

Business Logic

Cordys BOP 4.1 Fundamentals

CORDYS

- ◆ Event Driven Architecture
- ◆ Business Rules
- ◆ Business Rules in Cordys

- ◆ Event Driven Architecture:

- ◆ Events Triggers Action

- ◆ Periodic

- ◆ Periodic check if action is needed

- ◆ Event Driven Architecture

- ◆ More Efficient
 - ◆ More Agile
 - ◆ More Chaotic
 - ◆ Less Predictable

◆ Quality of Business Service:

- ◆ The ability to react fast and consistent on business events

◆ Business Events examples:

- ◆ New Quote
- ◆ New Customer
- ◆ Order is Cancelled
- ◆ Price Change
- ◆ Stock Change
- ◆ Transportation of supplies starts
- ◆

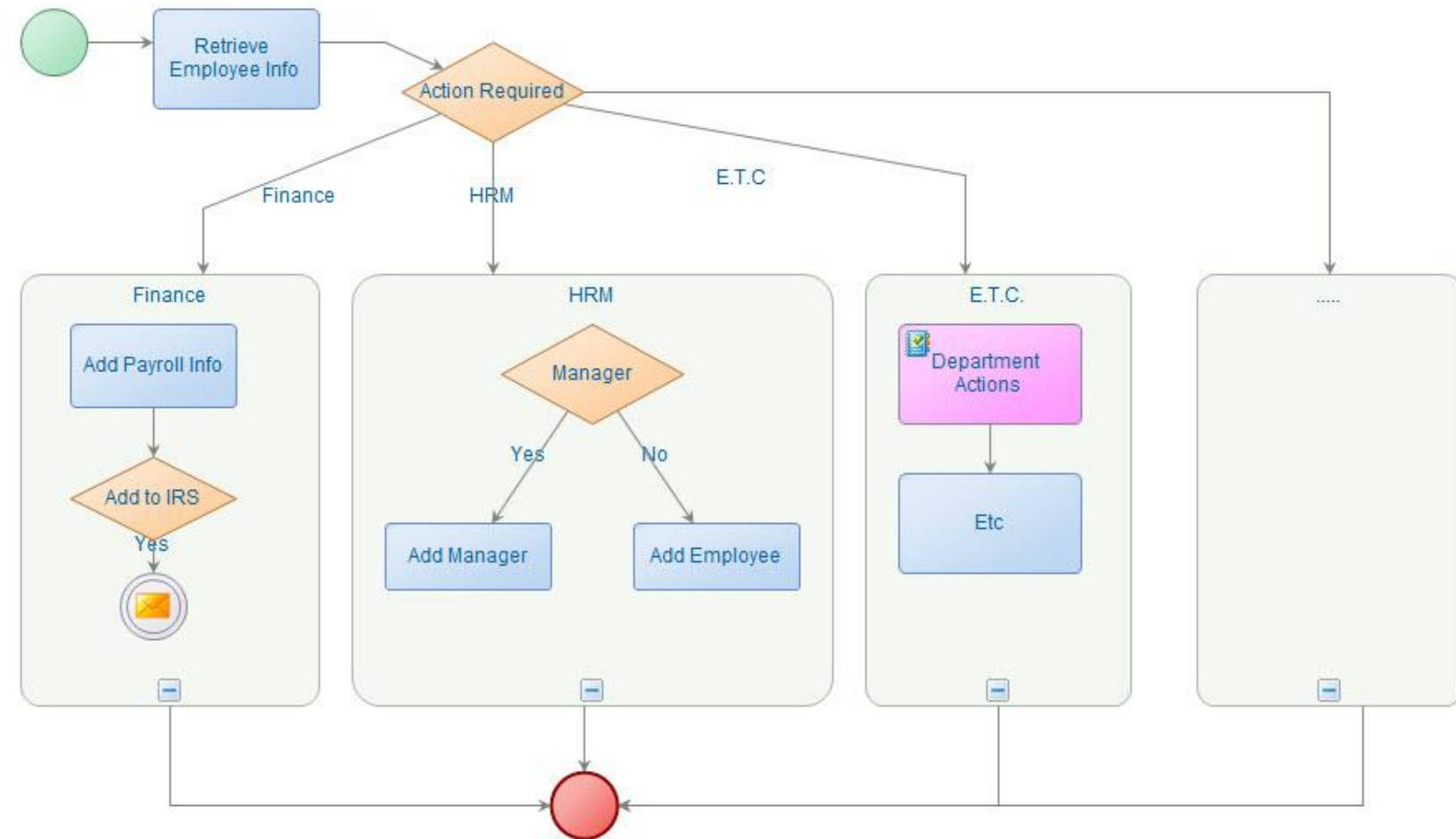


Quote becomes order

- ◆ Which activities have to be executed?
 - ◆ Order Material
 - ◆ Plan Resources
 - ◆ Send Invoice?
 - ◆ Celebrate?
- ◆ Which systems are involved?
- ◆ Which people are involved?

- ◆ Due to change in a Business Object (employee, order, customer, product etc.) a lot of different processes have to be executed.

◆ New Employee Hire BPM



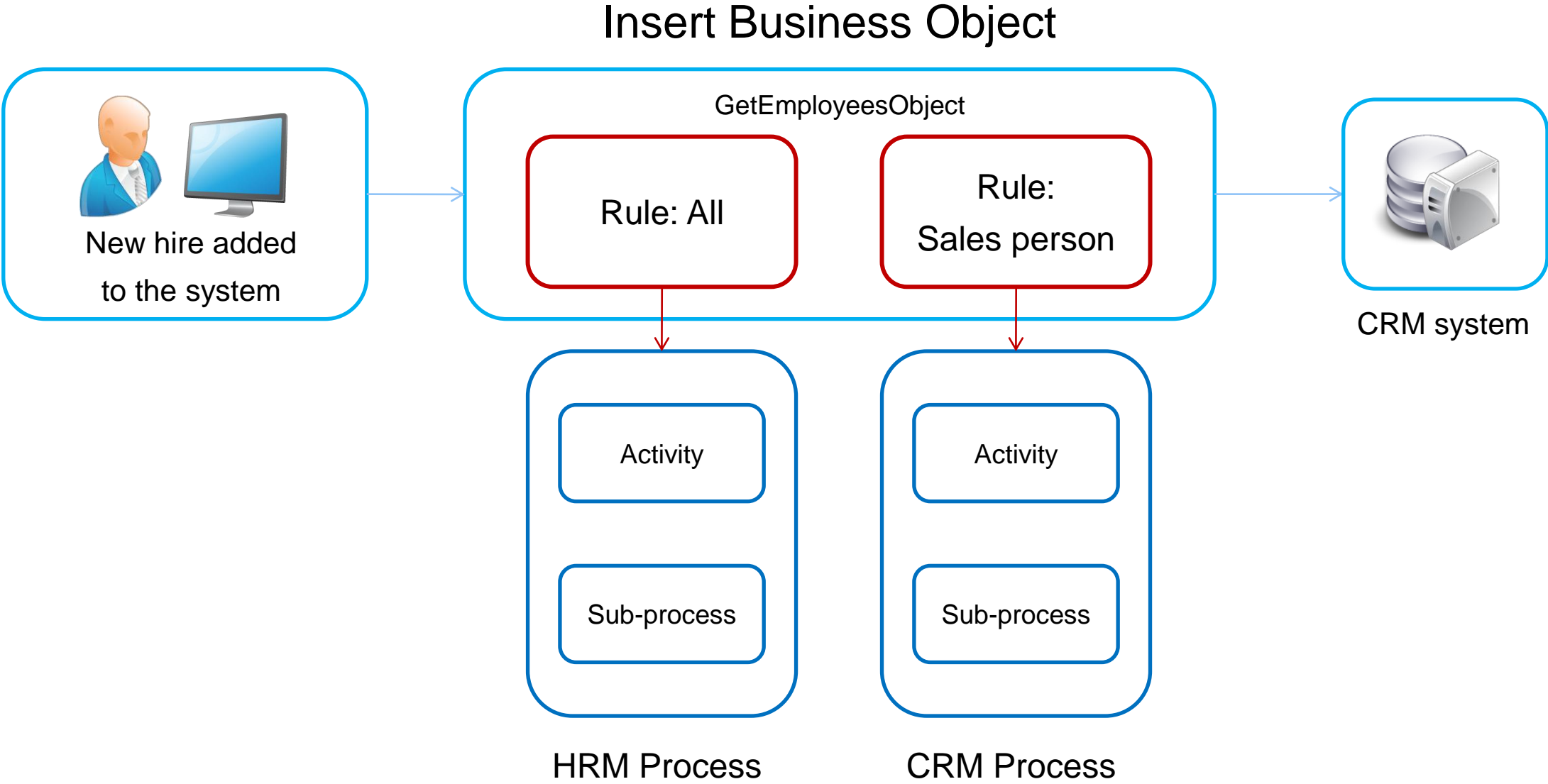
◆ Drawback

- ◆ Process becomes huge
- ◆ If one activity changes the whole process has to be rebuilt
- ◆ Processes are made over boundaries of domains (departments and companies)
- ◆ Process has to be changed if (sub) processes are added

◆ Solution

- ◆ Use Sub Processes (decouple Processes)
- ◆ Use Business Rules
 - ◆ Trigger Processes
 - ◆ Trigger Services
 - ◆

- ◆ Event Driven Architecture
- ◆ Business Rules
- ◆ Business Rules in Cordys



- ◆ Rules Support an Event Driven Architecture
 - ◆ More flexibility
 - ◆ Separation of concerns
 - ◆ Better maintainable
- ◆ Decision Logic is separated from BPMs
- ◆ Decision Logic (re)used in BPMs
- ◆ Business Logic is separated from Application Logic

- ◆ Event Driven Architecture
- ◆ Business Rules
- ◆ Business Rules in Cordys

◆ Implementation

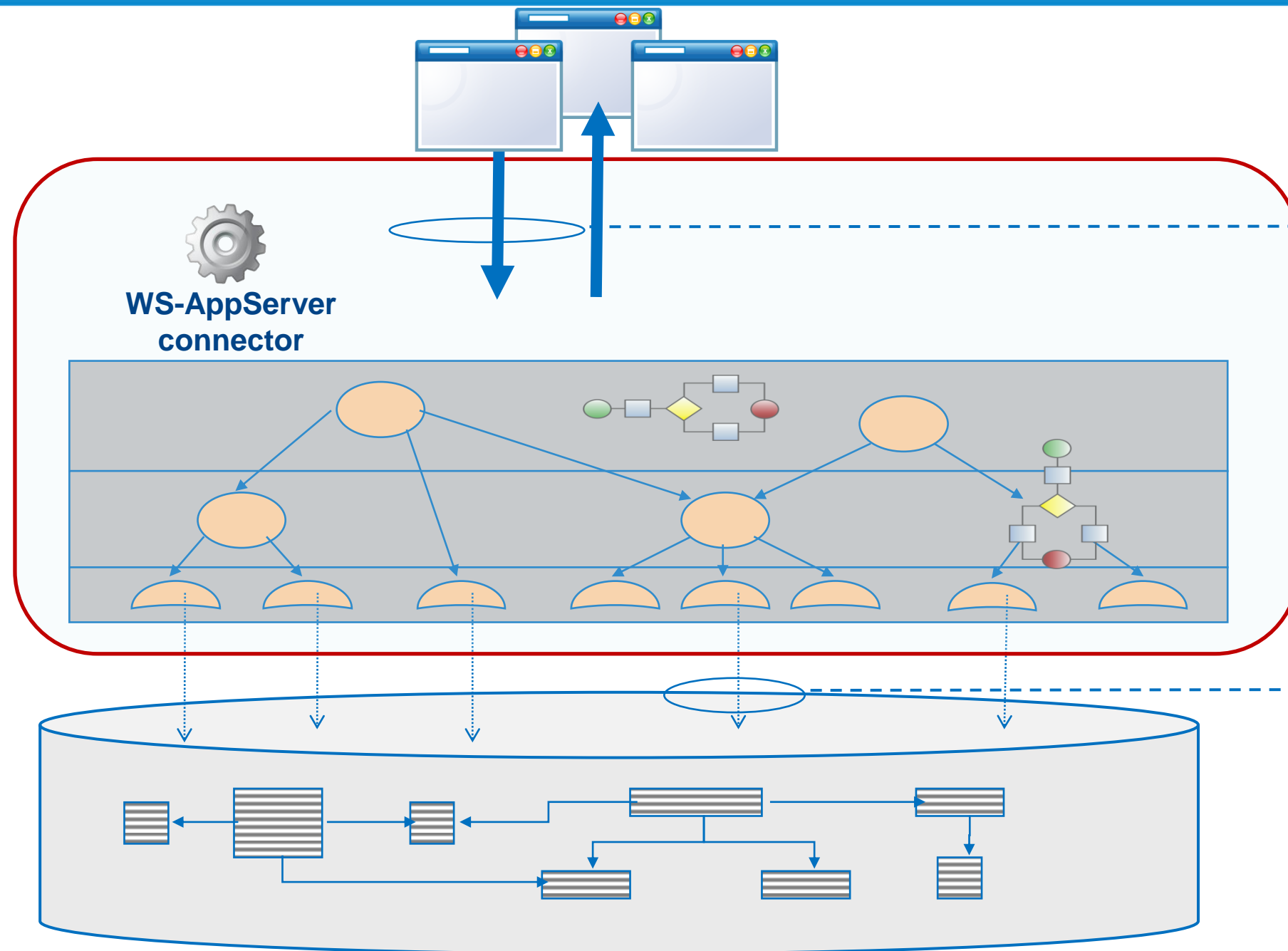
- ◆ Decision table
- ◆ Rule

◆ Usages

- ◆ Defined on WS-AppServer Package XML Schema,
 - ◆ to trigger actions on Insert, Update and/or Delete Transactions
- ◆ Defined on XML Schema
 - ◆ For decision table activity in a business process model

Rules on WS-AppServer Objects

CORDYS



- ◆ Presentation Logic
- ◆ Rules Logic (business)
- ◆ WS-AppServer Logic
 - ◆ Constraint
 - ◆ Access
 - ◆ Business
- ◆ Rules Logic (constraint)
- ◆ Database Logic

Decision Table

- ◆ Business View on Business Rules
- ◆ Group of Conditions defined on a Business Entity

Sales Manager - Decision Table*				
Conditions	Attributes	Office US Main	Office US East	Office UK
	Country	is USA	is USA	is UK
	Region	"CA", "TX", "AZ",	"VA", "WA", "NY",	
	ReportsTo	is ""	is ""	is ""
Actions	Set ReportsTo	Andrew Fuller	Steve Buchanan	Anne Dodsworth
	Inform Manager			Inform Manager
	Trigger Business Process		Create System Accounts	

Rule Definition Elements

◆ Condition

- ◆ Complex, technical conditions using functional libraries

◆ Actions

- ◆ Trigger Business Process
- ◆ Assign
- ◆ Trigger Web Services
- ◆

◆ Options

- ◆ Override
- ◆ Mutex
- ◆

The screenshot displays the CORDYS Rule Definition tool interface for a rule named "NewSalesEmployee - Rule*". The interface is divided into several panes:

- Functions:** A tree view on the left showing available functions categorized into Math Operators, Boolean Operators, Math Functions, String Functions, and Date-Time Functions.
- Rule Definition:** The main workspace showing the rule logic. It contains an "if" condition: `string-length(nor:Employees/nor:LastName) < 6`. If true, it triggers the action "Trigger Business Process-Send News Event".
- InputSchema:** A tree view showing the input schema for the "Employees" object, including fields like `nor:EmployeeID`, `nor:LastName`, `nor:FirstName`, `nor:Title`, `nor:TitleOfCourtesy`, `nor:BirthDate`, `nor:HireDate`, `nor:Address`, `nor:City`, `nor:Region`, and `nor:PostalCode`.
- Action-Trigger Business Process:** A pane for configuring the triggered business process. It shows:
 - Action Name:** Send News Event
 - Process Name:** NewsEvent
 - Message:** XML message structure for the event:

```
<NewsInput xmlns="http://schemas.company1.com/myapplication/salesprocesses">
  <News xmlns="http://schemas.company1.com/myapplication/salesprocesses">
    <Message xmlns="http://schemas.company1.com/myapplication/salesprocesses">
      <path xmlns="http://rules.commonontypes">concat("New sales employee ", nor:Employees/nor:FirstName, " ",
nor:Employees/nor:LastName)</path>
    </Message>
  </News>
</NewsInput>
```

◆ Rule Group

- ◆ Group rules (for example by context)
- ◆ Prioritize rules by rule group

◆ Rule Type

- ◆ Constraint – before the backend commit
- ◆ Business - after the backend commit
- ◆ Passive

◆ Templates (Action / Condition)

- ◆ Reusable for rules and decision tables

Thank You

Questions?