unibz SIEMENS DEPARTMENT OF COMPUTER SCIENCE UNIVERSITY OF OXFORD

Diagnostics of Trains with Semantic Diagnostics Rules

Ognjen Savković
Free University of Bozen-Bolzano

Evgeny Kharlamov, Martin Ringsquandl, Guohui Xiao, Gulnar Mehdi, Elem Guzel Kalayc, Werner Nutt, Mikhail Roshchin and Ian Horrocks

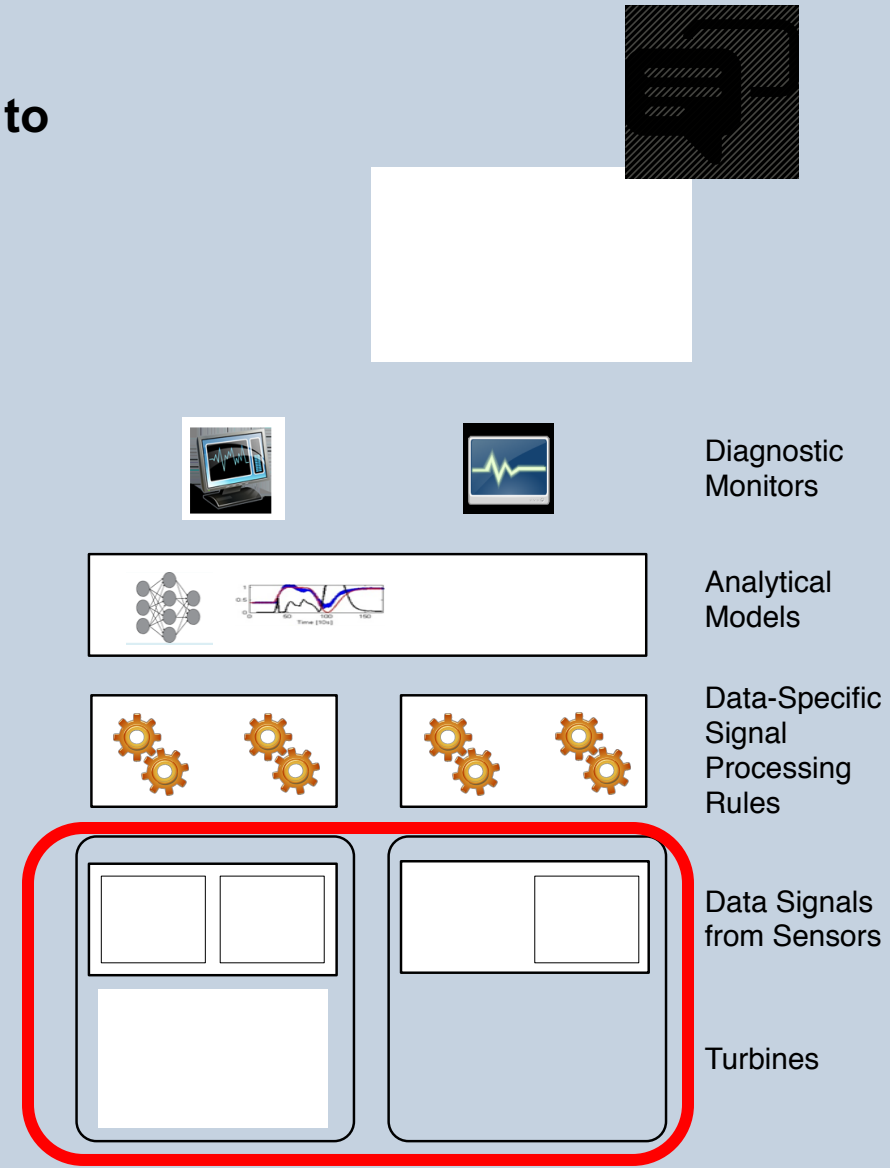
Turbine Diagnostics

Diagnostics of equipment is needed to

- maximise its up-time
- minimise maintenance costs

Diagnostic method

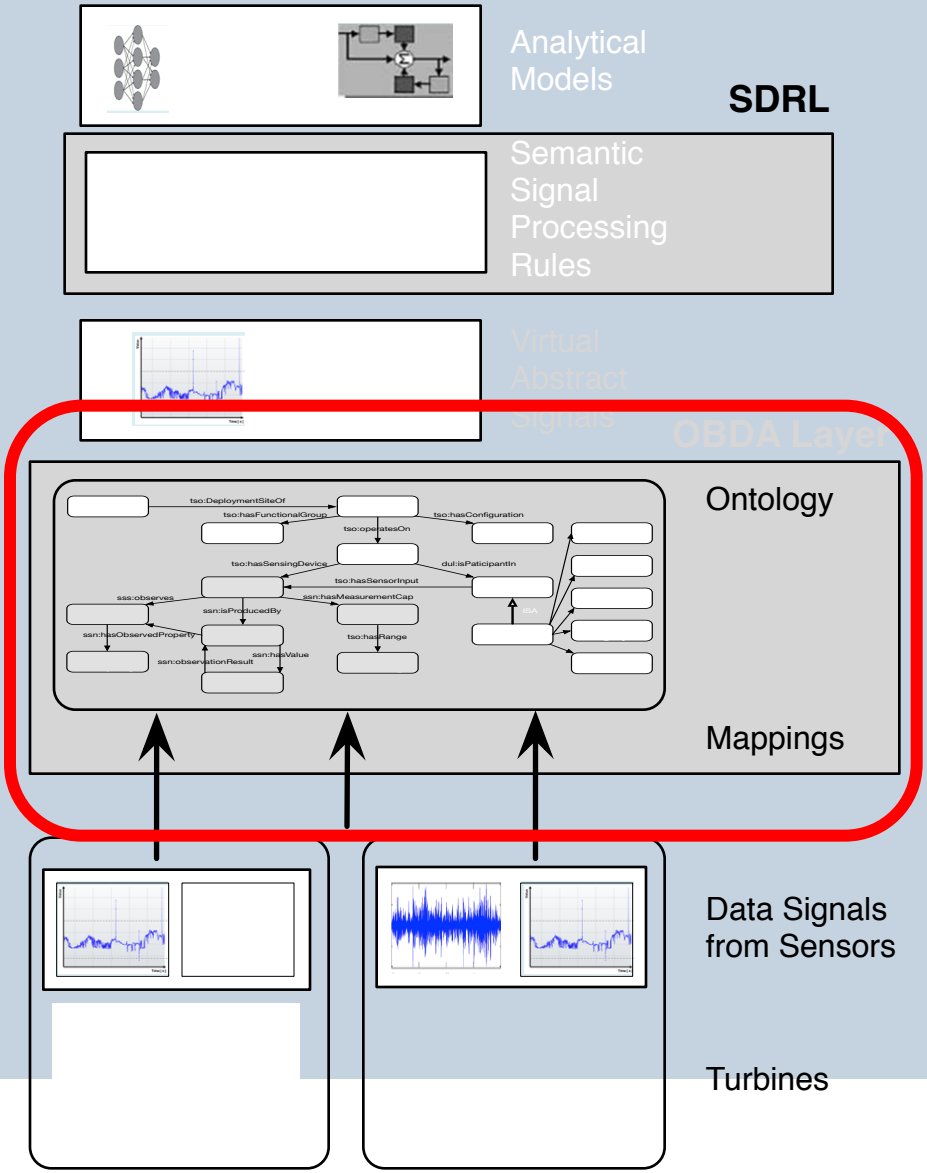
- turbines generate data
  - a turbine has about 2,000 sensors
  - sensors report temp, pressure. Etc
  - data ~“continuous” signals
- diagnostic programs
  - detect patterns in data
  - fire alarm messages
- analytical models
  - process data and messages



Our Proposal: to Fight Data Dependency

To rely on ontologies (OBDA)

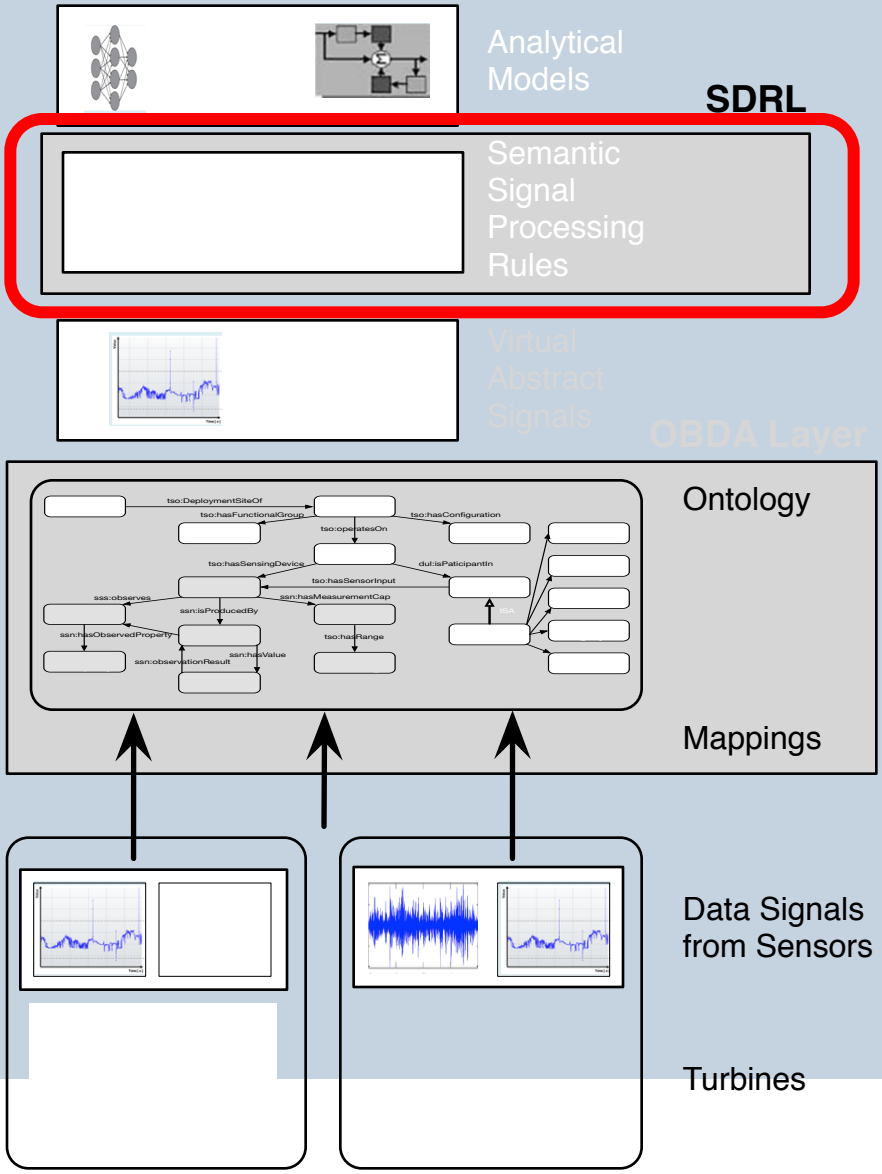
- ontologies
  - offer generic diagnostic vocabulary
  - capture background knowledge
- mappings
  - connect the vocabulary to data signals



Our Proposal: to Fight Data Dependency

A novel Semantic Diagnostic Rule-based Language (SDRL)

- to write SPRs over ontologies
- OBDA compatible
  - **FO-rewritability**



SDRL: Semantic Diagnostic Rule-based Language

Balance

- Expressiveness
  - capture the main features of the Siemens diagnostic language
- Efficiency
  - allows for FO-rewritability over OWL 2 QL ontologies
  - efficiency was verified: formally and with extensive evaluation
- (Re) Usability
  - simple and concise: engineers can significantly save time on diagnostic tasks
  - diagnostic programs can be easily reused across turbines
  - usability was evaluated with Siemens engineers

$$\mathsf{message}(\textit{“Purging over”}) = \mathsf{FlameSensor} : \mathsf{duration}(>, 10s) : \textit{after}[15s]\ \mathsf{PurgingStart} : \textit{after}[20s]\ \mathsf{PurgingStop}$$

$$\mathsf{PurgingStart} = \textit{avg}\ \mathsf{rotorStart} : \mathsf{value}(>, \textit{purgingSpeed}),$$

$$\mathsf{PurgingStop} = \textit{avg}\ \mathsf{rotorStart} : \mathsf{value}(<, \textit{nonPurgingSpeed}).$$

SDRL: Overview

Message Rules

$$message(m) = C.$$

$$\text{PurgingStart} = avg\ \text{rotorStart} : \text{value}(>, purgingSpeed),$$
$$\text{PurgingStop} = avg\ \text{rotorStart} : \text{value}(<, nonPurgingSpeed).$$

Complex Signal Expressions

$$\begin{array}{lllll} C & = & Q & | & \{s_1, \ldots, s_m\} \\ & & \alpha \circ C & | & C_1 : value(\odot, \alpha) \\ & & \text{agg } C_1 & | & C_1 : duration(\odot, t) \\ & & C_1 : \ align\ C_2 & | & C_1 : trend(direction). \end{array}$$

$$\text{agg} \in \{\text{min}, \text{max}, \text{avg}, \text{sum}\},$$
$$\circ \in \{+, -, \times, /\},$$
$$\odot \in \{<, >, \leq, \geq\},$$
$$align \in \{within, after[t], before[t]\},$$
$$direction \in \{\text{up}, \text{down}\}.$$

Meaning

- Message rules fire messages
  - if the class of (complex) signals C is not empty
- Complex expressions
  - define classes C of signals
  - By processing basic / raw sensor signals $s_1, \ldots, s_m$

FO-rewritability by encoding into metric Datalog

Signal processing expressions C	Encoding of C
Q	$\tau_C(x) \leftarrow Q(x), \text{value}(x, v).$
$\{s_1, \ldots, s_m\}$	$\tau_C(s_i) \leftarrow \text{value}(s_i, v)$, for each s_i
$\alpha \circ C_1$, where $\circ \in \{+, -, \times, /\}$	$\tau_C(f_C(x)), \text{value}(f_C(x), v) \leftarrow \tau_{C_1}(x), \text{value}(x, v), v = \alpha \circ v'.$
$C_1 : \mathit{value}(\odot, \alpha)$, where $\odot \in \{<, >, \leq, \geq\}$	$\tau_C(f_C(x)), \text{value}(f_C(x), v) \leftarrow \tau_{C_1}(x), \text{value}(x, v), v \odot \alpha.$
$C_1 : \mathit{duration}(\geq, t)$	$\tau_C(f_C(x)) \leftarrow \diamondsuit_{[0,t]} \boxplus_{[0,t]} \tau_{C_1}(x).$ $\text{value}(f_C(x), v) \leftarrow \tau_C(f_C(x)), \text{value}(x, v).$
$C_1 : \mathit{duration}(<, t)$	$\tau_C(f_C(x)) \leftarrow \tau_{C_1}(x),\ \neg\,(\diamondsuit_{[0,t]}(\boxplus_{[0,t]}\tau_{C_1}(x)).$ $\text{value}(f_C(x), v) \leftarrow \tau_C(f_C(x)), \text{value}(x, v).$
$\mathsf{agg}\ C_1$, where $\mathsf{agg} \in \{\mathsf{min}, \mathsf{max}, \mathsf{avg}, \mathsf{sum}\}$	$\tau_C(c), \text{value}(c, v) \leftarrow v = \mathsf{agg}[\![v_1 \mid \text{value}(x, v_1), \tau_{C_1}(x)]\!],$ where c is a fresh constant, $\mathsf{agg}[\![\cdot]\!]$ is an aggregation operator over bags
$C_1 : \mathit{after}[t]\ C_2$	$\tau_C(f_C(x_1)) \leftarrow \big(\tau_{C_1}(x_1)\big)\, \mathcal{U}_{[0,\infty)}\, \big((\neg\tau_{C_1}(x_1) \wedge \neg\tau_{C_2}(x_2))\, \mathcal{U}_{[0,t]}\, \tau_{C_2}(x_2)\big).$ $\text{value}(f_C(x_1), v) \leftarrow \tau_C(f_C(x_1)), \text{value}(x_1, v)$
$C_1 : \mathit{before}[t]\ C_2$	$\tau_C(f_C(x_1)) \leftarrow \big(\tau_{C_1}(x_1)\big)\, \mathcal{S}_{[0,\infty)}\, \big((\neg\tau_{C_1}(x_1) \wedge \neg\tau_{C_2}(x_2))\, \mathcal{S}_{[0,t]}\, \tau_{C_2}(x_2)\big).$ $\text{value}(f_C(x_1), v) \leftarrow \tau_C(f_C(x_1)), \text{value}(x_1, v)$
$C_1 : \mathit{within}\ C_2$	$\tau_C(f_C(x_1)) \leftarrow \Big(\big(\tau_{C_1}(x_1) \wedge \tau_{C_2}(x_2)\big)\, \mathcal{S}_{[0,\infty)}\, (\neg\tau_{C_1}(x_1))\Big)\ \mathcal{U}_{[0,\infty)}\, \big(\neg\tau_{C_1}(x_1)\big).$ $\text{value}(f_C(x_1), v) \leftarrow \tau_C(f_C(x_1)), \text{value}(x_1, v).$
$C_1 : \mathit{trend}(\mathsf{up})$	$\tau_C(f_C(x)) \leftarrow \tau_{C_1}(x), \neg\ \text{notTrendUp}_{C_1}(x)$ $\text{notTrendUp}_{C_1}(x) \leftarrow \tau_{C_1}(x), \text{value}(x, v_1), \diamondsuit_{(0,\delta]}(\text{value}(x, v_2), v_1 > v_2)$ where δ is a "small enough" positive real number $\text{value}(f_C(x), v) \leftarrow \tau_C(f_C(x)), \text{value}(x, v).$
$C_1 : \mathit{trend}(\mathsf{down})$	$\tau_C(f_C(x)) \leftarrow \tau_{C_1}(x), \neg\ \text{notTrendDown}_{C_1}(x)$ $\text{notTrendDown}_{C_1}(x) \leftarrow \text{value}(x, v_1), \diamondsuit_{(0,\delta]}(\text{value}(x, v_2), v_1 < v_2)$ where δ is a "small enough" positive real number $\text{value}(f_C(x), v) \leftarrow \tau_C(f_C(x)), \text{value}(x, v).$

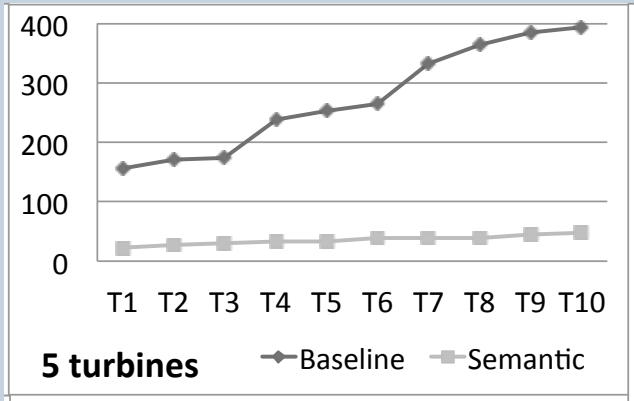
Usability Study

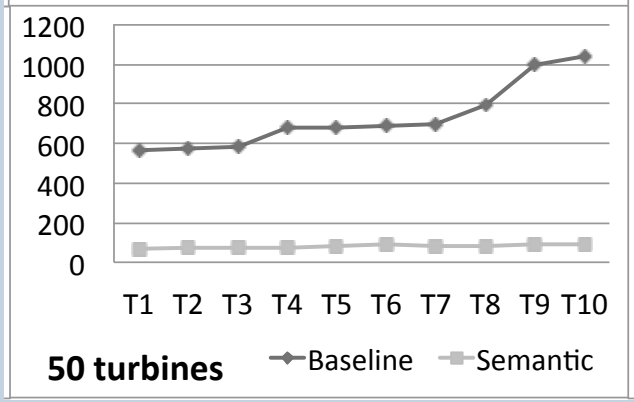
- Users: 6 Siemens engineers
- Tasks: 10 tasks, 3 levels of difficulty: low (3), medium (3), hard (4)
- Setting: write 10 tasks over 1, 5, 10, and 50 turbines
- Results: over 50 turbines 1.900 sec saved, 4 times faster

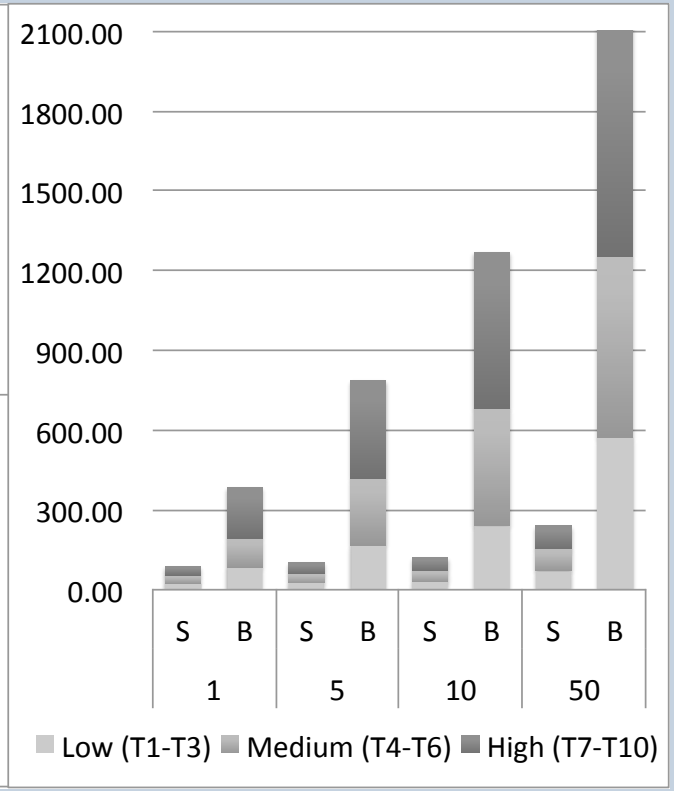
$$\begin{aligned}\text{message}(\text{“}\textit{Ready to Start}\text{”}) = \text{RampChange} : \ & \textit{after}[5m]\ \text{PurgingOver} : \\ & \textit{after}[11m]\ \text{PurgingAndIgnition} : \\ & \textit{after}[15s]\ \text{IgnitionToStand}.\end{aligned}$$

Usability Study

- Users: 6 Siemens engineers
- Tasks: 10 tasks, 3 levels of difficulty: low (3), medium (3), hard (4)
- Setting: write 10 tasks over 1, 5, 10, and 50 turbines
- Results: over 50 turbines 1.900 sec saved, 8 times faster







Conclusion

A novel Semantic Diagnostic Rule-based Language (SDRL)

- Higher-level data independent language
- to write SPRs over ontologies

Conventional Diagnostics

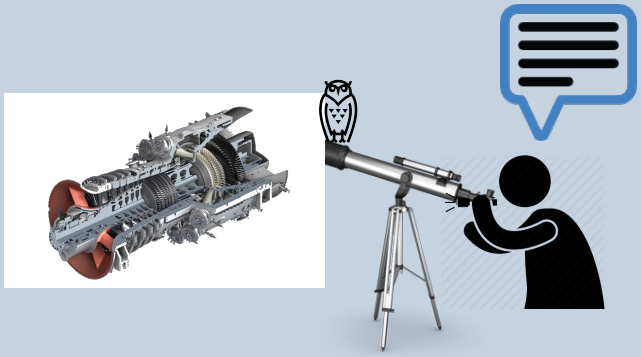
Main technical contribution: **FO rewritable**

- We proved that query answering is FO rewritable
- “Ontology-based data access (OBDA)” - compatible
- I.e. answering queries in SDRL can be checked using SQL

Semantic Diagnostics

Main practical impact

- implementation and evaluation of our approach
- efficiency and usability at Siemens



Next Steps in our Industrial Collaboration

Learning rules SDRL

- How to learn (fragments) of SDRL?

Improving data quality of industrial KGs by

- Learning Schemas over KGs (eg. SHACL constraints)
- Learning mappings from DB to KGs