

COMP3029J

Software Architecture

Software Architectural Styles (Virtual Machine Style)

DENG, YONGJIAN

Faculty of Computer Science, BJUT

Data Mining & Security Lab (DMS Lab)



北京工业大学
BEIJING UNIVERSITY OF TECHNOLOGY

Outline

1. Virtual Machine Style

2. Interpreter

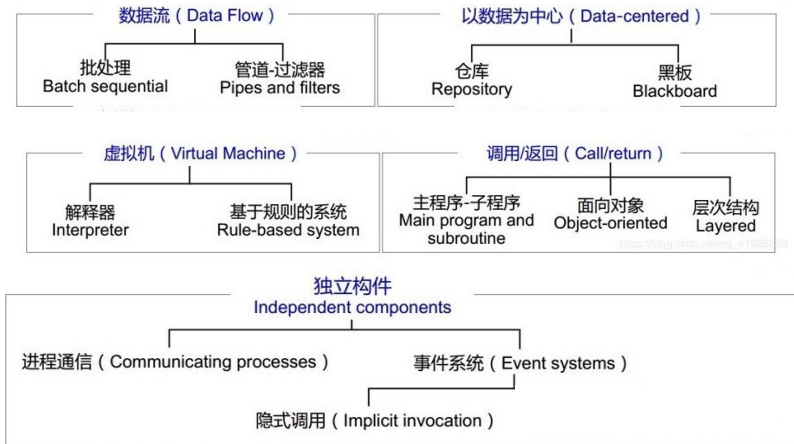
3. Rule-based System

4. Summary



北京工业大学
BEIHANG UNIVERSITY OF TECHNOLOGY

Taxonomy of styles



Outline

1. Virtual Machine Style

2. Interpreter

3. Rule-based System

4. Summary



Virtual Machine Style

Interpreters

- Simulate functionality which is not native to the hardware

Rule-based systems

- Specialization of an interpreter

Other

- Syntactic shells
- Command language processors



Outline

1. Virtual Machine Style

2. Interpreter

3. Rule-based System

4. Summary



北京工业大学
BEIHANG UNIVERSITY OF TECHNOLOGY

Interpreter Style

Problem: This pattern is suitable for applications in which **the most appropriate language or machine for executing the solution is not directly available**. The pattern is also suitable for applications in which the core problem is defining a notation for expressing solutions, for example as scripts. Interpreters are sometimes used in chains, **translating from the desired language/machine to an available language/machine in a series of stages**.

Context: The interpreter will most often be designed to bridge the gap between the desired machine or language and some (possibly virtual) machine or language already supported by the execution environment.



Interpreter Style

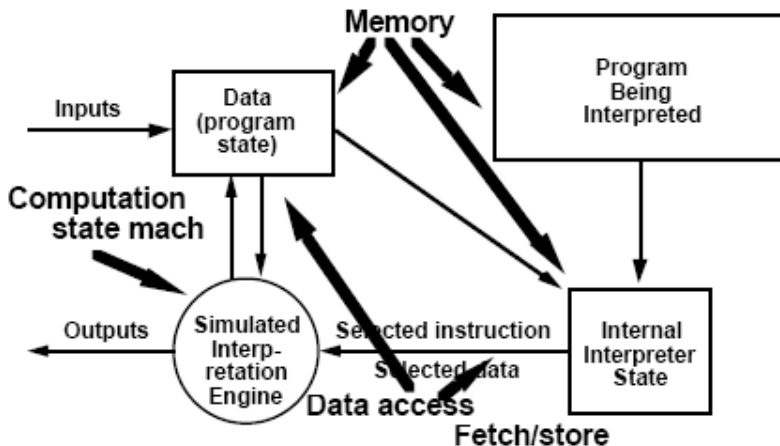
Solution

- System model: virtual machine
- Components: **one state machine** (the execution engine) and **three memories** (current state of execution engine, program being interpreted, current state of program being interpreted)
- Connectors: data access and procedure call
- Control structure: usually state-transition for execution engine; input driven for selection of what to interpret

Significant Variants: Expert systems are often implemented as interpreters for the collections of rules, or productions, that represent the expertise. Because the productions require a complex selection rule, specialized forms of interpreters have evolved.



Interpreter Style



Advantages of Interpreter Style

Functionality

- Can simulate non-native functionality

Testing

- Can simulate “disaster” modes (e.g. for safety-critical applications)

Flexibility

- Very general-purpose tool



Disadvantages of Interpreter Style

Efficiency

- Much, much slower than hardware
- Much slower than compiled system

Testing

- Additional layer of software to be verified



北京工业大学
BEIJING UNIVERSITY OF TECHNOLOGY

Applications of Interpreter Style

Interpreted language

- VB, Javascript, VBScript, HTML, Java bytecode, Matlab
- scripts, configuration files

Communication Protocol

user input

- Key combination in game



Outline

1. Virtual Machine Style

2. Interpreter

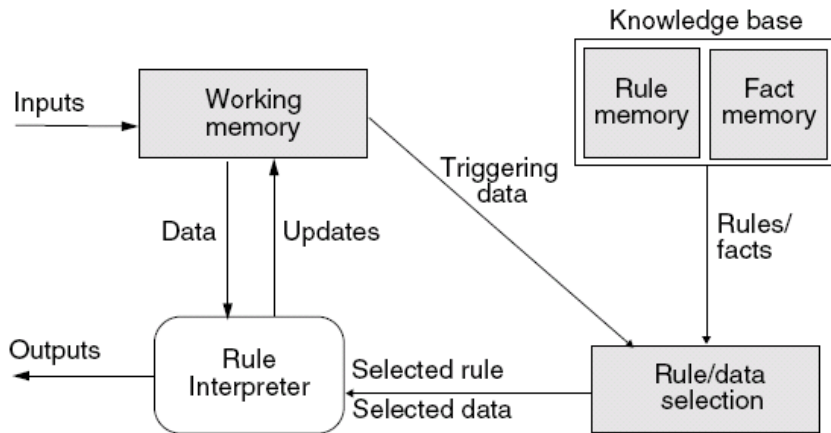
3. Rule-based System

4. Summary



北京工业大学
BEIJING UNIVERSITY OF TECHNOLOGY

Rule-based System



Characteristics of Rule-based System

Code to be executed (knowledge base)

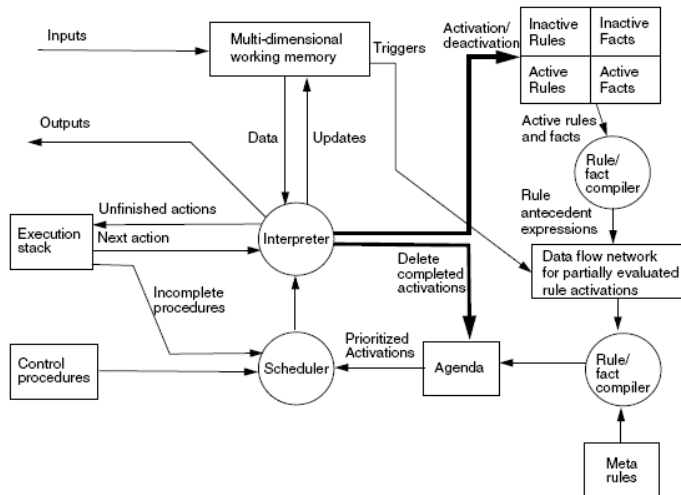
Interpretation engine (rule interpreter)

Control state of interpreter (rule/data selection)

Current state of the code (working memory)

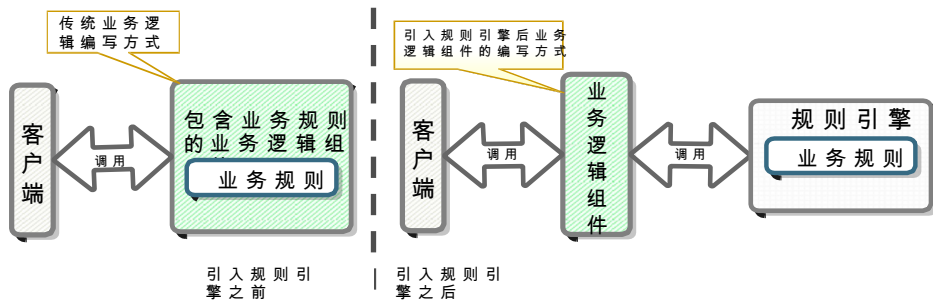


Complex Rule-based System (Example)



Application of Rule-based System

- Drools is the JBoss business logic intelligence module of Redhat
- www.jboss.org/drools/
- Drools is an expert system for processing rules



Application of Rule-based System

- **Declarative programming**
 - Rules engine allows you to say "what to do" instead of "how to do it"
- **Rule-based system is able to solve very difficult puzzles**
- **Logic and Data Separation**
- **Fast and flexible**
- **Centralization of knowledge**
- **Tool integration**
- **Good explanation mechanism**
- **Easy to understand rules**



Application of Rule-based System

Java语言表示，如果有一个人的名字是“Joe”，而且是个男性，就会输出他的名字跟性别。

```
If ( "Joe".equal( people.getName() ) )
{
    if( "Male".equal( people.getSex() ) )
    {
        System.out.println("This is a man, name is Joe.");
    }
}
```

```
rule "GoodBye"
when
    People( name = "Joe", sex = "Male" )
then
    System.out.println(("This is a man, name is Joe.");
end
```



Background of Drools

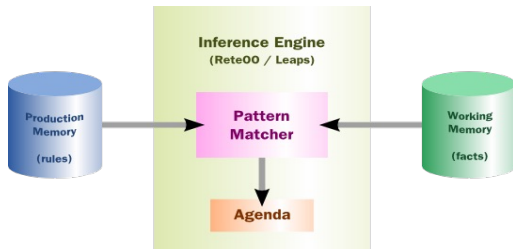
- **Huge system and complex business**
- **Business rules change frequently**
- **24-hour service**
- **Unified business management**
- **Reduce system maintenance and upgrade costs**



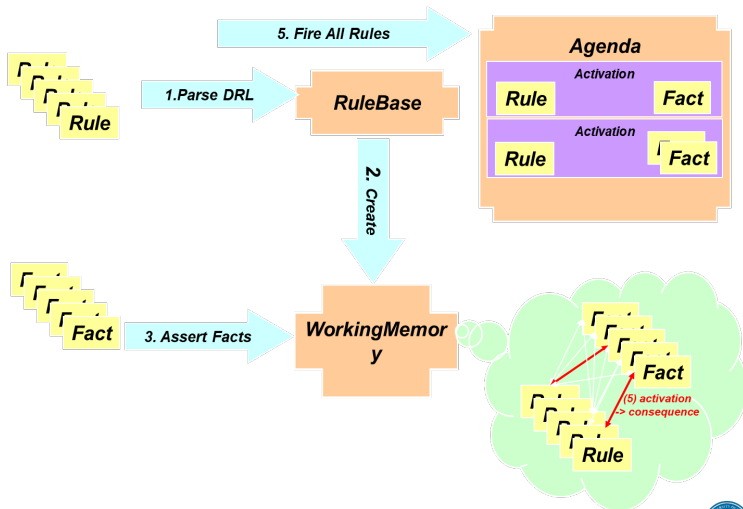
北京工业大学
BEIHANG UNIVERSITY OF TECHNOLOGY

Background of Drools

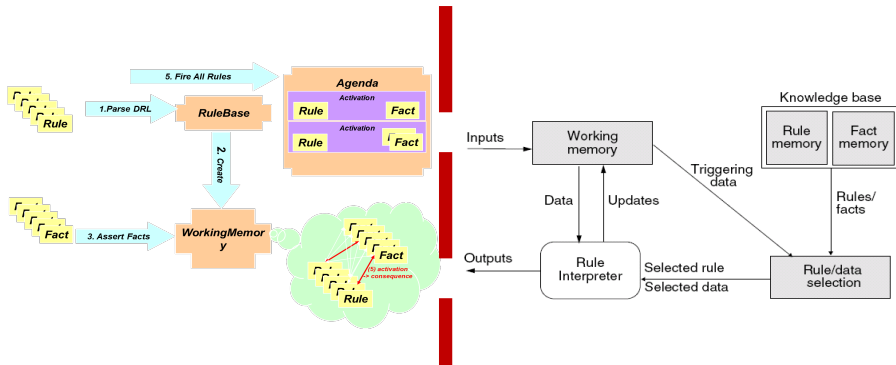
- The brain of many production rule-based systems is actually an inference engine that matches **Facts** and **Rules**
- When a match is found, the **Action** corresponding to the rule will be **triggered** (**Fire**)
- **Action**
 - Often change the state of the fact, or
 - Perform some "external" action on the application



Architecture of Drools



Drools & Rule-based System



Outline

1. Virtual Machine Style

2. Interpreter

3. Rule-based System

4. Summary



北京工业大学
BEIJING UNIVERSITY OF TECHNOLOGY

Summary

Interpreters (解释器)

- Simulate functionality which is not native to the hardware

Rule-based systems

- Specialization of an interpreter

Other

- Syntactic shells
- Command language processors



Summary

Interpreters (解释器)

- Simulate functionality which is not native to the hardware

Rule-based systems

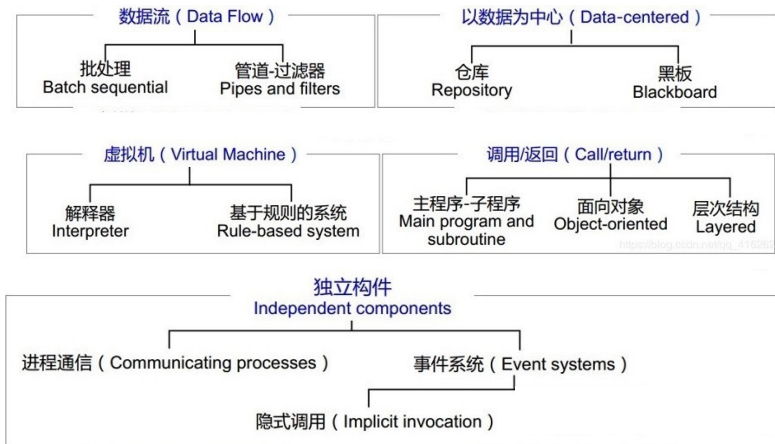
- Specialization of an interpreter

Other

- Syntactic shells
- Command language processors



Summary



Thank you!