1. Self-Management

Maintenance and configuration becomes more difficult as systems are increasingly more complex, with multiple distributed components. Systems should manage themselves: self-management.

1. Sensors: In the managed unit are triggered.
2. Autonomic manager analyses sensed values and determines a diagnosis.
3. Autonomic manager makes a remedy plan.
4. Effectors implement adaptation instructions.

2. Self-Management Knowledge

Self-management knowledge is modelled as follows:

- **Static model** – describes the structure of the system (systems, runnables, components, etc.).
- **Dynamic model** – describes the correct behaviour of the system, based on use-cases.
- **Management model** – describes reasoning rules, symptoms of misbehaviour, diagnoses, sensors, etc.

3. Requirements for the Representation of Self-Management Knowledge

- **Reasoning** – The representation must support logical rules to reason about concepts in the models.
- **Distributed knowledge** – In a distributed environment, the representation needs support for reasoning with distributed entities.
- **Usability** – a) Choose a representation that is familiar to the people supplying the self-management knowledge: system administrators, functional analysts, and system developers. b) The availability of tools to develop models (automatic syntax & constraints checks, IDEs, etc.) will increase its acceptance.

4. Semantic Web Satisfies these Requirements

Characteristics of the Semantic Web languages OWL+SWRL:

- **Reasoning** - Rules can be expressed in SWRL, which is closely integrated with OWL: rules can directly refer to OWL-concepts.
- **Distributed knowledge** - Support for Uniform Resource Identifiers (URI) that can identify and refer to resources stored on distributed locations.
- **Usability** – a) OWL has common and relevant features with UML, the de-facto standard in software development. The relationship between OWL and UML is officially described in the Ontology Definition Metamodel (ODM) specification. b) The Semantic Web community provides various tools and techniques for checking syntax and consistency of OWL documents (e.g., Protégé).

5. Example Scenario: Simplified Trading Application

This simplified distributed trading application (simplifiedTrading) is borrowed from Fortis Bank, Netherlands.

- Multiple (distributed) OWL-documents describe the static, dynamic, and management model of the self-managed system.
- **Static model**: simplifiedTrading is a composite system, containing a runnable (webApp), which has a component (paymentComp), which has one class (tradeCls).
- **Dynamic model**: The monitored use-case is sendPayment.
- **Management model**: The Sensor amountSensor monitors the variable amount in tradeCls.
- **Analysers**: The analyser raises the symptom lowAmountSymptom when the SWRL rule matches (lowAmountSymptom rule).

---

**Using Semantic Web Technology for Self-Management of Distributed Systems**

**Reza Haydariou, Michel Oey**

**Benno Overeinder, Frances Brazier**

(pezahay.michiel,bjo,frances)@cs.vu.nl

**Intelligent Interactive Distributed Systems Group**

**Department of Computer Science**

**Vrije Universiteit Amsterdam**