UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

****

AN INTEGRATED PAYROLL-PENSION SYSTEM

by

EVELYN L. HWANG

****

A thesis prepared under the supervision of Dr. H.F. Mitchell of the Department of Computer Science in fulfilment of the requirements for the degree of Master of Science

****

A C K N O W L E D G E M E N T S

I would like to express my appreciation to Southern Life Association for providing the working environment which contributed greatly to my understanding of the problem at hand. Special thanks are due to Ed Pryor, Keith Rentzke, Dennis Korck, and Keith Mattison for their help in providing helpful information during system design. I am particularly grateful to my thesis supervisor Dr. H.F. Mitchell for the guidance he has extended during the research period and the writing of this thesis. Finally, I would like to thank my husband for the constant encouragement he has given me this past year.

A B S T R A C T

An integrated payroll-pension system designed specifically for commercial users with limited computer resources is presented. This system combines in a single system the usual functions of both a payroll system and a pension system. Apart from economies which result from reduced software requirements and minimized duplication of files, the particular integration approach adopted leads to the inclusion, within the system design, of numerous capabilities normally associated only with a computerized personnel system. The integrated system has been so designed that it can serve eventually as a nucleus for a personnel information system.

T A B L E O F C O N T E N T S

C H A P T E R O N E

INTRODUCTION AND SUMMARY

"In terms of immediate practical effects, the overriding issue of tomorrow - and of the day after tomorrow - is neither the technological demands nor managerial problems. The overriding issue is people."(1) This viewpoint is indicative of the growing importance of the subject of personnel in general, and the significance of personnel planning in particular. In the present commercial world, various aspects of personnel operations are rapidly becoming computerized. However, personnel operations encompass such a wide scope that only very big companies and organizations can afford full scale computerized personnel systems. On the other hand, companies with access to computer facilities usually have computerized payroll systems. Less common, though also frequently encountered are automated pension systems. These two types of systems can serve as the nucleus of a fully expanded personnel system. Indeed, many companies which decide to computerize their entire personnel operations give priority to upgrading their payroll and pension functions.

The payroll and pension systems are linked closely despite their serving distinct functions. The payroll system normally has as its population group, employees in active service whereas the population group for the pension system is comprised of pensioners and active employees who are members

of the company's pension plan. The linkage between the two systems is tenuous when the company does not have its own pension plan and subscribes instead to a pension package provided by an external agency. But in an organization which has its own computerized pension system, the pension system can be closely allied with the payroll system to lay the foundation for computerizing personnel operations.

This paper presents an integrated payroll-pension system designed for companies which satisfy a general set of pre-requisites. The design of the system treats payroll and pension as two distinct components, each of which performs a unique set of tasks, yet both sharing considerable computational facilities and data storage areas. Apart from documenting the system design, this paper will attempt to analyse the factors that must be considered in arriving at the final design.

1.1 Background

At times the best way to study a problem, in this case to design a functional integrated payroll-pension system, is to work within the environment of an actual user. Although working with the requirements of a specific company can bias the design of the system towards the needs of that company, this bias can usually be kept inconsequential if sufficient care is taken during the course of system design. Furthermore, the advantages outweigh the negative aspects. Developing the system within the working environment of a potential user not only

clearly identifies the set of end users of the system but also benefits from a deeper understanding of the practical objectives and problems encountered by the end user. And most important of all, the EDP environment and computer facilities of such a company will help set the operational limits within which the designed system is to function.

Fortunately the design of the system presented here was conducted within the working environment of Company X, and greatly benefited from the aid and expertise of that company's personnel and data administration departments. Company X is an insurance company with an employee population of approximately 1200 people. It makes use of an ICL 1900 computer for the company's electronic data processing. The payroll function of Company X is currently being handled, unsatisfactorily, by a packaged system widely used in local commercial circles. The company has its own pension plan for company employees. The basic pension computations are done manually and subsequently input into its packaged payroll system merely for payslip printing. The immediate requirements of Company X are to substantially upgrade or replace its present payroll system and to completely automate its pension function. Eventually, the company aims to develop a full scale personnel system to meet its personnel management needs.

An analysis by the author on the payroll system of Company X revealed that the root of the company's difficulty with its payroll system is its use of a packaged system. A

packaged computer application system is designed to cater for various types of users. Unless very carefully designed, packaged systems usually cannot be adapted to deal with all the needs of an individual company, let alone handle the requirements of the company as it grows and expands. The packaged payroll system presently used by Company X may have been sufficient for the company in the past, but under present conditions, its performance is unsatisfactory. Its major deficiencies can be enumerated as follows:

1) Since it must cater for widely diverse users, the package is so complex that it prevents the user from making adjustments to the package as the user does not have absolute control over its set-up.

2) The history function of the package is unsuitable for Company X, being far too complicated to be of any value to the company.

3) Calculation procedures are vague and inapplicable in many instances. Since the user is prevented from revising system algorithms due to his limited knowledge of the entire system set-up, many calculations have to be manually performed and the results fed into the system to produce payslips.

4) Some of the output currently generated
   a) are unnecessarily repetitive and can be reduced to simpler and more compact forms,
   b) contain unnecessary information as well as formatted but unutilized space, and

c) are unsuitable in their existing formats and must be altered.

5) Additional reports needed by management cannot be produced.

6) Finally, the package cannot be used as a basis from which to develop a future computerized personnel system.

Thus, the alternatives are either to radically revise the application package currently used or to design a new payroll system more adaptable to company needs.

As for its pension function, Company X uses a series of manual recording procedures for pension records at present. In view of the complexity of the calculations involved and the record population that must be maintained, Company X finds these manual procedures to be excessively time consuming and inefficient. Hence the resolution to computerize the pension function. There are two ways of doing this: one is to develop a new system within the company's computer environment, and the other is to subscribe to a commercial pension package.

1.2 General Design Factors

It is always valuable to review the various factors a systems designer-analyst should consider before undertaking the design of a specific system. The problem should first be examined in relation to company resources. The primary question is whether the function under study needs to be computerized at all. A review of the company resources and objectives might disclose that computerization is not required. On the other

hand, computerization might be necessary. In an increasingly competitive business environment, computerization provides undeniable advantages. This is evidenced by an increasing number of faster and more sophisticated hardware as well as larger and more expensive computer staffs. Usually these developments arise from the desire of management for a better information base to improve coordination and administration, to strengthen planning, and to reduce the risk in decision making. But in other cases, the need to computerize may be due to the company's position in the industry. Usually when one company in an industry computerizes its functions, then all other companies in the industry are at a disadvantage until they follow suit, even though the cost of the new tool may be very steep in terms of investment, training, manpower or management education.(2)

Let us assume that computerization is called for. The next step is to determine how to use the computer most advantageously. For this, the analyst must access the individual systems that constitute the building blocks of the computer system of the company. It is important to anticipate the impact of the various alternatives (or of the proposed system if all other alternatives have been eliminated) on all existing systems. Moreover, if there are other systems pending implementation, then these systems must be evaluated together with the proposed system. Each system should be analysed for risk, security and return, for like any investment project, the priorities of implementation must be seen in the light of economic

and strategic factors. Crucial issues in determining implementation priorities include program resources required to develop each system, the development and implementation timetable for each, the number of systems and nonsystems personnel involved in the development effort, and finally the length of time before the system realizes a return.(3)

If the priority for the system is sufficiently high to merit implementation, then the system related factors enter the picture. Most companies start using the computer while still in the expansion stage. Therefore any new system needs to be designed with considerable foresight, bearing in mind not only the company's present needs but also its future anticipated needs. Such additional effort could well ensure that the system so designed remains functional for a reasonably long period of time without undergoing drastic revisions. Unfortunately, this general rule is ignored all too often, frequently resulting in a recently implemented system becoming obsolete rapidly and requiring either major revisions or even replacement by a new system. Within the South African context, there appear to be two related developments which bring about the rapid attrition of computerized systems, particularly personnel related application systems. One is that for various reasons, many companies requiring new systems simply take in a packaged system without adequately analysing what the package can and cannot do and, even more importantly, for how long the package

can serve company needs[1]. The other development is that companies whose current systems are on the verge of becoming out of date switch to packaged systems without prior careful study and without seriously considering the alternatives of revising their existing systems or of replacing their present systems by systems designed specifically to meet their needs[2]. Subsequently, a decision must be made whether the existing system should be revised or a new system needs to be developed. This involves determining the amount of work entailed in the revision: the vastness of the logic to be changed and the resources that have to be mobilized for the task. When the revisions are substantial and the user is not adequately familiar with the make-up of the system, frequently the best solution is to develop an entirely new system.

This brings us back to the problem confronting Company X. The revision required of its present payroll system is an extremely delicate task which would need an intimate knowledge of the packaged system's design and should not be attempted by any user without total control of the system. Thus the alternative is to develop a new payroll system which will be required to fulfil all presently utilized functions of the packaged system, but without the package's accompanying deficiencies. Besides, many reporting facilities and other

---

[1] Note that no disparagement of packaged systems is intended here. There are many excellent application packages commercially available today. But even these, when applied injudiciously, result in difficulties for the user.

[2] Apparently some academic institutions can fall prey to this tendency also. Note for example the recent difficulties the University of Cape Town has had with its payroll system.

long desired but unavailable functions can be incorporated in a system designed specifically for the company's use. As for the pension system of Company X, there are several reasons that argue against adopting a packaged system. Foremost is the fact that subscribing to a package will entail modifying their current pension procedures, which have functioned satisfactorily. Moreover, the experience of Company X has been that utilizing an off-the-shelf system has been only marginally successful in the past since the characteristics of Company X (as in the case with any company) - its personnel, management procedures, training needs, regulatory requirements, and its organizational structure - are unique. Techniques and approaches may be transferable, but the detailed design of an individual system usually is not.(4) Finally, choosing the correct package involves considerable time and expense spent in evaluating the packages currently available[3]. Thus for Company X, the proper decision is to develop new systems to replace its problematic payroll system and automate its currently manual pension function.

The initial decisions having been made, the systems designer must then obtain from the user the features required of the new systems and the constraints within which the design is to take place. These are pivotal factors in determining the type of design that will be most effective. In most cases,

---

[3] It might well be asked whether the system design presented here is suitable for any other user other than Company X. It is not the intention that the system design can be taken over in toto by any user. Rather the intention is for the design to serve as a model on which specific system designs can be based. If the system presented here is applied to other users, changes in programming specifications and perhaps even in the design structure may be required.

the system to be developed can be analysed in the following light: Is the system independent? Is it related to another set of systems? Can this system be made a subsystem[4] of another one? Is there another system pending development which is so closely related to this one that the design can be expanded to include both within the same framework? Both the payroll and pension systems can be designed as independent systems. However, since for Company X the two systems share not only numerous computational algorithms but also a common population set, combining the two functions to form a single system is strongly indicated. Moreover since the payroll population set records must be set up before the pension records, it is only logical to view the pension function as a subsystem which is to be integrated into the payroll system.

1.3 Summary

This paper presents the system design of an integrated payroll-pension system which is meant to serve as a model for a certain class of companies intending to computerize their payroll and pension functions. The system logic is new, derived independently of any existing system. The basic functions of the system are to: (a) maintain each employee's record upon entry into service up to his termination or resignation, (b) continue to care for a retired employee's record until his death

[4] A system is said to be independent if it has its own logic and does not depend on other sources for its information and output requirements. If in addition, a system shares the logic and data of a master system, it is said to be a subsystem of the master.

and the maturity of all his dependants' records, (c) perform a similar function as in (b) for in-service deaths. The integrated system has several advantages. The system

1) Is designed in such a way that the user has alsolute control over it, thereby allowing the user to revise the system when there is a need to do so;

2) Stores in permanent files all job and salary histories of employees together with any additional employee information which the user desires to keep in these files;

3) Provides new algorithms for pension calculations;

4) Maintains records of all types of loans issued by the company to its employees during their service periods, together with the loan amounts, interest rates, dates of issue, and other related data;

5) Determines the appropriate amount to be deducted for pension and medical aid from each employee's income, automatically adjusting the deduction whenever the conditions which determine it changes;

6) Provides new algorithms for computing the income tax deduction;

7) Computes the UIF employer and employee contributions based on the most recent UIF rules;

8) Computes a user specified yearly interest on the accumulated pension contributions to date of every employee;

9) Converts an employee to a pensioner upon his retirement;

10) Computes the annual pension (based on the final average salary which in turn is also computed), death benefits, and other related pension details;

11) Monitors the maturity dates of dependant allowances and performs all necessary changes in allowances whenever applicable without any additional input;

12) Includes the leave records of employees.

When applied to Company X, the integrated system, apart from providing the features listed above, resolves all difficulties the company is experiencing with its present payroll system and automates its pension function in a practical and efficient way. In addition, the new system fulfils all functions currently performed by the company's existing system, produces all existing reports, and provides new reports as well, for example, staff turnover reports and loan reports.

Finally, the integrated system has been so designed that it can be expanded to generate a data base to serve a full scale personnel system. While only a restricted group of users can afford to develop a full scale personnel system, the users of this system will have the facility to initiate particular aspects of personnel management planning, particularly in the areas of salary and incentives administration, promotion and training methods, in the monitoring of leave accumulation, the forecasting of future manpower needs, and the projection of future pension fund requirements.(5)

Naturally just as the new system has benefits, it also has limitations. The system, having been designed within the working environment of Company X, is biased towards companies characterized by a reasonably large staff and with limited computer resources, thereby preventing the inclusion of data structuring techniques that can be used if more computer resources were available[5]. Moreover the system is South African oriented. For example, the income tax deduction procedures are based on South African regulations[6]. The pension schemes to which the system can be adapted are most likely constrained by the range of pension schemes encountered locally.

In the succeeding chapters, the problem at hand is examined in greater detail and the solution adopted is more carefully discussed. Chapter 2 places the problem in perspective and discusses the functions and building blocks involved in a personnel system, a payroll system, a pension system, and finally an integrated payroll-pension system. In Chapter 3, a logical overview of the integrated payroll-pension system is given. Data organization and basic system operation are explained in the chapter. The final chapter is a technical presentation of the integrated payroll-pension system. The way the system works, the manner in which files interact with one another,

---

[5] This restriction is not so bad from a practical point of view if the availability of only limited computer resources is typical of most interested users.

[6] Although the specific regulations will be different, the algorithms developed might still be applicable elsewhere.

and the programming logic are described here. Finally, an estimate of implementation requirements for the system is provided.

C H A P T E R T W O

THE PROBLEM IN PERSPECTIVE

2.1 The Personnel System

Although frequently unrecognized and long ignored, the personnel department is a fundamental service department in any modern commercial organization. Apart from payroll and pension administration, the personnel department has many other responsibilities among which are recruiting and training of staff, and supplying management with information on personnel for use in job analysis and manpower requirement forecasting[7]. In recent years, business has become aware that personnel are amongst the most valuable assets of a company. Increasingly, it is recognized that the efficient management of human resources is often the key factor in profits and losses. Hence the momentum to bring personnel policies and practices up to date. An upgraded personnel system involves various factors. Efficient corporate planning is desirable. Accurate knowledge of employee capabilities are required. Similarly, career path planning and competitive salary policies are also needed.(6) To establish these conditions requires the creation of a data bank of accurate and up-to-date personnel information, for lack of data is frequently the primary impediment to effec-

[7] Indeed in some organizations, the personnel department is responsible only for personnel administration, the payroll and pension functions being under the jurisdiction of a different department in the organization, for example, the accounting department.

tive personnel planning. Usually such a data bank involves a large volume of data and is therefore expensive. Thus it is imperative that efficient data processing techniques be used to deal with the large volume of personnel information required by a modern company. The task before us therefore is clear: to design a computerized system which can serve as a nucleus for a fully expanded personnel system[8].

The personnel data bank should include every employee's biographical data, educational background, professional qualifications, training, details of recruitment, job description, work location, work conditions, salary, pension details, leave records, and performance records.(7) For many of these items to be useful in personnel planning, historical data is needed. However, the volume of historical data can be unexceptably large if a history of all items is kept indiscriminately or if historical data are kept in file indefinitely. Thus each company should review its requirements and decide what historical information to keep and for how long. For example, promotion requires a history of employee grade and occupation code changes. For analysis of past staffing and wastage, leave records which extend over a reasonably long period of time are essential. Salary reviews involve employee salary histories and past performance records. And determination of pension benefits must take into account past service and breaks in

---

[8] Design of a full scale personnel system is a major task and beyond the scope of this paper. For example, the personnel system being designed by IBM is estimated to require 20 man-years to complete.

service.

A significant proportion of the contents of the personnel data bank are data which are normally kept in standard payroll records. Whenever there is an existing payroll system in the company, the volume of data that is kept in the personnel file is considerably reduced. This is true particularly when the payroll function is a personnel department responsibility. Occasionally it is not, as a result of which coordination between the personnel system and the payroll system can be quite difficult. Nevertheless when such is the case, a link should be established between the two systems to enable the personnel system to access data from the payroll system. Apart from avoiding unnecessary and uneconomical duplication of information, this link gives the personnel system necessary momentum to keep it current. Indeed, a personnel system can be built around a well-designed payroll system.(8) The payroll system is usually up to date and invariably contains reliable data, thereby providing current data for much of the personnel data bank without additional effort and expense.

Pension records are another extremely reliable source of data. These records usually contain precise information on long term leaves and staff turnover, and can be used therefore for wastage analysis especially when a large proportion of staff are members of the company's pension plan. The main disadvantage of standard pension records is that they are not designed for statistical analysis.(9) Hence if the company does not

have a suitable computerized pension system, the design of a new pension system should provide for statistical analysis so as to interface optimally with the personnel system.

2.2 The Payroll and Pension Systems

In view of the central roles pension and payroll systems can play in a fully computerized personnel system, this section briefly reviews the standard features contained in these systems. However, before proceeding with the review, a note of caution is needed if proper perspective is to be maintained. A common misapprehension is that any computerized system is always better and cheaper than a manual one. This is not true of course. Certainly, a computerized system is more precise and more reliable, but it is usually also more expensive. Computerization becomes economical only when economies of scale enter the picture, that is, when the volume of data involved is large. Thus, small organizations can function efficiently with manual payroll and pension systems. However, any organization with a staff strength approaching 1000 people, and certainly those with larger staff, can justify computerization of these systems.(10) Henceforth it will be assumed that the discussion is applied to organizations sufficiently large to require computerized payroll and pension systems.

A company's payroll system is responsible mainly for paying the wages of company employees and keeping records of such payment. The payment period can be in terms of days,

weeks, or months. The payroll system can vary from rather simple systems to fairly sophisticated ones. The basic computations take into account earnings, deductions and all other allowances. The factors which must be considered consist of the regular salary, income tax deduction, medical aid contribution, pension fund contribution, overtime, various fringe benefits, allowances, bonuses, and loan repayments. The first five factors are standard. The others however, can mean different things to different companies. It is expedient to define the scope covered by these factors in our proposed solution. Fringe benefits granted to staff include paid holidays for employees, paid sick leave, executive leave, and continued payment of salary for some specified period to an employee injured while on duty. Allowances cover a wide range of special payments. Travelling allowances usually cover fares, hotel bills and other expenses incurred by employees while travelling on company business. Moving allowances pay for transferring employees and moving their household effects from one location to another location on a relatively permanent basis. Housing subsidy is also included among allowed allowances. Some allowances are issued as integral parts of basic earnings such as car allowances and specialist allowances, and are granted regularly every pay period. A bonus on the other hand is supplementary to, rather than part of the primary payment. Its main value is to motivate and provide extra incentive for employees. Bonus types considered here include the annual or so-called thirteenth-month bonus as well as the production bonus which

is usually given to senior executive staff, frequently in several annual instalments, with the amount depending on the company's annual profit.[9] Finally, loan repayments account for the fact that some companies issue loans to their employees. Loan types allowed for include study loans, car loans, and personal loans. Interest may or may not be charged on these loans depending on the loan type. Deductions corresponding to repayment of loans are subtracted from the employees' wages on a mutually agreed schedule.

The main components of a payroll system should include (1) computerized payslips, preferably containing only those items which are relevant to each employee, (2) computer generated reports to satisfy management needs and government requirements, (3) payment of wages directly to employees or into their bank accounts, (4) deductions for participation in insurance schemes and medical plan, if applicable, (5) automated income tax calculation and deduction, (6) overtime payments and other allowances, (7) loan repayments and monitoring of outstanding debts.

These features can vary from the very simple to fairly sophisticated procedures, depending on the requirements of the user and whether or not the payroll system is to serve as a stand-alone unit. For our purposes, if it is to play a central role in the development of an expanded personnel system,

---

[9] Strictly speaking, the production bonus is actually a fringe benefit extended primarily to top executives.

the payroll system needs to be flexible and thus requires reasonably sophisticated features. In addition, it must contain record keeping facilities not normally asked for in a standard payroll system.

Like the payroll system, pension system can play a significant role in the development of a computerized personnel system. The pension system picks up where the payroll system ends. Whereas the payroll system services only employees in active service, the pension system caters to employees who have retired and are eligible for pension and to dependants of deceased pensioners and employees who are eligible for pension benefits. The main tasks of a pension system are to determine the pension amount due to each pensioner, to compute benefits and other similar functions specific to company pension rules, to ensure that income tax deductions and all other suitable contributions are accounted for, and finally to arrange pension payment. It is clear that although the pension system services a larger population set than that serviced by the payroll system,[10] many functions are duplicated in the two systems. Thus many companies, particularly organizations with small staffs and even fewer pensioners, deal with the purely pension related aspects manually and use their payroll systems for pension payment and tax deduction through manual input of required pension data. This course is sensible for small organ-

---

[10] The pension system accesses the records of active employees, which it shares with the payroll system, for deduction of pension contributions. In addition, it also maintains the records of all pensioners and their dependants.

izations. But for companies of the size with which we are concerned, this approach has two distinct disadvantages. Considerable record keeping and management monitoring functions are sacrificed. Furthermore, the manual administration of the pension function for a large company can be expensive and tedious, regardless of whether the pension plan was developed specifically for the company or purchased from specialist pension or life assurance agencies.

The incessant introduction of new types of pension plans makes it impractical to present a thorough review of available schemes here. Sufficeth to note that different pension plans vary substantially both in cost to the company and in terms of contributions from and benefits to the employee. The simplest type of scheme grants the retiring employee a standard pension amount payable either as a lump sum or in regular instalments. At the other end of the spectrum, a scheme can allow the employee to adjust his pension rights upon retirement to include life assurance arrangements since, in the simpler schemes, pension benefits cease to apply when the pensioner dies. This can be adjusted in such a way that the pensioner agrees to receive reduced pension payments but with either continued payment of the same or even reduced benefits to the surviving spouse and eligible dependants upon the death of the pensioner. There are many complications involved; for example, whether the pension plan is a contributory one[11],

[11] A fundamental point in pension plan policy is the question whether or not employees should contribute to the cost of the plan. For some years past, the most advanced plans were non-

whether early retirement is allowed and if different pension benefits apply to early retirees.(11)

There are two basic types of calculations a pension system must undertake: regular deduction from salaries and corresponding accumulation of the contributions over the years of active service, and the computation of the pension benefit due to the employee at the time of his retirement or to his dependants in the event of his death. The latter computation can be quite complicated as is demonstrated in Appendix B.

The main components of a pension system should include: (1) computerized payslips, preferably including only items directly applicable to each pensioner, (2) computerized reports for management needs and as per government requirement, if any, (3) payment of pension benefits directly to pensioners or into their bank accounts, (4) deductions for participation in medical and insurance schemes, (5) automated computation of income tax deductions, (6) the capability to determine the final average salary on which pension benefits are normally based. As in the case of a payroll system, for our purposes, the pension system must include adequate record keeping facilities, with

---

contributory. But due to 'cost' and 'obligation', the case for a contributory plan was made. The cost of a generous pension plan is very substantial and few companies now feel that they can carry the full cost of the plan alone. Also in the non-contributory plan, the employee will be obliged to stay with the company for the sake of his eventual pension since there is no question of withdrawal of funds not paid in by him. This is true especially for employees being with the company for quite a number of years. Such 'golden handcuffs' type of practices are frowned upon increasingly by both companies and employees alike.

the added constraint that duplication of records and functions with the payroll system must be kept to a minimum.

2.3 The Integrated Payroll-Pension System

Although their spheres of responsibility are clearly demarcated, the payroll and pension systems have in common so many basic computational functions that integration of the two systems would be desirable, provided certain conditions are met. There are two sets of pre-requisites. The first set relates to the systems environment. The second set addresses the question of the goodness of fit of such an integration. It is essential first of all that

1) The company needs both functions. It does not matter if either of the functions is currently manually operated. Nor is it necessary for both functions to be under the jurisdiction of a single administrative department although this would be most desirable.

2) Basically the two systems should share the same population set.

These first set of conditions are minimal pre-requisites. The case for an integrated system is strengthened considerably if, in addition, within the company, the two functions are interrelated in the following ways:

3) All employees of the company are potential pension fund recipients.

4) Employee contributions to the pension fund are on a

regular basis.

5) The eventual pension received by a pensioner depends on his income history during his service period.

When these additional conditions are also satisfied, the two systems will fit well together and their integration can be seriously considered. Therefore the problem to be solved can now be posed as follows: Assuming that the preceding conditions are met, design an integrated payroll-pension system which serves both payroll and pension functions, yet minimizes software and file storage duplication.

It will be shown in the next two chapters that integration need not result in any loss to the functions that need to be performed. The integrated system can run as an ordinary payroll system as well as a standard pension system since the entire logic of both functions are incorporated into the integrated system logic. Moreover, substantial advantages are realized by the proposed system. Operation costs are reduced since only one system,albeit and an enlarged one, instead of two systems is in operation. Furthermore, economies result as a consequence of the reduction of storage requirements due to minimization of file and software duplication. Even more importantly, such an integrated system provides a nucleus as well as an impetus for the development of the full scale personnel system should this be required. It will provide core data useful for manpower planning and personnel operations. Moreover, data made available for the personnel system will consist

of up-to-date information since an integrated payroll-pension system must be kept current in order to perform properly. A large number of functions of a personnel system[12] are automatically dealt with as by-products of the integrated system. As the result of the centralization of information and the provision of management with timely and accurate personnel data, staff costs are likely to decrease through improved coordination by management and by the reduction of non-essential clerical and manual work engaged on the payroll and pension functions.

An interaction chart for the proposed integrated payroll-pension system is given in Figure 1. Observe that, in the chart, the most critical point in the logic is when an employee terminates his service in the company through retirement, resignation, or death. The proposed system is designed to maintain each employee's record from the time of his entry into service up to his termination, and to continue to care for a retired employee's record until his death and the maturity of all his dependants' records. The next two chapters will discuss in detail the logical and technical structures of the proposed system.

---

[12] This is possible since the calculation of payments includes different types of earnings and deductions which are the end results of some functions in the personnel system. Often these functions are not major enough to require the development of independent application systems. Yet they are essential parts of the personnel system used for planning and administration. Since these functions are delicately connected to the payroll function, they can be easily incorporated as segments of the integrated system.

C H A P T E R T H R E E

THE SOLUTION : A LOGICAL OVERVIEW

In this chapter, an overview of the integrated payroll-pension system is presented. The following points should be noted when following the system decription. The personnel and computer environments of Company X satisfy all the pre-requisites for such a system as mentioned in the preceding chapter. Even more noteworthy is the fact that Company X has virtually no organized centralized personnel system at present. Therefore this system can serve as the nucleus for the company's future centralized personnel system. Not all institutions want or even need such an expanded system as the integrated payroll-pension system. Many companies have an existing payroll system and require only a supplementary pension system or vice versa. In these cases, the system can be run with one of the two functions suppressed. However, although this involves no wastage in file storage used, using only one segment of the system can be extravagant in terms of programming development investment. Finally, there are various approaches in data organization and file structuring that can be adopted. For example, a particularly attractive approach is to structure data into a single file with either hierarchical or relational organization of data.(12) In this approach, only one file is accessed during system operation. Moreover, programs can be compacted into a small number of large routines,

and generally various data base type techniques may be utilized. However this approach can be relatively expensive for a company with limited computer resources. It is due to this reason primarily that the system presented here is designed using a multifile approach. Data items are separated into major data groupings and subsequently structured into files. Nevertheless the system design is such that the files in the system can be simply grouped together to form part of the personnel data base when suitable computer facilities become available and the need to do so arises. Meanwhile, programs are segmented into parts primarily to make efficient use of available computer resources and secondarily to fit in with the system file structuring technique adopted.

3.1 Data Organization

The most important element of a system is its data. In order to be able to produce correct reports and analyse them properly, the data used toward this end must be both adequate and accurate. Like most systems, the integrated payroll-pension system requires a variety of data types. Since the system deals primarily with people, most of the data required in the system are information about company employees. The information is in the form of personal details, statistical data, job and salary histories, leave details, identity data, pension details, payrates, allowances, deductions, etc. Data of such a wide range are required if the integrated system is to perform more than the basic payment function and if it is to serve as a

nucleus for the further development of an upgraded personnel system. With such a range of data, the system can produce much improved and hitherto unavailable in-depth reports. For example, information such as job and salary histories are useful for salary administration and job analysis but are not directly involved in the regular salary payment function. In the integrated system, data of this type are generated from the system whenever they are needed, as for instance when promotions are being considered or during salary reviews. Leave information is another example of the type of data that does not participate actively in the monthly cycle of paying employees. It is used only when the necessity for surveying past leaves arises, such as when management wants to examine employee absenteeism or when a wastage analysis is to be performed to determine the amount of company time lost due to leaves.

Thus not all the data in this system are used every cycle. A basic assumption in the data organization of this system is that data can be divided into active data and dormant data. All information are grouped into files accordingly on the basis of this assumption. However, as in any efficient system, the predominant concern is to organize data in such a way as to provide minimum storage expense and maximum processing efficiency.[13] Hence, at times, the strict segregation of

[13] Like many other systems, this system groups data into records which are in turn grouped into files. The number of files created depends mostly on the nature of the information required from each data group, on their roles in the basic system functions, and frequently, as in this case, even on the computer resources allotted the system.

dormant and of active information is sacrificed in order to avoid duplicate storage of data.

Active data can be divided further into two groups: one group consisting of data which vary with each cycle and the other group comprising data items which vary at a rate generally slower then the cycle rate, though information in this group can be updated cyclically when required. In this system, data of the former type are grouped into records designated as transaction records and those of the latter type - the more stable data - are grouped into master records. Transaction records and master records are not in one-to-one correspondence. As a rule of thumb, for every master record there are several transaction records. Indeed, if the data were not separated, the resulting record would expand and shrink in each cycle according to the number of transactions contained in the record. This is particularly disadvantageous for a small computer with limited storage facilities. Moreover, separation of active data into two record type groups facilitates processing and data access. Since output can generally be classified into reports based only on earnings and deductions information which are classed as transaction record data, and those based mainly on personal information which are master record data, further subdivision of active data into transaction and master record data groupings means that these reports can be produced by accessing and processing only the data necessary for each report.

Dormant data are not as inert as the name might suggest.

Although the information already stored in dormant data files will not usually be altered, nevertheless the information can be updated. Furthermore, the volume of dormant data will be incessantly increasing but generally at a rate slower even than the rate at which master records data are altered. As with active data, dormant data are subdivided into several different record grouping as well. Here, the subdivision is based on the specific system functions making use of dormant data. Thus in this integrated system, dormant data are grouped into job-salary history records, dependant records, and leave records. Among these, the first two record types play significant roles in pension computations. Employee in active service will have at least a job-salary history record and possibly dependant and leave records also. For a retired employee, only his dependant record is monitored by the system, the job-salary history and leave records having been purged at the time of the employee's retirement.

A job-salary history record provides the entire job and salary history of an employee up to some given point in time. It accounts for all promotions and salary increases during an employee's service lifetime as well as any exceptional type of allowance granted the employee. The dates on which promotions, salary increases or special allowances were granted determine the number of records required per employee. Alterations to the job-salary status on different dates correspond to different records. Moreover since the number of promotions and increases differs from employee to employee, different

record volumes are usually needed to provide complete job and salary histories for different employees. As has been mentioned earlier, job-salary history records are significant for forecasting salary requirements and job vacancies, planning career paths, in addition to providing the basis for computing the pension to be received by an employee upon his retirement, vis-a-vis the determination of his final average salary. Note that these are features that Company X needs, but is unable to obtain from its present system.

A dependant record plays an unusual role in the integrated system in that, though updated, it remains dormant so long as an employee, whether active or retired, is living. When an employee dies, his dependant record[14] changes status and becomes a record monitored cyclically by the system, participating actively in pension computations for dependant allowances. An employee's dependant record enables the system to judge whether or not the employee is survived by his widow[15] and to determine the number of dependants eligible for pension benefits. The information contained in the dependant record includes data which identifies the employee's surviving spouse - together with additional items kept primarily for actuarial statistics - and detailed information on employee's surviving children, for exam-

---

[14] That is, if the employee is survived by dependants at the time of his death.

[15] An extraordinary feature of many pension schemes is that survivor benefits are granted only to female spouses.

ple, their names, sex, birthdates. These data are used to determine which members of the family are eligible to receive survivor benefits and to calculate the amounts of the pension allowances involved.

Employee leave information is kept in two locations. An employee's master record contains only the current leave status of the employee. Details of previous leaves such as dates and lengths of leaves, and other leave related information are kept in leave records, with a different record representing each previous leave taken by the employee. Leave records are stored for a limited period, the period to be decided by the user, after which the information in the records are transferred to print or tape for future reference.

Five basic record types have been identified so far: master records and transaction records, both active types, and job-salary history records, dependant records and leave records, all dormant types. All records of these record types which pertain to an employee have the employee number as the record key, a key that is common to all five record types. Therefore if the computer resources available to the system permitted a single data file system design, packing all of the records into a single file storage area would be straightforward. However, since the computer environment of Company X precluded a single file system, the various record types are grouped into multiple files with all records of the same type being entered into the same file. This grouping results in five files: Master File,

Transaction File, Job-Salary History File, Dependant File and Leave File. Since records pertaining to a specific employee are identified by a common key, namely the employee number, records in one file are linked naturally to corresponding records in the other files.

Apart from the files mentioned above, the system makes use of three more data files - the Loan File, the Bank Parameter File, and the Transaction Parameter File. The Loan File differs from pevious files in that each record in the file is defined primarily by a unique, user assigned loan number instead of the employee number which is retained in the file as a minor key used mainly to link loan records with records in the other files corresponding to the same employee. Each record in the Loan File represents a loan granted by the company to a specific employee. It contains information such as the size of the loan, the date the loan was issued, the interest rate charged, the loan repayment schedule, and the outstanding balance, with the last item kept current every cycle by the system. Furthermore, the user can decide on the types of loans to be included. For Company X, the loan types consist of personal loans, study loans, and car loans. Other users presumably will be working with different loan type sets. The system uses the information contained in the Loan File to monitor the loans issued by the company to each and every employee. By means of the payroll segment of the system, the Loan File is used to account for, as per repayment schedules, all deductions from employees' salaries correponding to loan repayments and arrears, and to reflect

these deductions in the printed payslips. Each loan record is updated cyclically during the system run so long as there is a positive unpaid balance on the loan. Finally the Loan File can also be used to determine all loans incurred by an employee within some user specified duration as well as the frequency at which employees ask for company loans. The Bank Parameter File is used by the system to facilitate the direct payment of employee salaries into employee accounts in banks and building societies for employees who so choose. The bank parameter refers to the list of the different banks, their branches, and building societies where these employees and pensioners are depositors or clients. Information stored in this file consists of bank and building society codes, bank and building society names, corresponding addresses, and bank clearing numbers. The Transaction Parameter File is used by the system in each payment cycle to identify the types of all deductions from and all additional earnings to the employee's basic salary. The transaction parameter[16] refers to the list of all possible types of employee earnings and deductions involved in the system. In the system, there are at present 64 transaction types in the list, each of which is identified by a unique code. These codes are significant since each code represents a distinct programming routine in the system Some of the codes require more complicated processing than others, but all the codes are used in producing payslips and in generating

[16] Another name for transaction parameter might be earnings-deductions parameter. I have chosen the former because of its brevity.

the reports needed for accounting purposes. A detailed listing of these codes is given in Appendix C.

3.2 System Operation

The integrated payroll-pension system has three distinct run modes. These are the set-up mode, the standard mode, and the exceptional mode. The set-up mode is designed to be for one time use only. Its objective is to create all the files required to render the system operational. Once the system is up and so long as it remains operational, the set-up run mode need not be re-activated. The set-up run is initiated by the user feeding data punched through either cards or magnetic tapes into the system[17]. The input segment of the system is so designed that each set of data is assigned a specific input format, with the effect that all data encountered are instantly identified and properly channelled for processing and subsequent slotting into their proper places in the system storage areas. However before being entered into their designated files, the data undergo a basic system check, namely data validation, which ensures that only clean and accurate data are passed on to the file storage areas. Only after successfully passing the validation test are the data processed. The system identifies all validated data, sets up the relevant records to which the data belong, and enters the records into the proper files. When

---

[17] Payroll processing is a batch processing operation. A batch-controlled system, such as this integrated system, can be defined as a system in which it is necessary to accumulate related data usually input in sequential order, and to process this data simultaneously, usually involving the update of the various files as well.

all the data from cards or magnetic tapes are exhausted, file creation will have been completed. (see Figure 2) At this stage, the system becomes operational, and the files are ready to be used for runs in the other run modes. Moreover, all subsequent changes, be they addition of new data or alterations of existing data, to the data files are handled through the standard run mode.

The standard run relates primarily to the payment function of the system apart from dealing with new data input and file maintenance functions as mentioned above. The payment function refers to all salary and pension calculations required by the user, related file updates, and the production of standard reports. As indicated earlier, the system is capable of dealing with 64 different types of payment transactions. These transactions correspond to different processing streams, each of which involves a distinct programming routine. It should be stated here that the system makes no claims about the uniqueness of the algorithms used to tackle transactions and solve problems. Frequently, there are many ways of approaching the problem. In this system, the algorithms are chosen, bearing in mind the system design and data organization, to provide maximum efficiency in data usage and greatest user flexibility.

Finally, the exceptional runs include all non-set-up functions that are not covered in the standard run mode. In general, not all of these functions need be activated every cycle. Examples of exceptional runs may include data validation only

without any further processing, loan listings, leave analysis or job-salary reviews. The system allows the user to specify these listings to include all employees, only specific employees, or particular groups of employees. These employee groups can be chosen according to sex, age, race, salary level, position in th company, department codes and/or length of service in the company.

The run mode in which the system is to operate and the transactions it is to process are determined by sets of job control cards. The set-up run mode must be run independently from the other run modes whereas the standard and exceptional run modes can be combined to constitute a fourth run mode - the combined mode. Each combined run requires its own modified set of job control cards. The set-up run and many of the exceptional runs make use of only a limited number of components of the system. The system components are structured so that system operation proceeds in four logical phases: (1) data validation, (2) file preparation, (3) transaction processing, and (4) output. In the validation phase, all data entering the system are checked for errors. Only data which pass the validation test are used in the file preparation phase. Validated data are then forwarded to the system storage areas for entry into files. This is done by passing the data through a series of logical option checks. Each option consists of a series of algorithms which identifies the data, thereby enabling the system to determine the files to be accessed and prepared, as well as the

transactions to be processed. The second phase is completed when all files required for transaction processing are ready. In the set-up run, only these two phases are activated. For some of the exceptional runs, as when simply validating data, even the file preparation phase is not used. However, many of the exceptional runs, like the standard run, make use of the transaction processing and output phase as well. In the third phase of system operation, the transactions identified in the preceding phase are dealt with and the previously prepared data files are processed to produce updated files and reports, which are functions of the final phase. The main processing streams are described more fully later in this section. The output phase involves final updating of files and the generation of all specified reports. The latter is achieved by means of a set of output programs which compiles the information required by the reports from the updated files.

A more detailed overview of the four operational phases of the system is indicated in the system logic chart (see Figure 3). For ease of description in the text that follows it is assumed that data enter the system in the form of punched cards. Eight possible card types are catered for in the system. Each card type has a fixed set of possible card formats which enables the system to identify the nature of the data contained in each card. Card type I signals addition and maintenance of master and dependant records. Card type II is the termination advice which is required in the event of employee retirement,

resignation or death. Card type III is the loan advice which indicates the need to access the Loan File and either add a new record to the file or update information in an existing record in the file. Card type IV is the leave advice which instructs the system to access the Leave File for addition of new records or maintenance of existing ones. Card type V is the transaction card and indicates to the system which of the 64 payment transactions the system is to process. Card type VI is used to maintain existing job-salary history records. Card type VII contains bank parameter data and informs the system that the bank parameter listing is to be updated. Finally, card type VIII contains transaction parameter data to be used for amending the transaction parameters. Note that if new transaction parameters are added to the list, then new transaction algorithms, one per new parameter entry, must be added to the system transaction porgram routines. These eight different card types are handled differently within the system even as early as the data entry and validation phase.

When type I cards are read, entries on the punched cards indicate whether new records are to be added to the Master or Dependant File, or if existing records in these files are to be maintained. Some of the more important updates include changes in the basic salary, in the number of children, in the job designation, and in the marital status. Each of these changes triggers off a series of algorithms in the system. For example, a change in job designation affects the Job-Salary

History File. If the change in job status is accompanied by a change in the basic salary, like other changes indicated above, the job status alteration affects the income tax, pension and medical aid positions, and requires recomputation of the tax deduction, pension contribution, and possibly medical aid contribution. These algorithms can be quite lengthy. Consider the pension contribution calculations. When an employee's marital status changes, the system first checks the sex of the employee. If the employee is female, the pension contribution is not altered.[18] For a male employee, the alteration depends on his age at the time he joined the pension plan. The system determines the new pension contribution due, accesses his pension contribution transaction record, and changes the amount to be deducted from his salary. Now a change in the marital status effects a change in the tax status. The factors considered for the income tax deduction are, apart from the employee's income, the employee's age,sex[19], number of dependants and marital status. There are various ways of computing the monthly income tax deduction, all of which aim to free the employee from having to pay out an additional huge sum at the end of the tax year. The algorithm developed in this system takes into account personal details given by the employee and is such that the total income tax deduction accumulated over the tax year equals the

---

[18] This is a pension rule of Company X. Other pension plans may have different rules for female employees. Usually such changes correspond to a simple revision in the algorithm.

[19] Married males and married females have different types of income tax deductions under South African law.

annual tax deduction prescribed by the rules of the internal revenue office (see Appendix A for further details).

When type II cards are read, the system activates a string of algorithms for pension calculations. This card type can indicate three things, depending on the classification code entered in the punched card: a) death, retirement, or resignation of an employee, b) death of a pensioner, c) death of a pension widow. When an employee resigns from service he is entitled either to withdraw his total pension contribution with appropriate interest added, or to defer this amount until he reaches the retirement age, in which case the amount will be transferred to another fund used specifically for this purpose. The cases in this particular fund are treated on a case by case basis; therefore its details cannot be and are no longer part of this system. When an employee retires from service, the amount of his pension is based on the type of retirement under which he is classified. The different types of retirement are ill-health retirement, late retirement, early retirement, and normal retirement. In all these types, the employee's annual salary for all years are needed to calculate the final average salary from which the annual pension is determined. Therefore the job-salary history record of the employee is accessed. It provides the details of the salaries and bonuses that are needed in the computations. The same algorithms are used when an employee dies during service. The system treats this case as if the employee had resigned in ill-health. In addition to this, the

widow's pension and/or dependant allowances are determined using the calculated annual pension and the information contained in the dependant record of the deceased employee. When a pensioner dies, his widow's pension and/or dependant allowances are calculated directly since the pension received previously by the deceased pensioner will have been computed in an earlier cycle and recorded then. When a widow on pension dies and there are dependants who survive her, the allowances of these dependants will be increased to specified amounts (see Appendix B for further details).

When type III cards are read, the Loan File is accessed either for maintenance of existing records or for addition of new loan records. Maintenance on the amount of the loan issued and on the corresponding repayment value will cause the transaction record catering to this deduction to be accessed and the deduction amount on the record changed. A change in the interest rate on the loan is unlikely; but when it occurs, this change is reflected in the loan record and will be taken into account later during payslip computation when loan records are updated simultaneously. New records to be added are simply entered into the Loan File after passing the validation check. Simultaneously the system generates a transaction record for the employee specifying the amount to be deducted.

When type IV cards are read and maintenance is signalled, the leave records to be maintained will be accessed and the corresponding changes made. Otherwise, for new records,

the system first deducts the number of leave days taken, as indicated in the punched card,from the accumulated total leave due the employee which is recorded in the Master File in the field corresponding to the type of leave indicated in the card. When this is completed, the new record is written into the Leave File, and depending on whether or not a salary advance is made, a record is generated for the advance in the Loan File, in which the advance is treated as a special kind of loan. Simultaneously a transaction record is generated to account for the appropriate loan deduction.

For a standard payroll system, type V cards contain the most important data. The data on these cards consist of information on the various amounts to be added to and/or subtracted from salaries, and codes indicating which computations are to be performed by the system. As mentioned previously, there are 64 transaction codes currently allowed in the system. The codes include earnings, allowances, and deductions of different types. In addition, each code admits five possible input variations. These variations are identified by the variation codes - N, T, C, Z, Q. When an N variation is read, a new transaction record is created. The input card contains all the information required in setting up the new record. For a T variation, there are two possible alternatives: either a one time use new record is set up or an existing record is temporarily suspended, superceded meanwhile by the data in the T variation record for one run only, and then reinstated in the

next run. The choice of alternatives depends on whether or not there is an existing record of the same transaction code in the Transaction File under an identical record key. If not, the former option holds; otherwise the latter option is chosen. For a C variation, the system is instructed to access an existing record in the Transaction File and replace the value in the record by the new value entered in the punched card. When the system identifies a Z variation, it will fetch the specified existing transaction record and cancel the transaction permanently. Finally when a Q variation is encountered, a new transaction record is created and the system is alerted that instead of the user entering an actual amount, some other quantity is input which the system must then use to calculate the monetary equivalent.

When type VI cards are read, the Job-Salary History File is accessed only for maintenance of existing records. Changes in existing data can occur when an employee's job or salary has been misrepresented and needs to be corrected.

When type VII cards are read, the system prepares the Bank Parameter File for possible changes in existing records or addition of new bank parameter records, depending on the entry code input. Changes can be made on every item of information, except for the bank code, in any existing record in the file. Whenever a new bank code replaces an old one, a new bank parameter record must be entered. The old record is then removed by an additional entry changing the bank name of the

record to blanks.

When type VIII cards are read, the system likewise prepares the Transaction Parameter File for possible changes in existing records or addition of new transaction parameter records. Changes of data in existing records occur infrequently since each record consists of only the transaction code and the transaction name, and only the transaction name can be changed this way. The transaction code can only be changed by entering a new transaction parameter record. The old record is then marked for deletion by an additional entry changing the transaction name to blanks.

For an erroneous card entry, the entire card is displayed on the printer and an appropriate message is printed stating the types of errors found if they can be properly identified. Although the system can be instructed to exclude erroneous entries and proceed with transaction processing on validated data, the proper operational procedure is for the user to correct these errors and re-enter the corrected data set for revalidation and file preparation before allowing the data to undergo transaction processing.

After the transaction processing phase is completed, the output phase commences. There are two system functions in this final run phase: file update and report generation. Files which are updated are the Loan File and the Job-Salary History File. The loan balance in the loan record is reduced

whenever a deduction from employee earnings is made and credited towards the loan number. Updates of the Job-Salary History File involve accessing the Master File records which contain changes in job, salary, and/or special allowances, and entering these as new records into the Job-Salary History File. Finally, the system produces a report file containing payslip information for use by the Budget System of Company X[20].

In addition to the exception reports requested by the user, the system generates two basic sets of reports. The first set is generated by the system automatically in every cycle. This set includes payslips, a deductions summary, a schedule of bank transfers, cheques, and reports on detailed earnings and deductions of employees, on reconciliation of net pay, on personal information including special allowances, on monthly deduction totals, and on administration totals for balancing. The fundamental output in this set is the payslip. Payslip printing requires accessing of the Master File and the Transaction File. The Master File contains personal information such as employee name, sex, race, job designation, and birth dates, as well as administrative data such as department codes and cost center codes. The entire Transaction File, already updated in the transaction processing phase, is processed record by record to extract all earnings, deductions, and allowances information of every employee for printing in the payslips.

---

[20] The creation of the report file is easily suppressed if it is not required.

Different totals consisting of monthly accumulations of diff-rent types of earnings and deductions are first updated in the Master File and the new totals subsequently printed in the payslips. Some of these are totals on taxable earnings, non-taxable earnings, pension contributions, income tax deductions, insurance and reimbursive allowances.

The second set of reports is produced only at diff-erent time periods and involves additional processing. The system tests for three periods - beginning of calendar year, end of tax year, and beginning of company financial year.[21] At the beginning of the calendar year, the system re-initializes the total medical aid contributions for the calendar year to zero. This action is necessitated by the maximum annual limit imposed on medical aid contributions in a calendar year. If, for example, a member's salary increases sometime during the calendar year, then at the increased deduction rate, his annual contribution toward medical aid might exceed the prescribed limit. The system skirts this problem by adding the contri-butions accumulated within the calendar year prior to the salary increase and the projected contributions at the modified rate of deduction and compares the sum against the maximum amount allowed. If the contributions exceed the limit, the system automatically adjusts the contribution amount by computing a new deduction rate for the remainder of the calendar year. At

---

[21] Three tests are required because the tax year, company financial year, and calendar year can be different. If the years coincide, the number of tests can be reduced and the report generation function merged.

the end of the tax year, two reports required by the government are produced - the workmen compensation report and the employee tax certificate. After these reports are printed, the system re-initializes the relevant totals from the Master File used for producing the two reports to zeroes in preparation for the next tax year cycle. Finally at the beginning of the company financial year, specific pension calculations are automatically performed. The pension policy of Company X stipulates that every employee is entitled to interest on accumulated pension contributions at a specified rate compounded annually at the beginning of the company financial year. All employee records in the Master File are automatically accessed and the accrued interest added to their total pension contributions to date. As presently designed, new employees who have been with the company for less than a year are excluded in this function.[22] After the three tests are made and the additional processing, if any, completed, the system prepares all files for the next cycle. As a security procedure, files are copied to their respective backup files in order to prevent data loss in case of computer failure.

---

[22] Again, this is a pension rule of Company X.

C H A P T E R F O U R

THE SOLUTION : A TECHNICAL PRESENTATION

In the preceding chapter, the logical structure of the integrated system was discussed and a general system overview was presented. This chapter concentrates on the more technical details of the system. Specifically, the presentation centers on detailed file interactions during processing and extends to generation of output files and reports. Output requirements play a major role in the design of this system in that within the limitations imposed by the availability of the users' computer resources, the design structure of the system is geared primarily towards facilitating the generation of output. A schematic summary of the system flow is given in Figure 4, and file dependencies are listed in Appendix E.

4.1 The Set-Up Run: Data Validation, File Preparation

The set-up run begins by converting data from source documents via punched cards into computer storage areas, which are usually in the form of discs or tapes. As indicated in the preceding chapter, the data must be pre-grouped into different card types and sorted into appropriate sets for entry into their respective files. The process is facilitated by requiring the card types to conform to specific formats so that the system can recognize the data easily. The first sets of data converted by the system are those belonging to the Master

and Dependant Files. These data sets are processed by programming routine PROGØ1[23]. PROGØ1 validates the data before entering them into their designated storage areas. The validation depends on the specific data item being tested. PROGØ1 may test to see if a data item is numeric or alphanumeric, whether it conforms to a definite set of codes, or whether it corresponds to some fixed value. For example, PROGØ1 verifies whether each employee number is a nine digit number, whether date of birth is numeric, that the sex of the employee is always denoted either by an M or F, that the method of pay is either '1' (for cash), '2' (for cheque), or '3' (by credit transfer), and that department codes correspond to the codes within a fixed list.

After undergoing the validity check, all erroneous data are listed for the user to correct whereas validated data are slotted into their designated locations in the files. PROGØ1 provides the user with the option to either prooflist without file creation or prooflist with simultaneous file creation. It is recommended that for the first run, data should be input simply for validation and should not be entered immediately into the files. This allows for erroneous data to be corrected and subsequently re-entered into the system. Con-

---

[23] Processing in the system is performed by several major programming routines, identified in this chapter by a six character code, consisting of the characters "PROG" and two numerical digits used to identify the routine sequence. The division of the system processes into several major routines is aimed at insuring that the limited computer resources available to the user will be adequate to accomodate the different processing requirements.

current with the creation of the Master File and the Dependant File, PROGØ1 also generates, for every master record, a set of records for the Transaction File. Each record set generated consists of an earning record for the basic salary and four deduction records, one each for income tax, medical aid contribution, UIF, and pension contribution. The transaction values in the four deduction records are initialized to zeroes. Non-zero values are filled in if appropriate only after transaction processing since their final values depend on various types of earnings found elsewhere. Thus in the set-up run where the transaction processing phase is not involved, the transaction values on the deduction records remain zero. Nevertheless, all five records in the set are generated whenever a new master record is encountered. The creation of the Master and Dependant Files is not the only responsibility of PROGØ1. PROGØ1 is also used during the standard run for other functions.

The next programming routine activated is PROGØ2. PROGØ2 is used for the conversion of the Job-Salary History File. Two options for file creation are available. The first option is to begin all the history records from the present, and the second option is to include past history records as well. If the first option is chosen, the newly created Master File is accessed and all information required by the job-salary history records are extracted from it. The information extracted includes items such as the present job status and the basic salary of every employee. In this case, there is one history record for each master record. The second option is somewhat

more complicated since the history record for each employee has to start from the time the employee joined the company. This means that every employee might have more than one history record. When this is so, additional inputs are required, namely data on the historical job-salary profiles of all employees. All data pertaining to the job and salary status of employees are validated by PROGØ2. PROGØ2 groups all records with the same employee number and sorts these records into chronological order. Only if the entire record set of an employee is valid does PROGØ2 enter the information into the Job-Salary History File. Like PROGØ1, PROGØ2 also allows prooflisting without file creation; therefore erroneous data are listed and can be corrected and re-entered into the system. When the entire history record set of an employee has been entered into file, PROGØ2 accesses the employee's master record and extracts from it the employee's current job and salary status, which are added to the record set. Apart from creating the Job-Salary History File, PROGØ2 also maintains all history records and applies indicated changes to the information in the file during the standard run. However, once the file is set up, it is not responsible for creating new records for the file, this being the function of PROG11.[24] Finally, the user has the option of having PROGØ2 produce exception listings of history records.[25]

---

[24] A new record for the Job-Salary History File is generated by PROG11 from the Master File whenever there are changes in a master record which affects an employee's job or salary status.

[25] However, it is recommended that the exception listing is produce at the end of the standard run instead so that all changes and updates on the file will be reflected in the print-out.

The Leave File and the Loan File are the next set of files converted in the set-up run. Data belonging to these files are validated by PROGØ3. Aside from verifying the entry codes of the input data, PROGØ3 checks to see that there is an existing master record for every record entered in each of the two files. Only entries keyed by valid employee numbers are accepted for entry into the files. So far as the Leave File is concerned, no records for other files or any other by-product are generated during file conversion. On the other hand, as the Loan File is converted, PROGØ3 generates for every loan that is active a deduction record for the Transaction File with the loan type and its repayment value as well as repayment schedule indicated. Like the previous programming routine, PROGØ3 can produce exception listings or reports on employee loans and leaves upon request.

When PROGØ3 terminates, PROGØ4 is activated for the conversion of the Bank Parameter File and the Transaction Parameter File. The routines for the conversion of these parameters are relatively straightforward. Validation tests are required for numeric checks on bank codes and clearing numbers for the bank parameters. For the transaction parameters, the transaction codes undergo different validation tests: for earnings, the validation is a numeric check, and for deductions, it is an alphanumeric check (see Appendix C). PROGØ4 is also responsible for the maintenance of these two files. Likewise it can produce a listing of the different

bank codes and transaction codes, together with corresponding descriptions of these codes.

Thus the set-up run makes use of four programming routines[26], PROGØ1 to PROGØ4, to convert eight data files. Of the eight files, seven files are converted directly from input data. The exception is the Transaction File. Records for this file are automatically generated by the system during the conversion of other files. Maintenance of existing transaction records and the direct addition of new transactions occur during the standard run under the jurisdiction of PROGØ6.

4.2 The Standard Run: Data Validation, File Preparation, Transaction Processing, Output

The basic operations in the standard system run[27] consist of the addition, deletion, maintenance and update of records in the eight system files. These operations involve a number of algorithms which are incorporated within the 12 programming routines activated during the standard run. In the discussion that follows, it is assumed that all required files exist, having been created in a previous set-up run. The standard run kicks off with the maintenance of master and deduction records by PROGØ1. Three processing options are provided in PROGØ1. The particular option chosen for each

---

[26] PROGØ1 - PROGØ4 are also used in the standard run. However in the standard run, the other programming branches in these routines which were not activated in the set-up run are utilized.

[27] The description given here includes the exceptional runs as well as the standard run.

input card depends on the entry code specified. The first option allows the user to add new records to or change information in existing records in the Master File. The data entered, be they new records or for maintenance of existing ones, are validated by PROGØ1. The programming branch of PROGØ1 used for validating new records is the same branch that is activated during the set-up run. Therefore, as in the set-up run, PROGØ1 generates concurrently with data validation a set of five transaction records for each new master record encountered. In the case when existing master records are to be maintained, PROGØ1 will either access and directly update all the transaction records affected by the changes in the corresponding master records or it will trigger switches in the master records which are used later during transaction processing. For example, take the case of a salary increase. PROGØ1 will replace the current salary of the employee by the new salary in the employee's master record. The corresponding earning record in the Transaction File is also changed without any additional input from the user. Since an increase in salary may affect the amounts to be deducted for income tax, pension, and medical aid, the corresponding deduction records must take cognizance of these changes as well. PROGØ1 reflects these changes not by directly updating these records but by triggering switches, which have been designed in the employee's master record, whenever these deduction records are involved in any calculation further down the processing stream. The same holds for changes which affect the job-salary history records. Switches are provided for changes in salary,

in job status, and in the special allowances received by employees.

The second option of PROGØ1 that the user can choose enables the user to add records to and maintain existing records of the Dependant File. The addition of new dependant records is handled in much the same way as the addition of new master records. For maintenance, the relevant dependant records are changed accordingly. If the changes concern dependants of an active employee or a living pensioner, no transaction record need be updated. However this may not be so when the record involved is that of a dependant currently receiving allowances. In this case, the only change to his dependant record which can affect records in the Transaction File is a change in the dependant's marital status or a change due to his death. These changes involves recomputation or reassignment of dependant allowances. They are taken care of by PROGØ5 and will be discussed later.

The third option a user may choose is to reinstate an ex-employee. Reinstatement involves resurrecting all of the employee's old records in the different files. Since records of terminated employees are kept for only one tax year, this is possible only if the employee is reinstated within the same tax year as the year of his resignation. Otherwise, his rehiring is treated within this system as a new recruitment and data must be entered to create a new set of records with a new employee number for the employee. When resurrecting records, PROGØ1 retrieves from the Master File, Dependant File, and Job-Salary

History File the relevant records. These records will have been deactivated since the employee resigned from the company. PROGØ1 reactivates these records by changing their status from closed records to open records. However, all transaction records pertaining to the terminated employee will have been deleted at the time of termination; therefore, insofar as transaction records are concerned, PROGØ1 treats a reinstatement like a new employee and generates his regular earning and deduction records. The remainder of the transaction records will have to be entered later in the run.

After PROGØ1, PROGØ2 takes over. As was mentioned in the set-up run, PROGØ2 will now take charge of only maintaining the job-salary history records for all changes. Subsequently PROGØ3, also encountered during the conversion of Loan and Leave Files, is activated. This programming routine provides for the maintenance of the loan and leave records and is responsible also for the processing of the termination advice. In the standard run, PROGØ3 functions in three stages.

In the first stage, it adds new records to and maintains existing records of the Leave File. The same routines used during the set-up run are used to add new records to the Leave File. Moreover, with each new record encountered, the master record of the employee taking leave is retrieved and the number of leave days taken is subtracted from the accumulated number of leave days of the relevant leave type due the employee. This procedure ensures that the leave information in an employee's

master record is kept current at all times. If the leave involves a salary advance, PROGØ3 generates a loan record for the Loan File and a corresponding deduction record for the Transaction File. The deduction record ensures that the employee who receives a salary advance is not paid again for the period of the advance. If the advance exceeds the expected salary for the leave period, deductions are extended until the balance is paid. Leave records need to be maintained when an employee cuts short or lengthens his leave, in which case the employee's master record must be adjusted as well. PROGØ3 sees to it that both adjustments are made. In its second stage, PROGØ3 sees to the addition and maintenance of records for the Loan File. When new records are encountered, repayment schedules are generated for the Transaction File. Maintenance of loan records occurs whenever an entry in the record is altered. For example, an adjustment to the loan value is reflected only in the loan record. But a change in the repayment value is also reflected in the corresponding transaction record in the Transaction File. In the third stage, PROGØ3 accessed the master record of the employee identified in the termination advice. The information contained in the termination advice is entered into the master record for further processing by PROGØ5. Furthermore, PROGØ3 deactivates the master record and checks the stop payment code on the termination advice to decide whether or not all transaction records corresponding to the deactivated master are to be deactivated also. The reason for such a procedure is to enable the user to stop

all existing transactions for a terminated employee and replace them with another set, if any. If the transactions are not cancelled, PROGØ3 sees to it that an earning record is generated corresponding to pay in-lieu-of all leave entitled the terminating employee. If at the time of termination, the employee has taken leave in excess of the number of days allowed, PROGØ3 generates an earning record corresponding to a negative earning to be deducted from the employee's total earnings during payslip generation.

The next logical step after all termination advice have been processed is the computation of pension related benefits.[28] These calculations must precede payslip generation since they are involved in payslip computations for pensioners. The programming routine which makes these calculations is PROGØ5. Only deactivated master records and their dependant and job-salary history records are required for pension computations. PROGØ5 provides for eight different types of calculations corresponding to the following termination categories: 1) normal retirement, 2) late retirement, 3) early retirement, 4) ill-health retirement, 5) employee's death, 6) pensioner's death, 7) pension widow's death, and 8) employee's withdrawal from service. The particular calculation required for each deactivated master record depends on the category of termination entered by PROGØ3 into the record when it was deactivated.

---

[28] Although PROGØ4 can be inserted at this stage, it is recommended that the bank and transaction parameters be maintained in an independent run. The user can suppress PROGØ4 in the standard run cycle by simply not entering the corresponding input cards.

For the first five termination categories, PROGØ5 first determines the final average salary on which pension benefits and dependant allowances are based. The algorithm involved compares the average annual salary for the past N years of service with the highest annual salary averaged over any M consecutive years and chooses as the final average salary the larger of the two averages. The user can specify different values of N and M for each of these five termination categories.[29] When it has determined the final average salary, the algorithm proceeds to compute the annual pension. Factors common to the five categories in this stage of the calculation are the length of contributory service and the length of recognized non-contributory service. Moreover for late retirements, the period worked after retirement age can be taken into consideration for computing additional benefits. For early retirements, the difference between the specified and actual retirement ages is taken into account for reducing pension benefits, should this be required. For an employee's death, PROGØ5 also calculates death benefits and the amounts for the widow's pension and/or dependant allowances, if any. These amounts are also based on the deceased employee's final average salary. Finally for all five categories, a commutation amount not exceeding a user specified percentage of the annual pension is allowed and calculated if required. For the sixth termination category, the final average salary and annual pension computations are bypassed since presumably

[29] For an example of how this algorithm is applied, see Appendix B.

these were determined at the time of the pensioner's retirement. Thus, when a pensioner dies, PROGØ5 proceeds directly to the calculation of death benefits, and after testing to see that the pensioner is survived by a widow and/or eligible dependants, widow's pension and dependant allowances are determined. In the seventh termination category, when a pensioner's widow dies, PROGØ5 computes the increments to the allowances due the orphaned dependants, if any. For the last category, when an employee withdraws from service, PROGØ5 first computes the interest on his pension contributions and adds this interest to the accumulated total. Subsequently, PROGØ5 deactivates the employee's dependant records. Furthermore, for all eight categories, transaction records are generated whenever indicated; for example, as in pension payments, dependant allowances, etc. Moreover, PROGØ5 creates an identical copy of every master record deactivated under any of the first five categories. The copies differ from the deactivated originals in that the record status is changed to that of a pensioner. At the end of the standard cycle run, only these copies are retained, with all deactivated masters purged from the Master File. For deactivated master records of the sixth and seventh termination categories, PROGØ5 reactivates these records when there are eligible dependants who are entitled to pension benefits. Finally, PROGØ5 also includes a checking routine which monitors the maturity of active dependant records. Once a dependant record reaches maturity, it is deactivated and replaced by the next dependant, if any,

who becomes eligible for an allowance.[30] PROGØ5 sees to it that the transaction record affected by the change is altered accordingly.

There are two ways of generating records for the Transaction File during the standard run cycle: either indirectly by activating PROGØ1, PROGØ3, and PROGØ5 as indicated previously or directly by entering new records to PROGØ6. All valid inputs to PROGØ6 are checked against existing records for possible duplication[31]. When duplicate records are found, the information on the new record overrides the information on the previous one. This is not simply a validation feature. It is designed to enable the user to alter existing transactions easily. For example, suppose the user decides to cancel an allowance a particular employee has been receiving regularly. To do this in the integrated system, the user need only feed a new record into PROGØ6 with the employee number and the same transaction code as the one governing the issuance of the specific allowance, but with the variation code changed from an "N" to a "Z". PROGØ6 will then automatically cancel the transaction. Inputs for which no duplicate is found among existing records are entered into the file as new records.

After PROGØ6 completes its tasks, the standard run

---

[30] This holds in the event that the pension rule specifies a maximum number of dependants eligible to receive pension benefits, and the number of dependants for the employee or pensioner exceeds the limit.

[31] PROGØ6 treats two transaction records as duplicates if they have the same record key and the same transaction codes, even though values in other fields may differ, for example, the variation codes.

cycle moves into the output phase. The remaining six programming routines in the system are concerned with payslip computation, the production of reports and the final updating of specific files. The primary output produced by the integrated payroll-pension system are the payslips of employees and pensioners. These payslips are the end product of PROGØ7, the payslip computation routine. PROGØ7 performs all calculations needed to determine the gross earnings, the total deductions, and the resulting net pay for each employee. Before going into the processing sequence of PROGØ7, it is useful to have a general picture of the information contained in a payslip (see Figure 5). The information in a payslip can be divided into four groups - personal details, earnings, deductions, and statistical data. The personal details are used to provide additional identification of the payslip recipient. Earnings and deductions are used in the computation of the final pay received and are displayed to inform the employee of the factors used to compute the final amount shown. Statistical data indicate to the employee or pensioner the different totals that have been accumulated under his account to date. Some of these totals are used for income tax computation and therefore are initialized annually. Others are simply records of outstanding loan balances. Only three files are involved in payslip production: the Master File, the Loan File, and the Transaction File. The Master File provides the personal details required in the payslip. The Loan File is affected whenever an outstanding loan is partially or fully repaid. As a rule, during payslip computation, every

loan deduction transaction record results in the updating of the corresponding loan record. For instance, the outstanding loan balance needs to be reduced and the total amount paid up, both in terms of capital and interest repayments, must be increased. The Transaction File plays an even more important role. All earning and deduction transaction records take part in payslip production.

PROGØ7 is designed to handle a master record all the way to payslip printing before processing the next master record. Since PROGØ7 refers to the Master File through the Transaction File, this processing procedure requires that the Transaction File be organized in a specific way. Thus in the Transaction File, all records with the same record key (i.e., pertaining to the same employee or pensioner) are grouped together. In addition, all earning records within this set are placed ahead of the deduction records, with the records sorted sequentially in ascending order of the earning and deduction codes. PROGØ7 begins by picking up the first set of records in the Transaction File, uses it to refer to the corresponding master record, and checks whether the master record is a new or a deactivated master. In both of these cases, pro-rata computation of various earnings, allowances, and deductions may be indicated as, for example, when an employee is terminated or when a new employee is hired in the middle of the month, in which case the employee is paid according to the actual number of days worked. However, note that not all transactions are subject to the pro-rata computation. Those treated regular include overtime, lump sum ter-

mination pay, unpaid leave, and pay in-lieu-of leave. If the master record is neither new nor deactivated, PROGØ7 proceeds to process the earning records sequentially. Here, deactivated earning records are bypassed and only the active ones treated. Earning codes are classified into nine groups. Each group occupies a specific slot in the payslip and is identified by a unique group name. The codes are so designed that codes belonging to the same group are in numerical sequence. Since earning records in the Transaction File are also ordered sequentially according to the earning code, PROGØ7 effectively processes each of these groups separately one after the other. Each earning code indicates a distinct processing routine, although in some cases the algorithms involved are fairly straightforward. The more complex computations are described in Appendix D, and the simpler routines are discussed below.

Group 1: Salary. There are 9 earning codes in this group. Four of the codes refer to different kinds of pensions and the rest refer to earnings of active employees. The four pension codes refer to pension amounts that were computed by PROGØ5 when the employee became a pensioner. Since these values are fixed for the pensioners' entire lifetimes, no adjustments are involved and no further computation is needed. The rest of the codes refer to normal earning, part earning, unpaid leave, back pay, and lump sum termination pay. The earnings indicated by these codes are usually expressed in monetary units. However, the system allows transaction values for part earning and unpaid leave to be expressed in terms of the number of days

involved, from which PROGØ7 derives the equivalent monetary values (see Appendix D). The salary indicated in an employee's payslip is obtained by adding together all transactions in this group. PROGØ7 keeps a running totals of these earnings for other uses such as income tax computation.

Group 2: Overtime. There are 4 types of overtime in this system: normal overtime, Sunday overtime, public holiday overtime, and Saturday overtime. The transaction values indicated by the first three types are usually entered in terms of a number of hours instead of in monetary terms. Algorithms are provided for converting these transaction values to monetary units so that they may be entered into an employee's payslip. Distinct algorithms are designed to allow the user to specify different rules for these different types of overtime. Since overtime work on Saturday is usually paid on a fixed basis and only selected employees are entitled to this allowance, no conversion algorithm is required and the transaction value must be entered in monetary units directly. The values of these 4 types of transactions are added to form the overtime earnings indicated in the payslip. As with Group 1 earnings, PROGØ7 also keeps cumulative totals of these earnings.

Group 3: Allowances. There are only 3 earning codes involved here: special pension allowance, location allowance, and specialist allowance. All of these values are user input. The first code refers to a supplement to the pension whereas the other two refer to employee allowances. The two employee allowance codes are included among the special allowance types

in the job-salary history record of every employee. Therefore changes in these transaction values affect an employee's history record. During payslip processing, PROGØ7 compares these values with values of the previous cycle which are stored in the corresponding master record. If any of the values differ, a switch is triggered in the master record and this switch will be used to create a new history record later on in the system run. In the meantime, PROGØ7 replaces the original transaction value with the new value specified. As before, these allowances are added together to form the allowances shown in the payslip. Again, PROGØ7 keeps independent cumulative totals for these allowances.

Group 4: Reimbursive Allowance. There are 4 earning codes in this group: entertainment allowance, car allowance, miscellaneous reimbursements, and moving allowance. The first two codes are also classified as special allowances in the job-salary history records and are dealt with in the same way as the last two codes in Group 3. The values referred to by the miscellaneous reimbursements and moving allowance codes are entered by PROGØ7 directly into the master record and will be used when producing the report on these allowances. Again current totals for the values represented in this group are kept by PROGØ7.

Group 5: Miscellaneous Taxable Payments. There are 4 earning codes involved here: production bonus, medical aid grant, travel grant, and pay in-lieu-of leave. Production bonus is also a special allowance included in the job-salary history records and is treated in the same way as the special allowances

found in Groups 3 and 4. Medical aid and travel grants require no particular processing. They are input directly by the user in monetary units. The transaction value represented by the pay in-lieu-of leave code can be given in either monetary units or in terms of a number of days. PROGØ7 includes an algorithm which calculates the equivalent monetary value of the transaction. Simultaneously, PROGØ7 sees to it that the number of days indicated is subtracted from the accumulated unused leave of that particular leave type recorded in the master record. Running totals for this group are kept.

Group 6: Miscellaneous Non-taxable Payments. This group includes 4 earning codes: two for pensioners, representing two types of dependant allowances, and two for employees, consisting of the exam bonus and tax free gratuities codes. Values for these transactions are user input or generated by previously discussed programming routines as, for example, the dependant allowances. Totals are likewise kept for this group of earnings.

Group 7: Annual Bonus. Two types of bonus form this group: employee bonus and pensioner bonus codes. A pensioner usually can receive an annual bonus only once, in his first year of retirement, at the beginning of the company's financial year. For employees, an annual bonus can be given at the end of each year. PROGØ7 automatically generates the bonus amount received by every employee, on the basis of the employee's salary and service record for the year. However, for each employee, the user can override the computed value by entering

another value for the bonus. This allows the user to increase or decrease the bonus issued to any employee at his discretion. The annual bonus computation is always indicated by means of a header card which contains the current date for the run cycle.

Group 8: Petrol Reimbursement. Only 1 earning code falls under this group - petrol reimbursement. The value is user input and requires no further calculation. No cumulative totals are kept for this transaction.

Group 9: Free Benefits. Only 1 earning code is involved, - free benefits. The transaction value is user input and does not need any other calculation. PROGØ7 keeps a running total for values represented by this code.

Deduction codes are divided likewise into 11 groups. Deduction codes belonging to the same group are arranged in alphabetical sequence[32]. As with the earning code groups, each deduction group occupies its own slot in the payslip. The ways in which PROGØ7 processes these codes are described below.

Group A: Pension. There are 3 codes in this group. Each code is meant to correspond to a different type of pension fund, usually involving different contribution amounts.[33] These codes are mutually exclusive, unlike earning codes where

---

[32] For example, codes in Group A are labeled by A1, A2, A3, whereas those in Group C, say, are labeled by C1 and C2.

[33] The choice of allowing these codes in Group A has been influenced partly by South African legislation on race separation, i.e., two separate codes for the pension funds for Blacks and non-Blacks, and partly by the pension fund environment of Company X, thus a third pension fund specifically for its field agents.

more than one code per group can apply to the same employee. Two factors can cause a change in the contribution value of the transaction indicated by a code in this group. Either earnings involved in the pension contribution computations are altered or some information in the employee's master record which affect the contribution are adjusted. If relevant, the latter is always indicated by a switch in the master record. PROGØ7 tests to see whether a new pension contribution is to be computed, does the new calculations, and replaces the old value by the new one in the transaction record. It also maintains a cumulative total of all contributions by the employee for pension.

Group B: P.A.Y.E. This group contains 3 codes, corresponding to three distinct income tax deduction schedules.[34] The algorithms which determine the values of the deductions indicated are based on specific government rules and regulations. The value can change whenever employee earnings change or whenever a change in the master record which can affect P.A.Y.E. is found. As with Group A codes, the master record has a switch for the latter possibility. For income tax deductions, PROGØ7 keeps an annual total of deductions and initializes the total at the start of each tax year.

Group C: Medical Aid. There are 2 deduction codes in this group: medical aid and medical aid above limit. When

---

[34] This again is a peculiarly South African feature. Three types of P.A.Y.E. computations are required: regular income tax, Bantu tax, and Rhodesian tax. As designed, PROGØ7 accepts all three codes for pensioners, but only the first two for active employees.

PROGØ7 finds a medical aid code, it knows that a percentage of the annual salary must be deducted for medical aid contributions and reads this deduction rate from its location in the transaction record. PROGØ7 will test a switch in the corresponding master record to see if this deduction rate needs to be changed and does so if this is specified. Recomputation of the deduction is required when for example employee earnings change and exceed some set value. For the special cases where the annually accumulated amount contributed towards medical aid is likely to exceed the maximum annual limit, PROGØ7 changes the medical aid code to a medical aid above limit code and thereafter applies a special deduction rate which ensures that the maximum will not be exceeded.

Group D: UIF Contribution. Two codes are included here, one denoting employee contributions and the other employer contributions. The computations for both are based on fixed percentages of specific earning totals, which have been determined earlier during the processing of earning codes. PROGØ7 changes the deduction values whenever the earnings which form the totals are altered.

Group E: Insurance Premiums. Two codes corresponding to different kinds of insurance premiums are in this group. The values to be deducted are user input, so no additional processing is required. PROGØ7 keeps a running total for the first code, but not for the second one. Two codes are provided primarily for flexibility. This allows the user to use one or the other or both codes depending on the types of

insurance plans that are made available to employees.

Group F: Policy Loan. Only a single code falls under this group. It caters for a special type of loan that is not usually treated in detail by payroll systems. The transaction value must be user input. PROGØ7 does not keep a current total of this value.

Group G: Personal Loans. The deduction codes in this group denote the presence of deductions from an employee's salary to loans of various types of company loans incurred by the employee. Four types of deduction codes are allowed for, corresponding to the following loan types: personal loan, insurance loan, car loan, and study loan. Usually different interest rates are charged for these different loans. PROGØ7 reads off the interest rate to be charged on an outstanding loan balance from the transaction record, accesses the corresponding loan record, computes the amount of interest that has accrued, and adds this amount to the outstanding loan balance to get a new balance. When the new balance is less than the regular amount to be deducted, the prescribed amount is ignored and only the outstanding balance is deducted. Since these deductions affect the loan records also, PROGØ7 includes routines which update the loan record, corresponding to each loan transaction processed. These routines update the outstanding balance and enter the total interest paid to date in the loan record. Moreover, PROGØ7 ensures that the outstanding balance for each loan type is printed in the employee's payslip.

Group H: Mortgage Bond. There are 2 deduction codes here: mortgage bond and insurance bond. Both of these values are user input and no cumulative totals are kept by PROGØ7.

Group I: Medical Aid Claims. There is only 1 deduction code in this group. The value again is user input and no totals are kept.

Group J: Canteen & Phone Calls. Two codes fall in this group, representing repayment of canteen and company telephone usage accounts. The values are user input and totals are not kept.

Group K: Miscellaneous deductions. There are 10 codes in this final group. When PROGØ7 encounters any of them, it will search for the transaction value in the record and list the value together with its code in the payslip. All of these deductions are user input except possibly the last one, advances deduction, which is more likely to be system generated. This code usually indicates the absence of an employee for a particular period and is designed to be processed last, since it will be used to produce a nil payslip should this be necessary. Recall that a salary advance is treated as a special loan and that its payment is regarded as a deduction. As this transaction is processed, the net pay defined as the difference between gross earnings and total deductions is computed. PROGØ7 then compares the transaction value with the net pay. If the net pay is greater than or equal to the transaction value, the difference between the transaction value and the net pay becomes the final net pay that is displayed in the payslip. However, if the net

pay is less than the amount to be deducted, the amount actually deducted is the value of the net pay with the outstanding balance carried over as an arrear for the next pay period, and a nil payslip is produced. In both cases, PROGØ7 reduces the outstanding balance in the loan record accordingly. Note that if the loan balance before processing is already less than the prescribed amount to be deducted, the balance replaces the prescribed value and deduction from net pay proceeds as described above.

When all the transaction records pertaining to an employee have been processed, the different earning and deduction totals are added to the original totals in the employee's master record. These totals are used primarily for tax purposes and in some instances for inclusion in reports for management. Additional processing may be required at this stage. For example, if a female employee changes her tax status within the tax year, two sets of the totals needed for tax purposes are stored and two tax certificates are produced by the system at the end of the year. The totals for the month in which the status change occured are split into two groups and involve a pro rata computation. There are however two totals fields which require special handling. Whenever PROGØ7 notes that cycle in the current month is the first cycle of the calendar year, it initializes the annual total medical aid contributions to zero and starts a new total. Likewise, when the current cycle is the first cycle of the financial year, PROGØ7 calculates

the interest due on the total pension contribution to date, adds the interest to this total while initializing to zero the total annual pension contributions which are kept only for the duration of the financial year. Note that two different totals are involved here. The reason for keeping two separate totals is to make sure that interest is not paid on the contributions in the current financial year in the event an employee resigns within the year. The annual pension contributions total consists of only contributions made in the current financial year. The pension contributions to date total is kept current only to the last financial year and is incremented at the first cycle of the succeeding financial year. Only when all relevant totals have been updated is the payslip of an employee generated. After the employee's payslip is printed, PROGØ7 enters a record into the report file provided for the Budget System[35] mentioned in the previous chapter. Subsequently PROGØ7 repeats the procedure for the next master record, until the entire Master File has been processed.

After PROGØ7 is run, outputs are produced by the remaining programming routines. PROGØ8 is responsible for producing all outputs, including lists and forms, required

---

[35] This report file is used for budget analysis and the estimation of future allocation of salaries. Thus the information carried on each record consists of total earnings, overtime, and allowances for each employee. However, this does not include pensions since these are governed by the pension fund and are no longer part of the company's budget.

for actual payment to employees and a set of payslip related reports for management. Specifically, PROGØ8 first generates three sets of outputs, one set for the company's Accounting Department, another set for employees, and the third set for the banks and building societies involved. For the Accounting Department, two lists are produced: a cash payment list which contains the employee names and numbers, and the cash amounts they are to receive in the current pay cycle; and a payslip summary list which includes the gross earnings, total deductions, and net pay of every employee and pensioner whatever the method of payment. For the banks and building societies, PROGØ8 provides each bank or building society with a set of credit transfer advice, a deposit schedule containing the names of employees, their account numbers and the amounts to be credited to their accounts as per credit transfer advice, and lastly a single cheque covering the entire credit transferred. For employees who are paid by credit transfer, confirmation copies of the advice are produced. And cheques are printed for employees receiving cheque payment. Only the Master File and the Bank Parameter File are accessed by PROGØ8. To facilitate processing, PROGØ8 sorts master records according to method of payment with credit transfer first, then cheque payments, and finally cash payments. Moreover, master records in the first group are further sorted by bank codes. PROGØ8 deals with credit transfers first. Credit transfer forms are printed for one bank at a time with reference to the Bank Parameter File. The cheques for the credit transfers are then printed together

with the cheques made out directly to employees and pensioners. The cash payment list is generated next. Then PROGØ8 produces the deposit schedule list, and finally, the payslip summary list.

Three reports are then generated by PROGØ8 for management. All three involve only the Master File. The first of these is a report on the personal details of active employees. The report contains employee details, the list of special allowances received by each employee, his basic salary, and the amounts he contributes for pension and medical aid. The next report is a schedule of reimbursive allowances. This report is arranged by department and within each department grouping by sequential employee numbers. The final report is an administration accounting report which contains a list of various totals for the current cycle.

The next set of reports is produced by PROGØ9 and involves the Transaction File, Master File, and Transaction Parameter File. PROGØ9 subdivides the Transaction File into two working subfiles: earnings and deduction subfiles. This splitting facilitates the printing of the reports. From the deduction file, the deduction summary is printed. Essentially, this report contains, for each deduction code, a list of all employee numbers included in the deduction with their corresponding deduction amounts. The Transaction Parameter File is used to include descriptions of the deduction codes. A listing of monthly deduction totals is printed subsequently. This is

simply a list of deduction codes with corresponding descriptions, taken again from the Transaction Parameter File, and the total amounts deducted from employees earnings. This report is followed by a list of detailed deductions for each employee. This list is arranged by departments, and all employees in a department are listed under the department. Also following every employee name is a list of all types of deductions made from his salary for the pay period together with the specific amount involved in each type. After deductions have been dealt with, PROGØ9 utilizes the other subfile - the earning file. Output from this file consists of a list of the detailed earnings of each employee in every department. The list prints the names and numbers of employees of every department, and under each employee, all earnings affecting his pay for the pay cycle are listed. The last report produced by PROGØ9 is the reconciliation of net pay. This involves the current Transaction File and the Transaction File of the previous cycle. The two versions are compared record by record and a list of discrepancies between the two files is printed.

The last reports produced by the system in a regular cycle run is a printout of arrears, capital balances, and interest paid on different loans. PROGlØ produces this report based on the information contained in loan records. The information utilized in each record consists of the loan balance, interest rate and total interest paid, amount to be deducted per cycle, total amount already paid, and arrears if

any. After producing this report, PROG1Ø deactivates all loan records with a zero balance. These deactivated records will be deleted from the Loan File at the end of the current tax year when all the other files are also purged.

After the reports and outputs regularly required in a standard system run have been produced, the Job-Salary History File and the Master File must be updated. PROG11 initiates the updating by accessing both of these files. At this stage, the Master File still contains deactivated records of employees who have been terminated in the current pay cycle. PROG11 scans every record in the Master File, deactivates the job-salary history records corresponding to deactivated master records, and tests switches in active master records to determine whether there are changes which affect the job and salary status of employee. If it finds a switch on, PROG11 creates a new job-salary history record accordingly. While the Master File is read, PROG11 deletes each deactivated record from the file and stores the deactivated master in an extension file. Records in the extension file are kept until the end of the current tax year; so long as an ex-employee is reinstated within this time period, his master record can be resurrected. Deactivated job-salary history records which contain relatively more dormant data need not be deleted at this stage. The Job-Salary History File is purged of obsolete records at the end of the tax year.

The final programming routine PROG12 has two functions: to prepare all files for the next run and to produce two reports

(but only if the current cycle corresponds with the end of the tax year). PROG12 writes all files to their backup to safeguard against the possible loss of data due to unforeseeable circumstances. If the current cycle is the end of the tax year, PROG12 purges all files by re-copying only the active records from the backup into the original files. At the end of the tax year, PROG12 generates two reports even before the backup files are created. These are the employee tax certificates and the workmen compensation reports, both of which are required under government regulations. The programming routine is capable of producing up to two tax certificates per employee. But this occurs only if the tax status of the employee has changed during the past tax year. Concurrently, PROG12 also re-initializes all relevant totals fields to zero in preparation for the next tax year. Note that if the current run cycle does not coincide with the end of the tax year, not all files are purged, the two reports mentioned need not be produced, and the totals fields are left unaltered.

4.3 Programming and Implementation Requirements

There are twelve programs involved in the integrated payroll-pension system. Each of these programs consists of several routines which participate in file creation, maintenance, updating, computation, report generation,and purging. These programs have been discussed in the previous section. All the specifications for these programs have been drafted by the author. The specifications require only polishing to be ready for pro-

gramming and are available from the author upon request. It is estimated that approximately six months are required to develop this system, of which roughly four months should be allotted for program coding, testing and debugging, and two additional months for system testing and file conversion.

Program coding can be logically broken down into four parts: (1) input data processing program which in this system are PROGØ1 - PROGØ6; (2) computations on deductions and earnings - PROGØ7; (3) report generating programs, including the preparation of payments and the cost distribution reports which consist of PROGØ8 - PROG1Ø; (4) final update and purging of files involving PROG11 and PROG12. The first two parts alone may take 2 - 3 months to develop. Since the programs are not static, numerous changes may be necessary to improve various aspects of the programs. The coding involved for the last two segments should be fairly straightforward.

Personnel requirements for the six months period consist of a systems analyst and 2 - 3 full time programmers. This estimate is not based on the EDP strength of any particular company but rather on the author's previous experience with systems development. Thus the estimated implementation time-table may vary somewhat in accordance with the efficiency rating of the user's EDP personnel. Ideally the systems analyst should be in charge of system development, supervision of programming jobs, development of new program specifications if required, and the revision of existing specifications if needed.

The systems analyst should also be responsible for system testing and file set-up. The programmer's responsibility is to code, test, and debug the programs prior to system testing.

This system requires two disk packs for file storage and backups. Tapes may also be required for storing old records which are no longer needed by the system in its standard operation, but which might be of use for future statistical analysis. The major part of the system is designed to run in a monthly cycle. File maintenance however may be done as often as once a day but not less frequently than once a month if there are changes. Other minor runs occur whenever exception listings or reports not produced in the cycle run are wanted. Once the system is operational, part-time clerical work is sufficient to maintain the system. The author's program specifications are intended for coding in COBOL primarily because this is the computer language used by Company X and therefore the language with which its EDP staff has had the most experience. There are other factors which suggest that COBOL be used. This system is a batch-controlled system requiring efficient access to copious amounts of data, and COBOL has been specifically designed to be used for such systems(13).

Following is a suggested approach for implementing the integrated system:

1) Review program specifications and revise if necessary for the particular user;
2) Code programs according to the sequence in which they have been numbered;

3) Coded programs are to be tested while new programs are developed simultaneously;
4) Systems analyst prepares data and control cards for system testing, and organizes the collection of live data required for the set-up run;
5) System testing should cover all conceivable data types provided in the system, including erroneous entries;
6) All possible options should be tested;
7) File conversion to occur only after the entire system is satisfactorily tested;
8) Parallel runs with the existing system is to be done for at least one cycle run, with corresponding outputs of both runs checked against each other for possible discrepancies;
9) The system can be made operational only if the parallel run is successful.

FIGURE 1 : THE INTERACTION CHART

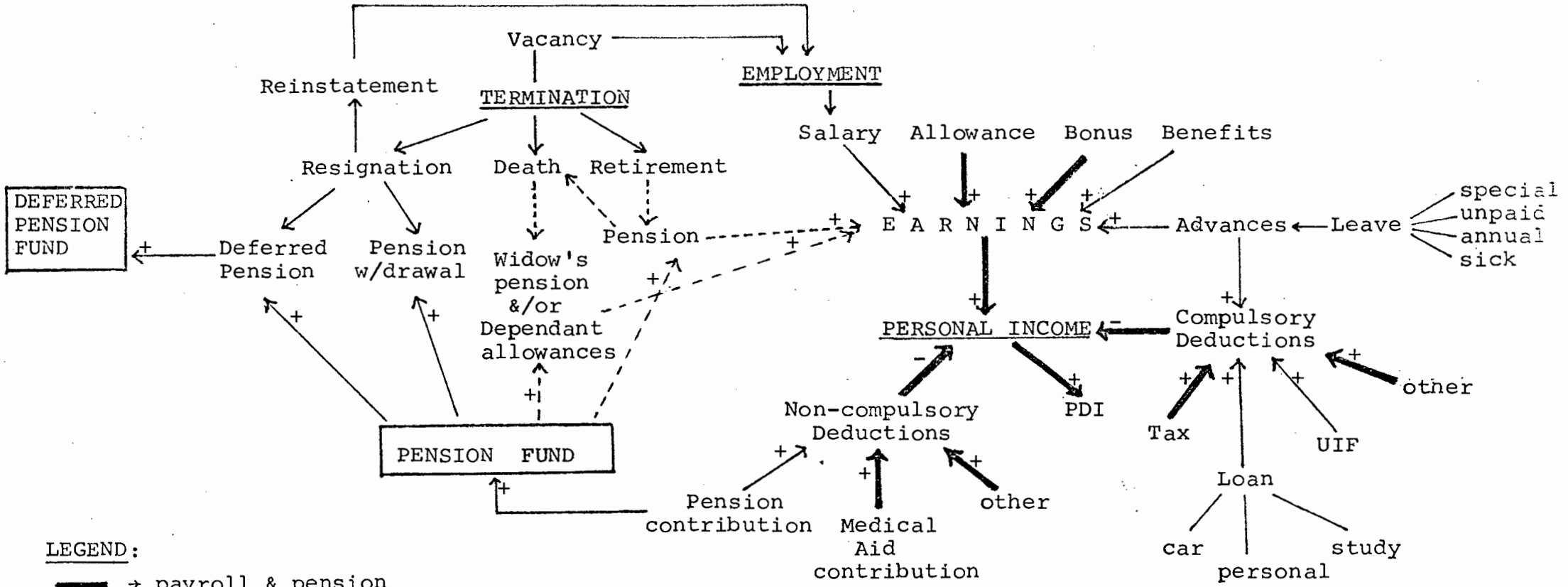


FIGURE 2 : THE SET-UP RUN - I/O REQUIREMENTS

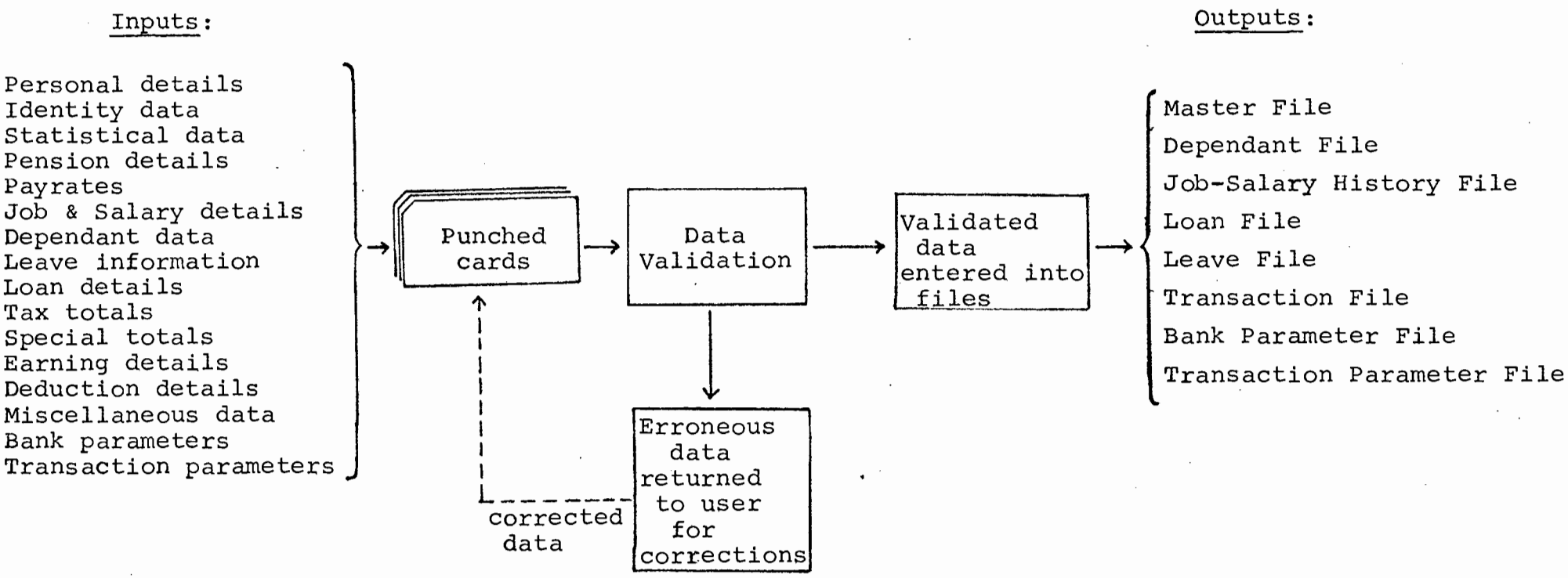
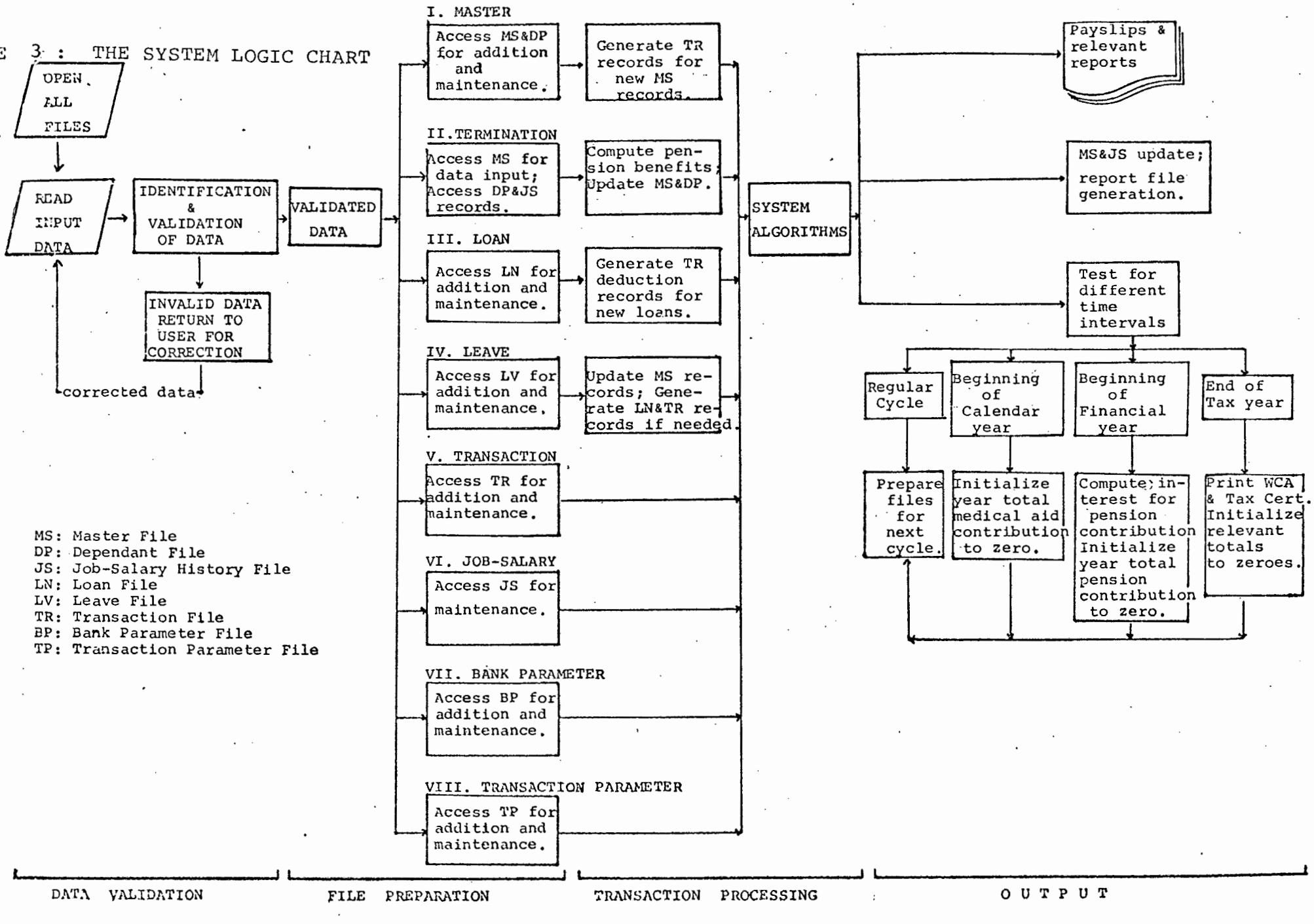
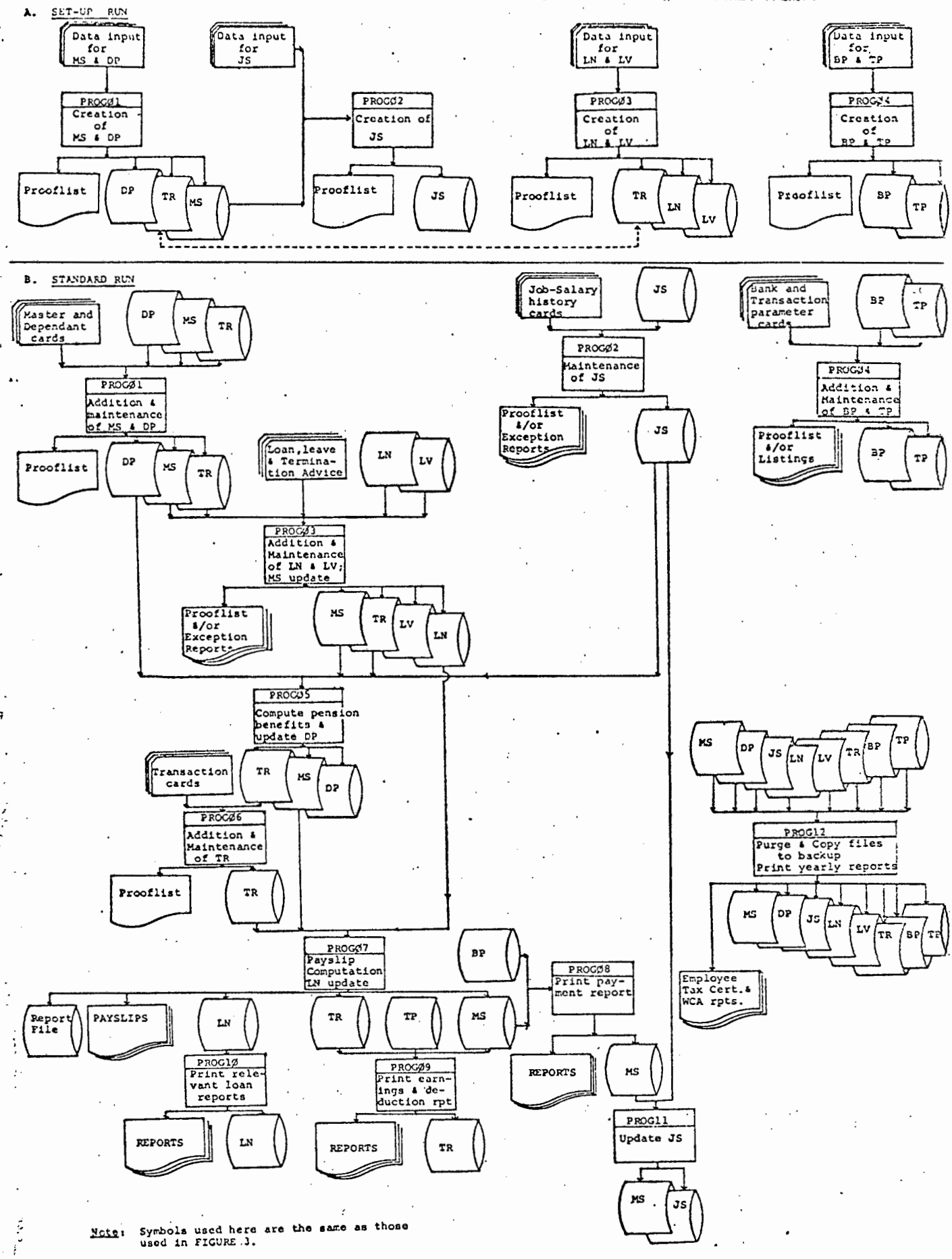


FIGURE 3 : THE SYSTEM LOGIC CHART



MS: Master File
DP: Dependant File
JS: Job-Salary History File
LN: Loan File
LV: Leave File
TR: Transaction File
BP: Bank Parameter File
TP: Transaction Parameter File

FIGURE 4 : THE SYSTEM FLOW CHART



Note: Symbols used here are the same as those used in FIGURE 3.

FIGURE 5 : THE PAYSLIP FORMAT

DATE

TAX REF. NO.	PROV.	RACE/SEX	STATUS	CHILDREN	BIRTH DATE	TAX WK.	DEPT.	OCCUPATION	EMPLOYEE NAME	EMPLOYEE NO.

EARNINGS

SALARY	OVERTIME	ALLOWANCES	REIMBURSIVE ALLOWANCE	TAXABLE PAYMENT	NON-TAXABLE PAYMENT	ANNUAL BONUS	PETROL REIMBURSMENT	FREE BENEFITS			GROSS EARNINGS

DEDUCTIONS

PENSION		P.A.Y.E		MEDICAL AID		U.I.F.		PREMIUMS		POLICY LOANS		PERSONAL LOANS		MORTGAGE BOND		MEDICAL AID CLAIMS		CANTEEN & PHONE CALLS				TOTAL DEDUCTIONS
CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	CODE	AMOUNT	

STATISTICAL DATA

TAXABLE EARNINGS	NON-TAXABLE EARNINGS	REIMBURSIVE ALLOWANCES	PENSION FUND	MEDICAL AID	TAX PAID TO DATE	STUDY LOAN BALANCE	CAR LOAN BALANCE	PERS. LOAN BALANCE	U.I.F. PAID TO DATE	INSURANCE	NET PAY

KEY TO MISCELLANEOUS DEDUCTION CODES:

K1	CAR LEASE	K6	RATES & TAXES
K2	CHARITIES	K7	RENT
K3	DEPOSITS ACCOUNT	K8	SPORTS CLUB
K4	NBS SUBSRIPTION SHARES	K9	TRAVELLING EXPENSES
K5	PERSONAL ACCOUNT	Z9	ADVANCES

A p p e n d i x A

CALCULATION OF P.A.Y.E.

The following is a general explanation of how PAYE is calculated in this system*. Four totals that are relevant to this calculation are held for each employee. These are:

TT-TAXABLE-EARNING - total monthly taxable earnings accumulated from the beginning of the current tax year to the previous month.

TT-PENSION-CONTRIB - total pension contributions from the beginning of the current tax year to the previous month.

TT-BONUS - total taxable bonuses from the beginning of the current tax year to the previous month.

TT-PAYE - total tax paid from the beginning of the current tax year to the previous month.

Aside from these four totals, there are three current month transaction sums needed for the calculation. These are:

TT-B - current month taxable earnings.

TT-C - current month taxable bonus.

TT-F - current month pension contribution.

The calculation is performed in three parts.

A. 1) Add TT-BONUS, TT-C, and TT-TAXABLE-EARNING.

2) Subtract TT-PENSION-CONTRIB from this sum.

---

* The calculation is performed in PROGØ7, the payslip computation routine.

3) Multiply the difference between TT-B and TT-F by the number of months remaining in the current tax year.

4) Add this amount to the result of 2). This gives the annual taxable earnings of the employee.

Summarizing:

Annual Taxable Earnings = (TT-TAXABLE-EARNING + TT-BONUS + TT-C) - TT-PENSION-CONTRIB + (TT-B - TT-F)(12 - (K - 1)),

where K = current tax month*

and $1 \leq K \leq 12$.

B. 5) The system looks up the annual taxable amount† on the tax tables and finds the annual tax to be deducted on this amount.

C. 6) TT-PAYE is deducted from the annual tax.

7) The resulting amount is divided by the number of months remaining in the current tax year.

Summarizing:

$$\text{PAYE (this month)} = \frac{\text{annual tax - TT-PAYE}}{12 - (K - 1)} .$$

---

* Month 1 in the South African tax year is March.

† Annual tax is not calculated on the basis of annual taxable earnings but rather on the basis of the annual taxable amount. This is the amount remaining after deducting the abatements from the annual taxable earnings as described in the PAYE Tax Manual. Married women are not included in this computation. Their annual remunerations are subjected to a different form of calculation which does not involve the tax tables. Instead the calculations consist of a set of standard algorithms provided in the Tax Manual. These algorithms are incorporated in PROGØ7.

Appendix B

PENSION COMPUTATIONS

The calculation of annual pension benefits and other pension related benefits are initiated by an employment termination advice. There are eight possible cases of pension computations, all of which are given below. The calculations depend primarily on the final average salary of the employee. The complex algorithm for computing the final average salary is presented prior to discussing the calculations for the eight pension cases.

Final Average Salary (FAS) Computations

The objective of the computation is to set the final average salary equal to the average of the employee's salary in his last three years of service* or the highest average salary from any five consecutive years, whichever is the larger amount.

1. Find MN = total number of months worked. MN $\geq$ 6Ø
2. Generate a Monthly Salary Table {S} from the job-salary history records of the employee. Each salary S includes the basic earning plus the production bonus. Production bonus occurs on a quarterly basis, and therefore must be spread evenly over the three months in the quarter before

* Except in the case of an ill-health retirement or an in-service death, where only the last year's salary is used in lieu of the last three years' earnings.

adding the result to S.

3. Compute the average monthly salary $\bar{S}$ over the last 3 years.

$$\bar{S} = \frac{\sum_{i=j}^{j+35} S_i}{36} \quad , \text{ where } j = MN-35.$$

4. Find the difference D between each monthly salary S and $\bar{S}$.

$$D_i = S_i - \bar{S} \quad , \text{ where } i = 1, 2, 3, \ldots, MN-35.$$

A Difference Table {D} is generated with D occurring MN-35 times.

5. If all $D_i \leq \emptyset$, FAS = $\bar{S}$ x 12.
This means that the average of last 3 years' earnings is higher than the average of any 5 consecutive years' earnings.

6. If there are some $D_i > \emptyset$, (which implies that there are some $S_i > \bar{S}$),

   a) Compute the sum of the differences D_i taken for each 6Ø months.

$$\bar{D}_i = \sum_{i=k}^{k+59} D_i \quad , \text{ where } k = 1, 2, 3, \ldots, MN-59.$$

   b) Search for the largest $\bar{D}_i$. The first entry will be the month i from which to compute FAS, (i.e., salaries of months i to i+59 will be used for FAS computation).

   c) FAS = $\bar{S}$ x 12 , where $\bar{S} = \frac{\sum_{i}^{i+59} S_i}{6\emptyset}$.

Here is a suggested method for doing 6.a and 6.b using COBOL subroutines. Assume that all underlined names have been defined in the Working-Storage Section of the program.

```
    MOVE 1 TO I.
    MOVE Ø TO SUM.
Ø3  ADD D(I) TO SUM.
    ADD 1 TO I.
    IF I IS EQUAL TO 6Ø NEXT SENTENCE ELSE GO TO Ø3.
    MOVE Ø TO DIFF.
    MOVE 1 TO I.
    MOVE 6Ø TO KK.
    MOVE Ø TO DB.                (DB is the same as D in 6.a)
1Ø  ADD SUM, D(KK) TO DB(I).
    SUBTRACT DIFF FROM DB(I).
    MOVE D(I) TO DIFF.
    MOVE DB(I) TO SUM.
    ADD 1 TO KK.
    ADD 1 TO I.
    IF I IS EQUAL TO M NEXT SENTENCE ELSE GO TO 1Ø.  (M = MN-58)
    MOVE Ø TO DBB.
    MOVE 1 TO I.
19  IF DB(I) IS GREATER THAN DBB NEXT SENTENCE ELSE GO TO 22.
    MOVE DB(I) TO DBB.
    MOVE I TO JJ.
22  ADD 1 TO I.
    IF I IS GREATER THEN 6Ø NEXT SENTENCE ELSE GO TO 19.
    EXIT.
```

The final JJ identifies the first month from which to compute the best FAS (i.e., salaries from month JJ to JJ+59 will be used for FAS computation). Hence, in 6.c, JJ represents the i in the equation.

Annual Pension (AP) Computations

Symbols used:

FAS - Final Average Salary
AP - Annual Pension
AWP - Annual Widow's Pension
DA - Dependant Allowance
LSB - Lump Sum Benefits

N - length of contributory service (from date joined to age 6Ø).
N'- length of recognized non-contributory service.
N"- length of contributory service after age 6Ø.
N_1- length of additional contributory service.
N_2- length of additional non-contributory service from age 25.
M"- number of months in N".
M'- number of months from date retired to age 6Ø.
n - number of eligible dependants.

Note: Retirement age = age 6Ø.
All lengths mentioned are defined as the difference between two inclusive dates given in the data files.

1. Normal Retirement

a. Calculate FAS based on the higher of:

i) last 3 years average

ii) any consecutive 5 years average.

b. Test for employee's final job status and compute for AP.

i) If job status is lower than Head-of-Department,

$AP = FAS(N/6Ø + N_1/6Ø + N'/12Ø)$, where ( ) $\leq$ 4Ø/6Ø.

ii) If job status is Head-of-Department or above,

$AP = FAS(N/6Ø + N_1/6Ø + N'/12Ø + N_2/12Ø)$, where

( ) $\leq$ 4Ø/6Ø, and $N_2 \leq 7$ and the lesser of

(1) and (2).

(1) $\frac{\text{age 6Ø date - entry age date}}{3}$,

(2) $\frac{\text{entry age date - age 25 date}}{2}$,

where entry age date = date of joining - N_1 - N'.

c. Test for possible commutation of pension:

i) If no commutation needed, AP will remain as is.

ii) If commutation is chosen, it should not exceed

one-third of annual pension.

AP'(after commutation) = AP(1 - commutation fraction).

Amount Commuted = (AP - AP') x commutation factor.

2. Late Retirement

a. Calculate FAS as in 1.a.(based on age 6Ø).

b. Compute AP" as in 1.b.(based on age 6Ø).

c. Calculate FAS" after age 6Ø (using last 3 years average).

d. Compute AP at retirement after age 6Ø :

$AP = AP"(1 + \frac{3}{4}\% \times M") + FAS"(N"/6Ø)$.

e. Test for possible commutation as in 1.c.

3. Early Retirement

a. Calculate FAS as in 1.a.

b. Compute AP" as in 1.b.(based on age retired).

Note: N is equal to the length from the date of joining to the date of retirement(which is before reaching the age of 6Ø).

c. Compute AP at retirement before age 6Ø :

$AP = AP"(1 - \frac{1}{2}\% \times M')$.

d. Test for possible commutation as in 1.c.

4. Ill-health Retirement

a. Calculate FAS based on the higher of:

i) the last year salary prior to retirement

ii) the highest average salary for 5 consecutive years.

b. Compute AP as in 1.b.

c. Test for possible commutation as in 1.c.

5. In-service Death

a. Calculate FAS as in 4.a.

b. Compute AP as in 1.b.

c. Test for possible commutation as in 1.c.

d. Compute LSB.

i) For married male :

LSB = 24 x monthly salary prior to death.

ii) For female and unmarried male :

LSB = 12 x monthly salary prior to death.

e. Compute AWP and/or DA.

i) If widow and dependants exist:

AWP = AP(before commutation) x 6Ø%.

DA = AP(before commutation) x 1Ø% x n , where $n \leq 3$.

ii) If no widow, and there are n dependants:

DA = AP(before commutation) x 25% x n , where $n \leq 3$.

6. Death of Pensioner

a. AP already exists.

b. Compute LSB: LSB = AP(before commutation).

c. Compute AWP and/or DA as in 5.e.

7. Death of Pensioner's Widow

a. Test if there are any eligible dependants existing. If so, DA = AP(before commutation) x 25% x n , where $n \leq 3$.

8. Withdrawal from Service

a. Compute total pension contribution due.

i) If the difference between the resignation date and

date of last interest update is less than one year, the current year pension contributions do not earn interest, only the pension contribution total as of last financial year.

Pension Contributions Due = total contributions (as of last financial year) x (1 + 3%) + contributions (current financial year total).

ii) If the difference between the resignation date and date of last interest update is not less a year,

Pension Contributions Due = (total contributions as of last financial year + contributions of current year) x (1 + 3%).

Appendix C

TRANSACTION CODES

There are 64 transaction codes used in the system which cater for different types of earnings, deductions, and allowances. Some of these codes are used for employees only, others are used for pensioners only, while the remaining codes are used for both employees and pensioners.

Earning and Allowance Codes: Codes under this category are all preceded by an "A".

Code	Description	Grouping	used for empl.	pens.	both
11	Normal Earning	Salary	*		
12	Part Earning		*		
13	Unpaid Leave		*		
14	Back Pay		*		
15	Lump Sum Termination Pay		*		
16	Pension Fund A (SSPF)			*	
17	Pension Fund B (FSPF)			*	
18	Annuities			*	
19	Bantu Pension Fund			*	
21	Normal Overtime	Overtime	*		
22	Public Holiday Overtime		*		
23	Sunday Overtime		*		
24	Saturday Allowance		*		
31	Location Allowance	Allowances	*		
32	Specialist Allowance		*		
33	Special Pension Allowance			*	
41	Entertainment Allowance	Reimbursive	*		
42	Car Allowance	Allowance	*		
43	Misc. Reimbursements		*		
44	Moving Allowance		*		
51	Production Bonus	Misc. Taxable	*		
52	Medical Aid Grant	Payments	*		
53	Travel Grant		*		
54	Pay in-lieu-of Leave		*		

Code	Description	Grouping	used for empl.	pens.	both
61	Exam Bonus	Misc.Non-taxable	*		
62	Tax Free Gratuities	Payments	*		
63	Dependant Allowance(SSPF)			*	
64	Dependant Allowance(FSPF)			*	
71	Annual Bonus	Annual Bonus	*		
72	Special Bonus			*	
81	Petrol Reimbursement	Petrol Reimburse-ment	*		
91	Free Benefits	Free Benefits	*		

Deduction Codes: Codes classified under this category are all preceded by a "D".

Code	Description	Grouping	used for empl.	pens.	both
A1	Pension Fund A (SSPF)	Pension	*		
A2	Superannuation Fund		*		
A3	Field Staff Pension Fund		*		
B1	PAYE	P.A.Y.E.			*
B2	Bantu Tax				*
B3	Rhodesian Tax			*	
C1	Medical Aid	Medical Aid			*
C2	Medical Aid above Limit		*		
D1	UIF Employee Contribution	U.I.F.	*		
D2	UIF Employer Contribution		*		
E1	Insurance Premium A	Premium			*
E2	Insurance Premium B				*
F1	Policy Loan	Policy Loans			*
G1	Personal Loan	Personal Loans	*		
G2	Personal Insurance Loan		*		
G3	Car Loan		*		
G4	Study Loan		*		
H1	Mortgage Bond	Mortgage Bond			*
H2	Insurance Bond		*		
I1	Medical Aid Claims	Medical Aid Claims			*
J1	Phone Calls	Canteen & Phone Calls			*
J2	Canteen		*		
K1	Car Lease	Miscellaneous	*		
K2	Charities				*
K3	Deposits Account				*
K4	NBS Subscription Shares		*		
K5	Personal Account				*
K6	Rates and Taxes		*		
K7	Rent				*
K8	Sports Club		*		
K9	Travelling Expenses		*		
Z9	Advances		*		

A p p e n d i x D

ALGORITHMS FOR SELECTED TRANSACTION TYPES

The transaction types presented here are those requiring special algorithms. These algorithms have been based on company rules and regulation governing the transaction types.

1. Part Earning (code:12)

   This earning code is used when an employee joins after the first of the month. Its effect is to reverse part of the full month's normal earning which has already been processed. It operates in the following ways: The user supplies the number of days missed and the system calculates the pay for that number of days in a negative value.

   $$\text{Part Earning} = -\left(\text{Basic pay} \times \frac{\text{number of working days missed}}{\text{total work days in the month}}\right).$$

2. Unpaid Leave (code:13)

   The user supplies the number of working days on leave. The system then calculates pay for these days and pays the amount in a negative value.

   $$\text{Unpaid Leave} = -\left(\text{Basic pay} \times \frac{\text{number of working days on leave}}{\text{total work days in the month}}\right).$$

3. Normal Overtime (code:21)

   The user supplies the number of hours worked overtime. The system then calculates overtime at whichever is the smaller of the following rates*: (1% of monthly basic pay) or (2.6/hour).

* These rates are user specified and may differ for different users.

Rate1 = .Ø1(Basic pay) x number of hours worked.

Rate2 = 2.6 x number of hours worked.

4. Public Holiday Overtime (code:22)

Only employees with less than R3ØØ monthly earning are entitled to this overtime pay. The user supplies the number of hours worked separately for each public holiday worked. Overtime payment is calculated according to the following formula: Pay whichever is the greater : (one ordinary day's pay + hours in excess of normal hours x double hourly rate) or (total hours worked at normal overtime hourly rate).

$$\text{Rate1} = \frac{\text{Basic pay}}{\text{work days in month}} + \left(\text{excess hours} \times \frac{2(\text{Basic pay})}{\text{work hours in month}}\right).$$

Rate2 = .Ø1(Basic pay) x number of hours worked.

5. Sunday Overtime (code:23)

Only employees with less than R3ØØ monthly earning are entitled to this overtime pay. The user supplies the number of hours worked separately for each Sunday worked. The system then calculates the overtime pay in accordance with the Factories Act:

a. If less than 4 hours are worked, it pays double time or one day's pay, whichever is the greater.

   Rate1 = .Ø1(Basic pay) x 2(number of hours worked).

   Rate2 = Basic pay/total work days in month.

b. If more than 4 hours are worked, it pays double time or two days' pay whichever is the greater.

   Rate1 = .Ø1(Basic pay) x 2(number of hours worked).

   Rate2 = 2(Basic pay)/total work days in month.

6. Pay in-lieu-of Leave (code:54)

The user* supplies the number of leave days for which the employee has to be paid. The system then calculates pay for those days.

$$\text{Pay in-lieu-of Leave} = \text{number of leave days} \times \frac{\text{Basic pay}}{\text{work days in month}}.$$

7. Personal Loans (codes:H1,H2,H3,H4)

The user supplies the amount of the loan and the amount to be deducted together with interest rate charged by means of a loan record, and the system will generate a transaction record for the loan containing instructions for monthly deduction from total earnings. Interest is calculated on the outstanding balance of each open loan every month. The interest paid is added to the accumulated interest paid. The system continues to deduct the instalment each month until the loan is completely repaid.

$$\text{Interest Paid} = \frac{\text{Interest Rate} \times \text{Loan Balance}}{12}.$$

$$\text{Total Interest Paid to date} = \text{Total interest paid (up to previous month)} + \text{Interest Paid}.$$

$$\text{Loan Balance to date} = \text{Loan open balance} + \text{Interest Paid}.$$

$$\text{Loan Final Balance} = \text{Loan Balance to date} - \text{Amount deducted monthly}.$$

---

* For terminating employees, the system automatically generates such transaction records, supplying also the balance of leave days due each employee.

Appendix E

FILES : ORGANIZATION AND DEPENDENCIES

This appendix consists of two tables. The first table briefly describes how the files introduced in Chapter Three are organized. The second table describes how files in the integrated system depend on each other. The detailed layouts for these files can be obtained from the author upon request.

Table 1 :

File Name	Organization	Record Keys	
		major	minor
Master File	IS	employee number	-
Dependant File	SD	employee number	record type
Job-Salary History File	SD	employee number	date
Loan File	IS	loan number	employee no.
Leave File	SD	employee number leave type	date
Transaction File	SD	employee number transaction code	-
Bank Parameter File	IS	bank code	-
Transaction Parameter File	IS	transaction code	-

Table 2 :

AFFECTED FILE \ SOURCE FILE	MS	DP	JS	LN	LV	TR	BP	TP
MS	-	1	0	0	1	1	0	0
DP	1	-	0	0	0	0	0	0
JS	1	0	-	0	0	2	0	0
LN	0	0	0	-	1	1	0	0
LV	0	0	0	0	-	0	0	0
TR	1	1	0	1	1	-	0	0
BP	0	0	0	0	0	0	-	0
TP	0	0	0	0	0	0	0	-

0 - not affected
1 - directly affected
2 - indirectly affected

R E F E R E N C E S

I. General:

Barber, David: The Practice of Personnel Mangement, Institue of Personnel Managment, London (1970).

Chruden, H.J. and Sherman, A.W. Jr.: Personnel Management, South-Western Publishing Co., Cincinnati, Ohio (1972).

Date, C.J.: An Introduction to Database Systems, Addison-Wesley, Reading, Mass.,(1975).

Dunn, J.D. and Rachel, F.M.: Wage and Salary Administration: Total Compensation Systems, McGraw-Hill Book Company, New York (1971).

Lupton, T. and Bowey, A.M.: Wages and Salaries, Penguin Education, Middlesex, England (1974).

Lynch, R.E. and Rice, J.R.: Computers: Their Impact and Use, Holt, Rinehart and Winston, Inc., New York (1975).

II. Specific:

(1) Hertz, David B.: New Power for Management: Computer Systems and Management Science, McGraw-Hill Book Company, New York (1969), p.193.

(2) ibid., pp. 179-182.

(3) ibid., p. 190.

(4) ibid., p. 195.

(5) Hussey, D.E.: Introducing Corporate Planning, Pergamon Press, Oxford (1971), pp. 124-137.

(6) McBeath, Gordon: Organization and Manpower Planning, Business Books Limited, London, 3rd ed.(1974), pp. 4-8.

(7) Bell, D.J.: Planning Corporate Manpower, Longman Group Ltd., London (1974), pp.124-126.

(8) ibid., pp. 127-129.

(9) ibid., p. 117.

(10) Dearden, J., McFarlan, F.W., Zani, W.M.: Managing Computer-Based Information Systems, Richard D. Irwin, Inc., Illinois (1971), p. 592.

(11) McBeath, G. and Rands, D.N.: Salary Administration, Business Books Ltd., London, 3rd ed.(1976), pp. 333-336.

(12) Martin, James: Principles of Data-Base Management, Prentice-Hall, Inc., New Jersey (1976), p. 87 and pp. 100-101.

(13) Boutell, Wayne S.: Computer Oriented Business Systems, Prentice-Hall, Inc., New Jersey (1968), pp. 169-170.