CIDR Center for Inherited Disease Research
Johns Hopkins University

Implementing a Third-Party Inventory System at CIDR

As a high-throughput genomics service provider, CIDR is home to wide variety of lab and computer equipment, some with very hefty price tags. These are thousands of items that must be tracked for accounting, warranty, and inventory purposes. While all of these items belong to CIDR, they were paid for and are actually owned by either NIH or Johns Hopkins University. In the latter case, such purchases must be linked to multiple budgets, and for some we must depreciate part of the cost over several years. Historically, we have used several large, complex spreadsheets to manage this information, and this scheme had become unwieldy. Accordingly, a couple of years ago we began evaluating off-the-shelf inventory management systems for possible adoption.

What we needed

CIDR was in search of an inventory system that would allow us to search on a variety of fields, such as serial numbers, manufacturer or vendor name, location, service tag number, etc., and that could easily generate a variety of reports filtered by various criteria (category, vendor, manufacturer, budget number, and so forth). The stakeholders included administrators and budget specialists, who enter and maintain equipment records, including purchase account information; lab directors, to generate reports for strategic planning, proposal writing, and contractual reporting obligations; and lab managers, responsible for hands-on equipment handling, to record warranties and service contracts, upgrades, and part replacements.

The types of equipment CIDR owns vary widely. Some examples:

- DNA sequencers
- Centrifuges
- pipetting robots
- powerful computer or network equipment
- components or replacement parts for any of the above, which needs to be tracked together with the enclosing equipment
- information about warranties and service plans (for which we need the system to generate timely alerts about expiration and renewal dates)

What we evaluated

Of several inventory software systems that we reviewed, most were geared towards very specific classes of equipment, most often networked computer equipment. Some of those we considered:

- Tracmor
    - Commercial (though touted as open source)
    - $99/month for 2500 assets
    - does not provide for custom fields
- GLPI:
    - Open source
    - LAMP-based
    - had best reviews and deserved the most attention
    - somewhat network-device oriented

- ezofficeinventory.com
    - Cloud-based
    - Good list of features
    - Positive reviews
    - Subscriptions start at $30/month
- Spiceworks
    - good reviews
    - focused on network scanning and IT device management

In the end, we selected WASP MobileAsset, a commercial Windows application popular in large, established logistics/merchandising companies, with a price of about $3000. With the MS SQL Server back end, reporting is easy to set up and configure; SQL Server Reporting Services (similar to Crystal Reports) is included in the standard SQL Server installation.

Implementation issues

We encountered interesting challenges during implementation, configuration, and the import of data from long-used spreadsheets. One of these was the non-homogeneous nature of records in the source spreadsheets. Two or more instances of the same equipment (e.g., a network switch, lab freezer, centrifuge) were recorded inconsistently under different ad-hoc categories (e.g., IT, Lab, Admin) that were created by different people who did equipment data entry over the years. There were other variations in initial data entry; for example, in the source spreadsheets, "type" and "category" of the same kind of equipment could often be different. A laptop could be recorded as either "laptop" or "computer;" a part for a blade server could go in as "network equipment" or "server part;" a workstation that came with an Illumina microarray scanner could end up in any of the categories "lab – major," "lab – minor," or "workstation."

Much work was involved in fitting the data model embedded in the MobileAsset schema: asset vs. asset type. By design, each asset in MobileAsset inventory application is assigned to an “asset type” – a category or classification for assets. For instance, you may have an asset type of “IBM Laptop X6000” whose attributes include the manufacturer, size specs, etc., of this particular type of computer. Particular computers of the same asset type, each with its own serial number and perhaps slightly different memory and disk size, are assigned as assets to, say, the “IBM Laptop X6000” asset type. They are then individually accounted for in budgets, service contracts, maintenance, room or person assignments, etc. In this case the asset type is still very specific, because it includes the model number, so a different IBM laptop (e.g., X6500) would require creation of a new asset type. Alternatively, the asset type can be more generic, e.g., Agilent Bioanalyzer, Fisher Bath, Illumina Heat Block, Dell Blade Server or HP Computer Workstation (as distinguished from Dell Laptop or HP Laptop). In our designation of asset types we leaned to the latter kind, yet still tried to strike some balance between being too specific and overly generic, by including the manufacturer in the asset type. Thus, we have asset types like "Dell Computer," "Dell Server," "Dell Computer Monitor," "HP Computer" and "Apple Laptop." After a few months of using the system, this seems to have been the optimal decision, as it makes searching and filtering within MobileAsset fairly intuitive.

All asset-type assignment to individual pieces of equipment was done in spreadsheets (modified copies of the source inventory spreadsheet) before data importation, with manual assignments for more generic types (e.g., Heat Block, Hybridization Oven, or Network Equipment) and some use of Excel vlookup() functions and less-than-straightforward heuristics for less generic types (e.g., Hoefer Fluorometer, Beckman Pipette Tool, or Perkin Elmer Robot).

Clean-up and importation of source data

There were several rounds of source data clean-up and importation through the MobileAsset data import API, resolving a certain category of errors with each iteration. A lot of minute detail, snags, and on-the-fly resolutions and fixes could be mentioned here, but these are typical issues of any data migration project, in which data is moved from one system of record to another.

Ongoing issues

CIDR's MobileAsset installation has been in use for several months now. As with any software application, issues arise periodically. There are various bugs and oddities in the software. For instance, on the Asset Details window, the last of nine tabs has all custom-created date, text, and number fields. These custom fields were essential to getting all our source equipment information into the MobileAsset system, since the standard pre-defined fields are insufficient to capture all detail (split budget numbers, NIH and Johns Hopkins tags, CIDR's own tags, internal order numbers, etc.). The many custom fields do not fit neatly into this last tab; the window cuts off at the bottom, and a couple of fields just don't make in in. Moreover, the window cannot be resized. Basically, then, two of our custom fields are unusable. Upon filing a ticket with WASP Support, we learned that the only way to the circumvent window size limitation is to change the Windows system font setting from "Medium – 125%" to the default one, "Smaller – 100%," which is a work-around rather than a solution.

Another disadvantage is that MobileAsset is not a true multi-window application: it is not possible to have two assets open for editing simultaneously, or to have an asset and an asset-type form open at the same time; one window must be closed before the other can be opened.

This kind of limitation is, unfortunately, common in the world of off-the-shelf applications. Fortunately, these and other shortcomings were not show-stoppers in our case. We were able to agree on using custom fields that are properly displayed by the application and to find other workarounds. The shortcomings of the application are further mitigated by the fact that full access to the database backend makes it easy to fix data entry mistakes and to generate reports and alerts. We remain hopeful for improvements in future versions of MobileAsset by its vendor, WASP.