Collaborative Softbots: Enhancing Operational Excellence in Systems of Cyber-Physical Systems

Ricardo J. Rabelo[1(✉)], Saulo Popov Zambiasi[2], and David Romero[3]

[1] Federal University of Santa Catarina, Florianópolis, Brazil
ricardo.rabelo@ufsc.br
[2] University of Southern Santa Catarina, Florianópolis, Brazil
saulopz@gmail.com
[3] Tecnológico de Monterrey, Monterrey, Mexico
david.romero.diaz@gmail.com

Abstract. This paper outlines the use of Collaborative Networks foundations at the intra-organizational level, applying them in the support of systems of Cyber-Physical Systems (CPSs), and of collaborative softbots in more particular. Five use case scenarios have been implemented upon a didactic shopfloor as a proof of concept. In spite of the complexity related to the knowledge modeling of softbots, the achieved results demonstrated the potentials of better human-automation symbiosis when groups of CPSs, information systems and humans have to cooperate and collaborate using collaborative softbots to improve operational excellence and human satisfaction in smart, social factories.

Keywords: Collaborative softbots · Industry 4.0 · Cyber-Physical Systems · Social factories · Service Oriented Architecture · Collaborative Networks

1 Introduction

Industry 4.0 can be defined as a production model characterized by an increasing digitalization and interconnection of smart products, services, manufacturing systems, value chains/networks and business models [1]. One of the core elements of Industry 4.0 looking at the digital transformation of production systems are the *Cyber-Physical Systems (CPSs)* [2, 3]. In general, a CPS can be defined as “an integrated environment, with computational and physical capabilities, such as sensing, communication and actuation, with feedback-loops where physical processes affect computations and vice-versa” [2, 3].

Although such CPS *capabilities* can be considered as still very advanced for many manufacturing enterprises, particularly for manufacturing SMEs, Industry 4.0 brings up new technical requirements for enterprise information and operational systems and technologies [2, 4]. They include: distributed and decentralized autonomy; cognition and control; failures, faults and anomalies supervision and resilience; adaptation and plug-and-play capabilities; cybersecurity; emergent behavior and self-organization; (big) data integration and interoperability; sensing and cooperative data collection;

Published by Springer Nature Switzerland AG 2019
L. M. Camarinha-Matos et al. (Eds.): PRO-VE 2019, IFIP AICT 568, pp. 55–68, 2019.
https://doi.org/10.1007/978-3-030-28464-0_6

virtualization/digitalization/simulation; data-driven control and optimization; collaboration in a system of CPSs; and symbiotic interaction of CPSs and humans [2, 4, 5]. These two last requirements are of particular interest in this paper.

Collaboration between CPSs involves many issues to handle [6]. *Collaborative Networks* [7] is the scientific area that offers a theoretical basis to study the collaboration phenomena including organizations and general entities. Camarinha et al. [1] have provided a long list of possible links between *Collaborative Networks* and *Industry 4.0,* including its application in networks of collaborative CPSs.

Despite the crucial importance the 'automation and control' part has in CPSs, many works in literature have underestimated the impact of the Industry 4.0 on the workers and, at the same, on the new systems' requirements needed to support the new types of *human-machine interactions* [8, 9]. Several authors (e.g. [9–11]) have pointed out the need for a shift in the way workers and machines interact when looking at boosting their cooperation and collaboration towards smarter symbioses, and hence, at enhancing their operational excellence and human satisfaction towards a cognitive, smart industry [10, 12].

Some approaches have been adopted for that, being *software robots* one of the most promising ones [13, 14]. A *software robot,* also called just as *softbot* [15], can be defined as a "virtual system deployed in a given computing environment that automates and helps humans in the execution of tasks by combining capabilities of conversation-like interaction, system intelligence, autonomy, and process automation". *Softbots* can be also seen as a powerful approach to facilitate the introduction of modern *digital lean manufacturing* and *Jidoka* concepts in terms of helping humans in quality control [16].

In this paper, we define *collaborative softbot* as "a software agent that reactively or proactively cooperates with other softbots, CPSs, information systems and humans helping its users to solve complex or unfamiliar problems, and/or to take care of distributed information requests".

In previous research [11] authors presented a work where a *single softbot* helped a single machine's operator in the execution of some tasks via a high-level and voice-enacted interaction. However, this showed to be not enough regarding collaborative CPSs [6] and Industry 4.0 systems requirements [2, 4, 5]. The works found out in the literature are either theoretical (e.g. in terms of issues or architectures to support CPSs human interactions [9]) or they implement simple scenarios, composed of one operator - one softbot - one CPS [11]. None of the works has tackled the problem of *collaborative softbots* in a system of CPSs.

This work presents a contribution to address this gap. A collaborative softbot's integrated environment has been implemented taking the Industry 4.0 design principles into account [2, 5] and deployed looking at an existing *FESTO didactic shopfloor* using a tool called *ARISA NEST* for deriving softbots.

This paper is organized as follows. Section 1 has introduced the problem and the objectives of the work. Section 2 summarizes the literature review. Section 3 depicts the Arisa NEST tool. Section 4 describes the implemented prototype and its results. Section 5 presents some preliminary conclusions and the next main steps of this work.

2 Basic Foundations and Related Work

In general, *softbots* in an Industry 4.0 scenario [e.g. 11] mean 'talking' to operators about their daily workflows, technical problems, and work-related topics [17].

Similar approaches to *softbots* were proposed in the past, mostly using "agents" [18] and "holonic" systems [19]. Although intelligence and autonomy had been a relevant motivation for their use, agents and holons were actually adopted to ease coordination, systems' architecture design, and integration or synthesis of disparate sources of information [20]. They are now being revisited thanks to their intrinsic features and potentials regarding Industry 4.0 technologies, such as Artificial Intelligence (AI) tools and Machine Learning techniques, which are more mature and robust to be used in real industrial environments, together with the advances on the Industrial Internet of Things (IIoT), Big Data Analytics, and Cloud Computing.

Some authors state that "agents" and "softbots" are equivalent concepts when considering their possible implementation capabilities, and that the difference in terminology just comes from the Schools where they were originated (distributed AI vs. software engineering) [11]. Another reading is that while "agents" and "holons" are more directed to provide some intelligence, autonomy and coordination to a system (usually via structured agent communication languages), "softbots" are more directed to help humans in automating user-customized actions and to intuitive chatting via natural-like language. As a matter of fact, agents can also have or implement such softbots properties and vice-versa.

Softbots do not aim at replacing the native user interfaces (i.e. human-machine/computer interfaces) between workers and machine tools, or of enterprise information systems, such as an ERP or MES. Instead, they simply represent a higher-level of human interaction with those systems, also skipping fixed and predefined menus that are typically accessed via keyboard and mouse. Depending on the way *softbots* are programmed and configured, and eventually learn and evolve, they can execute the actions that are actually defined in given business processes (e.g. Robotic Process Automation (RPA)) or in the functionalities of the CPS they represent (e.g. accessing information from IIoT devices and intelligent objects, or executing ad-hoc activities specifically designed to handle operators' needs).

A *softbot* can be exhibited as a simple icon in a computer's screen or as a sophisticated hologram elsewhere, for example. The information to be provided by the softbot to attend human requests or to interact with other systems can come from a direct invocation to the own systems' functionalities or from the access to cloud environments or local databases that store information generated by those systems.

Conceptually, a *softbot* can be associated to one single system (i.e. one CPS) or it can embrace many systems, becoming in this way a system of systems (i.e. one softbot representing many CPSs of e.g. a manufacturing cell).

Besides the more user-friendly way "softbots" represent in terms of human-machine/computer interaction, there are also some important potentials from the functional point of view. A compilation of this includes [11, 21]: (a) Automatic and/or autonomous execution of actions, from simple alarms to complex business processes, involving communication with diverse devices; (b) higher efficiency and execution correctness when compared to humans, especially in repetitive, hazardous, unsafe and/or unhealthy activities; (c) Filtering, reasoning and pre-selecting useful data to be used in different business contexts and problems; (d) More “quality time” for humans for more valuable activities, such as management and reasoning, instead of lower-value activities, like checking boring and repetitive tasks; (e) More intuitive and effective conversation, especially when compared to FAQs or text manuals in the Web; (f) Accurate, real-time, and up-to-date information as softbots can quickly have access to the right and authorized sources, also helping in the companies’ ICT governance; (g) Automatic generation of operational data and dashboards for further performance evaluation as well as for assisted, interactive operators’ decision-making and conflicts resolution (e.g. via automatic negotiation; interactive bargain, etc.); (h) “Standard” answers, which are important to guarantee that both experienced operators and the ones in training will receive the complete-enough and consolidated information needed; (i) Adaptive and more objective answers regarding operators’ experience, emotions, technical profile, business context, and personal goals’ status; (j) Learning and self-evolving behavior; and (k) Higher availability, being able to answer and process many actions simultaneously all the time.

Collaborative softbots in a system of CPSs does not mean only allowing CPSs to interact with each other, directly, but also ‘collaborating’ with companies’ systems and actors to more properly attend specific human requests or handle general issues. It is also important to point out that collaboration is a result of a rational process that a *softbot* (in this case) can have and that ends up looking for ‘external’ help [22]. This happens when it realizes it has not enough or not trustworthy enough information and capacity or capability to accomplish a given task respecting given requirements, or when it has no interest to accomplish the task on its own regarding its future plans or execution costs [22]. Therefore, *collaboration* does not mean “forcing” interactions between CPSs just to account that, but rather to support it when needed.

In terms of related works, few ones have been found out in the literature about *softbots* on the shop floor.

Schwartz et al. [13] proposed a concept of “hybrid teams” to face the increasing need for higher-level collaboration between humans, industrial equipment and software. Several potential hypothetical cases are described and the requirements for a generic architecture are presented. May et al. [23] proposed a taxonomy identifying the many aspects to be considered for implementing worker-centric systems in manufacturing. Nazarenko and Camarinha-Matos [6] elicited general requirements for collaboration between CPSs, which included the so-called “human orientation”. Kar et al. [17] identified various scenarios for softbots in IIoT environments and proposed a general system architecture, which is based on cloud computing and on a centralized chat platform to connect diverse chat channels. Caldarola et al. [24] proposed an architecture for a CPS with one chatbot, focusing on the problem of different semantic interpretation to better understand users’ requests. None of these five works has implemented their proposals or has addressed “collaborative softbots”. Similarly to [11], Kassner et al. [25] proposed a general architecture for what they called a “social factory”, implementing a softbot to interact with one single machine. The goal was to illustrate the potential benefits of this in a smart factory environment. In the same line, Dersingh et al. [26] developed a chatbot to monitor and record issues of a production line and to notify corresponding workers for appropriate actions. Longo et al. [27] implemented a framework to support the interaction of humans with physical equipment and their digital twins in a cyber-physical environment. The novelty of this work mainly relies on making users interact with a softbot that represents a virtual entity (i.e. a digital twin). Chen et al. [28] developed an engine that captures the production plan, and transform and adapt it to the skills and experience of the involved users so as to improve factory effectiveness and efficiency as well as the satisfaction of factory staff. Gnewuch et al. [29] made some experiments to evaluate the effects of how pre-designed delays in the conversation between users and systems could positively shape users' perception of chatbots. They realized at the end that delays in some situations can create a more human-like environment, and hence a better symbiosis between human and chatbot.

Despite the importance of all these works, none have addressed collaboration between groups of softbots linked to CPSs.

3 The ARISA NEST Tool

ARISA NEST[1] is an academic tool developed as a PaaS-based environment to allow the derivation of both single and groups of softbots. Softbots can be accessed both via desktop PCs and mobile phones running the Android operating system.

It has a number of commonalities, differences, pros and cons when compared to some recent tools for chatbots or softbots, like *Cortana, Siri, Alexa,* and *Watson*[2]. The purpose of this paper is not to compare them, but rather using one through which the main Industry 4.0 systems' requirements can be implemented [4]: interoperability, modularity, virtualization, real-time information, service-orientation, decentralization and autonomy, which is the case of ARISA NEST, for example.

ARISA NEST supports not only conversations between different softbots, but also the coding in the LUA language[3] of any kind of functionalities, which can be implemented both as scripts of simple functions and as more complex object-oriented programs. Programs can comprise since simple direct invocations to services (e.g. web services and REST) and external and legacy systems' APIs, till more complex orchestrations and choreographies. A *softbot* is internally organized under a Service Oriented Architecture (SOA) style and was inspired in the Belief-Desire-Intention (BDI) agent's architecture [22]. It is open to support different communication languages running under HTTP, including proprietary protocols, agent communication languages (ACL) and AIML (Artificial Intelligence Modelling Language) [22].

[1] ARISA NEST tool for softbots derivation – https://arisa.com.br/, developed in Brazil, in Portuguese.

[2] Cortana https://developer.microsoft.com/en-us/cortana, Siri https://www.apple.com/ios/siri/, Alexa https://developer.amazon.com/alexa, Watson https://www.ibm.com/watson/how-to-build-a-chatbot.

[3] LUA programming language – https://www.lua.org/.

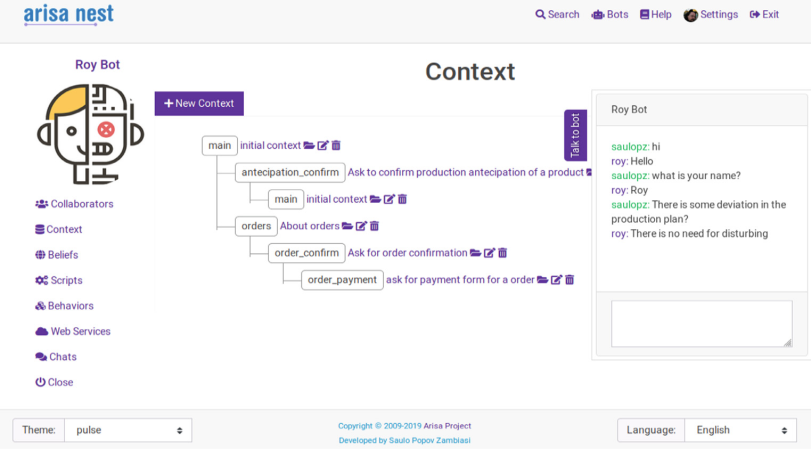
Broadly, a *derived softbot* has the following internal modules: (a) an *Interface,* involving possibilities to interact with other softbots via e-mail, instant messaging, social networks, text consoles, voice, etc.; (b) a *Toolbox,* as a general programming and configuration set of tools to customize and deploy softbots; (c) *Interoperability Services,* comprising a set of services to support the interoperability between the softbot and the systems it has to communicate with; and (d) a *Personal Assistant Manager,* responsible to define, configure, integrate, interoperate, and deploy a softbot as well as to manage the execution of all internal softbot's entities. Softbot's functionalities can be implemented as web services/SOAP protocol or REST.

A key element in the ARISA's model is the *context*, a concept got from AIML. All planned dialogues with users are modeled in XML and have to be carefully defined a priori. Messages are internally structured as forms. *Contexts* mean all the subjects (and the related key terms, including synonymous) users are supposed to talk about when asking things or ordering actions to the softbot. In other words, users can write (or say) a sentence whatever they want since the expected keywords are given. Regarding different idioms, ARISA NEST uses a Google library for automatic translations. New contexts, terms and flows can be added, removed or modified anytime during the softbot's lifecycle. It also uses Google APIs to support communication via voice.

Users are trained about the available contexts in the given softbot. Although all the contexts are organized as a tree (starting with the root node), any node can be directed reached by the internal engine so that the softbot can execute the request more effectively considering users' previous knowledge, and without having to go through fixed sequences of menus. However, in the case the user does not feel satisfied with the answer, he/she can keep asking to the softbot until he/she gets pleased with the provided content or receives a positive acknowledge about the requested action.

Figure 1 gives a broad view of ARISA environment. It has three main parts. Roughly, in the upper bar, *Search* option allows seeing all the derived softbots that are running as ARISA is a PaaS general environment; *Bots* lists the softbots derived by a given user so (s)he can edit them; *Help* provides general information about the ARISA platform, how to derive softbots, etc.; *Settings* refer to users' configurations (e.g. the softbot's name ['*Roy*'] and the icon); and *Exit* is to leave the environment.

In the left sidebar, by clicking on the softbot's icon, in the *Collaborators* option the user can remove or associate other users to the softbot; *Context* provides means to add and manage the subjects and key terms to be used in the conversations (as shown in Fig. 1); *Beliefs* allows to edit and manage the softbot's facts about the application environment it was derived for; *Behaviors* grants access to edit and manage the softbots' functionalities; *Web Services* allows registering the web services linked to the softbots' functionalities; *Chats* option shows all the messages exchanged with the softbot's 'friends' (other softbots and users). All this is exhibited in the central part of the screen. Finally, the *Close* option, which ends the session. The right sidebar is the chatting space, where the *language* to be used can be selected (this information is used to access the Google library, as explained before).



saulopz: hi
roy: Hello
saulopz: what is your name?
roy: Roy
saulopz: There is some deviation in the production plan?
roy: There is no need for disturbing

Fig. 1. ARISA NEST general user interface

ARISA NEST offers some basic computing security support. Being it a PaaS-based system, the first level is provided by the own server against cyber-attacks via firewalls. All the exchanged messages between softbots uses https with SSL connections meaning messages. The database where the facts which represent part of the knowledge used by the softbot to answer user questions is also protected from undesired injections. Users have a login and password as well as a valid e-mail address, guaranteeing users authentication.

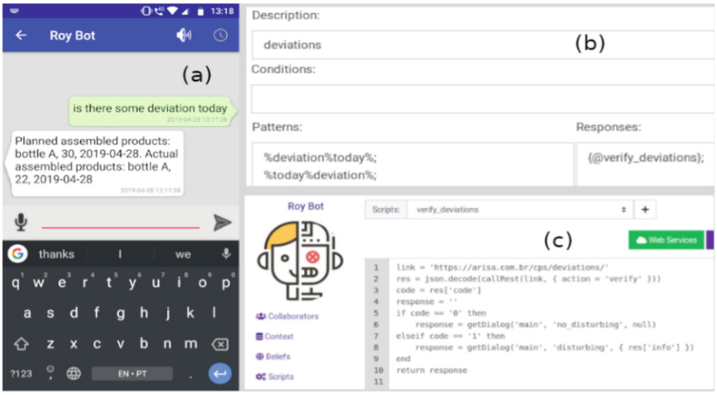
4 Implementation Scenarios and Results

The implementation of this work has considered the FESTO didactic plant existing in our research lab. It is composed of seven CPSs that work as described below.

Basically, a *distribution station* receives components according to the production plan. This is further received by the *testing station*, which checks several aspects to guarantee assembly conformity. In the case the component is not OK, it is put out of the production line; otherwise, the component is separated/sorted according to its size, weight, color and material by the *separating station*. The *processing station* does the planned machining operations (if any) in the sorted component and stores it in a *buffer*. The *pick & place station* starts the first phase of the assembling, picking components from the buffer according to the orders' due date indicated in the production plan. The *muscle press station* completes the assembling process joining different components to compose the final product. Finally, the *sorting station* takes the different final products and sorts them according to their types for further packing and delivery. Although these stations are organized to work in sequence, they are conceptually treated in this work as decoupled, autonomous and distributed CPSs. Each station is equipped with a PLC Siemens S7-1200 full DC and a set of sensors and actuators, having a link to the outside via a Profinet network and the OPC protocol.

Regarding the softbot's architecture, a web services-based wrapper has been built on top of each station. Each station/wrapper works in tandem with its 'manager' module, which represents the CPS and that is designed to work as the CPS 'connector' within the manufacturing environment. The *softbot part* is one of the macro functionalities of the manager. In [30] the authors detail this architecture. In the implemented scenario, each CPS is associated with one single station and it was derived to have one respective softbot. Two operators can interact with more than one softbot in a 'one worker, multiple machines' philosophy (OWMM).

Softbots can be used in many scenarios in the context of Industry 4.0. BCG [21] has outlined ten general scenarios where software assistants could be helpful in leveraging operational excellence, inclusiveness, satisfaction and motivation, safety and continuous learning. Plenty of examples and situations can be created out of this [see 11]. Five 'use cases' have been derived from that in this proof of concept work. Each one is presented below, which shows some GUIs and how it is generally handled in the ARISA tool.



is there some deviation today
2019-04-28 13:17:36

Planned assembled products: bottle A, 30, 2019-04-28. Actual assembled products: bottle A, 22, 2019-04-28
2019-04-28 13:17:38

Description: deviations

Conditions:

Patterns: %deviation%today%; %today%deviation%;

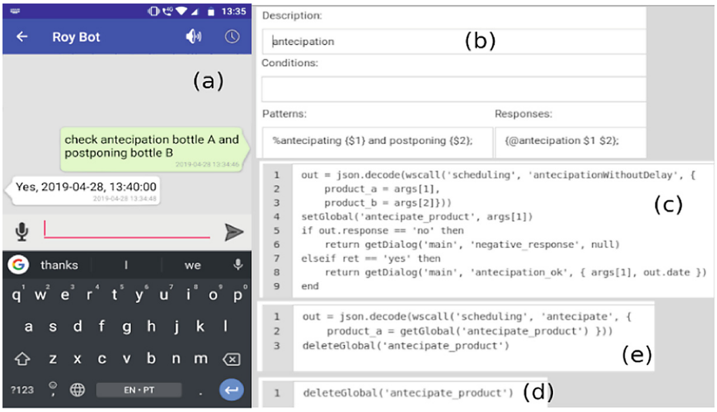
Responses: {@verify_deviations};

Scripts: verify_deviations

```
link = 'https://arisa.com.br/cps/deviations/'
res = json.decode(callRest(link, { action = 'verify' }))
code = res['code']
response = ''
if code == '0' then
    response = getDialog('main', 'no_disturbing', null)
elseif code == '1' then
    response = getDialog('main', 'disturbing', { res['info'] })
end
return response
```

Fig. 2. Softbot querying the MES system about production plan deviations

Use Case I: Operator 1 asks, via voice, to the *separating station* CPS's softbot called *'Roy'* (which is a name similar to a friend instead of a formal system) about current deviations in the production plan considering the number of final assembled products - counted by a specific sensor in the *sorting station* – due to some problems occurred during the work shift (see Fig. 2a). *'Deviation'* word and *'today'* are the key terms the softbot is prepared to hear, which in turn is handled by one of the CPS's manager services (see Fig. 2b). The softbot understands that *'today'* means checking what is the day today and takes this date as the target. The *separating* CPS's softbot *reacts* to this request, accesses the MES's database directly (there is no need to broadcast messages to other CPSs to try to get this information as it is normally stored in the MES or ERP) and sends the information back to Operator 1, noticing him about the expected and actual assembled products amount.



check antecipation bottle A and postponing bottle B
2019-04-28 13:34:46

Yes, 2019-04-28, 13:40:00
2019-04-28 13:34:48

Description:
antecipation

Conditions:

Patterns: %antecipating {$1} and postponing {$2};

Responses: {@antecipation $1 $2};

```
out = json.decode(wscall('scheduling', 'antecipationWithoutDelay', {
    product_a = args[1],
    product_b = args[2]}))
setGlobal('antecipate_product', args[1])
if out.response == 'no' then
    return getDialog('main', 'negative_response', null)
elseif ret == 'yes' then
    return getDialog('main', 'antecipation_ok', { args[1], out.date })
end
```

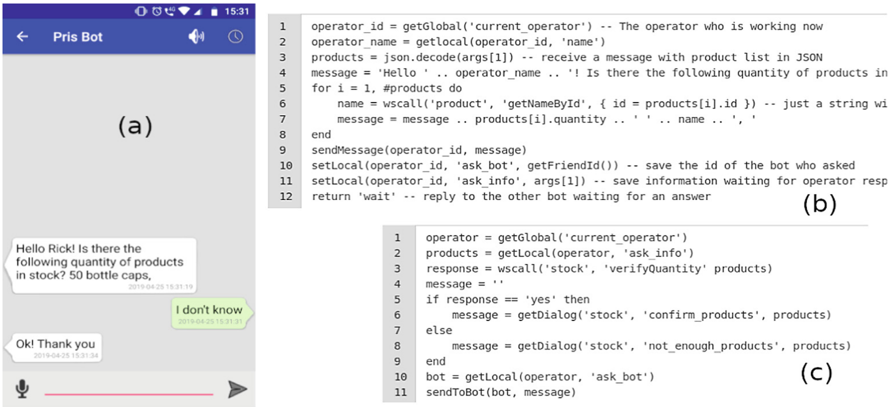
```
out = json.decode(wscall('scheduling', 'antecipate', {
    product_a = getGlobal('antecipate_product') }))
deleteGlobal('antecipate_product')
```

```
deleteGlobal('antecipate_product')
```

Fig. 3. Softbot querying the ERP system about the possibility of a production re-scheduling

Use Case II: Given the delays in the production plan (as fewer products have been assembled), Operator 1 asks, by typing in the keyboard, the *pick & place station's* softbot if it is possible to anticipate the production of *bottle A* without delaying *bottle B*'s due date (see Fig. 3a). Some typed words are taken by the softbot as key terms besides that fact that Operator 1 knows that such products' names are valid in the system. The softbot *reacts* to this, executing the proper CPS's manager service, and interacting with the ERP's scheduler module via its API (Fig. 3b and c). The scheduler can return two parameters after some calculations: 'NO', meaning this is not possible; 'YES' and *'date&time'* value, meaning it is possible, and that the activity linked to 'bottle A' at *pick & place* should start on the given date and time. This information is sent back to the Operator 1, together with a confirmation message 'YES' or 'NO' (Fig. 3d–e). (S)he evaluates that and, given the autonomy philosophy in Industry 4.0, (s)he takes the final decision in the case (s)he agrees on ('YES'). The softbot then invokes again the scheduler to update the production plan, which in turn should update the dispatcher's plan. The dispatcher updates the *pick & place's* PLC program so that the new production sequence can be performed. The operator also has the possibility to access the ERP database, stored in a cloud, to have a broader vision of the production via e.g. Gantt charts and performance indicators dashboards.

Use Case III: During the execution of the new schedule (previous case), the *pick & place's* softbot asks to the *distribution* softbot if there are bottle *caps* enough in stock to accomplish the task (see Fig. 4a). The distribution's softbot firstly asks Operator 2 (*'Rick'*) about it via voice. He answers '*I do not know*', and then the softbot access the inventory information stored in the ERP database (see Fig. 4b). The distribution softbot sends the stock amount ('50') to the pick & place's softbot, which verifies that this is not enough and notifies Operator 2 (see Fig. 4c) about it. In parallel, the softbot sends an e-mail to the purchasing department to warn it about that too. This also aims to demonstrate that interactions in an Industry 4.0 environment can be non-hierarchical, crossing many and different company's layers/departments directly.



Hello Rick! Is there the following quantity of products in stock? 50 bottle caps,
2019-04-25 15:31:19

I don't know
2019-04-25 15:31:31

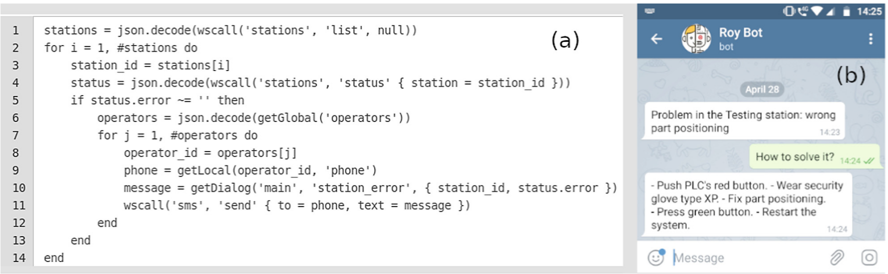
Ok! Thank you
2019-04-25 15:31:34

```
operator_id = getGlobal('current_operator') -- The operator who is working now
operator_name = getlocal(operator_id, 'name')
products = json.decode(args[1]) -- receive a message with product list in JSON
message = 'Hello ' .. operator_name .. '! Is there the following quantity of products in
for i = 1, #products do
    name = wscall('product', 'getNameById', { id = products[i].id }) -- just a string wi
    message = message .. products[i].quantity .. ' ' .. name .. ', '
end
sendMessage(operator_id, message)
setLocal(operator_id, 'ask_bot', getFriendId()) -- save the id of the bot who asked
setLocal(operator_id, 'ask_info', args[1]) -- save information waiting for operator resp
return 'wait' -- reply to the other bot waiting for an answer
```

```
operator = getGlobal('current_operator')
products = getLocal(operator, 'ask_info')
response = wscall('stock', 'verifyQuantity' products)
message = ''
if response == 'yes' then
    message = getDialog('stock', 'confirm_products', products)
else
    message = getDialog('stock', 'not_enough_products', products)
end
bot = getLocal(operator, 'ask_bot')
sendToBot(bot, message)
```

Fig. 4. Softbot querying the ERP system about raw materials inventory

Use Case IV: The *testing station's* softbot permanently and *pro-actively* checks the execution status of this station in the MES's database. In the case, it detects the station is currently stopped because the bottle's cap has been placed upside down, which was a fault detected by one of the testing's sensors. The softbot sends a message, via OPC, to the *testing*'s PLC to go into alarm as well as sends an SMS message to operators 1 and 2's smartphones (see Fig. 5a). Operator 1 was nearer to the station, confirms the problem exists and asks, by typing in the *testing*'s softbot how to solve the problem (see Fig. 5b). The softbot accesses the company's intranet and shows to him the exact part of the troubleshooting manual that explains the possible solutions (see Fig. 5c).



```
stations = json.decode(wscall('stations', 'list', null))
for i = 1, #stations do
    station_id = stations[i]
    status = json.decode(wscall('stations', 'status' { station = station_id }))
    if status.error ~= '' then
        operators = json.decode(getGlobal('operators'))
        for j = 1, #operators do
            operator_id = operators[j]
            phone = getLocal(operator_id, 'phone')
            message = getDialog('main', 'station_error', { station_id, status.error })
            wscall('sms', 'send' { to = phone, text = message })
        end
    end
end
```

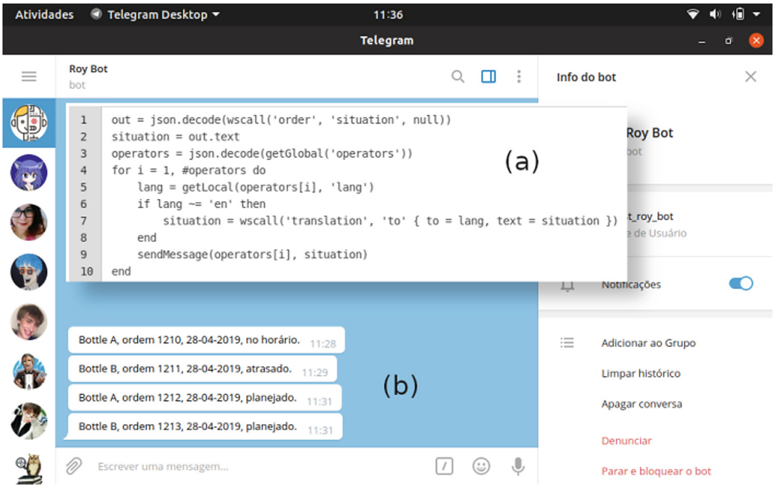
Problem in the Testing station: wrong part positioning
14:23

How to solve it? 14:24

- Push PLC's red button. - Wear security glove type XP. - Fix part positioning. - Press green button. - Restart the system.
14:24

Fig. 5. Softbot querying the station's troubleshooting manual

Use Case V: All softbots *proactively* show a real-time log in the FESTO central PC with the list of the orders in place, their due date, and if they are in time or delayed. This information is obtained from the MES database, which has all data in English. However, the softbot realizes that the operator present in the work shift is Brazilian, so it automatically translates the data to Portuguese to turn the communication more pleasant to the operator (see Fig. 6). "*No horário*" means 'in time'; "*atrasado*" means 'delayed'; and "*planejado*" means 'planned'.



```
out = json.decode(wscall('order', 'situation', null))
situation = out.text
operators = json.decode(getGlobal('operators'))
for i = 1, #operators do
    lang = getLocal(operators[i], 'lang')
    if lang ~= 'en' then
        situation = wscall('translation', 'to' { to = lang, text = situation })
    end
    sendMessage(operators[i], situation)
end
```

Bottle A, ordem 1210, 28-04-2019, no horário. 11:28

Bottle B, ordem 1211, 28-04-2019, atrasado. 11:29

Bottle A, ordem 1212, 28-04-2019, planejado. 11:31

Bottle B, ordem 1213, 28-04-2019, planejado. 11:31

Fig. 6 Softbot translates to a different language the data

5 Conclusions and Further Work

This paper has outlined the use of Collaborative Networks foundations [6, 7] at the intra-organizational level, applying them in the support of *collaborative softbots*.

Five collaborative softbots' use cases have been implemented to show the potentials of better human-automation symbiosis when groups of autonomous, heterogeneous and distributed CPSs, humans, and information systems have to cooperate and collaborate to improve operational excellence and human satisfaction in smart, social factories (e.g. [9, 25]). Although it was implemented in a local environment, the used service-oriented, open and distributed architecture [30] could support the communication with CPSs from other companies as well as services from external providers (so creating virtual organizations and inter-organizational scenario) since some implementation and governance guidelines are set up.

The implementation tried to comprise usual scenarios where *collaborative softbots* might be helpful. The actions were performed as planned, assisting human operators in some tasks, both in terms of interacting and collaborating when needed, and by automating tasks execution on behalf of him/her.

A small group of students was generally trained to use the softbots but knowing very well the subjects in advance. The results were very promising, showing that softbots can be more effective and user-friendly when they asked for some information and actions. This includes the access to information an individual operator usually would not have (or not so easily or promptly) from ERP and MES systems as well as from other CPS/softbots. However, the way softbots' knowledge-base is modeled, populated and maintained is crucial to guarantee answers accuracy and requires a very experienced software engineer and domain expert working together in the designing phase. Besides that, although the way the implemented softbots' wrappers had facilitated integration with physical CPSs - and regarding that this is a proof-of-concept work - some simplifications were made to overcome some tough usual interoperability issues when dealing with industrial systems integration.

The ARISA NEST tool could support all the defined requirements. However, being an academic tool, it has some limitations and some levels of complexity when deriving particular softbots. One of the aspects refers to softbots' behavior programming, although this limitation is also present in the other tools in the market.

Regarding that *softbots* can collaborate directly via messages exchange with any system, and even considering that Industry 4.0 preconizes non-hierarchical interactions, it is important to emphasize that some enterprise information systems (like ERP and MES) keep playing a crucial role in the companies and they are not supposed to be replaced by softbots. Besides *ad-hoc* business processes that can be designed looking at particular needs of softbots (e.g. actions out of the scope of e.g. ERP systems), the chatting part of softbots basically represents a more natural way operators can interact with systems.

This is an on-going R&D work. Next main steps comprise the consideration of machine learning techniques to support softbots evolution and operators' profiles to improve symbiosis between them; the integration of a BPM-like component to help developers in the softbots programming, execution and coordination; and the use of ontologies to better support semantic interoperability between users and softbots as well as between softbots and systems.

Acknowledgements. This work has been partially supported by CAPES Brazilian Agency for Higher Education, project PrInt CAPES-UFSC "Automation 4.0".

References

1. Camarinha-Matos, L.M., Fornasiero, R., Afsarmanesh, H.: Collaborative networks as a core enabler of industry 4.0. In: Camarinha-Matos, L.M., Afsarmanesh, H., Fornasiero, R. (eds.) PRO-VE 2017. IAICT, vol. 506, pp. 3–17. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65151-4_1
2. Lee, J., Bagheri, B., Kao, H.A.: Recent advances and trends of cyber-physical systems and big data analytics in industrial informatics. In: Industrial Informatics. IEEE (2014)
3. Colombo, A., Karnouskos, S., Kaynak, O., Yin, S.: Industrial cyber-physical systems – a backbone of the fourth industrial revolution. IEEE Ind. Electron. Mag. **11**(1), 6–16 (2017)
4. Hermann, M., Pentek, T., Otto, B.: Design principles for industry 4.0 scenarios. In: Proceedings of 49th IEEE Hawaii International Conference on System Sciences, pp. 3928–3937 (2016)
5. Lamnabhi-Lagarrigue, F., et al.: Systems & control for the future of humanity, research agenda: current and future roles, impact and grand challenges. Annu. Rev. Control **43**, 1–64 (2017)
6. Nazarenko, A.A., Camarinha-Matos L.M.: Towards collaborative cyber-physical systems. In: IEEE 2017 International Young Engineers Forum (YEF-ECE), pp. 1–6 (2017). https://doi.org/10.1109/yef-ece.2017.7935633
7. Camarinha-Matos, L.M., Afsarmanesh, H.: Collaborative networks: a new scientific discipline. J. Intell. Manuf. **16**(4–5), 439–452 (2005)

8. Romero, D., et al.: Towards an operator 4.0 typology: a human-centric perspective on the fourth industrial revolution technologies. In: Proceedings International Conference on Computers & Industrial Engineering (CIE46), Tianjin/China, pp. 1–11 (2016)
9. Romero, D., Wuest, T., Stahre, J., Gorecky, D.: Social factory architecture: social networking services and production scenarios through the social internet of things, services and people for the social operator 4.0. In: Lödding, H., Riedel, R., Thoben, K.-D., von Cieminski, G., Kiritsis, D. (eds.) APMS 2017. IAICT, vol. 513, pp. 265–273. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66923-6_31
10. Jones, A.T., Romero, D., Wuest, T.: modeling agents as joint cognitive systems in smart manufacturing systems. Manuf. Lett. **17**, 6–8 (2018)
11. Rabelo, R.J., Romero, D., Zambiasi, S.P.: Softbots supporting the operator 4.0 at smart factory environments. In: Moon, I., Lee, G.M., Park, J., Kiritsis, D., von Cieminski, G. (eds.) APMS 2018. IAICT, vol. 536, pp. 456–464. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99707-0_57
12. Kumar, N., Kumar, J.: Efficiency for Industry 4.0. Hum. Technol. **15**(1), 55–78 (2019)
13. Schwartz, T., et al.: Hybrid teams: flexible collaboration between humans, robots and virtual agents. In: Klusch, M., Unland, R., Shehory, O., Pokahr, A., Ahrndt, S. (eds.) MATES 2016. LNCS (LNAI), vol. 9872, pp. 131–146. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45889-2_10
14. Kopec, W., et al.: Hybrid approach to automation, RPA and machine learning: a method for the human-centered design of software robots. arXiv:1811.02213v1 (2018)
15. Kim, J.H.: Ubiquitous robot. In: Reusch, B. (ed.) Computational Intelligence, Theory and Applications. Advances in Soft Computing, vol. 33, pp. 451–459. Springer, Heidelberg (2005). https://doi.org/10.1007/3-540-31182-3_41
16. Romero, D., Gaiardelli, P., Powell, D., Wuest, T., Thürer, M.: Rethinking Jidoka systems under automation & learning perspectives in the digital lean manufacturing system world. In: IFAC Conference on Manufacturing Modelling, Management and Control (2019)
17. Kar, R., Haldar, R.: Applying chatbots to the internet of things: opportunities and architectural elements. Int. J. Adv. Comput. Sci. Appl. **7**(11), 147–154 (2016)
18. Barata, J., Camarinha-Matos, L.M.: Coalitions of manufacturing components for shop floor agility. Int. J. Netw. Virtual Organ. **2**(1), 50–77 (2003)
19. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L.: Reference architecture for holonic manufacturing systems. Comput. Ind. **37**(3), 255–274 (1998)
20. Lu, Y.: Industry 4.0: a survey on technologies, applications and open research issues. J. Ind. Inf. Integr. **6**, 1–10 (2017)
21. BCG Boston Consulting Group. https://www.bcg.com/ptbr/publications/2015/technology-business-transformation-engineered-productsinfrastructure-man-machine-industry-4.aspx
22. Ferber, J.: Multi-Agent Systems – An Introduction to Distributed Artificial Intelligence. Addison-Wesley, Boston (1999)
23. May G. et al.: A new huma-centric factory model. In: 12th Global Conference on Sustainable Manufacturing, pp. 22–24. Elsevier (2015)
24. Caldarola, E.G., Modoni, G.E., Sacco, M.: A Knowledge-based approach to enhance the workforce skills and competences within the industry 4.0. In: 10th International Conference on Information, Process, and Knowledge Management, pp. 56–61 (2018)
25. Kassner, L., Hirmer, P., Wieland, M., Steimle, F., Königsberger, J.K., Mitschang, B.: The social factory: connecting people, machines and data in manufacturing for context-aware exception escalation. In: 50th Hawaii International Conference on System Sciences (2017)
26. Dersingh, A., Srisakulpinyo, P., Rakkarn, S., Boonkanit, P.: Chatbot and visual management in production process. In: International Conference on Electronics, Information, and Communication, pp. 274–277 (2017)

27. Longo, F., Nicoletti, L., Padovano, A.: Smart operators in industry 4.0: a human-centered approach to enhance operators' capabilities and competencies within the new smart factory context. Comput. Ind. Eng. **113**, 144–159 (2017)
28. Chen, X., Bojko, M., Riedel, R., Apostolakis, K.C., Zarpalas, D., Daras, P.: Human-centred adaptation and task distribution utilizing levels of automation. In: 16th IFAC Symposium on Information Control Problems in Manufacturing, pp. 54–59 (2018)
29. Gnewuch, U., Morana, S., Adam, M., Maedche, A.: Faster is not always better: understanding the effect of dynamic response delays in human-chatbot interaction. In: 26th European Conference on Information Systems (2018)
30. Siller, H., Romero, D., Rabelo, R., Vazquez, E.: Advanced CPS service oriented architecture for smart injection molding and molds 4.0. In: IEEE 9th International Conference on Intelligent Systems, pp. 1–7 (2018)