Reallocating SRDB Functionality

CSG Meeting
January 25, 2005
Today’s Administrative Systems

This is an old slide, so “today” was originally about 3 years ago. The area in the black rectangle is what I will call the “Student Data Domain.” The changes since then are shown by the red text and lines. Financial Aid and BAR—the two areas that had converted from VSAM to DB2 in the nineties—are now able to present data in the QDB. Admissions has also fully converted to DB2 and is now starting work on making data available in QDB.
It is probably instructive to look at this domain from a slightly different perspective. Today, we have a mainframe transaction system that services “green screens” directly, supports a variety of campus systems (many through SRDB) and supplies reports programmed in either COBOL or Focus. The plan has been to move most of the reporting to a new reporting environment running against the new data warehouse. Secondly, the SRDB, which looks somewhat like an Operational Data Store, would become part of the data warehouse structure. That was, of course, before the totality of the SRDB functions were fully understood.
This gives rise to what was called the “Thin Mainframe.” Not that the mainframe has gone on a diet, but the idea was that by optimizing the transaction system for transaction processing and the data warehouse for reporting, we could free up considerable CPU time on the mainframe and provide faster turnaround on reporting. That, of course, in addition to the additional analytical capabilities that the warehouse would provide. Thus, with the transaction system redesign under way, it is now time to focus on the area inside the red oval: Campus Data Warehouse, reporting, and data feeds to the campus.
So, what are we doing?

We reviewed some 500 pages of listings of stored procedures and recognized that there are definite usage patterns. These were brought into clearer focus by asking the 15 or so top users, representing 80% of the objects in the SRDB, about how they used the information. (And, yes, the users with the largest number of objects also are the ones that generate most of the SRDB transactions.)

From this I created a first rough abstract machine model, which I then discussed with the developers of the student records system, the data warehouse and the identity management system. The model was iteratively modified, until it reflected how student systems ideally should be architected and interact with the rest of the campus systems portfolio. In a minute I’ll show you what that architecture looks like. You may already be familiar with this, since we distributed the results of the analysis to the query participants.

Right now we are where the line is on the slide. The next task is, of course, by far the most difficult to execute. The service architecture that results from the campus needs represents an ideal and an abstract one at that. However, we live in a world of real systems that we must build, legacy systems that must be accommodated and technical constraints that limit what is practical (or affordable).

The task now is to sort out how we can implement this architecture practically. Obviously it isn’t going to happen all at once, and there will be exceptions that “violate” the ideal architecture because we do have legacy systems and we do have third-party software. And, of course, some things are simply too expensive to do at this time, or it is imprudent. We are, for example, trying to put things in place as standards emerge from the various consortia and work groups. That isn’t always possible, of course, but in that case we’ll make an effort to minimize the rework at a later time. That’s the intent at any rate; we’ll see whether we have the smarts to actually do it.
Where the previous slide looked at this pretty much from the technical perspective, we should also look at this from the functional perspective. The transaction system records events during the student “life cycle.” The process is, of course, iterative; although usually recruited and admitted once, students will every quarter register, enroll (drop and add) various classes, receive instruction, and get grades.

To satisfy purists, who might recognize this as Porter’s value chain model, I’ve added “Alumni Management,” which is pretty much a “post sales service” activity. (Many would put Athletics there, too, by the way…)

Anyway, all of these processes generate data in the Student Records Database. Of we look at what is really there, it falls into two of the areas in the new data warehouse: “sources and uses of funds” and “academic process,” and mostly the latter. Those are the “facts” in the dimensional model; “student” is actually a dimension: a way to look at the facts. So one can look at the data from the perspective of registered student, enrolled student, 3rd week student, etc.

Also note the gray-blue items. These are functions currently performed by student systems, but they are not unique to the entity “student.” That is, basic information about a person (“demographics” for short) are collected for students as well as staff and faculty. They are attributes of the “person” entity, of which “student” is a subset. Likewise, students are only one group of entities that require “Authorization and Communication.” This is the information that determines who can access what and the address information so the university can communicate with students and they can interact among themselves – in a course chat room, for example. We’ll come back to that later.
So where the transaction system primarily supports the “life cycle” or “primary” processes, the data warehouse will provide the data for the “support” processes. Those may be pretty tightly coupled, such as Class Scheduling, or as far removed as “Institutional Research.” The Registrar has a whole slew of things that in the new architecture will be supported through the data warehouse. In that respect they pretty much “just another campus user” of the data warehouse.
I am sure that everyone in this group is familiar with abstract machine models to express a service architecture. As we started down the path of analyzing how we really want to allocate the various functions, it becomes clear that there are essentially three sets of services:

1. Those of the transaction system, dealing with the life cycle
2. “Published” services, which support global campus needs
3. “Private” services, which support internal operations of a particular campus unit.

Note that all of these are supported by a collection of data management services, but that the consumer of the student data services should not need to know the particulars of how they are supported. Note also that the items that were flagged earlier as being “improperly” attributed to students, are now shown as directory and authorization services.

If the original concept of SRDB as an early ODS were correct, it would roughly cover the ODS component of the Data Management Services layer. In actual fact, it covers something more like the gray rectangle. SRDB provides some ODS functions, although data warehouse purists might consider it more like the staging area of the data warehouse. In actual fact, SRDB also provides chunks of “business logic” for other campus applications, provides (part of) the authorization for various campus systems, and produces customized student directories. It supports various operational processes within the Registrar’s Office and, additionally, provides the middleware that bridges between mainframe production schedules and 24/7 web access.
When discussing the reallocation of function it is important to distinguish between the physical implementation of a service (in the technical sense) and the “consulting wrapper” that comes along with the service. The current transaction system cannot stay the way it is; therefore, what is in the SRDB also has to be changed. And it isn’t just the restructuring of the student data: single sign-on, enterprise directories, need for increased security at many levels, and virtual learning environments / course management systems all have an impact as well.

Also, some of the services provided by SRDB were created because the old technology wasn’t capable of supporting such modern things as the world-wide web. As we refresh the technology portfolio on campus it doesn’t make sense to layer middleware “glue code” on top of more glue code. Instead, we want to use these new technologies natively so they are an effective and efficient building block for the future.

But after all the technological wizardry has taken place, there remains the fact that the “real” expertise in student data is in the Registrar’s Office. The RO is expending significant resources to migrate to the new architecture because (1) in the long run it will allow more efficient support of the campus, (2) running a data center is not the core competency of the RO. 😊

The fear of the campus appears to be that when the technical changes are made, the Registrar’s support will also go away. I am not aware of a decision in that direction – admittedly well outside the scope of my mandate – but it would appear to me that should the budget pressure be high enough, then the Registrar can make that decision in any case. Should that happen, the campus will be far better off with far more user-friendly and “user-usable” – think metadata – tools.
So where are we right now? The gray-blue stuff from a bunch of slides ago is currently being addressed. We are trying to sort out how much of the student data domain should and can move into the EDIMI. Conceptually this is what directories do: authenticating and authorizing, and opening communication channels. As a practical matter we need to sort out what level of granularity can be supported in which environment. This is partly a technical matter, but it is probably also a licensing matter.

We may find that the architecturally and technically most desirable solution is not affordable, or we may find that the applications used on campus don’t integrate with the ideal model. So, at the moment we are working our way through understanding the options and figuring out what will actually meet these needs. It is likely that this will not be a static model, as standards evolve and systems are upgraded the implementation of the services may change. If we design it right, though, the impact on the campus should be minimal. That is one of the beauties of the service model: as long as the interface holds, nobody really cares how the service is implemented. [Yes, I know, theory and practice are usually much closer in theory than in practice…]
Some Authorization Issues

- In theory EDIMI is web-analogue to DACSS
  - Single campus-wide authorization service
  - Improves security and access control
- Practical impediments:
  - Required security model is beyond scope of EDIMI
  - Industry standards are still in the making
  - Course management systems implement their own
  - Inconsistent interface to laboratory access controls

Here is just a sampling of issues we are looking into. This is not an exhaustive list and your pet issue probably isn’t listed.
None too surprising, the interaction with campus systems is the most complex issue. We have already talked about the Registrar’s Office support issue, but there are technical complications as well. Ideally “nothing” changes, but that is highly unlikely. If you are making a remote procedure call now but will need to send an XML message in the future, you will not only have to change the code that produce the request. The stuff that comes back will look a bit different, too... Whether this means that the application is rewritten depends on how clean the application design is and on the middleware services that might be made available as part of the new architecture.

If you have a clean design, our expectation is that you will be able to make use of the “published” services. Most of the fear, uncertainty and doubt is concentrated in the “private” services. If things aren’t designed cleanly and/or you depend on a lot of support from the Registrar’s Office, what is going to happen? The answer to that question will depend a great deal on each individual situation. Just because the service interfaces haven’t been defined yet doesn’t mean that they will be “bad.” The intent is to make the transition as straightforward as possible. How that plays itself out is a function of the systems on both sides of the interface. Also, there are still some policy issues to be addressed: access authorization and methods, availability of work space, etc.

It is clear that this run-through today is only the beginning of a considerable and ongoing dialog over the next year or so as specifications are developed and test environments enabled. The next two slides give a sample, but by no means the full set, of issues that will be addressed.
Some myUCLA Issues

- **Posting of enrollment transactions**
  - Mainframe is not available 24/7 to process
  - SRDB provides a “store and forward” service
    - Registrar's policies still to be formulated

- **Gradebook**
  - Tightly coupled with the SRDB logic
    - Will require significant redesign
  - Most course management systems have their own
    - Unclear whether they provide better solution

If one component of student systems is up when another isn’t, we have to resolve the synchronization issue. Without making light of the fact that we need to solve this problem, it isn’t like nobody’s ever looked into it. It is essentially exactly the same problem Amazon.com has when you place an order over the web. The “real” inventory system may or may not be up; but you still can order your book. The gradebook provides yet another set of issues: everyone is at cross-purposes here. The Registrar has apparently mandated its use but, from what I know, it isn’t compatible with most CMS systems and from what Eric tells me, it may need considerable rewrite when data comes from a different source than SRDB.
Some Campus Concerns

- Many SRDB interfaces will have to be rewritten
  - Underlying (TXN) data model has been changed
  - Impact unclear until campus has Cognos experience
    - Quite a few naïve users in smaller units
    - A lot are one-of-a-kind procedures
- AIS’ USR process too slow for most changes
  - Fact: Registrar can continue to provide same service
    - Level of service doesn’t depend on technology
  - Fact: Users have more flexibility, if they so desire
    - Policies on control of use have still to be developed

Campus concerns are, understandably rampant. Of course, since the transaction data model and system changes, there will be further changes. How severe those are and whether the Cognos toolset makes them easy to deal with still remains to be seen. Even if that is an option for some units, it won’t be for others. The difficulty is that the people in the Registrar’s Office will also have to do the handholding; this is a resource bottleneck that we are trying to address.

In closing, I do want to address the urban myth that when the new data warehouse goes into production, everything that you want to get done becomes a USR and therefore turnaround will be terrible. In actual fact, the user community will have access to everything they are entitled to based on existing security and privacy constraints. Today the Registrar sits between the data and the user. The tools will enable you far more direct access, but the policies on doing that have not yet been determined. Of course, you may not want to (or have the ability) to make use of this capability on your own. I can’t promise that the RO is going to stay in the handholding business forever. They were never funded to do that; but there was no other way to do it. On the other hand, they are very concerned that people will use the data inappropriately, so I assume that they’ll want to keep a finger on what’s going on for quite a while.