Modeling Cyber-Physical Systems with Semantic Agents

Jing Lin, Sahra Sedigh, and Ann Miller
Department of Electrical and Computer Engineering
Missouri University of Science and Technology
Rolla, USA 65409
Email: {jlpg2,sedighs,milleran}@mst.edu

***Abstract*—The development of accurate models for cyber-physical systems (CPSs) is hampered by the complexity of these systems, fundamental differences in the operation of cyber and physical components, and significant interdependencies among these components. Agent-based modeling shows promise in overcoming these challenges, due to the flexibility of software agents as autonomous and intelligent decision-making components. Semantic agent systems are even more capable, as the structure they provide facilitates the extraction of meaningful content from the data provided to the software agents. In this paper, we present a multi-agent model for a CPS, where the semantic capabilities are underpinned by sensor networks that provide information about the physical operation to the cyber infrastructure. This model is used to represent the static structure and dynamic behavior of an intelligent water distribution network as a CPS case study.**

***Index Terms*—cyber-physical systems; multi-agent model; UML; semantic service; water distribution networks**

I. Introduction

The synergy between agent-based modeling and semantic technologies holds promise for the resolution of challenges posed by a broad range of complex systems, in particular *cyber-physical systems* (CPSs), where embedded computing and communication capabilities are used to streamline and fortify the operation of a physical system [1], [2]. In CPSs, sensors collect information about the physical operation of the system, and communicate this information in real-time to the computers and embedded systems used for intelligent control. These cyber components use computational intelligence to process the information and determine appropriate control settings for physical components of the system, such as devices used to control the flow of a physical commodity, e.g., water or electric power, on a line.

One of the fundamental challenges in research related to CPSs is accurate modeling and representation of these systems. The main difficulty lies in developing an integrated model that represents both cyber and physical aspects with high fidelity. Among existing techniques, agent-based modeling is a suitable choice, as it can encapsulate diverse attributes within one agent, while capturing the interaction among autonomous, heterogeneous agents that strive towards a common goal in a distributed fashion. Sensors are key to this approach, as they provide situational awareness to the agents and enable them to function based on the semantics of their mission and the specifics of their environment. The research presented in this paper aims to accurately model a CPS as a multi-agent system, where each agent is an independent entity that manages resources within its local scope. In the proposed model, information from the sensor networks is dynamically integrated with semantic services to support real-time decision support in the information-rich environment of a CPS.

The CPS domain used as a case study for application of this model is intelligent *water distribution networks (WDNs)*. In a WDN, physical components, e.g., valves, pipes, and reservoirs, are coupled with the hardware and software that supports intelligent water allocation. Fig. 1 depicts a sample WDN.

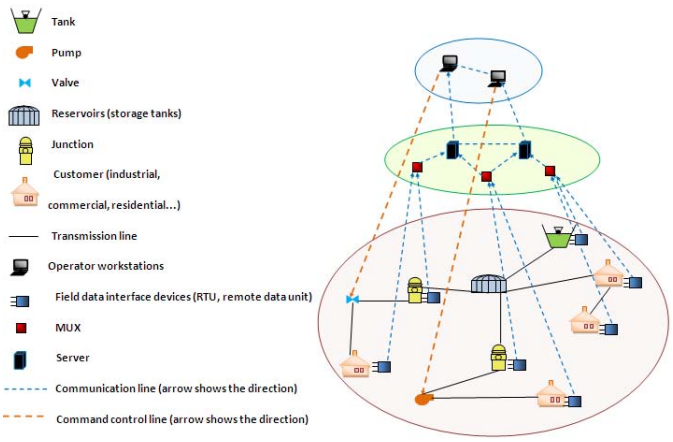

Fig. 1. Cyber and physical components of an intelligent WDN.

The primary goal of WDNs is to provide a dependable source of potable water to the public. Information such as demand patterns, water quantity (flow and pressure head), and water quality (contaminants and minerals) is critical in achieving this goal and beneficial in guiding maintenance efforts and identifying vulnerable areas requiring fortification and/or monitoring. Sensors dispersed in the physical infrastructure collect this information, which is summed by multiplexers and servers for hierarchical semantic interpretation. The processed and reasoned sensor data is then fed to distributed algorithms executing on the cyber infrastructure. These algorithms provide decision support to hardware controllers that are used to manage the allocation (quantity) and chemical composition (quality) of the water. The algorithms are implemented through

software executing on multiple distributed computing devices. In our model, the software is represented by several agents, each of which is capable of perceiving its environment, acting on that perception, communicating with other agents, and exhibiting behavior that fits its goal.

The remainder of this paper is organized as follows. Section II presents background information. In Section III, we present a UML multi-agent model that captures the static structure and dynamic behavior of a WDN, which serves as our case study within CPSs. An ontology is defined and associated with the definition of an agent in Section IV, where we also propose a sensor information hierarchy for interpretation of the semantics of raw data streams, and describe how semantic interpretation capabilities can be implemented through C++ on Matlab. Section V concludes the paper and describes future research directions.

II. Related Work

CPSs are an emerging research area, and the body of related literature is limited. A considerable fraction of related work examines critical infrastructure systems, which are prime examples of CPSs. Salient studies include [3], which investigates interdependencies among different components of critical infrastructure systems, and [4], which provides a relatively comprehensive summary of modeling and simulation techniques for critical infrastructure systems. System complexity has been identified as the main challenge in characterizing interdependencies in CPSs [5]. Other challenges include the low probability of occurrence of critical events, differences in time scales and geographical locations, and the difficulty of gathering the accurate data needed for modeling. These challenges are clearly articulated in the literature, but solutions are very scarce.

The need to use agent-based modeling for distributed complex systems has been investigated in [6]. The work in [7] adopts a distributed multi-agent architecture to analyze the observed information in real-time to adapt the multi-agent system to the evolution of its environment. To address the dependability of multi-agent systems, [8] improves the capability of calculating how critical an agent is to the system through its interactions with other agents and provides a framework that uses this information to ensure availability and reliability. UML 2.0 has been adopted to model agent-based or multi-agent systems as a formal specification language with precise semantics, as demonstrated in detail in [9].

Semantic agent technologies are typically closely associated with sensor networks, and several prototype systems or software architectures have been proposed based on the combination of the two. A prototype for battlefield information systems has been described in [10], where the stated goal is to dynamically integrate sensor networks with information fusion processes to support real-time sensing, interpretation, and decision-making in an information-rich tactical environment. In [11], an architecture and programming model for a semantic service-oriented sensor information platform has been presented. The use of autonomous semantic agents in developing a new software architecture for distributed processing environments has been proposed in [12].

The complexity of CPSs, as well as the necessity of capturing embedded computing and communication capabilities motivate the use of distributed agents and semantic services for representing the relationship between the cyber and physical infrastructures. In our work, the distributed semantic agent model augments the data acquisition of sensors in the CPS with ontological decision-making intelligence. The proposed model not only captures the complexity of the CPS in a clear and understandable way, but also takes accurate semantic interpretation into consideration. To our knowledge, our work is the first study to apply semantic agents to modeling of CPSs.

III. Multi-Agent Systems in UML

In this section, we present an agent-based model for an intelligent WDN, as a case study of CPSs. The model is represented in UML, and as such, creating a use case diagram is the first step for system analysis. A use case captures the interaction of a number of external actors with the system towards accomplishment of a goal. Fig. 2 shows the actor and use cases involved in an intelligent WDN. The use case diagram presented here can be generalized to other CPSs whose main goal is management of a physical commodity. Examples include power grids and intelligent transportation systems.

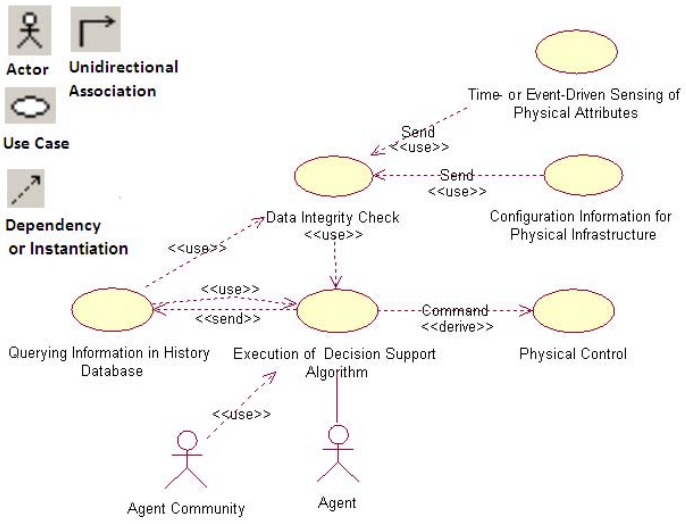

Fig. 2. Use case diagram for an intelligent WDN.

The CPS agent is the actor in the use case diagram, and associated with the decision support algorithm. For simplicity, only use cases associated with one agent are shown in Fig. 2; all other agents have similar use cases associated with them. As shown in the use case diagram, sensors collect information about the physical operation of the system on a time- or event-triggered basis. The collected information is aggregated by a multiplexer and sent to *Data Integrity Check* for intelligent semantic inference. The *Data Integrity Check*

use case utilizes three main data streams, specifically, raw data from the corresponding sensor, real-time data from nearby sensors for the same or related physical attributes, and the data from a history database. The second and third data streams mentioned are used for corroboration of the first, which is accomplished by checking for discrepancies in the values, whether in variation or in conformance to physical (hydraulic) laws that govern the operation of the physical infrastructure of the WDN. If no data is available from nearby sensors, as would be the case if all nearby sensors are in sleep mode, the history database will serve as a source of data for corroboration.

The *Decision Support Algorithm* uses three data streams: one data stream from the *Data Integrity Check*, another from the history database, and a third data stream from other agents. The *Decision Support Algorithm* is implemented through software code for intelligent management of physical commodities. The algorithm can make use of legitimate (corroborated) data whose integrity has been checked, and can also resort to history data for adjustment (rectification) of the calculated values in determining an appropriate strategy for resource allocation. Meanwhile, the local agent interacts and negotiates with other agents by sharing real-time information that provides a global perspective of resources in the system, and adjusts its own strategy accordingly.

Fig. 3 describes the state transition of data in one period, which is the time span from the point that data is collected (start state) until control has been exerted on the water consumption entity (end state). As agent-based modeling is a data-driven method, it is vitally important to track each state transition of the data. The condition that can trigger entry to or exit from a particular state has been specified in Fig. 3. The history state (encircled 'H') records the state of the system immediately before query of the history database takes place. Once the agent has finished data retrieval, the state reverts back to the original state before data storage, and the agent begins processing based on the combination of retrieved historical data and the originally collected data. The flow of the decision making procedure, whose goal is to allocate water (quantity), has been specified in the figure with two decision blocks (encircled diamonds).

IV. Semantic Interpretation Service

A. Sensor information ontology

Semantic interpretation is carried out on semantic streams, each of which is defined in a domain-specific ontology associated with the agent. The specific domain in this paper is intelligent water distribution. Generally, an ontology is a description, e.g., a formal specification of a program, of the concepts and relationships that can exist for an agent or a community of agents. The notion of ontology utilized in this paper is a model that describes semantic relations among components of the physical and cyber infrastructures, respectively, as well as the interdependencies across the cyber-physical boundary. Each component in the ontology model is a unique class in terms of programming implementation, with properties and parameters described in the class definition. The relations define how classes can be related to one another. Semantic interpretation is implemented through distributed software with capabilities of extraction, analysis, and processing of the semantic stream. The definition of an ontology for the WDN domain helps unify information presentation and permits software and information reuse, so as to reduce information redundancy during the process of semantic interpretation in the agents.

Fig. 4 shows the information hierarchy for failure detection through the semantic interpretation process. In the UML class diagram, each block represents one type of semantic stream in the intelligent WDN. The attributes of each class have been omitted in the interest of figure clarity. Details of the attributes are presented in Fig. 6, which shows pseudocode for the semantic service. Fig. 4 shows that a failure in the WDN can be detected by the agent in the event of device or information failure, the latter of which occurs when data falls outside a pre-defined safety range. Failures in the physical infrastructure of a WDN are of two main two types, physical failure due to excessive values of pressure and elevation, or biochemical failure due to excessive quantities of a biochemical substance or discovery of unknown biochemical materials. Cyber failures can be caused by human error or device malfunction. Each class identifies one type of semantic stream that can lead to failure in the CPS, and the ultimate determination of failure (or the overall interpretation) is carried out by the corresponding agent, which is in charge of all sensors deployed within its administrative scope.

The sensor information ontology captures the semantic entities (classes in the UML diagram) and the relations of events and objects, resulting in an intelligent reasoning procedure beyond what sensors can provide through detection alone. The ontology proposed in Fig. 4 is specific to the WDN domain, but can be readily adapted to other CPSs, such as smart power grids.

B. Model for semantic services

Based on the sensor information ontology proposed, we can develop components that convert semantics between classes in the information processing hierarchy, by extracting new semantic information from existing data streams. In other words, the components encapsulate the semantic service into a "black-box" containing the execution method, which takes as input information corresponding to events detected by sensors and generates as output a number of meaningful new events.

We propose a semantic service model that overlays the ontology defined in Section IV-A. The semantic services can be categorized into two types, i.e., a) the service that supplements input events with additional semantic annotation, and b) the service that produces new semantic streams. The first type of service can only identify additional properties carried by the input event. For example, a sensor has detected that the water pressure in a certain area has exceeded the safety threshold and reports this event to its semantic service component, which can be a superior sensor or multiplexer. The semantic service model associated with this component will add the geographical location as an additional identifier to distinguish

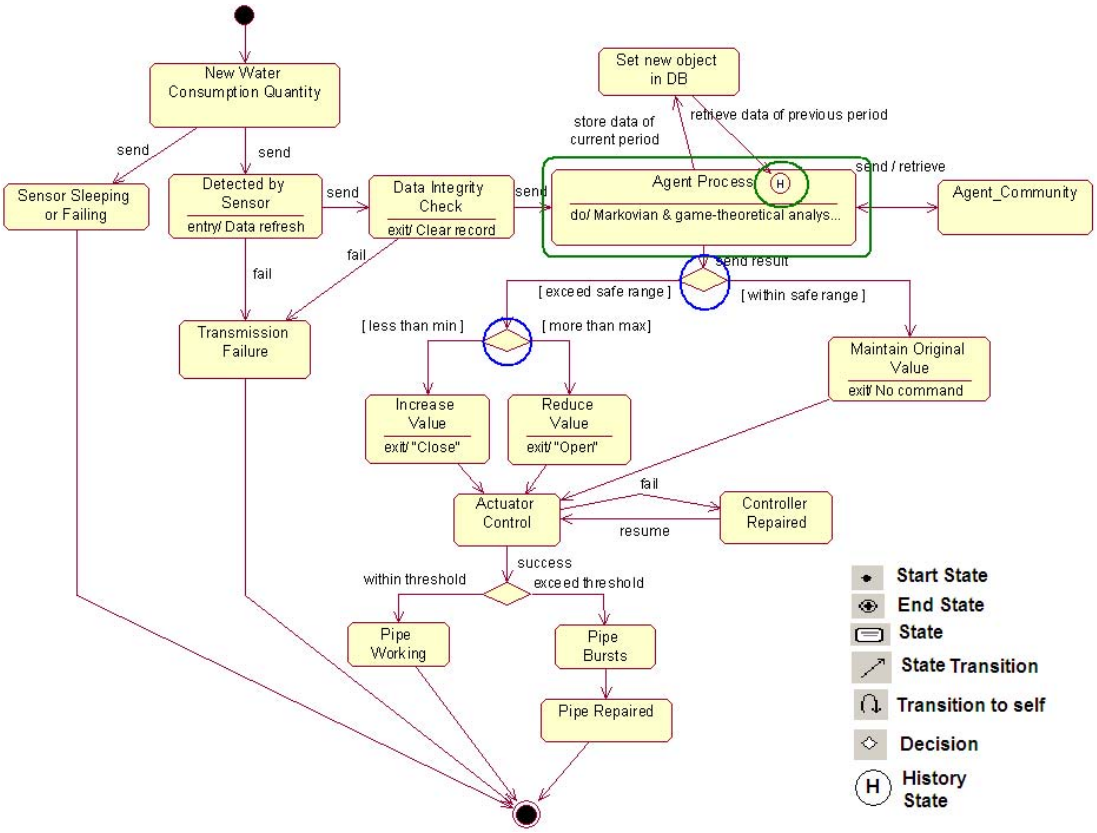

Fig. 3. State transition diagram for an intelligent WDN.

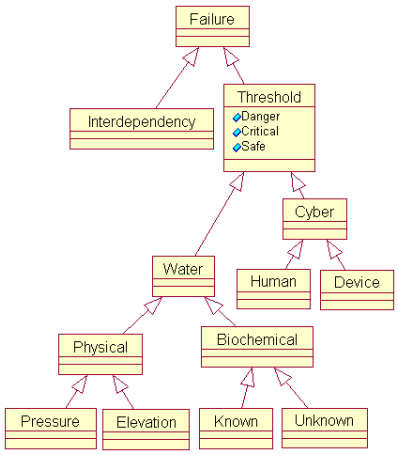

Fig. 4. Failure detection ontology (in UML) for an intelligent WDN.

this event from events reported from other areas. Such a functionality is particularly useful for distributed control and management in the context of CPS, where a service may not correspond to a centralized component that physically exists on one device; it can be physically implemented on several distributed devices, but logically exist as a single service.

The second type of service automatically terminates the input semantic stream, and uses the generated output semantic stream as the new stream propagating on the ontology. The essence of this type of service is semantic transformation, where the input and output events are different classes in the ontology. One typical semantic transformation is generalization. For example, in Fig. 4, an excessive pressure quantity will be interpreted as physical failure due to an abnormal pressure value. Later on, the semantic stream of physical failure will be propagated to a higher level for ultimate decision making, instead of the semantic stream of abnormal pressure quantity, which no longer exists.

The benefits of proposing such a semantic service model on an information ontology include the reduction of information redundancy, the provision of pre-processing and abstraction data to the agent, and the facilitation of semantic query by a user. A user can issue a query that requests that a certain data stream with desired semantics be provided to a certain component device to diagnose whether failure exists on the queried level.

C. Semantic agent framework

Fig. 5 illustrates how the agents use the information collected by sensors and the interpreted semantics based on the defined ontology. Raw data is obtained from sensor networks, and since each agent is an independent entity in charge of a particular geographical region, sensors located in distributed areas are managed by different agents (with possible overlap). For a semantic service component, the input semantic events

are preconditions of the service. The postconditions, i.e., the processed output semantics are provided to the agents for further computing.

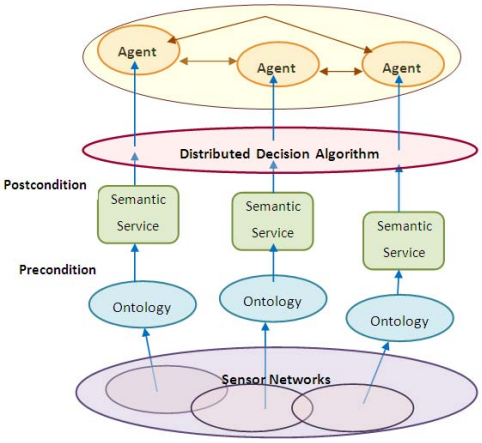

Fig. 5. Semantic agent framework.

The choice of C++ for implementation of the semantic service was motivated by several factors. Firstly, a service component is an information-rich component that needs to define the semantic service for execution, extract information from the input, and produce new semantics at the output. Such logical analysis is best implemented through a high-level programming language such as C++ or JAVA, rather than a computing tool. Secondly, a class in C++ is a good fit for our definition of the service component; the declaration of service properties and the execution method of the service can be encapsulated into one class. Thirdly, Matlab2008b can integrate C++ and support parallel computing, and the integration of the semantic service and computation of the algorithm in Matlab will make simulation of the CPS more compact and faster.

To implement the service in C++, the properties of the service are parameterized, and the execution method of the service becomes the corresponding method of the service class. For example, consider the *Pressure* to *Failure* branch of Fig. 4. Sensors are treated as services with only output semantics, which are parameterized into data that can be used by superior service components. Each component has been specified with a service name and parameters associated with the service. Each service takes the outputs of an inferior component as the input to its execution method, and inherits the parameters to ensure that attributes of a potential failure source (such as pressure, failure time, or location) are not lost during information propagation on the ontology. Pseudocode for our C++ implementation is shown in Fig. 6.

V. Conclusions and Future Work

CPSs are an emerging research area, and existing techniques for their modeling are quite limited. A number of related challenges were discussed in this paper, with focus on the importance of capturing interdependencies and streamlining semantic interpretation between the cyber and physical infrastructures. The use of agent-based modeling was proposed, and a case study of an intelligent WDN was presented to demonstrate the ability of the technique to capture various facets of the operation of a CPS. To reduce information redundancy in the model and simplify the data interpretation procedure of the agents, a semantic service model based on the definition of an ontology was proposed, and its implementation in C++ was presented. The proposed model reflects the semantics of intelligent water distribution, but can be modified for use in other CPS domains.

The modeling work presented in this paper is a preliminary step that will facilitate the broader goal of modeling CPSs. Future extensions to this work will incorporate sophisticated decision support algorithms, e.g., game theory, for the agents. The semantic service model implemented through C++ will be integrated with Matlab to facilitate the complex computation required. Provision of the semantic service in C++ to the decision support algorithm in Matlab will create an advanced simulation environment for CPSs, which can be invaluable to gaining a more profound understanding of the operation of CPSs.

References

[1] E. Lee, "Cyber physical systems: Design challenges," in *Proc. of the 11th IEEE Int'l. Symposium on Object Oriented Real-Time Distributed Computing (ISORC'08)*, May 2008, pp. 363–369.

[2] J. Sztipanovits, "Composition of cyber-physical systems," in *Proc. of the 14th Annual IEEE Int'l. Conference and Workshops on the Engineering of Computer-Based Systems (ECBS '07)*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 3–6.

[3] N. K. Svendsen and S. D. Wolthusen, "Analysis and statistical properties of critical infrastructure interdependency multiflow models," in *Proc. of the IEEE Information Assurance and Security Workshop (IAW '07)*, June 2007, pp. 247–254.

[4] S. M. Rinaldi, "Modeling and simulating critical infrastructures and their interdependencies," in *Proc. of the 37th Hawaii Int'l. Conf. on System Sciences*, 2004.

[5] P. Pederson, "Critical infrastructure interdependency modeling: The survey of U.S. and international research," Idaho National Laboratory, Tech. Rep., August 2006.

[6] C. M.Macal and M. J. North, "Tutorial on agent-based modeling and simulation Part 2: How to model with agents," in *Proc. of the 38th Winter Simulation Conf. (WSC '06)*, 2006, pp. 73–83.

[7] Z. Guessoum, N. Faci, and J. P. Briot, "Adaptive replication of large-scale multi-agent systems - towards a fault-tolerant multi-agent platform," in *Proc. of the 4th Int'l. Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS)*. ACM, 2005.

[8] M. de C.Gatti, C. de Lucena, and J. Briot, "On fault tolerance in law-governed multi-agent systems," in *Proc. of the 5th Int'l. Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS)*. ACM, 2006.

[9] B. Bauer and J. Odell, "UML 2.0 and agents: How to build agent-based systems with the new UML standard," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 2, 2005.

[10] G. Jiang, W. Chung, and G. Cybenko, "Semantic agent technologies for tactical sensor networks," in *Proceedings of the SPIE*, 2003, pp. 311–320.

[11] J. Liu and F. Zhao, "Towards semantic services for sensor-rich information systems," in *2005 2nd International Conference on Broadband Networks*, 2005, pp. 44–51.

[12] A. Elci and B. Rahnama, "Consideration on a new software architecture for distributed environments using autonomous semantic agents," in *Proc. of the 29th Annual International Computer Software and Applications Conference (COMPSAC'05)*, 2005.

```
Sensor{
water_sensor, geoID,[width,length,height], /* properties of sensors: water detection, geographical ID, location*/
  Outputs(pressure, elevation, biochemical, location);          /*the parameters can be detected by sensor* /
}

Pressure component:
Pressure_Service{
    service(pressure),                          /*service indicates execution method and the parameter is pressure*/
    Inputs (sensor(water_sensor, geoID, [width,length,height]));
      If (pressure > normal range)
        Outputs (pressure_normal (false), detected (pressure,geoID,T)); /*add the judgment result and time T*/
}

Physical component:
Physical_Service{
    service(physical_failure),
    Inputs(pressure_service(pressure_normal(false), detected (pressure,geoID,T)));
      If ((elevation < normal range) && (pressure_normal = false)) /*guarantees pressure is the unique reason*/
        Outputs(physical_normal(false), detected(physical_failure,pressure,geoID,T)); /*inherit inferior attribute*/
}

Water component:
Water_Service{
  service(water_failure),
  Inputs(physical_service(normal(false), (physical_failure,pressure,geoID,T)));
      If ((biochemical_normal = true) && (physical_normal = false))                    /*same as above*/
      Outputs(water_normal(false), detected(water_failure,physical _failure,pressure,geoID,T));
}

Threshold component:
Threshold_Service{
  service(failure),
  Inputs(physical_service(normal(false), detected(physical_failure,pressure,geoID,T)));
      Switch(detected(pressure))
      {
          Case (within range for safe): service terminates;
          Case (within range for critical): send (pressure,geoID,T) to database;
          Case (within range for safe): output system failure alert;
          Default: service terminates;
        }
  Outputs(system_failure_alert, detected( water_failure,physical _failure,pressure,geoID,T));
}
```

Fig. 6. Pseudocode for semantic service.