Efficient Model for Distributed Computing based on Smart Embedded Agent

Hassna Bensag, Mohamed Youssfi, Omar Bouattane
Laboratory SSDIA, ENSET
Hassan II University of Casablanca
Mohammedia 28999, Morocco

***Abstract*—Technological advances of embedded computing exposed humans to an increasing intrusion of computing in their day-to-day life (e.g. smart devices). Cooperation, autonomy, and mobility made the agent a promising mechanism for embedded devices. The work aims to present a new model of an embedded agent designed to be implemented in smart devices in order to achieve parallel tasks in a distribute environment. To validate the proposed model, a case study was developed for medical image segmentation using Cardiac Magnetic Resonance Image (MRI). In the first part of this paper, we focus on implementing the parallel algorithm of classification using C-means method in embedded systems. We propose then a new concept of distributed classification using multi-agent systems based on JADE and Raspberry PI 2 devices.**

Keywords—Distributed computing; parallel computing; Multi Agent System; Embedded computing; Raspberry PI 2

I. INTRODUCTION

Technological advances had imposed a growing intrusion of data processing tools as smart devices, giving us the opportunity to grow towards a continuous mobility.

Ambient intelligence does not merely adapt the technology to the human need, but also to the science demands by providing advanced embedded devices with high-level computing power. The low cost of some smart devices like raspberry Pi, and Arduino made them a fertile platform for high performance computing (HPC), an area that was previously very limited due to the cost and the complexity of HPC cluster. Today, thanks to smart devices advanced features, building a cluster to explore parallel computing has become even more cheaper and easier [19]. To fully exploit the cluster resources potential, strong jobs are partitioned into several tasks; these sub tasks are then distributed to multiple smart devices aiming to reduce the cost of communication, latency and execution time. Therefore, introducing some cooperating and social reasoning capabilities to these intelligent devices is necerray.

An intelligent agent is "a computer system, situated in some environment and capable of flexible and autonomous action in order to meet its design objectives" [15,18]. Multi-agent systems are based on the approach: compute corporately and autonomously. Even though multi-agent approach seems appropriate for raspberry devices, we must solve some agent effective implementation issues [20,16,18,17,12]

The multi-agent systems are used in many domains such as economy simulations, renewable energy, computer science and healthcare domain where image segmentation poses several issues [3,4,8,9,10 and 11]. In fact, when the image contains a large amount of data, the segmentation process takes a long time [5, 6].

In this article, we focus on the design of intelligent agents embedded in Raspberry Pi device. Also, the implementation of a parallel and distributed environment consisting of a middleware able to manage a set of embedded mobile agents and to provide a mechanism for load balancing and reducing communication cost. The goal is to overcome the distributed computing challenges and ensure a high-performance computing [13]. This paper aims to propose a new method for c-means classification applied to a cardiac MRI. The latter will be segmented on a parallel-distributed platform based on agents, which are embedded in Raspberry pi devices.

The second section of this paper consists in a review of all methods and tools used in the proposed system. The third section gives an overview of the distributed computational model. The proposed architecture is evaluated in section four with a case study using the distributed c-means algorithm. Section five presents the experiment results. Finally, conclusion and future work.

II. BACKGROUND

This section details selected methods, approaches and tools used in multi-agent system distributed in embedded devices.

A. Multi Agent System (MAS)

An agent is an encapsulated computer system, situated in an environment, and capable of performing flexible and autonomous action in order to meet its design objectives [9]. The main common agent's characteristics are autonomy, reactivity, proactivity, intelligence, adaptability, collaboration and mobility. Mobile agents have the additional ability to move from one machine to another [2,5,9,10 and 14]. Multi-agent system is used in different areas, offering strong models for complex and dynamic environments representation. MAS can also be used to simulate the behavior of complex computer systems, this simulation models can help designers and developers of complex computational systems. So, the multi-agent based simulation provides a good set of tools to manage complex systems for online resource allocation environments.

B. JADE Agent platform

JADE platform is distributed by Telecom Italia the copyright holder, in open source under the terms and conditions of the LGPL (Lesser General Public License Version 2) license [9]. JADE (Java Agent Development) is the most popular open source framework for the development of multi-agent systems, it is a framework fully programmed in JAVA language. It is a FIPA (Foundation for intelligent Physical Agents) compliant agent platform, composed of multiple containers which host and execute agents. The main goal of JADE platform is especially simplifying development while ensuring standard compliance through a comprehensive set of systems for agents and services [1]. Launching JADE platform triggers at least one container called Main Container, if there is other agents, they are registered with the main container [7] (Figure. 1).

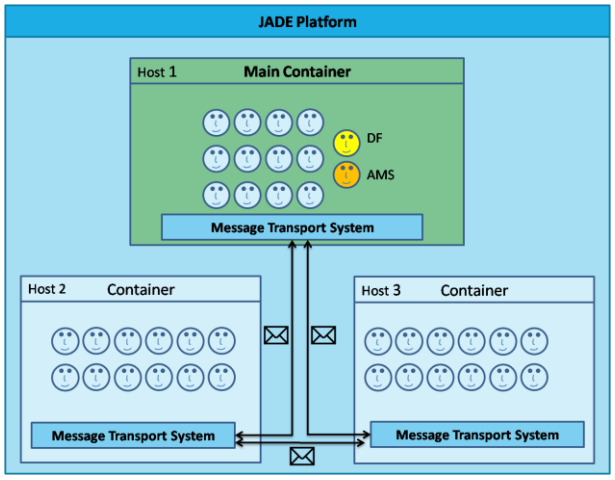

Fig. 1. JADE Agent platform

C. Raspberry PI

Raspberry PI card provides high speed, better accuracy, good flexibility and low cost solution for the development of embedded system equipped by ARM. Using this last board as development platform speed up the process of development. Raspberry pi Model B (as shown in Figure.2) is currently the most popular ARM board. Raspberry PI has a Broadcom BCM2835 system on a chip SoC, which includes an ARM1176JZF-S 700 MHz processor, VideoCore IV GPU, and is shipped with 512MB of RAM .It does not include a built-in hard disk, it uses instead an SD card for booting and long-term storage. It comes with two USB ports, RJ45 Ethernet port, HDMI port and RCA output on board.

Fig. 2. Raspberry PI 2 Model B

D. Distributed C-Means Algorithm

The C-means classification method as defined in [6], is an algorithm for image segmentation consisting of a partitioned groups of set S of n attribute vectors into c classes (clusters C_i , i= 1,…, c), generally based on different criteria segmentation : gray levels, texture or shapes. The main goal of the algorithm is to find the class centers in order to minimize the cost function by using:

$$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{k \in c_i} d(x_k, C_i)$$

Where:

C_i is the center of the i^{th} class

$d(x_k, C_i)$ is the distance between i^{th} center C_i and the k^{th} data of the set S

We use the Euclidean distance to define the objective function as follows:

$$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{k, x_k \in c_i} ||x_k - C_i||_2$$

The partitioned groups can be defined by a binary membership matrix U(c, n), where each element u_{ij} is formulated by:

$$\begin{cases} 1 \; if \; ||x_k - C_i||^2 \leq ||x_k - C_k||^2, \forall k \neq i \\ 0 \; otherwise \end{cases}$$

(i=1 tp c, j=1 to n; n is the total number of points in S).

Since a data must belong to only one class, the membership matrix U has two properties which are given in the following equations:

$$\sum_{i=1}^{c} u_{ij} = 1, \forall j = 1, \dots, n$$

$$\sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij} = n$$

The value C_i of each class center is computed by the average of all its attribute vectors:

$$C_i = \frac{1}{|c_i|} \sum_{k, x_k \in c_i} x_k$$

$|C_i|$ is the size or the cardinal of C_i.

The C-means classification is achieved using the following algorithm stages as illustrated in figure. 3:

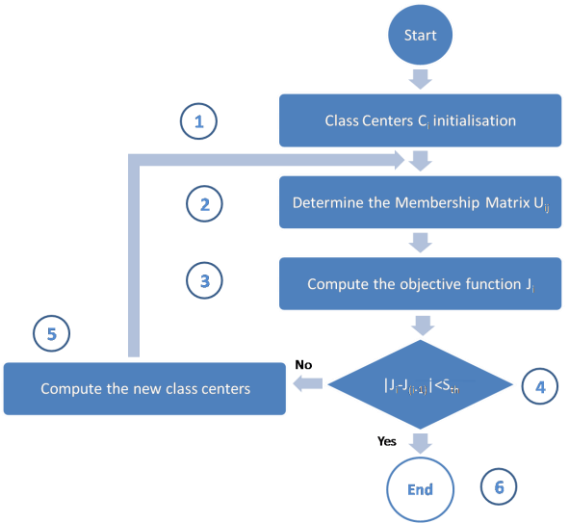

Fig. 3. Standard classification c-means process

III. Distributed Computational Architecture

In this section, we present distributed computational architecture which uses agent potential to create a cooperative multi agent platform. The system environment is described before detailing its main components.

A. Computing environment model

The proposed system aims to distribute tasks on a grid of embedded devices: raspberry pi. To perform tasks distribution in the described model (Figure 5), we must achieve the following three steps:

- **Initialization:** the launching of the platform is associated with the creation and initialization of the agent task dispatcher (ATD) using the task prepared by the application.
- **Joining the grid:** once the raspberry pi joins the network it is considered as an available resource and registers itself with the agent task dispatcher to start solving distributed tasks.
- **Task execution:** a task needs a local agent (agent local worker - ALW) and an available embedded remote agent (embedded remote agent - ERA) [15]. Once there are available ERAs, selected from the ATD repository, the execution process starts.

Beginning with task initialization, the process is then followed by task remote execution and ends up with task finalization.

B. Main component description

The proposed platform is a distributed and parallel architecture based on JADE middleware. As shown in figure 4, we distinguish three main components:

- **The main container:** The platform contains one active main container and all other containers joining the platform have to register with it. The main container hosts two special JADE agents: the AMS or agent management system, it keeps the repository of all intelligent agents of the platform. And the DF or directory facilitator; the yellow page service in which agent can register or find the available service in the platform. The main container is also a container where the main agent ATD is deployed.
- **The local container**: is created from the node responsible for the task distribution process initiation in the platform. It is where the ALW agent is hosted.
- **The remote Container**: this container receives groups of the ERAs so that each one can execute its task in parallel.

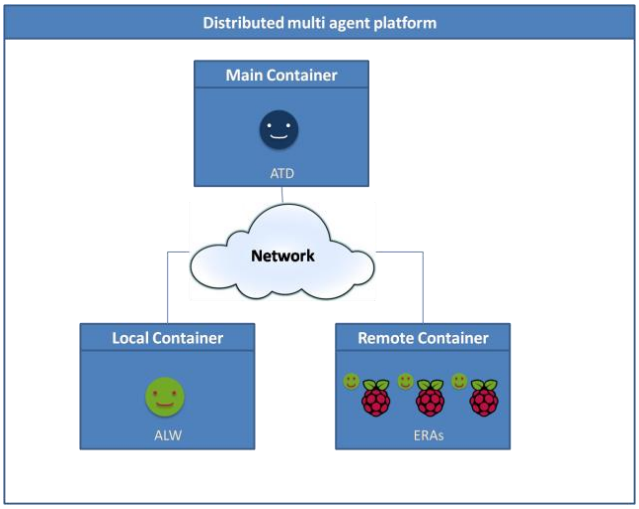

Fig. 4. Distributed multi agent platform

IV. Distributed C-Means Application

Detailed image segmentation according to distributed c-means algorithm is presented in this section. We highlight the segmentation process following the distributed computational model. To handle the raspberry pi characteristics heterogeneity a middleware is required in order to guarantee load balancing.

A. C-means segmentation process in the proposed approach

To prove the reliability of the proposed architecture, we take the c-means image segmentation process as a case study. The proposed architecture uses a C-means algorithm as a distributed program. In order to perform the distributed c-means classification implemented on a grid consisting of embedded devices, we should follow these steps:

- **Task preparation**: we should define the data and processing to be performed for the image c-means segmentation. In this application the user have to choose the source folder containing the image to classify. The classification treatments are defined in a java class in the application.
- **Task initialization**: Once the task is prepared, it is automatically added to the queue of ATD agent before being sent to ALW agent. This latter splits the image based on number (n) of available embedded devices (raspberry pi) provided by the ATD. Each elementary image is then distributed to a specific embedded remote agent (ERA).

- **Task execution**: when the data segmentation is executing, the ALW sends both elementary image and the initial class centers to ERAs. Each ERA determines the local class centers in order to compute the member ship matrix. Afterwards, it replies to the ALW agent message by sending the calculation results gathered from each ERA agent. ALW agent determines then the global class centers and computes the objective function J. The aim is to calculate the absolute value of the difference between J values in iteration i and i-1. If the result is bigger than threshold (Sth), the ALW agent sends a new class centers to the ERAs and the whole process is repeated until obtaining an absolute value lower than Sth. Finally the ERAs send the output elementary segmented images to ALW.
- **Task finalization**: in this step, the LAW assembles the elementary segmented images in order to display the c output segmented images for the classification where c corresponds to the class number.

B. Distributed middleware mechanisms for c-means algorithm

The proposed load distribution middleware aims to develop a multi-agent system to distribute tasks on a grid of embedded devices using intelligent agents. These agents are embedded in heterogeneous nodes of the platform and can dynamically execute the task they receive. We distinguish 3 different types of intelligent agents in this architecture:

- **Agent Task Dispatcher (ATD)**: ATD is the key element of the platform, which has several related functions. First, the ATD overviews the containers and registers agents. It must keep an up-to-date Active Agent Repository of all available intelligent agents. Second, the ATD agent is responsible to distribute tasks among ERAs able to solve tasks. The ATD keeps track of all partial tasks in its Task Allocation tables. Each partial task is marked as “unassigned”, “assigned” or “completed.” Finally, after completing their assigned sub-tasks, ERAs return the partial results to the ALW. After the results have been assembled by the ALW, the ATD deletes the corresponding partial tasks from its Task Allocation tables.
- **Agent Local Worker (ALW):** Any agent on the grid can act as an ALW of a distributed task. If an agent has a task that it cannot solve by itself, it can become an ALW and announce the task to other agents on the grid via the ATD. If there are other available intelligent agents in the grid capable of solving such a distributed task, the task execution process starts, and the ALW takes it over. This agent is the one responsible for achieving both initialization and finalization process.
- **Embedded Remote Agent (ERA):** Any intelligent agent that does not serve as an ALW of a distributed task at a given moment and has registered itself with the ATD is considered as ERA. Once created, these agents move to remote containers where they are supposed to execute their tasks.

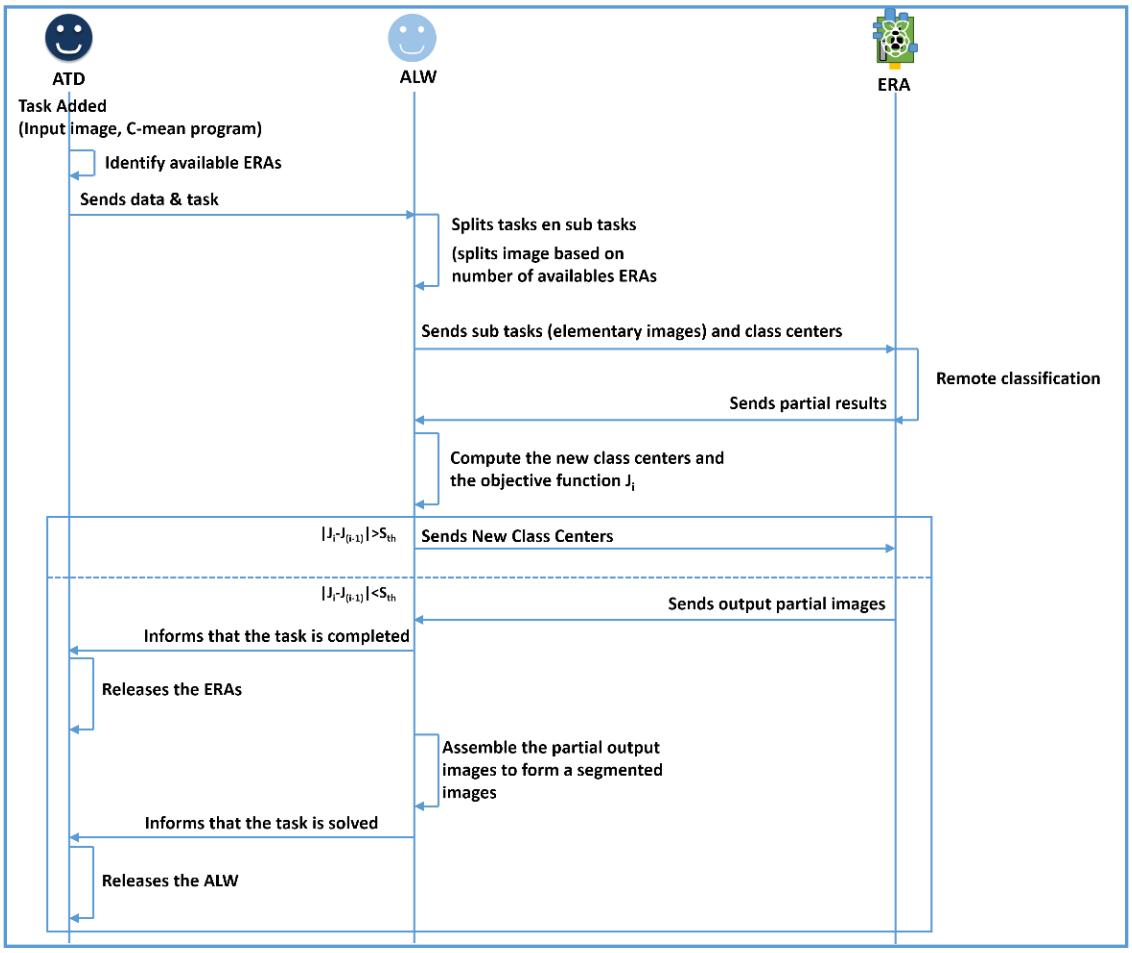

Fig. 5. Distributed Sequence diagram for distributed c-means classification

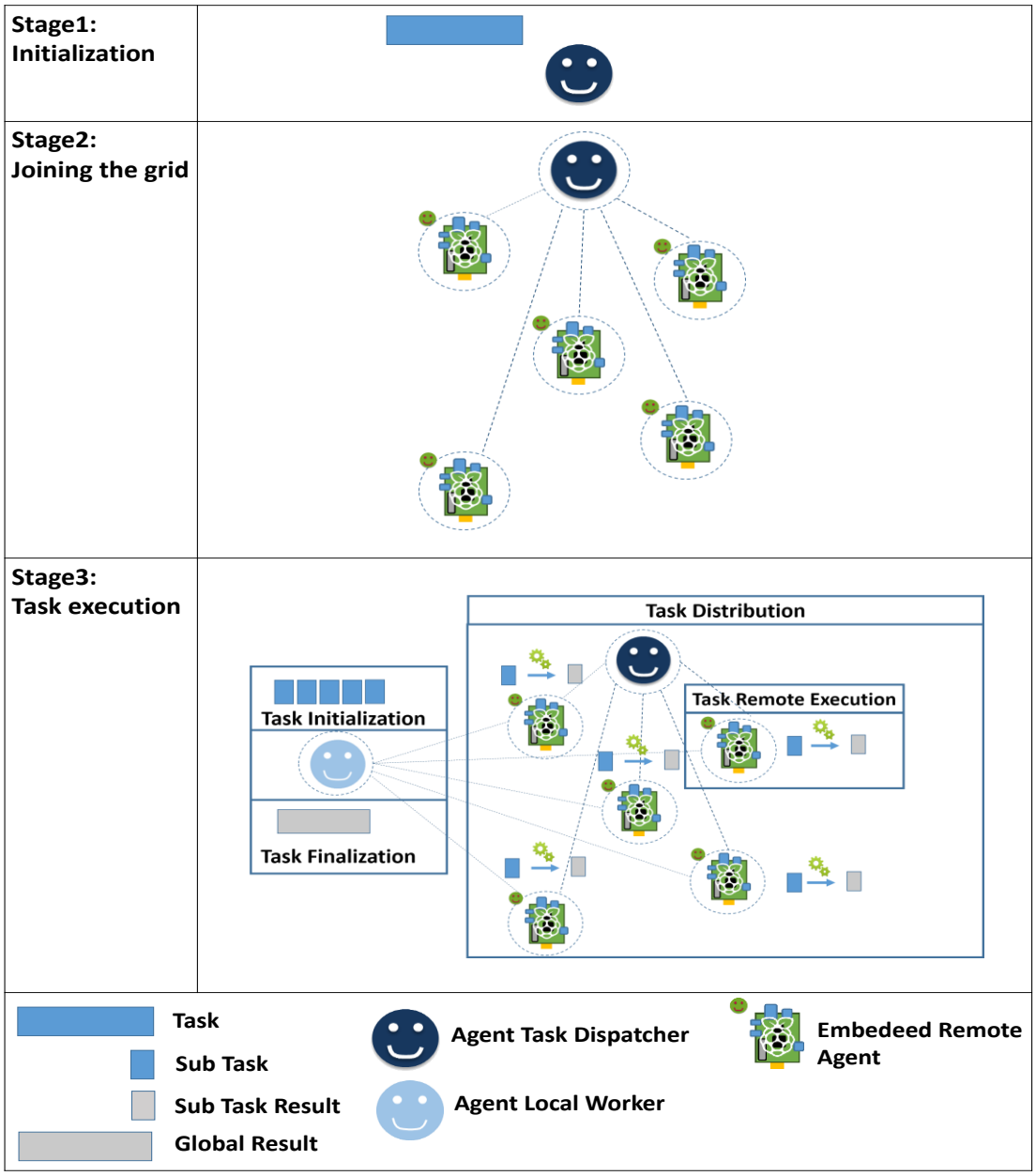

Fig. 6. Distributed model of task execution

V. Simulation and Results

To substantiate the architecture, we implement the c-means program in the platform according to the sequence diagram presented in Figure 6. We have used a cardiac MRI image. The experimental evaluation was done on the test-bed (Figure 7) with two Raspberry PI having the same configuration (Model B) and a third one (Raspberry PI) which host agent. The results are summarized in figure 8: The figure (a) corresponds to a human cardiac MRI, and the figures (b),(c),(d), are the segmented output images where each of them corresponds respectively to class centers (c1=0, c2=127, c3=255).

The first experiment confirms that the behavior of the real distributed systems is coherent with the simulation results, the algorithm converged to the final class centers (c1, c2, c3)=(13.00,99.00,220.00) right after the 8th iteration as shown in Figure 9.

In Figure 10, it's see clearly that from 16 agents the classification time of the two images achieves minimum values of 100 ms. Therefore, we do not need more than 16 devices to obtain this achievement time.

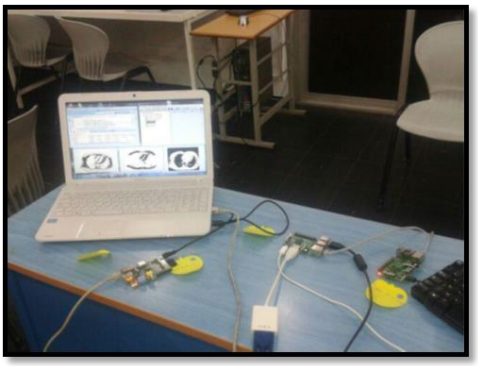

Fig. 7. Testbed

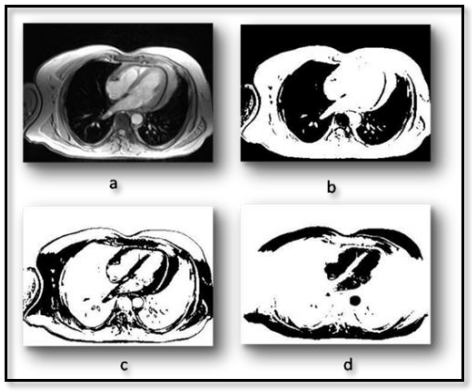

Fig. 8. Segmentation results

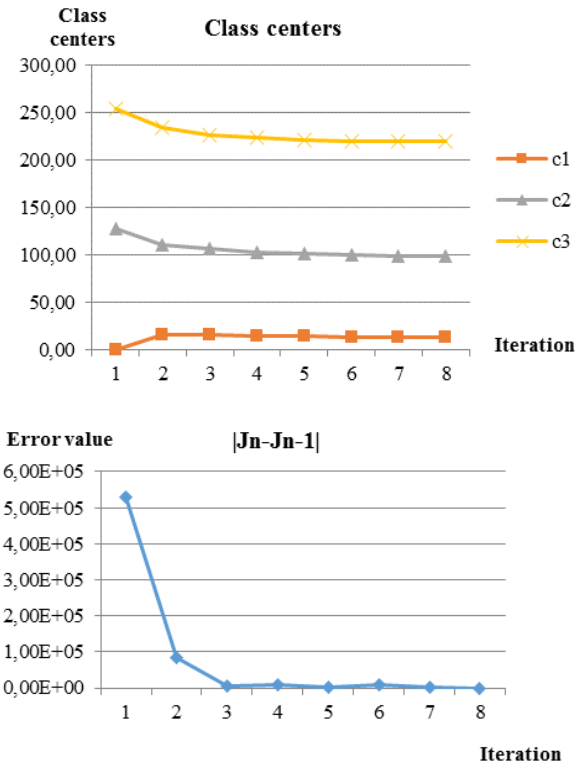

Fig. 9. Class centers (c1, c2, c3)=(0,127,254) And Error of the cost function

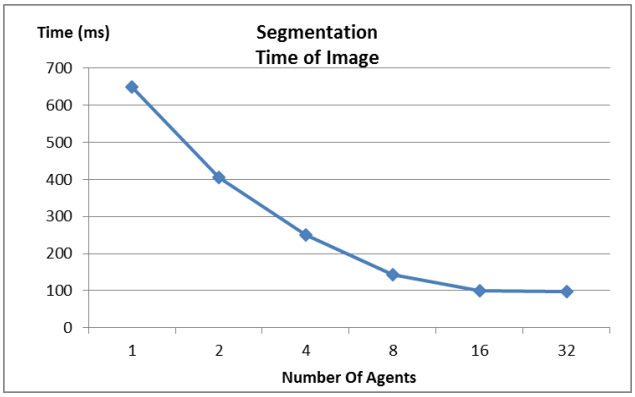

Fig. 10. Many agents by one smart devices

VI. CONCLUSION

The paper presents a distributed computing method with the raspberry PI to fill the gap between the distribution performance and the cluster cost. The proposed architecture can guarantee that, introducing a new model of an ambient agent designed to be implemented in low cost devices, such as raspberry Pi, to achieve parallel tasks in a distributed environment. To experimentally test the efficiency of the proposed system, the method was applied on objective medical image segmented by c-means algorithm, using JADE middleware and embedded agent based on raspberry Pi. The encouraging experiment results proved that the system fills all required features for a performing standard cluster. The proposed architecture opens new horizons towards new and more advanced systems including load balancing and internet of things.

REFERENCES

[1] S.Sotiriadis, N.Bessis, Y.Huang, P.Kuonnen, N.Antonopoulos, A JADE Middleware for Grid inter-cooperated infrastructures, international conference on advanced information networking and applications, 978-7695-4338-3, 2011.

[2] L, Zhang, Q.Wang, X.Shu, A mobile-Agent-Based Moddleware for wirless Sensor Networks Data fusion, International Instrumentation and measurement technology, 978-1-4244-3353-7, May 5-7 Singapore 2009.

[3] L.Chunlin, L.Layuan, A multi-agent model for service-oriented interaction in a mobile grid computing environment, Pervasive and mobile computing 7, 270-284, ELSEVIER, 2011.

[4] M.Youssfi, O.Bouattane, J.Bakkoury, M.O.Bensalah, A new massively parallel and distributed virtual machine model using mobile agents, international conference on multimedia computing and systems, 978-1-4799-3823-0, April 14-16 Marrakech, Morroco 2014.

[5] M. Youssfi, O. Bouattane, and M.O. Bensalah " On the Object Modelling of the Massively Parallel Architecture Computers", Proceedings of the IASTED Inter.Conf. Software engineering, Innsbruck, AUSTRIA, pp 71-78, February 16 - 18, 2010.

[6] O.Bouattane, B. Cherradi, M. Youssfi and M.O. Bensalah "Parallel cmeans algorithm for image segmentation on a reconfigurable mesh computer" ELSEVIER. Parallel computing, 37 pp 230-243, 2011.

[7] F.Bellifemine, A.Poggi, G.Rimassa, Developing Multi-agent systems with JADE, Intelligent Agents VII, pp.89-103, speinger 2001.

[8] M.Higashino, T.Hayakawa, K.Takahashi, T.Kawamura, K.Sugahara, Management of streaming multimedia content using mobile agent technology on pure P2P-based distributed e-Learning system, international conference on advanced information networking and applications, 978-0-7695-4953-8, March 25-28 Barcelona 2013.

[9] F. L. Bellifemine, G. Caire, and D. Greenwood, "Developing Multi-Agent Systems with JADE",Wiley, 2007.

[10] I.Satoh, Mobile Agent Middleware for dependable distributed systems, international conference on informatique technology interfaces, june 27-30, Cavtat, Croatia 2011.

[11] R.Abidar, K.Moummadi, H.Medromi, Mobile device and multi agent systems: an implemented platform of real time data communication and synchronization, international conference on multimedia computing and systems, 978-1-61284-730-6, 7-9 April Ouarzazate, Morocco.

[12] F.Bergenti, G.Caire, D.Gotta, Agenst on the move : JADE for android devices, CEUR workshop proceeding voal-11260, sepy. 25-26 catania, Italy 2014.

[13] Petr Kadera1, Petr Novak1, Vaclav Jirkovsky, Pavel Vrba1, Performance models preventing multi-agent systems from overloading computational resources, Automation, Control and Intelligent Systems, 2(6): 105-111, 2014.

[14] Abhilash Kantamneni, Laura E. Brown, Gordon Parker, Wayne W. Weaver, Survey of multi-agent systems for microgrid control, Engineering Applications of Artificial Intelligence 45, 192–203, ELSEVIER, 2015.

[15] H.Bensag, M.Youssfi, O.Bouattane, Embedded Agent for medical image segmentation, IEEE ICM 2015, 20-23 December Casablanca , Morocco

[16] F.Doctor, H.Hagras, V.Callaghan, "A type-2 fuzzy embedded agent for ubiquitous computing environments", Fuzzy Systems, Proceedings IEEE International Conference , 1105 - 1110 vol.2, July 2004.

[17] T.Leppnen, J.Riekki, M.Liu, E.Harjula, T.Ojala: Mobile agents-based smart objects for the internet of things, Internet of Things Based on Smart Objects, pp. 29–48. Springer, Heidelberg (2014)

[18] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish and H. Duman, "Creating an ambient-intelligence environment using embedded agents", Intelligent Systems, IEEE, vol. 19, no. 6, pp. 12-20, 2004

[19] C. Ramos, J. C. Augusto and D. Shapiro, "Ambient intelligence - the next step for artificial intelligence", Intelligent Systems, IEEE, 2008

[20] F.Ramparano, O.Boissier, "Smart Devices Embedding Multi-agent Technologies for a Pro-active World", The Ubiquitous Computing Workshop, Bologna, Italy, 16 July 2002.