Why semantics matter:
a demonstration on knowledge-based control system design

Wim Pessemier
ICALEPCS 2015
Melbourne
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

What are semantic models?
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations → meaning (semantics)
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations ➔ meaning (semantics)

Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations → meaning (semantics)

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations → meaning (semantics)
What are semantic models?

Models that describe
- pieces of information (data, descriptions)
- their relations → meaning (semantics)

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations meaning (semantics)
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations → meaning (semantics)
What are semantic models?

• Models that describe
  – pieces of information (data, descriptions)
  – their relations ➔ meaning (semantics)
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations → meaning (semantics)

Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations meaning (semantics)

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
What are semantic models?

• Models that describe
  – pieces of information (data, descriptions)
  – their relations \[\rightarrow\] meaning (semantics)

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
What are semantic models?

- Models that describe
  - pieces of information (data, descriptions)
  - their relations ➔ meaning (semantics)

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Where to apply them?
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Where to apply them?
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Where to apply them?
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

How to apply them?
How to apply them?

- Put them in a Knowledge Base and extract information!

Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

How to build them?
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

How to build them?

• Using an existing modeling language?
  - UML, SysML, ... : semantics not sufficiently formal
  - Modeling languages have no “programming” capabilities (loops, functions, if-then, ...)

WEB3O05
How to build them?

• Using an existing modeling language?
  – UML, SysML, … : semantics not sufficiently formal
  – Modeling languages have no “programming” capabilities (loops, functions, if-then, …)

• Using a Domain Specific Language (DSL)?
  – Internal DSL called Ontoscript
  – Based on coffeescript (~javascript)
  – Idea “adopted” from the Giant Magellan Telescope project [1]

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

How to build them?

- Example: model of an I/O module type

```
Model: http://mercator.iac.es/onto/models/external/beckhoff
```

```
220 id: "EL808B"
221 comment: "8-channel digital input terminal 24V DC, negative switching"
222 manufacturer: beckhoff.company

223 terminals:
224 1 -> symbol "11", comment: "Input 1"
225 2 -> symbol "12", comment: "Input 2"
226 3 -> symbol "13", comment: "Input 3"
227 4 -> symbol "14", comment: "Input 4"
228 5 -> symbol "15", comment: "Input 5"
229 6 -> symbol "16", comment: "Input 6"
230 7 -> symbol "17", comment: "Input 7"
231 8 -> symbol "18", comment: "Input 8"

232 channels:
233 234 1 -> terminals | terminals("1")
235 2 -> terminals | terminals("2")
236 3 -> terminals | terminals("3")
237 4 -> terminals | terminals("4")
238 5 -> terminals | terminals("5")
239 6 -> terminals | terminals("6")
240 7 -> terminals | terminals("7")
241 8 -> terminals | terminals("8")

242 soft_interface:
243 input1 <- { type 1, bool, comment: "Input 1" }
244 input2 <- { type 1, bool, comment: "Input 2" }
245 input3 <- { type 1, bool, comment: "Input 3" }
246 input4 <- { type 1, bool, comment: "Input 4" }
247 input5 <- { type 1, bool, comment: "Input 5" }
248 input6 <- { type 1, bool, comment: "Input 6" }
249 input7 <- { type 1, bool, comment: "Input 7" }
250 input8 <- { type 1, bool, comment: "Input 8" }

251 McState <- { type 1, bool, comment: "EtherCAT working counter state" }
252 McStateDelta <- { type 1, bool, comment: "EtherCAT state (NET, PDOP, UP, ..." }
```
Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

How to build them?

• Example: model of an I/O module type
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

• Example: model of an I/O module type

How to build them?
Example: model of an I/O module instance
Example: model of an I/O module instance
How to build them?

- Example: model of an I/O module instance
How to use them?
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Electrical design

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Electrical design
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Electrical design

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Electrical design

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Electrical design

Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Electrical design

I/O Module instance slot3

Digital input terminal to read the status of the SSI encoders of all 8 cover panels

System properties

Satisfies cover_sys:panelDesign.requirements.absFeedbackStatus

Module type summary

<table>
<thead>
<tr>
<th>ID</th>
<th>ELIO08</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>Beckhoff Automation</td>
</tr>
<tr>
<td>Description</td>
<td>8-channel digital input terminal 24V DC, negative switching</td>
</tr>
<tr>
<td>Used in</td>
<td>Cover (1), M3 (1)</td>
</tr>
</tbody>
</table>

[Diagram of I/O module with inputs and outputs]
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

---

Systems design

- Requirement absFeedbackStatus
  - The status of the absolute feedback shall be known

- Properties
  - Derived from: cover.sys:concept.requirements.monitorable
    - cover.sys:panelDesign.requirements.absFeedback
  - Satisfied by: cover.sys:panelDesign.parts.encoder
    - cover.elem:slot3
  - Declared by: cover.sys:panelDesign
Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Systems design

OntoManager @ Mercator Telescope

Design `panelDesign`
The design of the telescope cover panels

Requirements derivation matrix

<table>
<thead>
<tr>
<th></th>
<th>open</th>
<th>closed</th>
<th>moveActuator</th>
<th>moveActuatorStatus</th>
<th>absFeedback</th>
<th>absFeedbackStatus</th>
<th>locking</th>
<th>aluminum</th>
<th>obstructing</th>
<th>sealing</th>
<th>controlable</th>
<th>movableable</th>
<th>reliability</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>panelDesign</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>open</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>closed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moveActuator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moveActuatorStatus</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>absFeedback</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>absFeedbackStatus</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The status of the absolute feedback shall be known

aluminum
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Systems design

Design panelDesign

The design of the telescope cover panels

Requirements derivation matrix

<table>
<thead>
<tr>
<th></th>
<th>open</th>
<th>closed</th>
<th>moveActuator</th>
<th>moveActuatorStatus</th>
<th>absFeedback</th>
<th>absFeedbackStatus</th>
<th>locking</th>
<th>aluminum</th>
<th>obstructing</th>
<th>sealing</th>
<th>controllable</th>
<th>monitorable</th>
<th>reliability</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>open</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>closed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moveActuator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>moveActuatorStatus</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>absFeedback</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>absFeedbackStatus</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The status of absolute feedback shall be known

Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Electrical design

OntoManager @ Mercator Telescope

I/O Module instance slot3

Digital input terminal to read the status of the SSI encoders of all 8 cover panels

System properties

Satisfies cover_syspanelDesign.requirements.absFeedbackStatus

Module type summary

<table>
<thead>
<tr>
<th>ID</th>
<th>EL1088</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>Beckhoff Automation</td>
</tr>
<tr>
<td>Description</td>
<td>8-channel digital input terminal 24V DC, negative switching</td>
</tr>
<tr>
<td>Used in</td>
<td>Cover (1), M3 (1)</td>
</tr>
</tbody>
</table>

[Diagram of I/O Module instance slot3]
Electrical design

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

OntoManager @ Mercator Telescope

I/O Module instance slot3

Digital input terminal to read the status of the SSI encoders of all 8 cover panels

System properties
Satisfies cover_sys:panelDesign.requirements.absFeedbackStatus

Module type summary

<table>
<thead>
<tr>
<th>ID</th>
<th>EL1088</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>Beckhoff Automation</td>
</tr>
<tr>
<td>Description</td>
<td>8-channel digital input terminal 24V DC, negative switching</td>
</tr>
<tr>
<td>Used in</td>
<td>Cover (1), M3 (1)</td>
</tr>
</tbody>
</table>

Signals:
- LED1
- LED2
- LED3
- LED4
- LED5
- LED6
- LED7
- LED8

Inputs:
- Input 1
- Input 2
- Input 3
- Input 4

Power contact +24V
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Company Beckhoff

Logo

BECKHOFF

Summary

| Short name | Beckhoff |
| Long name | Beckhoff Automation |
| Description | Produces IPCs, PLCs, I/O, control panels, ... |

Products

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td>BK1101</td>
<td>EtherCAT Coupler with 10 switch</td>
<td>Cover (1), M1 (1), M3 (1), Telemetry (1), Timing (1)</td>
</tr>
<tr>
<td>EL1088</td>
<td>8-channel digital input terminal 24V DC, negative switching</td>
<td>Cover (1), M3 (1)</td>
</tr>
<tr>
<td>EL2008</td>
<td>8-channel digital output terminal 24V DC</td>
<td>Cover (1)</td>
</tr>
<tr>
<td>EL2024</td>
<td>4-channel digital output terminals 24 V DC, 2 A</td>
<td>M1 (1)</td>
</tr>
<tr>
<td>EL2124</td>
<td>4-channel digital output terminals 5 V DC</td>
<td>M1 (2)</td>
</tr>
<tr>
<td>EL2622</td>
<td>2-channel relay</td>
<td>Cover (5), M1 (1)</td>
</tr>
<tr>
<td>EL3024</td>
<td>4-channel analog input terminals 4...20mA, differential inputs, 12 bit</td>
<td>M1 (2), Telemetry (1)</td>
</tr>
</tbody>
</table>
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

## Electrical design

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR1101</td>
<td>EtherCAT Coupler with ID switch</td>
<td>Cover (1), M1 (1), M3 (1), Telemetry (1), Timing (1)</td>
</tr>
<tr>
<td>EL1088</td>
<td>8-channel digital input terminal 24V DC, negative switching</td>
<td>Cover (1), M1 (1)</td>
</tr>
<tr>
<td>EL2098</td>
<td>8-channel digital output terminal 24V DC</td>
<td>Cover (1)</td>
</tr>
<tr>
<td>EL2924</td>
<td>4-channel digital output terminals 24 V DC, 2 A</td>
<td>M1 (2)</td>
</tr>
<tr>
<td>EL2124</td>
<td>4-channel digital output terminals 5 V</td>
<td>M1 (2)</td>
</tr>
<tr>
<td>EL2622</td>
<td>2-channel relay</td>
<td>Cover (5), M1 (1)</td>
</tr>
<tr>
<td>EL3021</td>
<td>4-channel analog input terminals 4...20mA, differential inputs, 12 bit</td>
<td>M1 (2), Telemetry (1)</td>
</tr>
<tr>
<td>EL3102</td>
<td>2-channel analog input terminals -10...+10 V, differential input, 10 bit</td>
<td>M1 (2)</td>
</tr>
<tr>
<td>EL3164</td>
<td>4-channel analog input terminal 0...10 V, single-ended,16 bit</td>
<td>M1 (1)</td>
</tr>
<tr>
<td>EL3202-0010</td>
<td>2-channel input terminals PT100 (RTD) for 4-wire connection, high-precision</td>
<td>Telemetry (? )</td>
</tr>
<tr>
<td>EL3551</td>
<td>1-channel resistor bridge terminal (strain gauge)</td>
<td>M1 (3)</td>
</tr>
<tr>
<td>EL3581</td>
<td>Digital multimeter</td>
<td>Cover (1)</td>
</tr>
<tr>
<td>EL4008</td>
<td>8-channel analog output terminal 0...30V, 12 bit</td>
<td>Cover (1)</td>
</tr>
<tr>
<td>EL4922</td>
<td>2-channel analog output terminal 4...20mA, 12 bit</td>
<td>M1 (1)</td>
</tr>
<tr>
<td>EL5001</td>
<td>1-channel SSI encoder</td>
<td>M1 (1), M3 (1)</td>
</tr>
<tr>
<td>EL5002</td>
<td>2-channel SSI encoder</td>
<td>Cover (4)</td>
</tr>
<tr>
<td>EL5101</td>
<td>1-channel incremental encoder</td>
<td>M1 (1), M3 (1)</td>
</tr>
<tr>
<td>EL6001</td>
<td>RS-232 serial communication</td>
<td>Timing (1)</td>
</tr>
<tr>
<td>EL6608</td>
<td>IEEE 1588 external synchronisation interface</td>
<td>Timing (1)</td>
</tr>
<tr>
<td>EL6751</td>
<td>CANopen master/slave controller</td>
<td>M3 (1)</td>
</tr>
<tr>
<td>EL9070</td>
<td>Shield terminal</td>
<td>Telemetry (2)</td>
</tr>
<tr>
<td>EL9186</td>
<td>Potential distribution terminal, 8 x 24V</td>
<td>M1 (1), Telemetry (1)</td>
</tr>
<tr>
<td>EL9187</td>
<td>Potential distribution terminal, 8 x 0V</td>
<td>M1 (2), Telemetry (1)</td>
</tr>
<tr>
<td>EL9410</td>
<td>Power supply terminals for E-bus (with diagnostics)</td>
<td>M1 (2)</td>
</tr>
<tr>
<td>EL2509</td>
<td>Power supply terminals 5 V</td>
<td>M1 (2)</td>
</tr>
</tbody>
</table>
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?

Conclusions
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Electrical design

Connections

<table>
<thead>
<tr>
<th>Type (EI188)</th>
<th>Instance</th>
<th>Description</th>
<th>Connected to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td>Terminal</td>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>I1</td>
<td>Top 1 SSI encoder status</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>I2</td>
<td>Top 2 SSI encoder status</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>I3</td>
<td>Top 3 SSI encoder status</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>I4</td>
<td>Top 4 SSI encoder status</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>I5</td>
<td>Bottom SSI encoder status</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>I6</td>
<td>Bottom SSI encoder status</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>I7</td>
<td>Bottom SSI encoder status</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>I8</td>
<td>Bottom SSI encoder status</td>
</tr>
</tbody>
</table>

Interface

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
<th>Linked variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>input1</td>
<td>BOOL</td>
<td>Input 1</td>
<td>interface.parts.cover.parts.top.parts.p1.encoderErrorSignal</td>
</tr>
<tr>
<td>input2</td>
<td>BOOL</td>
<td>Input 2</td>
<td>interface.parts.cover.parts.top.parts.p1.encoderErrorSignal</td>
</tr>
<tr>
<td>input3</td>
<td>BOOL</td>
<td>Input 3</td>
<td>interface.parts.cover.parts.top.parts.p3.encoderErrorSignal</td>
</tr>
<tr>
<td>input4</td>
<td>BOOL</td>
<td>Input 4</td>
<td>interface.parts.cover.parts.top.parts.p4.encoderErrorSignal</td>
</tr>
<tr>
<td>input5</td>
<td>BOOL</td>
<td>Input 5</td>
<td>interface.parts.cover.parts.bottom.parts.p1.encoderErrorSignal</td>
</tr>
<tr>
<td>input6</td>
<td>BOOL</td>
<td>Input 6</td>
<td>interface.parts.cover.parts.bottom.parts.p2.encoderErrorSignal</td>
</tr>
<tr>
<td>input7</td>
<td>BOOL</td>
<td>Input 7</td>
<td>interface.parts.cover.parts.bottom.parts.p3.encoderErrorSignal</td>
</tr>
<tr>
<td>input8</td>
<td>BOOL</td>
<td>Input 8</td>
<td>interface.parts.cover.parts.bottom.parts.p4.encoderErrorSignal</td>
</tr>
<tr>
<td>WcState</td>
<td>BOOL</td>
<td>EtherCAT Working counter state</td>
<td>interface.parts.cover.parts.io.parts.slot3.wcState</td>
</tr>
<tr>
<td>InfoDataState</td>
<td>UINT</td>
<td>EtherCAT state (INIT, PREOEP, OP, ...)</td>
<td>interface.parts.cover.parts.io.parts.slot3.infoData</td>
</tr>
</tbody>
</table>
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

FunctionBlock **SM_CoverPanel**

- encoderErrorSignal
- initializationStatus
- operatorStatus
- operatingStatus
- config
- coverConfig

Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Name</th>
<th>Type</th>
<th>Initial value</th>
<th>Address</th>
<th>Description</th>
<th>Qualif</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAR_INPUT</td>
<td>encoderErrorSignal</td>
<td>BOOL</td>
<td></td>
<td>w1</td>
<td>Externally read error signal</td>
<td>OPC,UA:DA=1, DPC</td>
</tr>
<tr>
<td>VAR_IN_OUT</td>
<td>initializationStatus</td>
<td>InitializationStatus</td>
<td></td>
<td></td>
<td>INITIALIZED or INITIALIZING or ...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>operatorStatus</td>
<td>OperatorStatus</td>
<td></td>
<td></td>
<td>TECH or OBSERVER or ...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>operatingStatus</td>
<td>OperatingStatus</td>
<td></td>
<td></td>
<td>MANUAL or AUTO or NONE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>config</td>
<td>CoverPanelConfig</td>
<td></td>
<td></td>
<td>Configuration of the panel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>coverConfig</td>
<td>CoverConfig</td>
<td></td>
<td></td>
<td>Configuration of the cover</td>
<td></td>
</tr>
<tr>
<td>VAR_OUTPUT</td>
<td>actualStatus</td>
<td>STRING</td>
<td></td>
<td></td>
<td>Current status description</td>
<td>OPC,UA:DA=1, DPC</td>
</tr>
</tbody>
</table>
Software design

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

FunctionBlock SM_CoverPanel

Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Name</th>
<th>Type</th>
<th>Initial value</th>
<th>Address</th>
<th>Description</th>
<th>Qualif</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAR_INPUT</td>
<td>encoderErrorSignal</td>
<td>BOOL</td>
<td></td>
<td>%M1*</td>
<td>Externally read error signal</td>
<td>OPC UA DA=1, DPC</td>
</tr>
<tr>
<td>VAR_IN_OUT</td>
<td>initializationStatus</td>
<td>InitializationStatus</td>
<td></td>
<td></td>
<td>INITIALIZE or INITIALIZING or ...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>operatorStatus</td>
<td>OperatorStatus</td>
<td></td>
<td></td>
<td>TECH or OBSERVER or ...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>operatingStatus</td>
<td>OperatingStatus</td>
<td></td>
<td></td>
<td>MANUAL or AUTO or NONE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>config</td>
<td>CoverPanelConfig</td>
<td></td>
<td></td>
<td>Configuration of the panel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>coverConfig</td>
<td>CoverConfig</td>
<td></td>
<td></td>
<td>Configuration of the cover</td>
<td></td>
</tr>
<tr>
<td>VAR_OUTPUT</td>
<td>actualStatus</td>
<td>STRING</td>
<td></td>
<td></td>
<td>Current status description</td>
<td>OPC UA DA=1, DPC</td>
</tr>
<tr>
<td></td>
<td>statuses</td>
<td>CoverPanelStatuses</td>
<td></td>
<td></td>
<td>Statuses of the state machine</td>
<td></td>
</tr>
<tr>
<td></td>
<td>parts</td>
<td>CoverPanelParts</td>
<td></td>
<td></td>
<td>Parts of the state machine</td>
<td></td>
</tr>
<tr>
<td></td>
<td>processes</td>
<td>CoverPanelProcesses</td>
<td></td>
<td></td>
<td>Processes of the state machine</td>
<td></td>
</tr>
</tbody>
</table>
Software design

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

| VAR_INPUT | encoderErrorSignal | BOOL | %1* | Externally read error signal | OPC-UA.DA=1, OPC-UA.DA=2 |
| VAR_IN_OUT | initializationStatus | InitializationStatus | | INITIALIZED or INITIALIZING or... |
|           | operatorStatus | OperatorStatus | | TECH or OBSERVER or... |
|           | operatingStatus | OperatingStatus | | MANUAL or AUTO or NONE |
|           | config | CoverPanelConfig | | Configuration of the panel |
|           | coverConfig | CoverConfig | | Configuration of the cover |

| VAR_OUTPUT | actualStatus | STRING | Current status description | OPC-UA.DA=1, OPC-UA.DA=2 |
| statusSet | CoverPanelStatuses | | Statuses of the state machine |
| parts | CoverPanelParts | | Parts of the state machine |
| processes | CoverPanelProcesses | | Processes of the state machine |

Methods
• **startOpening()**

| Comment | Start opening the panel |
| Return type | RequestResults |
| Interface | Variable | Name | Type | Initial value | Address | Description | Qualifiers |
| Implementation | ```startOpening := THIS*.processes.startOpening.request();``` |

• **startClosing()**

| Comment | Start closing the panel |
| Return type | RequestResults |
| Interface | Variable | Name | Type | Initial value | Address | Description | Qualifiers |
| Implementation | ```startClosing := THIS*.processes.startClosing.request();``` |
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?

Conclusions
Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Software design

OntoManager @ Mercator Telescope

Library mtcs_cover

PLCopen XML serialization

<table>
<thead>
<tr>
<th>File</th>
<th>/home/wimpe/work/onts/ontomanager/env/ontomanager/generated/mtcs_cover.xml</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>File has been read</td>
</tr>
<tr>
<td>Code generation</td>
<td>Not running</td>
</tr>
</tbody>
</table>

Generate PLCopen XML  Download PLCopen XML

PyUAF serialization

<table>
<thead>
<tr>
<th>File</th>
<th>/home/wimpe/work/onts/ontomanager/env/ontomanager/generated/pyuaf/mtcs_cover.py</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>File has been read</td>
</tr>
<tr>
<td>Code generation</td>
<td>Not running</td>
</tr>
</tbody>
</table>

Generate pyUAF code  Download pyUAF code

File contents:

Show PLCopen XML code  Show PyUAF code

1 2 3 4 5 6 7
<xml version="1.0" encoding="utf-8">,
<wrapper name="http://www.plcopen.org/xml/mtcs-0300">,
<file Header company="Institute of Astronomy" product name="OntoManager" product Version="0.0.1" creation Date-Time="2015-10-09T12:23:49.239920">,
<context Header name="mtcs_cover" modification Date-Time="2015-10-09T12:23:49.239920">,
<coordinate info>,
<fbd>,
<scaling x="1" y="1" />>
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Software design

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Software design

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Software design

Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Software design

OntoManager @ Mercator Telescope

Library mtcs_cover

PLCopen XML serialization

<table>
<thead>
<tr>
<th>File</th>
<th>/home/wimpe/work/onts/ontomgr/env/ontomgr/generation/mtcs_cover.xml</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>File has been read</td>
</tr>
<tr>
<td>Code generation</td>
<td>Not running</td>
</tr>
</tbody>
</table>

Generate PLCopen XML  Download PLCopen XML

PyUAF serialization

<table>
<thead>
<tr>
<th>File</th>
<th>/home/wimpe/work/onts/ontomgr/env/ontomgr/generation/pyuaf/mtcs_cover.py</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>File has been read</td>
</tr>
<tr>
<td>Code generation</td>
<td>Not running</td>
</tr>
</tbody>
</table>

Generate pyUAF code  Download pyUAF code

File contents:

```xml
<xml version="1.0" encoding="utf-8">...
```

```python
from pyuaf import *
...
```
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Software design

- **Generated Python code (client side)**
  - Based on our OPC UA library “UAF”: [http://github.com/uaf/uaf](http://github.com/uaf/uaf)

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Software design

• Generated Python code (client side)
  – Based on our OPC UA library “UAF”: http://github.com/uaf/uaf

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Software design

• Generated Python code (client side)
  – Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: [http://github.com/uaf/uaf](http://github.com/uaf/uaf)

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: http://github.com/uaf/uaf

```python
>>> import opcua
>>> c = opcua.buildClient()
>>> print c.read(opcua)
```

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Software design

• Generated Python code (client side)
  – Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Software design

• Generated Python code (client side)
  – Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: [http://github.com/uaf/uaf](http://github.com/uaf/uaf)

```python
>>> import opcua
>>> c = opcua.buildClient()

print c.read ('opcua.MTCS.parts.ml')
```
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: [http://github.com/uaf/uaf](http://github.com/uaf/uaf)
Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: [http://github.com/uaf/uaf](http://github.com/uaf/uaf)
Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: http://github.com/uaf/uaf
Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Software design

• Generated Python code (client side)
  – Based on our OPC UA library “UAF”: http://github.com/uaf/uaf

```python
>>> import opcua
>>> c = opcua.buildClient()
>>> print c.read( opcua.MTS.CS.parts.m1.parts.axialSupport.regulatorPressure.average.bar.value.ADR() )
  - overallStatus  : Good
  - requestHandle : 1
  - targets[]
    - targets[0]
      - clientConnectionId : 0
      - status : Good
      - opcUAStatusCode  : 0
      - data : 1.10534573145
      - sourceTimestamp : 2015-10-13T11:33:01.825Z
      - serverTimestamp : 2015-10-13T11:33:01.825Z
      - sourcePicoseconds : 0
      - serverPicoseconds : 0
```
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

Software design

- Generated Python code (client side)
  - Based on our OPC UA library “UAF”: [http://github.com/uaf/uaf](http://github.com/uaf/uaf)
Why semantics matter?

- Currently in operation:
  - 1 PLC
  - 5 subsystems
  - 55 I/O modules
  - 159 PLC Function Block definitions (626 instances)
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

User Interface (HMI) running on the PLC

Results
Why semantics matter?

- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions

User Interface (HMI) running on the PLC
Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Results

• User Interface (HMI) running on the PLC
Why semantics matter?

• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions

Conclusions
So, why semantics matter?

Why semantics matter?
• What are semantic models?
• Where to apply them?
• How to apply them?
• How to build them?
• How to use them?
• Conclusions
So, why semantics matter?

1. Because every piece of information is just one query “away”

➔ organize, integrate, browse, find (query) information
So, why semantics matter?

1. Because every piece of information is just one query “away”
   - organize, integrate, browse, find (query) information

2. Because well defined semantics allow model verification
   - verify information

Why semantics matter?
- What are semantic models?
- Where to apply them?
- How to apply them?
- How to build them?
- How to use them?
- Conclusions
So, why semantics matter?

1. Because every piece of information is just one query “away”
   - organize, integrate, browse, find (query) information

2. Because well defined semantics allow model verification
   - verify information

3. Because they’re a key enabling technology for future “smart” systems
   - share information
Thanks!

Any questions?

wim.pessemier@ster.kuleuven.be