Chapter 4

Building fault tolerant ERTS

This chapter presents the policies to build fault tolerant ERTS by integrating the fault tolerance ERTS framework layers in an ERTS architecture. Its objective is to define a set of components that implement those classes which compose the network, communication, and intra-node layers and to define the necessary steps for its integration. It is organized as follows: Section 4.1 presents and overview of the necessary components for the construction of fault tolerant ERTS. Section 4.2 presents the particular characteristics of the ERTS components and the relation of fault tolerance ERTS framework classes that they implement. Section 4.3 describes the configuration phase that defines the necessary steps to integrate the component into an ERTS architecture. Finally, section 4.4 discusses about solutions presented in previous sections.

4.1. Overview

The fault tolerance ERTS framework that we proposed defines a set of abstract classes that allows the construction of ERTS from the principal layers which are network, communication, and intra-nodes operation. The framework layers are integrated into a set of bound components that represents the physical architecture and the communication network of the ERTS that it is desired to construct.

The construction of ERTS with the fault tolerance ERTS framework will ensure that is execution is fault free in the aspects covered by the three principal framework layers. It means that there are no needing for include additional components to the ERTS architecture to tolerate faults because the tolerance is included on the design itself. By consequence, if any of the three aspects (network, communication, intra-node operation) is not implemented by the fault tolerance ERTS framework there are no guaranty that this particular aspect be fault tolerant.

Figure 4.1 illustrates the three principal framework classes and an example of an ERTS architecture.

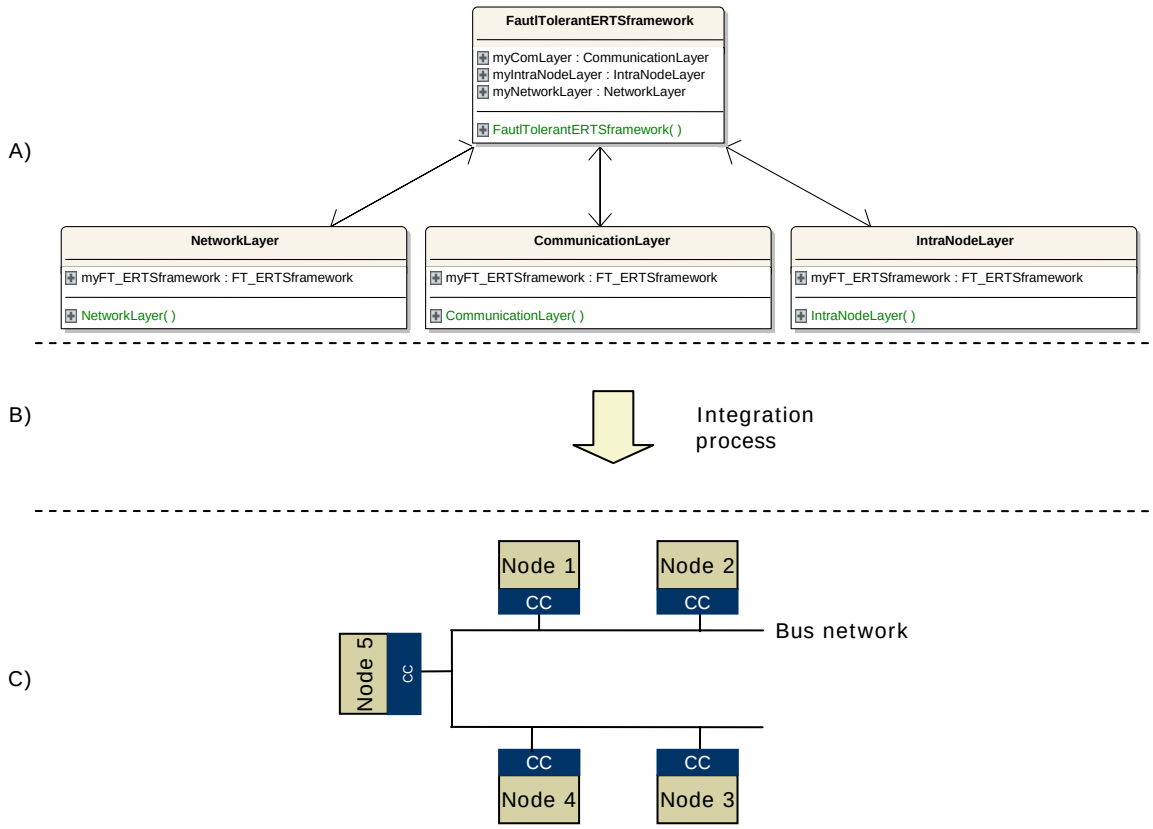

Figure 4.1 Framework classes and ERTS architecture

The figure 4.1 illustrates at A the network, the communication, and the intra-node layer classes of the framework. At B, the integration process of the classes presented at A into an ERTS components in order to build a fault tolerant ERTS. And at C, an example of an ERTS composed by five components that represent the ERTS architecture that is composed by five nodes and a bus network cable.

The integration process consists on insert the particular framework classes of the network, communication, and intra-node layers into the implementation of every component of the ERTS. For that reason each component is fragmented in three parts, where any part is designed for the implementation of one of the three principal classes of the framework.

In addition to this previous division and independently that every component of the ERTS is divided in the same form, it is necessary that different components implements different framework classes with the purpose of construct the ERTS layers. We propose three component types: the master component, the time component, and the normal component. Figure 4.2 illustrates the components fragmentation and the component types.

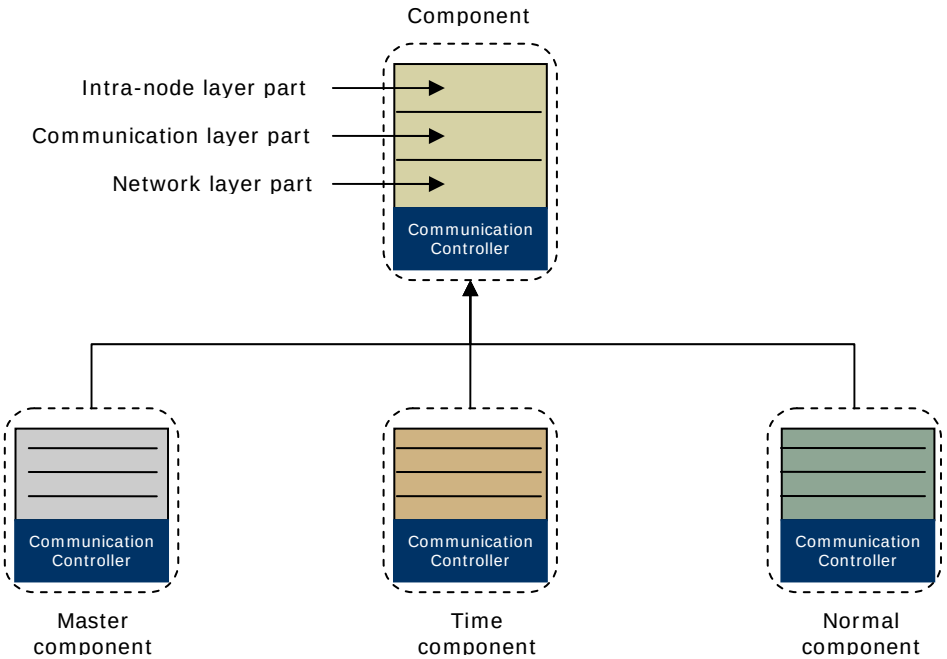

Figure 4.2 Component types

The ERTS includes one master component that generates the particular schedules for the tasks involved in the system, one or more time components that administrate the time assignment of the bus network, and several normal components that execute the functional operations of the ERTS.

4.2. ERTS components

The components on an ERTS implements different classes of the framework depending on the considered layer. Before to explain every component in particular is necessary to clarify that at network layer the entire components implement on their network layer part the same classes depending on the desired communication type (time-triggered, event-triggered, and mixed-triggered), at communication layer only the master and the time components implement some framework classes on their communication layer part, and at intra-node layer only the master node implements framework classes on its intra-node layer part. Next are the definitions for the master, time and normal components for the three layers.

4.2.1. Master component in communication layer

The master component implements the communication layer in any of its three different bus types (time-triggered, event-triggered, and mixed-triggered) with the purpose of calculate and adapted schedules for the ERTS operation.

Master components can be classified in three different types, each type corresponding and implementing on its communication layer part one of the three principal aspects of the communication layer: time-triggered, event-triggered, and mixed-triggered.

4.2.1.1. Time-triggered

It implements the time-triggered class included in the communication layer. It requires as input:

- The information for defining the time division multiple access protocol, that information is the network time unit value and the specification of the number of slots that are within a round.

- The time-triggered list containing the specification of the entire tasks that compose the ERTS with their associated real-time tasks properties.

- The implementation of one of the non-preemptive scheduling types (scheduling utilization bound or scheduling response time).

With the previous information the master component produces a global schedule table, with the use of the non-preemptive scheduling algorithm, which indicates the time in which every node of the ERTS is allowed to execute a particular task. The global schedule is fragmented for each node into particular schedule tables that only include the execution time of the tasks of that particular node.

A time-triggered master component additionally produces a reference schedule table based on the information of the time division multiple access protocol, in particular on the moment when each round begins.

4.2.1.2. Event-triggered

It implements the event-triggered class included in the communication layer. It requires as input:

- The event-triggered list containing the specification of the entire tasks that compose the ERTS with their associated real-time tasks properties.

- Selection of the priority types that the system requires (static or dynamic).

- The implementation of one of the preemptive scheduling types. Depending on the priorities type they can be Rate Monotonic Scheduling or Dynamic Monotonic Scheduling for static priorities, or Earliest Deadline First or Least Laxity First for dynamic priorities.

With the previous information the master component produces a template of a dynamical global schedule table with the capacity to be filled with the tasks that appear according to the occurrence of events. The assignation of priorities to tasks to the filling on the dynamical global schedule table is done with the use of the preemptive scheduling algorithm.

4.2.1.3. Mixed-triggered

It implements the mixed-triggered class included in the communication layer. It requires as input:

- The information for defining the time division multiple access mixed protocol, that information is the network time unit value, the specification of the mixed round (number of slots that are within a static round or number of slots available within a dynamic round), and the definition of the communication cycle.

- The time-triggered list containing the specification of the entire periodic tasks that compose the ERTS with their associated real-time tasks properties.

- The event-triggered list containing the specification of the entire non-periodic tasks that compose the ERTS with their associated real-time tasks properties.

- The implementation of two types of scheduling (preemptive and non-preemptive) from the scheduling model. The selection of each scheduling is similar to the used for the time-triggered master node and event-triggered master node.

With the previous information the master component produces a mixed global schedule table. At the beginning the table is generated, only with the scheduling of the periodic tasks, with the use of the non-preemptive scheduling algorithm; the rest of the schedule table is produced as a template with the capacity to be filled with the tasks that appear according to the occurrence of events. The assignation of priorities to tasks to the filling on the mixed global schedule table on

its non-periodic part is done with the use of the preemptive scheduling algorithm.

A mixed-triggered master component additionally produces a reference mixed schedule table based on the information of the time division multiple access mixed protocol, in particular on the moment when each round and each communication cycle begins.

4.2.2. Master component in intra-node layer

The master component implements the intra-node layer in order to generate a static distributed schedule or a fault tolerance schedule table for the ERTS, and to estimate the global ERTS time.

4.2.2.1. Functionality definition

It implements the static distributed schedule class, fault tolerance scheduling table class, and the replication heuristic class included in the intra-node layer. It requires as input:

- The task list containing the specification of the entire tasks that compose the ERTS with their associated real-time tasks properties.
- The algorithm graph representation that defines the manner in which operations and data-dependencies are related.
- The architecture graph representation that defines nodes and communication links among them.
- The distribution constraints description tables.
- The scheduling heuristic algorithm (optional for the case when a replication heuristic will be used).
- The backtracking algorithm.

With the previous information the master component can produce two kinds of schedules, depending if a replication heuristic class is implemented or not.

For the case when no replication heuristic class is provided explicitly, the scheduling heuristic considered is the defined in the AAA method, which is located in the static distributed schedule class included in the node layer. As result a static distributed schedule table is generated, it indicates the distribution order of the tasks over the entire ERTS components on their intra-node layer part.

In another case, the replication heuristic replaces the standard scheduling heuristic included by default in the static distributed schedule class, and it is used to produce a schedule. As result a set of fault tolerant scheduling tables is generated, it indicates the distribution order of the tasks over the ERTS entire components on their intra-node layer part, under a replicated philosophy.

4.2.2.2. Global time estimation

The ERTS that use some kind of schedule require that all components included in the ERTS being aware of the time on which the system is being executed, with the purpose of execute task in the accurate time in which it is established.

The master component defines the global time of the ERTS by maintaining a clock (physical or logical one [TV2002]). For simplicity, the definition of the global time is concerned to the functional aspects of the master component and it is included in its intra-node layer part.

The other components that take part on the ERTS must keep track of the time. They implement a mechanism for calculating a local time and eventually synchronize themselves with the global time of the master component. Again, the definition of the local time and the synchronization mechanisms are concerned to the functional aspects of the each node and they are included in

their intra-node layer part.

4.2.3. Time component in communication layer

The time component implements the communication layer only for the time-triggered and mixed-triggered buses with the purpose of sending reference messages at predefined moments. There can be from one to several time components within and ERTS. The time component implements the time-triggered class and the mixed-triggered class included in the communication layer.

A time component maintains a reference schedule table that indicates the accurate time when it is supposed to send reference messages. The reference schedule table is generated for the master component of the ERTS and transmitted to it. The reference messages can indicate the beginning of a round or a communication cycle, depending on the implemented class (time-triggered or mixed-triggered); in addition a time component contain the global time value to help other components to estimate their local time according to it.

4.2.4. Normal component in network layer

The network layer part, of the entire ERTS components, is always used. All components involved in an ERTS are considered normal components, independently if they are at the same time master components or time components. Normal components, no matter their associated functions, must implement, on their network layer part, the two principal classes included on the network layer which are the control unit class and the communication controller class.

4.2.4.1. Control unit

The control unit class is extended for the time-triggered node class, the event-triggered node type class, and the mixed-triggered node type class, in order to adapt the normal component to the selected communication type, which means, for a time-triggered communication there is a time-triggered component that matches, and the same for the event-triggered, and the mixed-triggered.

When master component or time component are not executing their particular functions, they act like normal nodes. Independently from the mission of generate and transmit schedule tables and priorities, a master component can execute functional operation of the ERTS, in other words, it can execute tasks as if it was a normal component; of course, only in the cases when functional tasks does not interfere with the particular operations associated to a master component. In the same form, independently from the mission of transmit reference message at determined time, a time component can execute functional operation of the ERTS, in other words, it can execute tasks as if it was a normal component; of course, only in the cases when functional tasks does not interfere with the transmission of reference messages.

4.2.4.2. Communication controller

The communication controller class is simpler than the control unit class; in general the entire components included in an ERTS must implement the communication controller class, no matter the communication type that they implement or their particular functions. The reason for this generalization is that data and error frames classes are implemented with the same format independently of the message type that is going to be included on them.

4.3. Configuration phase

The objective of the configuration phase is first to define the policies to define every component of an ERTS as a master, time, or normal component; and second, to define the execution steps for the classes instantiation in order to leave an ERTS ready for its execution.

4.3.1. Components selection

We had specified previously that any ERTS includes one master node, one or more time nodes,

and several normal nodes. The selection of which component of an ERTS is going to operate as master, time, or normal component is responsibility of the ERTS designer. Figure 4.3 illustrates an example of the components selection on an ERTS architecture.

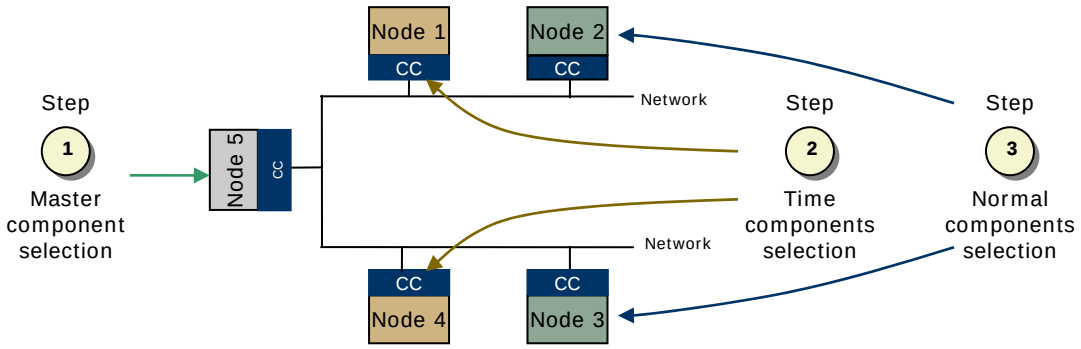

Figure 4.3 Components selection on an ERTS architecture

Figure 4.3 illustrates the ERTS architecture of the part C of the figure 4.1. The selection is organized in three steps; the first step is to select the master component (component 5 in the figure), the second step is to select the time component or components (components 1 and 4 in the figure), and the third step is to select the normal components, which are the rest of components that has not being selected yet (components 2 and 3 in the figure).

Time node duplication
Time node plays a vital roll within an ERTS that implements a time-triggered or mixed-triggered communication, for that reason sometimes it is possible to duplicate the time component, if the ERTS designer requires it. This is done by predefining more than one ERTS component to be potential time component. For example, in figure 4.3 component 1 and component 4 are selected as candidates to be the time component, but only one of them can be time component at a time, for that reason component 1 becomes the primary time component.

All the candidates to be the time component have an associated priority, assigned by the ERTS designer at the moment of their selection. When they recognize, by their reference schedule tables, that it is time to a reference message to be sent they check if there is already traffic on the bus and if there is already a reference message sent. If not the potential time component sends a reference message with its identifier and it is assumed that it is the primary time component.

Whenever a reference message with a higher priority is received the actual primary time component stops sending the reference message and synchronizes with the communication or round cycle given by the higher priority time component. Whenever a reference message with a lower priority is received the potential time component first synchronizes with the existing communication or round cycle and tries to become the primary time component by sending its own reference message at the start of the next cycle. The selection mechanisms presented before ensures that out of all error free potential time components the one with the highest priority eventually becomes active time component without violating the structure of the communication or rounds cycles.

4.3.2. Fault tolerant ERTS framework into ERTS components

We had defined the particularities of the master, time, and normal components and we had also defined the policies for the components selection on ERTS architectures. Now it is possible to define the final steps that every component must follow in order to leave the ERTS ready for starting its execution. The steps vary depending on two of the three principal framework layer (communication and intra-node), because in this case the network layer depends on the other two.

4.3.2.1.Communication layer with a time-triggered bus approach

This case is composed by four steps:

1. A particular component (chosen by the ERTS designer) is defined as master component.
    a. It implements the classes of a normal component.
    b. It defines the time division multiple access protocol.
    c. It receives as input (from the ERTS designer) the required information to be implemented as a time-triggered master component type.
2. The time-triggered master component generates
    a. The global schedule table and the particular schedule tables directed to each component.
    b. The reference schedule table.
3. One or several components (chosen by the ERTS designer) are defined as time components.
    a. They implement the classes of a normal component and after the particular functions of a time component.
    b. The time-triggered component sends to them the reference schedule table.
4. The rest of components that has not being considered yet are defined as normal components.
    a. They implement the classes of the network layer to become normal components.
    b. They implement the functional operations necessaries to produce the tasks that correspond to them. The functional operation codes are given by the ERTS designer, and the manner to know which tasks correspond to each component is located on the input information given at step 1.

4.3.2.2. Communication layer with an event-triggered bus approach

This case is composed by three steps:

1. A particular component (chosen by the ERTS designer) is defined as master component.
    a. It implements the classes of a normal component.
    b. It receives as input (from the ERTS designer) the required information to be implemented as an event-triggered master component type.
2. The event-triggered master component generates the template of a dynamical global schedule table that will be filled in execution time.
3. The rest of components that has not being considered yet are defined as normal components.
    a. They implement the classes of the network layer to become normal nodes.
    b. They implement the functional operations necessaries to produce the tasks that correspond to them. The functional operation codes are given by the ERTS designer, and the manner to know which tasks correspond to each node is also given by the ERTS designer.

4.3.2.3. Communication layer with mixed-triggered bus approach

This case is composed by four steps:

1. A particular component (chosen by the ERTS designer) is defined as master component.
    a. It implements the classes of a normal node.
    b. It defines the time division multiple access mixed protocol.
    c. It receives as input (from the ERTS designer) the required information to be implemented as a mixed-triggered master component type.
2. The mixed-triggered master component generates
    a. The mixed global schedule table.
    b. The reference mixed schedule table.
3. One or several components (chosen by the ERTS designer) are defined as time components.
    a. They implement the classes of a normal component and after the particular functions of a time component.
    b. The mixed-triggered component sends to them the reference mixed schedule table.
4. The rest of components that has not being considered yet are defined as normal components.
    a. They implement the elements of the network layer to become normal components.
    b. They implement the functional operations necessaries to produce the tasks that correspond to them. The functional operation codes are given by the ERTS designer, and the manner to know which tasks correspond to each node is located on the input information given at step 1.

4.3.2.4. Node layer with time-triggered communication

The more complex case occurs when it is desired to instantiate the three framework layers. An important observation is that the intra-node layer only can be instantiated with time-triggered communication, because the tasks distribution over the components and the schedule that it produces is fixed, and it cannot be altered in execution time (it does not accept dynamic factors).

This case is composed by four steps:

1. A particular component (chosen by the ERTS designer) is defined as master component.
    a. It implements the classes of a normal component before implementing the necessary classes to be classified as a time-triggered master component type.
    b. It defines the time division multiple access protocol.
    c. It implements the intra-node layer to generate the statically distributed schedule table or the set of fault tolerant scheduling tables (depending if the heuristic class is implemented). With the produced schedule table is possible to know the relation of which tasks correspond to every component in the ERTS.
    d. With the previous result it is possible to define the master component as a time-triggered master component.
2. The time-triggered master component generates.
    a. The global schedule table and the particular schedule tables directed to each

component.

b. The reference schedule table.

3. One or several components (chosen by the ERTS designer) are defined as time components.

   a. They implement the classes of a normal node and after the particular functions of a time component.

   b. The time-triggered component sends to them the reference schedule table.

4. The rest of components that has not being considered yet are defined as normal components.

   a. They implement the classes of the network layer to become normal components.

   b. They implement the functional operation codes necessaries to produce the tasks that correspond to them. The functional operations are given by the ERTS designer, and the manner to know which tasks correspond to each component is given by the schedule table produced when the intra-node layer was implemented.

4.4. Discussion

This chapter presented the policies to integrate the fault tolerance layers in an ERTS. Independently of have defined the fault tolerance layer model for ERTS it was necessary to have a model that enables the instantiation of such layers. This model was called component model and it divides the nodes that compose an ERTS in three types of nodes, which are master, time, and normal nodes.

The component model defines a hierarchy by nodes importance. It begins defining the master node that controls the configuration process and distributes the corresponding schedule tables to the rest of the nodes of the ERTS. It next defines the time node, just for the approaches that require it, that controls the time access over the bus. And finally, it defines the normal node that implements the applicative functions of the ERTS.

The instantiation is obtained by a configuration phase that defines a set of steps that establish the directives for the selection of the ERTS nodes as master, time, or normal nodes. The configuration phase steps vary depending on the selected fault tolerance layer with the finality of distribute the different types of nodes and implement on them the necessary fault tolerance elements. The configuration phase is relatively a simple process, understanding as a simple process like that with a few steps number in order to configure an ERTS; the maximum number of configuration steps is of four.