In Y. Lakhnech & S. Yovine (Eds.): FORMATS/FTRTFT 2004, LNCS 3253, pp. 167–182, Springer-Verlag, 2004

A Unified Fault-Tolerance Protocol*

Paul Miner[1], Alfons Geser[2], Lee Pike[1], and Jeffrey Maddalon[1]

[1] Mail Stop 130, NASA Langley Research Center, Hampton, VA 23681-2199, USA
{paul.s.miner, lee.s.pike, j.m.maddalon}@nasa.gov
[2] National Institute of Aerospace, 144 Research Drive, Hampton, VA 23666, USA
geser@nianet.org

Abstract. Davies and Wakerly show that Byzantine fault tolerance can be achieved by a cascade of broadcasts and middle value select functions. We present an extension of the Davies and Wakerly protocol, the *unified protocol*, and its proof of correctness. We prove that it satisfies validity and agreement properties for communication of exact values. We then introduce bounded communication error into the model. Inexact communication is inherent for clock synchronization protocols. We prove that validity and agreement properties hold for inexact communication, and that exact communication is a special case. As a running example, we illustrate the unified protocol using the SPIDER family of fault-tolerant architectures. In particular we demonstrate that the SPIDER interactive consistency, distributed diagnosis, and clock synchronization protocols are instances of the unified protocol.

Keywords: fault tolerance, protocol, SPIDER, Byzantine, reliability, Diagnosis, Interactive Consistency.

1 Introduction

Safety-critical real-time applications rely on basic fault-tolerant services such as interactive consistency (IC), clock synchronization (CS), and distributed diagnosis (DD, also called group membership). These services are usually rendered by distinct protocols that are designed, implemented, and validated separately. Examples of systems that provide these services are SAFEbus [HD92], TTA [Kop97], and MAFT [KWFT88]. Rushby presents an overview of how several architectures realize these fundamental services [Rus03].

Davies and Wakerly, in their ground-breaking paper [DW78], observed that Byzantine fault tolerance can be achieved through a cascade of middle value select functions. This is true when exact values are communicated, such as the payload messages in IC or the accusations in DD. It is also true when inexact values are communicated. By inexact values we mean values that range over

* This work was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046 while the second author was in residence at the National Institute of Aerospace, Hampton, VA 23666, USA.

the real numbers that may change by a bounded error during communication. Timing values for CS or analog sensor values are typical examples. Correct operation of a system crucially depends on both exact and inexact communication satisfying suitable validity and agreement properties.

We introduce a generalization and extension of the Davies and Wakerly protocol, which we call the *unified protocol*. Instances of this general protocol provide the core set of fault-tolerant services. We model the unified protocol formally and prove validity and agreement results for both exact and inexact data, under suitable fault assumptions. The exact case is precisely the inexact case with zero accumulated error. We then demonstrate how the unified protocol can be used as a basis for the IC, DD, and CS protocols for the SPIDER fault-tolerant architecture [MMTP02]. We have verified the unified protocol using PVS [ORSvH95], a semi-automated theorem-proving system developed at SRI. The PVS proof files are available on the web [SPI].

The original contributions of this paper include a formally verified generalization of the Davies and Wakerly protocol, adapted to exploit diagnostic information in the context of a hybrid fault model. In addition, we hope to rekindle interest in Davies and Wakerly's results, which provide an effective approach for Byzantine fault tolerance for real-time embedded applications.

The structure of this paper is as follows. Section 2 presents the unified protocol. Section 3 presents the assumptions and requirements for the protocols described in this paper. Section 4 presents the analysis of the protocol for exact communication, and then illustrates how the SPIDER IC and DD protocols are instances of the unified protocol. Section 5 presents the analysis when the communication can introduce error, then demonstrates how the SPIDER CS protocol is an instance of the unified protocol.

2 The Unified Protocol

The unified protocol is a multiple stage protocol which is constructed from a single basic operation: a middle value select. In this section, we describe the middle value select function and then present the unified protocol using it. We conclude with a mapping of the unified protocol to the SPIDER fault-tolerant architecture.

A *distributed system* is modeled as a graph with directed edges. Vertices are called *nodes* and directed edges are called *links*. We call s the *source* node, and d the *destination* node of the link (s, d). A *communication stage* is a set of source nodes, a set of destination nodes, and a set of of links between them. The absence of a link is modeled conservatively as a link fault. We allow both nodes and links to fail. However, we abstractly model link failures as failures of the source node [PMMG04].

2.1 Notation

We use i or j to refer to an arbitrary stage and k to refer to the total number of stages. In the first stage of a k-stage protocol, each member of the set N^0 of nodes broadcasts to all members of the set N^1 of nodes; in the second stage, each member of N^1 broadcasts to all members of N^2, and so forth, up through the k^{th} stage. Let us now fix an arbitrary stage $i+1$ for $0 \leq i < k$. The set of source nodes is N^i, and the set of destination nodes is N^{i+1}. We use $s, s_1, s_2, \ldots \in N^i$ to denote source nodes and $d, d_1, d_2, \ldots \in N^{i+1}$ for destination nodes. When we refer to a node without refering to a communication stage, we use $n \in N^i$. In the trivial example of a 0-stage protocol, no communication takes place and N^0 is the only set of nodes.

Now we turn our attention to the values that are transfered at each stage. We model payload data using real numbers. We augment the set of reals with certain special values to indicate error conditions. Specifically we define a type T by

$$T = \{\mathit{receive\_error}\} \cup \{\mathit{source\_error}^i \mid i \in \mathbb{N}\} \cup \mathbb{R}.$$

Let $v^i(s) \in T$ denote the value that $s \in N^i$ intends to broadcast in stage $i+1$. After communication in this stage, each destination d has a vector of values v^i_d, such that $v^i_d(s)$ is d's estimate of $v^i(s)$. If the message that d receives from s is obviously incorrect (for example, it does not arrive within the expected window or fails a cyclical redundancy check), then $v^i_d(s) = \mathit{receive\_error}$. The value $\mathit{source\_error}^i$ is a special message that is used to report the total absence of credible sources in stage $i+1$.

2.2 Middle Value Select

The main computation during the execution of a single stage of the protocol is a middle value select voting algorithm. This algorithm chooses the middle value from the vector of received values, v^i_d. For the data type T, we extend the natural order on the reals by the relations:

- $\mathit{receive\_error} < \mathit{source\_error}^0$,
- $\mathit{source\_error}^i < \mathit{source\_error}^j$ if $i < j$,
- $\mathit{source\_error}^i < x$ for all $x \in \mathbb{R}$.

Values from sources that are known to be faulty can be excluded from consideration. For this purpose, we define the *filtered eligible sources*, F^i_d, to be the set of sources whose values are included in the vote computed by node d. If the cardinality of F^i_d is even, any value between the two middle eligible values is an acceptable result, provided that all good nodes implement the same selection function. Let $\mathit{mvs}(F^i_d, v^i_d)$ denote the middle value of the received values from the filtered eligible sources.

2.3 Protocol

The unified protocol is composed of a cascade of individual communication stages. A k-stage protocol operates on the node sets $N^0, \ldots, N^k$. These sets may or may not be disjoint. For $0 \leq i < k$, the algorithm for stage $i+1$ is shown in Figure 1. Each destination node d maintains a set $E^i_d \subseteq N^i$ of *eligible* sources.

The set E_d^i is based, in part, on d's view of the failure status of the sources. Recognize that because of faults and errors during communication, $v_d^i(s)$ may differ from $v^i(s)$.

```
For stage $i+1$, let $s \in N^i$ and $d \in N^{i+1}$

  Communication: Each source $s$ broadcasts $v^i(s)$ to all destination nodes.
    For each destination $d$, $v_d^i(s)$ denotes the value received from source $s$.
  Computation: Each $d$ computes
    1. $I_d^i = \{s \mid v_d^i(s) = \mathit{receive\_error}\}$
    2. $F_d^i = E_d^i \setminus I_d^i$
    3. $v^{i+1}(d) = \begin{cases} \mathit{source\_error}^i, & \text{if } F_d^i = \emptyset, \\ \mathit{mvs}(F_d^i, v_d^i), & \text{otherwise.} \end{cases}$
```

Fig. 1. Unified Protocol

We assume that all correctly operating nodes share common knowledge of the communication schedule. In order to maintain integrity of the communication schedule, we require that correctly operating nodes be synchronized within a known precision. This synchrony provides a global time reference to manage the system's time-triggered communication. Synchrony is maintained by a CS protocol.

The protocol presented in Figure 1 generalizes the Davies and Wakerly (DW) protocol [DW78]. In the DW protocol, every stage has the same number of nodes. There is no such restriction on the unified protocol. Furthermore, the DW protocol does not use accumulated diagnostic information. At each stage, all nodes vote using identical sets of inputs. In the unified protocol, distinct nodes may compute the vote using nonintersecting vote sets. This capability enables the unified protocol to be analyzed using a weak hybrid fault assumption (see Section 3.4).

2.4 Application: SPIDER

The Scalable Processor-Independent Design for Electromagnetic Resilience (SPIDER) is a family of general-purpose fault-tolerant architectures. The SPIDER is designed at NASA Langley Research Center to support laboratory investigations into various recovery strategies from transient failures caused by electromagnetic effects [MMTP02]. The unified protocol is used in SPIDER to implement the IC, CS, and DD protocols. One instance of the SPIDER architecture consists of several Processing Elements (PE) communicating over a Reliable Optical Bus (ROBUS). All application-level functions take place on the PEs. To the PEs, the ROBUS operates as a Time Division Multiple Access (TDMA) broadcast bus.

The topology of the ROBUS is depicted in Figure 2. There are two types of nodes internal to the ROBUS. The Bus Interface Units (BIU) provide the only interface to the PEs. The Redundancy Management Units (RMU) provide the necessary replication for fault tolerance. There is no direct link between any pair of BIUs nor any pair of RMUs.

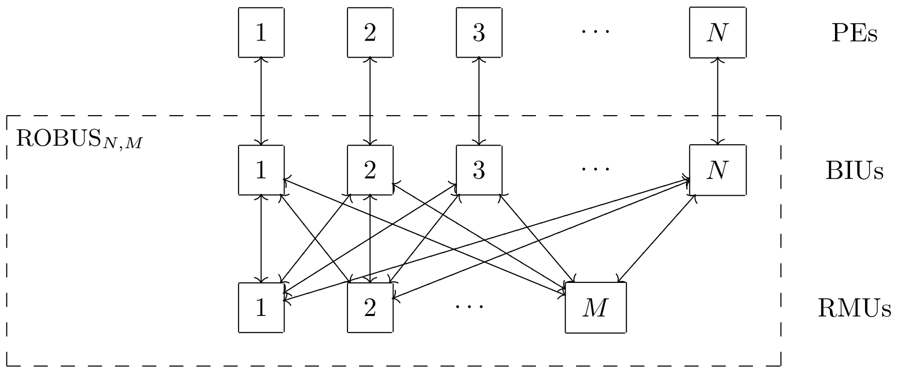

Fig. 2. ROBUS architecture

The primary uses of the unified protocol in the ROBUS are as 2- or 3-stage protocol instances. In a 2-stage instance, three sets of nodes are involved: N^0, N^1 and N^2. For our subsequent discussions of the ROBUS protocols, N^0 corresponds to a subset of the BIUs, N^1 corresponds to the RMUs, and N^2 corresponds to the BIUs. Communication is initiated from the BIUs (using information from their attached PE) who send their values to the RMUs. The RMUs apply the middle value select and send their results back to the BIUs. The BIUs then apply another middle value select and forward the result to the PEs. Provided the system fault assumptions are maintained, the unified protocol allows the ROBUS to provide strong guarantees about the timeliness and correctness of the communication between the various PEs.

3 Protocol Analysis

In this section, we explain the properties the unified protocol must satisfy: validity and agreement. After a description of the fault model, i.e., the covered kinds of faults, we define a fault assumption which constrains the number of faults of each kind. In the succeeding sections, we prove that the correctness conditions hold under this fault assumption.

3.1 Correctness Conditions

The unified protocol solves both the distributed consensus problem and the approximate agreement problem, as defined in [Lyn96]. The IC and DD protocols solve specific instances of the distributed consensus problem, and the CS protocol solves a specific instance of the approximate agreement problem. Validity, agreement, and termination conditions are specified for each kind of problem. The unified protocol obviously terminates for a finite number of stages, so we do not formally state or prove this condition.

Distributed Consensus Properties

Validity If all nonfaulty processes start with the same initial value $v \in V$, then v is the only possible decision value for nonfaulty processes.

Agreement No two nonfaulty processes decide on different values.

Approximate Agreement Properties

Validity Any decision value for a nonfaulty process is within the range of the initial values of the nonfaulty processes.

Agreement The decision values of any pair of nonfaulty processes are within ε of each other.

3.2 Fault Classification

Faults are classified according to the effect they have on the nodes of the system. We use a hybrid fault model from Thambidurai and Park [TP88] with one modification: benign nodes can sometimes behave as good nodes. The particular advantage of this modification is that many intermittent faults are now counted as benign, whence they are easy to mask. The nodes of the system are classified as follows:

Good Each good node behaves according to specification; that is, it always sends valid messages.

Benign Each benign faulty node either sends detectably incorrect messages to every receiver, or sends valid messages to every receiver.

Symmetric A symmetric faulty node may send arbitrary messages, but each receiver receives the same message.

Asymmetric An asymmetric (Byzantine) faulty node may send arbitrary messages that may differ for the various receivers.

A node that is not good is called *faulty.* A node is classified according to its worst error manifestation during the classification period. For example, it is possible for an asymmetric faulty node to behave in a manner that is observationally indistinguishable from a good node at times during this period. These classifications form a "behavioral hierarchy" such that benign nodes can behave as if they are good; symmetric nodes can behave as if they are benign or good, etc. We let G, B, S, and A denote the sets of good, benign, symmetric, and asymmetric nodes, respectively.

Good nodes always provide valid messages. Similarly, benign faulty nodes never provide misleading information. We define a set of nodes C, such that the worst case error manifestation of a source in C is ommissive.[1] That is, a node in C can send a valid message or an obviously incorrect message, but can never communicate an invalid message. From the definitions above, we know that $G \cup B \subseteq C$.

We attribute all faults to the communication, i.e., we assume that the processing of values by destination nodes is fault-free. We have described the rationale for this abstraction in [PMMG04].

For the analyses presented in sections 4 and 5, we introduce the following definitions and supporting facts. Unless we explicitly state otherwise, we assume $C \cap N^i \neq \emptyset$.

$$v^i_{\text{max}} \overset{\text{df}}{=} \max(\{v^i(n) \mid n \in C \cap N^i\})$$

$$v^i_{\text{min}} \overset{\text{df}}{=} \min(\{v^i(n) \mid n \in C \cap N^i\})$$

Lemma 1. *For all $n \in C \cap N^i$, if $v^i_{\text{min}} \in \mathbb{R}$, then $v^i(n) \in \mathbb{R}$.*

Lemma 2. *For all $s \in C \cap N^i$ and $d \in C \cap N^j$, if $v^i_{\text{min}}, v^j_{\text{min}} \in \mathbb{R}$, then*

$$|v^i(s) - v^j(d)| \leq \max(v^i_{\text{max}} - v^j_{\text{min}}, v^j_{\text{max}} - v^i_{\text{min}}) \ .$$

3.3 Eligibility Assumptions

In order to have a basis for agreement, we require that the sets of eligible sources differ only with respect to asymmetric sources.

Let $\mathcal{X}$ be a family of sets of nodes. We say that $\mathcal{X}$ satisfies the Eligible Sources Property if all its members differ only in asymmetric nodes.

Definition 1 (Eligible Sources Property (ESP)).

$$\mathit{ESP}(\mathcal{X}) \overset{\text{df}}{=} \forall X_1, X_2 \in \mathcal{X} : \ n \notin A \implies (n \in X_1 \iff n \in X_2) \ .$$

Let I^i_d, F^i_d be computed as in Figure 1. The families $\mathcal{E}^i, \mathcal{I}^i, \mathcal{F}^i, 0 \leq i < k$ of sets of *eligible sources*, *ignored sources*, and *filtered eligible sources* are respectively defined as follows:

- $\mathcal{E}^i \overset{\text{df}}{=} \{E^i_d \mid d \in N^{i+1}\}$,
- $\mathcal{I}^i \overset{\text{df}}{=} \{I^i_d \mid d \in N^{i+1}\}$,
- $\mathcal{F}^i \overset{\text{df}}{=} \{E^i_d \setminus I^i_d \mid d \in N^{i+1}\}$.

By definition, the filtered eligible sources inherit the Eligible Sources Property from their constituents:

Lemma 3. *If $\mathit{ESP}(\mathcal{E}^i)$ and $\mathit{ESP}(\mathcal{I}^i)$, then $\mathit{ESP}(\mathcal{F}^i)$.*

We expect that $\mathcal{E}^i$ is derived from accumulated knowledge about the $s \in N^i$, such that $\mathit{ESP}(\mathcal{E}^i)$. In addition, we expect that the models of communication be analyzed to ensure $\mathit{ESP}(\mathcal{I}^i)$ for all i. The property $\mathit{ESP}(\mathcal{F}^i)$ can then be deduced by Lemma 3.

[1] Azadmanesh and Kieckhafer [AK00] introduce the notion of a *strictly ommissive asymmetric* faulty node. In future work, we expect to extend our fault model to include this additional classification.

3.4 Fault Assumption

Nodes can exhibit incorrect behavior; that is, they can fail. We require an independence of failure between nodes. Moreover we assume that a certain minimum number of nodes are operating correctly. Engineering design and analysis has to guarantee the satisfaction of these assumptions to a specified probability. A DD protocol provides mechanisms that can increase the probability that a sufficient number of nodes are operating correctly [LMK04].

Our fault assumption contains two clauses. Each clause is an assumption used to guarantee that validity and agreement hold. Agreement is established using two different fault assumptions: *agreement propagation* and *agreement generation*. We name the clauses after the proofs in which they play a role.

The first clause is called the *Validity and Propagation Fault Assumption* (VPFA). It states that for each destination and each stage between j and k, the majority of eligible, non-benign nodes are good. Formally,

Definition 2 (Validity and Propagation Fault Assumption (VPFA)).

$$\mathit{VPFA}(j,k) \overset{\text{df}}{=} \forall i : \ j \le i < k \implies \forall d \in N^{i+1} : \ 2|G \cap E^i_d| > |E^i_d \setminus B|.$$

The second clause is called the *Agreement Generation Fault Assumption* (AGFA). It states that some stage between j and k is free of asymmetric, eligible nodes, and that the subsequent stages satisfy the VPFA. Formally,

Definition 3 (Agreement Generation Fault Assumption(AGFA)).

$$\mathit{AGFA}(j,k) \overset{\text{df}}{=} \exists i : j \le i < k \wedge \mathit{ESP}(\mathcal{E}^i) \wedge \mathit{VPFA}(i+1,k) \wedge \forall d \in N^{i+1} : |A \cap E^i_d| = 0 \ .$$

We have the following supporting lemmas:

Lemma 4. *For $d \in N^{i+1}$, if $2|G \cap E^i_d| > |E^i_d \setminus B|$, then $2|C \cap F^i_d| > |F^i_d|$.*

Lemma 5. *For $d_1, d_2 \in N^{i+1}$, if $|A \cap E^i_{d_1}| = |A \cap E^i_{d_2}| = 0$, $\mathit{ESP}(\mathcal{E}^i)$, and $\mathit{ESP}(\mathcal{I}^i)$, then $F^i_{d_1} = F^i_{d_2}$.*

3.5 Application: SPIDER

For the ROBUS architecture described in Section 2.4, we let N^{2i} denote the BIUs and N^{2i+1} denote the RMUs, for any k-stage SPIDER protocol and $0 \le i < k$. For $k \ge 2$, the SPIDER Maximum Fault Assumption is $\mathit{VPFA}(j, j+k) \wedge \mathit{AGFA}(j, j+k)$. This is equivalent to the following restatement of the SPIDER Maximum Fault Assumption [GM03]:

1. $2|G \cap E_r| > |E_r \setminus B|$ for all RMUs r, and
2. $2|G \cap E_b| > |E_b \setminus B|$ for all BIUs b, and
3. $|A \cap E_r| = 0$ for all RMUs r, or $|A \cap E_b| = 0$ for all BIUs b.

4 Exact Agreement

In this section, we analyze the unified protocol assuming exact communications. We prove validity, agreement propagation and agreement generation under two assumptions on the communication. This framework is a special case of inexact communication, addressed in the next section. However, this special case is simpler, so we present it first.

4.1 A Model of Exact Communication

For exact communication we assume that destinations receive exactly the messages sent by good sources, that messages from benign faulty sources are either correct or ignored, and that all destinations receive exactly the same messages from non-asymmetric sources.

More formally, we assume the following properties for the communication step in stage $i + 1$:

Assumption 1 *For all $s \in C \cap N^i$ and $d \in N^{i+1}$,*

– *$s \notin G$ and $v_d^i(s) = receive\_error$, or*
– *$v_d^i(s) = v^i(s)$.*

Assumption 2 *For all $s \in N^i \setminus A$ and $d_1, d_2 \in N^{i+1}$, $v_{d_1}^i(s) = v_{d_2}^i(s)$.*

These assumptions define an implementation requirement for the communication subsystem for any consensus protocol based on exact communication. The assumptions were constructed to ensure $ESP(\mathcal{I}^i)$.

4.2 Exact Agreement Results

In this section, we present the properties of the k-stage protocol presented in Section 2.3 using the communication assumptions presented in Section 4.1.

Theorem 1 (Upper Validity). *If $VPFA(j, j + k)$, then $v_{\max}^{j+k} \leq v_{\max}^{j}$.*

Proof. By induction on k.

The base case, $k = 0$, is trivial, so assume $k > 0$. By the induction hypothesis, we know that $v_{\max}^{j+k-1} \leq v_{\max}^{j}$. It remains to show that $v_{\max}^{j+k} \leq v_{\max}^{j+k-1}$. Choose $d \in C \cap N^{j+k}$ such that $v^{j+k}(d) = v_{\max}^{j+k}$. By $VPFA(j, j + k)$, we know that $2|G \cap E_d^{j+k-1}| > |E_d^{j+k-1} \setminus B|$. By Lemma 4, we know that $2|C \cap F_d^{j+k-1}| > |F_d^{j+k-1}|$. The pigeonhole principle ensures that there is an $s \in C \cap F_d^{j+k-1}$ such that $v^{j+k}(d) \leq v_d^{j+k-1}(s)$. Assumption 1 ensures that $v_d^{j+k-1}(s) = v^{j+k-1}(s)$. The definition of $v_{\max}^{j+k-1}$ ensures that $v^{j+k-1}(s) \leq v_{\max}^{j+k-1}$. □

Theorem 2 (Lower Validity). *If $VPFA(j, j + k)$, then $v_{\min}^{j} \leq v_{\min}^{j+k}$.*

Proof. Similar to the proof of Theorem 1. □

The following corollaries are direct consequences of Theorems 1 and 2.

Corollary 1 (Consensus Validity). *If $VPFA(j, j+k)$ and $v^j_{\max} = v^j_{\min} = v$, then $v^{j+k}_{\max} = v^{j+k}_{\min} = v$.*

Corollary 2 (Master-Slave). *If $VPFA(j, j+k)$, $v^j_{\min} \in \mathbb{R}$, and $v^j_{\max} - v^j_{\min} \leq \Delta$, then for all $s \in C \cap N^j$, $d \in C \cap N^{j+k}$ we have $|v^j(s) - v^{j+k}(d)| \leq \Delta$.*

Corollary 3 (Agreement Propagation). *If $VPFA(j, j+k)$, $v^j_{\min} \in \mathbb{R}$, and $v^j_{\max} - v^j_{\min} \leq \Delta$, then $v^{j+k}_{\max} - v^{j+k}_{\min} \leq \Delta$.*

Corollary 3 ensures that agreement among receivers will be at least as good as the agreement among the sources. However, it does not provide assurance that exact agreement will ever be achieved. Specifically, the presence of an eligible asymmetric faulty node in every stage can prevent exact agreement.

Theorem 3 (Agreement Generation). *If $AGFA(j, j+k)$ then $v^{j+k}_{\max} = v^{j+k}_{\min}$.*

Proof. By $AGFA(j, j+k)$, there is a $i < k$ such that $VPFA(j+i+1, j+k)$, and $|A \cap E^{j+i}_d| = 0$, for all $d \in C \cap N^{j+i+1}$. By Lemma 5, we know that $F^{j+i}_{d_1} = F^{j+i}_{d_2} = F$, for $d_1, d_2 \in C \cap N^{j+i+1}$. Since $F \subseteq N^{j+i} \setminus A$, Assumption 2 ensures that $v^{j+i}_{d_1}(s) = v^{j+i}_{d_2}(s)$ for all $s \in F$. Thus, $v^{j+i+1}_{\max} = v^{j+i+1}_{\min}$. From Corollary 1, we get $v^{j+k}_{\max} = v^{j+k}_{\min}$. □

4.3 Application: SPIDER Interactive Consistency

The SPIDER interactive consistency protocol [MMTP02] is an instance of the 2-stage unified protocol. The properties we require of interactive consistency are the distributed consensus properties as defined in Section 3.1.

Let s be some BIU that intends to send a value to all other BIUs. Next let v^2 be computed using a 2-stage exchange with $N^0 = \{s\}$, N^1 the set of all RMUs, and N^2 the set of all BIUs. The interactive consistency protocol for $d \in N^2$ is:

$$ic(d) = \begin{cases} v^2(d), & \text{if } v^2(d) = majority(F^1_d, v^1_d), \\ no\_majority, & \text{otherwise,} \end{cases}$$

where *no_majority* is a distinguished constant.

Theorem 4 (IC validity). *If $s \in G$ and $VPFA(0, 2)$, then $ic(d) = v(s)$.*

Proof. Since we have a singleton source set, $v^0_{\min} = v^0_{\max} = v(s)$. The result follows directly from Corollary 1 for $k = 2$. □

Theorem 5 (IC agreement). *If $AGFA(0, 2)$, then $ic(d_1) = ic(d_2)$.*

Proof. The result follows from Theorem 3 for $k = 2$. □

In addition, we are able to gather some diagnostic information about the source BIU, s. The following corollaries follow from Theorems 4 and 5, respectively.

Corollary 4. *If* $\mathit{VPFA}(0,2)$ *and* $ic(d) = \mathit{source\_error}^0$*, then* $s \notin G$*.*

Corollary 5. *If* $\mathit{AGFA}(0,2)$ *and* $ic(d) = \mathit{no\_majority}$*, then* $s \in A$*.*

4.4 Application: SPIDER Distributed Diagnosis

Distributed on-line diagnosis consists of two main parts. First, nodes accumulate evidence of faulty behavior by other nodes. Second, this local evidence must be reliably distributed to allow for global decisions.

There are several mechanisms for accumulating evidence of faulty behavior. There are indirect mechanisms, such as those provided by Corollaries 4 and 5. There are also several direct accusation mechanisms. These include communication resulting in *receive_error* and disagreement with results during an agreement propagation stage.

We let $D_n(\mathit{def}) \in \mathbb{N}$ represent node n's accumulated evidence against defendant def. If $D_n(\mathit{def}) = 0$, then n has no recent evidence of faulty behavior by def. A larger $D_n(\mathit{def})$ indicates more severe misbehavior by def.

We require that a good node can never make a false accusation. Formally, if $n \in C$ and $D_n(\mathit{def}) > 0$, then $\mathit{def} \notin G$. The role of the distributed diagnosis protocol is to achieve global consensus from locally gathered accusations. Strictly speaking, SPIDER does not require a distributed diagnosis protocol. It is possible for the locally gathered accusations to satisfy the required assumptions. However, by periodically exchanging diagnostic information, we can remove accumulated disagreement caused by asymmetric faults. This can increase the probability that our fault assumptions are true, thus increasing the predicted reliability of the system [LMK04].

The SPIDER DD protocol is a 3-stage instance of the unified protocol. The first two stages are to assure agreement among the BIUs. The third stage is to propagate this consensus diagnostic information to the RMUs.

Let $v^0(b) = D_b(\mathit{def})$, and v^2 and v^3 be computed using the 3-stage unified protocol. Thus, $v^0_{\max}$ is the most severe correct local accusation against def and $v^0_{\min}$ is the least severe accusation.

Theorem 6 (DD Validity). *If* $\mathit{VPFA}(0,3)$*, then for* $b \in C \cap N^2$*,* $r \in C \cap N^3$*,*

- *If* $v^2(b) > 0$*, then* $\mathit{def} \notin G$*.*
- *If* $v^3(r) > 0$*, then* $\mathit{def} \notin G$*.*

Proof. Both clauses are direct consequences of Theorems 1 and 2. □

Theorem 7 (DD Agreement). *If* $\mathit{AGFA}(0,3)$ *then for* $b \in C \cap N^2$*,* $r \in C \cap N^3$*,* $v^2(b) = v^3(r)$*.*

Proof. Follows from Theorem 3 and Corollary 1. □

The preceding results ensure consensus based on the BIUs local accusations against *def*. A similar protocol beginning with the RMUs ensures consensus based on the RMUs accusations against *def*. The maximum of the two results also satisfies validity and agreement.

5 Approximate Agreement

In this section we generalize the exact communication assumptions to accommodate error introduced in the communication phase. The results in Section 4 are all special cases of the results introduced in this section.

Analog information can be understood as a real valued, uniformly continuous function of time [Ros68]. Uniform continuity roughly means that the rate of change is bounded. For processing in a digital system, a digital approximation of the function value at a given moment is determined: the function is *sampled.* There are various sources of imprecision. For instance, the actual time of sampling may vary or the sampled value may be superposed with noise. The purpose of the inexact protocol is to reliably communicate values that may vary and may be further distorted during communication.

5.1 A Model of Inexact Communication

We model communication as in the exact case, but add terms representing the inherent imprecision of broadcasting inexact information. The error terms $\varepsilon_{\mathrm{l}}, \varepsilon_{\mathrm{u}}$, and ε are nonnegative reals. We define $\varepsilon \overset{\mathrm{df}}{=} \varepsilon_{\mathrm{l}} + \varepsilon_{\mathrm{u}}$.

We assume that messages from good nodes are correctly received within a known error tolerance, that messages from benign faulty nodes are either ignored or are correctly received within a known tolerance, and that only asymmetric nodes may introduce disagreement beyond ε in the communication phase. We allow the communication error bounds, ε_{l} and ε_{u}, to differ as the error may be biased. Formally, the assumptions for stage $i+1$ are:

Assumption 3 *For all $s \in C \cap N^i$ and $d \in N^{i+1}$:*

- $s \notin G$ *and* $v^i_d(s) = \mathit{receive\_error}$,
- $\mathit{receive\_error} < v^i_d(s) = v^i(s) < \mathit{source\_error}^i$, *or*
- $v^i(s) \in \mathbb{R}$ *and* $v^i(s) - \varepsilon_{\mathrm{l}} \leq v^i_d(s) \leq v^i(s) + \varepsilon_{\mathrm{u}}$.

Assumption 4 *If $s \in N^i \setminus A$, then for $d_1, d_2 \in N^{i+1}$:*

- $v^i_{d_1}(s) = v^i_{d_2}(s) < \mathit{source\_error}^i$, *or*
- $v^i_{d_1}(s), v^i_{d_2}(s) \in \mathbb{R}$ *and* $|v^i_{d_1}(s) - v^i_{d_2}(s)| \leq \varepsilon$.

When $\varepsilon = \varepsilon_{\mathrm{l}} = \varepsilon_{\mathrm{u}} = 0$, Assumptions 3 and 4 reduce to Assumptions 1 and 2. Thus, exact communication is a special case of inexact communication.

5.2 Approximate Agreement Results

The following results generalize the results from Section 4.2, by introducing the effects of bounded errors. By requiring $v^j_{\min} \in \mathbb{R}$ for these results, we avoid the clutter of defining arithmetic involving *source_error*. These results can be extended to handle such special cases.

Theorem 8 (Inexact Upper Validity). *If VPFA$(j, j+k)$ and $v^j_{\min} \in \mathbb{R}$, then*

$$v^{j+k}_{\max} \leq v^j_{\max} + k\varepsilon_{\mathrm{u}}.$$

Proof. Similar to proof of Theorem 1. □

Theorem 9 (Inexact Lower Validity). *If VPFA$(j, j+k)$ and $v^j_{\min} \in \mathbb{R}$, then*

$$v^j_{\min} - k\varepsilon_{\mathrm{l}} \leq v^{j+k}_{\min}.$$

Proof. Similar to proof of Theorem 1. □

Corollary 6 (Inexact Master-Slave). *If VPFA$(j, j+k)$, $v^j_{\min} \in \mathbb{R}$, and $v^j_{\max} - v^j_{\min} \leq \Delta$, then for all $s \in C \cap N^j$, $d \in C \cap N^{j+k}$ we have*

$$|v^j(s) - v^{j+k}(d)| \leq \Delta + \max(k\varepsilon_{\mathrm{l}}, k\varepsilon_{\mathrm{u}}).$$

Proof. Follows directly from Lemma 2 and Theorems 8 and 9. □

Corollary 7 (Inexact Agreement Propagation). *If VPFA$(j, j+k)$, $v^j_{\min} \in \mathbb{R}$, and $v^j_{\max} - v^j_{\min} \leq \Delta$, then*

$$v^{j+k}_{\max} - v^{j+k}_{\min} \leq \Delta + k\varepsilon\,.$$

Proof. From Theorems 8 and 9, we have $v^{j+k}_{\max} - v^{j+k}_{\min} \leq (v^j_{\max} + k\varepsilon_{\mathrm{u}}) - (v^j_{\min} - k\varepsilon_{\mathrm{l}}) \leq \Delta + k\varepsilon$. □

Theorem 10 (Inexact Agreement Generation). *If AGFA$(j, j+k)$ and $v^{j+k}_{\min} \in \mathbb{R}$, then*

$$v^{j+k}_{\max} - v^{j+k}_{\min} \leq k\varepsilon.$$

Proof. Similar to proof of Theorem 3. □

5.3 Application: SPIDER synchronization protocol

A clock is formalized as a function from clock time to real time. Clocks distributed in a system need to be re-synchronized periodically in order to prevent them from drifting too far apart. The two goals of synchronization are:

Accuracy All good clock readings are within a linear envelope of real time.
Precision At all times, the clock times of all good clocks differ by a bounded amount.

Prior formal models of fault tolerant clock synchronization [SvH98,Min93,Sha92] have established a systematic way to derive accuracy and precision from the following properties:

Accuracy Preservation The resynchronization time of a good clock is within the expected resynchronization times of good clocks, up to an error margin.
Precision Enhancement If the skew of good clocks is within a known bound at the time of protocol execution, then all good clocks are synchronized to within a tighter skew after protocol execution

Below we show how to prove accuracy preservation and precision enhancement, using validity and agreement properties of the unified protocol. The values communicated during this protocol are estimates of the real time that nodes should reset their clocks for the next period.

Let $c_n(T^{p+1}) \in \mathbb{R}$ denote the real time that node n expects to begin synchronization period $p+1$. Let $c'_n(T^{p+1})$ denote the real time that node n actually begins period $p+1$. Put another way, c_n models n's clock before resynchronization, and c'_n models n's clock after resynchronization.

The SPIDER synchronization protocol is a 3-stage instance of the unified protocol. The BIUs are N^0 and N^2, the RMUs are N^1 and N^3. Let $v^0(b_0) = c_{b_0}(T^{p+1})$, for BIU $b_0 \in N^0$. Then, for all $b \in N^2$, $r \in N^3$, define

$$c'_b(T^{p+1}) \overset{\mathrm{df}}{=} v^2(b)$$

$$c'_r(T^{p+1}) \overset{\mathrm{df}}{=} v^3(r)$$

The values ε_{l} and ε_{u} bound the variation of clock readings caused by drift, jitter, and differences in communication delay. Let $c_{\min}(p)$ and $c_{\max}(p)$ denote the minimal and maximal values of all $c_b(T^{p+1})$ such that c_b is a correct BIU clock at round p.

Within the ROBUS, we are principally concerned with the accuracy of the BIUs, as these provide time references for the PEs. If needed, a similar argument can be used to bound the accuracy of the RMUs.

Theorem 11 (BIU Accuracy Preservation). *If VPFA$(0,2)$ holds during synchronization period p, then for all good BIU clocks c'_b:*

$$c_{\min}(p) - 2\varepsilon_{\mathrm{l}} \leq c'_b(T^{p+1}) \leq c_{\max}(p) + 2\varepsilon_{\mathrm{u}}.$$

Proof. Follows immediately from Theorems 8 and 9. □

Precision results are given for the set of BIUs, the set of RMUs, and between the BIUs and RMUs. This last result provides the skew bounds necessary to reliably communicate within the ROBUS.

Theorem 12 (Precision Enhancement). *If AGFA$(0,3)$ then*

1. $|c'_{b_1}(T^{p+1}) - c'_{b_2}(T^{p+1})| \leq 2\varepsilon,$
2. $|c'_{r_1}(T^{p+1}) - c'_{r_2}(T^{p+1})| \leq 2\varepsilon,$

3. $|c'_b(T^{p+1}) - c'_r(T^{p+1})| \leq 2\varepsilon + \max(\varepsilon_{\mathrm{l}}, \varepsilon_{\mathrm{u}})$.

Proof. Clauses 1 and 2 each follow from Theorem 10 (Clause 1 using $AGFA(0, 2)$, and Clause 2 using $AGFA(1, 3)$). Clause 3 is a consequence of Clause 1 and Corollary 6. □

6 Concluding Remarks

We introduce a formal model of an extension of the Davies and Wakerly protocol, called the *unified protocol*. We prove that under a weak hybrid fault assumption, the unified protocol satisfies validity and agreement, both for exact and inexact communication. Three fundamental fault-tolerant protocols are shown to be instances of the unified protocol.

With the unified protocol, the analysis of fault-tolerance properties can be restricted to one general protocol. In this way, the unified protocol provides a useful abstraction layer: the analysis of the fault tolerance is not complicated by specific concerns of individual protocols. For the SPIDER architecture, this has resulted in simpler specifications. This in turn yields a simpler implementation and more transparent treatment of the separate functions. Although we have not yet performed the analysis, we believe that the SPIDER transient recovery and restart protocols are also instances of the unified protocol.

The unified protocol is flexible and can be adapted to other fault tolerant applications. In particular, it should be possible to adapt some of the arguments provided by Caspi and Salem [CS00] to bound the effects of computation error for locally computed control functions between communication stages.

In addition, we expect that our results may be extended to analyze other architectures. Similar arguments may be constructed under weaker fault assumptions. In particular, we intend to explore the benefits of extending our analysis to incorporate the strictly ommissive asymmetric classification introduced by Azadmanesh and Kieckhafer [AK00]. We also plan to explore a wider range of fault tolerant averaging functions within our PVS framework. Ultimately, we intend to provide a PVS library of reusable fault tolerance results.

References

[AK00] Mohammad H. Azadmanesh and Roger M. Kieckhafer. Exploiting omissive faults in synchronous approximate agreement. *IEEE Transactions on Computers*, 49(10):1031–1042, 2000.

[CS00] Paul Caspi and Rym Salem. Threshold and bounded-delay voting in critical control systems. In Mathai Joseph, editor, *FTRTFT*, volume 1926 of *Lecture Notes in Computer Science*, pages 70–81. Springer, 2000.

[DW78] Daniel Davies and John F. Wakerly. Synchronization and matching in redundant systems. *IEEE Transactions on Computers*, 27(6):531–539, June 1978.

[GM03] Alfons Geser and Paul S. Miner. A new on-line diagnosis protocol for the SPIDER family of Byzantine fault tolerant architectures. Technical Memorandum NASA/TM-2003-212432, NASA Langley Research Center, Hampton, VA, December 2003. In print.

[HD92] Kenneth Hoyme and Kevin Driscoll. SAFEbus^{TM}. In *11th AIAA/IEEE Digital Avionics Systems Conference*, pages 68–73, Seattle, WA, October 1992.

[Kop97] Hermann Kopetz. *Real-Time Systems*. Kluwer Academic Publishers, 1997.

[KWFT88] R. M. Kieckhafer, C. J. Walter, A. M. Finn, and P. M. Thambidurai. The MAFT architecture for distributed fault tolerance. *IEEE Transactions on Computers*, 37(4):398–405, April 1988.

[LMK04] Elizabeth Latronico, Paul Miner, and Philip Koopman. Quantifying the reliability of proven SPIDER group membership service guarantees. In *Proceedings of the International Conference on Dependable Systems and Networks*, June 2004.

[Lyn96] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[Min93] Paul S. Miner. Verification of fault-tolerant clock synchronization systems. NASA Technical Paper 3349, NASA Langley Research Center, Hampton, VA, November 1993.

[MMTP02] Paul S. Miner, Mahyar Malekpour, and Wilfredo Torres-Pomales. Conceptual design of a Reliable Optical BUS (ROBUS). In *21st AIAA/IEEE Digital Avionics Systems Conference DASC*, Irvine, CA, October 2002.

[ORSvH95] Sam Owre, John Rushby, Natarajan Shankar, and Friedrich von Henke. Formal verification for fault-tolerant architectures: Prolegomena to the design of PVS. *IEEE Transactions on Software Engineering*, 21(2):107–125, February 1995.

[PMMG04] Lee Pike, Jeffrey Maddalon, Paul Miner, and Alfons Geser. Abstractions for fault-tolerant distributed system verification. In *Proceedings of Theorem-Proving in Higher-Order Logics (TPHOLs)*. Theorem Proving in Higher-Order Logics (TPHOLs), 2004. Accepted. Available at `http://shemesh.larc.nasa.gov/fm/spider/spider_pubs.html`.

[Ros68] Maxwell Rosenlicht. *Introduction to Analysis*. Dover Publications, Inc., 1968.

[Rus03] John Rushby. A comparison of bus architectures for safety-critical embedded systems. Technical Report NASA/CR-2003-212161, NASA Langley Research Center, Hampton, VA, March 2003.

[Sha92] Natarajan Shankar. Mechanical verification of a generalized protocol for Byzantine fault-tolerant clock synchronization. In J. Vytopil, editor, *Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 571 of *Lecture Notes in Computer Science*, pages 217–236, Nijmegen, The Netherlands, January 1992. Springer-Verlag.

[SPI] SPIDER homepage, NASA Langley Research Center, Formal Methods Team. Available at `http://shemesh.larc.nasa.gov/fm/spider/`.

[SvH98] Detlef Schwier and Friedrich von Henke. Mechanical verification of clock synchronization algorithms. In Anders P. Ravn and Hans Rischel, editors, *Formal Techniques in Real-Time and Fault-Tolerant Systems*, number 1486 in LNCS, pages 262–271. Springer, September 1998.

[TP88] Philip Thambidurai and You-Keun Park. Interactive consistency with multiple failure modes. In *7th Reliable Distributed Systems Symposium*, pages 93–100, October 1988.