A Unified CAD-PLM Architecture for Improving Electronics Design Productivity through Automation, Collaboration, and Cloud Computing

Jonathan Friedman, Newton Truong, and Mani B. Srivastava
Networked and Embedded Systems Laboratory
University of California, Los Angeles
{jf8, dustintorres, mbs}@ucla.edu

***Abstract*—In electronics design, Computer Aided Design (CAD) tools manage part data in a logical schematic view (a part symbol) and a physical PCB view (a part footprint). Yet, a part has a third view, which CAD tools ignore – its supply data (Manufacturer part number, variant, distributor, etc). To manage this manufacturing view a broad class of tools known as Product Lifecycle Management (PLM) have evolved.**

A substantial chasm exists between the manufacturing and engineering views. More specifically, part data known to the supply chain (managed through PLM tools) and performance and specification data known to the engineering world (managed through CAD tools) must be manually integrated and managed by the design team. This leads to a substantial amount of redundant data entry into both tool chains with any error resulting in an inconsistency between design intent and fabrication.

In this work we introduce an entirely new approach to bridging the tool-chain divide a web-based architecture we call FriedParts. FriedParts exploits the recently available database-driven parametric part interfaces of CAD tools (like Cadence's Component Information System or Altium's Database Library Components) and web 2.0 automation to crawl data information providers like Octopart, Inc. and Digikey, Inc. and tie this information directly into the CAD tool at design time. It uses heuristics to suggest CAD symbols and footprints. Part search is handled from the website where cloud computing accelerates the search performance. The materials bill output from the CAD tool is then fed back into FriedParts which can automatically find second-source distribution, find alternate manufacturers, optimize purchasing, and perform other PLM functions. The amount of data entry by the designer is brought to almost zero. FriedParts stores the actual CAD data (part libraries) fostering verification and collaboration.

In a user case study, the average time to enter Digikey part number P1.0KGCT-ND, a Panasonic 1K-Ohm surface-mount resistor, was 6.1 seconds – including all of the round-trip time to the server. Compare this with more than ten minutes to do the identical task using a combination of an Excel-based CIS database and a popular online-based commercial PLM product. Further, the FriedParts solution resulted in zero data-entry errors and perfect user compliance, whereas three errors (inconsistencies) were found between the data entered into Excel and the online PLM.

Conceptually, FriedParts is a technology demonstration of the idea that CAD and PLM should share a single parts database to eliminate synchronization effort and errors, should exploit online information sources, and should simplify and automate data-entry tasks. FriedParts is open-source and will also be made available as a free service.

I. INTRODUCTION

Computer Aided Design (CAD) tools have all but eliminated manual drafting and for good reason. CAD offers more accurate drawing tools, efficient reuse of designs and drafting symbols, rapid duplication of finished designs, easier integration with manufacturing partners, and automated design rule verification. But along with their benefits, digitization of design information has usurped the semaphoric value of a hard-copy design document that engineers physically sign and hand to one another. In the pre-digital design age, little effort was required to keep a product's design details in order as multiple copies and variants of design documents were expensive and difficult to produce. In the digital age, the extreme opposite is true – copies and variants are easy to produce, while document control and consistency is extremely difficult.

In an effort to rein in design variant proliferation in a digital age, software has been employed, but much like the problem it attempts to solve, current design flows suffer from too many tools – and the resulting manual interfacing they entail. For the purposes of this work, we will focus on electronics design (and specifically the design and manufacture of Printed Circuit Board-based products), although much of the discussion forthcoming is broadly applicable.

II. ELECTRONICS DESIGN

The concept of a "part" in a design is actually a meta-construct consisting of a plethora of parameters and properties. It is useful to group these by job function into three broad "views" of the part. The engineer views the part as a logical entity which performs a specific electrical or mechanical function. These logical views are assembled into a schematic diagram which defines the functional behavior of the design. The schematic conveys the design intent to the Printed Circuit Board (PCB) designer who works with a physical view of the part, which includes the PCB footprint and/or 3D mechanical models (MCAD) to ensure package fitment and thermal and electro-magnetic compliance.

The designer's output is sufficient to manufacture the PCB, but insufficient to build the complete product. To populate the PCB with actual parts, they must be purchased, checked, shipped, and assembled. Numerous purchasing, accounting, manufacturing, and QA personnel are involved in this process, but they all interact with a view of the part we call the procurement view. The procurement view of a part consists of its manufacturer provided data (product name, part number, datasheet, etc), distribution and procurement data (distributor part numbers, prices, availability, etc), and quality control data (legal compliances, obsolescence plan, review and approval status, usage history, etc). The three views of the part are indicated in Figure 1.

Electronic CAD tools in this space manage part data in a logical schematic view (a part symbol) and a physical PCB view (a part footprint). Yet, some cursory support notwithstanding, CAD tools ignore the procurement view. To manage this procurement view a broad class of tools known as Product Lifecycle Management (PLM) have evolved to include most of the functions previously classified as Engineering Resource Planning (ERP). As AMR Research analyst Michael Burkett describes it, "PLM is [no longer] a single application but a process that crosses multiple business processes and technologies, i.e. marketing and supply chain" [1].

III. REIMAGINING THE CAD-TO-PLM BOUNDARY

Despite these advances, a substantial chasm still exists between the manufacturing and engineering views (Figure 1). More specifically,

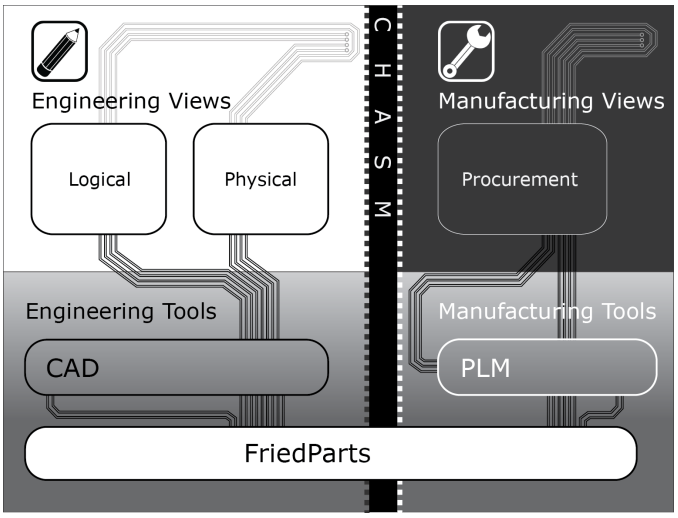

Fig. 1. Electronic CAD tools manage part data in a logical schematic view (a part symbol) and a physical PCB view (a part footprint). CAD tools largely ignore the procurement view. To manage this procurement view a broad class of tools known as Product Lifecycle Management (PLM) have evolved.

part data known to the supply chain (managed through PLM tools) and performance and specification data known to the engineering world (managed through CAD tools) must be manually integrated and managed by the design team. This leads to a substantial amount of redundant data entry into both tool chains with any error resulting in an inconsistency between design intent and fabrication. In the case of electronic small parts, this can result in subtle (even unnoticeable) part substitutions, which may yield dramatic product performance alterations.

In theory, Product Lifecycle Management (PLM) software allows companies to manage the entire lifecycle of a product efficiently and cost-effectively, from ideation, design and manufacture, through service and disposal [2]. In practice, however, those tasks are divided among computer-aided design (CAD), computer-aided manufacturing (CAM), computer-aided engineering (CAE), product data management (PDM), purchasing and accounting software (ERP), and, possibly, digital manufacturing tools. As a result, data must be collected from these tools and provided to the PLM tool. In effect, the PLM tool owns the procurement view, while CAD tools own the logical and physical ones. Until the PLM tool is the master of all three views, it will be unable to efficiently deliver the full promise of PLM. Only by reorganizing the CAD-PLM boundary will the chasm be eliminated.

IV. TARGET AUDIENCE

In enterprise scale operations, the chasm is bridged by manual labor. A parts librarian is assigned the duty of managing the CAD and PLM libraries and ensuring compliance and approval, hierarchy in the engineering department allows review of CAD data to ensure correct export versioning, product managers ensure BOM revisions are correct in the PLM, assembly technicians verify that the PLM BOM and the CAD BOM match prior to machine assembly of the PCB, purchasing agents handle contract negotiation with suppliers, and QA personnel provide redundant checking. This works well for companies with product sales volume sufficiently high to justify the number of personnel involved, but for more moderate operations this overhead is unaffordable.

For expediency, we define the term Small-to-Medium sized Design Bureaus (SMDB) to refer to this class of product building entities where enterprise-grade PLM tools are ill-applicable. It includes independent product design houses which do contract design and contact (or sub-contracted) manufacturing (Original-Design Manufacturers – ODM's), startup companies with electronic-centric product lines, and any small-to-medium sized companies where their product sales structure is built around, "low volume, high change" (catalog diversity). The market for SMDB services was valued at $100.65 billion in 2006 and is expected to grow to more than $219 billion by 2013 according to research firm Frost & Sullivan [3].

For this class, unless they can bridge the Chasm, common PLM solutions are essentially too big to be efficiently employed by SMDB's. SMDB's lack either the personnel to handle and maintain all of the data entry necessary to key in parts to the PLM and CAD tools or the will to do so, given the large number of designs from the wide array of clients they handle. The growth in complexity and scale of modern embedded electronics (and available reference designs) means even small design shops need enterprise-grade data management [4]. With the number of product lines growing and the life cycle of any given model shortening, even large enterprises realize that they must either increase staff, behave more like an SMDB, or turn further to outside help. In 2003, iSuppli found that more than 75% of such Original Equipment Manufacturers (OEM) expected to increase their use of SMDB's over the next three years [5].

A typical data entry procedure consists of entering the part number, manufacturer, manufacturer's part number, schematic symbol, and PCB footprint (at a minimum) into the CAD parts database. The manufacturer and part number must be re-entered into the PLM tool as well along with sourcing, pricing, and inventory information. The output from the CAD tool, the list of parts used in the design (e.g. the Bill Of Materials – BOM), must then be imported into the PLM and matched up to all the PLM parts. All of the data entry into the CAD tool and the PLM tool is human labor. Any mismatch, at any point in the information-chain, could result in manufacturing delays or worse – scrap.

V. FRIEDPARTS

In this work we introduce an entirely new approach to bridging the tool-chain chasm – a web-based architecture we call FriedParts (Figure 2). FriedParts exploits the recently available database-driven parametric part interfaces of CAD tools (like Cadence's Component Information System [6] or Altium's Database Library Components [7]) and Web 2.0 [8] automation to crawl data information providers (ex. Octopart, Inc. and Digikey, Inc.) and tie this information directly into the CAD tool at design time. It uses heuristics to suggest CAD symbols and footprints. Part search is handled from the website where cloud computing accelerates the search performance. The materials bill output (BOM) from the CAD tool is then fed back into FriedParts. Because the BOM is already keyed to FriedParts' unique identifiers, FriedParts can perfectly and automatically import the CAD-BOM ensuring perfect CAD-to-PLM materials build out alignment. Further, FriedParts can automatically find second-source distribution, find alternate manufacturers, optimize purchasing, and perform other PLM functions. The amount of data entry by the designer is brought to almost zero.

FriedParts stores the actual CAD data (part libraries) and project BOM's fostering verification and collaboration by allowing designers to rapidly search among existing symbols and footprints and recognize parts used by specific peers in specific prior work. This works exceptionally well for connectors and interfacing components when building, for example, a daughter card to a master system another engineer has already encountered. Locating and reusing the appropriate connectors, templates, and harnesses is simple with FriedParts

and facilitates the efficient transfer of institutional knowledge. For example, one of the Light Emitting Diodes (LED) in the database is marked by a user comment, "Polarity marking on some actual parts is inconsistent with the datasheet. Avoid." Another transistor part is noted, "[manufacturer] representative informed me that this part is scheduled for end-of-life two years earlier than previously announced. Avoid." Further, such knowledge is easily shared across institutions (we currently have three laboratories and one corporation sharing design data on the demonstration FriedParts system). Nearly every PLM tool offers commenting and knowledge sharing, but FriedParts, by integrating the CAD library and library file management, can explicitly guarantee that the comments and approval, pertaining the applicability of a specific symbol (and footprint or other model) to a specific part, are contextually correct.

Let us return to our previous workflow example. With FriedParts, the engineer begins by entering a global search term into the FriedParts' "Add New Part" webpage. This might be a manufacturer part number, a distributor part number, a keyword, or arbitrary phrase. FriedParts simultaneously performs the requested search over its local database, at Octopart.com, and at Digikey.com. The results are organized, combined, and then presented in list form. The engineer need only select the desired part from the list and click the "Easy" button (Figure 3). FriedParts then auto-fills the part's parameter fields, classifies the part, and guesses at its symbol and footprint based on a simple extensible set of heuristics developed by the authors. The engineer need only review the data already entered and make any necessary changes.

FriedParts even takes steps to mitigate the effort required to effect these changes. For example, the most common manual data entry step is the input of the CAD footprint and symbol. FriedParts maintains the official copy of the CAD libraries and parses them to extract the names of the symbols and footprints they contain. The engineer need only select the correct one from the presented pull-down list, which may be filtered (by typing) to accelerate the task of locating it (Figure 4). In a user case study, the average time to enter Digikey part number P1.0KGCT-ND, a Panasonic 1K-Ohm surface-mount resistor, was 6.1 seconds – including all of the round-trip time to the server. Compare this with more than ten minutes to do the identical task using a combination of an Excel-based CIS database and a popular online-based commercial PLM product. Further, the FriedParts solution resulted in zero data-entry errors and perfect user compliance, whereas three errors (inconsistencies) were found between the data entered into Excel and the online PLM.

VI. DESIGN PRINCIPALS

The goal of this paper is not to disparage existing PLM or CAD tools, but rather to suggest a relatively simple extension to the existing CAD-PLM architecture based around the following design principals. In so doing, existing PLM tools may be crafted into an extremely efficient package for the SMDB segment and perhaps offer even more value to the enterprise world.

VI.A. MAKE PLM THE MASTER

The PLM tool must own all of the part data, not just the resource-driven procurement view of it. To effect this, CAD libraries must be stored within the PLM's purview and control revision access to them (Figure 5). This will require a deeper understanding of the CAD tools, by the PLM creator, than previously necessary. The CAD tools must be database driven and populate parts directly from the PLM's database, not a separate (usually local) database as is common practice (more standardization of database schema on the part of CAD tool vendors would simplify this task immensely). Part data would, consequently, be consolidated in one database under one tool. All data-entry begins and ends in the PLM. It is an intrinsically safe approach that eliminates consistency errors since there is only one database and one point of interaction.

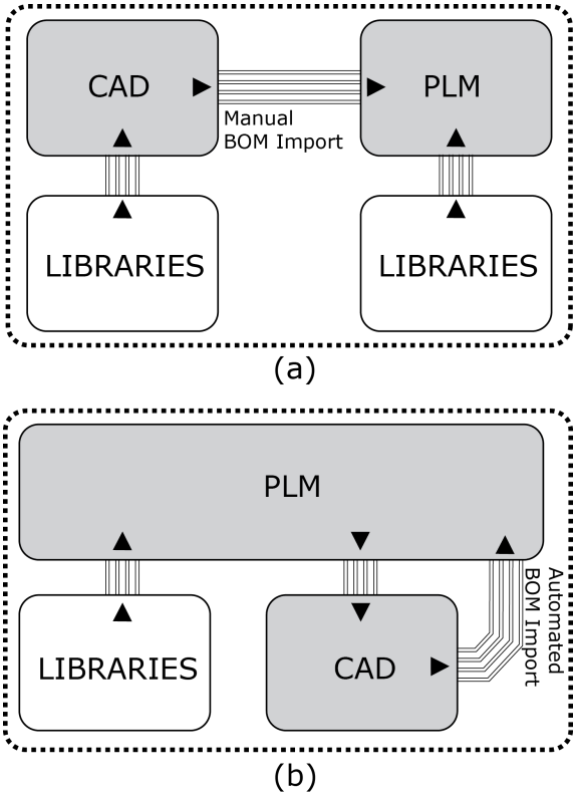

Fig. 5. (a) Current CAD tool to PLM tool interaction. Note the manual BOM alignment step during import/update to the PLM tool. (b) Proposed tool interaction approach – FriedParts. By eliminating the CAD tool's direct control of its libraries and bringing all user interaction to the PLM, CAD-to-PLM BOM interactions are automated since the CAD tool is already using PLM-aware parts.

VI.B. ELIMINATE MANUAL DATA ENTRY

SMDB engineers carry a design from conception to market. As competitive and economic pressures loom, even large enterprise engineering departments are fragmenting into ever smaller product development groups as the teams are asked to do more development with less resources. Our development group here at the Networked Embedded Systems Laboratory at UCLA consists of approximately ten engineers who work on, you guessed it, ten separate designs at a time. In this environment, keying in part data from a datasheet or manufacturer website is just an unacceptable waste of time and potential source of error. Computers excel at repetitive tasks, humans don't.

VI.C. LESS IS MORE

If intelligently selected, less features almost universally results in more perceived power. The success of the Apple iPodTM is a classic example. Restricting bad behavior and bad design practices not only results in improved quality, but improved efficiency in both the development and runtime performance of the PLM. A categorical example, with respect to CAD tools, is allowing multiple grid sizes, units, and off-grid electrical connections in schematics. Schematics are abstract graphical spaces whose exclusive purpose is to represent

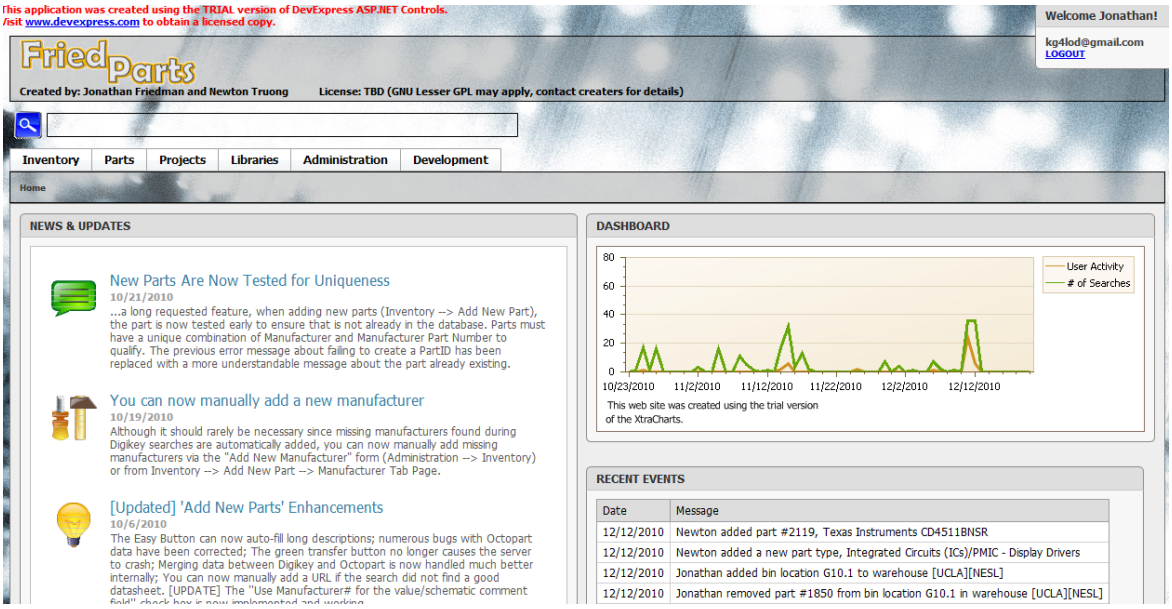

Fig. 2. The FriedParts homepage.

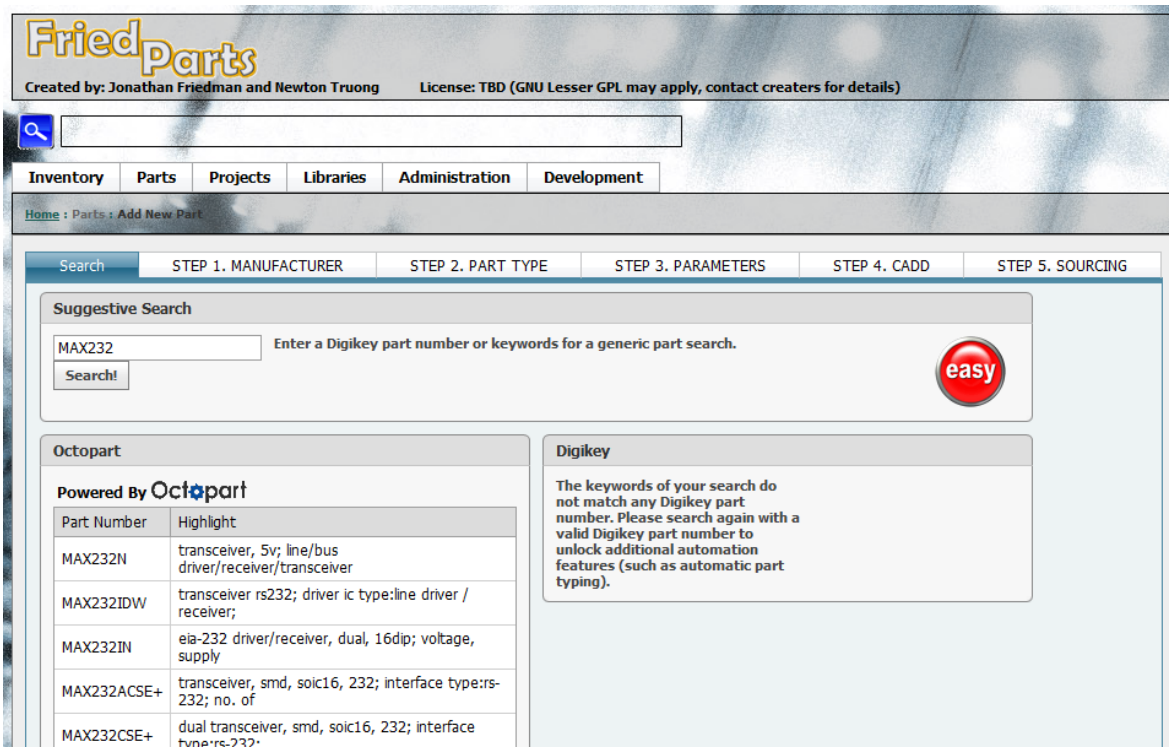

Fig. 3. The FriedParts' "Add a New Part" form exploits web-based component information databases to automate data entry.

electrical interconnection. There is literally no valid reason why off-grid points of electrical contact should be supported by any CAD tool – except to accelerate hair loss in design engineers. Commercial PLM tools come with powerful process flow editors, but the authors would prefer that these process editors be made obsolete by the inclusion of a single "best practices" design flow supported to the exclusion of all others. SMDB's are typically unaware of true best practices and cannot (or choose not to) afford to audit processes or hire efficiency experts to review internal procedure. Built-in, default, and enforced workflows allow SMDB's to benefit from the collected experience of quality assurance organizations without explicitly seeking their support and when improvements are made to workflow (and, correspondingly, to its implementation in the PLM tool) all of the customer organizations to that PLM tool immediately and automatically benefit.

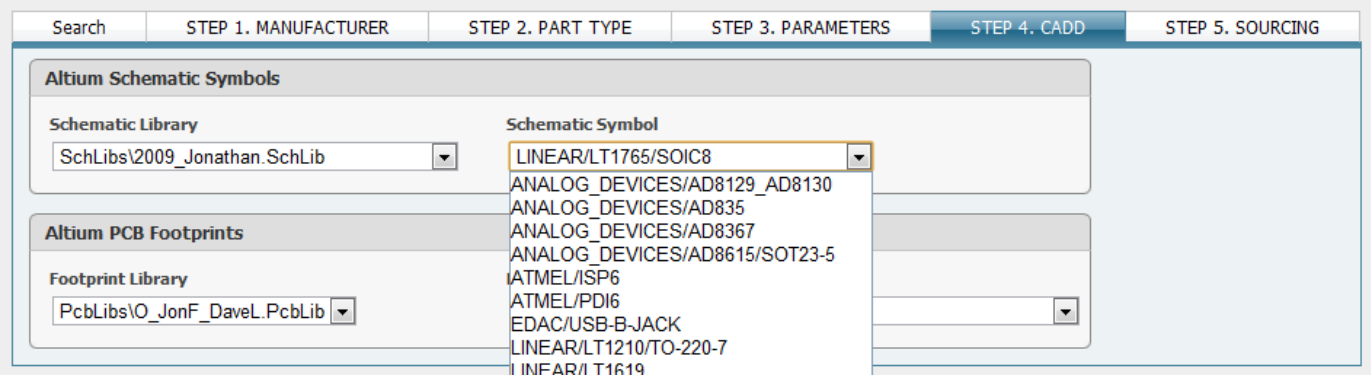

Fig. 4. FriedParts maintains the official copy of the CAD libraries and parses them to extract the names of the symbols and footprints they contain. The engineer need only select the correct one from the presented pull-down list.

VI.D. MAKE PROCESSES OBVIOUS

The power of PLM stems from its inherent ability to provide a real-time systemic view of product design and deployment. That requires that every person involved be comfortable employing the PLM. Immediate use by engineers with no PLM training requires obvious naming of forms, fields, and menus (avoid acronyms and jargon), inline help, tutorials linked from their respective data entry pages, and context aware notifications. We recommend that forms be grouped into a few broad categories that represent the stages of the design process to facilitate initial way-finding for first-time users. Individual data-entry forms should have their input fields grouped and ordered in such a way as to imply the correct procedure. In FriedParts forms, we use a tabbed approach with each tab corresponding to a ”step” in the process of, for example, entering a new part.

VI.E. DON'T MAKE USERS REMEMBER YET ANOTHER SET OF ACCESS CREDENTIALS

SMDB's do not employ personnel responsible for ensuring that all users follow all steps of every protocol. An SMDB PLM should take every step to make user compliance exceptionally easy and that begins with the front door. PLM's should not require complex create-a-new-user procedures, nor should they make the user create a new username or password. Instead, PLM's should support OpenID-based login through at least one of several popular federated login providers such as Google or Yahoo (FriedParts only currently supports Google federated logins). A new user to FriedParts can sign-up in less than twenty seconds if they have their Gmail open (already authenticated with Google). Further, no username or password is entered into FriedParts (this makes login so fast that it is unnoticeable) and therefore no risk of identity theft exists.

VI.F. USE SOCIAL REPUTATION RATHER THAN EXPLICIT SECURITY CONTROL

Managing access to the system is a complex and time consuming task that only makes sense in organizations large enough to need hierarchical access permissions due to a lack of trust among employees. With SMDB's, design teams are small enough that knowing who is responsible for a project or part is sufficient to result in appropriate behavior among all team members. CAD libraries are organized around individual users, rather than specific packaging technologies. We recognize that this may lead to some redundancy in stored footprints and schematic symbols, but better provides a reputation based framework for design integrity. When multiple part footprint options exist, you know exactly who created, maintains, or supports each alternative and can pick the one from the designer you believe most reliable. Further social features should be supported, such as commenting on contributed libraries and their contained symbols.

VI.G. ALLOW OUT-OF-ORDER DATA ENTRY

Data entry forms should be grouped into steps to facilitate compliance with procedure and automation (for example, it might be necessary to know the type of the part before we can automate selection of its footprint and value), however, processes should also allow out-of-order revision (random access). In FriedParts we use a tabbed approach where each step of the data entry process is a single tab. The user may either go through the tabs in order or jump around as appropriate. When validating the data entered, we present a list of errors to the user. When the user selects a specific error, the tab containing the corresponding field is brought to the front and highlighted.

VI.H. NO SETUP, RUN EVERYWHERE

PLM's must be globally accessible to maximize their utility. Several PLM tools have migrated online for this reason. While we agree with the general approach, we are concerned by several drawbacks which have emerged: (1) Some tools require extensive client security permissions, effectively turning a web-based tool into one which only runs on a computer that has been specifically configured for the task. (2) Some tools appear to be browser specific, (3) while others rely on uncommon plug-ins.

VI.I. PROVIDE ALTERNATIVES TO SUPPORT

To SMDB's the absolute cost of the system is more important than Return On Investment (ROI). Companies in this space cannot afford to think about engineering tool investments in many-year terms. They are concerned with immediate cash outlay and the cost of addiction to a service, whose pricing structure may increase at random with little warning and with little alternative. The cost structure for most commercial PLM tools is dwarfed by deployment and maintenance costs, an extreme example being Aras's Innovator PLM which is available as a free download – Aras making all of their revenue on support and training expenses. An effective SMDB PLM should be available both as a service and as one-time purchasable product which the buyer must host themselves. Altium licensing is a prime example of the success of this dual-natured approach [7].

VII. SYSTEM ARCHITECTURE

The FriedParts system architecture consists of three layers (Figure 6). User interaction takes place in either the FriedParts website or through their CAD tool, which communicates directly (read-only) with the FriedParts' database server. The website itself is spread across four services which may be hosted on one physical server or divided and duplicated as necessary to handle demand (the cloud computing model). The four services include a web server, which handles the browser interaction, a database server, which handles the actual data, a file server, which handles file uploads and downloads, and a periodic update service, which refreshes the database with new part status information it finds by continuously crawling Internet-based data providers. In our current implementation, the webserver is Microsoft Internet Information Server 6.0, the relational database runs atop Microsoft SQLServer 2005, the file server is Microsoft Windows Server 2003, and the update service runs as a local windows service on the file server. The database server is exposed through TCP port 1433 (the IANA standard [9]) to allow CAD tools to query the database in order to retrieve components directly through their own internal user interfaces. Data integrity requires that information retrieved from FriedParts (parts) is returned to it (BOM) in an orderly and controlled manner. To enforce this, data flow is uni-directional (read-only access over port 1433). By allowing the CAD tools to work directly with the FriedParts database, the unique part identifiers that the CAD tool will employ to recognize and track the part-to-originating-library relationship are known to, and actually assigned by, FriedParts (the FriedParts ID – FPID). If this were not the case, FriedParts might consider the common alternative identifiers used in CAD databases, which include manufacturer + manufacturer part number or an esoteric "Library ID" or "Database Part ID". The problem with these alternatives is that when the BOM moves out of the CAD tool and over to the PLM tool, the unique part identifier has lost its meaning. The PLM tool is unaware of the identifiers assigned by the CAD database tool and cannot automatically match them to existing parts in its own internal PLM parts database (Figure 5). Consequently, this step requires manual intervention. FriedParts assigns the unique part number (FPID) that is used to place parts in the design schematic. As such FriedParts will correctly and automatically recognize the BOM that comes out of this design. Most CAD tools that support database driven part placement can also report when the design contains parts that are not in the database. FriedParts will inform the user at the time of import if this error is not caught earlier.

FriedParts includes a rudimentary version control platform which exploits the available file server process. The file server stores the CAD libraries, project bill of materials, and other project engineering documents that are uploaded and retrieved through the website. The user is walked through the process of uploading files at the time of BOM import to ensure that a frozen state of the CAD engineering files is preserved. The user interface is designed to direct the user to upload only the appropriate files and the file type of the upload (file extension) is verified to encourage user compliance (Figure 7).

The update service, which exists as an independent server process, is periodically submitting queries to the data providers and updating records in the FriedParts database. It does so at the relatively slow rate of one part query every five seconds to reduce the traffic load imposed on the data providers to an imperceptible level. Digikey, in an apparent defensive measure against denial of service attacks [10], responds to search queries at a maximum rate of once per second per IP address. The update service, running at one fifth that speed, leaves sufficient space to allow interactive queries from FriedParts users to proceed unimpeded.

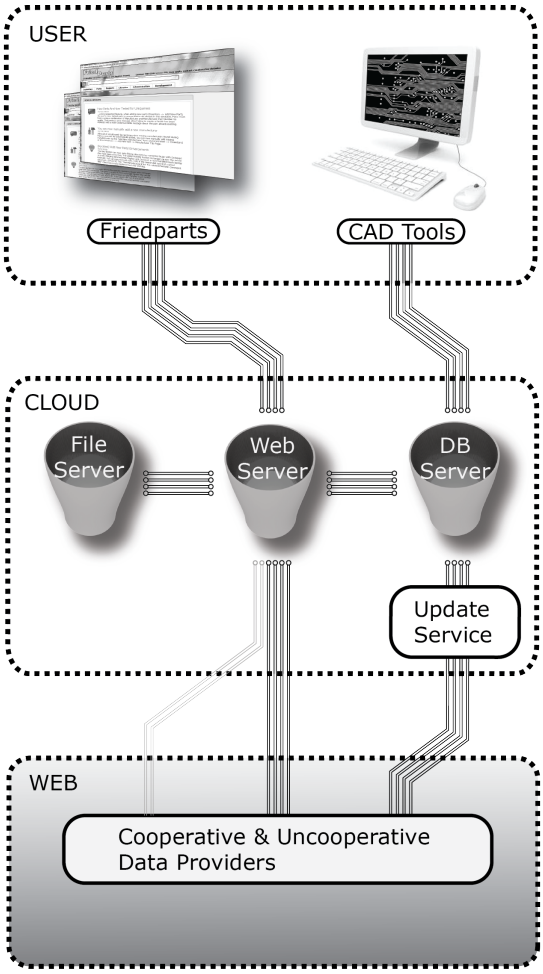

Fig. 6. The FriedParts' system architecture. The core infrastructure is divided into four cloud-hosted components which combine to run the website and manage its data.

When a part record is updated all of the projects in the system that use that part are flagged with any relevant notices. These include, price change, distributor out-of-stock, and compliance change (change in the description or parameters of the part). The collection and aggregation of market data by Octopart, and other data providers, allows the process of selecting components for purchase and cost optimization to be automated. FriedParts has sufficient information on alternate manufacturers and suppliers to automatically assemble purchase orders that minimize total cost while still providing complete BOM coverage. With the exception of shipping, where we just guess based on a stair-step model, FriedParts has all of the information it could need to calculate the lowest build cost. It attempts to save money by considering existing local inventory levels, unit quantities (distributors offer price breaks at discrete quantity levels – sometimes a complete reel is less expensive than a large number of parts), alternate distributors, and alternative parts.

While the authors could never reasonably argue that parts librarians and purchasing agents could ever be fully automated with no loss in job performance, FriedParts provides the SMDB a substantial improvement over the no-employee, no-time-for-optimization, alternative.

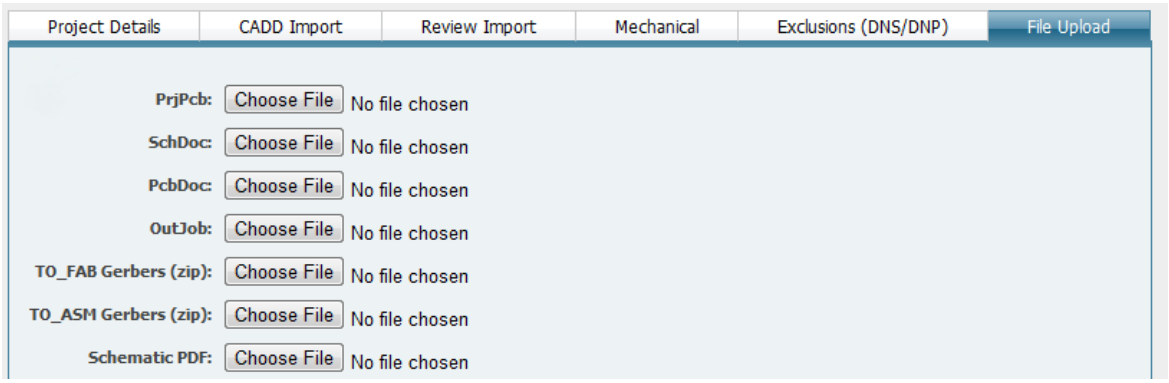

Fig. 7. FriedParts provides a limited version control system for a project's engineering files by requiring, checking, and storing files in the cloud during the BOM import process.

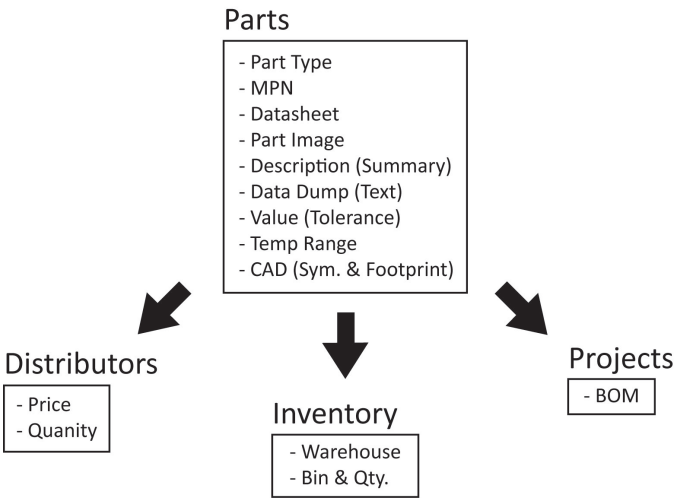

Fig. 8. A simplified view of the FriedParts data storage schema. Part parameters are reduced to a generic minimum. Further type-specific parameters are stored in a searchable text block rather than as explicit database fields to simplify the data entry automation.

VIII. DATA ARCHITECTURE

A plethora of schemes exist to store a manufactured part within the PLM. Each of the available PLM tools tracks a different set of parameters. The problem is complicated by the fact that what is interesting about a part, and therefore valuably stored as a parameter, depends on the type of the part itself. Commercial PLM's handle this task by providing a minimum set of attributes and allowing the user to specify additional part attributes that may be assigned to a part type or category. The authors take exception to this approach as it allows the user to, indirectly, change the database schema. A permutable data architecture substantially complicates the task of data entry automation. It is our implicit argument that the reduction in manual labor results in far more end-value to the SMDB engineer than being able to perform faster more precise parametric searches within the PLM tool. It is important to note that, at this stage in the product's development, the engineer has already chosen the part that he/she wants to use in the design. There is no need to store an entire datasheet's characteristics as parameter fields in the database. This type of extensive parametric search is reserved for the early design stages and can be done with currently available online tools from manufacturers and distributors.

The essence of how each part is represented in the database should

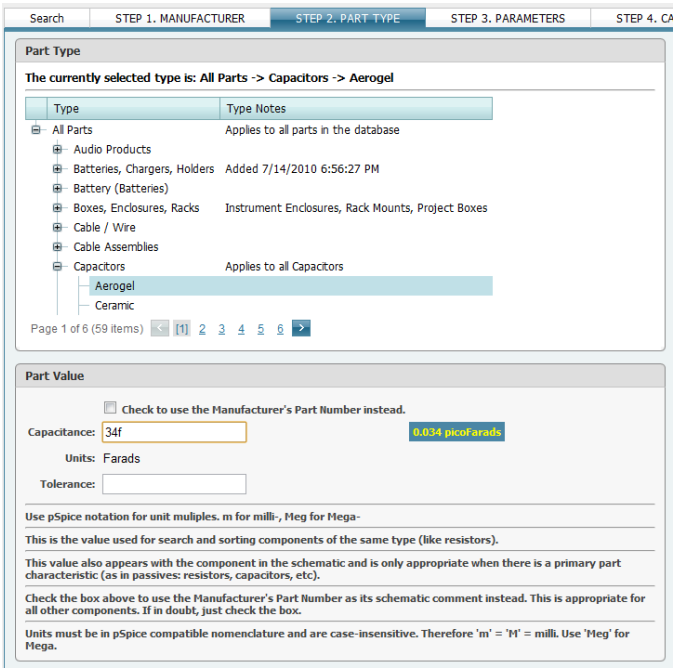

Fig. 9. Parts are assigned to a category (part type). The part type imparts meaning on the otherwise generic "Value" field. In this case, "Value" has been converted into "Capacitance". Having only one "Value" for a part type captures the benefits of type-specific parametric search without a user-permutable schema.

be reduced to the core essentials needed for unique identification. The parameters initially parsed out for a part (Figure 8) – such as Part Type, Datasheet, Part Image, Description, Value, and operating temperature range – are chosen because they represent the core describing attributes of a generic "part". Accordingly, this information is readily available on the Internet for any arbitrary part. Our web crawl program can take advantage of the spatial locality of the information, usually on a distributor's website, to mine all of the data an engineer will need to complete the part entry. Any additional data found will be stored as a text block to later support a limited parametric search. This is important if an engineer needs to check existing parts in the database as he or she can easily and immediately view these additional part parameters in the FriedParts website without having to download the datasheet.

The "Part Type" and "Value" fields (Figure 8) have some additional significance within FriedParts. The "Part Type" field is the category to

which this part belongs. Part types form a hierarchical (tree) structure and impart meaning on the otherwise generic ”Value” field. In the case of Figure 9, ”Value” has been converted into ”Capacitance”. Having only one ”Value” for a part type captures the benefits of type-specific parametric search without a user-permutable schema. Although not indicated in Figure 8, we track two attributes for each Part Type – the name of the Value for this part (ex. ”Capacitance”) and whether the value is numeric or textual. When a new part is being added, assigning the part’s type unlocks a substantial amount of additional automation. For example, once FriedParts knows the new part is a capacitor, it automatically recognizes common notations for capacitance values found in the part’s description, converts this into a number, and fills in the appropriate data entry form field. If the part were, instead, a logic microchip, such assumptions about the content of the part’s description no longer make sense. This value automation is implemented in an inheritable fashion. We assign the value name and determine value type (numeric/textual) by descending the tree from its selected leaf back to the root. We stop at the first part type along the way which has these name and type properties defined. In practice, this allows the value name ”Capacitance” to be assigned to the ”Capacitors” part type and automatically apply to all sub-types of capacitors (”Aerogel”, ”Ceramic”, etc). The root of the tree has the Value Name ”Value” and the Value Type ”textual”. If no further specification is found along the way, these values are used. We distill the remaining functions of a part into just three additional types of relationships. A part is supplied by one or more distributors (stock available for purchase), a part is stored in one or more local warehouses (parts-on-hand), and a part is used in one or more projects (BOM’s). Figure 8 captures this general simplicity in the FriedParts’ database schema.

IX. DATA PROVIDERS

The key principal in applying FriedParts methodology to SMDB targeted PLM tools is automation. FriedParts is implemented using ASP.NET with code-behind (business-logic) written in Visual Basic dot Net (VB.NET). VB.NET is an object oriented language that conveniently allows the authors to implement a generic data-provider automation class and then inherit from it to implement all of the code specific to that data provider. As a technology demonstration, we currently support two data providers: Octopart.com and Digikey.com. They are representative of a cooperative and uncooperative data provider respectively.

IX.A. Cooperative Providers (Octopart)

Octopart is a search engine that compares price and availability for electronic parts across multiple distributors. The data on the site comes from distributor feeds and is updated in real time [11]. The data on the site is made readily available for automated retrieval using an Application Programming Interface (API) exposed through JavaScript Object Notation (JSON). We call this type of data provider a cooperative provider because the information is easily and reliably obtained. JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is built on two structures: A collection of name/value pairs and an ordered list of values. All modern programming languages support these two data formats so JSON objects are language independent. For this reason JSON has become extremely popular for automating data exchange among websites on the Internet [12].

Datasheets	LNG992CFBW View all Specifications
Product Photos	LNG992CFBW
Catalog Drawings	LNG992CFBW
Standard Package	1
Category	Optoelectronics
Family	LEDs - <75mA, Discrete
Series	-
Color	Blue
Luminous Flux @ Current - Test	-
Current - Test	20mA

Fig. 10. A portion of a part webpage from Digikey.com. Note the row-associative nature of the table.

IX.B. Uncooperative Providers (Digikey)

Octopart has an authorized relationship with B&D, Bisco, Garrett, Gerber, Jameco, Masline, Newark InOne and Nu Horizons [13]. Data from these distributors is obtained directly from the companies through direct data feeds. Octopart does not have an authorized or official relationship with Allied Electronics, Digi-Key or Mouser. We call this second class of distributors, uncooperative data providers. Information from these providers must be captured directly from the markup code that creates the human-viewable webpage. In effect, FriedParts must navigate the distributor’s site and interpret the webpage as if it were a human visitor. Digikey, like many websites, uses the Hyper-Text Markup Language (HTML) to describe tables, where certain cells of those tables contain other tables. This table-within-table concept is known as nesting. The outer table layers are used to position and format graphical elements of the page in order to create the desired look of it, while tables deeper within the nested structure contain the actual data. Several permutations of the entire nested structure may appear in the page further complicating data identification.

There is no consistent way to predict the table structure or the depth of the nesting. FriedParts simplifies this problem by making several assumptions about the nature of distributor websites. First, the parametric part data that we are interested in is stored in a table (somewhere within one of the HTML ¡table¿ tags present in the page). Second, the data is row associative – meaning that every piece of information in the row is logically related to (grouped with) every other piece in some manner. Third, this concept of row-association also applies to nested tables (tables within tables). Intuitively (see Figure 10), this makes sense as the English language proceeds horizontally (left-to-right) before vertically.

To simplify the process of data extraction from the webpage and to make our parsing relatively immune to style or cosmetic changes instituted by the uncooperative data provider, we first refactor all of the webpage’s tables into a single table without any nesting. During this process links are treated as text. Our refactoring algorithm is illustrated in Figure 11.

This refactoring produces our text-data table. With the simplified table structure, it becomes trivial to locate, for example, the part’s ”Category”. Simply look for the row in the refactored table where the first column is ”Category” and report the value in the second column – in this case (Figure 10), ”Optoelectronics”.

A second table is constructed by scraping the page’s table content for links. The link-data table consists of three fields: (1) the text value found in the first cell of the row in which the link was found, (2) the link’s text, and (3) the web address to which the link points. The value

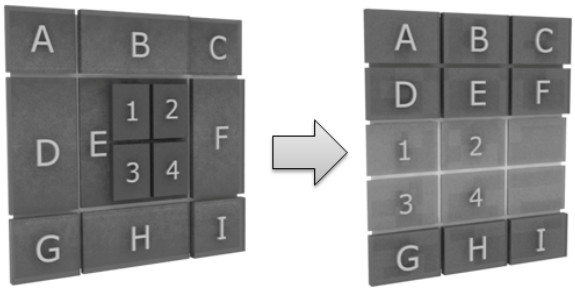

Fig. 11. A diagrammatic view of FriedParts' table refactoring.

of the first field in our link-data table is that it provides the appropriate context for the link. Referring to Figure 10, a link-data table entry for the first row might look like: Product Photos, LNG992CFBW, http://media.digikey.com/photos/Pa... Organizing link data in this manner makes locating links for the datasheet or a product photo trivial and largely independent of layout changes.

X. PRIOR ART AND RELATED WORK (THE PROXIMITY OF COMMERCIAL ALTERNATIVES)

Many excellent commercial PLM tools exist. It is not the aim of this work to disparage these products. Rather, the authors hope to illuminate a few areas in dire need of improvement in order to bring the enormous potential value that these tools represent to the small business (SMDB) marketplace.

Aras Innovator PLM [14] is an excellent, widely-used, open source tool that supports custom Dynamically Linked Libraries (DLL's) that may be developed by third parties [15]. It does not support direct interaction with CAD tools or their libraries. To remedy this, the authors had initially attempted to build FriedParts as an add-on module to Innovator but were ultimately unsuccessful in their attempt. External modules must fit within the existing Innovator User Interface (UI) framework preventing the necessary UI changes required to espouse our design principles. Innovator requires that each client workstation go through a security configuration. This restricts its use to pre-configured machines and it only supports the Microsoft Internet Explorer web browser. Arena Solutions, Inc. offers web-hosted on-demand BOM and change management software (e.g. an online PLM) [16]. Arena hosts the service on its own servers and, for performance reasons, manages all of their customer's data in a single database. This makes exposing direct access to the database dangerous to the service as a whole (it is not permitted). The absence of direct database access requires exporting and download of the PLM data to allow CAD access. This creates an independent CAD database (a snapshot of the PLM) that must be synchronized either manually or automatically (by purchasing and configuring an "Integration Automation" [17]). The very existence of the second database violates the core design principal of FriedParts and precludes an absolute trust in the integrity and timeliness of the data within the CAD database.

Numerous other PLM tools exist (Siemens' Teamcenter [18], PTC's PLM [19], Oracle's Agile [20], etc), yet each of these fails to comply with the FriedParts design principles in essentially the same way – they do not capture all three views of a part and thereby make the PLM tool the master of the data. What is needed is a turnkey solution with a single database mastered by the PLM tool and shared with the CAD tool. Enterprise PLM tools appear to place great focus on process accuracy (and for good reason – regulatory compliance, communication in large organizations, etc), while ignoring the larger concern to SMDB's – data accuracy. Ensuring data accuracy comes from minimizing the probability of data inaccuracy and that, in turn, comes from minimizing the laborious task of data entry.

XI. CONCLUSION

We are certainly not the first to recognize and struggle against the Chasm and both enterprise grade PLM tools and CAD tools are working to narrow the gap. The contribution of this work is to elucidate the ill-applicability of existing market solutions to SMDB's, to architect a broad solution, and to provide a working technology demonstration of the approach. FriedParts has supported more than thirty designs over two years in the Networked and Embedded Systems Laboratory (NESL) at the University of California, Los Angeles (UCLA), verifying the architecture's efficacy in optimizing SMDB product lifecycle management. At its core, FriedParts is built around three principles:

* MAKE PROCESSES OBVIOUS – Use a step-by-step approach to data entry form design. Provide only one way to do things.
* UNIFY CAD AND PLM PART DATABASES – Allow direct CAD access in real-time. Avoid copies and synchronization.
* AUTOMATE EVERYTHING – (to the maximum extent possible!)

XII. AVAILABILITY

In its current form, FriedParts is a turnkey solution for SMDB's using Altium Designer as their PCB CAD tool. FriedParts is open-source and will also be made available as a university-hosted free service on a limited basis. It is available at http://friedparts.nesl.ucla.edu and http://friedparts.com.

XIII. ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to Anyin Ye for her graphic design work on the figures, to Lila Ninh for her assistance in preparing the presentation slides, to Zainul Charbiwala for reviewing the manuscript, to Harish Agarwal at Octopart.com for his support, and to Andrew Arnott (DotNetOpenAuth), Karl Moore (HTML capture), Andy Powney (HTML parser for .NET), and the DevExpress Corporation (ASPxPerience) for their code contributions. The authors further thank Christine Collier and Jered Stoehr of Screaming Circuits (a division of the Milwaukee Electronics Companies), Nancy Viter of Sunstone Circuits, and Gil White and Mark Rodgers of DDi Corporation for their time, consultations, and industry perspectives.

This material is supported in part by the U.S. Office of Naval Research under MURI Award CR-19097-430345 and the UCLA Center for Embedded Networked Sensing. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the listed funding agencies.

REFERENCES

[1] Michael Hickins, When Titans Clash: PLM Versus ERP.: Internet-News.com, 2006.
[2] Siemens PLM. (2010, December) Defining PLM. [Online]. http://www.plm.automation.siemens.com/en_us/plm/definition/
[3] Suzanne Deffree, "Outsourcing by OEMs pushes EMS, ODM growth," Electronic Design News (EDN), September 13th 2007.
[4] Marty Hauff, "How to overcome the increasing management complexity of FPGA/PCB Pin synchronization," EE Times, p. 3, July 2008.
[5] Dennis Normile, "These slim margins are not by design," Electronic Design News (EDN), September 1st 2004.
[6] Cadence Corp. (2010) Cadence Capture CIS. [Online]. http://www.cadence.com/products/pcb/capture_cis/pages/default.aspx

[7] Altium, Inc. (2010, December) Altium Designer 10. [Online]. http://www.altium.com/
[8] Darcy DiNucci, "Fragmented Future," Print, vol. 53, no. 4, p. 32, 1999.
[9] Microsoft Corp. (2010, July) Article ID: 287932 - INF: TCP Ports Needed for Communication to SQL Server Through a Firewall. [Online]. $http://support.microsoft.com/kb/287932$
[10] Jon Erikson, "Denial of Service Attacks," in Hacking. The Art of Exploitation. San Francisco: No Starch Press, 2008, p. 251.
[11] Octopart, Inc. (2010) Octopart. [Online]. http://octopart.com/about
[12] (2010) JavaScript Object Notation Special Interest Group. [Online]. http://www.json.org/
[13] A. Meyer. (2008, November) Rhode Island School of Design risdpedia. [Online]. http://risdpedia.net/index.php/Octopart.com
[14] Aras Corporation. (2010) Aras PLM Software. [Online]. http://www.aras.com/
[15] Aras Corporation. (2010) Using Custom.dll's with Innovator. [Online]. http://www.aras.com/Community/wikis/kb/using-custom-dll-s-with-innovator.aspx
[16] Arena Solutions, Inc. (2010) Arena On-Demand BOM and Change Management. [Online]. http://arenasolutions.com/product/index.html
[17] Aras Corporation. (2010) Arena-Altium Designer Integration. [Online]. http://www.arenasolutions.com/images/pdf/rc_docs/datasheets/Arena_DS_Integration_Altium.pdf
[18] Siemens Corp. (2010) Teamcenter: Transforming the Process of Innovation. [Online]. http://www.plm.automation.siemens.com/en_us/products/teamcenter/index.shtml
[19] Parametric Technology Corporation (PTC). (2010) Windchill, PTC's PLM. [Online]. http://www.ptc.com/products/windchill/
[20] Oracle, Inc. (2010) Oracle and Agile. [Online]. http://www.oracle.com/agile/index.html
[21] Altium Corp. (2010) Altium Designer – Unified Data Model. [Online]. http://ad10.altium.com/#r10/explore/dm-UnifiedDataModel