<td>Data variable/Data structure</td>
</tr>
<tr>
<td>Artifact hierarchy</td>
<td>Data structure hierarchy</td>
</tr>
<tr>
<td>Artifact-state (control state)</td>
<td>Actual value of data variable</td>
</tr>
<tr>
<td>P-state machine</td>
<td>Software architecture with different scope and abstraction</td>
</tr>
<tr>
<td>Transition in P-state machine</td>
<td>Connector in architecture/Method call between methods</td>
</tr>
<tr>
<td>A-state machine</td>
<td>Inter-procedural program slicing for certain data variable</td>
</tr>
<tr>
<td>Petri-Net like executable state transition abstraction</td>
<td>Static function call graph</td>
</tr>
<tr>
<td>Runtime dynamic state transition</td>
<td>Dynamic function call graph</td>
</tr>
</tbody>
</table>
PASTA Main System Components
Conclusion

• Software and Process
• Process Program and Conventional Program
• Business Process and Software Process
• Process Program Design
• PASTA Modeling and Automation