Distributed Sensor Fusion Networks

Wilfried Elmenreich and Philipp Peti

Institut für Technische Informatik,
Technische Universität Wien, Vienna, Austria
{wil,pp}@vmars.tuwien.ac.at

Abstract — *This paper introduces a framework for processing sensor measurements with respect to their dependability. Each measurement is represented as an observation, i. e., a compound of a name, a measurement instant, the measured value, and a confidence marker indicating the dependability of value and instant. Sensor observations are processed by a network of fusion nodes in order to produce data with higher confidence. The confidence of each measurement is attached to the transmitted data by the confidence marker. The sensor fusion algorithms use a probabilistic model for sensor readings where the expected variance of a measurement corresponds directly to its confidence.*
The presented approach supports a modular system development, because the fusion algorithms are implemented only with respect to the interface specification and thus are independent of the actual control application. Second, it is possible to extend existing sensor applications with fusion tasks. Given that the necessary timing constraints can be satisfied, the modified application will show the same temporal behavior as the original application.

1 Introduction

Reactive computer systems interacting with their environment via sensors and actuators require a certain degree of reliability regarding the sensor information. However, since there is no such thing as a perfect sensor [1], such systems need to employ means of improving the data quality. To achieve this goal, different, and often redundant, information sources are fused to form a dependable perception of the environment.

Several reasons apply for a distributed design. First, the sensors of the application likely will be placed spatially apart from each other, so that a centralized approach will suffer from noise picked up by transmission of analog signals over long wires. Second, for a dependable systems it is unacceptable that a failure of a single component implies the failure of the whole system. Therefore, a fault-tolerant design with redundant components is desirable.

A further issue arising for reactive systems are real-time requirements. Real-time implies that the time instant, when a value was measured is of similar importance as the measurement value itself [2]. A measurement from a fast-changing property, for example, is of limited use, if the exact instant of the measurement is unknown.

It is the objective of this paper to present an approach for building distributed real-time sensor fusion networks. All data is generically modeled as a compound of a name, a measurement instant, the measured value, and a confidence marker indicating the quality of the data.These data is processed by fusion operators in order to produce a reduced set of data with higher quality. The presented framework supports the implementation of distributed small, controllable subsystems that interact with each other via a time-triggered communication system.

The remainder of the paper is organized as follows: The following Section 2 describes existing approaches found in the literature. Section 3 describes the architectural framework supporting operations with selective dependability. Section 4 examines some algorithms for the proposed operations. Section 5 sketches the integration of the proposed functions into the time-triggered fieldbus network TTP/A. The paper is concluded in Section 6.

2 Related Work

A scheme for confidence markers in digital systems is presented by Parhami in [3]. The proposed approach attaches so-called dependability tags to each data object and updates these tags according to operations performed on these data objects.

Another idea that contributed to the work in this paper is given by sensor validation for fieldbus nodes. So-called self-validating sensors are able to provide a standardized digital signal and generate diagnostic information. In the Oxford SEVA system [4], each measurement is delivered as a validated value together with the validated uncertainty and a measurement value status.

Finally, if there are multiple measurements of a property in a technical process, the question arises, how to further process these measurements until the data supports the required level of dependability. We assume that the taken measurements will have some degree of redundancy. Thus, it is possible to apply voting or competitive fusion algorithms [5] to refine the data.

3 Generic Sensor Fusion Framework

This section presents a generic framework for building sensor fusion networks. First we identify the relevant properties of a sensor observation and introduce so-called fusion operators, which act as a black box that consumes and produces observations. Subsequently, we will discuss the implementation of fusion operators with a weighted averaging and a selection algorithm.

3.1 Sensor Properties

A sensor can be seen as a small window providing a view of a property of a technical process. When modelling this view, often only the measured value is considered. We require a more comprehensive view of a sensor measurement that takes the following properties into account:

Value: Value denotes the result of a measurement. Generally, such a value can be discrete or continuous. We assume that all values are given in a digital representation.

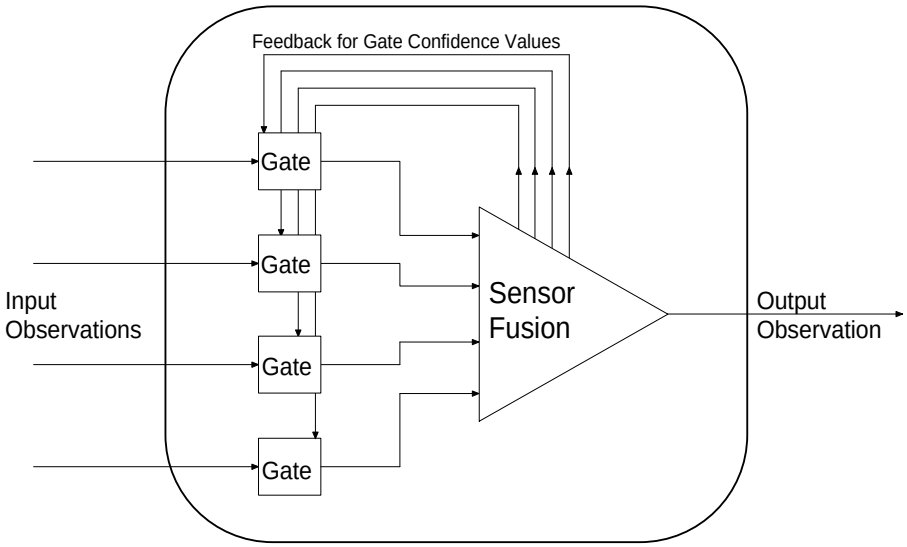

Figure 1: Structure of fusion operator

Description: Which property of the process has been measured? Which units are used? Which sensor has provided the measurement? In many systems the description of a sensor is not stored explicitly but defined implicitly in the way the system processes the sensor value. Usually, a description is static. Description properties can be multifaceted and therefore are difficult to model. There exist approaches using self-describing XML (Extensible Markup Language) data to provide efficient modelling and tool support for transducer networks [6].

Instant: The time when the value was observed. In a distributed system it is required that different nodes share a common notion of time in order to agree on the semantics of a given instant.

Confidence: The confidence is a dynamic property that annotates the quality of a value and instant. Quality, in this context, can denote uncertainty, precision, or jitter. Confidence values can be created in self-validating sensors or be derived by comparing multiple measurements of the same property. The confidence is influenced by static properties such as the accuracy of a sensor given in it data sheet as well as by dynamic properties, e. g., switching of metering ranges, sensor deprivation in a multi-sensor system, aging effects, etc.

3.2 Fusion of Observations

Based on the concept of an *observation* defined by Kopetz in [2], we extended the notion of *observation* by a confidence part in order to get a compound of

$$<\text{entity name, instant } t\text{, value } x\text{, confidence } c>$$

The *entity name* is a key to the description, the instant t defines the point in time when the respective measurement was made, x represents the measured value, and c is a confidence value that expresses the estimated dependability of instant and value. Since each

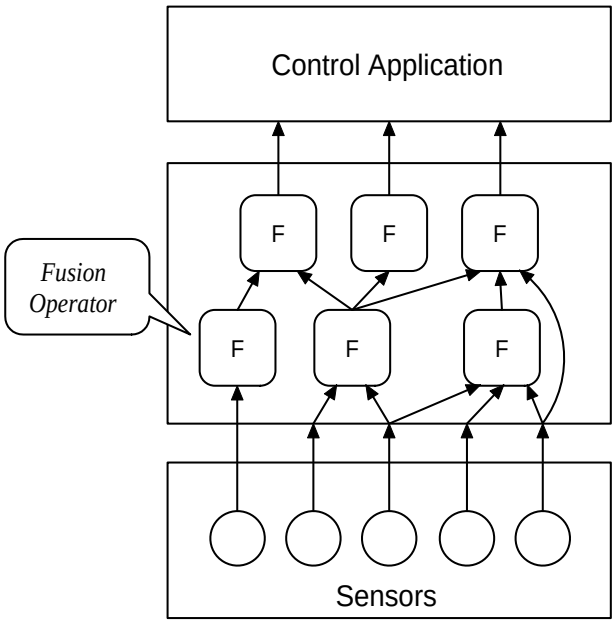

Figure 2: Structure of sensor fusion layer

observation has assigned a measurement instant, an observation is valid regardless of the time when it is processed.

When a new measurement is performed, the confidence value is introduced by the smart sensor. A self-validating sensor derives this confidence value as a result of a self-diagnosis function. If a sensor does not support self-validation, the confidence value is set to a standard value according to the *a priori* estimated average reliability of the sensor.

A *fusion operator* (depicted in figure 1) processes at least one observation as input and produces an observation as output. Several fusion operators can be clustered in an abstract network. Such fusion networks can be hosted on one or several physical fieldbus nodes. Besides the confidence value in the input observations, each input is assigned a *gate confidence value*. While the assignment of the observation confidence value is in the sphere of control of the data producer, the gate confidence value is in the sphere of control of the fusion operator for the purpose of feedback provision. Both, the gate confidence values and the observation confidence values, are combined in a *gate* to form a new confidence measurement for each observation. The resulting observations are then combined by sensor fusion algorithms.

These algorithms can either produce an enhanced observation of the properties observed by the individual sensors or produce a derived property, e. g., an acceleration value from speed measurements or the slippage by comparing rotation speed measurements of different wheels of a car (Anti-lock braking). Generally, the output observation of a fusion operator can differ in value, instant, and confidence from the input observations. The output observation is always assigned to a virtual sensor, which has an entity name that is different from the entities of the input observation. Fusion operators can be cascaded in a network as depicted in figure 2.

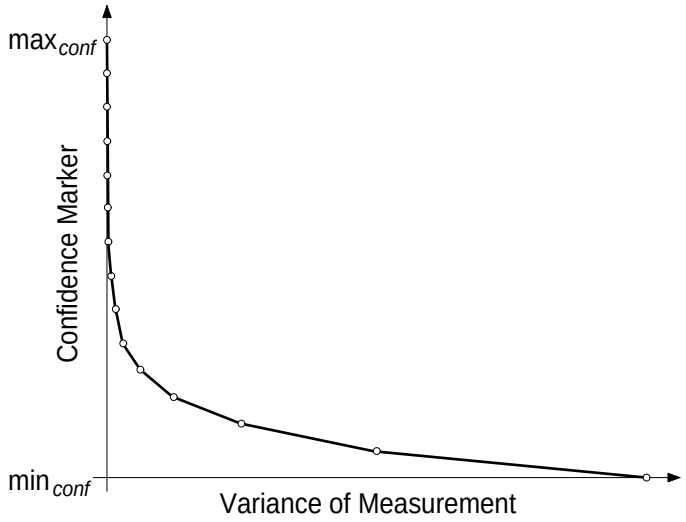

Figure 3: Conversion function for confidence/variance values (based on a logarithmic scale)

Since we assume an underlying digital network communication system, it is possible to copy the information of one observation and use the same sensor observation several times in different fusion operators. The control application uses at least one observation as input. It depends on the implementation of the control application if it uses the assigned confidence value when making control decisions.

3.3 Meaning of Confidence Values

The confidence measure will be introduced as an integer value between 0 and $conf_{max}$, where 0 is defined to be the lowest confidence and $conf_{max}$ is the highest confidence.

The confidence marker is interpreted as an estimator of the statistical variance $\mathbb{V}[S]$ of the measurement error.[1] The variance is the second moment of an arbitrary probability density function.

Using a linear transformation between confidence values and variance is not expedient, since the variances that indicate exact measurements are of greater interest than measurements with large variance. Therefore, we use a logarithmic scale to define the confidence values between min_{conf} and max_{conf} as depicted in Figure 3. Due to the expected computational load when doing logarithmic and exponential operations on embedded systems, we suggest the implementation of look-up tables for the conversion from confidence value to variance.

4 Fusion Algorithms

This section presents a set of fusion algorithms that can be used in a fusion operator. Since the presented architecture is open to custom implementations of fusion operators, it is easily possible to integrate other fusion methods (e. g., Kalman filter [8]) into the

[1]The *Guide to the Expression of Uncertainty in Measurement* [7] suggested statistical variance as a measure for uncertainty.

framework.

4.1 Confidence-Weighted Averaging

Averaging is one of the simplest forms of fusion. It comes with the advantages of noise reduction and simple implementation. We assume the incoming observations to be taken from a continuous entity. All observations under consideration must be made at approximately the same instant. The error functions of the incoming sensors are considered to be independent. The measurement values are fused by using a weighted average with the reciprocal variance values as weights:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i \cdot \frac{1}{\mathbb{V}[S_i]}}{\sum_{i=1}^{n} \frac{1}{\mathbb{V}[S_i]}} \tag{1}$$

where n is the number of input observations, x_i represents the measurement values and $\mathbb{V}[S_i]$ is the estimated variance for that measurement. The variance values are derived from the confidence marker using the conversion function depicted in Figure 3. The confidence value for the fused value is calculated according to equation 1 yielding the statistical variance of the output observation:

$$\mathbb{V}[S_O] = \frac{1}{\sum_{i=1}^{n} \frac{1}{\mathbb{V}[S_i]}} \tag{2}$$

The resulting variance is then converted to a confidence marker. The variance of the output values is always lower or equal than the variance of the best input observation. If the error independence condition is not true, the fused value is still correct, but its assigned confidence is overestimated. The error independence condition is difficult to fulfill when identical sensors or sensors of the same type of construction are fused. Thus, usually the best performance is achieved, when all input observations have the same confidence value, but stem from heterogeneous sensors.

A possible extension to confidence-weighted averaging is the application of a fault-tolerant averaging algorithm. This algorithm removes the t largest and the t smallest data values and then performs the weighted average as described above. Thus, at least t faulty measurements can be tolerated. The input set must consist of at least $2t + 1$ observations.

Finding the most distorting values among the variance-weighted values affords the following steps: First, the weighted average value $\bar{x}$ over all input observations has to be calculated according to equation 1. The values to be removed can be determined by finding the observations with the t largest and the t smallest values for $\frac{\bar{x}-x_i}{\mathbb{V}[S_i]}$. After removing $2t$ observations, the weighted average value and its corresponding confidence value can be derived.

The complexity of such an algorithm is $\Omega(n \log n)$, or, when regarding the number of faults, $\Omega(nt)$. This complexity still allows fast and efficient implementations even for great n and t, but the fault-tolerant averaging has a disadvantage in our application context. The confidence-weighted averaging algorithm performs better with an increasing number of inputs. Thus, removing t input values affects the gain of the fusion operation,

so that there is a tradeoff between the number of faults to be tolerated and the performance of the sensor fusion algorithm.

4.2 Selection

Some applications (e. g., classification) need to select one input out of a set of input observations. Usually, voting algorithms [9] are a good means to choose an appropriate output with improved dependability compared to the input observations.

A simple algorithm for *exact* voting using observations is described in the following. First, the reciprocal variance values of observations with identical values are added together in order to form a single variation value for each alternative. Then we select the first alternative (i. e., the one corresponding to the lowest value) with minimum variation. This selection method fulfills the following fairness criteria [10] for voting methods and has the property of replica determinism [11]:

Condorcet criterion: [2] *If there exists an alternative* A *that wins in pairwise votes against each other alternative, then* A *should be the winner of the election [10, page 1].*

Sketch of proof: An alternative *A* will only win against a different alternative *B*, if its variation is smaller than the variation of *B* or its variation is equal to the variation of *B* and the value of alternative *A* is smaller than the value of *B*. Since two alternatives will always differ at least in their values and "smaller than" is a transitive proposition, the Condorcet criterion will be fulfilled for all winning alternatives *A*.

Monotonicity criterion: *If alternative* A *is declared the winner under a voting method, and one or more voters change their preferences in a way to favor* A *(making no other changes), then* A *should still win [10, page 1].*

Sketch of proof: If an input that initially proposes an alternative *B* switches to the winner alternative *A*, then the variation of *A* will decrease, while *B*'s variation will either increase or *B* will drop out of the set of proposed alternatives. Thus, *A* will still win over *B*. Since the variations of all the other alternatives remain unchanged, alternative *A* will also win over all other alternatives.

Replica determinism criterion: *Every pair of two voters that get the same input data will elect the same winner consistently.*

Sketch of proof: Provided that all voters are using the same look-up table and arithmetics, all voters will calculate identical variations for every alternative in all voters. Since the algorithm itself is deterministic, it is thus guaranteed that replica determinism is maintained among multiple voters.

All three criteria are fulfilled by the proposed selection algorithm with the ancillary condition that all voters are provided with exactly the same input data. Hence, the fulfillment of the above criteria relies on a digital communication system that has to provide a reliable broadcast mechanism.

[2] Marquis de Condorcet, French mathematician of the eighteenth century

4.3 Fusing Observations from Different Instants

Synchronizing measurements and performing measurements at periodically *a priori* defined instants is an inherent capability of a time-triggered architecture as proposed in the following section. Thus, the k-th observation on an entity is guaranteed to be taken at the instant $t_k = t_0 + k \cdot \Delta t + \epsilon$ where t_0 is the first instant of measurement, Δt is the interval between two consecutive measurements and ϵ is the time jitter. The jitter is comparatively small in time-triggered systems ($|\epsilon| < \Delta t/100$). Therefore, we can ensure that a set of observations delivered by different sensor nodes are taken at the same instant. However, in case of omission faults, we have to deal with a set of observations with differing instants: some observations may have been updated to instant t_k and some observations which failed to be transmitted are only available as an older version t_{k-1}.

If the redundancy in the system is high enough, a convenient method to solve this problem is to keep only the most recent observations and discard the others. The fusion algorithms are then performed as described before with a reduced set of observations. This method is principally applicable if the periodical measurements happen on a sparse time set. Otherwise, if observations can take place at any instant of a fine-grained time line it is likely that only one observation remains as input for the fusion algorithm.

As an alternative to dropping old observations, a history of observations can be used to extrapolate the state of the real-time entity for the desired instant. Thus, all missing observations can be compensated with respective estimated values. The confidence of an extrapolated value should be respectively low due to the high expected deviation between real value and estimated value. Moreover, such practise can be critical if the output observation is used in feedback loops.

5 Application in a Fieldbus Network

A fieldbus network connects transducers (sensors and actuators) to a control system. While in former approaches transducers where point-to-point connected and instrumented by analog signals, actual fieldbus systems use a digital communication medium to interconnect the transducers with the control system. We decided to insert the sensor fusion processing between the transducer nodes and the control system. Since future fieldbus nodes are equipped with a local microcontroller for signal conditioning, it would be possible to implement the local sensor signal processing directly at the sensor. Conceptually, the sensor fusion will be strictly separated from the control application in order to avoid increasing complexity for the control application [12].

As a case study we used the time-triggered master-slave fieldbus protocol TTP/A. TTP/A is designed for predictable real-time communication in non-critical applications in the automotive and automation sector. The protocol [13] uses a time division multiple access (TDMA) bus arbitration scheme, which meets timing requirements for typical sensor fusion algorithms [14].

It is possible to address up to 254 nodes on a bus. One single node is the active master. This master provides the time base for a synchronized global time among all slave nodes. The communication is organized into rounds. A round consists of several slots. A slot is a unit for transmission of one byte of data. Data bytes are transmitted in a standard UART format. Each communication round is started by the master with a so-called fireworks

byte. The fireworks byte defines the type of round.

A TTP/A round (see Figure 4) consists of a configuration dependent number of slots and an assigned sender node for each slot. The configuration of a round is defined in the RODL (ROund Descriptor List). The RODL defines which node transmits in a certain slot, the semantics of each individual slot, and the receiving nodes of a slot. RODLs must be configured in the slave nodes prior to the execution of the corresponding round.

The TTP/A protocol offers a unique addressing scheme for all relevant data of a node like communication schedules, calibration data, and I/O properties. This addressing scheme is called Interface File System (IFS) [15]. The IFS provides a universal interface to the TTP/A network for configuration and maintenance tools as well as for applications running local on a node. The IFS is structured in a record-oriented format. The smallest addressable unit is a record of 4 bytes. All nodes contain several files with a number of records that can contain information for automatic configuration.

TTP/A supports data types for 8 bit and 12 bit digitized analogue data. It is possible to assign a measurement a four-bit confidence marker referring to the quality of the sensor observation. These confidence markers will be used to represent the quality of an observation as described in Section 3. Due to the fact, that the whole communication and computation is time-triggered with respect to a global time, the observation instant of each observation is known a priori to all nodes and may not be transmitted. The observation context is defined by its address in the IFS.

Figure 5 depicts a sample communication in TTP/A. Node *A* contains a sensor and a local IFS, node *B* contains the control application. The IFS acts as a temporal firewall [16] between communicating entities as follows: The sensor performs a measurement at instant t_{m1}. At time $t_{t1} > t_{m1}$ the protocol transports the data from the local IFS of node *A* to the local IFS of node *B*. The data arrives at time t_{a1} at node *B*. The control application can now use the sensor data until it is overwritten with a new value at time t_{a2}. Usually the control application has to hold a deadline t_{c1} to output a control signal. The output of the control application is also written to the IFS and further transported at time t_{c1} by the time-triggered protocol to the actuators of the system. Figure 7 shows the instants of communication in a timing diagram. The establishment of the temporal firewall is, that the task of the control application may be scheduled any time between t_{a1} and $min(t_{a2}, t_{c1})$ as long as the result is written to the IFS until the end of this duration.

By using knowledge about the timing behavior, it is possible to introduce sensor fusion into an existing TTP/A application. In our implementation, the fusion algorithms will be implemented on the same fieldbus node as the control application.

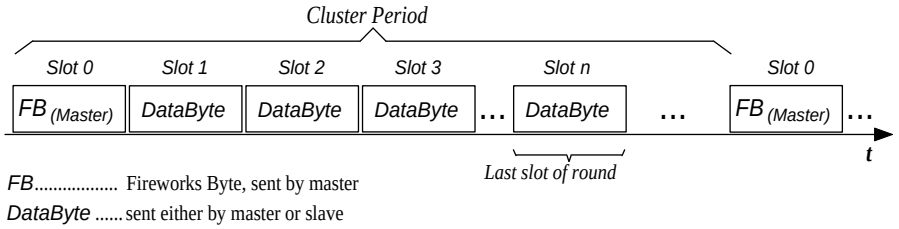

Figure 4: TTP/A Communication

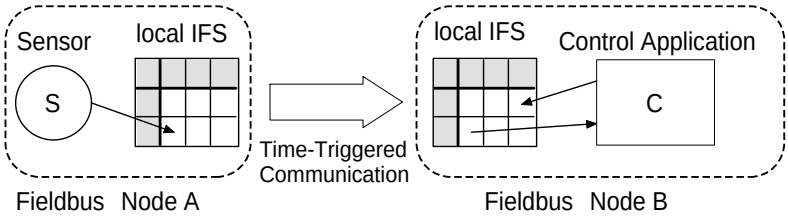

Figure 5: Data Processing in TTP/A

Figure 6 depicts an extension of the above described application with a fusion node. The fusion node is a virtual node implemented together with the control application in fieldbus node B. The incoming data is preprocessed by a fusion node until it is fed into the control application.

Taking the timing constraints from the above example we now arrive at the constraint in equation 3:

$$WCET_{fusion} + WCET_{control} \leq min(t_{a2}, t_{c1}) - t_{a1} \tag{3}$$

where $WCET_{fusion}$ is the worst case execution time of the fusion process and $WCET_{control}$ is the worst case execution time of the control application.

Given that the necessary timing constraints can be satisfied, the sensor fusion process is transparent to the control application process. The main advantages of this architecture are:

No changing of the network configuration: Existing TTP/A networks can be used to host sensor fusion algorithms. The system behavior will thus be improved without adding extra nodes. This eases the installation of sensor fusion methods significantly since no adaption of existing configuration data (RODL files) is needed.

Reuse of existing control applications: The presented approach can be used to transform an application that does not tolerate sensor faults into one that does. If the WCET analysis guarantees the timing constraint (Eq. 3) the modified application will show the same temporal behavior as the original application.

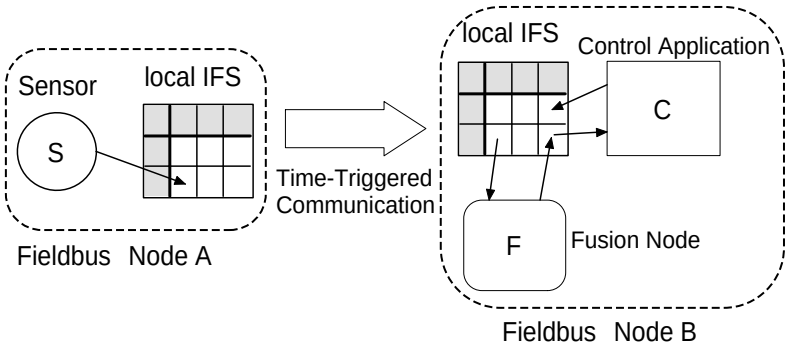

Figure 6: Data processing with fusion processor inserted

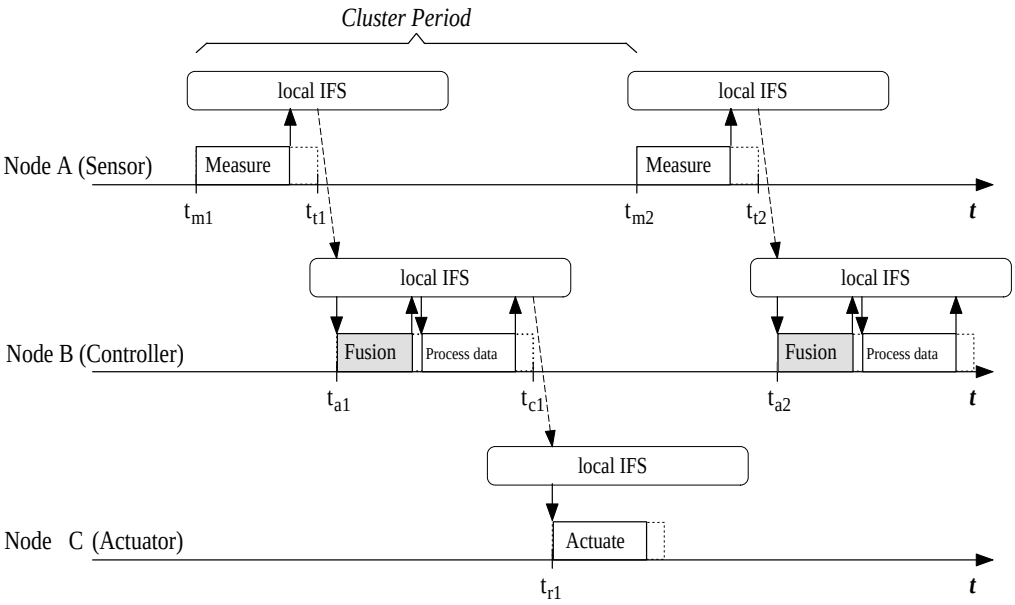

Figure 7: Communication and computation instants among three nodes

Fusion algorithm is independent of control application: The strict separation of the fusion process and the control application allows a modular system development. Thus small, controllable subsystems can be implemented and tested separately and finally joined together to form the final system.

Use of generic algorithms: The fusion algorithm is independent of the application, thus generic fusion algorithms can be used for a variety of applications.

There is still a possibility to enhance existing network without extensions to hardware and under reuse of existing software. However, this property cannot be guaranteed generally. If a system is already so compact, that there is no redundancy in sensor measurements and no space on the fieldbus controller to host the sensor fusion task, extensions are necessary anyway. Many applications, however, provide some redundancy or free resources. For those cases, our approach allows a smooth integration of the additional functionality.

6 Conclusion

We presented an architecture for processing sensor measurements with respect to their dependability. Each measurement is represented as a compound of a name, a measurement instant, the measured value, and a confidence marker indicating the reliability and preciseness. Sensor observations are processed by a network of fusion nodes in order to get data with higher confidence. The confidence of each measurement is attached to the transmitted data in form of the confidence marker. The sensor fusion uses a probability model of sensor readings where the expected variance of a measurement corresponds directly to its confidence. Besides the fusion of different values taken at approximately the same instant the paper presents also approaches for fusion observations of different instants.

We examined methods for integration of the presented sensor fusion algorithm into ex-

isting application in order to transform a real-time application that does not tolerate sensor faults into one that does. An important point is the stability of the timing behavior after the integration which is supported by a time-triggered communication system featuring a temporal firewall at each communication action. An analysis of the worst-case execution time of the resulting application is used to verify the timing constraints of the original application. When the necessary timing constraints can be satisfied, the modified application will show the same temporal behavior as the original application.

Since our approach allows the joining of fusion nodes with existing tasks into existing hardware nodes, resources in existing TTP/A networks can be used to host systems with improved behavior.

Acknowledgments

We would like to give special thanks to our colleague Wolfgang Haidinger who acted as mathematical advisor when we were at our wits' end. This work was supported in part by the Austrian Ministry of Science, project TTSB and by the European IST project DSoS under contract No IST-1999-11585.

References

[1] B. Liptak. OEM insight: Get serious about accuracy. *Control Design for Machine Builders*, May 2001.

[2] H. Kopetz. *Real-Time Systems, Design Principles for Distributed Embedded Applications*. Kluwer Academic Publishers, Boston, Dordrecht, London, 1997.

[3] B. Parhami. A data-driven dependability assurance scheme with applications to data and design diversity. In A. Avizienis and J. C. Laprie, editors, *Dependable Computing for Critical Applications*, volume 4, pages 257–282. Springer Verlag, Vienna, 1991.

[4] M. Henry. Sensor validation and fieldbus. *Computing & Control Engineering Journal*, 6(6):263–269, December 1995.

[5] A. Visser and F. C. A. Groen. Organisation and design of autonomous systems. Textbook, Faculty of Mathematics, Computer Science, Physics and Astronomy, University of Amsterdam, Kruislaan 403, NL-1098 SJ Amsterdam, August 1999.

[6] S. Pitzek. Description mechanisms supporting the configuration and management of TTP/A fieldbus systems. Master's thesis, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, 2002.

[7] International Organization for Standardization (ISO), Genève, Switzerland. *Guide to the Expression of Uncertainty in Measurement*, 1st edition, 1993.

[8] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the ASME, Series D, Journal of Basic Engineering*, 82:35–45, March 1960.

[9] B. Parhami. Voting networks. *IEEE Transactions on Reliability*, 40:380–394, August 1991.

[10] M. Corks. Evaluating voting methods. Survey paper, University of Waterloo, 1997.

[11] S. Poledna. *Replica Determinism in Fault-Tolerant Real-Time Systems*. PhD thesis, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, 1994.

[12] W. Elmenreich and S. Pitzek. The time-triggered sensor fusion model. In *Proceedings of the 5th IEEE International Conference on Intelligent Engineering Systems*, pages 297–300, Helsinki–Stockholm–Helsinki, Finland, September 2001.

[13] H. Kopetz et al. Specification of the TTP/A protocol. Technical report, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, March 2000. Available at http://www.ttpforum.org.

[14] W. Elmenreich and S. Pitzek. Using sensor fusion in a time-triggered network. In *Proceedings of the 27th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 369–374, Denver, CO, USA, November–December 2001.

[15] H. Kopetz, M. Holzmann, and W. Elmenreich. A universal smart transducer interface: TTP/A. *International Journal of Computer System Science & Engineering*, 16(2):71–77, March 2001.

[16] H. Kopetz and R. Nossal. Temporal firewalls in large distributed real-time systems. *Proceedings of the 6th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS '97)*, pages 310–315, 1997.

About the Authors

Wilfried Elmenreich was born in 1973. Before going to the Vienna University of Technology, he studied at the Engineering School for Electrotechnics and Control in Weiz, Styria. In 1998, he received a Master's degree in computer science and in 2002 a doctoral degree in technical sciences both from the Vienna University of Technology in Austria. His research interests include real-time and embedded systems, smart transducer networks, fieldbus communication, and sensor fusion.

Philipp Peti received his Master's degree in computer science in 2001 from Vienna University of Technology in Austria. His research focuses on networked embedded systems, fault-tolerant systems and real-time systems. Currently, he is working on his PhD thesis on diagnosis in distributed embedded real-time systems.