Efficient handling of distributed video analysis results: An agent-based data warehousing approach

Cyril Carincotte
carincotte@multitel.be

CARETAKER IST-4-0227231

Content Analysis and Retrieval technologies to Apply Knowledge Extraction to massive Recording
✓ Introduction

✓ **Data warehousing and video monitoring system**

✓ **Agent-based data warehousing architecture**

  - Source data structure and exchange
  - Stored knowledge schema
  - Agent Oriented Programming (AOP)
    - Process-oriented agents
    - Data-oriented agents
  - Storage specifications
  - Performance evaluation

✓ **Conclusion & perspectives**
Today’s Video Content Analysis (VCA) enables:

- **reliable extraction of low-level features**
  (motion of moving objects, trajectories, . . .)
- **complex semantic event recognition**
  (activity recognition, abandoned luggage detection, . . .)

However produced knowledge in monitoring systems is not exploited in an efficient way:

- Use of the extracted information in the
  - **Search** (query on DB by type of event, time, …)
  - **Retrieval** (retrieve all videos matching this event features)
  - **Data management context** (off-line process of DB content)

Need to propose well-designed Video Monitoring systems!
Introduction

State-of-the-art in Video Monitoring systems

- VSAM [2000]
  - Object/event database of known objects

- RETRIEVE [2002]
  - MPEG-7 meta-data descriptors used to describe video content

- SfinX [2004]
  - Sequence data representation stored in events database

- IBM-S3 [2005]
  - Meta-data indexed into predefined tables (IBM DB)

Target: Propose a flexible system handling unknown events supporting a posteriori modification of used schema.

Use Data Warehouse for storage task!
Data warehousing & video monitoring system

Data Warehouse ???

✓ (Specific) **Multi-dimensional database** allowing
  ➢ information analysis and management.
  ➢ processing not achievable within classical databases.

✓ Data warehouses
  ➢ specifically designed for **reporting and analysis purposes**.
  ➢ support of management decisions.

Target:
  ➢ Exploit data warehouse capabilities in a video analysis environment
    ▪ Ability to support complex queries and analysis (data mining, ...) on
      stored data without slowing down on-going analysis task.
Data warehousing & video monitoring system

Data Warehouse in which environment?

Audio/Video acquisition

Low-level processing

I/O handling

DB access & storage

High-level processing

Two levels of knowledge:

- first layer of primitive descriptors, extracted from the raw data streams.
- second layer of higher semantic events from which rich meta-data can be produced.
Hierarchical modelling of:
- the VCA (A & C)
- I/O handling and DB access via interaction modules (B & D)

Data Warehouse (E)

Query/retrieval on DB:
- handle by dedicated interaction module (F)
Agent-based data warehousing architecture

Source data structure and exchange (1)

Two XML schemas (.xsd files) defining:
- primitive descriptors level outputs,
- higher semantic events level outputs.

- Need to distribute the XML results over the network
  - Use RSS technology
Agent-based data warehousing architecture

Source data structure and exchange (2)

- XML documents wrapped in RSS feeds
- Data exchange reduced to RSS handling

```xml
<rss version="2.0">
  <channel>
    <title>IST FP6-027231</title>
    <link>http://www.ist-caretaker.org/</link>
    <description>CARETAKER_VCA</description>
    <category>WP4</category>
  </channel>
</rss>
```

RSS wrapping

RSS feed sent over the network

Raw XML file

RSS wrapping

RSS wrapping
Agent-based data warehousing architecture

Stored knowledge schema (1)

Which format for storage of XML results in Data Warehouse?
Agent-based data warehousing architecture

Stored knowledge schema (2)

Target:
- XML schema (.xsd files) \textit{a priori} unknown
- System not aware (before initialization) of XML structures
  - \textit{no a priori} system optimization or database schema definition.

✓ XML meta-data considered as knowledge we want to
  - \text{structure},
  - \text{manage} and
  - \text{store}
  in an optimized and \textit{unsupervised way}.

RDF: framework used for defining and sharing metadata by means of graphs.

Resort to RDF (semantic web) to represent and share knowledge.
Main advantages of RDF are:
- stored knowledge is incrementally built
- no schema need to be a priori determined before being instantiated.
- duplicated literals in XML files (very frequent) are not duplicated in DW

RDF graphs are sets of triples, also called statements, of the form subject predicate object.

Let’s define:
- \( U \): an infinite set of RDF URI references
- \( B = \{ N_i : i \in \mathbb{N} \} \): a set of RDF blank nodes
- \( L \): an infinite set of RDF literals (e.g. string, int, ...)

A triple \((s, p, o) \in (U \cup B) \times U \times (U \cup B \cup L)\) where \(s\) is a subject, \(p\) a predicate and \(o\) an object is called an RDF triple.
Agent-based data warehousing architecture

Agent-Oriented Programming (1)

What are Agents?
- autonomous/adaptative/reactive entities
- applications able to work on their own on dedicated sets of data related to their core domain

Why using AOP?
- high degree of flexibility, autonomy and proactivity
- Prove to be well-suited in such context (EasyMeeting, [2004])

Two kinds of agents in system:
- process-oriented agents, in charge of the RSS and knowledge flow management,
- data-oriented agents, handling the data itself.
Agent-based data warehousing architecture
Agent-Oriented Programming (2)

✓ **Process-oriented agents**
  - Implemented with JACK (agent-oriented development environment)
  - Agents in charge of the RSS and knowledge flow handling.

- Have to handle data
  - Tricky business.
  - Dev. of dedicated data-oriented (data-oriented agents)
Agent-based data warehousing architecture

Agent-Oriented Programming (3)

System

RSS Flow

| Correspondence relation | External data transfer | Internal data transfer |

High-level analysis

Low-level analysis

Data flow
Agent-based data warehousing architecture

Agent-Oriented Programming (4)

✅ Data-oriented agents
- implemented in pure Java,
- form a set of agents corresponding to XML documents
- Not XML document containers or “translations”
  - Agents are themselves XML documents able to
    - self-replication, i.e. replicate its structure and content,
    - self-reification, i.e. retrieve the knowledge stored in the data warehouse and to instantiate its structure with it,
    - self-exportation to XML,
    - self-exportation to RDF/XML and N-Triples, and optimization of the graph produced in order to minimise reification time and the required space,
    - updating its components,
    - modifying its structure from new data,
    - merging with other autotroph agents of possibly different types,
    - and applying a mask on the exported data.
Agent-based data warehousing architecture

Agent-Oriented Programming (5)

✓ Typical process flow for data-oriented agent

➢ New XML file received by system
  ▪ Type analyse (low/high level)
  ▪ Matching skeleton replication
  ▪ Skeleton modification
  ▪ Content update
Agent-based data warehousing architecture

Storage specifications

✓ Storage based on 3store (MySQL backed RDF triple store).

✓ Dedicated cache (red/black tree)

✓ Particular synchronization mechanisms

✓ Knowledge permanently stored through commit (nonvolatile DW property)
Agent-based data warehousing architecture

Performance evaluation (1)

- VCA simulators producing XML as I/O
- XML files
  - metro station monitoring (~13min @ 25fps)
  - Low-level stubs
    - Moving object detection (person, group of persons, ...)
  - High-level stubs
    - Activity recognition (stay in platform, queue at ticket vending machines, ...)
- 3Store knowledge base
  - more than 4,000,000 triples
Agent-based data warehousing architecture

Performance evaluation (2)

- System performance markedly improved when using cache memory (for DB access operations, such as insertions, queries and reifications)

- Interest of RDF optimization
  - Reification time from 3store with non-optimized RDF structure is ~1600 ms (not reported in Tab.)
  - Reification time with optimized RDF structure is ~40 ms, i.e. 40 times faster.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Average time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reification (from cache)</td>
<td>0</td>
</tr>
<tr>
<td>Reification (from 3store)</td>
<td>40</td>
</tr>
<tr>
<td>Query (from cache)</td>
<td>3</td>
</tr>
<tr>
<td>Query (from 3store)</td>
<td>50</td>
</tr>
<tr>
<td>Agent replication</td>
<td>0</td>
</tr>
<tr>
<td>Agent initialization</td>
<td>5</td>
</tr>
<tr>
<td>Agent → XML</td>
<td>5</td>
</tr>
<tr>
<td>Agent → RDF</td>
<td>5</td>
</tr>
<tr>
<td>Insertion (in cache)</td>
<td>0</td>
</tr>
<tr>
<td>Insertion (in 3store)</td>
<td>70</td>
</tr>
</tbody>
</table>
Conclusion

✓ Open, generic and flexible monitoring system
  ➢ Handle meta-data without being aware of structure before initialization

✓ Main contributions
  ➢ Use of a data warehousing architecture
    ▪ RDF graphs used to manage and store data
  ➢ Combined with Agent-Oriented Programming
    ▪ Dedicated kind of agents proposed (autotroph agents)
  ➢ Optimizations to improve system reactivity
    ▪ RDF graph depth restriction
    ▪ Dedicated cache memory implementation

✓ Performance evaluation
  ➢ Local tests with real meta-data showed very promising results
Perspectives

- Real-scale evaluations within entire monitoring environment
  - Acquisition system & audio/video streaming,
  - Video content analysis, metadata generation and RSS publication,
  - Metadata handling and storage in data warehouse
  - Query/retrieval of stored knowledge through GUI
Thank you!

Questions?

For further information about CARETAKER project (FP6-0272312), please visit
http://www.ist-caretaker.org/