Objective: Software Implementation of Efficient Fair Queueing (FQ) Schemes to serve as Analysis/Comparison tool

1. SCOPE

This project aims to implement Fair Queueing Algorithm in original and modified form as described below. Initial work will begin by implementing original FQ algorithm as proposed by Nagle et al. in its original form to simulate under various real world variables.

Practical implementation of FQ algorithm poses various obstacles ranging from scalability to efficiency and space issues. Modified “efficient” implementation of FQ algorithm will consider ways to improve FQ efficiency for high network load by alternate implementation of round robin schemes [3], optimal buffer space requirement by using alternate storage data structures, minimizing CPU utilization by alternate round robin strategy while still guaranteeing close to 100% fairness. [2][4]

2. GOALS

2.1 Research Goals

1. ANALYSIS OF FQ ALGORITHM UNDER MULTIDIMENSIONAL CONSTRAINTS:
   Implementation of FQ Algorithm on custom developed software for unified analysis of FQ algorithm under various real world constraints that can be changed in real time.

2. POSSIBLE IMPROVEMENT IN FQ ALGORITHM EFFICIENCY AND BUFFER REQUIREMENT:
   Analysis and implementation of modified FQ Algorithm to obtain higher efficiency, low buffer requirement, less computing intensive while still maintaining the global invariants of fairness, promptness and buffer distribution.

2.2 Software Development Goals

1. MODULAR OBJECT ORIENTED DESIGN:
   Software will be developed in conformity to industry standard patterns. Forward Engineering tools such as Rational XDE will be used in initial phases of Software Development Life Cycle (SDLC) for generation of code skeleton and UML modeling.

2. SCALEABILITY:
   Software will be build using layered approach thus further implementation can be made on top of its engine so that it can be used to simulate other queueing algorithms in future as well.

3. INTUITIVE INTERFACE AND ANALYSIS PROVISION:
   It is planned that all simulation will be shown graphically on screen and user can change most of the parameters via on screen interface. Status of input queues can be monitored on graph. While throughput, buffer status can also be analyzed on multidimensional graph.

3. PHASES AND DELIVERABLE TIMELINE

Current proposed break up of project in two phases is as follows:
PHASE 1:

- **Delivery Date:** ~7/4/2003
  To be completed by Middle of current quarter.

- **Deliverables:** Software with accompanied demonstration.

- **Expected Functionality:** Developed software will provide the following proposed functionality.
  - FQ implementation as proposed by Nagle et al
  - Real time changeable parameters such as packet size, arrival rate at input etc.
  - Real time graphical analysis facility of performance measures.

PHASE 2

- **Delivery Date:** ~7/5/2003
  To be completed by end of current spring quarter.

- **Deliverables:** Software with accompanied demonstration. Facility for Presentation is requested at end of course.

- **Functionality:** Software at the end of this phase will provide the following proposed functionality:
  - Modified FQ algorithm that may include Weighted Fair Queueing (WFQ), Deficit Round Robin (DRR) or any other possible improvement that is explored during phase 2 study and analysis period.
  - Another possible improvement of FQ algorithm is use of alternate data structure for efficient buffer usage and improved efficiency as proposed by Keshav [2]. Implementation of these high performance data structures will be explored that makes hardware implementation of FQ algorithm possible by decreasing space requirement for routers that service large number of ports.

4. SOFTWARE DEVELOPMENT LIFE CYCLE

Each phase of software development is divided into

4 sub-phases which will consist of industry standard division for iterative development.

![Phase 1 and Phase 2 Timeline](image)

**Fig 1:** Shows expected starting and completion dates of the two phases

Study and Analysis will be based on research papers in field on FQ and self study. Development will be conducted using RAD tools so that main emphasis remains equally balanced towards research and analysis as well as software development. Close to end of SDLC will commensurate the testing phase to rectify any errors.

5. DEVELOPMENT ENVIRONMENT

Software will be developed using Microsoft Visual Studio .NET 7.0.9466

- Language: Microsoft C#
- .NET Platform on Win32 system
- Rational Rose XDE
- Visio Enterprise Architect
- .NET GDI+
6. REFERENCES

[1] Analysis and Simulation for Fair Queueing Algorithm - Demers, Keshav, Shenker