

M. Eng Thesis Proposal

**TSAFE**  
**Building a Trusted Computing Base**  
**for Air Traffic Control Software**

by  
Gregory Dennis  
Software Design Group

Submitted to the  
Department of Electrical Engineering and Computer Science  
May 14, 2002

Supervisor  
Daniel Jackson

**Abstract.** This paper examines TSAFE, a proposed new decision support tool for air traffic controllers. Air traffic control software systems are highly complex pieces of machinery. The various heuristics and intricate algorithms such systems employ usually bring them out of reach of standard verification and modeling techniques. By abstracting away and simplifying many of these intricacies, I begin to investigate how one could build a Trusted Computing Base for TSAFE, upon which further layers of complexity may later be added. I describe my work on a TSAFE prototype and propose future thesis research on this topic.

## **Introduction**

Air traffic control software systems are highly complex systems of machinery. They are typically ridden with complex heuristics and intricate algorithms, many of which are developed in a very ad hoc manner. As a result, they are generally large and messy and intractable in terms of analysis and verification. In my thesis work, I plan to steer TSAFE in the opposite direction. That is, I hope to turn it into a small, trusted, verifiable component.

In this paper, I recount the research of TSAFE that I and others have conducted to date. Much of this work was done in conjunction with the NASA Ames Research Center and the MIT International Center for Air Transportation. I begin by giving a brief overview of particular air traffic control terms and concepts that one should be familiar with before reading the rest of this paper. I describe the motivation for building a tool like TSAFE and then detail each of its components. I then discuss our efforts to more fully understand TSAFE through problem analysis. The paper continues by detailing our ongoing work on a TSAFE prototype and suggests avenues for further research.

## **1. Overview of Air-Traffic Control**

Before further describing the details of TSAFE, there are a number of terms and concepts the reader should become familiarized with first.

**Fix.** A fix is a two-dimensional geographic position on the earth's surface.

**The Flight Plan.** Commercial flights fly according to predefined flight routes. Each route consists of a sequence of fixes, which when connected, form a two-dimensional path from the point of departure to the point of arrival. The flight route is issued to a flight prior to take off as part of the flight plan. The plan also includes additional flight information, such as the cruise altitude, cruise speed, and the aircraft type and equipment. Even though the flight route is part of the flight plan, the two terms are often used interchangeably.

**Trajectory.** A trajectory is four-dimensional path (latitude, longitude, altitude, and time components) along which an aircraft may fly. The flight route is not classified as a trajectory because it does not have a time dimension.

**Conflict.** A conflict is a predicted violation of separation, an instance in which two flights will come within a minimum regulatory distance from one another.

**Controller.** The essential job of an air traffic controller is to prevent conflicts from occurring. More generally, they are responsible for maintaining safe and efficient flow of aircrafts.

## **2. Overview of TSAFE**

The Tactical Separation Assisted Flight Environment, or TSAFE, is a proposed new decision support tool for air traffic controllers. Unlike CTAS, which performs long-term conflict prediction on the order of 20-40 minutes, and unlike TCAS, which provides last minute avoidance maneuvers for collisions seconds away, TSAFE is intended to detect conflicts that are generally between 3 and 7 minutes in the future. It was initially proposed by Dr. Heinz Erzberger, Senior Scientist for Air Traffic Management at the NASA Ames Research Center.

### **2.1 Motivation**

In today's Air Traffic Control system, it is the air traffic controllers that are primarily responsible for maintaining aircraft separation. Controllers accomplish this by surveilling radar data for potential conflicts and issuing clearances to pilots to alter their trajectories accordingly. Ground-based automated tools play only a supportive role in this process [1].

Under this current system, the airspace within the United States operates at only half its potential capacity. Experience has shown the controllers' workload limits to be the fundamental limiting factor in the current model. Despite attempts at resectorization—dividing the airspace into smaller sectors to disperse controller workload—and development of enhanced ATC decision support tools, researchers have failed to circumvent these fundamental workload limits. Erzberger claims that exploiting the full airspace capacity requires a new paradigm. He refers to this new paradigm as the *Automated Airspace Concept* (AAC) [1].

Under the AAC framework, automated mechanisms would play a primary role in maintaining aircraft separation. Aircraft would remain in direct connection with a ground-based automated system, which would transmit conflict alerts and air traffic control clearances via a persistent data link. (The concept includes a fail-safe mode in which controller-pilot communication takes over in emergencies and other special circumstances.) By shifting much of the responsibility of aircraft separation from controllers to automated systems, AAC should significantly increase airspace capacity [1].

TSAFE is not intended to be a realization of the Automated Airspace Concept. Rather, it is meant to be a last line of defense for inevitable imperfections in the AAC model. That is the reason it is important for TSAFE to be designed as a Trusted Computing Base—a highly dependable piece of software. Erzberger's hope is that the development of TSAFE will bring automated tools to the forefront of air traffic control operations, and thereby bring us one step closer to fulfilling the Automated Airspace Concept [1].

### **2.2 TSAFE Components**

TSAFE consists of five main components: (1) a Conformance Monitor, (2) a Trajectory Synthesizer, (3) a Conflict Detector, (4) a Conflict Avoidance Module, and (5) a Controller Interface.

### **2.2.1 Conformance Monitor**

Conformance monitoring is the process of determining the degree to which a flight is adhering to a prescribed path. In other words, conformance monitoring is asking the question “Is a flight where it’s supposed to be?” If the answer is no, the flight is often said to be *not conforming*, or *blundering*.

Conformance monitoring typically involves comparing a flight’s current state against some *conformance basis*. In simpler models, the conformance basis may just be the flight plan. More sophisticated models will allow for several conformance bases at once. In such a system, if a flight is found to be not conforming to a particular basis, the Conformance Monitor may dynamically hypothesize a new conformance basis and perform a new comparison. The new conformance basis may account for possible controller clearances and directive that were given, but never actually entered into the ATC system.

TSAFE falls somewhere in between these two approaches. It uses the flight plan, flight plan amendments, cruising altitude, controller clearances, and other intent information to generate a 3-Dimensional volume in which the aircraft should ideally be located. We have given this volume the name *planned path*. The job of TSAFE’s Conformance Monitor is to compare an aircraft’s current position and heading against this planned path.

For each flight, the Conformance Monitor calculates both a horizontal and vertical conformance status. There are eight separate horizontal and eight separate vertical conformance statuses, each indicating a different degree of flight conformance. As a result, TSAFE may place a flight into one of sixty-four different conformance status combinations at any given time.

### **2.2.2 Trajectory Synthesizer**

TSAFE requires access to two different types of trajectory predictions, a *nominal trajectory* and a *dead reckoning trajectory*. A nominal trajectory assumes the flight will eventually rejoin its planned path, regardless of its current state. The dead reckoning trajectory, on the other hand, assumes the flight will stay “on course.” This may simply mean that the flight retains its current velocity and heading, though the exact meaning may differ between implementations.

### **2.2.3 Conflict Detector**

The Conflict Detector, or Conflict Probe, will then probe along one or both of the nominal and dead reckoning trajectories, searching for points at which two flights break legal separation. Precisely which trajectories are used will vary according to the conformance statuses of the flights in question. The Detector will provide timely and reliable warnings to controllers should any imminent loss of separation be detected.

### **2.2.4 Conflict Avoidance Module**

In response to high-risk conflict warnings, the Conflict Avoidance Module calculates appropriate avoidance maneuvers. The maneuver is not intended to deliver a long-term conflict-free trajectory, but rather a brief controller clearance that should steer the aircraft out of danger for about three minutes. Nevertheless,

the modules would need to factor in nearly all flight state, flight intent, weather, airspace geometry, and relevant geography data available in order to make this calculation.

### **2.2.5 Controller Interface**

The avoidance maneuvers and conflict warnings are relayed to the controller in the form of visual and aural signals via the Controller Interface. Later enhancements to the interface may offer voice recognition technology so as to limit the number of keyboard commands the controller must execute.

## **3. Problem Analysis**

We began our analysis of TSAFE by constructing diagrams and models that helped us visualize how TSAFE operates. These diagrams enabled us to develop a deeper understanding of how TSAFE's components interact and brought to light a number of questions we had yet to answer.

### **3.1 Information Flow Diagram**

We began by constructing an **Information Flow Diagram**, so as to give us a better grasp on how data flows between modules in TSAFE. This diagram can be found in Appendix A.

The flow of data can be thought to begin on the right-hand side of the diagram at the *Static and Dynamic Databases*. The Static Database contains information we expect not to change throughout TSAFE's operation. As the diagram illustrates, this includes the locations of navaids and fixes as well as aircraft performance models and data regarding airspace geometry. The information in the dynamic database, on the other hand, is expected to change rapidly. This database would store information such as to the positions and velocities of flights, flight plans and routes, and weather data.

The *Planned Path Synthesizer* uses the aircraft state, intent, and performance models, along with weather information provided by the databases to construct a three-dimensional volume of the flight's planned trajectory, or *planned path*. The *Conformance Monitor* compares the flight's actual state information to the planned path to determine to what degree, if any, the flight is deviating from where it should be. This deviation is then classified into one of many different conformance statuses, which is then outputted by the Conformance Monitor.

Based on the given conformance status, the *Probing Strategy Function* chooses a conflict probing strategy that should be associated with the flight. This can include the types of trajectories to probe along—nominal, dead reckoning, or both—the look ahead times to use along each trajectory, as well as whether or not to conduct a vertical airspace search. The *Conflict Probe* follows the probing strategy dictated by the Probing Strategy Function, invoking the *Nominal* and *Dead Reckoning Trajectory Synthesizers* to generate the nominal and dead reckoning trajectories as needed. The Trajectory Synthesizers draw information from the Databases for their calculations.

Once the conflicts have been determined, TSAFE must notify the controllers of them and possibly suggest avoidance maneuvers. If the conflict is low-risk, i.e. it is not scheduled to occur for some sufficiently long period of time, the conflict is sent only to the *Alerting Mechanism*, which generates an appropriate level of alert and notifies the controller via the *Controller Interface*.

If the conflict is high risk, it is sent to both to the Alerting Mechanism and the *Conflict Avoidance Module*. The Conflict Avoidance Module must pull nearly all available information from the databases and from previous calculations in order to calculate and appropriate avoidance maneuver. Again, the Alerting Mechanism generates an alert with a level corresponding to the severity of the conflict. Both the avoidance maneuver and the alert are relayed to the controller via the Controller Interface.

### **3.2 Computation Graph**

The Information Flow Diagram forced us to precisely enumerate the various components TSAFE, but we felt it too strongly implied a specific architecture. The diagram also brings the modules to the forefront, when it was the data that is needed and the composite structures that must be calculated that we were primarily interested in. We concluded that creating the dual graph of the IF Diagram—which would put the data objects (positions, flight plans, conflicts, etc) at the nodes and move the computations (trajectory synthesis, conformance monitoring, etc) to the edges—would serve as a better aid in analyzing the nature of TSAFE.

This decision prompted the creation of the TSAFE Computation Graph, which can be found in Appendix B. This graph is essentially the dual graph of the Information Flow Diagram, with improvements. It links each data component to the data components that are necessary for its derivation. Fittingly, it shows the *conflict* to be the final result, or purpose, of all prior derivations, and shows all the intermediary computations TSAFE must perform along the way. (At this point in our study of TSAFE, we had decided to simplify the problem and not look at conflict avoidance or alerting, so they are not included in this diagram).

### **3.3 Abstract Alloy Model**

Clearly, much effort has been invested into understanding the workings of TSAFE. But how do we know that at a fundamental level, the design of TSAFE even allows for accurate conflict prediction? The logic inherent in TSAFE's structure may in fact prohibit this.

This is what I intended to investigate in building an Alloy model of TSAFE. The model I built was highly abstract and placed the computational complexities of trajectory prediction and conflict detection into a black box. The model boiled the problem domain down to just flights, trajectories, locations, and points of time, and sought to investigate one hypothesis:

*All other things being equal, if flights are following the same trajectories in a given time instance that they were following in the previous instance, the predicted conflicts in both instances should be the same.*

The Alloy Analyzer could find no counterexamples to this assertion up to a scope of four, a sufficient scope it seemed for this model. This gave me confidence that at least at theoretic level, TSAFE should work. The full Alloy model can be found in Appendix C.

## **Chapter 4: Prototype Development**

At this point in our work with TSAFE, we felt we had exhausted the potential for the type of high-level problem analysis we had been conducting. There were only so many issues we could flush out at the black-box level before engaging in some sort of implementation. So we decided to build a toy model of TSAFE—a prototype—that we would use to explore this tool’s potential. We set forth to investigate how one could enforce a uniform set of information safety policies on TSAFE and ensure the correctness of its internal mechanisms, i.e. how one could make TSAFE a Trusted Computing Base (TCB).

### **4.1 ETMS Data**

One of the initial hurdles we had to surmount in the beginning stages of development was how to obtain access to up-to-date information on flight positions, flight plans, flight plan amendments, etc. across the U.S. We gained access to this data by connecting to an Enhanced Traffic Management System (ETMS) feed. ETMS is used by the FAA to “to predict, on national and local scales, traffic surges, gaps, and volume based on current and anticipated airborne aircraft” [2]. It is not typically used for conflict detection and resolution purposes because it obtains updated information at a relatively slow rate (at approximately 15 minute intervals); however it has so far proven adequate for our purposes.

### **4.2 Our Simplifications**

Before beginning the prototype, we made a series of deliberate simplifications and approximations of the initial proposed design. First off, we decided to focus our attention on a small geographical area of the United States. At the time of this writing, our attention is focused on the airspace above the state of Massachusetts. This allows us to filter out most ETMS flight position and fix/navaid/airport information. The aim is to test the prototype a small scale before setting our sights on larger geographical areas.

In addition, our prototype does not include the Conflict Avoidance Module. Conflict resolution involves very complicated heuristics, and we have eliminated it to give ourselves a more manageable problem with which to begin our prototype research. As will be discussed, we also made some significant simplifications to the original TSAFE methods for conformance monitoring and trajectory synthesis.

Our principle simplification with regard to conformance monitoring was to use flight plans and their amendments as the only conformance bases for flights. This differs from Erzberger’s initial design in which he proposes continually-mutable conformance bases. Under his proposal, if a flight is deemed to not be conforming to the issued flight plan, TSAFE could dynamically hypothesize a new conformance basis against which the flight could be compared.

We do not allow for this dynamic derivation of alternate conformance bases. In our model, the operation of the Conformance Monitor is reduced to performing a static comparison of a flight's position and heading against its flight plan. Note that this means the conformance monitor does not make use of any trajectories, and therefore has no dependencies on the Trajectory Synthesizer. In addition, we do not provide for a multitude of conformance statuses. Instead, our conformance status is just boolean: either the flight is conforming or it is blundering.

We have also simplified and modified conflict detection from TSAFE's proposed strategy. Unlike TSAFE's approach, if our prototype deems a flight to be conforming, we will not look along either the dead reckoning or the nominal trajectory. Instead, we will probe a trajectory that runs along the flight's planned path. We will refer to this as the *planned trajectory*. If a flight is blundering, we will use both the planned trajectory and the dead reckoning trajectory.

Note that this greatly simplifies trajectory synthesis by eliminating the notion of a nominal trajectory, which would have otherwise required us to implement intensely complex calculations. We make a further simplification to trajectory synthesis by ignoring vertical intent information.

### **4.3 Desired Features**

There are a number of specific features we want the prototype to include, most of which, at the time of this writing, have already been implemented. First off, we want the ability to visually display flights, fixes, blunders, routes, trajectories, and conflicts. That does mean to imply we are expecting to implement a highly sophisticated trajectory synthesizer or conflict probe, but simply that we just wish to display the trajectories we have synthesized and the conflicts we have detected regardless of their accuracy.

We would also like our prototype to be generalized enough to fit a number of different conformance monitoring, trajectory synthesis, and conflict prediction schemes. Indeed, a prototype architecture that is not adaptable to other calculation schemes will not be useful for making a claim about air traffic control systems generally. Thus, our prototype should allow us to "plug-in" different conformance monitoring, trajectory synthesis, and conflict prediction modules without altering its overall architecture.

We also plan to include record and playback capabilities. A user should be able to press a record button, and thereby trigger all future incoming messages to be written to a log. The user should then be able to stop recording, restore the state of the airspace at the time recording began, and playback all recorded events. This is an essential feature to incorporate because it allows ATC controllers to review the events leading up to near misses. It will also prove very helpful in debugging our system.

### **5. Future Thesis Work**

The principal goal of this project is to investigate how we can achieve all the desired features listed in 4.3 while fulfilling the requirement of ultra-reliability. In

essence, we must continue to investigate how to develop a Trusted Computing Base for TSAFE and decide how do we know when we have one.

There are a number of techniques we can use to approach this problem. One strategy would be to check our code against our proposed design. We could develop models of our code and attempt to analyze these models against certain design constraints. Alloy would prove useful to us in this regard.

We could also attempt to show that our design meets the requirements of the problem domain. Using ‘Problem Frames’ techniques we could come to a clearer understanding of the requirements of our tool at the intersection with the problem domain [3]. By building models of our design and of the requirements, we can attempt to demonstrate that the former implies the latter.

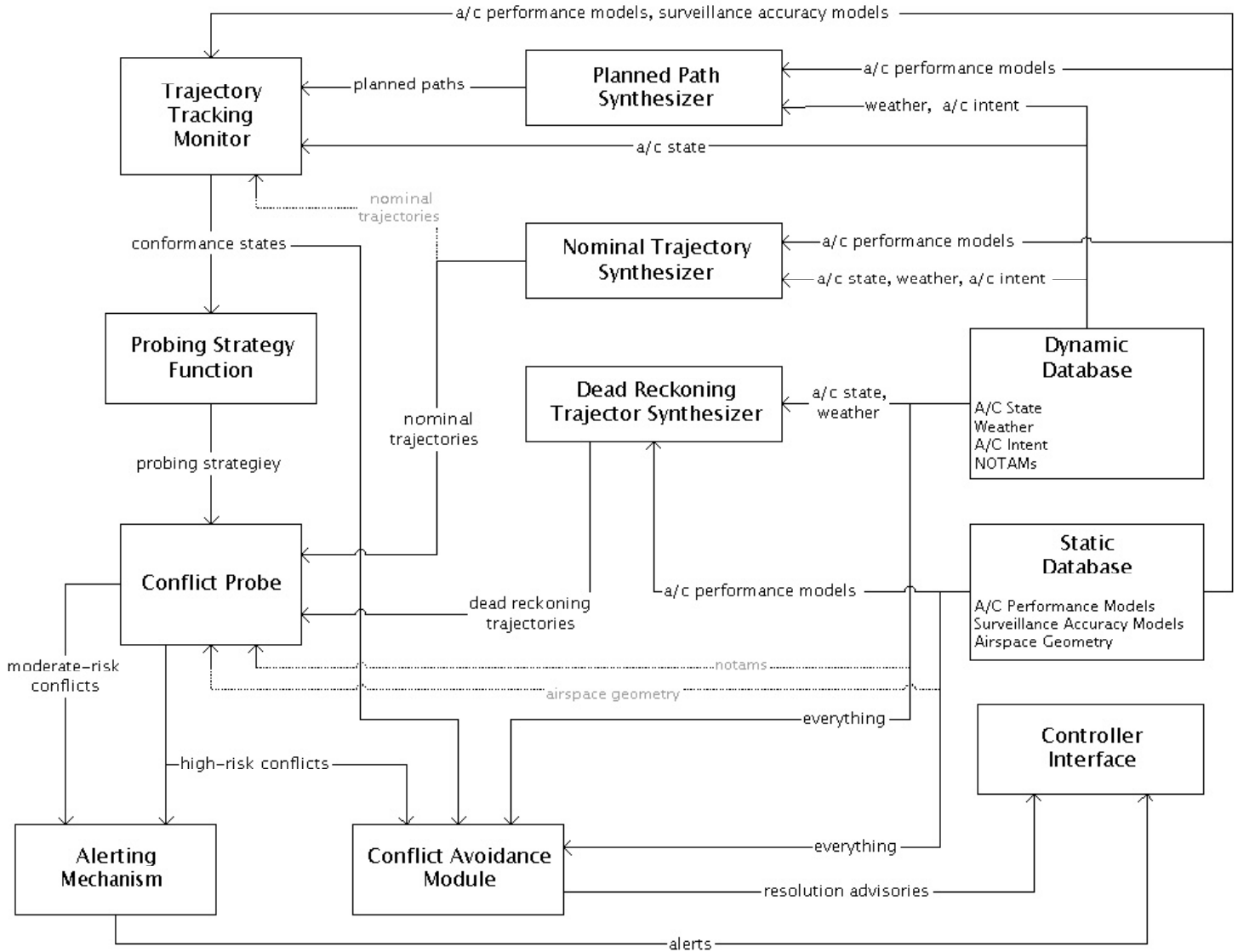
We could also try to expand the scope of our model in various directions. For instance, we can make attempts to expand the geographic scope to cover a larger land area. We can plug-in various trajectory synthesizers and conflict probes and compare their performances.

In short, the future of this project is still, to a great extent, open-ended, but will take on a more concrete form as the work on it intensifies over the summer and the following fall.

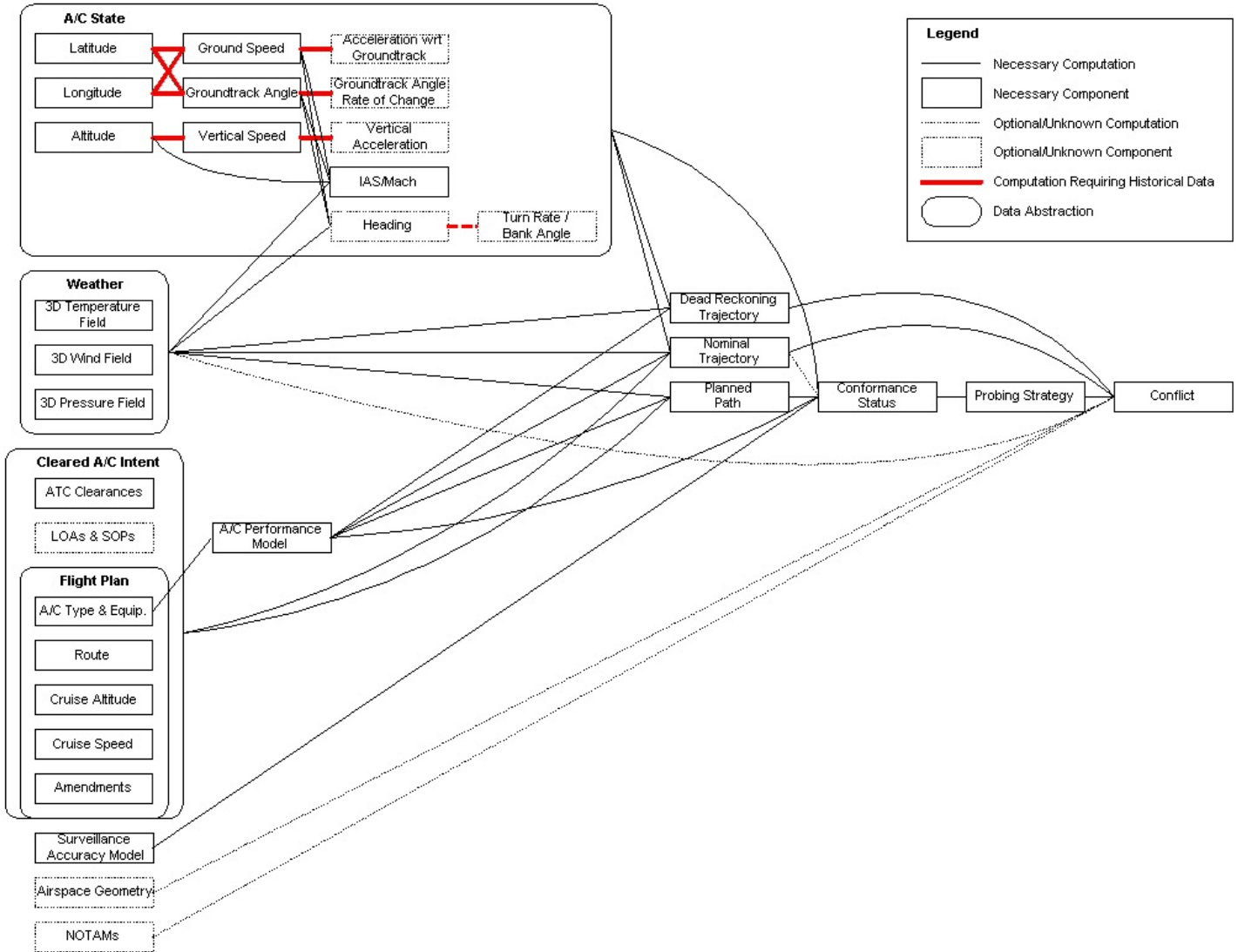
## References

- [1] H. Erzberger, “Next Generation ATC System: The Automated Airspace Concept,” July 10, 2001.
- [2] “Enhanced Traffic Management System (ETMS),” Available at HTTP: <http://www.fly.faa.gov/Information/ETMS/etms.html>
- [1] M. Jackson, Problem Frames: Analyzing and Structuring Software Development Problems, Addison-Wesley, 2001.

# Appendix A: Information Flow Diagram



# Appendix C: Computation Graph



## **Appendix C: Alloy Model**

Will be here soon