ITC303 Project Proposal

For

Team Decided

Business Case

In today's modern world computing is becoming more fundamental to everything we do, and with that comes a focus on creating highly available and distributed services. With high availability services, many computers work together to make a service achieve the greatest uptime possible, this includes being fault-tolerant to various outage scenarios. Consensus algorithms play a large part into building these systems.

The two most practical features for consensus algorithms are:

- Election of a leader/coordinator in a an environment built to sustain faults, and on-the-fly election of new leaders in cases of fault
- Maintaining a consistent log amongst a distributed network of computers

This proposal covers writing a code library which would allow for an application developer to implement consensus/fault-tolerance into their services as easily as possible at the code level. This could allow for creating highly available services hosted by one company, or due to the consistent log feature amongst networked computers it could allow for services to run distributed amongst an applications users instead of hosted by the creator.

Joshua and Sean 07-03-2018

Project Scope

The project will include the design and writing of a software library, implementing extremely strict testing procedures, making the library publically available, and also releasing the library with an example program with most common use cases implemented.
Due to the technical nature of the project, much deliberation on research and design will be included especially at the start. Being that the target use case of this library will be in fault tolerant systems there will be a large focus placed on testing.

Depending on actual time taken, the project has a range of completion stages possible to end on prematurely if required.

1. Working prototype
    a. Implementing the minimal features required to achieve a node consensus library
    b. All reasonable use cases are unit tested
2. Production level prototype
    a. Implementing as many reasonable features as possible
        i. Better leader selection, identifying ideal next leader
        ii. Network library hooks for using an existing application's networking instead
        iii. Security, preshared key and encryption
        iv. Event logging
        v. Log compaction
        vi. Dynamically resizing cluster
3. Feature full production
    a. Project is confirmed thread safe
    b. All noted features worth adding to library are added
    c. Code is publically released as usable and fully featured
    d. Code is thoroughly unit tested, aiming for 100% code coverage
    e. Relay server released to resolve NAT issues for WAN nodes

Project Feasibility

This project has been chosen due to it's reliability of completion and minimal risk. It's underlying consensus algorithm has been chosen due to it's understandability. It's language, IDE, and version control method have been chosen based on the team's existing skill set which match a target market for the library. Due to it's technical nature, the team was decided to be kept small with only two people, this is to allow for joint design and pair programming. This project is certainly doable with current technology; evidence of this is that it's been previously implemented at various other levels of feature completion and in various other languages. The project is setup with multiple possible premature completion stages so that the project can survive large internal issues and a release a working product, however minimally working, at a "completion" stage can still occur.