# DEVELOPMENT AND MAINTENANCE DYNAMICS OF A SOFTWARE BUSINESS FIRM

**Pratap K. J. Mohapatra**

**Department of Industrial Engineering and Management**
**Indian Institute of Technology, Kharagpur**
**Kharagpur - 721 302**

## Absract

This paper demonstrates, in a system dynamics framework, the effect maintenance activities have on development of new software. Buoyant demand, rising backlog of work, increased delivery delay, pressure to increase programming productivity, and consequent rise in the number of defects in the software result in a fall of quality image of the company in the eye of its customers. Falling delivery delay and quality images can reverse the trend of orders. Added to this problem is the fact that the software industry today is experiencing large employee turnover. Hiring new programmers affect both productivity and quality of products. The simulation results display cyclic fluctuation of business over a 12-year period. The policy tests indicate that the policy of hiring experienced staff and of being able to retain them for a considerable length of time gives the best system behavior compared to the other policies.

## Introduction

The cost of software maintenance is increasing steadily during the past twenty years. Whereas it was between 35% and 45% during the 1970's, it is touching 80% now. Jones (1986) and Pressman (1992) voice concern for such increasing maintenance activities in the information system organizations. Jones speaks of a cyclic wave of high development activity followed by high maintenance activity. Pressman is of the opinion that the intangible costs comprise of lost opportunity to develop new products, customer dissatisfaction, and decreased productivity.

In this paper we have considered the case of a software consultancy firm which has been generally doing well, but, which is experiencing, in recent times, a slow growth in its business associated with a lot of fluctuations.

## The Software Business Firm

The firm does not wish to go through the trauma of ups and downs of business. Bulk of the new orders, the firm receives every month, used to be from its old customers. Of late, however, many of these customers are turning to its competitors.

The firm initially thought that its high price quotations are responsible for such a phenomenon. Many in the firm however believe that the price differential is rather small. They think that customers give much more importance to the quality of the product.

The firm rigorously follows the procedure of quality assurance and testing and is generally proud of its high quality product. Additionally, it attempts to promptly attend to field errors whenever they are reported to it, particularly during the stage of installation of the software. In the process, many times they have to call back their programmers from newly assigned development works and ask them to attend to the reported errors.

The firm has noticed, on average, that about 12%-15% of its programmer force have to direct their time and energy to correcting errors that surface during installation and that about 10 errors are fixed by a person in a month.

The firm also receives customer complaints from time to time but rarely attends to them unless they are under maintenance contract.

The firm has found that an employee stays with it for, on overage, only five years. While recruiting new programmers, it considers the work load vis-a-vis the current level of programmers. It puts every programmer in training before (s)he is asked to actively participate in the development task. These new programmers, however, are less productive than the experienced ones and quite often take the help of the experienced programmers during the development process.

The firm has observed that in general, customers, particularly new customers, want to get their products in time. Some in the firm believe that late deliveries may have driven away some customers to its competitors.

## Conceptualizing the Problem

The problem faced by the firm is one of fluctuation in its business which is otherwise growing. Although the causes of this problem are not obvious, one wonders whether the following factors may be contributing to the occurrence of this problem:

The company may not be recruiting software personnel at the desired rate. Whenever it has more number of software personnel relative to the business it has, it does a good job both in terms of quality and delivery schedule; but falters when it has considerably high business than it can handle with its software personnel. In case of the former, the market image of the firm shoots up and it is likely to get more and more customer orders. In case of the latter, however, delivery schedules are not met and quality of the software suffers, leading to loss of goodwill among the potential customers and to loss of potential sale.

With reference to the discussion made above, two factors appear to be important in the context of the problem faced by the firm. When the firm is having a good time with a lot of business with it than it can handle with its programmer force, schedules may overrun, and quality of its product may suffer resulting in quite a considerable number of programmers spending their time in correcting errors than getting engaged in new developments. This worsens the delivery delay position of the firm. This can adversely the new customer orders.

The second factor that can precipitate the problem of the firm is that if many new programmers are around then the overall programmer productivity will be lowered due to not only their low productivity but also due to communication and on-the-job training overheads because experienced programmers have to spend some time on these new programmers to familiarize them to the new environment.

We now make our hypotheses more explicit with the help of graphic tools. Figure 1 shows how the Firm, the Market, and the Competitors interact in the context of the problem being faced by the firm. The Market places orders with the company and receives the software, when ready, from the Firm. While deciding the software orders to be placed with the firm, the Market considers the delivery delay position of the Firm, the quality of software developed by the Firm, and the delivery delay and software quality images of the Competitors.

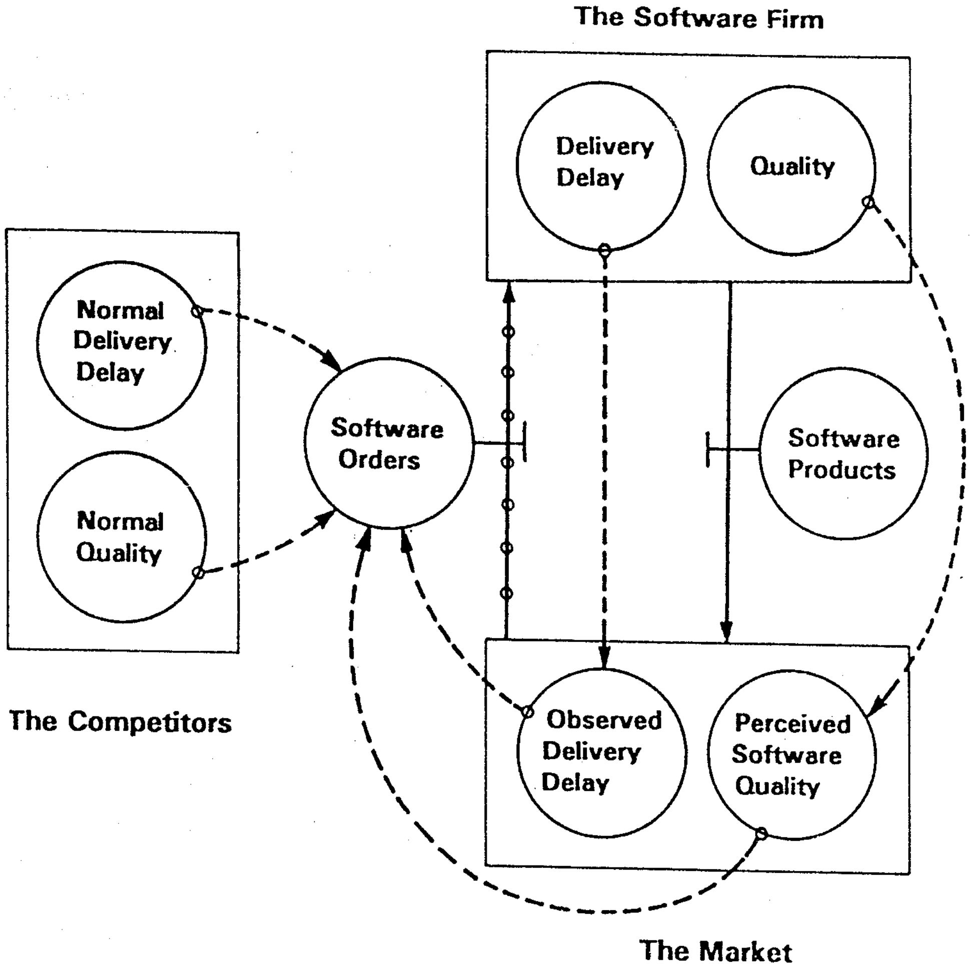


**Fig. 1 The Firm-Market-Competitors Interactions**

Figure 2 is the sectoral overview diagram that shows, in some detail, the sectoral interactions within the Firm and the interactions of these sectors with the Market. The following four sectors are considered:

*The Firm:*

(1) Human Resource, (2) Development & Delivery, and (3) Quality

*The Market:* (4) Order Formation.

As before, the market uses information on delivery delay and quality of software before placing orders with the Firm. Within the Firm, schedule overruns can affect quality and prompt the firm management to hire additional manpower. As new programmers join, the development and delivery may be delayed for some time before they pick up.

## The Model

As visualized in the last section, the two physical flows taking place in this situation are (1) The flow of Software Orders, and (2) The flow of Programmers.

Figure 3 is a structure diagram for the flow of software orders. As software orders are

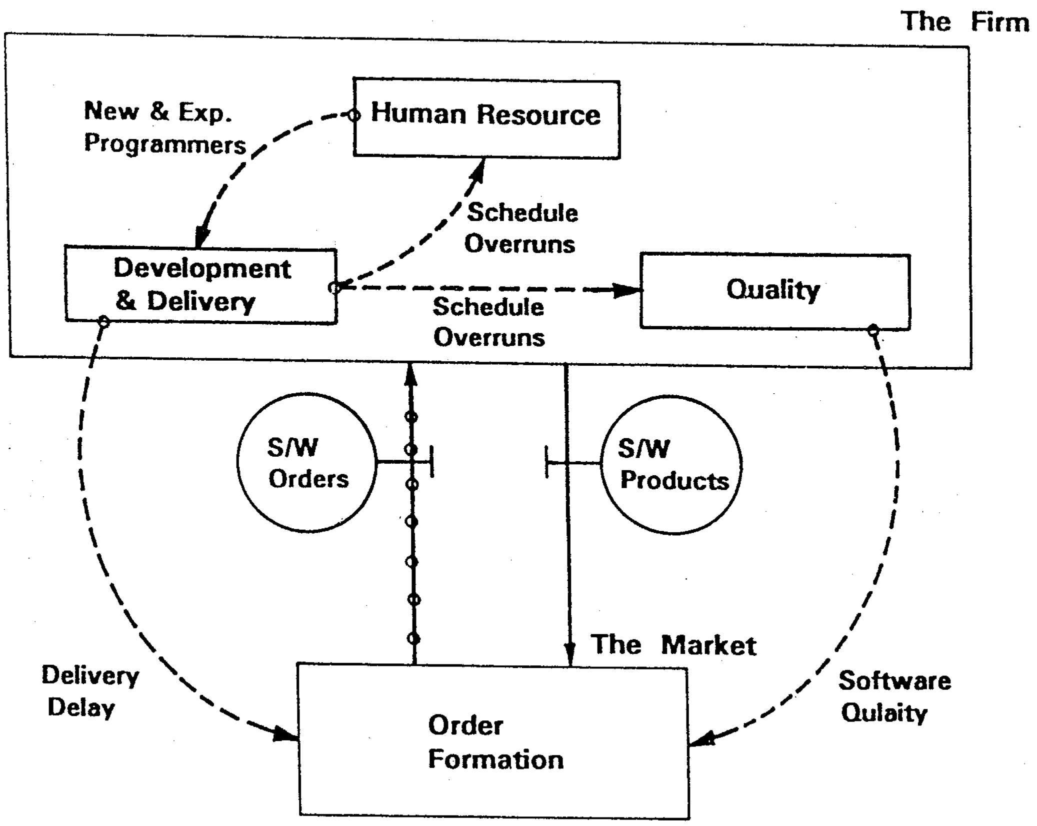


Fig. 2 The Sectoral Overview Diagram

received backlog of work increases. Subsequently, when deliveries are made, the size of the backlog reduces. It is assumed in this model that customer orders are decided on the basis of delivery delay position of the firm and the quality image of the firm in the market. Delivery of software is made as soon as they are developed according to the user specifications. It is assumed that the development rate of software depends on the number of new and experienced programmers, their productivities, and pressure on the programmers to maintain the schedule.

Figure 4 is a structure diagram for the flow of programmers. After programmers are recruited, they are put under training. After they complete the training programme, they are assigned to software development tasks. However, they take quite some time to become experienced enough in the art and science of software development. Experienced programmers are assumed to leave the firm after they serve the firm for about eight years (one year in training, two years in gaining useful experience, and five years as experienced programmers).

In the model the recruitment policy of the firm is assumed to depend on the number of programmers on roll and the requirement of programmers to do the development and corrective works for the software.

## *Order Formation*

We assume that under normal conditions of quality and delivery delay the firm's normal order rate is constant at 50 Kilolines of code per month. But with varying quality and delivery

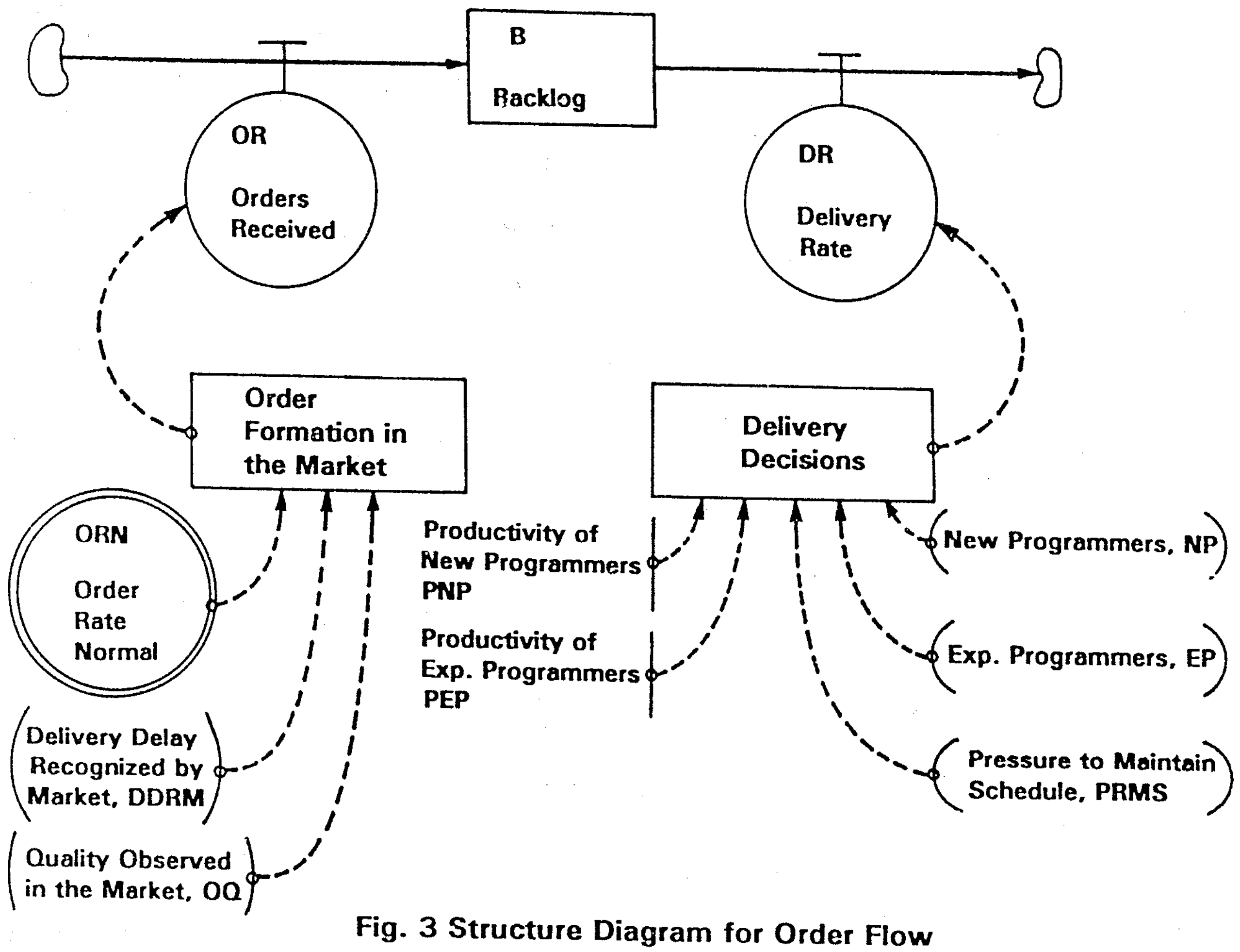

B
Backlog
OR
Orders Received
DR
Delivery Rate
Order Formation in the Market
Delivery Decisions
ORN
Order Rate Normal
Delivery Delay Recognized by Market, DDRM
Quality Observed in the Market, OQ
Productivity of New Programmers PNP
Productivity of Exp. Programmers PEP
New Programmers, NP
Exp. Programmers, EP
Pressure to Maintain Schedule, PRMS

**Fig. 3 Structure Diagram for Order Flow**

delay levels, the actual order it receives will change. Higher quality and lower delivery delay values relative to their normal values in the market will increase the orders received figure, and lower quality and higher delivery delay values will decrease it. The effects of quality and delivery delay are modelled as multipliers. The market share of the company is the average delivery to the market relative to the potential demand. Delivery delay recognized by market is a smoothed version of delivery delay recognized by the company.

Order rate multiplier from quality is likewise defined. Here quality of a software is defined in terms of number of errors per thousand lines of code. Quality perceived by the customers is the impression created by the users about the software they are using when the errors that pass the installation phase surface.

## *Software Development & Delivery*

It takes, on average, about a year to develop a software and deliver it to a customer. The firm therefore always has a backlog of orders with it. As new orders are received, backlog increases; and as deliveries are made to the customers, backlog comes down.

Here, delivery rate is taken synonymous to development rate. It is assumed here that software development productivity of a new programmer is low relative to that of an

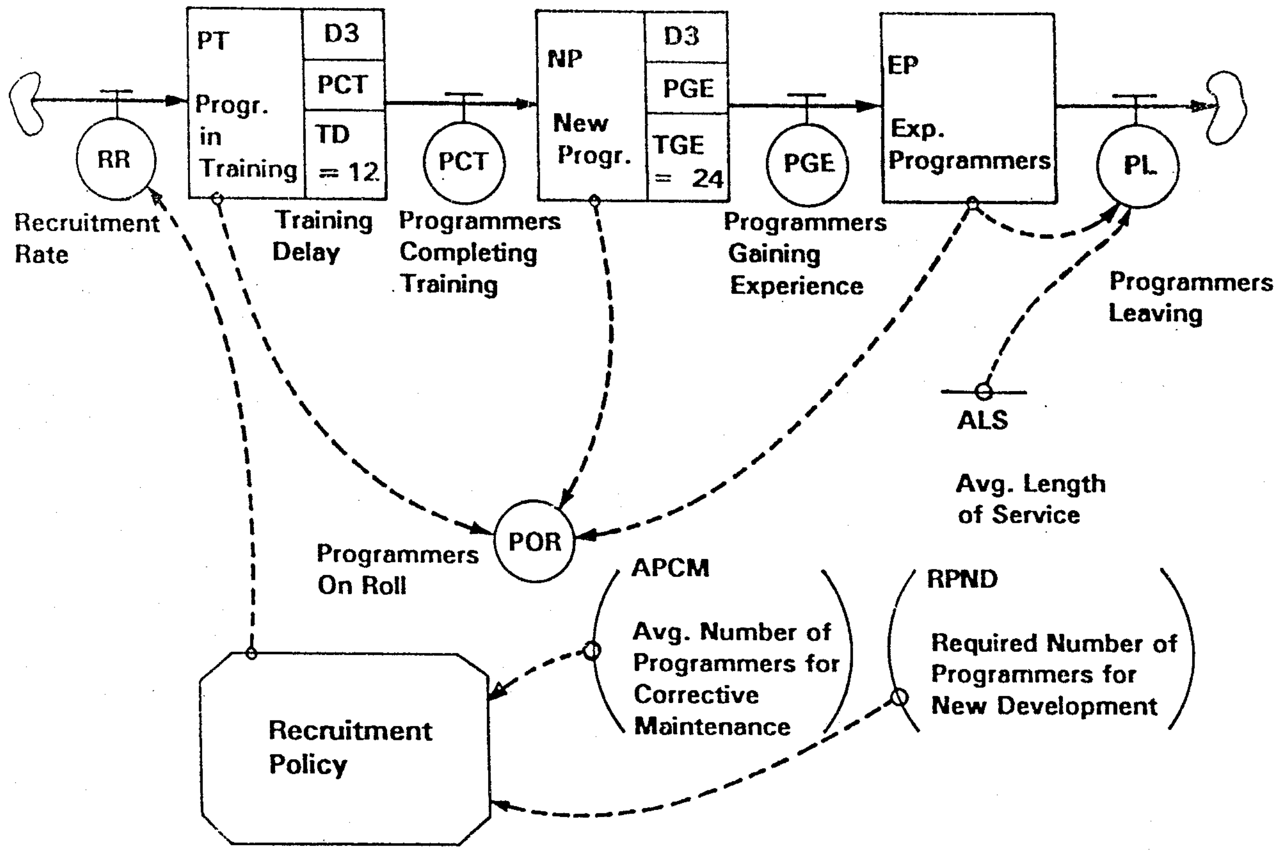


Fig. 4 The Structure Diagram for Flow of Programmers

experienced programmer. It is further assumed that on average, an experienced programmer spends about 25% of his(her) time on a new programmer, exposing him(her) to the nuances of the programming environment of the firm. Further, programmers engaged in removing errors are not available for developmental work.

It is assumed that at times of schedule overruns, the programmers will be under pressure and will try to be more productive, perhaps by cutting down their communication overheads, personal times, and even working overtime.

As delivery rate rises, delivery delay falls, and as backlog rises, delivery delay rises. So delivery delay can be modelled as a ratio of backlog and average delivery rate. As the management observes that its delivery delay position is worsening, it is under mounting pressure to maintain the schedule.

## *Software Quality*

Quality is defined here as the average number of errors per kilo lines of code. It is assumed here that on average and under normal conditions, one error per one thousand lines of code remains latent in the software at the time of its delivery to the customer. However, as pressure mount due to schedule overruns, more errors pass through the testing phase to the customers. As the software once installed and customers work on it, there errors surface and are corrected by the firm. It is assumed here that about 80% of the errors that pass on to the customer get detected during the installation stage of the software and the firm is under obligation to correct the errors.

The rest 20% of the errors may be detected, however, after this period of installation for which the firm is under no obligation to correct.

## *Human Resources*

We can discern programmers in three different groups: (1) those who are under training, (2) those who have completed training, but are still new and inexperienced, on average with less than two years of experience, and (3) those who may be considered as experienced.

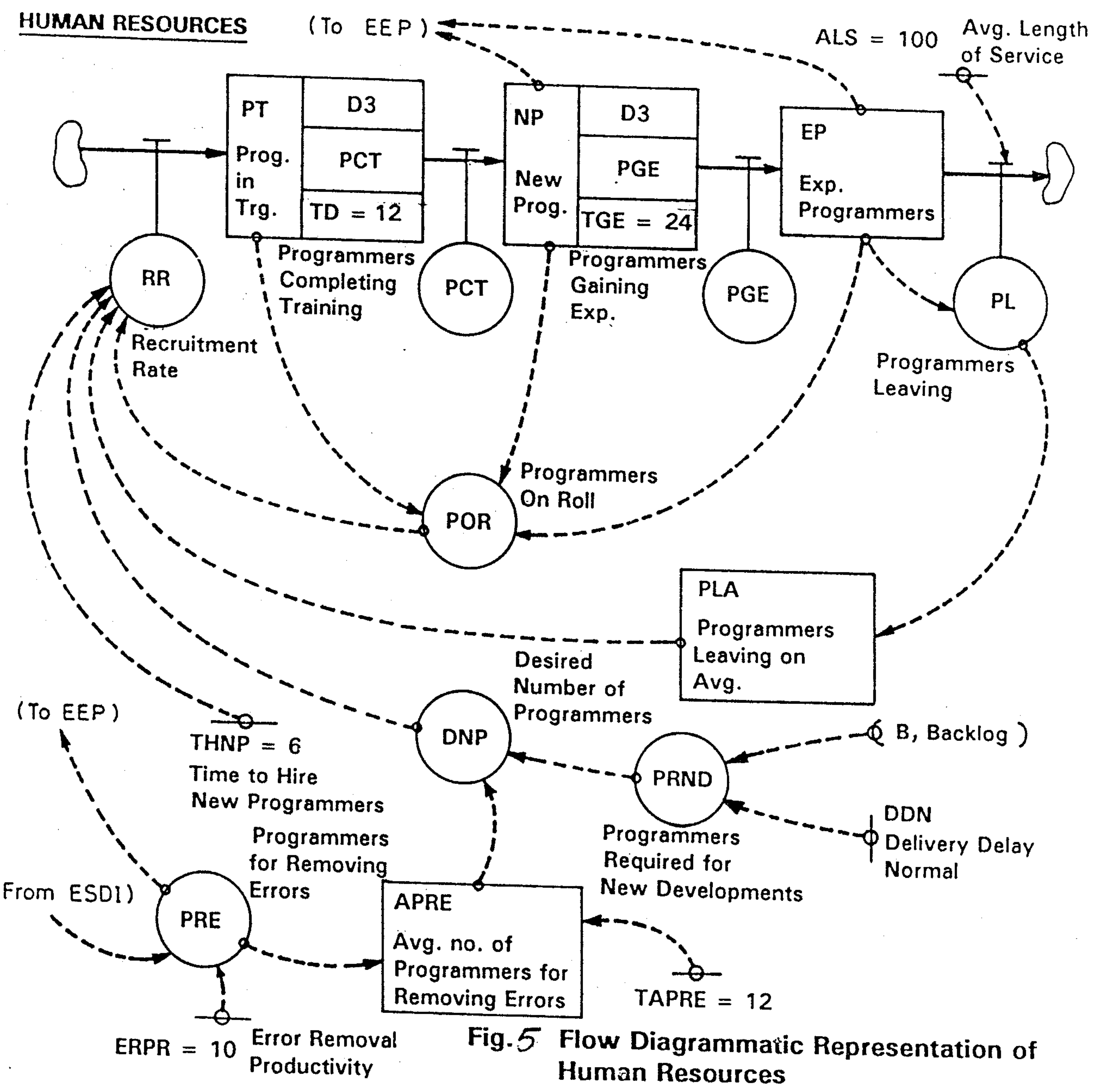


Fig. 5 Flow Diagrammatic Representation of Human Resources

While recruiting fresh programmers, the firm considers the outstanding backlog of work and the average rate at which programmers leave the firm.

The desired number of programmers equals the average number of programmers for removing errors and the required number of programmers who can work on the backlog to give the delivery on schedule.

Training and gaining experience are considered third-order delays. The average length of service of an experienced programmer is taken as 100 months.

Error removal productivity is taken as 10 errors/KLOC-month. So persons for removing errors is given by total number of errors surfaced in a month divided by error removal productivity.

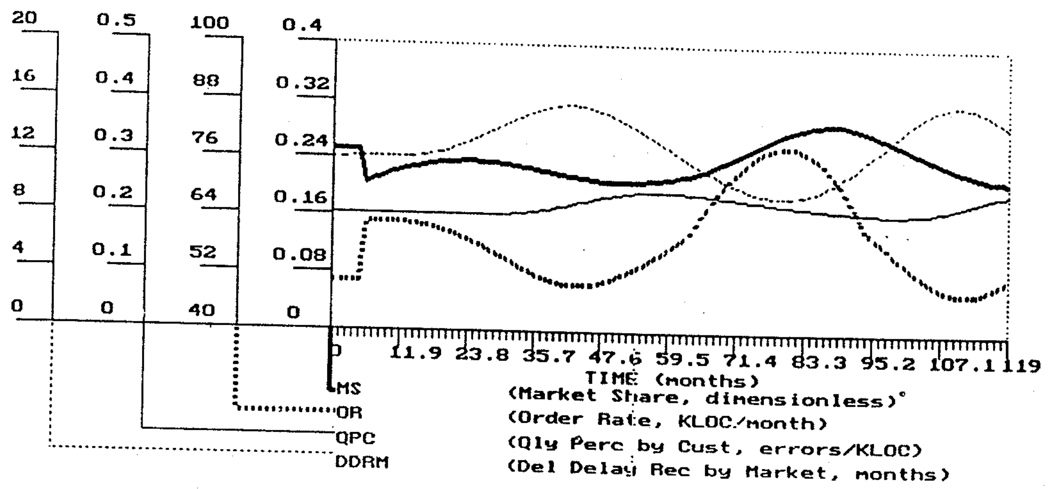
Figure 5 shows graphically the details of modelling for the human resources sector.

## The Base Run

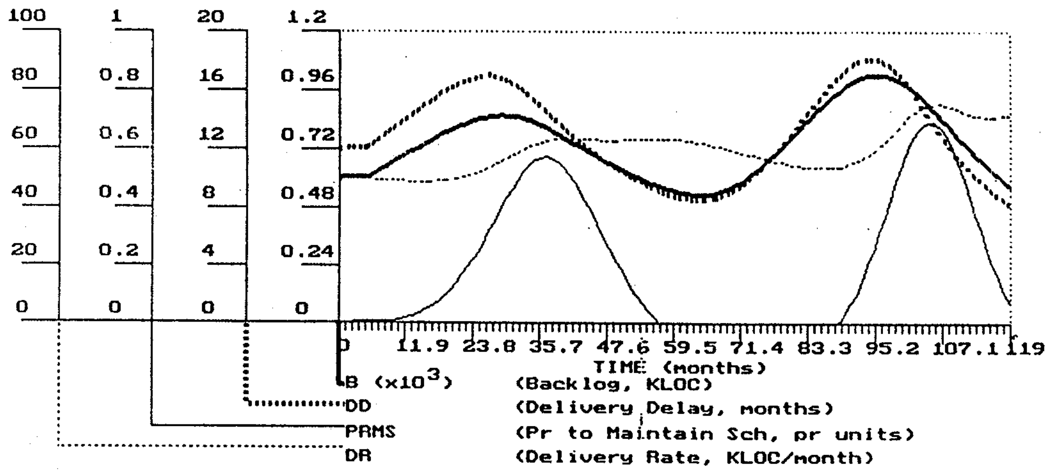
We define market share as the ratio of average order rate received by the company to the potential demand in the market. We assume the normal market share to be 25% and initial potential demand to be 200 KLOC/month. Further we assume that the potential demand undergoes a step change immediately after the fifth month.

We now simulate the model. Figures 6 and 7 show the response of the model to the step change in the potential demand.

We see in these figures that as the potential demand undergoes a step change, order rate rises, but slowly falls to quite a low level before rising and falling again. Market share shows similar rise and fall (Fig. 6). This can be explained by the fact that as order rate rises, backlog rises, and delivery delay and pressure to maintain schedule rises (Fig. 7). With this pressure rising, backlog no doubt falls, but quality (errors/KLOC) and even quality after error removal suffer. This ultimately reduces the customer order to the firm. Maintenance of rising number of errors take away programmers from new development. To reduce the delivery delay to a normal value, the firm recruits more programmers who undergo training. Programmers in training rise; after a few months they join as new programmers who, after a few months, become experienced; thus programmers show a rise after a while.



STEP CHANGE IN POTENTIAL DEMAND
FIG. 6 VARIABLES IN THE ORDER FORMATION SECTOR



STEP CHANGE IN POTENTIAL DEMAND
FIG. 7. VARIABLES IN THE DEVELOPMENT & DELIVERY SECTOR

Paradoxically, when the number of programmers is the highest, the customer order rate to the firm falls, reducing backlog, and reducing the delivery delay and the latent errors in the software. In turn, the customer order rises again; thus the cycle repeats.

We see that the firm displays fluctuating business over a 12-year period. It is unable to take advantage of a permanent, positive change in the potential demand in the market.

## Testing Policy Changes

We now try to evaluate the effect of a few policy changes.

*Policy I*

Here we assume that the firm creates an environment quite conducive for the career growth of the programmers so that they continue to serve the firm for a much longer length of time. Here we assume that on average, a programmer would stay with the firm for 10 years.

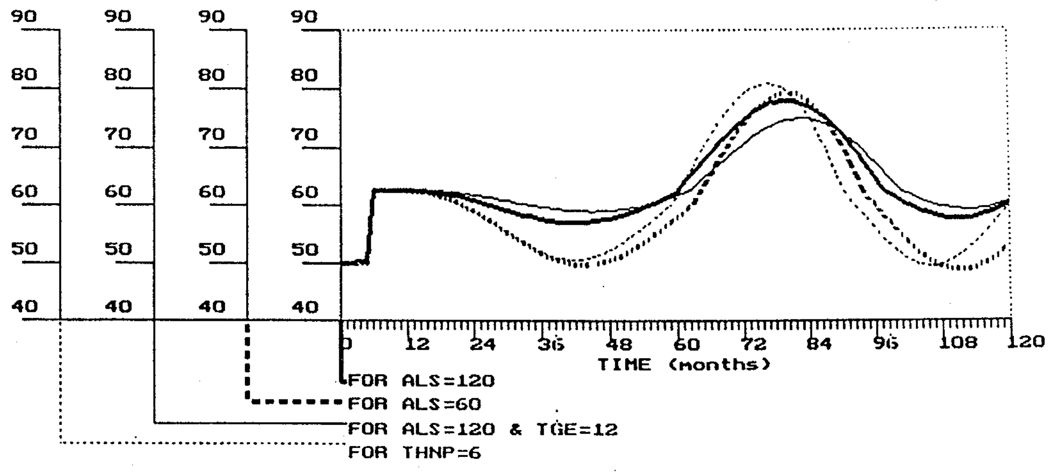
*Policy II*

Here we assume that the firm, in addition to creating an environment with a view to holding the programmers for a longer period of time, recruits programmers who take much shorter time to gain experience. To implement this policy we assume that the average length of service is 10 years and the time to gain experience is 12 months.

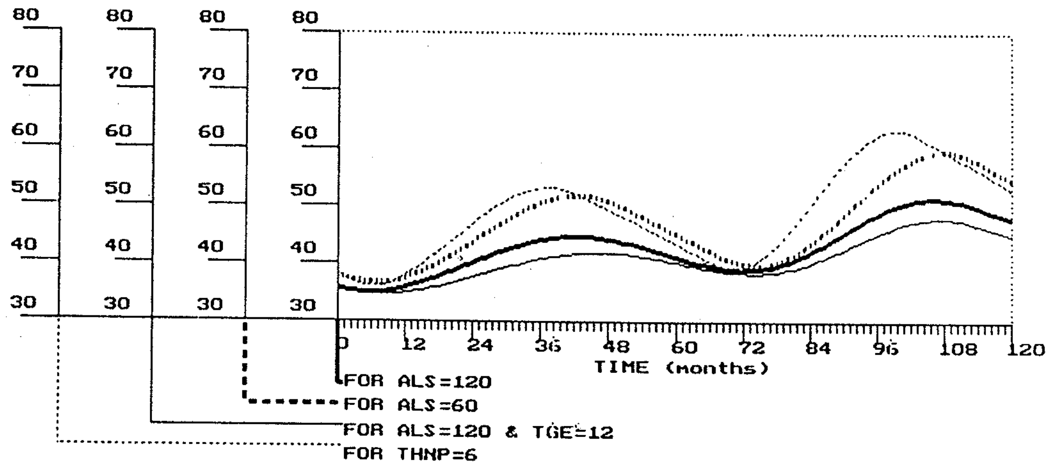
*Policy III*

Here we assume that the firm reduces the time to hire new programmers. It reduces this time from 12 months to 6 months.

Figures 8 and 9 show the variation of order rate and programmers on roll respectively for these policies and for the base policy where the average length of service is only 60 months. A comparison of these figures indicates that the policy II (where the programmer is assumed to stay for a longer period of time and where the time to gain experience is smaller) gives a much improved behaviour of the model compared to the base run and to the other two policies.



OR -- TIME
FIG. 8 : CHANGING HUMAN RESOURCE POLICIES



POR -- TIME
FIG. 9 CHANGING HUMAN RESOURCE POLICIES

## References

Jones, T. C. (1986), Programming Productivity, New York: McGraw-Hill.
Pressman, R. S. (1992), Software Engineering: A Practitioner's Approach, New York: McGraw-Hill International Editions, Third Edition.