Appears in: *Reliability Engineering & System Safety 92 (2007) 1538–1550.*
Special Issue on Safety, Reliability and Security of Industrial Computer Systems.

Modular Formal Analysis of the Central Guardian in the Time-Triggered Architecture [1]

Holger Pfeifer *, Friedrich W. von Henke

Abtl. Künstliche Intelligenz
Fakultät für Informatik
Universität Ulm
D–89069 Ulm

Abstract

The Time-Triggered Protocol TTP/C constitutes the core of the communication level of the Time-Triggered Architecture for dependable real-time systems. TTP/C ensures consistent data distribution, even in the presence of faults occurring to nodes or the communication channel. However, the protocol mechanisms of TTP/C rely on a rather optimistic fault hypothesis. Therefore, an independent component, the central guardian, employs static knowledge about the system to transform arbitrary node failures into failure modes that are covered by the fault hypothesis.

This paper presents a modular formal analysis of the communication properties of TTP/C based on the guardian approach. Through a hierarchy of formal models, we give a precise description of the arguments that support the desired correctness properties of TTP/C. First, requirements for correct communication are expressed on an abstract level. By stepwise refinement we show both that these abstract requirements are met under the optimistic fault hypothesis, and how the guardian model allows a broader class of node failures to be tolerated.

The models have been developed and mechanically checked using the specification and verification system PVS.

* Corresponding author.
Email addresses: Holger.Pfeifer@uni-ulm.de (Holger Pfeifer), friedrich.von-henke@uni-ulm.de (Friedrich W. von Henke).
[1] This research was supported by the European Commission under the IST project NEXT TTA (IST-2001-32111).

1 Introduction

The Time-Triggered Architecture (TTA) [1–3] is a distributed computer architecture for the implementation of highly dependable real-time systems. In particular, it targets embedded control applications, such as *by-wire* systems in the automotive or aerospace industry [4,5]. For these safety-critical systems fault tolerance is of utmost importance. The Time-Triggered Protocol TTP/C constitutes the core of the communication level of the Time-Triggered Architecture. It furnishes a number of important services, such as atomic broadcast, consistent membership and protection against faulty nodes, that facilitate the development of these kinds of fault-tolerant real-time applications. However, these protocol mechanisms rely on a rather optimistic fault hypothesis and assume that a fault is either a reception fault or a consistent send fault of some node [6]. In order to extend the class of faults that can be tolerated a special hardware component, the so-called *guardian*, is introduced [7]. A guardian is an autonomous unit that protects the shared communication network against faulty behaviour of nodes by supervising their output. The original bus topology of the communication network employed local bus guardians, which were placed between the nodes and the bus. In the more recent star topology, central guardians are used in the hub of each star. The guardian makes use of static knowledge available in a TTA-based system to transform arbitrary node failures into those that are covered by the optimistic fault hypothesis. For example, the time interval during which a given node is allowed to access the shared communication network is statically determined in a TTA system and known *a priori*. The guardian can hence control the correct timing of message transmissions by granting write access to the network only during a node's pre-defined time slot.

The goal of this work is to formally model TTP/C guardians and analyse their fault tolerance properties. In particular, we aim at describing the benefits of the guardians by giving a precise specification of the assumptions on which the derivation of the properties is based. Formal analysis can provide an additional source of confidence in correct behaviour of a system, which is particularly important in the context of safety-critical systems. Several aspects of TTP/C and related protocols have therefore been formally modelled and analysed, including clock synchronisation [8], group membership [9–11], and the startup procedure [12, 13]. A detailed overview of formal analysis work for the Time-Triggered Architecture is given by Rushby [14]. While so far the protocol algorithms of the time-triggered protocol have been the focus of the formal analyses cited above, we concentrate in this paper on the communication properties of TTP/C, thereby complementing and extending previous work.

To describe the behaviour and properties of the communication network and

the guardians we develop various formal models, which are organised in a hierarchical fashion. We start by specifying the desired correctness properties of the communication in an abstract form. Subsequently, in a process of stepwise refinement, more detail is added to this initial abstract model. On the next level of the hierarchy, we consider a TTP/C system without guardians. We show that in this case the strong, optimistic fault hypothesis is necessary to guarantee correct communication. Another model then introduces guardians and specifies their behaviour. At this level we demonstrate that the optimistic assumptions can be relaxed, which leads to a fault hypothesis that covers a broader class of faults.

The development of the models is in the spirit of, and builds on the work on modelling TTP-related aspects that has been carried out previously [8, 10, 15]. Specifically, it continues the use of the PVS specification and verification system [16] to both specify the model and the properties to be verified, and develop formal proofs that the model satisfies the stated properties. Previous work has demonstrated the suitability of PVS for this type of tasks. The formal models that are presented in this paper have been developed, and the proofs of their properties have been mechanically checked, with the PVS system.

The paper is organised as follows. In Section 2 we give a brief overview of the main aspects of the Time-Triggered Architecture. Section 3 describes the structure of the models and motivates their organisation. Details of the components of the formal models are elaborated in Section 4. Finally, we conclude in Section 5.

2 Brief Overview of the Time-Triggered Architecture

In this section we only briefly describe the main aspects of the Time-Triggered Architecture to the extent that is required for this paper. For more detailed presentations we refer to [3, 17, 18].

In a Time-Triggered Architecture system a set of *nodes* are interconnected by a real-time communication system. A node consists of the host computer, which runs the application software, and the communication controller, which accomplishes the time-triggered communication between different nodes. The nodes communicate via replicated shared media, the communication *channels*. There are two common physical interconnection topologies for TTA. Originally, the channels were replicated, passive buses, while in the more recent star topology the nodes are connected to replicated central star couplers, one for each of the two communication channels.

The distinguishing characteristic of time-triggered systems is that all system

activities are initiated by the passage of time [19]. The autonomous TTA communication system periodically executes a time-division multiple access (TDMA) schedule. Thus, access to the communication medium is divided into a series of intervals, called *slots*. Every node exclusively owns certain slots in which it is allowed to send messages via the communication network. The times of the periodic message sending actions are determined *a priori*, that is, at design time of the system. The send and receive instants are contained in a message schedule, the so-called *message descriptor list (MEDL)*. This scheduling table is static and stored at each communication controller. It thus provides common knowledge about message timing to all nodes. A complete cycle during which every node has access to the network exactly once is called a *TDMA round*.

Messages are used as a life-sign of the respective sender, and whenever a node receives a correct frame on at least one of the channels it considers the sender correct. Correctness of a frame is determined by each receiving node according to a set of criteria. A node considers a frame correct if it is well-timed, i. e. arrives within the boundaries of the TDMA slot, the physical signal obeys the line encoding rules, the frame passes a CRC check, and the sender and receiver agree on the distributed protocol state, the so-called *C-state*. One of the desired correctness properties of TTP/C is that all correct nodes always agree on whether or not a message is considered correct.

The Time-Triggered Protocol is designed to provide fault tolerance. In particular, the protocol has to ensure that non-faulty nodes receive consistent data despite the presence of possibly faulty nodes or a faulty communication channel. The provision of fault tolerance is based on a number of assumptions about the types, number, and frequency of faults. Altogether, these assumptions constitute the so-called *fault hypothesis*. The main assumption for the algorithms implemented in TTP/C is that a fault manifests itself as either a reception fault or a consistent send fault of some node [6]. In particular, the TTP/C services rely on transmission faults being consistent. That is, messages must be received correctly by either all non-faulty nodes or none. Moreover, nodes are assumed not to send messages outside their assigned slots. With respect to faults of the communication network, it is assumed that the channels cannot spontaneously create correct messages, and that messages are delivered either with some known bounded delay or never. With regard to the frequency and number of faults, TTP/C assumes that only one node becomes faulty during a TDMA round, and that there is at most one faulty node or one faulty channel at a time.

However, the Time-Triggered Architecture can tolerate a broader class of faults by intensively using the static knowledge that is present in the TDMA schedule. This allows to transform arbitrary failure modes of nodes into either send or receive faults that can be tolerated by the protocol. The guardians, which

are dedicated components of the communication system, monitor the temporal behaviour of the nodes. As the right to access the communication channels is statically determined, the guardians can bar a faulty node from sending a message outside its designated slots. Thus, timing failures of nodes are effectively transformed into send faults.

Moreover, guardians can protect against a particular class of *Byzantine* faults, the so-called *slightly-off-specification* (SOS) faults. A component is called SOS-faulty if it exhibits only marginally faulty behaviour that appears correct to some components, but faulty to others. A slightly-off-specification timing fault could occur if the transmission of a node terminates very close to the end of its scheduled transmission interval; thus, some receivers might accept the message while others might consider it mistimed. Because the duration of a particular transmission is known beforehand, the guardian can prevent such a cut-off scenario. A node must begin its transmission during a pre-defined period of time after the start of its slot, otherwise the guardian would terminate the right to access the communication network. Thus, the guardian can effectively prevent cut-off SOS faults, provided that the transmission interval is chosen long enough to ensure that a transmission fits the interval whenever it is started in time. Specifically, TTP/C guardians protect against SOS faults in the line encoding of frames at the physical layer, SOS timing faults, transmission of data outside the designated sending slots, masquerading of nodes, and transmission of non-agreed critical state information [7].

3 Bird's Eye View of the Formal Models

The overall goal of modelling the communication network is to provide a concise description of the arguments that support the following three main correctness properties of the TTP/C communication:

- *Validity:*
  If a correct node transmits a correct frame, then all correct receivers accept the frame.
- *Agreement:*
  If any correct node accepts a frame, then all correct receivers do.
- *Authenticity:*
  A correct node accepts a frame only if it has been sent by the scheduled sending node of the given slot.

Once these properties are established, they can be exploited in subsequent analyses of protocol algorithms. This is preferable, since it is generally more feasible to base an analysis on properties of a supporting model or theory, rather than on the mere definitions of the model itself.

In order to facilitate the deduction, the formal proofs of these properties are decomposed into a series of smaller steps, and a hierarchy of corresponding models has been developed. Each of the single models focuses on a particular aspect of the communication. Altogether, we have identified the following four suitable model layers:

- General specification of the reception of frames.
- Channels without guardians, requiring a strong fault hypothesis.
- Channels with guardians, requiring only a weaker fault hypothesis.
- Different network topologies: local bus guardians and central guardians.

Each of the models contributes a small step towards proving the desired correctness properties. The steps themselves are each based on a set of assumptions or preconditions. Put in an abstract, and maybe also slightly oversimplified way, in each model layer i one establishes a theorem of the form

$$assumptions_i \Rightarrow properties_i$$

The idea is to design the different models in such a way that the properties on one level establish the assumptions on the next. Ultimately, the models are integrated and the reasoning is combined, yielding a chain of implications of roughly the following kind:

$$\begin{aligned} assumptions_0 \Rightarrow \quad & properties_0 \\ = \quad & \text{or} \quad \Rightarrow \\ assumptions_1 \Rightarrow \quad & properties_1 \\ = \quad & \text{or} \quad \Rightarrow \\ & assumptions_2 \Rightarrow \ldots \Rightarrow properties_f \end{aligned}$$

The final properties, $properties_f$, correspond to the desired main correctness properties of the TTP/C communication as specified above, while the initial assumptions, $assumptions_0$, describe what constitutes the basic fault hypothesis.

We are going to briefly summarise the main aspects of the four model layers. At the bottom, the model describes the reception of frames by the nodes. Here, the various actions that nodes take in order to judge the correctness of the received frame are formalised. This amounts to considering the transmission time and the signal encoding of the frame, and the outcomes of the CRC check and the C-state agreement check, respectively [18]. The main correctness properties of the communication network are then expressed in terms of these notions. The assumptions of this model layer concern requirements about the functionality of the communication channels. In particular, they describe properties of the

frames that a channel transmits, such as signal encoding or delivery times, and reflect the hypothesis about possible faults of the communication network. In essence, this model establishes a proposition that informally reads as follows:

$$general\_channel\_properties \Rightarrow Validity \wedge Agreement \wedge Authenticity \quad (1)$$

On the next level, we model the transmission of frames through channels that are not equipped with guardians. The goal is then to derive the assumptions of the basic model, as covered by the expression *general_channel_properties*. However, in order to do so, a strong hypothesis on the types of possible faults of nodes is necessary. This strong fault hypothesis requires, for instance, that even a faulty node does not send data outside its sending slot, and nodes never send correct frames when they are not scheduled to do so. Using our informal notation, we can sketch the reasoning at this level as follows:

$$strong\_fault\_hypothesis \Rightarrow general\_channel\_properties \quad (2)$$

Guardians are employed to transform arbitrary node faults into faults that are covered by the strong fault model. Thus, the strong fault hypothesis can be replaced with weaker assumptions about the correct behaviour of the guardians. The functionality and the properties of the guardians are formally specified in the third model of the hierarchy, where the following fact is established:

$$weaker\_fault\_hyp. \wedge generic\_guardian \Rightarrow general\_channel\_properties \quad (3)$$

Ideally, we would have liked to demonstrate directly that – together with the guardian properties – the weak form of the fault hypothesis implies the strong one. However, it turned out to be rather challenging to accomplish a formal proof for this fact and hence we had to revert to reasoning according to (3).

The model of the guardians is generic, as it does not, for instance, stipulate the type of guardian to be used in the communication network. The final level of our hierarchy models each of the two typical topologies of a TTP/C network: the bus topology and the star topology. In the former, each node of the network is equipped with its own local bus guardian, one for each channel, while in the latter the guardians are placed into the central star-coupling device of the channels. In this model layer we show that the properties of the guardians are independent from the choice of a particular topology, given that both the local bus guardians and the central guardians implement the same algorithms. Hence, we establish the following facts:

$$local\_bus\_guardian \Rightarrow generic\_guardian \quad (4)$$
$$central\_star\_guardian \Rightarrow generic\_guardian \quad (5)$$

The hierarchic arrangement of the models for the communication network allows for a concise description of the dependencies of the three main correctness properties. At the basic level the fundamental prerequisites are described that are necessary for the desired correctness properties to hold, while the subsequent levels express what must be assumed from the nodes and guardians, respectively, to satisfy these prerequisites. In particular, the treatment precisely explains the benefits of introducing guardians into the communication network.

4 Modular Formal Analysis of TTP/C Communication

In this section we present the main details of the formal models for the communication network according to the hierarchy that has been set out in the previous section. Although the formal models have been developed as a set of PVS modules (i.e., theories), the presentation is in the style of a mathematical transcription of these PVS modules. Similarly, we will explain the essential steps of the major proofs in an informal way; nevertheless, all proofs presented in this section have been developed and mechanically checked using the PVS theorem prover.

4.1 Modelling the Reception of Frames

We start the presentation of our hierarchy of models at the bottom level. This model provides a formalisation of how the communication of TTP/C essentially works. In particular, the actions taken by the nodes when sending or receiving frames are described. In addition, the desired correctness properties of TTP/C communication are stated at this level. The reference points of this formalisation are the TTP/C specification document [18] and the protocol developer's discussion of the fault assumptions in TTP/C [7]. In the sequel, we will refer to this model layer as the *ground model*.

In our model we consider a network with an undetermined but fixed number of communication channels. This is a generalisation of the TTA, which typically involves only two channels. In the formal model, we divide communication between nodes into three phases: the sending of frames by a sending node, the transmission of the frame on a channel, and the reception of the frame at the receiving nodes. To model the reception of frames we introduce a function $rcvd(n, c, r)$ to denote the frame a receiving node r receives in slot n on channel c. Similarly, $sent(n, c, p)$ denotes the frame that a node p has sent in slot n on channel c, while $transmit(n, c)$ models the frame that is transmitted on channel c in slot n. The ultimate goal of the ground model of our hierarchy

is to precisely state the relationship between these entities and to prove the correctness properties *Validity*, *Agreement*, and *Authenticity* explained in the previous section.

To this end, we first need to introduce notions expressing the checks a receiver carries out to determine the correctness of the frame received, and stating the conditions under which these checks are assumed, or required, to succeed. First of all, we formalise the notion of frame status. In TTP/C, frames can be *null* frames, *valid* frames, or *correct* frames. Frames that are neither null frames not valid, are called *invalid* frames, while valid frames that are not correct are called *incorrect*. TTP/C furthermore distinguishes tentative frames and frames that have other errors; the former relates to situations in the implicit acknowledgement process of nodes, while the latter refers to illegal mode change requests. These types of frames are, however, not considered in our model. The status of the frame received by node r on channel c in slot n will be denoted by $\mathit{frame\_status}(n, c, r)$.

Frames are considered (syntactically) valid if the frame is transmitted during the receive window of the receiving node, no code violations are observed during the reception, and no other transmission was active within the receive window before the start of the frame. For a frame to be considered correct, it has to pass both the CRC check and the C-state agreement check [6].

Now we can formally state the desired correctness properties introduced in the previous section. The node scheduled to send in slot n is denoted $\mathit{sender}(n)$, while we use the (overloaded) notation $\mathcal{NF}^n$ to denote both the set of non-faulty nodes and the non-faulty channels in slot n; consequently, $r \in \mathcal{NF}^n$ and $c \in \mathcal{NF}^n$ indicate that node r and channel c are non-faulty in slot n.

Property 1 (Validity) *For all slots n, there exists a channel c such that if the sender of slot n sends a correct frame on c then all non-faulty nodes will receive this frame and assign the status* correct *to it:*

$$\begin{aligned} \exists c : \mathit{sender}(n) \in \mathcal{NF}^n \wedge \mathit{sends\_correct}(n, c, \mathit{sender}(n)) \Rightarrow \\ \forall r \in \mathcal{NF}^n : \mathit{rcvd}(n, c, r) = \mathit{sent}(n, c, \mathit{sender}(n)) \wedge \\ \mathit{frame\_status}(n, c, r) = \mathit{correct} \end{aligned}$$

Here, the predicate $\mathit{sends\_correct}(n, c, \mathit{sender}(n))$ subsumes what is considered a correct sending action of a node: the sending node sends a non-null frame, does so at the specified time, the frame carries the correct C-state information and the physical signal obeys the line encoding rules.

Property 2 (Agreement) *All non-faulty nodes consistently assign the sta-*

tus correct *to a frame received on a non-faulty channel c:*

$$p \in \mathcal{NF}^n \wedge q \in \mathcal{NF}^n \wedge c \in \mathcal{NF}^n \Rightarrow$$
$$\mathit{frame\_status}(n, c, p) = \mathit{correct} \Leftrightarrow \mathit{frame\_status}(n, c, q) = \mathit{correct}$$

Property 3 (Authenticity) *A non-faulty node r assigns the frame status* correct *to a frame received on a non-faulty channel c only if it was sent by the scheduled sender of the slot:*

$$r \in \mathcal{NF}^n \wedge c \in \mathcal{NF}^n \wedge \mathit{frame\_status}(n, c, r) = \mathit{correct} \Rightarrow$$
$$\mathit{rcvd}(n, c, r) = \mathit{sent}(n, c, \mathit{sender}(n))$$

In order to prove that these desired correctness properties hold for our model, several preconditions must be satisfied. These conditions concern the relationship between the frames sent by a sending node, transmitted through the channel, and received by the receivers, as expressed by the functions $\mathit{sent}(n, c, p)$, $\mathit{transmit}(n, c)$, and $\mathit{rcvd}(n, c, r)$, respectively. Therefore, we have to axiomatise the intended meaning of these functions. First of all, we formalise what is expected from a correct receiver. If a node r is non-faulty, we assume that it receives the frame that is transmitted on a given channel c:

$$r \in \mathcal{NF}^n \Rightarrow \mathit{rcvd}(n, c, r) = \mathit{transmit}(n, c) \tag{6}$$

However, even faulty nodes cannot receive other messages than those transmitted. Hence, nodes either receive whatever is transmitted on a channel, or nothing in the case of a reception fault:

$$\mathit{rcvd}(n, c, r) = \mathit{transmit}(n, c) \ \vee \ \mathit{rcvd}(n, c, p) = \mathit{null} \tag{7}$$

As for the sending nodes, we would like to model that non-faulty nodes send frames in their own sending slots, and remain silent otherwise. In some situations however, e. g. during the start-up of a TTP/C system or during the re-integration process of a node, the scheduled sender might not be fully integrated in the system and therefore, although being non-faulty, will not send at all in its sending slot. To also cope with these situations, we assume that in their sending slots non-faulty nodes either send a correct frame on all channels, or do not send any frame on any channel:

$$p = \mathit{sender}(n) \wedge p \in \mathcal{NF}^n \Rightarrow$$
$$(\forall c : \mathit{sends\_correct}(n, c, p)) \vee (\forall c : \mathit{sent}(n, c, p) = \mathit{null}) \tag{8}$$

Next we need to constrain the behaviour of the channels. As we intend to examine both channels with and without guardians, we now state certain requirements that must be satisfied by either configuration in order to maintain the correctness properties.

First, we like to express that non-faulty channels deliver the frame sent by some node. However, a faulty node might try to send a frame on a channel outside its assigned sending slot, which could then interfere with the frame sent by the scheduled sending node. For our ground model, we do not want to constrain the behaviour of the channel in this case, but allow the channel to either transmit one of the frames sent by the interfering nodes, or block all transmissions, or transmit a corrupted frame.

Requirement 1 *A non-faulty channel either transmits the frame sent by some node p, or nothing, or a corrupted frame.*

$$c \in \mathcal{NF}^n \Rightarrow (\exists p : \mathit{transmit}(n, c) = \mathit{sent}(n, c, p)) \\ \vee\ \mathit{transmit}(n, c) = \mathit{null} \vee\ \mathit{corrupted}(\mathit{transmit}(n, c))$$

However, this requirement is not sufficient, as it allows trivial, and rather useless, solutions such as channels that never transmit anything. In order to exclude these unwanted cases, we require that a non-faulty channel must transmit the frame of the scheduled sending node in situations where no other node interferes. We use the predicate $\mathit{single\_access}(n, c)$ to express that there is at most one node sending on channel c in slot n. Technically, this expression is an abstract parameter of our model; its interpretation depends on the concrete implementation of the channels, and it will be defined later in the subsequent refining model layers.

Requirement 2 *If the scheduled sender exclusively accesses the channel and sends a correct frame, then this frame is transmitted by the channel.*

$$c \in \mathcal{NF}^n \wedge \mathit{single\_access}(n, c) \wedge \mathit{sends\_correct}(n, c, \mathit{sender}(n)) \Rightarrow \\ \mathit{transmit}(n, c) = \mathit{sent}(n, c, \mathit{sender}(n))$$

The last of the basic requirements for non-faulty channels accounts for the fact that channels are passive entities and thus cannot generate frames by themselves. Here, the expression $\mathit{sends}(n, c, p)$ is an abbreviation for $\mathit{sent}(n, c, p) \neq \mathit{null}$.

Requirement 3 *Channels can only transmit what has been sent by some*

node.

$$transmit(n, c) \neq null \Rightarrow \exists p : sends(n, c, p)$$

We are now going to derive the proofs of the three main correctness properties for the TTP/C communication. However, as we will see, the assumptions and requirements listed so far are not sufficient to allow such a derivation. Consequently, we need to further constrain the behaviour of both the channels and the sending nodes in order to achieve correct communication. We will introduce the additional requirements as they become necessary in the derivation of the proofs.

4.1.1 Proof of Validity

The proposition *Validity* states two aspects of the reception of a correct frame sent by the scheduled sender in a given slot: first, all non-faulty receivers must receive the frame sent by the sender, and, second, all of them must accept this frame. With respect to the first part, we can derive from (6) that all non-faulty receivers receive the frame that is transmitted by the channel c. Furthermore, Req. 2 states that a non-faulty channel transmits the frame sent by the sender, provided that there is no other node accessing the channel. This latter clause gives rise to another requirement on the communication network:

Requirement 4 *In every slot, there is at least one non-faulty channel that is accessed by at most one node.*

$$\exists c \in \mathcal{NF}^n : single\_access(n, c)$$

Note that this is a rather strong requirement, and one that is impossible to satisfy for a channel without further measures, because it requires a certain behaviour of faulty nodes, which is outside the control of a channel. As we will see in the subsequent sections, the treatment of this requirement is one that distinguishes the communication model with guardians from one without. With this requirement we can prove the first part of *Validity*, i.e., that all correct nodes receive the frame sent by the sender.

As for the second part, we need to demonstrate that all correct receivers assign the status *correct* to the frame, that is, that they see a non-null, valid frame that passes both the CRC check the C-state agreement check. Non-emptiness of the frame can be proved from the fact that a correct sender sends a correct, and thus non-null, frame. A correct frame will always be transmitted by a non-faulty channel due to Req. 2, and Req. 4 ensures that such a non-faulty channel does indeed exist. Concerning the validity of the frame we have to consider the transmission timing of the frame and its signal encoding on the physical

layer. The latter is given by the same line of arguments as we demonstrated that the frame is not a *null*-frame: a correct sender sends a correct frame, which includes a correct signal encoding, and a correct channel will transmit this frame.

Next, we focus on the transmission timing of the frame. In order for the frame to be received by a non-faulty receiver within its receive window, the sending node and the receiver must be synchronised. Moreover, the values defining the nominal sending time of a frame and the start and end of the receive window, respectively, must be chosen such that the possible slight differences among the readings of the nodes' local clocks allow for the receivers to open their receive windows at "the right time". If we presuppose that non-faultiness of nodes encompasses that they are synchronised and that the window timing parameters are set correctly, it suffices to show that the sender sends its frame in time, and that the channel has a transmission delay that is bounded by some given bound $\widehat{d}$. The first is given by the fact that the sender sends a correct frame, and therefore the transmission time is correct. The second, however, must be stated as another requirement on the correct behaviour of a channel:

Requirement 5 *The transmission time of a correct frame on a non-faulty channel does not deviate from the sending time by more than some bounded delay d.*

$$
\begin{aligned}
& c \in \mathcal{NF}^n \wedge \mathit{sends\_correct}(n, c, p) \wedge \mathit{single\_access}(n, c) \Rightarrow \\
& \qquad \exists d : d \leq \widehat{d} \ \wedge \ \mathit{transmission\_time}(f') = \mathit{send\_time}(f) + d \\
& \mathit{where}\ p = \mathit{sender}(n), f = \mathit{sent}(n, c, p), f' = \mathit{transmit}(n, c)
\end{aligned}
$$

The last characteristic of a valid frame is unique transmission, which is ensured by Req. 4.

So far we have shown that the received frame is considered valid by non-faulty receivers. We are thus left to examine the CRC check and the C-state agreement check. As for the first, an incorrect CRC checksum is used to signal transmission faults. As the channel c under consideration is a non-faulty one, it is reasonable to assume that this includes the fact that no transmission fault occurs on c. Therefore we can conclude that a frame received by a non-faulty receiver on a non-faulty channel will pass the CRC check.

Finally, we consider the C-state agreement check. For the frame to pass the check, the C-state encoded in the frame has to correspond to the receiver's C-state. For this to be the case, two things must be ensured: first, the sender and the receiver must have equal C-states, and, second, the sender provides a correct encoding of its C-state in the frame. The first part corresponds to the functionality of the clique avoidance mechanism of the TTP/C group

membership algorithm, which is responsible for maintaining a single clique of nodes during system operation, that is, a single group of nodes that has equal C-states. We abstract from this protocol property by assuming that our notion of non-faultiness of nodes includes that two non-faulty nodes belong to the same (single) clique and thus have common C-states.

The second part, however, gives rise to another requirement on the behaviour of a correct channel.

Requirement 6 *A non-faulty channel transmits a frame with a correct signal encoding only if the frame sent provides a correct encoding of the sender's C-state.*

$$c \in \mathcal{NF}^n \wedge transmit(n,c) = sent(n,c,p) \wedge sends(n,c,p) \wedge$$
$$signal\_encoding\_OK(transmit(n,c)) \Rightarrow$$
$$cstate\_encoding\_OK(n, sent(n,c,p), p)$$

We can summarise that with the requirements introduced above the *Validity* property can be derived.

4.1.2 Proof of Agreement

To prove *Agreement*, we have to demonstrate that if some non-faulty receiver considers a received frame correct, then all non-faulty receivers do so. To establish this property we have to prove the same six characteristics of the received frame as for *Validity*, that is, reception of a non-null frame, the three properties of valid frames, and the two correctness checks. Each of these six cases can be proved using the same requirements as the proof of *Validity*. The structure of the proof of the agreement property is very similar to that of the *Validity* property; we therefore omit a detailed description.

4.1.3 Proof of Authenticity

For *Authenticity* we are required to show that if a frame is considered correct by a correct receiver then this node has in fact received the frame sent by the scheduled sender of the slot. In order to derive this fact we note that since the receiver considers the frame correct, we know that the six characteristics that define correct frames hold. This implies, for instance, that the receiver has detected a non-null frame. As channels only broadcast frames that have actually been sent by some node, cf. Req. 3, we know that there is an originator of the frame and that the receivers have received the frame sent by this sending node, say p. Hence, we only need to prove that this node p is in fact the

scheduled sender of the current slot. However, the facts established so far are not sufficient to do so, and hence we need to introduce one final requirement on the behaviour of channels.

Requirement 7 *A non-faulty channel transmits a correctly sent frame only if it originates from the scheduled sender of the given slot.*

$$c \in \mathcal{NF}^n \wedge \mathit{sends\_correct}(n, c, p) \wedge \mathit{transmit}(n, c) = \mathit{sent}(n, c, p) \Rightarrow$$
$$p = \mathit{sender}(n)$$

This requirement, together with the precondition that the received frame is considered correct by the receiving node, enables us to prove that the originator of the frame is indeed the scheduled sender of the given slot.

This concludes the derivation of the three desired correctness properties for the communication of TTP/C and the requirements they are based on. In the following two sections we will describe under which fault hypotheses these requirements can be met, both for a scenario with and without bus guardians.

4.2 Strong TTP/C Fault Hypothesis

In the previous section we have described a formalisation of the reception of frames by a node and have stated seven requirements that must be satisfied by the sending nodes and the channels in order to establish the desired correctness properties. In this section we are now going to give a formal model for sending nodes and channels and examine how the requirements stated above can be met. First, we consider the scenario where the channels are simple passive entities that broadcast the frames sent by a sender without further mechanisms, before, in the next section, we analyse a refinement of this model that incorporates bus guardians.

The ground model presented in the previous section expresses certain required properties of the entities $\mathit{sent}(n, c, p)$ and $\mathit{transmit}(n, c)$, which model the behaviour of the sending nodes and the channel, respectively. In a technical sense, these entities are parameters of the model. We now give an interpretation to these parameters for a network without guardians and show that the general requirements are satisfied for these interpretations.

First, we give a definition of the predicate $\mathit{single\_access}(n, c)$. This predicate is intended to model the case where only the scheduled sender sends a frame, and no other node interferes. Hence, we define it as true if there are no two

different nodes that send a non-null frame on the channel in the same slot:

$$single\_access(n, c) := \forall p, q : sends(n, c, p) \wedge sends(n, c, q) \Rightarrow p = q \qquad (9)$$

The interpretation of $sent(n, c, p)$ and $transmit(n, c)$ is given in an axiomatic style, and the set of axioms essentially constitutes the fault hypothesis of the guardian-free setting. In this setting, we cannot say anything about the frame transmitted by a channel other than that it depends on what is sent by the sending nodes. This is in contrast to the scenario with guardians, where we can, for instance, express that a guardian will not broadcast a frame if it does not originate from the scheduled sender.

Hypothesis 1 *A non-faulty channel without a guardian will transmit a frame sent by a node p if no other node accesses the channel in the given slot n.*

$$c \in \mathcal{NF}^n \wedge sends(n, c, p) \ \wedge \ (\neg \exists q : q \neq p \wedge sends(n, c, q)) \ \Rightarrow$$
$$transmit(n, c) = sent(n, c, p)$$

This hypothesis is sufficient to prove Req. 2 of the ground model; to see this, note that with the definition of *single_access*, the premise of Req. 2 implies that of Hyp. 1.

In order to prove Req. 1, we must assume that a channel can only transmit a non-null frame if it has been sent by some node:

Hypothesis 2 *If a channel broadcasts a non-null frame, then there is a corresponding node that has sent this frame.*

$$transmit(n, c) \neq null \ \Rightarrow \ \exists p : sends(n, c, p)$$

With this assumption we can now prove Req. 1 of the ground model: either the frame transmitted on a channel is a null frame, or if it is not, then by Hyp. 2 there is a sending node p, and the channel transmits the frame sent by p according to Hyp. 1.

Note that Hyp. 2 is actually identical to Req. 3 of the ground model. At this level, we cannot further constrain the behaviour of the channels more than what is expressed by Hyp. 1; consequently, some of the requirements of the ground model have to be restated as hypotheses on the channels for the guardian-free case. This is also true for Req. 5, which constrains the possible delay in the delivery of a frame on a non-faulty channel:

Hypothesis 3 *The delivery time of a frame on a non-faulty channel does not*

deviate from the transmission time by more than some bounded delay d.

$$c \in \mathcal{NF}^n \wedge sends(n, c, p) \wedge f' \neq null \wedge \neg\, corrupted(f') \Rightarrow$$
$$\exists d : d \leq \widehat{d} \wedge transmission\_time(f') = sending\_time(f) + d$$
$$where\ f = sent(n, c, p), f' = transmit(n, c)$$

Thus, Req. 3 and Req. 5 are trivially satisfied by our model. Note, however, that the corresponding assumptions are by no means just inadmissible simplifications of the matter. On the contrary, these hypotheses are direct formalisations of the strong fault hypothesis of the "raw" TTP/C protocol [7].

We proceed by extending our model in order to also derive the remaining three requirements of the ground model.

Considering Req. 6, we need to ensure that a non-faulty channel only transmits frames that contain a correct encoding of the sender's C-state. Since in the guardian-free setting, the channels will transmit whatever is sent by the sending node, the responsibility for providing a correct C-state encoding is with the sender. Note that this must be true not only for the scheduled sender, but extends to all nodes, even faulty ones, and hence is a rather strong assumption.

Hypothesis 4 *Frames sent must contain a correct encoding of the sender's C-state.*

$$sends(n, c, p) \Rightarrow cstate\_encoding\_OK(n, sent(n, c, p), p)$$

Requirement 7 states that correct frames must only be sent by the scheduled sender of a given slot. However, for the guardian-free case a channel cannot prevent other nodes from sending. Hence, in order to prove this property for this model, we need to introduce a corresponding hypothesis and assume that the behaviour of the sending nodes is in compliance with the sending schedule.

Hypothesis 5 *Correct frames must only be sent by the scheduled sender of a given slot.*

$$sends\_correct(n, c, p) \Rightarrow p = sender(n)$$

We are left to prove Req. 4, which states that in every slot there exists at least one non-faulty channel that is not accessed by more than one sending node. To prove this fact we have to make a series of assumptions about the number and behaviour of faulty nodes and channels. First of all, we need to assume that a non-faulty channel exists at all times.

Hypothesis 6 *In every slot, there is at least one non-faulty channel.*

$$\exists c \in \mathcal{NF}^n$$

TTP/C is based on a *single fault assumption*, i. e. at any given time there is at most one faulty component in the network. Consequently, there cannot be more than one faulty node present in any given slot.

Hypothesis 7 *There is at most one faulty node in every slot.*

$$p \notin \mathcal{NF}^n \wedge q \notin \mathcal{NF}^n \Rightarrow p = q$$

In deriving Req. 4, we first consider the case where there is no faulty node. By Hyp. 6 we know that there exists a non-faulty channel c. To establish Req. 4 we must therefore show that at most one node sends a frame on c. Since the scheduled sender of the given slot is allowed to send a frame, we must ensure that no other node can send. We can establish this fact if we assume that a *non-faulty* node does not send anything outside its designated sending slots, which is a reasonable assumption to make.

$$p \neq sender(n) \wedge p \in \mathcal{NF}^n \Rightarrow \neg\, sends(n, c, p) \tag{10}$$

Now suppose that there is a faulty node, p say. If p is the scheduled sender of the given slot we are done, because then all other nodes are non-faulty, by Hyp. 7, and do not send a frame, see Hyp. 10. Therefore consider the case where the faulty node p is not the current sender. In order to prevent a collision on the channel we must require that p, even if it is faulty, does not send.

Hypothesis 8 *Nodes other than the sender of a slot, including faulty ones, will not send data on every non-faulty channel outside their assigned sending slots.*

$$p \neq sender(n) \Rightarrow \neg\, \forall c \in \mathcal{NF}^n : sends(n, c, p)$$

Note that the hypothesis as stated only requires that p does not send on at least one of the non-faulty channels. Again, this is a strong hypothesis, as it constrains the behaviour even of *faulty* nodes.

This completes our derivation of the seven requirements of the ground model for a communication network without guardians. In order to establish the requirements we have stated a series of hypotheses. Besides describing the intended behaviour of correct nodes and channels, these assumptions directly reflect the strong fault hypothesis of the "raw" TTP/C protocol [7]. We have shown that this fault model is sufficient to prove the requirements of the

ground model, and thus established the desired correctness properties for the communication in a TTP/C network. What makes this set of hypotheses strong or optimistic is the fact that assumptions are not restricted to non-faulty nodes, but also encompass faulty ones, cf. Hyp. 4, 5, and 8. In the following section, these hypotheses will be replaced with weaker ones about the behaviour of *non-faulty* guardians.

4.3 Guardians

In the scenario described in the previous section, where a channel transmits a frame whenever there is no concurrent access to it, strong assumptions about the behaviour of the sending nodes have to be made in order to satisfy the requirements stated in the ground model. In particular, some of the assumptions even concern the behaviour of faulty nodes, such as that sending nodes always provide a correct C-state in the frame, or that correct frames are sent only by the scheduled sender of a slot. Whenever one relies on a certain benignity of faults one has to examine how well the fault assumptions are covered by the system. If such an analysis is difficult, or leads to the result that the probability of a fault being outside of the scope of the assumed fault hypothesis is not negligible, it is advisable to aim to eliminate, or at least weaken, assumptions about the behaviour of faulty components. To this end, guardians are used in the Time-Triggered Architecture to avoid certain fault scenarios, such as, for instance, faulty nodes accessing the bus outside their assigned slots.

In this section we describe the formalisation of abstract guardian components. The formalisation is abstract in the sense that it does not restrict the kind of the guardian and the topology of the communication network; we will show in the subsequent section how this abstract model can be refined either to a bus topology, where each node has its own local guardians, or to a star topology with central bus guardians.

We state a number of hypotheses on the expected behaviour of a non-faulty guardian and show that they are sufficient to prove the requirements of the ground model. Thus, the desired correctness properties for the communication of TTP/C are satisfied for a communication network with guardians.

In our model, we use $g(c)$ to denote the guardian of channel c. We think of a guardian as having incoming links from each of the nodes of the network, and corresponding outgoing links. The task of a guardian is to receive the frames sent by the nodes, analyse them and relay them to the other nodes according to certain rules. Obviously, these rules would prescribe, among other things, that only the frame of the scheduled sender of a slot is relayed. To describe the functionality of a guardian we use a function $relay(n, g(c), p)$ that denotes

the frame the guardian $g(c)$ relays from node p in slot n.

By the following hypotheses we describe what is expected from a non-faulty guardian. To distinguish the guardian hypotheses from the ones presented in the previous section, they are labelled with capital letters instead of numbers. First, if the scheduled sender of a slot sends a correct frame, then the guardian should relay this frame:

Hypothesis A *If the scheduled sender of a slot sends a correct frame, then a correct guardian relays this frame.*

$$p = sender(n) \ \wedge \ g(c) \in \mathcal{NF}^n \ \wedge \ sends\_correct(n, c, p) \Rightarrow$$
$$relay(n, g(c), p) = sent(n, c, p)$$

Conversely, frames of nodes other than the scheduled sender must not be relayed:

Hypothesis B *A non-faulty guardian must not relay frames of nodes other than the scheduled sender.*

$$p \neq sender(n) \wedge g(c) \in \mathcal{NF}^n \ \Rightarrow \ relay(n, g(c), p) = null$$

In addition to this basic functionality of supervising the correct message schedule, the guardian performs several other analyses in order to prevent fault propagation and possible SOS faults. First, if a sending node does not start to send its frame within the nominal sending window, the guardian closes the window with the effect that a null frame is relayed.

Hypothesis C *A non-faulty guardian will relay a frame only if it is being sent in time.*

$$p = sender(n) \wedge g(c) \in \mathcal{NF}^n \wedge \neg\, sending\_time\_OK(n, sent(n, c, p), p) \Rightarrow$$
$$relay(n, g(c), p) = null$$

Furthermore, if the signal encoding of the frame sent by a node violates the coding rules such that the guardian cannot decode the signal, it will end the broadcast of the frame prematurely, thus corrupting the frame.

Hypothesis D *A non-faulty guardian will corrupt a frame if the signal encoding of the frame violates the coding rules.*

$$p = sender(n) \wedge g(c) \in \mathcal{NF}^n \wedge \neg\, signal\_encoding\_OK(sent(n, c, p)) \Rightarrow$$
$$corrupted(relay(n, g(c), p))$$

Finally, if the C-state encoded in a frame does not correspond to the guardian's own C-state, then the guardian aborts the transmission of the frame, and the relayed frame will be corrupted. This serves, for example, to protect a node that is about to integrate into the cluster against so-called *masquerading* nodes that provide an incorrect MEDL position within the C-state.

Hypothesis E *A non-faulty guardian will corrupt a frame if the C-state encoded in the frame does not correspond to the guardian's own C-state.*

$$
\begin{aligned}
& g(c) \in \mathcal{NF}^n \wedge \neg\, cstate\_encoding\_OK(n, sent(n, c, p), p) \\
& \qquad \Rightarrow corrupted(relay(n, g(c), p)) \\
& where\ p = sender(n)
\end{aligned}
$$

These hypotheses describe the supervising functionality of a guardian. In addition, a non-faulty guardian is expected to behave in a reasonable way. First, guardians are assumed to be passive entities in the sense that they can only relay frames that have actually been sent by some node. In other words, guardians cannot generate valid frames by themselves.

Hypothesis F *Guardians are passive and can only relay frames that have actually been sent by some node.*

$$relay(n, g(c), p) \neq null \;\Rightarrow\; sends(n, c, p)$$

The next assumption on the functionality of a guardian concerns the timing behaviour. In order to fulfil Req. 5 of the ground model we must assume that a guardian delivers a relayed frame with a bounded delay.

Hypothesis G *A non-faulty guardian relays a frame with a bounded delay.*

$$
\begin{aligned}
& g(c) \in \mathcal{NF}^n \wedge sends(n, c, p) \wedge f' \neq null \Rightarrow \\
& \qquad \exists d : d \leq \widehat{d} \;\wedge\; transmission\_time(f') = sending\_time(f) + d \\
& where\ f = sent(n, c, p),\ f' = relay(n, g(c), p)
\end{aligned}
$$

The final two assumptions concern the number and kinds of possible faults of the guardians. First, we assume that for all slots the guardian of at least one of the channels is non-faulty.

Hypothesis H *For every slot n, there is at least one channel with a non-faulty guardian.*

$$\exists c : g(c) \in \mathcal{NF}^n$$

A faulty guardian may fail only in such a way that it delays the delivery of a frame for an arbitrary amount of time and thus effectively does not relay any non-null frame in the given slot n.

Hypothesis I *A faulty guardian fails silently and does not relay any frame.*

$$g(c) \notin \mathcal{NF}^n \Rightarrow relay(n, g(c), p) = null$$

Some of the requirements of the ground model involve the abstract predicate $single\_access(n, c)$ and we must hence give an interpretation to this predicate for a communication network with guardians. Since the guardians are intended to just prevent the simultaneous access of a channel by two different nodes, we define this predicate to be always true:

$$single\_access(n, c) := true \tag{11}$$

Finally, we say that a channel is non-faulty if its corresponding guardian is.

$$c \in \mathcal{NF}^n := g(c) \in \mathcal{NF}^n \tag{12}$$

To complete the formalisation of the guardian model, we need to define what is meant by the frame a channel broadcasts, i. e. we require a definition of the function $transmit(n, c)$. Obviously, this function definition must reflect the frame that a guardian relays for some node p. On the other hand, there cannot be frames from more than one node be transmitted per slot. Consequently, we say that a frame is transmitted on a channel c if there is a node p such that the guardian $g(c)$ of channel c relays that frame for p, and does not relay any frame for all nodes other than p. The technical definition of $transmit(n, c)$ proceeds in two steps. First, we define a predicate $unique_c^n(f)$ to be true, if in slot n the guardian of channel c relays frame f for some node, but relays no frames for any other node:

$$\begin{aligned} unique_c^n(f) := \exists p : relay(n, g(c), p) = f \ \wedge \\ \forall q : q \neq p \ \Rightarrow \ relay(n, g(c), q) = null \end{aligned} \tag{13}$$

For the definition of $transmit(n, c)$ we use Hilbert's choice operator ϵ, where $\epsilon(S)$ denotes some arbitrarily chosen element from a given set S. Here, the set S consists of those frames f for which the predicate $unique_c^n(f)$ is true. Obviously, this set can contain at most one frame; consequently, if the set is non-empty, simply this unique frame is chosen. In the other case where the set is empty, i. e. no frame satisfies the *unique*-predicate, the ϵ-operator returns

an arbitrary frame, for which no special properties can be deduced.

$$transmit(n, c) := \epsilon(unique_c^n) \tag{14}$$

In the remainder of this section we present the arguments that show that this definition of $transmit(n, c)$ and the hypotheses stated for a guardian are sufficient to satisfy the requirements of the ground model. To this end, we state two properties of $transmit(n, c)$. First note that a non-faulty guardian either transmits the frame sent by the scheduled sender of a given slot, or a corrupted frame, or a null-frame.

Proposition 1 *A non-faulty guardian either transmits the frame of the scheduled sender, or a corrupted frame, or a null-frame.*

$$\begin{aligned} c \in \mathcal{NF}^n \;\Rightarrow\; & transmit(n, c) = sent(n, c, sender(n)) \;\vee \\ & corrupted(transmit(n, c)) \;\vee\; transmit(n, c) = null \end{aligned}$$

If in addition we know that the scheduled sender of a slot sends a correct frame, then the guardian indeed broadcasts this frame.

Proposition 2 *A non-faulty guardian transmits a correct frame if it is sent by the scheduled sender of slot n.*

$$\begin{aligned} & c \in \mathcal{NF}^n \;\wedge\; sends\_correct(n, c, sender(n)) \;\Rightarrow \\ & \quad transmit(n, c) = sent(n, c, sender(n)) \end{aligned}$$

We briefly sketch the proofs of these properties. First note that, according to Hyp. B, a non-faulty guardian does not relay a frame for nodes other than the scheduled sender of the given slot. Moreover, since guardians will not produce frames by themselves, see Hyp. F, the only frame that is relayed by a non-faulty guardian is the one sent by the scheduled sender of the slot. Hence, this frame satisfies the *unique*-predicate, and therefore $transmit(n, c)$ equals the frame that is relayed by the guardian for the scheduled sender of the current slot. Proposition 1 holds because, depending on whether or not the scheduled sender sends a correct frame, the guardian either relays this frame according to Hyp. A, or blocks or corrupts the frame following Hyp. C, D, or E.

The second proposition is a specialisation of the first, where we know that the scheduled sender sends a correct frame. In this case, the guardian relays this frame and by the same reasoning as above this frame is tranmitted on the channel.

These two propositions provide the connection between the requirements of the ground model, which are stated in terms of the expression *transmit*, and the hypotheses of the guardian model, which are very similar, but involve the *relay* forms. To derive the general requirements, we see that Req. 1 of the ground model directly follows from the first of the propositions above, while the second proposition implies Req. 2. Requirement 3 can be proved from the similar assumption that guardians do not send frames by themselves, see Hyp. F. Requirement 4 follows from the assumption that there always exists a non-faulty guardian, see Hyp. H, and observing that the predicate $single\_access(n, c)$ is always true. Requirement 5 on the bounded delay of transmissions follows from Hyp. G, while the assumption concerning the encoded C-state of a frame, Hyp. E, is used to prove Req. 6. Finally, Req. 7 follows from the combination of the two propositions above.

Having shown that all of the requirements of the ground model are satisfied in the guardian model, we can deduce that the three correctness properties *Validity*, *Agreement* and *Authenticity* hold for the guardian model. In comparison to the model without guardians, weaker hypotheses are sufficient to prove the requirements. In particular, we do no longer need to make any assumptions about the behaviour of faulty nodes, thus a broader class of node faults can be tolerated by a TTP/C network using guardians.

4.4 Local vs. Central Guardians

In this section we briefly discuss how the guardian model described above can be applied to both an interconnection network with a star topology and one with a bus topology. In the former, central guardians are used, which are usually located at the centre of each communication channel, i.e. at the star coupler. Thus, the above guardian model can be directly matched to this setup, since the denotation $g(c)$ appropriately models the central guardian device at the star coupler of channel c.

In a connection network that uses the bus topology every node is equipped with its own local guardian, typically one for each channel. Therefore, one would rather use a function *lbg* to denote particular guardian devices, such that $lbg(p, c)$ is the local bus guardian of node p for channel c. Nevertheless, the functionality of the bus guardians can be described in the same way as in the star-topology model by formalising assumptions about the frames relayed by the local bus guardians, as expressed by the function $relay(n, lbg(p, c), p)$. In order to use the abstract guardian model of the previous section we only need to combine the local bus guardians of all nodes that supervise a particular channel c to one logical entity. Thus, the expression $g(c)$ would denote a function that yields for a given node its local bus guardian that controls

channel c. Formally:

$$g(c) := \lambda p : lbg(p, c) \tag{15}$$

Consequently, the system of local bus guardians at channel c is considered non-faulty, if for all nodes p the local bus guardian $g(c)(p)$ is non-faulty:

$$g(c) \in \mathcal{NF}^n \;:\Leftrightarrow\; \forall p : lbg(p, c) \in \mathcal{NF}^n \tag{16}$$

To summarise, the abstract guardian model can be arranged in a way that the formalisations of guardians for both a star-based topology and a bus topology can be derived as an instance of this model. The details are, however, mainly of a technical nature and do not provide any further conceptual insight; therefore, they are omitted here. At the bottom line we can state that, as long as the same algorithms and supervising functions are implemented in either guardian type, both the local bus guardians and the central guardians of a star coupler provide the functionality to satisfy the requirements stated in the ground model and thus ensure that the main correctness properties for the communication of TTP/C hold.

5 Conclusions

The goal of formally analysing aspects of the Time-Triggered Architecture is to provide mathematically substantiated arguments that architecture and algorithms provide certain services and satisfy certain critical properties. This is to support the claims that the architecture meets the high reliability requirements of safety-critical applications in the automotive or aerospace domain.

In this regard we have presented a formal analysis of the guardian-based communication of TTP/C. We have developed a series of formal models of the interconnection network that are hierarchically structured and formalise different aspects of the communication of TTP/C nodes at various levels. The ground level provides a precise specification of the desired correctness properties of the TTP/C communication. It states several requirements that must be satisfied for the channels in order to guarantee that the correctness properties hold. These requirements serve as an interface of the model. In a process of stepwise refinement we have proved the validity of these properties for TTP/C by showing that the interface requirements hold for the refined model layers. The organisation of the model hierarchy not only facilitates the formal proof by dividing it into manageable steps. It also reflects the structure of what constitutes the Time-Triggered Protocol, viz. the communication controllers of the nodes, and the guardians. The former provide the fault-tolerant protocol

services on the basis of strong fault assumptions, which, in turn, are guaranteed by the guardians. Thus, one of the benefits of our formal analysis is that the formal models yield a concise formal description of the respective purposes and dependencies of these components, and precisely state the assumptions about the behaviour of a guardian, which previously had been stated only informally [7].

One of the characteristics of the formal models is their abstract nature. Abstraction is a fundamental prerequisite for the feasibility of formal analysis, as it allows for both structuring the models by providing abstract interfaces and hiding details unnecessary or irrelevant for the formal analysis and the demonstration of critical properties. An adequate structure of formal models allows one to concentrate on particular aspects of a TTA system, such as the behaviour of the guardians in the communication network. Different items can then be analysed separately from each other, assuming certain properties of other models where necessary. Moreover, abstract interfaces of the models also provide a certain degree of genericity, which enables one to express the commonalities of a range of designs in a coherent way. For instance, one of the model layers provides a generic treatment of the guardians. The model can then be refined to either a central guardian-based view, or to a model for local bus guardians, thereby covering the two typical network topologies of a TTA system.

The formal models presented in this paper have been developed with the specification and verification system PVS, and all proofs have been mechanically checked using PVS's theorem prover. Although the individual proofs of most of the properties and facts are relatively simple and straightforward, the use of a mechanical proof assistant has been found very valuable. One of the difficulties in developing formal models and proofs is to keep track of all details and the dependencies of the various properties. PVS is particularly useful for such tasks, as it does not only check the proofs provided for the claimed properties, but also provides bookkeeping functions to ensure that there are no gaps in the chain of arguments for a given fact. Moreover, if changes are made to a formal model, PVS requires all proofs of properties that depend on the changed model to be re-run. Thus, if changes cause proofs to be no longer valid, these will not go undetected.

A mechanism that has been found particularly useful is PVS's support of *theory assumptions*. In a PVS theory one declares the relevant entities of a formal model, and states – and then proves – the properties these entities have. Theories can be parameterised, and one can state certain assumptions about concrete interpretations of these parameters. The properties within such a parameterised theory are then based on these assumptions. If such a theory is instantiated, that is, the parameters are given concrete interpretations, PVS automatically creates proof obligations that require to show that the

stated assumptions indeed hold for the given interpretations. We have employed this mechanism for specifying our ground model of the general reception of frames. This model is parameterised with the entities describing the sending of frames by nodes, $sent(n, c, p)$, or the transmission of frames by a channel, $transmit(n, c)$, among others. The general requirements described in detail in Section 4.1 are expressed as PVS assumptions on these parameters. The desired correctness properties, such as *Validity* or *Agreement*, are proved relative to these assumptions. The formal models on the higher hierarchy levels that describe the strong fault model and the guardian model, cf. Sections 4.2 and 4.3, respectively, instantiate the ground model. Thus, PVS generates proof obligations that correspond to showing that the general requirements of the ground model are valid for the provided interpretations of $sent(n, c, p)$ or $transmit(n, c)$. This way, PVS provides support to ensure that eventually all claimed facts are indeed proved.

The analysis of the properties of the communication network of TTA has supported the claim that the functionality of the guardians ensures that arbitrary node failures are converted into fault modes the TTP/C protocol algorithms can tolerate. Thus, the strong fault hypothesis of TTP/C can be replaced by a weaker, minimal fault hypothesis on the correct behaviour of the guardians, which has two direct advantages. First, applications of TTA can rely on the architecture to tolerate a broad class of faults, and, second, protocol algorithms of TTP/C can be designed for and analysed under the strong fault model, which allows for simpler algorithms and significantly facilitates formal analysis.

References

[1] H. Kopetz, The Time-Triggered Approach to Real-Time System Design, in: B. Randell, J.-C. Laprie, H. Kopetz, B. Littlewood (Eds.), Predictably Dependable Computing Systems, Springer-Verlag, 1995.

[2] H. Kopetz, The Time-Triggered Architecture, in: Proc. 1st Intl. Symp. on Object-Oriented Real-Time Distributed Computing (ISORC), 1998, pp. 22–31.

[3] H. Kopetz, G. Bauer, The Time-Triggered Architecture, Proceedings of the IEEE 91 (1) (2003) 112 – 126.

[4] G. Heiner, T. Thurner, Time-Triggered Architecture for Safety-Related Distributed Real-Time Systems in Transportation Systems, in: Proc. 28th Intl. Symp. on Fault-Tolerant Computing (FTCS), IEEE Computer Society, 1998.

[5] T. Ringler, J. Steiner, R. Belschner, B. Hedenetz, Increasing System Safety for By-Wire Applications in Vehicles by Using a Time-Triggered Architecture, in: W. Ehrenberger (Ed.), Proc. 17th Intl. Conf. on Computer Safety, Security and

Reliability (SAFECOMP), Vol. 1516 of Lecture Notes in Computer Science, Springer-Verlag, 1998, pp. 243–253.

[6] G. Bauer, H. Kopetz, W. Steiner, Byzantine Fault Containment in TTP/C, in: Proc. Intl. Workshop on Real-Time LANs in the Internet Age (RTLIA), 2002, pp. 13–16.

[7] G. Bauer, H. Kopetz, W. Steiner, The Central Guardian Approach to Enforce Fault Isolation in the Time-Triggered Architecture, in: Proc. 6th Intl. Symp. on Autonomous Decentralized Systems (ISADS), 2003, pp. 37–44.

[8] H. Pfeifer, D. Schwier, F. von Henke, Formal Verification for Time-Triggered Clock Synchronization, in: C. Weinstock, J. Rushby (Eds.), Dependable Computing for Critical Applications (DCCA) 7, Vol. 12 of Dependable Computing and Fault-Tolerant Systems, IEEE Computer Society, 1999, pp. 207–226.

[9] S. Katz, P. Lincoln, J. Rushby, Low-Overhead Time-Triggered Group Membership, in: M. Mavronicolas, P. Tsigas (Eds.), Proc. 11th Intl. Workshop on Distributed Algorithms (WDAG), Vol. 1320 of Lecture Notes in Computer Science, Springer-Verlag, 1997, pp. 155–169.

[10] H. Pfeifer, Formal Verification of the TTP Group Membership Algorithm, in: T. Bolognesi, D. Latella (Eds.), Formal Methods for Distributed System Development – Proc. of FORTE XIII / PSTV XX, Kluwer Academic Publishers, 2000, pp. 3–18.

[11] A. Bouajjani, A. Merceron, Parametric Verification of a Group Membership Algorithm, in: W. Damm, E.-R. Olderog (Eds.), Proc. 7th Intl. Symp. on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT), Vol. 2469 of Lecture Notes in Computer Science, Springer-Verlag, 2002, pp. 311–330.

[12] A. Merceron, M. Müllerburg, G. Pinna, Verifying a Time-Triggered Protocol in a Multi-Language Environment, in: W. Ehrenberger (Ed.), Proc. 17th Intl. Conf. on Computer Safety, Security and Reliability (SAFECOMP), Vol. 1516 of Lecture Notes in Computer Science, Springer-Verlag, 1998, pp. 185–195.

[13] W. Steiner, J. Rushby, M. Sorea, H. Pfeifer, Model Checking a Fault-Tolerant Startup Algorithm: From Design Exploration To Exhaustive Fault Simulation, in: Proc. Intl. Conf. on Dependable Systems and Networks (DSN), IEEE Computer Society, 2004.

[14] J. Rushby, An Overview of Formal Verification for the Time-Triggered Architecture, in: W. Damm, E.-R. Olderog (Eds.), Proc. 7th Intl. Symp. on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT), Vol. 2469 of Lecture Notes in Computer Science, Springer-Verlag, 2002, pp. 83–105.

[15] J. Rushby, Systematic Formal Verification for Fault-Tolerant Time-Triggered Algorithms, IEEE Trans. on Software Engineering 25 (5) (1999) 651–660.

[16] S. Owre, J. Rushby, N. Shankar, D. Stringer-Calvert, PVS: An Experience Report, in: D. Hutter, W. Stephan, P. Traverso, M. Ullman (Eds.), Applied

Formal Methods (FM-Trends), Vol. 1641 of Lecture Notes in Computer Science, Springer-Verlag, 1998, pp. 338–345.

[17] G. Bauer, H. Kopetz, P. Puschner, Assumption Coverage under Different Failure Modes in the Time-Triggered Architecture, in: Proc. 8th IEEE Intl. Conf. on Emerging Technologies and Factory Automation (ETFA), 2001, pp. 333–341.

[18] TTTech, Time-Triggered Protocol TTP/C High-Level Specification Document, `http://www.tttech.com/technology/specification.html` (2003).

[19] H. Kopetz, The Time-Triggered (TT) Model of Computation, in: Proc. 19th IEEE Real-Time Systems Symposium, 1998, pp. 168–177.