

Getting your IT systems to work together: It's not a technical issue

Stephen E. Lipka, Ph.D.

If you're thinking about integrating your information technology (IT) systems, or even buying a new system to integrate into your application suite, you're probably already researched a number of methods and potential products to enable this integration. However, finding the right technology for an integration project is only part of the challenge. In fact, it's only one of five key issues to address before starting an integration project.

Successful system integration in any company requires the following management-level decision-making steps:

- identify the company's business concepts and terminology,
- determine data ownership,
- define the integrity required of cross-system transactions,
- select the right technology, and
- clean up data before integration.

Agreement on Business Concepts

As humans, we take for granted the meaning of certain concepts, such as a "package deal," and we are flexible about details. Systems aren't so flexible; they need precision. Consider the following situation:

- A potential customer relationship management (CRM) system recognizes package deals as separately identified and priced "products" comprised of other products. For example, a book and CD package is recognized as one product for \$29.95. The order and the invoice show the package with its price and the two included product items.
- Your finance system and process consider this package as a series of products and a discount. The book is \$22.95, the CD is \$13.00, and the discount for buying the two products in a package is \$6.00. The invoice shows the package with no price, instead showing three line items with individual prices (including the discount that appears as a negative number).

These two systems cannot be integrated without some translation. If your team does not discover this mismatch until after the actual integration programming begins, you will suffer escalating costs and delayed implementation.

To avoid surprises such as this—and, in fact, to define

your requirements—thorough information mapping is necessary. This can be a substantial task, but the payoff is a significant reduction in future risk.

In the author's experience, it is advantageous to begin information mapping with foundation data—the basic concepts on which others are built. For example, define what you mean by a contact or a company. How many addresses does each have? Is an address an independent entity? How are phone numbers or products treated? Are products grouped by type (e.g., books, seminars, online databases), brands, and product bundles? How are they structured and interrelated?

Once you define these basic concepts, it is easier to understand more complex concepts, such as orders, and their relationship to products and customers.

Knowing the structure of your information can help you integrate in two ways: First, if you intend to buy off-the-shelf systems, it will help you narrow the choices to compatible combinations. Also, if you choose not to restrict your choice so narrowly, this knowledge will help you downgrade less-compatible systems. Failure to understand these compatibility issues up front will cost money and time in the long run.

Ownership of Data

Although information is a company asset, it's simply unrealistic to expect everyone to share the same sense of ownership of all data. Some group or organization within the company (and some system) must "own" products, prices, and customers. Only one database or record for each should exist, so the key questions are: "Who is responsible for keeping data up-to-date, and which system houses each master list?" If your company intends to use one all-encompassing system, then this question of ownership is moot. However, this is rarely the case.

The most beneficial way to choose ownership is to partition your business process to minimize intersystem coupling. For example, if your sales organization recruits customers and maintains the relationships, it may be best for the sales system to "own" customers' information. The sales force can update the contact information, classify and segment it, and perhaps even qualify customers for credit. It's not necessary for the financial system to know about a customer until an order is placed.

On the other hand, if the financial group and its system

were to own the customer data, every change must go through the finance system before the sales system has the latest information. In this case, either sales staff would need to call finance staff to make changes they learn about (increasing organizational coupling) or the sales system will need to send a transaction to the finance system describing the update (increasing system coupling).

If you choose ownership, you will make organizations accountable for the quality of their data. If you choose ownership carefully, you will also minimize the number of inter-system transactions your team must construct.

Transactional Integrity

Today's database management systems operate consistent with a concept known as transactional integrity—either all of the updates in a transaction are recorded or none are.

Consider, for example, your own personal financial management. When you transfer money from your checking account to your savings account, you would never accept a situation in which the money came out of the first account but, because of a system interruption, never went into the second account. Such a mid-transaction failure would separate your money from you and would put the bank's books out of balance. Transactional integrity guarantees all or nothing, no matter how many parts there are to the transaction.

Although applications running on a single database have the benefit of transactional integrity, applications running on separate databases will not. You must construct transactions so that they implement cross-system integrity or at least make visible any partially failed transactions and provide a means for reconciliation.

For example, a company recently wanted to integrate an existing CRM system with a new finance system. In specifying the transactions, they required the receiving system to acknowledge acceptance or rejection of each transaction. Thus, if the CRM system sent an order, the financial system sent an acknowledgment that the order was accepted or rejected. The CRM system then knew to alert users when the order failed. (Note that the acknowledgment from the financial system was not the actual invoice with the invoice number, which is a business transaction all its own, and which may not happen for days.)

Consider, again, your personal financial situation. You can pay your credit card balance with a check (an unacknowledged transaction) and wait for the next credit card statement (a business acknowledgment), or you can send a check by courier that tracks delivery (an acknowledged transaction) and wait in comfort for the next credit card statement (the business acknowledgment). If you never get a statement showing receipt of your check, which method of shipment would give you more evidence that you made a payment?

Companies concerned with certain legislated financial reporting requirements and good accounting practices requiring auditing of all financial transactions must build transactional integrity into cross-system transactions.

Transactional integrity, incidentally, gets more complex when integrating more than two systems. How you implement the acknowledgment is a matter of the technology you choose.

Once you address this key issue, you will have defined cross-system integrity requirements for the integrating transactions you'll use to bridge the two systems.

Choosing Technology

In most literature, integration technology discussions are often accompanied by a fanatical fervor and a good dose of hype. It is possible to approach the matter logically: Choose your integration technology just as you choose any application. In other words, define your requirements and judge the candidates.

Consider some sample areas in which to express requirements. Do you need:

- transactional integrity,
- real-time integration (or are batched transactions sufficient),
- data mapping (e.g., temperature is °F in one system and °C in the other, or price is in cents in one system and dollars in the other),
- concept mapping (e.g., the package deal example mentioned earlier), or
- integration of two systems (in which several data transfer tools might do) or more systems (for which more sophisticated transaction management is needed)?

Other questions to consider include:

- Do your existing applications have limits on the technologies they can use, thus limiting your choices?
- Do some of your applications (commercially available packages) already have integration technology embedded, limiting your choices?
- Will your integrated system be subject to scrutiny in audits, and, if so, what requirements exist for control and auditing?
- Will the chosen integration technology be used only between these systems, or will it be adopted as a company standard? Would the latter choice change your requirement for vendor longevity and quality of support?

Once you address these key issues, you will have chosen the mechanism for integration, and you can proceed to construction.

Data Hygiene and Integration

Data hygiene—cleaning up the data—may not seem related to integration, but consider the following situation one company faced:

The company constructed an integrated CRM–financial system from two application packages and an integration engine, replacing two standalone implementations. One month before launch, they began looking at the data that would populate each system from the replaced source systems. They found a number of situations in which bad data would lead to automatic creation of much more data:

- One company had two names in one system (e.g., “IBM” and “I.B.M.”), both of which were unintentionally separate accounts. There was also another name (e.g., “International Business Machines”) in the other system. This presented an obstacle to initial integration. They knew that, at time of launch, multiple names for the same corporate customer would result in the creation of multiple accounts in the new accounting system.
- The old CRM system had more than 2,000 accounts, many of which were inactive. The old accounting system also had more than 2,000 accounts, many of which were inactive, but not the same inactive accounts as noted in the CRM system. After data cleaning took place, fewer than 800 active accounts matched. Therefore, they knew that without archiving inactive accounts, the new financial system would have a set of approximately 1,200 inactive accounts (the number of “dead” accounts remaining after identifying the 800 active accounts) plus an additional 1,200 inactive accounts propagated from the CRM system.
- The old CRM system was not extended or modified much to avoid upgrade complications. In the old system, clever sales representatives considered unused fields as the perfect place to save information not housed elsewhere. For example, a field labeled “contact’s manager” was used by one representative for the name of the contact’s assistant and by another representative as a tickler/reminder. If the newly integrated systems were to use this field, “garbage” would be propagated from system to system.

Data hygiene and lack of standards have effects in two circumstances: Integrating two existing systems (or integrating their replacements) requires that the data be consistent across systems when integration is established. In addition, it’s problematic when there is no owner system for some data or when, for some reason, duplicate databases exist in two systems that are integrated. For example, if sales staff enters “Microsoft” as an account in their system and finance staff enters “Microsoft Corporation” into their system, cross-system transactions will not recognize the

account names refer to the same company. This reinforces the need for data ownership.

Conclusion

Integration projects that only focus on technology will suffer many surprises. In the author’s experience, technology is the enabling tool but the real issues and challenges center around integrating the business processes and information.

Definitive recommendations include:

- Map your business concepts so that organizations and systems destined to be integrated talk the same language.
- Decide which organization and which system owns each information entity. The owning organization will be responsible for standardizing the data across the company. The owning system for the data will be the source for all other systems.
- Decide whether data passing between systems to be integrated is just data (e.g., transaction system downloading to a data warehouse) or whether it’s really a business transaction. If it’s a transaction, decide the level of required transactional integrity and ability to be audited.
- Choose your integration technology application package by defining the requirements and evaluating the alternatives.
- Clean your data before you integrate and keep it clean to avoid propagating garbage across systems.

Stay aware of these definitive recommendations and your IT systems will work as well together as you intend them to.

About the Author



Stephen Lipka is a Principal of Avatar Strategic Partners, where he serves companies that want to improve their performance through better use of information technology. His career spans 31 years in management, consulting, and product development. Steve received his Ph.D. in Computer Science at SUNY Stony Brook. Reach Steve at slipka@AvatarSP.com.



Visit us at www.AvatarSP.com