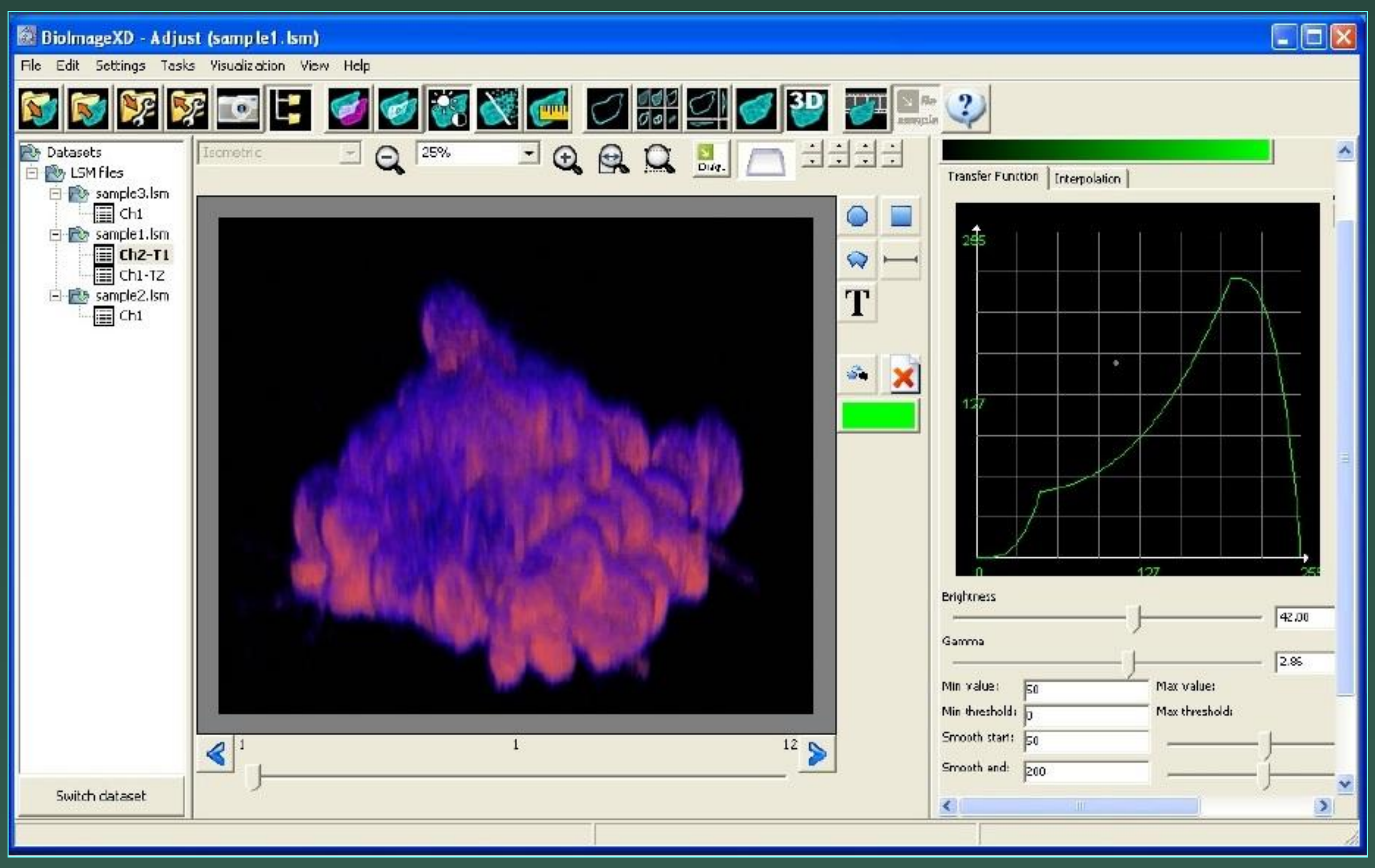
Programming Biomedical Smart Space Applications with *BioImageXD* and *PythonRules*

Natalia Díaz Rodríguez[1], Pasi Kankaanpää[2], M. Mohsin Saleemi[1], Johan Lilius[1], Iván Porres[1]
1.Department of IT, Åbo Akademi University, Turku, Finland.
2. Department of Biochemistry and Food Chemistry, University of Turku, Turku, Finland.
{ndiaz, msaleemi, jolilius, iporres}@abo.fi, pkanka@utu.fi

Motivation

In biomedical sciences, increasingly complex technologies are constantly being introduced, but the interoperability of devices, tools and data is not always simple. We propose a semantic approach and ***PythonRules*** as a way to increase functionality and programming productivity in heterogeneous spaces, through adaptable and scalable development of **Smart Space** applications.
BioImageXD software, for processing, analyzing and visualizing multidimensional biomedical images, is presented as a use case.

BioImageXD Software

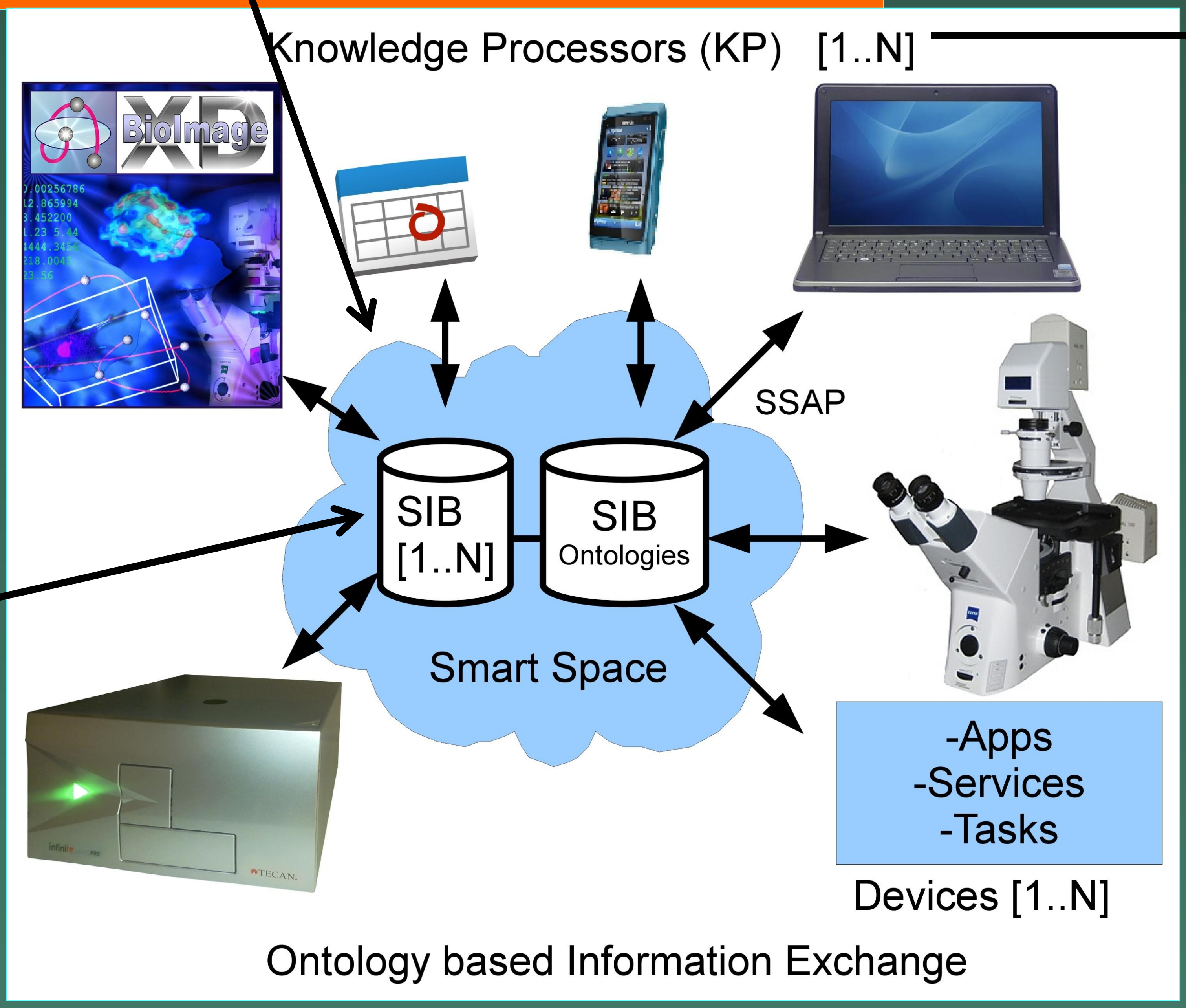


Methodology: Ontology based application development in Smart Spaces

SMART SPACE: An abstraction of space encapsulating both information from a physical space and access to this information allowing devices to join and leave the space.

NOKIA'S SMART-M3 (An implementation of Smart Space):

- A multi-part, multi-device and multi-vendor **open source cross-domain platform** for agents to communicate.
- **Semantic Information Broker (SIB):** RDF repository for information storage, sharing & management.



KPs implement functionality by inserting/retrieving/querying information.

1. ***Smart-M3* Ontology Library Code Generator** (OWL->Python/C API).
2. **Middleware** to abstract the communication with the SIB.
3. ***PythonRules*** for Smart Space programming to ease interaction (handling of namespaces, RDF, queries) and provide higher abstraction for **fast specification of the space's behavior**.

Knowledge Processor Programming with *PythonRules*

Python Rule Syntax:

With() // When() >> Then()

- **With** class handles Existence Assumptions in the Smart Space.
- **When** class handles Conditions.
- **Then** class handles Actions.

```
with = With([labWorkerCalendar, meetingEvent,
    bxdRun, meanFilter])
when = When(meetingEvent.getProperty("Title")
    == "Lab Weekly Meeting")
then = Then([bxdRun.new(BXDBatchAnalysis),
  bxdRun.setProperty(ProcessID = 86),
  bxdRun.setProperty(ProcessDate = today),
  bxdRun.setProperty(FileIN ="./sample.lsm"),
  bxdRun.setProperty(FileOUT ="./resultsFile"),
  bxdRun.setProperty(Timepoints= "1"),
  bxdRun.setProperty(Channels= "1"),
  bxdRun.setObject("AppliesFilter", meanFilter),
  bxdRun.runCommandLine(twoDaysBeforeMeeting)])
rule = PythonRule(with, when, then)
# Running the whole RuleSoup...
ruleSoup.addRule(rule)
ruleSoup.runAllRules()
```

Example

Example: ***PythonRules*** can be used to program, according to the researcher's calendar, when to start running time consuming or complex experiments, remotely. As alternative to the classical batch processing the ontology generated Python classes trade off the shortness of code for the command readability and understanding, aiding also at data sharing and process automation.

```
python BioImageXD.py -b -f Mean -o ./meanOutput
     -T 0 -c 0 -i ./sample.lsm
```

Classic *BioImageXD* Batch Processing

Conclusion: When the programmer does not deal with RDF directly, but mainly with Python statements, fast prototyping of mashup applications becomes easier. We introduce comparable functionality to SPARQL or SWRL, by "abstracting away" the technical details. ***PythonRules*** provides software power in a more comprehensive interface, in which learning OWL, RDF or query languages is not needed for taking advantage of the Semantic Web's benefits. The potential of the tool is demonstrated in a bioimaging scenario, in which batch processing of image filtering is remotely programmed to run according to the researcher's calendar.

ACKNOWLEDGMENT: This research work is funded through the ICT-SHOCK DIEM (Devices and Interoperability Ecosystem, www.diem.fi) project by TEKES (Finnish Funding Agency for Technology and Innovation). The authors thank Lassi Paavolainen, University of Jyväskylä, Finland, for his help with *BioImageXD* and TUCS (Turku Centre for Computer Science) Graduate School (www.tucs.fi).