The use of IBM SanFrancisco core business processes in human resources scheduling

by E. J. Jaufmann, Jr.
D. C. Logan

The authors review their experience in adapting IBM SanFrancisco™ software to use outside of the financial problem domain. The IBM SanFrancisco warehouse and order management core business processes are explored, showing how the components and processes contained within the core business processes are being adapted for use in a scheduling and human resource tracking system to reduce product investment and increase reliability. Additionally, techniques for migrating from relational database "legacy" systems to SanFrancisco core business processes are discussed.

Provider Solutions Corporation is a technology company focused on the development of Internet offerings and software component technology for the health care industry, with particular emphasis on the post-acute segment. Over the last 12 months we have been migrating the entire product line of the company to a technology platform based on IBM SanFrancisco* software. During that migration we have been constantly reviewing what components and processes within IBM SanFrancisco software can be used to reduce the migration effort and replace portions of our technology, with the goal of limiting the components created by us to only those not already available in the SanFrancisco software.

As described by IBM, "IBM SanFrancisco is a Java** -based set of components that allows developers to assemble server-side business applications from existing parts, rather than [to] build [them] from scratch. Pretested SanFrancisco components enable developers to build and modify business applications quickly. Cross-platform applications can be built once and run on a wide range of servers, including Windows NT**, OS/400* [Operating System/400], AIX* [Advanced Interactive Executive], Solaris**, HP-UX**, and Reliant UNIX**. SanFrancisco is the largest server-side Java initiative in the industry and is a key element in IBM's Application Framework for e-business."[1]

When we started looking at the content provided by the SanFrancisco software, it first appeared to be focused solely on the financial aspects of business. An examination of SanFrancisco core business processes, including warehouse, order management, general ledger, and accounts receivable/payable, seemed to show no core business process that would assist in the development of a nonfinancial health care Internet offering.[2] This is not to say that the company hierarchy process and other SanFrancisco common business objects could not be used, but

there appeared to be no business processes that could be used in their entirety.

What we saw was not a surprise, since IBM SanFrancisco software is often described as an object or component framework with financial business process content.

The problem domain of scheduling and human resource tracking defines a large number of health care business requirements including:

- Selection of resources (i.e., health care workers) for one or more appointments based on the type of resource needed for that appointment, relative distance of the resource and location of appointment, personal preferences of patient and resource (smoking, animals, stairs), and previous contact between resource and patient
- Searching for available resources in other resource pools based on distance and business relationship between the searching organization and the potential supplying organization
- Certification of a resource to work at an appointment that will be paid under a particular managed care contract within a particular jurisdiction. This includes checking on whether the resource meets all human resource requirements of both the contract and the jurisdiction, including—but not limited to—licensing, training, and immunization.

We decided to revisit use of the SanFrancisco core business processes despite the effort we had already invested in deriving SanFrancisco entities from our legacy applications. We concluded that the investment in the creation of the entities that mapped to the relational data was relatively small, and therefore, even if none of the new entities was reusable with the core business processes, the expected reduction in continued investment would outweigh the loss from abandoning the existing entities.

Taking an unbiased look

Among the entities included with IBM SanFrancisco software[3] are: warehouse, product, lot, unit, serial number, order, quotation, customer, contract, and invoice. The business processes in SanFrancisco software include: picking, stock transfer, back-to-back ordering, and discounts.

In our "fresh look" at the SanFrancisco core business processes, we considered which processes and entities map well to the processes and entities in our application domain and what questions pertain to the mapping. For example:

- What is a scheduling organization?
- How do resources relate to the organization?
- In an extended organization, when and why does one organization go to another for resources?

As we looked at the questions listed above and continued to review the SanFrancisco software, some basic concepts became apparent:

1. Each branch or office of an organization has a set of resources associated with it.
2. Each resource associated with an organization has a finite availability, and once a resource is assigned to an appointment, that part of that resource's availability used for the appointment was "sold" and no longer available.

These two concepts had striking similarities to statements that could be made about a warehouse and product, namely:

1. Each warehouse of a company has a set of products associated with it.
2. Each product within a warehouse has a finite number of units, and once sold, that unit cannot—when it is a stock-controlled product—be sold again.

Further logical tests were undertaken to see whether other SanFrancisco components and business processes would be usable in our domain. One of these tests produced the following result:

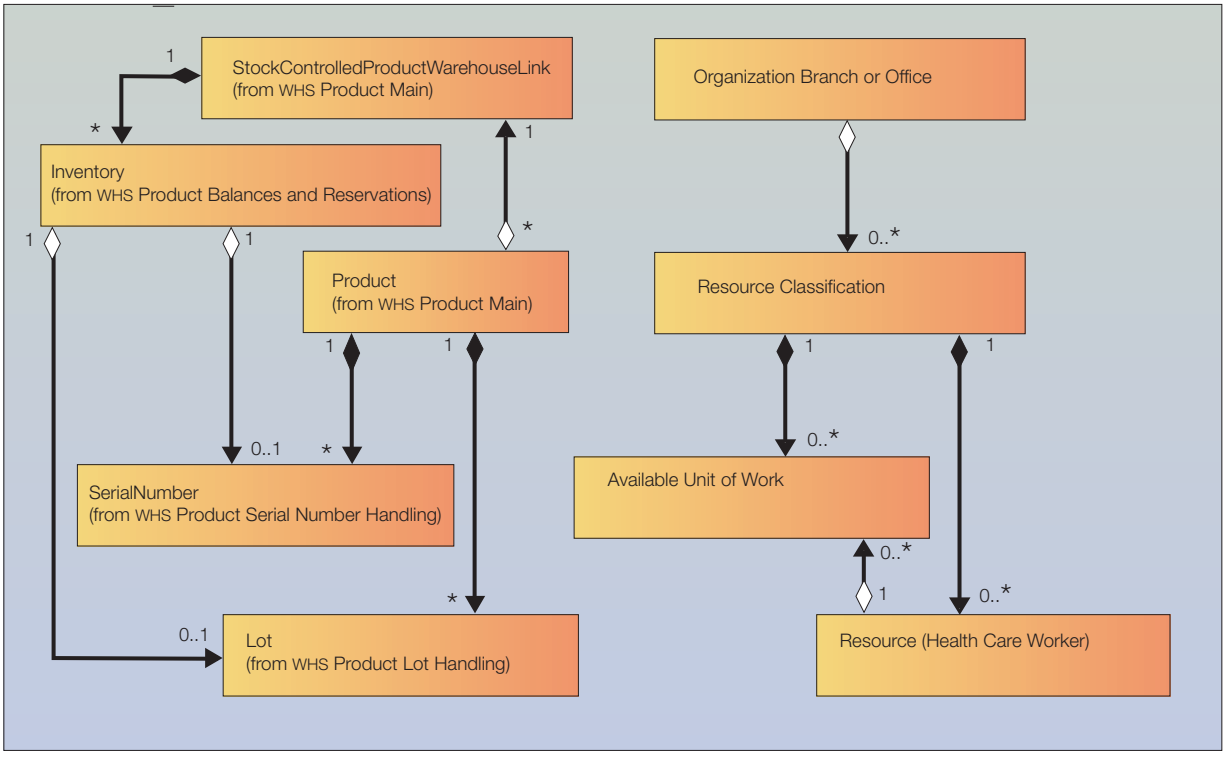
> Upon not having available resources to fill an appointment within an organization, check with other organizations, taking into account geographic and pricing considerations, for the same type of resource.

The previous statement has similarities to the following statement:

> The SanFrancisco warehouse sourcing process will search the default warehouse and, if the default cannot fill the order with the needed number of units of the ordered product, then based on the warehouses associated with a product, search through the available warehouses to find one that can source the needed units of a product.

Such tests started us down a line of questioning concerning the relationship between our entities—resource, appointment, patient, payer, and organiza-

Figure 1 How health care entities correspond to SanFrancisco components



tion; and the SanFrancisco components—product, business partner, and warehouse. Did a resource really correspond to an IBM SanFrancisco product? How did the various processes and other components associated with the product map to our modeling and "legacy" products?

Mapping to SanFrancisco

After some review of the IBM SanFrancisco component documentation, including SanFrancisco business scenario documentation and javadoc, we found the following parallels:

1. A SanFrancisco product maps to a resource classification rather than a resource, for example: registered nurse, occupational therapist, home health aide, and personal caregiver.
2. A SanFrancisco lot, a collection of units of a product, maps to a specific resource.
3. A serial numbered unit of a product within a lot maps to a specific unit of work for a resource (such as a unit of time or a care visit) that is available. The serial number of the unit contains information about the actual date and time of availability.
4. A SanFrancisco warehouse maps to a branch office of an organization.

The mappings described above and shown in Figure 1 allow the entire process of scheduling appointments with qualified caregivers to be modeled within the SanFrancisco components and business processes already available within the warehouse core business process (CBP). Additionally, since the warehouse CBP is directly linked to the order management CBP, the components and processes within that CBP become available as well. Some examples follow:

1. It is always preferable to have a continuum of care from caregivers experienced with a patient. Therefore, we always try to assign resources previously assigned to a patient to new appointments for that patient. The IBM SanFrancisco warehouse CBP has planning and picking policies that govern what product, lot, and serial-numbered units

are withdrawn from inventory at what warehouse to fulfill an order. In our case, this means that by using the extension point of the picking policy, we can look for the availability, within a skill (product), of previously assigned resources (lot) that have the correct time available (serial numbered units) for the appointments to be filled.
2. As stated previously, should a resource not be available in the branch or office that received the appointment request originally, then other branches, offices, or organizations with relationships with the originating branch or office should be queried for a suitable resource. This case appears to be covered in the planning policy by configuring which warehouses stock a particular product in the SanFrancisco warehouse CBP. The extension point will allow specific tests for geographical distance (logistics) and cost to be taken into consideration in the planning. Additionally, SanFrancisco order management covers back-to-back orders. In the case where another organization's (warehouse) resource's time (unit) would be used to fulfill an appointment (order), a back-to-back order is automatically generated to go to the other warehouse.

Introducing SanFrancisco order management with its coordination with warehouse appears to provide more potential reuse capability for developing applications in the health care problem domain than we originally understood.

Comparing individual patient appointments and a series of prescribed appointments with SanFrancisco orders, the following logical similarities were identified:

1. When a patient requests an appointment or a series of appointments to be filled with resources (products ordered), an estimate of the cost (quotation) can be created. SanFrancisco order management already has an order life cycle taking into account the creation of quotations from customer requests for products.
2. When a patient agrees with the estimate (quotation), appointments can be booked (order), and the patient's insurance and other sources of payments can be checked to determine the financial risk (credit check). SanFrancisco order management already has extension points for credit policies and overrides of credit holds on orders.
3. When a patient has received the services of a resource (order delivery), the various payers for the service, including insurance carriers, government agencies, and guarantors are sent an invoice (invoice). SanFrancisco order management has extension points for generating invoices on verification of delivery of orders to customers.

Development process

How do SanFrancisco mappings impact our development methodology?

Provider Solutions Corporation develops all of its products using the Unified Software Development Process (USDP) that was developed and documented by Jacobson, Booch, and Rumbaugh of Rational Software Corporation.[4] USDP is a full life-cycle process covering the structuring of requirements through testing, packaging, and distribution of software systems.

The primary tool used to document the various artifacts developed with the USDP is the Unified Modeling Language (UML),[4] a set of diagrams representing different aspects of systems, including use case diagrams that structure requirements, collaboration and interaction diagrams that document the interaction of objects, and class diagrams that document the static relationship between and the attributes of classes.

Each project is divided into a number of separate subprojects that we refer to as "slices." Each slice is a deliverable subproject, with defined scope, the aspects of which suggest a high potential for successful delivery. Each slice cycles through four phases: inception, elaboration, construction, and transition.

Inception centers around the structuring of business requirements. The artifacts delivered include use case diagrams, the domain entity model, which is a form of class diagram containing the entity objects representing major nouns in the use case diagrams, and the prototype user interface.

Elaboration extends the artifacts from inception producing: use case diagrams with greater detail, including secondary functionality; the prototype user interface, also containing greater detail; collaboration or interaction diagrams, showing the interaction between objects discovered in development of the use case diagrams; and class diagrams, giving full implementable detail to the classes discovered in the collaboration diagrams, including class relationships.

Table 1 USDP phases and SanFrancisco adaptations

Phases	USDP	Adaptation of SanFrancisco Software
Inception	Use case diagrams	Compare use case diagrams to SanFrancisco CBPs to identify candidate CBPs
	Domain entity diagrams	Review domain entities for proper inheritance from and relationship with SanFrancisco core business objects (CBOs)
	Prototype user interface	
Elaboration	Detailed use case diagrams	Compare use case diagrams to SanFrancisco CBPs to make final CBP identifications
	Detailed domain entity diagrams	Review domain entities for proper inheritance from and relationship with SanFrancisco CBOs and patterns
	Detailed prototype user interface	
	Collaboration and interaction diagrams	Incorporate CBPs concentrating object modeling on policies and other objects required for incorporation of CBPs
		Model business logic to entity collaboration for business process that cannot be replaced with CBPs
	Class diagrams	Model inheritance and relationships of classes to be generated with SanFrancisco code generator
Construction	Collaboration and interaction diagrams	Same as elaboration, necessary detail
	Class diagrams	Same as elaboration, adding relationship detail and schema mapping, and code generation
Transition	Configuration	Configure SanFrancisco CBP and classes for testing

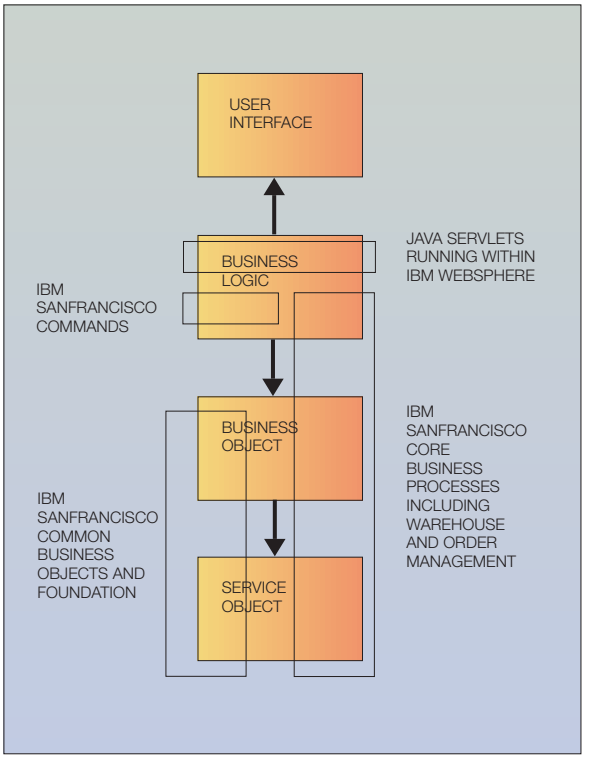
Construction extends the artifacts from elaboration to produce highly detailed collaboration and class diagrams, ready for coding, including inheritance among classes.

After we model the requirements by means of use cases described above, we refine the use cases and start object modeling. This stage is where SanFrancisco makes a major impact. We compare the delineated problem with the available SanFrancisco components and processes. We review our current product and then look at the potential reduction in investment, both immediate and long term, that could be derived through the reuse of SanFrancisco components.

Our next step is to refine the use cases and start object modeling. Again, SanFrancisco makes a major impact. We review our current product and then look at the potential reduction in investment, both immediate and long term, through the use of the SanFrancisco components and processes. We undertake more detailed use cases and domain entity models with this comparative information, which is integrated into our models. Last, we start our object modeling—specifically collaboration and class diagrams—using SanFrancisco components, through inheritance, extension, and direct means.

We have adapted these phases to include the incorporation of SanFrancisco content as much as pos-

Figure 2 Package relationships and mapping of SanFrancisco to architecture



sible. Table 1 shows the USDP phases and the adaptation for use with SanFrancisco.

The incorporation of IBM SanFrancisco common business objects (CBOs) and CBPs into the USDP has shortened our product development life cycle significantly. As an example, efforts to implement one product, Provider Solutions Outcomes Manager Internet Offering, was estimated at 12 staff years without SanFrancisco. We accomplished implementation of version one using SanFrancisco in under four staff years.

In our use of USDP, we have developed a logical architecture separating the objects within our systems into four separate packages. Figure 2 illustrates the dependency relationships between the packages and how various IBM SanFrancisco components and processes map to the logical architecture. To review the responsibilities of each package, see Table 2.

By reusing both SanFrancisco CBOs and CBPs, we were able to eliminate a significant amount of modeling and construction. Service objects are provided by SanFrancisco and require no modeling or development. Business objects are either 90 percent provided by SanFrancisco when modeled and generated or 100 percent when used within a CBP. Business logic is about 60 percent provided by SanFrancisco when modeled and generated or 80 percent when used within a CBP.

In particular, as mentioned earlier, CBPs provide extension points, often in the form of policies, which allow a CBP to be customized without the need to alter the actual CBP directly.

Legacy considerations

How are the "legacy" applications integrating with IBM SanFrancisco component technology?

In order to migrate some of our products, we have successfully undertaken the approach of using relational tables to create objects within SanFrancisco entity owning extents.[4] Entity owning extents are SanFrancisco object collections that integrate well with existing relational database tables. They allow the tables to be used to migrate a function from an existing application base to a new SanFrancisco-based application with the only change being the location from which the end user will access the migrated functionality. Mapping relational database tables to objects is accomplished through the use of the SanFrancisco Schema Mapping Tool. This tool maps individual attributes of a class to fields in one or more relational database tables.[5,6]

One function migrated as stated above is the intake/referral function of the Provider*FIRST*** Scheduler. This function, which allows a new patient to be added and initial scheduling to be undertaken by an admitting professional, was successfully migrated from the available version of the product to our SanFrancisco-based version of the product. The SanFrancisco version of the intake/referral functionality is being distributed as part of the entire product. The admitting professional uses the SanFrancisco-based intake/referral functionality, including thin client interface, while other users employ all other functionality as they have in the past. With a large number of entities being created by this migration process, the question was also raised as to how the CBPs would be used with these entities.

In many cases the point of integration is the relational database, as indicated previously. In some cases, we have had to take an alternative approach.

Table 2 Object packages and their responsibilities

Package	Responsibilities
User interface	Objects associated with presentation services—not limited to classic visual display, but includes nonvisual forms of communication such as transfer of data to other systems, as well as printing and voice response
Business logic	Components encapsulating business processes
Business objects	Business entities such as patient, employee, and company
Service objects	Objects that separate business objects from the physical implementation, such as persistence storage

The movement of scheduling information to the warehouse/order management CBP requires some duplication of work. One approach we took to synchronize the relational database and the IBM SanFrancisco warehouse/order management persistence store scans the "legacy" database for new entries. In this approach, we instantiate a command object with the information from each entry. Each command, depending on the type of entry, executes the appropriate command in the warehouse/order management-based application to duplicate the work.

We have tried to minimize the need for this type of synchronization as much as possible by creating sets of related functionality that allow a clean delineation between the SanFrancisco component-based functionality and "legacy" functionality. This has been moderately successful, with the exception of caregiver availability and patient orders, which are duplicated in both technologies.

Another approach we have used successfully is to utilize "mirrored" commands when completing a particular business process. One command executes the business process using the objects derived from the "legacy" tables, while the other executes the same business process using the SanFrancisco components. Another command is created that is used by the client logic to execute the particular business process. This "master" command simply executes both of the "mirrored" commands within a single transaction. This dual execution ensures that either both, or neither, of the command results are permanent. Should one command fail, the transaction will be rolled back, ensuring that both business processes either succeed or fail.

In summary, the use of the USDP clearly defines all of the business requirements being satisfied by the products developed by Provider Solutions Corporation. Use case, collaboration, and class diagrams as part of the total development life cycle outlined in the USDP[4] created an understanding that was used to review SanFrancisco CBPs for possible employment within our product development efforts.

By setting aside preconceptions concerning use of the CBPs and matching the processes and components contained in the CBPs with concepts within the problem domain, in this case health care, we were able to achieve use of the CBPs in unique ways. As one example, the warehouse and its constituent business components and processes store human resource availability and match the best resource available with an order for care.

Last, by schema mapping SanFrancisco objects to relational data we are able to develop new SanFrancisco-based functionality while still using the same database for already-existing systems. This coexistence allows a stepwise migration to the new systems based on SanFrancisco CBOs and CBPs instead of the need for a "big bang" conversion process, where all systems and all functions must be converted at once.

Conclusion

We believe IBM SanFrancisco components and processes are much more than a financial problem domain framework. The key to making this idea work is good modeling and analysis of the problem domain, followed by an open-minded review of the available SanFrancisco components and processes with respect to the developed models. Our experience is that this approach has the potential to radically reduce product development investment for highly complex systems outside of the financial problem domain.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Sun Microsystems, Inc., Microsoft Corporation, Hewlett-Packard Corporation, Reliant Software, or Provider Solutions Corp.

Cited references

1. *Introduction to IBM SanFrancisco*, SanFrancisco documentation, IBM Corporation.
2. *Deploying Applications—Persistent Object Planning*, SanFrancisco documentation, IBM Corporation.
3. *SanFrancisco Warehouse Management User Guide* and *Order Management User Guide*, SanFrancisco documentation, IBM Corporation.
4. I. Jacobson, G. Booch, and J. Rumbaugh, *The Unified Software Development Process*, Addison-Wesley Publishing Co., Reading, MA (1999).
5. IBM Web site at http://www-4.ibm.com/software/ad/sanfrancisco/tools.html#SchemaMappingTool.
6. *Deploying Applications—Using the Schema Mapper*, SanFrancisco documentation, IBM Corporation.

Accepted for publication December 16, 1999.

Edwin J. Jaufmann, Jr. *Provider Solutions Corporation, 4905 West Laurel Street, Suite 200, Tampa, Florida 33607 (electronic mail: Edwin.Jaufmann@Providersolutions.net).* Mr. Jaufmann has been Chief Information Officer since January 1998. He has over 20 years experience in information systems management, architecture, design, and development. Prior to joining Provider Solutions, he spent nine years as a consultant on the management and architecture of large projects to *Fortune* 500 companies including Aetna, MetLife, Bell Atlantic, and SIAC (Securities Industry Automation Corporation). Previously, he served as Vice President of Systems Development for the Chase Manhattan Bank, responsible for the Chase Automated Clearinghouse. Mr. Jaufmann has earned the designation of Certified Computing Professional from the Institute for Certification of Computing Professionals.

Daniel C. Logan *Provider Solutions Corporation, 4905 West Laurel Street, Suite 200, Tampa, Florida 33607 (electronic mail: Dan.Logan@Providersolutions.net).* Mr. Logan is Product Architecture Manager. He has 11 years of experience in the design and implementation of information systems and five years of experience in object-oriented analysis, design, and implementation. Prior to working at Provider Solutions, he spent nine years developing systems for the financial services area of Aetna, Inc. Mr. Logan received a bachelor of science degree in mathematics and statistics from the University of Connecticut.