# 

Synopsis

Mechanical Archer

[1 Description](#h.abxkijgl9a39)

[2 Requirements for Contribution](#h.7wsmn0h1cmu8)

[3 Continuation](#h.his1zvkmlfs4)

[3.1 Project State](#h.qv3elyw18u6e)

[3.2 Setup](#h.bsme0u6kbhgz)

[3.3 Documentation](#h.veij7l563gng)

[3.4 Known Issues](#h.e2ia1lvcxlve)

[4 Members](#h.5f8r3ysnuykr)

**Last Updated: 15 July 2015**

# 

# 

# 1 Description

The Mechanical Archer project is an effort by the RIT Archery Club to create a remotely fired bow and arrow. It takes the form of a metal frame containing one linear actuator with a firing block to draw, release, and retract the bowstring; one linear actuator to raise and lower the back end of the frame for vertical aiming; and a combined gear motor and rotary encoder on the frame's wheels for horizontal aiming. The actuators and motors are electrically controlled by an Arduino Mega 2560. The Arduino will accept commands from a computer running a C++ network client and state machine, which will in turn accept user input via a local frontend client (Java is the expected language of choice for the client, but this may change). The Society of Software Engineers has supplied the Arduino's Ethernet shield to the Software team for networked communication. This document will provide a high-level overview of the software components for the project.

# 2 Requirements for Contribution

* C/C++ source code editor and compiler (g++ or equivalent)
* (tentative) Java JDK 1.7 or higher (version subject to change).
* Github account and client (or command line access)
* Mechanical Archer Trello board access
* Arduino environment and drivers

# 

# 

# 3 Continuation

## 3.1 Project State

Mechanical Archer Revision 1 succeeded in drawing and firing a bow with a manual reload. Revision 2 added aiming and computer control; however significant issues arose in the machine's consistency. Revision 2.5 is focused on moving the state machine logic to a C++ client on the computer, performing minimal sensor-based (rather than state-based) validation before performing allowing the machine to move. The C# network client used with Revision 2 is being rewritten with a backend in C/C++ and a frontend in Java due to the proficiency of much of the software team in those languages.

## 3.2 Setup

Arduino code can be downloaded from Github, then compiled and flashed to the board using the standard Arduino IDE, assuming the computer used has the appropriate drivers and configuration. The C++ client files can be downloaded from Github and compiled directly with g++ or another appropriate C++ compiler (in the near future, a C++ makefile will be written to simplify the build process). The anticipated Java files can also be downloaded and compiled with the standard javac compiler provided in JDK 1.7 or higher. The Arduino must be connected directly to the computer with an Ethernet cable. On any Unix-derivative operating system, no special requirements are needed to connect the backend and frontend process. Compatibility with Windows is anticipated in the future.

## 3.3 Documentation

* Github

<https://github.com/mja2637/MechArcher>

* Trello (currently inactive)

<https://trello.com/board/mechanical-archer/50666dba653f0b062437f92f>

Access will be granted to Mechanical Archer project members, RIT Archery officers, and Society of Software Engineers officers on request.

## 3.4 Known Issues

Inconsistent

# 4 Members

* Mechanical Archer Project Head
  + Eric Roth - [edr2964@rit.edu](mailto:edr2964@rit.edu)
* Electrical Team
  + Eric Roth (Team Lead)
  + Justin
* Mechanical Team
  + Derek – (Team Lead)
  + Evan Natter – [esn8034@rit.edu](mailto:esn8034@rit.edu)
* Software Team
  + Angel Lomeli (Team Lead) – [axl5646@rit.edu](mailto:axl5646@rit.edu)
  + Evan Natter – [esn8034@rit.edu](mailto:esn8034@rit.edu)
  + Tim Ascencio – [ta3755@rit.edu](mailto:ta3755@rit.edu)
  + Amy Mays [amm9649@rit.edu](mailto:amm9649@rit.edu)