Verbal Communication Protocols in Safety-Critical System Operations

Peter Bernard Ladkin
University of Bielefeld CITEC and Causalis Limited

For the Handbook of Technical Communication, ed. Gibbon & Mehler

Background: Systems, Safety, Agents, Joint Behavior

A "system" is best defined as a collection of objects that exhibit joint behavior. An object which exhibits behavior is often known as an agent; so a system is a collection of agents. Engineering systems are designed by humans to fulfil a particular purpose, to achieve a set of goals, thus they may be deemed "teleological". Besides those other agents in the system with which an agent might interact, system agents interact with – that is, exhibit joint behavior with - objects not in the system. These jointly-interacting but non-system agents constitute the "environment" of the system.

Consider a desktop or laptop computer. Such a system consists of agents – chips and other electronic components sitting on a motherboard, screen and keyboard, connections to other devices known as "peripherals". A user can be considered to belong to the environment, through which interaction takes place over very narrow channels: keyboard input on the one hand and screen output on the other. Such interaction is not necessarily best analysed or understood using linguistic constructs. Neither is desktop computer use in its computational function per se either safety-critical or safety-related.

However, there are systems consisting of discrete agents in which joint behavior involves explicit communication, and in which this communication may have implications for safety. Consider a highway system, with a junction uncontrolled by STOP signs or traffic lights. You drive up to the junction and stop; to your right, another vehicle drives up and also stops. The driver gestures with her hand towards you, and moves her hand across the windshield in your direction of travel. You take her to be waving you across, and accelerate. She was, however, attempting to indicate to you the traffic jam on the other side, and that you cannot therefore proceed, expecting you to conclude that she, however, can. She also accelerates; you both collide. The interaction, though performed by means of gesture, admits analysis in linguistic terms because, well, that is how it was just explained what the miscommunication consisted in!

A road system is only partially engineered. Some teleological transport systems are more thoroughly engineered, down to the communication amongst agents, for example railways and air traffic. Both involve safety-critical behavior, as does road transport. We shall consider some cases in which technical, partially- or fully-controlled communication amongst system agents and the environment plays a fundamental role critical for safety.

What does "safety-critical" mean? The concept "safety" is defined in a number of ways in the engineering literature when talking about the safety of systems. One definition is: freedom from accidents [Leveson 1995], where an accident is defined as unwanted but not necessarily unexpected behavior of some kind. Another definition, in the international standard governing the functional safety of systems involving programmable-electronic components: freedom from unacceptable risk [IEC 61508]. Risk is defined here in the usual way as the combination of the probability of harm occurring and the severity of that harm, where harm is considered to be physical injury or damage to the health of people, or damage to property or the environment.

Much behavior which has consequences which may be critical to safety is regulated within a system by human agents – for example, a policeman directing traffic at a busy intersection. Other examples: air traffic control is intended to ensure the physical separation of participating aircraft; train dispatching protocols coordinate the behavior of trains running in opposing directions, or different-speed trains running in the same direction, on single-track lines.

Traditionally, these protocols are purely verbal, but increasing use is made of automated helper systems, such as the ASCII-text CDPLC system for relaying ATC clearances and acknowledgements between computers on board and in air traffic control centers, with initiation and checking performed by humans, or the computer-based protocol-aiding systems on board regional dispatched trains on Austrian railways [Stadlmann 2004]. Such protocols are synchronous, in the sense that both initiating agents are interacting in real time, whether the communication is mediated by digital-electronic devices or not.

Not all communications involving human agents take place between two human agents, however. Central monitoring system messages in complex modern aircraft determine the status of various critical systems and relay changes in status to an electronic bulletin board which the pilots read. These communications are different from the verbal-exchange protocols in that one communication channel, from the central monitoring system to the pilot's screen, is designed asynchronously, including its execution. The execution is controlled by synchronous events, but the system adapts according to algorithms which have been designed in the engineering workshop largely before the aircraft was built, and certainly before it flew its present trip.

We consider some incidents in which these communications went wrong, and how they may be repaired. Potential or actual solutions to the problems exhibited are often not primarily linguistic but involve technical measures to which linguistic features contribute.

The verbal protocols used in air traffic control are crucial to the safety of air travel, a major mode of transport used by increasingly many. Thus arises the question why more attention has not been paid by researchers and engineers to this domain. Here are some possible reasons.

First comes a social reason. The skills of safety-critical system analysis and development consist in hazard analysis, safety requirements derivation, risk analysis, risk avoidance and mitigation; understanding and knowing how to use the basic concepts and techniques. To the author's knowledge, there are few linguists, with the possible exception of Cushing (below), who possess these skills to the necessary degree. Similarly, computational-linguistic skills, skills in formal semantics and syntax, and in discourse analysis and the engineering of controlled languages, are generally not to be found amongst safety-critical system engineers.

Second, the verbal protocols in, say, air traffic control work nowadays very well. Major accidents in which air traffic control behavior has played a role most often involve passivity – a controller could possibly have intervened in anomalous aircraft behavior, but somehow did not. Such situations are not primarily linguistic and so are of less interest here. In the first half of 2010, major accidents disproportionately involved a loss of control or of situational awareness, and non involved difficulties with verbal crew-ATC protocols – see for example [Ladkin 20100903].

There are two notable commercial air accidents in the past 15 years in which aspects of the verbal ATC-flight crew protocols played a causal role. We analyse the linguistic aspects of both these situations.

The other set of verbal protocols considered are those for train dispatching on non-signalled single-

track lines. Such lines are relatively rare in Europe (although not in, say, Australia, where such lines may be hundreds of miles long, and exhibit quite different phenomena), and relatively sparsely used (for otherwise they would be signalled). So the system is not as pervasive as that for air traffic. The protocols are simpler, less involved, and arguably more robust. These characteristics enable technical solutions to reliability and accuracy problems, as will be shown, which solutions could be considered exemplars for how one may go about the task of increasing reliability, accuracy and safety in the air-traffic environment.

Case Study: The TCAS Collision Avoidance System in the Überlingen Midair Collision, 2002

On 1 July, 2002, a Tupolev 154M commercial jet transport aircraft, operated by Bashkirian Airlines (BTC), a Russian airline, was flying westwards at night over Southern Germany towards a destination in Catalunya. A Boeing 757 operated by the cargo airline DHL was flying northbound over Switzerland, at the same Flight Level 360 (representing a nominal altitude of 36,000 feet in a "standard" atmosphere). Both were operating under Instrument Flight Rules (IFR), compulsory at this Flight Level. Skyguide, the Swiss air traffic control organisation, had control of both aircraft in its Zürich center, and accordingly had responsibility for separation of the aircraft.

The controller on duty was operating two positions, some meters apart, because colleagues were on break. He was working primarily with other traffic at one position, and only noted the convergence of the two aircraft close to the limit of the separation he was required to ensure (7 nautical miles lateral and/or 1,000 ft vertical separation). Another air traffic control facility at Karlsruhe had noticed the convergence, but was unable to contact Zürich Center through the dedicated communication channel, which was undergoing maintenance. An automatic ``early warning'' system installed at Zürich Center was also undergoing maintenance and did not trigger.

The controller issued a verbal avoidance manoeuvre to BTC to descend immediately. However, both aircraft received a Resolution Advisory from their on-board Airborne Collision Avoidance System (ACAS) devices, both TCAS II Version 7.0 from the company ACSS, virtually simultaneously with this instruction. The avoidance manoeuvres advised by TCAS are strictly vertical: one aircraft is advised to climb, and the other to descend. The rates at which these manoeuvres are to be accomplished are also normed in the TCAS protocol: a smooth 1/4g acceleration to a climb, resp. descent rate of 1,500 feet per minute is to be performed. TCAS advised to BTC an immediate climb, and to DHL an immediate descent

DHL descended. The BTC commander also instructed his Pilot Flying (PF) to descend. 7 seconds later, the air traffic controller repeated his descend instruction to BTC with an note to ``expedite'', for traffic which he mistakenly described as at the ``two o'clock'' relative position. (The direction of flight is "twelve o'clock"; two o'clock is two segments to the right, so two-twelfths of 360°, 60° right of BTC's direction of flight.) BTC was in fact at ``two o'clock'' to DHL; DHL was correspondingly at ``ten o'clock'' to BTC. This was a cognitive slip by the controller. Such slips are not uncommon, and normally inconsequential. In this case, however, it caused the BTC commander to believe he was in a three-aircraft conflict, with DHL, whose lights the crew could see and had identified at their ten o'clock position, and with an unknown aircraft which his TCAS display was not ``seeing'', at his two o'clock position [BFU 2004]. (Ladkin had speculated
that this might have been so already in [Ladkin 20020812]. The present analysis is based heavily on the final paper [Ladkin 2004].)

7 seconds later, DHL received an ``iterated advisory'' to ``increase descent'' (to a normed rate of 2,500 fpm). 9 seconds after that, DHL informed air traffic control that he was in a ``TCAS descent''.

Air traffic control procedures are such that the controller is no longer responsible for separating

traffic responding to TCAS Resolution Advisories until it is reported to himher by the participants that they are ``Clear of Conflict''. However, a controller may continue to provide information to participants during the manoeuvres. The air traffic controller conformed with this procedure.

11 seconds after DHL informed the controller of the TCAS descent, the two aircraft collided.

TCAS is a system which of which the agents are a visual and aural display, along with “advisories”, issued according to a programmed algorithm. The agents which activate the avoidance manoeuvre are the human pilots of the involved aircraft. The communication between TCAS agent and human is visual, and aural (involving commands expressed as English sentences). What of communication between ATC and TCAS or ATC and aircraft crew? The TCAS “philosophy” is that TCAS only triggers an RA when ATC separation has “failed”; it is an on-demand system implementing a function based on the hypothesis that ATC has failed to fulfil its function. Hence TCAS is considered as a system, by some only involving the two programmed boxes and associated electronics, but this is analytically inadequate because the crucial joint behavior, the avoidance manoeuvre is initiated and performed by the human crews of the aircraft; the TCAS boxes issue “advisories”. So TCAS boxes plus flight crew of each aircraft belong to the “system”. ATC is not considered to be part of the system: TCAS is predicated on failure of ATC function and one wants to remedy this failure through TCAS, not incorporate the failure into the system itself.

However, this part of the TCAS “philosophy” (i.e. modus operandi as construed by the designers) is also inadequate to the reality, as shown clearly in this accident. The controller issued information from which the BTC crew could infer the presence of a third aircraft with whom they also had a conflict. This aircraft was “phantom”, the statement containing this information mistaken. But BTC took this information as correct, which is appropriate given the understood roles of the participants, and made a decision based on it.

They were in fact presented with a decision problem: one can only go up or down; one sees an aircraft which TCAS is likely telling one to avoid; there is another aircraft which one does not see (either visually or metaphorically via TCAS display), but which the controller sees, and given the sense of the controller's instruction is likely at the same flight level or above. Two “threatening” aircraft; only two directions (up/down), either choice exacerbates one conflict or the other. BTC choose to descend, and have been widely criticised in the literature for so choosing. Ladkin suggested that decision to have been rational, given the crew's understanding of the state of the airspace and its occupacy, and gave reasons in [Ladkin 2004]. This is not the focus of the observations here. Neither is the apparent TCAS algorithm discrepancy which failed to issue a so-called “reverse RA” to DHL when TCAS sensed that separation was not being insured by the first RA, for that is a pure system issue, and not linguistic. (This anomaly has been fixed by Change Proposal 112E in the most recent version of TCAS, Version 7.1.)

The relevant observation here concerns the causal efficacy of mistaken information in a communication that was legitimate according to protocol but involved a not-uncommon cognitive slip. This information gave the BTC crew a mistaken understanding of the state of the airspace around them: they were considering two “threats”, not one. In fact, all three participants in this interaction, the two TCAS participants and the Zürich Center controller, had at this point three mutually incompatible understandings of the situation.

The situation can be expressed as simply as states of parameters.

Parameters:
posn: vertical and horizontal position; correspondingly *Dposn* for DHL position and *Bposn* for BTC position, with its two components:

hpn: horizontal position; correspondingly *Dhpn* and *Bhpn*
alt: altitude (vertical position); correspondingly *Dalt* and *Balt*
RA: an RA has been issued
sense: the sense of the RA is known, correspondingly *Dsense* and *Bsense*, which should be opposite
move: the up/down/level horizontal motion of an aircraft, correspondingly *Dmove*, *Bmove*.
phantom: there is a (third) aircraft at BTC's two o'clock

The collection of values of these parameters will be called here the *state* of the airspace. To indicate who has what information, and who does not have it, + and – signs are used. The state consists of the values of the parameters thus:
Dposn, *Bposn*, +*RA*, *Dsense* down, *Bsense* up, *Dmove* down, *Bmove* down, --*phantom*

However, a rational reconstruction of the state of knowledge of the three main actors during the crucial initial few seconds of the encounter yields

ATC: +*Dposn*, +*Bposn*, --*RA*, --*Dsense*, --*Bsense*, --*Dmove*, *Bmove* level, --*phantom*
DHL crew: +*Dposn*, +*Bhpn*, --*Balt*, +*RA*, +*Dsense* down, +*Bsense* up (inferred), +*Dmove* down, --*Bmove*, --*phantom*
DHL TCAS: +*Dposn*, +*Bposn*, +*RA*, +*Dsense* down, +*Bsense* up (inferred), +*Dmove*, +*Bmove*, --*phantom*
BTC crew: +*Dhpn*, -*Dalt*, +*Bposn*, +*RA*, +*Dsense* down (inferred), +*Bsense* up, --*Dmove*, +*Bmove* down, +*phantom*
BTC TCAS: +*Dposn*, +*Bposn*, +*RA*, +*Dsense* down (inferred), +*Bsense* up, +*Dmove*, +*Bmove*, --*phantom*

A simple observation is that all these five understandings of system state are different. Are they compatible? BTC crew's state is compatible with that of no other participant, because of the value of phantom. DHL crew's understanding is compatible with that of DHL TCAS. ATC's understanding is incompatible with that of no other participant, because of the value of *RA*. DHL TCAS and BTC TCAS are mutually compatible (which we would hope!).

Ladkin called these rational reconstructions of the partial system state, given the knowledge of each agent, the agent's *Rational Cognitive Model* (RCM) of the situation [Ladkin 2009]. Note that two of the agents are boxes, DHL TCAS and BTC TCAS.

It is normal that in the design of safety-critical systems states are identified which can be deemed to be *hazardous,* that is, roughly speaking, states in which the chances of an ensuing accident are much increased over those chances in a non-hazardous state (there are in fact various different notions of hazard in various engineering domains, which this loose statement attempts to cover). This state, in which many but not all significant agents have incompatible understandings of the partial state, is obviously hazardous – the aircraft collided seconds later!

Hazard analysis (HazAn) attempts to identify all hazardous situations in system operation, and avoid them or mitigate the possible consequences. HazAn is required by most standards for development of safety-critical systems, as here also. It is clear from the above that hazardous states can issue from communication of mistaken information. The questions which interest communications specialists, and which must be answered, are which hazardous states thereby ensue? And what to do about them? Analysis of this example here has indicated that answers may be found using techniques of finite-state combinatorics and analysis, techniques which are well-known to formal linguists.

Ladkin has proposed principles for the design of such systems, which principles can help avoid

such hazard states as the above [Ladkin 2009]. One such principle is the *Rational Cognitive Model Coherence Criterion*, that all participants' RCMs, their partial understandings of state, are compatible. Another is the *Mutual Cognisance of Relevant Parameters* criterion, that agents who need to know the value of certain parameters should have the true values available. Here, ATC did not know RAs had been issued until well into the interaction; and DHL did not know that BTC was also descending. It is of course a far cry from knowing which principles were violated to being able to devise protocols which adhere to them! For example, one instance of MCRP violation is 8we may hope!) solved by CP112E. The other may be solved by downloading RA status, when two are issued, automatically to ATC. However, controllers have legitimate worries about the intended use of such information and their responsibilities upon reception of it, and such worries have not been resolved at time of writing.

Case Study: Controlled Collision with Terrain on approach into Cali airport, Colombia, 1995.

On 20 December, 1995, an American Airlines Boeing 757 commercial passenger jet aircraft was flying into Cali airport in Colombia at night. Cali airport lies in a long, narrow valley surrounded by high mountains, and the runway lies along this valley, aligned at 10° right of magnetic north, allowing one to land to the south (on the end labelled Runway 19, the 19 standing for 190° magnetic on the compass) or to the north (on RWY 01, at 10° magnetic). To approach, an aircraft has to descend within the valley directly aligned with the runway, towards one end or the other, so at a heading of 10° magnetic or 190° magnetic, depending. The crew were used to overflying the airport, turning back, and landing to the north on RWY 01. However, on a sparse-traffic night they were offered a “direct” “straight-in” approach to RWY 19. They were unfamiliar with that approach, and hurried to find it in their charts (called “approach plates” because each is printed on one standard-format page. They may be found in Appendix C of the accident report in [Aeronautica Civil, Cali 1996].)

There was also some informational confusion. This account follows closely the account in [Gibbon&Ladkin 1996]. Here is the pilot-ATC exchange, taken verbatim from the accident report, from the original approach clearance to the end of reported conversation.

> *Approach replied, "Roger 965 is cleared to the VOR DME approach runway one niner, ROZO Number One arrival, report Tulua VOR"*
>
> *The flightcrew readback was, "Cleared the VOR DME one niner ROZO one arrival, we'll report the VOR, thank you Sir"*
>
> *Cali approach immediately clarified with, "Report Tulua", and the flightcrew immediately acknowledged, "Report Tulua"*
>
> *The flightcrew referred to the cockpit chart package (approach publications) after ATC instructions to "Report Tulua"*
>
> *Flightcrew discussion took place about the navigational aids to be used in the ROZO 1 Arrival, specifically their position relative to Tulua*
>
> *About 30 seconds later the flightcrew requested, "Can American Airlines 965 go direct to ROZO and then do the ROZO arrival sir?"*
>
> *Several radio transmissions then took place: Approach replied, "affirmative direct ROZO one and then runway one niner, the winds calm". The flightcrew replied, "all right, ROZO,*

the ROZO 1 to 19, thank you, American 965". And the controller stated, "Affirmative, report Tulua and twenty one miles, 5000 feet". The flightcrew acknowledged, "OK report Tulua, twenty one miles at 5000 feet, American 965"

Gibbon and Ladkin remark that AA965's request (after the phrase `*About 30 seconds later*') demonstrates a linguistic expectation. Going `*direct to ROZO and then do the ROZO arrival*' only makes sense if one expects that navigation fix ROZO is at the beginning of the ROZO One arrival. Which it isn't. A look at the approach plate in Appendix C of the report [*op.cit.*] shows that the ROZO navigational beacon is at the end of the ROZO arrival, not at the beginning. In the US, all such arrivals are named after the start point, not the end point as here.

Gibbon and Ladkin suggest that this reversal of the naming convention common in the US may have induced semantic dissonance in the US crew, which can lead to confusion with respect to temporal as well as spatial location (see, for example, [Gibbon 1995]). However, they note that the opposite is true with road naming conventions in a variety of cultures, e.g., close to where the author lives in Germany, between-town roads are named with the presumed goal: the very same road is called the *Wertherstrasse* (Werther road) in Bielefeld, and the *Bielefelder Strasse* (Bielefeld road) in Werther. The entry point for the ROZO One arrival is in fact the *Tulua* navigation beacon, some 22 nautical miles before the *Initial Approach Fix* (IAF) for the RWY 19 approach, which IAF is itself 9 nautical miles before the ROZO beacon, which is the *Final Approach Fix* (FAF) for the approach to RWY 19, 2.6 nautical miles before the threshold of RWY 19.

Ladkin and Gibbon note that the ATC response `*affirmative*' to this request is incorrect. AA965 cannot procedurally fly what they requested; therefore the controller's confirmation of that request is wrong. An ATC "*clearance*", which is what the flight crew and the controller are discussing, is both a commitment by ATC to keep the airspace clear for the space and time slots indicated and a mandatory (but revisible) routing instruction. Any clearance must therefore be a possible routing. Which the putative clearance contained in AA965's request is not. (When asked why he replied "*affirmative*", or "yes", when he knew that the correct reply was "*negative*", or "no", the controller cited relative status – he conceived of his to be much lower than that of airline pilots – and a consequent unwillingness to be seen to contradict people of higher status, even when he could see they were mistaken [Aeronautica Civil – Cali 1996]).

Ladkin and Gibbon also note the following words spoken by ATC: `*direct ROZO one*' (a complete phrase, as indicated by the following conjunction, `*and*'.) and that it is a syntactically incorrect phrase in any grammar of pilot-controller speech. The word `*direct*' must be followed by the name of a navigation fix. If "ROZO" is interpreted as the fix, the word `*one*' is then superfluous. The entire phrase `*ROZO one*' denotes an arrival, and an arrival name cannot correctly follow the word `*direct*'. They suggest that the lexical ambiguity between fix (a point) and arrival (a procedure) enabled production of this syntactically incorrect and semantically confused phrase.

The Rational Cognitive Models of the participants may be reconstructed thus, as they evolve through time.

ATC: *ROZO One arrival* [Tulua fix to ROZO fix]; *then RWY 19 approach* [ROZO fix to RWY 19].

AA965: Confirm: *ROZO One arrival* [unknown fix to unknown IAF RWY 19]; *then RWY 19 approach* [unknown IAF to RWY 19]

ATC: *Report overhead Tulua fix* [at start of ROZO One arrival]

AA965: Confirm: *Report overhead Tulua fix* [?expectation: at end of ROZO One arrival?]

AA965: Request: *fly direct to ROZO fix and then ROZO One arrival* [ROZO to unknown IAF]

ATC: Affirm request. {Semantically inconsistent phrase}; *and then ROZO One arrival* [ROZO to RWY 19]

AA965: Confirm *ROZO fix and then ROZO One arrival* [ROZO to unknown IAF] *and then RWY 19 approach* [unknown IAF to RWY 19]

ATC: Affirm. *Report overhead Tulua fix*.

AA965: Confirm. *Report overhead Tulua fix*.

We can represent the data structures here, giving the semantics of the phrases, by pairs: *<x,y>* represents flying from overhead Fix *x* to overhead Fix *y*; and by sequencing: *<x,y>;<z,w>* represents flying *<x,y>* and then *<z,w>*, and *z* must equal *w* (that is, either identical, or two different designations for the same fix). We may also compose, namely *<x,y>;<z,w>* may be represented by *<x,y=z,w>*. Then the syntax and semantics of the interchange proceeds as follows. Each line has three components, namely agent:syntax:semantics, with the components separated by colons. "Arr" abbreviates "arrival" and "App" "approach". Elements of a sequence are denoted by "*.first*", "*.second*", etc. Here, the Initial Approach Fix for the RWY 19 approach is the point at 21.0 nautical miles on the 013° radial of the Cali VOR navigation beacon (identifier CLO) [Aeronautica Civil – Cali 1996]. I abbreviate by CLO013/DME21.0

ATC: ROZO One Arr; RWY 19 App: *<Tulua,IAF=CLO013/DME21.0,FAF=ROZO,RWY 19>*

AA965: ROZO One Arr; RWY 19 App: *<unknown,unknown=IAF,unknown=FAF,RWY 19>*

ATC: Say overhead *Tulua*: say overhead *clearance.first*

AA965: We say overhead *Tulua*: say overhead *clearance.unknown* .
Supposition: *clearance.unknown* is *clearance.second* or later

AA965: Request ROZO and then ROZO One Arr: *<ROZO,unknown=IAF,unknown=FAF,RWY 19>*

ATC: Affirm, {inconsistent} and then ROZO One Arr: *inconsistent*

AA965: ROZO and then ROZO One Arr and then RWY 19 App:
<ROZO, unknown=IAF,unknown=FAF,RWY 19>

ATC: Affirm, Say overhead *Tulua*: say overhead *clearance.first*

AA965: We say overhead *Tulua*: say overhead *clearance.unknown*

It is clear from this representation of the syntax and semantics of these communications that the syntax is ambiguous and the RCMs of the participants are inconsistent, and remain so despite repeated attempts to unify them. This sequence shows a clear violation of RCMC and MCRP.

The following then happened. Because AA965 considered ROZO to be *clearance.first*, they attempted to fly direct. Unbeknowst to them, there was another navigational beacon, with the selfsame abbreviated identifier and the same frequency as ROZO, lying roughly at the nine o'clock position and within reception range. Upon entering the data for ROZO in the flight management computer, the autopilot then began a left turn towards this other beacon, ROMEO, which went unremarked by the crew for some significant portion of a minute. They deviated from the line of the

valley, and now had mountains in between them and the airport. However, they continued to descend into the airport. Upon realising the aircraft was flying in the wrong direction, the crew finally initiated flight towards the right fix, but there were now mountains in the way. As the aircraft flew towards a mountain, the ground proximity warning sounded, but the “escape manoeuvre” was not successful and the aircraft hit the mountain.

The above analysis proceeded, as did the analysis of the Überlingen midair collision, by representing the cognitive states of participant agents by simple data structures. Unlike in the Überlingen case, some concerted attempt was made to reconcile the RCMs of the participants through the usual ATC-pilot communication format, but this format was ambiguous (strictly: the same phrases were allowed different semantics). Also, the ATC mistake in the Cali situation was deliberate, engendered by social concerns, rather than being inadvertent as with Überlingen.

An attempt to enforce RCMC and MCRP in this case would involve disambiguating the language. This is formally straightforward: for example, one could require participants to enumerate explicitly the semantics - their understood sequence of fixes - saying “*unknown*” where they do not know, allowing a knowledgable communication partner to precisify. However, this would involve a major change in ATC-pilot communication. It is not clear whether such a major change would avoid more accidents than it might engender, for every change requires some time for it to mature in use.

We may observe that, once again, simple finite-state methods enable an adequate analysis of the communications and suffice to generate suggestions for avoiding or mitigating hazards.

The Linguistics of ATC-Pilot Communication: Work of Stephen Cushing and Extensions

The linguist Stephen Cushing investigated linguistic phenomena in ATC-pilot communication leading to incidents and accidents on behalf of NASA in 1987-9. His results were presented in [Cushing 1994], predating the accidents above and their analysis.

Cushing compiled a superb collection of examples, largely drawn from NASA's Aviation Safety Reporting System, an anonymised reporting facility guaranteeing immunity for reporters, intended to identify trends as well as individual occurrences or potentially hazardous situations so that these could be preempted if necessary by regulators in the hope of reducing accident statistics.

Cushing identified problems of ambiguity (as we considered in the case studies above), homophony (similar-sounding phrases having different meanings, for example similar aircraft call-signs and consequent misunderstandings of a crew that a clearance intended for another aircraft was in fact for them), the influence on meaning of spoken punctuation and of intonation. He identified problems of reference: uncertain reference, unclear addressee and unclear hand-off (an air traffic control procedure for assigning an aircraft to a different controller), which overlap with problems with homophony. Then there are problems of inference, including implicit inference, lexical inference, and false assumptions, as we have seen above concerning the Cali accident. Then come problems of repetition, especially involving readbacks, which we have also seen in the Cali case study above. There are problems with numbers and their abbreviations, in particular for radio frequencies, in navigational procedures, for runways, and for altimeter settings (communications of the local barometric pressure). He noted problems with radios, problems of compliance, and general problems of technical communication loss such as lost messages or garbled message contents. Although such problems are formally recognised and adherence to the controlled language of ATC-pilot communication is supposed to be a prophylaxis, Cushing's investigations showed just how big a gap there is between the reality of everyday ATC-pilot communication and the solutions offered by the controlled language promulgated by the aviation regulatory authorities [FAA ATC Phraseology].

One reason for the continuing discrepancies may well be found in work of the organisational sociologist Jens Rasmussen [Rasmussen 1997]. He observed that human agents in their everyday work devise ways of getting their job done optimally. That is, optimally for them, and the agent may well have different goals from the designers of the system within which heshe is working. Rasmussen noted in many accidents how these individual optimisations, which can be observed and which he called "migration to the boundary" (MttB), his phrase for coming up against the limits of what one can optimise, conflicted either with overall system safety goals, or with each other to engender hazardous system states and behaviors. Most ATC-pilot communication is routine, and much of it can be expressed by use of much shortened phrases amongst knowledgeable pilots and controllers: elisions and suchlike. And these optimisations, such as elisions, rarely lead to miscommunication, lead even more rarely to hazardous situations, and these themselves lead even more rarely to accidents. So MttB in formalised communication has immediate and pervasive benefits and only occasional, very rare, disadvantages. Which even more occasionally result in accidents, sometimes with heavy loss of life. Which of course is why the formalised communication, not foreshortened by MttB-derived modifications, is promulgated in the first place.

Cushing suggested a "long-term solution" to the disadvantages derived from the pervasive linguistic phenomena which he had enumerated. The solution consisted in a computer-supported "intelligent" voice interface for aviation communications, resting on a formal language. He proposed an architecture for the system, and use of some computer-supported-linguistic tools, such as LEX and BISON, based on a formal grammar of aviation communications. He made extensive use of SW available on Apple Macintosh computers, at that time limiting the applicability of the work since Apple did not have a large market share of personal computers.

Hilbert and Ellermann showed, however, that the grammar proposed by Cushing was formally inadequate for these communications, in that it did not correctly represent them, allowing some phrases that were inappropriate, and ruling out other phrases in officially-sanctioned use [Ellermann&Hilbert 2001]. They went on to show that accurate pilot-ATC communications could in fact be represented by a formally-simpler grammar in extended Backus-Naur form (EBNF), which makes it suitable for standard computer-linguistic parsing and analysis tools [Ellermann&Hilbert 2002]. In this work, Ellermann and Hilbert also implemented a parser for their grammar on a Unix-based computer system. Unix had a much broader installed base than Apple systems at that time. Nowadays, Apple operating systems are also Unix-based.

Hölz and Hettenhausen designed and partially implemented a limited computer corpus of aviation communications for published accidents [Hölz&Hettenhausen 2001]. This task is made particularly difficult in that such communications are not publicly available, in part through agreements with professional-pilot unions for reasons of privacy and employee protection in the workplace. Indeed, such transcripts are not available in most cases even within airlines. They are sometimes made available, say by the US National Transportation Safety Board, in the case of significant accidents, but other investigative authorities, such as the Transportation Safety Board of Canada, redact heavily and summarise, through concerns for the privacy of participants and victims. The chances of building a representative corpus are thereby limited. However, Döring, McGovern and Sanders [Döring&MacGovern&Sanders 2001] showed using the limited Hölz-Hettenhausen corpus that semantic parsing techniques on the corpus enabled unique correlation of spoken phrases in the transcripts with their formal Ellermann-Hilbert equivalents in a high proportion, some 80-90%, of cases. This work validates Cushing's intuition that computer linguistics can be fruitfully applied to improve aspects of day-to-day controller-pilot communications with respect to the unsafe features he identified in his compilation of examples.

The question is, however, still unanswered what sort of computer-supported tools would most benefit these communications from the point of view of enhancing safety, and which such tools

stand a reasonable chance of being implemented and successfully introduced into service. Limited versions of such communications are already flying, for example the CPDLC communications, which are digital-computer-mediated communications whereby the sender uses a keyboard, the receiver a screen, with a button for acknowledgement of reception and/or clearance. As of writing, CPDLC has been successfully used for communications on trans-oceanic flights for almost two decades.

One possible future development could see verbal statements of agents parsed by computer, transmitted digitally as text after checking for compliance, consistency, completeness, and common forms of failure, and voice-synthesised at reception. The state of the art is a long way from having such proof-of-concept prototypes, and even further from a practically workable system which would be accepted by the aviation community.

Such systems are feasible. We turn now to a domain in which such tools have been successfully implemented, the linguistically somewhat more simple domain of train dispatching and train announcement procedures.

Train Announcement Procedures (TAP, German: *Zugmeldeverfahren*) are one of the two procedures on German railways for coordinating traffic on single-track lines without signals. The other is train dispatching (*Zugleitbetrieb*, ZLB), for which computer support will be considered, below.

Train Dispatching and Train Announcement Procedures

Train dispatching and TAP are forms of train control used in German railway operations on sparsely-used single-track railway lines without signal support (signals are required by most railway authorities on all tracks which have above a certain frequency of train movements). Trains halt at a stopping point, usúally a railway station, and negotiate a verbal protocol with a train controller in order to proceed to the next stopping point. The region of line between two stopping points is called a block, and the protocol must ensure that no two trains may proceed in the same block at the same time under central control. It must be possible, for track maintenance, rescue, and shunting purposes to allow trains to move at line-of-sight braking speeds under certain circumstances, but a train must not be allowed to proceed under normal driving conditions into a block in which some line-of-sight operations are taking place, and line-of-sight operations must not be allowed to proceed in a block in which a train is proceeding under normal driving conditions. Basically, the two allowable states for one block are: one train only, or everyone is being slow and careful and can stop before hindrances.

Train dispatching operates with one train controller, who controls the movements of all trains on a specific section of track (a block). A controller using dispatching communicates with train drivers using dedicated radio.

TAP is a protocol in which two train controllers, usually the station masters at the stations at each end of a block, coordinate the dispatch of trains into the block between themselves first, and then communicate permissions with the train drivers. The station masters communicate with each other using dedicated telephone lines, and with the train drivers using dedicated radio.

The verbal protocols for train dispatching and TAP are strict, use defined phraseology, and are part of German rail law.

The protocols do most often support the operational safety requirements, but mistakes are sometimes made.

Case Study: the Warngau Single-Track Train Collision, 1975

On Sunday 8th June 1975 18:32h, two passenger trains crashed into each other near Warngau, Germany. The passenger trains both entered the single-track line between Warngau and Schaftlach after receiving permission to continue. One train was lifted off the track and fell to the side. The accident killed 44 people, including the two engine drivers, and injured 122. Damage amounted to about 4 million Deutsche Mark (about €2 million). Verbal execution of protocols played a significant role.

The single-line track between Warngau and Schaftlach is 4.8 km long. It is part of the line between München (Munich) and Lenggries, in Bavaria. Automatic Train Protection, automatic braking on passing a signal at danger, is installed and functions through inductive-electromagnetic trackside and on-board equipment (Indusi). Indusi was required for the line because it was designed for trains travelling faster than 100 kph. However, traffic density on the line was sufficiently low that compartmentalisation into blocks was not required between stations. Thus the track between Warngau and Schaftlach constituted one block (German *Blockabschnitt*) to which access was controlled by station masters/controllers (*Fahrdienstleiter*) in Warngau and in Schaftlach using TAP. The block contained one level-crossing (grade crossing), protected by warning signals.

Under TAP, trains must be *offered (OFF)*, *accepted (ACC)*, *checked out (PERM)* and *reported back (ACK)* between station masters. These procedures are as follows. Suppose a train at holding point (here, station) A wishes to progress to holding point/station B.

- OFF: if he does not regard the block as *occupied*, station master A tells station master B that he has a train ready to enter a specific block.
- ACC: station master B confirms that the train can enter (if the block is not already *occupied*), and holds the block as *occupied* until the train arrives at the B side of the block, at his station, and leaves. This arrival event will be assured visually, and verbally by an ACK, below.
- PERM: station master A notifies the train driver that the train may enter the block and notes the train's time of departure.
- ACK: station inspector B notifies station inspector A that the train has cleared the block.

The OFF-ACK pair guards against opposite-direction trains colliding head-on, for station master B will know if he has already sent a train into the block, and station master A will have marked the block as *occupied*. The OFF-ACK pair guards against same-direction trains colliding head-on-tail. I forego a formal analysis of the TAP protocol here; we will see such an analyis below for train dispatching.

On the day of the accident train 3591 was waiting at Warngau, ready for departure. Train 3594 S (scheduled sundays only) was waiting at Schaftlach, ready for departure. Both station masters offered their trains (OFF), and both interpreted the communication for their respective train as if they had received ACC. Both cleared their trains to depart.

The apparent misunderstanding was determined to be a result of not using the fixed phrases. It was reported that the station masters conversed using a Bavarian dialect and “cut corners” in the execution of the protocol. If so, this would then be an example of MttB. Unfortunately, there is no transcript of the communications publicly available.

According to the written schedule, train 3591 had to wait for two trains coming from Schaftlach: train 3592 and train 3594 S. The station master at Warngau had to handle three trains in a short time (about 9 minutes), and as well as sell tickets and answer passengers' questions. He was the

only person in the Warngau station.

Other causal factors than miscommunication played a role. For example, a feature of train timetabling for opposite-direction trains, “air intersections”, *Luftkreuzungen*, theoretical positions within a block at which trains following a timetable would cross were the block to consist of two one-way tracks, was used in the pictorial timetable for these stations for these trains. In fact, the EBO regulations explicitly disallowed *Luftkreuzung* in timetable construction. It was nevertheless common practice at this time to use them, to allow station masters some scheduling flexibility in case trains were not running on time.

Also, between Warngau and Schaftlach, there were two signals guarding a level crossing (grade crossing) for trains. The signal indicator lights start blinking when a train's first axle passes a contact point on the track. If a train from the opposite direction passes its respective contact point, then the control light does not blink. In this case the train driver must come to a full stop before the crossing. Train 3591 passed the contact point first; train 3594 S passed its respective contact point after 3591 passed his, but the driver of 3594 S did not react to the lack of blinking signal, which should have indicated to him that he should commence an emergency stop. The trains were in a blind curving section of track, so visual contact was not possible until late in the convergence. Either visual contact or braking of train 3394 S would have reduced the severity of the collision. This is not the only case in German railway history in which safety information is conveyed by a lack of action, here the lack of operation of the signal light, before shortly later an accident ensued. The Brühl derailment accident [Brinkmann&Lemke 2003] is another example. Our concern here is not, however, with the analysis of visual communications.

The Warngau collision and the Cali accident have some communication phenomena in common. Non-standard phraseology was used, in this case local dialect and ambiguous phrasing – MttB at work - and the result was an inconsistent understanding of overall system state by the participants: RCM and MCRP were again violated, obviously.

This raises the question, raised above for aviation communications, whether computer support can help secure such operations. In this case, there are working computer-supportive systems for the alternative to TAP, train dispatching. Train dispatching without local signal support is supported on some Austrian railway lines run by the company Stern & Hafferl, such as the local railway in the region around Linz, using a system developed at the University of Applied Sciences at Wels (Fachhochschule Wels) by a team led by Burkhard Stadlmann [Stadlmann 2004]. Stadlmann's work represents the state of the practice in computer-supported train dispatching, but it holds minimal interest from the linguistic point of view, since the system is primarily informational to the agent.

If that is state of the practice, then what about state of the art?

Computer Support with a Proven-Correct Protocol Implementation

Bernd Sieker has considered the Zugleitbetrieb protocol defined by German railway law for non-federal railways. There are many such lines run by private railway companies in Germany, but only a few, mostly in Sachsen around Dresden, are non-signalled. Most lines, including all those around Bielefeld where we work, are supported by signalling systems to some degree.

In his PhD thesis, in German [Sieker 2010], Sieker started from a formal expression in logic of the above basic requirement of Zugleitbetrieb, the one-block one-train criterion, with the distinction between central control and line-of-sight operations. He proceeded to derive the protocol along the lines indicated in the law [FV-NE], by using a technique of computer-system design elaboration known as formal refinement. Sieker derived a series of ever-more-elaborate state machines to

implement the verbal protocol formally using computational techniques, and each state machine was formally proven using formal logic to mimic the more abstract state machines higher in the development hierarchy. The penultimate step consisted of factoring a state machine representing the global system state into different state machines for each agent, in such a manner that it could be proven that the interactions of the agent state-machines formed exactly the global state machine from which they were derived.

The final step was implemented by Phil Thornley of SparkSure, who implemented the agent state machines in the annotated programming language SPARK [Barnes 2003] and proved formally, using the required annotations supplementing the executable code, that the executable computer programs implemented exactly and precisely the agent state machines derived by Sieker.

The result is a fully formally analysed SW system, which runs a verbal protocol precisely, and which has been formally proved to satisfy the basic safety requirement of Zugleitbetrieb. That means that running the SW is guaranteed to execute the protocol logically correctly (Sieker discovered during the course of the work that the protocol did not define actions in all reachable states. The required actions were mostly obvious, and were added. The protocol which Sieker's development runs is strictly speaking a formal completion of the legal protocol).

Of course, there can still arise hardware problems with the machine, and it may be that the compiler which translates the SPARK source code into machine language for the machine is inexact, but these are problems which are common to all computer support of any task, are not specific in any way to mimicking verbal protocols, and have been addressed by the computer industry. Indeed the issue of unreliable compilation has been at the center of work by the developers of SPARK, the company Altran-Praxis, for two decades.

The details of Sieker's work here generally follow an account in [Ladkin&Sieker&Stuphorn 2009]. This account focused on the safety properties of the development – for example, the novel approach to hazard analysis, derivation of a demonstrably-complete set of safety requirements, and aspects of the development that would and would not generalise to other situations – which are of less interest to us here.

The highest-level language, called Level 0, required to express the basic safety requirement is simple, indeed astonishingly simple. This simplicity allowed a manual selection of safety requirements, which is complete in the sense that one can easily show that they cannot be logically strengthened.

The next language level, Level 1 and further language levels were defined through the usual type of refinement process familiar to computer scientists, in which the extensions of the language were carefully controlled in small steps. The entire functional operation of the system at each level could be expressed in terms of a global finite-state machine, which state machines were formally proved through simple, short propositional-logic arguments to refine each other, sometimes through addition of extra requirements, which then were added to the list of safety requirements.

The final refinement step involved transforming the global state machine into a set of agent state-machines, one representing a driver and one a train controller, which communicate by means of message-passing. The agent state-machines are expressed in a structure called a Message Flow Graph (MFG), for which a formal semantics has already been defined [Ladkin&Leue1995]. By this means the MFG could be formally proved to implement the global state machine from which it was refined.

The MFG agents were then implemented as SPARK procedure skeletons with the appropriate

annotations by Phil Thornley of SparkSure, and the annotation proved to implement the MFG.

The entire development ensured complete traceability between very-high-level safety requirements and SPARK source code. Suppose such a system were to be implemented as either an automated dispatching system, with computers replacing the human agents, or, more practically, as a support system which checks that the required steps have been performed by the human agents. Then the risk of using the system resides entirely in the hardware and communications systems used, as well as in the compiler used to compile the source code, and in human factors such as whether the system is used as intended. Any risk inherent in the logic of the program design itself has been eliminated, and shown to be eliminated by the logical proofs. The risk of this computer-based system has thereby been reduced to that of other, generic risks, which data from other, unrelated computer projects may be used to assess.

The architectural scheme of the development is an example of a method called Ontological Hazard Analysis, proposed by myself and developed inter alia by Sieker and Sanders [Stuphorn&Sieker&Ladkin 2009, Ladkin&Sanders&Sieker 2012]. The Hazan aspects of the development interest us marginally here. The structure of the development is given in the following diagram.

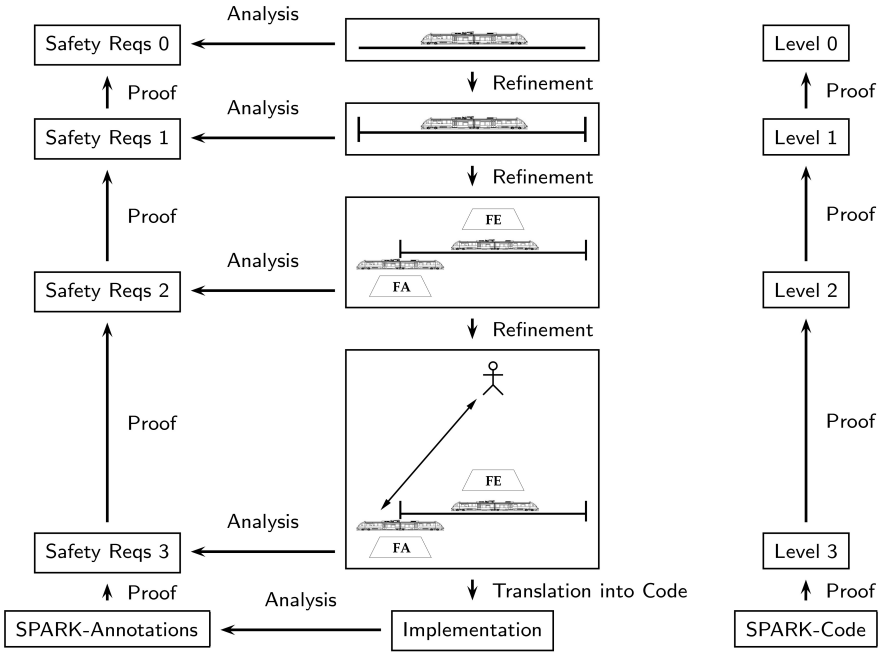


Fig. 1 Structure of the OHA

Level 0

The goal of the highest specification level, Level 0, is to provide a description language adequate for expressing the intuitive safety requirement of train dispatching, the one-block one-train requirement with exceptions, that is so simple that (a) we can define safety axioms to which all applications experts can assent, and (b) at the same time ascertain that these axioms are both *correct* and *complete* relative to the expressions of the language.

Fig. 2. Schematic Representation of Level 0

The Level 0 language is a logical language, which includes sorts of objects, properties of objects in those sorts, and relations between objects; a basic predicate-logical language with no complex features. Sorts and relations are as follows.

Table 1. Level 0 Sorts

Sort	Description
Vehicle	Any train or other vehicle operating on tracks
Block	A section of a track inside or outside a station

Table 2. Level 0 Relations

Relation	Description
inA(F,S)	Train F is in Block S
ZV(F,S)	ZV(F,S) Train F may occupy Block S under central responsibility (normal scheduledoperation)
LV(FS)	ZV(F,S) Train F may occupy Block S under local responsibility (special case)

Determining Safety Axioms. Using elementary propositional logic as well some semantic domain knowledge, Sieker is able to determine that there turn out to be only 6 safety postulates on Level 0 from consideration of a couple of dozen non-equivalent statements from a total of 256 statements before semantic reduction. We use the following shorthand notation for a train F1 and one block S: LV(F1,S) = LV1, ZV(F1,S) = LZ1, inA(F1,S) = in1; similarly for train F2. The Safety Postulates at Level 0 are enumerated in Table 3. It is not shown here how the safety postulates were derived, for this is not our main concern.

Table 3. Safety Postulates at Level 0

Safety Postulate	Description
$ZV1 \Rightarrow \neg LV1$	If a train is in a block under central responsibility it cannot be there under local responsibility
$\neg LV1 \wedge in1 \Rightarrow ZV1$	If a train is in a block and is not there under local responsibility then it is under central responsibility
$in1 \wedge ZV1 \Rightarrow \neg LV1$	If a train is in a block under central responsibility it cannot be in that block under local responsibility
$(F1 \neq F2) \Rightarrow (LV1 \Rightarrow \neg ZV2)$	If a train is in a block under local responsibility another train under central responsibility cannot be in that block
$(F1 \neq F2) \Rightarrow (in1 \Rightarrow \neg ZV2)$	If a train is in a block another train under central responsibility cannot be in that block
$(F1 \neq F2) \Rightarrow (ZV1 \Rightarrow \neg ZV2)$	If a train under central responsibility is in a block, another train under central responsibility cannot be in that block.

Level 1: The First Refinement Level

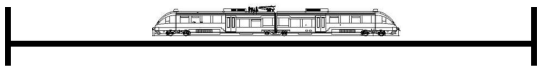


Fig. 3. Schematic Representation of Level 1

The generic block of Level 0 is refined by introducing the new sorts Track and Station. No other modifications are undertaken, in accordance with the principle of keeping refinement steps simple. This results in the sorts in Table 4.

Table 4. Level 1 Sorts

Sort	Description
Vehicle	Train or other track vehicle
Block	A track section
Track	A piece of track in the station
Station	A station where messages are exchanged

On Level 1, there are then 10 relations, distributing the sorts over the relations. Meaning Postulates define what each Level 0 sort and Level 0 relation means in terms of the Level 1 language. Using these Meaning Postulates, we arrive at 12 Safety Postulates for Level 1.

Level 2

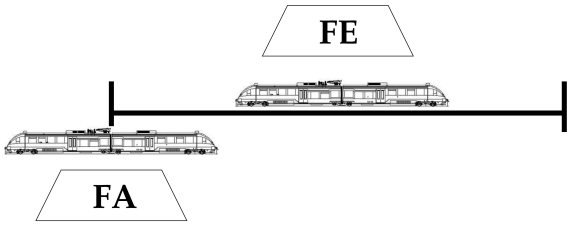


Fig. 4. Schematic Representation of Level 2

At Level 2, no new sorts are added, but additional relations concerning ‘clearances’ are added, as shown in Table 5.

Table 5. Level 2 Relations

Relation	Description
FA(F,A,B)	Train F, in station A, has asked for clearance to go to station B
FE(F,A,B)	Train F, in station A, has received clearance to go to station B
AFE(F,A,B)	Train F, in station A, has been denied clearance to go to station B
KH(F,A,B)	No obstructions are known for train F to go from station A to station B

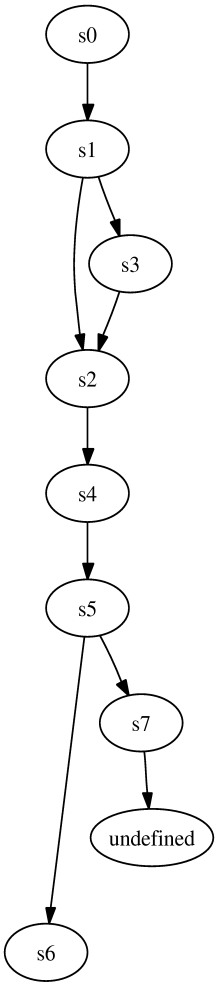
At this point, Sieker is now able to build a state-machine representing the global states of clearances which represents a train journey. The state-machine is shown in Figure 5, which is presented as a Predicate-Action-Diagram [Lamport 1995].

Three simple Meaning Postulates and elementary logic lead to only two new Safety Postulates, which can be expressed informally as:

- if no obstructions are known and clearance has been given, the block can be occupied under central responsibility
- clearance for a block cannot be given for a second train, if clearance has already been given for a train for the same block in either direction.

New hazards identified at this level are simply the negations of the newly-identified Safety Postulates:

- Clearance has been given, and no obstruction is known, but the conditions for occupying the block under central responsibility have not been met.
- Clearance has been given for two trains for the same block at the same time.



State	Description
s0	inZ(T , A)
s1	∧ inZ(T , A)
	∧ FA(T , A, Next(T , A))
	∧ ¬ FE(T , A, Next(T , A))
s2	∧ inZ(T , A)
	∧ FA(T , A, Next(T , A))
	∧ KH(T , A, Next(T , A))
s3	∧ inZ(T , A)
	∧ FA(T , A, Next(T , A))
	∧ ¬ FE(T , A, Next(T , A))
	∧ ¬ KH(T , A, Next(T , A))
	∧ AFE(T , A, Next(T , A))
s4	∧ inZ(T , A)
	∧ FA(T , A, Next(T , A))
	∧ FE(T , A, Next(T , A))
	∧ KH(T , A, Next(T , A)
s5	∧ zw(T , A, Next(T , A))
	∧ FE(T , A, Next(T , A))
	∧ KH(T , A, Next(T , A))
	∧ ¬ LV(T , S)
s6	inZ(T , A) = s0
s7	∧ btw(T , A, Next(T , A))
	∧ FE(T , A, Next(T , A))
	∧ ¬ KH(T , A, Next(T , A))
	∧ ¬ LV(T , S)

Fig. 5. Level 2 state-machine (a Predicate-Action Diagram)

Level 3

Level 3 includes the specific defined communications between train drivers and a train controller. Message types correspond to the states in which the trains can be, and are designed according to the message types prescribed in the regulations for German non-state-owned railways (VDV 2004).

Table 6. Message types at Level 3

Message Type	Description
FA	Request for Clearance (Fahranfrage)
FE	Clearance (Fahrerlaubnis)
AFE	Denial of Clearance (Ablehung der Fahrerlaubnis)
AM	Notification of Arrival (Ankunftmeldung)

In addition, we define relations to describe sending and receiving of messages, as shown in Table 7.

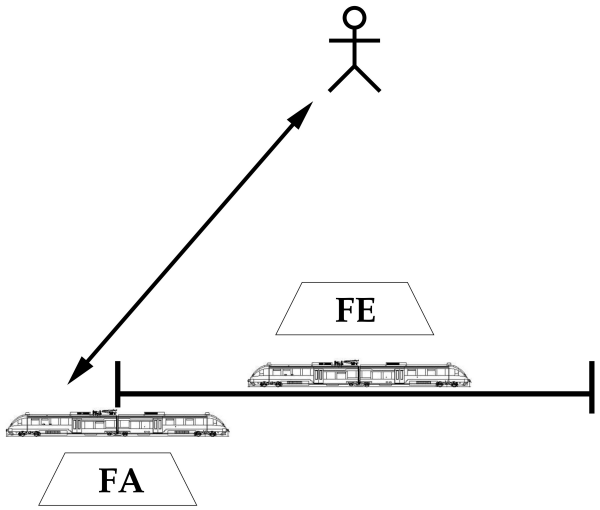


Fig. 6. Schematic Representation of Level 3

Table 7. Relations at Level 3

Relation	Description
Sent(MT,T,A)	Message of type MT, concerning train T and station A has been sent.
Recd(MT,T,A)	Message of type MT, concerning train T and station A has been received.

Note that the sender and receiver of the message are implicit. Messages of type FA and AM are always sent by the specific train driver to the train controller, messages of type FE and AFE are always sent by the train controller.

Through appropriate Meaning Postulates, the state machine of Level 2 can be augmented to include communications. This now more complex state machine can be transformed into a Message Flow Graph (MFG), to make the communications visually clear. The MFG represents the individual agents and their changing states as vertical lines, message passing between agents as angled lines. The MFG can be formally shown to define the same global state machine as the Predicate-Action-Diagram for this level. The MFG is used as the starting point to define the SPARK implementation and the SPARK verification conditions are determined by hand to define the MFG of Figure 7.

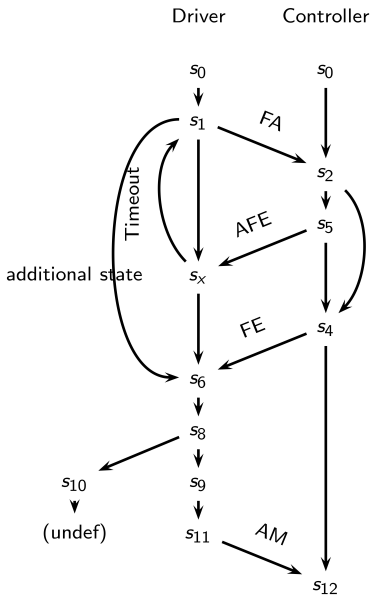


Fig. 7. The Message Flow Graph

Table 8. States corresponding to the Message Flow Graph

MFG-Trans.	Driver-State	Controller State	Global State
s0	inZ(T , A)A)	–	inZ(T , A)
s0 → s1	∧ inZ(T , A)	--	∧ inZ(T , A)
	∧ Sent⟨FA, T , Next(T , A)⟩		∧ Sent⟨FA, T , Next(T , A)⟩
s1 → s2	--	Recd⟨FA, T , Next(T , A)⟩	∧ inZ(T , A)
			∧ Sent⟨FA, T , Next(T , A)⟩
			∧Recd⟨FA, T , Next(T , A)⟩

The Step to Code: Implementation in SPARK

SPARK is based on a subset of the Ada language. It uses annotations to denote data and information flow and to specify pre- and post-conditions for functions and procedures. The SPARK toolset includes a static code analyser which uses the annotations to prove the absence of run-time errors, such as division by zero, buffer overflows and other bounds violations, before the code is actually compiled.

Typical Example of SPARK annotations corresponding to the MFG

```
procedure Send_FA (DS : in out Driver_State);
--# global out Messages.Out_Queues;
--# derives Messages.Out_Queues from
--# DS
--# & DS from
--# *;
--# pre D_State(DS) = D_S0;
--# post To_S1(DS~, DS);
```

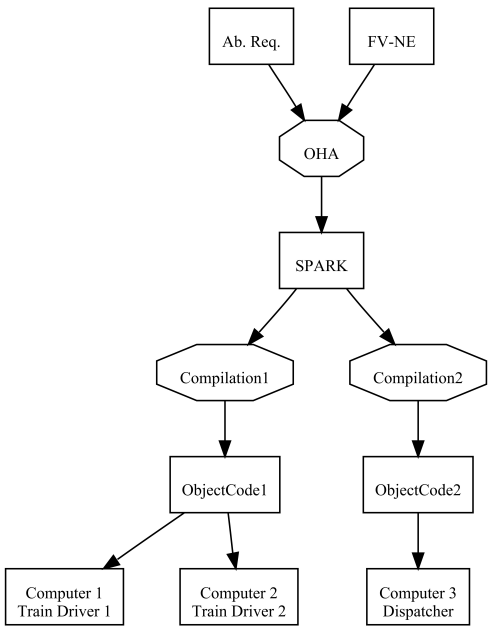


Fig. 8. The Development-Flow Graph of the SPARK verbal-protocol simulation

Conclusions of the FV-NE Project

This is another case study in which techniques associated with finite-state systems are adequate for analysing a verbal safety-critical communications protocol. Through the thorough use of formal-refinement techniques, it was possible to prove formally in propositional logic that the derived system expressed in executable SPARK source code fulfilled the basic system safety requirement without exception. One could well imagine a helper computer system, installed on trains and at the train controllers' positions, which reiterated the steps of the verbal protocol as executed by the train drivers and controllers. This system is guaranteed to execute the protocol unambiguously and correctly, and therefore could have helped avoid such an outcome as at Warngau. Furthermore, Sieker's work has shown that the verbal protocol is itself logically correct, in that it logically fulfils the basic safety requirement.

What is most remarkable about this development in the context of this survey of communication in critical systems is how far the technology has come from the tools generally used by computational linguists.

Rigor and Accuracy in Verbal Protocols: General Conclusions

We have considered verbal safety-critical communications in transportation systems, namely air and rail travel. This domain forms a small but, it has been argued, clearly delineated subspecialty of safety-critical system science at the borders of general and computational linguistics.

We have seen how techniques from finite-state system science can profitably be used to analyse, as well as in the last instance to help execute, the protocols required for these communications in order to enhance safety.

Two case studies from air travel have been considered in detail, and some safety issues effectively addressed by design principles have been discussed. However, considering work by Cushing as well as extensions by Ellermann, Hilbert and others has shown just how far we have to go in this area to reach a state-of-the-art computer-supported system which would enhance safety in these critical communications by eliminating or mitigating features which have been shown to compromise that safety.

The requirements and protocols of train dispatching are much simpler. In this case, helper systems are already deployed (state of the practice) in Austria, derived from work of Stadlmann et al. But his methods are not based on linguistic analysis, and a state of the art development based on careful linguistic analysis has been exhibited by the research work of Sieker, described broadly above.

All these methods, with the possible exception of those of Stadlman, employ mainly finite-state system techniques.

It has been remarked how far the system developed by Sieker lies from the typical considerations of computational linguistics. One may speculate that, as verbal-protocol systems become more well understood, the phenomena of interest to computer linguists will recede into the background, to be supplanted by standard techniques of reliable- and safe-SW-system development as known to SW safety system engineers. For indeed we should wish to control these phenomena, as Cushing and the case studies have shown. It has been suggested that general techniques are emerging with which we may profitably continue to go about this task: simple abstract data-structures representing state; finite-state-machine techniques; formal refinement; simple formal languages based on the ontology of predicate logic (maybe sorted predicate logic); meaning postulates to relate levels of refinement;

and of course formal proof of properties of simulation.

References

[Aeronautica Civil, Cali 1996] Aeronautica Civil of the State of Colombia, Aircraft Accident Report, American Airlines Flight 965 near Buga, Colombia, 20 December, 1995. Report published 1996, available from http://www.rvs.uni-bielefeld.de/publications/compendium/incidents_and_accidents/cali_american_airlines_b757.html

[Barnes 2003] John Barnes, High-Integrity Software: The SPARK Approach to Safety and Security, Addison-Wesley, 2003.

[BFU 2004] Bundesstelle für Flugunfalluntersuchung/German Federal Bureau of Aircraft Accidents Investigation, Investigation Report AX001-1-2/02 (the mid-air collision near Überlingen, Germany on 01 July 2002), May 2004. Available in English and German from http://www.bfu-web.de

[Brinkmann&Lemke 2003] Christian Brinkmann and Oliver Lemke, Analysis of the Brühl Railway Accident, invited talk at the 2nd Bieleschweig Workshop on Systems Engineering, 1-2 July 2003, Braunschweig. Slides available from http://www.tu-braunschweig.de/ifev/veranstaltungen/bieleschweig/bieleschweig2/

[Cushing 1994] Steven Cushing, Fatal Words, University of Chicago Press, 1994.

[Döring&McGovern&Sanders 2001] André Döring, Mark McGovern and Jan Sanders, Computational Analysis of Cockpit-Voice-Recording Transcripts, RVS Technical Note RVS-Occ-01-07, Faculty of Technology, University of Bielefeld, 11 November 2001. Available at http://www.rvs.uni-bielefeld.de/publications/Papers/doering-01-07.pdf

[Ellermann&Hilbert 2001] Martin Ellermann and Mirco Hilbert, Review of the Cushing Grammar, RVS Technical Note RVS-Occ-01-02, Faculty of Technology, University of Bielefeld, 23 July 2001. Available at http://www.rvs.uni-bielefeld.de/publications/Papers/hillermann-critique.pdf

[Ellermann&Hilbert 2002] Martin Ellermann and Mirco Hilbert, Building a Parser for ATC Language, RVS Technical Note RVS-Occ-01-05, Faculty of Technology, University of Bielefeld, 18 February 2002. Available at http://www.rvs.uni-bielefeld.de/publications/Papers/hillermann-parser.pdf

[FAA ATC Phraseology] Federal Aviation Administration of the United States of America, Order JO 7110.65T, Air Traffic Control, February 11, 2010. Available from http://www.faa.gov/documentLibrary/media/Order/ATC.pdf

[FV-NE] Bundesrepublik Deutschland (Federal Republic of Germany), Fahrdienstvorschriften für die Nichtbundeseigenen Eisenbahnen (Train Control Regulations for Non-State Railway Operations).

[Gibbon 1995] Dafydd Gibbon, The Parable of Next Thursday, 12 November 1995. Available at http://coral2.spectrum.uni-bielefeld.de:/~gibbon//nextthur.html

[Gibbon&Ladkin 1996] Dafydd Gibbon and Peter Ladkin, Comments on Confusing Conversation at Cali, 7 February 1996. Available at http://www.rvs.uni-bielefeld.de/publications/Incidents/DOCS/Research/Rvs/Misc/Additional/Reports/cali-

comment.html

[Hölz&Hettenhausen] Oliver Hölz and Thomas Hettenhausen, Building a Corpus for Cockpit-Voice-Recorder Transcripts, RVS Technical Note RVS-Occ-01-06, Faculty of Technology, University of Bielefeld, 23 October 2001. Available at http://www.rvs.uni-bielefeld.de/publications/Papers/hettenhoelz-report.pdf

[IEC 61508] International Electrotechnical Commission, International Standard 61508, Functional Safety of Electrical/Electronic/Programmable-Electronic Safety-Related Systems, 2nd Edition, 2010.

[Ladkin 20020812] Peter Bernard Ladkin, ACAS and the South German Midair, RVS Technical Note RVS-Occ-02-02, Faculty of Technology, University of Bielefeld, August 2002. Available at http://www.rvs.uni-bielefeld.de/publications/Reports/ACAS-Midair-www.html

[Ladkin 2004] Peter Bernard Ladkin, Causal Analysis of the ACAS/TCAS Sociotechnical System, Keynote Paper, 9th Australian Workshop on Safety-Related Programmable Systems, in volume 47 of Conferences in Research and Practice in Information Technology, 2004. Available at http://crpit.com/abstracts/CRPITV47Ladkin.html as well as http://www.rvs.uni-bielefeld.de/publications/Reports/SCSS04.pdf

[Ladkin 2009] Peter Bernard Ladkin, Securing the Interface: Safety-Critical Interaction Between Humans and Robots, Extended Abstract of Keynote Talk, 4th IET System Safety Conference, 2009.

[Ladkin 20100903] Peter Bernard Ladkin, Fully Automatic Execution of Critical Manoeuvres in Airline Flying, http://www.abnormaldistribution.org/2010/09/03/fully-automatic-execution-of-critical-manoeuvres-in-airline-flying/ 3 September 2010.

[Ladkin&Leue 1995] Peter Ladkin and Stefan Leue, Interpreting Message Flow Graphs, Formal Aspects of Computing 7(5):473-509, 1995. Available through http://www.rvs.uni-bielefeld.de/publications/abstracts.html#FAC-MSC

[Ladkin&Sanders&Sieker 2012] Peter Bernard Ladkin, Jan Sanders and Bernd Sieker, Safety Analysis of Computer-Based Systems, Springer-Verlag, forthcoming 2012.

[Lamport 1995] Leslie Lamport, TLA in Pictures, IEEE Trans. Soft. Eng. 21(9):768-775, 1995. Available through http://research.microsoft.com/en-us/um/people/lamport/pubs/pubs.html

[Leveson 1995] Nancy Leveson, Safeware, Addison-Wesley 1995.

[Rasmussen 1997] Jens Rasmussen, Risk Management in a Dynamic Society: A Modelling Problem, Safety Science 27(2/3):183-213, 1997.

[Sieker 2010] Bernd Manfred Sieker, Systemanforderungsanalyse von Bahnbetriebsverfahren mit Hilfe der Ontological-Hazard-Analysis am Beispiel des Zugleitbetriebs nach FV-NE (System Requirements Analysis of Railway Operations with Ontological Hazard Analysis: An Example of Train Dispatching in German Non-State-Railway Law). Doctoral Dissertation, Faculty of Technology, University of Bielefeld, 2010. Available through http://bieson.ub.uni-bielefeld.de/volltexte/2010/1782/

[Stadlmann 2004] Burkhard Stadlmann and Helmut Zwirchmayer, Einfaches Zugleitsystem für Regionalstrecken, Signal+Draht 06/2004.

[Stuphorn&Sieker&Ladkin 2009] Jörn Stuphorn, Bernd Sieker and Peter Bernard Ladkin, Dependable Risk Analysis for Systems with E/E/PE Components: Two Case Studies, in Chris Dale and Tom Anderson (eds.) Safety-Critical Systems: Problems, Process and Practice, Proceedings of the Seventeenth Safety-Critical Systems Symposium, Brighton, UK, 3-5 February 2009, Springer-Verlag London, 2009. Also available at http://www.rvs.uni-bielefeld.de/publications/Papers/StupSiekLadSSS09.pdf