XIRAF - Ultimate Forensic Querying

By

Wouter Alink, Raoul Bhoedjang, Peter Boncz and Arjen de Vries

Presented At

The Digital Forensic Research Conference

DFRWS 2006 USA Lafayette, IN (Aug 14th - 16th)

XIRAF
Ultimate Forensic Querying

DFRWS - August 15, 2006

Wouter Alink, Raoul Bhoedjang
Netherlands Forensic Institute

Peter Boncz, Arjen de Vries
Centrum voor Wiskunde en Informatica

Introduction

XIRAF

“An XML Information Retrieval Approach to Digital Forensics”

Collect, manage, and query information extracted from digital evidence

Outline

- Problem statement
- XIRAF approach
- XIRAF architecture
- Forensic application areas
- Initial experiments
- Conclusion

Typical investigation steps

1. Media capture
2. Feature extraction
3. Analysis
4. Reporting

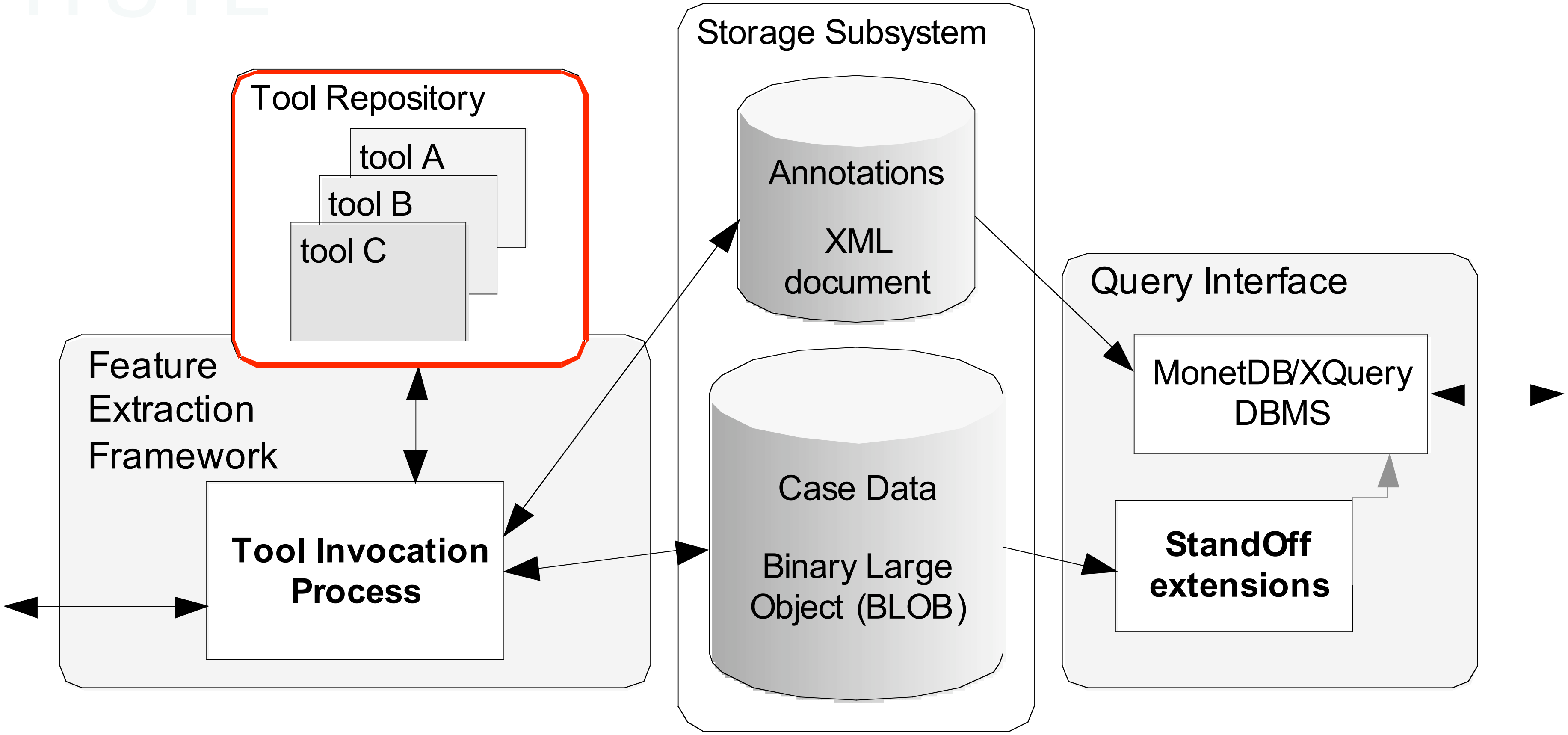
Problem identification

- Large amounts of data
  - Investigation restricted by deadlines
  - Too much information to track manually
- Diversity of data and tools
  - Many different formats
  - Many stand-alone forensic tools

Approach

- Clean separation between feature extraction and analysis
- A single, XML-based output format for tools
- XML database technology to analyze extracted features
- Use of existing forensic analysis tools

XIRAF architecture

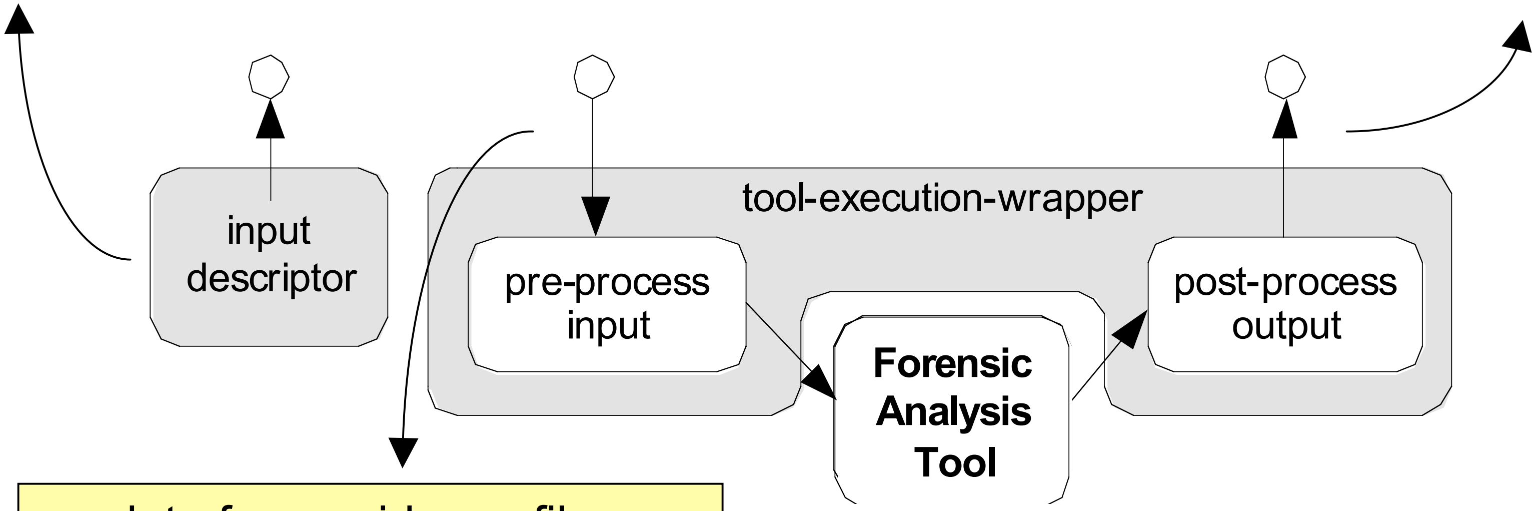


Tool wrapper

```
<photo>
  <camera>Canon<camera>
  <taken-on>
    <date>15-12-2005</date>
  </taken-on>
</photo>
```

- metadata (features/traces)
- new view of the original data

```
//file[mime="image/jpeg"]
```

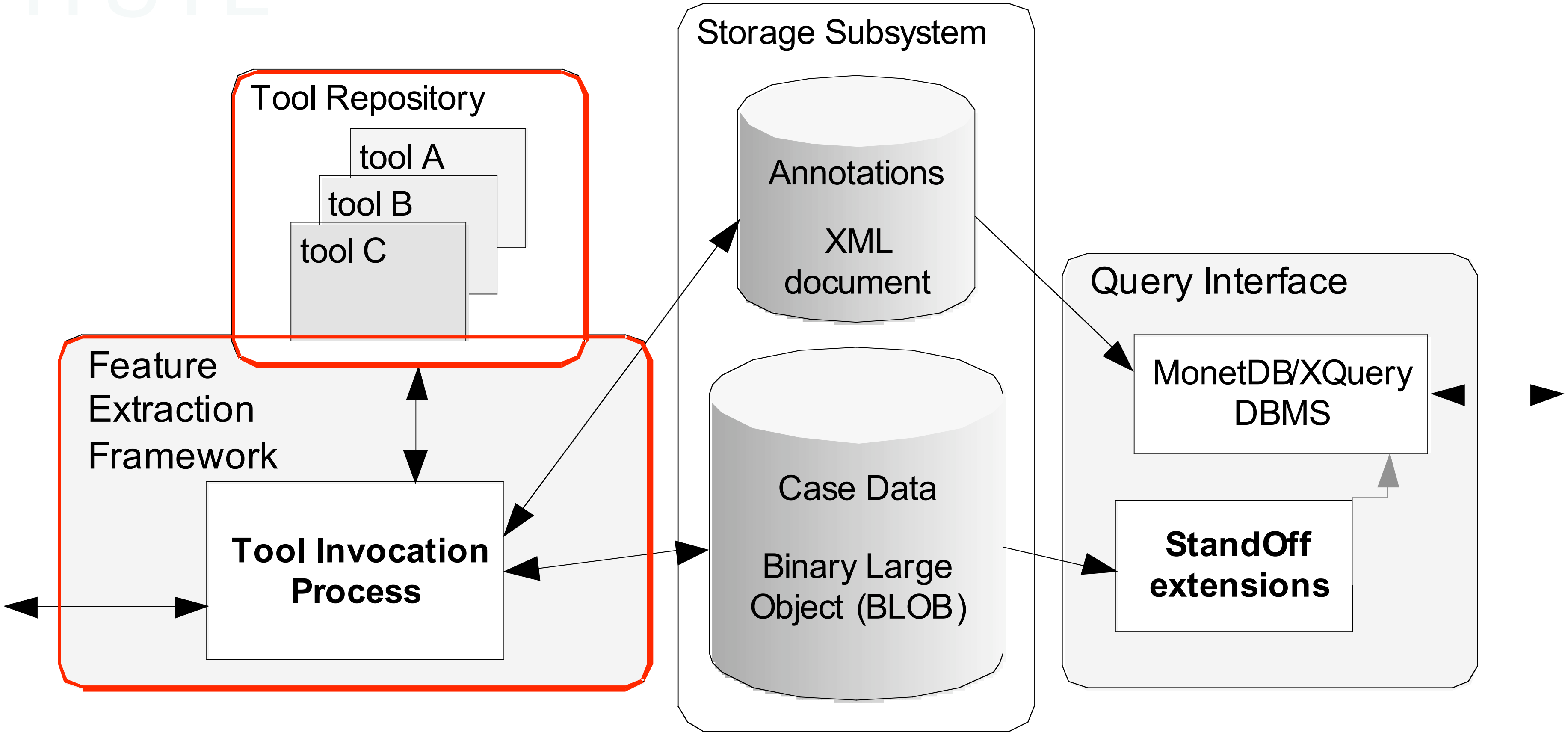


- data from evidence files `Photo03.jpg'
- Optional:
  additional metadata

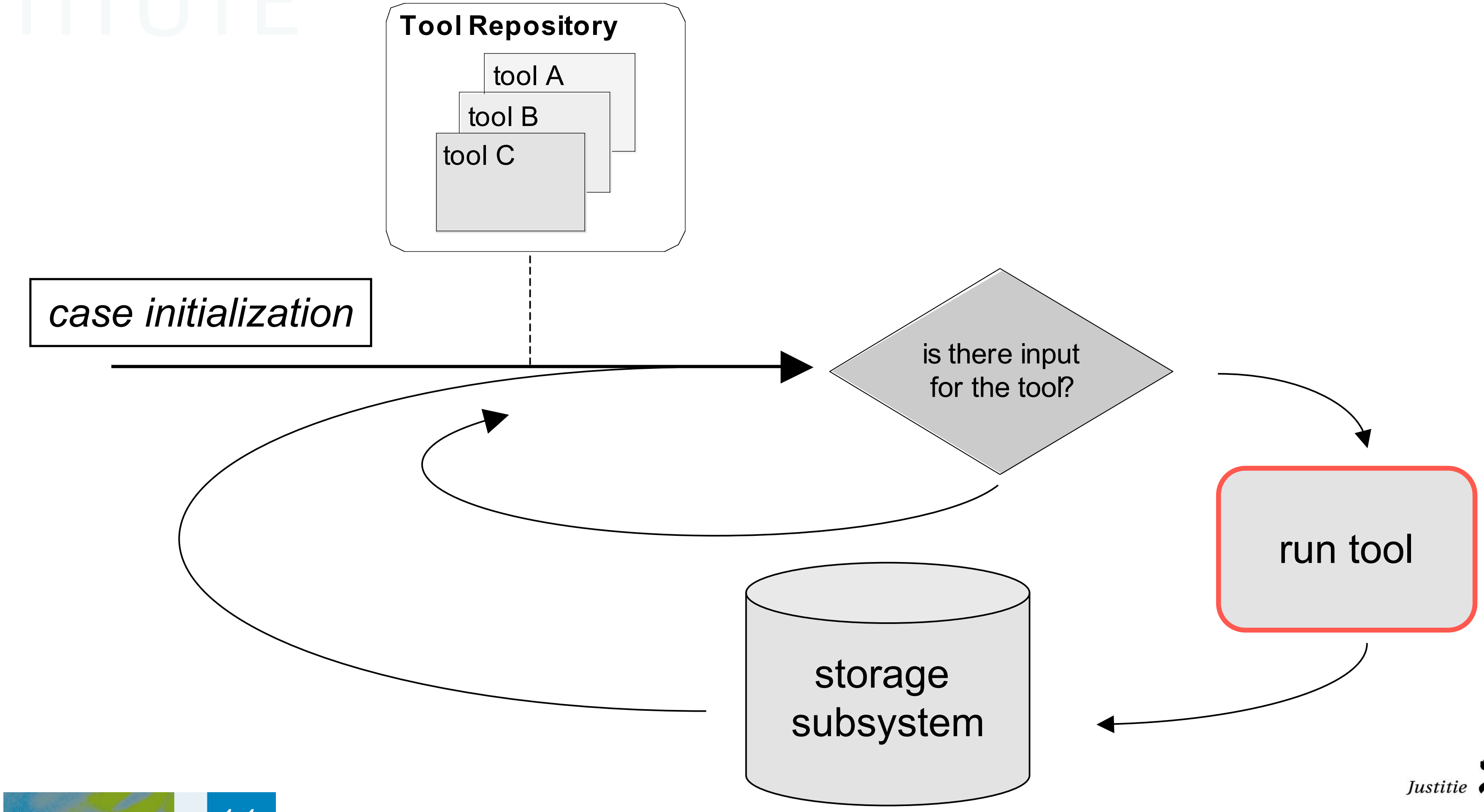
Tool repository

- Feature extraction tools
- Gain knowledge about an ‘object’:
  - volume
  - file-system
  - image
  - email
- Some of the wrapped tools:
  - file-system dissector
  - windows registry analyzer
  - EXIF-data parser
  - carving tool
  - IE-history parser
  - Hashing tool

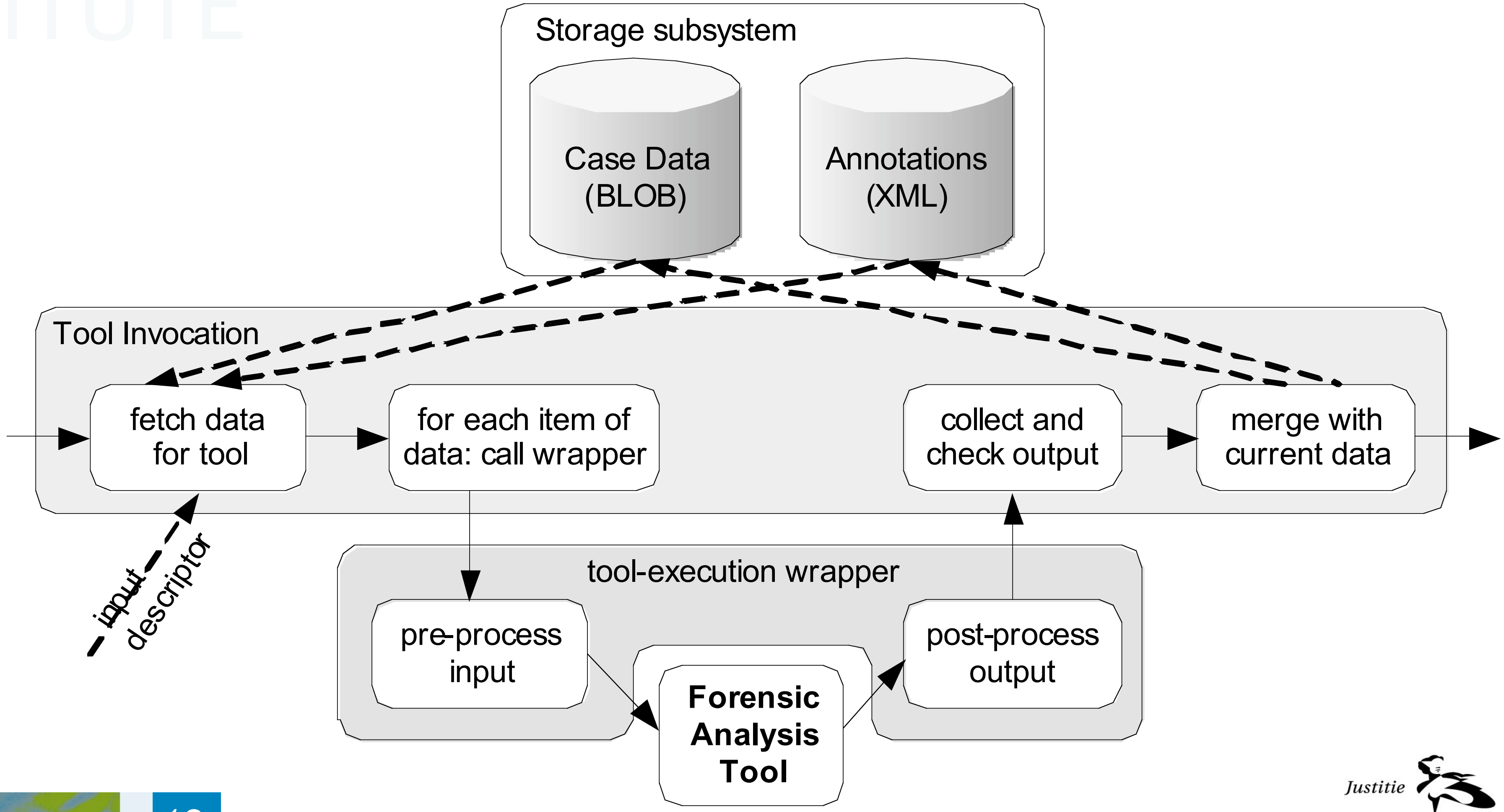
XIRAF architecture



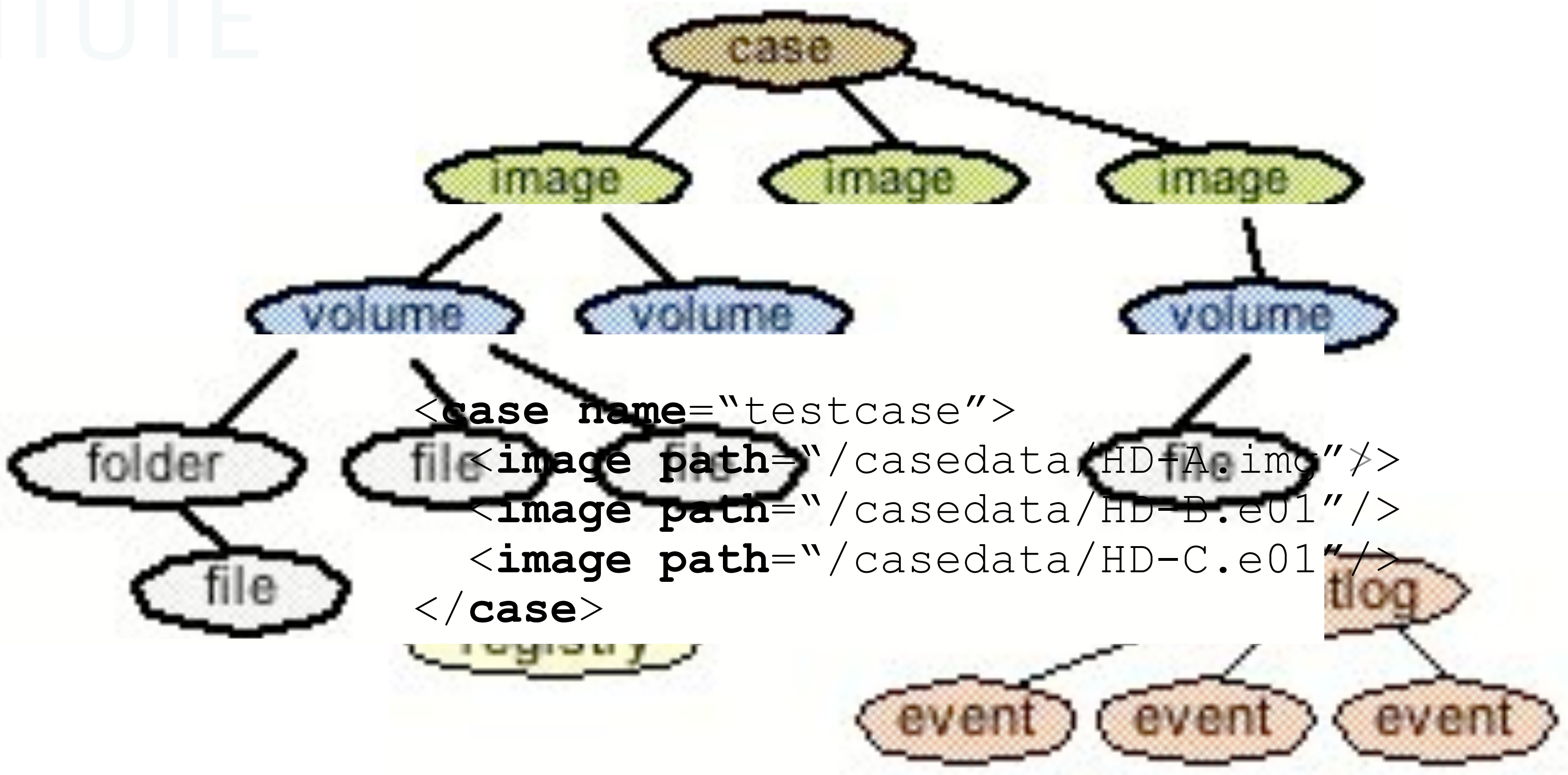
Feature extraction framework



Feature extraction framework



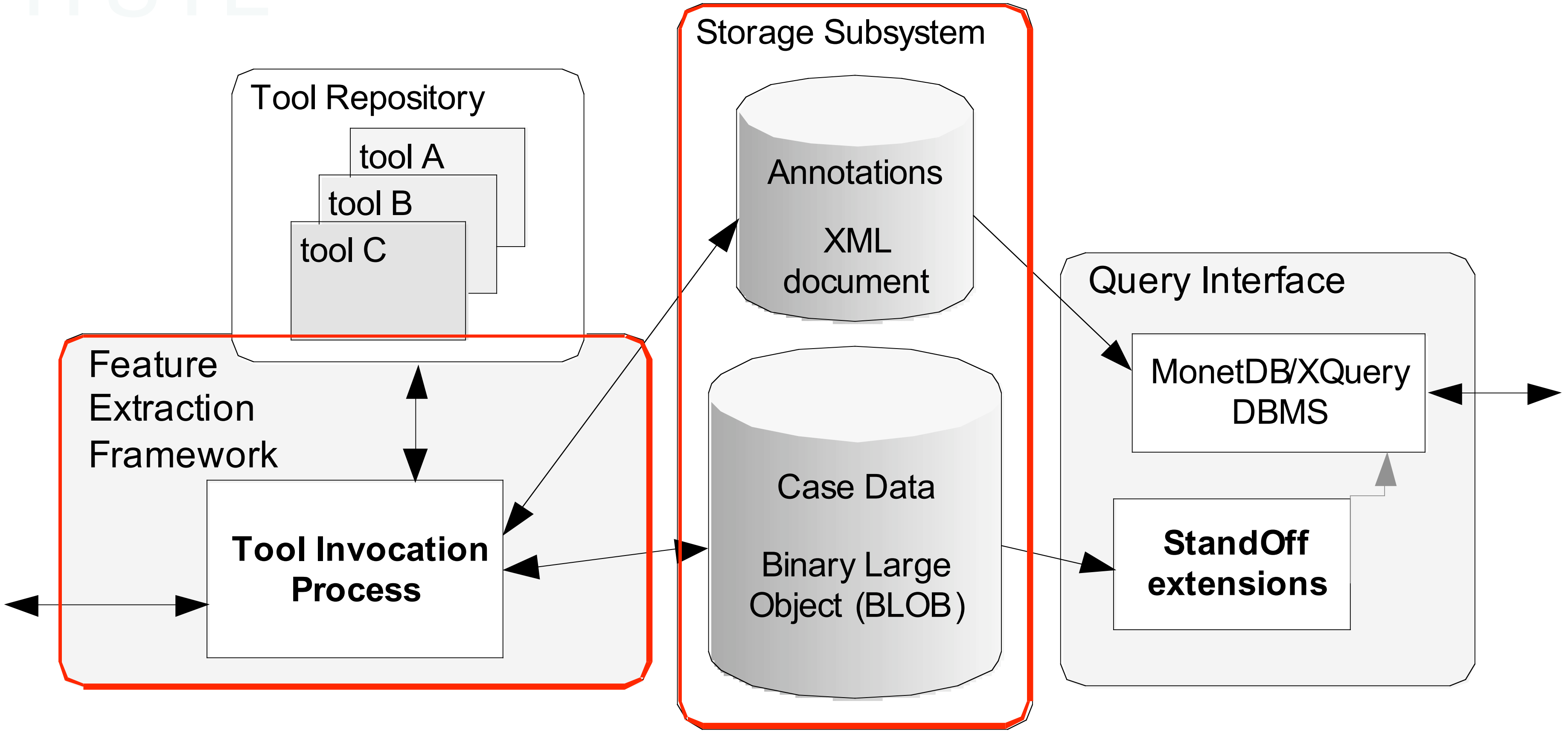
Feature extraction



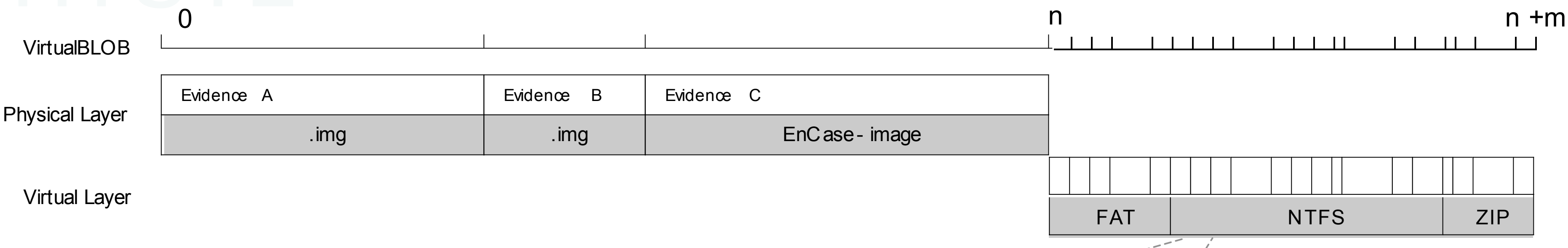
```
<case name=“testcase”>
  <image path=“/casedata/HD-A.img”/>
  <image path=“/casedata/HD-B.e01”/>
  <image path=“/casedata/HD-C.e01”/>
</case>
```

```
  <image path=“/casedata/HD-C.e01”>
    <volume label=“MP3”/>
  </image>
</case>
```

XIRAF architecture



Virtual BLOB and XML



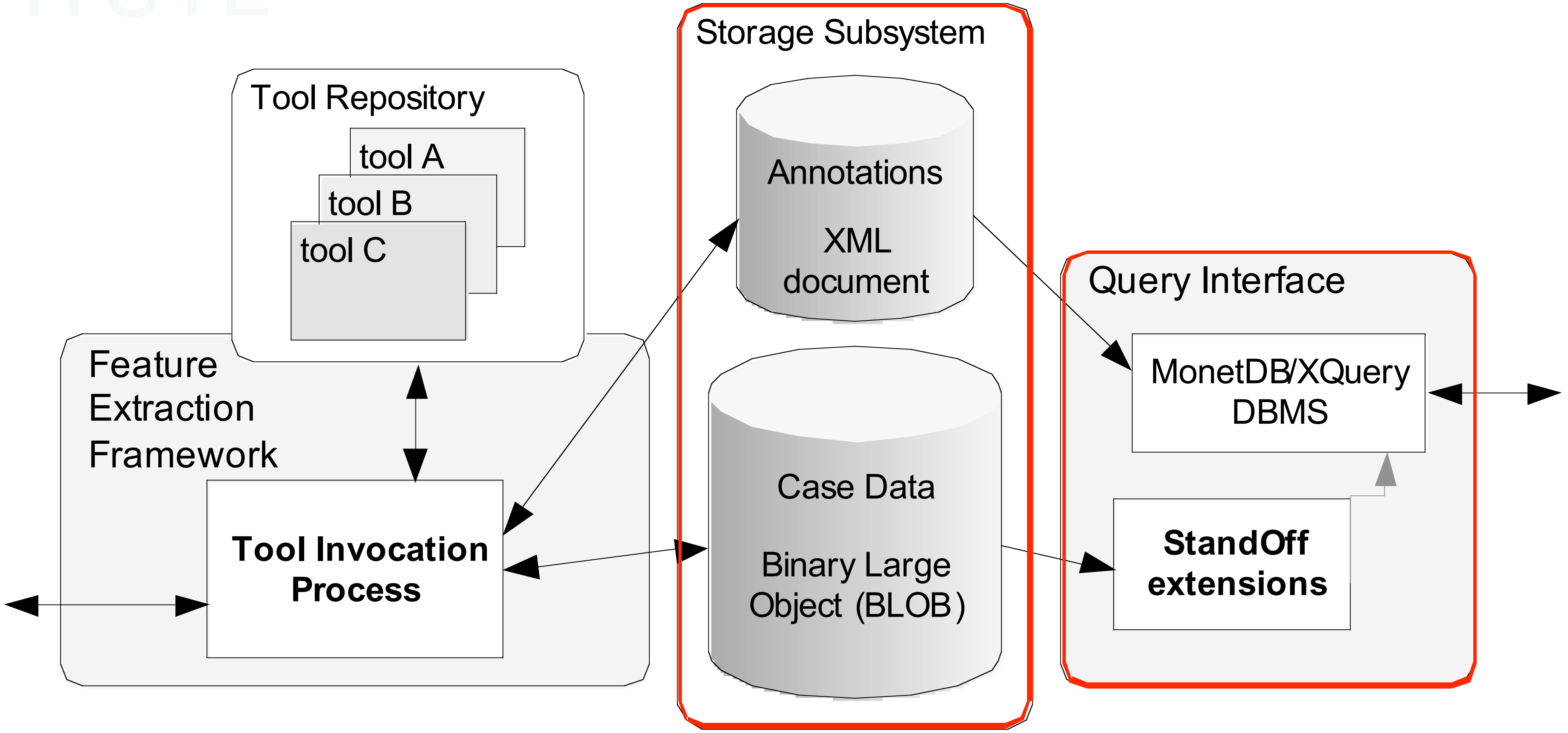
```
<case name="testcase">
  <image path="/casedata/HD-A.img" start="0" end="19999"/>
  <image path="/casedata/HD-B.img" start="20000" end="29999"/>
  <image path="/casedata/HD-C.e01" start="30000" end="59999"/>
</case>
...
<volume type="FAT" start="1000" end="19999"/>
<volume type="NTFS" start="35000" end="39999"/>
```

```
...
  <file name="Photo03.jpg" start="70000" end="74999">
    <size>5000</size>
    <mime>image/jpeg</mime>
    <modified><date>2006-08-15T09:10:00</date></modified>
  </file>
```

Storage subsystem

- Virtual BLOB mapping
  - evidence files
  - alternative representations
- Single XML document
  - extracted features
  - references to layout

XIRAF architecture



XQuery language

- Database language:
  - large XML documents
  - sorting/grouping/selecting/(updating)
- Example: timeline
  - different tools produce date-elements

```
for $i in doc(“case.xml”)//date
order by $i
where $i > $lowerbound
    and $i < $upperbound
return $i
```

Forensic application areas

- search for keywords, MD5s, URLs

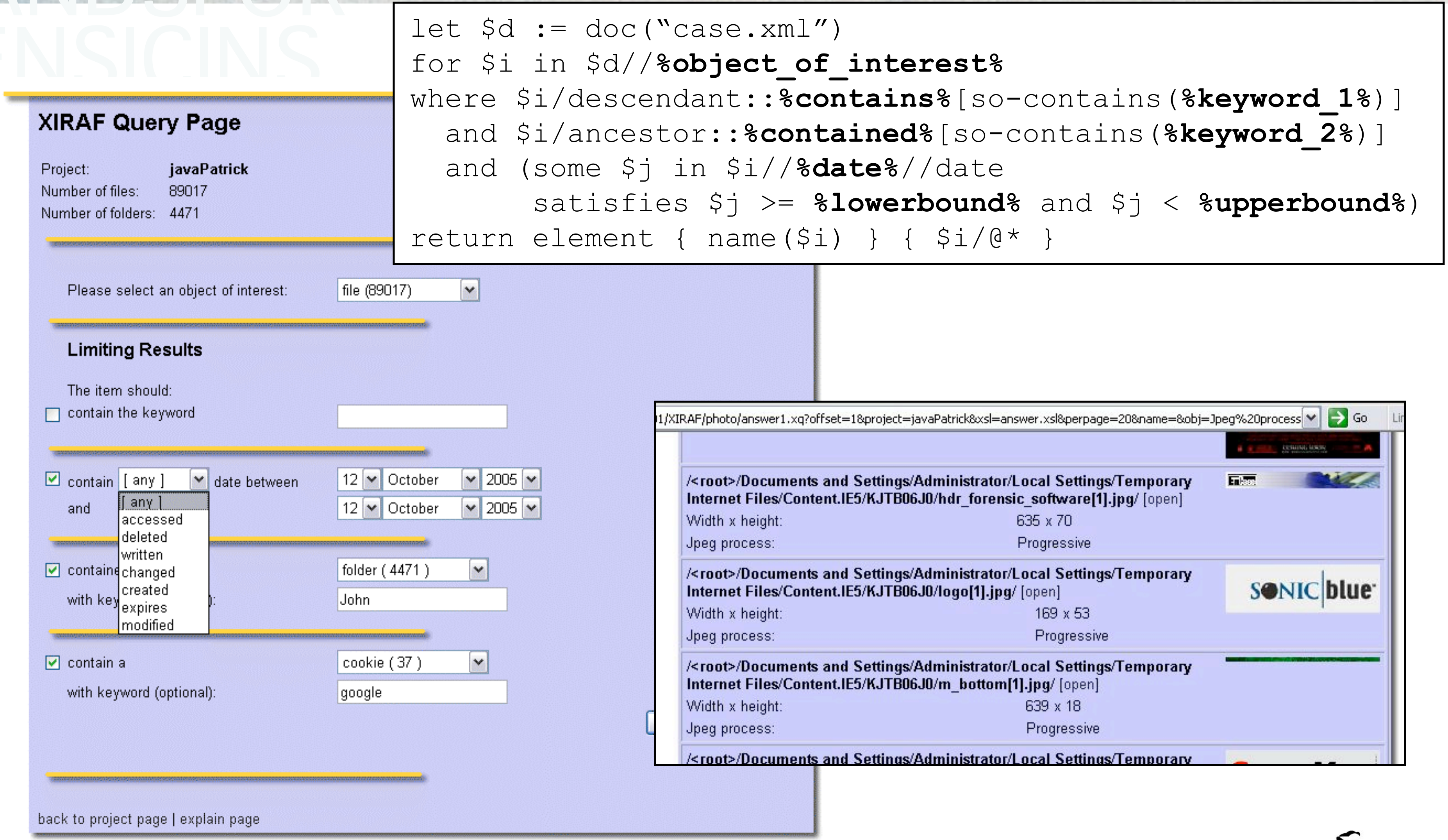
```
for $i in doc(“case.xml”)//file
for $j in doc(“CP-hashes.xml”)//md5
where $i/md5 = $j
return <file> { $i/@name } </file>
```

```
let $word_list :=
      doc(“terrorism-words.xml”)//word
for $i in doc(“case.xml”)//*
where some $i in $word_list
      satisfies blob-contains($i,$j)
return element { name($i) } { $i/@* }
```

Benefits

- Exploit exhaustive runs of tools
- Use knowledge from previous investigations
- Integrated data schema

- Added functionality:
  - XQuery extensions to relate XML to Virtual BLOB content



```
let $d := doc("case.xml")
for $i in $d//%object_of_interest%
where $i/descendant::%contains%[so-contains(%keyword_1%)]
  and $i/ancestor::%contained%[so-contains(%keyword_2%)]
  and (some $j in $i//%date%//date
       satisfies $j >= %lowerbound% and $j < %upperbound%)
return element { name($i) } { $i/@* }
```

XIRAF Query Page

Project: **javaPatrick**
Number of files: 89017
Number of folders: 4471

Please select an object of interest: file (89017)

Limiting Results

The item should:

☐ contain the keyword

☑ contain [any] date between 12 October 2005
and 12 October 2005

☑ containe folder (4471)
with key : John

☑ contain a cookie (37)
with keyword (optional): google

back to project page | explain page

1/XIRAF/photo/answer1.xq?offset=1&project=javaPatrick&xsl=answer.xsl&perpage=20&name=&obj=Jpeg%20process

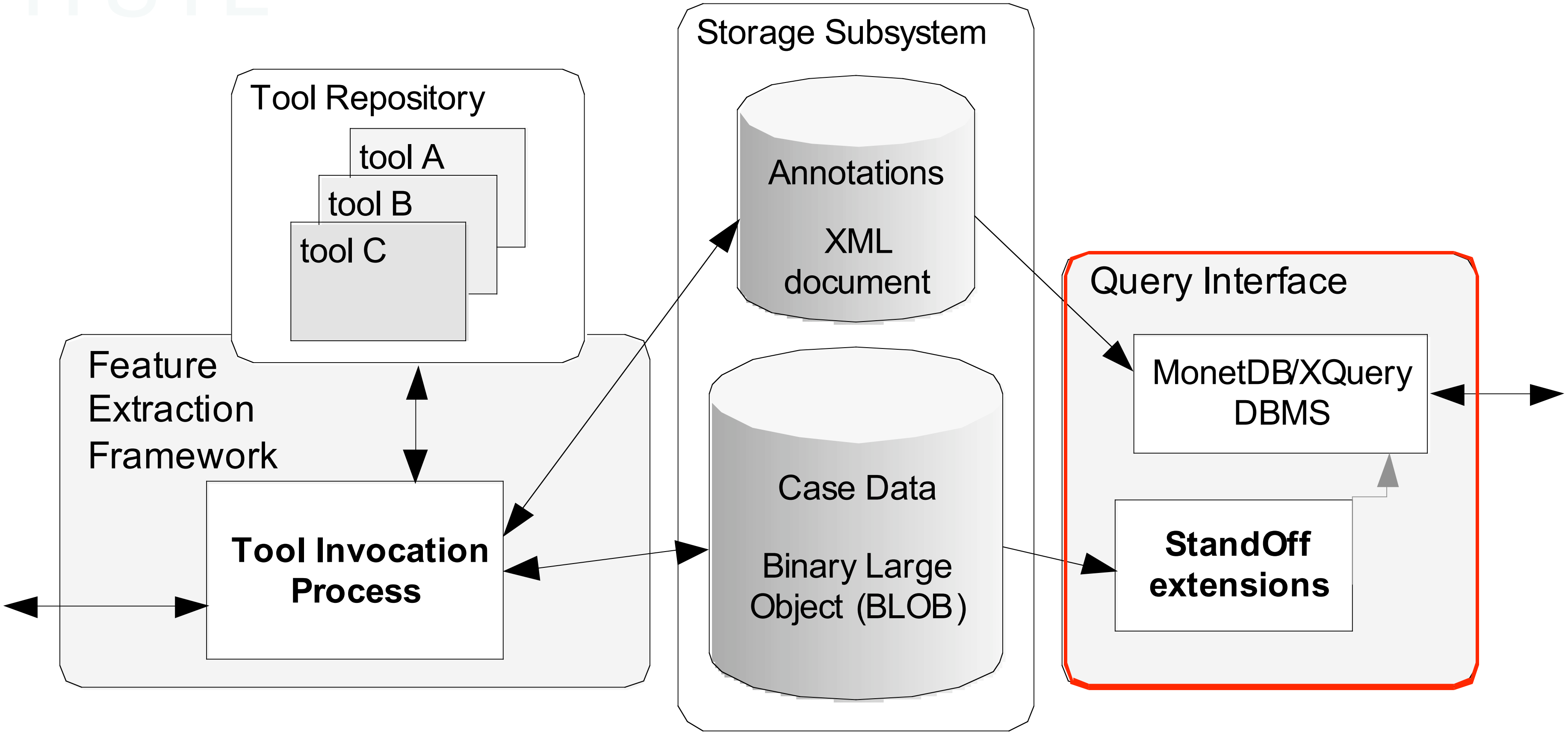
/<root>/Documents and Settings/Administrator/Local Settings/Temporary Internet Files/Content.IE5/KJTB06J0/hdr_forensic_software[1].jpg/ [open]
Width x height: 635 x 70
Jpeg process: Progressive

/<root>/Documents and Settings/Administrator/Local Settings/Temporary Internet Files/Content.IE5/KJTB06J0/logo[1].jpg/ [open]
Width x height: 169 x 53
Jpeg process: Progressive

/<root>/Documents and Settings/Administrator/Local Settings/Temporary Internet Files/Content.IE5/KJTB06J0/m_bottom[1].jpg/ [open]
Width x height: 639 x 18
Jpeg process: Progressive

/<root>/Documents and Settings/Administrator/Local Settings/Temporary

XIRAF architecture



Initial Experiments

- Evidence: 2 hard disks
  - (2 x 120GB)
- ~200MB XML
  - ~2.5M elements
- Recognized ~90000 files
  - file-systems / unallocated space
- ~500000 timestamps
  - file-system, registry, EXIF, .LNK, log-entry, cookie, etc

Conclusion

- Separation of feature extraction and analysis seems a viable approach
- Integrated querying of multiple tools becomes possible

Status & Future Work

- Prototype implementation (Java/Python)

- Make system production-ready
- More tools, query patterns
- Connect XIRAF to existing knowledge-bases

More information

- xiraf-info@holmes.nl
- http://www.forensischinstituut.nl/
- http://monetdb.cwi.nl/