K-ACE: An Innovative Multi-Component and Multi-Level Architecture

Doctoral Consortium

Valentina Pitoni
University of L'Aquila, DISIM, Italy
valentina.pitoni@graduate.univaq.it

ABSTRACT

My research work concerns Intelligent Software Agents and their applications to "Cyber-Physical Systems" (CPS), i.e. computer systems able to operate and to interact actively with the environment in which they are situated. In CPSs, several components and subsystems interact among themselves, with human users and with the physical environment, and employ forms of intelligent reasoning. Myself and my research group propose a new multi-component multi-level architecture called K-ACE, which provides a high degree of flexibility in the system's definition.

KEYWORDS

Multi-Agent Systems, Computational Logic, Multi-Context Systems

ACM Reference Format:
Valentina Pitoni. 2018. K-ACE: An Innovative Multi-Component and Multi-Level Architecture. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 2 pages.

1 INTRODUCTION

"Cyber-Physical Systems" (CPS), which generally operate under real conditions, may suffer from failures or malfunctions (e.g. in hardware/software) or the environmental conditions may change and influence their behavior and performance. For this reason, adaptivity (e.g., components evolve in time, may join or leave the environment, or become momentarily unreachable) and flexible interaction between components are particularly important.

The research group led by my supervisor has worked to define software architectures for this kind of systems. Mainly:

- DACMACSs (Data-Aware Commitment-based managed Multi-Agent-Context Systems), which are an extension of the work of [5] and of [1]. The DACMACS approach provides a quite general model of multi-agent systems with explicit elements of knowledge representation, as in fact a special agent called *institutional agent* is in charge of locating agents and contexts based on their roles, and manages a common terminological component (global TBox) specified in terms of DRL-Lite Description Logics, and an explicit communication component based upon *commitments*. DACMACSs agents, also possibly exchanging ontological definitions, are able to flexibly interact with heterogeneous external data/knowledge sources. Such knowledge sources are called components, or *"contexts"*, and each of them can be formalized as $C_i = \langle c_i, L_i, kb_i, br_i \rangle$, where c_i is the context's *name*, L_i the logic it is based upon, kb_i is a knowledge base according to L_i, and br_i is a set of *bridge rules*, seen below.
- ACEs (Agent Computational Environment): [3, 4] are defined as a tuple $\langle A, M_1, \ldots, M_r, C_1, \ldots, C_s, R_1, \ldots, R_q \rangle$ where module A is an agent program written in any agent-oriented language, the M_is are "Event-Action modules", which are special modules aimed at Complex Event Processing, the R_js are "Reasoning modules", which are specialized in specific reasoning tasks, and the C_ks are contexts.

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

2 THE NEW ARCHITECTURE

The new architecture that I have contributed to define is called **K-layers-ACE**, or simply **K-ACE**, and is designed as a multi-layered uniform architecture which smoothly integrates many features of DACMACS-ACE required for CPSs.

Each agent in a DACMACS can be an ACE. So, a DACMACS is now renamed K-ACE and can be composed not only of agents and contexts, but also of other (lower-level) K-ACEs. This "nesting" is allowed over an arbitrary number "K" of layers.

K-ACE systems may evolve in time. Thus, we will more properly define a K-ACE relative to a time T (state/time instants are represented as integer numbers as customary in Linear Temporal Logic). Agents, contexts and lower-level K-ACEs can either join and leave the system, so the system composition can change. Also, not every component is allowed to reach every other one: rather, accessibility can be subject to permissions, and is defined via a reachability relation, which itself evolves in time, that dynamically establishes which component can reach which other. Below are the formal definitions.

Definition 2.1. Let a 1-ACE agent be an extension of an ACE agent, where:

- a 1-ACE is characterized by a unique name and a list of roles that the agent can assume;
- a 1-ACE's main agent is equipped with a local ABox;
- a 1-ACE definition includes a "main" agent program and (i) a number internal reasoning modules; (ii) the list of external contexts (components) that the agent can access.
- in 1-ACE (as in ACE/DACMACS) the interconnection between components is managed by the *Bridge-rules* construct, borrowed from MCS [1]. The form of a bridge rule is:
  $s \leftarrow (c_1 : p_1), \ldots, (c_j : p_j), not\ (c_{j+1} : p_{j+1}), \ldots, not\ (c_m : p_m).$
  where the head s is added to the "destination" component's knowledge base in consequence of each atom p_r, $r \leq j$, belongs to the consequences of component c_r, while instead

each atom p_w, $j < w \leq m$, does not belong to the consequences of component c_s. Each c_i can be:
- a constant denoting a component name;
- a term of the form $m_i(k_i)$ that we call *context designator*, which indicates the *kind* of component to be queried, to be substituted by a constant denoting a component name before bridge-rule execution;
- an expression of the form *role@Ag*, denoting a query to the "institutional" agent that will return the name of a component with the required role.

Definition 2.2. Let a K-ACE (K $\geq$ 1) at time T be a tuple

$$\langle N^T, X^T, Y^T, T^T, A^T, R^T, C^T, B^T, E^T, I \rangle$$

identified by a unique name n_K, a role r_{n_K}, and an expression denoting its "institutional" agent $Inst@n_K$, where:

(i) N^T is the list of 1-ACEs' which are part of the K-ACE at time T; each one is identified by a unique name and by its role(s);
(ii) if K > 1, X^T is a finite list of K'-ACEs, K' = K −1, i.e., the lower-level subsystems taking part in the K-ACE at time T; each one is identified by a unique name $n_{K'}$ and by its role(s) $r_{n_{K'}}$, and by an expression denoting their institutional agent $Inst@n'_K$;
(iii) Y^T is a set of contexts' names, listing the contexts which are externally defined and globally known at level K and time T;
(iv) T^T and A^T are the global TBox and Abox respectively, and are common to all agents participating in the system;
(v) R^T is a reachability relation, establishing:
  (a) which elements of N^T can reach each other, thus specifying constraints on inter agent communication;
  (b) which elements of Y^T are reachable by each element of N^T, i.e, which contexts are reachable by each agent;
  (c) which elements of X^T are reachable by each agent in N^T;
  (d) which elements of Y^T are reachable from the outside, precisely from higher-level K-ACEs (if any); such external agents are designated herein via the standard name *outag*.
  We introduce a special distinguished predicate $kreach(C_1, C_2, T)$ which is true whenever component C_2 is accessible from component C_2 at time T according to R^T. The binary version $kreach(C_1, C_2)$ takes T to be the current time.
(vi) C^T is a contractual specification, B^T is a Commitment Box (CBox), E^T is a set of predicates denoting events, and all these elements are exploited by the institutional agent for the management of (commmitment-based) inter-agent communication. Note that communication among 1-ACEs A and B is possible at any time t only if $kreach(A, B, t)$ holds.
(vii) I is a specification for the institutional agent $Inst@n$; specifically, the institutional agent is in charge of inter-agent communication, of contexts' identification via their roles, and of acting as an interface with the lower-level K-ACEs (see (ii)); a query to the institutional agent for obtaining a (set of) agent(s) or a (set of) context(s) with role *role* issued by the 1-ACEs is now of the form $role@Inst@n_K$, and returns the set of agent/contexts with the specified role; we assume that the institutional agent is a special 1-ACE, which does not include external contexts, but that can encompass reasoning modules to be exploited for performing its functions; we assume $Inst@n$ to have direct access to all other K-ACE's elements, and to be able to communicate with institutional agents of lower-level K'-ACEs by means of queries $role@Inst@n_{K'}$.

Therefore, we have constructed a modular architecture where:
- the basic elements are agents, as 1-ACEs, and contexts;
- agents can be part of a K-ACE, which provides, via the institutional agent and with the support of a number of specialized knowledge bases, a suitable infrastructure for communication and for the location (by role) of other agents and of required knowledge bases;
- K-ACEs can be in turn part of other (higher-level) K-ACEs, where the interface among levels is provided by the institutional agents.

In our approach, bridge rules are applied by means of special "trigger rules" that we adopt to activate not only bridge-rule execution, but also inter-agent communication. Specifically, given 1-ACE A, its agent program may include, e.g., trigger rules of the form:

$Q(\hat{y})$ **enables** *communication*$(A, B, Payload, T)$

where $Q(\hat{y})$ is the triggering condition, that must be true in present agent's state, and *communication*$(A, B, Payload, T)$ denotes a communicative act occurring at time T with origin A and destination B where the *Payload* is understood, in FIPA terms (http://fipa.org), as comprising the message according to the specific language adopted. Time T is assumed to be automatically added.

In addition, in 1-ACE, by using "metric" and interval linear temporal logic, we can have trigger rules of the form:

$OP\langle m, n\rangle\ Q(\hat{y})$ **enables** $A(\hat{x})$

where bridge rule with head $A(\hat{x})$ is enabled if $Q(\hat{y})$ has become true never/sometimes before time m, or always in the interval $[m, n]$, according to the LTL operator OP.

3 CONCLUSIONS AND FUTURE WORK

In the extended version of this abstract there are detailed explanations about the semantics of the approach, and its complexity. Related work, including [2], the JaCaMo methodology and the Holonic Agents approach, is discussed. Future work will concern techniques for verification, validation and evaluation of K-ACEs (e.g. expressiveness, scalability and robustness). We also intend to elaborate an execution semantics for K-ACE, by extending the execution semantics provided for DACMASs in [5] in terms of a transition system constructed by means of a suitable algorithm.

REFERENCES

[1] Gerhard Brewka, Thomas Eiter, Michael Fink, and Antonius Weinzierl. 2011. Managed Multi-Context Systems. In *Proc. of IJCAI 2011*, Toby Walsh (Ed.). IJCAI/AAAI, 786–791.
[2] Gerhard Brewka, Stefan Ellmauthaler, Ricardo Gonçalves, Matthias Knorr, João Leite, and Jörg Pührer. 2018. Reactive multi-context systems: Heterogeneous reasoning in dynamic environments. *Artif. Intell.* 256 (2018), 68–104.
[3] Stefania Costantini. 2015. ACE: a Flexible Environment for Complex Event Processing in Logical Agents. In *Engineering Multi-Agent Systems, Third International Workshop, EMAS 2015, Revised Selected Papers (Lecture Notes in Computer Science)*, Luciano Baresi Matteo Baldoni and Mehdi Dastani (Eds.), Vol. 9318. Springer.
[4] S. Costantini and Andrea Formisano. 2013. Budget-Constrained Reasoning in Agent Computational Environments. In *Proceedings of AAMAS 2016, 15th Intl. Conf. on Autonomous Agents and Multi-Agent Systems*. IFAAMAS/ACM. Extended Abstract, to appear.
[5] Marco Montali, Diego Calvanese, and Giuseppe De Giacomo. 2014. Specification and Verification of Commitment-Regulated Data-Aware Multiagent Systems. In *Proc. of AAMAS 2014.*