PESC Members Approve XML Postsecondary Transcript

In an overwhelming show of support, 100 percent of votes cast by Members of the Postsecondary Electronic Standards Council (PESC) were in favor of accepting the XML Postsecondary Transcript as a PESC approved standard.

While current PESC policy requires 80 percent of votes cast be favorable for a candidate to be approved as a standard, the Members have clearly indicated their need for standards, their support for XML, and their confidence in the process for how standards are developed and approved within the higher education community.

Final versioning will be made on the XML Postsecondary Transcript schemas and accompanying Data Dictionary and final documentation will be available on the PESC website shortly.

This achievement marks a significant milestone for the higher education community and for PESC. In May of 2000, a small workgroup in PESC issued an educational white paper on a new technology called eXtensible Mark-Up Language (XML). The white paper is available at http://www.pesc.org/publications/white-papers.asp.

At that time, everyone was talking about XML, but not a lot was known about it. The workgroup concluded that there is strong promise for XML as it was being used already, but warned that a lack of standards for XML could lead to a multitude of differing proprietary implementations. To alleviate this, the workgroup recommended that PESC promote a single, unified approach to XML business standards.

The PESC Board of Directors responded to this challenge presented by the workgroup and formed the XML Forum for Education (now called the Standards Forum for Education). On August 4, 2000, the inaugural meeting of the Standards Forum took place at PESC’s headquarters at One Dupont Circle in Washington, D.C.

Since then, we, as a community, have worked hard to learn as much as we can about XML—we’ve educated each other on the competing needs of data...
across campus sectors and departments, and spent a lot of time identifying the strategic issues as much as the technical ones. The overarching goal was not to simply put out standards, but to move the community forward, together.

Much time was spent discussing business processes and the desired end result or goal of a business process. Before releasing any standard, two major foundational elements were needed: A data dictionary that standardizes data definitions across the admissions, registrars, and financial aid areas; and a model for data and architectural structuring.

Now with an all encompassing data dictionary and data/architectural model, we have laid the groundwork to develop other standards. These standards can pull a group of data elements from the dictionary to create a business transaction; or if new data elements are needed, they can be added to the dictionary.

Already in the works are development efforts by the National Student Clearinghouse to add supporting transactions for requesting and acknowledging the Transcript; and Mapping Your Future is looking to develop XML transactions to support the reporting on students who have completed counseling sessions.

To support standards, the US Department of Education’s Office of Federal Student Aid (FSA) has gifted to PESC and to the community, an XML Registry and Repository. This wonderful tool, that will store XML schemas for free downloading, will be available in June 2004.

A policies and procedures manual will also be released over the summer. This manual identifies all the requirements and processes that the Standards Forum and the community should follow in developing standards.

Lastly to ensure that developed standards are meeting the needs of the community, PESC is tracking the use and adoption of standards. For example, we’ve heard from the Chancellor’s Office of the California Community College System, whereby they have required that systems providers support this new PESC approved standard.

For anyone looking to communicate their specific use and adoption of the XML Postsecondary Transcript, please contact Michael Sessa or Ane Johnson directly at Sessa@PESC.org 202-293-7383 or Johnson@PESC.org 202-263-0296 respectively.

For more information on the XML Postsecondary Transcript, please visit www.PESC.org whereby a current version is available. Note that the final ratified version (the final version to use) will be available shortly as the PESC Board of Directors must officially accept the Members vote and final administrative versioning needs to occur.

Many, many people have helped bring PESC and the community to this point. Thank you to all of you as we would not have reached this milestone without the direct involvement of community leaders and participants.

Approved, From Page 1

FSA Prepares Five-Year Plan

Under the direction of COO Terri Shaw, the U. S. Department of Education’s Office of Federal Student Aid (FSA) recently circulated to a small group a draft five-year plan describing FSA’s strategic direction, objectives, goals, and success measures. The plan takes into account: PBO Legislation, Customer Needs, The President’s Management Agenda, GAO’s List of High-Risk Issues, and ED’s Strategic Plan. The final plan is being crafted now and should be available for the public in the near future.

Terri is serving as a keynote speaker at the 1st Annual Conference on Technology and Standards. She will discuss FSA’s Data Strategy, and take questions from the audience.
Workgroup Forms to
Develop Loan Counseling Schemas with MYF

BY CATHY MUELLER
EXECUTIVE DIRECTOR OF MYF

The process of exchanging entrance and exit data may be made easier in the future with the development of a standard format. To accomplish that standard, Mapping Your Future, Inc. is requesting assistance from any financial aid industry organization interested in developing an XML schema for entrance and exit counseling data.

The steps to developing an XML schema and next steps were discussed on a recent conference call, hosted by the Postsecondary Electronic Standards Council (PESC) to kickoff the effort. Mapping Your Future will be organizing this collaborative effort and encourages anyone interested in the schema to join the team.

If you are interested in participating in this effort or to obtain the meeting notes from the first conference call, contact Cathy Mueller, Mapping Your Future Executive Director, at Cathy.Mueller@Mapping-Your-Future.org or 940-497-0741 or Adele Marsh (AES), leader of the Mapping Your Future Technology Team at AMarsh@AESsuccess.org or 717-720-2711. The next conference call is scheduled for May 27 at Noon eastern; 11 a.m. Central; 10 a.m. Mountain; 9 a.m. Pacific.

Mapping Your Future is a national collaborative, public-service project of the financial aid industry-bringing together the expertise of the industry to provide free college, career, financial aid, and financial literacy services for schools, students, and families.

PESC On The Road

PESC continues to ensure awareness and communications through attending community meetings and events.

In March 2004, PESC Executive Director Michael Sessa attended the first Technology Summit for No Child Left Behind, sponsored by the US Department of Education, held in St. Louis.

Ane Johnson, PESC’s Membership Coordinator, attended both Datatel’s User Group Conference and SCT’s Annual Summit, held in March in Washington DC and Philadelphia, respectively.

PESC was also present at FSA’s Annual Spring Conference held in March in New York City. PESC held its Spring Workgroup Summit in New York City as well, in partnership with FSA (notes from the Summit begin on page 12 of this newsletter.)

In April, Michael also attended the Electronic Authentication Partnership meeting sponsored by the Center for Strategic and International Studies held in Washington DC; and the AACRAO Annual Conference held in Las Vegas.
Interview with Tom Reynolds

Vice President, Software Development and Quality
Datatel, Inc.

Tom joined the Datatel team in 1980 as a Product Manager for Datatel’s flagship Colleague solution. He has since held several positions leading to his current position as Vice President of Software Development and Quality. Tom was instrumental in the development of the Madison Technology Center at Datatel’s headquarters in Fairfax, Va. Tom has also served as a member of IEEE, ASQC and TEC.

■ Please tell me a little about Datatel in terms of when it was founded, what its function is, the number of clients it serves and where its headquarters are located.

Datatel is an employee-owned company founded in 1968. We are the leading provider of fully-integrated, enterprise information management solutions for higher education. Our headquarters is in Fairfax, VA, with offices in San Francisco, CA, Dallas, TX, and Buffalo, NY. We serve more than 620 client institutions throughout North America. For the past 25 years, Datatel has been exclusively focused on meeting the needs of colleges and universities, helping them operate more efficiently so they may better serve their constituents.

■ How many trading partners does Datatel have and how does Datatel handle trading partner communications?

Datatel works formally with over 25 trading partners, as well as numerous industry organizations and associations, to provide complete, service-based, information management solutions for colleges and universities. Our Marketing organization is responsible for building long-term business partner relationships that add significant value for our clients and fully leverage the products and services of each organization. Our clients rely on us to optimize the relationships among our partners, so the combined solution is one that is both integrated and easy to manage.

We have ongoing communications with our partners to keep all parties apprised and involved with key business, marketing and technical issues. Our partners participate with us throughout the entire client life cycle, from
pre-sales, through implementation, and post-implementation support. They participate with us at trade shows and at national and regional user group meetings.

Datatel follows a unique deployment approach with our partners. The collaboration is based on true integration and real-time data exchange between products, using our EDX technology. We base our integration efforts on industry standards, many of which are supported by PESC. We provide single sign-on capabilities for ease of use, and we utilize LDAP for identification and access. The IMS standards are used for data exchange with our learning management system partners; XML is used for transactions with the Department of Education and the Department of Homeland Security in Financial Aid COD transactions and SEVIS regulatory data exchange. We also employ EDI standards for transcript exchange with multiple partners and solutions.

What are the biggest technological concerns we face in higher education? What is Datatel’s position in regards to this concern?

The most pressing concern for higher education in regard to IT is the complexity of the technology environment. IT’s role is extremely challenging – meeting the business demands of the institution while managing the requirements of an ever-changing regulatory environment. The college campus of today is a heterogeneous environment of software applications, technologies and vendors. It must be able to satisfy the demands of customers, whether they are students, faculty, staff, or alumni. Assembling the right mix of technologies, business solutions, product offerings, and vendors can make all the difference in institutional effectiveness.

Another key concern is ensuring the compatibility of IT offerings, such as tracking version levels of the various technologies and tools employed in those solutions. Solutions must leverage resources on all levels to simplify what is an increasingly complex collection of technology standards, tools, vendors, and infrastructure needed to deliver service to constituents at all times. They must also see their way clear to the next generation of technology, and know how they will get there without interrupting service along the way.

Datatel works with clients in every aspect of this complex environment. For nearly three decades, we have boasted the best client retention rates in our market, and over 99 percent of our clients are on the most current release of our software. That’s pretty remarkable, considering all the changes in the technology world, and the continued renewal, growth and enhancement of our product offerings. Clients don’t want to change software as they grow, or when technology changes. They have a lot invested in their business practices. So, even as they reinvent the way they do things, they don’t want to throw out everything and start over.

Our products and technical solutions are unique. All our clients use the same software – single source code – regardless of their hardware, database system, or the size of their institutions. We have leveraged and pioneered the deployment of application generators and the use of metadata, generating code and business rules that are independent of specific database technologies, yet communicate to those environments in their native interfaces. Clients can implement their unique processes using parameters as tailored business rules. These rules are compatible within the framework of the development tools. Regardless of how they were deployed, their particular business models continue to deliver services, even when the need to change technologies, architectures, or delivery methods emerges.

Datatel invests as much in how to get to the next generation of products and technology as we do in the products or technologies themselves. We partner with our clients in a complete range of strategic planning and execution tasks – from business advisory services based on proven Malcolm Baldrige methodologies, strategic planning services, IT project management services, and implementation services, as well as new product offerings.

How does Datatel address privacy and security?

Privacy and security are paramount concerns for our clients today. By their very nature, college and universi-
ty environments are distinguished by a diverse community of users with an abundance of time and curiosity about technology and the applications they use. Higher education institutions typically employ a common network infrastructure shared by students, faculty, and administrators, and, to some extent, the community at large.

Our clients use standard authentication systems such as LDAP and Kerberos, and we ensure encryption in all communication transactions between client and server connections. Datatel solutions employ certificates for trusted connections, and our applications and tools permit data to be secured at the row, column, or cell level based on roles. Datatel’s Professional Services group is often called upon to help advise and implement reasonable and effective security and privacy policies.

**How does Datatel support standards? How does Datatel support PESC?**

Datatel supports standards in a variety of ways. We are members of PESC and have had staff involvement in the technical groups and PESC’s annual conference committee. We use the standards in our solutions, and actively support them at the client level.

**Is interoperability achievable?**

From an application point of view, we still have a way to go to achieve interoperability. The standards give you a good base from which to start, though they need to be extended to achieve the level of interoperability our clients want. Web services technologies can help, as well as the use of middleware. We use EDX, our own electronic data exchange tool, to apply industry standards and to interoperate with our partners, using the same set of business rules regardless of entry point into the system. Clients are assured of data integrity, consistency and ease of use. So, it’s possible and being done, though it has limitations.

**Datatel’s two main competitors are generally considered to be SCT and Peoplesoft. SunGuard recently acquired SCT and there have been many reports about Oracle’s desire to obtain Peoplesoft. What do you attribute to the sudden interest of these larger companies in postsecondary-specific technology companies? Is this a sign of a changing landscape in regard to enterprise systems and software solutions for higher education?**

Acquisitions are changing the landscape a bit, though neither SunGard nor Oracle is entirely new to the higher education market. Through its purchase of Bi-Tech and BSR, SunGard has been offering financials software and fund-raising software to higher education for nearly a decade. They have openly stated that acquisition is their primary means of expansion into the higher education market.

Oracle has also been active in higher education for more than a decade. They first offered their own financials, and then human resource software, to large 4-year institutions. Following several initiatives, either alone or in partnership with financials clients in higher education, Oracle recently released a student information system targeted toward large research institutions. So, what we are seeing now, and industry analysts have affirmed, is a continuation of this expansion strategy. The challenge lies in consolidating competing corporate cultures, business models, and multiple product lines, without disenfranchising segments of multiple user communities.

**Datatel recently drafted a white paper on application generators. What are the key points about this white paper?**

Datatel has been innovative and unique in using advanced technologies to create sustainable, renewable and affordable solutions for colleges and universities. We have developed and deployed application development tools, code generators, and metadata repositories specifically focused on delivering solutions for the business of higher education. Using these tools, methods, and strategies has enabled us and our clients to overcome the technical barriers that have traditionally required complete replacement of hardware and software when...
radical technological change or reengineered business processes are brought into play. We have incorporated our understanding of the recurring design patterns evident in higher education processes, and developed the tools to capture and preserve these technical and business rules in a technology-independent way.

As architectures have evolved from the monolithic mainframe, to distributed processing, to the Web, the way software is conceived has changed significantly. By capturing the business rules of the institution in reusable form, we have been able to preserve the institutional investments in business processes, while changing the technology used underneath. Rather than rewriting applications for different hardware, operating systems, databases, or new types of user interface, we have reused the accumulated knowledge captured in metadata and business rules to regenerate applications for the target environment. We have been able to build on our understanding of generative software development methods that we then incorporate into new versions of our tools and methodologies.

At the same time, we have proven the viability of the techniques through our success with our clients. We believe these methods reduce the cost to our clients over the long term, help them enhance service to their constituents, take advantage of new developments in technology, and not have to start over as their businesses grow and change.

To what should we all be paying more attention?

We need to continue listening and working with our clients and the marketplace. From a management standpoint, the problems we solve in technology are similar to those of the past. So, too, are the issues of implementing change. We are creatures of habit. The students of today, and I mean students in the context of traditional and lifelong learners, and the community of higher education, have grown with different expectations and experiences than we did. So we need to continue to learn from their experiences and expectations, and use our experience expertise to deliver service in new, innovative and effective ways.

What can the community look forward to from Datatel in the future?

Datatel remains exclusively committed to the higher education marketplace and to our clients. We are releasing an exciting stream of new products and technology in the coming year. The next major release of our Colleague software introduces significant architectural advances to enable true database independence with native connectivity.

We will be delivering a new generation of Web capability, replacing our current model, as well as introducing numerous new Web solutions. Datatel’s Envision development tool will add a new forms generator, making Web solutions easier to develop and deploy. Our business intelligence tools enable clients to take advantage of native environments to create and deploy data marts and warehouses for their business needs. Our workflow engine and workflow processes are exciting tools to streamline business processes and lower cost on campus. We will also introduce a new generation of tools to enable the Systems Administrator to easily manage and deploy new and updated software across the institution.

Datatel will expand its involvement with PESC and the overall standards community. We’ve deployed our software incorporating XML and IMS standards, and we are greatly involved in architecting the financial aid community standards as part of NCHELP.

Datatel continues to be firmly entrenched in the higher education community. We take the entire higher education marketplace seriously, not just the technology it uses. We are closely involved with higher education associations, and we are dedicated to promoting the Malcolm Baldrige criteria for performance excellence in Education, collaborating with government agencies that greatly influence the business requirements for automation and accountability in our educational systems. We also continue to fund the Datatel Scholars Foundation to give back to the community and provide educational opportunities to students.
Minimum and Optimal PC Requirements
for the 2005-2006 Award Year (from FSA)

<table>
<thead>
<tr>
<th>Minimum Configuration</th>
<th>Optimum Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM or fully IBM-compatible PC</td>
<td>Intel/AMD compatible processor</td>
</tr>
<tr>
<td>1.2 GHz (processor speed)</td>
<td>Intel Pentium 4 Processor – 2.80 GHz/333 MHz</td>
</tr>
<tr>
<td>512 MB RAM</td>
<td>1 GB SDRAM</td>
</tr>
<tr>
<td>60 GB Hard Drive</td>
<td>80 GB Hard Drive</td>
</tr>
<tr>
<td>56 Kbps Modem (that meets or is upgradeable to V.90 standard)</td>
<td>DSL Line or Higher Speed</td>
</tr>
<tr>
<td>All Configurations</td>
<td></td>
</tr>
<tr>
<td>3.5”/1.44 MB Diskette Drive</td>
<td>3.5”/1.44 MB Diskette Drive</td>
</tr>
<tr>
<td>Monitor and video card capable of Super Video Graphics Adapter (SVGA) (800x600) resolution (small fonts only) or higher</td>
<td>Monitor and video card capable of Super Video Graphics Adapter (SVGA) (800x600) resolution (small fonts only) or higher</td>
</tr>
<tr>
<td>Windows compatible Keyboard with Microsoft compatible mouse</td>
<td>Windows compatible Keyboard with Microsoft compatible mouse</td>
</tr>
<tr>
<td>Laser printer capable of printing on standard paper (8.5”x11”)</td>
<td>Laser printer capable of printing on standard paper (8.5”x11”)</td>
</tr>
<tr>
<td>48x CD-ROM Drive or higher with sound board (*Recommended CD-RW drive)</td>
<td>48x CD-ROM Drive or higher with sound board (*Recommended CD-RW drive)</td>
</tr>
</tbody>
</table>

Note: The optimum configuration described above is recommended and is readily available at modest prices. However, if your school will not send or receive more than 4,000 records (ISIR or COD) in a batch, the minimum configuration will suffice.

FSA does support:
Windows 2000
Windows XP Professional

FSA will continue support until July 1, 2006
Windows 98
Windows 98 SE
Windows ME

FSA does not support:
Windows 95
Windows XP Home
OASIS hosted a Symposium on Reliable Infrastructures for XML, April 26-27, 2004, in New Orleans. The event, which was open to the public, offered a forum for the international community to exchange ideas and present results of standards work-in-progress. Attendee identified unaddressed topics in need of standards development and areas where coordination between efforts would promote interoperability. Other program highlights included a user panel on “Implementing Standards to Achieve Reliable Transacting,” which explored the importance of open standards to users and the ways industry groups are implementing standard-oriented architectures into their own work. Seventeen OASIS Technical Committees met following the Symposium to advance standard data exchange for e-business, Web services, security, and legal applications. First meetings of the new OASIS Electronic Business Service Oriented Architecture (ebSOA) Technical Committee, the OASIS Web Services Notification (WSN) Technical Committee, and the OASIS Web Services Resource Framework (WSRF) Technical Committee were held at the event. For more information visit http://www.oasis-open.org/events/symposium/

In an article titled, XML Security: Ensure Portable Trust with SAML, Manish Verma discusses the objectives, architecture and basic concepts behind single sign on technology. In addition the article defines a standard mechanism for representing the information that needs to be exchanged, and defines a standard for the exchange. To access the article, visit http://www-106.ibm.com/developerworks/xml
Government officials confirmed the suspicions of college officials that officers at ports of entry into the United States frequently do not consult SEVIS, when scrutinizing foreign students and scholars in the hope of weeding out potential terrorists, according to a Chronicle of Higher Education article. “Some college officials say they are exasperated. Their international offices have created new positions, worked thousands of hours, and spent millions of dollars to make sure they meet federal guidelines for Sevis,” according to the article. The Homeland Security Department has not yet reached its goal of giving the primary customs officers access to Sevis, and at this point there is no indication of how long that will take. To access the full article visit http://chronicle.com/daily/2004/04/2004041502n.htm.

While many college and universities have been allowing electronic signatures for the release of transcript and other personal student-data without reprimand from the Department of Education for some time now, the Department had issued rules requiring a handwritten signature from the student for such transfers of data. The Department officially registered a rule change in the Federal Register on April 21, that now allows for electronic signatures to be used in all instances where handwritten signatures were previously required, according to a Chronicle of Higher Education article. The rule will go into affect on May 21. The Department has indicated that a future notice will explain the types of electronic signatures that will be acceptable, until that time colleges are directed to follow e-sign guidelines already in place for federal student loan transactions, according to the article.

In an article based on his keynote speech delivered during the XML Europe 2004 conference in Amsterdam, Edd Dumbill discusses the current state of XML. The article follows XML from “Refinement of the XML Core” through standards development and applications to a final conclusion that “XMLs Great Strengths Endure.” To access the article visit http://www.xml.com/pub/a/2004/04/21/state.html.

According to a IBM developer-Works article, “The idea of binary XML has always hung around the margins of XML discourse.” The article acknowledges the concerns surrounding storage of XML and transmission capacity in message transport formats. The article addresses options for compressing XML and the possible formats it may take. To access the article in its entirety visit http://www-106.ibm.com/developerworks/xml/library/x-tip-
In an attempt to speed adoption of PKI technology, OASIS has launched the OASIS PKI Action Plan. The Plan builds on the results of a series of surveys conducted by the OASIS PKI Technical Committee with IT staff who have deployed or attempted to deploy it. The PKI Action Plan addresses some of the primary obstacles to widespread PKI adoption; these adoption barriers include: poor or missing support in software applications, high costs, poor understanding of PKI among senior managers and end users, interoperability problems and lack of focus on business needs. The action plan can be accessed at http://www.oasis-open.org/committees/pki/pkiactionplan.pdf.

Twenty-four colleges and one public-school district have formed a consortium to share software and course materials developed at member institutions, according to a Chronicle of Higher Education article. The non-profit group, called the CampusEAI Consortium, first project is focused on helping members install and expand campus portal software. The group’s first project is an effort to help members install or expand campus portal software. Member institutions have already set up a software library, which contains 60 applications that any members can use on their campus networks. In the next few months, the consortium hopes to start sharing course materials and other digital content, according to the article.

The Massachusetts Institute of Technology has released its long talked about course-management software that it will give away free to help institutions create and run distance-education programs. The software, called Caddie.net, is designed to help institutions build Web portals to run multiple facets of their online-education programs, including registration, course management, and online testing.

Sun Microsystems Inc. recently announced that it has entered into a “broad cooperation agreement” with Microsoft Corp. and settled all outstanding litigation, according to an InforWorld article. The agreement includes technical collaboration, giving access to each other’s server technology, as well as Sun’s licensing of Microsoft’s communications protocols and Microsoft support of some Sun products. For more information visit www.infoworld.com/article/04/04/02/HNsunmssettle_1.html.

In an interview with Tim Berners-Lee, founder of the World Wide Consortium, the Software Development Times discusses the future of W3C in setting standards for the Internet and Berners-Lee’s vision for the Web. To access the interview in its entirety, visit www.sdtimes.com/news/099/story3.htm.
Standards Forum - Ed Hauser, SCT

- New make up of Steering Committee
  - Bob King, Citibank
  - Adele Marsh, AES/PHEAA
  - Paul Hill, FSA/ED
  - Ed Hauser, SCT
  - Bruce Marton, University of Texas - Austin

- Change in Standards Forum structure
  - defining process for developing standards
  - creation of policies and procedures manual
  - FSA development of an XML registry and repository

- Completion of XML Postsecondary Transcript

- Focus on funding and support of the Forum and PESC
  - need to ascertain more technical resources
  - process of developing standards is cumbersome due to the dependence on volunteer support
  - focus on membership

Policies & Procedures Manual - Jason Elwood, Miami University-DARS

- Brief rundown of process as outlined so far
  - submission of potential standard
  - creation of workgroup
  - submit potential standard to review committee
  - public comment period (after review of work by review committee and necessary changes are completed by workgroup)
    - open to entire public for 30 days
    - public comments posted on website and published in The Standard with response from workgroup on comments
• add appendix to supporting documents of standard with comments and responses
  • voted on by membership (must achieve 80% consensus)
  • ratified by Board of Directors
  • published as standard
  • approved standards will be easily accessed through our website and completely open to the public for download

• Implementation of standards
  • participation by vendors will drive adoption of PESC-approved standards
  • SCT, by implementing XML transcript into banner, involves 800 schools in the use of the transcript thus driving adoption

• Other Issues
  • how will we handle skipping steps? Example: The Common Record schemas don't need more public comment.
  • creation of an intellectual property policy
  • collaboration principles
    • financial viability (membership only voting privileges) vs. open collaboration (adoption and involvement)
  • need for marketing tools for membership to use
    • i.e. copies of The Standard, a flyer, and/or PESC PowerPoint presentation
    • "knowing what is coming will generate interest" meaning, advertising upcoming endeavors (such as work on NSC and DARS schemas) will peak the interest of outside players

**XML Registry & Repository** - Holly Hyland, FSA/ED

• **Gifted to PESC and the community by FSA**

• **Available for use by May 31, 2004**

• **Registry & Repository development**
  • based upon ED's XML Framework
  • developed collaboratively
  • registry built to ebXML standards
  • custom made solution, but will move to off the shelf at a later date
  • web-based application
- Core components data dictionary has been added
- Will be available through PESC and FSA websites

**XML Standards and the Registry**
- All standards (after PESC approval) of higher education will be represented in the registry (not just financial aid and admissions/ registrar)
- Need for consistent use of standardized data
- Central access point for managing:
  - Core components
  - Sector libraries
  - XML message specifications
  - Enumerated lists
- Information accessible, but version controlled

**Standard Student Authentication** - Michael Sessa, PESC

**Overview of Student Authentication issue**
- No standard way of authenticating
- Privacy and policy issues (laws vary state to state)
- Want higher ed to agree to adhere to standard authentication principals
- Problems with liability and levels of assurance
- Desire to standardize the ED pin and use it to authenticate on other sites
- Who assigns id?
- Penn State expressed interest in the continued use of SSN
  - Conflicts with certain privacy laws
  - ED uses SSN to authenticate e-sign, but required an act of Congress

**Electronic Authentication Partnership**
- Began by the CSIS, NACHA, and the OMB
- PESC has become involved in order to represent the interests of education
- Goals of the EAP were to establish business processes within and across verticals for authentication
- As an organization that has been responsible for setting standards for ACH (Direct Deposit, online payments, etc), NACHA provides a good model from which Authentication standards can be developed

**FSA involvement**
- Paul Hill indicated that the staff at FSA have a learning curve to over
come (in terms of issues concerning authentication) and that after training and
a gathering of the stakeholders FSA will be able to discuss next steps.
- FSA want a certification practice statement. Identifying students is only
  the first step of the authentication problem

**Single Institutional Identifier** - Jerry Bracken, BYU

- **Single ID is unlikely** - should focus on identifying sets of policies

- **Foreign Institutions**
  - need a way of identifying foreign schools due to legislation regarding
    homeland security
  - AACRAO does not assign codes to international schools
  - foreign school names change often and there are transliteration issues to
    consider when assigning codes
  - there are federal agencies in each country that creates codes
    - Use these codes? May overlap.
    - Or use country codes as the prefix to any assigned code
      (ISO Alpha Code)

- **Who updates the codes?**
  - those who own the codes are not always the best choice

- **ED wants to reduce the number of identifiers to 3 from the current 16.**

- **Numbering Schemes**
  - must be aware of these schemes as they may be incompatible with the
    business logic behind the use of certain numbers

- **Limitation of the use of certain codes due to intellectual property issues.**

- **Postsecondary Institutional Code Crosswalk update**
  - **Who?**
  - **What should be included?**
  - **How often should it be updated?**
Using Application Generators to Preserve Business Knowledge

March 26, 2004

Stephen B. Ornburn
Pete Nalli
Tom Reynolds

Datatel, Inc.
4375 Fair Lakes Court
Fairfax, VA 22033
1.800.DATATEL
703.968.9000

www.datatel.com
Using Application Generators to Preserve Business Knowledge

Stephen B. Ornburn  
GBC-Group, Inc.  
sbo@gbc-group.com

Pete Nalli  
Datatel, Inc.  
vpn@datatel.com

Tom Reynolds  
Datatel, Inc.  
tar@datatel.com

March 26, 2004

Abstract

Application generation is a powerful method for developing software. Datatel has worked with generative programming methods for many years and has learned a number of pragmatic lessons in how to successfully use application generators to build and maintain large software products. Datatel has succeeded with generative programming, in large measure, because of how it has organized business specifications for use by generators. Business specifications consist of small-grained, architecture independent components. Application generators combine these small components and generate fragments of implementation code. Finally, these code fragments are embedded in code templates suited to the target architecture. Thus, business specifications provide a single source describing application function and behavior, independent of the target architecture or software technology.
Using application generators to preserve business knowledge

Stephen B. Ornburn
GBC-Group, Inc.

Pete Nalli & Tom Reynolds
Datatel, Inc.

March 26, 2004

Abstract
Application generation is a powerful method for developing software. Datatel has worked with generative programming methods for many years and has learned a number of pragmatic lessons in how to successfully use application generators to build and maintain large software products. Datatel has succeeded with generative programming, in large measure, because of how it has organized business specifications for use by generators. Business specifications consist of small-grained, architecture independent components. Application generators combine these small components and generate fragments of implementation code. Finally, these code fragments are embedded in code templates suited to the target architecture. Thus, business specifications provide a single source describing application function and behavior, independent of the target architecture or software technology.

1.0 Generating applications directly from business specifications
Knowledge is power! And, the knowledge needed to operate a modern institution of higher education is costly to develop and all too often poorly preserved. The analysis and development of business knowledge—the business rules, information models, collaboration patterns and workflows—is often an institution’s single largest IT investment. Unfortunately, in many cases, the only permanent documentation of these results is deeply embedded in a software system’s detailed design and source code where it can lay forgotten and neglected until changes are required or the software itself must be rewritten.

The task of modifying or rewriting software can be an expensive undertaking, requiring considerable effort and expense to rediscover and reanalyze business rules and other related knowledge. To avoid or postpone this rediscovery and reanalysis, IT departments often struggle to keep old systems in service, delaying investments in new software technologies and system architectures.

Postponing the use of new technologies and architectures to delay the expense of rediscovering and reanalyzing business knowledge is an unsatisfactory strategy. The widespread availability of computing power and internetworking, and the emergence of distributed systems have, over the last thirty years, driven a steady stream of innovations. Some of the most important innovations include distributed transaction controllers, messaging and remote procedure calls, threads, location transparency, object migration, and, most recently, web services. This stream of innovation will continue unabated as the
industry responds to new challenges such as building reliable, high-performance, high-availability systems that are neither owned nor operated by a single organization and provide any-time, any-where, any-device access by users. Other challenges likely to drive future innovations include the need for computer-mediated collaboration, database management of unstructured data, and the automation of commonplace objects. In this environment, postponing the use of new technologies and architectures effectively postpones significant value creation for an institution and its users.

So how is an institution’s IT department to take advantage of these new value-creation opportunities without the high cost of repeatedly rediscovering and reanalyzing the processes and rules by which the institution operates?

The answer is to capture business knowledge once, referencing that single, authoritative source of business knowledge whenever software technologies are changed and applications must be regenerated. Fortunately, there is a software development paradigm, generative programming, that meets this requirement. Generative programming allows developers to take advantage of new computing technologies without incurring the costs of rewriting or substantially modifying applications. In Datatel’s approach to generative programming, specifications representing business knowledge are captured and maintained separately from any specific application software or system architecture. Datatel has built and maintains a set of generators that process these business specifications, generating software that conforms to current architectural standards. When new computing technologies emerge and architectural standards change, the software architects make the applicable changes to the generators and are ready to produce the next generation of applications from the original body of business specifications.

Business knowledge controls an institution’s operations and reflects the institution’s uniqueness

Generators create application components which encapsulate business knowledge and run on target architectures

Target architectures change in response to the economics and innovation in the software industry

Figure 1: Generators build software from business specifications for multiple architectures

---

The relationships between business knowledge, generators, and architectures are illustrated in figure 1. This diagram depicts two institutions that have developed distinct operating models represented by the separate bodies of business knowledge. Assuming this business knowledge has been represented as specifications, the generators can be used to build the applications. Furthermore, the applications can be migrated from the current architecture to another by changing the generator. This provides the opportunity to create additional value for the institution by replacing an outmoded architecture with one leveraging desirable new technologies.

Datatel has developed and delivered a library of core specifications, tools for creating and maintaining business specifications, and standardized target architectures and generators. Datatel users all use the same notation and data structures to capture their business specifications. Changes in the target architecture, generators, core specifications, notations and tools are driven by the needs of the user community, as expressed in various forums and through work with leading innovators.

The use of business specifications, application generators, and standard architectures provides a means of overcoming one of the chief barriers to change in large IT systems—the mismatch between the architectures of the existing and desired systems.²

Section 2, which follows, illustrates several approaches to renovating software systems, including the use of application generators. Section 3 explains how the generative paradigm works, introducing several examples of how they have been used to transform business applications to a new architecture. Section 4 considers important principles for organizing business specifications and building application generators. Section 5 compares the generative approach described in this paper to other common approaches, including Model-driven architecture and .NET Enterprise Templates. Section 6 draws some conclusions about the promising future for application-generation technology.

2.0 Renovating software architectures

In the absence of application-generator technology, IT departments typically seek to postpone the time when they will need to re-implement software to accommodate a new architecture. Mostly they try to mix and match software components based on disparate technologies and design patterns, with several different types of architectural mismatches as likely outcomes. These architectural mismatches are difficult to overcome but not insurmountable. Unfortunately, their resolution can add significant complexity, time, cost, and technical risk to system development. Several examples illustrate these difficulties.

- **Information**—Differences in data architecture can impact business rules and information. For example differences in how and when business rules are applied

to compute values such as account balance can result in errors that can be difficult to diagnose.

- **Communication**—Differences in the communication architecture can also complicate systems integration. For example, with an event-based system built around a dispatch loop, considerable engineering can be required to interconnect it with other systems employing other patterns such as publish-subscribe or asynchronous messaging.

- **Orchestration**—Integrating software can sometimes require complex orchestration as existing methods and data structures from the various components are used together in new ways.

There are two long-standing approaches to addressing architectural mismatches. The ‘banyan tree’ approach is the world of hackers—architectural mismatches are bridged with one-off, custom solutions. At the other extreme is the ‘root-bound’ approach in which data architecture and communication and coordination mechanisms are over generalized to solve the current problem with the hope of avoiding future architectural mismatches. Both approaches are risky, and this risk has been the cause of major software-project failures.\(^3\) Successful software departments generally look for a middle ground between these two extremes.

The banyan-tree approach causes rapid deterioration in reliability and increases in maintenance costs. Each successive change adds complexity by compromising the original architecture. With increased complexity there is a greater chance of unintended side effects - reduced reliability, increased instability, poor performance and increased support costs. Typically, the developer limits the risks to existing functionality by making changes in cloned modules (the banyans). This duplication of code leads to a stovepipe architecture that exhibits poor performance, inconsistent behavior, many maintenance and support headaches, and results in an increasingly sluggish development and delivery pipeline.

\(^3\) D. Dikel, D. Kane, S. Ornburn, B. Loftus, and J. Wilson. “Applying software product-line architecture,” *IEEE Computer*, Aug. 1997: 49-55. Figure 2 is, with permission of the authors, from an expanded discussion of these principles in D. Dikel, D. Kane, and J. Wilson, *Software Architecture: Organizational Principles and Patterns*, (Prentice-Hall, 2001).
In the root-bound approach, designs are generalized to avoid the possibility of architectural mismatches in the future. Eventually, even these applications must be changed or integrated in unanticipated ways. In the meantime, the excessive generality contributes to complexity and poor performance. Furthermore, the first release of a generalized architecture often entails greater risks and takes longer to build and deploy.

Some software products swing between these two extremes as though on a pendulum, as illustrated in figure 2. In one commonly observed pattern, the product team reacts to the banyan-tree architecture and its ever-lengthening release cycle by taking time out from new features to simply clean up the existing product. Because the cleanup had been long postponed, the team over-engineers the architecture, resulting in a root-bound solution.

There are three common ways to achieve the middle ground, avoiding the extreme swings of the pendulum: on-going refactoring, middleware, and application generators.

**Refactoring**—The term refactoring is often used to describe the process of modifying source code, particularly where the changes affect its structure and design but not its external behavior. This “cleaning up” of the code is intended to reconcile the architectural mismatches, remove hacked solutions, and make the code more maintainable. When asked to add new behavior, the developer can begin by refactoring the software, preserving its original behavior but giving it a new structure. By separating the refactoring from the addition of new behavior, the developer can more easily separate issues associated with the new design pattern from those related to the new behavior. Refactoring can include the repartitioning of components as well as changes in the mechanisms used to communicate information and coordinate behavior. With the new structure in place, the developer can proceed to add new behaviors. Experience suggests that refactoring works best when changing the internals of individual program units. Refactoring becomes significantly more difficult when introducing a new pattern that involves many units in large programs.

**Middleware**—Middleware attempts to mask mismatches in data architecture and communication and coordination mechanisms by wrapping the existing code in additional software. For example, a system of adaptors and a message hub with message routing and transformation capabilities can address many mismatches in data and communication architectures. Additional features supporting workflow and distributed transactions can provide orchestration across components or systems. Other means of software integration available from middleware providers include web services, message hubs, portal-style user interfaces, and metadata interchange. Middleware vendors are now promoting their products’ orchestration capabilities, emphasizing how their products can be used to build composite applications from a collection of legacy systems.

---

Middleware works best when interconnecting large, heterogeneous sets of applications, but can be complex to implement and expensive to support. It also introduces new points of failure and additional processing overhead, potentially reducing overall reliability and performance.

The use of middleware can, in the short term, mask architectural mismatches but, in the end, the use of middleware only postpones the inevitable—replacing the applications because of obsolete architectures.

**Application generators**—None of the approaches to architectural renovation discussed to this point address the problem of preserving business knowledge as a valuable asset separate from the software in which it is used. Instead, they offer ways of postponing or limiting the scope of software rewrites. Only through the use of application generators is there a clean separation between managing business specifications, defining system architectures, and generating application components suited to the intended architecture. It is because of this separation that the use of application generators allows for the introduction of consistent architectures and accelerates the adoption of new technologies.

### 3.0 Application generators explained

A software development paradigm employing application generators divides the software development and maintenance problem into four activities:

1. Capture, analyze, and preserve business knowledge;
2. Define architectures and design patterns for new systems;
3. Build runtime environment; and
4. Construct and use the generators themselves.

Figure 3 illustrates the interplay of these four activities. The system architect defines the target architecture, the metadata for the business specifications, and the patterns for generating application components. The application components are derived from the business specifications but must be capable of executing in the runtime environment. Thus, the system architect must also define the patterns to be followed when synthesizing application components from various types of business specifications. Other developers build the generators, runtime components, and tools for managing the business specifications. The business analysts can then use these tools to capture business specifications and subsequently generate, deploy and execute application components.

Application components are never truly insulated from the architecture. The architecture includes runtime services responsible for managing application components. As a consequence, application components must conform to the architecture by following design patterns, including fragments of system code, and supporting required interfaces and attributes.

In Datatel’s tool set, the runtime environment is a Datatel-developed application server. As illustrated in figure 4, this application server is built by assembling a number of
components. The process of assembling the runtime environment is similar to snapping together LEGO-style components.\(^6\) When fully assembled, the runtime system provides the *containers* and *services* needed to manage the execution of the application components produced by the generators.

In many modern systems, e.g., Java and .NET, an application component runs in a container that will handle the component’s execution—the container starts and stops the component, manages resources, and processes requests for services. For the application component to interact with its container, the architecture may require that the component include some system code in addition to the business logic. Furthermore, the architecture may require that the business logic itself be organized according to certain patterns.

---

\(^6\)LEGO, the registered trademark of LEGO Company, comes from the Danish words "Leg Godt," which means "Play well." In Latin it means, "I put together." [http://www.lego.com/eng/info/default.asp?page=brand].
Generators ensure that the application components follow the right patterns and include the right system code. Most development tools for the Java and .NET environments are limited to ensuring the correct system code is included in the application component, leaving the task of adding business logic to the software developer. In contrast, a full-fledged generator will go further, adding to the application component business logic derived from previously collected business specifications.

The use of application generators protects developers from changes in system architecture. Whenever the component/container model or other part of the runtime system changes, the application components can be regenerated using updated generators. To illustrate this point, several examples, drawn from Datatel’s experience, are briefly introduced.

**Virtual Machine (VM) independence**—In recent years, the software industry has seen a steady stream of changes to component models, the containers responsible for managing components, and the virtual machines (VMs) in which containers and components execute. A generative approach can be used to change component models or move a component from one VM to another. For example, by means of generator changes, Datatel has demonstrated that application components can be run in the UniData, Java, and .NET VMs. Of course, changes to the application-server containers are required to support multiple VMs and to allow communication between virtual machines when the virtual machines employ different technologies.

**Database independence**—With the unpredictable popularity of various relational database systems, it is increasingly important that enterprise applications be database-independent, running equally well on Oracle, SQL Server, and IBM database systems. Reengineering an application that was originally designed for a particular database so that it is now database-independent can be extremely challenging. Database-specific code must be removed from application

---

7 Similarly, if the business specifications change, the existing generators can be rerun, updating all affected application components.
components and database-specific constructs must be removed from the business specifications. Both of these tasks can be performed using application generators. One approach is to construct a database access layer that provides the application with a consistent interface regardless of the underlying database. Then, with appropriate generator changes, the applications can be re-generated to use these new database access routines.

- **User-Interface (UI) independence**—In many enterprise applications, including Datatel’s, early user interfaces followed a simple architecture that did not separate presentation logic from business logic. In modern interfaces, presentation and business logic are separated into distinct components. With application generators, this separation can be achieved simply by changing the way business specifications are processed.

- **Direct-manipulation UI**—The transition from forms-based user interfaces to UIs supporting direct manipulation operations such as drag-and-drop and cut-and-paste requires a complete change in UI architecture. The most common architecture for direct-manipulation interfaces is commonly known as model-view-controller (MVC). In MVC, the screen displays one or more views of an underlying data model. Controllers track user gestures such as mouse movements and keyboard activity. In response to user gestures, controllers modify screen state indicating events such as change of focus. If indicated, controllers will also invoke operations on the data model. Changes to a data model will generally trigger changes in the associated views. Ordinarily, introducing MVC into a UI architecture is a fundamental change, requiring a rewrite of all UI components and screens. When a generative approach is used, developers can introduce MVC by adding services to the runtime environment and making generator changes. As a consequence of the generator changes, new UI components can be generated from the original business specifications.

- **Structured Workflows**—Many existing enterprise applications are designed to support a number of individual users, each interacting with his or her own set of forms. Newer applications also support multiple users collaborating in cross-functional workflows. Application generation has been used to simplify the introduction of multi-user workflows into existing enterprise applications. Each step in a workflow is a separate business transaction. By means of generator changes the business transactions are instrumented to interact with the workflow management system, which is responsible for handling uncommitted data and providing services such as routing and escalation.

These five examples illustrate how a generative approach to software development can insulate applications from changes in the technical architecture, thereby protecting the institution’s investment in its business specifications.
4.0 Creating, organizing, and using business specifications

Application generation has a long history. There have been many successes, but with every success comes new challenges. As progress is made, many ideas have been tried in various combinations—rules-based programming, algorithm synthesis, design reuse, automated component composition, and wide-spectrum, extensible, and problem-specific languages. Overall, work on application generation has produced a rich set of results and has significantly influenced the design of modern software development environments. Successful application generation depends in large measure on structuring the business specifications.

In Datatel’s experience, four important principles for organizing business specifications have emerged: (1) clear context for the application of business rules, (2) small-grained specification elements, (3) logical data independence, and (4) explicit declaration of business rules and other attributes. When these principles are followed, generators can more easily construct application components.

To build application components from these business specifications, application generators must perform three types of operation—compilation, transformation, and translation. Compilation weaves together the business specification, collecting and editing small-grained specifications into a single text specifying a particular component. Transformation rewrites the specification to improve it in some way such as simplifying data access, optimizing an algorithm, or reducing memory use. Translation is the final step in which the transformed specification is re-written in a conventional programming language.

Compilation may also occur as part of the translation phase, combining generated code with code templates and library routines. The library routines implement common operations and data structures, and code templates supply system code through which the component will interact with its runtime environment.

For application generators to achieve ‘industrial strength,’ it is necessary to occasionally extend the business specification language so solutions to newly important business problems can be better described. It must also be possible to modify generators to handle these new types of specification or generate to new target architectures. Application generators work best if they are designed to build a relatively specific type of software system. For example, Datatel has designed its application generators for building and maintaining a specific software product line for higher education. This focus simplifies the design of the generators, permitting very specific assumptions about patterns of use, interfaces, system architecture, and runtime environments.

Of these steps—compilation, transformation, and translation—the most important for large-scale application generation is the compilation of small-grained specifications into a business specification describing a complete application component. For this

---

1 Early programming systems, such as FORTRAN, also used compilation, transformation, and translation operations. FORTRAN would translate mathematical formula into an assembly language, which it then combined with library routines implementing common operations such as for floating-point arithmetic. Transformations were also sometimes used to improve the code, e.g., factoring out common sub-expressions or optimizing the use of CPU registers.
compilation to succeed on large problems, the business specifications must be organized using the four principles mentioned at the beginning of this section.

The first principle for organizing business specifications is to provide clear context for application of business rules. This context helps structure the compilation process, so that several standard types of component can be generated from business specifications, including

- User screens,
- Business transactions,
- Workflows,
- Batch processes—essentially iterative business transactions, and
- Reports derived from transactional and historical data.

To ensure consistency across the application, the same business rules are referenced by multiple components. The details of the code generated from a business rule are determined in large measure by the type of component into which it is being embedded.

The second principle of organizing business specifications is to build specifications by linking together small-grained, reusable elements. Types of small-grained specifications include

- Views (constructed from relations, representing derived types),
- Relations (representing base types),
- Tuples (rows of columns in a relation or view, denoting entities),
- Columns (both stored and virtual), and
- Groups (collected sets of columns and subgroups, essentially objects).

Business rules and other attributes can be attached at any level in this hierarchy. Structurally, this model has much in common with object-oriented class hierarchy with multiple inheritance. Data from a view may be affected by business rules attached to any column in the view, rules attached to the view as a whole, or to the records, relations and columns from which the view has been derived.

Visual tools are used to guide the business analyst as he or she builds up the definition of a group by linking together other elements and attaching additional business rules to the group. Typically, an application component is associated with a group that has collected together all of the elements to be used by the component. A single group may be associated with several types of component such as a screen, business transaction and workflow, allowing the generators to create a set of complementary, interoperating components.

The third principle is logical data independence. With logical data independence, the application developer is separated from the inevitable complexity of the underlying logical data model. Specifically, the developer can define a group as a set of columns without considering how to navigate the views and relations of the underlying logical data model. It is left to the application generators to compute the logical connections among data and generate queries as required by the business specifications.
Logical data independence also allows database analysts the convenience of refactoring the information model as required, without impacting developers or the components on which they are working.

The fourth and final principle is the explicit declaration of specification elements, including their business rules and other attributes. The types of business rule that can be attached to a specification element include:

- Validation rules, which check the contents of individual fields when data is entered or changed;
- Derivation rules, which compute the content of virtual fields, e.g., totals and averages;
- Grouping rules, which define sets, where the sets can contain fields and other groups;
- Constraints, which ensure consistency within a set of fields, e.g., debits = credits,
- Sequencing rules, which structure multi-step processes such as workflows, and
- Other attributes indicating type, display characteristics, and updateability.

A component, such as a user screen, is defined as a set of fields to be displayed. From this specification the generator is able to retrieve and weave together the business rules and other attributes for the referenced fields and for any underlying groups, records, views, and relations. When weaving the rules together, the generator makes some assumptions about the order of evaluation—field and group level rules first, followed by rules attached to views and their records, and finally rules attached to relations and their records. Since a view may derive from multiple base tables, and rules lower in the hierarchy may override higher-level rules, the information model is essentially object oriented. The information model and associated business rules are the results of long and careful analyses of an institution’s operations. Application generators provide a means of preserving and reusing this analysis, even if the underlying technical architecture of the application is changed.

5.0 The future of application generation

Some aspects of application generation technology are being incorporated into modern application development environments. This section contrasts the approach to application generation described in this paper to two other approaches currently receiving industry attention, model-driven architecture and .NET templates. Both are being used to help developers manage the complex software technologies required in modern distributed systems and for graphical user interfaces. Even though both approaches can be used to generate a correct technical framework for the application, neither helps developers fill in the framework with business logic.
5.1 Model-Driven Architecture
The Object Management Group (OMG) has established standards for Model-Driven Architecture (MDA). MDA differs from traditional development in that MDA developers generate code from higher-level models, expressed as Unified Modeling Language (UML) diagrams. The code-generation tool uses templates stored in a pattern repository. Of many benefits ascribed to MDA, one of the most important is increased developer productivity.

Software development using MDA includes the following steps:

1. Develop UML diagrams for the domain model, independent of any particular implementation technology (J2EE, Microsoft .NET, CORBA, etc.). This UML model, called a Platform-Independent Model (PIM), represents the core business services and components.

2. Build lower-level UML diagrams showing the technology-specific design patterns for various types of services and components. This lower-level model, called the Platform-Specific Model (PSM), refines the PIM and can be generated using tools.

3. Generate the framework code for the application from the PSM. In the case of J2EE, the MDA generator would create most of the servlets, Java Server Pages (JSPs) and Enterprise Java Beans (EJBs). The generated software would include all the required technology-specific code, leaving developers to fill in details that could not be modeled using UML, such as business logic.

MDA anticipates that development departments will build or modify PIM-to-PSM and PSM-to-code generators to suit their needs. It seems likely that MDA generators will be created or customized to fit software product-line architectures. Some software product-line managers are hoping that MDA will help reduce time-to-market for software and will allow them to create more varied product-lines, addressing the specialized needs often found in highly segmented markets.

Recent controlled experiments comparing MDA with traditional development methods confirm the promise MDA holds for increasing the productivity of development teams. In one study, the MDA team required 35% less time to complete a system than a similar team using traditional methods. Based on these types of early results IBM, IBM Rational, Oracle, Microsoft and Borland are moving to incorporate MDA or similar capabilities into their application development environments.

5.2 .NET Enterprise Templates
Microsoft’s Enterprise Templates serve a similar purpose to MDA but are specifically designed for use in building applications for Microsoft runtime environments such as

---


.NET. An Enterprise Template generates the code skeleton for a developer to follow when implementing a specific software component. With Enterprise Templates, technical architects can build templates for use by other developers, thus ensuring that complex software technologies are correctly used and coding and design standards are followed.

Enterprise templates include both static and dynamic elements. A template can be as simple as some generic code for which the developer must “fill in the blanks.” A template can also contain dynamic elements which execute “on the fly” when a project is created. For example, a template can add dynamic content (custom namespaces, boilerplate code) to the files of a project, perform administrative operations during project creation (for example, use the event log to track the projects developers are creating), and even perform infrastructure tasks (for example, custom source control operations).

Enterprise Templates can achieve some of the same results as the transformations used in MDA. Some technical architects believe Microsoft’s Enterprise Templates to be easier to create than MDA transformations, but because the templates are primarily static structures with some scripted components, they also provide the architect with less ability to express complex design transformations. For this reason and because of its diagramming capabilities, many architects prefer MDA.

There are enough differences between Enterprise Templates and Model-Driven Architecture to ensure that the market for application development environments will be dynamic for a number of years to come. Whether using Enterprise Templates, MDA, or some other approach, most developers are now generating complex software infrastructures that they can later fill in with handcrafted business logic.

5.3 A business-rule approach
A truly powerful application generator would both produce the complex software infrastructure and automatically fill in the business logic, with the business logic being generated from information stored in the business specification repository. Donald G. Ross’s Business Rule Approach describes one approach to documenting and using business rules.

Simple rule-based systems are best understood by comparing them to relational databases. As with relational databases, rule-based systems employ some basic conventions, including some notations for representing data. Relationships within the data are denoted by predicates and can be represented as tables. The rows of data within a relation are called facts. A key difference between rule-based systems and relational databases is the use of rules sensitive to the semantics of the data. A rule-based system can include rules defining additional predicates not explicitly represented by tables, e.g.,

---

the students-on-probation relation where “students on probation include those currently enrolled and having grades less than a C average.”

In enterprise systems other types of rules are needed as well. For example, the system must make it convenient to express and use rules involving time and events—e.g., if payment is late, a late fee is charged; if payment is very late, the account is closed; if the account has just been closed, send the customer mail; if an account has been closed, customer-originated payments are allowed but new charges are not. In summary, Ross identifies three main types of rules—constraints, derived values, and sequencing rules. All three types are used in Datatel’s business specifications.

Finding effective and efficient ways to use these rules is traditionally the work of a programmer. Handcrafted code, while expensive to produce, can execute quite efficiently. In some architectures, to lower development costs, inference engines are used to automate the processing of rules. Unfortunately, inference engines can be inefficient, deciding at runtime, sometimes by trial-and-error, the order in which rules are to be executed. Application generators strike a balance between handcrafted code and the use of inference engines. Application generators hardwire the order in which rules are to be used, but do not synthesize optimized algorithms.

In Datatel’s experience, the approach to using rules depends on the nature of the problem being solved. Much of the computation performed in enterprise systems involves relatively rudimentary reasoning, e.g., using one or two rules to validate a field value or multi-field constraint. When the reasoning is rudimentary, code generation is simply a matter of retrieving rules from the business specification repository, translating them, and appropriately embedding them in the component. In some cases, however, the reasoning is more complex, requiring that a number of rules be chained together, possibly with some backtracking to find the right combination of rules. For these cases, a rules engine may be used. In a few cases, where significant analysis is required to find an efficient algorithm, the programmer may want to embed the rules using handcrafted procedures. Examples of tasks that, for efficiency, should be handcrafted include degree audit, matching payments to invoices, and processing refunds. Even when handcrafting an algorithm, the developer needs to reference the business-specification repository to ensure consistent, complete, and correct use of the applicable rules.

Datatel’s generator architecture is an example of an industrial-strength application generator that separates the developer from the details of the technical architecture and provides the tools needed to collect and use business rules.

6.0 Conclusion

The computer science literature documents a long history of investigation of how computers can be used to automate the development of computer software, beginning with macro assemblers and FORTRAN. Continued advances in computing power, combined with ongoing analysis of software development tasks, reveal a steady stream of

---

13 Modal logic is the study of reasoning about time and tense. Results from modal logic have been used extensively in computer science, both in artificial intelligence and in reasoning about the behavior of real-time, distributed systems.
opportunities for additional automation. The close connections between computation, logic, and human reasoning lead to many opportunities for automating some of the more complex design tasks that are currently the province of human software developers.

The wide-spread belief within the software engineering community is that visual modeling, combined with some sort of generative or template-based reuse technology, will help developers remain productive even as they work with the increasingly complex technologies required in enterprise-scale distributed applications.

Datatel’s approach to application generation carries this an important step further, demonstrating how generators can be used to automatically translate and embed business logic in application components. Datatel’s methods for generating application components are well-suited for building large business applications, allowing application developers to focus on the definition of business specifications and insulating them from the details of the runtime architecture and the process of weaving business specifications together into components.

Datatel’s approach to application generation is particularly valuable to developers caught between the accelerating rate at which architectures and technologies are changing, on one hand, and the ever-larger investment in business knowledge, on the other. The Datatel approach helps developers preserve, in a single authoritative source, an institution’s single most powerful IT asset, the business knowledge embedded in its software applications.

The efficacy of this application generation approach is demonstrated by the fact that Datatel has propagated a single set of Colleague business specifications into multiple server and database environments. The consequence is a simpler system to install, upgrade, support and maintain.
Acknowledgements

The authors wish to acknowledge a number of contributions. Kelly Tucker, Sue Kumpf, and Dr. Laird Sloan provided valuable insight into the customer value created through the use of generative programming methods and tools. Harrison Smith carefully explained the benefits and challenges of developing and maintaining business specifications. Richard Henninger explained, in great detail, the metadata, the evolution of the target technical architecture, and the design and operation of the application generators themselves. Dr. Sloan also provided much appreciated detailed reviews of earlier drafts of this paper.

About the Authors

Stephen B. Ornburn, sbo@gbc-group.com, has been researching, writing, teaching and consulting on software engineering and software architecture for over twenty years. Through his company, GBC-Group, he provides consulting services to developers of software-intensive products and services. Steve’s work on reverse engineering, application generation, software reuse, and software development methodology has been published in IEEE Computer, IEEE Software, CrossTalk, and leading conferences and workshops. Steve’s education includes a bachelor's degree in industrial engineering from Northwestern University, and a master's degree in computer science from Georgia Institute of Technology.

Pete Nalli, vpn@datatel.com, is Director of Architecture and Software Development for Datatel. Pete has been designing and building software development tools since the mid 70s. In the early 80s, Pete co-designed Datatel’s first application development environment, which used application generators to build Datatel’s first fund-raising solution, Benefactor. This generative programming environment was also used to build Colleague, an integrated information technology solution designed specifically for higher education. In the 90s, Pete headed up Datatel’s research and development group and was instrumental in moving Datatel's architecture from a single database focus to its current three-tier, database-independent design. For over thirty years, Pete has been an innovator of development tools and technical architectures with a constant vision for the future that incorporates solving the business needs of higher education institutions.

Tom Reynolds, tar@datatel.com, is Vice President, Software Development and Quality for Datatel, Inc. Tom joined the Datatel team in 1980 as the original product development manager for Colleague. He has provided leadership in many areas of the company, including marketing, distributor sales, as CIO, and in new product development. During Tom’s tenure at Datatel, the company received the U.S. Senate Productivity and Quality Award (SPQA), which is the highest quality award presented by the State of Virginia. Tom attended the Catholic University of America, where he earned his BA degree.