ON THE CONFIGURATION OF FAULT TOLERANT DISTRIBUTED PROCESS CONTROL SYSTEMS

G. D. Hassapis
Dpt. of Computer and Electronics Engineering
Faculty of Electrical Engineering
School of Engineering
University of Thessaloniki 540 06
Greece

ABSTRACT. There are microcomputer based systems, organized in a hierarchical and distributed architecture basis and used to control and manage large industrial plants. They consist of independent fault tolerant microcomputer subsystems which are usually grouped together into physical structures, called nodes, and communicate between each other through a local area network. Since these systems operate costly and dangerous plants they require high reliability and availability. An availability model for this class of systems and its computational procedure, based on the Laguerre transform[1], is developed in this work. It may be used to assess the availability of system architectures with different fault tolerant configurations and number of functions per node. Then, depending on the application requirements, an architecture and its best configuration may be selected. The system behaviour is described by a multivariate stochastic process. It is assumed that at time t or while the system is in repair only one hardware or software module may fail. The use of the model is demonstrated by an application example.

1. INTRODUCTION

A distributed process control system can be any system of computers and peripherals utilized to perform monitoring and control functions of large process units. However, commercially available systems [2],[3],[4], which are classified as distributed process control systems, follow a hierarchical and distributed hardware architecture and they are loaded with a library of software packages which are able to perform well-defined control, monitoring, plant management and system self-diagnosis functions. Such a distributed system may be devided into four levels. These are the Input/Output the Data Acquisition and Control, the Network and the Operator's Console/Supervisory Computer levels.

At the Input/Output level a number of independent modules, usually corresponding to a printed circuit card, receive and send various types of analog and digital signals from and to the process. Each module

S. Tzafestas et al. (eds.), System Fault Diagnostics, Reliability and Related Knowledge-Based Approaches, Vol. 2, 327–344.

may include more than one data channels and communicates with modules of the Data Acquisition and Control level.

The microcomputer based modules of the Data Acquisition and Control level perform either input signal filtering, smoothing, conversion to engineering units, alarm settings and process database updating or PID control of a number of process variables. Each module at this level communicates with the process through the circuitry of the Input/Output level and with the Operator's Console/Supervisory Computer level through the circuitry of the Network level.

The distributed control systems use fault tolerant schemes at all levels. The existence of these schemes reduce the possibility of loosing control of critical process loops. The usual forms of fault tolerance are: (1) the duplication of each microcomputer[6]; (2) the shared memory approach[2], (3) the allocation of an active standby module for every N main modules[5] and (4) voting configurations[6].

The Network level is a local area network providing a medium for the intercommunication between the Data Acquisition and Control level and the Operator's Console/Supervisory Computer level. The network types used are the ring-based, cluster-based and broadcast based systems [4]. Network fault tolerance is usually achieved by simple parallel redundancy of all the functional modules of the network.

The modules of the Operator's Console/Supervisory Computer level provide a human interface to the operator and allow the execution of higher level applications, such as performance optimization, energy management, sequencing and scheduling. Also, there is provision for connecting modules of this level in double parallel redundancy schemes.

One of the major performance indices considered during the installation of distributed control systems is their availability. The value of this index depends on the architectural characteristics of a system, the failure probabilities of its hardware and software components and the repair policy followed. Since these systems have extensive self-diagnosis routines and highly modular construction, their hardware maintenance is limited to the replacement of the parts declared fauly by the self-diagnosis program. Unfortunately, this policy cannot be applied to software maintenance. A software fault may appear when the system is used in a specific application environment and its repair may result to the introduction of other latent faults in the package.

This paper presents a model to assess the time-dependent availability of a distributed control system configuration. To apply this model the system architecture and its distributed software must be able to be devided into independent hardware modules and software packages. Also, it must be possible to identify the hardware modules required to run each package. In the case of the considered commercially available distributed control systems this information is usually provided upon request from the system manufacturer in a schematic or chart form[3],[5]. Failure data for each hardware module and software package must also be requested from the manufacturer, unless the experience of other system users is available.

2. DEFINITIONS AND ASSUMPTIONS

Let us consider a distributed control system consisting of independent hardware modules and independent software packages, all of which are subject to random failures. The system is assumed to control M process loops by means of M output signals. Whenever an output signal assumes an unsafe value due to a fault in a hardware or software package, the system is considered to be in a failed state. A repair activity is then undertaken and the system is brought back to operative state. More formally the following definitions and assumptions are considered:

(1) The states of the system are devided into two disjoint subsets. One subset represents the operational system states and the other the failed states. A state is considered operational when the values of the system outputs drive the controlled process to a correct and safe state. If at least one output assumes a dangerous or incorrect value then the system is considered to be in a failed state. We now introduce the following notation.

$S_0 = [\ v_0\ ]$ is the set of operational states.

According to the given definition there is only one operational state.

$S_F = [\ v_i,\ i=1,2,\ldots\ 2^M-1]$ is the set of failed states.

(2) Each operational or failed system state may be mapped to a set of hardware and software components states. That is, for v_0 there is the set:

$$A = [\ a_l,\ l=1,2\ \ldots\ L]$$

where a_l is a vector of $\mu+\nu$ binary digits. Each one of the first μ digits represents the state of a hardware module and similarly each one of the last ν digits represents the state of a software package. The failed state of a hardware module or a software package is denoted by 0 and the operational state by 1. Because of the fault tolerant structure of a distributed control system it may include vectors with hardware and software components in failed state. Similarly, S_F is mapped to the set:

$$B = \bigcup_{l=1}^{L} B_l$$

where $B_l = C_l U D_l U E_l$

The subset C_l is the set of the hardware-software states which differ from the operational state a_l only in the value of the state of a hardware module. That is,

$$C_l = [\ c_l^r,\ r=r_1,r_2\ \ldots\ r_k\ ] \quad r_i \in [1,\mu\ ],\ i=1,2\ \ldots\ k$$

The subset D_1 contains the hardware-software state vectors which differ from the operational state a_1 only in the value of the state of a software package. That is,

$$D_1 = [\, d_1^z,\ z=z_1,z_2 \ldots z_{p_1}],\quad z_i \in [1,\nu\ ],\ i=1,2 \ldots p_1$$

The subset E_1 contains all the remaining hardware-software state vectors corresponding to the S_F set.

Further, the A set may be devided to subsets of the form:

$$A_1^1 = [\, a_1^r,\ r=r_1,r_2 \ldots r_{q_1}],\quad r_i \in [1,\mu\ ],\ i=1,2 \ldots q_1$$

where a_1^r are the hardware-software state vectors of A which differ from the state a_1 in the value of the r hardware module. Similarly

$$A_1^2 = [a_1^z\ ,\ z=z_1,z_2 \ldots z_{w_1}]\quad z_i \in [1,\nu\ ],\ i=1,2 \ldots w_1$$

(3) A software package may fail only when the hardware components required to run this package are functioning. The hardware components, however, may fail while the software package is under repair.

(4) A hardware component constitutes an alternating renewal process where up times have a probability distribution function(p.d.f) $H_i(x)$ $1 \leqslant i \leqslant \mu$. The down time of any hardware component is exponentially distributed with rate λ_o .

(5) The maximum number of software errors in a software package is limited to a number K.

(6) At time t=0 the state vector of the hardware and software components of the system may be any member of the A set. Each software package has a number of errors n_i, $1 \leqslant i \leqslant \nu$ which is a discrete nonnegative random variable with probability vector,

$$\underline{b}^{(i)T} = (\, b_0^{(i)}, b_1^{(i)}, \ldots\ b_K^{(i)}\ ),$$

where $b_j^{(i)} = P[n_i=j],\quad j=0,1,2 \ldots K$

(7) The probability that there are n_i errors remaining in the i software package after a repair, provided that there are k_i errors at the beginning of the repair is $p_{k_i n_i}$. It is assumed that $p_{00}=1$ and $p_{0n_i}=0$.

(8) If there are n_i errors in the i software package upon the completion of the repair and if there are not hardware failures influencing the operation of the package, then the up-time of the package has cummulative distribution function(c.d.f) $A_{i;n_i}(x)$, , $n_i=0,1,2 \ldots K$.

It is assumed that for $n \geq 1$ $A_{i:n_i}(x)$ is absolutely continuous with p.d.f $a_{i:n_i}(x)$. By definition $A_{i:0}(x)=0$, as if there are no errors in the software package, then there will be no failures.
(9) The failures of any hardware component occur according to a Poisson process with parameter λ_o . Similarly the failures of any software package occur according to a Poisson process, there is however for each package a different parameter λ_i , i=1,2, ... V .
(10) It is assumed that during a very small period of time dt only one hardware component or one software package may fail. When the system is under repair there is the possibility to occur either an additional hardware or software fault.
(11) The following random processes are introduced.

a(t) the hardware software state at time t.

$m_i(t)$ the number of failures of the ith software package at time t.

$n_j(t)$ the number of errors in the jth software package at time t.

(12) The given initial conditions of the a(t), $m_i(t)$ and $n_j(t)$, $1 \leq i \leq \mu$ $1 \leq j \leq V$ are $a(0-)=a_0$, $a(0+)=a_1$, $1 \in [1,L]$, $m_i(0)=0$ and $n_i(0)=n_i$, where $a_0=00 \ldots 0$
(13) The lifetimes and repair times of all the software and hardware components ate mutually independent,
(14) System repair is initiated when it enters one of the states of the B set.

3. MODEL DESCRIPTION

We wish to determine the probability that a distributed control system is operational at time t. If we define the multivariate process:

$$X(t)=[\, a(t),m_1(t),m_2(t) \ldots m_V(t),n_1(t),n_2(t) \ldots n_V(t) \,] \quad (1)$$

then the system will be operational at time t if X(t) is in one of the discrete states $(a_1,m_1,m_2 \ldots m_V,n_1,n_2, \ldots n_V)$, $1 \in [1,L]$, $m_i \in [0,K]$ and $n_i \in [0,K]$, i=1,2 ... V .
Then, it can be readily seen that the probability of a distributed system being operational at time t, that is its availability is:

$$A(t)=P[a(t)=v_0]=\sum_{i=1}^{V}\sum_{j=1}^{V}\sum_{n_i=0}^{K}\sum_{n_j=0}^{K}\sum_{l=1}^{L} P(X(t)=(a_l,m_1 \ldots m_V,n_1 \ldots n_V)) \quad (2)$$

To find $P[X(t)=(a_1,m_1,m_2 \dots m_\nu,n_1,n_2, \dots n_\nu)]$ one has to evaluate the rates of transition from any state of the set A either to states of the same set or to states of the B set and vice versa, taking into consideration the assumptions of the previous section.

Formally these transitions satisfy the relationships:

$$P(X(t+dt)=(c_l^r, m_1,\dots m_\nu,n_1\dots n_\nu)/X(t)=(a_l, m_1\dots m_\nu, n_1\dots n_\nu))= g^{l:r}_{m_\zeta:n_\zeta}(t).dt \tag{3}$$

If it is impossible to have such a state transition for a certain value of l, then $g^{l:r}_{m_\zeta:n_\zeta}(t)=0$

$$P(X(t+dt)=(d_l^\zeta, m_1,\dots m_\zeta+1,\dots m_\nu,n_1,\dots n_\zeta,\dots n_\nu)/X(t)=((a_l, m_1\dots m_\zeta\dots m_\nu, n_1\dots n_\zeta\dots n_\nu))= w^{l:\zeta}_{m_\zeta:n_\zeta}(t).dt \tag{4}$$

$$P(X(t+dt)=(a_l^r, m_1,\dots m_\nu,n_1\dots n_\nu)/X(t)=(a_l, m_1\dots m_\nu, n_1\dots \dots n_\nu))= \tau^{l:r}_{m_\zeta:n_\zeta}(t).dt \tag{5}$$

$$P(X(t+dt)=(a_l^\zeta, m_1\dots m_\zeta+1\dots m_\nu, n_1\dots n_\zeta\dots n_\nu)/X(t)=(a_l, m_1\dots m_\zeta\dots m_\nu, n_1\dots n_\zeta\dots n_\nu))= \psi^{l:\zeta}_{m_\zeta:n_\zeta}(t)\,dt \tag{6}$$

If we consider that during the repair time of a failed module another hardware module fails, one has:

$$P(X(t+dt)=(a_l, m_1\dots m_\nu,n_1\dots n_\nu)/X(t)=(c_l^r, m_1\dots m_\nu, n_1\dots \dots n_\nu), s=0)= p^{l:r:s}_{m_\zeta:n_\zeta}(t).dt \tag{7}$$

where r and s are the two failed hardware modules, $1\leqslant r\leqslant\mu$, $1\leqslant s\leqslant\mu$.

$$P(X(t+dt)=(a_l, m_1\dots m_\zeta+1\dots m_\nu, n_1\dots n_\zeta\dots n_\nu)/X(t)= (d_l^\zeta, m_1\dots m_\zeta\dots m_\nu, n_1\dots n_\zeta\dots n_\nu), r=0)= \varphi^{l:\zeta:r}_{m_\zeta:n_\zeta}(t).dt \tag{8}$$

where r and ζ are the hardware and software components which have failed, $1 \leqslant r \leqslant \mu$, $1 \leqslant \zeta \leqslant \nu$.

Similarly, when a software failure occurs during the repair time of a hardware or a software component then:

$$P(X(t+dt)=(a_1, m_1 \dots m_\zeta+1 \dots m_\nu, n_1 \dots n_\zeta \dots n_\nu) / X(t) = (c_1^r, m_1 \dots m_\zeta \dots m_\nu, n_1 \dots n_\zeta \dots n_\nu),\ \zeta \to 0)) = \gamma_{m_\zeta+1:n_\zeta}^{1:r:\zeta}(t) \qquad (9)$$

where r and ζ are the failed hardware and software component respectively $1 \leqslant r \leqslant \mu$, $1 \leqslant \zeta \leqslant \nu$.

$$P(X(t+dt)=(a_1, m_1 \dots m_\zeta+1 \dots m_\nu, n_1 \dots n_\zeta \dots n_\nu) / X(t) = \qquad (10)$$
$$(d_1^\zeta, m_1, \dots m_\zeta \dots m_\xi \dots m_\nu, n_1 \dots n_\zeta \dots n_\xi \dots n_\nu),\ \xi\ 0) = f_{m_\zeta+1:m_\xi+1:n_\zeta:n_\xi}^{1:\zeta:\xi:}(t).dt$$

where ζ , ξ , are the two failed software modules, $1 \leqslant \zeta \leqslant \nu$, $1 \leqslant \xi \leqslant \nu$.

Suppose now that X(t) has just entered the state $(c_1^r, m_1 \dots m_\nu, n_1 \dots n_\nu)$ at time t because of the failure of the r hardware component. Immediately the module repair starts. It may, however, be interrupted by either the failure of another hardware module or the failure of a software component, the operation of which does not depend on the initially failed hardware module. It has been assumed that these interruptions occur according to a poisson process with a parameter λ_o for any hardware module and λ_i for the ith software package. Under these conditions, the effective repair time is obtained by adding the repair time y of the first failed hardware module and the repair time of the other failed hardware or software component. If we denote this conditional effective repair time density in the case of the two hardware failures by $h_y^{eff}(x)$, one has[1]:

$$h_y^{eff}(x) = \lambda_o y \exp(-\lambda_o y) h^{(1)}(x-y) \qquad (11)$$

where $h^{(1)}(x)$ is the convolution of h(x) with itself, and unconditioning with respect to y, one obtains:

$$h^{eff}(x) = \int_o^\infty h_y^{eff}(x)\, h(y)\, dy \qquad (12)$$
$$h(x) = \exp(-\lambda x)$$

where the * denotes convolution.

Similarly, in the case of a hardware and softwate failure the effective repair time density is:

$$q_{r:n_\zeta}^{eff}(x) = [\lambda_o x \exp(-\lambda_o x)\, a_{\zeta:n_\zeta}(x)] * [h(x) * a_{\zeta:n_\zeta}(x)] \qquad (13)$$

where n_ζ is the number of errors in the software package at the beginning of the repair and r is the failed hardware component.

If we consider a possible failure of the ζth software package followed either by a failure of the ith hardware module or the ξth software package, then the effective repair time densities $\pi^{eff}_{\zeta:r:n_\zeta}(x)$ $\rho^{eff(x)}_{\zeta:\xi:n_\zeta:n_\xi}$ will be respectively:

$$\pi^{eff}_{\zeta:r:n_\zeta}(x) = [\lambda_\zeta x \exp(-\lambda_\zeta x)\, a_{\zeta:n_\zeta}(x)] * [h(x) * a_{\zeta:n_\zeta}(x)] \tag{14}$$

$$\rho^{eff}_{\zeta:\xi:n_\zeta:n_\xi}(x) = [\lambda_\xi x \exp(-\lambda_\xi x)\, a_{\xi:n_\xi}(x)] * [h(x) * a_{\zeta:n_\zeta}(x)] \tag{15}$$

n_ζ and n_ξ are the number of errors in the ζth and ξth software package respectively, at the beginning of the repair.

For $\underline{X}(t)$ to enter the state $(a_1, m_1 \ldots m_\nu, n_1, \ldots n_\nu)$ from the state $(c^r_i, m_1 \ldots m_\nu, n_1 \ldots n_\nu)$ at time t it should have entered the state at time t-x and remained there until x expires, $0 < x < t$. This probabilistic argument is expressed by the relationship:

$$p^{1:r:s}_{m_\zeta:n_\zeta}(t) = \int_0^t g^{1:r}_{m_\zeta:n_\zeta}(t-x)\, h^{eff}(x)\, dx \tag{16}$$

Similarly, one has:

$$\gamma^{1:r:\zeta}_{m_\zeta+1:n'_\zeta}(t) = \int_0^t g^{1:r}_{m_\zeta:n_\zeta}(x)\, q^{eff}_{r:n_\zeta}(t-x)\, dx \tag{17}$$

$$\varphi^{1:\zeta:r}_{m_\zeta+1:n'_\zeta}(t) = \int_0^t p_{n_\zeta:n'_\zeta}\, w^{1:\zeta}_{m_\zeta:n'_\zeta}(t-x)\, \pi^{eff}_{\zeta:r:n_\zeta}(x)\, dx \tag{18}$$

$$f^{1:\zeta:\xi}_{m_\zeta+1:m_\xi+1:n'_\zeta:n_\xi}(t) = \int_0^t p_{n_\zeta n'_\zeta}\, p_{n_\xi n'_\xi}\, w^{1:\zeta}_{m_\zeta:n_\zeta}(t-x)\, \rho^{eff}_{\zeta:\xi:n_\zeta:n_\xi}(x)\, dx \tag{18a}$$

$$\tau^{1:r}_{m_\zeta:n_\zeta}(t) = g^{1:r}_{m_\zeta:n_\zeta}(t) = \lambda_o t \exp(-\lambda_o t) \tag{18b}$$

$$\psi^{l:\zeta}_{m_\zeta+1:n_\zeta}(t) = w^{l:\zeta}_{m_\zeta+1:n_\zeta}(t) = \lambda_\zeta t \exp(-\lambda_\zeta t) \qquad (18c)$$

The probability $P[X(t)=(a_1, m_1 \ldots m_\nu, n_1 \ldots n_\nu)]$ may be expressed as the total amount of probabilistic flow which the state $(a_1, m_1 \ldots m_\nu$ $n_1 \ldots n_\nu)$ has received in $[0,t)$ reduced by the amount of flow that went out of the state. That is,

$$P(X(t)=(a_1, m_1 \ldots m_\nu, n_1 \ldots n_\nu)) = \int_0^t \Big[\sum_{l=1}^{L} \Big(\sum_{r=1}^{\mu} \sum_{s=1}^{\mu} p^{l:r:s}_{m_\zeta:n_\zeta}(t) +$$

$$+ \sum_{n'_\zeta=1}^{K} \sum_{r=1}^{\mu} \sum_{\zeta=1}^{\nu} \gamma^{l:r:\zeta}_{m_\zeta:n'_\zeta}(t) + \sum_{\zeta=1}^{\nu} \sum_{r=1}^{\mu} \sum_{n'_\zeta=0}^{K} \varphi^{l:\zeta:r}_{m_\zeta:n'_\zeta}(t) +$$

$$+ \sum_{\zeta=1}^{\nu} \sum_{\xi=1}^{\nu} \sum_{n'_\zeta=0}^{K} \sum_{n'_\xi=0}^{K} f^{l:\zeta:\xi}_{m_\zeta:n_\zeta}(t) - 2 \sum_{r=1}^{\mu} g^{l:r}_{m_\zeta:n_\zeta}(t) -$$

$$- 2 \sum_{\zeta=1}^{\nu} \sum_{n'_\zeta=0}^{K} w^{l:\zeta}_{m_\zeta+1:n'_\zeta}(t) \Big] dt \qquad (19)$$

In the next section a numerical procedure is presented for calculating this state probability and hence the availability index of equation (3).

4. COMPUTATIONAL PROCEDURE

Thr Laguerre transform[1] provides an algorithmic framework for the evaluation of multiple convolutions. The transform employs the Laguerre functions to map a square integrable function into a discrete sequence of rapidly decreasing coefficients. For example, if we assume that $a(x)$ is a rapidly decreasing function which is square integrable, then this function has the Fourrier-Laguerre series expansion:

$$a(x) = \sum_{n=0}^{\infty} a_n^+ l_n(x) \quad , \quad a_n^+ = \int_0^\infty a(x) l_n(x) \qquad (20)$$

where

$$a_n^+ = \int_0^\infty a(x) l_n(x) \qquad (21)$$

$l_n(x)$ is the Laguerre function[7, pp 584-586] given by :

L
$$l_n(x) = \exp(-x/2)\, L_n(x) \tag{22}$$

n=0,1,2 ... and $L_n(x)$ is the Laguerre polynomial:

$$L_n(x) = \exp(+x)/n! \,+(d/dx)^n x^n \exp(-x) \tag{23}$$

For the convolution $c(x)= \int_0^x a(x-y)b(y)dy$, where a(x) and b(x) are square integrable functions , rapidly decreasing it is:

$$\tilde{c}_n = \sum_{j=0}^{n} \tilde{a}_{n-j} . \tilde{b}_j \tag{24}$$

where $\tilde{c}_n$, $\tilde{\alpha}_n$, and $\tilde{b}_n$ are the Laguerre sharp coefficients and related to the coefficients of the Fourier-Laguerre series expansion by the relationships:

$$c_n^+ = \sum_{m=0}^{n} \tilde{c}_m, \quad a_n^+ = \sum_{m=0}^{n} \tilde{a}_m , \quad b_m^+ = \sum_{m=0}^{n} \tilde{b}_m \tag{25}$$

The Laguerre sharp coefficients may be evaluated from the relationship

$$\sum_{n=0}^{\infty} \tilde{a}_n u^n = \ddot{a}((1+u)/2(1-u)) \tag{26}$$

where $\ddot{a}(\frac{1+u}{2(1+u)})$) is the function derived if in the Laplace Transform of a(x) we let $s=\frac{1+u}{2(1-u)}$

The truncation point of the series representation may be provided by the

$$\int_0^{\infty} x^i a(x)dx = 4^i \sum_{n=0}^{\infty} (-1)^n n^i \tilde{a}_n , \qquad 0 \leqslant i \leqslant 2 \tag{27}$$

when the moment values are known or may be computed. Experimental observations[1] suggest that if the truncation point is chosen to satisfy a given accuracy for the moment value, then the series representation of the a(x) function also satisfies the same accuracy.
If we let

$$\bar{W}(x) = \int_x^{\infty} w(y)dy \tag{28}$$

$$\text{then} \qquad \tilde{w}_n = -2\tilde{w}_n + 4 \sum_{m=0}^{\infty} (-1)^m \tilde{w}_{n+m} \tag{29}$$

An efficient algorithm evaluating the state probabilities of the previous section may be derived if the mathematical formulae just described, are utilized. This algorithm evaluates first the sharp Laguerre coefficients of the following probability distribution densities:

$$g^{l:r}_{m_\zeta:n_\zeta}(x),\ h(x),\ w^{l:\zeta}_{m_\zeta:n_\zeta}(x),\ {}^{\bullet}a_{\zeta:n_\zeta}(x) \tag{29a}$$

$${}^{1}h(x)=\lambda_o x\exp(-\lambda_o x)\,h(x),\quad {}^{1}a_{\zeta:n_\zeta}(x)=\lambda_o x\exp(-\lambda_o x)\,a_{\zeta:n_\zeta}(x), \tag{29b}$$

$${}^{2}a_{\zeta:n_\zeta}(x)=\lambda_\zeta x\exp(-\lambda_\zeta x)\,a_{\zeta:n_\zeta}(x),\quad {}^{3}a^{\xi:n_\xi}_{\zeta:n_\zeta}(x)=\lambda_\zeta x\exp(-\lambda_\zeta x)\,a_{\xi:n_\xi}(x)$$

This is achieved by applying iteratively equation (26) At each iteration the accuracy of the first moment of the distribution densities is checked and if it is found less or equal to 1% the evaluation process of the sharp Laguerre coefficients is terminated.Of course, it is assumed that the integral of equation (27) can be evaluated analytically. Next, the sharp Laguerre coefficients of the state transition probabilities probabilities, defined by the relationships (16) to (18c) and the effective repair time densities,satisfying equations (11) to (15), are easily computed by applying equation (24).
The evaluation of the state probability $P[X(t)=(a_1,m_1 \ldots m_\nu,n_1, \ldots n_\nu)]$ is not made with the same method because this function may not be be square integrable. However, if we consider the derivative of the state probability, denoted by W(t), according to equation (19) it will be Laguerre transformable and have:

$$\tilde{w}_n=\sum_{l=1}^{L}\Big[\sum_{r=1}^{\mu}\sum_{s=1}^{\mu}\tilde{p}^{l:r:s}_{m_\zeta:n_\zeta:n}+\sum_{n'_\zeta=0}^{K}\sum_{r=1}^{\mu}\sum_{\zeta=1}^{\nu}\tilde{\gamma}^{l:r:\zeta}_{m_\zeta:n'_\zeta:n}+\sum_{\zeta=1}^{\nu}\sum_{r=1}^{\mu}\sum_{n'_\zeta=0}^{K}\tilde{\varphi}^{l:\zeta:r}_{m_\zeta:n'_\zeta:n}+$$

$$+\sum_{\zeta=1}^{\nu}\sum_{\xi=1}^{\nu}\sum_{n_\zeta=0}^{K}\sum_{n_\xi=0}^{K}\tilde{f}^{l:\zeta:\xi}_{m_\zeta:n_\zeta:n}-2\sum_{r=1}^{\mu}\tilde{g}^{l:r}_{m_\zeta:n_\zeta:n}-2\sum_{\zeta=1}^{\nu}\sum_{n_\zeta=0}^{K}\tilde{w}^{l:\zeta}_{m_\zeta+1:n_\zeta:n} \tag{30}$$

It should, however, be noted that $p(s)=-\frac{1}{s}\,w(s)+p(0)$ where p(s) and w(s) are the Laplace transforms of the state probability function and its derivative.; $p(0)=\prod_{i\leq\nu}\delta_{m_i,0}\cdot b_{n_i}$, $\delta_{m_i,0}=1$ when $m_i=0$ $\delta_{m_i,0}=0$ $\quad m_i\neq 0$,

Then according to equation (29) the state probability has the following Fourier-Laguerre series expansion:

$$P(X(t)=(a_l,m_1\ldots m_\nu,n_1\ldots n_\nu))=\sum_{m=0}^{\infty}\sum_{n=0}^{m}\tilde{\tilde{w}}_n(1-l_n(t))+p(0) \tag{31}$$

The algorithm uses this last equation to evaluate the state probabilities when l=1,2, ... L . The truncation point of the series expansion of the state probability is initially selected equal to the truncation point of the probability distribution densities (29a) to (29b) and the state probability is computed for a period of time equal to the system useful lifetime. Using this value as a basis a final value of the state probability is computed recursively, satisfying the desired accuracy bound.

5. AN APPLICATION EXAMPLE

Let us consider two distributed control systems of the type described in section 1. The architecture of each one consists of the hardware and software modules shown in Figures 1 and 2 respectively. In Tables I and II information on the hardware modules required to run each software package is also provided. From these tables the states of the A set may be obtained. The statistical features of the various components of each system are given in Table III.

Using the previously discussed algorithmic procedure the state probabilities of the basic node of the two distributed control systems are evaluated. The resulting availability index of the two systems for various control loop functions are illustrated in Figure 3. Comparing the obtained curves one can conclude that the second system architecture presents better availability and cost characteristics for small number of loops. If, however, the number of loops is increased, i.e to 64, the cost and availability characteristics of the two architectures become practically the same. On the contrary, for a large number of loops i.e 128 the first architecture is superior.

6. CONCLUSIONS

In this paper stochastic relationships are developed for the evaluation of the availability of various architectural configurations of distributed control systems. These relationships are applicable when from past system operational experience one may consider that:
(1) within a short period of time it is very unlikely to occur more than one failure of a hardware module or a software package.
(2) The p.d.f of the failure rate, up-time and down-time of the various hardware and software components may be expressed by square integrable rapidly decreasing, time-dependent functions, the Laplace transform of which can be found analytically.
(3)At the beginning or after the repair of a software package the errors cannot exceed a finite, nonnegative and relatively small number.

The availability is considered to be the sum of all the state probabilities of a multivariate process corresponding to the system operational states. For the numerical evaluation of these probabilities a procedure is proposed using the Laguerre transform which simplifies the numerical computation of multiple convolution integrals. Since the required CPU time of the numerical procedure increases significantly when the number of software errors increases the method seems to be cost effective for a relatively small number of software errors.

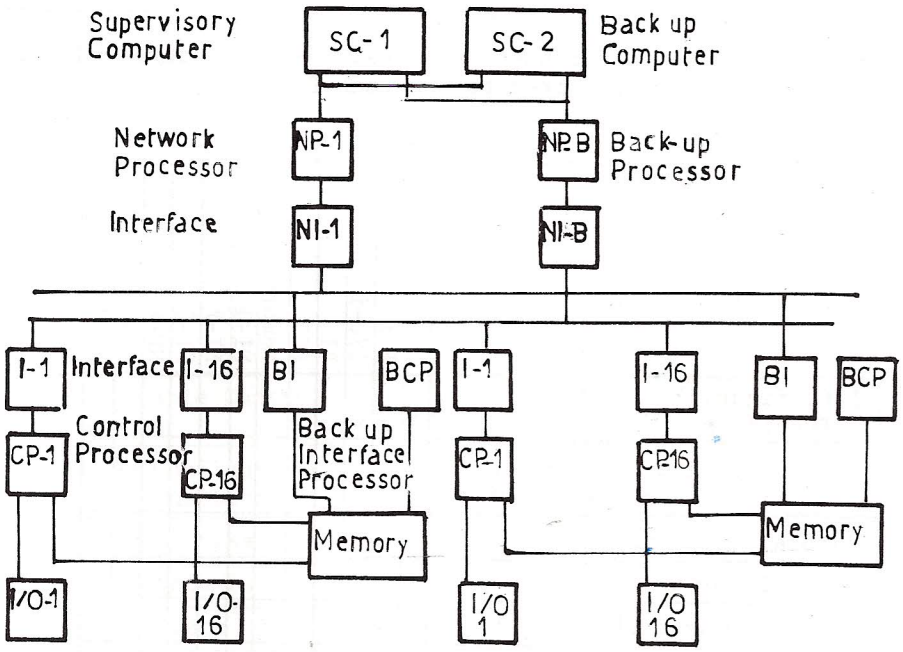

Figure 1: Architecture of the first system

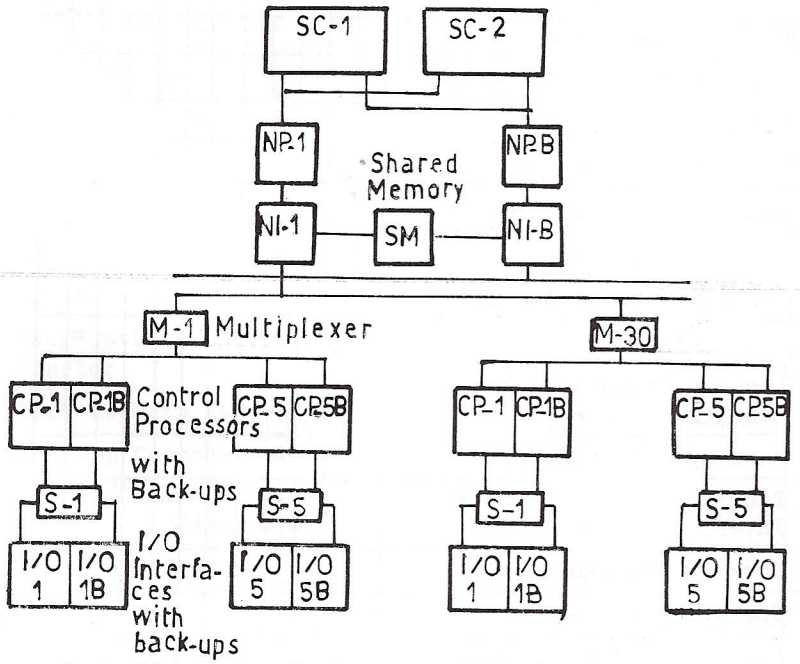

Figure 2: Architecture of the second system

TABLE I

Software Packages for the first System Architecture and the Hardware Modules required to run each Package

	Program Name	Hardware Module Name												
		CP-i	I-i	I/O-i	BI	BCP	Memory	NI-1	NI-B	NP-1	NP-B	SC-1	SC-2	
1	Configurable Control Functions	*		*										
2	Network Interface of Control Functions Circuits	*	*											
3	Local Data Base	*												
4	I/O Interface	*		*										
5	Fault Diagnostics					*	*							
6	Back-up Configurable Control Functions			*		*	*							
7	Back-up Network Interface of Control Functions Circuits				*	*	*							
8	Network Interface							*		*				
9	Back-up Network Interface								*		*			
10	Network Control									*				
11	Back-up Network Control										*			
12	Diagnostic Error Handler											*		
13	Data Base Manager											*		
14	Console Network Interface											*		
15	Configurator											*		
16	Back-up Diagnostic Error Handler												*	
17	Back-up Data Manager												*	
18	Back-up Console Network Interface												*	
19	Back-up Configurator												*	
20	Operating System											*		
21	Back-up Operating System											*		

TABLE II

Software Packages for the second Systen Architecture and the Hardware Modules required to run each Package

	Hardware Module Name / 0 Program Name	CP-i	CP-iB	S-i	I/O-i	I/O-iB	M-j	NI-1	SM	NI-B	NP-1	NP-B	SC-1	SC-2
1	Configurable Control Functions	*												
2	Back-up Configurable Control Functions		*											
3	Local Data Base	*							*					
4	Back-up Data Base		*											
5	I/O Interface	*		*	*									
6	Back-up I/O Interface		*	*	*									
7	Diagnostics and Back-up Transfer		*	*										
8	Network Interface						*	*						
9	Back-up Network Interface						*			*				
10	Network Diagnostics and Back-up Transfer								*			*		
11	Dignostic Error Handler												*	
12	Data Base Manager												*	
13	Console Network Interface												*	
14	Configurator												*	
15	Back-up Diagnostics Error Handler													*
16	Back-up Data Base Manager													*
17	Back-up Console Network Interface													*
18	Back-up Configurator													*
19	Operating System													*
20	Back-up Operating System													*
21	Network Control								*		*			
22	Back-up Network Control									*		*		

TABLE III
Features of the considered Fault Tolerant Distributed Control Systems

$g^{j:r}_{m_\zeta:n_\zeta}(x) = w^{j:\zeta}_{m_\zeta:n_\zeta}(x)$	$0.5\,x\,\exp(-0.5\,x)$
$h(x)$	$\exp(-0.1x)$
K	4
$q_{\zeta:n_\zeta}(x)$, $1 \leq \zeta \leq v$, $0 \leq n_\zeta \leq K$	0, $n_\zeta = 0$ $((x^{4-n_\zeta})/(4-n_\zeta)!)\exp(-x))$
$a_{\zeta:n_\zeta}(x)$ $1 \leq \zeta \leq v$ $0 \leq n_\zeta \leq K$	0, $n_\zeta = 0$ $\frac{\gamma_{n_\zeta}\cdot\beta_{n_\zeta}}{\gamma_{n_\zeta}-\beta_{n_\zeta}}(\exp(-\gamma_{n_\zeta}x)-\exp(-\beta_{n_\zeta}x))$
$P_\zeta = (p_{n_\zeta n'_\zeta})$ $1 \leq \zeta \leq v$, $0 \leq n_\zeta \leq 4$	1.000 0.000 0.000 0.000 0.000 0.700 0.150 0.100 0.030 0.020 0.300 0.425 0.150 0.100 0.025 0.125 0.225 0.400 0.150 0.100 0.075 0.125 0.200 0.400 0.200
$\underline{b}^{(i)T}$ $1 \leq i \leq v$	0 0 0 0 1

Note: Both systems are considered to have the same statistical features

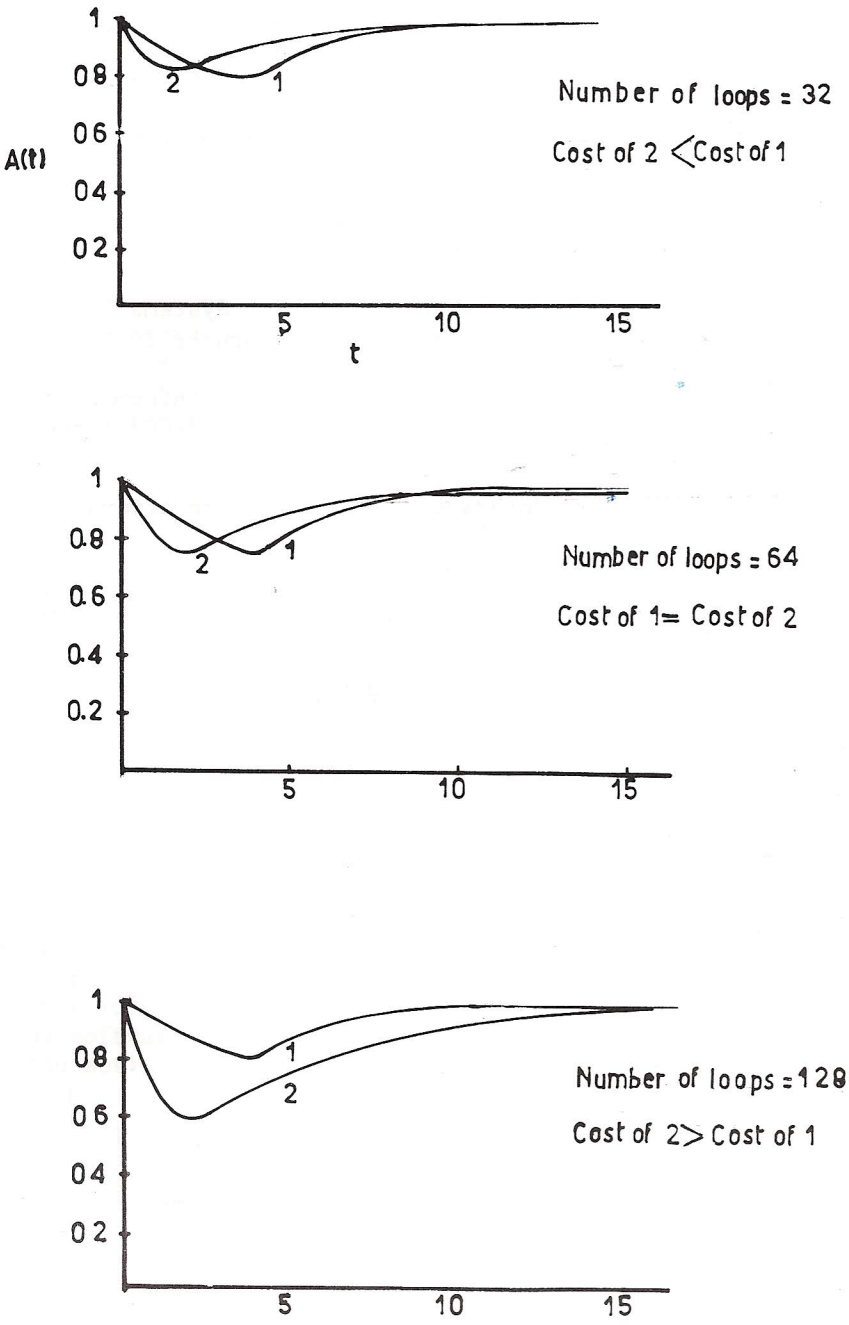

Figure 3: Availability curves

REFERENCES

[1] Ushio Sumita and Yasushi Masuda, ' Analysis of Software Availability/Reliability under the Influence of Hardware Failures', IEEE Trans. on Software Eng., SE-12, 1, January 1986, 32-41

[2] Foxboro Co., ' Spec-200 Micro Control System Overview,' TI 280-100 1984

[3] Honeywell Inc., ' TDC 2000 System Overview ,' Reference SY-10-01, 1979

[4] James D. Schoeffler, ' Distributed Computer Systems for Industrial Process Control,' IEEE Computer, 17, 2, February 1984, 11-18

[5] Combustion Eng., ' Taylor Mod 300 Process and Information System,' Dependability and Maintenance Concepts,' SDS-32E008 Issue 1, September 1985

[6] Jim Machulda, ' Fault Tolerant Control: You Can Afford it,' InTech, 32, 9, September 1985

[7] V.I. Smirnov, A course on Higher Mathematics, Vol. 3, Part 2,Pergamon Press, 1964

[8] V.S. Pagachev, Theory of Random Functions, Pergamon Press, 1965