A Cognitive Agent Architecture for Feedback Control Scheme Design

Georgios M. Milis, Demetrios G. Eliades, Christos G. Panayiotou, and Marios M. Polycarpou
KIOS Research Center for Intelligent Systems and Networks
Department of Electrical and Computer Engineering
University of Cyprus
Nicosia, Cyprus
Email: {milis.georgios, eldemet, christosp, mpolycar}@ucy.ac.cy

***Abstract*—We present a novel architecture for the design of feedback control schemes, aiming to automate the cognitive process performed by human experts when designing control schemes for certain systems. The work starts with the identification of types of cyber-physical modules participating in a feedback control scheme. Each module is defined as a functional mapping between inputs-vectors and an output vector. The inputs and outputs are then mapped to a semantic space formed by linguistic variables that model the expert knowledge. These semantic annotations of the modules are then exploited by a Cognitive Agent with semantic reasoning capabilities, to achieve the online configuration of a feedback control scheme, given a set of specifications and a database of available modules' implementations. The adopted knowledge modelling and reasoning techniques are driven by past efforts of the World Wide Web Consortium on the semantic composition of Web services. The applicability of the method is tested via a paper-based simulation of a use-case from the smart buildings domain.**

I. Introduction

The recent efforts towards the Internet-of-Things (IoT) paradigm, lead to the design of cyber-physical systems [1], which consist of the physical-engineered modules and the cyber modules that offer advanced internet-enabled communication and computation capabilities. The characteristics of these modules turn them into promising candidates for the adoption of large-scale systems monitoring and control applications, where the topology and dynamics of the systems are complex and difficult to handle with traditional monolithic architectures. Typically, there is a need for smaller modules to work autonomously to accomplish certain tasks in some part of the system and effectively co-operate with the rest of the system as to address higher-level and heterogeneous monitoring and control challenges [2].

Today's engineered systems more and more employ several cyber and physical modules, like sensors for monitoring system states, electrical and mechanical actuators, controllers and a number of other software tools for signal processing, estimations and online-learning tasks. Utilising the capabilities of these modules in implementing efficient systems' monitoring and control, can considerably help in saving energy, reducing economic cost and improving societal welfare. A great challenge, however, faced in relation with these systems is the fact that the utilized modules may need to change during operation, due to replacements, availability of new technologies or developments of new monitoring and control capabilities.

During the last two decades, various methodologies have been developed and proposed for system monitoring and control, ranging from applications of the classical control theory of linear systems to techniques with control law adaptation capabilities [3], to methodologies with online learning capabilities of unknown dynamics, which combine model-based analytical redundancy and computational intelligence tools, i.e. neural networks, radial basis functions, etc. [4]. Going one step further, the control community has also been addressing the online re-configurability challenge of control systems [5], including also the identification of newly introduced dynamics in a plug-and-play fashion [6], [7].

Designing a feedback control scheme for a certain system is a complicated procedure which relies on the knowledge and (semantic) reasoning capabilities of human experts. In practice, a human expert should have a broad background knowledge of tools (e.g., classical control design methods, non-linear system control methods, online learning methods based on computational intelligence, state-estimation methods, etc.) and in which situations these are best suited, in order to make an informed selection that fully exploits the available measurements, plant and actuation constraints. However, it is very rare, if not impossible, to find and employ a human expert of such breadth of knowledge whenever a feedback control scheme is required for a certain system. An additional drawback in current practices is the lack of mechanisms to allow online (and where possible automatic) replacement of individual modules or of the overall control scheme.

The explicit incorporation of semantics in the control systems' design has been proposed in 1988 by Prof. Ervin Rodin in its article titled "Semantic Control Theory" [8]. Prof. Ervin proposed an architecture, which allows the control system to reason upon the modelled knowledge and select appropriate controllers from a pre-existing pool. A more system-theoretic and philosophical approach to presenting the concepts of semantic control systems can be found in [9]. More recently, concrete examples of using ontological (semantic) knowledge models are observed. For instance in the smart buildings context, *DOGont* [10] deals with the current issues of domotic environments, that is, the existence of many vendors, each with

separate not compatible solutions, the existence of different technologies, different protocols, different device features, etc. Moreover, an early effort introducing the need for a modular architecture for the control system design, has been discussed in [11]. In the Computational Intelligence domain, a significant milestone was the approval of the IEEE Standard for Fuzzy Markup Language (IEEE Std 1855-2016), which specifies an interoperability framework for fuzzy logic controllers [12], [13].

The above motivate our work for the design of an architecture and a more systematic methodology which is able to utilise expert knowledge and cognitive reasoning based on semantics, to reproduce part of the reasoning procedure of a human expert. This would allow new modules to be gradually deployed as they become available, by automatically configuring the feedback control scheme. The aim is to achieve a syntactically and semantically valid composition and to facilitate interoperability with the other available modules using a common framework for efficient exchange of data and knowledge. The concept is driven by past work for the semantic composition of purely cyber entities (Web services) [14]. The envisioned impact from the adoption of a module-based feedback control design, is the faster exploitation, testing and demonstration of academic research results (such as new learning, control and optimisation algorithms), in industrial applications. To demonstrate the application of the proposed architecture, we present a paper-based simulation of a use-case where a Cognitive Agent designs feedback control schemes for smart-building-related plants. It is emphasized that our work focuses on the online configuration of feedback control schemes, using existing modules, and does not focus on the design of any new feedback control algorithm or module.

The paper is organized as follows: Section II formulates the problem by revisiting the feedback control theory from the literature and breaking the feedback control scheme into a set of individual modules. Section III then presents the design of the proposed architecture, the knowledge space with the semantic annotations of the modules and the Cognitive Agent which utilizes semantic reasoning towards the automatic configuration of feedback control schemes. Section IV presents the use-case. Finally, Section V concludes the paper and discusses future directions.

II. Problem formulation

Typically, a feedback control scheme is specifically designed for a certain system/plant, taking into account the system's measurable variables, known dynamics, available actuation capabilities, as well as other relevant information. The output of the feedback control scheme is the action applied on the system via the system inputs. In a discrete-time implementation, the system input $v(k)$, at time k, is a vector of signals produced by the actuators. In the general case, the feedback control scheme implementation, can be considered as composed of sub-modules, some of which are basic (mandatory) for all implementations of feedback control schemes while others are required only in certain cases. These are discussed in the sequel, adopting the approach introduced in [15] for the module-based analysis of fault-detection schemes.

A. Basic Modules

First of all, a control scheme is always implemented to offer a service on a specific plant, therefore the first basic module of a feedback control scheme is the "Plant" itself. The dynamics of the plant, adopting a discrete time formulation, are generally described by $x(k) = f_p(x(k-1), v(k), w(k), \phi(k), h(k); \zeta_p)$, where $x(k) \in \mathcal{R}^n$ is the vector of state-variables describing the plant, $f_p(\cdot)$ is the function representing the plant's dynamics, $x(k-1)$ is the vector of system-state memory, $v(k) \in \mathcal{R}^m$ is the input signal produced by controlled actuators, $w(k)$ is the input signal produced by uncontrolled sources, $\phi(k)$ is a signal modelling faults introduced in the plant's dynamics, $h(k)$ is the input signal produced by third interdependent systems and ζ_p is a set of other plant parameters.

The second basic module of a feedback control scheme is the "Actuator", given by $v(k) = f_a(u(k); \zeta_a)$, where $v(k) \in \mathcal{V}$ is the system input signal discussed earlier, $f_a(\cdot)$ is the implementation of a function that produces the signal acting on the controlled system, $u(k)$ is a computed signal that drives the action and ζ_a is a set of other parameters required by the available actuation implementation.

The third basic module in a feedback control scheme is the "Controller" given by $u(k) = f_c(y(k), r(k), \hat{x}(k), \hat{g}_p(k); \zeta_c)$, where $u(k)$ is the control decision signal defined earlier, $f_c(\cdot)$ is the implementation of a control method to derive the signal, $y(k)$ is the signal representing the plant's feedback as given to the controller, $r(k)$ is the desired system state trajectory, $\hat{x}(k)$ is the estimated system state (optionally used), $\hat{g}_p(k)$ is the estimated value of unknown plant dynamics (optionally used) and ζ_c is a set of parameters required by the adopted controller implementation.

As an example, the control function f_c may be a *bang-bang* controller that compares the measured states $y(k)$ with the desired states $r(k)$ and returns a vector of binary signals indicating whether the measurements are greater than the desired values or not. Another example may be a fuzzy control implementation, where the control decision is the defuzzification of a linguistic value, e.g., "fast", which was the output of the triggering of a set of fuzzy rules on the fuzzified system output.

A fourth basic module of a feedback control scheme is the "Sensor", which undertakes the task of measuring the state of the system and is given by the function $y(k) = f_s(x(k), v(k), w(k), \phi(k), h(k); \zeta_s)$, where $y(k)$ is the signal produced by the installed sensing devices, $f_s(\cdot)$ is the implementation of the system measuring given the available sensing devices, $x(k)$ is a vector of the system states, $v(k), w(k), \phi(k), h(k)$ are the various plant's input vectors discussed earlier and ζ_s is a set of parameters required by the adopted implementation of the sensing. For instance, the sensing parameters may correspond to measurement accuracy

given by manufacturer or location of the device derived from expert knowledge about the system operation, etc.

In summary, at a minimum, the feedback-control scheme is composed of the modules specified above, forming a composite function $f_a \circ f_c \circ f_s$. That is, the input to the system is a function of the control decision which in turn is a function of the system measurements.

B. Advanced Modules

In addition to the four basic ones, additional modules may be required by certain feedback-control schemes. For instance, the estimation signal of the system states, $\hat{x}(k)$, may be computed by a separate module. In that case, a "State-Estimation" module can be considered, given by $\hat{x}(k) = f_e(\hat{x}(k-1), \hat{g}_p(k), y(k), u(k); \zeta_e)$, where $\hat{x}(k)$ is the estimated system states signal at the current time step, $f_e(\cdot)$ is the adopted implementation of the state-estimation, $y(k)$ and $u(k)$ are the vectors of system's measured output and known control signal respectively, ζ_e is a set of other parameters required by the adopted implementation, $\hat{x}(k-1)$ is the vector of estimated previous system state, and $\hat{g}_p(k)$ is the estimated value for unknown plant dynamics (if required). For instance, the State-Estimation module may correspond to a "Kalman filter" which produces estimates based on some prior knowledge about the states, a measurement vector and certain parameters of measurement and state's uncertainty; it can also be a "Luenberger observer" which, based on a known model of system dynamics and the available measurements, produces estimates of the state.

Furthermore, in the case of having a system model with unknown dynamics $g_p(\cdot)$ (part of $f_p(\cdot)$), a "Learning Module" can be utilized, to learn the unknown function using a suitable approximation structure (e.g., neural network, polynomial function, radial-basis functions, wavelets, etc.), such that $\hat{g_p}$ approximates g_p. This module can be described in general by $\hat{g}_p(k) = f_\theta(y(k), u(k); \zeta_\theta)$, where $\hat{g}_p(k)$ is the estimated value of the unknown function, $f_\theta(\cdot)$ is the adopted online learning implementation and ζ_θ are any other parameters required by the adopted implementation (e.g., the convergence rate, knowledge about the structure of the function).

In some cases, the measured system output needs to be processed by a separate module before being fed to the controller. For instance, if a system state is measured by more than one sensor, we may want to fuse the measurements and use the computed signal in the controller; alternatively this could correspond to data validation/reconstruction. This step is undertaken by a "Pre-control Function", defined as $y(k) = f_y(y_a(k); \zeta_y)$, where $y(k)$ is now the processed system output, $f_y(\cdot)$ is the adopted measurement processing implementation, $y_a(k)$ is the actual sensor measurements and ζ_y are any other parameters required by the adopted processing implementation (e.g., knowledge about the proximity of devices to the state location). Then, the controller implementation $f_c(\cdot)$ receives as input the signal produced by the function $f_y(\cdot)$ instead of the actual measurement $y_a(k)$.

Similarly, the output of the controller, $u(k)$ may need to be processed before fed to the actuators. For instance, consider the case where a single control signal needs to drive two actuators. This can be implemented by a "Post-control Function", given by $u_a(k) = f_u(u(k); \zeta_u)$, where $u_a(k)$ is the processed control decision, $f_u(\cdot)$ is the adopted control signal processing implementation, $u(k)$ is the actual control signal and ζ_u are any other parameters required by the adopted processing implementation (e.g., knowledge about the type of actuation devices). Then, the actuators receive the signal $u_a(k)$ instead of the control signal $u(k)$.

C. Modules Database

All implementations of modules (functions) of the types discussed above, can be considered as being elements of a set $\mathcal{F}$, thus forming a database of modules. The set $\mathcal{F}$ is a super-set of the finite-cardinality type-sets of modules, as:

$\mathcal{F} = \mathcal{F}_p \cup \mathcal{F}_a \cup \mathcal{F}_c \cup \mathcal{F}_s \cup \mathcal{F}_e \cup \mathcal{F}_\theta \cup \mathcal{F}_y \cup \mathcal{F}_u$

In addition, it can be seen from above analysis that all modules are essentially functional mappings between certain inputs to certain outputs. The sets of inputs and outputs of a module are defined here as $\mathcal{T}^{(\omega)}$ and $\mathcal{O}^{(\omega)}$ respectively, where ω is the module. For instance, the inputs set of a specific sensor implementation f_s^1 is given by $\mathcal{T}^{(f_s^1)}$, containing all individual inputs and parameters from the vectors $x(k), v(k), w(k), \phi(k), h(k); \zeta_s$. The outputs set is given by $\mathcal{O}^{(f_s^1)}$ and contains the elements of the vector $y(k)$. The feedback control scheme considers a fixed choreography of types of modules. That is, the plant's output will always be measured by sensing devices, the pre-control processing functions will always use the sensors' output as input and they will give input to a controller, and so on.

D. Cognitive Agent

Depending on the system and the given specifications, an expert engineer would have selected and designed a feedback-control system using specific instances of the basic modules and possibly utilizing additional modules from the advanced ones, as well as specific domain knowledge expertise. In other words, to make a decision, the expert engineer relies on reasoning which considers the available knowledge about the domain and the feedback-control engineering, including the associated semantics of each module.

The challenge addressed in this work is to design an architecture with the ability to utilize pre-modeled expert knowledge and a set of feedback-control specifications for the online design and configuration of a suitable feedback-control scheme, for a large class of systems. The decision about the configuration of the scheme, can be formulated as:

$$\sigma = f_\sigma(\mathcal{G}, \mathcal{S}) \tag{1}$$

where σ is a decision vector that models the selection of specific modules from the subsets of $\mathcal{F}$ defined earlier, $f_\sigma(\cdot)$ is a function implementing the reasoning and the decision about the scheme configuration, $\mathcal{G}$ is the available experts'

knowledge space modelled as a graph, $\mathcal{S}$ is the set of feedback-control (semantic) specifications given to the function (e.g., the characteristics of the desired state). The elements of σ are indexes of the sets of modules defined earlier, such that $\sigma = [\sigma_{f_p}, \sigma_{f_a}, \sigma_{f_c}, \sigma_{f_s}, \sigma_{f_e}, \sigma_{f_\theta}, \sigma_{f_y}, \sigma_{f_u}]^\top$.

Then the configuration of the feedback control scheme can be defined as the operator:

$$f : \Sigma \times \mathcal{F} \mapsto \mathcal{I} \tag{2}$$

where f is a generic mapping operator, Σ is the space of configuration decisions, $\mathcal{F}$ is the space of all available modules and $\mathcal{I}$ is the space of all configurations of the feedback-control scheme.

The next sections provide details about the proposed architecture for the implementation of the functions f_σ and f, emphasizing on the knowledge space and the reasoning mechanism.

III. Architecture Design

The feedback-control architecture, which implements the functions described in the previous section, is depicted in Fig. 1. The top-part of the figure illustrates the *Plant* on which the feedback-control service is provided. The plant has a controlled input vector $v(k)$, an internal state vector $x(k)$ and dynamics described by a function from the set $\mathcal{F}_p$. The plant may be also affected by uncontrolled inputs, i.e. disturbances (vector $w(k)$), faults (vector $\phi(k)$) and inter-dependencies with other plants (vector $h(k)$).

The middle-part of the figure shows the composite *Feedback Control Scheme*. The input to the feedback-control scheme from the plant, is the vector of measurable properties of the system; we assume these to be the plant's states $x(k)$. The output is the signal $v(k)$ produced by the actuators and affecting the plant's states. It is noted that the white boxes (functions) refer to the sets of available modules of each type, as described in Section II and not to specific instances. The selection of specific implementations is performed through the decision signal σ given by the Cognitive Agent (blue dashed line).

The bottom layer illustrates the design of the *Cognitive Agent*. As discussed in Section II, the Cognitive Agent function f_σ utilizes the stored knowledge in the Knowledge Graph $\mathcal{G}$ (including the semantic annotation/characterization of the described modules), as well as any given specifications $\mathcal{S}$, and produces a decision as to what instances of modules to adopt for the feedback control. The decision signal σ is passed to the function $f(\cdot)$ that invokes the selected instances found in the database of modules $\mathcal{F}$. An important added-value feature, is that the databases can be also enriched online, via internet-based remote services or directly by human experts, through appropriate interfaces.

A. Knowledge Space and Cognitive Reasoning

Give the analysis performed earlier, the task of the Cognitive Agent is to utilize pre-defined knowledge about the individual

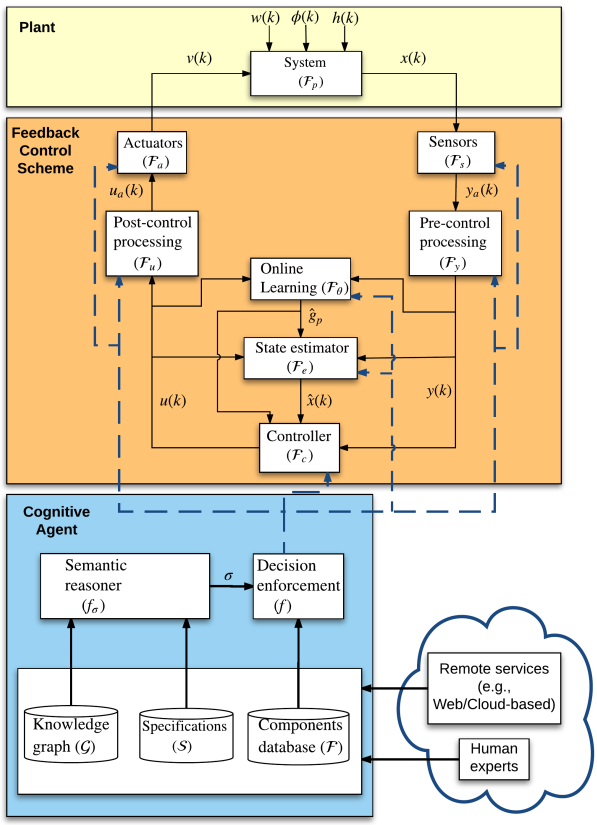


Fig. 1. Block diagram of the architecture. Top: the plant on which feedback control service is provided. Middle: the Feedback Control Scheme. Bottom: the Cognitive Agent

modules and their operation environment and check the matchings between their outputs and inputs in a pipeline fashion. The matching is performed at a semantic level and is therefore called "semantic matching". In order to check for a semantic matching of an output to an input, the Cognitive Agent first explores the mappings of the output and input variables to an "(expert) knowledge space". These are called "semantic annotations" and are defined as: $\lambda : \mathcal{T}^{(\omega)} \mapsto \Lambda \subset \mathcal{G}$, for inputs and correspondingly for outputs, where Λ is the set of all possible semantic annotations of modules in the knowledge space.

The semantic matching is defined as another function, $\rho : \Lambda \times \Lambda \mapsto \{True, False\}$. That is, the semantic matching function takes as input a pair of semantic annotations (of an output and an input) and returns "True" if the matching is confirmed and "False" otherwise.

What remains to be defined is the (semantic) knowledge space $\mathcal{G}$ and the actual meaning of the semantic annotation and semantic matching functions. An early version of the knowledge space has been defined, adopting the theory of bipartite graphs, in recent publications by the same authors [16], [17], with case-studies from the "Water Distribution Network" and "Smart Grid" domains. In this paper we extend the work adopting a fundamentally similar but formulation-

wise different approach. We model the expert knowledge with linguistic variables that take values from discrete sets of linguistic terms. For the purposes of this work, the semantic annotation of input and output variables is performed using three properties to which they refer to: "location", "physical quality" and "measurement unit". These correspond to three linguistic variables: $l \in \mathcal{L}$, where $\mathcal{L}$ is the space of possible locations, $q \in \mathcal{Q}$, where $\mathcal{Q}$ is the space of possible physical qualities, and $d \in \mathcal{D}$, where $\mathcal{D}$ is the space of possible measurement units. The combination of these three spaces, defines a 3-dimensional knowledge (discrete) space ($\mathcal{G}$), the elements of which are the semantic annotations of inputs and outputs.

The above is clarified in the sequel through an illustrative example. Consider a plant representing a one-room office with its temperature state. A feedback control scheme is deployed in the office, comprising an electric heater introducing heat energy in the office, a sensor measuring the temperature in degrees Celsius and a simple proportional controller that receives the measurement, compares it to a pre-defined desired value and decides the level on which to operate the heater. In this example we have three modules: a sensor f_s^1, an actuator f_a^1 and a controller f_c^1. Also, the plant dynamics are represented by f_p^1. The sensor has an inputs-set $\mathcal{T}^{(f_s^1)}$ and an outputs-set $\mathcal{O}^{(f_s^1)}$. The respective sets for the other modules are defined in the same way. In addition, expert knowledge is assumed defining these linguistic spaces: $\mathcal{L} = \{l_1 : \text{'office'}, l_2 : \text{'ambient'}, l_3 : \text{'ceiling'}\}$, $\mathcal{Q} = \{q_1 : \text{'temperature'}, q_2 : \text{'heat-energy'}, q_3 : \text{'on-off'}\}$ and $\mathcal{D} = \{d_1 : \text{'Celsius'}, d_2 : \text{'Fahrenheit'}, d_3 : \text{'kW'}, d_4 : \text{[1,0]}\}$. Relations (mappings) between the linguistic terms are also defined. These are formulated with matrix operators, the elements of which take the value 1 if the two terms are semantically related and the value 0 otherwise. For instance, we define the following operators:

$$M_1 : \mathcal{L} \mapsto \mathcal{L} = \begin{array}{c} \\ l_1 \\ l_2 \\ l_3 \end{array} \begin{array}{c} \begin{array}{ccc} l_1 & l_2 & l_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \end{array},$$

$$M_2 : \mathcal{L} \mapsto \mathcal{L} = \begin{array}{c} \\ l_1 \\ l_2 \\ l_3 \end{array} \begin{array}{c} \begin{array}{ccc} l_1 & l_2 & l_3 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array},$$

$$M_3 : \mathcal{D} \mapsto \mathcal{Q} = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array},$$

where M_1 models the relation "is-part-of" or "contains" between two locations, M_2 models the relation "is-adjacent-to" between two locations and M_3 models the relation "is-measurement-unit-of" or "is-measured-in-unit" between measurement units and physical qualities.

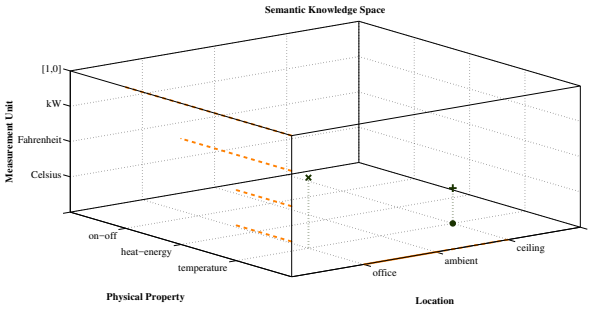


Fig. 2. The current knowledge space, showing the semantic mapping of inputs and outputs of the sensor f_s^1. Input: filled circle mark; Output: Cross '+' mark; Semantic mappings: dashed lines; Specifications: 'x' mark

Given the above described knowledge space, Fig. 2 illustrates the semantic annotation of the input and output of sensor f_s^1, as example. The sensor has one input, t_s, with semantic annotation $\lambda(t_s) = \{l_3 : \text{'ceiling'}, q_1 : \text{'temperature'}\}$, modelling the fact that the sensor is installed on the ceiling of the room and senses the physical quality "temperature". This is shown on the figure with a green-filled circle mark at the specific point in space. The output of the sensor, o_s, has a semantic annotation $\lambda(o_s) = \{l_3 : \text{'ceiling'}, q_1 : \text{'temperature'}, d_1 : \text{'Celsius'}\}$, modelling the fact that the sensing signal is further given in degrees Celsius. The output annotation in the knowledge space is shown by a green cross '+' mark. For completeness, the defined semantic mappings modelled by the operators M_1, M_2, M_3 above, are shown by yellow dashed lines. If two points are connected, it means that if the expert knows something modelled by the first point, the knowledge modelled by the second point can be inferred automatically. Finally, we assume that the desired office temperature has been set to 73^oF, which transforms to the following set of (semantic) specifications for the feedback control scheme: $\mathcal{S} = \{l_1 : \text{'office'}, q_1 : \text{'temperature'}, d_2 : \text{'Fahrenheit'}\}$. The specifications are shown on the figure with a green 'x' mark.

B. Semantic Reasoning Algorithm

Having defined the above, the reasoning is implemented by the algorithm outlined below:

- Start with a set of "Actuator" modules that do not violate the specifications.
- Find a "Controller" module with outputs that match semantically with the "Actuators" inputs. If there are remaining inputs and/or required parameters, check whether matching can be achieved by the intervention of "Post-control Function" module(s).
- For the selected controller, find all "Sensor" modules that produce outputs that match with the controller's input. Check also "Pre-control Function" modules.
- In all cases, check if specifications are met by the selections, otherwise re-iterate within modules.
- If all successful, create the decision signal σ such as to enforce the selection of the matching modules.

This work neither focuses on the selection of modules based on their performance nor does it consider any sensitivity analysis of the selections. The Cognitive Agent will implement the feedback control scheme using the first set of modules that will be found matching. In case of inability to close the loop with matching modules, the algorithm terminates and informs inability to configure a feedback control scheme with currently available modules. Moreover, it is noted that the method is currently applied for the configuration of control schemes in plants with slow dynamics, where the time required for the re-configuration of the scheme does not affect the stability characteristics. The execution of the algorithm will be clarified through a use-case in the next section.

IV. USE-CASE

We consider a use-case where the Cognitive Agent's Semantic Knowledge Space is provided as in Table I. Then, the

TABLE I
CURRENT STATUS OF SEMANTIC KNOWLEDGE SPACE

Specifications	$\mathcal{S} = \{l_1$: 'office', q_1 : 'temperature', d_2 : 'Fahrenheit'$\}$
Plant f_p^1	Inputs: $\mathcal{T}^{(f_p^1)} = \{\{l_1$: 'office', q_2 : 'heat-energy'$\}\}$
	Outputs: $\mathcal{O}^{(f_p^1)} = \{\{l_1$: 'office', q_1 : 'temperature'$\}\}$
Sensor f_s^1	Inputs: $\mathcal{T}^{(f_s^1)} = \{\{l_3$: 'ceiling', q_1 : 'temperature'$\}\}$
	Outputs: $\mathcal{O}^{(f_s^1)} = \{\{l_3$: 'ceiling', q_1 : 'temperature', d_1 : 'Celsius'$\}\}$
Controller f_c^1	Inputs: $\mathcal{T}^{(f_c^1)} = \{\{l_1$: 'office', q_1 : 'temperature', d_1 : 'Fahrenheit'$\}\}$
	Outputs: $\mathcal{O}^{(f_c^1)} = \{\{l_1$: 'office', q_3 : 'on-off', d_4 : [1,0]$\}\}$
Actuator f_a^1	Inputs: $\mathcal{T}^{(f_a^1)} = \{\{l_3$: 'ceiling', q_3 : 'on-off', d_4 : [1,0]$\}\}$
	Outputs: $\mathcal{O}^{(f_a^1)} = \{\{l_3$: 'ceiling', q_2 : 'heat-energy', d_3 : 'kW'$\}\}$

Cognitive Agent starts the execution of its semantic reasoning algorithm to configure the feedback control scheme. The algorithm's steps are illustrated in Fig. 3a. All outputs of modules are marked with the cross sign and all their inputs are marked with circles. Moreover, each module is shown with different colour and their inputs and outputs are connected by same colour lines, modelling their internal transfer functions. Finally, the modules are numbered in the order they are explored by the algorithm, which helps following the checks for output-input semantic matchings. It can be seen that the semantic annotation of the actuator's output (cross 2) matches the semantic annotation of the plant's input (circle 1) if projected on the "Locations-Physical Property" plane and then move along the semantic relation, which models that the "ceiling" "is-part-of" the "office" (defined by M_1 operator). Then, the semantic annotation of the controller's output (cross 3) matches the semantic annotation of the actuator's input (circle 2) by considering the relation "contains" defined by the M_1 operator. Continuing, the semantic annotation of the sensor's output (cross 4) cannot match with the semantic annotation of the controller's input (circle 3), since there is no semantic path to connect them. At this stage, although the rest of the matchings are confirmed, the algorithm terminates stating inability to configure a feedback control scheme with the available modules.

However, we consider that the database of modules is enriched online with a pre-control processing function f_y^1 which converts temperature degrees Celsius to degrees Fahrenheit, as shown in the Table II.

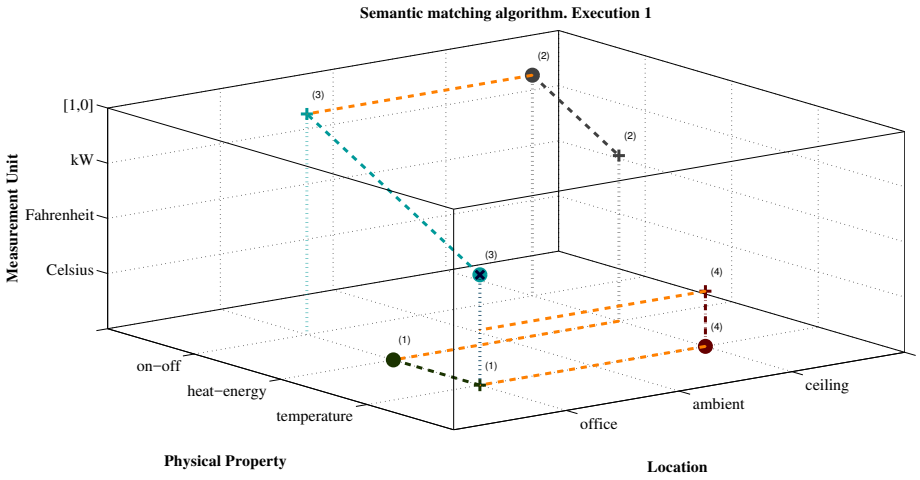
TABLE II
EXTENDING THE KNOWLEDGE SPACE WITH ADDITIONAL KNOWLEDGE

Pre-control proc. f_y^1	Inputs: $\mathcal{T}^{(f_y^1)} = \{\{d_1$: 'Celsius'$\}\}$
	Outputs: $\mathcal{O}^{(f_y^1)} = \{\{d_2$: 'Fahrenheit'$\}\}$

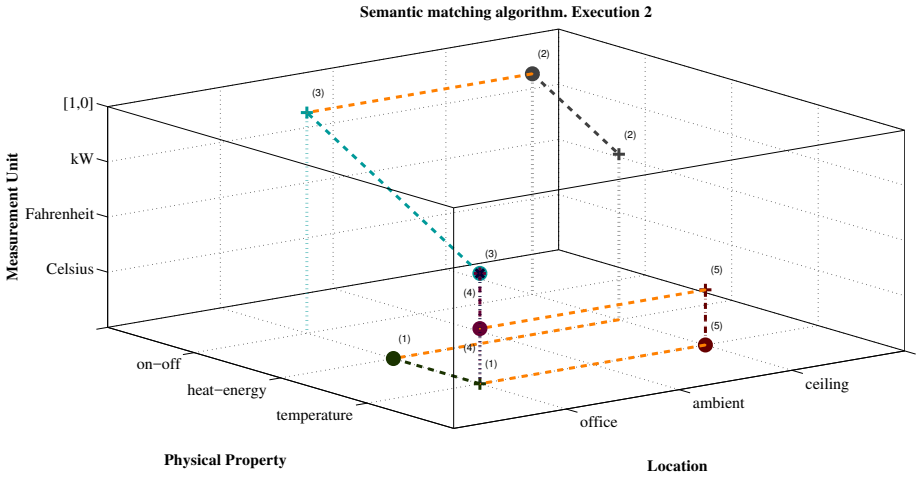
Fig. 3b shows the semantic annotations of the inputs and outputs of the additional module. It can be seen that the input of the controller (circle 3) can be now connected with the output of the function f_y^1 (cross 4) since the latter transforms the signal in relation to the measurement unit. In the same way, the output of the sensor (cross 5) can be connected to the input of the function (circle 4), exploring the semantic relation "is-part-of" between the ceiling and the office. We can also see that the specifications (marked with 'x') coincide with an output-input semantic matching, which means that the conformance to them is taken into consideration by the system. The algorithm is therefore able to initially configure ($\mathcal{I} = 0$) the feedback control scheme using the set of modules $\{f_p^1, f_a^1, f_c^1, f_y^1, f_s^1\}$.

At some future time $k+N$ of the control system operation, the plant becomes equipped with a sensor f_s^2 that measures the opening of a window on the office wall in "percentage". In addition, a new controller (f_c^2) is uploaded in the modules' database, which implements fuzzy logic decision based on temperature estimation and window opening inputs. For instance, the controller may define appropriate linguistic variables for the temperature, e.g., "high"/"low" and the window opening, e.g., "little"/"much", so as to make decision for operating the heating device at "low"/"high" level. A Luenberger observer [18] f_e^1 is also uploaded in the database, able to implement state-estimation for the plant and already considering internally the transformation of the temperature measurement units. The pre-control function that transforms Celsius to Fahrenheit is removed from the database to facilitate clarity of the presentation. The details of the implementations of the modules are not the focus of this work. Finally, a new specification is introduced, requiring to "consider the opening of the window". New semantic annotations are introduced in the knowledge space, as indicated in III.

Figure 4 illustrates the execution of the algorithm given the newly introduced changes in the modules. It can be seen that nothing changes with the matching of the actuator's output to the plant's input. Then, the algorithm infers that both controllers' outputs are candidates to connect to the input of the actuator (see cross 3,3b). Moreover, the inputs of both controllers (red circles 3,3b) are matched with outputs



(a)



(b)

Fig. 3. Illustration of semantic annotations and matchings

TABLE III
SECOND KNOWLEDGE SPACE EXTENSION

Sensor f_y^2	Inputs: $\mathcal{T}^{(f_y^2)} = \{\{l_4$: 'window', q_4 : 'opening'$\}\}$
	Outputs: $\mathcal{O}^{(f_y^2)} = \{\{l_4$: 'window', q_4 : 'opening', d_5 : 'percentage'$\}\}$
Controller f_c^2	Inputs: $\mathcal{T}^{(f_c^2)} = \{\{l_1$: 'office', q_1 : 'temperature', d_2 : 'Fahrenheit'$\}, \{l_4$: 'window', q_4 : 'opening', d_5 : 'percentage'$\}\}$
	Outputs: $\mathcal{O}^{(f_c^2)} = \{\{l_1$: 'office', q_3 : 'on-off', d_4 : [1,0]$\}\}$
State-estimator f_e^1	Inputs: $\mathcal{T}^{(f_e^1)} = \{\{l_1$: 'office', q_1 : 'temperature', d_1 : 'Celsius'$\}\}$
	Outputs: $\mathcal{O}^{(f_e^1)} = \{\{l_1$: 'office', q_1 : 'temperature', d_2 : 'Fahrenheit'$\}\}$
Specification $\mathcal{S}$	$\{l_4$: 'window', q_4 : 'opening'$\}$

from available sensors (see light blue cross 5b and brown cross 5 through the state estimation module 4b). However, the only option that satisfies also the specification at point $\{l_4$: 'window', q_4 : 'opening'$\}$ is the selection of controller f_c^2 which uses the measurement of the window opening as well. Therefore, the Cognitive Agent terminates the execution with the successful configuration of the feedback control scheme ($\mathcal{I} = 1$) using the set of modules $\{f_p^1, f_a^1, f_c^2, f_e^1, f_s^1, f_s^2\}$.

In the same way, the Cognitive Agent is able to use more complex knowledge spaces and semantic operations in order to explore the semantic knowledge space and the respective modules' semantic annotations and subsequently configure the feedback control schemes.

V. CONCLUDING REMARKS AND FUTURE PLANS

The work presented in this paper, introduced an effort to automate the design process of a feedback control scheme. The expert engineering and domain knowledge have been modelled using well established knowledge representation techniques and an agent has been designed with the ability to reproducing part of the cognitive process and reasoning performed by the engineer when designing the feedback control scheme. The

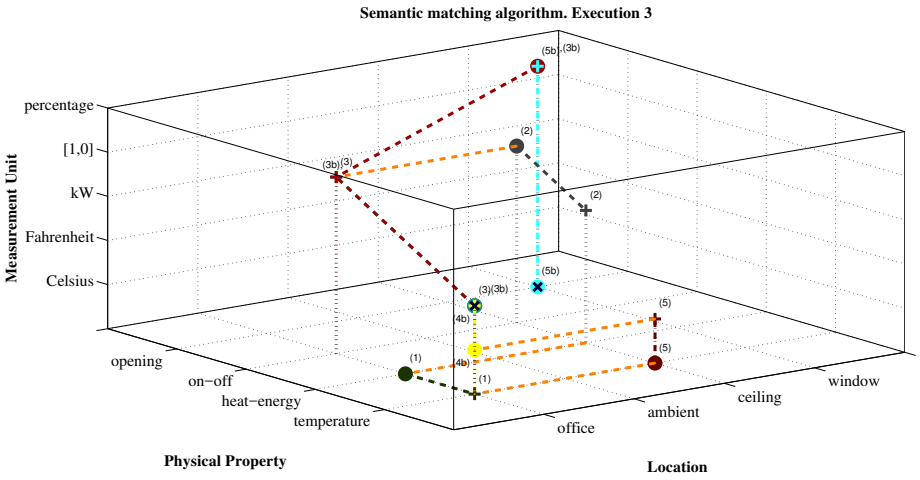


Fig. 4. Illustration of the third semantic matching

primary impact of these results is the fact that it defines clear semantic interfaces between parts of the feedback control process and enables industrial set-ups and/or academic prototypes to allow online plugging-in of new implementations of modules. Immediate future plans foresee the enrichment of the knowledge space to facilitate more complex reasoning tasks, the development of techniques to infer knowledge from data and encode it in the knowledge space and the development of a web platform prototype offering the described services of the Cognitive Agent. Furthermore, depending on the modelled expert knowledge, the Cognitive Agent will support the configuration of feedback control schemes in heterogeneous domains (e.g., water distribution networks, electric power grid, smart buildings). We aim at exploring how knowledge from one domain and/or configuration decisions in one domain can affect the configuration of control systems in another domain.

ACKNOWLEDGMENT

This work is partially funded by the European Research Council (ERC) under the project ERC-AdG-291508 "Fault-Adaptive Monitoring and Control of Complex Distributed Dynamical Systems" (FAULT-ADAPTIVE).

REFERENCES

[1] P. J. Antsaklis, B. Goodwine, V. Gupta, M. McCourt, Y. Po Wu, M. Xia, H. Yu, and Z. Feng, "Control of cyberphysical systems using passivity and dissipativity based methods," *Eur. J. Control*, vol. 19, no. 5, pp. 379–388, 2013.

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

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

[4] J. Farrell and M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*, N. J. W. Hoboken, Ed. J. Wiley, 2006.

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

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

[7] J. Stoustrup, "Plug & Play Control: Control Technology Towards New Challenges," *European Journal of Control*, vol. 15, no. 3-4, pp. 311–330, Aug. 2009. [Online]. Available: http://ejc.revuesonline.com/article.jsp?articleId=13584

[8] E. Rodin, "Semantic control theory," *Appl. Math. Leu.*, vol. 1, no. 1, 1988.

[9] C. Joslyn, "Semantic control systems," *World Futures: Journal of General Evolution*, vol. 45, no. 1-4, pp. 87–123, 1995.

[10] E-Lite, "DogOnt," 2012. [Online]. Available: http://elite.polito.it/dogont

[11] M. Boasson, "Control systems software," *Automatic Control, IEEE Transactions on*, vol. 38, no. 7, pp. 1094–1106, 1993.

[12] G. Acampora and V. Loia, "Fuzzy control interoperability and scalability for adaptive domotic framework," *IEEE Transactions on Industrial Informatics*, vol. 1, no. 2, pp. 97–111, May 2005.

[13] *1855-2016 — IEEE Standard for Fuzzy Markup Language*, IEEE Std., 2016.

[14] G. C. B. D. M. Z. Mrissa, M., *Context and Semantic Composition of Web Services*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 266–275. [Online]. Available: http://dx.doi.org/10.1007/11827405_26

[15] G. Milis, D. Eliades, C. Panayiotou, and M. Polycarpou, "A cognitive fault-detection design architecture," in *IJCNN, World Congress in Computational Intelligence*, 2016.

[16] G. M. Milis, D. G. Eliades, C. G. Panayiotou, and M. M. Polycarpou, "Semantic mediation in smart water networks," in *Computational Intelligence, 2015 IEEE Symposium Series on*, Dec 2015, pp. 617–624.

[17] G. M. Milis, M. Asprou, E. Kyriakides, C. G. Panayiotou, and M. M. Polycarpou, "Semantically-enhanced configurability in state estimation structures of power systems," in *Computational Intelligence, 2015 IEEE Symposium Series on*, Dec 2015, pp. 679–686.

[18] D. Luenberger, "Observers for multivariable systems," *IEEE Transactions on Automatic Control*, vol. 11, no. 2, pp. 190–197, Apr 1966.