TOWARDS AN ARCHITECTURE FOR MOBILE CLOUD ROBOTICS

José M. Reyes Álamo, Benito Mendoza and Aparicio Carranza
New York City College of Technology, USA

ABSTRACT

Cloud computing is a ubiquitous term used today for computational models that rely on interconnected computers over the Internet offering greater processing power and storage capabilities than stand-alone solutions. Cloud computing has been applied to diverse fields including cloud robotics and mobile cloud. Cloud robotics is a paradigm where robots offload their heavy computations and storage needs to the cloud while focusing on simpler computation tasks. Mobile cloud computing (MCC) is the reliance on mobile devices such as smartphones and tablets for data gathering and simple processing while using the cloud for resource intensive tasks. We propose a mobile cloud robotics architecture that combines these two concepts. Three different architectural models for mobile cloud robotics are provided: the Lego NXT Mobile Cloud Robotics Architecture, the Android Powered Lego NXT Mobile Cloud Robotics Architecture, and the Hybrid Lego NXT Mobile Cloud Robotics Architecture. These architectural models combine several tools including the Lego NXT robot, the Lego Java Operating System (leJOS), a Python interpreter, and the Seattle cloud computing platform. Details on the first two architectural models for the mobile cloud robotics architecture with their advantages and disadvantages are provided. We also argue that a hybrid architecture that combines features of the original two could offer the best solution as it has a good trade-off in terms of performance and cost.

Keywords: *Cloud Computing, Cloud Robotics, Mobile Cloud, Architecture, Java, Python.*

INTRODUCTION

Cloud computing has become a ubiquitous term used today for computational models where tasks are executed by several interconnected computers offering greater capabilities in terms of processing power and storage than stand-alone solutions. Researchers have applied the principles of cloud computing to the field of robotics as well as to mobile environments resulting in the emergence of topics such as cloud robotics and mobile cloud. Cloud robotics refers to the use of the cloud for applications involving robots (Chen, Du, & García-Acosta, 2010; Hu, Tay, & Wen, 2012). Robotics applications usually require lots of computations which are limited by the capability of the robot. Cloud computing offers significantly more resources making it possible for robotics applications to become more sophisticated. Embedded devices also have limited capability and other limitations such as battery life. The concept of mobile cloud computing (MCC) relies on using mobile devices such as smartphones, tablets, and embedded devices to detect phenomena, while offloading the heavy computation and storage needs to the cloud (Dinh, Lee, Niyato, & Wang, 2011).

In this paper we propose a mobile cloud robotics architecture that will integrate concepts from cloud robotics and mobile cloud. For practical and pedagogical purposes we designed this architecture to work with the popular and relatively inexpensive Lego NXT robot. The idea is to allow the robots to communicate among themselves and also to connect to a cloud computing platform. We believe this architecture will provide a platform that will ease the development and deployment of applications and be especially useful for research projects involving mobile applications and robotics as well as for teaching purposes to introduce students to the mobile cloud and cloud robotics fields. The rest of the paper is organized as follows: the next section presents background information on different tools used in this project. The mobile cloud robotics architecture section details the architecture models we are proposing followed by the applications section that lists some of the projects we are working on where this architecture will be applied. Finally we presents our conclusions & future work.

BACKGROUND

In this paper we propose a mobile cloud robotics architecture that will facilitate the research, development, and testing of applications in fields such as mobile cloud and cloud robotics. In this work several technologies are being used including the Lego NXT robot, the Lego Java Operating System (leJOS), a Python interpreter, and the Seattle cloud computing platform. The next paragraphs provide background information about some of these tools.

Lego NXT Robot

The Lego NXT robot contains a programmable brick with a set of sensors and servo motors. The Lego NXT robot main processor is a 32-bit ARM with 256 kilobytes of flash memory for program storage, and 64 kilobytes of RAM for data storage during program execution. The Lego NXT robot contains a programmable brick, three servo motors, one light sensor, one ultrasonic sensor, one sound sensor, and two touch sensors. The NXT brick contains three output ports used to power the motors and four input ports used to connect the sensors. To get data from the sensors, an extra processor is included with 4 kilobytes of flash memory and 512 bytes of RAM. For connectivity the Lego NXT robot supports Bluetooth for wireless, and USB for wired. (Lew, Horton, & Sherriff, 2010)

Lego Java Operating System (leJOS)

The Lego Java Operating System (leJOS) (Pedersen, Nørbjerg, & Scholz, 2009) is an open source firmware replacement for the Lego NXT robot that provides an implementation of a Java Virtual Machine. leJOS supports having an alternative runtime environment that allows programming the robots using the Java programming language. The leJOS system includes a library of Java classes that implement the leJOS NXJ Application Programming Interface (API). This API consists of a library of Java classes for desktop programming that communicates with the brick via Bluetooth or USB. It also includes a set of tools for debugging, firmware replacement, compiling, and uploading programs into the robot. Some of the features supported by leJOS include: object oriented programming support, Java types such as float and String, most of the java.lang, java.util and java.io classes, recursion, threads, synchronization, arrays, exception handling, and a well-documented robotics API (“LeJOS, Java for Lego Mindstorms,” n.d.).

Python

Python is a high-level interpreted programming language that supports multiple programming paradigms and focuses on readability. In this project we are considering several tools to allow execution of Python code within the Lego NXT robot. These tools are Jython, PyMite and NXT-Python. Jython a Python interpreter written in Java (“The Jython Project,” n.d.). PyMite is a lightweight Python interpreter that works on embedded devices (“PyMite: python-on-a-chip,” n.d.). NXT-Python is a library that supports Python commands to control the Lego NXT brick remotely (“nxt-python - A pure-python driver/interface/wrapper for the Lego Mindstorms NXT robot. - Google Project Hosting,” n.d.). Executing Python code is important for using other tools including the Seattle cloud computing platform explained next.

Seattle Cloud Computing Platform

Seattle is an educational platform very useful for teaching concepts in a variety of topics such as cloud computing, networking, and distributed systems. Seattle is a free to use, community-driven effort supported by resources donated by its users (Cappos, Beschastnikh, Krishnamurthy, & Anderson, 2009). To use Seattle the user installs the software onto their computer and assigns a portion of the computer’s resources to it. The user then receives resources from other computers around the world that are also using the Seattle platform. Seattle runs on different operating systems including Windows, Mac OS, Linux, and FreeBSD. It can also run on mobile devices such as Android phones and jail broken iPhones (“Seattle,” n.d.). An attractive feature of Seattle is that programs running on it are sandboxed and securely isolated from the rest of the programs running on the same computer. It also has hard resource guarantees making it difficult for erroneous or malicious code to by-pass them. Seattle users can also run their programs on computers all over the Internet as Seattle is widely deployed on hundreds of computers worldwide. Seattle comes with a simple programming language RePy which is a subset of Python but that is expressive enough to allow users to create interesting applications. Code written for Seattle is portable and runs across all supported systems.

MOBILE CLOUD ROBOTICS ARCHITECTURE

In this section we present a set of models for the mobile cloud robotics architecture we propose. The first model is the Lego NXT Mobile Cloud Robotics Architecture. The second model is an Android Powered Lego NXT Mobile Cloud Robotic Architecture. The third architecture is a hybrid model that combines features of the previous two. For each one of the models, we show the architecture for a single device followed by a diagram of a network of devices under that architectural model.

The first architecture we propose consist of a Lego NXT robot with the following sensors: sound, ultrasonic, touch, and motion. We also used three servo motors. The Lego NXT robot original operating system is replaced with leJOS. In order to execute Python code inside the Lego NXT robot we need to install a Python interpreter. There are several Python interpreters that can be used and currently we are working with Jython and PyMite (Pedersen et al., 2009). The Python interpreter is necessary among other things to support RePy, a Python dialect that the Seattle cloud computing platform understands. In this architecture there is also a server that sits between the Lego NXT robot and the Seattle cloud computing platform. The Lego NXT robot will communicate with the server via Bluetooth. The server will be online and will become a gateway responsible for sending and receiving commands from the Seattle cloud computing platform. This server will also be responsible for performing some local computations. Figure 1 shows and example of this architecture for a single Lego NXT robot connected to the server, and this server connected to the Seattle cloud. This idea can be extended for multi-robotics applications by having several interconnected robots forming a network as depicted in Figure 2. This network of Lego NXT robots will follow the architecture just described by relying on Bluetooth for communicating among themselves and with the server. We belief this interconnectivity will allow for elasticity and scalability by allowing messages to be forwarded to other robots reaching the server eventually.

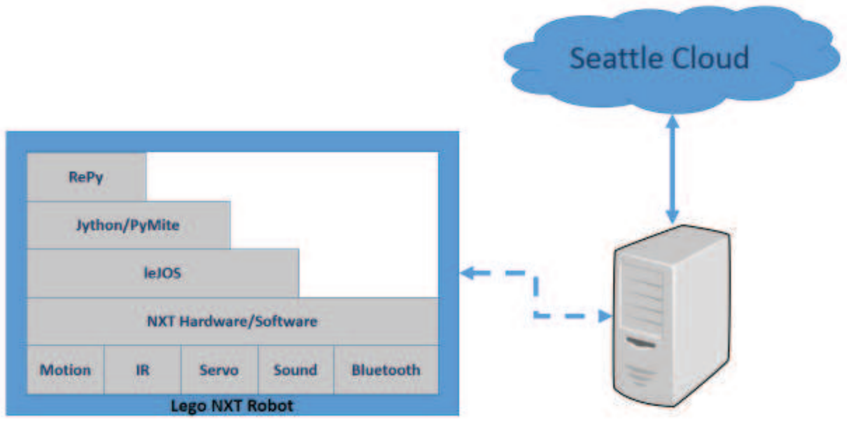

Figure 1: *Model 1: Lego NXT Mobile Cloud Robotics Architecture*

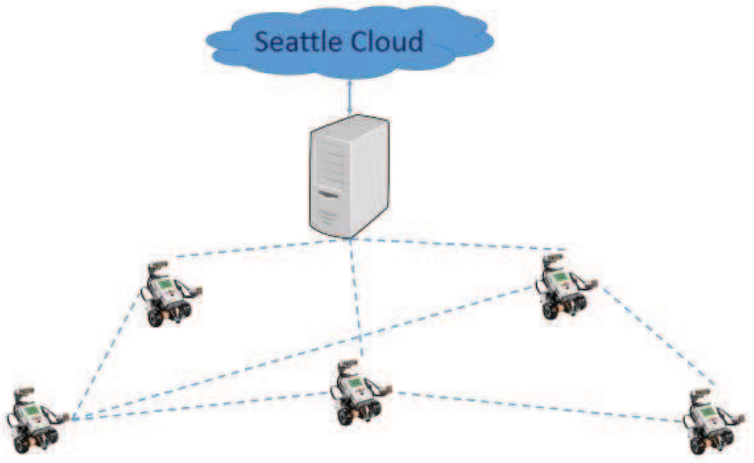

Figure 2: *Network of Robot under the Lego NXT Mobile Cloud Robotics Architecture*

The second architecture we present has a similar configuration as the first one consisting of a Lego NXT robot with sound, ultrasonic, touch, and motion sensors as well as the three servo motors. The operating system running on the Lego NXT robot is leJOS, but this robot is enhanced with an Android device as shown in Figure 3. The Android devices takes care of the tasks such as execution of Python and RePy code and communicating with the server that sits between the Lego NXT robot and the Seattle cloud platform. The Lego NXT robot communicates with the Android device using Bluetooth, while the Android device communicates with the server using Bluetooth or WiFi, as Android devices support both protocols. The server will be online and be responsible for sending and receiving commands from the Seattle cloud computing platform as well as performing some local computations. A network of NXT robots under this architecture is shown in Figure 4. The fact that Android devices supports Bluetooth and Wi-Fi and are capable of being online all the time will allow the Lego NXT robots under this architecture to reach all of their peers in the network resulting in improved connectivity. This configuration will allow for even more elasticity and scalability as now all devices can reach each other and the Lego NXT robots can focus on managing the sensors and the actuators, while the Android device handles communication and computation.

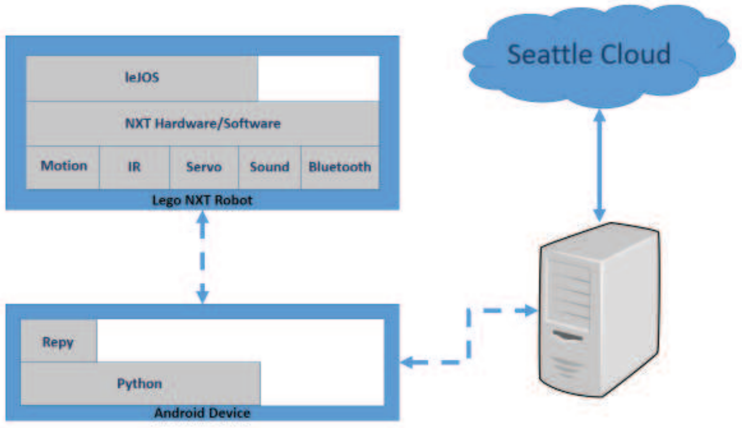

Figure 3: *Model 2: Android Powered Lego NXT Mobile Cloud Robotics Architecture*

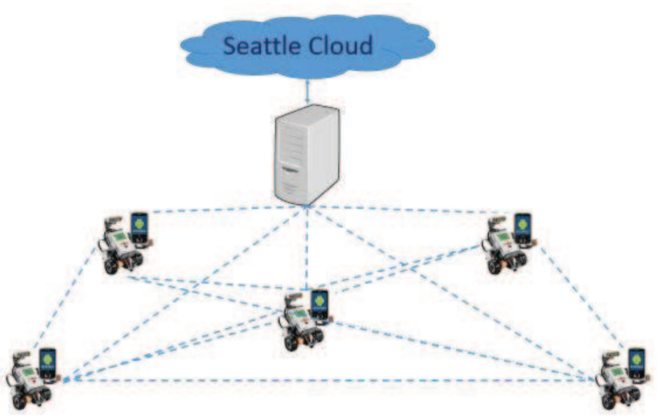

Figure 4: *Network of Robots under the Android Powered Lego NXT Mobile Cloud Robotics Architecture*

The third architecture we propose is a hybrid of the previous two models. This hybrid architecture consist of a Lego NXT robot with sound, ultrasonic, touch, and motion sensors plus the three servo motors. For this model the Lego NXT robot also runs leJOS, and have a Python interpreter installed on it that executes Python and RePy. But for this architecture only a subset of the Lego NXT robots will be powered by an Android device. Figure 5 shows an image of the hybrid architecture. Under this hybrid architecture, the Android powered Lego NXT robots will rely on the Android device to communicate among themselves and with the server, while the robots without an Android device will rely on Bluetooth to communicate with each other and to reach the server. The server handles the communication with the Seattle cloud computing platform and performs some local computations. A network of robots under the hybrid architecture is shown in Figure 6.

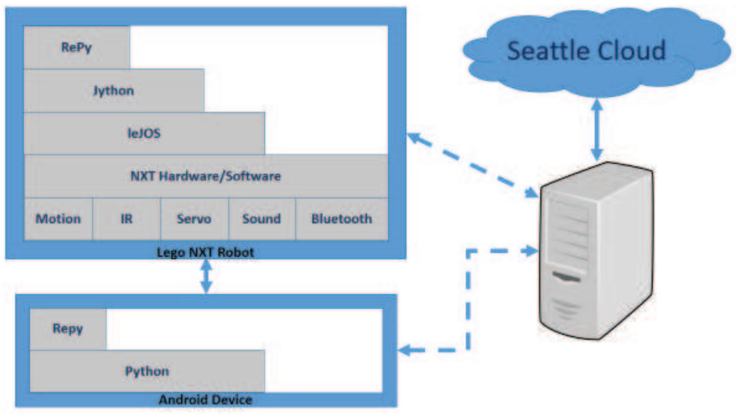

Figure 5: *Model 3: Hybrid Lego NXT Mobile Cloud Robotics Architecture*

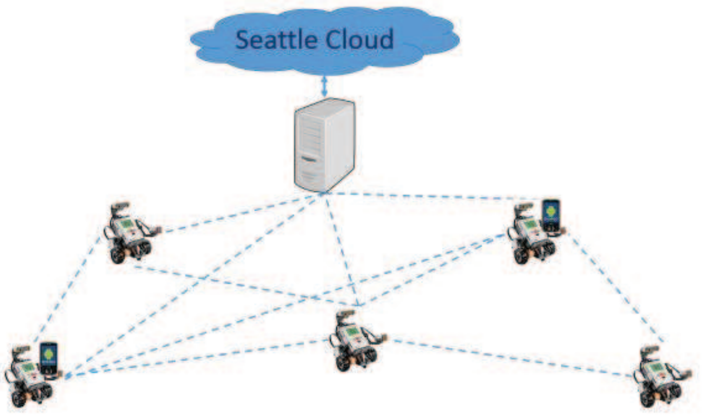

Figure 6: *Network of Robots under the Hybrid Lego NXT Mobile Cloud Robotics Architecture*

ANALYSIS AND DISCUSSION

We propose three mobile cloud robotics architectural models: the Lego NXT Mobile Cloud Robotics Architecture, the Android Powered Lego NXT Mobile Cloud Robotics Architecture, and the Hybrid Lego NXT Mobile Cloud Robotics Architecture. Among the advantages of the first model is that it will maximize the use of the Lego NXT robot by executing Java, Python, and RePy code inside the unit while relying on the cloud for complex computations and storage. However this might come at a performance cost as the processor of the Lego NXT robot is simple, the robot has a very limited memory, and relying only on Bluetooth for communications might create situations in which connectivity is weak or lost disrupting applications. To deal with some of these issues we proposed the Android Powered Lego NXT Mobile Cloud Robotics Architecture. This second model that enhances the Lego NXT robot with an Android device at first this seems like an ideal solution because of the extra computation and communication capabilities that the Android device offers. However this architecture adds another level of complexity by having two different devices. Experts in both, Android and Lego NXT robots will be needed for development, testing, maintenance, and troubleshooting. Also adding Android devices will increase the cost and the development time.

Given the advantages and disadvantages of the previous architectural models we also propose the Hybrid Lego NXT Mobile Cloud Robotics Architecture. Under this hybrid architecture the Lego NXT robots will be able to communicate with each other and the Android powered devices will have extra connectivity. Under the hybrid model there is no exclusive dependence on the Android device as the Lego NXT robots will be able to handle most of the communication and computation themselves, but those enhanced with an Android device will have more computational capabilities. Another consideration is that Android devices can communicate with the Seattle platform as well. So even though communication with the Seattle cloud computing platform will be the responsibility of the server, in case that the server cannot be reached, the Android devices might take over this task potentially eliminating the single point of failure found in the first model. Another important consideration is cost. The first model offers the lowest cost but also less connectivity and computational power, while the second model offers the greatest connectivity and computational power but at a greater cost as each Lego NXT robot has an Android device. Under the hybrid model only a subset of the Lego NXT robots will be powered by an Android device, resulting in improving connectivity and computational capabilities with respect to the first model and at the same time reducing cost with respect to the second model. We believe that this hybrid model offers a good tradeoff between the simplicity and cost of the first model and the connectivity and computational capabilities of the second.

APPLICATIONS

The proposed mobile cloud robotics architecture will provide the necessary tools for developing, deploying, and testing several applications. Currently we are working on different projects where this architecture will become handy. One of these applications is a coordinated detection mechanism. When a robot detects certain phenomena in the environment it might request help from other robots to confirm the phenomena detected. By communicating with the server and the cloud, other robots can be instructed to look for the phenomena as well. This can be useful to detect hazardous conditions such as a smoke, fire, or noise. If the phenomena is confirmed by other units then depending on the intention of the application, the robots might move closer or further away from the area.

Another application is in the field of swarm robotics, a paradigm for the coordination of multiple robots based on local interactions using simple individual robotic nodes. This approach emerged from the field of artificial swarm intelligence, as well as the biological studies of insects, ants and other fields in nature (Holland & Melhuish, 1999). The cloud infrastructure may be used to emulate the environment where artificial pheromones (e.g. light, temperature, and visual marks) are stored by the Lego NXT robots. Thus, in a cooperative and distributed way the Lego NXT robots can create a virtual map of the environment using their sensing, computational, and position capabilities. Applications of this approach include smart-cleaning, efficient patrolling, and efficient exploring of unknown spaces (Altshuler, Bruckstein, & Wagner, 2005).

This architecture will also be useful for pedagogical purposes. The Lego NXT robot and the leJOS operating system has been used successfully in advanced software engineering courses (Lew et al., 2010). Also students seem to be very interested in mobile devices. This architecture will facilitate teaching students the basics of robotics, cloud computing, and mobile devices programming. It will also provide the tools for students to work in interdisciplinary problems and gain experience in these fields.

CONCLUSIONS & FUTURE WORK

In this work we proposed an architecture for having a mobile cloud robotics infrastructure using the Lego NXT robot as an example. We presented three different models: the Lego NXT Mobile Cloud Robotics Architecture, the Android Powered Lego NXT Mobile Cloud Robotics Architecture, and the Hybrid Lego NXT Mobile Cloud Robotics Architecture. We argue that the Hybrid Lego NXT Mobile Cloud Robotics Architecture offers the best tradeoff between the simplicity and cost of the first model and the connectivity and computational capabilities of the second. We also showed different application where this architecture will be relevant including a coordinated detection mechanism, swarm robotics, and for teaching advanced topics in software engineering. Future work includes finishing the implementation of some features of the architecture, debugging, and more testing. We are also looking forward to integrate other technologies such as the Raspberry Pi and Arduino. Performance analysis and implementation of some of the applications is on-going work.

REFERENCES

Altshuler, Y., Bruckstein, A. M., & Wagner, I. A. (2005) Swarm robotics for a dynamic cleaning problem. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005* (pp. 209–216). doi:10.1109/SIS.2005.1501624

Cappos, J., Beschastnikh, I., Krishnamurthy, A., & Anderson, T. (2009) Seattle: a platform for educational cloud computing. In *Proceedings of the 40th ACM technical symposium on Computer science education* (pp. 111–115). New York, NY, USA: ACM. doi:10.1145/1508865.1508905

Chen, Y., Du, Z., & García-Acosta, M. (2010) Robot as a Service in Cloud Computing. In *2010 Fifth IEEE International Symposium on Service Oriented System Engineering (SOSE)* (pp. 151–158). doi:10.1109/SOSE.2010.44

Dinh, H. T., Lee, C., Niyato, D., & Wang, P. (2011) A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless Communications and Mobile Computing*, n/a–n/a. doi:10.1002/wcm.1203

Holland, O., & Melhuish, C. (1999) Stigmergy, Self-Organization, and Sorting in Collective Robotics. *Artificial Life*, *5*(2), 173–202. doi:10.1162/106454699568737

Hu, G., Tay, W.-P., & Wen, Y. (2012) Cloud robotics: architecture, challenges and applications. *IEEE Network*, *26*(3), 21–28. doi:10.1109/MNET.2012.6201212

LeJOS, Java for Lego Mindstorms. (n.d.). Retrieved October 22, 2013, from http://www.lejos.org/

Lew, M. W., Horton, T. B., & Sherriff, M. S. (2010) Using LEGO MINDSTORMS NXT and LEJOS in an Advanced Software Engineering Course. In *2010 23rd IEEE Conference on Software Engineering Education and Training (CSEE T)* (pp. 121–128). doi:10.1109/CSEET.2010.31

nxt-python - A pure-python driver/interface/wrapper for the Lego Mindstorms NXT robot. - Google Project Hosting. (n.d.). Retrieved November 11, 2013, from http://code.google.com/p/nxt-python/

Pedersen, R. U., Nørbjerg, J., & Scholz, M. P. (2009) Embedded programming education with Lego Mindstorms NXT using Java (leJOS), Eclipse (XPairtise), and Python (PyMite). In *Proceedings of the 2009 Workshop on Embedded Systems Education* (pp. 50–55). New York, NY, USA: ACM. doi:10.1145/1719010.1719019

PyMite: python-on-a-chip. (n.d.). Retrieved November 11, 2013, from http://code.google.com/p/python-on-a-chip/

Seattle. (n.d.). Retrieved October 23, 2013, from https://seattle.poly.edu/html/

The Jython Project. (n.d.). Retrieved November 11, 2013, from http://www.jython.org/