Chapter 5

Generating Linked Data in Real-time from Sensor Data Streams

NIKOLAOS KONSTANTINOU

DIMITRIOS-EMMANUEL SPANOS

Outline

Introduction

Fusion

The Data layer

Rule-based Reasoning

Complete Example

Problem Framework

Rapid evolution in ubiquitous technologies

Pervasive computing is part of everyday experience

- User input
- Information sensed by the environment

Parallel decrease of the price of sensors

IoT and M2M

Streamed Data (1)

Need for real-time, large-scale stream processing application deployments

Data Stream Management Systems

- Managing dynamic knowledge
- Emerged from the database community
- Similar concern among the Semantic Web community

Streamed Data (2)

Numerous challenges

- Large scale
- Geographic dispersion
- Data volume
- Multiple distributed heterogeneous components
  - Sensor, sensor processing and signal processing (including a/v) components
- Vendor diversity
- Need for automation

Data Stream Management Systems

Fill the gap left by traditional DBMS's

- DBMS's are not geared towards dealing with continuous, real-time sequences of data

Novel rationale

- Not based on persistent storage of all available data and user-invoked queries

Another approach

- On-the-fly stream manipulation
- Permanent monitoring queries

Context-awareness, IoT and Linked Data (1)

Context

> Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves

Context-aware systems

- Knowledge of the environment in which they are acting
- Extract and use information from the environment in order to adjust their functionality according to the incoming information
- Behavior according to the existing conditions
- Tightly connected to the IoT vision

Context-awareness, IoT and Linked Data (2)

The IoT vision

- Aims at connecting (large numbers of) sensors deployed around the world
  - Focus shifts to automated configuration of filtering, fusion and reasoning mechanisms that can be applied to the collected sensor data streams

Ubiquitous, pervasive, context-aware

- The ability of a system to “read” its environment and take advantage of this information in its behaviour

Context-awareness, IoT and Linked Data (3)

Challenge

- Heterogeneity in systems
  - Capture, store, process, and represent information
- Information is generated in large volumes and at a high velocity

Common representation format

- Making systems interoperable
- Allows information to be shared, communicated and processed

Context-awareness, IoT and Linked Data (4)

Linked Data

- An efficient approach towards filling in this gap
- A common, reliable and flexible framework for information management
- Information homogeneity
- Facilitates information integration

Outline

Introduction: Problem Framework

Fusion

The Data layer

Rule-based Reasoning

Complete Example

Information Fusion (1)

Fusion

The study of techniques that combine and merge information and data residing at disparate sources, in order to achieve improved accuracies and more specific inferences than could be achieved by the use of a single data source alone

- Leverages information meaning
- Partial loss of initial data may occur
- Several fusion levels

Information Fusion (2)

Algorithm

- Online (distributed)
  - Each node can take decisions based only on its perception of the world
  - Each algorithm execution is based on the knowledge of only a local node or a cluster of nodes
- Offline (centralized)
  - There is a need of a central entity maintaining system-wide information

Fusion nodes can

- Act as a server (push)
- Harvest information (pull)

Information Fusion (3)

Information fusion vs integration

- Fusion takes place in the processing steps
- Integration refers to the final step
  - The end user’s gateway to (integrated) access to the information

Fusion Levels

Early fusion

Signal level

- Signals are received simultaneously by a number of sensors
- Fusion of these signals may lead to a signal with a better signal-to-noise ratio

Feature level

- A perceptual component must first extract the desired low-level features from each modality
- Typically represents them in a multidimensional vector

Late fusion

Decision level

- Combines information from multiple algorithms in order to yield a final fused decision
- May be defined by specific decision rules

JDL Fusion Levels (1)

A process model for data fusion and a data fusion lexicon

Intended to be very general and useful across multiple application areas

Identifies the processes, functions, categories of techniques, and specific techniques applicable to data fusion

JDL Fusion Levels (2)

Process conceptualization

- Sensor inputs
- Source preprocessing
- Database management
- Human-computer interaction
- Four key subprocesses (following next)

JDL Fusion Levels (3)

Level 1 – Object Refinement

- Combines sensor data together to obtain a reliable estimation of an entity position, velocity, attributes, and identity

Level 2 – Situation Refinement

- Attempts to develop a description of current relationships among entities and events in the context of their environment

JDL Fusion Levels (4)

Level 3 – Threat Refinement

- Projects the current situation into the future to draw inferences
  - E.g. about friendly and enemy vulnerabilities, threats, and opportunities for operations

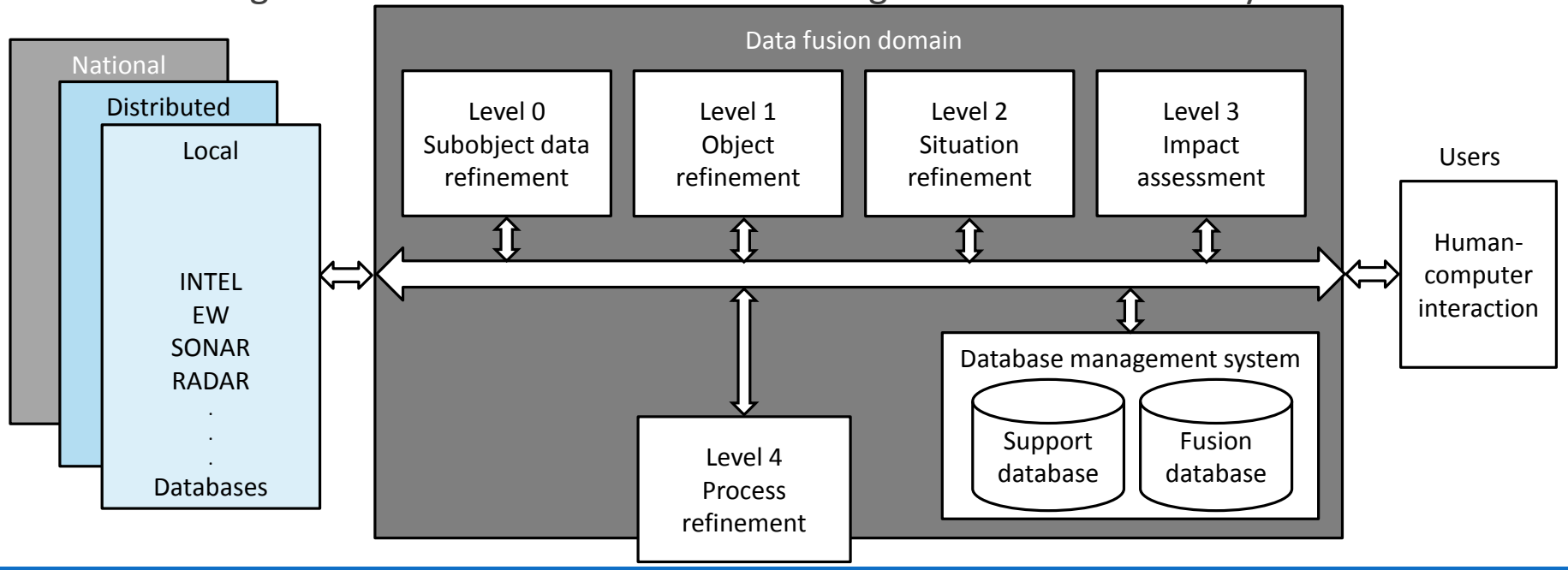
Level 4 – Process Refinement

- A meta-process which monitors the overall data fusion process
  - Assesses and improves the real-time system performance

JDL Fusion Levels (5)

Level 5 – Cognitive or User Refinement

- Added in revisions
- Introduce the human user in the fusion loop
  - The aim is to generate fusion information according to the needs of the system user



Outline

Introduction: Problem Framework

Fusion

The Data layer

Complete Example

The Data Layer (1)

Metadata of multimedia streams

Semi-structured vs structured

- No significant technological challenges into converting semi-structured documents into RDF

Data is produced in the form of streams

- Originating from a set of heterogeneous distributed sources
- No starting or ending point
- A strategy must be devised
  - Define how new facts will be pushed into the system and old facts will be pushed out of it

The Data Layer (2)

Data flow

- Use of a middleware
- Semantic annotation → Knowledge Base
- Ability to answer semantic queries

Modeling Context (1)

Challenges

- Complexity of capturing, representing and processing the concepts
- Expressivity of the description of the world
  - Expressive enough in order to enable specific behaviors
  - Not as complex as to render the collected information unmanageable

Common representation format and vocabulary

- Ensure syntactic and semantic interoperability in the system
- Enable integration with third party data sources

Modeling Context (2)

Uniform context representation and processing at the infrastructure level

- Better reuse of derived context by multiple data producers and consumers

Ontology-based descriptions

- Offer unambiguous definitions of the concepts and their relationships
- Allow further exploitation of the created Knowledge Base
  - Higher level, intelligent, semantic queries
- Formalize knowledge about the specific domain

Semantic Web Technologies in Sensor Networks

Research focuses on modelling

Goal: enable higher level processing for event/situation analysis

- Define ontologies for sensor measurement and sensor description
- Use a meta-ontology (e.g. SUMO)
  - Define a sensor ontology under it
  - Use it to annotate, query, and discover sensors
- Base concepts: Resource, Actor, Environment
  - Parts that describe the state of the resources
  - Entities operating in the resources
  - Surrounding environment conditions

Semantics for Situation Awareness (1)

Situation Theory Ontology (STO)

- A unified expression of the Situation Theory
- Models events/objects and their relationships in OWL
- Can be extended with classes and relations that correspond to the actual application scenario
- Two additional ontologies integrated, in order to be able to use it in a real sensor fusion environment
  - The Time ontology
  - The WGS84 Geo Positioning ontology

Semantics for Situation Awareness (2)

SSN ontology

- Describes sensor data in the Linked Sensor Data context

Semantic Sensor Web (SSW)

- Framework for providing meaning for sensor observations
- Bridges the gap between
  - XML-based metadata standards of the Sensor Web Enablement
  - RDF/OWL-based vocabularies driving the Semantic Web

Modeling Context using Ontologies

A Local-As-View setting

- Optimal when new types of sensors need to be integrated and there is a relatively high variety in structure
- Preferred approach
  - Better than translating user queries from the global schema to each one of the local ones in order to retrieve, aggregate and return results to the user

Data produced by sensors is eventually stored in its Knowledge Base

- Reasoning
  - Infer knowledge which corresponds to events
- Situation assessment
  - Cannot be performed at lower fusion levels

Sensor Data

Audiovisual

- Audio/video stream

Non-audiovisual

- Any kind of streamed sensor observations
  - Temperature, RFID tags, etc.

Incoming data is subject to filtering

- Not all information is stored and processed
- Only the subset that is meaningful and needed by the processing modules
  - E.g. discard out-of-range values

Challenges in Homogenizing Sensor Data

Numerous standards to annotate various areas of sensor data

- MPEG-7 for audiovisual
- Federal Geographic Data Committee (FGDC) for geographical information

Accuracy of the annotations

Convenience of updates and maintenance

Added value to the content itself

- Not always clear how the user can benefit from the existence of such annotations

Tracker errors

- False, missing, incorrect annotations

A Two-step Approach

Annotate sensor data according to the nature of its tracker

Homogenize the data under a common vocabulary

- Containing the captured values and semantics

Real-time vs. Near-real-time (1)

A real-time system

- Must satisfy explicit bounded response time constraints to avoid failure and present consistency regarding the results and the process time needed to produce them
- Emphasis in predicting the response time and the effort in reducing it
  - Response time: the time between the presentation of a set of inputs and the appearance of all the associated outputs

A real-time sensor fusion system

- Produces certain alerts (outputs)
- Connected to the appearance of certain inputs (events)

Real-time vs. Near-real-time (2)

Near real-time

- E.g. systems that schedule their operations at fixed time intervals
- Frequency of the updates depends on the application
  - E.g. in a surveillance scenario, more frequent updates would be needed than in an environmental monitoring scenario

Sensor inputs/outputs

- Real-time signals
  - E.g. audiovisual streams
- Near-real time/asynchronous messages
  - E.g. RFID readings

Data Synchronization and Timestamping (1)

Synchronization

- Rules of the form

  if $event_1$ occurred before $event_2$ then…

Data Synchronization and Timestamping (2)

Two kinds of timestamps

- The time the event was recognized
  - Local time at the node that made the measurement
  - Problem: how to synchronize the sensor network to a common clock
- The time the event arrived in the fusion node
  - Fusion node
    - A node that fuses information incoming from other nodes
  - No need of a common clock
    - The fusion node will timestamp events upon their arrival
  - Followed when there are no great delays in communicating messages

Data Synchronization and Timestamping (3)

Timestamping can be

- Distributed
  - At each node
- Centralized
  - At a central node, maintaining a common clock

Windowing

A mechanism to assure continuous data processing

In order to process newly generated information properly, the system will not have to take into account all existing information

Maintain a working memory window

Streams are unbounded

- Cannot fit into memory in order to be processed as a whole

Windowing-related Decisions

The measurement unit

The size

The window behavior

- Sliding windows
- Tumbling windows
- Landmark windows
- Partitioned windows
- Predicate windows

Rules applied real-time are restricted to the current information window

The (Distributed) Data Storage Layer (1)

Generated information needs to be stored and processed before being communicated to the system

Each node maintains its perception of the real world

- Physically stored in a local database

Multi-sensor stream processing systems purposed to function under a heavy load of information

- Preferred database design geared towards scalability
- I.e. decentralized to the maximum extent possible
  - Not centralizing collected information

The (Distributed) Data Storage Layer (2)

An approach in order to maintain scalability

- Each node keeps the amount of information required for its local operation
  - Local database schema has to relate only to the hosted components
- Restrict information communicated throughout the system
- Communicates to the central node only higher level information
  - E.g. detected events or entities

Relational to RDF in Sensor Data Streams

Produced messages will have to be eventually converted to RDF and ultimately inserted in an ontology

A mapping layer

- Map the relational schema to the semantic schema

Mapping Layer (1)

Push strategy

- Forward data to the ontology using semantic notation as soon as they are generated
- Advantages
  - Transformations are executed fast
  - The ontology is always up-to-date
- Disadvantages
  - Each lower level node will have to implement its own push method
  - Risk that the ontology will be populated with data even when no query is sent to the semantic layer

Mapping Layer (2)

Pull strategy

- Transform relational data to semantic on request
  - I.e. during query time
- Similar to RDF Views
- Advantages
  - Actual mapping defined at semantic level
  - Data transformed on request
    - The ontology will accumulate instances needed for the actual query evaluation
- Disadvantages
  - Could lead to longer response times during queries

Outline

Introduction: Problem Framework

Fusion

The Data layer

Rule-based Reasoning

Complete Example

Rule-based Stream Reasoning in Sensor Environments (1)

Reasoning

- Infer implicit information
- Use the ontology structure and instances in order to draw conclusions about the ongoing situations
- Based on a set of provided rules, applied on the created knowledge base

Extend the knowledge base by using rules

- To describe complex situations/events
- To create alarm-type of objects if certain conditions are met
- To depict the desired behavior and intelligence

Rule-based Stream Reasoning in Sensor Environments (2)

Event-condition-action pattern

on *event* if *condition* then *action*

An event is a message arrival indicating a new available measurement

Two distinct sets of rules

- Mapping rules
- Semantic rules

Rule-based Stream Reasoning in Sensor Environments (3)

Mapping rules

- Specify how sensor measurements represented in an XML-based format will be mapped to a selected ontology
- Can fetch data from the XML-like message and store it into the ontology model in the form of class instances
- Can be perceived as the necessary step bridging the gap between semi-structured data and ontological models

Rule-based Stream Reasoning in Sensor Environments (4)

Semantic rules

- Derive new facts, actions or alerts
  - Based on existing facts and knowledge
- Perform modifications on the ontology model
- Depend on the specific domain or deployment scenario
- Involve high-level concepts , meaningful to humans
  - E.g., “when a certain area under observation is too hot, open the ventilating system”
  - The “too hot” conclusion will probably be inferred from a set of current observations coupled with the knowledge stored in the ontology

Rule-based Stream Reasoning in Sensor Environments (5)

Rule-based systems

- Combine real-time data with stored sensor data
- Fire rules based on data obtained from sensors in real-time and classified as ontology instances

Rule-based problem solving

- An active topic in AI and expert systems
- Classic approach comes from work on logical programming and deductive databases
- Conventional rule engine implementations based on the Rete algorithm

Rule-based Stream Reasoning in Sensor Environments (6)

Stream reasoning

Performing reasoning on a knowledge base comprising stable or occasionally changing terminological axioms and a stream of incoming assertions or facts

Will lead the way for smarter and more complex applications

- E.g. traffic management, fastest route planning, environmental monitoring, surveillance, object tracking, disease outburst detection, etc.

Rule-based Stream Reasoning in Sensor Environments (7)

RIF – Rule Interchange Format

- W3C recommendation
- A core rule language
- A set of extensions (dialects) that allow the serialization and interchange of different rule formats

Also RuleML and SWRL

Also using SPARQL CONSTRUCT

- SPIN being an extension to this approach

Rule-based Reasoning in Jena (1)

Jena Semantic Web Framework

- The most popular Java framework for ontology manipulation
- Includes an inference engine, can be used as a reasoner
- Includes a number of predefined reasoners
  - Transitive reasoner
  - RDFS rule reasoner
  - OWL, OWL mini, OWL micro
  - DAML micro reasoner
  - A generic rule reasoner that can be customized to meet specific ad hoc application demands

Rule-based Reasoning in Jena (2)

Forward chain rules (body ⟶ head)

- When the body is true, then the head is also true

Built-in rule files of the form:

```
[rdfs5a: (?a rdfs:subPropertyOf ?b), (?b rdfs:subPropertyOf ?c) -> (?a rdfs:subPropertyOf ?c)]
```

Symmetric and transitive properties in OWL:

```
[symmetricProperty1: (?P rdf:type owl:SymmetricProperty), (?X ?P ?Y) -> (?Y ?P ?X)]
[transitivePropery1: (?P rdf:type owl:TransitiveProperty), (?A ?P ?B), (?B ?P ?C) -> (?A ?P ?C)]
```

Rule-based Reasoning in Jena (3)

Builtin primitives

- Functions that can be used in the place of rule predicates
  - isLiteral(?x), notLiteral(?x),
  - isFunctor(?x), notFunctor(?x)
  - isBNode(?x), notBNode(?x), etc.
  - Custom builtin primitives can be developed

Rule-based Reasoning in Virtuoso (1)

Reasoning essentially a set of rules applied on the RDF graph

Relatively simple reasoning

- Scalability instead of rich inference capabilities

Rule sets

- Loaded using the rdfs_rule_set function
- User can specify a logical name for the rule set, plus a graph URI
- Are provided as a context to user queries
- Can be referenced by SPARQL queries or endpoints

Rule-based Reasoning in Virtuoso (2)

Queries return results as if the inferred triples were included in the graph

- Inferred triples generated by reasoning (the rule set) are generated at runtime, are not physically stored

Outline

Introduction: Problem Framework

Fusion

The Data layer

Rule-based Reasoning

Complete Example

Complete Example

Information originating from a distributed sensor network

Architecture of a Multi-Sensor Fusion System

- Based on GSN
- Operating at all JDL levels
- Create Linked Data

Fusion and its potential capabilities

Combine semantic web technologies with a sensor network middleware

Blend ontologies with low-level information databases

Proof-of-Concept Implementation

An LLF node

Two processing components

- A Smoke Detector
- A Body Tracker
- Each component hosted on a computer with a camera
  - RTP streams

An HLF node

A Central node

- Overall system supervision

The GSN Middleware (1)

Needed in order to perform LLF

Open-source, java-based

Allows processing data from a large number of sensors

Covers LLF functionality requirements in sensor data streams

The GSN Middleware (2)

Virtual sensor

- Any data provider (not only sensors)
- Configuration file in XML
  - Processing class
  - Windowing
    - Time- or tuple-based sliding window size
  - Data source
  - Output fields

The GSN Middleware (3)

GSN Servers

- Can communicate between them
- Support input from more than one data stream
- Can form a network
  - Allow information collection, communication, fusion, integration

Data acquisition

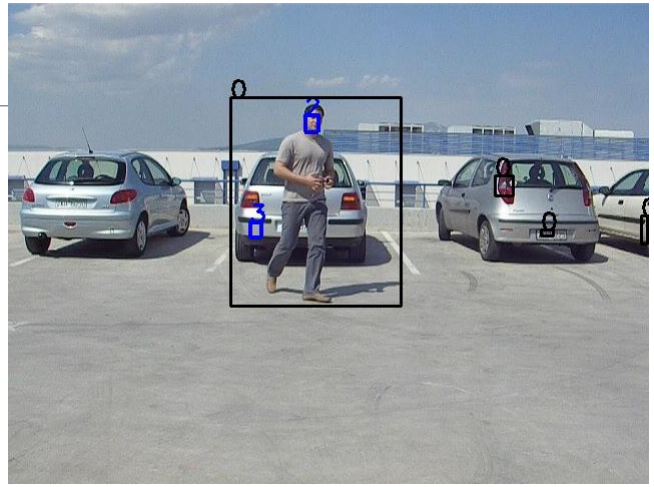
- An SQL-like procedural language
  - Combine and fuse the information
  - User-defined LLF functions
- GSN takes care behind the scenes about crucial issues
  - E.g. thread safety, synchronization, etc.

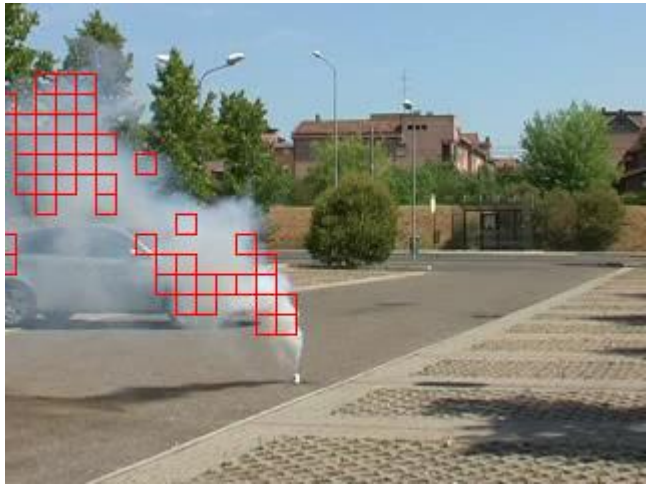
Low Level Fusion (1)

Camera generates an RTP feed with its perception of the world

Feed processed by signal processing components

- The Body tracker
  - Reports NumberOfPersons
- The Smoke detector
  - Reports NumberOfSmokeParticles
- Both components
  - Receive a POLL command at fixed time intervals
  - Return an XML file





Low Level Fusion (2)

Sampling the video source takes place asynchronously

- Camera streams at 25 fps
- 500 ms between two consecutive polls suffices for LLF

Tracking a person is more demanding than detecting it

- Implies comparing consecutive frames

Asynchronous processing

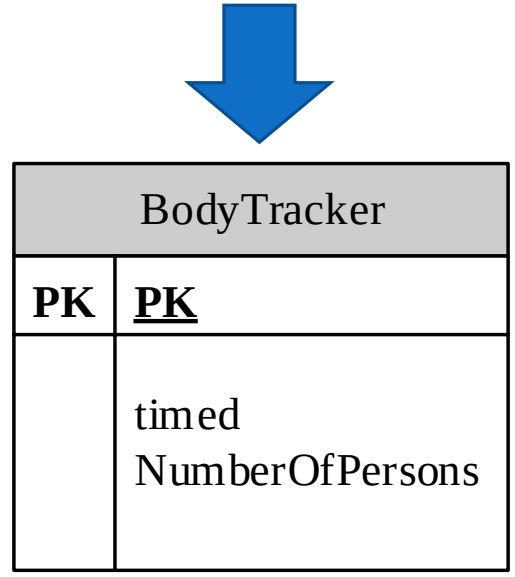
- The camera fps rate not aligned with the produced message rate

Low Level Fusion (3)

Virtual sensor XML configuration files

- Body tracker
  - PK: primary key (auto-incremented integer)
  - Timed (timestamp)
  - NumberOfPersons (integer)
- Similarly for the smoke detector
  - ExistenceOfSmoke (boolean)

```
<virtual-sensor name="BodyTracker">
  …
  <output-structure>
   …
   <field name="NumberOfPersons"
          type="int" />
  </output-structure>
  …
  <storage history-size="1m" />
</virtual-sensor>
```



Low Level Fusion (4)

LLF virtual sensor definition

- Two data providers (source streams)
- Produce events only when the fusion conditions are satisfied
  - A person and smoke are detected, concurrently

```
SELECT source1. NumberOfPersons AS v1,
       source2. ExistenceOfSmoke AS v2
FROM source1, source2
WHERE v1 > 0 AND v2 = "true"
```

Low Level Fusion (5)

Fusion

- Results provided by taking into account the inputs from both the sensors
- More processing components, sensors, sensor types and more complex fusion conditions can be integrated into the system

Low Level Fusion

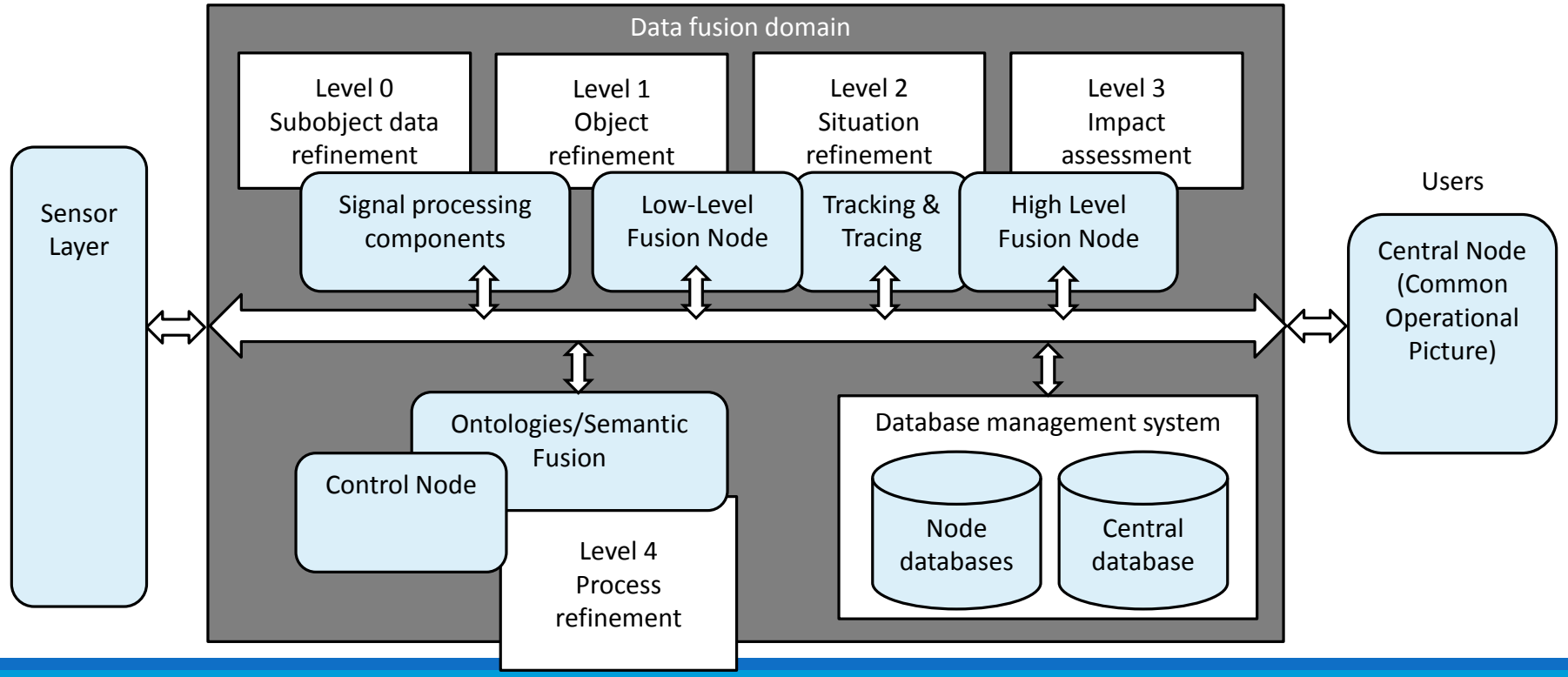
- Allow only important information to be forwarded to the upper layers

No semantic enrichment so far

A Sensor Fusion Architecture (1)

From theory to practice

Address matters that correspond to all JDL levels



A Sensor Fusion Architecture (2)

Collected data is leveraged into meaningful and semantically enriched information

- I.e. transformed from signals to higher level information

All levels of multi-sensor data fusion are applied to the data

A Sensor Fusion Architecture (3)

JDL Level 0 – Subobject data refinement

- Signal Processing components operate at this level
- System collects observation data
  - Sensory data input mostly from Electro-Optical sensors (e.g. cameras, microphones, etc.)
- Virtualizes the inputs of the sensors
- Performs spatial and temporal alignment

A Sensor Fusion Architecture (4)

JDL Level 1 – Object refinement

- System analyzes the collected data
- Extracts features, objects, and low level events
- Detected objects may include persons or vehicles
- Low level events may include movements
- Implementation can be based on e.g. GSN

A Sensor Fusion Architecture (5)

JDL Level 2 – Situation refinement

- System aggregates and fuses the objects, events and the general context in order to refine the common operational environment
- System is able to perform object fusion and tracking, and identify situations in the system, in a manner that is not feasible by a sensor alone

A Sensor Fusion Architecture (6)

JDL Level 3 – Impact assessment

- Events and objects refined at JDL Level 2 are subsequently analyzed, using semantically enriched information, resulting at the inferred system state
- High Level Fusion (HLF, e.g., using Virtuoso's rule engine) operates at this level
  - Fuses the information, enabling reasoning that can infer and assess potential impacts
    - E.g. situations and respective alerts

A Sensor Fusion Architecture (7)

JDL Level 4 – Process refinement

- System can refine its operation by providing feedback to the sensor layer
- Ontologies/Semantic fusion component
  - Perform analysis on the system-wide high-level collected information in order to infer events and potential risks and threats
  - Hosted at the Central Node
  - Monitor and curate the whole system

A Sensor Fusion Architecture (8)

Level 5 – Cognitive or User Refinement

- A higher level, introduced in revised versions of the JDL model
- Control Node component
  - Hosted at the Central Node
  - Responsible for issuing commands back to the sensors
- Common Operational Picture component
  - System front-end
  - Provide a visualization of the system state
    - Deployed sensors, detected objects, sensed events and inferred threats
  - Allows for human-computer interaction

A Sensor Fusion Architecture (9)

Two types of databases to support system operation

- Support databases
  - Materialized as a node database, kept locally at each node
  - Collected data are kept close to their source of origin, allowing for each node to be configured and maintained according to its environment and system's needs
  - Allow system to scale
- Central Node database
  - Plays the role of the fusion database, where higher level fused information is kept

High Level Fusion Example (1)

Introduction

- Ontologies
  - Powerful means to describing concepts and their relations
    - Entities, events, situations, etc.
    - JDL levels 2 and 3
  - Main information gathering point in the proposed architecture
- Reasoning
  - Infer new knowledge
  - Cannot be performed at the LLF level

High Level Fusion Example (2)

The scenario

- Demonstrate inference capabilities of the proposed architecture
- Full use of the data chain
  - From low level sensor data to high level complex event detection and situation awareness/threat assessment
- Detect threats to public safety and associate an action plan
- I.e. detect persons and smoke in an area of interest, e.g. a petrol station

High Level Fusion Example (3)

Configure GSN to use Virtuoso as its backend at the HLF node

Develop RDF views in Virtuoso over the sensor data

Use the Situation Theory Ontology (STO)

- The STO:FocalSituation class marks significant situations that prompt action by security personnel

High Level Fusion Example (4)

Perform reasoning in Virtuoso

- Triple store populated with data from the Smoke detector and Body tracker components
- Reasoning associates the smoke detection event with a criticality factor
  - According to the events and geospatial information modeled in STO

The scheduler component in Virtuoso

- Update the triple store via RDF views from the sensor inference database
- Subsequently invoke on the reasoning

High Level Fusion Example (5)

Query to retrieve focal situations with their event related details

- Time, location, textual descriptions

```
SELECT ?focalsituation ?timeTxt ?locTxt
WHERE {
  ?focalsituation STO:focalRelation ?event .
  ?event STO:hasAttribute ?time .
  ?event STO:hasAttribute ?location .
  ?time rdf:type STO:Time .
  ?time time:inXSDDateTime ?timeTxt .
  ?location rdf:type STO:Location .
  ?location rdfs:label ?locTxt .
}
```

High Level Fusion Example (6)

Integration with emergency departments achieved by sending details on significant threats found in SPARQL result sets to a Web service endpoint

- The smoke event together with location data is forwarded to emergency personnel

More details in:

- Doulaverakis et al. An approach to intelligent information fusion in sensor saturated urban environments. EISIC 2011, Athens, Greece