IBM T.J. Watson Research Center

# Specifying and Enforcing High-Level Semantic Obligation Policies

**Zhen Liu,**
**Anand Ranganathan,**
**Anton Riabov**

© 2006 IBM Corporation

## The first part of ECA is *E*

- **Obligation Policies often take the form of ECA (Event-Condition-Action) rules**
- **In this work, we focus on the *Event* part of ECA**
- **Large scale pervasive and distributed computing systems often produce event streams containing large volumes of low-level events**
  - Logging and audit messages
  - Events from sensors
  - Unstructured data (containing text and multimedia) from video cameras, microphones
- **Policies are often at a higher-level**
- **Hence often difficult to specify policies**

## How can I specify the *E* in an ECA rule more easily?

- **Specify obligation policies based on high-level events**
  - derived after performing appropriate processing on raw, low-level events
- **Semantic obligation policy specification language called *Eagle***
  - based on high-level events
  - represented as RDF graph patterns
  - based on terms defined in domain OWL ontologies
- **Policy Enforcement through a Compiler**
  - Uses an AI Planning approach to build a workflow for producing a stream of the required high-level events as specified in a policy.
  - Workflow consists of a number of event sources and event processing components, which are described semantically

## Key advantages of Eagle Policies

- **Ease of Policy Specification and Maintenance**
  - Policies are independent of low-level events or of specific processing components
- **Highly expressive**
  - Use of Description Logic Reasoning for matching an event (or a stream of events) to a policy
- **Adaptive**
  - Use of planning allows adaptation to current conditions of a system

## Example from a call center domain

- **Customer Service Department of a company receives and makes VoIP calls**
- **Managers create policies that specify actions for certain kinds of events**
  - If there are many dropped calls
  - If the average call waiting time is high
  - If an employee is discourteous
- **Raw events in system**
  - VoIP based call traffic
  - Session management events (call transfer, call termination, etc.)

## Example Policy

- **If the courtesy level of an employee in a VoIP service call is**
  - less than 0.4, then an instant message alert should be sent to the manager;
  - if the courtesy level is less than 0.2, then the employee should not be allowed to answer any further customer service calls.

## Example Policy

```
Policy EmployeeCourtesyPolicy
    OnEvent
        Containing ?EmpName, ?MgrEmail, ?CurtLevel
        Where
            ?Employee      a :Employee;
                          :tookCall ?EmpVoIP_Call;
                          :hasName ?EmpName;
                          :hasMgr ?Manager;
                          :employedBy EGS.
            ?EmpVoIP_Call a :ServiceVoIP_Call;
                          :withCurtLevel ?CurtLevel.
            ?Manager    a :Manager;
                          :hasEmail ?MgrEmail.
    PerformAction
        IMServ.sendAlert(?MgrEmail, ?EmpName)
                If (?CurtLevel < 0.4)
                AND (IMServ.isOnline(?MgrEmail))
    PerformAction
        CallRouteServ.suspendEmp(?EmpName)
                If (?CurtLevel < 0.2)
```

Name of Policy

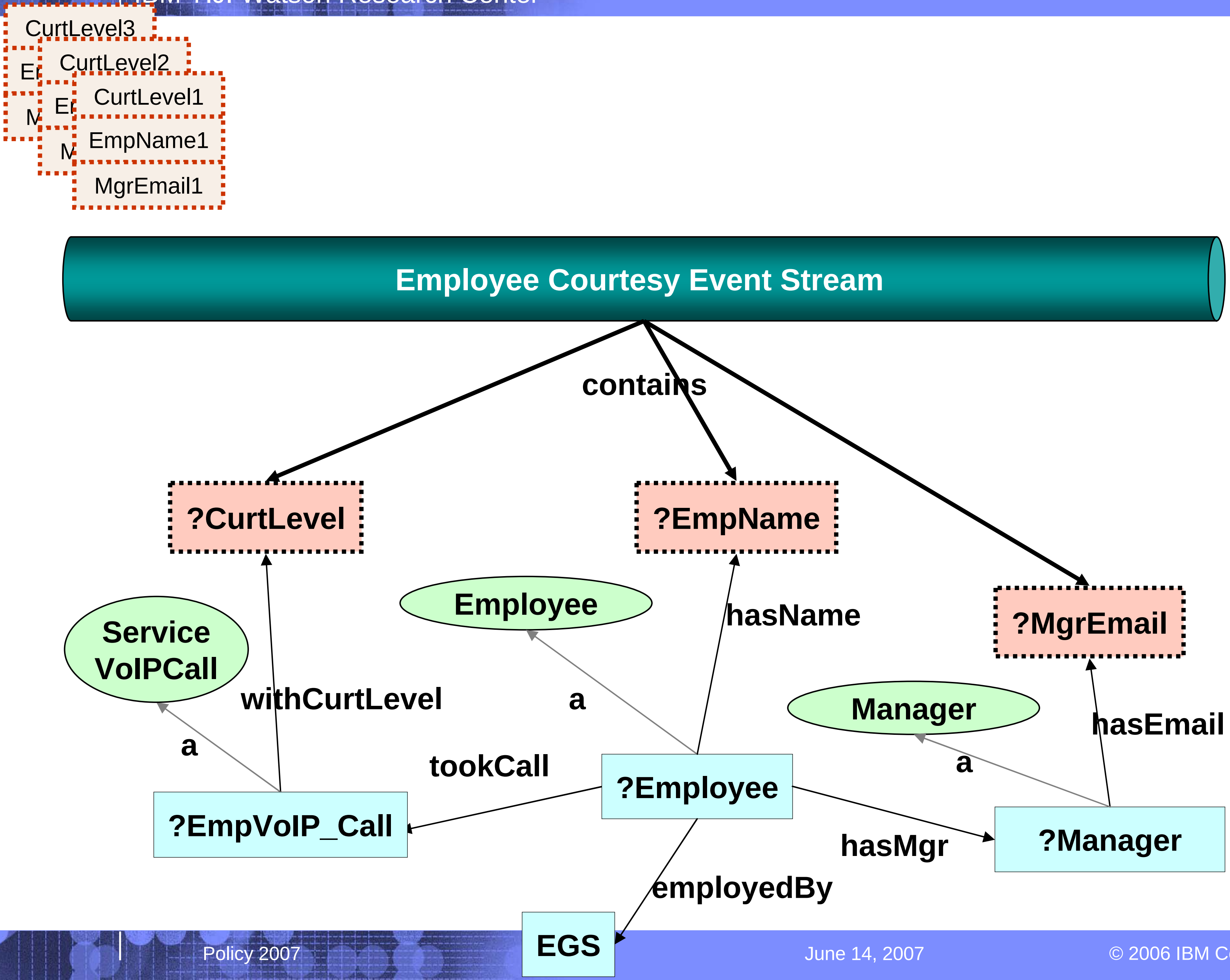
Data Elements contained in Event

Semantic Description of Event (expressed as an RDF graph pattern)

Actions to be Performed under given conditions

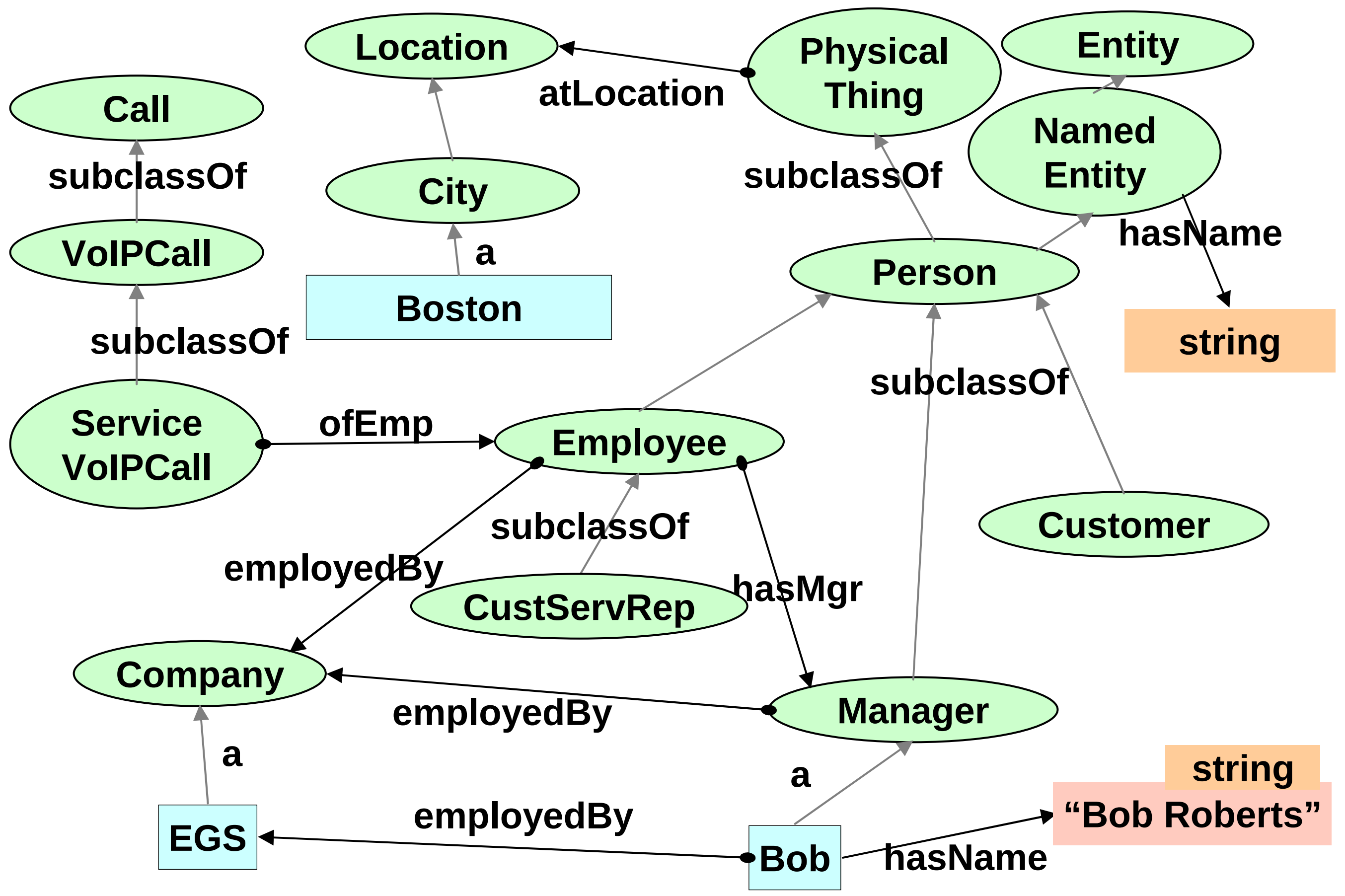
## Event Specification in Eagle

- **Streaming model of events**
- **An Eagle Policy describes the data elements that must be contained in the event**
  - Described as a set of variables that are bound to values by a given event on the stream
- **Also specifies the semantics (or the *meaning*) of the elements in an event**
  - Described as a set of constraints on the variables
  - Expressed as an RDF graph pattern


CurtLevel3
CurtLevel2
CurtLevel1
EmpName1
MgrEmail1
Employee Courtesy Event Stream
contains
?CurtLevel
?EmpName
?MgrEmail
Service VoIPCall
Employee
Manager
withCurtLevel
hasName
hasEmail
a
a
a
tookCall
?Employee
?EmpVoIP_Call
?Manager
hasMgr
employedBy
EGS

## Terms in Event Specification defined in OWL Ontologies

- **OWL Ontologies Describe Entities in Domain**
  – Classes, Properties, Individuals



## The Eagle Policy Language

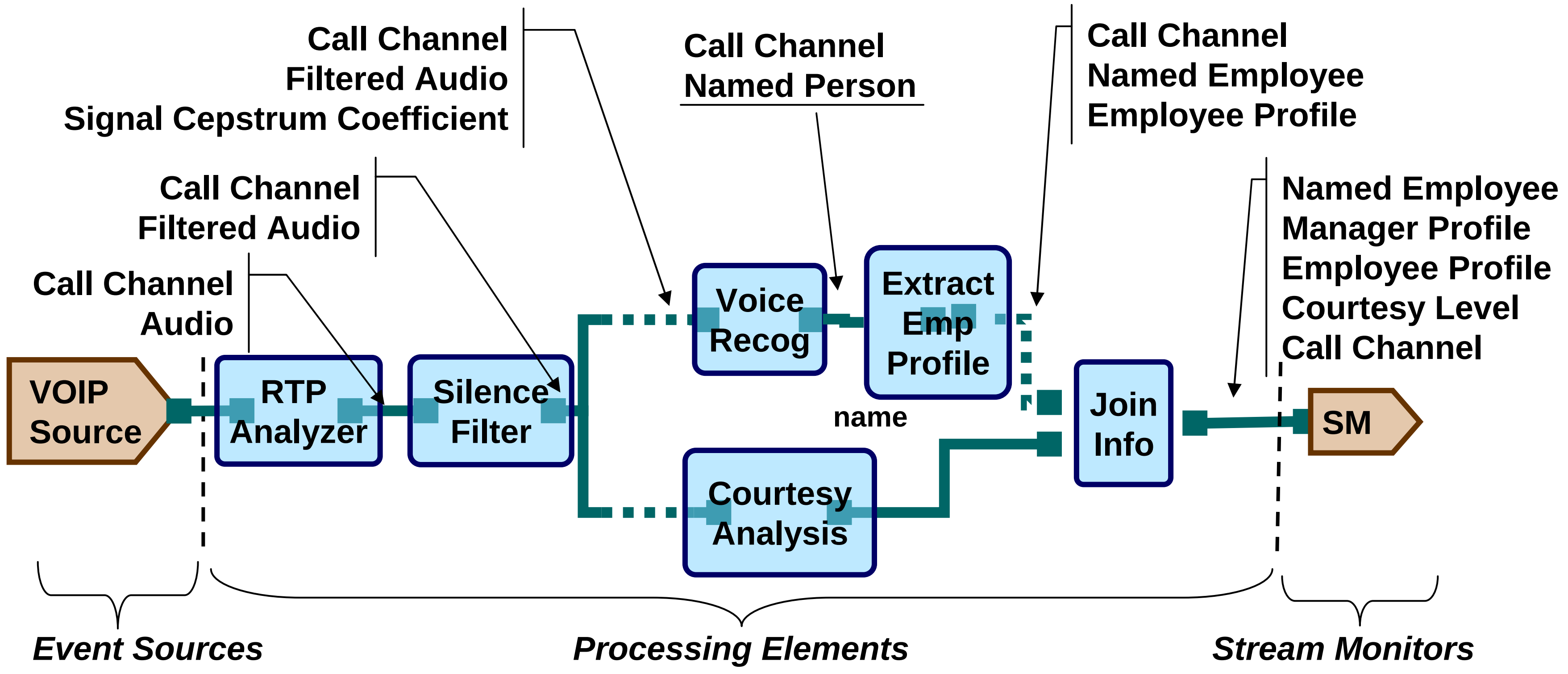
```
Policy <PolicyName>
  OnEvent
          Containing <VariableSet>
          Where <Graph Pattern>
  PerformAction <ActionDefnition_1>
          If <ConditionDefinition_1>
…
  PerformAction <ActionDefnition_n>
          If <ConditionDefinition_n>
```

## Policy Enforcement consists of 2 steps

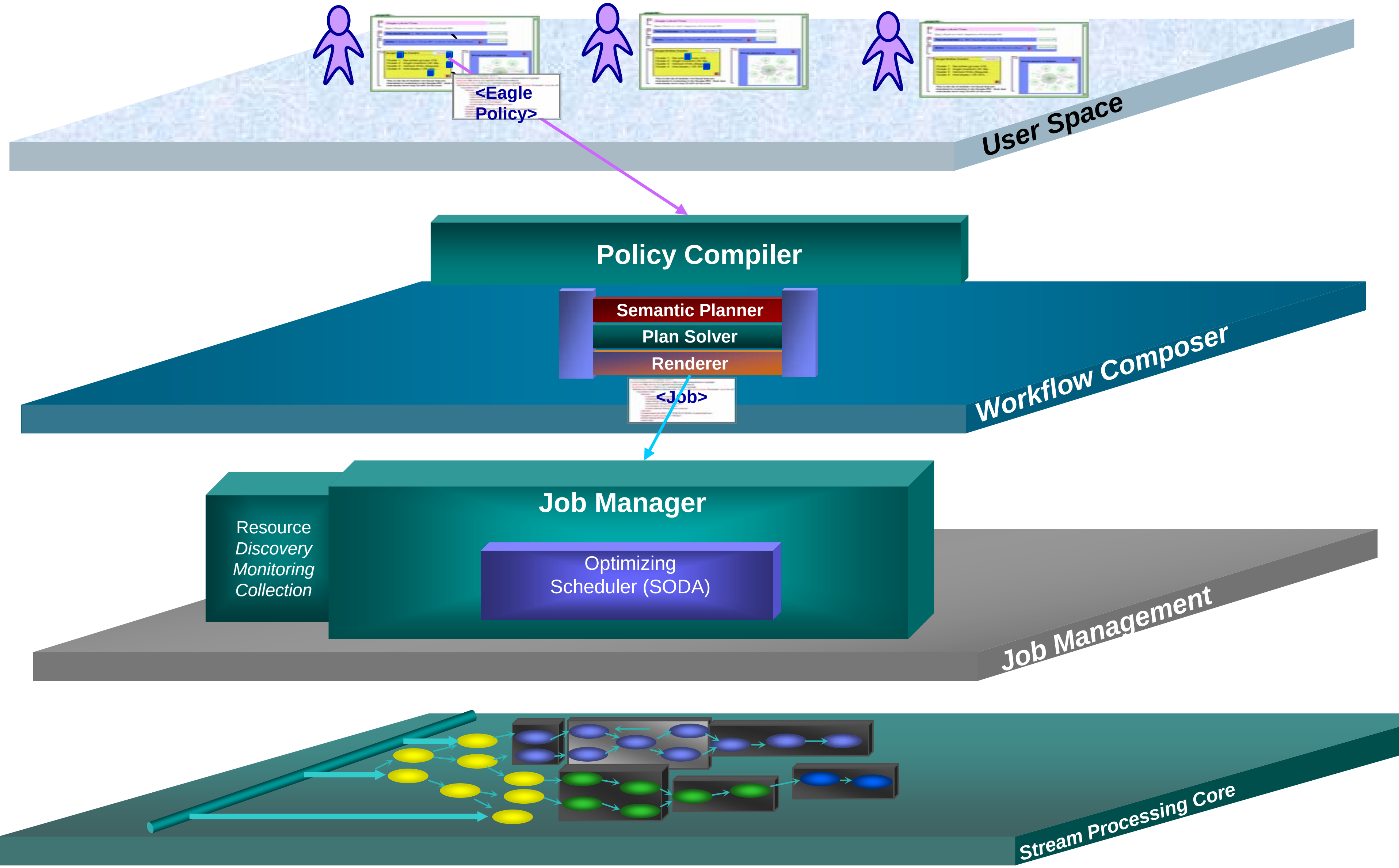
- **Policy Compilation**
  - Create a workflow that can generate a stream of events that match the pattern defined in the policy
  - Matching is based on Description Logic (DL) reasoning
- **Stream Monitoring**
  - Monitor events on the matched stream and perform actions if certain conditions are true.

## Policy Compiler

- **Constructs Workflow (DAG) of sources and processing elements to produce high level event**

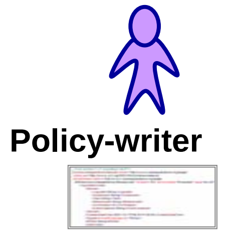


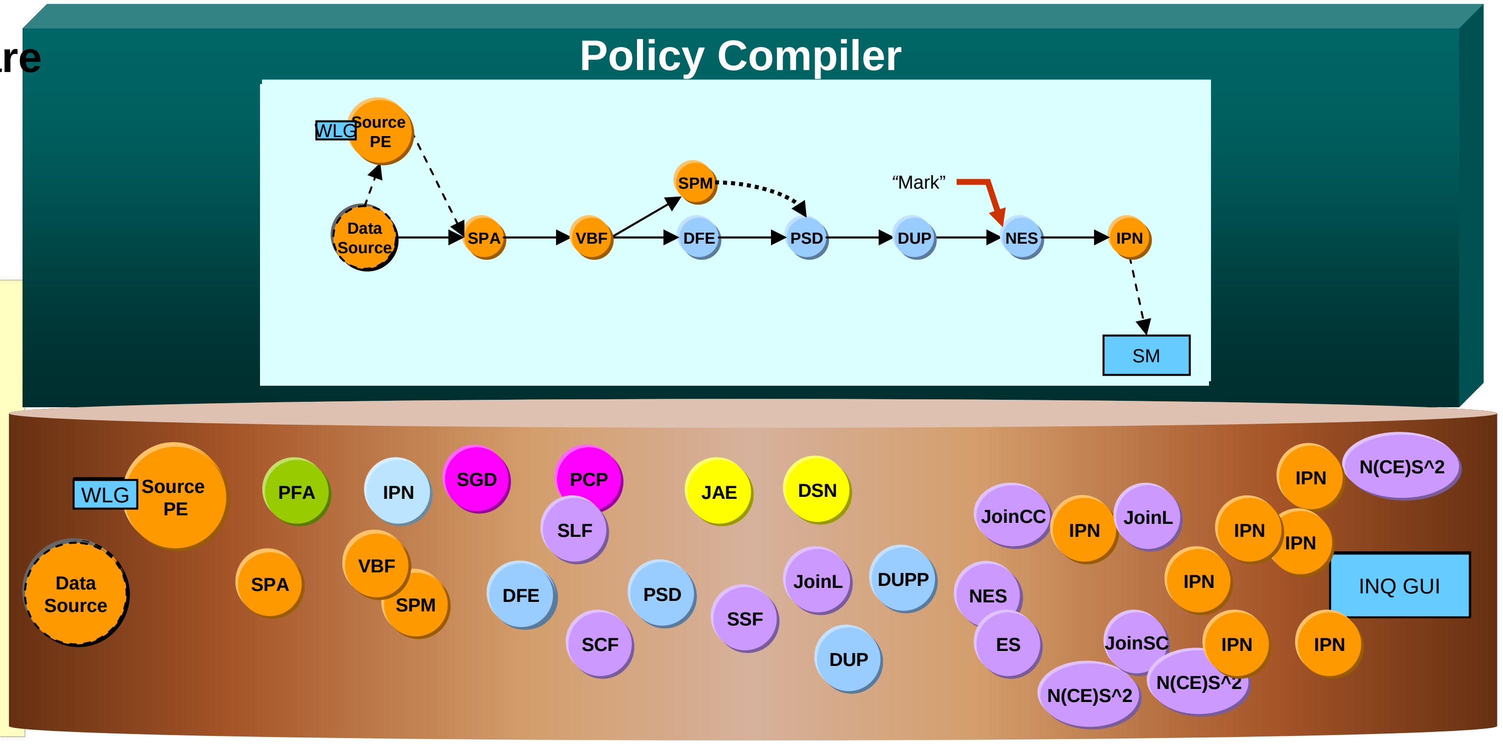
# System S Stream Processing System



## Goal of Policy Compiler: Automatic Generation of Workflows

- **Ease of specification**
  - Policy-writer doesn't have to be aware of actual event sources and software components
- **Shorter Time to Enforcement**
- **Reusable Software Components**





| | |
|---|---|
| **SPA** | → Streaming Protocol Annotator |
| **VBF** | → Value-Based Filter |
| **SCF** | → Stream Courtesy Finder |
| **JoinCC** | → Join Cust & Emp to Courtesy Level |
| **IPN** | → Identified Person Notification |
| **SGD** | → Speech Gait Detection |
| **PCP** | → Progressive Conversation Pairing |
| **JAE** | → Join Asynch Events |
| **DSN** | → Denoise via Social Network |
| **DUPP** | → Dig Up Person Pair Info |
| **N(CE)^S** | → Employee Selector |
| **SPM** | → Stream Progression Monitor |
| **DFE** | → Distributed Front-End |
| **PSD** | → Progressive Speaker Detection |
| **IPN** | → Identified Person Notification |

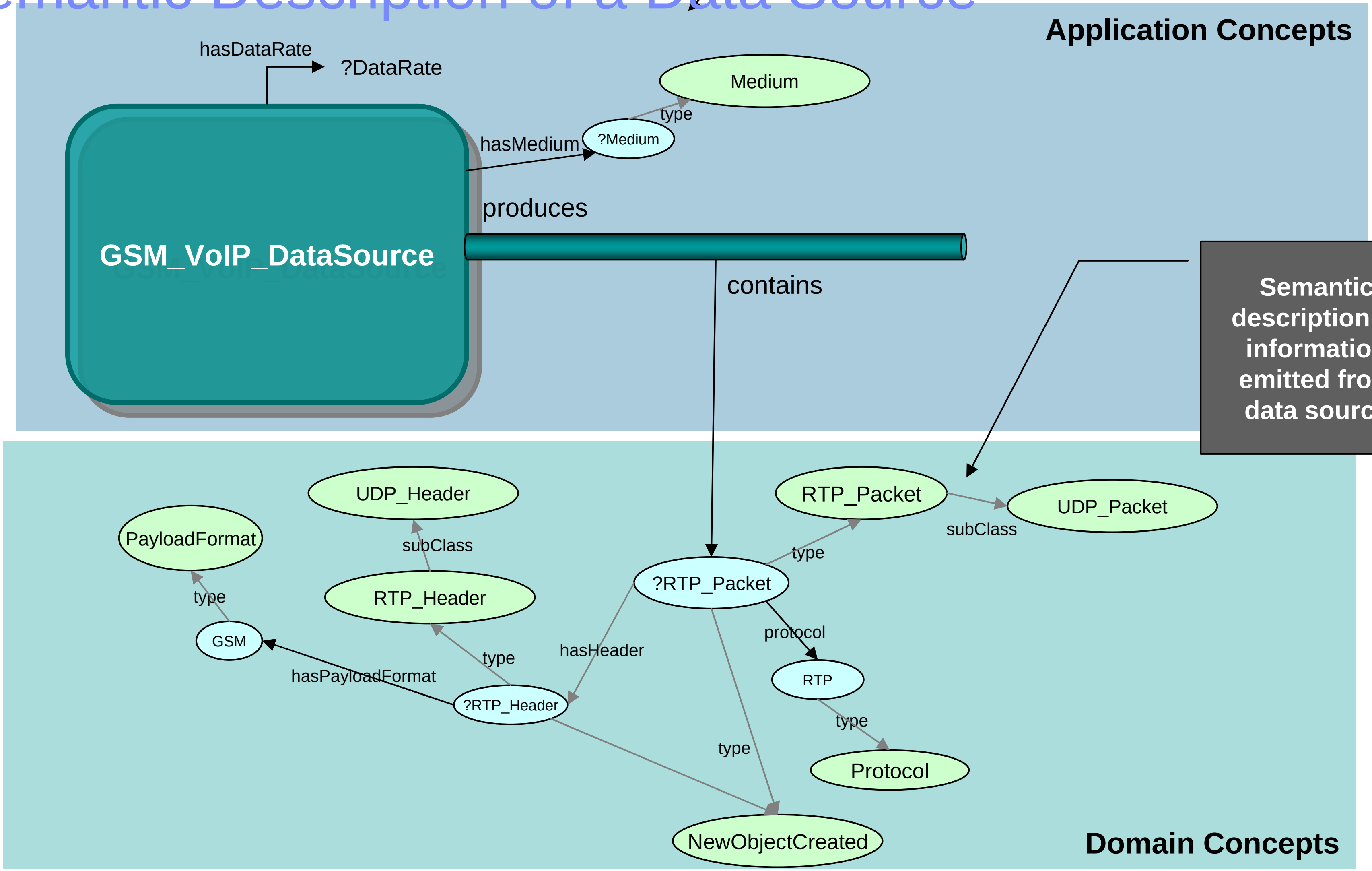
## What is required for policy specification?

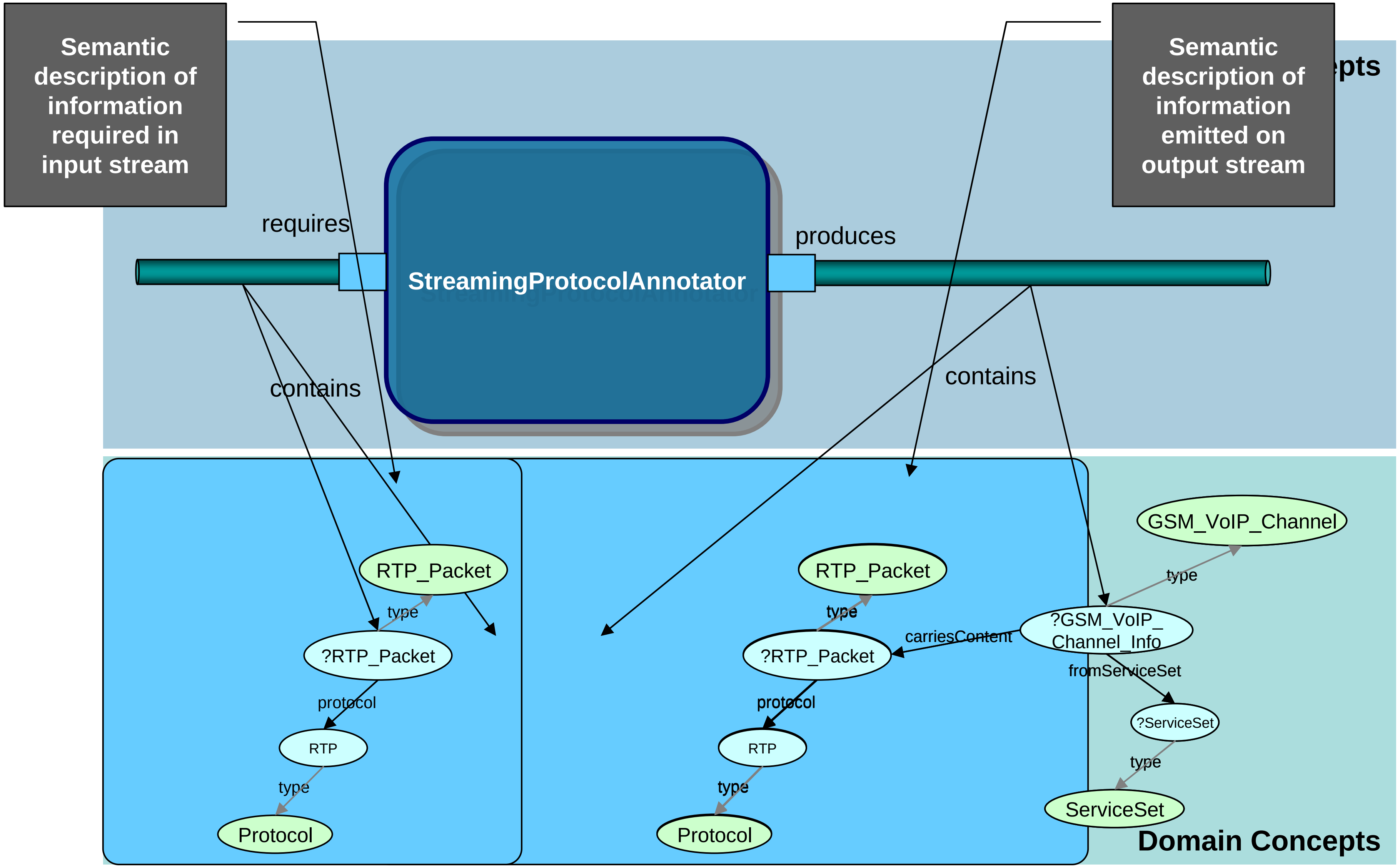
- **Expressive semantic descriptions of**
  - Outputs of different streaming event sources
  - Input requirements and output capabilities of stream processing components
- **Method for determining if a stream produced from an event source can match the input requirements of a stream processing component**
- **Method for determining if any stream matches the event specification in the policy**

# Semantic Description of a Data Source

Information about the Data Source

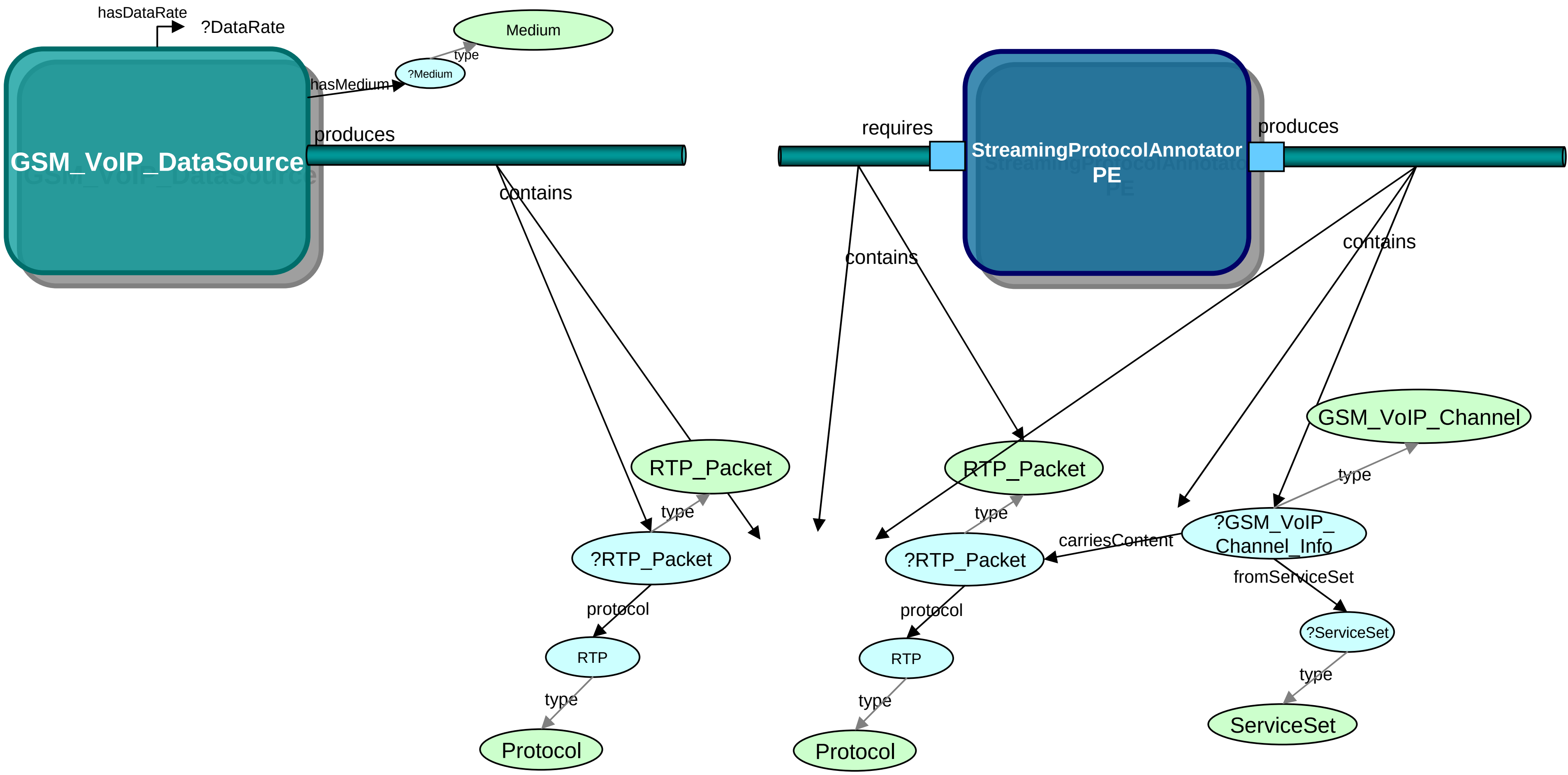
Semantic description of information emitted from data source





Semantic description of information required in input stream
Semantic description of information emitted on output stream
pts
requires
produces
StreamingProtocolAnnotator
contains
contains
RTP_Packet
type
?RTP_Packet
protocol
RTP
type
Protocol
RTP_Packet
type
?RTP_Packet
carriesContent
?GSM_VoIP_Channel_Info
GSM_VoIP_Channel
type
fromServiceSet
?ServiceSet
type
ServiceSet
protocol
RTP
type
Protocol
Domain Concepts

## Using Semantic Descriptions to Match and Assemble Plan Graph Components



## Policy Compilation through use of AI Planning

**We have developed a semantic planner for composing a workflow that can…**

- Make use of rich, expressive domain knowledge in ontologies
- Perform Description Logic (DL) reasoning to allow flexible matching
- Be scalable even in the presence of possibly time-consuming DL reasoning
- Be capable of planning on streams

## 2-phase semantic planning algorithm

- **First phase, which occurs offline**
  - Takes OWL files describing components and data sources.
  - Uses a DL reasoner to obtain all inferred facts about the components
  - E.g. Asserted : Data Source produces Traffic Video Segments
    Inferred : Data Source produces Video Segment
- **Second phase, which occurs when the system is given a goal, for which plans have to be found.**
  - Searches for plan using more complete description of components
  - Uses a forward chaining, branch and bound search to produce optimal plans
  - No DL reasoning required in this phase
  - Checks data security and privacy constraints

## Stream Monitoring

- **Stream monitoring components are instantiated to monitor high-level events produced by workflow**
- **Performs actions if given conditions in Eagle Policy are satisfied**

© 2006 IBM Corporation

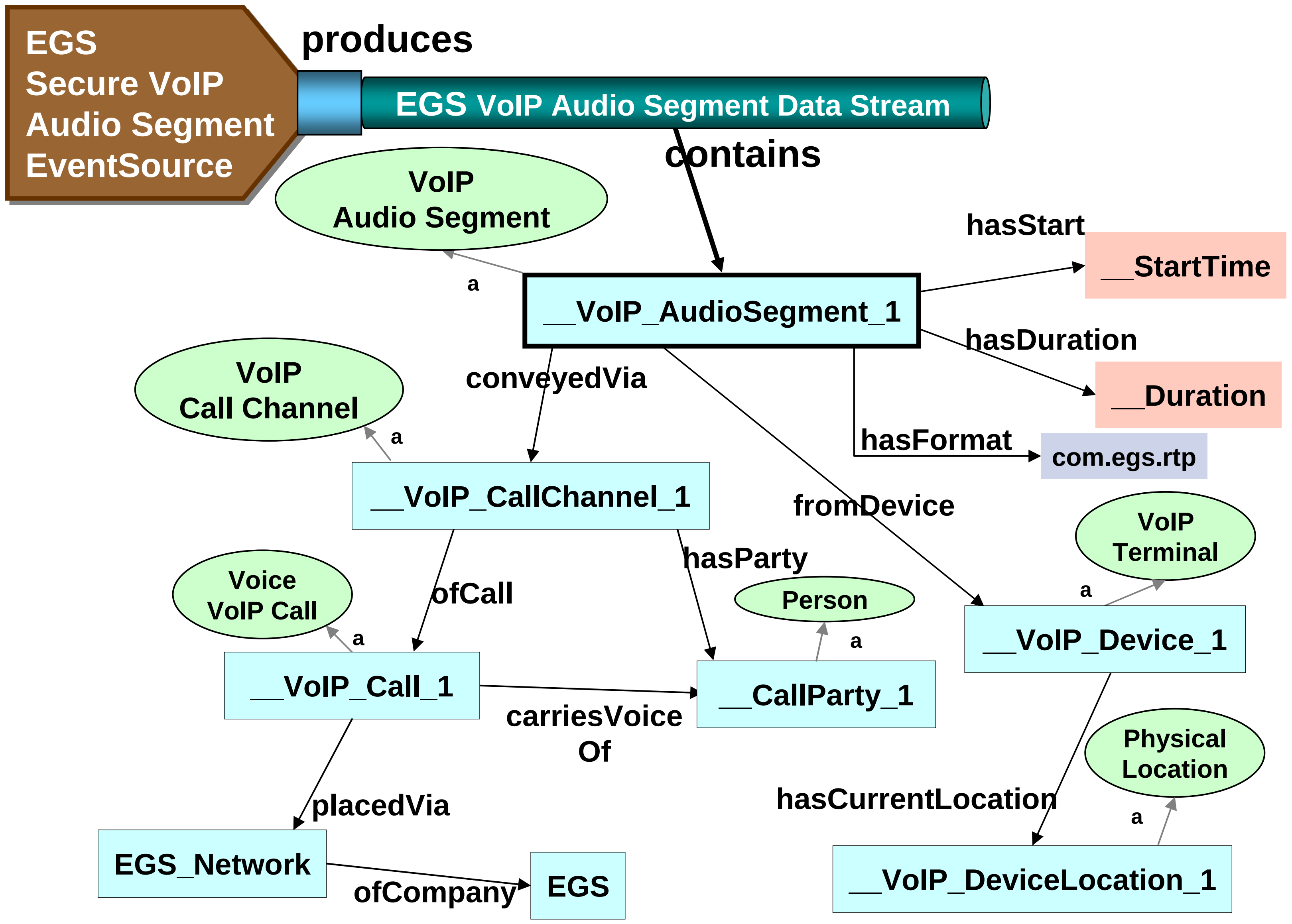
## Conclusion

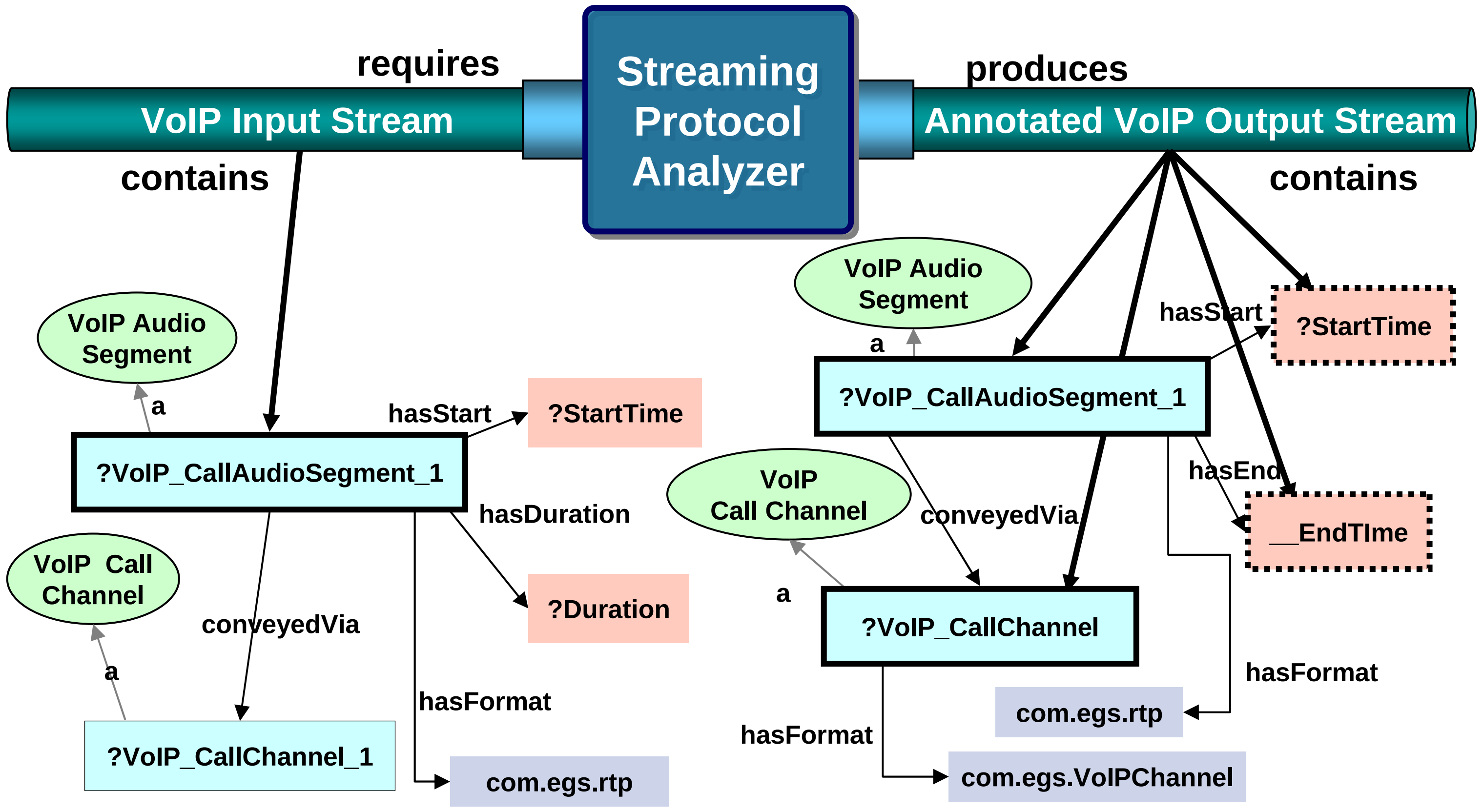
- **Method for specifying high-level obligation policies based on high-level events**
  - Using RDF Graph Patterns to describe the event
  - Using terms defined in ontologies to define the semantics of the event more precisely
- **Method for matching a stream of events to an event pattern**
  - Uses DL reasoning
- **Method of compiling a policy by constructing a workflow of processing elements and event sources**
  - Workflow produces event stream that satisfies high-level event pattern

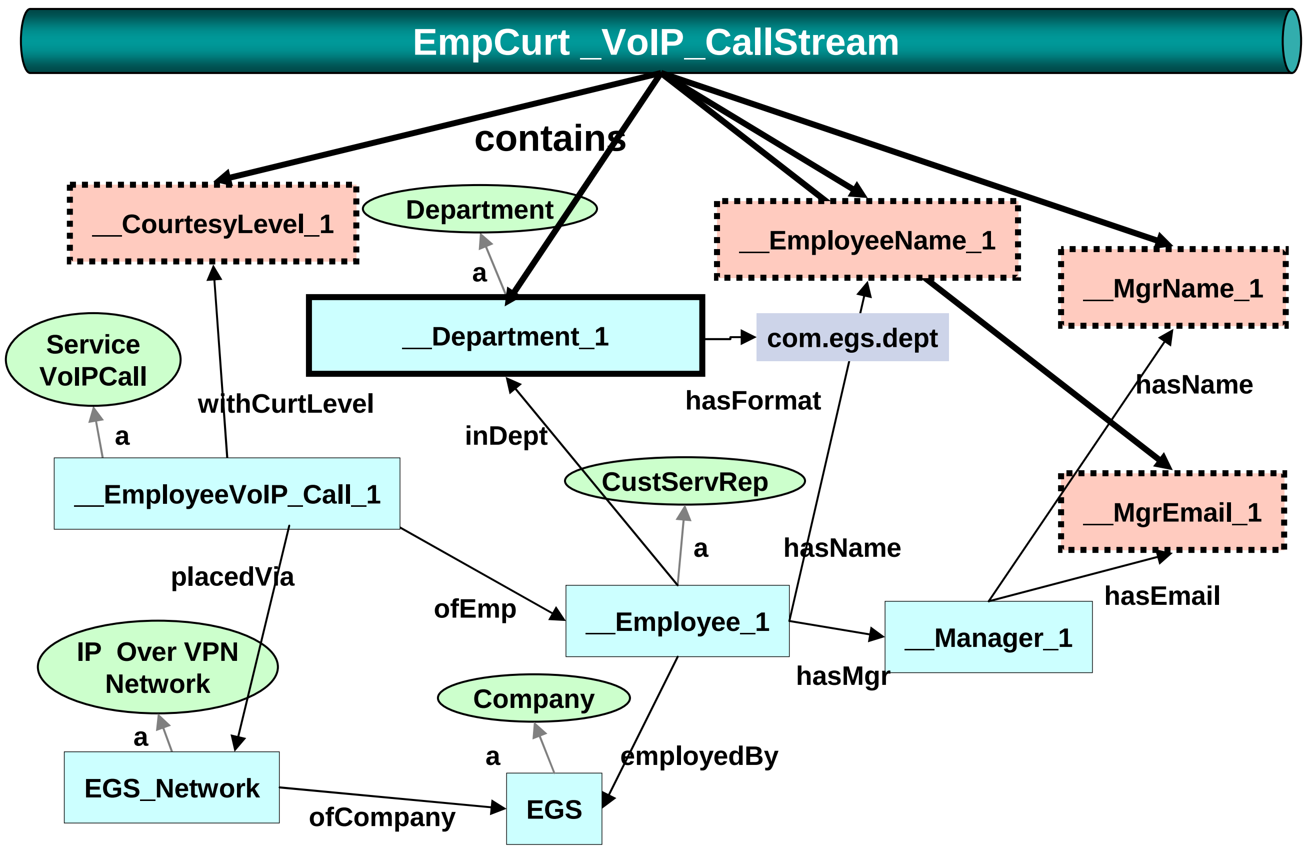
## Grazie e Boun Giorno

## Delle Domande??

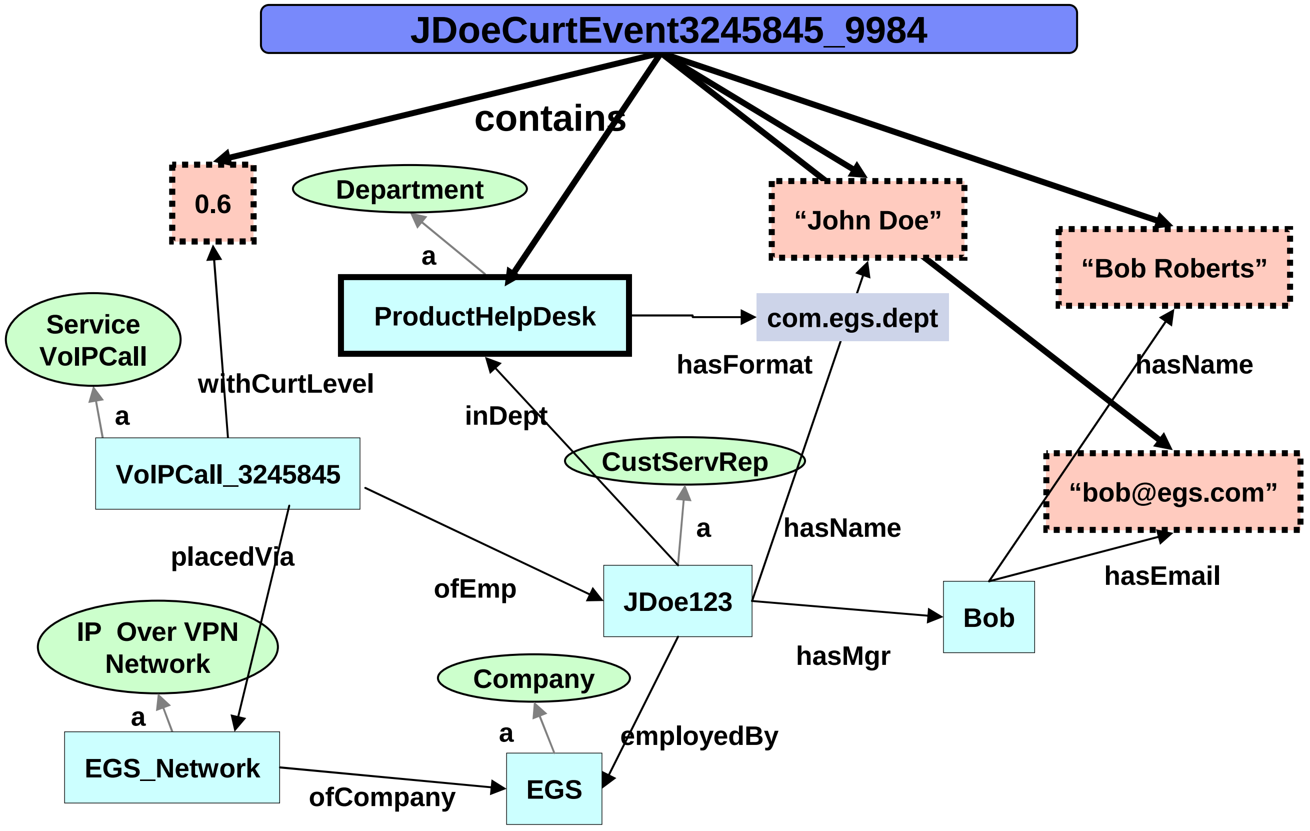
## Backup


EGS Secure VoIP Audio Segment EventSource
produces
EGS VoIP Audio Segment Data Stream
contains
VoIP Audio Segment
a
__VoIP_AudioSegment_1
hasStart
__StartTime
hasDuration
__Duration
hasFormat
com.egs.rtp
conveyedVia
VoIP Call Channel
a
__VoIP_CallChannel_1
fromDevice
VoIP Terminal
a
__VoIP_Device_1
hasParty
Person
a
__CallParty_1
ofCall
Voice VoIP Call
a
__VoIP_Call_1
carriesVoice Of
placedVia
EGS_Network
ofCompany
EGS
hasCurrentLocation
Physical Location
a
__VoIP_DeviceLocation_1

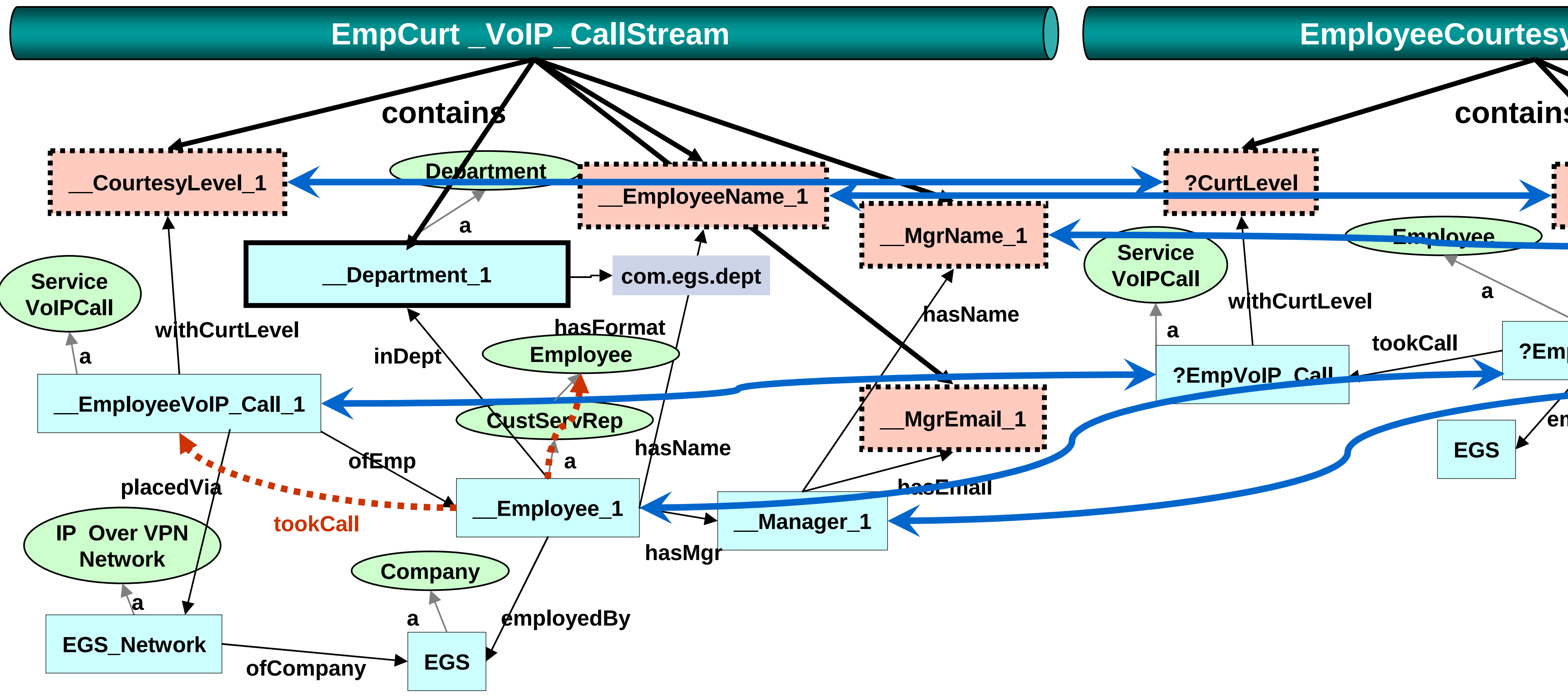

requires
VoIP Input Stream
Streaming Protocol Analyzer
produces
Annotated VoIP Output Stream
contains
contains
VoIP Audio Segment
a
?VoIP_CallAudioSegment_1
hasStart
?StartTime
hasDuration
?Duration
VoIP Call Channel
a
conveyedVia
?VoIP_CallChannel_1
hasFormat
com.egs.rtp
VoIP Audio Segment
a
?VoIP_CallAudioSegment_1
hasStart
?StartTime
hasEnd
__EndTIme
VoIP Call Channel
a
conveyedVia
?VoIP_CallChannel
hasFormat
com.egs.rtp
hasFormat
com.egs.VoIPChannel



© 2006 IBM Corporation



© 2006 IBM Corporation


EmpCurt _VoIP_CallStream
EmployeeCourtesy
contains
contains
__CourtesyLevel_1
Department
a
__EmployeeName_1
__MgrName_1
?CurtLevel
Employee
Service VoIPCall
__Department_1
com.egs.dept
Service VoIPCall
withCurtLevel
a
hasName
withCurtLevel
a
hasFormat
inDept
Employee
tookCall
?Emp
a
?EmpVoIP_Call
__EmployeeVoIP_Call_1
CustServRep
__MgrEmail_1
em
ofEmp
a
hasName
EGS
placedVia
hasEmail
tookCall
__Employee_1
__Manager_1
IP Over VPN Network
hasMgr
Company
a
a
employedBy
EGS_Network
EGS
ofCompany