# Assessing the Effect of Failure Severity, Coincident Failures and Usage-Profiles on the Reliability of Embedded Control Systems

Frederick T. Sheldon
Applied Software Engineering Research
Oak Ridge National Laboratory
Oak Ridge, TN 37831 USA
001-865-576-1339
SheldonFT@ornl.gov

Kshamta Jerath
School of EECS
Washington State University
Pullman, WA 99164 USA
001-425-883-0392
kshamtaj@verizon.net

## ABSTRACT

The increasingly ubiquitous use of embedded systems to manage and control our technologically (ever-increasing) complex lives makes us more vulnerable than ever before. Knowing how reliable such systems are is absolutely necessary especially for safety, mission and infrastructure critical applications. This paper presents a structured compositional modeling method for assessing reliability based on characteristic data and stochastic models. We illustrate this using a classic embedded control system (sensor-inputs | processing | actuator-outputs), Anti-lock Braking System (ABS) and empirical data. Special emphasis is laid on modeling extra-functional characteristics of severity of failures, coincident failures and usage-profiles with the goal of developing a modeling strategy that is realistic, generic and extensible. The validation approach compares the results from the two separate models. The results are comparable and indicate the effect of coincident failures, failure severity and usage-profiles is predictable. **Keywords** design, measurement, performance, reliability

## 1 INTRODUCTION

*Reliability*, the probability that a system will deliver its intended functionality for a specified period and under specific conditions, *is one inherently important measure of quality* [1]. Structured models allow the system's reliability to be determined from the reliabilities of its constituent (possibly numerous) components. The key is to find the right level of complexity that is reasonably tractable (see Fig. 1). Complex embedded systems are composed of numerous components. The probability that the system survives (effective down to an acceptable level of degraded performance) depends directly on each of the components. Reliability analysis can provide an understanding about the likelihood of failures and an increased insight about inherent system *weaknesses* [2].

In [3], the authors present Stochastic Petri Net (SPN) models of a vehicle dynamic driving regulation (DDR) system with sub-system representations of the (1) Anti-lock Braking System (ABS), (2) Electronic Steering Assistance (ESA), (3) traction control (TC) and, (4) a composed model of all three. Lets consider but one component of the total system with the idea that the developed model can be extended to the larger context of interacting/interdependent components. Moreover, this approach considers the extra-functional characteristics of coincident failures (CFs), severity of failures (SFs) and usage-profiles (UPs). One significant failure characteristic is severity. Severity of a failure is the impact it has on the overall operation of the system (i.e., the hazard posed, in functional terms, to correct system operation) [1]. Predicting reliability based on such characteristics provides an objective and concrete way to assess risk (e.g., tradeoffs to integrity levels which establish the consequence of failure). Severity is an important candidate to weigh the data used in reliability calculations and must be incorporated into the model to determine the probability that the system survives, including efficient or acceptable degraded operation[1]. Failure severity has been studied in the context of gracefully degrading systems –using Markov models to model a multiprocessor system in [5] and a set of radars in an air traffic control system in [6].

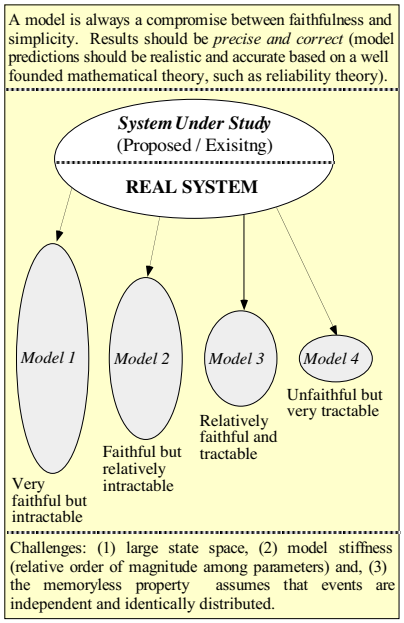


**Fig. 1: Precise realistic models.**

Further, if a system does not contain redundancy –that is, if every component must function properly for the system to work –and if component failures are statistically independent, then the system reliability is simply the product of the component reliabilities (i.e., the failure rate of the system is simply the sum of the failure rates of the individual components [7]). The assumption that failures occur independently (in a statistical sense) in hardware components is a widely used and often successful model for predicting the reliability of hardware

[1] Severity relates to failures in a dangerous mode that would impair safety integrity. Two relevant parameters are the overall failure rate and the probability of failure to operate on demand. The former parameter is especially important when continuous control is necessary to maintain safety. The latter parameter (availability) is used in the context of protective systems. Higher safety integrity demands a lower probability that safety-related systems will fail to carry out the required functions [4].

© 2004 Association for Computing Machinery. ACM acknowledges that this contribution was authored or co-authored by a contractor or affiliate of the U.S. Government. As such, the Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.

SAC'04 March 14-17, 2004, Nicosia, Cyprus
Copyright 2004 ACM 1-58113-812-1/03/04…$5.00

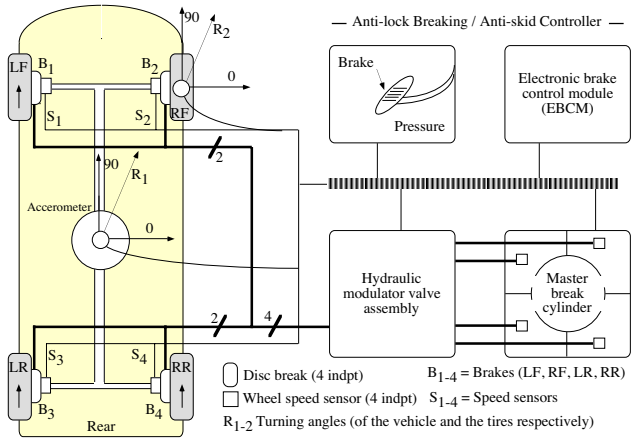


**Fig. 2: Top-level schematic of sensors/ processing / actuators.**

devices. However, components generally interact with each other during operation, and a faulty component can affect the probability of failure of other components [8]. Such failures are coincident in the sense that failure of one component increases the probability of failure in another[2].

Researchers have considered modeling correlation between failures. Two schools of thought have emerged, distinguished by the definition of the basic events of interest, and are called Correlated failures and Differentiated causes [9]. Correlated failures were first considered by Eckhardt and Lee [10], and later extended by Littlewood and Miller [11]. Differentiated causes was proposed by Arlat, Kanoun and Laprie [12] and adopted by others.

System reliability also depends on its usage profile –users interact with the system in an intermittent fashion, resulting in operational workload profiles that alternate between periods of *active* and *passive* use. Reliability describes the service that is actually delivered by the system as opposed to a system's capacity to deliver such service [13]. When considering usage profiles, faults need not necessarily cause failures since they can be repaired; failures occurring during "active" use of the system only, should contribute to reliability calculations. Investigations in this field have mainly been experimental, using empirical data from measurements of real systems to correlate workload with various measures of dependability [14, 15]. On the analytic side, probabilistic models are used to obtain workload-related dependability measures [5, 16-18].

Our structured compositional modeling (SCM) method uses readily available tools (i.e., stochastic modeling abstraction (SMA) *patterns* and classical Markov theory) to incorporate the characteristics of *failure severity*, *coincident failures* and *usage-profiles*. A model is developed for the ABS of a passenger vehicle that exemplifies SCM method [19]. These characteristics provide an assessment using a more realistic and extensible modeling approach (i.e., dependent only on the availability of data needed to assign component failure rates). In particular, the strategy adopted is innovative in terms of how the nonfunctional properties are integrated into the Stochastic Petri Net (SPN) and Stochastic Activity Network (SAN) formalisms.

SPNs and SANs provide concise and intuitive representations that are used to automatically generate the underlying Markov process. The strength of this approach is only limited theoretically by the three universal challenges in analytic Markov modeling – large state space, stiffness and the memoryless property which assumes events are independent and identically distributed. Further, employing the identified modeling abstraction patterns, as opposed to conducting experiments and observations of the real system (measurements), are two separate but complementary practices in any performance evaluation process methodology. For obvious reasons, it was beyond the scope and means of this research to validate the predictive results against real data (e.g., instrumenting/observing system failures and demonstrating causal relationship (beyond deductive interpretation). Nevertheless, as a means to strengthen confidence, we used two different stochastic formalisms to carry out the reliability analysis (see Figs. 12 and 13). The results of analyzing the two similar (i.e., same system different formalisms and consequently different abstraction patterns) models using the Stochastic Petri Net Package [20] and the UltraSAN [21] tools are discussed and compared.

# 2 EMBEDDED SYSTEM DESCRIPTION

The ABS is an integrated element of the total braking system (Fig. 2). In ABS-deficient systems, applying excessive pressure on the brake pedal, or panic slamming the brake pedal, can cause wheels to lock up, skidding, loss of stability and control. The ABS prevents wheel lockup by modulating brake pressure to permit steering control while braking. This system is easily composed from the general form of reactive embedded control: (1) sensor inputs and control inputs (a combination of continuous and discrete parameters respectively), (2) control law processing and, (3) outputs to actuators (e.g., protection, dampening devices) that maintain stability (e.g., smooth/dampen transient states)[3].

## 2.1 Components (Cmpts) and Functioning

The ABS consists of the following major components (cmpts). (1) Wheel Speed Sensors –measures wheel-speed and transmit information to an electronic control unit, (2) Electronic Control Unit (ECU Controller) –receives information from the sensors, determines when a wheel is about to lock up and controls the hydraulic control unit, (3) Hydraulic Control Unit (HCU, Hydraulic Pump) –controls pressure in the brake lines, and (4) Valves –present in the brake line of each brake and are controlled by the HCU to regulate pressure in the brake lines.

When a driver applies the brakes, wheel sensors monitor the rotational speed of each wheel. The ECU "reads" signals from the sensors and compares the speed of each wheel. If one wheel is slowing at a faster rate than the others, the ECU sees that the wheel is beginning to lock up and orders the HCU to reduce the line pressure to that wheel's brakes. The HCU reduces pressure (*pulsing* the brake line) in that particular brake line by opening its valves. Once the wheel resumes normal operation, the HCU restores pressure to its brake. Keeping all tires just below the lock-up threshold maintains the highest steering capacity.

ABS complexity prevents a direct analysis. A series of abstraction steps are necessary to deduce system predictions from measures (i.e., observations) taken from the real system. Initially the system model is created at a detailed level and the collected data are used to parameterize the model. In the second *abstraction* step, the computational model is created which allows an easier and more efficient system analysis [22]. The key element therefore is to identify the essential system cmpts, the different ways they interact and introduce various simplifying assumptions. Table 1 presents all cmpt parts and their failure rates distinguished

[2] Common cause (mode) failure analysis determines potential failures in multiple [sub-] systems that would undermine the benefits of redundancy, because of the appearance of the same failures in multiple parts at the same time (and/or the appearance of a cascading effect that impacts the ability of the effected [sub-] system to function).

[3] Real-time in this context places hard deadlines on the delivery of outputs which if not met, will cause loss of stability (so-called timing correctness).

**Table 1: Failure rates associated w/ critical failure states**

| Component | # | Base Failure Rate | Probability | | |
|---|---|---|---|---|---|
| | | | Degraded Operation | Loss of Stability | Loss of Vehicle |
| Wheel Speed Sensor | 4 | 2.00E-11 | 0.38 | 0.62 | - |
| Pressure Sensor | 4 | 1.50E-11 | 0.64 | 0.36 | - |
| Main Brake Cylinder | 1 | 1.00E-11 | - | - | 1.0 |
| Pressure Limiting Valve | 2 | 6.00E-13 | - | 0.22 | 0.78 |
| Inlet Valve | 4 | 6.00E-13 | - | 0.18 | 0.82 |
| Drain Valve | 4 | 6.00E-13 | - | 0.19 | 0.81 |
| Toggle Switching Valve | 2 | 6.00E-13 | 1.0 | - | - |
| Hydraulic Pump | 2 | 6.80E-11 | - | - | 1.0 |
| Pressure Tank | 2 | 2.00E-12 | - | - | 1.0 |
| Controller | 1 | 6.00E-12 | 0.2 | 0.4 | 0.4 |
| Tubing | 1 | 3.00E-12 | 0.33 | - | 0.67 |
| Piping | 1 | 4.00E-12 | 0.33 | - | 0.67 |

by the 3 different critical failure states[4].

## 2.2 Assumptions

Various assumptions about the system include the following:

(1) Three *modes of operation*: For the purpose of this discussion, the different modes of operation of the system (in presence/absence of failures of different severity) are assumed to be: (i) normal operation, (ii) degraded operation, and (iii) lost stability mode; in increasing order of severity. Critical failures seriously impact the operation of the system, and are assumed to cause loss of vehicle. Further, if sufficient cmpts of the system have failed to impact the system operation (either degraded operation or lost stability mode), the sum of those failures is assumed to be critical, causing loss of vehicle.

(2) Three *modes of operation*: The different modes of operation of the system (in presence/absence of failures of different severity) are assumed to be: (i) normal operation, (ii) degraded operation, and (iii) lost stability mode; in increasing order of severity. Critical failures seriously impact the operation of the system, and are assumed to cause loss of vehicle. Further, if sufficient cmpts of the system have failed to impact the system operation (either degraded operation or lost stability mode), the sum of those failures is assumed to be critical, causing loss of vehicle.

(3) *Lifetime of passenger vehicle*: Essentially the average hours of operation for a passenger vehicle range from 300-600 hrs /year and the average lifetime is 10-15 years. Thus, the average life span of a passenger vehicle ranges from 3000 - 9000 hrs. This estimate is important while considering the duration for which to carry out the reliability analysis.

(4) *Interdependencies among cmpts*: To model coincident failures, several dependencies among system cmpts are assumed. Only those inter-relationships between cmpts depicted as solid arrows in Fig. 3 are explicitly modeled. All other possible inter-relationships between cmpts (edges with circle heads) have been ignored. Further, for modeling purposes, we assume a four channel four sensor ABS [23]. The SMA (stochastic modeling abstraction) pattern is easily modified to represent other ABS schemes.

# 3 STOCHASTIC PETRI NET MODELS

The SMA patterns developed to model failure severity (FS) and coincident failures (CF), as well as usage-profiles (UP) are presented in [19][5]. Here, models are discussed in Petri net form and the code is presented for explanation wherever necessary.

The ABS is represented as a composition of all-important cmpts, as shown in Fig. 4. Cmpts are sorted into two groups: *central* and *axle*. The cmpts under *axle* are further divided according to the corresponding *wheel* – FRWheel (Front Right), FLWheel (Front Left), RRWheel (Rear Right) and RLWheel (Rear Left). A Speed Sensor, finds its place under the appropriate wheel groupings. A cmpt like the Hydraulic Pump, one for each axle, finds its place in the *axleCentral* group under the *axle* place. Only one Main Brake Cylinder exists and is under the *central* category. Each cmpt has its own SMA pattern (or sub-model), shown as dashed rectangles in Fig. 4.

## 3.1 Modeling Severity of Failures

The composed model (Fig. 4) also depicts the ABS under normal, degraded and lost stability operational conditions. The places *degraded_operation*, *loss_of_stability* and *loss_of_vehicle* model the severity of failure. The system is functioning normally when there are no tokens in any of these three places. The model is initialized with a single token in the *start* place. When the *central_op* and the *axle_op* transitions fire, a token is deposited in each place that represents a functioning cmpt of the ABS. The operation of each cmpt is now independent of every other cmpt (except where CFs are modeled explicitly).

The controller SMA pattern is seen in Fig. 5. Every cmpt either functions *normally* shown by the *controllerOp* transition, or *fails* shown by the *controllerFail* transition[6]. A failed cmpt may either cause degraded operation, loss of stability or loss of vehicle (shown as the *controllerDegradedOp*, *controllerLOSOp* and *controllerLOV* immediate transitions). The probability of any one of these three transitions occurring (based on measures of the real system) is different for each cmpt (see Table 1). When the failure causes either degraded operation or loss of stability, the cmpt continues to operate (token recycled back to the *controller* place), though the failure rate increases by $10^2$ and $10^4$ respectively. Loss of vehicle (indicated by a token in *loss_of_vehicle*), extreme loss of stability (indicated by three tokens in *loss_of_stability*) or extreme degraded operation (indicated by five tokens in *degraded_operation*) signify critical failures and determine the halting condition for the model.

## 3.2 Coincident Failures (CFs)

To model CFs, several dependencies among system cmpts are assumed (see Fig. 3). CFs are modeled in a manner similar to severity. The failure coincidence of two cmpts is modeled by causing the failure of one cmpt (to degraded operation or loss of stability) to increase the failure rate of the dependent cmpt. The failure of a cmpt A to a degraded mode causes the failure rate of a "related" cmpt B to increase by $10^2$. The failure of cmpt A to a

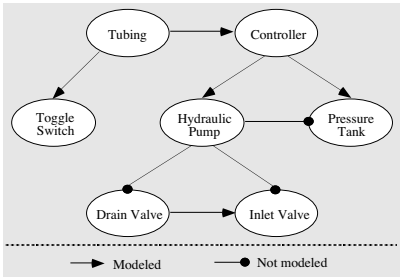


**Fig. 3: Component interdependencies.**

[4] The data was obtained from DaimlerChrysler. The failure rates listed in Table 1 however are dummy values. The real values are protected under a non-disclosure agreement.

[5] The SPNs were input to the Stochastic Petri Net Package (SPNP v. 6) tool in CSPL (C-based Stochastic Petri net Language).

[6] For Markov analysis, the time to failure of all components is assumed to be exponentially distributed.

lost stability mode causes the failure rate of a cmpt B to increase by $10^4$[7].

The function that calculates the failure rate of the transition *controllerFail* is shown in Fig. 6, which assumes a tubing malfunction affects the operation of the controller. Thus, when calculating the failure rate of the controller, the normal rate is increased by $10^2$ if the tubing has failed causing degraded operation (i.e., indicated by a token in the *tubingDegraded* place). Only a few CFs have been represented in the model. However, CFs between or among other cmpts can be easily accommodated by suitably modifying the failure rate function (as in Fig. 5) for the relevant cmpts using the same rule.

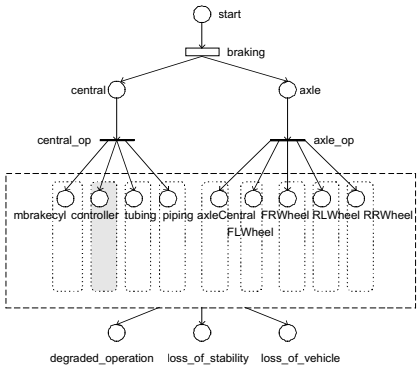


**Fig. 4: Composed SPN for the ABS.**

## 3.3 Modeling Usage-Profiles (UPs)

The global ABS model is represented as a composition of all-important constituent cmpts and remains unchanged (Fig. 4). To incorporate the UPs in the ABS model, the model of each individual cmpt, like the *controller* depicted in Fig. 5, is extended as shown in Fig. 7. The figure shows the *controller*, with the bold lines indicating the additions to the model. In case of a failure (*failedController*), the model differentiates between the two situations regarding whether the system was in active use (along the branch to transition labeled *mu*) or not (along the branch to transition labeled *alpha*). The parameter 1/*mu* indicates the mean duration of active use while the parameter 1/*alpha* indicates the mean duration of passive use. Active use (parameterized as a rate) is assumed to be exponentially distributed. If the failure occurs during the active period (*inUseController*), the system either continues to operate in the degraded (*controllerDegradedOp*) or lost stability mode (*controllerLOSOp*), or causes loss of vehicle (*controllerLOVOp*). In the case where the failure occurs during passive use of the system (*repairableController*), the fault can be repaired and an infinite repair rate is assumed (all repairs occur instantaneously, the system continues to operate as if no failure had occurred). The model may be extended to associate a cost with each time the failed cmpt is repaired).

To avoid state explosion, the model is simplified to incorporate the usage parameters while calculating the failure rate for each cmpt (see Fig. 6). The modified function for calculating the failure rate in light of the UPs is shown in Fig. 8. Essentially, the failure rate (considering only usage-parameters) is the sum of the failure rates *mu* and *lambda*. The value of these usage parameters was factored by the actual failure rate of the cmpts to avoid model stiffness. The *mu* value is 2.5 for low-usage periods and 250 for high-usage periods.

## 3.4 Extensibility of the SPN SMA Patterns

The SPN patterns developed for modeling CFs and severity and usage-profiles are easily extensible. The global SPN Model can be extended to include other cmpts deemed relevant to the ABS by including their corresponding sub-models. The sub-models, in turn, would be simple reproductions of the sub-models for other cmpts with different failure rates and probabilities. The model, developed for the four channel four sensor ABS, can be adapted to model other schemes of the ABS, by suitably changing the numbers of the relevant cmpts modeled (by either removing/adding the respective place, or updating the failure rate). Inter-dependencies between other cmpts (or among more than two) culminating in CFs can be modeled by updating the failure rates of the relevant cmpts.

Different SF groupings can be realized by simply extending the sub-models of the cmpts to include the necessary places depicting the severity-level. For example, the model representing the controller cmpt (Fig. 5) can be updated to have a transition (in addition to *controllerDegradedOp*, *controllerLOSOp* and *controllerLOVOp*) and a place (in addition to *degraded_operation*, *LOS* and *L O V*) to represent an additional failure severity level. The SMA patterns representing UPs can be extended to represent different usage-parameters (i.e., intensity of workload) by changing the value of *mu*. The model can also be extended to associate a cost with each time the failed cmpt is repaired by adding an additional place to track the number of times the cmpt is repaired (i.e., denoted by the number of tokens in this place).

# 4 SANS SMA PATTERNS

The SMA patterns, which use SAN primitives, were developed to assess severity and the effects of CFs in the context of different UPs are presented in [19]. The composed model for the ABS is shown in Fig. 9 and consists of three individual sub models: *Central_1, Central_2* and *Wheel*. The *Wheel* subnet is replicated to model the four wheels of the vehicle (SMA patters in the form of SANs are input to UltraSAN graphically.). The division into these three sub-groups facilitates the representation of CFs. As depicted in Fig. 3, the inlet valve and the drain valve (in the *Wheel* subnet) are correlated, as are the cmpts listed under the group *Central_2* (grayed subnet). All cmpts under *Central_1* are assumed to be independent of each other. This grouping avoids replicating the subnets unnecessarily (for modeling SFs and CFs) and thereby mitigates the state explosion problem.

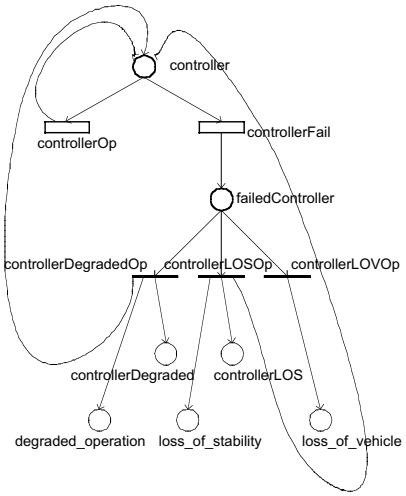


**Fig. 5: SPN model of the controller.**

## 4.1 Severity of Failures (SFs)

All subnets when combined to form the composed model share some common places: *degraded*, *LOS*, *LOV* and *halted*. The first three places model the severity of failure, while the *halted* place is used to determine the halting condition. The *Central_2* subnet is shown in Fig. 10. The presence of tokens in *degraded*, *LOS* and *LOV* represent degraded system operation, loss of stability and

```
double controllerRate()
{ double controller_rate = 0.0000006;

  if (mark("controllerLOS") > 0) return controller_rate * 10000;
  if ((mark("controllerDegraded") > 0) || (mark("tubingDegraded") > 0))
          return controller_rate * 100;
  return controller_rate;}
```

**Fig. 6: Variable failure rate function for coincident failures.**

[7] No data was available to confirm or validate this assumption.

loss of vehicle respectively (the same concept used in the SPN models). The system is operating normally when there are no tokens in any of these three places.

The subnet is instantiated with a single token in the *central_2* place. The *central2_op* activity fires and deposits a token in each of the five places: *hydraulicPump*, *pressureTank*, *toggleSwitch*, *controller* and *tubing*. The portion of the subnet for the *controller* cmpt is highlighted in Fig. 10 and discussed here in the context of SFs. The *controllerFail* activity models the failure of the controller. There are three possible outcomes of this activity. The *controller* either fails causing degraded operation (with probability 0.2, output gate *controllerDegraded_out*), or causes loss of stability (with probability 0.4, output gate *controllerLOS_out*), or causes loss of vehicle (with probability 0.4, output to *LOV*). In the former two cases the controller continues to operate in a degraded manner, as is evident by the recycling back of the token to the *controller* place. Further, the failure rate in this situation increases by $10^2$ (for degraded) and $10^4$ (for loss of stability) respectively. Table 2 gives the code snippet that achieves this.

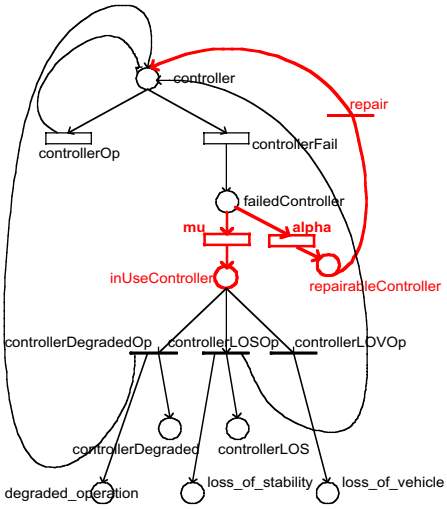


**Fig. 7: Controller SPN w/ usage-parameters.**

**Table 2: Activity rates model SFs and CFs.**

| Activity | Rate | Probability (Cases) 1 | 2 | 3 |
|---|---|---|---|---|
| controller Fail | MARK(controllerLOS)!=0? controllerRate*10000: (MARK(controllerDegraded)!=0 \|\| MARK(tubingDegraded)!=0 ?controllerRate*100 :controllerRate) | 0.4 | 0.4 | 0.2 |
| Hydraulic PumpFail | MARK(controllerLOS)!=0? hydraulicPumpRate*10000: (MARK(controllerDegraded)!=0 ? hydraulicPumpRate*100 : hydraulicPumpRate) | 1.0 | - | - |

Now, lets consider the *controllerFail* activity of Table 2[8]. If the controller fails causing degraded operation (i.e., MARK(controllerDegraded)!=0), it continues to function manifest by recycling the token back to the *controller* place, and the failure rate for the *controllerFail* activity increases by $10^2$ (i.e. controllerRate*100). Similarly, if the controller fails causing loss of stability (i.e., MARK(controllerLOS)!=0), (again) it continues to function manifest by recycling the token back to the *controller* place, and the failure rate for the *controllerFail* activity increases by $10^4$ (i.e. controllerRate*10000).

## 4.2 Coincident Failures (CFs)

As in the SPN models, failure coincidence of two cmpts is modeled by causing the failure of one cmpt (to degraded operation or loss of stability) to increase the failure rate of the dependent cmpt. The failure of a cmpt A to a degraded mode causes the failure rate of a "related" cmpt B to increase by $10^2$. The failure of cmpt A to a lost stability mode causes the failure rate of the dependent cmpt B to increase by $10^4$. Table 2 shows the failure rates for the activities modeling the controller and the hydraulic pump (other cmpt failures are modeled similarly).

Consider the *controllerFail* activity. Since failed tubing (in degraded mode) is assumed to affect the controller, if the *tubingDegraded* place is tokenized (i.e., MARK(tubingDegraded) != 0), the failure rate for the controller increases by $10^2$ (i.e., controllerRate*100). Similarly, the failure rate for the hydraulic pump (*hydraulicPumpFail*) increases by $10^4$ if the controller has failed causing loss of stability (or, if the controller is operating in a degraded mode, the failure rate increases by only $10^2$). CFs between other cmpts (or among more than two) can be modeled in a similar fashion.

## 4.3 Usage-Profiles (UPs)

The composed structure of the SAN model remains unchanged for modeling usage-profiles. Individual cmpt models within each subnet are updated to handle usage. To avoid the state explosion problem (for a cmpt in passive and active modes), the model is simplified to incorporate usage parameters while calculating the failure rate itself for each cmpt. The modified construct calculates the rate for each failure activity in light of the usage-profile (Fig. 11, using standard C).

The parameter 1/*mu* indicates the mean duration of active use for a given cmpt. To calculate the failure rate of the cmpt, the actual failure rate is added to the active usage rate (*mu* factored by the actual failure rate to avoid stiffness). The remaining constructs for SFs and CFs remain unchanged. The value of *mu* is assigned 2.5 for low-usage or 25 for high-usage periods.

## 4.4 SAN model Extensibility

The SMA patterns developed for modeling CFs and severity, and usage-profiles are easily extensible to include other cmpts deemed relevant to the system by adding other subnet patterns, or including additional cmpt(s) as part of an existing subnet. A new cmpt is modeled and composed in the same way as other existing cmpts by instantiating the new pattern with its own failure activity, corresponding output cases and probabilities.

Adding more cmpts can cause state space explosion due to the interleaving of the different token configurations within and among all the subnets of the composed model. The strategy should avoid using multiple places to denote multiple instances of the same cmpt where possible (e.g., instead of using two places to denote two axles, use a single place with the associated activity having a failure rate twice the failure rate of one axle). Using one SMA pattern and the Replicate primitive to model two axles enables the State Lumping Theorem to minimize the creation of unnecessary state space.

The model, developed for the 4-channel-4-sensor ABS, can be adapted to model other schemes, by suitably changing the

```
double controllerRate()
{ double controller_rate = 0.0000006;
  // usage parameter
  controller_rate += controller_rate * mu();
  if (mark("controllerLOS") > 0) return controller_rate * 10000;
  if ((mark("controllerDegraded") > 0) || (mark("tubingDegraded") > 0))
        return controller_rate * 100;
  return controller_rate;}
```

**Fig. 8: Variable failure rate function models usage-params.**

[8] UltraSAN requires the failure rate to be specified in a single statement, hence the use of the special if-then-else construct.

numbers for the relevant cmpts modeled (by either removing/adding the respective place, or updating the failure rate as described above). Different groupings for SFs can be realized by simply modifying the cmpt sub-models to include the necessary places representing the severity level. The severity levels can be altered by changing the number of tokens in each of the "severity" places that are necessary to trigger the halting condition. Different severity-levels can also be modeled by multiplying the failure rate of the affected cmpt by a different scalar (other than 100 and 10000 for degraded mode and lost stability as was used here). Inter-dependencies between other cmpts (or among more than two cmpts), which cause CFs, can be modeled by updating the rates of the activities that model failure of those cmpts.

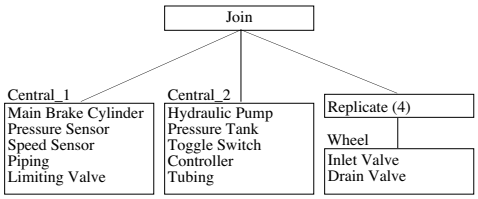


**Fig. 9: The composed model of the ABS.**

The SAN model representing UPs can be updated to represent different usage-parameters or intensity of workload by simply changing the value of *mu* in Fig. 11. The model can be extended to associate a cost with each time the failed cmpt must be repaired; by adding an additional place to keep track of the number of times the cmpt has been repaired (denoted by the number of tokens in this place).

# 5 ANALYSIS

The reliability of the system at time *t* is computed as the expected instantaneous reward rate at time *t*. To determine the reliability of the system, transient analysis of the developed models was carried out and the reliability measured between 0 and 50K hrs. The time duration was deliberately conservative, even though the average life span of a passenger vehicle ranges from 3000 – 9000 hrs, the reliability measures were determined for up to 50K hrs[9].

Since, it is beyond the scope (and the means) of this research to validate the results from the analytic experiments against real data, two different stochastic formalisms have been used to carry out the reliability analysis. The transient analysis of the developed SPN models resulted in 164,209 tangible markings, of which 91,880 were absorbing. The running time of the solver on the models was ~144-168 hrs. The developed SAN models were solved at a tolerance of $10^9$, and resulted in the generation of 859,958 states. The running time of the solver on the models was ~120-144 hrs. The results, from the analyses of each of the (SPN and SAN) models developed, using the SPNP and the UltraSAN tools respectively, are presented and compared in this section.

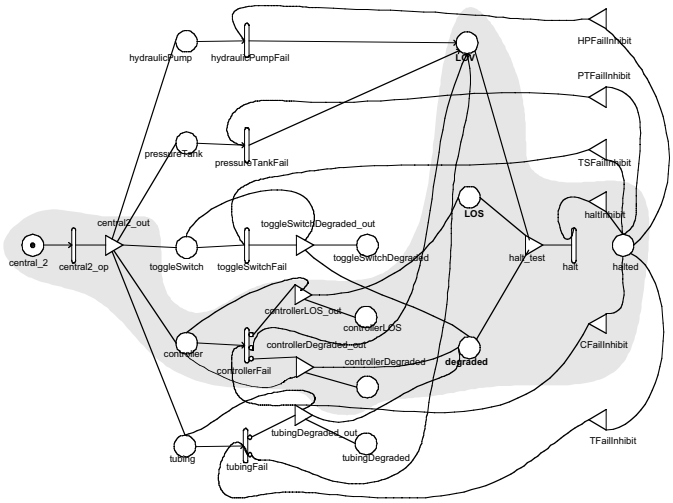


**Fig. 10: *Central_2* subnet with the controller highlighted.**

## 5.1 Coincident and Severity of Failures

The SPN / SAN model results representing SFs and CFs are shown in Fig. 12 (LHS). The Y-axis gives the measure of interest –reliability; while the time range (0-50K hrs) is on the X-axis. The interval between time points was not constant for the entire time range (i.e., the X-axis is not linear [interval size increases with time]). As expected, the reliability steadily decreases with time. The box highlights the vehicle average lifetime range.

The two SPN model curves (representing SFs + CFs failures and without either) for CFs are completely overlapped. The Mean Time to Failure (MTTF) for the CFs model (784,856 hrs) is approximately 421 hrs less than the model without CFs (785,277 hrs). For the *limited*[10] number of CFs that have been modeled, the difference of 421 hrs in the two cases is considered well within the confidence interval.

For the SAN models, the reliability functions diverge perceptibly after around 1K hrs of operation, and the difference continues to increase with time. At 50K hrs, the reliability has dropped down to 0.21 when CFs are modeled, and down to 0.30 when CFs are not modeled, a difference of 0.09 in reliability for the two cases within 50K hrs. The MTTF at 50K hrs when CFs are not considered is 29,167 hrs, and when considered is 25,409 hrs, a difference of 3,758 hrs.

The range difference in the reliability values produced by the two different formalisms may be attributed to the way the reliability reward is defined in each. The SPN reward rate was defined as a single set of discrete 0/1 values, while the SAN reward rate function models a range between 0 and 1 (a function of the number of tokens in the *degraded*, *LOS* and *LOV* places). Therefore, the different rewards accumulate at different rates, and this explains the disparity in the reliability values at any given point in time. It is evident that representing SFs/CFs in the model contributes to the fidelity of reliability prediction given the aforementioned assumptions.

## 5.2 Representing Usage-Profiles

The SPN / SAN model results representing UPs are shown in Fig. 12 (RHS). For the SPNs, the reliability of the system with heavy usage decreases steeply in the first 1K hrs of

```
MARK(componentBLOS)!=0 ?
(componentARate+componentARate*mu)*10000 :
(MARK(componentBDegraded)!=0 ?
(componentARate+componentARate*mu)*100 :
(componentARate+componentARate*mu)).
```

****** Is Equivalent to: ******

```
if(MARK(componentBLOS)!=0)
  return (componentARate+componentARate*mu)*10000;
else if(MARK(componentBDegraded)!=0)
  return (componentARate+componentARate*mu)*100;
else     return (componentARate+componentARate*mu);
```

**Fig. 11: Construct to model usage-profiles.**

[9] Transient analysis was carried out using the Stochastic Petri Net Package (SPNP) version 6 (for the SPN models) and UltraSAN version 3.5 (for the SAN models) on a Sun Ultra 10 (400 MHz, 500MB memory).

[10] One may speculate that there is some kind of relationship (perhaps linear) between the number of dependencies modeled and the difference observed in the graphs and the MTTF values.

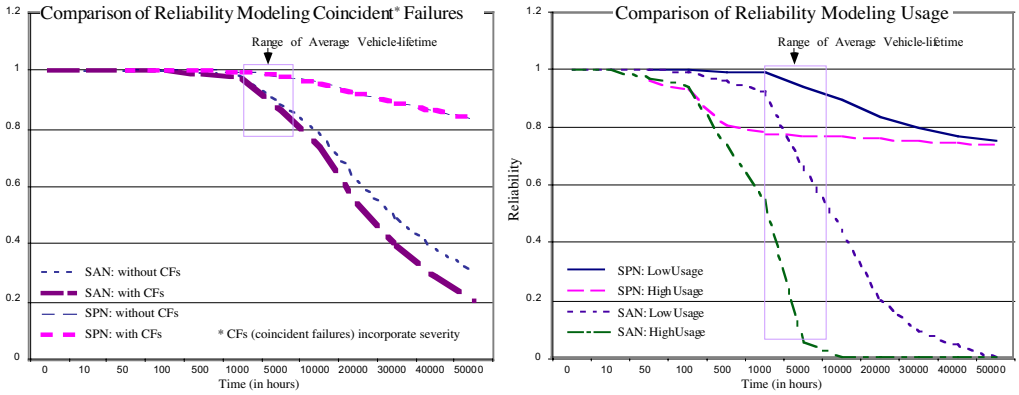


**Fig. 12: Comparison of SPN and SAN model results.**

operation. The reliability of the system with moderate usage decreases after 2.5K hrs of operation and steadily thereafter. MTTF for the high-usage case is ~771,023 hrs compared to 775,112 hrs for the low-usage case, a difference of 4,089 hrs.

For the SAN models, the reliability of the system with heavy usage decreases steeply past 100 hrs of operation. The reliability of the lightly used system decreases only after 100 hrs of operation and steadily thereafter. The expected vehicle lifetime is 3,000-9,000 hrs. Reliability for the high-usage profile drops from around 0.55 down to approximately 0.05. For the same duration, the reliability for the low-usage drops from 0.9 to only 0.5, a difference of approximately 0.45 after 10K hrs of operation. Thus, aggressive use of the system, causes the reliability to drop rapidly (as expected) than when the system is used conservatively. The affect of usage is more pronounced using the SAN SMA patterns. Table 3 provides a summary of comparison criteria and results.

**Table 3: Comparison criteria/results for SPN/SAN models.**[11]

| Criteria/Results | SPN Model | SAN Model |
|---|---|---|
| Assumptions | Same | Same |
| Reliability Measure | Defined as a set of 0/1 rewards | Function: # of tokens in degraded, LOS & LOV places |
| Number of states generated | 164,209 states | 859,958 states |
| Environment/Platform | Same | Same |
| Running Time of Solvers | 144-168 hrs | 120-144 hrs |
| Reliability at 9K hrs (With severity and coincident failures vs. without) | 0.95792578 vs. 0.95792653 | 0.73672 vs. 0.786 |
| Difference in reliability at 9K hrs | 0.00000075 | 0.04928 |
| Reliability at 9K hrs (Low vs. High-usage) | 0.89621556 vs. 0.76658329 | 0.4455167 vs. 0.3130521 |
| Difference in reliability at 9K hrs (Low-usage minus High-usage) | 0.12963227 | 0.1324646 |

# 6 SUMMARY

The objective of this study was to exemplify a generic (extensible and realistic) SCM method for analyzing reactive embedded systems emphasizing nonfunctional properties (i.e., focusing on SFs, CFs and UPs). We represented these characteristics in the special case of the ABS characterized by empirical data. We discussed the SCM strategy and the extensibility of the SMA patterns. The results from comparing two analyses using different modeling formalisms were compared, in the absence of other validation procedures (see Fig. 13). The goal of exemplifying this approach as a generic extensible SCM method in this domain was achieved through the discussion including how to incorporate greater complexity and and/or modifications with respect to the aforementioned assumptions.

## 6.1 Conclusion

The characteristics of SFs, CFs and UPs were successfully incorporated into the model developed for the ABS of a passenger vehicle. This resulted in a more realistic model (with real data being used to determine failure rates) despite the somewhat unrealistic IID (Independent and Identically Distributed) assumption that is necessary using Markov theory. The degree of complexity and the level of abstraction that is feasible to solve for the desired predictive measures suffers (as is common knowledge) from combinatorial explosion.

To combat the state space problem, this paper asserts by example, a structured compositional modeling approach (using SMA patterns) that can be used to combat the combinatorial explosion of states in the analysis of large systems.

(1) A *composition rule* describes how the nonfunctional properties of the system model can be determined from the nonfunctional properties of its cmpts, without knowledge of their internal structure. In this way, each cmpt is simpler and smaller than the composed system. Composition requires the model be evaluated by checking the consistency and completeness of its specification (e.g., using deductive methods and ensuring that the model is mathematically well founded).

(2) *Abstraction* ensures that high-level abstractions of cmpt groupings are preserved by their SMA patterns are faithfully precise and correct). The stochastic modeling formalisms/tools automate parts of the compositional process and transformation from the expressive form of the model (e.g., Petri net) into a Markov model and its solution.

The model specification method is illustrated using an ABS system that cannot be solved quickly (due to the combinatorial explosion of states) without compositional reasoning and abstraction. The approach we are promoting ensures that the patterns used are both precise and correct –the model accurately predicts the real result and is mathematically well founded.

## 6.2 Future Work

The ABS is a small part of the DDR (Dynamic Driving Regulation) system consisting of other subsystems like the ESA (Electronic Steering Assistance), the TC (traction control), and the PT (power transmission) as shown in Fig. 13. SMA patterns can be developed for the ESA, TC and the PT sub-systems as well. To achieve more realistic models, this approach can be extended to incorporate other closely related sub-systems for analysis of a more complex composed system for determine how different configurations and dependencies affect overall system reliability/ availability). To extend the SCM method as described in our example to achieve the composition of multiple sub-systems, further abstractions as well as high performance computing resources (e.g., terascale) are necessary.

[11] Assumptions and data used here were the same in both cases, while the reliability reward measure was different. The states generated for the SPNs were much less than those for the SAN models, but SPN solver running time was longer. The reliability results at 9K hrs (the expected end-of-life time point) are also provided.

Both modeling formalisms lend themselves to sensitivity analysis at various levels. The models developed so far can be used to carry out sensitivity analysis for the system under study to identify the cmpts that are most likely to fail and thereby making the system susceptible to critical failures. Armed with such knowledge, system/software architects can make informed decisions as they pertain to how inherently reliable or safe their choices may be and/or make economic/cost tradeoffs.

The complete DDR system is a very complex, and a model capturing all of its essential features/characteristics would itself be very complex, precluding an efficient analytical solution. In this case, the model must be studied carefully to avoid unnecessary replication and/or addition of unimportant factors that would aggravate the state space problem.

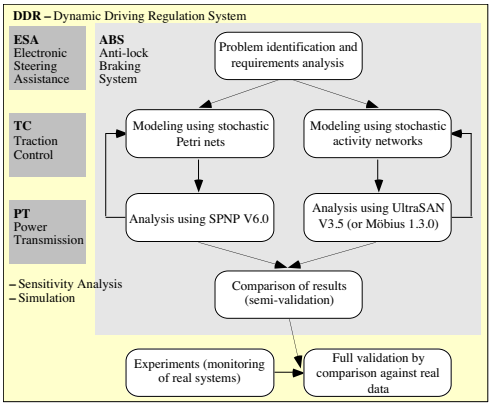


**Fig. 13: Scope of Sys composition/validation.**

Various (confidential/proprietary) studies exist that record the effect of various system cmpts, usage profiles and their failure rates on safety and reliability properties. The key strategy for validation includes the measures and procedures used for confirming that each safety function conforms to the specified system safety requirements [4]. One would like to have comprehensive data collected that could account for the contribution of CFs and usage information: data about (1) the effect of degraded operation/loss of stability on cmpt failure rate, (2) the correlation of failures between cmpts, (3) the effect of demand/usage on failure rates and, (4) precise quantification of workload durations. Such data would provide evidence to validate the analyses and help to further evolve the developed models (i.e., SMA patterns) to make more precise and correct predictions.

# 7 REFERENCES


1. M. A. Vouk, "Software reliability engineering," *presented at the RAMS*, Los Angeles, CA, 2000.
2. K. Jerath and F. T. Sheldon, "Reliability analysis of an anti-lock braking system using Stochastic Petri Nets," in *Proc. Workshop PMCCS 5*, Erlangen, Germany, 2001, pp. 56-60.
3. F. T. Sheldon, S. Greiner and M. Benzinger, "Specification, safety and reliability analysis using Stochastic Petri Net models," in *Proc. IWSSD*, San Diego, CA, 2000, pp. 123-132.
4. *Functional safety of electrical/electronic/programmable electronic safety-related systems*, CEI/IEC Std. 61508, 1998, Part 4 in Sect. 3.
5. F. A. Gay, "Performance evaluation for gracefully degrading systems," in *FTCS-9*, Madison, 1979, pp. 51-58.
6. M. Hecht, D. Tang and H. Hecht, "Quantitative reliability and availability assessment for critical systems including software," in *Proc. CCA*, Gaithersburg, Maryland, 1997.
7. D. P. Siewiorek and R. S. Swarz, *Reliable computer systems: Design and evaluation*. 2ed. Woburn, MA: Digital Press, 1992, pp. 908.
8. G. Balbo, Professor, Universita di Torino, Italy. Personal Comm. EEF-Sum-Sch. FMs/Perf. Anal., Holland, July 2000.
9. J. B. Dugan, "Experimental analysis of models for correlation in multiversion software," *IEEE ISSRE*, 1994, pp. 36-44.
10. D. E. Eckhardt and L. D. Lee, "Theoretical basis for the analysis of multiversion software subject to coincident errors," *IEEE TSE*, 11(12), pp. 1511-1517, 1985.
11. B. Littlewood and D. R. Miller, "Conceptual modeling of coincident failures in multiversion software," *IEEE TSE*, vol. 15(12), pp. 1596-1614, 1989.
12. J. Arlat, K. Kanoun and J.-C. Laprie, "Dependability modeling and evaluation of software fault-tolerant systems," *IEEE TC*, vol. 39(4), pp. 504-513, 1990.
13. J. Meyer, Professor, U. of Michigan, Ann Arbor, MI. Personal Comm. at PMCCS5, Erlangen, Germany, Sept. 2001.
14. M. C. Hsueh, R. K. Iyer and K. Trivedi, "Performability modeling based on real data: A case study," *IEEE TC*, vol. 37(4), pp. 478-484, 1988.
15. X. Castillo and D. P. Siewiorek, "Workload, performance and reliability of digital computing systems," in *IEEE Symp. FTC*, Portland, ME, 1981, pp. 84-89.
16. J. Meyer and L. Wei, "Analysis of workload influence on dependability," in *IEEE Symp. FTC*, Tokyo, 1988, pp. 84-89.
17. L. M. Malhis, W. H. Sanders and R. D. Schlichting, "Numerical evaluation of a group-oriented multicast protocol using SANs," in *PNPM*, Durham, 1995, pp. 63-72.
18. M. A. Qureshi and W. H. Sanders, "The effect of workload on the performance and availability of voting algorithms," in *Proc. MASCOTS*, Durham, NC, 1995, pp. 217-224.
19. K. Jerath, "Modeling and Stochastic Analysis of Embedded Systems Emphasizing Coincident Failures, Failure Severity and Usage-Profiles," Masters Thesis, WSU, Pullman, 2002. http://www.csm.ornl.gov/~sheldon/public/Jerath-Thesis.pdf.
20. G. Ciardo, J. K. Muppala and K. Trivedi, "SPNP: Stochastic Petri Net Package," in *Proc. of Int. Workshop on Petri Nets and Performance Models*, Kyoto, Japan, 1989, pp. 142-151.
21. J. Couvillion, et al., "Performability modeling with UltraSAN," *IEEE Software*, vol. 8(5), pp. 69-80, 1991.
22. F. T. Sheldon and S. Greiner, "Composing, analyzing and validating software models to assess the performability of competing design candidates," *ASE*, vol. 8, pp. 239-287, 1999.
23. R. Bosch, *Automotive Handbook*. 4 ed. Manchester, Tennessee: Bentley Pubs, 1993, pp. 852.


**Author Biographies:**

***Sheldon*** (M'1991) is a research staff member at ORNL. He was a Professor at WSU, CU and research staff at DaimlerChrysler, Lockheed Martin, Raytheon and NASA Langley/Ames Research Centers. His research is concerned with developing and validating models, methods and supporting tools for the creation of safe, secure and correct software/systems. He received his Ph.D. at UTA in 1996 and founded the SEDS (Software Engineering for Dependable Systems) Lab in 1999.

***Jerath*** (M'2002) received an MS in CS from WSU in August 2002 and her BS in CE is from Delhi College of Engineering India. Her main research interests are software components and reliability modeling. Currently, a Software Design Engineer at Microsoft (SharePoint Portal Grp), she was also a Software Engineer at IBM Global Services, in the area of web application development and three-tier client server applications. She is a SUN certified Java programmer. She received an honorable mention for Outstanding Woman Graduate Student from the Assoc. of Faculty Women at WSU ('02) and is recipient of the All American Scholar and National Collegiate Computer Science Awards ('02). She was awarded the chief minister's gold medal and the lieutenant governor's gold medal for best performance in undergraduate studies as well as the IEEE P. Kundu gold medal for outstanding performance in industrial training.