Design of a User-Centric RaaS Application

C. Brooks
Western Kentucky University
Computer Science Department
connor.brooks499@topper.wku.edu

W. Seymour
Western Kentucky University
Computer Science Department
william.seymour656@topper.wku.edu

M. Galloway
Western Kentucky University
Computer Science Department
jeffrey.galloway@wku.edu

ABSTRACT

The cost and training to properly operate consumer-grade robots creates a potential obstacle to users that only require specific data or functionality from these robots. The Robot as a service (RaaS) paradigm models robots in a service oriented architecture. In this project, we create an RaaS application that enables users to request specific data or functionality from robots without having to consider the specifics of controlling these robots. We design a framework to be used in an RaaS application that allows users access to data and functionality through a web application. The system is designed to support an initial use case of aerial photography requests. Considerations are taken to make this system require minimal human intervention for successful operation. Future work on the project is discussed, including the necessary progress for full implementation of the system, future experiments, and potential further expansions of the project.

CCS CONCEPTS

• **Computer systems organization ~ External interfaces for robotics** • *Networks ~ Cloud computing*

KEYWORDS

Robots as a Service, Cloud Computing, Robotics

1 INTRODUCTION

As consumer robots continue to rise in popularity, the control of these robots is increasingly allocated to individuals with little to no training. In some cases, such as toy drones, consumers purchase robots explicitly with the intent of controlling these robots. In other cases, such as aerial photography, the product desired by consumers is either the data collected or the function performed by the robot. In cases that fall into the latter category, the process of learning to control the robot is an obstacle to consumers receiving the data they desire.

Additionally, some use cases for consumer-grade robots involve only periodic or even one-time collection of data or performance of a function. An example of such a use case is the collection of photographs or other data surrounding a specific event. In this scenario, the purchasing of a robot specifically for this limited use is undesirable due to the cost of the robot. Users in such scenarios would benefit from the capability to "rent" a robot's functionality for a limited period of time.

The obstacles in both of these cases can be addressed by the introduction of a system which coordinates registered robots to fulfill user requests. This system requires the capability to keep track of multiple robots and allocate missions to a specific robot (or group of robots). This involves translating user requests into specific missions to be completed and maintaining states for each connected robot.

While a robot control system as described could be used in conjunction with professional "robot pilots", advances in autonomous systems allow for consideration of a control system with minimal human supervision required. In this case, a mission must be translated into a sequence of commands to be followed, then uploaded to the allocated robot. Each robot must have some onboard system that enables it to autonomously complete missions of the supported types and return to dedicated base stations. Then, in the case of data retrieval, data must be uploaded to the control system, processed (if necessary), and given to the user.

An applicable concept to the implementation of such a system is that of Robots as a Service (RaaS) [1]. RaaS models robots as a unit in a service-oriented architecture. Through the approach of viewing robots as a service unit in a cloud computing paradigm, RaaS provides a system design that allows users to interact with robots through an abstracted layer.

The proposed system follows the RaaS model by providing a system architecture that fully abstracts robot data and functionality from direct control of the robot. In addition to the concept of RaaS, another applicable concept is that of cloud robotics. Cloud resources could be used during missions by the robots to process sensor data.

The rest of the paper is organized as follows: Section 2 covers related work on this topic. Section 3 lays out the design of the system, both in terms of the application architecture and the design of experimental robots used. Section 4 discusses future work, and since this project is still in progress, this includes experiments to be performed as well as future extensions of the basic project goals. Finally, Section 5 provides the conclusion.

2 RELATED WORK

As described in the previous section, Chen et al. [1] introduced the RaaS architecture for cloud computing. The authors also present a prototype of an RaaS implementation that allows high school students to program robots in C# and Visual Programming Language. Experiments are performed on the CPU usage of the processor used to host the RaaS system.

Kehoe, Patil, Abbeel, and Goldberg [2] surveyed cloud robotic systems. The authors suggest an approach they title Robotics and Automation as a Service (RAaaS) for cloud robotics. RAaaS provides an interface for robot operators to utilize cloud robotics through specifying from available software for robot control.

Liu et al. [3] created a complete framework for Cloud-Enabled Robotic Systems (CERS). The authors make use of the Robot Operating System (ROS) to create a robot network. Users are able to request services, which results in missions being assigned to specific robots. Experiments are done on offloading image processing and video tracking to the cloud.

Koubâa [4] created a system called RoboWeb that virtualizes robotic hardware as resources that can be accessed through the Web. The authors create a service-oriented framework for interacting with robots. Users are able to browse active robots, reserve a specific robot, and perform experiments through uploading and running ROS programs.

Miratabzadeh et al. [5] created a cloud-based architecture for autonomous robots. This architecture is broken into three primary subsystems: a middleware subsystem, a background tasks subsystem, and a control subsystem. The implementation of their system is done using ROS to connect robots to each other and to cloud resources.

The contribution of this project is the design of an RaaS application that focuses on user interaction with autonomous robots for various user requests. This projects provides several areas of study including the translation of user requests into specific missions, the gathering of data on autonomous robots, and the maintenance of autonomous robots with minimal human supervision.

3 DESIGN

3.1 Application Architecture

Our application is designed with an initial use case in mind of gathering photo or videos of a specific geographical location. The application is not limited to this use case, however, and is designed to be expandable. The sequence of actions taken when a user requests data is shown in Fig. 1.

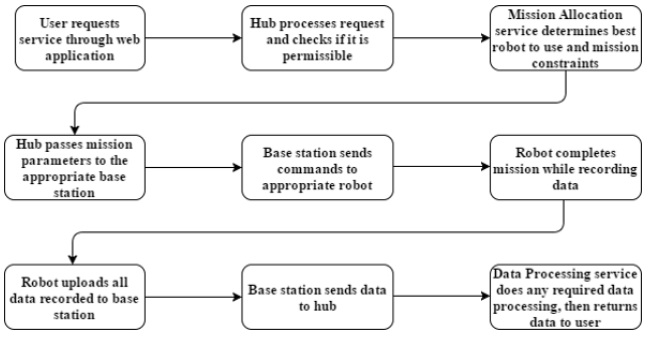

Figure 1: The flow of a use case in which a user requests visual data of a geographical location. The various steps listed in this figure are completed by several different components within the application.

The design of the application involves three top-level components. First, A cloud-based "hub" that hosts the web server, the database of robots, and services such as mission allocation and data processing units. Second, a base station that communicates with local robots and relays information between the hub and individual robots. Finally, the third component is made up of individual robots that have the capability to autonomously complete missions and record required data. This architecture is pictured in Fig. 2.

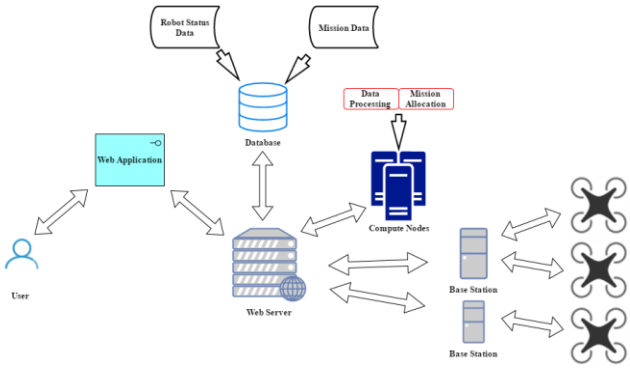

Figure 2: The basic setup of our RaaS application. Users request data through the web application. A mission allocation service uses robot data to both select a robot and create a command sequence for the robot to follow. The appropriate base station interfaces with the robot in order to start the mission and receive data once the mission is completed. Any processing to the data needed is done in the hub, and the user is notified when the final data is ready to be viewed.

3.2 System Setup

The hub will be implemented as a cloud platform with a dedicated virtual machine for the web server. Compute nodes are used to fulfill the mission allocation and data processing services, as well as any other services developed. A database will keep track of robot status and location by base station. Base stations will consist of local servers at designated robot storage areas.

A ROS layer will be used to facilitate communication between the hub and individual base stations. The hub will act as a master node and each base station as a functional node. This allows the base stations to communicate with one another through the hub. Such communication will take place in instances where a robot takes off from one base station and ends its mission at another base station. In this case, the end base station would communicate with the original base station once the robot so that the original base station can drop the robot from its list of local robots.

Communication between the base station and robots will also be accomplished using a secondary ROS layer. Individual base stations will act as the master nodes in this layer, with each robot registering as a functional node that publishes and subscribes to its respective base station.

3.3 Robot Design

LIAM – discuss details of the robot, including hardware used and how it will communicate/receive instructions, etc.

4 FUTURE WORK

4.1 Intended Experiments

This project's intent is to create an RaaS application with a focus on ease of usability and minimal human intervention required. Experiments to be conducted include investigating load balancing in the hub, methods of allocating missions to specific robots and generating a sequence of commands for a robot to follow, and determining how to charge a user based on the cost of completing the mission and uploading data to the hub.

Since an additional area of focus for this project is requiring minimal human intervention in the system during operation, experiments will also be done on monitoring battery life of robots. For fully autonomous operation with no human intervention during day-to-day operation, a method of autonomously charging the robots must be implemented. This must take place at the base stations, requiring further development of the implementation of these stations. Until this is done, human supervisors can be notified when a robot's battery is running low in order to manually charge the battery.

Finally, one other area for future experiments is data sharing between robots. Since these robots do not exist in insolation, it would be advantageous to leverage the fleet of robots in certain cases. Potential experiments could include reuse of navigation data for similar missions.

4.2 Security

Security concerns in this project are twofold. First, proper security practices are necessary to prevent users from accessing data they should not be able to access or breaking into the system in any way. This is common to most applications.

However, this project provides a second type of security concern, which is that of preventing users from using robots to collect data or perform a function that is illegal or undesirable. One primary example in the sample use case of aerial photography is that users should not be able to request photographs or video of private property that they do not own. Such data collection could constitute privacy violations, and a system that allows users to request real-world data must take this into account when fulfilling user requests.

Some potential methods of addressing this concern include forbidding data collection of any data that is not publicly allowable or creating a user privilege system that contains an authorization process for requesting private data or certain sensitive functionalities.

4.3 Heterogeneous Robots

One large extension of the project that will be done is to increase the types of robots and the types of services included in the system. While our initial prototype for the system uses a single type of robot and a single type of user request, further progress in the project will incorporate robots with varying capabilities and designs. The types of data or functionality able to be requested by users will be increased with the introduction of heterogenous robots.

The introduction of heterogeneous robots will also affect the mission allocation service. This service must then select from robots able to complete the required type of mission, and may need to translate mission commands into different forms based on the type of robot selected for the mission. A potential solution to this problem is the introduction of a driver layer that translates general mission commands for a specific type of robot. Such a layer could be placed either in the base station or in the hub as a separate service.

5 Conclusion

In this project, we work to create a system that acts as an RaaS application to allow users to interact with robots through simple requesting of data or services. An initial framework of the system has been designed and explained.

Future work in the project includes the full implementation of this system, as well as experiments into topics such as autonomous battery monitoring and charging, cost prediction of robot missions, and leveraging fleet data. Extensions of the project include further security measures to prevent users from abusing the system and the introduction of heterogeneous robots with varying capabilities.

REFERENCES

[1] Chen, Y., Du, Z., and Garcia-Acosta, M. 2010. Robot as a service in cloud computing. In *Proceeding of the 2010 Fifth IEEE International Symposium on Service Oriented System Engineering* (pp. 151 - 158). IEEE.

[2] Kehoe, B., Patil, S., Abbeel, P., and Goldberg, K. 2015. A survey of research on cloud robotics and automation. *IEEE Transactions on Automation Science and Engineering, 12(2)*, pp. 398-409. IEEE.

[3] Liu, B., Chen, Y., Blasch, E., Pham, K., Shen, D., and Chen, G. 2014. A holistic cloud-enabled robotics system for real-time video tracking application. In *Future Information Technology* (pp. 455-468). Springer, Berlin Heidelberg.

[4] Koubâa, A. 2014. A service-oriented architecture for virtualizing robots in robot-as-a-service clouds. In *International Conference on Architecture of Computing Systems* (pp. 196-208). Springer International Publishing.

[5] Miratabzadeh, S. A., Gallardo, N., Gamez, N., Haradi, K., Puthussery, A. R., Rad, P., and Jamshidi, M. 2016. Cloud robotics: A software architecture: For heterogeneous large-scale autonomous robots. In *World Automation Congress, 2016* (pp. 1 – 6). IEEE.

[6] Liam's citation (pixhawk?)