# MODELLING AND ANALYSIS OF SOFTWARE RESOURCES IN MODULAR SPC SWITCHING SYSTEMS - SOME ASPECTS OF DIMENSIONING AND OVERLOAD CONTROL

by P. Tran-Gia
Dept. of Communications, University of Siegen, W.-Germany

Contribution to the Fourth International Conference on Software Engineering for Telecommunication Switching Systems, Warwick, GB, July 1981

ABSTRACT

A queueing model for two software machines handling calls in tandem is presented. Software machines are components of stored program controlled (SPC) switching systems with modular software structure.

Dimensioning and overload control aspects are discussed and a call acceptance and regulation method, which increases the call completion rate of the switching system, is developed and investigated.

Numerical results are given for dimensioning purposes, whereby the blocking probabilities for call requests under different traffic conditions are determined. The calculation is based on a two-dimensional birth and death process under stationary conditions.

## 1. INTRODUCTION

The software structure of modern switching systems must meet a wide range of requirements, of which the modularity is often the most essential. To provide this modularity feature, a typical switching system software is usually decomposed into functional modules or *software machines* (SM's), e.g. for call control, signalling, device control, metering, maintenance etc..(c.f.fig.1). The software machines can be located in one processor as in the case of a single processor system or in different processors as in decentralized multiprocessor systems.

Each software machine consists of a program part and an individual data part. The program part contains the function-dependent programs and the data part is reserved for permanent and semipermanent system data, which are required to perform the SM's function, e.g. call oriented data.

The data part of a software machine has also a modular structure and is divided into units, namely *call control blocks* (CCB's), which store necessary data for a call in its active period, e.g. state of the call, expected event, subscriber identity etc..

Communication between SM's is only allowed in the form of messages interchanged between them, even for data transfer. At the lowest level, subcall events are messages which are interchanged by a message (or subcall) handler via a logical bus.

As a consequence, the programs of a particular software machine can only access data of its own CCB's. This has the advantage that incorrect data access by other SM's is prevented and the separation of data is ensured. On the other hand, certain call oriented data must be stored several times in different SM's and data transfer for updating purposes must be provided.

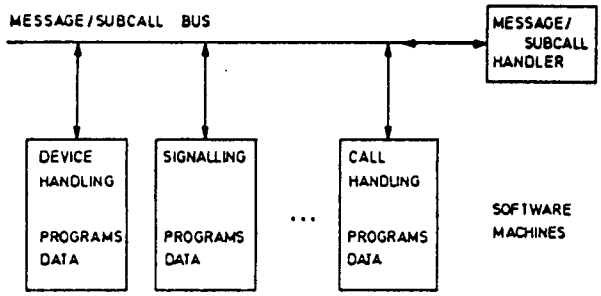



*Fig.1 Modular software structure for SPC switching systems*

A call request may be dealt with by different SM's in parallel or in tandem. The activated software machine seizes one of its CCB's during the corresponding active period to store actual data for the call request.

Calls can only be completed if all CCB seizures have been done successfully (c.f.fig.2). If a call transferral is blocked, an ineffective processor occupancy must be registered and the call completion rate is less and secondary blocking is said to have occurred.

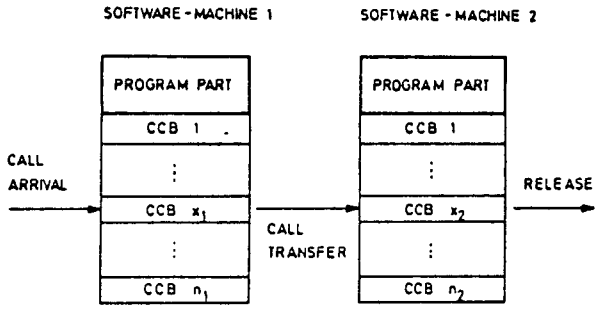



*Fig.2 Call-acceptance and call-transfer in two software machines serving calls in tandem*

Therefore, the dimensioning of the number of CCB's in software machines and the CCB seizing scheme will be important considerations influencing the secondary blocking effect, the call completion rate and further the real time performance of a modular switching system.

The aims of the paper are :

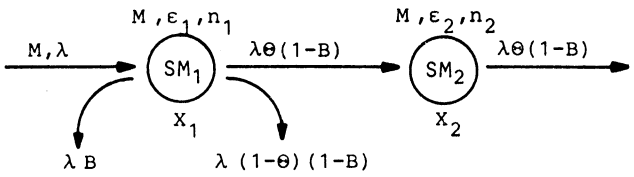
- To investigate the dimensioning problem of finding the required number of call control blocks, by modelling two software machines serving calls in tandem.

- To analyse the call blocking effect in the case of a two-machine system, where a call acceptance regulation method, which avoids the secondary call blocking effect and optimizes the call completion rate, is assumed.
- To define and estimate control mechanisms for overload situations by modifying the model used in the above.

## 2. MODELLING

Fig.3 shows a basic model which describes the call acceptance and transfer mechanism for two software machines serving call requests in tandem.

This model differs from a multistage many-server blocking system on one essential point, that the call acceptance condition depends not only on the actual state of the first server group ($SM_1$), but also on the state of the whole system, as described in the equations (2.1) and (2.2) below.




$n_i$ *number of servers in the* $i^{th}$ *SM*
$X_i$ *number of busy servers in the* $i^{th}$ *SM*
$\frac{1}{\varepsilon_i}$ *mean service time in the* $i^{th}$ *SM*

*Fig. 3 Model of two software machines controlling calls in tandem*

$SM_1$ could be considered as the call set-up software machine for handling normal calls and also special facility calls. After the set-up phase, a call will be transferred with transfer probability $\Theta$ to the second software machine, which could be thought of as the call control module performing call control tasks during the conversation phase.

The complementary probability 1-$\Theta$ models calls with incompleted dialing, unsuccessful calls etc..

The holding time of a call in the first SM depends on the number of messages or subcalls produced per call and their response time. Considering all call types with their diverse numbers of subcalls, the holding time (or service time) in the call set-up phase is approximated to be negative exponentially distributed. This is also valid for the conversation phase modelled by the second software machine.

Therefore, the arrival and service processes in this model are considered under the following assumptions:

- Call arrivals constitute a Poisson process with rate $\lambda$.
- The service time of individual calls in $SM_1$ and $SM_2$ are independent, exponentially distributed random variables with mean $1/\varepsilon_1$ in the first and $1/\varepsilon_2$ in the second software machine.

In Fig. 3, the notation M indicates that the arrival and service processes are of the Markov type. The maximum number of CCB's is $n_1$ and $n_2$ for $SM_1$ and $SM_2$, respectively.

In this paper, the following method for the call acceptance regulation is proposed; it ensures that a call transferring from the first SM always finds a free CCB in the second SM so that no secondary call blocking can occur:

a) $x_1 \leq n_1$ ... (2.1)

limitation of calls in $SM_1$ due to capacity $n_1$

b) $x_1 + x_2 \leq n_2$ ... (2.2)

limiting number of calls in both SM's

Condition b) corresponds to the reservation of a secondary CCB at call initialization.

The call acceptance mechanism described above will increase the call completion rate because calls that have entered the system have a higher priority to seize system resources.

## 3. ANALYSIS

### 3.1 State probabilities

The model described above can be investigated as a two-dimensional birth and death process using standard techniques of queueing theory [6].

The following symbols are used :

| | |
|---|---|
| $\lambda$ | arrival rate for calls |
| $\varepsilon_i$ | service rate for $SM_i$ |
| $\Theta$ | transfer probability |
| $P(x_1,x_2)$ | state probability for an arbitrary instant to have $x_1$ and $x_2$ active calls in $SM_1$ and $SM_2$ respectively. |

Since the input and service processes have the Markov property, the state probabilities are described by the following set of Chapman-Kolmogorov difference equations in the steady state :

$$P(x_1,x_2).(\lambda+\varepsilon_1 x_1+\varepsilon_2 x_2) = \lambda P(x_1-1,x_2)$$
$$+(1-\Theta)(x_1+1)\varepsilon_1 P(x_1+1,x_2)$$
$$+\Theta(x_1+1)\varepsilon_1 P(x_1+1,x_2-1)+(x_2+1)\varepsilon_2 P(x_1,x_2+1)$$
$$(x_1<n_1;\ x_1+x_2<n_2;\ P(x_1,-1)=P(-1,x_2)=0)$$

$$P(n_1,x_2).(\varepsilon_1 n_1+\varepsilon_2 x_2) =$$
$$\lambda P(n_1-1,x_2) + (x_2+1)\varepsilon_2 P(n_1,x_2+1)$$
$$(x_2<n_2-n_1)$$

$$P(x_1,x_2).(\varepsilon_1 x_1+\varepsilon_2 x_2) =$$
$$\lambda P(x_1-1,x_2) + \Theta(x_1+1)\varepsilon_1 P(x_1+1,x_2-1)$$
$$(x_1 \leq n_1\ ;\ x_1+x_2=n_2\ ;\ P(-1,x_2)=0\ )$$

....(3.1)

The total number of equations is

$$k = (n_1+1)(n_2 - \frac{n}{2}1 + 1) \qquad ...(3.2)$$

This set of equations corresponds to the state transition diagram shown in Fig.4. The equations are unsymmetrical in $x_1$ and $x_2$ and it does not seem possible to find a closed form for the probability generating function.

The state probabilities are obtained numerically from equation (3.1) using a relaxation method, whereby the normalization equation

$$\sum_{x_1=0}^{n_1} \sum_{x_2=0}^{n_2-x_1} P(x_1,x_2) = 1 \qquad ...(3.3)$$

is considered.

These probabilities form the basic requirement for the calculation of the call blocking probability under stationary conditions.

## 3.2 Call blocking probability

According to the call acceptance conditions (2.1) and (2.2), the blocking probability B for call requests is given by :

$$B = \sum_{x_2=0}^{n_2-n_1-1} P(n_1,x_2) + \sum_{x_1=0}^{n_1} P(x_1,n_2-x_1) \qquad ...(3.4)$$

This blocking probability is composed of two terms ; the first is due to the condition (2.1), the second term relates to condition (2.2).

A close approximation using product form solution has also been investigated. However, such approximation is only applicable for a limited range of parameters, and shall not be discussed further in this paper.

## 4. RESULTS AND APPLICATIONS

### 4.1 Sizing of number of call control blocks

In this section, several examples are calculated to show how the blocking probability for call requests is influenced by the offered traffic (standardized by $A_2/n_2 = \lambda/\varepsilon_2 n_2$), the number of CCB's ($n_1$ for $SM_1$ and $n_2$ for $SM_2$) and the transfer probability $\theta$, where the call acceptance method in (2.1) and (2.2) is assumed. The number $n_2$ of CCB's in $SM_2$ is fixed.

Numerical results are given for different numbers of CCB's and incoming traffic conditions. Fig. 5 shows the behaviour of the call blocking probability B as a function of the offered traffic.

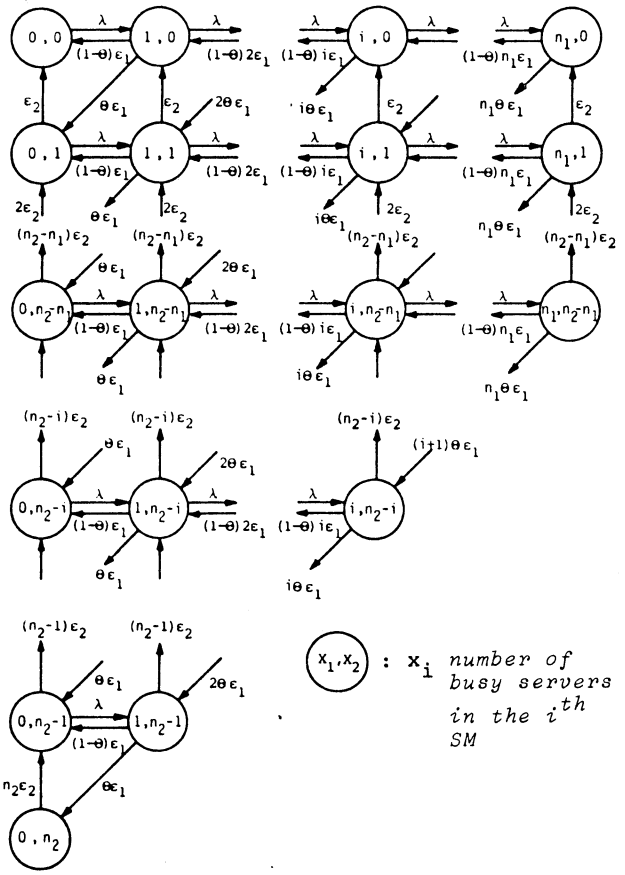



*Fig.4 State Transition Diagram*

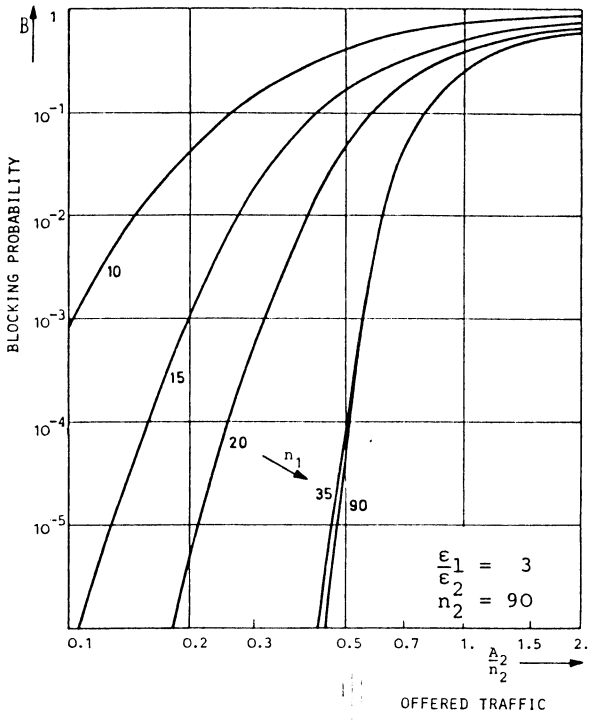



*Fig.5 Call blocking probability B vs offered traffic*

The curves show that above a certain value of $n_1$, the effect of condition (2.1) will diminish compared with the effect of condition (2.2). This is further illustrated by Fig. 6, where the dependence of call blocking probability B on the number $n_1$ of CCB's in $SM_1$ is shown. Above the value $n_1$ = 40, B can not be further reduced by increasing $n_1$ and is only affected by $A_2/n_2$.

For large $n_1$ and $n_2$, a proportionality has been observed between the ratio of holding times ($\varepsilon_1/\varepsilon_2 = A_2/A_1$) and the location of the breakpoint of the call blocking probability B.

Taking into account in addition the influence of the transfer probability $\Theta$, figure 7 shows the blocking probability as a function of $n_1$. For dimensioning purposes, this figure can give the required number $n_1$ in $SM_1$ for a realistic range of the transfer probability $\Theta$.

## 4.2 Optimal CCB allocation for two SM's serving calls in tandem

If a fixed total number n of CCB's (i.e. a fixed memory range) is reserved for the two SM's (see fig. 2), the blocking probability B of call attempts is strongly dependent on the values chosen for $n_1$ and $n_2$ ($n_1+n_2 = n$ ).

Fig.8 shows that a minimum of B can be found for different traffic conditions. This minimum varies according to the ratio of traffic $A_2/A_1 = \varepsilon_1/\varepsilon_2$ and the transfer probability $\Theta$.

The sensitivity of this minimum is more strongly dependent upon the values of $n_1$, $n_2$ and the transfer probability $\Theta$ than upon the incoming traffic intensity.

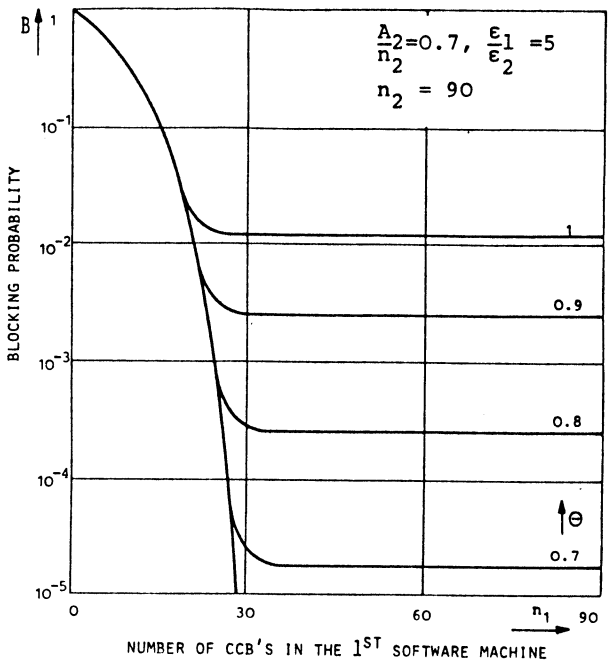



*Fig.7 Call blocking probability B vs number $n_1$ of CCB's in the first SM*

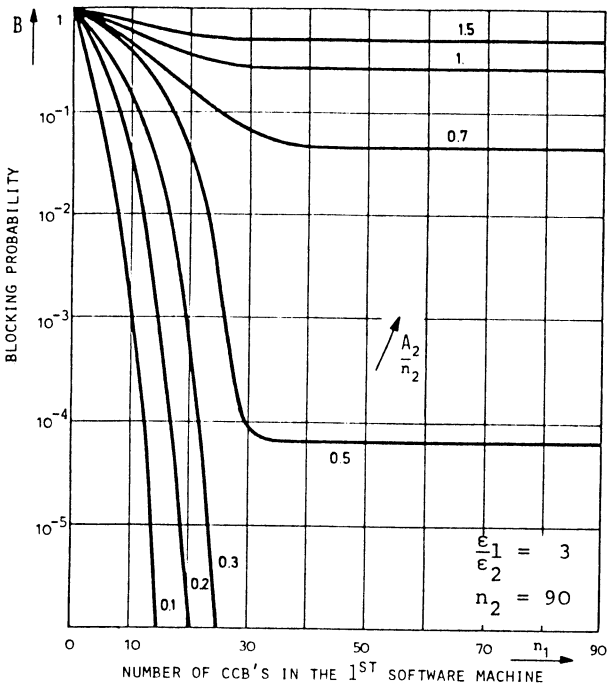



*Fig.6 Call blocking probability B vs number $n_1$ of CCB's in the first SM*

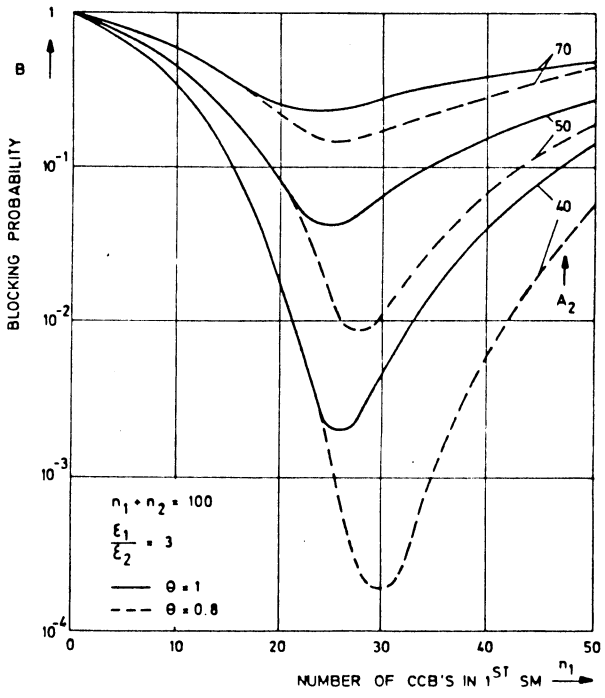



*Fig.8 Call blocking probability B vs number $n_1$ of CCB's in the first SM*

### 4.3 CCB management in overload situations

Overload may be caused by statistical fluctuations, seasonal effects, loss of resources through breakdown etc..

To guarantee proper system functioning in short-term overload situations, various strategies for congestion control can be used e.g. controlling the call acceptance rate, dynamic provisioning of additional resources, changing of schedules, etc..

In terms of the above basic resource model, an overload control strategy which reduces the blocking probability B has been investigated and quantitatively analyzed. This method, described below, can be implemented on condition that the two considered software machines are controlled by the same processor. This allows a short term changing of $n_1$ and $n_2$, which are considered as semipermanent data.

By this strategy, calls may be accepted even when there is no CCB available in the second SM. After the first phase, a call keeps its CCB in the first module for further call control tasks performed by the second SM.

This strategy can only be applied if both software machines are located and controlled in the same processor.

Using this overload control method the call acceptance conditions (2.1) and 2.2) are replaced by the following conditions

a) $x_1 \leq n_1$ (4.1)

b) $x_1 + x_2 \leq n_0$ (4.2)

where $n_2 \leq n_0 \leq n_1 + n_2$

$n_0$ is the maximum number of calls allowed in both SM's in the overload period.

Fig. 9 shows how the call blocking probability B can be further reduced by increasing $n_0$.

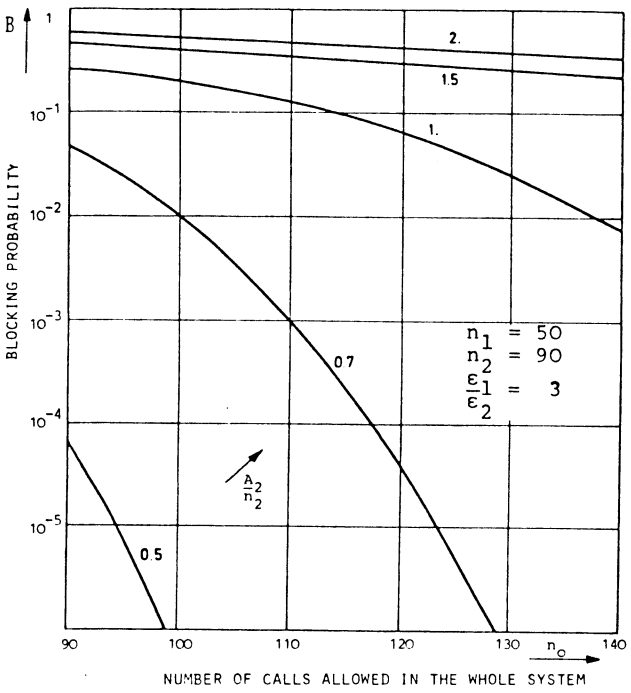



*Fig.9 Call blocking probability B vs number $n_0$ of calls allowed in the whole system*

The consequences of this strategy are

- Minimizing the influence of repeated call attempts and reduction of ineffective processor load
- Reduction of data transfer via messages between software machines in overload situations (Under normal conditions, data transfer is necessary if a new CCB is seized in the second SM).

The improvement of resource utilization is obtained at the cost of short-term controlled violation of the SM's data separation.

## 5. CONCLUSION

The numerical results given in section 4 relate to a basic queueing model for two software machines handling calls in tandem with a call acceptance and regulation method.

This provides a possibility to dimension the number of call control blocks in stored program controlled switching systems with modular software structure.

This basic model can be extended to describe several software machines operating in tandem with other interfaces, e.g. in the case where the call set-up SM is subdivided into incoming and outgoing signalling parts, and originating and terminating call control parts.

Finally, an overload control method which increases the call completion rate has been presented, for which results for call blocking probability are provided.

## ACKNOWLEDGEMENTS

The author would like to thank Professor P.J. Kuehn for helpful discussions and for his continued interest in this work . The programming efforts of Mr. W. Fischer are greatly appreciated.

## REFERENCES

[1] Arima,T.; Kobayashi,H.; Makino,H.; Aso,T. " A new switching processing architecture and its operating system for a distributed hierarchy software structure ", Int. switching symposium record, Paris 1979, pp. 943-947.

[2] Gardiner,A.; Katzschner,L.; Van der Straeten,C., " Software design for digital exchanges ", El.Comm., Vol.54, No 3, 1979, pp. 199-204.

[3] Carruet,V.; Rideau,A.; " Software structure and methodology ", El.Comm., Vol.54, No 3, 1979, pp. 178-185.

[4] Lawson,D.A., " Software architecture ", El.Comm., Vol.54, No 3, 1979, pp.225-230.

[5] Botsch,D., " Die EWSD-Software - Ein Realzeitprogrammsystem in der höheren Programmiersprache CHILL ", Telcom report 2(1979), pp. 184-189.

[6] Cooper,R.B., " Introduction to queueing theory ", Macmillan Co., NewYork 1972.