PAPER *Special Issue on Dependable Computing*

Analyzing the Impact of Data Errors in Safety-Critical Control Systems*

Örjan ASKERDAL†, **Magnus GÄFVERT**††, **Martin HILLER**†††, *and* **Neeraj SURI**††††, ***Nonmembers***

SUMMARY Computers are increasingly used for implementing control algorithms in safety-critical embedded applications, such as engine control, braking control and flight surface control. Consequently, computer errors can have severe impact on the safety of such systems. Addressing the coupling of control performance with computer related errors, this paper develops a methodology for analyzing the impacts data errors have on control system dependability. The impact of a data error is measured as the resulting control error. We use maximum bounds on this measure as the criterion for control system failure (i.e., if the control error exceeds a certain threshold, the system has failed). In this paper we a) develop suitable models of computer faults for analysis of control level effects and related analysis methods, and b) apply traditional control theory analysis methods for understanding the impacts of data errors on system dependability. An automobile slip-control brake-system is used as an example showing the viability of our approach.

key words: *safety-critical systems, control systems, error modeling, error analysis*

1. Introduction

With the increasing use of computers for implementing control algorithms, control systems become more vulnerable to computer level failures. Thus, this paper focuses on understanding the impact of computer level data errors on system dependability (Errors caused by faults in the computer nodes are often classified as either data errors, or timing errors. A data error occurs when the computer node delivers data that is incorrect, and a timing error occurs when the computer delivers data at an incorrect point in time. In this paper, we focus the analysis to the impact of data errors on system dependability. Methods for analyzing the effects of timing errors on control systems were presented in e.g., [5], [10].).

Manuscript received March 31, 2003.
Manuscript revised June 27, 2003.

†The author is with the Department of Computer Engineering, Chalmers University of Technology, Sweden.
††The author is with the Department of Automatic Control, Lund Institute of Technology, Sweden.
†††The author is with the Volvo Technology Corporation, Sweden and Technische Universität Darmstadt, Germany.
††††The author is with the Department of Computer Science, Technische Universität Darmstadt, Germany.

*This paper is based on the work presented at IEEE 2002 Pacific Rim International Symposium on Dependable Computing. The research was funded by NUTEK (DICOSMOS P11762-2) and EC NextTTA (IST-2001-32111).

Data errors and their effects on computer functionality is an intensively researched area, e.g., [8]. However, recent results [1], [9] show that many data errors will have a limited impact on control performance, i.e., control systems often have an inherent resiliency or inertia to data errors. These results were obtained experimentally using fault-injection, e.g., [8], for system validation. However, this technique requires a prototype of the system (or at least a detailed model), which generally is not available in the early design phases. *Thus, the specific scope of this paper is to develop a systematic design stage analytical basis for estimating the impacts of different data errors on the control applications.* We envision our approach to be used in early design phases as a design level guide to adapt fault tolerance techniques to enhance control system dependability as needed.

Error effect analysis is an extensively developed area, e.g., [8]. Analysis of effects on system stability of data errors caused by EMI bursts was investigated in [6]. However, as catastrophic failures in safety-critical system may occur before the system reaches instability, we base our definition of system failure on thresholds of the magnitude and duration of the control error, i.e., the difference between the reference (desired) value of a controlled physical process property and the actual value of this property (e.g., in a system controlling cabin temperature in a car, the control error would be the difference between the temperature the driver has requested and the actual temperature measured by the sensors). *The analysis of this paper is focused on finding errors posing to threat the system, i.e., data errors that result in large control errors.*

As an example of error impacts, Fig. 1 shows the output signal of a control system. At start, the output value follows the reference value (desired output), i.e., the control error is 0. Then, at time s, a transient fault occurs affecting the output vale. Depending on which bit (or bits) that are corrupted, the magnitude of the error will differ, as indicated by the solid lines. The maximum acceptable control error, defined by the system designers, taking into account noise levels and other effects, is plotted as dotted lines in Fig. 1. The plot to the left describes a data error in bit(s) with low significance which never exceeds this level, and thus, does not need to be handled, whereas in the right plot,

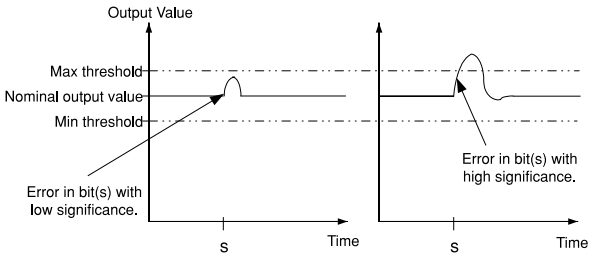

Fig. 1 Transient fault influence on the control error.

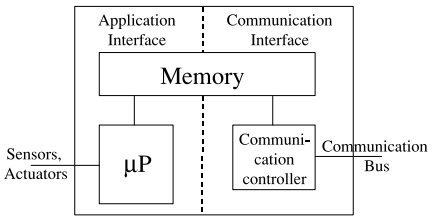

Fig. 2 A General architecture of a computer node.

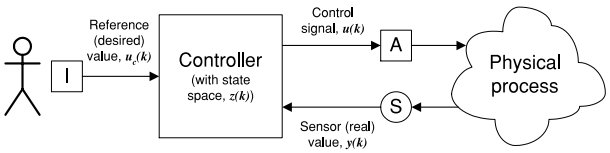

Fig. 3 General depiction of a controller and the corresponding physical process.

an error has occurred in more significant bit(s) and the level is exceeded. With the proposed analysis method, the magnitudes of errors (i.e., which bit errors), that are inherently tolerated by the control system can be estimated. Also, it is possible to estimate the time until the output value has returned to its correct value after an error has occurred.

The organization of this paper is as follows: First, we detail the control system model and related assumptions in Sect. 2. In Sect. 3, we introduce classes of possible error models for different types of computer node faults. In Sect. 4, we describe different analysis methods for determining how the control performance is affected by different types of faults and we exemplify them through a test case in form of a simple automotive slip controller. We summarize the conclusions and discuss future research directions, Sect. 5.

2. Assumptions and Models

A distributed control system typically consists of sensors, actuators, computer nodes and a communication network connecting these components. We restrict this paper to the analysis of the impacts of errors caused by faults occurring in the computer nodes. We primarily consider errors caused by hardware (HW) faults and introduce models for simulation of these (based on data bit manipulation).

The Computer Node Consider the generic computer node shown in Fig. 2. Such a node conceptually contains two interfaces, a communication interface and an application interface. The communication interface handles the information exchange with other nodes (e.g., sensor and/or control signal values). The communication is conducted according to a protocol, stating the packaging and encoding of data. Received data is transferred to the application interface, where it is utilized together with data received from internal data sources (e.g., local sensors) in the control algorithms. The control signals are then transferred to their corresponding actuators.

The actual implementation differs among computer nodes. In some nodes, communication and calculation of control signals may be managed by the same main processor, whereas in others, the main processor only handles calculations and a specific controller manages network communication. For such nodes, data can be transferred between the controller and main processor through an internal bus or a dual-port memory.

Different implementations have different component failure probabilities. *This paper analyzes the impact of data errors occurring in control signals and/or internal states* (*including propagated errors*). It is up to the system designers to estimate the probability of different errors to occur, taking the specific component failure intensities into account. For instance, if internal fault tolerance techniques are used in the various components of the system, such as error correcting codes in memory, the error intensity will be reduced.

The Controller We provide a basic description of a generic controller structure implementing a control function at a level detailed enough to communicate the ideas presented in the paper. The interested reader is referred to the appendices/references for more details.

Conceptually, a control system is set to control a certain physical process (see Fig. 3). This is achieved using a set of actuators (**A** in Fig. 3) for affecting the physical process and sensors (**S** in Fig. 3) for monitoring the effects of the actuators. The user of the controller (which may be a human user or an external computer system) provides the controller with a reference signal $u_c(k)$ via some interface (**I** in Fig. 3). The controller will then attempt to change the physical process with control signal $u(k)$ to the actuators, such that the sensor value $y(k)$ is as close to the reference value as possible, using information about the history of the system (the stored state space $z(k)$. As we consider controllers implemented on microprocessors, we assume that it executes in discrete steps, i.e., the controller reads reference values and sensor values and calculates control signals periodically. This is the case for many dynamic systems which constitutes an important subset of control systems upon which data errors can have severe impacts. In Fig. 3, and subsequently, k indicates the

k^{th} time-step.

Typically, a control system has non-linear effects (a non-proportional relationship between the value of the control signal and the resulting physical value read by the sensors). However, linear models are often used as they in many cases are sufficient approximations at the normal system operating point and reduce the complexity. Thus, we focus the error impact analysis on linear system models in this paper. The generalized feedback controller in Fig. 3 is defined as:

$$\begin{aligned} u(k) &= Cz(k) + D^{u_c}u_c(k) + D^{y}y(k) \\ z(k+1) &= \Phi z(k) + \Gamma^{u_c}u_c(k) + \Gamma^{y}y(k) \end{aligned} \tag{1}$$

Equation (1) describes a generic linear controller with internal states, e.g., a PI-controller (we have a controller with a proportional feedback (P) and an integrated feedback (I)) or a state-feedback controller with observer states and integral action. The control signal $u(k)$ is calculated using the reference value $u_c(k)$, the sensor value $y(k)$, and an intermediate value $z(k)$. This intermediate value is actually a vector constituting the state space (history) of the controller containing for instance integrator states, which are used to provide a certain level of history that guarantee that the correct output is reached despite constant disturbances, and observer states, which are used to estimate signals which influence the control algorithm, but may be impossible to measure. In our example in Sect. 4.1 we use one integrator state and one observer state.

Φ, Γ^{u_c}, Γ^{y}, C, D^{u_c} and D^{y} are matrices containing control constants, used as weights in the equations. The control constants are amplifying factors that are set by the designer to give the desired control performance, e.g., making the physical value assume the reference value as fast as possible (after a change of the reference value) without exceeding the reference value (i.e., overshooting).

Using pseudo-code, Eq. (1) is implemented as:

```
repeat:
   uc := read_from_interface();
   y := read_from_sensors();
   u := compute_control_signal(z,uc,y);
   write_to_actuator(u);
   z := update_controller_state(z,uc,y);
   wait_for_next_sample;
```

Previous research [5], [6] has investigated the effect of data errors on system stability. However, a system may fail before it reaches instability. Thus, the severity of errors is more related to how much the output of the system differs from the desired value (i.e., the control error). Therefore, we alternatively use the following (application dependent) requirements on the control error: **C1:** The control error must not exceed a certain specified limit. **C2:** A certain level of the control error is only acceptable (from a control perspective) for a limited duration. If any of these two requirements is violated, we define the system to have failed.

As most systems have a certain inherent inertia, the control error is dependent on i) the dynamics of the system, ii) how the reference value changes, and iii) external disturbances. Thus, the impact of a fault on the system is not only dependent on the specific fault, but also the current operational state of the system (the current operational state is called the operating point). We assume that the designer has identified the most sensitive operating point and set the requirements on the control error for that point.

Having introduced the general structure of the computer nodes and the controllers, in the next section we discuss how external disturbances, such as errors, can be modeled using the same mathematical framework as the controller equations in Eq. (1).

3. Classification and Modeling of Data Errors

Based on the assumptions presented in Sect. 2, we describe our approach for analyzing the impact of data errors on the system. However, we first discuss how data errors can be modeled using the same mathematical framework as in Eq. (1). Subsequently, in Sect. 4, we present an example control system and describe analysis methods for understanding the impacts of data errors on the control system.

In Sect. 2, we introduced general recurrence equations (see Eq. (1)) for calculating control signals (u) and the state space (z). Data errors can affect the behavior of the controller either by disrupting the calculated control signal (u) and/or the state space of the controller (z). Therefore, we model errors as additive terms as:

$$\begin{aligned} u(k) &= Cz(k) + D^{u_c}u_c(k) + D^{y}y(k) + \boldsymbol{\eta_u(k)} \\ z(k+1) &= \Phi z(k) + \Gamma^{u_c}u_c(k) + \Gamma^{y}y(k) + \boldsymbol{\eta_z(k)} \end{aligned} \tag{2}$$

where $\boldsymbol{\eta_u(k)}$ and $\boldsymbol{\eta_z(k)}$ are the functions describing disturbances due to faults in the computer (i.e., errors) executing the control equations. Henceforth, we define expressions for $\boldsymbol{\eta_u(k)}$ and $\boldsymbol{\eta_z(k)}$ as suited for different error types.

As illustrated in Fig. 2, the internal components of a computer node are: communication controllers, memories, microprocessors, and internal communication buses. All these components may be affected by faults and the impacts of these faults will be dependent on the function of the component, which bit(s) the fault affects (high significance versus low significance) and how often the fault is activated.

As the communication controller handles the information exchange with other nodes, quantities measured by sensors may become erroneous due to faults in the communication controller. In Eq. (1), the reference value (u_c) and the output value (y) are such quantities. Thus, a communication controller fault could result in any of these variables having an incorrect value.

All data used for the calculation of the controller equations are stored in memory circuits. The control constants defining the behavior of the controller (Φ, Γ^{u_c}, Γ^{y}, C, D^{u_c} and D^{y} in Eq. (1)) are likely to be stored in non-volatile memory, such as Flash or EEPROM. Variables (u_c, z and y in Eq. (1)) are stored in RAM. Although the impact of faults affecting memory is very dependent on the implementation of the memory, the most common fault model is to change the value of one or more bits in the memory. Thus, this error will propagate when the memory cell with incorrect data is accessed when calculating Eq. (1).

The data path of a microprocessor consists of caches, registers, busses and functional units (i.e., ALU, multiplier, etc.). If a fault occurs in any of these parts, it will affect the ongoing calculation if the faulty part is activated, rendering the result of that calculation to be incorrect.

The communication across the computer node is generally via data buses. If a fault occurs on such a bus, the data (variables and constants) currently being transferred will be affected and the result may be a data error in any of the calculated data.

On this background of how error propagation affects the calculations of the controller, we next discuss the temporal dimension of data errors and discuss how disturbances/errors of various occurrence frequencies and persistency levels affect the controller.

3.1 Classifying Occurrence Rates of Data Errors

We divide the possible errors (i.e., η_u and η_z, Eq. (2)) into classes based on their rate of occurrence. In Sect. 3.2 we define the magnitudes for the errors.

Class A: Persistent Errors This class consists of errors which will affect almost all calculated samples (from when the error first occurs and then subsequent samples). If the errors are caused by faults in the memory storing the control constants, they can substantially change the behavior/structure of the system (disturbances with this effect are denoted structural in control theory). This type of persistent errors is not considered further in this paper (tools for more detailed analysis of structural disturbances are provided by robust control theory, see e.g., [11]). However, if the errors are uncorrelated with the control and state signals (e.g., have similar magnitude at longer time intervals, for instance caused by faults in memory cells storing the most significant bits of variables which change their value infrequently), they can be modeled by the step-function[†], $\theta(k-s)$.

Class B: Sporadic Errors This class includes errors that occur so infrequently that the system has time to return to a correct state before the next error occurs, i.e. the impacts of errors are not superimposed. Thus, these errors can be modeled as temporary impulse disturbances, described with the pulse-function[††], $\delta(k-s)$. Such errors are for instance caused by transient faults, occurring with low intensity, and intermittent-permanent faults in the functional units of the microprocessor that are activated occasionally and propagated with low probabilities.

Class C: Frequent Errors This class consists of errors not covered by the two previous classes, i.e., errors that do not affect every sample, but frequently enough for their impacts on the system to be superimposed. The impacts of these errors can be modeled as stochastic processes, i.e., a function that assumes random values and whose properties are described by its mean and variance, which either will assume 0 (when no error affects the system) or the magnitude of the error (when an error does affect the system), m, see Sect. 3.2. Such errors could be caused by transient faults occurring with high intensity, memory faults in the least significant bits of variables (which change value relatively often) and functional units intermittent-permanent faults.

3.2 Data Formats and Error Magnitudes

In the previous subsection, we discussed the occurrence rate of errors caused by different types of faults. In this subsection we focus on the magnitudes of the errors.

The error magnitude will be dependent on the formats used for representation of data in the calculations performed by controller. Two commonly used formats to represent numerical data are floating-point and fixed-point values. Floating-point values give better accuracy than fixed-point values for a given number of bits, but require either floating-point units (i.e., more expensive microprocessors) or additional software routines (adding to the total execution time of the calculations). We now define the error magnitudes that can occur for the different formats.

In the IEEE floating-point standard [4], numbers are represented with a sign bit, s, a fraction part with value f (23 bits in the single precision format) and an exponent part with value e (8 bits in the single precision format). A decimal number a is represented as:

$$a = (-1)^s(1.f)2^{(e-127)} \tag{3}$$

The range of representable values[†††] is then for the single precision format $R = [-(2-2^{-23})\cdot 2^{128}, (2-2^{-23})\cdot 2^{128}]$, with a resolution (minimum difference between two non-identical numerical values) of $Q = 2^{-23}$. With

[†] $\theta(k-s) = 0$ for $k < s$ and 1 for $k \geq s$, where s denotes the point in discrete time where the error occurs.

[††] $\delta(k-s) = 1$ for $k = s$, else 0, where s denotes the point in discrete time where the error occurs.

[†††] Often some values are used to represent special entities, e.g., ∞, reducing the usable range.

Table 1 Example of error magnitudes for single-bit errors in different data formats.

Erroneous bit	*Error magnitude, floating-point, single format (e_* is the value of the remaining exponent bits)*	*Error magnitude, fixed-point integer, N=32, M=23*
0	$2^{-23} \cdot 2^{(e-127)}$	2^{-23}
16	$2^{-7} \cdot 2^{(e-127)}$	2^{-7}
31	$(1.f) \cdot (2^{(e_*+1)} - 2^{(e_*-127)})$	2^{8}

this format, the magnitude of a bit error is dependent on the values of the other bits, but if it occurs in the most significant bits of the exponent, it will result in very high error magnitudes, see Table 1. Even if the physical limitations of actuators, sensors, etc., limit the immediate impacts of such errors, the state update (see Eq. (1)) can be seriously perturbed. This problem is addressed in [9] by adding executable assertions that check that data do not exceed their specified limits.

In the fixed-point format [3], numbers are represented in two-complement form, with a total of N bits, of which $M < N$ bits are used as fractional bits. Hence, a decimal number a is represented by the bit sequence $a_{N-1} \ldots a_0$, such that

$$a = 2^{-M} \left(-a_{N-1} 2^{N-1} + \sum_{i=0}^{N-2} a_i 2^i \right) \tag{4}$$

The $(N-1)th$ bit carries the sign information. The representable value range is $R = [-2^{N-M-1}, 2^{N-M-1} - 2^{-M}]$, with the resolution $Q = 2^{-M}$.

With this representation, the error magnitudes will be in the same range as the control signals (assuming that the data has been properly scaled). Thus, a single bit error in bit $n \in [0, N-2]$ will have the magnitude:

$$m = (-1)^{a_n} 2^{n-M} \tag{5}$$

where a_n is the correct bit value. A bit error in the sign bit $n = N-1$ will have the magnitude:

$$m = -(-1)^{a_{N-1}} 2^{N-M-1} \tag{6}$$

Some examples of magnitudes for single-bit errors are given in Table 1. Note that burst errors, affecting multiple bits, will correspondingly have magnitudes being the sum of the individual bit error magnitudes. For this study, we limit the scope to single-bit errors.

Fixed-point values will be used in the continuation of this paper, but the proposed analysis methods are valid also for floating-point values.

3.3 Error Impacts on the Generalized Controller

Looking at the errors described in previous subsections, we now define expressions for modeling their impacts on the generalized controller of Eq. (1). The expressions are summarized in Table 2, based on the discussions in Sect. 3.1 and Sect. 3.2. The first column defines the error class, the second column is the type of the disturbance that the error will cause and the third column shows the mathematical expressions describing the disturbances, η, for modeling the error in Eq. (2). Here, $m_A(k)$ is an error magnitude uncorrelated with the magnitude of the control and state signals, $m_B(s)$ the error magnitude at time s, $m_C(k)$ a stochastic process describing the error magnitude when a frequent error occurs, θ the step-function, δ the impulse function and s the point in discrete time when the first error occurs (i.e., the error occurs in a specific sample).

Table 2 Impacts of errors occurring at time s.

Error class	*Nature of resulting disturbance*	*η: Mathematical expression for disturbance*
A:Persistent	Step	$m_A(k)\theta(k-s)$
B:Sporadic	Impulse	$m_B(s)\delta(k-s)$
C:Frequent	Stochastic Process	$m_C(k)\theta(k-s)$

4. Analytical Methods for Understanding the Impact of Data Errors on Control Systems

In the previous section we defined disturbance-models for the impacts of different data errors on the system. In this section we first give an example of a control system. Then we present how different analysis methods can be used to understand the impact of errors on control systems, using the disturbance-models (Sect. 3.3) and the example.

4.1 Example — Brake-Slip Controller

A brake-slip controller is used to control the wheel-slip λ on a car during braking. This is used to avoid situations where the car loses its grip of the road and starts skidding. The sampling time is set to $t_h = 0.01$ s. The brake-slip controller normally operates in the region $0 < \lambda < 0.2$, where 0 corresponds to no slip, i.e., no braking, and 0.2 very hard braking where the wheels are almost locked. It is essential that the brake force on the left and right side of the car is balanced, otherwise there will be a resulting torque on the vehicle, and the car will turn during braking, which may lead to a hazardous situation. To avoid this, we require that the control error may not exceed $|\lambda_{desired} - \lambda| < 0.01 = C1$, if the reference value has not changed during the last 0.4 s (i.e. no change in the braking force is requested). We do not set any requirement for the duration of the control error ($C2$) in this example.

The process can be described as a linear discrete-time system:

$$\lambda(k) = \frac{B(q)}{A(q)} u(k) \tag{7}$$

where the polynomials B and A are of degree 1 and 2, respectively. We used the generalized controller in

Eq. (1) and instantiated it according to traditional control design for achieving desired control performance (for design details, see Appendix B). This resulted in the following values of the control constants:

$$\Phi = \begin{pmatrix} 1 & 0 \\ 0 & 0.2534 \end{pmatrix}, \quad \Gamma^{u_c} = \begin{pmatrix} 0.1237 \\ 0.01378 \end{pmatrix},$$
$$\Gamma^{y} = \begin{pmatrix} -0.1237 \\ 0.7328 \end{pmatrix}, \qquad C = 63.55 \quad 94.64 \ , \tag{8}$$
$$D^{u_c} = 13.01, \qquad D^{y} = -161.4$$

The first value in the controller state space (i.e., the top values of Φ, Γ^{u_c} and Γ^{y}) is an integrator state and the second value (the bottom values) is an observer state.

To validate that the desired control performance was reached with this instantiation, a case where the driver suddenly applies full brakes, was investigated. At normal driving conditions (dry asphalt road), this manoeuvre corresponds to a slip change from $\lambda = 0$ (no braking) to $\lambda = 0.1$ (full braking). Fig. 4 shows how the output signal and the control signal, are changed during the manoeuvre. Please note that different scales are used between the plots in this and subsequent figures.

As seen in the left plot of Fig. 4, the output value assumes the reference value within 0.4 s (which was the requirement) and without any major overshooting (i.e., exceeding the reference value). Looking at the control signal, the right plot of Fig. 4, it first increases quickly to make the output signal assume the desired value (the reference value) as fast as possible. Then it decreases to avoid overshooting and stays constant on the level required to give the desired slip of $\lambda = 0.1$. The actuators were considered to be able to deliver the maximum required braking force. Thus, all requirements were fulfilled and the control instantiation accepted.

Fig. 4 can be used for comparison of how the system signals are changed due to normal changes of the reference values (braking) and how they are changed due to different types of errors caused by computer node faults (we will detail the impact of errors on control system in the following subsections).

To study the impact of bit errors, the data format needs to be set. Assume that the available word-length is $N = 32$ bits. Based on the numerical values in Eq. (8) and the signal magnitudes of Fig. 4, the fixed-point numerical implementation is chosen as $M = 23$, which gives the numerical range $R = [-256.0, 256.0 - Q]$ with a resolution of $Q = 2^{-23} \approx 1.19 \cdot 10^{-7}$.

In the following subsections, we describe how the impacts of data errors on control systems can be investigated using different analysis methods, adopted from control theory, and exemplify these using the control system example (i.e., the brake-slip controller) given in this subsection.

4.2 Sensitivity Analysis

In control theory, the sensitivity function is used to determine how the system is affected by disturbances occurring in a particular signal. It is calculated by determining the impulse response function, $h(k)$, for how changes of that specific signal affect the output value. This function is then transformed from the time domain to the frequency domain (to the closed-loop transfer-function, $H(z)$) to determine how much disturbances with different frequencies are amplified (or attenuated). This makes it possible to determine which occurrence rates of disturbances the system is most sensitive to. For more details about the sensitivity functions, see Appendix A.

As shown in Sect. 3.3, errors can be described as disturbances in the calculation of the control signal, η_u, and state space, η_z, (Henceforth, we will denote disturbances affecting the integrator state as η_1 and disturbances affecting the observer state as η_2.) see Eq. (2). Thus, we can determine the impact of different errors on the system, by calculating the sensitivity functions for each disturbance.

To exemplify this approach, the sensitivity functions for errors affecting the example given in Sect. 4.1 were calculated. The results are shown in Fig. 5, where the x-axis shows angular frequency and the y-axis how much sine shaped disturbances (errors) of each angular frequency are amplified. As control signals are sampled, errors will be rectangular shaped, consisting of various frequencies. However, the main frequency component will be the occurrence rate of the error, and thus, analyzing the impact of errors with this frequency will in most cases give a good assumption.

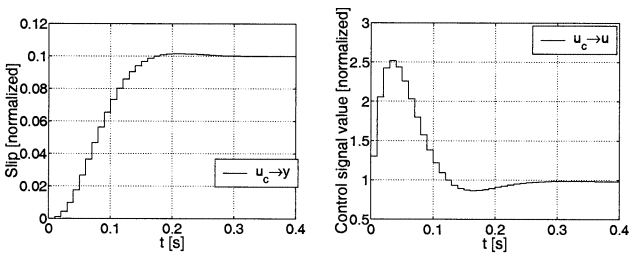

Fig. 4 Step-input command-signal response of closed-loop system. The plots have different scales.

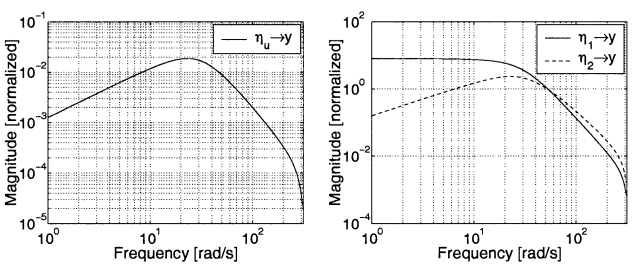

Fig. 5 Sensitivity functions from η_u (control signal), η_1 (integrator), and η_2 (observer) to output, y. The plots have different scales.

As can be seen, the curve for errors affecting the integrator state (η_1 in the right plot of Fig. 5) assumes the highest value at almost all frequencies, i.e., the system is most sensitive for these errors. The highest value for the sensitivity function of η_1, about 8, is assumed at 0 Hz, which means that the magnitudes of constant errors affecting multiple consecutive samples, i.e., persistent errors, of the calculation of the integrator state, will be amplified about 8 times and have the largest impact on the system output. As the sensitivity is greater than 0, the errors will result in a steady-state influence on the process output (i.e., the system will never return to the reference value).

For the control signal (η_u) and the observer state (η_2) in Fig. 5 persistent errors (frequency = 0 Hz) are rejected (asymptotically assume 0, meaning that the impact of the error will decrease to 0). Instead the maximum value for their sensitivity functions are reached for frequent errors at 23.2 rad/s = 3.7 Hz. Thus, for errors affecting the control signal and observer state, the largest slip impact (the error that is amplified most) will be for errors occurring with this frequency.

4.3 Impulse Response Analysis

To more accurately estimate the impacts of sporadic errors, one can study how the output value is affected by impulse disturbances occurring in the control algorithm (η_u and η_z in Eq. (2)), by calculating the impulse response functions, $h(k)$ (described in Sect. 4.2 and in Appendix A). These functions show how the output value is changed due to impulses (sporadic errors) with magnitude 1, affecting the calculation of the different signals of Eq. (2). As we consider linear systems, *the impact on the system will be proportional to the magnitude of the error.* Thus, the impact of an error with magnitude m will be[†] $mh(k)$.

The impulse response analysis also reveals how fast the output returns to the correct value, due to the inherent robustness of the system (i.e., without adding any error recovery), after an error has occurred. This time can be used to determine how often errors can occur without their impacts being superimposed, i.e., it can be used for classifying errors into sporadic errors or frequent errors. It should be noted that due to the linearity, the time constant is independent of the magnitude of the error. This means that if the response to a error with a certain magnitude have decreased to x% of its maximum value after a certain time, errors with other magnitudes will also have decreased to x% of their maximum value, after the same time.

To exemplify this analysis method, the output and control signal responses to impulse data errors for the example described in Sect. 4.1 were calculated. Fig. 6 shows how the slip is affected (the nominal value is set to 0) by an impulse (sporadic error) affecting the calculation of the control signal (the left plot) and state space (the right plot). Fig. 7 shows how the control signal is affected for the same impulse affecting the control signal (the left plot) and state space (the right plot).

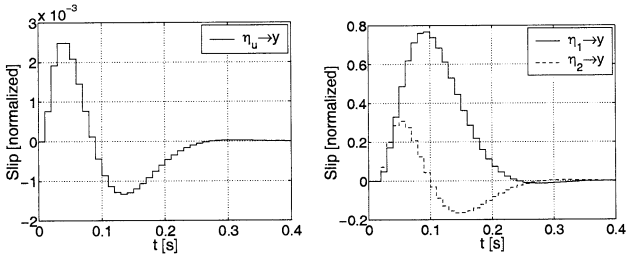

Fig. 6 Output signal responses to impulse data errors in η_u (control signal), η_1 (integrator) and η_2 (observer). The plots have different scales.

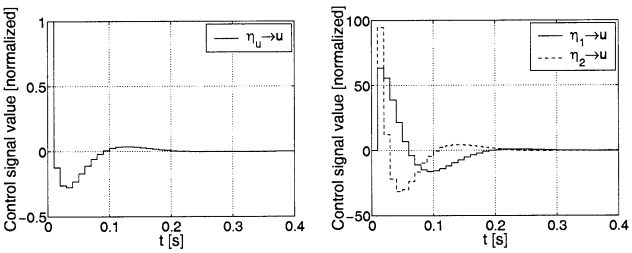

Fig. 7 Control signal responses to impulse data errors in η_u (control signal), η_1 (integrator) and η_2 (observer). The plots have different scales.

It can be seen that sporadic errors have larger impact when affecting the states compared to the control signal (i.e., the slip diverges more from its nominal value of 0 in the left plot compared to the right plot of Fig. 6. This is due to the large scaling of the state variables when calculating the control signal (matrix C in Eq. (8)) and that the errors will be stored in the states of the controller (the old values of the state z in Eq. (2) are used for calculating the new state and the control signal), and thus, affect several samples, whereas control signal errors will only directly affect one sample.

Furthermore, it can be seen that the integrator state is more sensitive to sporadic errors than the observer state (the magnitude is higher for the solid line $\eta_1(k)$in the right plot of Fig. 6 than for the dashed line $\eta_2(k)$). However, looking at the right plot of Fig. 7 it can be seen that errors in the observer state generate control signals with higher magnitudes (the dashed line have a higher maximum magnitude than the solid line).

It can also be seen from Fig. 6 that the system returns to the nominal slip, 0, about 0.4 s (in this case 40 samples, $t_h = 0.01$ s) after the error occurred. This means that if errors do not occur more frequently than this, the effect of them will not be superimposed.

[†]For an error that manifests as an impulse with magnitude m at time $s = 0$, i.e., $\eta = m\delta(k)$, the impact on the process output will be described by $y(k) = \sum_{l=0}^{\infty} h(l)m\delta(k-l) = mh(k)$.

4.4 Norm Analysis

We base our failure criteria on magnitude and duration of the control error, see Sect. 2. To get numerically comparable values, the following norms are useful:

$$\|h\|_1 \triangleq \sum_k |h(k)| \quad \text{and} \quad \|h\|_\infty \triangleq \sup_k |h(k)| \qquad (9)$$

where sup denotes the supremum function. The norm $\|h\|_\infty$ is the maximum absolute value of the error resulting from the disturbance. The norm $\|h\|_1$ is the sum of the absolute control errors over time. The essential difference is that $\|h\|_\infty$ only gives information on the largest size of the error. When combined with $\|h\|_1$, information on the duration of the error is also assumed. The choice of controller state-realization (i.e., which values for the control constants of Eq. (1) that are chosen) will affect the size of both norms ($\|h\|$). In particular, $\|h\|$ can be made arbitrarily small with a scaling of the state variables. In the example described in Sect. 4.1, all state realizations are scaled such that a step-input command-signal results in state-variables with unit stationary value, see the plot in Fig. 4 (b).

If the failure criteria are defined using the previously described norms, the maximum single bit error, n, which is tolerated can be found through:

$$\begin{aligned} n &< n_{lim\infty} := M + \log_2(C1/\|h\|_\infty) \\ n &< n_{lim1} := M + \log_2(C2/\|h\|_1) \end{aligned} \qquad (10)$$

where $C1$ and $C2$ are the failure limits specified by the designer (e.g., for the brake-slip controller, in Sect. 4.1, $C1$ was specified but not $C2$) for the failure criteria described in Sect. 2 and M is the number of fractional bits used in the fixed-point representation. It follows that multiple errors in bits satisfying $n < n_{lim} - 1$ do not lead to failure.

The impulse-response norms for the example described in Sect. 4.1 are shown in Table 3. The largest bit for which a single-bit or multiple-bit error does not result in failure (i.e. the control error exceeds $C1 = 0.01$, see Sect. 4.1), according to Eq. (10), is shown in Table 4.

Table 3 Impulse-response norms calculated from Eq. (9).

Norms	$h_u(k)$	$h_1(k)$	$h_2(k)$
$\|h\|_1$	0.0245	8.28	3.08
$\|h\|_\infty$	0.00248	0.766	0.307

Table 4 Largest bit-number for which a single or multiple bit impulse-error is tolerated (calculated from Eq. (10) for the failure criterion $C1$).

	η_u	η_1	η_2
Single bit error	25	16	18
Multiple bit errors	24	15	17

4.5 Step Response Analysis

Persistent errors change the dynamics of the closed-loop system, and thus, control performance is generally substantially affected. Therefore, most of these errors need to be handled, and thus, the analysis method in this paper is focused on how fast errors need to be detected in order for the system to recover before it fails.

In many cases, errors occurring several samples in a row with high constant magnitude (i.e., $m_A(k) = m_A$ in Table 2), are the most harmful errors, i.e., the errors that in shortest time will result in failures (the errors with the highest magnitude can be identified with sensitivity analysis, see Sect. 4.2). Such errors will, at least for the first samples, have similar impacts as step disturbances (the step function, θ, is described in Sect. 3.1). Thus, we can study the system response to step disturbances to find when (after how many samples) the failure limits are reached for different error magnitudes[†].

To exemplify this approach we calculate the step response functions for the brake-slip controller described in Sect. 4.1. The result is shown in Fig. 8. For persistent errors with constant magnitude affecting the calculation of the control signal (the left plot) or the observer state (the dashed line, $\eta_2(k)$, in the right plot) of Eq. (2), the slip returns to the nominal value 0 within 0.4 s. This is due to the fact that the controller uses an integrator state designed to mask the effect of constant disturbances. However, when such errors affect the integrator state directly (the solid line in the right plot), the slip will never return to its nominal value.

The number of samples between the fault occurrence and until the $C1$ or $C2$ limits are exceeded, can be used as a measure of the required system recovery

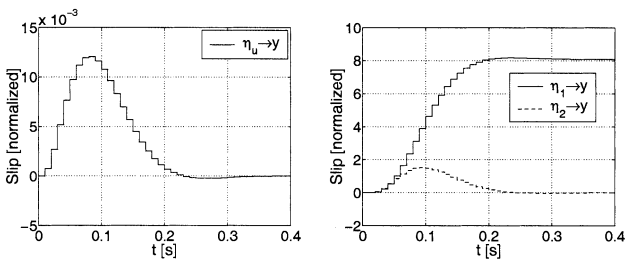

Fig. 8 Output responses to step fault disturbances in η_u (control signal), η_1 (integrator) and η_2 (observer). The plots have different scales.

[†]For a fault that manifests as a step with magnitude m on state z occurring at time $t = 0$, i.e., $\eta_z = m\theta(k)$, the impact on the process output will be described by $y(k) = \sum_{l=0}^{\infty} h_z(l) m\theta(k-l) = m\sum_{l=0}^{k} h_z(l)$. Using this equation we can find the first sample (if any), k, at which the requirements are no longer fulfilled, that is, the time within which the system needs to be recovered to avoid system failures.

time. As an example, consider an error occurring in the integrator state (the solid line in the right plot of Fig. 8). Closer inspection of the plot shows that after 0.02 s (two samples) the control error reaches 0.0483 for an error with magnitude† 1, which is well above $C1 = 0.01$ ($C1$ was specified in Sect. 4.1). This means that recovery must have been initiated before 0.02 s after occurrence of the error.

4.6 White Noise Response Analysis

For determining the impacts of persistent errors which magnitude is not constant but still uncorrelated with the control and state signals, we describe the errors as uncorrelated stochastic processes, with variance σ^2. Then, a requirement on the maximum tolerated variance on the slip (a new failure criterion **C3**), can be specified. To determine which errors violate this requirement, we can calculate the resulting variance†† of errors with different variances similar to Eq. (10) as:

$$n < n_{lim2} := M + \log_2(C3/\|h\|_2^2) \tag{11}$$

4.7 Simulations

Even if the previously described analysis methods will provide important information on the effect different data errors will have on the system, simulations may be necessary to get more detailed information on specific errors (mainly frequent errors). As we are looking at linear systems, the impacts of several errors occurring closely in time (i.e., another error occurs before the system has returned to the correct state after the previous error) can be simulated by superimposing the impact of each error, for instance by using a tool such as Simulink (a toolbox for MATLAB, [7]). In the simulation environment it is possible to define when different disturbances (errors) should occur and which magnitudes they should have. Then, the resulting output can be observed to see whether any failure criteria is exceeded, see Fig. 9. Simulations can be time-consuming, but by utilizing the information gained from the analysis methods, only a small number of errors is generally necessary to simulate.

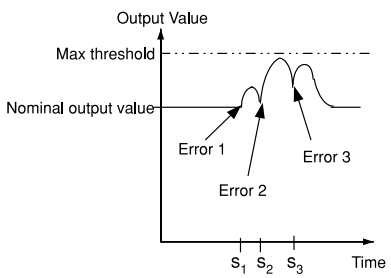

Fig. 9 Simulations of the impact of three errors with different magnitudes occurring close in time.

†For an error occurring in the most significant bit (i.e., with magnitude $2^{31} \cdot 2^{-23} = 2^8$), the control error will be (due to the linearity): $0.0483 \cdot 2^8 \approx 12.4$, i.e., much higher.

††The variance can be calculated through: $E[y^2] = E\left[\left(\sum_l h(l)\eta(k-l)\right)\left(\sum_j h(j)\eta(k-j)\right)\right] = \sum_l h(l)^2\sigma^2 = \|h\|_2^2\sigma^2$, where η determines which calculation of the control algorithm Eq. (2) that is affected and $\|h\|_2^2 \triangleq \sum_k h(k)^2$.

5. Conclusions and Future Work

We have described how analysis methods adopted from control theory can be used for understanding the impact of data errors on control systems. The results obtained from the analysis of an automotive brake-slip controller showed that many transient faults are tolerated by the control system and that the part most sensitive to data errors is the integrator state. This indicates that many results found from fault injection experiments [1], [9] can be estimated from analysis already at early design phases. Such information is valuable for safety-critical system designers when deciding on which fault tolerance techniques are efficient to use.

More specifically, using the described analysis it is possible to estimate: i) the control performance degradation (i.e., the control error) different errors cause, ii) the maximum allowed recovery time before a system failure occurs, iii) how often errors can occur without their impacts being superimposed, iv) the error magnitude dependency on the data format used. We have also used the analysis to design a controller that takes control signal limitations into account, to reduce the impact of data errors, [2].

In future work we will investigate how the analysis methods presented in this paper can be used when designing executable assertions.

Acknowledgment

The authors want to thank Prof. Björn Wittenmark for valuable comments on this work.

References

[1] J. Cunha, R. Maia, M. Rela, and J. Silva, "A study of failure models in feedback control systems," DSN-01, pp.314–323, 2001.
[2] M. Gäfvert, B. Wittenmark, and Ö. Askerdal, "On the effect of transient data errors in controller implementations," American Control Conference, 2003.
[3] H. Hanselmann, "Implementation of digital controllers — A survey," Automatica, vol.23, no.1, pp.7–32, 1987.
[4] "IEEE standard for binary floating-point arithmetic," ANSI/IEEE Std 754-1985, 1985.
[5] H. Kim and K. Shin, "On the maximum feedback delay in a linear/nonlinear control system with input disturbances caused by controller-computer failures," IEEE Trans. Control Syst. Technol., vol.2, no.2, pp.110–122, 1994.

[6] H. Kim, A. White, and K. Shin, "Effects of electromagnetical interference on controller-computer upsets and system stability," IEEE Trans. Control Syst. Technol., vol.8, no.2, pp.351–357, 2000.
[7] The MathWorks, Inc., Using Simulink Version 3, Dynamic System Simulation for MATLAB, Natick, MA, Jan. 1999.
[8] D. Pradhan, Fault-Tolerant Computer Systems Design, Prentice Hall PTR, 1996.
[9] J. Vinter, J. Aidemark, P. Folkesson, and J. Karlsson, "Reducing critical failures for control algorithms using executable assertions and best effort recovery," DSN-01, pp.347–356, 2001.
[10] B. Wittenmark, J. Nilsson, and M. Törngren, "Timing problems in real-time control systems: Problem formulation," American Control Conference, pp.2000–2004, 1995.
[11] K. Zhou and J.C. Doyle, Essentials of robust control, Prentice Hall, Upper Saddle River, New Jersey, 1998.
[12] K. Åström and B. Wittenmark, Computer-Controlled Systems, third ed., Prentice Hall, 1997.

Appendix A: The Closed-Loop System

Let

$$\begin{aligned} x(k+1) &= \Phi_p x(k) + \Gamma_p u(k) \\ y(k) &= C_p x(k) \end{aligned} \tag{A·1}$$

with $x \in \mathbb{R}^{n_x}$, $\Phi_p \in \mathbb{R}^{n_x \times n_x}$, $\Gamma_p \in \mathbb{R}^{n_x \times 1}$, and $C_p \in \mathbb{R}^{1 \times n_x}$ be an arbitrary state realization of the process subject to control. Combined with the controller, Eq. (2), this yields the closed-loop system

$$\xi(k+1) = \Phi_{cl}\xi(k) + \Gamma_{cl}^{u_c} u_c(k) + \sum_z \Gamma_{cl}^{\eta_z}\eta_z(k) + \Gamma_{cl}^{\eta_u}\eta_u(k) \tag{A·2}$$

$$y(k) = C_{cl}^{y}\xi(k) \tag{A·3}$$

$$u(k) = C_{cl}^{u}\xi(k) + D^{u_c}u_c(k) + \eta_u(k) \tag{A·4}$$

with $\xi = (x; z)$ and

$$\begin{aligned} \Phi_{cl} &= \begin{pmatrix} \Phi_p + \Gamma_p D^y C_p & \Gamma_p C \\ \Gamma^y C_p & \Phi \end{pmatrix} \quad \Gamma_{cl}^{u_c} = \begin{pmatrix} \Gamma_p D^{u_c} \\ \Gamma^{u_c} \end{pmatrix} \\ \Gamma_{cl}^{\eta_z} &= \begin{pmatrix} 0 \\ e_z \end{pmatrix} \qquad \Gamma_{cl}^{\eta_u} = \begin{pmatrix} \Gamma_p \\ 0 \end{pmatrix} \\ C_{cl}^{y} &= \begin{pmatrix} C_p & 0 \end{pmatrix} \qquad C_{cl}^{u} = \begin{pmatrix} D^y C_p & C \end{pmatrix} \end{aligned} \tag{A·5}$$

where e_z is the z:th column of the unit matrix $I_{n_z \times n_z}$. The impulse-responses from the computation-errors $\eta_z(k)$ and $\eta_u(k)$ to the process output $y(k)$ are

$$\begin{aligned} h_z(k) &= \begin{cases} 0, & k \le 0 \\ C_{cl}^{y}\Phi_{cl}^{k-1}\Gamma_{cl}^{\eta_z}, & k > 0 \end{cases} \\ h_u(k) &= \begin{cases} 0, & k \le 0 \\ C_{cl}^{y}\Phi_{cl}^{k-1}\Gamma_{cl}^{\eta_u}, & k > 0 \end{cases} \end{aligned} \tag{A·6}$$

The process output-responses to the disturbances η_z and η_u are now described by

$$\begin{aligned} y(k) &= \sum_{i=0}^{\infty} h_z(i)\eta_z(k-i) \\ y(k) &= \sum_{i=0}^{\infty} h_u(i)\eta_u(k-i) \end{aligned} \tag{A·7}$$

respectively. The sensitivity functions

$$\begin{aligned} H_z(z) &= C_{cl}^{y}(zI - \Phi_{cl})^{-1}\Gamma_{cl}^{\eta_z} \\ H_u(z) &= C_{cl}^{y}(zI - \Phi_{cl})^{-1}\Gamma_{cl}^{\eta_u} \end{aligned} \tag{A·8}$$

are the $\mathcal{Z}$-transforms of the impulse responses, and describe the frequency responses from η_z and η_u to the output y. Evaluation of the sensitivity functions at $z = e^{i\omega t_h}$, where t_h is the sampling time, gives the stationary response of the closed-loop system for pure sinusoidal inputs $\eta_z(k) = \sin(k\omega t_h)$ and $\eta_u(k) = \sin(k\omega t_h)$ as $y(k) = |H_z(e^{i\omega t_h})|\sin(k\omega t_h + \arg(H_z(e^{i\omega t_h})))$ and $y(k) = |H_u(e^{i\omega t_h})|\sin(k\omega t_h + \arg(H_u(e^{i\omega t_h})))$ respectively. Please refer to [12] for more details.

Appendix B: The Brake-Slip Controller

The wheel-slip is the normalized relative velocity between the rotating wheel and ground: $\lambda = (r\omega - v)/v$, where r is the wheel radius, ω the wheel angular velocity, and v the speed of the car. The braking force of a wheel is proportional to the wheel slip for $0 \le \lambda \lesssim 0.2$, as $F_b = F_z C_\lambda \lambda$, where F_z is the normal load on the tire and C_λ is a tire-stiffness parameter. A simple quarter-car model of the slip dynamics is

$$v/\alpha\frac{d\lambda(t)}{dt} + \lambda(t) = \beta/\alpha\tau_b(t) \tag{A·9}$$

with $\alpha = mgr^2C_\lambda/4J$, $\beta = r/J$, where m is the car mass, and J is the wheel inertia. The braking torque $\tau_b(t)$ is described by the actuator dynamics

$$T_a\frac{d\tau_b(t)}{dt} + \tau_b(t) = K_a u(t) \tag{A·10}$$

Combining Eq. (A·9) and Eq. (A·10), and discretizing with zero-order-hold (ZOH) sampling with period t_h results in the second order open-loop system Eq. (7).

Numerical values used in the example are $v = 30$ m/s, $m = 2000$ kg, $J = 16$ kgm^2, $r = 0.4$ m, $C_\lambda = 5$, $F_z = 5000$ N, $g = 9.81$ kgm/s^2, $T_a = 0.05$ s, and $K_a = 1000$.

A RST-controller [12] is designed to obtain a closed-loop system with poles in a Butterworth pattern with $\omega_{cl} = 30$ rad/s and opening angle 45°. The controller includes integral action. An observer pole is introduced at $2\omega_{cl}$. A modal form state-realization of the controller is found in Eq. (8).

Örjan Askerdal received his MS in electrical engineering degree in 1997 and his Ph.D. degree in computer engineering in 2003 at Chalmers University of Technology, Sweden. His research interests include analysis, design, implementation and testing of safety-critical computer-controlled systems.

Magnus Gäfvert received his MSc in Engineering Physics in 1996 and his Ph.D. in Automatic Control in 2003 at Lund Institute of Technology, Sweden. His recent research is focused on modeling and control systems in automotive applications.

Martin Hiller received his MS in Computer Science and Engineering in 1996 and his Ph.D. in Computer Engineering in 2002, both from Chalmers University of Technology, Sweden. His research has been focused on analysis and design of software for dependable embedded systems. In 2001, he received the William C. Carter Award for his work on analysis of propagation of data errors in software. Hiller is currently with the Volvo Technology Corporation, Sweden, and is also a post-doc at TU-Darmstadt, Germany.

Neeraj Suri received his MS and Ph.D. degrees from the Univ. of Massachusetts at Amherst. He currently holds the TU-Darmstadt Chair Professorship in "Dependable Embedded Systems and Software" at TU Darmstadt, Germany. Prior to TU Darmstadt, he held the Saab Professorship and was on the faculty at Boston University. His research interests focus on design, analysis and assessment of dependable systems and services. Suri serves as an editor for ACM Computing Surveys covering Embedded Systems and was an editor for IEEE Trans. on Parallel and Distributed Systems. He is a member of IFIP WG 10.4 and on the board for Microsoft's Academic Advisory Board for Trustworthy Computing. More research and professional details are available at www.deeds.informatik.tu-darmstadt.de