Abstract—The major focus of the U.S. Next Generation Air Transportation System (NextGen) is maintaining safety while increasing air transport capacity [1]. Safe and efficient operations on the airport surface are crucial, since airports are a primary means of access to the national air space. In particular, new concepts for real-time conflict detection and resolution (CD&R) systems for surface movement are needed to mitigate collision risk as airport traffic density increases. Assessing alternative CD&R concepts is challenging, however, because no standardized methods exist for comparing early-stage conceptual designs for cyber-physical systems, a term which refers broadly to technologies like CD&R that feature spatially distributed but tightly integrated hardware and software components. In this paper we introduce a method to compare conceptual designs for cyber-physical systems. In particular we introduce Information Flow Diagram (IFD) analysis as a means to model coupled system faults, which together might result in unacceptable safety risk. An IFD analysis is used to compare four proposed CD&R architectures and to identify which architectures successfully mitigate all fault sequences considered.

Index Terms—Conflict Detection and Resolution, Cyber-physical System, Fault Analysis
I. Introduction

As the population size increases and more people travel by air, it is important that safety of the national air transportation system is maintained at current levels. A number of initiatives have been enacted in order to address safety and performance of the national air transportation system under an increased throughput. The most notable U.S. initiative is NextGen, which involves many aspects of air travel including but not limited to: reconsideration of aircraft travel routes, incorporation of new navigation and surveillance technologies, and new methods for detecting and resolving potential conflicts before they occur [1].

Conflict detection and resolution (CD&R) is an essential aspect to ensuring aircraft safety during all phases of a flight, including during surface movement. In this paper, we investigate the problem of concept design for a surface CD&R system. We will refer to various designs for CD&R systems as architectures, each which consists of human operators, software algorithms, sensors, communication links, and decision protocols that support the CD&R process.

Analyzing surface CD&R architectures is challenging, in part because they involve distributed human controllers and automated algorithms operating on aircraft and on the ground [2, 3]. Few tools exist for analyzing and comparing such systems, which are more broadly classified as cyber-physical systems (CPS). In general terms, a CPS is a system comprised of both computational and physical elements whose actions and outputs are closely coupled when trying to achieve system level goals [4].
A straightforward method for comparing alternative early-stage CD&R designs is needed to identify which architectures are most promising. Standard conceptual design tools [5] are not well suited to quantifying the safety of a complex CPS, such as a surface CD&R architecture. It is nontrivial, for instance, to determine how interactions between human and software components might lead to a fault. To help fill this gap, this paper introduces a modeling approach we call the *Information Flow Diagram* and applies it to evaluate and compare four candidate surface CD&R architectures.

The remainder of this paper is organized as follows. First, the components of a surface CD&R system are summarized and grouped to provide a tractable model describing how information flows among those components. A risk analysis approach is then introduced, which combines model reduction with expert systems analysis to identify weaknesses (and possible mitigations) for candidate CD&R architectures. Next, four specific surface CD&R architectures are introduced and analyzed. Finally, a brief discussion indicates that only one of the architectures meets all critical safety requirements.

**II. Information Flow Diagram Analysis of CD&R Architectures**

In this section, we introduce the Information Flow Diagram (IFD) as a model reduction tool that supports evaluation of conceptual design options for a new CPS. The central idea of an IFD analysis is to trace how information moves through a CPS and how it impacts sequential decisions or actions. In this analysis, *information bottlenecks* are identified as communication links that compress or delay available information such that not all relevant data available at the
input of the communication process can be reconstructed at the output. CPS model simplification is achieved by clustering together system components where they are not specifically separated by an information bottleneck.

A. Agents

System components or subsystems for which communication does not impact performance are defined as agents. By this definition, agents may be individual humans, machines or teams of humans and/or machines.

Four types of agents are considered in modeling integrated surface CD&R architectures. These agents include: pilots (P), air traffic control ground or local controllers (ATC), air-side automated systems (ASA), and ground-side automated systems (GSA), as summarized in Fig. 1. Agents make decisions and perform actions, which together impact the safety of aircraft, crew, and passengers.

Fig. 1. Conflict Resolution Agents
B. Layers

To minimize complexity, we model the CD&R process as a series of sequential events, and group these events into *layers*, which are sets of events among which information flow is not restricted (i.e. no information bottleneck). The sequential nature of events (and layers) means that information always flows in one direction, from high-level sources (e.g. sensors) down to the agents who control individual aircraft (e.g., pilots or possibly ASA).

In the analysis of surface CD&R, we have identified three primary information bottlenecks that divide the integrated CD&R system into a series of four layers: (1) *sensing*, (2) *resolution specification*, (3) *decision* and (4) *execution*. These four layers are described in the following paragraphs. A list of the actions associated with each layer is given in Table I.

<table>
<thead>
<tr>
<th><strong>TABLE I</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>OPERATIONS ASSOCIATED WITH EACH LAYER</td>
</tr>
<tr>
<td><strong>Sensing Layer</strong></td>
</tr>
<tr>
<td>Collection of raw data (e.g. aircraft positions and velocities)</td>
</tr>
<tr>
<td>Sensor data communication (e.g. transmission of raw data)</td>
</tr>
<tr>
<td><strong>Resolution layer</strong></td>
</tr>
<tr>
<td>Data association (e.g. matching radar returns to aircraft)</td>
</tr>
<tr>
<td>Sensor fusion (e.g. combining redundant data to maximize accuracy)</td>
</tr>
<tr>
<td>Trajectory identification (e.g. compiling data from many time points into a “track”)</td>
</tr>
<tr>
<td>Conformance monitoring (e.g. detecting when an aircraft track strays from a flight plan)</td>
</tr>
</tbody>
</table>
Conflict detection (e.g. detecting when an aircraft collision is imminent)
Resolution generation (e.g. generating a plan to mitigate a conflict)
Resolution negotiation (e.g. discussion among agents that influences resolution generation)
Resolution translation (e.g. conversion of resolution into a form interpretable by other agents)
Resolution communication (e.g. transmission of resolutions from each resolution provider)

**Decision Layer**

- Selection of the “best” resolution (when multiple options are available)
- OR fusion of multiple resolutions to construct a single self-consistent resolution
- Resolution translation (e.g. conversion to a form interpretable by other agents)
- Resolution communication (e.g. transmission of a single resolution to individual aircraft)

**Execution layer**

- Implementation of resolution action (e.g. pilot actions to avoid conflict)

1) Sensing Layer

The first layer of the CD&R architecture consists of a suite of sensors used for navigation and surveillance. Historically, navigation sensors have been classified as those used to guide and control aircraft, while surveillance sensors have been used by air traffic controllers to detect aircraft nonconformance. The historical definition of sensor groupings are blurred by modern communication capabilities, which can convert off-board surveillance data into navigation data or on-board navigation data into surveillance data (e.g. GPS data are used for surveillance in Automatic Surveillance Dependant-Broadcast, or ADS-B) [6].

For this document we have grouped sensors based on how they are communicated rather than on how they are used. Consequently, we have three distinct sensor groups: (1) navigation sensors,
which are located on one aircraft and possibly communicated to the tower and other aircraft, (2) surveillance sensors, which are ground-based sensors, possibly communicated to aircraft, and (3) vision sensors, which are human visual observations, possibly communicated via verbal descriptions [6-9]. Navigation data consist of raw state information provided by aircraft-based sensors such as: positions, headings, and velocities obtained from GPS fused with inertial measurements. Navigation data are assumed to be communicated through ADS-B to all aircraft as well as to the tower [6]. Surveillance data consist of multilateration, primary radar, and secondary radar measurements. Surveillance data are assumed to be communicated by Traffic Information Services-Broadcast (TIS-B) [10]. Vision data consist of visual observations made by the air traffic controller or pilots. Vision data are assumed to be communicated via voice.

2) Resolution Specification Layer

The next layer in the CD&R architecture is the resolution specification layer. In the resolution specification layer, sensor data are processed in order to detect potential conflicts and to generate conflict resolutions when necessary. In general, agents may be empowered to generate different resolutions, each of which may involve one or more aircraft. For instance, the pilot and ATC may have different ideas about how to resolve a potential conflict, and the ATC may provide a resolution for all aircraft while the pilot provides a resolution for only his/her own aircraft.

In some cases, it may be important to model conflict resolution plans that evolve over time as communication makes additional information available. For instance, a pilot might formulate and announce an initial resolution plan to a ground controller but later revise that plan based on
verbal communication with the controller. To model such scenarios, we make a distinction between two types of plans: fixed plans and evolving plans. Fixed conflict resolution plans are issued once. Evolving conflict resolution plans can be updated in a sequential nature as more information becomes available. Most typically, evolving plans are generated by a human agent (pilot or ATC) based on shared verbal communication with other human agents. Whereas fixed plans are always delivered within a specific duration of time, evolving resolutions are delivered continuously.

3) Decision Layer

The next layer in the CD&R architecture is the decision layer. In the decision layer it is assumed that an agent (or consensus of agents) is granted authority to decide on a single resolution to implement. In this sense, the decision layer compresses information; the role of the agent (or consensus of agents) identified as the decider is to select or construct a single resolution (or no resolution) when multiple resolutions are given. Generally, the decider will simply select one of the plans from the set provided by all resolution specifiers. In some cases, however, a decider may be allowed to construct a fused resolution that draws together elements suggested by more than one specifier. In constructing a fused resolution plan, the onus is on the decider to ensure that the fused plan is self-consistent. Additionally, the decider may always elect to ignore a resolution (if for instance, that resolution appears to be a false alarm). Thus, if only one resolution plan is given to the decider, the decider must choose whether or not to pass that resolution plan to the next layer. By extension, if no resolution actions are provided to the decider, then the decision layer can take no further action.
4) Execution Layer

The final layer in the CD&R architecture is the execution layer. The execution layer models aircraft physical motion that might occur either during nominal operations or during an emergency response to a conflict. In a broad sense, the execution layer encompasses all aircraft interacting on (or near) the airport surface. In order to represent typical fault scenarios, however, only a small number of aircraft need be considered (i.e., those directly involved in a potential conflict). The agents controlling each aircraft (labeled *executers*) are either pilots or air-side automation (ASA) agents. Executers are expected to implement the resolution plans selected by the decider unless those plans are deemed unsafe.

C. Branch-Risk Analysis

An important function of IFD analysis is to abstract choices made in conceptual design such that a rapid but meaningful analysis of their impact on system safety can be conducted. In order to compare alternatives, we use IFDs to quantify the ability of various CD&R architectures to mitigate fault-scenario risk.

1) IFD Branch Structure

Fault scenarios are investigated using IFD *branches*, which are ordered sequences of failure events occurring at one or more layers. The sequences of failure events are coupled together through the passing of information (correct or corrupted) from layer to layer through the IFD
information bottlenecks. Consequently, event sequencing is constrained to follow the IFD description of Table I. Failure probabilities are assigned to each agent on each layer (each sensor, resolution specifier, decider, and executer) when information is corrupted or prevented from arriving at the next layer. A risk analysis that considers all possible unique branches will be called a *Branch-Risk Analysis*. A branch-risk analysis relies on human experts to identify detailed worst-case scenarios associated with each branch.

2) Branch-Risk Probability Assignment

In order to assess overall branch-risk, we assign risk probabilities to each fault event in the IFD. For the purpose of this paper, probabilities for each event (or *node*) on a branch are assigned in a qualitative sense, depending on whether the event is *extremely probable, probable, improbable,* or *extremely improbable*. These four qualitative assessments are subsequently quantified using a process defined by in the FAA Safety Handbook [11]. The mapping from qualitative description to quantitative risk assessment is summarized in Table II.

<table>
<thead>
<tr>
<th></th>
<th>Extremely Probable</th>
<th>Probable</th>
<th>Improbable</th>
<th>Extremely Improbable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability</td>
<td>$10^{-1}$</td>
<td>$10^{-3}$</td>
<td>$10^{-6}$</td>
<td>$10^{-9}$</td>
</tr>
</tbody>
</table>

To determine the probability of each collision event (associated with a particular branch) all of the conditional event probabilities on the branch are multiplied.
\[ P(\text{Branch}) = P(\text{Event}_1)P(\text{Event}_2 \mid \text{Event}_1) P(\text{Event}_3 \mid \text{Event}_1, \text{Event}_2) \]
\[ \quad \cdots P(\text{Event}_n \mid \text{Event}_1, \ldots, \text{Event}_{n-1}) \]  

(1)

Here \(P(\text{Branch})\) indicates the risk for a branch consisting of an ordered set of events \(i\) (numbered 1 through \(n\)), each with an event probability \(P(\text{Event}_i)\). In general, an event may be conditioned on the previous event, but also on any or all prior events.

At each node in the IFD, success requires that an agent follow the rules provided to it, complete its actions in a timely manner, and successfully transmit data to agents at the next layer. In other words, it is assumed that a fault only occurs when an agent produces corrupt data, given a healthy data input. The process of defining success and failure based on action and not outcome avoids double counting of failure probabilities. For branch-risk analysis a probability of 1 is assigned to successes, since most often actions are assumed successful. This means successful events need not be considered in assigning fault probabilities to a branch sequence. Consequently, in naming branches we identify only the failures that occur along the branch. For example, a \(R_{P1}\)-D-x branch refers to a case in which a failed resolution provided by Pilot 1 (\(R_{P1}\)) and a subsequent decision failure (D) contribute to collision (x). The overall collision risk for this branch would be the product of the conditional probabilities for these three failure events, multiplied by a prior (discussed in the following section).

3) Prior Conflict Probability
In order to evaluate the probability of collision for each branch, it is important to consider that imminent conflicts are, in fact, rare. As such, a prior probability is associated with each branch. We assume that general conflict-imminent situations are *probable* according to FAA definitions (see Table II). However, some conflict situations are riskier than others. In this work, we consider two particularly severe but rare (i.e. *improbable*) conflict situations: the high kinetic energy scenario and the bad-weather scenario.

The first scenario of interest is that in which the conflict is not observable until one or both aircraft are already moving at high speed. We refer to this scenario as high kinetic energy or *High KE*. Such an event might be the result of a runway incursion, such as when one aircraft proceeds unexpectedly past the hold short line (without clearance) and moves onto an active runway on which another aircraft is taking off or landing. Were it to occur, a high KE scenario would result in a number of correlated system faults. Importantly, in the high KE scenario, significant communication latencies for ADS-B and TIS-B may mean that available data are old, pre-dating the occurrence of the runway incursion such that conflict detection is not possible using those data. As such, the high KE scenario results in likely sensor faults.

The second scenario of interest is that in which the conflict occurs during low visibility conditions caused by bad weather. We refer to these operating conditions as a *bad-weather* scenario. Multiple vision sensing faults are likely to occur in this bad-weather scenario (e.g. because of fog). Also, if the pilot’s vision is poor, it is assumed that the pilot is more likely to behave erratically, resulting in an elevated probability of a pilot execution error.
4) Expert Systems Analysis

Ultimately, domain experts convert abstract branch models into detailed fault scenarios and assign fault probabilities based on those scenarios. In this regard, the IFD analysis provides two benefits. First, consolidating a multitude of potential system faults into only four broad categories (sensing, resolution specification, decision, and execution) reduces the combinatorial explosion that would otherwise occur in more detailed fault modeling [12]. Second, the fact that the number of unique branches is small allows human experts to discuss every branch.

In branch analysis, experts first brainstorm detailed system faults (hardware, software, communication, human, etc.) that might explain each branch. Next, experts identify the most likely of these fault mechanisms and assign associated event probabilities. This expert systems analysis is systematic, in the sense that every branch is discussed; however, the analysis is still open ended, in the sense that experts brainstorm worst-case sequences of system faults. This creative, brainstorming activity is important for identifying subtle, correlated fault sequences, which might be missed were fault probabilities assigned in a more automated fashion. The outcome of the IFD analysis is an identification of the worst-case fault sequence (or sequences) for each branch.

In order to define fault probabilities, experts must have a clear definition of the term *fault*. For CD&R analysis, a layer-by-layer definition is provided in Table III.
### TABLE III
BRANCH-RISK ANALYSIS LAYER ASSUMPTIONS

<table>
<thead>
<tr>
<th>Sensing Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success:</strong> Sensors are successful when measurements are generated (inclusive of nominal errors) and passed in a timely manner to intended recipients on the resolution layer.</td>
</tr>
<tr>
<td><strong>Fault:</strong> A sensor fails if it generates data corrupted by:</td>
</tr>
<tr>
<td>• An abnormally large error, which might result in a false conflict detection, a missed conflict detection, or an incorrect conflict resolution</td>
</tr>
<tr>
<td>• Latency associated with communication delays (e.g., via ADS-B, TIS-B, CPDLC, etc.), which might cause data to arrive too late for a resolution action to be initiated</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resolution Specification Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success:</strong> A successful resolution occurs when an agent correctly processes data, as in the following three cases.</td>
</tr>
<tr>
<td>• Correct Detection &amp; Resolution: Agent correctly performs conflict detection and formulates a resolution plan that can avert the conflict, given prompt action by pilots</td>
</tr>
<tr>
<td>• Fault Mitigation: As above, but in the case of one or more corrupted sensor inputs; (note, in this paper, it is assumed inconsistent sensor inputs are resolved by majority vote)</td>
</tr>
<tr>
<td>• Corrupt Resolution due to Bad Data: Resolution procedure correctly applied to corrupt inputs, resulting in a corrupt output; (note, this case not counted as fault to avoid double counting)</td>
</tr>
<tr>
<td><strong>Fault:</strong> A faulted resolution occurs when corrupt output data are produced from good input data, for example, as caused by a software problem (failure to converge or to detect a “corner case”) or communication problem.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decision Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success:</strong> A successful decision action occurs when decision agents consider possible resolutions and arrive at consensus pilot actions to avert a collision. Types of successful decision actions include the following cases.</td>
</tr>
<tr>
<td>• Consistent Decision: Pilots and controllers agree to implement a consistent set of resolution actions</td>
</tr>
<tr>
<td>• Fault Mitigation: A consistent decision (as described in the previous bullet) results because the negotiation process rejects corrupt resolutions that appear to be outliers</td>
</tr>
<tr>
<td>• Corrupt Decision due to Many Corrupt Resolutions: Given too many corrupt resolutions (e.g. more than half), it is assumed that the decision process may also be corrupted</td>
</tr>
<tr>
<td><strong>Fault:</strong> A decision failure generally indicates that human agents did not arrive at a consensus decision or the negotiation begins with safe resolution plans but drifts to an unsafe decision.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Execution Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success:</strong> Each pilot executes the consensus resolution unless the pilot feels the consensus resolution is unsafe. Successful execution actions include the following three cases.</td>
</tr>
<tr>
<td>• Conflict Resolved: Pilot implements correct resolution and avoids collision</td>
</tr>
<tr>
<td>• Fault Mitigation: Pilots individually select their resolution strategies given a corrupt decision</td>
</tr>
<tr>
<td>• Corrupt Execution due to Corrupt Decision: If the decision is corrupt, and if pilots heed that decision</td>
</tr>
<tr>
<td><strong>Fault:</strong> Execution faults might occur, for instance, due to a lapse in attention by the pilot or due to a mechanical failure.</td>
</tr>
</tbody>
</table>
III. Architectures of Interest

Using IFDs and Branch-risk Analysis, we will compare alternative CD&R architectures of interest. The architectures investigated will be referred to as “Baseline”, “Option A”, “Option B”, and “Option C”. A detailed description of these four architectures has been reported previously [13]; as such, this summary provides only a brief overview of each architecture.

In constructing IFDs for each alternative architecture, we consider, in this paper, only the case involving conflicts between two aircraft, whose pilots are designated (P1, P2). Each aircraft is assumed to be equipped with an air side automated CD&R system (ASA1, ASA2). Conflict resolution is conducted by pilots and air side automation in coordination with an air traffic controller (ATC) and a ground side automated CD&R system (GSA). The relationships among these agents will first be described for a baseline system, and subsequently for the three alternative options.

A. IFD of Baseline Architecture

Whereas current day CD&R is performed entirely by human agents (Pilots, ATC), the Baseline architecture seeks to maintain safety in denser traffic patterns by introducing automation, both on the aircraft and in the control tower. The relationship of human and automated agents in the Baseline architecture is represented compactly as an IFD in Fig. 2 and described in more detail, on a layer-by-layer basis, in the paragraphs below.
1) Baseline Sensing Layer

The Baseline architecture assumes that all three classes of sensor are available: navigation, surveillance and vision [14]. ADS-B and TIS-B, drawn as horizontal bars in Fig. 2, serve as communication links to distribute Navigation data (Nav) and Surveillance data (Sur), respectively. Latency for ADS-B transmissions can be up to 2 seconds, and latencies corresponding to TIS-B transmissions can be up to 3.25 seconds [10, 15]. Because verbal communication of visual data is imprecise, it is assumed that visual information is not broadcast.

2) Baseline Resolution Layer

At the resolution layer human and automated agents process sensor data to detect conflicts and generate resolution actions. The potential providers of resolutions include all automated systems (ASA1, ASA2, and GSA), as well as all human operators (Pilot1, Pilot2, and ATC). Automated systems generate optimized plans based on set protocols, while human operators offer
resolutions based on training and past experience. Thus, each human operator might propose a different resolution given the same sensory information. In either case, we make no assumptions about compatibility of resolutions provided by different agents (whether human or automated), even when resolutions are generated from the same sensor data.

3) Baseline Decision Layer
The decision layer takes into account that the resolution actions provided by each human or automated system may be inconsistent; hence a reconciliation process is needed to ensure that pilots act in a consistent manner. We assume the reconciliation is conducted by human agents, who attempt to reach a consensus decision via voice communications. It is expected that a set of human operators typically requires at least 9 seconds to converge on a decision (using a median response time of three seconds [16] for each of three human operators). If a consensus decision is not reached by the time pilots must take action, then pilot actions may be inconsistent, leading to an elevated risk of collision.

During consensus building, it is assumed that human agents create evolving resolutions (that may change over time), but that the automation generates fixed resolutions (that do not change once published). We assume that the initial human resolution is generated at the same time that automated systems publish their resolution actions. Automated resolutions are not necessarily shared with other human agents; however, human agents have an option of sharing automation-generated resolutions via voice communication.

4) Baseline Execution Layer
The execution layer concerns the implementation of individual resolution plans by all pilots. Ideally, these plans are consistent. If a consensus decision is available, it is assumed that pilots will generally act upon this consensus. Whether or not a consensus decision is reached, the pilot has ultimate authority and responsibility to implement a safe resolution action. Thus, a pilot may opt to disregard a partially or fully negotiated resolution if he/she deems it unsafe.

B. Alternative Architectures

Three option architectures add capability to the Baseline Architecture.

1) IFD of Option A

Option A introduces rapid reconciliation among automation systems (ASA, GSA) to produce a common, consistent resolution plan. In this way the air side and ground side automated systems can be thought of as a single fused automation system. Ideally, a reconciled automation resolution, consistent for all air side and ground side automated systems, could be displayed to the human operators at the same time they are forming their own conflict resolutions.

The primary difference between the Baseline IFD and the IFD for Option A is that, in Option A, all the automated systems are clustered together at the resolution layer, in essence acting together as a single agent. An IFD for Option A can be seen in Fig. 3.
Option A offers two distinct advantages over the Baseline architecture. First, automation reconciliation eliminates inconsistencies in resolutions produced by disparate automation systems. Second, automation reconciliation provides a common ground for discussion and negotiation by pilots and the air traffic controller during the decision process. A common ground likely speeds up verbal negotiations to reach a consensus.

2) IFD for Option B

Option B is the same as the Baseline case except intent information is available to the pilots and the ATC. Intent information is assumed to be clearance information or possibly a limited trajectory description sent over Controller Pilot Data Link Communications (CPDLC) channels.
Intent information can be thought of as an additional sensor technology allowing pilots and the controller to more effectively predict farther into the future.

The primary difference between the Baseline IFD and the IFD for Option B is that, in Option B, intent information is added to the sensor layer and distributed (via a CPDLC data bus) to resolution-generating agents. The IFD for Option B can be seen in Fig. 4.

There are two distinct advantages to adding intent information to the Baseline architecture. First, intent information allows for more accurate predictions of aircraft trajectories and thus enables more sensitive and accurate conflict detection, with reduced latency. Intent information effectively makes it less likely that dangerous runway incursions occur (such as when aircraft find themselves on a collision course at high speed). Second, intent serves as additional
information that can be used to resolve sensor ambiguities. For example, if two sensors disagree, then intent data can act as a “tie breaker” to determine which sensor is more likely to be correct.

3) IFD for Option C

Option C combines Option A and Option B, thereby providing both automated reconciliation and intent information. The information flow diagram for Option C can be seen in Fig 5. Option C offers the advantages of both Option A and Option B, as described in the preceding sections.

![Fig. 5. IFD for Option C](image)

IV. Architecture Comparison

Architectures were analyzed by applying IFD modeling and quantifying risks associated with each IFD branch (i.e., each unique combination of layer faults). In this study, IFD-based analysis
was conducted by two domain experts, a faculty member and a Ph.D. student at Tufts University. A full description of the IFD analysis, on a branch-by-branch basis, can be found in [13].

Key results of the complete analysis are tabulated in Table IV, below. Each column of the table corresponds to an alternative CD&R system architecture (Baseline, Option A, Option B, or Option C). The table presents the worst-case risk over four groups of branches, where each group consists of branches whose first fault occurs at a particular layer. For example, the upper left box indicates that the highest risk for any branch of the Baseline architecture for which the first fault occurs at the sensor layer; this worst-case risk was identified to be $10^{-9}$ per hour. (All risk probabilities are listed on a per hour basis.)

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Option A</th>
<th>Option B</th>
<th>Option C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor</td>
<td>$10^{-9}$</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
<td>$10^{-12}$</td>
</tr>
<tr>
<td>Resolution Specification</td>
<td>$10^{-13}$</td>
<td>$10^{-13}$</td>
<td>$10^{-13}$</td>
<td>$10^{-13}$</td>
</tr>
<tr>
<td>Decision</td>
<td>$10^{-9}$</td>
<td>$10^{-9}$</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
</tr>
<tr>
<td>Execution</td>
<td>$10^{-11}$</td>
<td>$10^{-11}$</td>
<td>$10^{-11}$</td>
<td>$10^{-11}$</td>
</tr>
</tbody>
</table>

To ensure system safety, the FAA requires that catastrophic events be improbable, meaning that they occur with a probability less than $10^{-9}$ [11]. All branch probability estimates are intended to be accurate plus or minus an order of magnitude; hence, any branch with a risk of $10^{-9}$ or greater represents a potential concern for future system certification. As such, all risks of $10^{-9}$ or greater
are shaded in Table IV. Of the four architectures considered, only Option C mitigates risk on all branches.

It is instructive to consider the most threatening branches identified by the IFD analysis in more detail. For the Baseline architecture, three branches resulted in risk probabilities of $10^{-9}$. (Risks for all other Baseline architecture branches were lower.) These three highest risk branches are summarized in Table V. These include two branches with their first fault at the sensor level and one branch with its first fault at the decision level. In the latter case (labeled D-x), the expert system analysis identified two worst-case fault sequences of equal risk; these scenarios are labeled Case 1 and Case 2 in Table V.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Option A</th>
<th>Option B</th>
<th>Option C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nav-Nav-Sur-x</td>
<td>$10^{-9}$</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
<td>$10^{-12}$</td>
</tr>
<tr>
<td>Nav-Nav-Sur-D-x</td>
<td>$10^{-9}$</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
<td>$10^{-12}$</td>
</tr>
<tr>
<td>D-x (Case 1)</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
</tr>
<tr>
<td>(Case 2)</td>
<td>$10^{-9}$</td>
<td>$10^{-9}$</td>
<td>$10^{-12}$</td>
<td>$10^{-12}$</td>
</tr>
</tbody>
</table>

*High Risk Branches Involving Sensor Faults:*

The highest-risk branches with a fault at the sensor layer (Nav-Nav-Sur-x and Nav-Nav-Sur-D-x) are both high KE scenarios. When a conflict involves aircraft moving at high speed, pilot action may be required immediately (within a matter of seconds). In this timeframe, sensor latencies
become critical. Sensor latencies of 2-3 seconds (as allowed for ADS-B and TIS-B) are likely to result in conflict detection and resolution based on aged, and therefore misleading, surveillance data. Given that latencies are likely to be present at all times, even under nominal conditions, we assume latency faults are essentially deterministic (probability 1) for all sensor data sent through an ADS-B or TIS-B communications link. Thus, no agent has access to up-to-date ADS-B data from both conflicted aircraft (“Nav-Nav” fault). Likewise, no flight deck has access to up-to-date TIS-B data (“Sur” fault). However, timely surveillance data are assumed to be available to the tower, since these data are not transmitted via TIS-B messaging.

As a result of sensor latency faults during high KE scenarios, all but one resolution is corrupted. ASA resolutions receive two faulted sensor inputs (Nav and Sur) out of two. Pilots receive two faulted sensor inputs (Nav and Sur) out of three. GSA receives one faulted sensor input (Nav) out of two. In all these cases, at least half of sensor inputs are corrupt, so the resulting resolution is also considered to be corrupt. The only exception is ATC, who generates a healthy resolution after receiving only one faulted sensor input (Nav) out of three. Because all but one resolution are assumed corrupt, the decision process is either corrupt or fails to converge. In the former case (Nav-Nav-Sur-x), the decision is corrupt because fewer than half its input resolutions are healthy. In the latter case (Nav-Nav-Sur-D-x), it is assumed that no consensus is reached. The no consensus case is highly probable given the time pressure of the high KE scenario. For the Baseline architecture both branches have a collision risk probability of $10^{-9}$, reflecting a probability of $10^{-6}$ of a high KE fault and a $10^{-3}$ risk of a collision given a corrupt or faulted decision.
The risk of these fault sequences is mitigated by the introduction of intent information, which reduces the probability of a high KE scenario to only $10^{-9}$, and the overall branch risk to only $10^{-12}$. Thus, these sensor-layer fault sequences do not threaten the Option B or Option C architectures, both of which benefit from intent information.

*High Risk Branch Involving Only a Decision Fault:*

The other high-risk branch involves only a decision fault (D-x). There are two predominant fault sequences associated with this branch. The details of the fault sequence depend upon the operating conditions when a conflict occurs (nominal or high KE), and are equally likely. For this reason, both fault sequences must be considered. The fault sequences (with probabilities given in parentheses) are the following. The Case 1 fault sequence occurs under nominal operating conditions ($10^{-3}$), when it is probable that a consensus decision will not converge ($10^{-3}$) leading to probable collision ($10^{-3}$). The Case 2 sequence occurs during a high KE scenario ($10^{-6}$), when it is virtually deterministic that no consensus decision will be reached ($10^{0}$) making a conflict probable ($10^{-3}$).

Different mitigations are required for both fault sequences. Automated reconciliation resolves the Case 1 fault sequence by providing common ground to help human agents arrive more quickly at a consensus decision for nominal conflict situations, reducing the risk of a decision fault ($10^{-6}$) and the overall Case 1 branch risk ($10^{-12}$). In a high KE scenario (such as Case 2), automated reconciliation does not significantly reduce the risks of a decision fault. However, intent information does reduce the risk of entering a high KE conflict ($10^{-9}$), and so the overall Case 2 collision risk is also reduced ($10^{-12}$). Only the Option C architecture, which features both
automated reconciliation and intent broadcast, resolves both the Case 1 and Case 2 fault scenarios, and thus only this architecture reduces all D-x branch risks to $10^{-12}$.

V. Conclusion

A complex cyber-physical system was modeled and analyzed using Information Flow Diagrams (IFDs). Specifically, IFD Analysis was introduced to model four alternative conflict detection and resolution architectures for airport surface movement. IFD analysis demonstrated that both automated reconciliation and intent information have positive effects on reducing collision risk probabilities. Additionally, of the four architectures considered, only Option C (featuring both automated reconciliation and intent information) resolves all major fault conditions.
REFERENCES


