Towards a Semantically Enhanced Control Architecture

Georgios M. Milis, Christos G. Panayiotou and Marios M. Polycarpou

Abstract— **The design of feedback control systems typically assumes a fixed number of sensors and actuators. However, in today's large-scale complex systems there is often a need to replace faulty sensors/actuators with new ones or to deploy additional sensors of possibly different characteristics compared to the old sensors. In such cases, it may be necessary to deactivate, redesign and reconfigure the feedback control system, which implies system interruptions and significant economic costs. This paper provides a first step towards a semantically enhanced feedback control architecture, where it is possible to automatically reconfigure the sensor/actuator components without having to redesign the feedback control law. The proposed architecture uses ontology-based semantic mediation. This paper focuses on the sensor case and utilizes a simple example to illustrate the approach. It is emphasized that the proposed architecture will be even more beneficial for achieving interoperability in the new paradigm of *system of systems* where systems may be added or removed and the expectation is that the overall system should continue to operate optimally.**

I. INTRODUCTION

The evolution of the *system of systems* paradigm necessitates the easy interconnection and interoperability of multiple systems each including one or more control systems. The control systems consist of several components, from sensors to actuators to feedback control and decision algorithms, being implemented in hardware and/or software. Especially in large-scale systems, and with the advancements in information and communication technologies, these components tend to be implemented as separate subsystems that are deployed as spatially distributed entities.

The design of feedback control systems is usually based on a fixed configuration of components (sensors and actuators), with certain knowledge of their specific characteristics. However, sometimes there is a need to replace a faulty sensor or install some additional sensors. New sensors may have different characteristics (e.g. due to unavailability of the same type of components or due to upgrading to new technology), which would require readjustment or redesign of the overall feedback control system. Let's consider the simple example of temperature control for a building with multiple rooms, where a controller is designed to regulate the temperature of each room to specific desired values. The controller receives the measurements of temperature sensors that are installed in the rooms and controls the power of installed electric radiators. Replacing a faulty sensor component with another one, which measures in different unit and with different accuracy, might require the deactivation, re-design and reactivation of the control system. Also, the deployment of a new electric radiator that cannot directly interpret the control signal, might again require re-design of the controller. However, it is not practical or economic to deactivate an entire control system for reconfiguration or redesign. The situation can only be overcome if the control system has built-in flexibility to accommodate and adapt to such changes online.

An approach widely adopted in research areas like the *fault diagnosis*, as well as, the *adaptive* and *fault-tolerant control* is to design intelligent control type algorithms, aiming to facilitate the flexibility of the control system with respect to on-line adaptation and learning of system uncertainties and/or time variations. The classical model-based approaches for fault detection, isolation and identification procedures in linear and nonlinear systems are well described in [1], [2]. Approaches to designing fault tolerant and reconfigurable control systems are presented in [3]. In [4] the author also addresses the issue of fault-tolerant sensor, actuator and control systems. The textbooks [5] and [6] comprise comprehensive information sources on approaches to design adaptive controllers, while [7] provide a methodology for designing adaptive approximation based control systems.

The increasing scale and complexity of control applications, led [8] and [9] to propose considering the *Systems of Systems* perspective where each component is to be treated as an independent system in terms of operation and management. Although the control system components are not always completely independent, the ideas of a component-oriented architecture seem promising.

The semantically enhanced architecture presented in this paper is complementary with the idea of Plug and Play, which aims at exploring new designs that enable automatic reconfigurability in cases of components' changes. The authors in [10] explain that after installing a new device, the control system needs to become aware of the new situation, by acknowledging any new signal, determining its relation to the running control process and identifying the extent to which that signal can be useful for meeting the objectives. The importance of *situational awareness* for effective and efficient adaptation of control systems to changes is also emphasized in [11].

The motivation of the work presented in this paper lies in the understanding that the existence of an awareness framework in the environment where the changes happen, would simplify the design and even improve the efficiency of feedback control algorithms and subsequently the proper online adaptation/reconfiguration. In summary, the contribution of this work lies in exploring the ontology-based semantic

The authors are with the KIOS Research Center for Intelligent Systems and Networks, Department of Electrical and Computer Engineering, University of Cyprus, {milis.georgios, christosp, mpolycar}@ucy.ac.cy

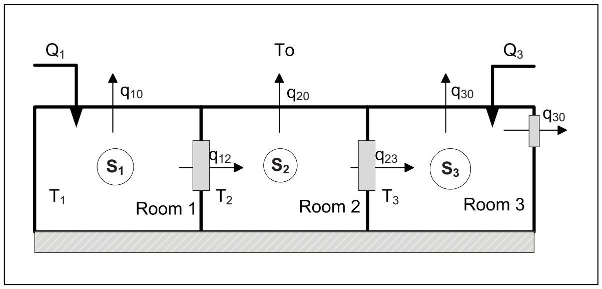

Fig. 1: System used in the motivating example

interoperability concepts and utilizing them to design a flexible control architecture. The remainder of this paper is organized as follows: Section II presents and discusses the proposed architecture with a motivating example. Then, Section III provides a detailed description of the ontology-based semantic interoperability concept. Section IV presents the proposed implementation of the semantically enhanced architecture followed by simulations and results in section V. Finally, section VI concludes the work and provides insights to interesting future research paths.

II. ARCHITECTURE

A. Motivating Example

A system comprising three rooms is considered, as shown in Fig. 1. The objective of the employed feedback control is to regulate the temperature in each room (T_i) to reference set points $r_i,\ i \in \{1,2,3\}$. The temperature in each room is measured with sensors $S_i,\ i \in \{1,2,3\}$ while the outside temperature T_0 is also measured. To keep the example simple, a single, centralized controller is assumed. The thermal energy transferred from room i to the environment or other rooms is denoted with $q_{i,j}$ ($i \in \{1,2,3\},\ j \in 0,1,2,3\ i \neq j$). In addition, Rooms 1 and 3 are equipped with one electric radiator each, that is able to generate thermal energy ($Q_j,\ j \in \{1,3\}$) to heat the room.

B. Semantically Enhanced Control Architecture

To design a control architecture, the designer needs to obtain a model of the system and design a feedback control algorithm taking into consideration the a priori knowledge about the system and the components used. The result is a system that comprises of a set of components (hardware and software) that cooperate to achieve the desired objectives. After launching, if for any reason it is required that a component is changed and the new component is not 100% compatible with the already installed components, then the designer will need to manually make any redesigns/reconfigurations. Alternatively, the control algorithm may have embedded intelligence to gradually adapt to the new situation (e.g., use an adaptive control law). The latter approach, though it can work is some cases, may be inefficient due to high complexity and does not have a way to utilize any information that could have been available a priori. Each of the approaches has its own advantage: the former offers the human ability to perform changes based on a priori knowledge and the latter offers automation. To combine the advantages of both approaches one should design a system where the components are able to understand the environment in which they operate and communicate efficiently with other components in order to be automatically reconfigured as a result of a change. Once the reconfiguration takes place, the system can continue incorporating adaptation and learning as required.

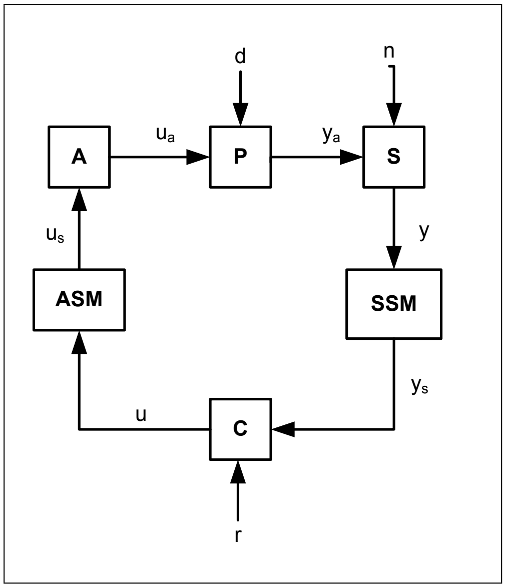

Fig. 2: Semantically Enhanced Control Architecture

The automatic reconfiguration requires the availability of an intermediary layer among the components. This layer will be responsible for supporting the components by exchanging meaningful messages, as well as, applying mediation processing (e.g. perform transformations of the physical signals) on the messages as required. To be able to perform the transformations online, the signals should not be fed directly to the components that will use them but pass through dedicated mediation components which will act as the proxy. This can be realised through the architecture in Fig. 2.

The system consists of the plant P which takes an input u_a from the actuator(s) (A) (and/or a disturbance d) to produce an output signal y_a which is measured by the sensors (S). In a typical feedback controlled system, the sensor measurements (y - possibly corrupted with some noise n) are *directly* used by a controller C which generates a control signal u that is used to *directly* drive the actuator(s). In the proposed architecture (Fig. 2), the feedback loop is enhanced with the introduction of a *Sensors Semantic Mediator* component (SSM) and an *Actuators Semantic Mediator* component (ASM). SSM and ASM act as the interfaces between the sensors and actuators and the rest of the control system, in order to achieve flexible interaction. It is noted that the connections between the components in Fig. 2 represent only the exchange of numerical signals. The communication between the components and the SSM/ASM for the exchange of semantic information (as explained below) is not illustrated but assumed existing.

Each of the SSM and ASM contains the following:

1) *Knowledge base (memory)*: a set of (pre-inserted by experts) knowledge facts and rules that can be also used to infer knowledge not directly encoded.
2) *Objects' database*: the actual storage of components that are not hardware implementations. This database may store pre-designed tools and functions that can be employed by the mediator when required, e.g. to transform signals.

The knowledge base enables the control system components to improve their inter-operation efficiency, by using expert knowledge and understanding of the environment and exchanging semantically meaningful data. This ability to cooperate by exchanging meaning is called *semantic interoperability*. The knowledge can be represented in a machine-readable format with the help of *ontologies*. The two terms are introduced in the following section.

III. SEMANTIC INTEROPERABILITY

In the *European Interoperability Framework*[1] (EIF), semantic interoperability is defined as the ability of components to share and use meaning. In the introduced motivating example, the sensors produce measurement signals that can be exploited by the controller by comparing them to the desired temperature $\mathbf{r}$ of the rooms and deciding on proper control signals. To improve the operation, as also stated in [12], an intermediary layer is required, to enable the components to share semantic knowledge. A sensor, upon its installation, may send information of the form: *I am a sensor, I measure temperature in Celsius degrees with accuracy* $\pm 3\%$ *and I am installed in Room 1.* Similarly, the controller may send information: *I am a temperature controller, I need to receive the temperatures of the rooms in Fahrenheit degrees, etc.* These descriptions of the components are called *semantic annotations* and will become clear in Section IV. Implementing a mechanism by which the components can exploit this knowledge, will improve their ability to achieve their goals. The implementation of the semantic messages can be done using pre-defined domain vocabularies (ontologies) to semantically annotate the components and provide structure to the messages.

A. Ontologies

In philosophy, ontology is the study of things that exist. According to [13], ontologies are content theories about *types* of objects, *properties* of objects and *relations* between objects, within a specified *domain of knowledge*. For instance, temperature sensors and electric radiators may be considered as types of objects. A sensor then may have properties, like the measurement unit, the accuracy, etc.. Finally, each *sensor* may relate to the measurement of specific state (room temperature) in the model. Ontologies have been already proposed in [14] for the use of sensors by expert systems. Ontological representation of knowledge is used also in [15] to semantically annotate process models. Also, in [16] the ontologies are used to capture the reasoning steps included in problem solving methods.

IV. PROPOSED SEMANTIC INTEROPERABILITY IMPLEMENTATION

SSM has a number of inputs and a number of outputs to connect to a controller. ASM has inputs from controller(s) and outputs that can be fed to actuators. Inputs and outputs can be of different type, e.g. wired, wireless. Both components are implemented such as to be able to relay inputs to outputs as required. In addition, the components are able to apply specific transformations on the inputs before connecting them to the outputs. Finally, the components are able to execute software algorithms and pass the results to outputs.

A. Knowledge-base and Objects' database

First, an ontology is required that will serve as the knowledge base for the semantic mediators to use for their rational decisions. As a starting point, the Semantic Sensor Network Ontology ([17]) is used for the proof of concept built in this work, which is a recent effort towards designing a general-purpose sensor network ontology, building upon standardisation efforts by the Open Geospatial Consortium (OGC) ([18]) and the Semantic Web activity of World Wide Web Consortium[2] (W3C). This ontology is extended here with objects and relations from the general control system domain, to build a common language for the components to facilitate situational awareness and interoperability. The components of a closed-loop system are annotated as objects of the ontology with a three-tuple of the form:

$$o_i = \{ID_i, Type_i, Relations_i\},\ i \in \{1, 2, ...\}$$

That is, the annotation of a component o_i consist of a unique identity of the component (ID_i), a definition of its type ($Type_i$) and a definition of its relations ($Relations_i$) to other components.

The following sets are defined to represent the types of objects:

$\mathcal{M}$: The set of system models (e.g. a state-space model of the system)

$\mathcal{S}$: The set of sensors (e.g. the temperature sensor installed in Room 1)

$\mathcal{C}$: The set of controllers (e.g. a proportional controller)

$\mathcal{A}$: The set of actuators (e.g. the electric radiator installed in Room 3)

$\mathcal{X}$: The set of states of the process (e.g. the state (temperature) of Room 2)

$\mathcal{P}$: The set of physical properties (e.g. the *temperature*)

$\mathcal{MU}$: The set of measurement units (e.g. the unit *Celsius*)

$\mathcal{F}$: The set of functions/tools (e.g. the function transforming Celsius to Fahrenheit degrees, an observer for the estimation of a non-measured state, etc.)

[1] www.ec.europa.euisadocumentsisa_annex_ii_eif_en.pdf

[2] www.w3.org

The “superset” $\mathcal{W}$ is also defined, which is the set of all such sets, for example $\mathcal{W} = \{\mathcal{M}, \mathcal{S}, \mathcal{C}, \mathcal{A}, \mathcal{X}, \mathcal{P}, \mathcal{MU}, \mathcal{F}\}$. Then, the type of each object is represented by a vector of the form:

$$Type_i = [x^{\mathcal{M}}, x^{\mathcal{S}}, x^{\mathcal{C}}, x^{\mathcal{A}}, x^{\mathcal{X}}, x^{\mathcal{P}}, x^{\mathcal{MU}}, x^{\mathcal{F}}],$$

where:

$$x^{\mathcal{J}} = \begin{cases} 1 & \text{if } o_i \in \mathcal{J} \\ 0 & \text{otherwise} \end{cases}, \mathcal{J} \in \mathcal{W}$$

For instance, the type of a controller is represented by the vector $[0, 0, 1, 0, 0, 0, 0, 0]$. Next, the relations between the objects are defined:

$$r_l(o_i, o_j, ...) = \begin{cases} 1, & \text{if } o_i, o_j, \cdots \text{ have the relation } l \\ 0, & \text{otherwise} \end{cases}$$

where $i, j \in \{1, 2, ..., N\}$ $j \neq i$, $l \in \mathcal{L}$ and $\mathcal{L}$ is the set of pre-defined possible relations between the objects. For the purposes of this paper, the following relations have been defined: $\mathcal{L} = \{1$ ($o_i \in \mathcal{C}$ is designed for $o_j \in \mathcal{M}$), 2 ($o_i \in \mathcal{X}$ represents $o_j \in \mathcal{P}$), 3 ($o_i \in \mathcal{S}$ measures in $o_j \in \mathcal{MU}$), 4 ($o_i \in \mathcal{M}$ models state $o_j \in \mathcal{X}$), 5 ($o_i \in \mathcal{C}$ requires $o_j \in \mathcal{X}$ in units $o_k \in \mathcal{MU}$), 6 ($o_i \in \mathcal{MU}$ is transformed to $o_j \in \mathcal{MU}$ through $o_k \in \mathcal{F}$), 7 ($o_i \in \mathcal{S}$ measures $o_j \in \mathcal{X}$)$\}$

Let’s assume the objects $o_1, ..., o_8$ be objects of types $\mathcal{M}, \mathcal{S}, \mathcal{C}, \mathcal{A}, \mathcal{X}, \mathcal{P}, \mathcal{MU}$ and $\mathcal{F}$, respectively. Then, $r_1(o_3, o_1) = 1$ if the controller o_3 is designed for the model o_1 and $r_5(o_3, o_5, o_7) = 1$ if the controller o_3 requires state o_5 to be measured in units o_7. For completeness, it is noted that several constraints are enforced, e.g., on the number of times a specific object can appear as argument in specific relations. For instance, a *controller* can be designed only for one *process model*, whereas a *state* may be measured by any number of *sensors*.

Further to the knowledge directly encoded within the ontology, hidden knowledge can be obtained through logical inference. For instance, o_5 may be the temperature of Room 1 and o_2 may be the sensor measuring this temperature in o_7 (Fahrenheit degrees). Also o_3 may regulate the state o_5 and assume it is measured in o_7. The controller is designed based on the model o_1 which models the state o_5. Whether the output of sensor o_2 can be used as input to the controller o_3, is not encoded with direct relation in the ontology. However, this can be inferred by implementing reasoning rules, serving as meta-relations. An example of such rule might be:

$g_1(o_i \in \mathcal{C}, o_j \in \mathcal{S}) = r_1(o_i, O^M) \cdot r_4(O^M, O^X) \cdot r_7(o_j, O^X) \cdot r_3(o_j, O^{MU}) \cdot r_5(o_i, O^X, O^{MU})$

where $O^{\mathcal{M}}$, $O^{\mathcal{X}}$ and $O^{\mathcal{MU}}$ are variables that can take as value any object of the proper type as required by the relation in which they are given as arguments. In words, the rule says that sensor o_j is compatible with controller o_i IF AND ONLY IF there is a process model for which the controller is designed and that model assumes a state that is measured by the sensor in a measurement unit and the controller requires that specific state to be measured exactly in the same measurement unit. The rule function is executed for every possible combination of specific objects, resulting in:

$$g_1(o_i, o_j) = \begin{cases} 1 & \text{if } o_i \text{ can use the output of } o_j \\ 0 & \text{otherwise} \end{cases}$$

The annotations of the components are either provided by a human or downloaded automatically from a dedicated database of such annotations, similarly to the way the operating systems of computers obtain drivers for peripheral devices.

B. Example scenario

Given that n is the number of states, n_a is the number of measured states ($0 \leq n_a \leq n$), m is the number of control inputs required by the installed actuators and m_a is the number of control inputs produced by the controller ($0 \leq m_a \leq m$), an example usage scenario with SSM and ASM is given next.

The output from the sensors, vector $y \in \mathbb{R}^{m_a}$, is connected to m_a inputs of SSM. Every time a change is identified in the inputs of SSM, the latter requests the semantic annotation of the new sensor. Then, n outputs of SSM (vector $y_s \in \mathbb{R}^n$) are connected to the controller which is also semantically annotated in the same way. Every time there is a change, SSM checks the knowledge base and, if necessary, executes rules to infer whether the output of sensors can be fed directly to the controller, whether a transformation is required on any signal, even whether a signal is missing and must be generated otherwise. The result is that SSM binds specific input signals to specific outputs. E.g. SSM will directly bind the output of a sensor installed in Room 1 that measures temperature in Fahrenheit, as required by the controller, but will first transform it if a new sensor measures temperature in Celsius degrees. On the ASM side, the controller output signal ($u \in \mathbb{R}^{m_a}$) is connected to the inputs of ASM. The outputs of ASM ($u_s \in \mathbb{R}^m$) are connected to the actuators. ASM is assumed to have full knowledge to properly bind its inputs to its outputs. Summarizing, the SSM makes sure that the controller receives as input the appropriate vector signal $y_s \in \mathbb{R}^n$ and the ASM guarantees that the actuators receive the correct vector signal $u_s \in \mathbb{R}^m$.

V. SIMULATION EXAMPLE

The example here adopts the system presented in Section II and shown in Fig. 1 and is only a first step towards developing a semantically enhanced feedback control architecture. Using Newton’s law of cooling, $Q = cA(T_i - T_j)$, $i \neq j$, with Q the heat transfer, c the heat transfer coefficient, A the area of the surface through which the heat flows and T_i, T_j the temperatures in the two sides of the surface, the following linear state-space model of the above closed-loop system is derived:

$$\begin{aligned} \dot{T} &= AT + Bu + DT_0 \\ y &= CT \end{aligned} \tag{1}$$

where:

T is a vector in $\mathbb{R}^3$, representing the temperatures of the three rooms

T_0 is the temperature of the ambient that is measured but not controlled

y is the measurement vector in $\mathbb{R}^{n_a}$, with $0 \leq n_a \leq 3$ the number of measurements that depends on the number of installed sensors in the rooms

u is a vector in $\mathbb{R}^2$, representing the controlled input

A, B, C and D are the state, input, output and ambient temperature transition matrices respectively

The control system is designed with a semantically enhanced architecture as in Fig. 2. At this stage, it is assumed that the system operates without noise affecting the measurements of sensors. Moreover, only SSM is employed, that is, semantic mediation is provided only between the sensors and the controller. To achieve the control objective, that is to regulate the temperature of the rooms to the desired values (vector $r \in \mathbb{R}^3$), a controller of the form $u = -KT + Lr + M$ is employed (K, L, M are design matrices), designed with the assumption that the three states are measured in Fahrenheit degrees.

The sensors and the controller are semantically annotated using the ontology in SSM. The ontology and the knowledge base used for the simulations, were implemented in SWI Prolog [3] while the MATLAB tool was used for the rest of the implementations. For the simulations, the desired temperatures were set to $r = [85\ 72\ 77]^T$ $({}^0F)$ while the maximum power of the electric radiators was $u_{max} = [2000\ 2000]^T\ W$. The ambient temperature was modelled as the absolute of a sine with 40^0F minimum value, 70^0F maximum value and $T_{pr} = 48h$ period .

A. Simulation scenario

Initially it is assumed that three sensors are deployed, measuring the temperatures of the rooms in Fahrenheit degrees. The outputs of the sensors are connected to respective inputs of SSM. In this case, the semantic mediation identifies that no mediation is required and binds the output of all sensors directly to the inputs of the controller.

At time t_0, the sensor deployed in Room 3 stops transmitting data. SSM notices the change and activates a *reduced-order observer* ([19]) that is able to estimate the 3rd state given the underlying system's model and the available two states. Then SSM binds the two available measurements to the respective inputs of the controller and the value produced by the estimator to the remaining input.

At time t_1, a new sensor is deployed and annotated. This new sensor measures the temperature of Room 3 in degrees Celsius. In this case, SSM binds the first two state measurements directly to the inputs of the controller. At the same time it detects and performs the transformation from Celsius to Fahrenheit and binds that value to the remaining controller's input. In any case, the SSM mediation ensures that the controller continues receiving the expected measurements' vector $y \in \mathbb{R}^3$.

B. Results and discussion

Fig. 3 shows the real and estimated values of the regulated temperature of each room when no SSM is employed in the scenario. At $t_0 = 5h$, when the temperature of Room 3 is not measured the controller is assumed to have built-in intelligence to use the last available value. At $t_1 = 29h$, the newly installed sensor measures the temperature of Room 3 in Celsius. It is shown that the real temperatures of rooms 2 and 3 are driven to high values, well above the reference. This is the result of the wrong control instructions given to the electric radiator in Room 3 due to the wrong situation interpretation, as also shown in Fig. 5.

The scenario is executed again, this time with active SSM component. The results are shown in Fig. 4 and Fig. 6. Due to the semantic mediation, the missing temperature is estimated by the observer, thus preventing any unwanted effects. Also, when the new sensor is installed, the "problematic" measured values of the temperature in Room 3 do not affect the performance of the controller. It is shown that even after t_1, the regulation of the temperature of all rooms continues as desired.

As evident from the simulation results, the semantic mediation proved to be highly beneficial for the control system which was given the intelligence to be flexible and remain stable even after an otherwise problematic change that occurred.

VI. CONCLUDING REMARKS

In this paper we introduced a semantically enhanced architecture for feedback control systems. This architecture takes advantage of the knowledge representation capabilities provided by the use of ontologies in order to achieve semantic interoperability of the components. The results of the simple simulation scenario indicate a positive potential of the proposed architecture and, at the same time, generate the need to further investigate certain issues related to the limitations of the approach, performance, robustness and stability. In order to demonstrate the benefits and impact that could be achieved with this technology once mature, there is a number of things to consider (e.g. exploiting the measurements of different physical properties through their relations, exploiting the spatial and temporal properties of components that enter and leave the environment at runtime, etc.).

Finally, it is noted that the development of appropriate ontological models of the controlled environments is a complicated task that involves many challenges (accuracy, scalability, completeness), however, it is still a research area worthy of investing.

ACKNOWLEDGMENT

This work is partially funded by the European Research Council (ERC) under the project ”Fault-Adaptive Monitoring and Control of Complex Distributed Dynamical Systems”.

[3] www.swi-prolog.org

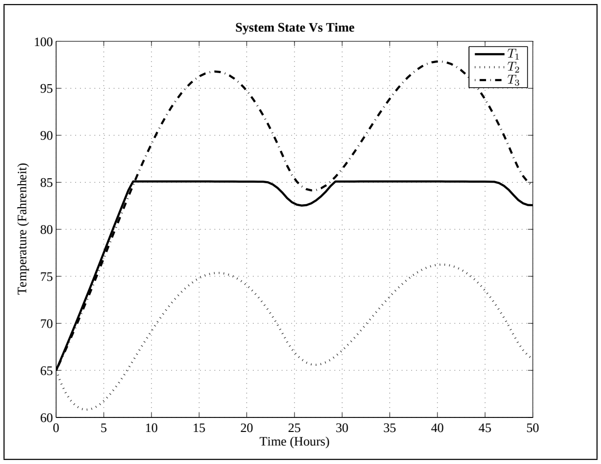

Fig. 3: Temperatures of rooms - No SSM mediation

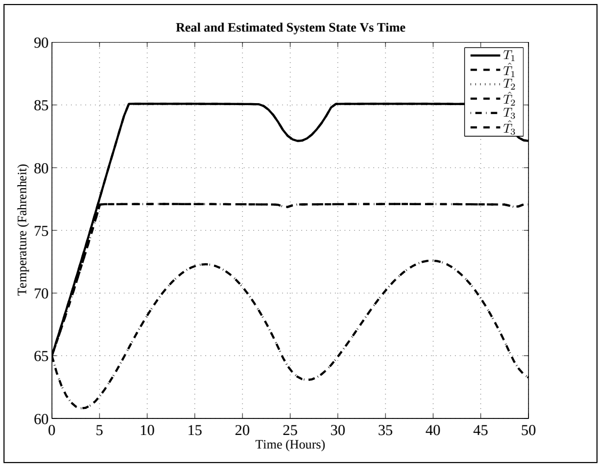

Fig. 4: Real and estimated temperatures of rooms - With SSM mediation

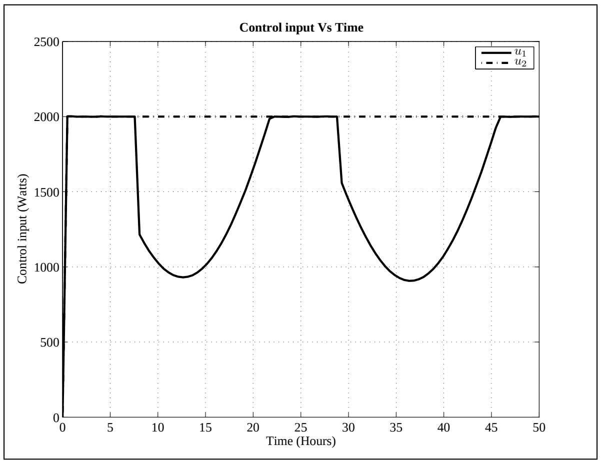

Fig. 5: Thermal power by electric radiators - No SSM mediation

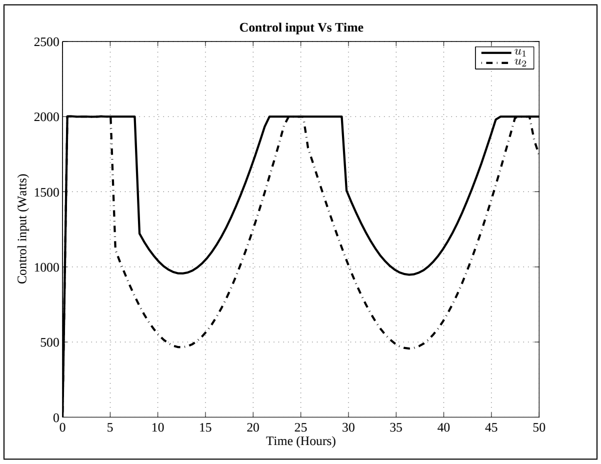

Fig. 6: Thermal power by electric radiators - With SSM mediation

REFERENCES

[1] J. Gertler, *Fault detection and diagnosis in engineering systems*. CRC, 1998.

[2] J. Chen and R. J. Patton, *Robust model-based fault diagnosis for dynamic systems*. Kluwer Academic Publishers, 1999.

[3] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and fault-tolerant control*. Springer Verlag, 2003.

[4] R. Isermann, *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Verlag, 2006.

[5] K. J. Astrom and B. Wittenmark, *Adaptive Control, Addison-Wesley*. Addison-Wesley, 2d ed. ed., 1994.

[6] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[7] J. Farrell and M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*. J. Wiley, 2006.

[8] T. Samad and T. Parisini, "Systems of Systems," in *The Impact of Control Technology* (T. Samad and A. Annaswamy, eds.), available at www.ieeecss.org, 2011.

[9] W. A. H. Thissen and P. M. Herder, "System of Systems Perspectives on Infrastructures," in *System of Systems Engineering* (M. Jamshidi, ed.), ch. 11, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2008.

[10] T. Knudsen, "Awareness and its use in Plug and Play Process Control," *Convergence*, pp. 4078–4083, 2009.

[11] J. Stoustrup, "Plug & Play Control: Control Technology Towards New Challenges," *European Journal of Control*, vol. 15, pp. 311–330, Aug. 2009.

[12] A. Sheth, C. Henson, and S. Sahoo, "Semantic Sensor Web," *IEEE Internet Computing*, vol. 12, pp. 78–83, July 2008.

[13] B. Chandrasekaran, J. Josephson, and V. Richard Benjamins, "What are ontologies, and why do we need them?," *IEEE Intelligent Systems*, vol. 14, pp. 20–26, Jan. 1999.

[14] C. J. Goodwin and D. J. Russomanno, "Ontology integration within a service-oriented architecture for expert system applications using sensor networks," in *Expert Systems*, pp. 409–432, Blackwell Publishing Ltd, 2009.

[15] Y. Lin and H. Ding, "Ontology-based Semantic Annotation for Semantic Interoperability of Process Models," in *Science And Technology*, 2005.

[16] D. Fensel, E. Motta, S. Decker, and Z. Zdrahal, "Using Ontologies For Defining Tasks, Problem-Solving Methods and Their Mappings," 1997.

[17] N. Holger and M. Compton, "The Semantic Sensor Network Ontology : A Generic Language to Describe Sensor Assets," 2009.

[18] M. Botts, G. Percivall, C. Reed, and J. Davidson, "OGC Sensor Web Enablement : Overview And High Level Architecture," in *5th International ISCRAM Conference*, no. May 2008, (Washington, DC, USA), pp. 713–723, F. Fiedrich and B. Van de Walle, 2008.

[19] D. Luenberger, "Observers for multivariable systems," *Automatic Control, IEEE Transactions on*, vol. 11, pp. 190–197, Apr. 1966.