REAL-TIME OPEN SOURCE TRAFFIC CONTROL SOFTWARE FOR THE ADVANCE TRAFFIC CONTROLLER

by

JUSTIN LOGAN KEY
B.S. University of Central Florida, 2004
M.S. University of Central Florida, 2005

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the College of Graduate Studies
at the University of Central Florida
Orlando, Florida

Fall Term
2012

Major Professor: Essam Radwan

ABSTRACT

Under the initiative of Department of Transportation (DOT) a safety-critical, dual redundant, open source traffic signal control application is currently being developed. The system named SCOPE, for Signal Control Program Environment, currently implements standard 8-phase NEMA logic and the National Cooperative Highway Research Program 3-66 preemption logic. SCOPE is designed to be part of the Advanced Traffic Controller (ATC), making use of API standard 2.06b to integrate with the hardware. Safety-critical status is achieved through redundancy of application logic that constantly compares expected signal phase information. From baseline requirements, engineers independently program application code, one using Ada95 and the other using C++.

The Traffic EXperimental Analytical Simulation Model, a microscopic single-intersection vehicular simulation, was used for initial validation and testing of the functionality of the system. The second demonstration of the SCOPE, used actuated detector data collected from a recording of a live intersection. Actuator calls were placed on SCOPE at the same times the vehicles triggered the detectors in the video (assuming the vehicles were not in-queue). Using SCOPE the real-world traffic was not only right-of-way safely yielded, but the traffic flow state time average time in-queue reduced. The final phase of testing will occur when the DOT performs Formal Qualification Testing, which is scheduled for 2013. Upon validation and subsequent release to the open source community SCOPE will provide users the ability to replace the proprietary application software residing in ATC cabinets. Transparency will be provided into another aspect of the traffic control signal thus taking the initiative of ATC one step further.

This dissertation is dedicated to my parents for their love and support throughout my education and my life and to my fiancé for her love and support throughout my Ph.D. studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE: INTRODUCTION

Under the initiative of U.S. Department of Transportation a safety-critical, dual redundant, open source traffic signal control application is currently being developed. The system named SCOPE, for Signal Control Program Environment, currently implements standard 8-phase NEMA logic and some concept of the National Cooperative Highway Research Program 3-66. SCOPE is designed to be part of the Advanced Traffic Controller (ATC), making use of API standard 2.06b to integrate with the hardware. In addition to executing on the ATC platform, SCOPE can run on desktop workstations and PowerPC Linux based prototype boards. It is easily ported to any CPU. Safety-critical status is achieved through redundancy of application logic that constantly compares expected signal phase information. From baseline requirements, engineers independently program application code, one using the strongly typed Ada95 which is popular for mission critical systems and the other using the statically typed C++ which is popular for embedded systems. The Traffic EXperimental Analytical Simulation (TEXAS) Model is currently used for validation and testing with Formal Qualification Testing to occur late in 2011. Upon validation and subsequent release to the open source community SCOPE will provide users the ability to replace the proprietary application software residing in ATC cabinets. Transparency will be provided into another aspect of the traffic control signal thus taking the initiative of ATC one step further.

Traffic Control Signal Definition

Traffic signal controllers are a signaling mechanism positioned at road intersections and pedestrian crosswalks to control competing flows of traffic and ensure that conflicting or dangerous traffic signals are not permitted. They are responsible for synchronizing solid-state lamp switching of any number of traffic lights, also known as traffic signals, and stop lights in an area. Virtually everyone in every city in

the developed world places their own and their passengers' physical safety in a signal's allocation of right-of-way. When properly implemented, traffic signal controllers provide significant decreases in travel time, fuel consumption, and emissions, as well as some increases in safety.

The operation of a traffic signal controller can be described in terms of cycle length, signal phases, offsets, scope and mode. The cycle length is the total time required to complete one sequence of signal phases, and it typically lasts 60 to 120 seconds for a four-legged intersection. A phasing plan defines when a traffic signal changes states. The offset between successive traffic signals is the difference in time between the start of their respective green light states. The scope is the level of interaction the traffic signal controller has with other controllers, for this research the focus will only be on:

- Individual Intersection Control – A single traffic signal operates without affecting the operation of other traffic signals.

Finally traffic signal controllers are categorized by their individual mode of operation, for an individual intersection the operations include:

Pre-timed – The controller sets signal phases and the cycle length based on a predetermined schedule which is created from historical data.

Actuated – Cycle length and phases can be adjusted, with possibility of some phases being skipped, based on traffic flow. The green time for is a function of the traffic flow that can be varied between minimum and maximum lengths.

Semi-Actuated - The major street has a constant green signal except when a demand is registered by the

minor street detector and is best suited for locations with low volume minor street traffic.

Full Actuated - All approaches to an intersection have detectors and assignments of the right of way are made in accordance with traffic demand. This control is best where the demand proportions from each leg of the intersection are less predictable.

Traffic Responsive – A signal, or group of signals, use inputs from detectors to chose an appropriate timing scheme from a library of different schemes. Libraries can be selected based on various data analyzing procedures, whether it is current or future prediction, pattern matching of traffic patterns.

Adaptive Control Strategies (ACS) - The most advanced traffic signals, they receive real-time data through detectors to create a timing plan. No library of timing plans is needed, which is ideal for areas with high rates of growth, where libraries would be outdated frequently.

Need for Traffic Control Signals

As the population continues to grow, the demand on the existing transportation infrastructure will become increasingly hard to meet. With roads and highways unlikely to keep pace due to cost and dwindling land supply, the use of traffic control signals will be critical to operating our current roadway systems at maximum capacity. Traffic signals generally provide the greatest payoff for reducing surface street congestion when compared with other methods, such as widening roads (*1*). These devices can help ease congestion without the cost and environmental impact of road expansion. When properly implemented, traffic signal controllers provide significant decreases in travel time and fuel consumption. These decreases provide a great cost and environmental benefit, as the fuel consumed by vehicles stopping and idling accounts for approximately 40% of network wide vehicular fuel consumption (*2*).

When a traffic control signal is properly timed, it is invaluable for safety of motorists and pedestrians. In particular, signals reduce high-fatality rate accidents such as motorist-pedestrian and right angle (T-bone) collisions (3).

As beneficial as properly placed and maintained traffic control signal can be, an unwarranted or improper timed signal can lead to just the opposite effects. Increases in accident frequency, fuel consumption, delay, disobedience of signals, and use of inadequate alternate routes are all consequences of poor timing (4). Once installed traffic control signals are given little thought, and are often ignored unless citizens complain about their operation. Under normal circumstances new installations, maintenance, and retiming activities are often delayed or canceled due to budget issues. The current economic situation faced by many local governments only amplifies this problem. In fact, more than half of the signals in North America are in need of repair, replacement, or upgrading (*4*).

Many of the nation's signals in need of repair could be improved by updating equipment or by simply adjusting and updating timing plans (*5*). Many agencies have no program for monitoring the applicability of signal timing plans to the current traffic patterns, and it is not uncommon to find agencies that have not re-timed coordinated signals for five years (*6*). However for the vast majority of signals a paradigm change to a modern advance controller would yield the greatest benefit. Despite a slow start, computer models have begun replacing manual settings and optimization of signal timing plans. These powerful models use historical data and computer simulation to create an optimal signal timing plan that either maximizes bandwidth or minimizes total delay. Today's traffic signal control varies in complexity, from simple systems that use pre-timed plans based on historical data, to adaptive, also known as advanced, signal control which optimize in real-time plans for a network of signals according to traffic conditions.

While pre-timed signal plans are expensive to prepare and keep up to date (*7*), moving to a more advanced signal controller scheme is not without expense either. These signals incur a higher up-front cost, and while the maintenance cost may be reduced, proper maintenance is still required. For new signals, local governments are faced with the dilemma of whether or not to stretch their budget to invest in the more expensive system. While officials may understand new signals will save them money in the long run, justifying the cost to citizens and their current year's budget might not be an easy task. For locations with existing infrastructure the dilemma can be even greater, should they remove the old system or hang on to what they have?

No matter what equipment or design decisions are chosen, upgrading a traffic control system has a hefty price tag. The process cannot be done without thorough studies and planning by a qualified traffic engineer. However, the cost associated with signal installation and improvement doesn't have to be as great as it is today. Most traffic signals are proprietary, sole-source acquisitions which tie local governments into long-term contracts. The equipment and contracts have inflated costs, promote non-competitive business tactics, and requires use of their product for future installations. These "tie-in" sales practices have led to litigation, calls for non-proprietary industry standards, and the creation of in-house systems(*8*)(*9*)(*10*).

In addition to inflated costs that proprietary software tends to incur, there also exist problems with the proprietary nature of the software performance data. Statistics such as the number of software faults and conflicting traffic signals are not made public. Conflict Management Units (CMUs) are system-independent electronic devices attached to traffic signal controllers that prevent conflicting signals from being displayed. To the average citizen, CMUs cause the blinking "all-red" signal, that they see every

once in a while when the traffic signal is out. While CMUs prevent accidents from occurring, they provide little insight into the software logic causing the failure. That data remains property of provider, and while the CMU makes the system safety-critical the software is not.

CHAPTER TWO: LETRITURE REVIEW

Recent Strides for Controller Openness

In 2000 when the city of New York went to upgrade their aging traffic control signals they made sure to protect the interests of the City over the long term. Having begun to upgrade their system in the past and stopped due to ballooning costs they knew not to repeat their mistakes. This time the vendor was required to provide the source code and a full development system. All equipment had to adopt the National Transportation Communications for ITS Protocol (NTCIP) standards for actuated signal control, no "custom" (or semi-custom) implementations of NTCIP would be allowed. The City and subsequent third parties were allowed to make modifications to the software. Ownership of the software was left to the vendor; license rights allowed the City to deploy the software or derivative products at all intersections within the City. While making a significant push for openness in software, the solution fell short on adhering to a standard on the controller front. The city, “decided that the 2070 construct (with all internally interchangeable modules) was too expensive for the relatively simple intersections within the City.” (*8*) They mandated a functional and size compatible solution, but the internal construction, including the processor and memory, was left to the vendor.

In 2004 the State Attorney General Bill Lockyer filed an antitrust lawsuit against Econolite Control Products claiming that the “tie-in” sales practice of traffic signal equipment was illegal. The lawsuit stated Econolite was forcing contractors bidding on public projects to buy certain equipment at inflated prices as a condition for using other "proprietary" equipment which only the company can provide and this had cost Southern California taxpayers millions of dollars (10). A spokesman for Lockyer’s office stated, “when electrical contractors pay inflated prices…they are passed on to the taxpayers. Primarily what we're looking at is to make sure that these guys stop so that taxpayers don't get stuck with the

tab." (10) According to the lawsuit the use of the Econolite signal controller required use of Econolite signal lights and preemption systems that may have been purchased elsewhere for less. In 2006, a court agreed and found Econolite liable for the use of “tie-in” sales practices (11).

Need for Redundant Traffic Control Signal Software

Present day traffic control systems are essentially ancillary in nature, i.e. they do not directly control the motorist but simply provide indication (*12*). However a faulty indication, as say with conflicting green signals, could easily cause a hazard resulting in any combination of property damage, serious injury, or death. The traffic signal’s ability to provide control or mitigation of hazards is the reason it is considered safety-critical. For many safety critical systems, redundancy is the only acceptable method to achieve high operational reliability (*13*), yet the capabilities of existing traffic controllers do not include software redundancy.

Safety Critical Software

Medical devices, aircraft flight control, weapons, and nuclear reactors are common examples of safety-critical software as their failure would most likely directly lead to the loss of human life. However some examples may be less obvious, from out of order traffic lights that contribute to individuals being involved in an accident, to structural engineering design tools whose fault leads to a building collapse, to a bug in the compiler used in the create another piece of safety-critical code. A safety-critical system is simply any system whose failure could result in loss of life, significant property damage, or damage to the environment (14). Simply put safety-critical software must be reliable because someone’s life depends on it.

Creating Safety Critical Software

The usual method to attain reliability of software operation is fault-avoidance (or intolerance) (*15*). This simply means that all defects are eliminated prior to the software being fielded. However in most software the elimination of all defects is never attained and a crash or an erroneous result is inevitable. This observation leads to the conjecture that for reliable software operation, redundant software in some form is required to detect, to isolate, or recover from effects of undetected software defects (*16*). Three important methods of creating fault-tolerant software systems have been developed, namely N-Version Programming, Recovery Block, and Consensus Recovery Block (*13*). For all these methods, increasing the redundancy, within the software system is essential.

N-Version Programming

N-version programming (NVP) is defined as the independent generation of $N \geq 2$ functionally equivalent programs from the same initial specification. More simply stated, NVP is a method or process in software engineering where multiple functionally equivalent programs are independently generated from the same initial specifications (17). The major objectives of the NVP process are to maximize the independence of version development and to employ design diversity in order to minimize the probability that two or more member versions will produce similar erroneous results that coincide in time for a decision (consensus) action (18). In turn the result is more reliable software operation due to built-in fault tolerance and redundancy.

In general the steps of *N*-version programming are:

Creation of an initial specification of the presenting the functional requirements of the software being developed. The specification should unambiguously define (17):

1. the function to be implemented by an N-version software unit;
2. data formats for the special mechanisms: comparison vectors (“c-vectors"), comparison status indicators (“cs-indicators"), and synchronization mechanisms
3. the cross-check points (“cc-points") for c-vector generation;
4. the comparison (matching or voting) algorithm; and
5. the response to the possible outcomes of matching or voting

Using the specifications, two or more versions of the program are independently developed, each by a group that does not interact with the others. Whenever possible the implementations of these functionally equivalent programs use different algorithms and programming languages (13).

Some *N*-version execution environment (NVX) is developed which runs the *N*-version software and makes final decisions of the *N*-version programs as a whole given the output of each individual *N*-version program (13).

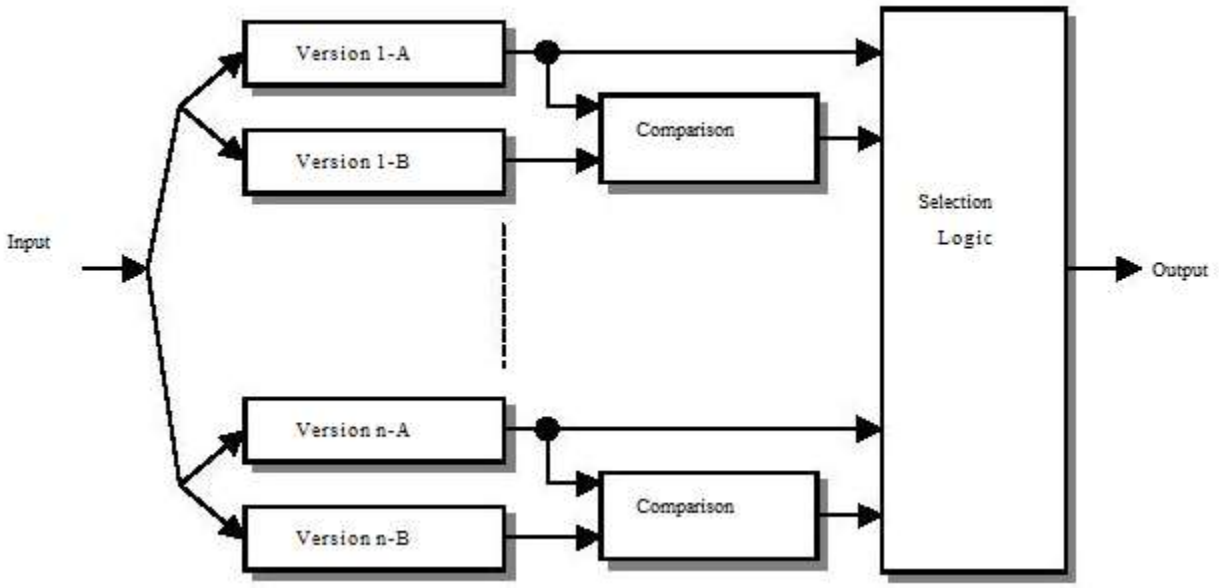


Figure 1: N-Version Programming

Recovery Block / Consensus Recovery Block

The Consensus Recovery Blocks approach combines N-Version Programming and Recovery Blocks to improve the reliability over that achievable by using just one of the approaches. Acceptance tests in the Recovery Blocks suffer from lack of guidelines for their development and a general proneness to design faults due to the inherent difficulty in creating effective tests. The use of voters as in N-Version Programming may not be appropriate in all situations, especially when multiple correct outputs are possible (13). Consensus Recovery Blocks uses a decision algorithm similar to N-Version Programming as a first layer of decision and a second layer using acceptance tests (Recovery Blocks). Although more complex than either of the individual techniques, the this combined approach, if properly implemented, can result in more reliable results.

Signal Timing

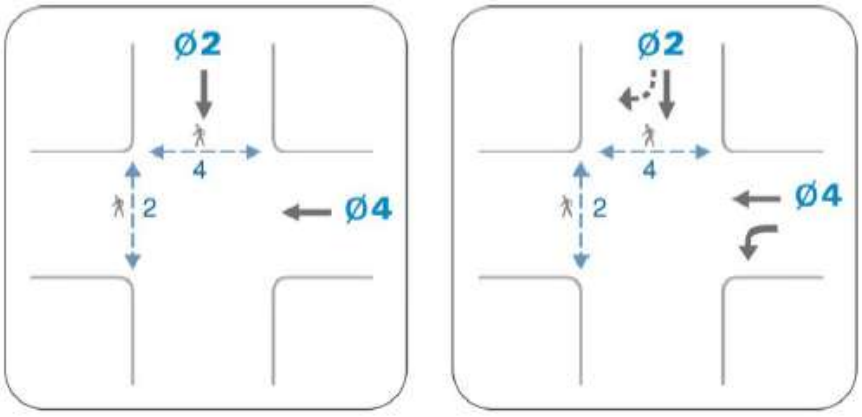
Definitions and Terminology

Traffic signal controllers implement a timing plan that consists of a pre-timed or actuated mode, or a combination of the two. A pre-timed controller has a predetermined and fixed cycle length, phase plan, and phase times. This makes coordinating with adjacent pre-timed signals easy, since the start and end of green are predictable. Pre-timed control is ideally suited to closely spaced intersections where traffic volumes and patterns are consistent on a daily or day-of-week basis. Such conditions are often found in downtown areas. They are also better suited to intersections where three or fewer phases are needed (19). For an actuated controller cycle length, phase plan and phase times are controlled by detector actuations. Phasing represents the fundamental method by which a traffic signal accommodates the various users at an intersection in a safe and efficient manner.

Many of the terms used to describe pre-timed and actuated control are often used incorrectly by professionals or publications (20). The Signal Timing Manual uses the following terminology when describing signal control:

- Phase – the total of the green, red and yellow interval for a given movement(s) (Figure 2a).
- Split – The time assigned to a phase (green and the greater of the yellow plus all-red or the pedestrian walk and clearance times) during coordinated operations. May be expressed in seconds or percent.
- Movement – describes the user type (vehicle or pedestrian) and action (turning movement) at an intersection. Two different types of movements include those that have the right of way and

those that must yield consistent with the rules of the road or the Uniform Vehicle Code (Figure 2b).



(a) (b)

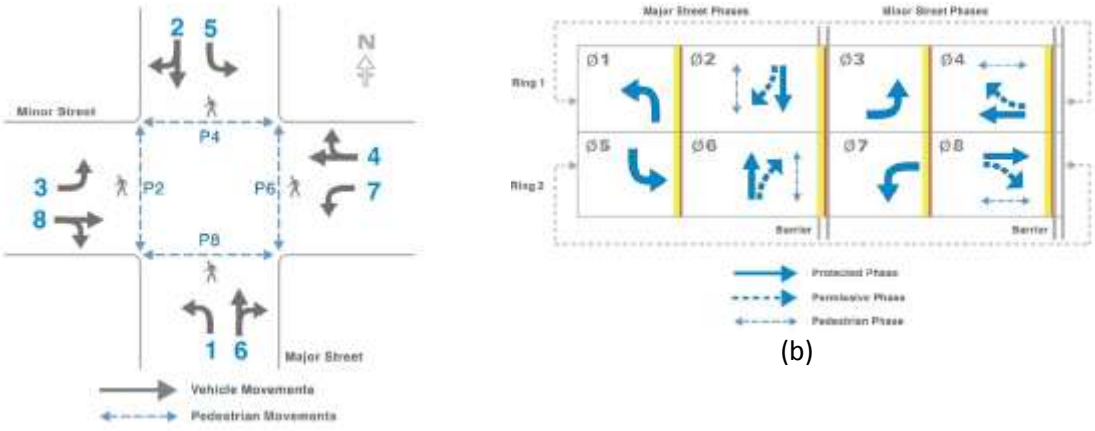
Figure 2: (a) Phasing Diagram (b) Movement Diagram

- Phase Pair – A combination of two phases allowed within the same ring and between the same barriers. For example phase pair 1&2 can operate concurrently with 5&6, and 3&4 can operate concurrently with 7+8 (Figure 3a).
- Ring – A series of conflicting phases that operate in sequence.
- Barrier - A separation of intersecting movements in separate rings to prevent operating conflicting phases at the same time.

and the following terms when describing actuated control:

- Minimum Gap - A volume density parameter that specifies the minimum green extension when gap reduction is used.
- Minimum Green - A parameter that defines the shortest allowable duration of the green interval.

- Extend - A detector parameter that increases the duration of a detector actuation by a defined fixed amount.
- Gap Out - A type of actuated operation for a given phase where the phase terminates due to a lack of vehicle calls within a specific period of time (passage time).
- Max Out - A type of actuated operation for a given phase where the phase terminates due to reaching the designated maximum green time for the phase.
- Queue -A line of vehicles, bicycles, or persons waiting to be served by a phase in which the flow rate from the front of the queue determines the average speed within the queue. Slowly moving vehicles or people joining the rear of the queue are usually considered part of the queue.
- *Call* - An indication within a controller that a vehicle or pedestrian is awaiting service from a particular phase or that a recall has been placed on the phase.
- Recall - A call is placed for a specified phase each time the controller is servicing a conflicting phase. This will ensure that the specified phase will be serviced again. Types of recall include soft, minimum, maximum, and pedestrian.
- Minimum Recall - A parameter which results in a phase being called and timed for at least its minimum green time whether or not a vehicle is present.
- Maximum Recall - The maximum recall parameter causes the controller to place a continuous call for vehicle service on the phase. It results in the presentation of the green indication for its maximum duration every cycle as defined by the maximum green parameter for the phase. When the maximum recall parameter is selected for a phase, the maximum green timer begins timing at the beginning of the phase's green interval, regardless of the presence of a conflicting call or lack thereof.



(a) (b)

Figure 3: (a) Phase Pair Diagram (b) Ring And Barrier Diagram

Ring And Barrier Structure

The modern U.S. practice groups phases into a continuous loop (or ring) and separates the crossing or conflicting traffic streams with time between when they are allowed to operate (20). By making movements sequential or adding a barrier between the movements conflicting phases are avoided. In Figure 3b a dual ring controller, which uses a maximum of eight phases (or traffic control lights) to accommodate the eight movements. Ring 1 contains phases 1 through 4, and ring 2 contains phases 5 through 8. The two rings operate independently, except that their control must cross the “barrier” at the same time (20). The barrier separates the east-west movements from the north-south movements so as to operate without giving the right-of-way to conflicting movements at the same time. This allows phase pair 1 and 2 can operate concurrently with phase pair 5+6. Phase pair 3+4 can operate concurrently with phase pair 7+8. These phase pairs are also known as concurrency groups because they can time together.

Actuated Control

Research has shown that the best form of isolated operation occurs when fully-actuated controllers are used. Actuated controllers operate most effectively when timed in a manner that permits them to respond rapidly to fluctuations in vehicle demand (*21*). Basic actuated control relies on the phasing parameters that change in accordance with sensor inputs. The minimum green time attempts to allocate just enough time for stopped vehicles to partially cross the intersection or pedestrians to cross the street. One method that can be used to calculate the minimum green is:

Minimum Green = 5 + 2n

Where:

n is the number of vehicles that can be stored between the stop line and the far detector in one lane. This is determined by dividing the distance (in feet) between the stop line and the detector by 25, since 25 is the average vehicle length plus stopped-headway in feet (22).

Each vehicle requires enough green time to travel from the detector to the intersection. This is referred to as passage time, vehicle extension, or gap. Gap refers to the distance between vehicles as well as the time between vehicles. Each successive vehicle actuation increases the phase green time and when no opposing calls exist, the controller rests. Extensions continue to be timed, but with no effect on the green interval. Passage time is calculated as follows:

Passage Time = D / S

Where:

D is the distance from the stop line to the detector in feet.

S is the speed on the approach in feet per second (22).

When opposing traffic does exist, the maximum time the green interval can be extended is referred to as the maximum green time (Figure 4).

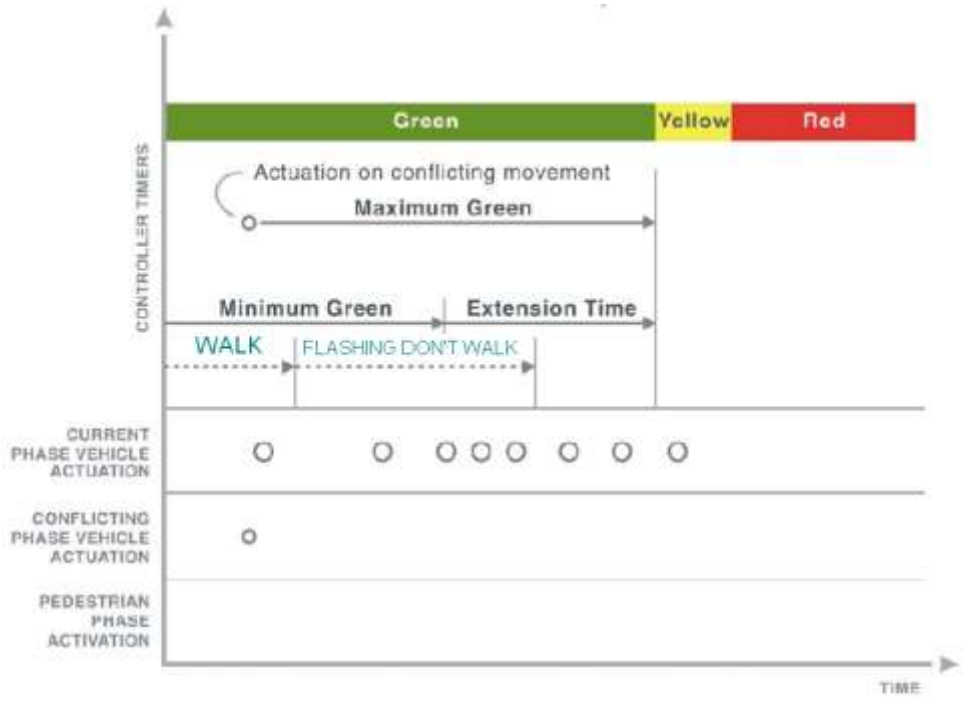


Figure 4: Actuated Interval Concept

When actuation is triggered from another phase, the maximum green timer is started. The current green will be held until the time between actuations is greater than the preset unit extension or gap. If a gap is detected, the yellow change interval will begin and the controller will transition to the next phase in sequence. If the change to the next phase is due to detection of a gap it is referred to as a gap-out, if it is due to the maximum green time being reached it is referred to as termination by maximum green or max-out.

At the completion of the green interval, the yellow interval begins. The yellow signal alerts drivers that the red signals is imminent and proceed if too close to stop other to stop. The following equation is generally used to determine the proper change interval:

Yellow Time = t + S / (2a +- 64.4 g)

Where:

t is the perception/reaction time of the driver in seconds (typically 1.0 second).

S is the speed on the approach in feet per second.

a is the deceleration rate in feet per second (typically taken as 10 feet per second squared).

g = approach grade, percent of grade divided by 100 (add for up-grade and subtract for downgrade) (22).

The all-red interval follows the yellow interval, and gives vehicles in the intersection time to clear before conflicting lanes are given right of way. It is common for the all-red interval to be one to two seconds, but on slower speed approaches, it is not unusual to use a very short duration of 0.0 to 0.5 seconds.

Red Time = (W + L) / S

Where:

W is the Width of intersection in feet.

L is the length of vehicle in feet (typically taken as 20 feet).

S is the speed on the approach in feet per second (22).

Actuated Control Parameters

Research has shown that the best form of isolated operation occurs when fully-actuated controllers are

used (20). Phase recall, passage time, simultaneous gap, and dual entry are all common parameters used when designing an actuated controller. A recall places a call for a conflicting phase, whether there are any detector triggered calls for that phase. According to the NTCIP these phase option parameters have four variations: minimum recall (also known as vehicle recall), maximum recall, pedestrian recall, and soft recall (23).

The most commonly used recall mode is minimum recall, and it is timed for at least the minimum green whether a detector has been triggered or not. The call is cleared upon start of green for the affected phase and placed upon start of the yellow change interval. It is frequently used to give major road traffic the right-of-way regardless of demand on minor roads. Maximum recall places a continuous call on the phase, resulting in the maximum green time. The maximum recall is desirable if a fixed-time operation is desired, detectors are not used or out of service or , or left-turns are associated with thru traffic and gap-out is not desired. Similarly for pedestrian recall, a continuous call is placed for pedestrian service on the phase. The use of the pedestrian recall is applicable when pedestrian detection is not used or out of service, or there is high pedestrian traffic. The soft recall parameter places a call on the phase in the absence of a serviceable conflicting call, resulting in the minimum green time being used. The use of soft recall is applicable when there is a desire to give major-road green time when demand for the conflicting phases is absent.

TS2 Cabinets

The National Electrical Manufacturers Association (NEMA) standardizes equipment used to facilitate and expedite the safe movement of pedestrians and vehicular traffic. As new technologies become available they seek industry help in developing these standards to accommodate technology advances and new

incorporate standard practices. The NEMA Standards Publications define three major specifications for a traffic controller. These specifications include environmental (i.e., temperature, shock, etc.), traffic control logic (i.e,. phases, rings, safety, etc.) and the cabinet interface (i.e., A, B, C connectors and BIU SDLC interface).

Most jurisdictions use NEMA TS-1 and TS-2 or Model 170 actuated controllers (22), with TS2 being the newest standard. TS2 specifies controllers and cabinets more fully than the TS1 or 170/179 standards by covering auxiliary functions. These auxiliary functions include coordination of multiple intersections, preemption for emergency vehicles or trains, time-based control which varies signal coordination throughout the day, and an automatic flash triggered by a manual switch, time switch or system command. It also assures safer operation and enhanced diagnostics when compared to the older TS1 or 170/179 standards (24). However, The TS2 lacks requirements that enable interchangeability of sub-components or software between controllers form different manufacturers (FHWA Traffic control system handbook 2005). The TS2 standards assume that the whole controller will be replaced when the system changes. Controllers that follow the TS 2 standards are called NEMA controllers and the manufacturer provide the software along with the controller. TS2 controllers offer more flexibility in assigning traffic signal phases in order to control many complex or unique situations. There are four timing rings, up to sixteen vehicle and pedestrian phases, and each phase can be assigned to any ring (25).

Evaluation of Existing Traffic Signal Control Programs

Programs were considered to serve as the core of SCOPE and complexity metrics analysis was used to determine their suitability. McCabe metric analysis was chosen over Halstead methods because of the

availability of free analysis tools. A structured software procedure, method, or function should have a McCabe Cyclomatic complexity of less than or equal to 12 (McCabe suggests 10, other projectshave been very successful using a limit of 15). The McCabe complexity is a measure of quantity, where calculations measure the number of linearly independent paths through a program's source code. Another type of metric analysis is Essential Complexity which measures the "quality" of a software system. It is calculated by removing all primitives from a procedure's control flow and then computing the Cyclomatic complexity on what remains. There is no magic number for Essential Complexity.

Summary of Criteria

Although the requirements of the project have changed since the project began, the open source programs where analyzed using the following criteria:

1. Can it be ported to an Advanced Traffic Controller Architecture?
2. Can NCHRP 3-66 concepts be incorporated?
3. Can it be interfaced to CICAS?
4. Can it be interfaced to ACS Lite?
5. Does it have a default steady state?
6. Is the software well documented?
7. Does it make use of exception handling?
8. Does its McCabe Cyclomatic complexity fall within the recommend value of less than or equal to 12?
9. What is its Essential Complexity and how does that compare to others under analysis?
10. Is the nesting level of loops reasonable?
11. Is the error diagnostic system comprehensive and straight forward?

Program 1 - California Partners for Advanced Transit and Highways (PATH)

Dr. Marco Zennaro developed the Berkeley Adaptive Traffic Control System Protocol (Berkeley ATCP2070**)** at the University of California Berkeley. It was developed specifically for the Econolite Model 2070 Advanced Traffic Controller. It was released under GPLv2 in May of 2008 and its current (and only) version is 1.0. According to Dr. Zennaro, it is meant to provide interoperability and scalability. Unfortunately, only the "core" program (batcp.cpp) was available. The core program includes several C++ header files (such as modes.h, types.h, signal.h, and process.h) which are needed for compilation. In addition, the batcp.cpp is coupled to the operating system (OS9) through the include of OS9def.h.

Base Evaluation Criteria

Can it be ported to an Advanced Traffic Controller Architecture?

YES. It already runs on an Advanced Traffic Controller. It is not running under Linux but can be ported.

• ***Can NCHRP 3-66 concepts be incorporated?***

YES, but not easily done. The application software is a single file, batcp.cpp.

• ***Can it be interfaced to CICAS?***

YES. but not easily done. See above (single file problem).

• ***Can it be interfaced to ACS Lite?***

YES. but not easily done. Same as above (single file problem).

• ***Does it have a default steady state?***

YES.

• ***Is the software well documented?***

YES. Yes, the comment to code ratio is 21%.

- ***Does it make use of exception handling?***

NO. There are zero exception handlers. Errors are not caught which could lead to software crashes.

- ***Does its McCabe Cyclomatic complexity of fall within the recommend value of less than or equal to 12?***

NO. The McCabe Cyclomatic complexity for the PATH software averaged 19.94.

- ***Does its McCabe Essential Cyclomatic complexity fall within the recommend value of of less than or equal to 12?***

NOT PERFORMED. The free tool used to perform metric analysis on the PATH program did not contain an essential cyclomatic function.

- ***Is the nesting level of loops reasonable?***

YES . Hand inspection of the software showed no nested looping.

- ***Is the error diagnostic system comprehensive and straight forward?***

NO. There is no diagnostic error system.

Additional Criteria from SCOPE requirements

- ***Is the software Open Source?***

YES. It is released under GPLv2

- ***Does it already contain or use industry standard NTCIP protocols?***

NO. But they can be added (again, not easily)

- ***Can it be integrated or is it integrated with the TEXAS Mode, a single-intersection vehicular traffic simulation, to validate the results?***

YES. Before it is integrated into the TEXAS Model, each procedure in the program would have to be broken out into separate modules. The program does not make use of any object oriented attributes (inheritance, dynamic polymorphism, encapsulation, etc). The program would need to be re-designed and an interface to the TEXAS Model added.

• ***Can it be integrated with CORSIM?***

YES. Last response applies here also.

Program 2 - ATI Dual Redundant Base Software

Advanced Technologies, Incorporated (ATI) developed a dual redundant base traffic intersection controller prototype as part of the Phase I effort for 06-FH1. This prototype could control an intersection and contained the railroad preemption concept of NCHRP 3-66. The prototype used a configuration file to "define" the intersection. The number of intersection approaches, traffic signals, lanes, and crosswalks are all modifiable without software constraints.

Base Evaluation Criteria

- ***Can it be ported to an Advanced Traffic Controller Architecture?***
- **YES**. The software developed by ATI can be ported to any software or hardware environment. It already runs under Linux and Windows.
- ***Can NCHRP 3-66 concepts be incorporated?***
- **YES**. The software developed by ATI has already incorporated the train preemption concept of NCHRP 3-66. The software is modular and object oriented. The employees of ATI who are working on this Phase II effort know the software well because they wrote it.
- ***Can it be interfaced to CICAS?***

- **YES**. Currently, there is not a formal specification for the interface between CICAS-V and the traffic controller. ATI's primary Ada95 software is object oriented. Adding an interface module is doable.
- ***Can it be interfaced to ACS Lite?***
- **YES**. Current implementations of ACS Lite use the NTCIP standard. ATI's Statement of Work states we will be NTCIP compliant.
- ***Does it have a default steady state?***
- **YES**. The current Phase I prototype includes steady state processing of an intersection. However it does not include transitioning to a default state upon detection of errors. Many different error detection techniques are included in the software. A detected error was displayed as a warning (miscompare and/or log message) but the prototype did not drop into the steady state.

• ***Is the software well documented?***

YES. The primary software has a comment to code ratio of 22%. The comment to code ration of the secondary software is 23%. The code is easily understood. Both the secondary software and primary software were written using formal coding standards.

• ***Does it make use of exception handling?***

YES. There are **395** exception handlers in the primary software alone.

• ***Does its McCabe Cyclomatic complexity fall within the recommend value of of less than or equal to 12?***

YES. The McCabe Cyclomatic complexity measurements for ATI's auto-generated and redundant software are well under 12 (1.5 and 1.21 respectively).

• Doe***s its McCabe Essential Cyclomatic complexity fall within the recommend value of of less than or equal to 12?***

YES. The McCabe Essential Complexity measurements for ATI's auto-generated and redundant software are well under 12 (1.5 and 2.27 respectively).

• ***Is the nesting level of loops reasonable?***

YES. The analysis provided by our software case tools showed looping levels of less than 1 for both the auto-generated code and the ATI written code. This implies there is no delay induced because of nested loops.

• ***Is the error diagnostic system comprehensive and straight forward?***

YES. Errors are handled by a central error handling system.

Additional Criteria from SCOPE requirements

• ***Is the software Open Source?***

YES. All software developed by ATI under this contract is by definition open source.

• ***Does it already contain or use industry standard NTCIP protocols?***

NO. NTCIP standards and protocols were not used in Phase I because we did not have to interface to outside systems.

• ***Can it be integrated or is it integrated with the TEXAS Model, a single-intersection vehicular traffic simulation, to validate the results?***

YES. ATI's primary Ada95 software is object oriented. Adding an interface module is simple.

• ***Can it be integrated with CORSIM?***

YES. We will dynamically link interface libraries with CORSIM on a Windows based PC to allow CORSIM to drive our software (containing the TSCP) executing on the development board. This is the way the University of Idaho uses CORSIM.

Program 3 - The InSync Adaptive Traffic Signal Controller

InSync is an adaptive traffic signal system developed by Rhythm Engineering©. The system is claimed to automatically optimize local traffic signals and coordinates signals along roadway arterials based on real-time traffic demand. The system utilizes cameras coupled with image processing of vehicles queues to adjust traffic signal timings in an adaptive fashion. The software is written is written in C++ language and it is a proprietary software (not open source system). The software is capable of communicating with NEMA and 2070 controllers alike (InSync Traffic-Adaptive System White Paper).

When a sensor of this system is placed in emergency/fog mode, InSync will access 4-weeks of historic green split data for specific TOD/DOW at that particular approach. This data is then normalized into a split time to place in the controller until the sensor is functioning again properly. If communications between networked intersections fail, individual processors will continue to perform local optimization functions. Because the software is not open source, it was not considered for SCOPE, however, the functionality of the system was studied.

Program 4 – MIT Intelligent Transportation System Program (MITSIMLab)

MIT's Intelligent Transportation Systems (ITS) program developed the MITSIM Lab to evaluate

the impact of the alternative of the traffic management system design. According to the MIT

Intelligent transportation systems web site, http://mit.edu/its/mitsimlab.html, the software

incorporates a traffic management simulator (TMS) that can be used to evaluate:

1. Ramp control (ramp metering)
2. Freeway mainline control
    a. Lane control signals (LCS)
    b. Variable speed limit signs (VSLS)
    c. Portal signals at tunnel entrances (PS)
3. Intersection control
4. Variable Message signs (VMS)
5. In-vehicle route guidance

The software has an open source version that requires the Linux operation system. It calls for the "Redhat Linux 7.3 distribution" to compile the source code. The files can be downloaded from the MIT's Intelligent Transportation System Program website at: http://mit.edu/its/MITSIMLabOSnew.html. MITSIM was examined to see how other open source traffic programs are implemented Some files contain excellent headers with attributes: Class Name, File Name, Class Type, Derivation, Layered, Friends, C++ Version, Calls to, and Library, while some did not. The software has detailed installation instructions and a 116 page user's manual. Similar to the ATI's Phase I prototype, there is a way to simulate an eight-phase dual-ring traffic signal controller. The Traffic Management System (TMS) portion of this software was considered for the core logic of the SCOPE system. However, the software is extremely complex. The average McCabe complexity figure for the TMS C++ classes is 23.92. That might be overlooked if the code was adequately commented. But, the comment to code ratio is only 10

percent. This is significantly less than both the PATH software and the ATI Phase I prototype software under consideration.

Program 5 – Software Controller Interface Device (CID) II

The National Institute for Advanced Transportation Technology, University of Idaho, developed a real-time interface between a 170, 2070 and NEMA TS 1 and TS 2 traffic controllers and application software running on Windows 98, Windows ME or Windows 2000 (Brian Johnson et al, 2001).

Listed below are applications of the software:

1. A real-time interface between the TSIS/CORSIM traffic simulation running on a computer and 170, 2070 and NEMA TS1 and TS2 traffic controllers (hardware-in the-loop simulation). The simulation runs with the real traffic controller instead of a generic model in the simulation, resulting in more realistic simulations that can be used to test traffic signal plans or train new engineers.
2. A suitcase tester, in which a laptop computer and a CID are used to test the settings of a traffic controller and simulate full operation of the controller. This allows signal timing and progression to be checked under multiple scenarios prior to field installation.
3. A hardware tester that can be used to test the operation of the CID periodically and test the continuity in the cables connecting the CID to the traffic controller.

In addition, the AASHTO Green book and the MUTCD were reviewed, both books only include suggestions for the logic to be used in the signal operation and the signal timing, but there was no mention of the software operating traffic signal controllers.

Summary of Research Findings (Pros and Cons)

INSYNC ADAPTIVE TRAFFIC SIGNAL CONTROLLER

Not open source, was not considered to be used as base software for SCOPE.

MITSIMLab

Open source, however, not considered for core base logic because of the complexity of the software and lack of extensive comments that might overcome the complexity.

PATH SOFTWARE

PROS:

- The main benefit to the software developed by Dr. Marco Zennaro is it has been run on an Advanced Traffic Controller (Econolite Model 2070).
- The software is well commented.
- The software creates an ATCP sensor server, an ATCP actuator server and a “lookup” server.

CONS:

- The software uses hard-coded strings to specify paths.
- There is no error handling.
- It is not POSIX compliant.
- It does not run under Linux but instead is tied to OS9.
- All initialization logic is hard-coded.
- Magic numbers are used.
- Threading not used. Not interruptable (preemptable)

- Most of the logic is contained in a single file that has an C++ extension but does not use C++ the way it is meant to be used.
- No easy method to scale software.

INTERSECTION SOFTWARE DEVELOPED BY ATI:

PROS:

- Safety Critical (Dual Redundant, Software Watchdog Timers, Protected Types, Exception Handling).
- The software is well commented.
- Auto-Generated from UML Design.
- Object Oriented techniques used (inheritance, encapsulation,and association)
- Tasking model allows easy incorporation of preemption.
- The software is not complex based upon metric analysis.
- The software is modular and can be easily interfaced to other systems.
- The software uses an initialization file to define an intersection. This makes scalability simple.
- The software is portable. The primary software already runs under Linux and Windows. It should also run under any POSIX compliant operating system.

CONS:

- The software uses terminology unfamiliar to subject matter experts.
- The dual-redundant approach, while it promotes safety, requires additional independent programmers for the redundant software.

- Headers are currently missing from the redundant implementation.

Software Selection for Phase II

The software developed by ATI during Phase I was selected for Phase II with the following caveats derived from the Task 1 research:

1) Use Canny Quach's in-depth knowledge of the LA-TSCP software to ensure our software contains the same base functionality.
2) Use the interfaces developed by Dr. Marco Zennaro to guide us when porting our finished software to an Advanced Traffic Controller.
3) Use the MIT developed MITSIM documentation as a guide when developing our SCOPE installation instructions and user's manual.

Advantages of Approach

1) ATI's principle investigator and other engineers wrote the software and are intimately familiar with it.
2) Unlike other software examined, the ATI code itself has safety mechanisms built in.
3) The ATI software is the least complex and best documented of all programs evaluated.
4) The ATI primary software is written in Ada95, the same language used in flight control systems, nuclear power plants, and other safety critical applications.
5) The ATI software is extremely portable and is POSIX compliant.

CHAPTER THREE: RESEARCH OBJECTIVES AND METHODOLOGY

Research Objectives And Methodology

The Signal Control Program Environment (SCOPE), is a safety-critical, software logic redundant, open source traffic signal control software initiative currently being developed under the initiative of U.S. Department of Transportation. SCOPE is being developed on Advanced Traffic Controller (ATC) compliant PowerPC hardware. Once complete, SCOPE will provide the ability to replace the proprietary application software residing in ATC cabinets. SCOPE will take the city of New York's solution to traffic control systems one step further, while at the same time increasing the safety of the software. Source code will be made public and communication standards will be followed. The benefits of this approach might include:

- Increased software efficiency as evaluation of logic, algorithms, and design by industry professionals, researchers, and programmers will be made possible.
- Increased software stability and safety as countless developers will have the ability to search code for bugs and exceptions.
- Spurred interest in traffic signal control as software will no longer be proprietary or require large licensing fees.
- Ability of corporations and municipalities to alter traffic control logic to best meet a given city's needs.
- Ability of government traffic engineers to perform signal re-timing.
- Increased competition in signal timing maintenance for localities wishing to use a third party.

SCOPE is unique in the fact that it is open source software, developed using only open source toolsets from requirements though formal qualification testing. By creating SCOPE in this open source fashion, it will provide a platform for industries (see Appendix page 111) and researchers not only to enhance and expand the SCOPE software but to try out and create new sensors and algorithms. The most exciting prospect and budding research could be to use SCOPE to advance the DOT's initiative of the Connected Vehicle. The idea behind the Connected Vehicle Research program is to create interoperable networked wireless communications among vehicles, the infrastructure, and passengers' personal communication devices to make driving safer, smarter and greener [26]. Two-way Digital Short Range Communication provides connectivity between the vehicle and intersection computers warning drivers of upcoming traffic signal changes and possible traffic violations. There is also a possibility of this technology being used for advanced situations such as crash avoidance, cruise control adjustments and autonomous driving. Intellidrive is currently being tested in Texas to help avoid collisions at intersections between emergency vehicles and conflicting traffic. SCOPE can play a major role in expanding the development and creativity of this initiative to smaller entities that would have been barred in the past due to proprietary nature of other systems.

Another characteristic that makes SCOPE unique is that it is safety-critical software that makes use of N-version programming (NVP), or more simply redundant algorithms for fault detection. NVP is defined as the independent generation of N>2 functionally equivalent programs from the same initial specification. More simply stated, NVP is a method or process in software engineering where multiple functionally equivalent programs are independently generated from the same initial specifications (*14*). The major objectives of the NVP process are to maximize the independence of version development and to employ design diversity in order to minimize the probability that two or more member versions will produce

similar erroneous results that coincide in time for a decision (consensus) action (*18*). In turn the result is more reliable software operation due to built-in fault tolerance and redundancy. For SCOPE, NVP is achieved by two independently programmed versions of the traffic signal control logic running concurrently on the ATC. Before the command is issued to execute the next split the results of the Ada95 and C++ logic are compared, with a miscompare resulting in the blinking all-red condition.

Research Tasks / Requirements

The design of SCOPE calls for independently programmed dual redundant timing control logic. The logic controlling the primary software (Ada95) will be completed by engineers at ATI. The secondary logic (C++) and user interfaces are being be developed for this of the candidacy proposal. While SCOPE is part of a large scale project for the DOT with many requirements, the following requirements pertain to the design of the secondary software.

Table 1: Requirements applicable to secondary software for SCOPE project

Requirement	Priority	Verification & Validation
Software shall be open source	High	Inspection
Software shall be dual redundant	Medium	Inspection
Secondary shall be programmed in C++	Medium	Inspection
Secondary shall receive time at 10 Hz rate from primary and perform timing calculations	High	Testing
Secondary shall process heartbeat at 1 Hz rate from primary	Low	Testing
The system shall fall into steady state after 5 miscompares	High	Testing
Watchdog timers shall be used to monitor software	High	Testing
3-legged, 4-legged and Texas Diamond intersection types shall be implemented	High	Testing
Industry standard nomenclature shall be used	High	Inspection
Pre-timed logic shall be implemented	High	Testing
Selected NCHRP 3-66 algorithms shall be implemented	High	Testing
UML shall be used to design code	Medium	Inspection
Configuration management software shall be used for code repository	High	Inspection
Code shall be documented	High	Inspection
Code shall be commented	High	Inspection
Code shall be capable of running on PowerPC platform	High	Testing
Code shall be capable of running on ATC	High	Testing
Code shall be capable of interfacing with Houston cabinet	High	Testing
Code shall be capable of interfacing with TEXAS Model	Medium	Testing

The following requirements were delivered as tasks need to be completed for the research objectives of this candidacy proposal to be met.

Table 2: Derived requirements for completing research objectives

Derived Requirements	Derived Priority	Verification & Validation
Investigate existing traffic control software	High	Inspection
Translate PATH software to C++	High	Inspection
Implement logic for various intersection types	High	Testing
Implement logic for pre-timed control	High	Testing
Implement logic for actuated control	High	Testing
Intergrate with TEXAS Model	Medium	Testing
Create a JAVA GUI for entering parameters	Low	Testing
Create an Android Tablet app for entering parameters	Low	Testing
Port C++ code to PowerPC platform	High	Testing
Port C++ code to ATC	High	Testing
Port C++ to Houston cabinet	High	Testing
Design and build cable to connect PowerPC and Houston cabinet	Medium	Testing

CHAPTER FOUR: SOFTWARE DESIGN

Software Description

Prior to embarking on the design, evaluation of the project requirements and existing traffic control systems was performed. The creation of a large-scale system, needed to integrate with, and make use of, several industry standards required careful planning. The software designed had to produce a safety-critical traffic signal control program that could easily incorporate new algorithms such as those developed under NCHRP 3-66. Slated as the application logic for the ATC, the software was required to use the communication protocol defined by the ATC API standard 2.06b. The decision was made to develop the application using Ada95 as the primary logic, with C++ as the secondary logic. Using a baseline of requirements and an agreed upon interface API, the Ada95 and C++ software was independently developed by two software engineers, one specializing in Ada95 and the other C++. The Unified Modeling Language (UML) tool Umbrello was extensively used to document the requirements and designs, but like the source code was never shared between engineers.

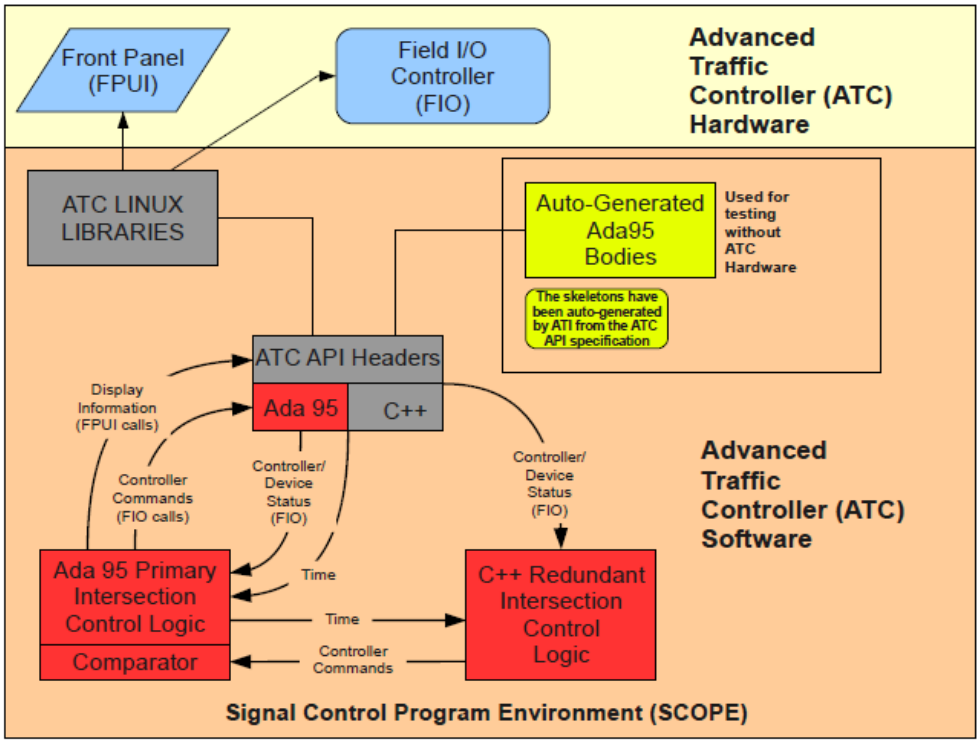


Figure 5: Overview of ATC using SCOPE Application

Ada95/C++ Communication & Synchronization

The Ada95 software (primary) communicates with the C++ software (secondary) through the use of standard socket connections. The primary software acts as a server, creating a socket and accepting connections from the C++ client if it can be found. If a connection is not made after several attempts, the primary software begins execution of the control loop. The primary software still runs and controls the intersection without the redundant software executing. Under normal operating conditions a connection is established and the primary sends the secondary an initial message containing the current time. Periodically, time and request for status packets are sent to the secondary software by the primary software. In turn, the secondary software sets its clock or replies to the status request. The reply status of the secondary is compared to the primary status to determine if the software is in sync and producing

the same results. Up/down redundancy counters and thresholds are used to determine critical faults.

Both the primary software and the secondary software periodically register with watchdog timers set at two second intervals. If the watchdog timer expires without hearing from the primary and secondary software an indication is currently displayed on the GUI, for the testing phase, and later will be logged as a failure to the ATC. A future enhancement being considered for SCOPE is secondary taking the role of the primary if for any reason the primary stopped communication with the watchdog.

Preemption Design

One of the key elements of designing ATC compliant software is the ability to immediately respond to preemption events (train, ambulance, pedestrian, etc). For example, a preemption event would be the approach of an ambulance triggering an “all-red” intersection. Another example would be a pedestrian preemption altering minimum green times to accommodate the pedestrian clearance interval. Phase I of SCOPE incorporated NCHRP 3-66's railroad crossing preemption algorithm. Preemption is handled by checking for an event every 50 milliseconds when the primary software is in steady state. Instead of using hardware interrupts, SCOPE's main thread used the delay feature of Ada95 to release control of the processor and allow external events to be processed by subtasks. This method ensured that the software had no greater than a 50 millisecond response time to a preemption event.

For Phase II, the SCOPE primary processing uses a more deterministic method of execution. Instead of assigning a separate task to every independent traffic phase, SCOPE has a main task that contains a sequential execution loop. The loop executes at an adjustable rate (currently set to 100 ms). Polling for external events occurs at the end of each pass. As part of Phase II, SCOPE's sponsors chose additional NCHRP 3-66 actuated concepts to be incorporated into SCOPE. This six month task began after SCOPE

was integrated on an ATC.

ATC Integration

The ATC provides an open-architecture hardware and software platform to support signal control applications requiring a field-implementable controller. An ATC system consists of three main components: cabinet, software, and controller (27). System standardization is vital to the Intelligence Transportation System's push for next-generation signal control operations. As such, Phase II requires SCOPE be integrated with an ATC compliant hardware controller. For initial development SCOPE was ported to run on a PhyCore MPC5200B PowerPC SBC to simulate the controller. As Phase II progresses, the simulator will need to be replaced to make way for real Peek ATC-1000 hardware (***Figure 6***).

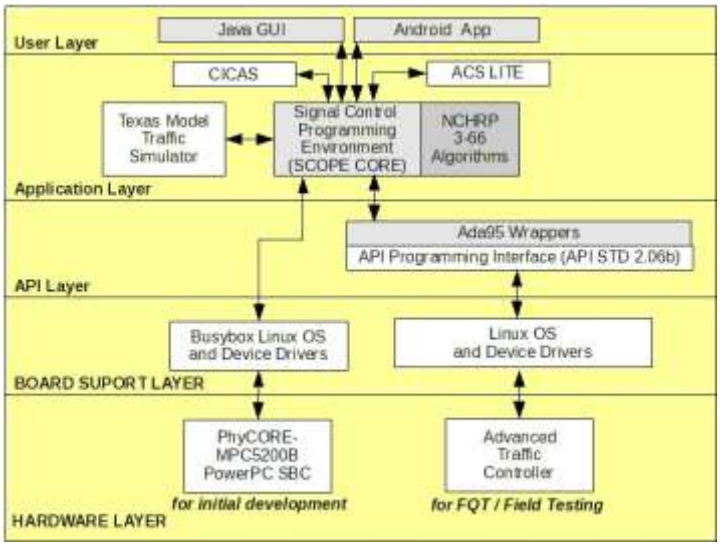


Figure 6: Optimal SCOPE System Overview.

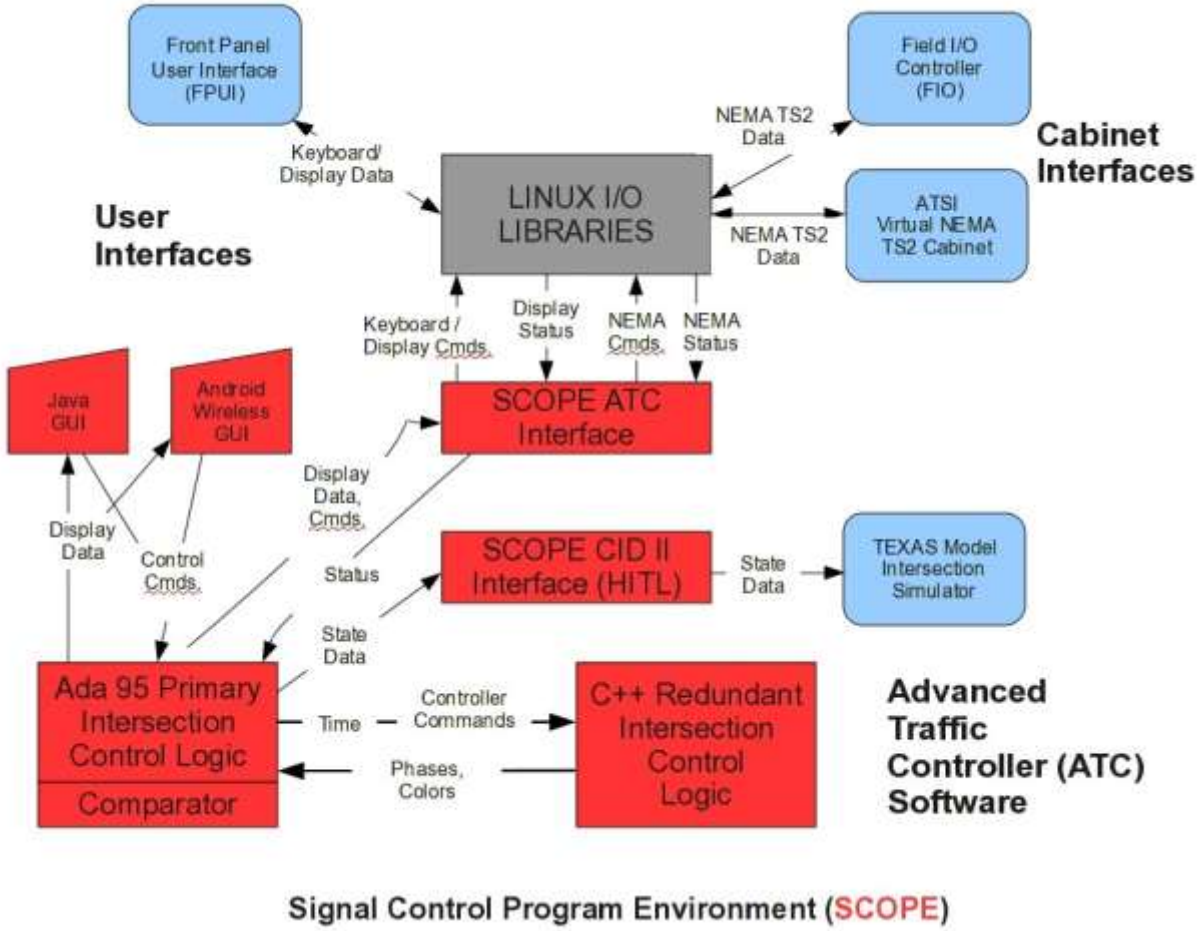


Figure 7: Data Flow Diagram

Interface to Primary Software

This section contains the requirements and definitions for the interface between the Ada95 Primary Signal Intersection Control Program and the C++ Secondary Intersection Control program.

Table 3: Commands

Primary / Secondary Command Names and Values

Name	Value	Definition	Direction
Heartbeat	0	Secondary is Alive	Secondary to Primary
Send Results	1	Request Secondary State	Primary to Secondary
Secondary State	2	Secondary Intersection State	Secondary to Primary
Preempt	3	Stop and Start New State	Primary to Secondary
Min Green Time	4	Set a Minimum Green Time	Primary to Secondary
Max Green Original Value	5	Set a Maximum Green Timeinitial value	Primary to Secondary
True Max Time	6	Set the True Max Green Value	Primary to Secondary
Extension Time	7	Set the initial Extension Time	Primary to Secondary
Consecutive Fails	8	Set the Consecutive Failure Constant Number	Primary to Secondary
Change Clear	9	Set a Clear Value	Primary to Secondary
Change Split	10	Modify a Split Time	Primary to Secondary
Change Mode	11	Switch Processing Mode	Primary to Secondary
Sending Time	12	Current Time	Primary to Secondary
Adjustment	13	Set the Max Green Adjustment Value	Primary to Secondary
Actuated Trigger	14	Extension Green Processing	Primary to Secondary
Actuated Mode	15	Set Actuated Processing Mode	Primary to Secondary
Gap Times	16	Set Gap Times (Actuated)	Primary to Secondary
Time Before Reduction	17	Set Time Before Reduction (Actuated)	Primary to Secondary
Time To Reduce	18	Set Time To Reduce (Actuated)	Primary to Secondary
Min Gap Times	19	Set Minimum Gaps (Actuated)	Primary to Secondary
Sync Message	20	Start of New Green Phase	Primary to Secondary
Debug Level	21	Logging Level (0 off)	Primary to Secondary

Data sent from the Primary to the Secondary

Sending Time

Description: Time is sent to the secondary from the primary at the beginning of the main processing

loop. It is used to keep the secondary software running in lock-step with the primary software. Seconds are seconds from midnight UTC of January 1, 1970 (epoch).

Size: 72 bits (command + data)

Structure:

Table 4: Sending Time Structure

Bits	Description	Possible Value(s)	Data Type
0 .. 7	Command	12	8 bit unsigned
8 .. 43	Seconds	0 .. Max Unsigned	32bit unsigned
44 .. 71	Microseconds	0 .. Max Unsigned	32 bit unsigned

Frequency of Message: Sent once every 100 milliseconds.

Send Results

Description: A command sent from the primary program to the secondary program that instructs the secondary program to send its intersection state values back to the primary. This command is sent when the primary finishes its main processing loop.

Size: 8 bits (command)

Structure:

Table 5: Send Results Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	1	8 bit unsigned

Frequency of Message: Sent once every 100 milliseconds.

Change Clear Time

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify a Yellow or Red clear time. This command is sent when the primary receives a command to change a clear time.

Size: 32 bits (command + data)

Structure:

Table 6: Change Clear Time Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	9	8 bit unsigned
8 .. 15	Clear_Time_To_Change	0 – Red 1 – Yellow	8 bit unsigned
16 .. 31	New_Time	0 – 200	16 bit unsigned

Frequency of Message: Sent on Event.

Change Split

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify a split time. This command is sent when the primary receives a command to change a split time.

Size: 32 bits (command + data)

Structure:

Table 7 : Change Split Time Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	10	8 bit unsigned
8 .. 15	Split_Time_To_Change	0- 15	8 bit unsigned
16 .. 31	New_Time	0 – 200	16 bit unsigned

Frequency of Message: Sent on Event.

Change Mode

Description: A command sent from the primary program to the secondary program that instructs the secondary program to change mode. This command is sent when the primary receives a command to change its processing type (i.e., pre-timed to CICAS).

Size: 32 bits (command + data)

Structure:

Table 8: Change Mode Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	11	8 bit unsigned
8 .. 31	New_Mode	1000 – Pre-timed 1001 – Actuated 1002 – CICAS 1003 - Low Level 1004 – Adaptive	24 bit unsigned

Frequency of Message: Sent on Event.

Preempt

Description: A command sent from the primary program to the secondary program that instructs the secondary program to stop processing and change state. There are several variants. If the immediate indicator is set, the phases immediately transition to the new phases set in the message. If the delay indicator is set, the phases wait for the amount of time specified before changing to the new phase. If the when_done indicator is set, the current phases should complete before changing to the new

specified phases.

Size: 64 bits (command + data)

Structure:

Table 9: Preempt Command Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	3	8 bit unsigned
8 .. 15	Indicator	0001 – Immediate 0002 – Delayed 0003 – When Phase Done	8 bit unsigned
16 .. 47	Delay Time	0.0 – 300.0 seconds	Float
48 .. 55	New_Phase_0	1 – 16	8 bit unsigned
56 .. 63	New_Phase_1	1 – 16	8 bit unsigned

Frequency of Message: Sent on Event.

Min Green Time

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify a minimum green time. This command is only used in actuated mode. It is sent when the primary changes a minimum green time.

Size: 32 bits (command + data)

Structure:

Table 10: Set Minimum Green Time Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	4	8 bit unsigned
8 .. 15	Min_Green_Time_To_	0 – 15	8 bit unsigned

	Change		
16 .. 31	New_Time	0 – 200	16 bit unsigned

Frequency of Message: Sent on Event.

Max Green Original Value

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify a maximum green time. This command is only used in actuated mode. It is sent when the primary changes a maximum green time.

Size: 32 bits (command + data)

Structure:

Table 11: Max Green Original Value Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	5	8 bit unsigned
8 .. 15	Max_Green_Value_To_Change	0 – 15	8 bit unsigned
16 .. 31	New_Value	0 – 200	16 bit unsigned

Frequency of Message: Sent on Event.

True Max Time

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify the true max green time. This command is only used in actuated mode. It is sent when the primary changes a true max green time. The “true max” green time is actually the greatest allowed value for the maximum green time. This maximum green time gets adjusted by an “adjustment” value up to the true max time.

Size: 32 bits (command + data)

Structure:

Table 12: True Max Time Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	6	8 bit unsigned
8 .. 15	True Max Time to Change	0 – 15	8 bit unsigned
16 .. 31	New_Value	0 – 200	16 bit unsigned

Frequency of Message: Sent on Event.

Extension Time

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify an extension time. This command is only used in actuated mode. It is sent when the primary changes an extension time. The extension time is used to extend the minimum green value.

Size: 32 bits (command + data)

Structure:

Table 13: Extension Time Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	7	8 bit unsigned
8 .. 15	Extension Time to Change	0 – 15	8 bit unsigned
16 .. 31	New_Value	0 – 200	16 bit unsigned

Frequency of Message: Sent on Event.

Consecutive Failure Constant

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify the consecutive failure's constant. This command is only used in actuated mode. It is sent when the primary changes the consecutive failures constant.

Size: 32 bits (command + data)

Structure:

Table 14: Consecutive Failure Constant Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	8	8 bit unsigned
8.. 15	Number of Failures before Action	0 – 15	8 bit unsigned
16 .. 31	Spare	N/A	16 bit unsigned

Frequency of Message: Sent on Event.

Adjustment

Description: A command sent from the primary program to the secondary program that instructs the secondary program to modify the minimum green adjustment constant. This command is only used in actuated mode. It is sent when the primary changes the minimum green adjustment constant.

Size: 32 bits (command + data)

Structure:

Table 15 : Adjustment Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	13	8 bit unsigned
8 .. 15	Spare	N/A	16 bit unsigned
16.. 31	Spare	N/A	16 bit unsigned

Frequency of Message: Sent on Event.

Actuated Trigger

Description: A command sent from the primary program to the secondary program that instructs the secondary program to perform extension processing for minimum green.

Size: 32 bits (command + data)

Structure:

Table 16: Actuated Trigger structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	14	8 bit unsigned
8 .. 15	Actuated Trigger_Element	0 – 15	8 bit unsigned
16 .. 31	New_Value	0- The channel has no call – and there has been no change in this status since this frame was last transmitted (no call – no change). 1- The channel has a call - and there has been no change in this status since this frame was last transmitted (constant call - no change). 2- The channel has no call - and there has been a change in this status since this frame was last transmitted (call has gone away). 3- The channel has a call - and there has been a change in this status since this frame was last transmitted (new call).	16 bit unsigned

Frequency of Message: Sent on Event.

Actuated Mode

Description: A command sent from the primary program to the secondary program that instructs the secondary program to perform actuated processing based on the mode sent.

Size: 32 bits (command + data)

Structure:

Table 17: Recall Mode structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	15	8 bit unsigned
8 .. 15	Phase	0 – 15	8 bit unsigned
16 .. 31	New_Value	Presence (0) Recall Min (1) Recall Max (2), Max Out (3), Gap Out (4)	16 bit unsigned

Frequency of Message: Sent on Event.

Gap Time

Description: A command sent from the primary program to the secondary program that instructs the secondary program to set a gap time

Size: 64 bits (command + data)

Structure:

Table 18: Gap Time structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	16	8 bit unsigned
8 .. 15	Phase	0 – 15	8 bit unsigned
16 .. 31	Spare	N/A	16 bit unsigned
32 .. 63	Gap Time	2.0 – 5.0	Float

Frequency of Message: Sent on Event.

Time Before Reduction

Description: A command sent from the primary program to the secondary program that instructs the secondary program to set a reduction wait time

Size: 64 bits (command + data)

Structure:

Table 19: Time Before Reduction structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	17	8 bit unsigned
8 .. 15	Phase	0 – 15	8 bit unsigned
16 .. 31	Spare	N/A	16 bit unsigned
32 .. 63	Time Before Reduction	0.0 – 200.0	Float

Frequency of Message: Sent on Event.

Time To Reduce

Description: A command sent from the primary program to the secondary program that instructs the secondary program to set a reduction time

Size: 64 bits (command + data)

Structure:

Table 20 : Time To Reduce structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	18	8 bit unsigned
8 .. 15	Phase	0 – 15	8 bit unsigned
16 .. 31	Spare	N/A	16 bit unsigned
32 .. 63	Time To Reduce	0.0 – 200.0	Float

Frequency of Message: Sent on Event.

Minimum Gap Time

Description: A command sent from the primary program to the secondary program that instructs the secondary program to set a min gap time

Size: 64 bits (command + data)

Structure:

Table 21: Minimum Gap Time structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	18	8 bit unsigned
8 .. 15	Phase	0 – 15	8 bit unsigned
16 .. 31	Spare	N/A	16 bit unsigned
32 .. 63	Minimum Gap Time	0.0 – 200.0	Float

Frequency of Message: Sent on Event.

Send Sync

Description: A command sent from the primary program to the secondary program that tells the secondary to sync frame processng with the primary.

Size: 32 bits (command + data)

Structure:

Table 22: Send Sync Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	20	8 bit unsigned
8 .. 15	Ring	1 .. 2	8 bit unsigned

Frequency of Message: Sent on Event.

Debug Level

Description: A command sent from the primary program to the secondary program that tells the secondary at what debug level to run at (0 – none). The higher the level, the more information that is logged.

Size: 32 bits (command + data)

Structure:

Table 23: Debug Level Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	21	8 bit unsigned
8 .. 15	Debug Level	0 .. 3	8 bit unsigned

Frequency of Message: Sent on Event

Data sent from the Secondary to the Primary

Heartbeat

Description: An alternating value (0,1) sent to the primary from the secondary to let the primary know the secondary is still executing. When the primary software receives the secondary heartbeat, it makes a call to an internal watchdog timer. If this timer is not called at least every second, a problem is declared.

Size: 16 bits (command+data)

Structure:

Table 24 : Heartbeat Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	Command	0	8 bit unsigned
8 .. 15	Data	0,1	8 bit unsigned

Frequency of Message: Sent once every 900 milliseconds.

Secondary State

Description: Sent from the primary program to the secondary program when the primary requests the secondary's state data. The primary program then uses this data to insure both programs are computing

the same values.

Size: 8 bits (command)

1280 bits (data)

Total – 1288 bits.

Command Structure:

Word Number	Size in Bytes	Size in Bits	Field Name	Possible Value(s)	Data Type
0	1	8	Command	2	unsigned

Data Structure:

Table 25: Secondary State Structure

Word Number	Size in Bytes	Field Name	Possible Value(s)	Data Type
0	64	Phase	1 .. 8	8 by 8 matrix of unsigned
16	64	Splits	0 – 300 (in seconds)	16 element array of float
32	64	New_Splits	0 – 300 (in seconds)	16 element array of float
48	4	Red Clear	0 – 300 (in seconds)	Float
49	4	New Red Clear	0 – 300 (in seconds)	Float
50	4	Yellow Change	0 – 300 (in seconds)	Float
51	4	New Yellow Change	0 – 300 (in seconds)	Float
52	4	Speed	0 – 50 (in seconds?)	Float
53	2	Current Phase	1 through 8 (each element)	2 element array of Unsigned Byte
53	2	Control Mode	1002 - CICAS 1000 – Pre-timed 1004 - Adaptive 1003 – Low	Short integer
54	2	New Control Mode	1002 - CICAS 1000 – Pre-timed 1004 - Adaptive 1003 – Low	Short Integer

Word Number	Size in Bytes	Field Name	Possible Value(s)	Data Type
54	2	Current Color	Red (0), Yellow (1), Green (2)	2 element array of Unsigned Byte
55	1	Status	Initialized (10)	Unsigned Byte
55	3	Spare	N/A	3 Unsigned Bytes

Frequency of Message: Sent on command by primary to secondary

Data sent from the Primary to the GUI

Display Data

Description: Data displayed on the test GUI. This data is sent by the primary to the GUI over a TCP/IP channel. The data is sent once the main processing loop has finished a complete iteration.

Size: 164 bytes

Structure:

Table 26: GUI Display Data Structure

Byte	Field Name	Possible Value(s)	Data Type
1	Display_Pri_Sec_Validation	0 (False), 1 (True)	8 bit Boolean
2	Display_Secondary_Status	0 (False), 1 (True)	8 bit Boolean
3	Display_Status	0 – 255	Byte
4	Display_Detector_Value	0 – 64 (0 = do nothing)	Byte
5	Display_Split_Counter	0 .. Max Float	Float
9	Display_Control_Mode	Pre-timed (1000), Actuated (1001), CICAS (1002) Low_Level (1003) Adaptive (1004)	Short_Integer
11	Display_New_Control_ Mode	Pre-timed (1000) Actuated (1001) CICAS (1002) Low_Level (1003) Adaptive (1004)	Short_Integer
13	Display_Rc	0 .. Max Float	Float
17	Display_New_Rc	0 .. Max Float	Float
21	Display_Yc	0 .. Max Float	Float
25	Display_New_Yc	0 .. Max Float	Float
29	Display_Current_Phase	Each byte ranges 0 – 7	2 Element Array of Byte
31	Display_Current_Color	0,1,2 (Red, Yellow, Green)	2 Element Array of Byte
33	Display_Speed	0 .. Max Float	Float
37	Display_Splits Display Transition Times*	0 .. Max Float	16 Element Array of Float
101	Display_New_Splits Display_Current_Max_ Green_Times*	0 .. Max Float	16 Element Array of Float

Frequency of Message: Sent once every major processing iteration.

Note - In actuated mode, the current transition times are displayed instead of split times. Transition times are equal to maximum green + yellow clear + red clear. In addition, the current value of max green is displayed instead of New Splits.

Data sent from the GUI to the Primary

GUI to Primary

Description: Commands and Data sent from the User GUI to the Primary Ada95 SCOPE program.

Size: 697 Bytes

Structure:

Table 27: GUI To Primary Data Structure

Byte	Field Name	Possible Value(s)	Data Type
1	Simulate_Secondary	0 (False), 1 (True)	8 bit Boolean
2	Forever	0 (False), 1 (True)	8 bit Boolean
3	Stop	0 (False), 1 (True)	8 bit Boolean
4	Output_Interface_Data	0 (False), 1 (True)	8 bit Boolean
5	Time_To_Stop	0 .. Max_Float	32 bit Float
9	Debug_Level	0 .. 3 (None,Low,Med,High)	16 bit ShortInteger
11	Traffic Simulation	0 (False), 1 (True)	8 bit Boolean
12	Start	0 (False), 1 (True)	8 bit Boolean
13	Intersection Type	0 (4 Leg), 1 (3 Leg), 2 (Diamond)	Byte
14	Spare 1	N/A	Byte
15	Spare 2	N/A	Byte
16	Spare 3	N/A	Byte
17	New_Rc	0 .. Max Float	Float
21	New_Yc	0 .. Max Float	Float
25	Spare	0 .. Max Float	Float
29	New_Splits	Each Element 0 .. Max Float	16 Element Array of Float
93	New_Current_Color	0,1,2 (Red, Yellow, Green)	Byte
94	New_Control_Change	0 .. 255	Byte
95	New_Mode	Pre-timed (1000) Actuated (1001) CICAS (1002) Low_Level (1003) Adaptive (1004)	Short_Integer
97	New_Min_Green_ Times	Each Element 0 .. Max Float	16 Element Array of Float
Byte	**Field Name**	**Possible Value(s)**	**Data Type**
161	New_Max_Green_ Original_Times	Each Element 0 .. Max Float	16 Element Array of Float
225	New_True_Mx_Grn_Time	Each Element 0 .. Max Float	16 Element Array of Float

Byte	Field Name	Possible Value(s)	Data Type
289	New_Default_Extension Times	Each Element 0 .. Max Float	16 Element Array of Float
353	New_Consecutive Failures_Allowed	1 .. 10	Integer
357	New_Extension_Time Increment	1.0 .. 100.0	Float
361	New_Actuated_Mode	Presence(0) Recall Min (1) Recall Max (2), Max Out (3), Gap Out (4)	16 Element Array of Byte
377	New_Actuators	0 The channel has no call – and there has been no change in this status since this frame was last transmitted (no call – no change). 1 The channel has a call - and there has been no change in this status since this frame was last transmitted (constant call - no change). 2 The channel has no call - and there has been a change in this status since this frame was last transmitted (call has gone away). 3 The channel has a call - and there has been a change in this status since this frame was last transmitted (new call).	64 Element Array of Byte
441	Gap Time	2.0 – 5.0 Seconds	16 element array of Float
505	Time Before Reduction	0.0 – 200.0 Seconds	16 element array of Float
569	Time to Reduce	0.0 – 200.0 Seconds	16 element array of Float
633	Min Gap Time	0.0 – 200.0 Seconds	16 element array of Float

Frequency of Message: Sent by GUI when user hits START button.

Sent by GUI every 100ms only if GUI detects new input from user.

TEXAS Model Interface

The TEXAS Model for Intersection Traffic Control has been integrated into SCOPE. The TEXAS Model is a

single intersection simulation model developed at the University of TEXAS under the lead of Dr. Thomas Rioux.

Data sent from the TEXAS Model to the Primary

Description: Commands and Data sent from the TEXAS Model to the Primary Ada95 SCOPE program.

Size: 72 bits

Structure:

Table 28: TEXAS Model to SCOPE Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	CID II ID	0 – 256	Byte
8	Phase 1 Detector	0 (False), 1 (True)	1 bit Boolean
9	Phase 2 Detector	0 (False), 1 (True)	1 bit Boolean
10	Phase 3 Detector	0 (False), 1 (True)	1 bit Boolean
11	Phase 4 Detector	0 (False), 1 (True)	1 bit Boolean
12	Phase 5 Detector	0 (False), 1 (True)	1 bit Boolean
13	Phase 6 Detector	0 (False), 1 (True)	1 bit Boolean
14	Phase 7 Detector	0 (False), 1 (True)	1 bit Boolean
15	Phase 8 Detector	0 (False), 1 (True)	1 bit Boolean
16 .. 23	SPARE	0	Byte
24	SPARE	0	1 bit Boolean
25	SPARE	0	1 bit Boolean
26	SPARE	0	1 bit Boolean
27	SPARE	0	1 bit Boolean
28	Pedestrian 2 Detector	0 (False), 1 (True)	1 bit Boolean
29	Pedestrian 4 Detector	0 (False), 1 (True)	1 bit Boolean
30	Pedestrian 6 Detector	0 (False), 1 (True)	1 bit Boolean
31	Pedestrian 8 Detector	0 (False), 1 (True)	1 bit Boolean
32 .. 39	SPARE	0	Byte
40 .. 47	SPARE	0	Byte
48 .. 55	SPARE	0	Byte
56 .. 63	SPARE	0	Byte
64	SPARE	0	1 bit Boolean
65	SPARE	0	1 bit Boolean
66	SPARE	0	1 bit Boolean

Bits	Field Name	Possible Value(s)	Data Type
67	SPARE	0	1 bit Boolean
68	Restart	0 (False), 1 (True)	1 bit Boolean
69	SPARE	0	1 bit Boolean
70	IO Mode 1	0 (False), 1 (True)	1 bit Boolean
71	IO Mode 2	0 (False), 1 (True)	1 bit Boolean

Frequency of Message: Sent based on TEXAS Model input parameters.

Data sent from the Primary to the TEXAS Model

Description: Commands and Data sent from the Primary Ada95 SCOPE program to the TEXAS Model.

Size: 64 bits

Structure:

Table 29: SCOPE To TEXAS Model Structure

Bits	Field Name	Possible Value(s)	Data Type
0 .. 7	CID II ID	0 – 256	Byte
8	Phase Data	See CID II Format Below	3 Bytes
9	Overlap Data	Future Implementation	2 Bytes
9	Pedestrian Data	Future Implementation	2 Bytes

Frequency of Message: Sent every 100 ms.

CHAPTER FIVE: SOFTWARE TEST

Test & Validation Considerations

A variety of errors and deficiencies will be encountered during the specification and design of complex systems, each having a different root cause, and care must be taken to ensure their early detection and correction (close to their source), if breeding effects are to be avoided, and reliably safe systems developed (12). It is well known in the software world that the majority of fault detection and correction occurs during the testing phase, and that this can be most costly and timely phase of any project. It has been estimated that approximately 40%-50% of the total amount of software development resource is testing (28). And the process of testing software for a safety-critical system is much more rigorous than for other types of software; however the same basic principals apply.

Testing starts with the developer, who can aid in the testing of safety critical software by properly commenting the code, avoiding complex design patterns, and architecting the system that anticipates the need for assessment. Coding standards should be established at the beginning of the project and should be used to achieve consistency. Every functions, class variable, logic block and calculation should be explained or justified, as an operational that seems obvious or trivial to the developer might not be so to someone else looking at the code. Documentation that is clear and complete significantly aids the test team or subsequent developer. The simpler a design is the easier it will be for reviewers assess the code, find defects, and have a high level of confidence in their work. The ultimate result is code that is easier to maintain, test, and correct. A modular design is also beneficial to testing, as it lends itself to code that anticipates assessment. When compartmentalized code is written properly it isolates each part of the program which simplifies the creation of unit test. Unit testing has several benefits as it

allows testing to begin earlier in the development cycle, reduces time to find defects, and eases the pain of integration – since the components should be largely defect free.

Tools & Methods

Before the software was integrated with external components or simulators, the application had to work as specified by the requirements. While small bugs were found along the way, the majority of the time spent debugging problems related to communication and results synchronization between the primary and secondary. This was as expected though as the primary and secondary, were not only in the early stages of development but also being developed by different engineers restricted from seeing one another's code. Application logic and APIs were not always implemented per the requirements and sometimes there were disagreements about what they entailed. Without the ability to step-through the other side of the application code a heavy importance was placed on proper debugging statements. User-defined parameters allowed various levels of diagnostic information to be displayed either to a file or a terminal.

As development progressed, the number of user configurable items and results outgrew the configuration text file and terminal window used during the early development stages. Knowing the trend was only going to continue as intersections are added in later phases of development the decision was made to create a graphical user interface (GUI) (Figure 8a, Figure 12, Figure 13, Figure 14). This addition allows users the ability to easily alter default configuration data and interpret synchronization results.

Testing with Traffic Simulator

The SCOPE software makes use of the Traffic EXperimental Analytical Simulation (TEXAS) Model for

Intersection Traffic. The TEXAS model is a microscopic single-intersection vehicular traffic simulation model of an at-grade intersection or diamond interchange (Figure 8b). The TEXAS Model is open-source software licensed under the terms of the GNU General Public License and made available on Windows X86 and Linux X86 systems. The user interface is written in Java and the model is written in FORTRAN, therefore an interface classes was written to allow the primary software to communicate with the TEXAS Model FORTRAN code. Using the TEXAS Model allowed SCOPE to be tested in a single intersection environment.

(a) (b)

Figure 8. (a) Original Stand Alone GUI designed for SCOPE. (b)TEXAS Model Interface Panel.

The TEXAS Model allows simulation of three to six leg intersections, with a maximum of six inbound and six outbound lanes per leg. Initial software testing consisted of a simulation of a four leg intersection with three inbound and outbound lanes per leg. The initial vehicle classes included cars, buses, and trucks with driver classes ranging from slow to aggressive. The TEXAS Model simulates an intersection that allows right turn on red with 3 detectors per lane. As part of the SCOPE a parser was developed that

modified the TEXAS Model's FORTRAN code so that it could compile using GCC (gfortran). The TEXAS Model contains a Hardware-In-The-Loop (HITL) feature that connects it to a controller interface device (CID-II) (Figure 9). The HITL source code was modified to connect and communicate to SCOPE using standard sockets. The information exchanged between SCOPE and the TEXAS Model includes phase, overlap, pedestrian, and detector data.

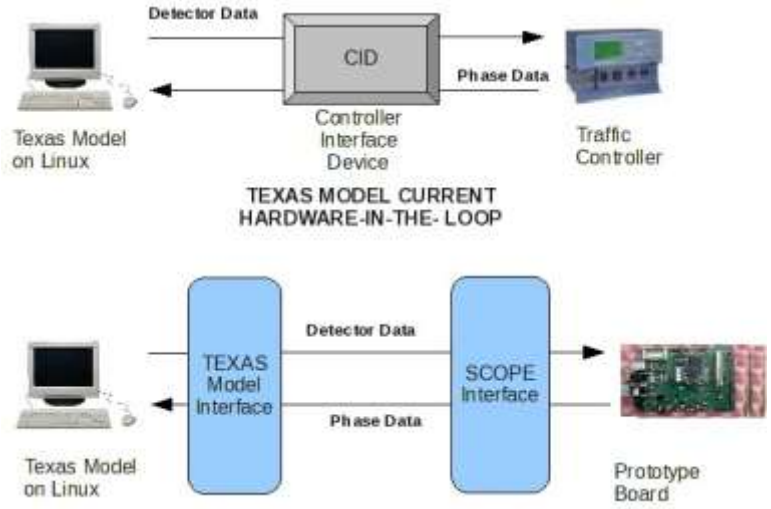
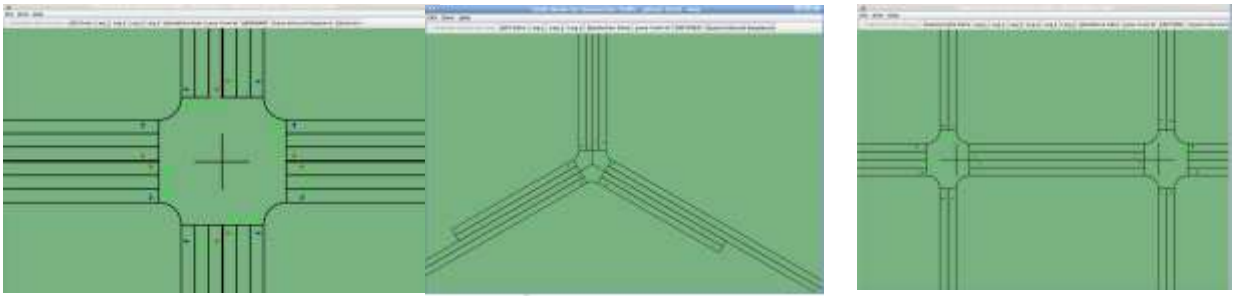


Figure 9. TEXAS Model Modified to use SCOPE.

The TEXAS Model has been used to validate SCOPE's control of a standard 4 legged intersection, a Y shaped 3 legged intersection and a “TEXAS Diamond” intersection (Figure 10). SCOPE was tested with varying numbers of lanes per leg along with varying traffic patterns. These intersection models are shown below.



(a) (b) (c)

Figure 10. (a) 4 Legged. (b) Y** shaped 3 Legged**. (c) TEXAS Diamond.

A video of the TEXAS Model integration demo has been uploaded to:

http://www.youtube.com/watch?v=xeMDpeu9fhQ.

Integrating with Peek ATC-1000

Before the development phase for the actuated control began the software was integrated onto the Peek ATC-1000 (Figure 11a). In doing so the SCOPE logic was allowed to be routinely tested and validated on a field production ATC. The Peek ATC-1000 was connected to an ATSI TVC 3500 Virtual Cabinet through the use of the SDLC port to provide full emulation of a NEMA TS 2 standard cabinet (Figure 11b). The software correctly set the signals on emulated traffic controllers and avoided any conflicting traffic patterns for all pre-timed and actuated logic programmed. Negative cases such as miscompares between the primary and secondary were tested to ensure the blinking “all-red” signal would be given if conflicting results were given. The software logs all miscompares and errors on the ATC to give developers full insight into the cause of the problem such that corrective changes can easily be made to the logic.

Figure 11. (a) SCOPE Running on Peek ATC communicating with Android Phone App. (b) VC 3500 Virtual Cabinet Interface.

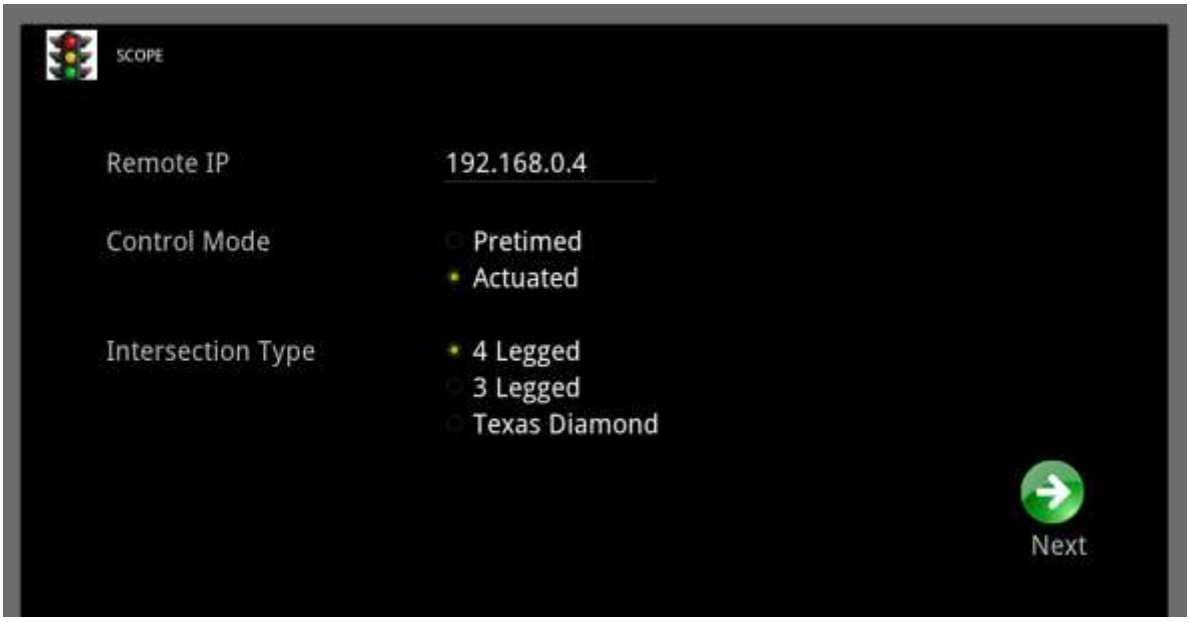


Figure 12: Android Tablet App Startup Screen

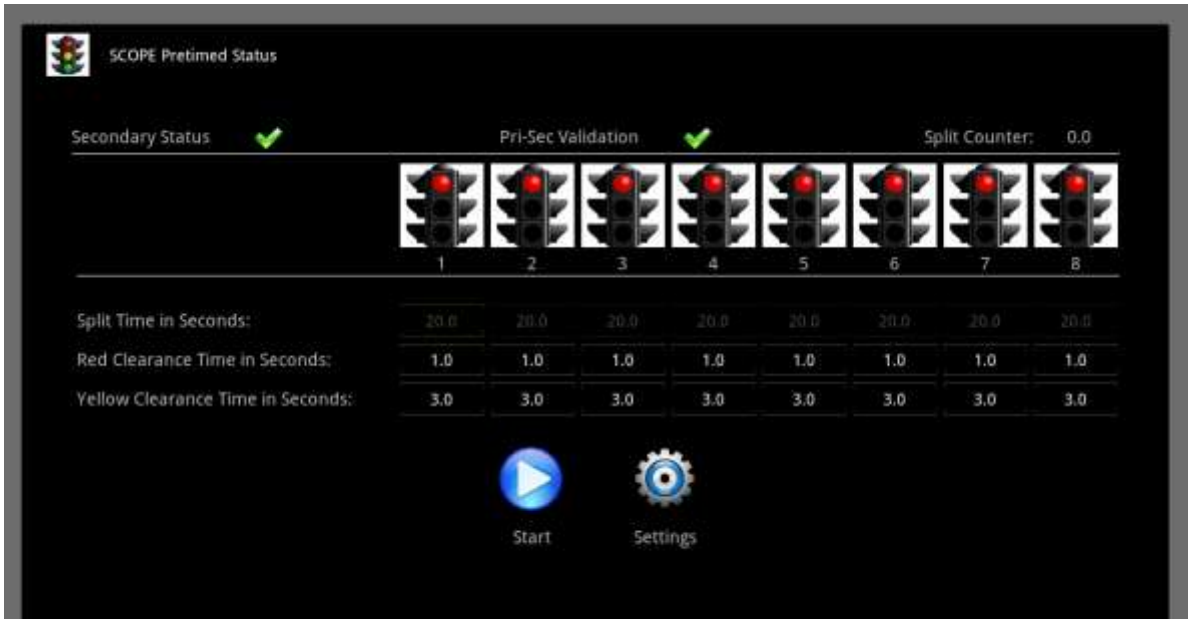


Figure 13: Android Tablet App Pre-timed Status Screen

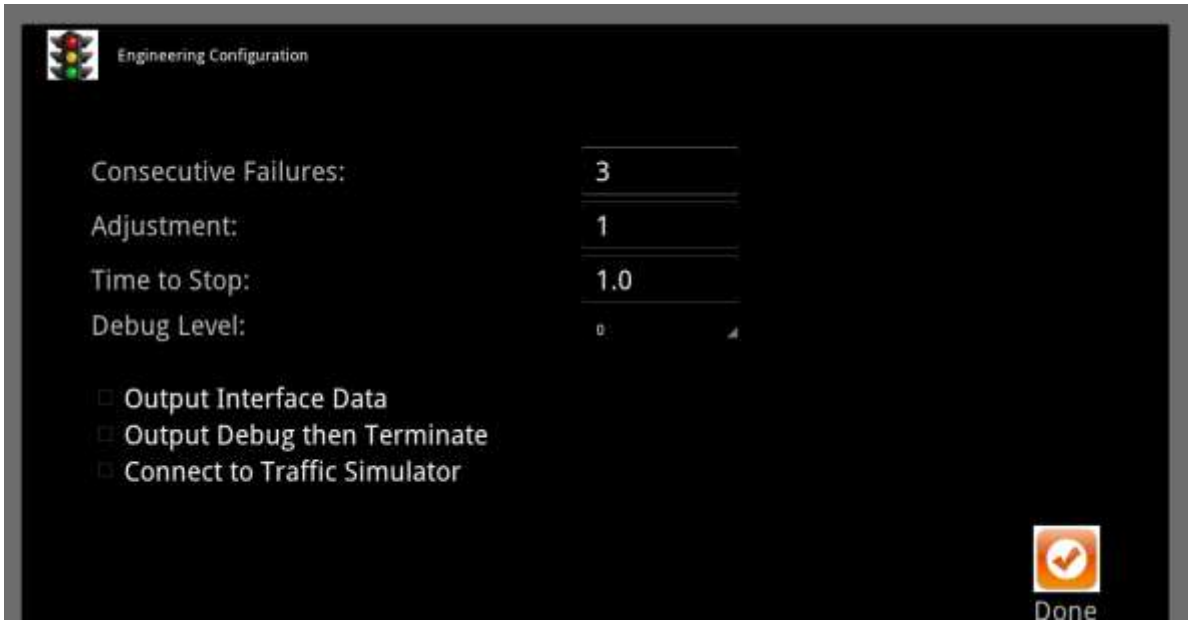


Figure 14: Android Tablet App Settings Screen

A video of the SCOPE software running on the Peek ATC that demonstrates miscompare logging has been

uploaded to: http://www.youtube.com/watch?v=GW2NARzrG3I.

Formal Qualification Testing

Field validation will serve as SCOPE's Formal Qualification Testing (FQT) and will be conducted at a laboratory intersection. It is scheduled to begin immediately following the NCHRP 3-66 integration phase and is a four month effort. The location will be chosen by our D.O.T. Contract Officer Technical Representative (COTR) and representatives of the Federal Highway Administration. Test plans and procedures are currently being developed to validate SCOPE meets the levied by the DOT. These plans and procedures will contain a step-by-step instructions and a checklist to ensure SCOPE is working as designed. Each of the considered testing locations makes use of actual traffic controllers, vehicles and actuators for verification and data collection.

Current locations under consideration include The Virginia Tech Transportation Institute (VTTI) “Smart Road”, the Turner-Fairbank Highway Research Center (TFHRC), and the California Partners for Advanced Transit and Highways (PATH) facility located at the Richmond Field Station. The TFHRC in McLean, Virginia is the home of the Federal Highway Administration's Office of Research, Development, and Technology. The VTTI “Smart Road” is a closed research facility owned and maintained by the Virginia Department of Transportation. Part of this smart road will be an intersection containing two high speed approaches and two low speed approaches. The intersection will contain customized controllers, vehicle sensors, and wireless communications. The smart road itself is 2.2 miles long and was made possible by a cooperative effort between the Virginia D.O.T, the FHWA, the Virginia Transportation Research Council, the University of Virginia, Virginia Tech, and the VTTI. According to the VTTI website, “Current plans are to add an at-grade intersection to the Smart Road so that its features will allow intersection-related

research. This intersection will not connect to any public road at any time in the future, it will be for testing purposes only." The TFHRC contains an intelligent test intersection used for intersection collision avoidance systems being developed by the infrastructure consortium (sponsored by Cooperative Intersection Collision Avoidance Systems Initiative). The NCHRP 3-66 section of our development must be compliant with the systems developed under the CICAS initiative. Performing formal qualification at this intersection at TFHRC would allow ATI to work alongside CICAS testers.

Figure 15: TFHRC Test Site

PATH at University of California, Berkley has a test intersection located at the Richmond Field Station. It is a full size intersection containing an ITS-340 controller cabinet, PC-104 infrastructure computers, and in pavement loop detectors on 3 legs of the intersection. In addition, it has a video system, Canoga micro-loops, a radar, and mini-loops with wireless links.

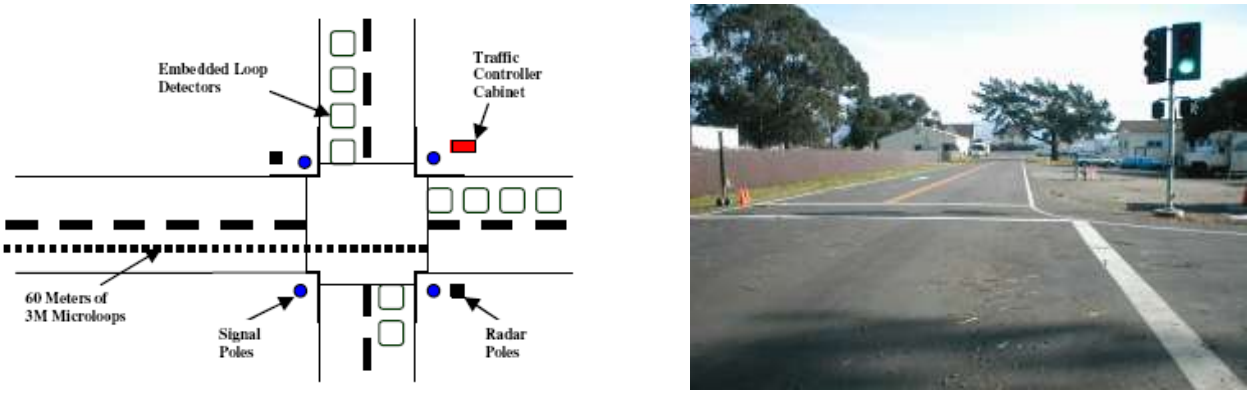


***Figure 16**: PATH Test Site*

.

CHAPTER SIX: RESEARCH METHODOLOGY

Although SCOPE has been tested on the using a microscopic simulation, run on the ATC and integrated into a signalized cabinet – the need to demonstrate it safely and efficiently controlling an intersection using real-world data existed. Formal Qualification Testing of SCOPE is still scheduled to occur, however it has been postponed in order to incorporate new features such as NTCIP standards. The test procedures, equipment and controlled environment afforded by a test site will not be available to SCOPE until the next calendar year. In order to validate the effectiveness of SCOPE an experiment was designed and performed.

Methodologies for this research are described in the following subsections, listed and summarized below:

1. Experiment definition: describes the process of defining the problem, a proposed solution, and hypotheses to be tested.
2. Experiment design: describes the steps taken to choose an intersection for the experiment to validate the effectiveness of the models.
3. Data Collection: describes the steps taken to choose an intersection for the experiment, a description of the intersection, how the data was recorded and how the configuration of the detectors compares to other intersections.
4. Data extraction: describes the methodologies by which field data was taken from videos and field notes. Data is representative of the raw, unorganized information used in the experiment. It provides all that is needed for the live intersection but still needs to be rearranged for use in feeding the simulation.

5. Data processing: describes the efforts taken to prepare the raw data to be used by the SCOPE simulation driver.
6. Data analysis: describes determining the peak hour for the experiment, and characterizing the intersections traffic flow.
7. Simulation: describes the addition of adding the simulation and data recording capability to the SCOPE GUI.

Experiment Definition

Throughout the paper the need and justification for dual-redundant open-source traffic controller software has been presented. This paper has also stated that SCOPE will meet all the requirements, bringing with it the benefits that such software entail. The final step in the process was to test SCOPE using data from a live intersection and prove its ability to improve the effectiveness of a given intersection.

This research involves the testing of various actuated models and understanding of traffic. To test the effectiveness of the various modes, several experiments had to take place. The goal was to determine a set of experiments that would rigorously test the Advance Traffic Controller and the chosen hypothesis using real-world samples. The experiment is separated into three sections: hypotheses tested, evaluating existing intersection control effectiveness and comparing the effectiveness of various modes of SCOPE.

Hypotheses

Quality data is the key to monitoring and validating traffic controller operations. The typical data source for this information is usually extracted using vehicle counting, vehicle detector data, or recording of various vehicle action times. By collecting real-world data from an intersection and providing that data

into SCOPE allowed the following hypothesis to be tested:

- SCOPE provides the capability to safely yield right-of-way.
- SCOPE is capable of working in real-world intersections and is capable of passing Formal Qualification Testing (FQT).
- SCOPE provides a system that incorporates traffic models which can be configured to improve the effectiveness of an existing intersection.
- SCOPE provides a testbed for evaluation of traffic models, and the open-source nature of the project allows new models to be easily integrated.

Existing System / Live Intersection

Collection of data from a live intersection provided information which was beneficial for the evaluation of the current system and the testing and comparison to SCOPE. The length of time vehicles spent in-queue was used to characterize the current intersection, while the times vehicles and pedestrians triggered detectors run the SCOPE software.

Comparison to SCOPE

By collecting the times at a live intersection at which a pedestrian or unimpeded vehicle triggered a detector, this data could be used to run SCOPE and evaluate the effectiveness of the system using various models. To run SCOPE using the data collected a software simulation component was written to read the call times and types and place calls on the system at the given time. The simulation would also be responsible for collecting the average vehicle time in-queue per lane. The resulting average times of vehicles in-queue was then compared to the existing live intersection.

Experiment Design

The experiment was designed such that SCOPE would run using data from a live intersection, such that the results of various models could be validated against the existing intersection. From the recording of the live intersection the time in-queue per lane was recorded for all cars and summed. For the simulation, SCOPE was run with the actuated data sent to primary software at the same intervals as the detectors were triggered in the video (see Assumption #3 below). Based on the color of the light and assumption #2 listed below it was decided what action the vehicle would take with respect to stopping. If the vehicle stopped the amount of time the vehicle spent in-queue until the light turned was summed up for each lane. The differences in times between the live intersection and a various models of SCOPE were compared to evaluate efficiency.

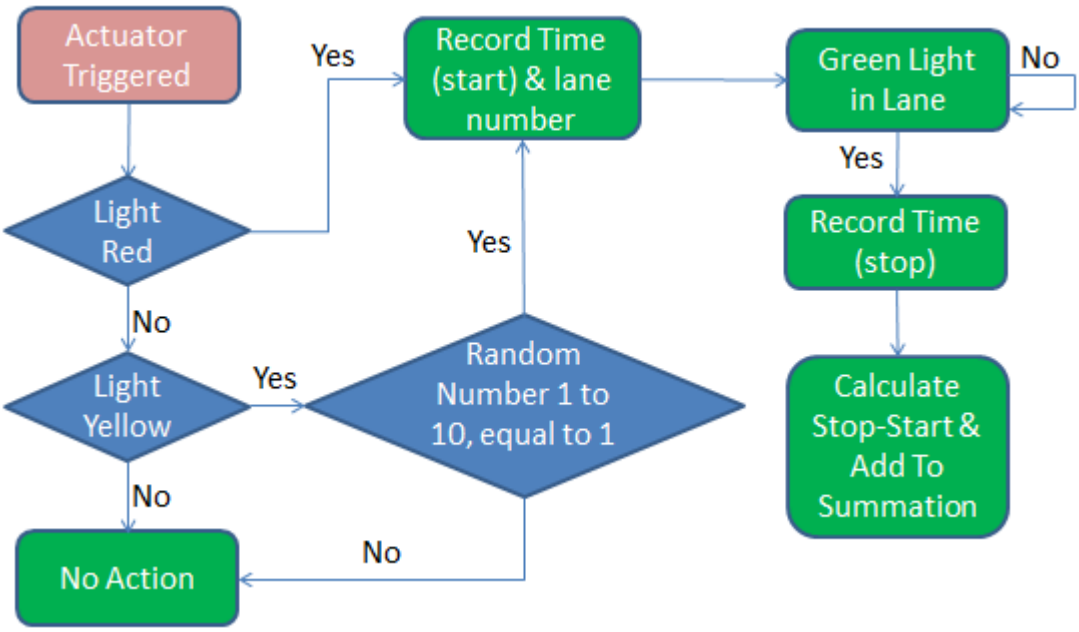


Figure 17: Simulation Flowchart for In-Queue Time Per Lane

Assumptions/ Estimation

1. An average start-up loss time for the intersection can be used to determine when the light turned green. Start-up lost time occurs when a traffic signal changes state from red to green. Some amount of time elapses between the signal changing from red to green and the first queued vehicle moves across the stop bar. This assumption is necessary because the video does not have a view of the traffic signals to determine when the green signal occurs.

   g = m - l

   where:

   g is the time the light turned green

m is the time the first car in the queue began moving

l is average start-up loss time

2. For the simulation to reflect real-world conditions it was necessary to estimate if a vehicle approaching a yellow light would go or stop. This decision time is known as the dilemma or indecision zone. The driver of the vehicle may have difficulty deciding whether to stop or proceed through the light. Abruptly stopping could result in a rear-end collision, while proceeding may produce a t-bone collision. Table 30 shows the boundaries of the dilemma zone, and Figure 18 illustrates dilemma zone boundaries for a vehicle approaching an intersection at 35 mi/h (FHWA - Traffic Control Systems Handbook: Chapter 6. Detectors). The data indicates that 90 percent of motorists will decide to stop, lies 4.5 seconds from intersection. And 10 percent of the motorists will decide to stop, is 2 seconds from the intersection. With the major arterial detectors being approximately 4 seconds from the stop bar for the purpose of the simulation it was assumed 90 percent would stop. For the minor arterial, where the detectors were located directly in front of the stop bar for the purpose of the simulation it was assumed 100% would proceed.

Table 30: Dilemma Zone – Probability of Stopping

Approach Speed		Distances from intersection for 90% and 10% probabilities of stopping			
mi/h	km/h	90% values in ft	10% values in ft	90% values in m	10% values in m
35	56	254	102	77	31
40	64	284	122	87	37
45	72	327	152	100	46

mi/h	km/h	90% values in ft	10% values in ft	90% values in m	10% values in m
50	80	353	172	108	52
55	88	386	234	118	71

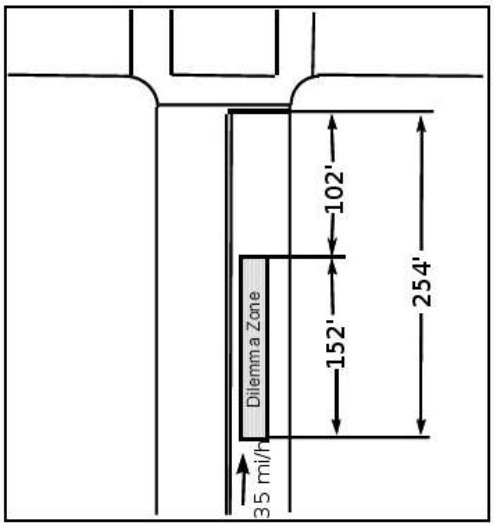


Figure 18: Dilemma Zone for 35 mi/h approach (representative of Gemini Blvd)

3. To estimate the time in-queue for vehicles had SCOPE been controlling the intersection, it was necessary to record the time at which the vehicle would have triggered the sensor in free-flow traffic environment. Hence if the light was not red or if loss time was not a factor – at what time would the vehicle have caused the sensor to be triggered.

Data Collection

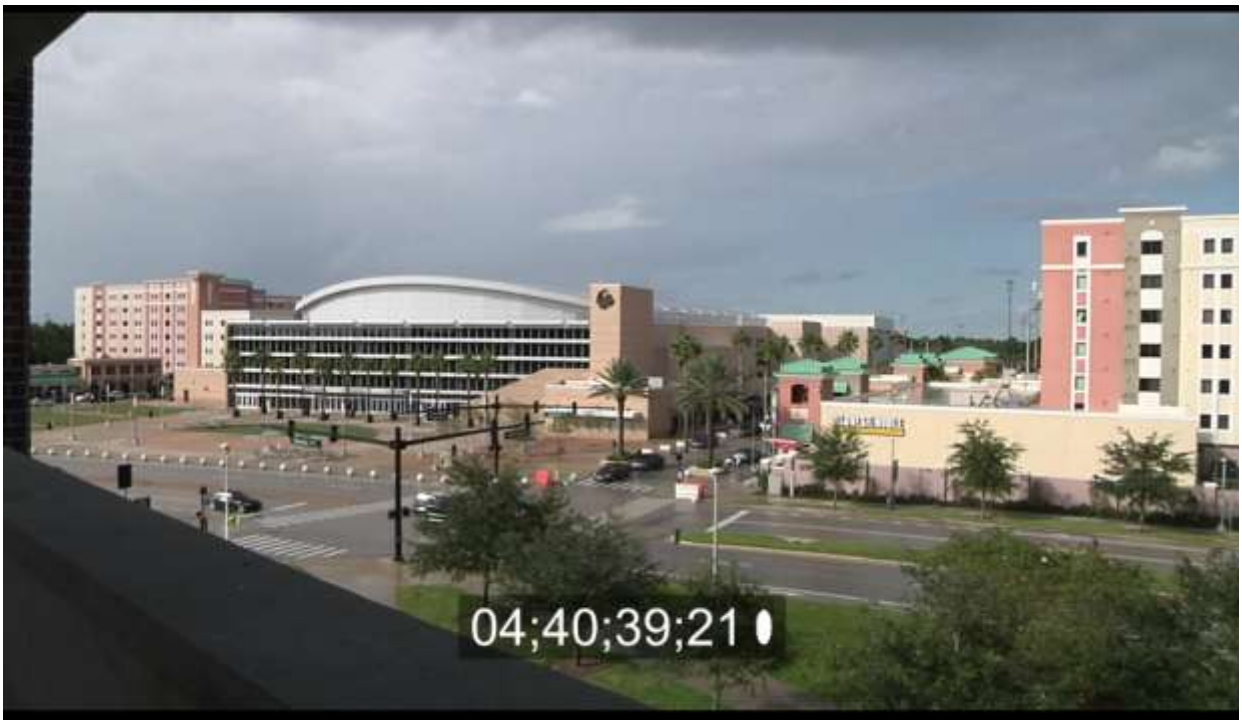


Figure 19: Frame of Video Capture from Intersection

The intersection of Gemini Blvd and Plaza Drive on the University of Central Florida campus was chosen for the data collection portion of the experiment. The nearby location of a four-story parking garage was the first key reason for selecting this intersection. The approximate fifty foot height of the top story of the garage and lack of visual obstructions provided a clear field of view of all four approaches (Figure 19). The second key reason was the intersection only used inductive-loop detectors and pedestrian buttons for its actuated modes. Systems using combinations of infrared (IR) or video capture, or a combination with inductive-loops, would have provided additional complexities in knowing when the call was placed on the system.

Gemini Boulevard is the major arterial and is a four lane roadway, while Plaza drive is the minor arterial

and is a two lane roadway. The only right-of-way given to any of the approaching lanes is a standard vertical red, yellow and green signal face arrangement. No left or right turn indications are given to any approach (Figure 20). The decision to make a left or right turns must be made by drivers is based on other vehicle actions and gaps in oncoming traffic.

***Figure 20**: (left) Map of Intersection. (right)Signal at Intersection.*

Data collection occurred between the times of 4:30 p.m. to 6:30 p.m. on a Monday evening. The timing of the collection was chosen as that is when the volume for that road is at or near its highest. These times are representative of when the peak hour was most likely to occur on the intersection on any given Monday. The two hour video was examined, and the peak hour of 4:30 p.m. to 5:30 p.m. was chosen for the experiment. The length of one hour was selected as count periods for traffic studies generally occur in five, fifteen or sixty minute intervals (29).

During the times of the experiment the roads leading up to the intersection are highly traveled by students, especially those attending classes after work, and people leaving work and using the roads as a

shortcut around a nearby major state road. Given the large traffic volume, a video of the intersection was recorded during the specified times. This allowed for the data collection to occur using two observers, instead of six or more that would have been required to record data on the field. In this experiment one observer was atop the parking garage filming the intersection and another was near the intersection recording times of pedestrian calls.

Detectors

The size, type and placement guideline for a detector to actuated approach varies by state and agency standards. In the Idaho Transportation Department Traffic Design Manual (2008) a basic detector configuration has a 6 foot stop bar detector at the stop bar and a second one 10 feet upstream from it.

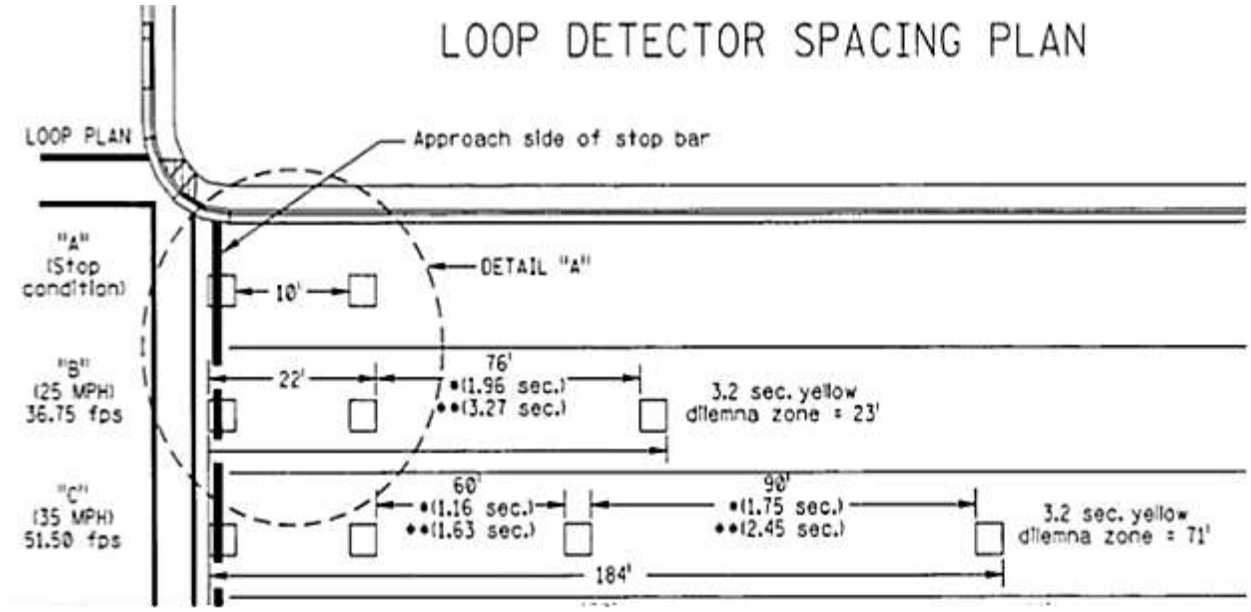


Figure 21: Idaho Transportation Department Suggested Detector Placement

In the configuration shown in Figure 21, for the 35 mi/h approach of the major arterial used for this study would require four detectors.

The FHWA Traffic Detector Handbook defines conventional control as using a single small area detector. The minimum green interval is set to provide time to clear a standing queue between the sensor and the stop bar. The unit extension sets a value for both the allowable gap to hold the green and the travel time from sensor to stop bar. For a conventional control the allowable gap is usually 3 or 4 seconds, so the sensor ideally located 3 or 4 seconds upstream from the intersection. This sensor position would appear to be the most efficient for accurately timing the end of green after passage of the last vehicle of a queue. However, a long minimum green is created at approaches with speeds greater than 25 to 30 mi/h because of the longer sensor setback. Therefore, the principle is amended to locate sensors 3 to 4 seconds of travel time from, but not more than 170 ft from the stop bar. Table 4-2 displays the application of this principle to determining sensor location and associated timing parameters as a function of vehicle approach speed.

Table 31: FHWA Traffic Detector Handbook Suggested Detector Placement

Approach speed	**Detector setback from stop bar to inductive-loop detector**	**Minimum green time**	**Passage time**		
mi/h	km/h	Feet	meters	seconds	Seconds
15	24	40	12	9	3.0
20	32	60	18	11	3.0
25	40	80	24	12	3.0
30	48	100	30	13	3.5
35	56	135	41	14	3.5
40	64	170	52	16	3.5

At the intersection in this study the major and minor arterials use this conventional control with one detector. The advantage of this single sensor approach is that the cost of installation is minimized.

However, this type of control does not screen out false calls for green as occurs with right turns on red. These false calls from vehicles making a right turn from the minor arterial led to yielding the right of way to lanes with no cars. According to the table provided by the FHWA the intersection in this study's major arterial (35 mi/h and detector setback of 110 feet) would ideally be at 135 feet.

Data Extraction

The data extracted from the video and in the field to test the above hypotheses are discussed in this section. The primary data collected was actuated call times, pedestrian call times, number of vehicles in-queue and average time in-queue. It was important to use real-world data as these inputs tested the system to next level past the TEXAS model microscopic simulation.

Data used to validate the SCOPE system are defined as follows:

- Actuated call times: The time at which a vehicle runs over an inductive loop detector and the actuated call is placed on the system.
- Pedestrian call times: The time at which a vehicle runs over an inductive loop detector and the actuated call is placed on the system.
- Vehicles in-queue: The number of vehicles in-queue. Where in-queue is the count of all vehicles stopped at the intersections or with a speed lower than 5 mi/h. This also refers to any vehicles that arrive and stops during a red signal plus the vehicles arriving and stopping during the green indication.
- Average time in-queue: The average time that all vehicles spend waiting in the queue for a given lane.

As mentioned before, the pedestrian call times were extracted in the field by an observer during the study and the actuated call times were collected by reviewing the video. The time on the watch of the pedestrian observer was noted once the record button was pressed, such that a timer could later be overlaid on the video for synchronization purposes. For free flow traffic, those vehicles that were identified as vehicles driving through the approach under unimpeded traffic conditions, the time in the video the vehicle crossed the detector a given detector was recorded for the actuated call time. For non-free flow traffic, those vehicles that were identified as vehicles driving through the approach under impeded traffic conditions the estimated time at which the vehicle would have tripped the detector had it not been impeded was recorded for the actuated call time. Non-free flow traffic included cars stopping due to a red or green light (stopped delay) or slowing down due to other vehicles impeding their free flow (deceleration delay).

Data Preparation

Once all the data was collected and extracted the remaining step of data work was to prepare the data in the format the simulator would be able to parse. Excel was used to store the data at the times the actuated or pedestrian call occurred, which actuator was triggered and the type of call in columns A, B, and C in Table 32 respectively. From there Excel was used to calculate the elapsed time from the previous call of any call type and finally a space delimited string was created and placed in column C and D in Table 32 respectively.

Table 32: Example Data File

Simulation Time	Actuator	Call Type	Time Elapsed (ms) from previous call	Data for Simulator
4:40:00	6	new_call	0	0 6 new_call
4:40:02	2	new_call	2000	2000 2 new_call
4:40:07	8	ped_call	5000	5000 8 ped_call

Upon completing the data entry in the excel spread sheet, the final space delimited string was placed in a raw file for use by the simulator. The resulting raw file was in the format:

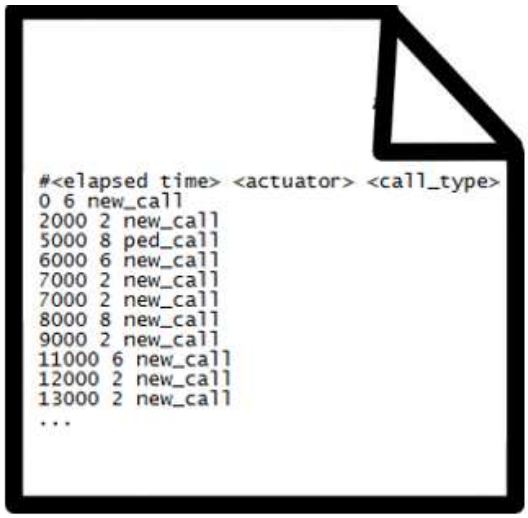


Figure 22: Example Simulator Format Data File

The entire datasheet can be found in APPENDIX C: LIVE DATA CAPTURE.

Data Analysis

The Flow rates are collected through point measurements, and require measurement over time. It is simply the number of vehicles counted crossing a given point divided by the elapsed time and computed using the following formula:

q = N/T

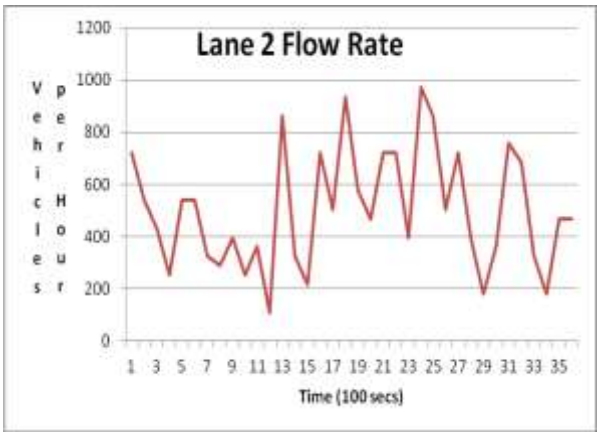
where:

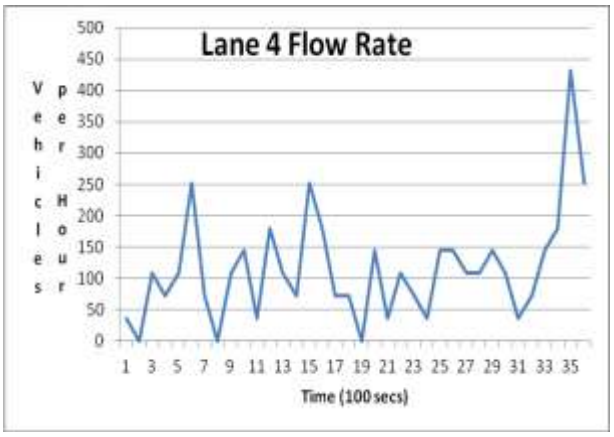
q is the flow rate

N is the number of vehicles counted

T is the elapsed time

Flow rates are typical expressed in terms of vehicles per hour, although the actual measurement interval can be less. The 1985 Highway Capacity Manual (HCM 2000) suggests at least 15-minute intervals, although shorter interval can still be beneficial. Figure 23 shows the flow rates varying with time for each lane in the intersection during the time of the experiment.





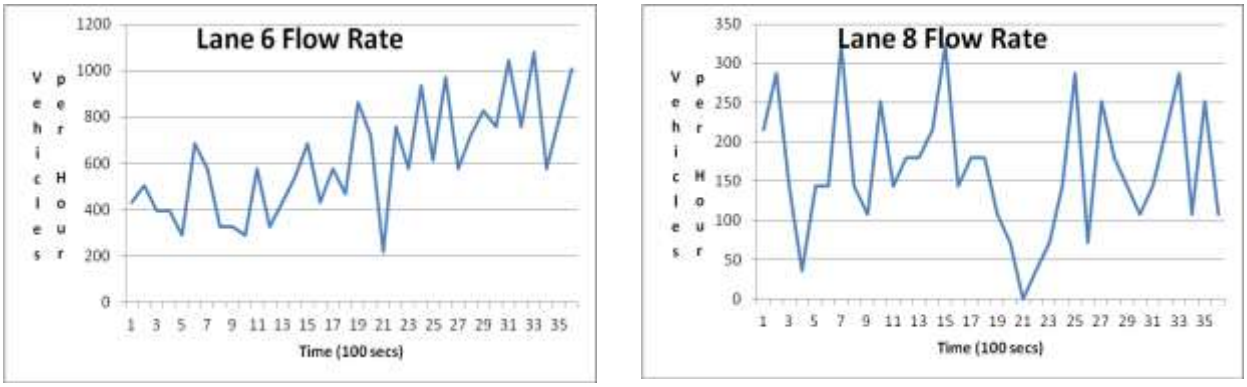


Figure 23: Flow Rates per Lane During Peak Hour

Simulation

The original GUI used for SCOPE was the testbed for the experiments as the GUI had visibility to all status and configuration data for the given four way intersection. The GUI was first expanded to allow the input of pedestrian calls in a similar fashion to the existing actuated calls (Figure 24). While not necessarily need for the experiment, it more easily allowed for the testing and validation of the interface between the GUI and the primary software, and the primary and secondary software which were critical to the experiment. In the same manner the actuated panel on the GUI created a thread to monitor checkbox presses, the simulator created a thread to place the calls at the time specified. A user clicking a checkbox on the GUI to place an actuated or pedestrian call would be equivalent to the simulations raw data file placing the call.

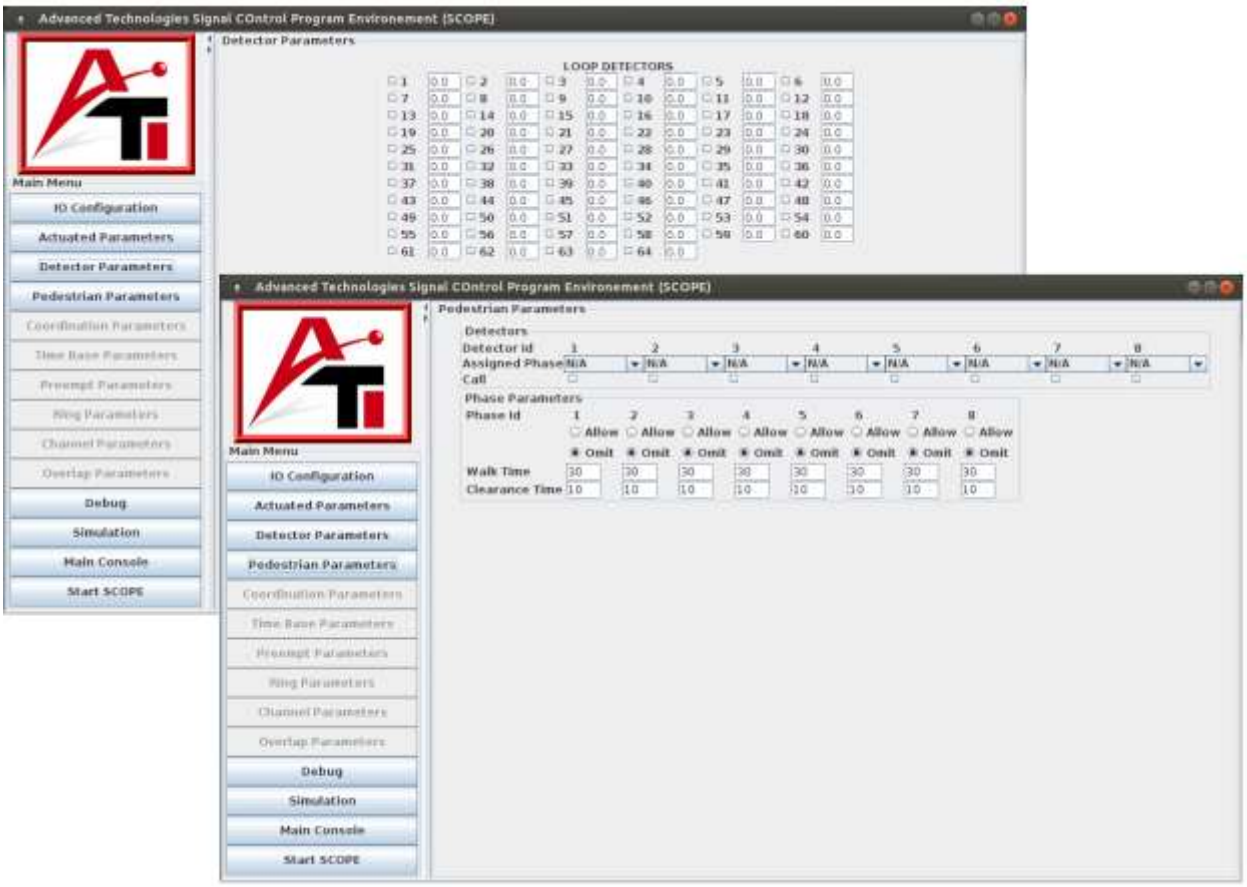


Figure 24: (Top) Actuated Call Interface (Bottom) Pedestrian Call Interface

An addition console was added to the existing GUI which allowed a user to select to run the simulation from a file and allowed visualization of actuated and pedestrian calls. A small dot was placed on the lane of the vehicle or pedestrian placing the call which allowed for the verification of the video and simulation being in-sync. The main console allowed for easy monitoring of conflicting calls and switching between the main and simulation console allowed for confirmation of successful operation.

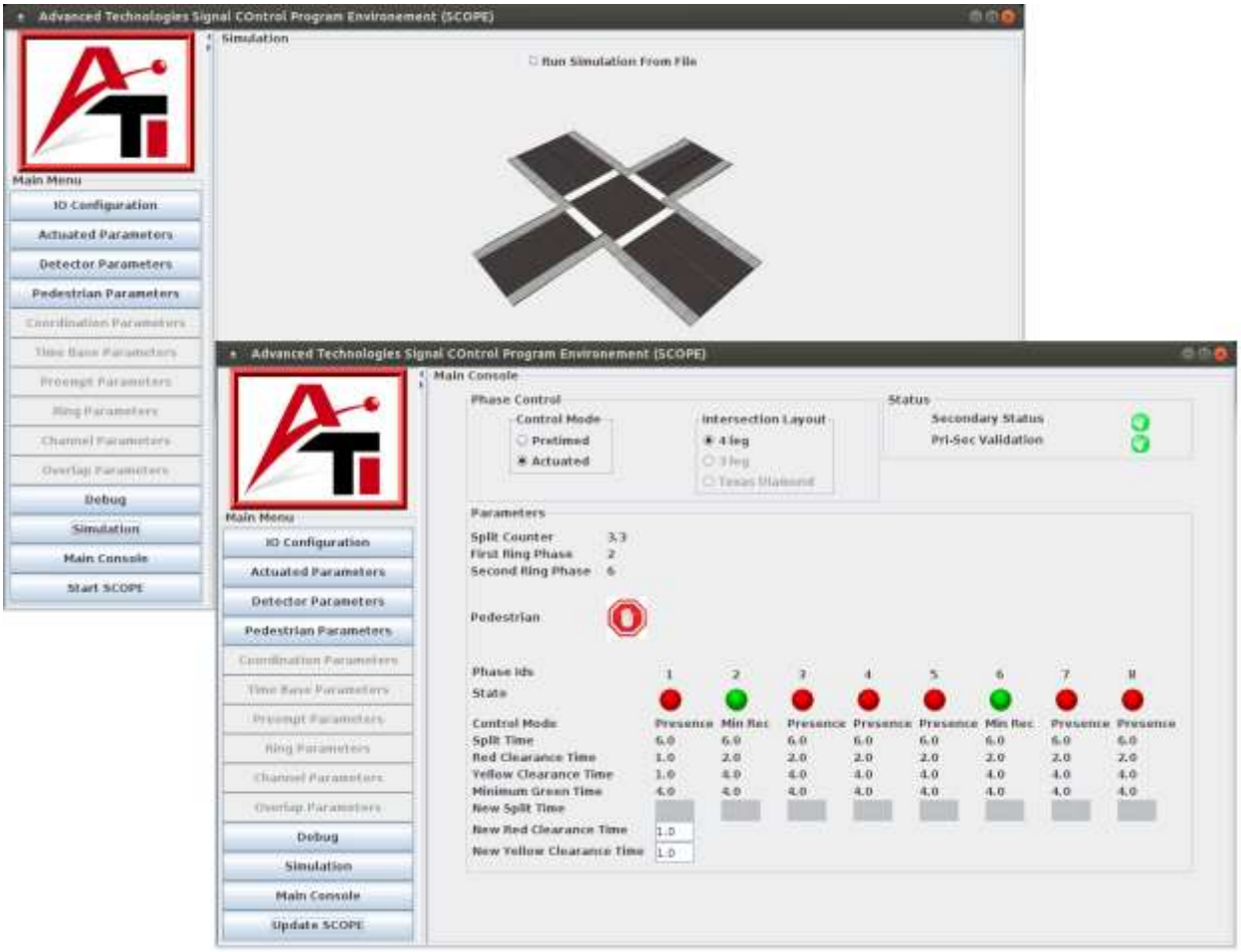


Figure 25: (Top) Simulation Visualization (Bottom) Main Console

CHAPTER SEVEN: ANALYSIS AND RESULTS

Several models were tested in order to the ability of SCOPE to yield the right-of-way without placing conflicting calls on the system and to compare their efficiency to the system currently in use at the intersection. The first section of this chapter is dedicated to the analysis of the time spent in-queue for the vehicles in the video operating under the current system. The second section of this chapter will focus on the various traffic models under various configuration parameters compare to the existing system.

Gemini Boulevard & Plaza Drive

The traffic controller at the corner of Gemini Boulevard and Plaza Drive is adaptive traffic control system installed and run by Siemens Traffic Solutions called SCOOT, which stands for Split Cycle Offset Optimization Technique. "Basically [SCOOT] looks at how many cars are coming through an intersection each day at a certain time of day, and then it goes, 'Well, yesterday I had this many cars, today I've got this many cars, so I need to change my times to allow that many cars to get through,' " said Sgt. John Moore (30). The intersection handled the traffic demands of the recorded peak hour extremely well, allowing for the safe and efficient movement of vehicles and pedestrian.

During this peak hour 1396 vehicles were witnessed crossing through the intersection. Each of the 1396 vehicles had the time which they ran over the inductive loop detector logged (see APPENDIX C: LIVE DATA CAPTURE). Also recorded and extracted from the video was the amount of time that each vehicle that was required to stop spent in-queue (see

APPENDIX D: LIVE DATA CAPTURE). As shown in Table 33 approximately 40 percent of the vehicles in the major arterial were forced to stop due to a red light, with the minor arterial requiring 55-62 percent of the vehicles to stop (Table 33).

Table 33: Percentage of Vehicles Stopped Per Lane

Lane	Vehicles Per Lane	Vehicles Stopped by Red Light	Percentage Stopped
8	126	78	61.90%
4	104	57	54.81%
2	504	201	39.88%
6	607	249	41.02%

Of the vehicles required to stop, those in the major arterials were required to wait on average 16.5-18.9 seconds, while the cars in the minor arterials were required to wait on average 20.1-25.6 seconds (Table 34).

Table 34: Average Time In-Queue for Vehicles Stopped Per Lane

Lane	Average Time In-Queue (Stopped Vehicles)
8	25.6026
4	20.0702
2	16.4627
6	18.8956

Although the SCOOT system handled the intersection extremely well, upon reviewing the video of the live intersection a couple areas of general improvement became apparent.

- The major arterial's value for minimum green time was too short and preemption by the minor

arterial was occurring too often.

- The quick yielding of right-of-way to the few vehicles in the minor arterial came at a cost to the majority of drivers. It was causing more overall time spent by vehicles in the intersection waiting at red lights. It also led to the queue on the major arterial not always being emptied on the cycle of the light – leading to overall congestion.
- Increasing the minimum green time of the major arterial by 10 or 15 seconds, would still allow for an adequate service of the minor arterial queue.
- Right turns from the minor arterial lanes, although rare, did cause false green lights for the minor arterials.

Simulation

Table 35 shows the number of vehicles in-queue, the average time those vehicles spent in queue and the total amount of time spent by all vehicles in-queue for the intersection of Gemini Boulevard and Plaza Drive. This data will serve as the benchmark for the performance comparison of SCOPE 's various modes of operation.

Table 35: Average Time Per Lane & Total Time of Vehicles In-Queue at Gemini/Plaza

Lane	Count	Average
LANE 8	78	25.6026
LANE 4	57	20.0702
LANE 2	201	16.4627
LANE 6	249	18.8956
Total Time (secs)		11155

Minimum Recall

Minimum recall mode has the controller place an actuated call for service on the phase. The split time of the phase is at least the minimum green, regardless of whether there is demand on the movement. The actuated call is cleared when the light turns green for the phase and the call is placed at the start of the yellow interval. This mode may be used when vehicle detection is not working properly.

Minimum recall is the most frequently implemented recall mode. It is frequently used for the major arterial phases (commonly designated as phases 2 and 6 – the lights on Gemini Boulevard in this experiment) at semi-actuated non-coordinated intersections. This use ensures that the controller will always return to the major-road through phases regardless of demand, thus providing a green indication as early as possible in the cycle.

Run #1

Run #1 was configured with a minimum green time of 40 seconds on the major arterials, true max set to 70 seconds and an extension time of 5 seconds. The minor arterials had a minimum green time of 3 seconds, true max set to 20 and an extension time of 5 seconds. Using these parameters there was a slight decrease in the number of minor arterial vehicles in-queue, however a slight increase in time which they were required to stop. The effect on the major arterial was a poor showing compared to the benchmark with an increase in vehicles in-queue and larger time in queue. Overall there was a 28 percent increase in the total amount of time spent by vehicles in queue.

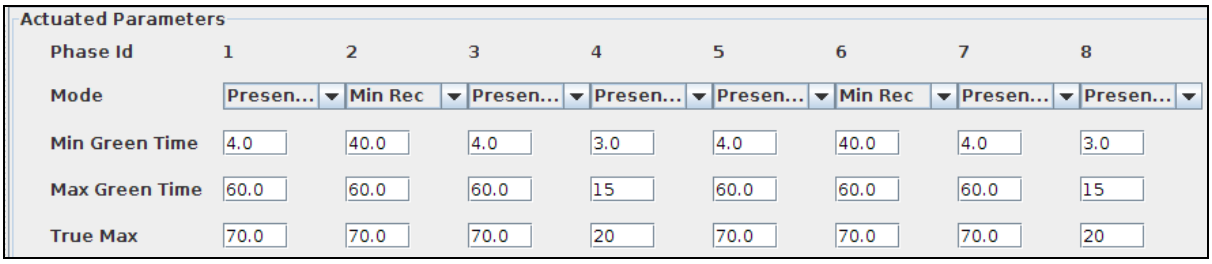


Figure 26: Actuated Parameters for Run #4

Run #2

Run #2 was configured with a minimum green time of 40 seconds on the major arterials, true max set to 80 seconds and an extension time of 5 seconds. The minor arterials had a minimum green time of 3 seconds, true max set to 20 and an extension time of 5 seconds.

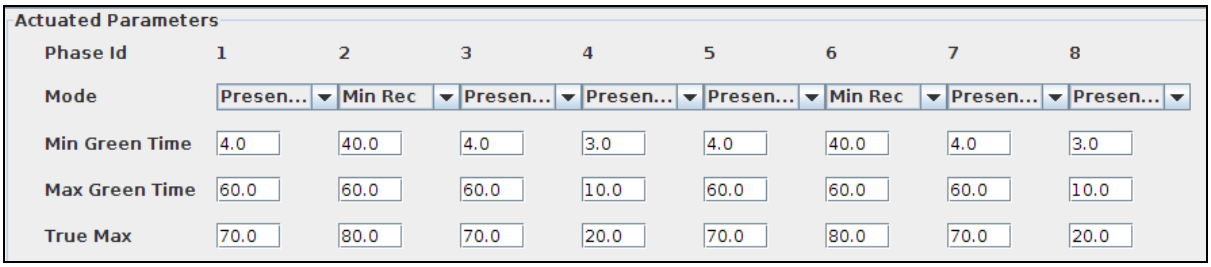


Figure 27: Actuated Parameters for Run #4

Table 36: Average Time Per Lane & Total Time of Vehicles In-Queue Using Simulation with Minimum Recall Mode (Left) Run #1 (Right) Run #2

Lane	Count	Average
LANE 8	70	20.67
LANE 4	55	26.86
LANE 2	238	22.86
LANE 6	295	20.29
Total Time (secs)		14350.4

Lane	Count	Average
LANE 8	64	21.37
LANE 4	54	24.73
LANE 2	250	23.78
LANE 6	325	22.83
Total Time (secs)		16067.9

Using these parameters for the minimum recall /presence mode every metric fared worse than the

benchmark, with a 44 percent increase in the total amount of time spent by vehicles in queue.

Maximum Recall

The maximum recall mode has the controller place a continuous call for service on the phase. This mode causes the presentation of the green indication for the maximum duration on every cycle. Regardless of the presence of a conflicting call, when the maximum recall parameter is selected for a phase the maximum green timer begins timing at the beginning of the phase's green interval.

There are at least three common applications of maximum recall (4):

- Fixed-time operation is desired: Each phase is set for maximum recall. The maximum green setting used for this application should be equal to the green interval durations associated with an optimal fixed time plan.
- Vehicle detection is not present or is out of service: Maximum recall for a phase without detection ensures that the phase serves the associated movement. However, maximum recall can result in inefficient operation during light volume conditions (e.g., during night times and weekends) and should be used only when necessary. In some of these situations, a lower maximum green or MAX 2 (50 to 75% of the typical MAX GREEN value) may be desirable.
- Gapping out is not desired: Maximum recall can be used to prevent a phase from gapping out. An example application of this is under coordinated operations where a left turn phase is lagging. By setting the lagging left turn phase to maximum recall, the phase will time for its maximum duration, allowing the adjacent coordinated phase to also time.

Run #3

Run #3 was configured with a maximum green time of 60 seconds on the major arterials, true max set to 80 seconds and an extension time of 5 seconds. The minor arterials had a maximum green time of 10 seconds, true max set to 20 and an extension time of 5 seconds.

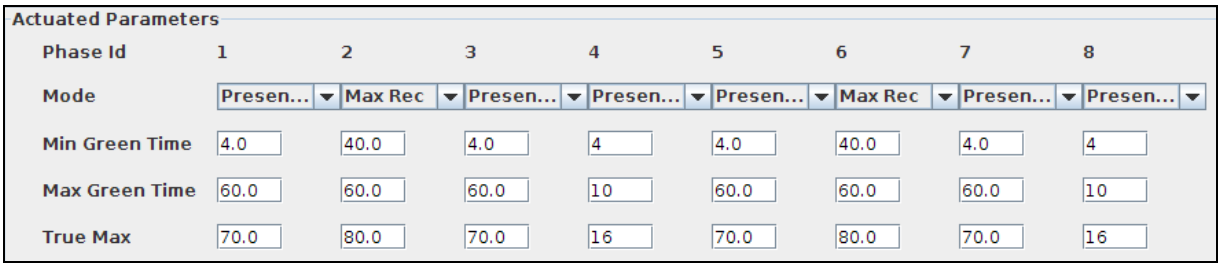


Figure 28: Actuated Parameters for Run #3

Run #4

Run #4 was configured with a maximum green time of 50 seconds on the major arterials, true max set to 60 seconds and an extension time of 5 seconds. The minor arterials had a maximum green time of 10 seconds, true max set to 16 and an extension time of 5 seconds.

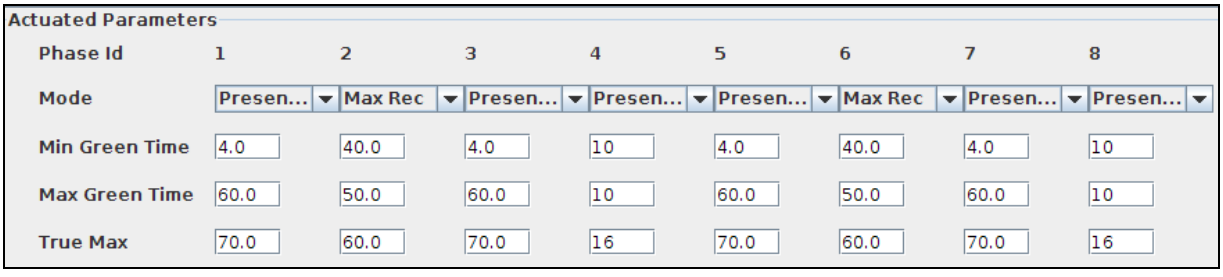


Figure 29: Actuated Parameters for Run #4

Table 37: Average Time Per Lane & Total Time of Vehicles In-Queue Using Simulation with Maximum Recall Mode (Left) Run #3 (Right) Run #4

Lane	Count	Average
LANE 8	98	25.55
LANE 4	91	26.82
LANE 2	93	5.82
LANE 6	108	6.01
Total Time (secs)		6134.86

Lane	Count	Average
LANE 8	89	26.35
LANE 4	71	24.56
LANE 2	115	6.77
LANE 6	149	8.97
Total Time (secs)		6203.99

Using these parameters for the maximum recall mode/presence the total time spent in-queue was greatly reduced compared to the benchmark. While the number of vehicles required to wait and their in-queue times increased for the minor arterials, the increase was relatively small. However the saving for the vehicles in the major arterial was substantial, with the worst case average in-queue time being half of that of the benchmark. Overall, for the worst-case of run #4, there was a 45 percent decrease in the total amount of time spent by vehicles in-queue. This shows that simply increasing the minimum green times in run #1 and #2, would have shown results with improved efficiency as well.

Max Out

In max out by the green time is extended by green extension each time the passage timer value is less than or equal to the gap time - only up to the maximum green time. Upon reaching the maximum green time the system transitions off upon receiving a conflict call. At the end of each extension the system compares the passage timer to the Gap Time to see if the system should extend again. Note the system only check passage time when conflicting call. Once a maximum green time is reached and a conflicting call is present the system transitions off that phase. Prior to max green time the system uses the passage time to decide whether to transition off the phase.

Run #5

Run #5 was configured in minimum recall with a minimum green time of 40 seconds and maximum green time of 60 seconds on the major arterials. The minor arterials was configured in max out mode with maximum green time of 14 seconds and maximum green time of 8 seconds. An extension time of 2 seconds and a gap time of 5 seconds were used.

Figure 30: Actuated Parameters for Run #5

Run #6

Run #6 was configured in minimum recall with a minimum green time of 40 seconds and maximum green time of 60 seconds on the major arterials. The minor arterials was configured in max out mode with maximum green time of 14 seconds and maximum green time of 8 seconds. An extension time of 3 seconds and a gap time of 6 seconds were used.

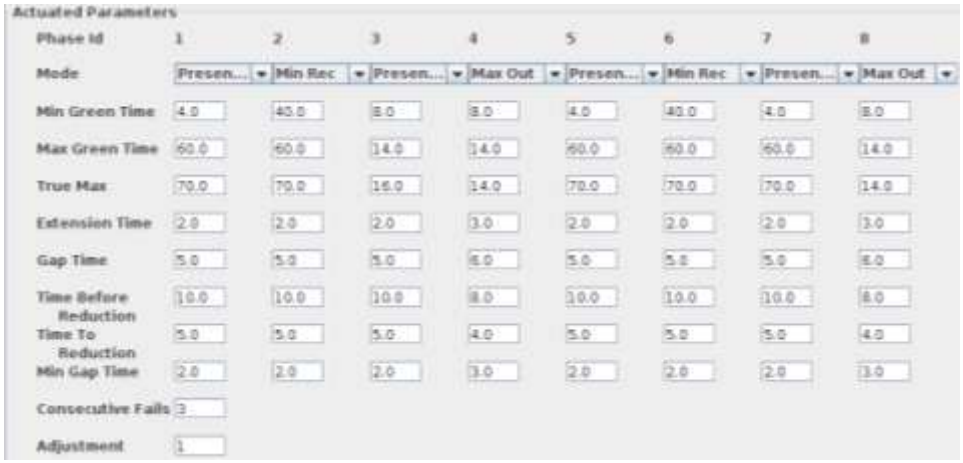


Figure 31: Actuated Parameters for Run #6

Table 38: Average Time Per Lane & Total Time of Vehicles In-Queue Using Simulation with Max Out Mode (Left) Run #5 (Right) Run #6

Lane	Count	Average
LANE 8	97	24.19
LANE 4	72	25.33
LANE 2	101	5.95
LANE 6	108	7.01
Total Time (secs)		5528.22

Lane	Count	Average
LANE 8	93	25.31
LANE 4	61	27.11
LANE 2	121	5.67
LANE 6	139	7.99
Total Time (secs)		5804.22

Using these parameters for the max out the total time spent in-queue was greatly reduced compared to the benchmark. While the number of vehicles required to wait and their in-queue times increased for the minor arterials, the increase was relatively small. However the saving for the vehicles in the major arterial was substantial, with the worst case average in-queue time being 52 percent of that of the benchmark. Overall, for the worst-case of run #4, there was a 48 percent decrease in the total amount of time spent by vehicles in-queue.

Gap Out

In this mode the software behaves just like max out mode. The one difference is that once the system

has a conflicting call after the minimum green time a timer, called the time to reduction timer, is started at the time of the conflict call. The green times continue to be extended using the same logic as the max out mode. The difference is that once the time to reduction timer is greater than or equal to the time before the gap time starts to be reduced. When reduction timer is greater than or equal to time before reduction a new timer called gap down time is started.

If (Gap Time > Min Gap) and (reduction timer >= Time Before Reduction)

Gap Time = Original Gap Time - ((Original Gap Time – Min Gap)/Time to Reduce)*gap down

Every time the end of the green extension is reached and there is a conflict call the gap time is computed (once the time before reduction time is passed and until the gap time equals the minimum gap) and the green is extended by the extension time. If the maximum green time is reached without the gap out terminating the phase then the maximum green time will terminate the phase.

Run #7

Run #7 was configured in minimum recall with a maximum green time of 40 seconds and maximum green time of 60 seconds on the major arterials. The minor arterials was configured in gap out mode with maximum green time of 14 seconds and minimum green time of 8 seconds. An extension time of 2 seconds, a gap time of 5 seconds, a time before reduction of 10 seconds, and a time to reduction of 5 seconds were used.

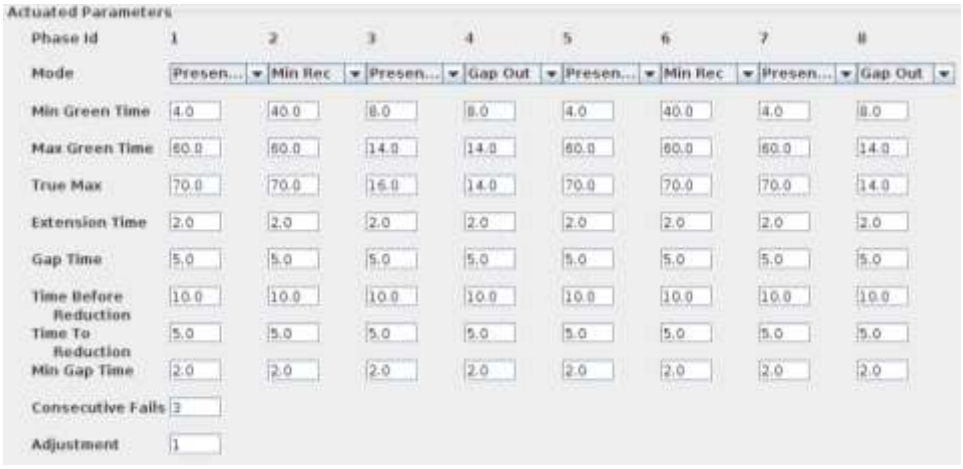


Figure 32: Actuated Parameters for Run #7

Run #8

Run #8 was configured in minimum recall with a maximum green time of 40 seconds and maximum green time of 60 seconds on the major arterials. The minor arterials was configured in gap out mode with maximum green time of 14 seconds and minimum green time of 8 seconds. An extension time of 3 seconds, a gap time of 6 seconds, a time before reduction of 10 seconds, and a time to reduction of 5 seconds were used.

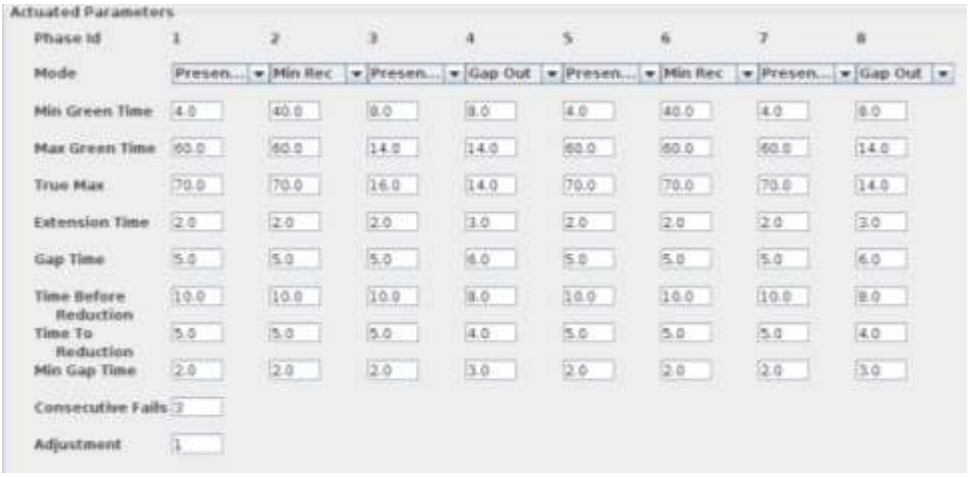


Figure 33: Actuated Parameters for Run #8

Table 39: Average Time Per Lane & Total Time of Vehicles In-Queue Using Simulation with Gap Out Mode (Left) Run #7 (Right) Run #8

Lane	Count	Average
LANE 8	89	26.35
LANE 4	71	24.56
LANE 2	115	6.77
LANE 6	149	8.97
Total Time (secs)		6203.99

Lane	Count	Average
LANE 8	93	26.35
LANE 4	72	24.56
LANE 2	123	6.77
LANE 6	157	8.97
Total Time (secs)		6459.87

Using these parameters for the gap out mode the total time spent in-queue was greatly reduced compared to the benchmark. While the number of vehicles required to wait and their in-queue times increased for the minor arterials, the increase was relatively small. However the saving for the vehicles in the major arterial was substantial, with the worst case average in-queue time being 58 percent of that of the benchmark. Overall, for the worst-case of run #8, there was a 42 percent decrease in the total amount of time spent by vehicles in-queue.

CHAPTER EIGHT: CONCLUSION

With populations increasing and municipalities' budgets decreasing the need for transportation infrastructure will become increasingly hard to meet. Using properly timed traffic control signals is critical to operating current roadway systems at maximum capacity. The days of traffic signal proprietary communication protocols, inflated service contracts, forced sole-source acquisitions, and "tie-in" sales practices are numbered by the advancement of the ATC systems. This fact will be a giant leap in reducing the cost of traffic signal maintenance and replacement, enabling cities and towns to use their limited budget to maximize their traffic signal upgrades. SCOPE is an open source, safety critical signal control application and provides the next logical step in the openness of today's traffic signal controllers.

All the hypotheses made before the experiment began were found to be correct:

- SCOPE provides the capability to safely yield right-of-way.
- SCOPE is capable of working in real-world intersections and is capable of passing Formal Qualification Testing (FQT).
- SCOPE provides a system that incorporates traffic models which can be configured to improve the effectiveness of an existing intersection.
- SCOPE provides a testbed for evaluation of traffic models, and the open-source nature of the project allows new models to be easily integrated.

While every run in the experiment did not produce results that decreased wait times of vehicles, every model tested if properly configured had the ability to do so. It is important to note that while discrepancies will exist between the accuracy in timing between a computer simulation and human

recording times, SCOPE was found in some runs to be significantly better performing than the live intersection. These results are not presenting the case that SCOPE outperformed SCOOT, as the logic of the intersection appeared to be similar to SCOPE's gap or max out model. The intersection even by visual inspection was obviously improperly timed, being too preemptive on the minor arterial. An increase in the minimum green time would have been greatly beneficial to the live intersection. Requiring 20 or so vehicles on the minor arterial to wait an additional average of 5 seconds, could yield huge benefits for approximately 80 to 100 vehicles on the major arterial. Had SCOOT's model been open-source it could have been integrated into SCOPE so that parameters could be tweaked and all things being equal they could have been evaluated by the simulation.

Once the NTCIP library is completed, SCOPE will undergo formal validation; with the source code will be made public shortly after. Upon release, SCOPE will reap the benefits that open source software provides. From the software's perspective the benefits of open sourcing include bug detection, better fault detection, and better design. For local governments that incorporate the open source software they will have complete visibility into the logic and interfaces behind the system. SCOPE will not decrease the need for properly maintained pre-timed signal plans, but it may reduce the cost of the maintenance and integration of advanced controls. Maintenance, integration and signal timing could be performed outside the sometimes inflated contract prices by local, state, or third-party engineers. At the completion of Phase II a solid safety-critical intersection control system will be delivered to the DOT. Once SCOPE lands in the open source community, the possibility for new features and ingenuity is endless.

CHAPTER TEN: FUTURE WORK

The current state of SCOPE allows it to control an intersection in a small to medium city. However, larger cities (and some smaller municipalities) contain Central Management Stations (CMS) that use the NEMA National Transportation Communications for ITS Protocol (NTCIP) standard to communicate between a Central Management Station and a traffic controller. In order for SCOPE to be competitive in the marketplace, it must incorporate the NTCIP standard. Therefore a stand-alone NTCIP interface will be developed that will permit SCOPE to communicate with NTCIP compliant central management stations. Because this interface will be modular and loosely coupled, any external program needing NTCIP compliance will be able to incorporate it.

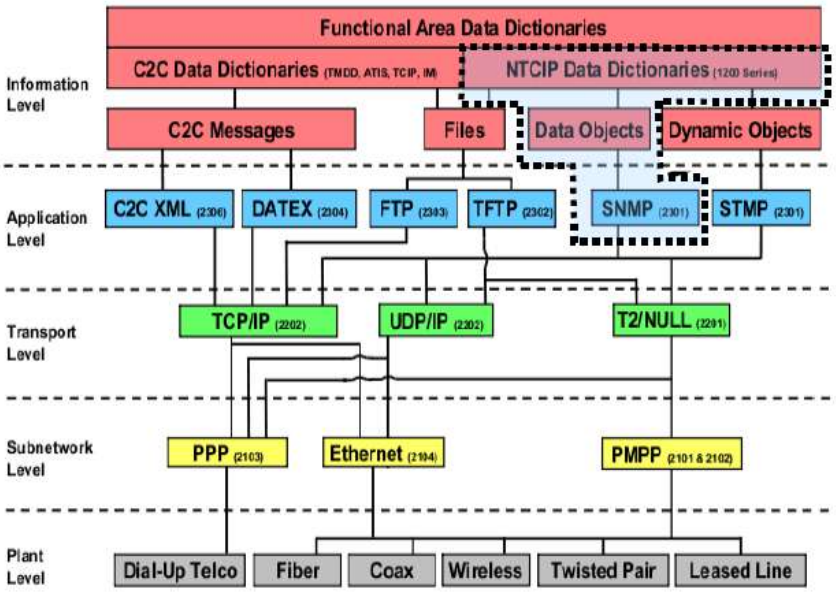


Figure 34: NTCIP Framework

Figure 34 shows the NTCIP Framework. An NTCIP API library that allows user applications (such as SCOPE) residing in the application layer to mutate and access the NTCIP 1201 and 1202 traffic controller

data objects in the information layer. So that the NTCIP interface is useful to other programs, from vendors independent of the SCOPE effort, the NTCIP interface will be a separate entity and is de-coupled from SCOPE. In other words, the NTCIP interface will be a standalone library capable of being included by any vendor's product. Figure 35 shows the ATI NTCIP architectural design with SCOPE included. The NTCIP interface library will contain APIs for the C, C++, and Ada programming languages.

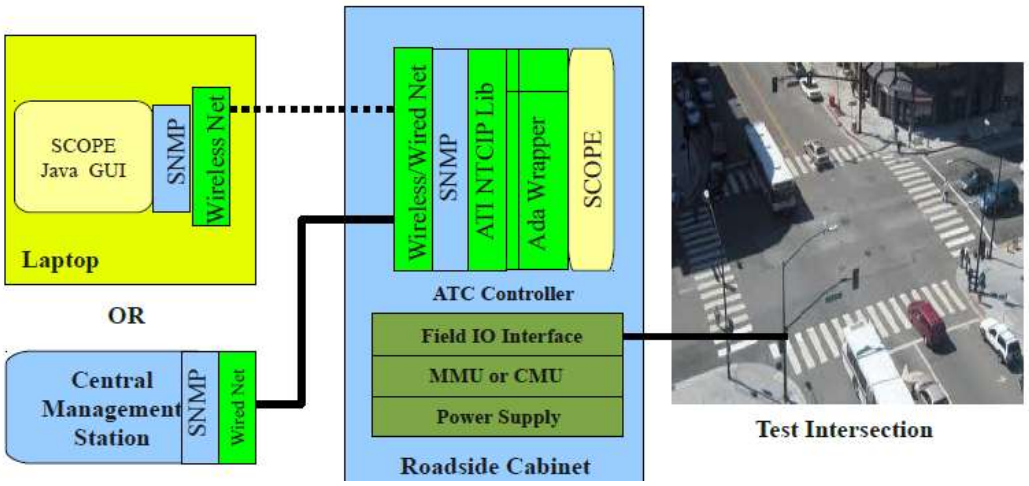


Figure 35: SCOPE / NTCIP Architecture

APPENDIX A: INDUSTRY INTEREST IN RESEARCH

Industry Interest In Research

SCOPE has already received interest from a traffic control company based out of Canada. In the following email an engineer from the company states they are working on developing video monitoring software to optimize traffic flow and mirrors the needs for an open-source solution laid out in this paper.

On 07/19/2011 03:12 PM, Nicholas Jankovic wrote:

Greetings,

I work for a Canadian traffic monitoring company named Miovision and we are in the process of developing an new adaptive traffic control solution for a customer. The goal is to optimize traffic flow through their downtown core using embedded video analytics modules at each intersection (basically a very smart intersection sensors). At this stage of the design process, we have a good handle on the embedded system and video analytics; however, we still need a mechanism to invoke changes at the intersection level. Ideally, we would like to have our embedded video system communicate optimized phase changes directly to the traffic controller. Unfortunately, **there is no clear means of communicating such information to proprietary traffic control applications**. Therefore, we would like to evaluate the feasability of interfacing with SCOPE using your Java interface or possibly modifying SCOPE to accept external video sensor inputs over an external serial connection. Conveniently, we have a Peek ATC-1000 in our lab and our customer already has Linux-based ATC controllers installed at their intersections, so SCOPE appears to be a good alternative to communication to proprietary applications using NTCIP. I am aware that SCOPE will not be publically available until October, but any additional information on SCOPE would be greatly appreciated.

Thank you.

Nicholas Jankovic

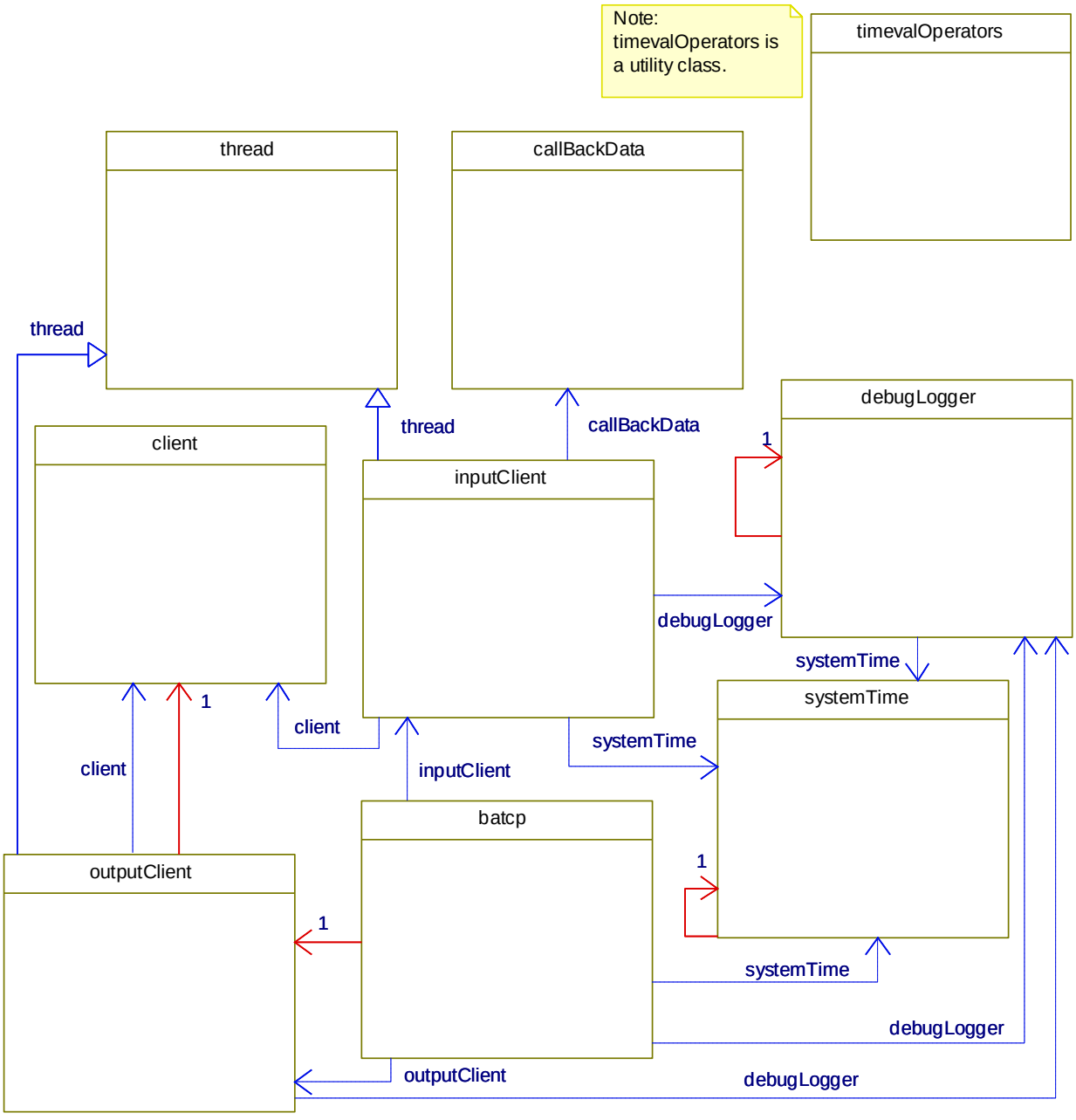
Applications Embedded Developer

Miovision Technologies Incorporated

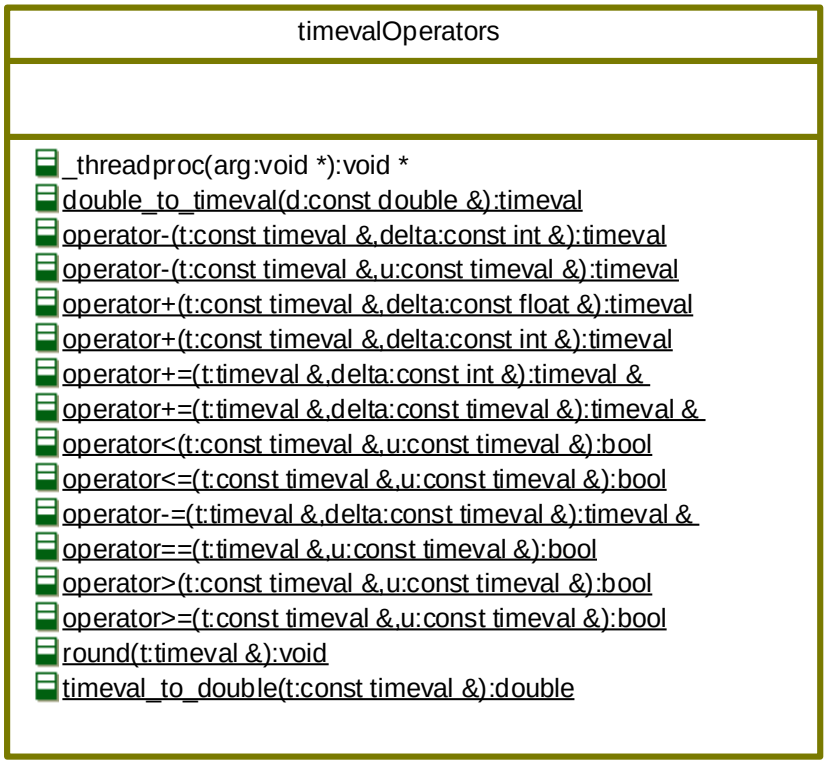
APPENDIX B: UML DIAGRAMS

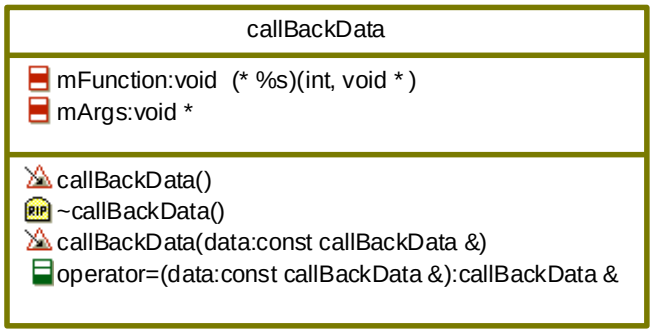
Unified Modeling Language (UML)

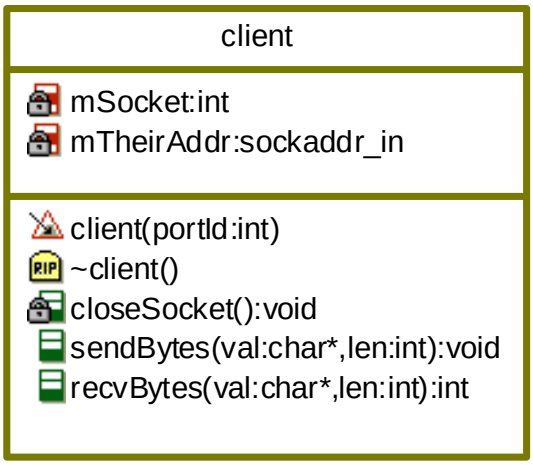
Class Diagram

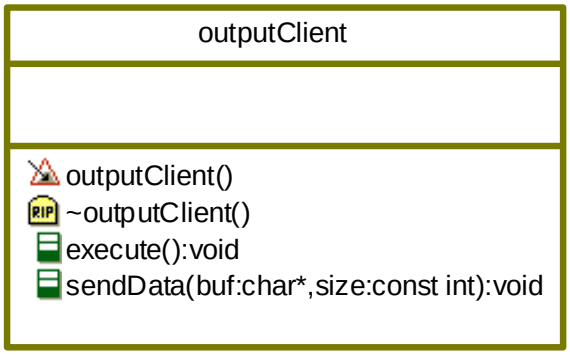


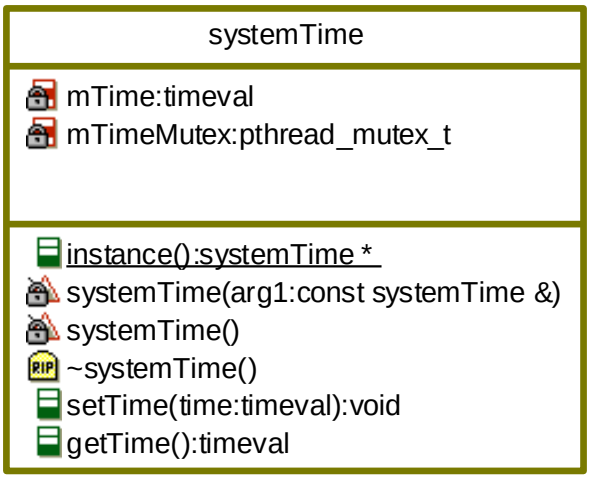
Class Operations And Attributes



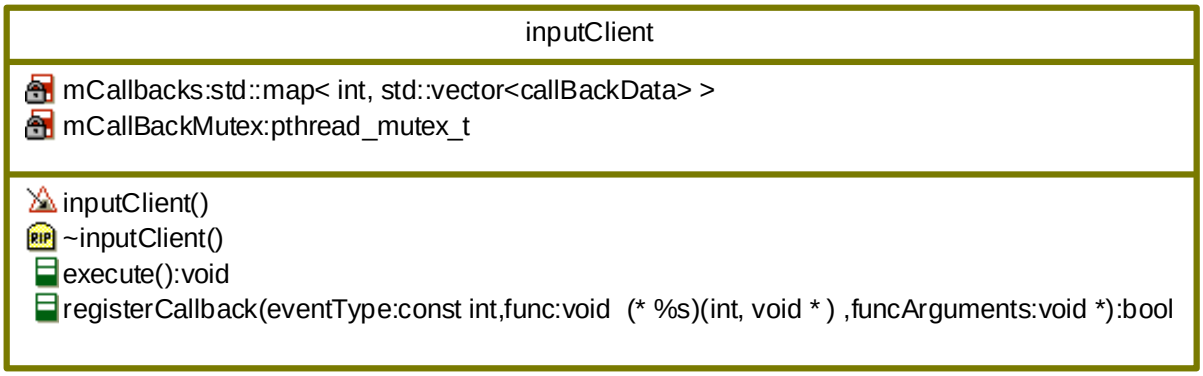


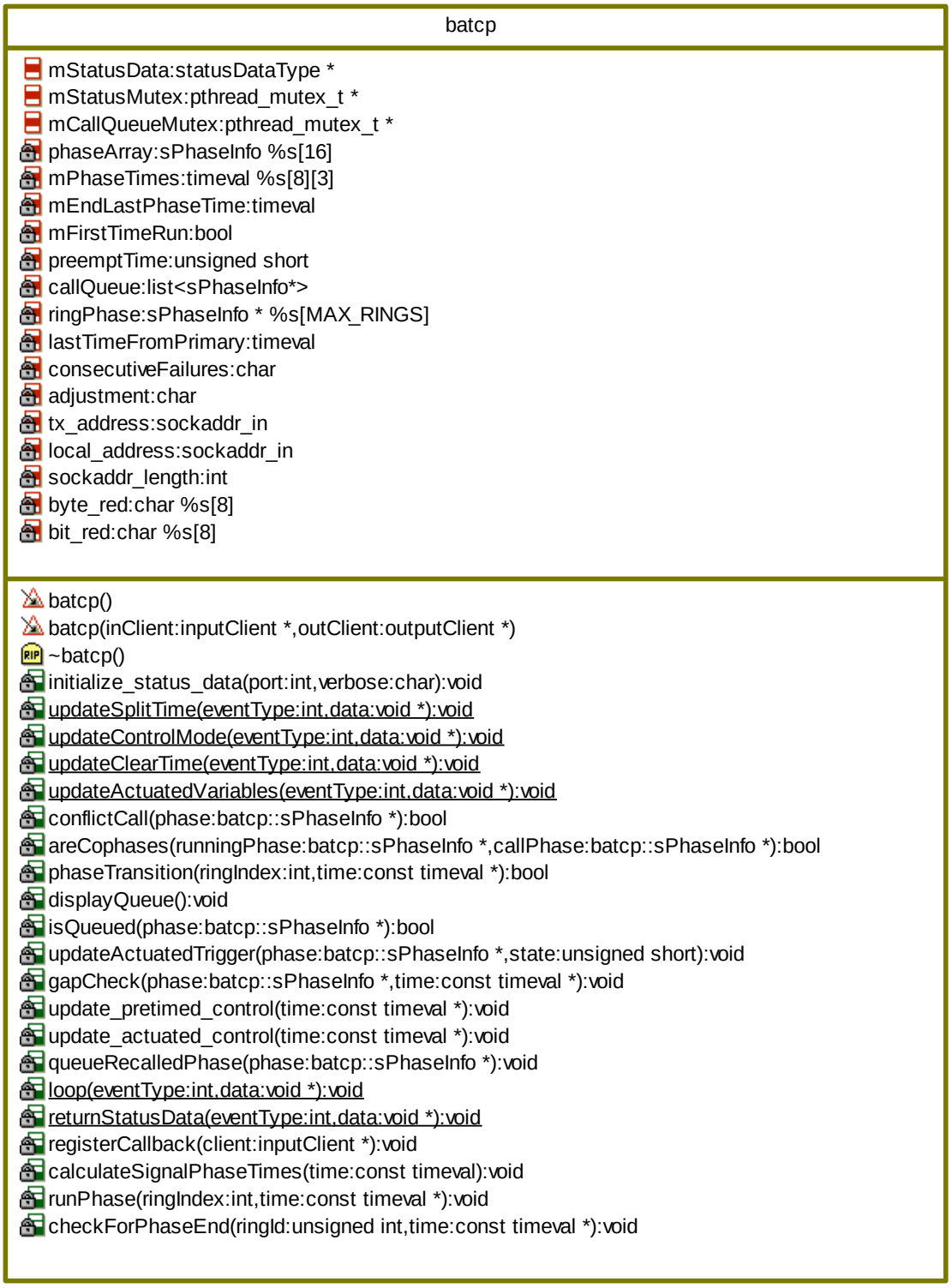



outputClient
outputClient()
~outputClient()
execute():void
sendData(buf:char*,size:const int):void

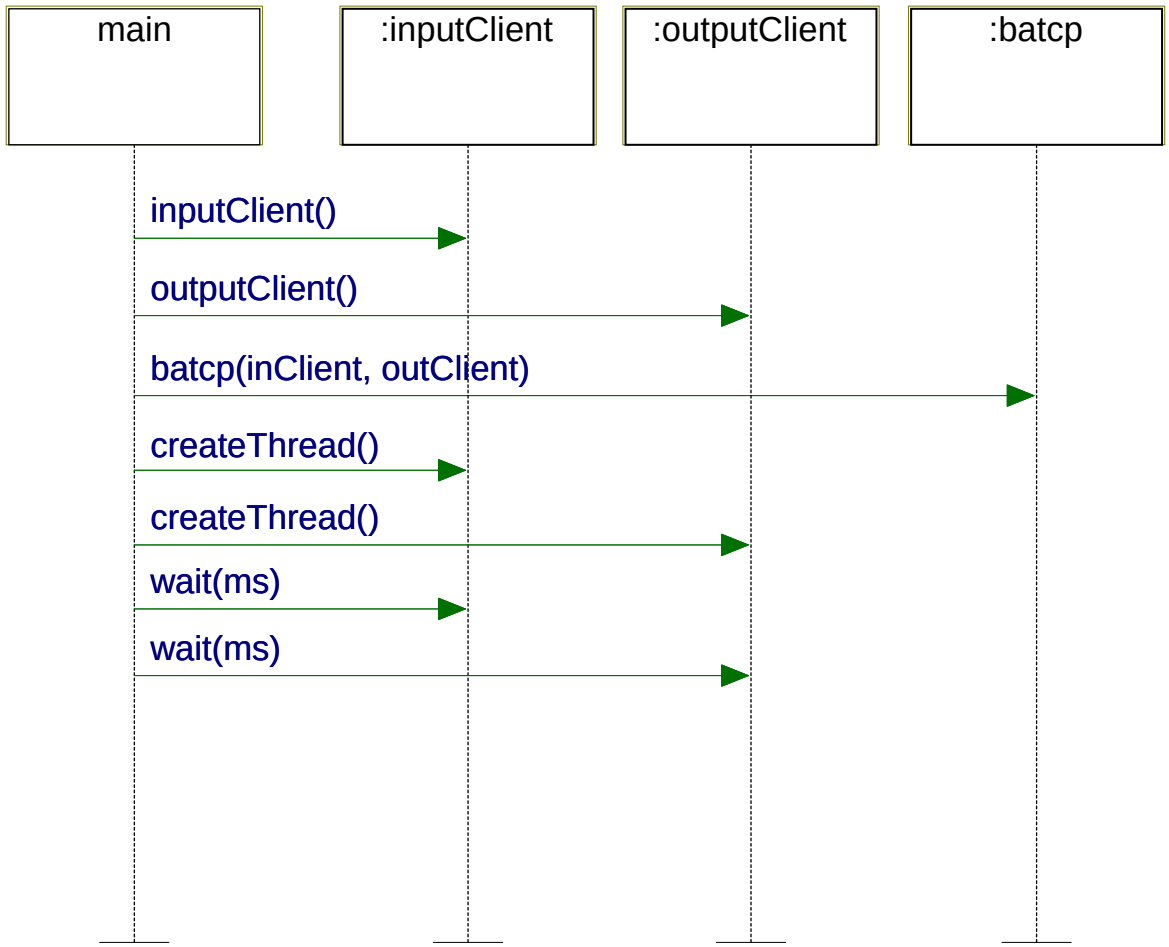

systemTime
mTime:timeval
mTimeMutex:pthread_mutex_t
instance():systemTime *
systemTime(arg1:const systemTime &)
systemTime()
~systemTime()
setTime(time:timeval):void
getTime():timeval


thread
mActive:bool
mQuitThreadFlag:bool
mThreadFunc:void (* %s)(void *)
mThreadArgs:void *
mThread:pthread_t
thread()
thread(name:const char *)
~thread()
init():void
createThread():bool
createThread(func:void (* %s)(void *) ,args:void *):bool
stopThread(ms:const int=500):bool
isActive():bool
wait(ms:const unsigned=INFINITE):bool
kill():bool
getId():pthread_t
setPriority(priority:const int):bool
quitFlag():bool
execute():void
cleanup():void
debugLogger
mDebugLevel:int
instance():debugLogger *
debugLogger()
~debugLogger()
setDebugLevel(debugLevel:int):void
print(statement:const char *,debugLevel:int,timeStamp:const timeval *=NULL):void

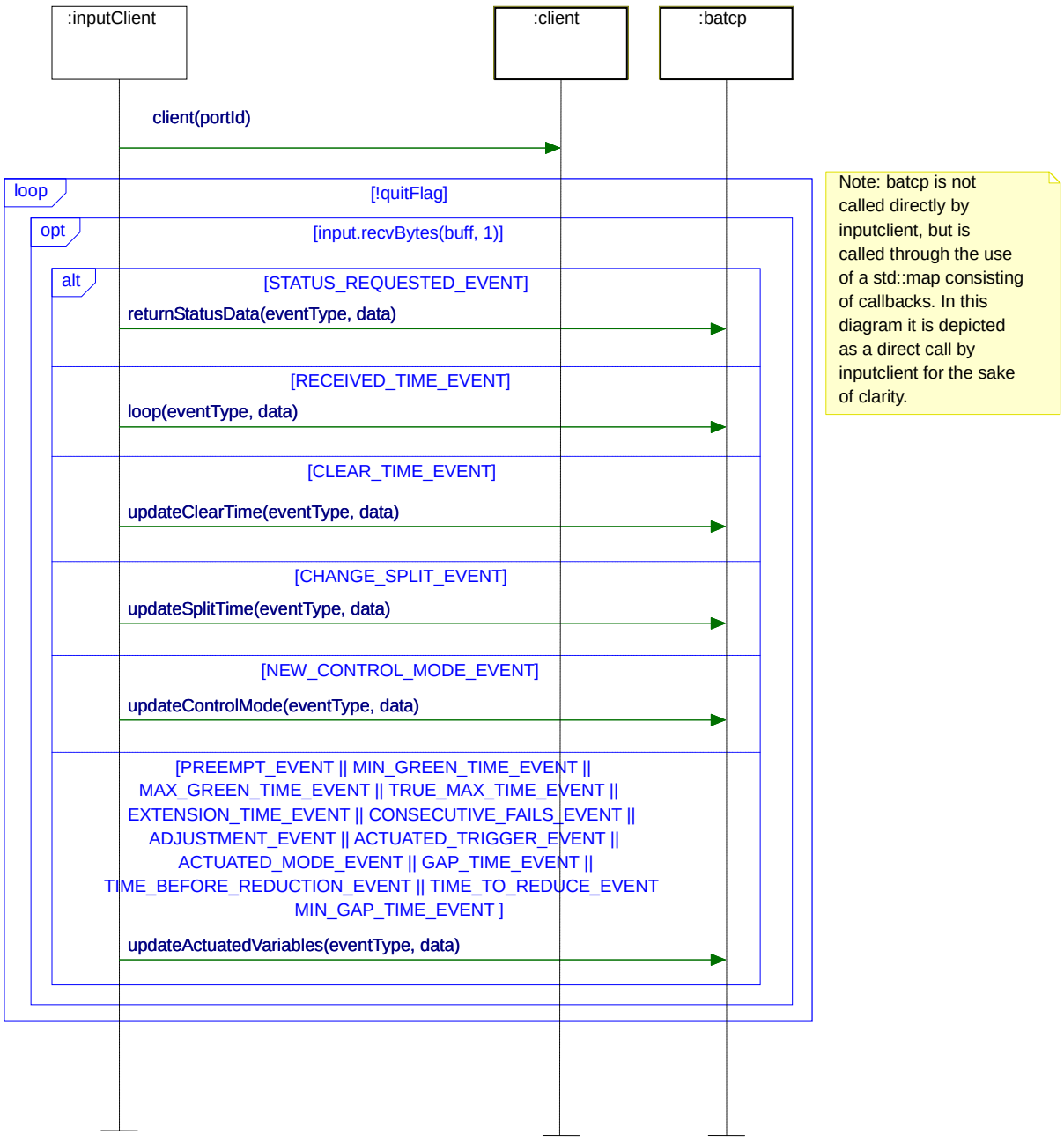

inputClient
mCallbacks:std::map< int, std::vector<callBackData> >
mCallBackMutex:pthread_mutex_t
inputClient()
~inputClient()
execute():void
registerCallback(eventType:const int,func:void (* %s)(int, void *) ,funcArguments:void *):bool


batcp
mStatusData:statusDataType *
mStatusMutex:pthread_mutex_t *
mCallQueueMutex:pthread_mutex_t *
phaseArray:sPhaseInfo %s[16]
mPhaseTimes:timeval %s[8][3]
mEndLastPhaseTime:timeval
mFirstTimeRun:bool
preemptTime:unsigned short
callQueue:list<sPhaseInfo*>
ringPhase:sPhaseInfo * %s[MAX_RINGS]
lastTimeFromPrimary:timeval
consecutiveFailures:char
adjustment:char
tx_address:sockaddr_in
local_address:sockaddr_in
sockaddr_length:int
byte_red:char %s[8]
bit_red:char %s[8]
batcp()
batcp(inClient:inputClient *,outClient:outputClient *)
~batcp()
initialize_status_data(port:int,verbose:char):void
updateSplitTime(eventType:int,data:void *):void
updateControlMode(eventType:int,data:void *):void
updateClearTime(eventType:int,data:void *):void
updateActuatedVariables(eventType:int,data:void *):void
conflictCall(phase:batcp::sPhaseInfo *):bool
areCophases(runningPhase:batcp::sPhaseInfo *,callPhase:batcp::sPhaseInfo *):bool
phaseTransition(ringIndex:int,time:const timeval *):bool
displayQueue():void
isQueued(phase:batcp::sPhaseInfo *):bool
updateActuatedTrigger(phase:batcp::sPhaseInfo *,state:unsigned short):void
gapCheck(phase:batcp::sPhaseInfo *,time:const timeval *):void
update_pretimed_control(time:const timeval *):void
update_actuated_control(time:const timeval *):void
queueRecalledPhase(phase:batcp::sPhaseInfo *):void
loop(eventType:int,data:void *):void
returnStatusData(eventType:int,data:void *):void
registerCallback(client:inputClient *):void
calculateSignalPhaseTimes(time:const timeval):void
runPhase(ringIndex:int,time:const timeval *):void
checkForPhaseEnd(ringId:unsigned int,time:const timeval *):void

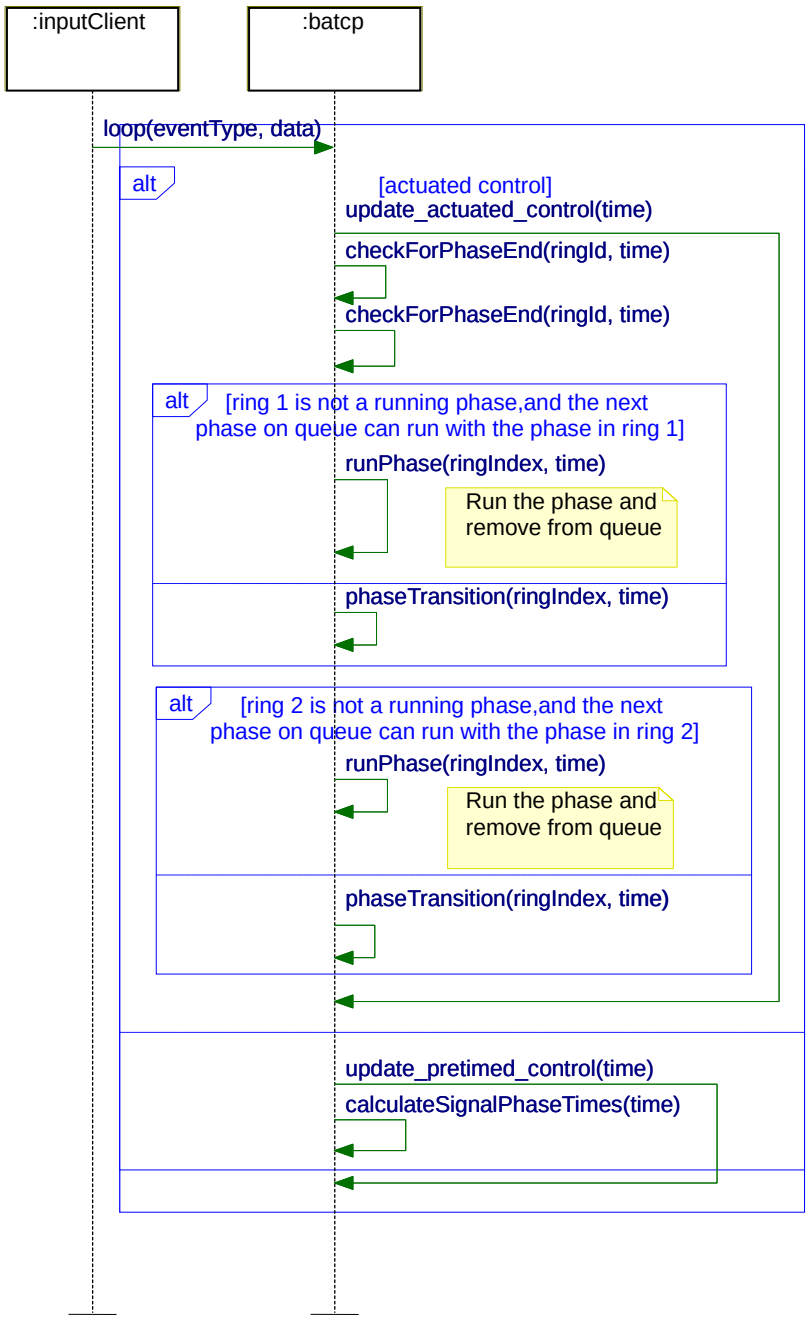
Sequence Flow
Initialization



Incoming Message



Incoming Time


:inputClient
:batcp
loop(eventType, data)
alt
[actuated control]
update_actuated_control(time)
checkForPhaseEnd(ringId, time)
checkForPhaseEnd(ringId, time)
alt
[ring 1 is not a running phase,and the next phase on queue can run with the phase in ring 1]
runPhase(ringIndex, time)
Run the phase and remove from queue
phaseTransition(ringIndex, time)
alt
[ring 2 is not a running phase,and the next phase on queue can run with the phase in ring 2]
runPhase(ringIndex, time)
Run the phase and remove from queue
phaseTransition(ringIndex, time)
update_pretimed_control(time)
calculateSignalPhaseTimes(time)

APPENDIX C: LIVE DATA CAPTURE

0 2 new_call
2000 6 new_call
2000 8 ped_call
3000 2 new_call
4000 2 new_call
4000 6 new_call
6000 2 new_call
6000 2 new_call
8000 2 new_call
9000 8 new_call
11000 2 new_call
13000 2 new_call
13000 6 new_call
16000 2 new_call
17000 2 new_call
18000 6 new_call
19000 2 new_call
19000 6 new_call
21000 2 new_call
21000 8 ped_call
22000 2 new_call
22000 6 new_call
24000 2 new_call
26000 2 new_call
27000 2 new_call
32000 4 new_call
38000 8 new_call
42000 2 new_call
47000 6 new_call
47000 6 new_call
48000 6 new_call
50000 2 new_call
56000 8 new_call
58000 2 new_call
70000 6 new_call
71000 6 new_call
74000 2 new_call
78000 6 new_call
80000 8 new_call
101000 6 new_call
108000 8 ped_call
112000 8 new_call
113000 6 new_call
114000 6 new_call
114000 8 new_call
115000 6 new_call
115000 8 ped_call
118000 6 new_call
120000 2 new_call
124000 6 new_call
128000 2 new_call
128000 2 new_call
129000 6 new_call
129000 6 new_call
130000 6 new_call
142000 2 new_call
143000 2 new_call
145000 2 new_call
146000 2 new_call
152000 2 new_call
154000 2 new_call
157000 2 new_call
158000 6 new_call
158000 6 new_call
159000 6 new_call
160000 2 new_call
164000 2 new_call
164000 6 new_call
165000 2 new_call
165000 6 new_call
168000 2 new_call
173000 2 new_call
176000 8 ped_call
180000 8 new_call
187000 8 new_call
193000 8 new_call
202000 6 new_call
205000 6 new_call
208000 2 new_call
208000 8 new_call
210000 2 new_call
236000 6 new_call
236000 8 ped_call
239000 8 new_call
240000 6 new_call
243000 4 new_call
244000 4 new_call
248000 6 new_call
253000 6 new_call
253000 8 new_call
254000 6 new_call
261000 4 new_call
264000 6 new_call
268000 6 new_call
270000 2 new_call
271000 2 new_call
272000 2 new_call
274000 6 new_call
276000 2 new_call
278000 2 new_call
281000 2 new_call
282000 2 new_call
284000 2 new_call
289000 2 new_call
291000 2 new_call
291000 6 new_call
302000 2 new_call
303000 2 new_call
305000 2 new_call
306000 2 new_call
310000 2 new_call
323000 8 new_call
325000 2 new_call
325000 6 new_call
337000 2 new_call
347000 6 new_call
359000 4 new_call
369000 6 new_call
372000 6 new_call
374000 6 new_call
376000 6 new_call
381000 6 new_call
383000 6 new_call
395000 6 new_call
395000 6 new_call
397000 4 new_call
399000 6 new_call
401000 2 new_call
403000 2 new_call
404000 2 new_call
405000 8 new_call
406000 2 new_call
409000 2 new_call
409000 6 new_call
411000 2 new_call
412000 2 new_call
419000 8 ped_call
421000 4 new_call
425000 2 new_call
429000 4 new_call
434000 2 new_call
434000 6 new_call
435000 6 new_call
436000 6 new_call
441000 6 new_call
441000 6 new_call
447000 2 new_call
455000 8 new_call
458000 2 new_call
460000 2 new_call
465000 2 new_call
471000 2 new_call
472000 8 new_call
491000 2 new_call
492000 4 new_call
494000 6 new_call
499000 6 new_call
510000 6 new_call
511000 6 new_call
512000 6 new_call

513000 6 new_call
516000 6 new_call
520000 4 new_call
525000 2 new_call
526000 4 new_call
529000 2 new_call
534000 4 new_call
540000 6 new_call
544000 2 new_call
549000 6 new_call
550000 6 new_call
551000 6 new_call
553000 2 new_call
553000 6 new_call
555000 2 new_call
555000 6 new_call
555000 6 new_call
556000 2 new_call
557000 6 new_call
558000 2 new_call
560000 6 new_call
561000 2 new_call
563000 2 new_call
564000 6 new_call
564000 8 ped_call
565000 2 new_call
568000 2 new_call
570000 2 new_call
574000 2 new_call
584000 2 new_call
585000 4 new_call
588000 4 new_call
588000 8 new_call
589000 6 new_call
589000 8 new_call
590000 4 new_call
591000 8 new_call
592000 6 new_call
592000 6 new_call
595000 4 new_call
596000 2 new_call
596000 6 new_call
605000 4 new_call
611000 4 new_call
613000 8 new_call
614000 2 new_call
632000 8 new_call
638000 8 ped_call
639000 6 new_call
640000 6 new_call
642000 6 new_call
642000 6 new_call
643000 6 new_call
647000 6 new_call
650000 6 new_call
654000 6 new_call
657000 8 new_call
660000 8 new_call
663000 8 new_call
664000 6 new_call
666000 8 new_call
669000 8 new_call
671000 6 new_call
671000 8 new_call
685000 6 new_call
686000 6 new_call
690000 2 new_call
691000 2 new_call
692000 6 new_call
694000 2 new_call
694000 2 new_call
694000 6 new_call
695000 2 new_call
695000 2 new_call
695000 6 new_call
697000 2 new_call
698000 6 new_call
699000 2 new_call
700000 2 new_call
701000 2 new_call
702000 2 new_call
702000 2 new_call
702000 8 ped_call
704000 2 new_call
704000 2 new_call
707000 2 new_call
710000 6 new_call
724000 6 new_call
731000 2 new_call
735000 8 new_call
742000 6 new_call
773000 6 new_call
773000 6 new_call
775000 6 new_call
775000 6 new_call
777000 6 new_call
793000 8 new_call
797000 8 new_call
798000 6 new_call
805000 6 new_call
807000 6 new_call
807000 4 new_call
808000 6 new_call
813000 6 new_call
814000 6 new_call
816000 8 new_call
818000 2 new_call
818000 2 new_call
821000 2 new_call
823000 6 new_call
824000 8 ped_call
825000 6 new_call
835000 2 new_call
840000 2 new_call
842000 2 new_call
843000 2 new_call
845000 2 new_call
860000 6 new_call
861000 6 new_call
862000 4 new_call
866000 2 new_call
867000 4 new_call
876000 2 new_call
878000 2 new_call
887000 8 new_call
904000 4 new_call
906000 4 new_call
909000 6 new_call
909000 8 ped_call
910000 6 new_call
910000 8 new_call
911000 6 new_call
924000 4 new_call
924000 8 new_call
930000 4 new_call
935000 6 new_call
940000 6 new_call
945000 2 new_call
946000 8 new_call
947000 2 new_call
949000 2 new_call
949000 6 new_call
957000 2 new_call
960000 8 ped_call
969000 2 new_call
979000 6 new_call
984000 8 ped_call
986000 6 new_call
994000 2 new_call
995000 8 new_call
996000 2 new_call
1002000 2 new_call
1004000 2 new_call
1016000 6 new_call
1019000 6 new_call
1019000 6 new_call
1020000 8 ped_call
1025000 8 new_call
1026000 2 new_call
1035000 2 new_call
1047000 6 new_call
1049000 6 new_call
1053000 6 new_call
1053000 6 new_call

1055000 6 new_call
1061000 6 new_call
1064000 6 new_call
1075000 6 new_call
1076000 6 new_call
1077000 8 new_call
1083000 2 new_call
1083000 2 new_call
1086000 2 new_call
1086000 6 new_call
1087000 6 new_call
1088000 2 new_call
1090000 6 new_call
1093000 6 new_call
1094000 2 new_call
1094000 8 ped_call
1095000 4 new_call
1097000 2 new_call
1110000 4 new_call
1120000 8 new_call
1122000 6 new_call
1122000 4 new_call
1124000 4 new_call
1131000 2 new_call
1131000 6 new_call
1131000 4 new_call
1133000 6 new_call
1152000 4 ped_call
1153000 2 new_call
1161000 2 new_call
1172000 8 ped_call
1176000 6 new_call
1182000 8 new_call
1185000 6 new_call
1185000 6 new_call
1186000 6 new_call
1188000 6 new_call
1188000 8 new_call
1189000 6 new_call
1195000 8 new_call
1208000 2 new_call
1219000 2 new_call
1219000 2 new_call
1222000 6 new_call
1224000 2 new_call
1224000 6 new_call
1226000 2 new_call
1226000 2 new_call
1227000 2 new_call
1228000 2 new_call
1229000 6 new_call
1230000 2 new_call
1231000 6 new_call
1232000 6 new_call
1233000 2 new_call
1233000 8 new_call
1234000 6 new_call
1236000 6 new_call
1236000 6 new_call
1237000 2 new_call
1239000 6 new_call
1239000 4 new_call
1240000 6 new_call
1241000 2 new_call
1246000 6 new_call
1248000 2 new_call
1250000 2 new_call
1250000 4 ped_call
1253000 2 new_call
1256000 2 new_call
1257000 8 new_call
1258000 2 new_call
1262000 2 new_call
1266000 2 new_call
1266000 4 new_call
1269000 8 new_call
1270000 2 new_call
1270000 8 ped_call
1280000 2 new_call
1289000 8 new_call
1291000 2 new_call
1291000 2 new_call
1296000 2 new_call
1299000 6 new_call
1312000 6 new_call
1314000 2 new_call
1315000 6 new_call
1315000 6 new_call
1322000 6 new_call
1324000 6 new_call
1328000 6 new_call
1328000 4 new_call
1328000 8 new_call
1341000 8 new_call
1349000 6 new_call
1349000 6 ped_call
1354000 2 new_call
1355000 2 new_call
1355000 2 new_call
1356000 6 new_call
1359000 2 new_call
1360000 8 new_call
1362000 2 new_call
1373000 8 new_call
1375000 6 new_call
1377000 4 new_call
1381000 2 new_call
1382000 8 ped_call
1386000 6 new_call
1387000 6 new_call
1390000 6 new_call
1393000 6 new_call
1396000 6 new_call
1397000 2 new_call
1398000 8 new_call
1399000 2 new_call
1402000 6 new_call
1402000 8 new_call
1407000 2 new_call
1420000 8 new_call
1424000 6 new_call
1426000 6 new_call
1426000 8 ped_call
1431000 8 new_call
1433000 8 ped_call
1436000 8 new_call
1438000 4 new_call
1439000 8 new_call
1443000 8 ped_call
1448000 6 new_call
1450000 6 new_call
1451000 6 new_call
1451000 6 new_call
1452000 6 new_call
1454000 6 new_call
1456000 6 new_call
1456000 6 new_call
1458000 6 new_call
1459000 6 new_call
1462000 2 new_call
1465000 2 new_call
1474000 8 new_call
1477000 4 new_call
1478000 4 new_call
1479000 4 new_call
1486000 6 new_call
1486000 6 new_call
1486000 4 new_call
1487000 6 new_call
1487000 4 new_call
1491000 2 new_call
1491000 6 new_call
1491000 4 new_call
1495000 2 new_call
1497000 6 new_call
1498000 2 new_call
1499000 6 new_call
1502000 2 new_call
1502000 2 new_call
1502000 6 new_call
1503000 2 new_call
1504000 2 new_call
1504000 4 ped_call
1505000 6 new_call

1506000 6 new_call
1507000 2 new_call
1508000 2 new_call
1509000 8 ped_call
1510000 2 new_call
1510000 6 new_call
1512000 2 new_call
1514000 2 new_call
1514000 6 new_call
1515000 2 new_call
1516000 2 new_call
1518000 2 new_call
1518000 2 new_call
1522000 4 new_call
1527000 6 new_call
1528000 2 new_call
1528000 4 new_call
1540000 8 new_call
1544000 2 new_call
1544000 2 new_call
1545000 2 new_call
1546000 2 new_call
1550000 2 new_call
1557000 8 new_call
1564000 8 ped_call
1565000 6 new_call
1573000 6 new_call
1576000 6 new_call
1579000 6 new_call
1581000 6 new_call
1582000 6 new_call
1589000 4 new_call
1598000 4 new_call
1599000 2 new_call
1600000 2 new_call
1605000 2 new_call
1605000 4 new_call
1606000 2 new_call
1607000 2 new_call
1607000 2 new_call
1611000 2 new_call
1613000 6 new_call
1619000 2 new_call
1619000 6 new_call
1620000 6 new_call
1620000 6 new_call
1624000 6 new_call
1625000 6 new_call
1626000 2 new_call
1627000 6 new_call
1629000 8 new_call
1630000 6 new_call
1632000 6 new_call
1632000 6 new_call
1635000 4 new_call
1635000 8 ped_call
1638000 6 new_call
1643000 6 new_call
1647000 8 new_call
1648000 2 new_call
1651000 2 new_call
1652000 2 new_call
1652000 2 new_call
1653000 8 new_call
1654000 2 new_call
1655000 2 new_call
1655000 6 new_call
1663000 8 new_call
1672000 6 new_call
1698000 6 new_call
1699000 6 new_call
1707000 4 new_call
1710000 6 new_call
1711000 6 new_call
1714000 6 new_call
1717000 8 new_call
1724000 6 new_call
1725000 2 new_call
1727000 4 new_call
1728000 2 new_call
1728000 2 new_call
1730000 2 new_call
1731000 2 new_call
1732000 2 new_call
1734000 2 new_call
1735000 2 new_call
1739000 2 new_call
1740000 2 new_call
1744000 6 new_call
1745000 2 new_call
1747000 2 new_call
1748000 2 new_call
1750000 2 new_call
1750000 6 new_call
1750000 6 new_call
1751000 6 new_call
1751000 6 new_call
1752000 2 new_call
1752000 8 ped_call
1754000 2 new_call
1754000 2 new_call
1754000 6 new_call
1755000 2 new_call
1755000 6 new_call
1758000 2 new_call
1758000 6 new_call
1760000 2 new_call
1761000 6 new_call
1765000 2 new_call
1767000 2 new_call
1772000 8 new_call
1775000 8 ped_call
1778000 2 new_call
1790000 2 new_call
1791000 8 new_call
1794000 2 new_call
1795000 2 new_call
1803000 6 new_call
1804000 8 new_call
1807000 2 new_call
1821000 2 new_call
1824000 6 new_call
1824000 6 new_call
1826000 6 new_call
1826000 6 new_call
1828000 6 new_call
1828000 8 new_call
1834000 6 new_call
1835000 6 new_call
1839000 6 new_call
1840000 6 new_call
1840000 8 new_call
1850000 6 new_call
1853000 6 new_call
1854000 6 new_call
1854000 6 new_call
1856000 6 new_call
1858000 2 new_call
1858000 2 new_call
1859000 2 new_call
1860000 2 new_call
1861000 2 new_call
1864000 6 new_call
1865000 2 new_call
1868000 2 new_call
1868000 2 new_call
1871000 6 new_call
1878000 2 new_call
1878000 2 new_call
1879000 2 new_call
1881000 6 new_call
1884000 6 new_call
1885000 6 new_call
1886000 6 new_call
1890000 2 new_call
1891000 2 new_call
1892000 6 new_call
1894000 2 new_call
1894000 6 new_call
1896000 6 new_call
1900000 2 new_call
1904000 2 new_call
1908000 2 new_call
1908000 6 new_call

1914000 6 new_call
1918000 8 new_call
1919000 2 new_call
1929000 6 new_call
1932000 6 new_call
1932000 4 new_call
1939000 4 new_call
1946000 4 new_call
1949000 4 new_call
1955000 2 new_call
1959000 6 new_call
1962000 6 new_call
1962000 6 new_call
1964000 6 new_call
1965000 6 new_call
1967000 6 new_call
1970000 6 new_call
1971000 6 new_call
1980000 6 new_call
1984000 6 new_call
1984000 6 new_call
1985000 6 new_call
1986000 8 ped_call
1987000 6 new_call
1988000 6 new_call
1990000 2 new_call
1990000 6 new_call
1992000 2 new_call
1993000 2 new_call
1993000 2 new_call
1994000 2 new_call
1996000 2 new_call
1999000 2 new_call
1999000 2 new_call
1999000 6 ped_call
2002000 2 new_call
2002000 6 new_call
2002000 4 new_call
2003000 2 new_call
2007000 2 new_call
2015000 6 new_call
2019000 2 new_call
2034000 2 new_call
2034000 2 new_call
2035000 2 new_call
2038000 2 new_call
2039000 6 ped_call
2040000 2 new_call
2040000 2 new_call
2043000 2 new_call
2043000 2 new_call
2046000 2 new_call
2046000 2 new_call
2049000 2 new_call
2050000 2 new_call
2050000 2 new_call
2051000 2 new_call
2052000 2 new_call
2053000 2 new_call
2054000 6 new_call
2094000 6 ped_call
2099000 6 new_call
2100000 6 new_call
2101000 6 new_call
2102000 6 new_call
2104000 6 new_call
2104000 6 new_call
2105000 6 new_call
2106000 2 new_call
2106000 6 new_call
2107000 2 new_call
2117000 2 new_call
2120000 6 new_call
2121000 2 new_call
2122000 2 new_call
2123000 2 new_call
2124000 2 new_call
2124000 2 new_call
2127000 2 new_call
2129000 2 new_call
2134000 6 new_call
2134000 8 new_call
2138000 2 new_call
2140000 6 new_call
2140000 4 new_call
2141000 2 new_call
2141000 4 new_call
2145000 6 new_call
2146000 2 new_call
2146000 2 new_call
2147000 6 new_call
2148000 2 new_call
2148000 6 new_call
2154000 2 new_call
2165000 6 new_call
2166000 2 new_call
2166000 6 new_call
2167000 2 new_call
2167000 6 new_call
2170000 6 new_call
2172000 6 new_call
2173000 6 new_call
2177000 2 new_call
2178000 6 new_call
2178000 6 new_call
2182000 4 new_call
2183000 2 new_call
2201000 4 ped_call
2206000 4 new_call
2226000 6 new_call
2236000 6 new_call
2237000 6 new_call
2238000 6 new_call
2238000 6 new_call
2240000 6 new_call
2241000 6 new_call
2241000 6 new_call
2243000 6 new_call
2245000 6 new_call
2248000 2 new_call
2250000 8 new_call
2251000 8 ped_call
2254000 6 new_call
2258000 2 new_call
2259000 2 new_call
2260000 2 new_call
2261000 2 new_call
2261000 2 new_call
2262000 2 new_call
2262000 6 new_call
2265000 2 new_call
2267000 6 new_call
2276000 2 new_call
2280000 2 new_call
2282000 6 new_call
2283000 2 new_call
2293000 6 new_call
2296000 6 new_call
2301000 2 new_call
2301000 6 new_call
2302000 6 new_call
2304000 2 new_call
2304000 6 new_call
2305000 6 new_call
2306000 6 new_call
2307000 2 new_call
2308000 6 new_call
2308000 6 new_call
2310000 6 new_call
2313000 6 new_call
2314000 6 new_call
2318000 6 new_call
2320000 6 new_call
2333000 2 new_call
2333000 8 new_call
2336000 2 new_call
2336000 4 new_call
2336000 8 new_call
2337000 2 new_call
2340000 2 new_call
2342000 2 new_call
2343000 2 new_call
2344000 2 new_call
2344000 2 new_call

2345000 2 new_call
2347000 2 new_call
2348000 2 new_call
2353000 2 new_call
2359000 6 new_call
2360000 6 new_call
2360000 6 new_call
2360000 8 new_call
2360000 8 ped_call
2363000 2 new_call
2363000 6 new_call
2363000 6 new_call
2364000 2 new_call
2364000 2 new_call
2365000 6 new_call
2365000 6 new_call
2369000 6 new_call
2370000 2 new_call
2372000 6 new_call
2373000 6 new_call
2374000 2 new_call
2376000 6 new_call
2382000 2 new_call
2382000 2 new_call
2385000 2 new_call
2386000 6 new_call
2388000 6 new_call
2389000 2 new_call
2389000 6 new_call
2391000 2 new_call
2394000 2 new_call
2394000 2 new_call
2402000 6 new_call
2404000 6 new_call
2405000 6 new_call
2410000 2 new_call
2412000 8 new_call
2414000 2 new_call
2420000 2 new_call
2421000 2 new_call
2423000 2 new_call
2424000 2 new_call
2424000 6 new_call
2425000 8 new_call
2430000 2 new_call
2430000 2 new_call
2430000 6 new_call
2433000 6 new_call
2433000 6 new_call
2434000 2 new_call
2436000 6 new_call
2437000 2 new_call
2437000 8 ped_call
2438000 2 new_call
2438000 8 new_call
2441000 4 new_call
2449000 4 new_call
2449000 8 new_call
2453000 8 new_call
2454000 8 new_call
2461000 4 new_call
2463000 6 new_call
2469000 4 new_call
2472000 8 new_call
2473000 2 new_call
2474000 2 new_call
2475000 2 new_call
2476000 6 new_call
2477000 2 new_call
2477000 4 new_call
2479000 2 new_call
2480000 2 new_call
2480000 6 new_call
2481000 2 new_call
2482000 6 new_call
2483000 2 new_call
2484000 2 new_call
2484000 6 new_call
2486000 2 new_call
2486000 2 new_call
2489000 2 new_call
2492000 2 new_call
2492000 6 new_call
2496000 6 new_call
2498000 6 new_call
2499000 6 new_call
2501000 6 new_call
2501000 6 new_call
2504000 6 new_call
2505000 6 new_call
2506000 6 new_call
2508000 6 new_call
2511000 2 new_call
2511000 6 new_call
2513000 6 new_call
2514000 6 new_call
2516000 6 new_call
2518000 2 new_call
2518000 2 new_call
2518000 6 new_call
2520000 2 new_call
2520000 6 new_call
2521000 2 new_call
2522000 6 new_call
2527000 6 new_call
2528000 2 new_call
2532000 6 new_call
2535000 2 new_call
2539000 4 new_call
2548000 2 new_call
2550000 2 new_call
2555000 2 new_call
2556000 8 new_call
2556000 8 ped_call
2560000 4 new_call
2563000 2 new_call
2564000 2 new_call
2564000 6 new_call
2564000 6 new_call
2566000 6 new_call
2567000 6 new_call
2570000 6 new_call
2570000 6 new_call
2572000 2 new_call
2573000 6 new_call
2575000 6 new_call
2577000 6 new_call
2584000 6 new_call
2586000 6 new_call
2590000 4 new_call
2591000 6 new_call
2598000 2 new_call
2603000 2 new_call
2603000 8 new_call
2610000 2 new_call
2623000 2 new_call
2623000 2 new_call
2624000 2 new_call
2625000 2 new_call
2627000 6 new_call
2627000 6 new_call
2629000 2 new_call
2630000 2 new_call
2631000 6 new_call
2632000 2 new_call
2633000 2 new_call
2633000 6 new_call
2634000 2 new_call
2635000 2 new_call
2635000 2 new_call
2636000 6 new_call
2637000 2 new_call
2640000 2 new_call
2640000 6 new_call
2642000 2 new_call
2642000 2 new_call
2642000 6 new_call
2645000 6 new_call
2647000 6 new_call
2648000 6 new_call
2649000 2 new_call
2650000 6 new_call
2655000 6 new_call
2656000 2 new_call

2659000 2 new_call
2659000 4 new_call
2665000 4 new_call
2665000 8 new_call
2673000 8 new_call
2674000 8 new_call
2677000 8 new_call
2678000 4 new_call
2687000 8 new_call
2694000 6 new_call
2694000 6 new_call
2696000 6 new_call
2696000 8 new_call
2699000 6 new_call
2702000 6 new_call
2704000 6 new_call
2710000 6 new_call
2724000 4 new_call
2725000 8 new_call
2732000 2 new_call
2733000 2 new_call
2736000 2 new_call
2737000 2 new_call
2739000 2 new_call
2744000 8 new_call
2748000 6 new_call
2748000 6 new_call
2749000 2 new_call
2749000 6 new_call
2750000 6 new_call
2752000 6 new_call
2753000 2 new_call
2754000 6 new_call
2754000 8 new_call
2755000 6 new_call
2756000 2 new_call
2758000 6 new_call
2758000 6 new_call
2759000 6 new_call
2761000 6 new_call
2762000 4 new_call
2764000 6 new_call
2766000 6 new_call
2767000 8 new_call
2767000 8 ped_call
2768000 6 new_call
2773000 6 new_call
2774000 6 new_call
2791000 2 new_call
2792000 6 new_call
2792000 4 new_call
2796000 2 new_call
2797000 2 new_call
2801000 2 new_call
2802000 4 new_call
2807000 2 new_call
2807000 6 new_call
2807000 6 new_call
2813000 6 new_call
2815000 6 new_call
2816000 4 new_call
2816000 8 new_call
2817000 6 new_call
2823000 6 new_call
2824000 6 new_call
2825000 8 new_call
2826000 6 new_call
2833000 8 new_call
2844000 6 new_call
2855000 4 new_call
2863000 6 new_call
2876000 6 new_call
2877000 6 new_call
2879000 6 new_call
2880000 6 new_call
2881000 6 new_call
2884000 4 ped_call
2886000 8 new_call
2887000 6 new_call
2888000 6 new_call
2891000 6 new_call
2894000 6 new_call
2895000 6 new_call
2896000 2 new_call
2896000 6 new_call
2897000 6 new_call
2897000 6 new_call
2898000 2 new_call
2899000 2 new_call
2900000 2 new_call
2900000 6 new_call
2902000 2 new_call
2902000 2 new_call
2904000 6 new_call
2906000 2 new_call
2906000 2 new_call
2908000 2 new_call
2912000 6 new_call
2916000 6 new_call
2916000 6 new_call
2919000 6 new_call
2923000 6 new_call
2924000 6 new_call
2925000 4 new_call
2927000 2 new_call
2931000 2 new_call
2946000 2 new_call
2954000 6 new_call
2958000 6 new_call
2959000 6 new_call
2961000 6 new_call
2962000 6 new_call
2962000 6 new_call
2964000 6 new_call
2964000 6 new_call
2967000 4 new_call
2968000 6 new_call
2970000 6 new_call
2974000 2 new_call
2976000 8 new_call
2976000 4 ped_call
2989000 6 new_call
2989000 6 new_call
2991000 8 ped_call
2993000 6 new_call
2996000 8 new_call
3005000 6 new_call
3005000 6 new_call
3007000 6 new_call
3011000 6 new_call
3012000 6 new_call
3014000 6 new_call
3016000 2 new_call
3024000 6 new_call
3027000 2 new_call
3027000 6 new_call
3027000 6 new_call
3028000 2 new_call
3028000 6 new_call
3030000 2 new_call
3031000 2 new_call
3032000 2 new_call
3032000 6 new_call
3033000 8 new_call
3034000 6 new_call
3036000 2 new_call
3036000 6 new_call
3037000 2 new_call
3042000 6 new_call
3043000 8 ped_call
3044000 2 new_call
3044000 2 new_call
3045000 6 new_call
3046000 2 new_call
3047000 2 new_call
3047000 2 new_call
3047000 6 new_call
3050000 2 new_call
3051000 6 new_call
3052000 6 new_call
3053000 2 new_call
3055000 6 new_call
3055000 6 new_call
3057000 2 new_call

3058000 6 new_call
3060000 6 new_call
3061000 6 new_call
3061000 6 new_call
3062000 4 ped_call
3064000 6 new_call
3064000 6 new_call
3071000 2 new_call
3073000 2 new_call
3073000 8 new_call
3073000 8 ped_call
3075000 2 new_call
3076000 6 new_call
3082000 2 new_call
3083000 6 new_call
3084000 2 new_call
3089000 6 new_call
3111000 2 new_call
3111000 8 new_call
3113000 2 new_call
3116000 8 new_call
3118000 2 new_call
3124000 4 ped_call
3126000 8 new_call
3134000 8 new_call
3135000 6 new_call
3136000 6 new_call
3139000 6 new_call
3140000 6 new_call
3143000 2 new_call
3144000 6 new_call
3144000 6 new_call
3144000 6 new_call
3145000 6 new_call
3148000 2 new_call
3156000 2 new_call
3161000 2 new_call
3164000 2 new_call
3166000 2 new_call
3169000 8 ped_call
3170000 2 new_call
3171000 6 new_call
3171000 6 new_call
3174000 2 new_call
3174000 6 new_call
3174000 6 new_call
3175000 6 new_call
3176000 2 new_call
3176000 2 new_call
3177000 2 new_call
3181000 2 new_call
3181000 6 new_call
3182000 6 new_call
3184000 6 new_call
3184000 4 ped_call
3186000 6 new_call
3187000 2 new_call
3188000 2 new_call
3190000 6 new_call
3192000 2 new_call
3195000 2 new_call
3196000 8 new_call
3197000 6 new_call
3198000 6 new_call
3198000 6 new_call
3200000 6 new_call
3200000 6 new_call
3202000 6 new_call
3202000 6 new_call
3205000 6 new_call
3205000 6 new_call
3205000 4 ped_call
3206000 8 new_call
3209000 6 new_call
3210000 2 new_call
3217000 2 new_call
3219000 6 new_call
3223000 4 new_call
3225000 2 new_call
3226000 6 new_call
3227000 2 new_call
3227000 4 new_call
3227000 8 new_call
3229000 2 new_call
3231000 2 new_call
3244000 4 new_call
3244000 8 new_call
3250000 8 new_call
3250000 8 new_call
3256000 6 new_call
3257000 8 new_call
3264000 6 new_call
3265000 6 new_call
3267000 6 new_call
3268000 6 new_call
3272000 6 new_call
3272000 8 new_call
3274000 6 new_call
3276000 6 new_call
3278000 6 new_call
3280000 6 new_call
3280000 6 new_call
3284000 6 new_call
3285000 6 new_call
3287000 6 new_call
3288000 6 new_call
3290000 6 new_call
3291000 6 new_call
3291000 8 new_call
3294000 6 new_call
3296000 2 new_call
3296000 6 new_call
3297000 2 new_call
3298000 6 new_call
3299000 2 new_call
3299000 6 new_call
3300000 2 new_call
3301000 2 new_call
3302000 2 new_call
3302000 6 new_call
3303000 2 new_call
3307000 6 new_call
3308000 8 ped_call
3311000 6 new_call
3319000 6 new_call
3334000 6 new_call
3336000 4 new_call
3337000 6 new_call
3338000 6 new_call
3338000 4 new_call
3340000 6 new_call
3341000 4 new_call
3351000 6 new_call
3354000 6 new_call
3359000 6 new_call
3359000 6 new_call
3362000 6 new_call
3365000 2 new_call
3366000 6 new_call
3370000 6 new_call
3385000 8 new_call
3389000 6 new_call
3389000 4 new_call
3389000 8 new_call
3397000 4 new_call
3402000 6 new_call
3404000 6 new_call
3407000 4 new_call
3407000 8 new_call
3407000 6 ped_call
3411000 4 new_call
3412000 4 new_call
3412000 8 new_call
3413000 4 new_call
3425000 2 new_call
3425000 6 new_call
3426000 6 new_call
3427000 2 new_call
3428000 2 new_call
3428000 6 new_call
3429000 2 new_call
3430000 2 new_call
3430000 2 new_call
3431000 8 new_call

3434000 2 new_call
3434000 6 new_call
3435000 6 new_call
3437000 6 new_call
3440000 6 new_call
3444000 6 new_call
3445000 6 new_call
3445000 4 new_call
3447000 6 new_call
3448000 6 new_call
3450000 6 new_call
3450000 6 new_call
3451000 2 new_call
3454000 2 new_call
3455000 2 new_call
3456000 2 new_call
3458000 4 new_call
3458000 8 new_call
3459000 4 new_call
3462000 4 new_call
3463000 8 new_call
3464000 4 new_call
3467000 4 new_call
3468000 4 ped_call
3469000 4 new_call
3471000 8 new_call
3478000 2 new_call
3485000 6 new_call
3486000 2 new_call
3489000 6 new_call
3489000 6 new_call
3491000 6 new_call
3491000 6 new_call
3493000 8 ped_call
3496000 6 new_call
3511000 8 new_call
3519000 4 new_call
3524000 4 new_call
3527000 4 new_call
3529000 4 new_call
3532000 4 new_call
3535000 2 new_call
3547000 6 new_call
3550000 6 new_call
3550000 6 new_call
3551000 8 new_call
3553000 6 new_call
3553000 6 new_call
3554000 2 new_call
3556000 2 new_call
3556000 6 new_call
3556000 6 new_call
3557000 6 new_call
3558000 2 new_call
3558000 6 new_call
3558000 6 new_call
3559000 2 new_call
3560000 2 new_call
3560000 6 new_call
3561000 6 new_call
3562000 2 new_call
3564000 6 new_call
3566000 6 new_call
3567000 6 new_call
3569000 6 new_call
3569000 6 new_call
3571000 8 new_call
3572000 6 new_call
3572000 6 new_call
3574000 6 new_call
3574000 6 new_call
3576000 6 new_call
3578000 2 new_call
3580000 6 new_call
3581000 6 new_call
3582000 2 new_call
3584000 6 new_call
3585000 2 new_call
3585000 4 new_call
3589000 6 new_call
3590000 2 new_call
3590000 4 new_call
3592000 6 new_call
3592000 6 new_call
3593000 2 new_call
3595000 2 new_call
3602000 2 new_call
3603000 2 new_call

APPENDIX D: LIVE DATA CAPTURE

Lane 6 Time In-Queue (249 Vehicles)

2	21	31	13	20	11	2	3	14
2	25	30	4	18	18	33	1	28
12	23	21	4	7	21	31	1	17
22	21	19	2	2	24	29	30	15
22	20	19	2	20	30	6	34	5
16	37	19	17	21	30	5	36	16
13	36	17	29	22	27	4	35	12
14	30	15	27	16	13	3	32	10
22	22	13	32	14	13	33	32	6
14	14	10	32	8	12	33	32	4
2	19	5	31	4	2	33	32	3
2	13	29	31	32	2	18	32	3
2	13	25	19	32	1	35	27	3
24	28	14	19	32	4	33	10	2
24	6	35	18	32	37	31	5	1
25	16	35	13	6	36	20	25	1
31	12	34	13	20	20	18	13	30
19	6	34	12	18	18	30	9	30
20	6	6	8	14	5	23	32	29
17	4	33	3	8	31	20	32	26
15	3	33	33	29	28	20	32	23
25	3	31	22	29	27	14	32	2
26	21	26	21	29	21	14	32	22
22	21	21	16	29	14	9	32	13
20	15	17	30	32	12	8	32	2
20	11	29	29	32	26	6	27	
10	27	26	29	22	3	4	22	
22	34	14	21	12	3	4	15	

Lane 2 Time In-Queue (201 Vehicles)

1	29	23	5	33	18
2	25	15	4	22	6
4	29	9	3	19	15
5	10	17	2	15	11
6	6	17	5	3	10
8	30	17	13	2	10
28	17	14	13	37	5
31	2	13	14	30	4
14	24	12	15	29	3
2	15	10	17	28	1
20	17	6	19	19	20
19	16	34	19	9	32
18	15	33	18	1	31
13	5	33	11	13	32
11	3	32	11	31	25
7	22	26	10	32	21
7	21	19	10	25	19
6	17	34	4	23	17
5	17	33	12	19	13
4	21	31	9	20	16
3	3	2	9	17	10
5	20	4	7	9	
6	26	35	7	17	
9	32	17	6	17	
9	33	25	5	13	
13	23	26	3	11	
12	19	32	5	1	
24	12	7	31	26	
34	8	3	31	40	
20	5	2	30	32	
17	3	29	29	32	
13	36	23	28	30	
12	25	10	27	26	
10	31	8	26	10	
8	27	7	22	9	
31	27	5	34	5	

Lane 8 Time In-Queue (78 Vehicles)

17	10	33
53	8	9
28	34	1
27	17	37
9	4	28
16	29	26
11	5	13
3	36	32
6	33	
3	29	
5	23	
40	2	
26	51	
5	37	
8	34	
51	12	
35	3	
7	30	
30	52	
11	38	
9	76	
13	75	
10	62	
2	54	
25	43	
23	16	
9	8	
3	37	
3	25	
73	16	
72	7	
67	19	
45	30	
30	30	
20	38	

Lane 4 Time In-Queue (57 Vehicles)

6	13	21
5	9	19
6	9	17
5	7	14
4	5	7
6	7	24
4	7	15
16	6	13
15	5	54
24	5	37
23	19	39
24	17	27
13	45	25
12	43	33
73	47	10
65	42	10
43	76	3
15	11	7
24	6	7

REFERENCES

[1] Michael Meyer, “A Toolbox for Alleviating Traffic Congestion and Enhancing Mobility”, Publication IR-054B, Institute of Transportation Engineers, 1997.

[2] Daniel Fambro, et. al, “Benefits of the Texas Traffic Light Synchronization (TLS) Grant Program II”, Publication 3010-1F, Texas Transportation Institute, 1995.

[3] “Traffic Signals & Flashing Beacons”, Washington County Public Works, Transportation Division.

[4] Peter Koonce, et. al, “Traffic Signal Timing Manual”, Publication FHWA-HOP-08-024. FHWA, U.S. Department of Transpiration, 2008.

[5] J.M. Morales, “Improving Traffic Signal Operations: A Primer”, Federal Highway Administration, Washington, DC, 1995.

[6] K.J. Fehon, “Adaptive Signals—Are We Missing the Boat?”. Institute of Transportation Engineers District 6 Meeting, Sacramento, CA, 2004.

[7] M.C. Bell and R.D. Bretherton. “Ageing of Fixed-Time Traffic Signal Plans”, Institution of Electrical Engineers Second International Conference on Road Traffic Control, 1986, pp. 77-80.

[8] Robert Rausch and J.M. Cheeks, “The New York City, NY, USA, Traffic Signal System: A Standards-Based Approach”, *Institute of Transportation Engineers Journal*, Vol. 75, No. 3, 2005.

[9] Erin Ehlinger and PB Farradyne. “Successful Traffic Signal System Procurement Techniques”, Publication FHWA-OP-02-032. FHWA, U.S. Department of Transportation, 2002.

[10] “Attorney General Lockyer Files Taxpayer-Protection Antitrust Suit Against Traffic Signal Firm”, State of California Department of Justice, Office of the Attorney General, California. http://ag.ca.gov/newsalerts/release.php?id=643. Accessed June 5, 2010.

[11] “Antitrust Highlights”, State of California Department of Justice, Office of the Attorney General, California. http://ag.ca.gov/antitrust/highlights.php**.** Accessed January 15, 2011.

[12] W. Reed. “Safety Critical Software in Traffic Control Systems”, *IEE Colloquium on Safety Critical Software in Vehicle and Traffic Control,* 1990, pp. 2/1-2/5.

[13] R.K. Scott, J.W. Gault, and D.F. Mcallister. “Fault-tolerant software reliability modeling”, *IEEE Transactions on Software Engineering*, Vol. SE-13, No. 5, 1987, pp. 582-592.

[14] John Knight. “Safety critical systems: challenges and directions”, 24th International Conference on Software Engineering, Orlando, Florida, 2002.

[15] A. Avizienis. “Fault-Tolerance and Fault- Intolerance: Complementary Approaches to Reliable Computing”, International Conference on Reliable Software, Los Angeles, California, 1975 pp. 458-454.

[16] Limng Chen and A. Avizienis, “N-Version Programming: A Fault-Tolerance Approach to Reliability of Software Operation.” Twenty-Fifth International Symposium on Fault-Tolerant Computing, Vol., Iss., 27-30 Jun, 1995, pp.113.

[17] Liming Chen; Avizienis, A., Fault-Tolerant Computing, 1995, ' Highlights from Twenty-Five Years'., Twenty-Fifth International Symposium on, Vol., Iss., 27-30 Jun 1995, Page:113.

[18] Avizienis. *The Methodology of N-version Programming,* John Wiley & Sons, Inc., 1995.

[19] Orcutt, F.L. *The Traffic Signal Book*. Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[20] Koonce, P., Rodergerdts, L., Lee, K., Signal Timing Manual. FHWA-HOP-08-024, Federal Highway Administration, Washington, DC. 2008.

[21] Ibid “National Traffic Signal Report Card.”

[22] Henry, RD, “Signal Timing on a Shoestring”, Federal Highway Administration, FHWA-HOP-07-006, March 2005.

[23] “National Transportation Communications for ITS Protocol” January 2005. http:/ntcip.org, Accessed July 8, 2011.

[24] “NEMA TS2 Standards: The “Intelligent Cabinet”. Econolite. Anaheim. California. http://www.econolite.com/assets/pdf/nemats2.pdf. Accessed July May 15, 2011.

[25] Mn/DOT traffic Signal Timing and Coordination Manual, Mn/DOT Office of Traffic Engineering and Intelligent Transportation Systems, June 2002.

[26] “Intellidrive: The Benefits of Infrastructure to Vehicle Information Transfer”. Siemens. http://www.itssiemens.com/en/t_nav141.html. Accessed August 5, 2011.

[27] “ITE Advanced Transportation Controller Family of Standard”, Research and Innovative Technology Administration (RITA), U.S. Department of Transportation, 2006.

[28] S. Yamada, T. Ichimori and M. Nishiwaki, *Optimal allocation policies for testing-resource based on a software reliability growth model*, Mathematical and Computer Modeling, Vol. 22, 1995, pp. 295-301.

[29] R. Roess, E. Prassas and W. McShane, *Traffic Engineering – Third Edition*. Prentice Hall, Englewood Cliffs, New Jersey, 2004.

[30] Iies, B. “Light problem addressed: Police give green light on shorter traffic signals”. Central Florida Future. http://www.centralfloridafuture.com/mobile/light-problem-addressed-1.2141722. Accessed October 15, 2012.

[31]