ICL Partner

Competing Against Lane-Based Products

MB-RPSSU-950 Issue 2
September 15, 1995

A never-ending stream of "lane-based" POS products continues to impact the market; it seems that anyone with grocery experience who can program a desktop database can't resist writing their own system. "Lane-based" refers to architectures where the central POS functions, files and storage are housed in the terminal as opposed to a "System" architecture where the terminals are secondary to a central device. Many lane-based products offer connections to a "collector" or "consolidator" PC or other device, but this does not alter the fact that the product is lane-based".

The people selling lane-based PCs are happy to capitalize on the exaggerated image of PC-POS with the unsophisticated buyer. Often, these products are sold as "nirvana" solutions, which are promised to be perfectly "open," perfectly portable, perfectly redundant, high performance, and very low cost with all the features in the world. Of course, the grocer ends up with none of the above, and a lot less than expected.

Lately, the industry has been turning ***away*** from lane-based architectures in favor of the traditional system architectures. With **ISS*45*** and RPS/Supermarket, the RETAIL*partner* is able to offer a true system architecture with more power and flexibility than lane-based products. This bulletin is to presents information to help the RETAIL*partner* compete more effectively against such systems. Below, we'll discuss:

- Lane-based architecture,
- The Core Software Platform
- Open Systems
- The Portability Myth
- The Low-Cost Myth
- The Changing Industry View

And we'll finish with a Product Comparison.

LANE-BASED ARCHITECTURE

The typical architecture of a lane-based product is to use a standard PC as the POS terminal. Of course, there is nothing wrong with PC terminals themselves. The problem is with the accompanying mindset that PC desktop software platforms, tools, and the resulting software are automatically appropriate to POS—they're not.

Anyone who's done a little database work might think about writing a POS application. Most of us get over this temporary insanity, however, and move ahead with our lives. But with the availability of PC terminals it's easy to get fooled into thinking that one can program a POS terminal just like a desktop PC, and for some people the temptation is just too great.

The basis of lane-based architecture is the independence of each terminal at the checkout counters. Each terminal has its own set of files, own operating software, own application software and its own database. It is purely internally-directed, and whole-store issues are secondary. Again, there's nothing wrong with terminal independence, and in fact this "Replication" of POS function is an important feature of **ISS*45*** architecture. The problem with Lane-Based architectures stem from the mindset of the developer, who's typically pretty good at programming desktop databases. Unfortunately, this, is far easier than executing a sophisticated design for an On-Line-Transaction-Processing (OLTP) system like POS.

So, the focus is naturally on the features, not on the architecture, not the timing nor positioning of data, nor how data is transitioned between system elements. The end result is almost invariably a lane-based system, where a standard, low-cost database package is put to work running POS transactions. The system is fine in a one-lane store; but add a second lane and the difficulties begin.

Fundamentally, the problems with lane-based architecture stem from the fact that a store system is ***not*** a collection of independent terminals. First, it is essential to consolidate data, reliably and in real-time. Next, all data in everywhere in the system must be in synch, regardless of end-of-day or cashier sign-on status. Third, the full system must be manageable from a single point. Fourth, central office functions and data integrity are as important as POS functions.

The answers to these issues cannot be solved by using Novell Lite or any other office-style LAN as the "instant glue" in an attempt to fuse independent registers into a true system. If the applications are designed as independent POS, then they will remain independent, LAN or no LAN.

THE CORE SOFTWARE PLATFORM

The quality of a POS application can be no better than its platform. For low cost or speed to market, almost all lane-based systems have been designed using off-the-shelf desktop packages. In mission-critical applications, like a POS system however, these typically have fundamental weaknesses. These shortcomings aren't in the area of database features, but instead in the way the data is held and updated.

Good TP Databases	Limited TP Databases
QDX	Access
SQL Server	dBase
Oracle	FoxPro
Informix	Clipper
Sybase	R:Base

Fundamentally, there is a difference between a "desktop" database and a "transaction-processing" ("TP") database. A desktop database is a great tool to keep lists and store information for sorting and retrieval, but it is not up to the tasks required of a full transaction processing system. Point-of-Sale is a transaction processing application.

For example, TP databases can provide "Commit" type functions, where data can be temporarily applied until other system functions are able to verify proper application, then the update is committed. "Rollback" is another important database subsystem, where previously applied data can be uncommitted. A database product like Sybase, for example, can even ensure real-time updates across all data screens in a system (and this can also be accomplished in **ISS*45*** reports and IQL-generated screens). ICL's QDX system even includes physical disk locations in its indexing algorithm, virtually eliminating disk seek time. Such systems are rock solid and recover from catastrophic power or hardware failures without data damage. Typical "xBase" desktop applications (dBase, R:Base, Clipper, FoxPro) cannot supply this kind of solid platform.

4GLs, often with their own internal database, are frequently used to speed up development, but there's no free lunch. Besides many of the same weaknesses as mentioned above, these

systems use layer upon layer of software to accomplish tasks. The resulting performance can be very, very poor as the terminal slogs through these application layers to perform even the most straightforward jobs.

The important thing to realize about these issues is that ***no amount of application programming can make up for these platform shortcomings.*** If your foundation is weak, you can't make up for it with a fancy screen, or a LAN or an extra file test somewhere. For example, the xBase architecture, frequently used in lane-based POS, relies on an archaic database structure with a very simple file header that was designed for microcomputers nearly 15 years ago. Data corruption problems are legendary if anything goes wrong.

A recent article in DBMS, a technical magazine specializing in database management systems, discussed this issue in a recent article, "***The Limits of PC Databases***":

> *The consensus among leading mainframe and client/server consultants is echoed by Database Associates' Nagraj Alur, who says, "PC databases [such as] Paradox, FoxPro and dBASE are for single-user applications." ... Others may concede that PC databases are fine for small tactical "workgroup" applications accessing small databases. ... When you get to multiuser DBMSs, you will find that PC systems typically have less sophisticated concurrency controls than most client/server and host-based DBMSs. The chief goals of concurrency controls are increased concurrency and data consistency. This results in performance degradation...*

WHO'S OPEN ANYWAY?

What's an open system? Lane-based vendors blur the distinction between "industry-standard" and "open", but these are not remotely the same thing. According to the Open System Foundation, a true open system must offer three things:

1. System Access
2. Data Access
3. Functional Access

System Access is the capability to get to the system via hardware or software. These issues are usually handled through the use of standard hardware and software platforms, such as DOS, PCs and RS-232. This makes it possible to add applications or new hardware devices such as MICR readers to a system. As a result, most new systems can handle the System Access requirement pretty well.

Data Access is more difficult, especially with lane-based system. Access to data is very limited. How, for example, can you read the data in a given terminal, and from where, using what system or tools? Often, the use of an off-the-shelf database is referenced, but just being able to open and write to a file will not do you any good unless all the "fallout" from this transaction is also handled. This is one of the big advantages of **ISS*45*** and RPS/Supermarket, both of which have tremendous Data Access capabilities via the API. For example, using a single call, a user will be able to update the host controller service, the replicated/alternate service, and all the terminals automatically and instantly. This is because real Data Access in a system is accomplished by ***Systems*** and ***Methods***, not by standard platforms.

Finally, you need ***Functional Access***. As we've asked before, how can you call a system "open" if you can't change the way it works? Both RPS/Supermarket and **ISS*45*** offer powerful tools that can change the features of the system and add whole new functions.

When compared to a real definition of system "openness", the lane-based systems on the market today fail the test. While System Access is typically provided, Data Access and Functional Access are typically weak or missing altogether, while **ISS*45*** and RPS/Supermarket provide real power in these areas.

THE PORTABILITY MYTH

Sellers of lane-based POS like to talk about how they protect the buyer's investment by running on standard PC terminals. The pitch is something like "and if you don't like the software, you can just run someone else's software on the same terminal since it's industry-standard."

Reality is different. Even if the PC is perfectly standard (and some aren't), these terminals typically include special interface boards, keyboards, printers and peripherals which require special drivers and handlers in the software. If the drivers and handlers aren't there for those particular peripherals, then the software won't run on the terminal. Furthermore, these drivers don't transfer from system to system, and most companies hold them confidentially.

So, most lane-based systems support few kinds of terminals, and the only other software that may run on that terminal is another lane-based system which was also designed for that specific terminal. And since SASI, for example, wants to sell SASI terminal hardware, their system runs on SASI, and since Casio wants to sell Casio terminals, their system runs on Casio. It is very unlikely that you'll see a Casio application to run on SASI hardware, (and even less likely the other way around).

These promises have been such a problem that a portability test was suggested at a recent presentation by consultant Jerry Morton to the FDRSG. Vendors with promises using terms like "software portability" or "commodity-based" or "open system" should be asked these questions:

- Can the software run on ***any*** PC, including standard 386, 486 and Pentium platforms?
- Can you unbundle the POS hardware to take advantage of any peripheral components from any vendor?
- Are replacement parts for the POS hardware available on the open market?
- Can the POS hardware be purchased from any POS vendor?

As you can imagine, this kind of litmus test will wilt the extravagant claims of "open portability" and the like.

THE LOW-COST MYTH

Another favorite technique of lane-based vendors to play upon industry misinformation on the cost of PC systems. Because of the destructive and highly publicized $2,000 Sears/Compu-Add deal for POS a few years back (Compu-Add is now bankrupt), some buyers have taken on a mindset that POS can be purchased anywhere for $2,000 per terminal. Lane-based salespeople play to this assumption of lower cost when competing against other types of products, and the truth is conveniently reserved until later.

In reality, of course, the PC doesn't make a lot of difference in the POS system and, in fact, it will often ***add*** cost. Let's look at a lane of POS and see how a PC-based terminal will affect the price compared to a non-PC terminal:

- Scanners and Scales: this accounts for about half the price of a lane of hardware, and there is no difference in cost.
- Printer: a printer of the same quality for a PC terminal will not cost any less.
- Cash drawer: no difference.
- Keyboard: no difference.
- Displays: typically, a VGA display will be supplied with a PC terminal vs. a 2-line display otherwise. A full-size VGA screen will often cost a little less than a 2 x 20 display, but small VGA screens that fit into typical POS counters will cost more.

- Controller: so, you're down to the processor unit itself. An industry-standard PC board will cost a little less than a built-for-purpose terminal board. On the other hand, lane-based POS systems need to have (1) a hard disk, (2) more memory, (3) special interfaces out to the peripherals and (4) a LAN card. All of these items are either unnecessary or included in a purpose-built terminal. These add up to a cost ***disadvantage*** for the PC terminal.

Increasing the cost of PC terminals further, the architecture, database, 4GL, and LAN issues with typical lane-based products also force the use of high-powered PCs in order to get adequate performance. For example, many users are rebelling at being forced to buy the 12 or 16 MB or RAM (that's up to $800 a lane) required for Windows terminal performance.

To sum up, it should be pretty clear that you can't provide a lane of PC terminal POS for less money than an equivalent quality purpose-built terminal. If you see lower prices, it is due to the lower quality of the individual components, especially in the printer, keyboard and cash drawer areas. Everything else is pretty much a wash.

Again, this is not a criticism of PC terminals themselves—ICL manufactures a wide variety of TeamPoS PC terminal configurations, and PC terminals are clearly the future of POS—only the methods sometimes used to sell them. We have frequently seen bait-and-switch tactics used where cheap PCs (even 386SX models) are originally quoted as the terminal platform, but once the order is won, the buyer is moved to big 486 units with high clock speeds to bring performance up to minimum acceptable standards.

THE CHANGING INDUSTRY VIEW

A study was recently presented by Store Systems Consulting, one of the industry's leading POS consultants. The survey group covered major chains as well as independents, and concluded that the fad of lane-based POS systems is dwindling out. Some of the major reasons were issues that RETAIL*partners* have been aware of for a long time:

- Proper consistency and synchronization data is almost impossible to enforce. Out-of-synch systems is one of the most consistent problems of lane-based systems.
- System-wide data integrity is critical: Replication of POS is nice, but Replication of the back office is just as important.
- The higher reliability of terminals makes controller systems more attractive than they have been in the past.
- Instant access to current data and totals usually a problem.
- Lane monitoring, and especially multiple lane monitoring are not possible.
- You can't run any program on any lane.
- The "Data Deluge" from EPS, shopper profiles, in-house charges, and basket analysis is impractical to manage at lane level.

Especially important has been the problem that lane-based systems provide no workable Replication for back office functions. Even if each terminal has its own file, the fact that one failed terminal doesn't affect other terminals has nothing to do with replication of the central POS files. With the importance of central data, back office redundancy has become as important as fail-safe checkout, and lane-based architectures do not provide proper redundancy in a back office.

Because of these factors, and because a system-oriented POS has none of these drawbacks, most grocers stated that they are now more interested in the more powerful architectures. Sophisticated MIS buyers understand the difference, and will only consider full system-oriented architectures like IBM 4690 and ICL's **ISS*45*,** RPS/Supermarket, S18 and ISS400

systems. But **ISS*45*** and RPS/Supermarket are the only system-oriented POS product designed exclusively for the independent grocer.

A PRODUCT COMPARISON

So now let's compare **ISS*45*** and RPS/Supermarket with a typical lane-based product.

Architecture: **ISS*45*** and RPS/Supermarket have true system-based architectures, with full data control and proper data positioning. Current data is always available at the central system point. Data in the terminals is properly synchronized. All system transactions are handled by a rock-solid Transaction Log system from which all files are simultaneously updated.

Lane-based systems usually rely on collection routines at end-of-day. Real-time data and totals are not available from a central point. Data batches must often be applied after-hours only. If collection is "trickle fed", files in a central PC or Master are updated directly and are susceptible by corruption by LAN errors or bad data. Once corrupted, there is no rebuild possible.

Replication, Resiliency & Recovery: With RPS/Supermarket if a terminal fails, is turned off, or is taken off-line, there is no loss of data, either temporary or permanent. Other terminals are not affected, nor is the controller set. Controller redundancy is elegant, fast and automatic. Even dynamic rejoin with automatic re-synchronization of data is provided without front-end slowdown or operator intervention. Files and totals are built and can be rebuilt and updated from the Transaction Log, simultaneously in the Primary and Secondary.

You can turn off an RPS terminal in the middle of a transaction, lose power, etc. and when power is restored the terminal will return to the transaction exactly where it left off. (Don't try this with a lane-based PC terminal.)

ISS*45* offers full Replication, Resiliency and Recovery of all four critical functions, POS, Back Office, Applications and Communications. No other POS system approaches **ISS*45*** in these "3 Rs".

Lane-based systems typically write directly to the files. If they're corrupted, they're gone. Dual disk mirroring is often employed to provide some kind of safety net at the single master unit, but this only provides redundancy for a physical disk failure: if files get corrupted, they are simultaneously corrupted, and no recovery is possible. A physical disassembly of the master and swapping disks is often required here. On other systems, failure of the master unit requires that all terminals be physically rebooted to run stand-alone. When the master is restarted or repaired, all terminals must again be physically rebooted to be brought on line**.**

Hardware Access: The **ISS*45*** and RPS/Supermarket back office PCs are not dedicated exclusively for system functions, even when the store is in full operation. Both PCs are available for any purpose. This is not the case with central office PCs of lane-based systems.

Concurrency: RPS/Supermarket and **ISS*45*** are two of the few systems available that support simultaneous file maintenance sessions. Both PCs, plus workstation PCs, plus FM, etc. etc. can all update, add to and modify the files simultaneously. Batches, including promotion batches, are sophisticated, and date/time activated.

On the other hand, the architectures of lane-based POS have difficulty just keeping lanes in synch, let alone being sophisticated about it. For example, if a terminal is off-line during a file maintenance update, nothing will tell you that the update isn't there. When someone finally figures out that the terminals have the wrong prices, you may have to take the system down and do a full terminal reload over the LAN.

Features: The **ISS*45*** and RPS/Supermarket feature sets have been built up over fifteen years, migrating the features forward into stronger and stronger implementations with consistent performance optimization. Just as an indication, there are over 130 screens of configuration parameters for the **ISS*45*** POS functions alone.

Moreover, ICL has demonstrated the speed at which new features can be added to our systems. For example, **ISS*45*** will have three or four major releases in 1995, each one full of important and significant features. These aren't "catch-up" features, either. At this point, **ISS*45*** has been left to compete with itself, so many of the new features of this system are breaking new ground and putting increasing distance between ICL and competing products.

Lane-based systems have been recently written from scratch with tremendous financial pressure to get the product to market. Slick-demonstrating features such as "Produce ID" are used to get attention, but many traditional and necessary POS features just aren't there. Other key features are implemented in a minimal way to get a "check in the box" on a feature/function matrix, but the depth of the feature, its flexibility, and set-up options just won't be there. In many cases, there is so much left undone that competitors don't even know all the things they don't do. POS is a complex and wide-ranging application and there is no magic: it will be years before they catch up.

Features, II: RPS/Supermarket and **ISS*45*** offer "heavy" system features such as multiple cashier monitoring (without performance degradation), Suspend/Recall (with partial tender), available on any terminal in the store, delayed cashier balancing, 24-hour operation, in-house charges with real-time data keeping. FM verification and maintenance works directly with the core system files.

Such features as these are difficult or impossible to implement properly in lane-based architectures, not only exposing the inherent weakness of such systems, but precluding important and valuable features.

Hardware: Compare ICL's TeamPoS or 9518/200 terminals with the typical PC-ECRs used on most lane-based systems which were developed with low manufacturing cost as the primary objective.

Open Systems: We told that story above on Page 3—you don't need to hear this again. But do look at:

- ***CustomLink:*** **RETAIL*partners*** can respond to their long-term customers' needs with CustomLink. There are versions for both **ISS*45*** and RPS/Supermarket. Transactions at the front end can be added or heavily modified. All CustomLink development is protected and carries forward unaltered with new RPS or **ISS*45*** maintenance and feature releases (which would be impossible even with full source code). In fact, **ISS*45*** and RPS/Supermarket are the only ***practically*** customizable POS systems available.
- ***Definable Item File:*** **ISS*45*** and RPS/Supermarket are the only products on the market with a definable item file. Resellers or users can define, name, report, update and maintain their own set of fields for whatever purpose necessary. Give it a try—you'll be amazed at how straightforward it is to make and maintain these changes.
- ***The API:*** provides software developers the same set of system tools and calls used to implement all back office features in **ISS*45*** and RPS/Supermarket, down to the file management level. This makes it easy to develop, add or interface new applications to the system. In fact, developers consistently tell us that their applications run better on RPS and **ISS*45*** than they do on other applications because of the power of the full API and the automatic data replication it brings. Lane-based POS systems typically offer little in this area**.** Some capabilities to

modify or add reports may be provided with some lane-based systems, but no customization tools or full-featured API is available.

Performance: RPS/Supermarket performance, at both the front end and the back office is fast and efficient. As for **ISS*45***, performance is already legendary with its virtually instantaneous response time and immediate data movement.

With RPS/Supermarket, system scan capability, even with no fast-mover file, is at least 108 items per minute, per terminal, with 30 terminals. RPS has been performance-optimized over many years, and the results show with quick response for the clerk.

With **ISS*45*** clerks can scan and print 3½ items per second per terminal with over 200 terminals in a system.

No lane-based system can match RPS's performance, let alone **ISS*45's*** even scanning from a separate disk in every terminal. With the layers upon layers of software required to build and run applications with 4GLs and high-level database language and application generators (used to cut development time) performance has been a major problem for all such systems, mitigated only partially by mandates for PCs such as 66 MHz '486DX units (or faster) running every terminal.

A frequently-referenced performance problem with a typical lane-based product is how a feature like PLU Lookup or Produce ID[1] puts tremendous stress on the system, and if more than one clerk is using it, the entire system slows down perceptibly. In fact, with some such products, use of these features by three or four clerks acn bring the system to a virtual halt. More than one user is giving up and is replacing such systems with RPS or **ISS*45***. Others, promised anything and everything are filing lawsuits.

By the way, don't fall for "faster PC hardware coming down the pike will fix your performance problem". Just like your copy of Microsoft Word 6 doesn't run any faster on a Pentium than WordStar ran on a 8088 PC-XT 12 years ago, software will always use up any hardware performance available to it. So, if they promise to never enhance the system, yes, a P6 will help. Otherwise, if it's slow now it'll always be slow.

There is no magic. Even with the best tools and the most modern PCs, you can't build a serviceable POS system overnight, and a first-rate one will take years to refine. The sheer number of features in grocery POS, the divergent legal requirements in different states, and the requirement for all features to work perfectly together requires an experienced development team with years of operating system, low-level hardware, driver/handler, database and POS application experience. Then you need the trial-by-fire gained from installations in many Independents.

To Your Success,

Tony

Tony van Seventer
Director: Supermarket Systems

[1] Never mind that you can't really see the difference between different kinds of apples because of the graphics limitations...