Inventing Systems Engineering

John Aris

The J. Lyons food and catering company in 1949 undertook an ambitious project known as LEO (Lyons Electronic Office). This created, for the first time anywhere, computer hardware and software for business applications. The author describes the history and applications legacy of this successful endeavor, using personal reminiscences and letters from one of LEO's founding participants.

The year is 1949. Your company's board of directors has just approved an ambitious new project called LEO. You are responsible for creating the software (as we would now call it) for a series of substantial business applications for a computer. They include a sophisticated payroll system, invoicing, innovative stock control, and production scheduling. You also have a remit to improve the business by using the computer in whatever other ways you can make feasible. The hardware does not exist. Nobody in the world has ever specified or designed a business application for a computer. Nobody has ever written a business program. You have two years in which to work out the principles then design the jobs, get them programmed and working, ensure they yield an economic return, and satisfy users—all from scratch.

The man who was faced with this interesting challenge, and who met it triumphantly, was David Caminer, manager of the Systems Research Office of J. Lyons and Company. This article[1] is based on, and frequently quotes, conversations and correspondence with him. I also gratefully acknowledge valuable suggestions from another early LEO pioneer, Frank Land.

More complete accounts of the LEO project can be found in two books: *LEO—The Incredible Story of the World's First Business Computer*,[2] which contains a 150 page history of LEO by Caminer, with contributions by twelve other authors who were personally involved with LEO; and the similarly titled *LEO, the First Business Computer*,[3] in which Peter Bird tells the story with more emphasis on the engineering aspects.

Definitions

A difficulty in writing about this early period is the terminology. At the time the terms for everything from client liaison and requirements scoping, through coding and operating, to implementation in the field, were "systems and programming" (for the activity) and "programmer" (for the practitioner). In this article I use "systems engineering" and "systems engineer" in this same comprehensive sense; "systems analysis" for the phase of scoping, requirements analysis, and specification; "systems design" for the phase of turning a requirements specification into a set of defined programs with detailed input layouts, file layouts, output layouts, and technical provisos (such as reconciliations to be carried out); and "programming" for the phase of coding, testing and integration. The phases of course tended to overlap. I hope those who use these terms in rather different senses will indulge me. It should also be mentioned that throughout this paper I use the term "LEO" to refer to the project, the machine or the staff: the context will make clear which is intended.

Background

It is remarkable that J. Lyons, a catering and food manufacturing company, should have initiated such a project, the first of its kind in the world.[4] There was, however, good reason. The computer project was a natural, though ambitious, development of Lyons' pioneering approach to office work since the 1920s.

Lyons' main business activities were running restaurants and teashops, and making food products—principally bakery products, tea, and ice cream—that were for sale in the company's catering establishments, to other caterers, and to the public). The business, by its nature, involved a very large number of low-value transactions, and the administration for these transactions was laborious, tedious, and expensive to a degree that threatened the health and growth of the company. Lyons had always invested in modern and efficient manufacturing processes; in the 1920s they set their minds to a similar policy of investment in modern and efficient administration.

The mainspring of this was John Simmons, a Cambridge Wrangler[5] whom Lyons engaged in 1923. Under his leadership during the 1920s and 1930s all Lyons' administrative processes were studied with great intellectual rigor, redesigned, and redesigned again. The objective was always that the processes must be as effective and as economic as Simmons' impressive team, together with the best available technology, could make them. Lyons, always under the imperative of their multiplicity of low-value transactions, became world leaders in this process engineering, then known as O & M. Systems analysis and systems design (though of course not computer systems analysis and computer systems design) were integral parts of O & M. Pioneering applications of calculating and accounting machines and a world's first use of microfilm were typical of the systems that Simmons' team devised. Simmons declared, as a pre-LEO lesson from the past, "record keeping was of greatest value when it showed what ought to be done, rather than merely what had been done."[6]

It was therefore not amazing that in 1947 members of Simmons' team were the first in the world to see that the "electronic brains" invented during World War II might have a role to play in business administration. Not only did they have this wild idea, they followed it up and established its feasibility, leading to the Lyons board's remarkable decision in 1949.

The LEO project set up by that decision was to design and build a computer based on the Cambridge University electronic delay storage automatic calculator (EDSAC) and to mount a series of systems on it that would improve further, in effectiveness and economy, upon those already honed by twenty years of high-grade O & M work. The EDSAC had triggered Lyons' decision by carrying out its first successful run that very day. The LEO systems were to include payroll, stock control, sales invoicing, and whatever else could fruitfully be devised (the feasibility studies had even envisaged word processing, which, it turned out, did not become economically viable for thirty years). This was not an experiment or a pilot project: it was expected to pay its way.

> There was no question of imposing applications. They had to be "sold" to the client departments.
>
> —Caminer[7]

The key figures in the LEO project were Simmons, with board-level responsibility; Raymond Thompson, on whose ideas it was founded and who was in overall charge; John Pinkerton, who was to design and build the hardware; and Caminer, who was to design and implement the applications. This article concentrates on Caminer's area of responsibility.

The challenge

> The job I was given was to prepare production work for the computer system and to have the work ready to run as soon as the not yet fully designed equipment was itself ready to run it. Everything—coding standards, program construction, and systems documentation—had to be created from scratch. The aim was always to be ready to use whatever facility Pinkerton was bringing to fruition.
>
> —Caminer[8]

Caminer joined J. Lyons as a management trainee on Simmons' team at the age of 21 in 1936. He served in the infantry in World War II and then rejoined Lyons with the responsibility of assembling the management accounts for its board—a task which (then as now) was less straightforward than might appear, it involved dealing with issues of incompatibility in terminology and data, of error trapping, of interpretation, and of politics. He then became manager of the Systems Research Office, which as its name implies was a forward-looking team, some twenty strong, considering new systems approaches and new technologies. He was thus the natural choice to take responsibility for LEO's systems and programming—though he was not released to do the job full time until May 1950.

In dealing with the challenge[9] outlined at the beginning of this article Caminer had a number of important advantages. He had been trained in Lyons' elite O & M team. He had a long personal experience with the company: its systems, people, and culture. He was able to recruit other individuals to the project with a similar background, experience, and knowledge (though not, in the early days, very many of them). Time in the Army had given him further experience in the benefits of good order and military discipline when tackling demanding objectives. Finally, he had the trust and support of Simmons and Thompson, and through them the trust and support from the company as a whole that Simmons' team had built up.

Difficulties

Perhaps, on the basis of all that, the challenge sounds not so daunting after all. Was it just a matter of applying the well-understood Lyons approach to systems analysis and design and coding the results? Hardly.

Business tasks were going to call for long, complex programs with complicated data and results, many alternative paths, and much dealing with exceptions.

> Apart from the basic principles and objectives it emerged that there was little relationship between the analysis for a job using conventional equipment and that for an integrated application on the computer. What had started as draughts had become three-dimensional chess.
>
> —Caminer[10]

One obvious difficulty was that the systems to be implemented were already highly developed in their noncomputer form. There was no fat on them but economic savings were nonetheless required. The computer was notionally a wonderful tool, but copious data was going to have to be entered into it, and copious results were going to have to be printed. These operations would not only hold back the computer's electronic speed, but would introduce staff costs, particularly for data preparation, which were absent in the noncomputer systems. Achieving net savings was not going to be easy.

A less obvious difficulty was that a prime advantage of the computer was integration—tackling business processes end-to-end and without human intervention once the basic data was entered. But just how far should that principle be taken? Where did the law of diminishing returns set in and where did feasibility give out? There was no basis, either of experience or of theory, for answering those questions.

A third difficulty arose from the very absence of human intervention, which the project targeted. Humans are intelligent and flexible. When something looks strange they notice it and perhaps correct it or perhaps raise a query. Computers are not like that. The whole strategy for detecting and dealing with data errors was going to have to be rethought in minute detail and redesigned.

Fourth, there was a group of difficulties arising from the computer itself. It had to be reconfigured halfway through the project because the original intention of using magnetic tape storage proved beyond the technology of the day. The computer's electronic speeds, though they seemed impressive at the time, were slow and the internal storage was very limited. The computer's reliability was poor, which was a very serious issue when the daily operations of the business were to depend upon it. What could be done about this, and what contingency planning was required?

Fifth, how were the new systems, once specified and programmed, to be validated? Computer time for testing was going to be a rare and precious commodity on a unique machine, and business users were not going to tolerate a trial and error experience when the systems went live.

Sixth, programming was going to be an even more novel activity than systems design. The few computers that then existed were all in universities, research, and defense establishments and were used for mathematical and scientific tasks. Programming these tasks was a chore carried out by the mathematicians with a little help from the local computer team. Programming was not thought of as an important, or even an interesting, discipline in its own right—but mathematical skills were thought to be necessary for doing it. For most mathematical and scientific tasks the programs were relatively concise. Business tasks, on the other hand, were going to call for long, complex programs with complicated data and results, many alternative paths, and much dealing with exceptions. Writing, testing, and maintaining such programs was going to require discipline and professionalism. Professional mathematicians were not available to perform the task (and would have found it deeply unsatisfying if they had been available). Of course no software tools of any kind existed.

Some of these difficulties were obvious from the start and solutions for them could be planned. Some difficulties gradually manifested themselves and the LEO team had to retrofit solutions to the work already done. Some problems were apparent only to imaginative insight or could be solved only by intuition.

It might be asked whether experience of punched-card systems took some of the sting out of the difficulties. Punched cards were something of a halfway house between noncomputer and computer systems. Did they not throw light on integration, on coping with the absence of human intervention, on systems testing, on writing complex programs? The answer is no, because Lyons' experience of punched-card systems consisted mainly of considering and rejecting them on cost-effectiveness grounds. In most cases Lyons found that the cost of data preparation and handling could not be compensated by savings from the limited integration such systems could offer. Whether it would have helped if Lyons had had more experience of implementing punched cards is an interesting question (my guess is yes, but not much).

Faced by this situation, Caminer started four broad lines of action. Principles and standards for systems and programming work had to be prepared (and kept under review as experience developed); a team had to be assembled; research on how to write and test business programs was necessary; and the specification and design of the first applications had to be tackled.

The LEO approach was to optimize memory occupancy and runtime by clever coding, that is, minimizing the number of instructions written or to be obeyed.

Principles

Many of the early principles and standards, based on a combination of Lyons' O & M experience and Caminer's very acute intuition of what was needed, lasted throughout the project and long after:

- The scope and aims of each application must be searchingly and creatively established, defined, and discussed with the client to reach agreement.
- Requirements specifications must be well written, well understood by the users, and have client agreement.
- Specifications should be kept as unchanged as possible during implementation.
- Systems should be as comprehensive as possible, dealing with exceptions as well as normal situations, but should not attempt "a bridge too far" (plenty of room for intuition here).[11]
- System benefits must be explicit and quantified.
- An imaginative examination of all kinds of possible pitfalls in the system must be made and evasive actions taken.
- Detailed flowcharts of paths through the system must be drawn and kept up to date.
- Data must be entered only once.
- All data must be rigorously checked for credibility by the program and rejected if unacceptable but with allowances made for later reinsertion, and amendable if found to be in error.
- Form design, for both computer input and computer output, is crucial.
- Reconciliations on the computer's and the program's internal workings should be calculated and displayed. (This principle was eventually discarded in the early 1960s, by which time reliability had greatly improved.)
- Programs must be carefully checked by a second programmer before trying them on the computer. The computer's time must be regarded as a means of verifying correctness, not finding errors. (This principle is perhaps still valid, though cheap computer time has led to near universal neglect of it.)
- No error is the computer's fault, it is the fault of the people who should have corrected it or who should have allowed for the possibility of it happening.
- Spare space must be left in all programs for future modifications.
- Orderliness in thought and in documentation is essential.

The team

Assembling the original team was not a huge task. It consisted of five people including Caminer; all (except one new management trainee) were established Lyons staff.

> It is worth emphasizing the extent to which Lyons did see the computer project as another venture in which the resources of the company would participate organically rather than as something deposited complete on the company from outer space.
>
> —Caminer[12]

Business programming

One of the team, Derek Hemy, had a particular aptitude for programming and program design. He undertook the first steps in LEO's programming research in conjunction with the EDSAC team at Cambridge. A number of important ideas came from Cambridge, such as indirect addressing. One of the Cambridge team sketched, with Hemy, the first outline of a payroll program but, as mentioned above, mathematical programming was very different from business programming. Cambridge made

> **The principle was first to establish the outlines of a system and then to amplify them through successive levels of detail.**

much use of standard subroutines (such as square root, trigonometry functions) that could be assembled into programs, whereas the LEO team, after one abortive experiment, found little use for this technique. The LEO approach was to optimize memory occupancy and runtime by clever coding, that is, minimizing the number of instructions written or to be obeyed, whereas this was less of a priority for mathematicians. The LEO team emphasized the importance of breaking programs into discrete, comprehensible stages and annotating them so other programmers could more readily understand what they did—vital for both program checking and subsequent modification. Coding sheets, memory layout sheets, file layout sheets, and other documentation had to be designed and standardized.

Applications

Caminer personally carried out the specification and flowcharting of the earliest applications: a notable historic first. The use of flowcharts was itself innovative, having recently been introduced as a system design aid by Lyons' Systems Research Office. Caminer had the help of Lyons' O & M staff, who of course were still involved and conducted most of the liaison in detail between the LEO project and the users. Hemy also turned out to have a talent for estimating program sizes and timings, which was invaluable in the systems design phase.

> What could go wrong was as important to identify as what was needed to go right. Errors and stoppages had to be guarded against, not accepted as inevitable, for all that the art was in its infancy.
>
> —Caminer[13]

The principle was first to establish the outlines of a system and then to amplify them through successive levels of detail. The typical sequence was that Caminer, with his thorough knowledge of Lyons' processes, drafted the scope and aims of each system and reached agreement on them with user management. He then went on to detail inputs, outputs, calculations to be performed, and any other key issues. At this point he would also draft a (necessarily long and elaborate) requirements specification, including outline flowcharts, which, with the aid of O & M, was agreed upon and signed off by the users. Inputs and outputs were now frozen and, with the help of Hemy, the computer design was worked out. This design would evolve during implementation as memory and running time considerations became clearer.

Lyons had always assumed that LEO's first application would be the company payroll. However as the project progressed four further major jobs were developed almost in parallel with the payroll, all of which broke new systems ground. Teashops Distribution was a time-critical daily stock replenishment system for some 200 retail outlets. Reserve Stores was a stock control and production scheduling system. Tea Blending not only tracked stocks of about 300 types of unblended tea—available or in transit from growers—but provided Lyons, for the first time, with full, detailed cost and availability data for the complex problem of mixing these into branded blends—effectively a decision support system.[14] Bakery Wholesale Rails was a highly integrated set of procedures involving dispatching, invoicing, sales accounting, sales statistics, directions to packers on carton sizes, packers' bonuses, and management reports. The output from the bakery application went to thousands of small and large retail stores selling food products throughout Britain. It was the first system in which the public outside the computer owning organization received and used computer output as part of their everyday work.

The first system that Caminer specified in detail was the payroll. In the project's feasibility phase (pre-1949) the outlines of a computer payroll system—inputs, brought forward and carried forward files, outputs—had been completed (there was a perfectly valid half page description of the payroll in the 1947 report that inspired the LEO project) but Caminer now had to go into all the massive detail that would put flesh on those bones.

Caminer needed to address many payroll systems issues. What should be in those files and how much file space would it require? Exactly what inputs and outputs were needed? What must the input and output forms look like? Precisely what credibility checks must be carried out on the data? Exactly what calculations must be performed (for example, what were the tax calculations on advances of pay to those going on holiday)? What exception routines were needed? What aspects of payroll might be "a bridge too far" (as it turned out, only team bonus payments)? What possible pitfalls might there be in the logic of the system, in its implementation, in the regular operation of the computer, in the surrounding manual procedures, and what should be done about them? What reconciliations were needed? What contingency plans? What management information might be a valuable by-product? And so on. Specifications for noncomputer payrolls had of course been written before, but many points had to be covered in this computer system that no previous one had addressed.

The payroll was the first such specification but Teashops Distribution was not far behind. Unlike payroll, the teashops application was not an adaptation of a precomputer system and had not been an explicit part of the LEO remit. It was an entirely fresh look at an old but recalcitrant problem. The problem was that each of the 200 teashops had to reorder hundreds of bakery and kitchen products every afternoon for the following day to ensure freshness and avoid waste (these were the days of rationing). The products then had to be manufactured, packed, and delivered overnight. The preparation of the orders was a big and tedious task every day for the teashop manageresses. Once the teashops' order forms were delivered (physically) to Lyons' Hammersmith works there was formidable paperwork necessary to initiate production and dispatch, and the time available was limited. It was certainly too short a time to allow an extensive data-preparation operation followed by a series of computer runs before production could begin. At the same time, the computer was potentially an ideal way of optimizing production runs, assembly of individual teashops' orders, van loading, and the necessary documentation.

The new system was Caminer's invention. He discovered, in true O & M fashion by studying piles of previous transactions, that orders for any one teashop fell into a pattern dependent on the day of the week. Armed with this knowledge, he devised a system whereby the computer produced for each manageress each day a suggested reorder. The manageress now had only to indicate any changes she wanted to make, vastly reducing the data preparation load. The manageresses dictated any changes over the telephone directly to keypunch operators (such use of the telephone was itself innovative in the early 1950s). The resulting punched cards went into the computer in batches and the computer could do its part in good time for production. Not quite a real-time job, perhaps, but very much a time-critical one—probably the first of its kind in the world by at least five years. Most computer applications throughout the 1950s processed data after the business event, rather than on the critical path before it.

Three types of work were carried out appreciably earlier than the major applications and became important ingredients in the evolution of systems engineering.

Preliminaries

> It all seemed so obvious that what Simmons described as the "incredible speed" of the computer would have to be matched by a medium that was equally state of the art. The fact that with planning and programming skill a balanced system could be put together with paper tapes as the main current data medium and with punched cards as the main data storage medium had not been explored.
>
> —Caminer[15]

The LEO team had initially envisaged magnetic tape as the main device for carrying data, and magnetic tape drives were installed in the early days. However the technology chosen proved unworkable at that time, so punched card equipment had to be installed instead. This setback delayed the major applications significantly, so they were not in fact the first jobs to run on LEO.

Three types of work were carried out appreciably earlier than the major applications and became important ingredients in the evolution of systems engineering. They were the test programs for the computer; the Bakery Valuations job, which had the distinction of being the first regularly run (though not major) business computer application; and a series of mathematical jobs undertaken on an opportunist basis for customers outside Lyons.

It became clear early in the project that test programs were a nontrivial requirement. This computer, uniquely for its time, was going to have to give dependable service to a business. Teashops Distribution was particularly time critical, but all the jobs had a degree of time criticality. For example, payroll had to be ready (and correct) for payday; tea stock reports had to be available to production planners when needed; and delays anywhere were likely to be very damaging to the credibility with business management that LEO crucially needed to sustain.

However 1950 valve-based technology was inherently fault prone. Specifying and programming the test programs became a challenging application in its own right. The detailed requirements were worked out and documented (and continually updated as experience accumulated), flowcharts were drawn, and programs were coded and independently checked. These programs had to exercise each logical function and each circuit. Though this test application bore little resemblance to a business system it was sophisticated, and its use of systems engineering skills was an important stage in their evolution.

The Bakery Valuations job was a modest one, not demanding bulk input and output and not especially time critical. It was scheduled at Caminer's instigation when it became clear that the major applications would have to be postponed while new input and output mechanisms were designed and developed to take the place of magnetic tape. There was some resistance from higher management to introducing such a trivial job: it was not the kind of major advance that the Lyons board was looking for from its LEO investment. Nonetheless it was useful, it provided economic savings, it would yield—for the first time—experience of regular running of a job, and it would be a satisfying and tangible achievement for the systems engineering team.

Caminer, again, specified and flowcharted the system, which valued the weekly output of each bakery, bakery dispatches to each sales outlet, and stocks awaiting dispatch. The results went to top management. He built reconciliations and restart points into the programs. It went live in November 1951 and ran every week thereafter. The system saved money and produced its reports faster than the previous noncomputer system. Perhaps more significantly, it provided the experience of using data from the outside world, with all the fallibilities that had previously been circumvented by intelligent human interpretation on the job. It also turned out to be a valuable extra test program for the hardware and even contained an alternative piece of code to be activated if one particularly refractory circuit was malfunctioning!

> It scarcely mattered whether a scientific job was executed today or tomorrow or the day after. It was generally being accomplished altogether faster and with less human effort than had hitherto been conceivable.
>
> —Caminer[16]

Mathematical jobs came in when it became known in the market that the computer under development could offer some capacity, particularly for work that was not time critical and had no need for bulk input or output. Typical jobs were weather forecasting, ballistic tables for the army, and flutter calculations for aircraft design. These jobs were again subjected to, and influenced, the evolving systems and programming disciplines, and furthermore were operated in a revenue-earning context.

This first experience of live operation cast new light on the issue of hardware unreliability and how to deal with it. Restart procedures were introduced, which allowed long program runs to be repeated in part rather than as a whole if they had encountered hardware trouble. It sharpened thinking and sophistication for the design of in-program reconciliations. It also illuminated issues of accuracy in rounding off large numbers.

Implementation tasks

Programmers of the early applications, particularly the major jobs, faced three particular challenges. The first was retaining an overall intellectual grasp of the sprawling complexity of the programs while coding in machine code (though not, mercifully, binary). Flowcharts, breaking the program into short, logically coherent stages, and annotation were vital to this.

The second challenge was fitting each program and its data into the memory (there was of course no backing store). This required minimization of the number of instructions in the program and tight packing of data. Data had to be packed more tightly still on the brought forward and carried forward files, which were held on the slow medium of punched cards. The cards were ingeniously used to their utmost by holding binary numbers horizontally rather than decimal numbers vertically: this was one of the many clever expedients devised as the project proceeded to cope with the limitations of the hardware. Another was forming instructions by program for later execution ("if *X*, plant an instruction to go to *Y* ten steps ahead").

The third challenge was to minimize runtime. Major file processing programs and major mathematical tasks took hours not minutes to execute. As computer time was scarce and valuable and as the mean time between faults was short, reducing these runtimes was a priority. One instruction saved in a main loop could reduce overall running time by many minutes. Unfortunately it was often only possible to save running time at the cost of increasing memory occupancy and vice versa. Satisfactory compromises could sometimes be achieved only by redesigning the program structure of the system.

These challenges will be unfamiliar to the great majority of today's programmers. They made programming much more interesting but were expensive in skilled people's time—which, however, was cheap in relation to computer time. An interesting reflection of the skill employed is that the runtimes for the payroll, estimated on the assumption that the file holding medium would be magnetic tape, were faster in practice despite having to use much slower punched cards.

Independent checking eliminated many program errors, but inevitably not all. Debugging online was discouraged because of the scarcity of computer time, but nonetheless some expertise in it was developed, usually after midnight. The debugging involved peering at binary patterns on a cathode-ray tube, with some additional help from listening to patterns of machine activity on a loudspeaker. The alternative was to study memory dumps, also in binary, offline. Test data came in two varieties. First the programmer devised his or her own, trying to cover all paths through the program and all permissible and impermissible data variants. When he or she was satisfied, real data, more voluminous and typical but less logically stringent, was acquired from the prospective users for a further round of tests. Only when all glitches revealed by either set of test data were fully explained and corrected could pilot runs with the users begin.[17]

LEO people

> The rapidity and the success of the achievement owed much to the preparatory thinking that had anticipated the decision to proceed, but a great deal, too, was due to the closeness of the working relationship between myself and John Pinkerton.
>
> —Caminer[18]

A feature of the whole project was how closely the systems and programming and the engineering teams worked together. There was little difficulty in doing so, as both teams were very small, largely made up of long-term Lyons people, and located together. The benefits were significant. The programmers were able not only to influence the hardware's logic design (for example, by calling for hardware instructions for conversion between binary and decimal) but also to understand issues, such as circuit reliability, which directly affected their own work. Similarly, the combination in the same individuals of the systems analysis, systems design, and programming roles led to a work optimization, which was highly necessary given the limitations of the technology. The loss of this close, interdisciplinary cooperation, though no doubt inevitable as the computer industry grew, has been sad.

It was clear that for regular time-critical business applications to succeed, hardware and software were not enough. Professional data preparation and professional computer operating would also be required. For data entry two media were used: punched paper tape, where variable length fields were needed, and conventional (decimal) punched cards, where mechanical sorting was a requirement. For punched cards the discipline of keypunching, checked by rekeying and comparison, was well established and the appropriate equipment was available. This was not the case for paper tape. LEO designed a paper tape "comparator" specifically to carry out the checking. A team of punch operators was assembled—this was the world's first computer data preparation section. Operating again presented new challenges. Nothing in the nature of operating system software existed. Much manual intervention between programs, often including card sorting and printer plug board changes, was required. Decisions had to be made when unexpected stoppages occurred, as they often did. Again a specialized team was formed.

As the reader will have realized, the work on the major and minor applications and the ever-increasing understanding of systems engineering was intellectually fascinating and absorbing. It was also extremely demanding. The team frequently worked all night. Caminer was a hard taskmaster and his wrath was terrible—though it was reserved for occasions when it was deserved (cutting corners, ignoring disciplines, making the same mistake twice), and he worked himself as hard as the others.

> I believe that I was regarded as unreasonable in those days. There seemed no other way. There was so much to do and so little time to do it in,

> and resources were so limited. To keep ahead of expectations in Systems Engineering and to produce a stream of dependable, economically viable integrated jobs meant years of grinding, dedicated work by the application team. I myself was completely dedicated to the work in hand and placed it before family commitments and other interests. I expected others to behave similarly on the particular jobs with which they were concerned, and to be ready to help out in a comradely manner when one of their colleagues needed a hand. When they had been engaged they had been warned that the work would be both mentally and physically exacting. That had frightened no one whom we wanted away. But looking back it must have been a sore trial. My own wife was totally resilient in dealing with my odd hours and absences and our young children saw little of me. Happily the other young wives were tough too, though it must have been less easy for them to understand the voracious demands of what we were about.
>
> —Caminer[19]

The team now began to be expanded, with individuals drawn from within Lyons but soon also with new graduates, and questions of how to select and train them had to be considered. This gave rise to an aptitude test (essentially a simple exercise in programming) and a training course, both designed by Caminer. Both ran regularly (with occasional adaptations as the technology developed) for some ten years and were much praised by LEO's customers.

> Month-long training courses were established which converted newcomers in a short time into active contributors. The existing staff were the instructors and the evaluators. Students quickly became teachers.
>
> —Caminer[20]

The challenge accomplished

The new input and output system came online in 1953 and the first pilot runs of the payroll took place that June. The Lyons organization was understandably cautious about committing anything as sensitive as the pay of their employees to a system as innovative and vulnerable as this, so a long series of pilot, then parallel runs took place.

Parallel running, as its name implies, involved comparing the computer's results, in detail, with those produced by the noncomputer system and explaining any differences (by no means were the errors always by the computer system!). Though costly and tedious, this process was essential not only to getting the computer system right but also to building management's confidence in it. Parallel running was an accepted Lyons practice for noncomputer systems, but once again it was the first time that system proving on such a scale had been done by anybody. Eventually the first live run took place in February 1954, and 10,000 employees were on the system by the summer of that year. Lyons would not go beyond that number while there was only one computer, and contingency plans for making emergency payments remained in place for some years. However, they were never needed. The system ran every week and though, of course, there were both hardware and software problems from time to time it never failed to deliver.

There was extensive press coverage of the first live runs. *The Economist*, for example, was moved to futuristic speculation:

> A year or so ago, a suggestion that one of those thinking robots, the electronic brains, should be put to tasks so mundane as the counting of pounds, shillings, and pennies for a weekly wage packet would have been greeted with general scepticism. Now that it has actually happened, preconceived opinions about the type of machine suitable for office and accounting work have received a severe jolt… Is this the first step in an accounting revolution or merely an interesting and expensive experiment? There are those who do not believe in the desirability of introducing anything as esoteric as electronics into business routine at all. Others believe that there is a limited field for electronic methods, providing that they fit into, and do not disrupt, established business systems. But there is a third group—of which Lyons is one—who consider that a major revolution in office methods may be possible… Electronic computors [*sic*] were essential for the development of atomic physics and are rapidly becoming a necessity for aircraft and missile development. Might they not also have a valuable, if less spectacular, contribution to make to improving business efficiency? …To save time and labour is always worth while. But what may be more important in the long run is the prospect of doing calculations on electronic computors that would simply not be possible by normal methods because of the time that they would take. This has opened up fresh fields in science; it may conceivably do the same in business.[21]

Teashops Distribution was not far behind the payroll. It went live in October 1954 and ran regularly and dependably every day thereafter. Its inventiveness, and in particular the work it saved the manageresses every day, was quickly recognized and acknowledged. After only a few days of operation, the daily report from the Wembley teashop said "the head staff at this shop would like to give thanks for LEO. This is a wonderful time-saver, work saver, and we are grateful for it." As the manageresses were a notoriously crusty group this was a most satisfying achievement.

Reserve Stores and Tea Blending also successfully went live during this period. Reserve Stores was a short-lived job because its necessity went away with the abolition of rationing. Tea Blending continued to run (on successive generations of hardware) for at least 25 years and may still be in operation in recognizable form today—which would make it the world's oldest business application.

The LEO team, therefore, had gone from the completely green field of 1949—no hardware, no experience anywhere of computer systems analysis, computer systems design, or business programming—to four major live applications and a large number of minor ones in just more than five years. Except for the magnetic tape problems that time could have been considerably shorter. The major applications were sophisticated even by current standards— they also were economically successful and satisfied their users. The number of people who had achieved all this was extraordinarily small: up to the go-live of Bakery Valuations, fewer than 20 people in total (management, design and maintenance engineers, "programmers," secretaries, data preparation staff) had taken part. By 1954 the systems engineering team had risen to about ten, none of them of course with any previous experience.

Spreading beyond Lyons

This is an impressive story, but of course it is only the beginning of the story of systems engineering. The leading role in that story continued to be LEO's for at least another five years, as a number of striking new developments now took place. Further major applications for Lyons were initiated. Outside customers, impressed by what Lyons had achieved, began to commission systems from the LEO team. Lyons management not only approved the building of a second machine for the company but also agreed that LEO could make machines for customers. The teashops company was now in the computer business. The systems engineering principles and practices developed during the formative period turned out to be generally applicable.

The first major outside customer was the Ford Motor Company. They knew and respected Simmons through the Institute of Office Management (now the Institute of Administrative Management in London), of which he was a leading light. Ford managers visited Lyons to see the payroll system and commissioned a payroll system of their own to run on LEO's machine. Obviously they could not use the Lyons software as it stood, but the principles were similar and quite a lot of the program code was actually copied. Caminer produced a draft specification in May 1955; the order was placed in August, parallel running started in November, and the first live run was in December. For a major custom-built system this timescale still seems incredible.

Many other customers followed, initially for jobs to be run on LEO's machines (the second went live in May 1957) and then for machines of their own. The first customers had strong O & M teams of their own and an approach similar to that at Lyons was possible—with the LEO systems engineers collaborating with the customers' O & M. Caminer personally specified the Stewarts and Lloyds (a steel firm) payroll and others. Indeed members of the LEO team were still writing specifications and designing systems for customers—often in the form of elaborate presales proposals—in the 1960s.

However, a notable difference gradually became apparent between customer and Lyons work, that is, the customers had reduced room for maneuver in systems analysis and design. In Lyons there was a general understanding that the efficiency and effectiveness of any business process could and should be improved, and that computerization was an excellent opportunity for doing so. For many customers radical change in working practices was not an option. Top management were in most cases not sufficiently committed to push such radical change through, even if they recognized it as desirable. These managers were therefore looking for something that saved money but did not aim at uncomfortably realized optimization. In many cases they had a background in punched card systems, and saw a computer as an enhancement of unit record equipment rather than as a major, liberating innovation. LEO was able to satisfy such customers, but the team was always happiest dealing with the minority of adventurous customers who saw the computer's true potential.

There were indeed customers who expanded LEO's thinking. One early example is that Lyons had resolved that alphabetic (in addition to numeric) printed outputs were an expensive luxury and that all results should be coded as numbers. The first customer to buy a LEO machine, the tobacco manufacturer W.D.& H.O.Wills, saw the extra flexibility and user friendliness of alphabetic printing and insisted upon it.

In the 1950s and early 1960s adventurous customers for business systems may have been commoner in the United Kingdom than in the United States. There was an important difference in the economics of computing between the two countries. In the US labor costs were high, so cautious computer systems—easy to implement but falling far short of exploiting computers' true potential— could readily be justified by quite minor staff savings. As a result, most early US business systems did not depart far from the pattern established using punched cards (much to the benefit of IBM). In the UK, by contrast, labor was cheaper so unadventurous systems were harder to justify.

The strategy by which LEO specified and designed adventurous customers' systems (it could be regarded as an early manifestation of outsourcing) was successful but could not be sustained indefinitely. Customers were encouraged to hire their own professional staff, often from LEO, and gradually took over these tasks for themselves. However, by the late 1950s the majority of the world's sophisticated business computer applications were still LEO designed, or designed by systems engineers trained by LEO, in accordance with the principles laid down by Caminer and his staff, although by then the competition was beginning to catch up.

Commentary

The list of LEO world firsts in systems engineering is impressive but perhaps a more important question is which of LEO's achievements lasted to become part of world best practice. It is also interesting to ask what, with hindsight, LEO got wrong.

The list of firsts goes like this:

- Lyons employees conceived the idea of using computers for business administration before anybody else.
- Caminer specified the first business applications of computers both inside and outside Lyons and, with his team, designed the systems.
- These early specifications and designs pioneered flowcharts, file design, form design for input and output documents, in-program reconciliations, and restart procedures.
- More subtly, LEO developed the concepts of systems integration (entering data once and squeezing the most out of it) and of handling error-prone data without human intervention.
- The developed systems involved business process reengineering, and included time-critical and decision support systems as well as systems initiating operational action rather than reacting to it.
- Hemy established the disciplines for writing the long, complex programs characteristic of business systems and, with his team, wrote the first such programs.
- LEO created the disciplines of checking programs offline, debugging them, and carrying out large scale pilot and parallel running.
- LEO set up the first professional computer data preparation unit and the first professional computer operators.

What were the lasting achievements? By the time I joined LEO (in 1958, some five years after parallel running of the Lyons payroll began, and a few months after the delivery of the first LEO machines to outside customers), there was a highly professional systems engineering department, including customer employed as well as LEO staff, constantly applying all the lessons of the early applications. Further experience was of course continually adding to good practice but the principles were remarkably unchanged and survived the advent of magnetic tape, operating systems, and other upheavals. Most of the principles remain sound today; though some are now concealed in software (for example, operating procedures) and some have been overtaken by technological advance (for example, the need for in-program reconciliations). Some are neglected though they should not be (such as flowcharting and offline program checking).[22]

LEO, and to a lesser extent its successors English Electric and ICL, continued to teach those principles and their further enhancements for many years. However other computer companies by the early 1960s were teaching their own versions, and the cohesion and discipline of LEO's approach gradually disappeared. Much of that approach was obvious good practice and was reinvented (and is still being reinvented) elsewhere. LEO's historical claim is not that its thinking was unique but that it did it first and it did it whole heartedly.

What did LEO get wrong in the field of systems engineering?[23] In the early days it went up some blind alleys (for example, a very complex set of flowcharting symbols or a program structure based on closed subroutines like those used for mathematical algorithms), but perhaps more importantly it lacked the resources or time to try alternative approaches and then use the winner.

> We always had to choose right first time and not being on Olympus didn't always succeed.
>
> —Caminer[24]

LEO may have missed an opportunity to reuse systems and coding (moving towards software packages), but the hardware constraints of space and time made anything other than intelligent Chinese copying infeasible until the late 1950s. LEO indeed got so good at Chinese copying that it missed the importance of forward compatibility, which IBM exploited so well in the 1960s. It was very dependent on high-quality individuals working extremely hard, which was not sustainable over the long term. The LEO approach was inward looking and in an ideal world the group would perhaps have deliberately spread its hard-won good practice to universities, business schools, software houses, consultants, other computer manufacturers, and so on. However, the time was not ripe, effort was scarce, and the task did not appeal to Lyons, not least for commercial reasons. Some dissemination did take place through the export of LEO people, not only to customers but to other companies both in the UK and abroad.

Systems engineering was invented in many places in many ways, it must be admitted. It emerged from punched card and other office machine techniques in the manufacturers of those devices and in the customers who used them, from engineering practice in engineering companies, from accounting practice in financial departments, from trial and error everywhere. David Caminer and his team did it very early, very successfully, through a combination of prior experience, intuition, and conscientious and imaginative learning on the job, and set a standard of best practice that was unique for its time.

References

1. This paper is a revised and expanded version of one presented and published under the same title in *Proc. Kiev Symp. Computers in Europe: Past, Present, and Future*, Int'l Charity Foundation for History and Development of Computer Science and Technology, 1998
2. D. Caminer et al., *LEO—The Incredible Story of the World's First Business Computer*, McGraw-Hill, New York, 1998. (UK edition: *User-Driven Innovation*, McGraw-Hill, Maidenhead, 1996.)
3. P. Bird, *LEO, the First Business Computer*, Hasler Publishing, Wokingham, 1994.
4. The background is examined more closely in J. Hendry, "The Teashops Computer Manufacturer," *Business History*, Vol. 29, No, 8, 1986 and in F.F. Land, "The First Business Computer: a Case Study in User-Driven Innovation," *Proc. Kiev Symposium*, 1998, and in this Annals issue.
5. Cambridge University gave the title of "Wrangler" to its top mathematics graduates.
6. J.R.M. Simmons, *LEO and the Managers,* Macdonald, London, 1962. p. 25.
7. Note from Caminer to Aris, May 1998.
8. Note from Caminer to Aris, May 1998.
9. D. Caminer, "LEO and Its Applications: The Beginning of Business Computing," *Computer Journal*, Vol. 40, No. 10, 1997.
10. Note from Caminer to Aris, May 1998.
11. D. Caminer, "...And How to Avoid Them," *Computer Journal*, Vol. 1, No. 1, 1958.
12. Note from Caminer to Aris, May 1998.
13. Note from Caminer to Aris, May 1998.
14. These systems are described in F.F. Land, "Systems Analysis for Business Applications," *Resurrection*, Summer 1996.
15. Note from Caminer to Aris, May 1998.
16. Note from Caminer to Aris, May 1998.
17. Fascinating material on the early days of programming, supplied by Derek Hemy, will be found in Bird, op. cit., pp. 52-62.
18. Note from Caminer to Aris, May 1998.
19. Note from Caminer to Aris, May 1998.
20. Note from Caminer to Aris, May 1998.
21. "Electronic Abacus," *The Economist*, 13 March 1954, pp. 789-791.
22. Comparisons between LEO and modern practice can be found in J.B.B. Aris, "Systems Design—Then and Now," *Resurrection*, Summer 1996.
23. The wider question of why LEO did not sustain its world lead in business computing to become a dominant supplier is intriguing but beyond the scope of this paper. It is the subject of a so far unpublished note from Caminer, dated 19 July 1998, to the author.
24. Note from Caminer to Aris, August 1998.

John Aris joined LEO Computers immediately after graduating in classics from Oxford in 1958. He worked for LEO and its successors until 1975, when he became head of computer development at Imperial Group. From 1985 to 1990 he was director and chief executive of the UK National Computing Centre. Since 1990 he has been associated with the IMPACT Programme, which he founded.

Readers can contact the author by e-mail at johnbaris@aol.com.