

**Potential Issues and/or Opportunities for the Department of Health and Family  
Services in  
Moving From Microsoft Office 97 to Office 2003**

**Part II**

**Fred Buhr  
Data Administrator/Consultant  
Center for Uniformity, Security and Privacy  
Bureau of Information Systems  
Department of Health and Family Services**

**June 2004**

**Target Audience:** Denise and Ted

**Objective:** To present a comprehensive overview that will inform administrative decisions concerning implementation of the Office 2003 System in the Desktop Operating System/Office Suite Upgrade (OS2U) Project.

## Table of Contents

I.	Case Study - "DMT 25 Forms / Publications Requisition" E-Form.....	3
	• Description of the DMT 25 E-form and Process	
	• Question: Will the Word 97 electronic version of the DMT 25 function as designed when DHFS migrates from Word 97 to Word 2003?	
	• Question: What new interface features are activated by virtue of moving from Word 97 to Word 2003?	
	• Question: Would there be advantages in re-programming the DMT 25 as a Smart Document?	
	• Question: Would there be advantages in re-programming the DMT 25 as an InfoPath form?	
II.	New Strategic Functionality – A “Forms-Centric” Architecture.....	11
	• Question: What would a Forms/Publications Center scenario based on a "forms centric" architecture look like?	
	• Question: What is a “forms-centric” architecture?	
	• Question: Would a DHFS "forms-centric" architecture conflict with the emerging DOA Enterprise Architecture?	
	• Question: What would be the steps in developing an XML based “forms-centric” information architecture?	

# I. Case Study - "DMT 25 Forms/Publications Requisition" E-Form

## Current System

For a number of years, automation of the DMT 25 was targeted to be the first electronic forms project to be designed using the Adobe Designer (formerly JetForm) and was to be made accessible through the FormFinder Catalog. Because technical and software incompatibility problems delayed implementation of both the designer and catalog, an initial study and cost estimate for automating through traditional programming and project development was requested and completed by BIS. A cost estimate of approximately \$30,000 was attached to the project. Because of the cost, it was decided not to pursue a programming solution but rather to explore the use of Word 97 as a forms design tool with accessibility for customers provided through the current forms lists on the DHFS intranet and internet sites. The Word 97 solution was rolled out to the DHFS Forms Managers and internal (DHFS WorkWeb) intranet customers in April 2004 and, barring unforeseen problems, will be rolled out to internet customers by the middle of June 2004.

The Word 97 solution utilizes an electronic document that is transmitted from customers to the DHFS forms managers as an e-mail attachment. Dedicated DMT 25 mailboxes have been established and instructions as to which mailbox the form should be sent are available to customers. The electronic version of the DMT 25 functionally replicates the multi-ply hardcopy version by activation of select fields through Visual Basic for Applications (VBA) "programming" that is activated by pressing "secret" key combinations.

Fillable fields available to the customer relate to shipping address and number and title of forms being requested.

Pressing the special key combination of "alt+m" activates forms manager's fields relating to changed quantities, codes for certain situations, and a field for sign off and date. Forms managers via e-mail then send the form to the Forms Publications Center for processing.

The screenshot shows a complex form with multiple sections. At the top, there are fields for 'SHIP TO:' and 'FROM:'. Below this is a table with columns for 'FORMS', 'QUANTITY', 'FORM TITLE', 'FORM NUMBER', 'FORM CODE', and 'FORM DATE'. At the bottom, there are sections for 'REMARKS', 'ORDER FILLED BY', and 'CODE'. Arrows from the text boxes point to various fields in the form, including the shipping address, the table headers, and the 'ORDER FILLED BY' field.

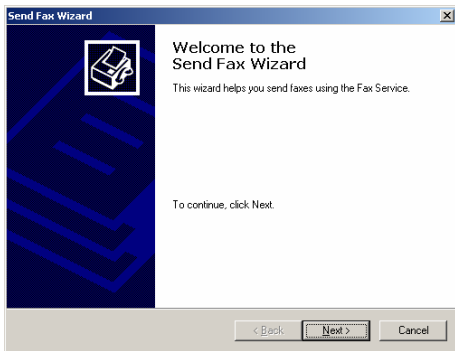
The special key combination "alt+p" activates a special printing process in the Forms/Publications Center that prints the whole form on two ply (carbon less NCR) paper at the same time it sends the shipping address information to a DYMO label maker for shipping label(s). The Center staff use the two (white, yellow) ply form as a pick list. The white copy with handwritten "Order Filled By" information is sent along with the order to the customer while the yellow copy is retained, as necessary, for back orders and as a source for keying information into the center's inventory system.

**Question: Will the electronic DMT 25 form itself function, as designed, when DHFS migrates from Word 97 to Word 2003?**

As a basis for looking at this question, a copy of the Word 97 form was saved as a Word 2003 document on a standalone laptop. The two forms were then compared on monitors, side by side. Allowing for differences in the Word programs themselves, there do not appear to be any visual differences in presentation of the form itself. As far as I can see, both Word 97 and Word 2003 present the form identically.

Testing for functionality, in customers' mode, no differences were noted. All fields were appropriately either active or inactive. Likewise, no differences were noted in managers' mode. The "alt+m" mode toggle functioned correctly.

However, testing the forms' center mode revealed that "alt+p" invoked a native Word 2003 macro that invoked the "Send Fax Wizard". Rather than firing the macro that prints the whole document through the dot matrix printer and the shipping label through the DYMO label maker, the wizard appeared.



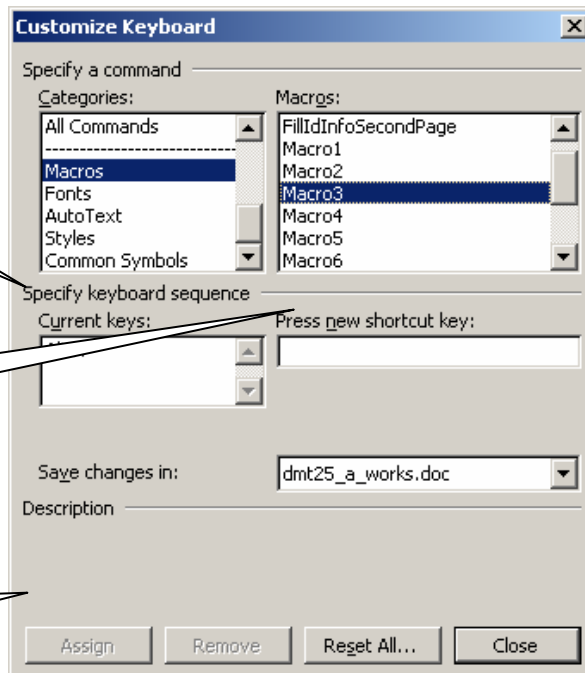
Rather than waiting for the migration, the keyboard combination should be changed to something that isn't being used by Word prior to rolling out the form on the internet.

The procedure for changing keyboard shortcuts is similar in Word 2003 to Word 97. Choose "Tools, Customize" At the bottom of the Customize dialog box, click the keyboard button. Word then opens the Customize Keyboard dialog box shown below.

This is the keyboard sequence that must be changed.

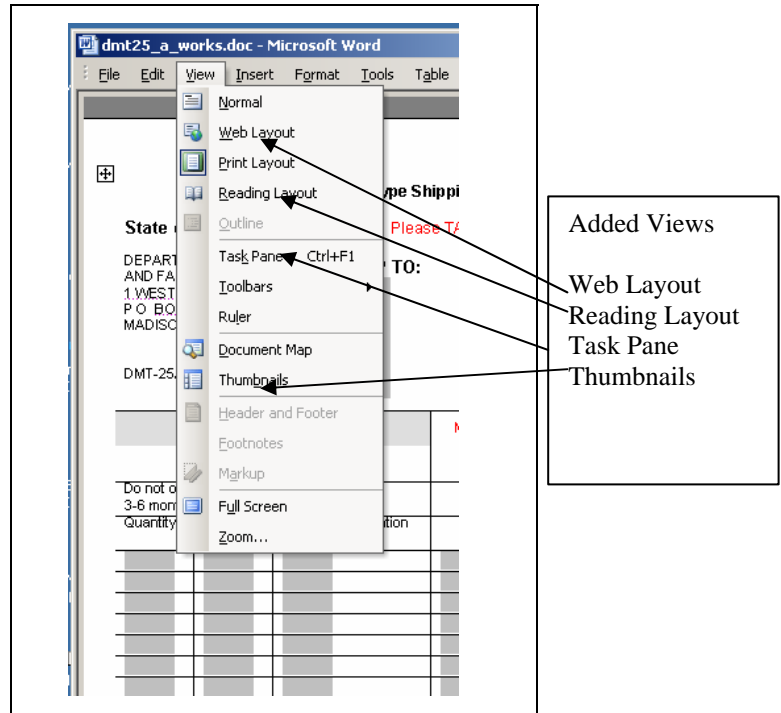
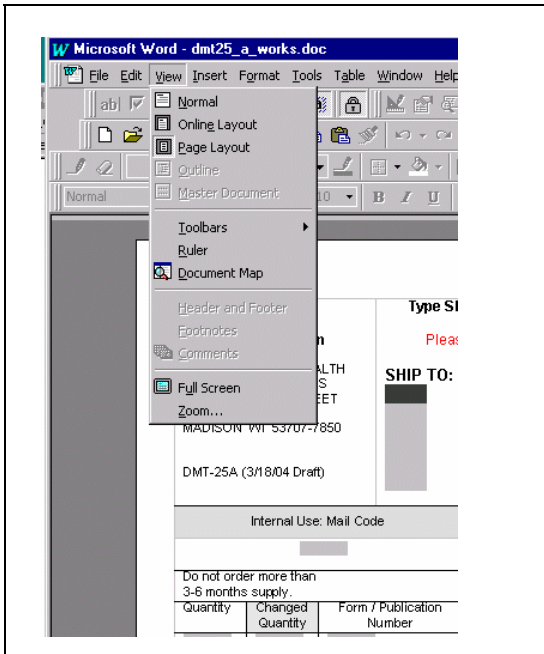
Pressing a different keyboard sequence will enter the keys in this field.

The "Assign" button will then become active and when pressed will change the keyboard sequence for the





A very noticeable change is that of the dropdown menus in the form "View" in the tool bar. Most significant are added views for Web and Reading Layouts, Task Pane and Thumbnails.



Although not very useful for the DMT25, users will find the reading layout to be of a great deal of use for many documents. Word 2003 has an optimized a view for reading - rather than editing or creating. This view divides the document in to small pages that fit completely within the Word window, simulating a book. One or two pages can be presented at a time and the text is presented in a large, smoothed font. The layout is designed for the newer flat screen monitors. I recall reading an article that indicated the appearance, although acceptable, is much less effective when viewed on the older, curved screen monitors. Since I've only tested Office 2003 with flat screened monitors, I have not been able to observe the different effects.

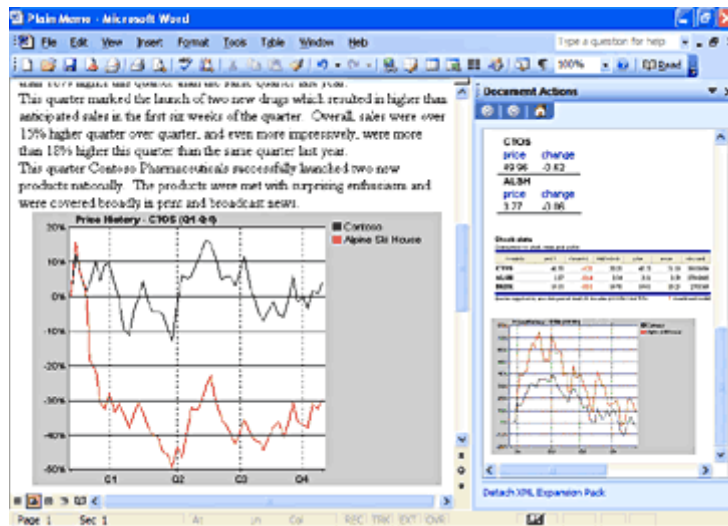
**Question: Would there be advantages in re-programming the DMT 25 as a Smart Document?**

I haven't had the opportunity to explore "smart documents" but the concept seems to have a great deal of potential for usefulness. I've taken the following information directly from the *Microsoft Office 2003 Editions Product Guide, September 2003*:

"Smart documents are XML-based solution development and deployment platforms that are available in Word 2003 and Excel 2003. Smart documents help developers to tackle business-process problems by quickly building document-based solutions that combine the advantages of the Word and Excel desktop programs with the advantages of Web services. Benefits of these solutions include easier deployment and updating of the solution, and a larger selection of tools for solution development.

Smart documents bring relevant information directly to the task at hand through a new "Programmable task pane" user interface. With smart documents, XML solutions can be created to enable business processes and help users complete forms and other documents, and then link that information to back-end systems that support XML. As the user clicks through the document, the program modifies itself to present the appropriate functions and help to complete a task.

Smart documents can easily be incorporated into business processes such as expense reporting, contracts, or anything else that might pass through multiple hands or systems while being authored, or that require information from back-end sources. A Smart Document solution can also include custom Smart Tags that are designed specifically to run only in that solution, or any other code that the developer would like.



*Smart documents use the new "Programmable task pane" shown on the right to bring relevant information directly to the user.*

Using smart documents makes solution deployment much more flexible. Solution code can now be deployed once to a central server location, and Word and Excel will securely download and cache it locally when the smart document is opened. Updates to the solution code and even the document template itself can be delivered the same way, as a single update to the server location for the smart document solution, without having to deploy directly to hundreds or thousands of clients, just as with Web-based solutions. But

unlike Web-based programs, the Smart Document solutions can operate even when the computer is offline, because they use the Office programs rather than a browser.

Smart documents are new, and their use is virtually unlimited. The following are a few example scenarios.

- A purchase order solution using Excel might be linked to a business rule in a back-end server that takes the User ID and applies the user's current signing limit to the order directly in Excel, while the user is completing the form.
- A social services solution could be created that translates forms into different languages or assists applicants in completing the form, with context-sensitive help.
- Depending on the user type or ID, or on input from the user, a Word document could reconfigure itself to include the necessary sections that the user needs to complete—for example, a performance review form that has different sections for management and employees. "

**Question: Would there be advantages in re-programming the DMT 25 as an InfoPath form?**

Yes, particularly because the DMT 25 could easily be integrated as a part of a business application that would automate the Forms/Publications Center's ordering and re-ordering process. Both the paper and electronic versions of the DMT 25 are structured so that the data entered by the customer and subsequent data entered by the forms managers and forms center staff are bound to the DMT 25 itself.

Key to the advantages is the fact that an InfoPath solution would allow for separation of data from the form itself.

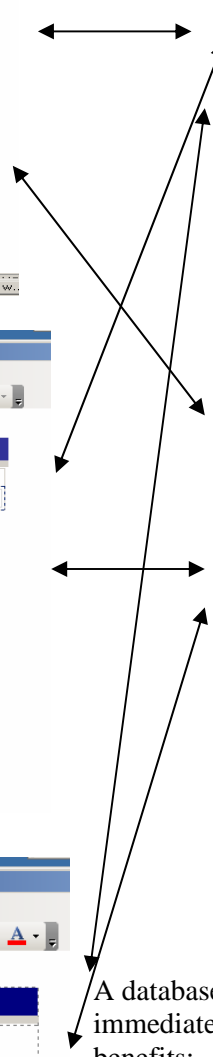
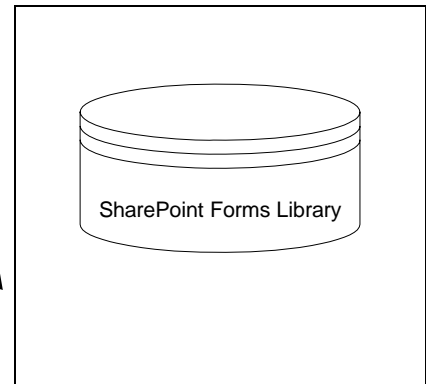
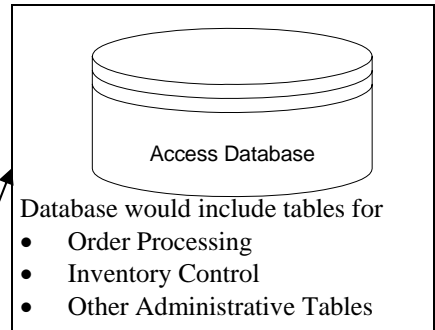
The data entered by the customer would be saved and transmitted in a .xml file. The schema of the form would be saved in a .xsd file.

InfoPath has its own design mode for form development. Within design mode, it is possible to save an InfoPath form template as individual files and then work with them in other programs. Separating the data from the form would allow

for automation of the total information exchange process. Microsoft designed InfoPath as a new desktop application and set out the following core design goals and main architectural features:

- Build a hybrid tool combining the best of a document editing experience with the rigorous data-capture capabilities of forms.
- For input and output, use XML documents belonging to a custom-defined schema.
- Provide structural editing.
- Provide flexible views.

Using InfoPath 2003, it would seem to be feasible to combine and transform the Word97 design into an order entry and inventory system that would automate most aspect of the process. Since it wouldn't be necessary to dislodge or interrupt the current process, while developing and testing a an Office 2003 solution, automating the Forms/Publications Center business process could serve as a laboratory type setting to test a "Forms XML Centric" approach. A Microsoft Office Access 2003 data base would be sufficient for testing and could be easily upgraded to SQL Server as needed.



A database for storing orders would immediately have the following key benefits:

- Customer friendly search for forms titles
- Customer connection to inventory availability information
- Customer informed immediately as to supply limits
- Easy integration with other Windows Services
- Easy reuse of information for reporting and business intelligence

Further, scaling up, the Access 2003 database could be expanded to include inventory information that is now a separate database and process.

## II. New Strategic Functionality - A "Forms-Centric" Architecture

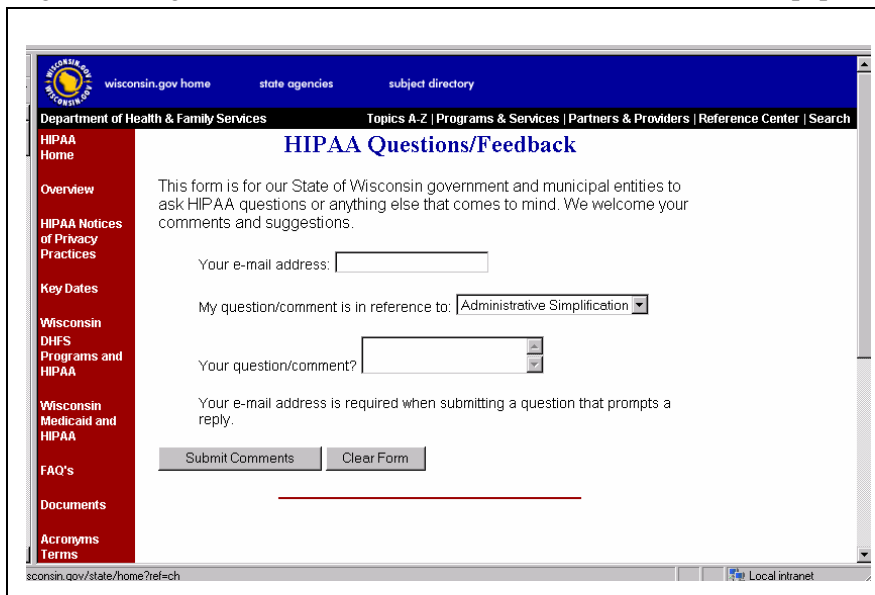
### Question: What would a Forms/Publications Center scenario based on a "forms-centric" architecture look like?

The architecture would be based upon the legal definition of "forms". Quoting from Module 1: Forms and Forms Management training: "Wisconsin State Statute 16.97(5p) reads; "Form" means any written material, by whatever means, printed, generated or reproduced, with blank spaces left for the entry of additional information to be used for the purpose of providing information, collecting information or requiring action in any transaction involving this state.

The definition of a form is very broad. It is meant to include paper forms, computer generated print on demand or fill forms, computer screens or any other media that contains predefined script and blank spaces or fields to allow for the insertion of information. In DHFS, if instructions for completing a form are a separate document, it is to be handled as a form.

The most distinguishing factor in the definition of a form is that it contains blanks, spaces, or fields that allow for the entry of data in a predetermined format. This is what distinguishes a form from other documents."

The distinguishing factor of a form, "that it contains blanks, spaces, or fields that allow for the entry of data in a predetermined format" applies to electronic forms developed in Microsoft Office 2003 (as well as earlier versions). Following is an example of an HTML form generated through FrontPage 2000. In a "forms-centric" architecture, all forms, paper or electronic, would



The screenshot shows a web browser window displaying a form titled "HIPAA Questions/Feedback". The page header includes the Wisconsin Department of Health & Family Services logo and navigation links. The form itself has a red sidebar with links like "Overview", "HIPAA Notices of Privacy Practices", "Key Dates", "Wisconsin DHFS Programs and HIPAA", "Wisconsin Medicaid and HIPAA", "FAQ's", "Documents", "Acronyms", and "Terms". The main content area contains a text box for "Your e-mail address:", a dropdown menu for "My question/comment is in reference to:" (set to "Administrative Simplification"), and another text box for "Your question/comment?". Below these are "Submit Comments" and "Clear Form" buttons. A note states: "Your e-mail address is required when submitting a question that prompts a reply." The browser's address bar shows "scconsin.gov/state/home?ref=ch" and the status bar indicates "Local intranet".

be subject to standards and be identified with a unique number. The HIPAA illustration may also have use in categorizing various form states. The form rendering (the graphic) that one sees in the browser, in e-forms nomenclature is identified as a "GUI". "GUI" is

the acronym for Graphical User Interface. In the past, in text based systems, similar input screens were most often simply referred to as "screens". Often, the perception of what a form is, relates to the paper based form that has been in use for so long. Electronic forms usually have two states, one relating to the GUI, for input while the second relates to a printed presentation which looks like the paper based form. Fidelity in representation of a paper based form is often seen as being most desirable.

The major technological advancement in Office 2003, relates to bringing XML to the desktop. By way of the HIPAA illustration, FrontPage 2003 has a source code editor that supplies an XML

tool bar with appropriate commands. XML code, as has been shown in earlier sections of this paper, is a tagged code that looks very much like HTML. HTML (in contrast to XML) defines a specific set of elements so that browsers can universally interpret the language, and has relatively loose rules of syntax. FrontPage 2003 has the ability to apply XML rules of syntax so that all the code on a Web page will conform to XML rules.

Bringing XML to the desktop means that data can be decoupled from other aspects of the document used for data collection by way of an XML schema. While the HTML tags describe how to render or display the data, the XML tags describe the data itself. Thus, regardless of where the information resides in whatever document, and regardless of how it is formatted, as long as it is tagged within the context of an XML schema, it is accessible through XML.

The XML desktop solution is particularly relevant to DHFS. Our agency already employs tightly defined standards for data capture on paper and (to a somewhat less extent) online and to a much less extent when captured through e-mail. Formalizing the structures of these forms where they exist and developing categories and formalizing structures of forms transmitted through e-mail is simply a matter of developing schemas that describe the information. Since many of our forms have been developed using Word and Excel, a nearly seamless transition can be made to an XML based system, requiring little or no retraining of employees or replacement of existing systems.

- XML-enable desktop applications enable agencies to capture data pre-tagged as XML, to facilitate storage, re-use and exchange of information.
- Use of the familiar Office environment minimizes retraining.
- XML and XML Web services promote interoperability across platforms and systems, minimizing replacement or upgrade of existing systems.

The Microsoft Office System 2003 presents an opportunity to implement a forms and records architecture based upon current unconnected (stovepipe) applications and systems. It further represents an opportunity to design and develop applications based upon business processes that are currently in place. Legacy systems and processes need not be disturbed or displaced.

It seems to me, that we might be at a point where DHFS could establish an information architecture through wide participation of staff across the Department by focusing on forms as the central organizing vehicle. Since (nearly) all data utilized by the Department is represented on a form that is or has been stored in the DHFS Forms/Publications Center and (for the most part) is recorded on the inventory maintained in the forms center, it would be possible (not easily done, but possible) to comprehensively review all the forms and list all the (different) data elements for each form.

The rapidly maturing XML technology along the evolution of forms design tools as InfoPath and Adobe Forms Designer provides a most rare opportunity to specify an information architecture that encompasses legacy and current systems, is programming language agnostic, and platform independent. Furthermore, the forms automation approach can also help instill in users, in a non-confrontational way, the necessary discipline and incentive to participate in establishing and maintaining enterprise wide data dictionaries. Each time the collection of highly structured data occurs, the E-forms system would be available to facilitate the effort, by and for anyone in the organization.

Starting with the notion that virtually all data collected and stored within the Department of Health and Family Services is or was a part of a form, the 5,500 plus forms that are stored and made available through the forms center.

The new Microsoft Office System 2003 itself is an implementation the Microsoft .Net Framework which defines everything – from intrinsic variables, right up to full-blown applications – as explicit objects. The Framework contains hundreds of objects which are defined as classes. These objects are the fundamental building blocks of .NET programming. The .NET Framework uses a naming scheme to organize all its classes: **namespaces**. A namespace is a group of similar classes or objects. The names of namespaces are usually quite short but always very accurate in describing what classes they contain.

Use of namespaces in .NET stem from their use in XML schemas, many situations require the structure of an XML document to be well-defined in order for software to usefully interact with it. There are far too many objects to usefully put into a list. But since the namespaces are organized hierarchically, it's quite easy (and intuitive) to browse through them and find what is available.

Namespaces in .Net are easily distinguishable from namespaces in XML. Although namespaces in XML look like URLs (Universal Resource Locators) they are actually URIs (Universal Resource Identifiers) and serve as identifiers rather than locations. A “forms centric” architecture is basically one of documenting, classifying and identifying the schemas that already exist as data elements on forms. The 2003 versions of Word, Excel or InfoPath (as Adobe Forms Designer) can be used as graphical XML forms design tools to associate XML name tags with data elements on forms.

From a theoretical standpoint, a paper form, as a type of document, inherently possesses remarkable properties. It can, and is most often used to, gather highly structured information in a highly structured way. The structure itself ensures that the data can be efficiently and effectively processed, shared, and used with a minimum of confusion. And it can be done with little or no additional process in terms of data interpretation, conversion, or reformatting. If a data element is not already represented on a form somewhere in an organization, the need for it may be questionable. Taken in the aggregate, the data elements included in the forms of an agency provide a comprehensive view of the data that drives the business. The basic frustration with forms is that the very same data must be re-supplied so many times. Forms design tools utilizing XML can electronically remove that frustration. But, a serious amount of upfront work is necessary and explicit definitions of data elements must be developed and adopted.

IntraFormNumber	IntraTitle	_IntraTitle
CFS0149A	Family History Questionnaire Medical / Genetic Pregnancy and Delivery Information	81
InterFormNumber	InterTitle	_InterTitle
CFS-0149A	Family History Questionnaire Medical / Genetic - Pregnancy and Delivery Information	83
CenterFormNumber	CenterTitle	_CenterTitle
CFS0149A	FAMILY HIS. QUEST. MED/GEN PREGNANCY & DELIVERY	48

This is an example of the same form being listed in three different directories,

The Department of Family Services likely has over 6500 forms with the exact number difficult to determine because there is no single listing that can be considered to contain the authoritative records for forms. The Forms/Publications Center maintains an inventory of forms that contains approximately 5545 numbers and titles. A listing maintained by the Department on its Internet Site lists approximately 587 forms, some of which are included in the Forms Center

inventory. Another listing, of approximately 1669 forms, some of which are listed on the Internet site and maintained in the Forms Center inventory, is maintained on the DHFS Work Web.

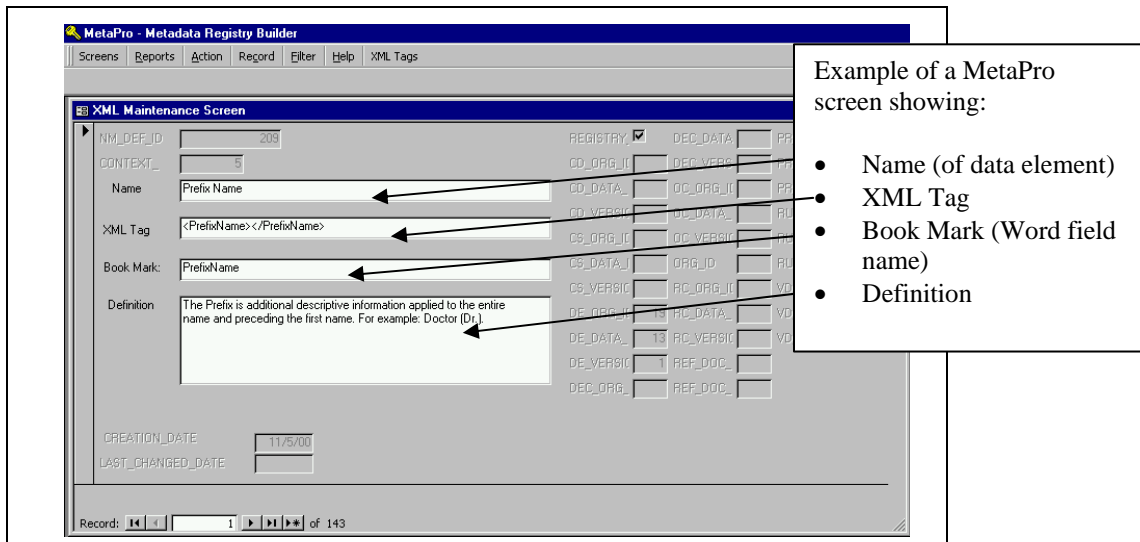
In order to develop a “forms-centric” architecture, it would be necessary to develop an intuitive system of classification that would enable both users of forms and administrators to understand the architecture. An intuitive “namespaces” classification scheme could be developed for the Department of Health and Family Services (DHFS) generally following a hierarchical structure (as does .Net and XML) but related to organizational structure (identified by an “as of date”) showing ownership of a specific schema identified by an XML schema namespace attached to a specific form.

Using the DMT25 for example, a fully qualified organization path leading to its XML schema might look like: <schema xmlns = “<http://DHFS.DMT.BIS.CUSP.DMTS0025.xsd>”</schema>. The schema itself (the .xsd file) would contain all data elements found within the form. The classification scheme could easily be expanded to the State enterprise level by similarly identifying all forms with the higher level organizational structure namespace, for example, beginning with an agency designation such as DNR or DWD, etc. I’ve seen an estimate of there being 48,000 forms on a statewide basis.

At any rate, for purposes of this scenario, the Forms/Publications inventory system would be expanded so as to be inclusive of all forms, paper or electronic, GUI (graphical user interface) screens, and print only (high fidelity forms representations). In addition to information relating to inventory supplies and reorder points, there would be information concerning form ownership, versions, and all data elements included in the **MetaData Registry**. A print version of each form and screen would be available for all users. Where forms are a part of a record series covered under a Records Retention/Disposition Authorization (RDA), such information would also be available.

The expanded inventory system would be maintained by Divisional Forms Managers under the supervision of the DHFS Forms and Records Officer. New schemas for forms would be proposed by Divisional Forms Managers with approval the DHFS Forms and Records Officer.

Closely related to the inventory system, would be a registry of all data elements that would be maintained by a Registrar under supervision of the DHFS Forms and Records Officer. The



registry would be modeled on the MetaPro Data Registry (beta version) that was used for documenting the DHFS Common Core Data Standards but expanded to include XML tags for all data elements. The MetaPro Registry would maintain a comprehensive set of definitions for all data elements and related XML tags.

**MetaPro - Metadata Registry Builder**

Recent Additions | Contact Us  
EPA Home > MetaPro

**MetaPro**  
Metadata Registry Builder

MetaPro is a registry that allows entry for definitions related to the following concepts:

- Conceptual Domains
- Classification Schemes
- Data Concepts
- Data Elements
- Object Classes
- Organizations
- Permissible Values

MetaPro is a tool that allows users to create their own metadata registries. A distributable beta version was developed using the MS Access database management system. The development was sponsored by the United States Environment Agency (EPA) and the former Health Care Financing Administration (HCFA). Because of funding constraints, the distributable version has not been developed further.

MetaPro, a tool that allows users to create their own metadata registries, was designed to foster exchange of metadata and encourage reuse of metadata attributes of data elements and value domains. MetaPro is available in two versions: the hosted version of MetaPro that uses the Oracle database management system and resides on an EPA UNIX server; and the distributable version of MetaPro, developed using the MS Access 97 database management system that will be available for use on Personal Computers (PC). The development of MetaPro was sponsored by the United States Environmental Protection Agency (EPA) and the Health Care Financing Administration (HCFA).

**What Is a Metadata Registry?**

organizations use registries as repositories of standard data elements used as models for data developing applications.

**How Can a Metadata Registry Help an Organization Manage Its Sources of Metadata?**

Various government agencies and nongovernment organizations are responsible for managing data sets. For example, the U.S. Geological Survey manages Hydrologic Unit Codes (HUC), the U.S. Environmental Protection Agency maintains an addressing standard and manages the official list of Zone Improvement Plan Codes, and the National Institute of Standards and Technology manages various data representation lists. These are a few examples of code sets that can be incorporated into a metadata registry. standard metadata and code sets can improve the quality of managed data and reduce duplication.

**Why Use an International Standard-Based Metadata Registry?**

MetaPro is based on the International Organization for Standardization/International Electrotechnical Commission (ISO) 11179 metadata standard, which describes the specification of a registry for managing metadata to ensure that the representation of relevant characteristics is consistent and accurate. This standard is designed to foster the exchange of metadata between members of the information technology community.

MetaPro's database design is based upon the current version of the metamodel for shareable data, which was set forth in the American National Standards Institute (ANSI) X 3.285 standard (November 1999) and which is currently being incorporated into a revision of ISO 11179, Part 3, Specification and Standardization of Data Elements, Basic Attributes of Data Elements (November 1999).

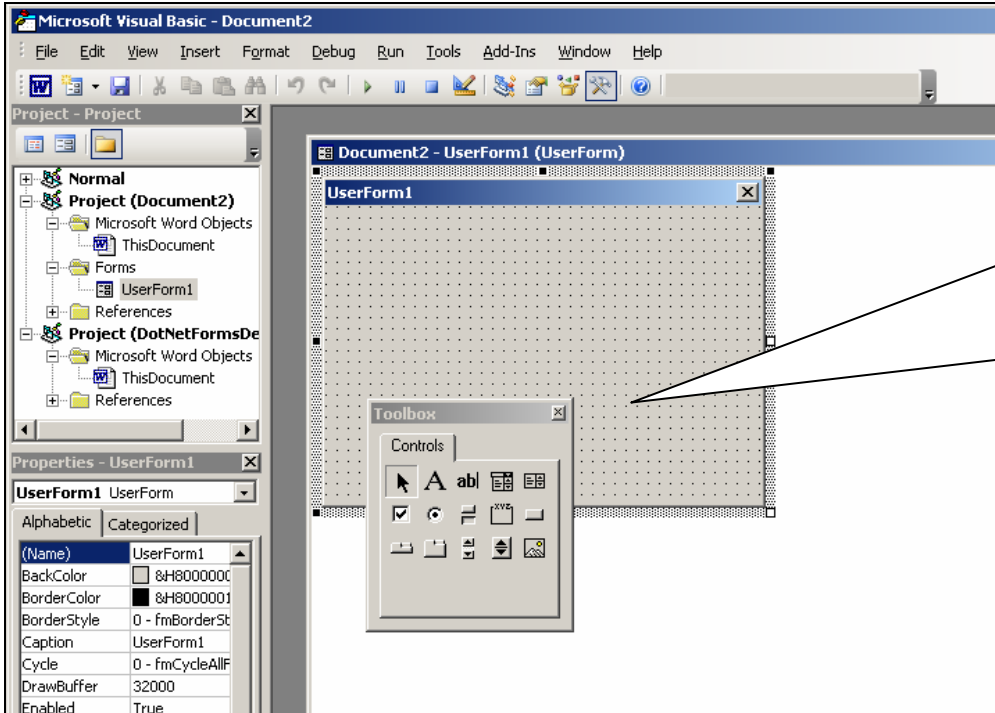
For additional information click below:

- [MetaPro - Hosted Version](#)
- [MetaPro - Distributable Version - Currently Under Development](#)

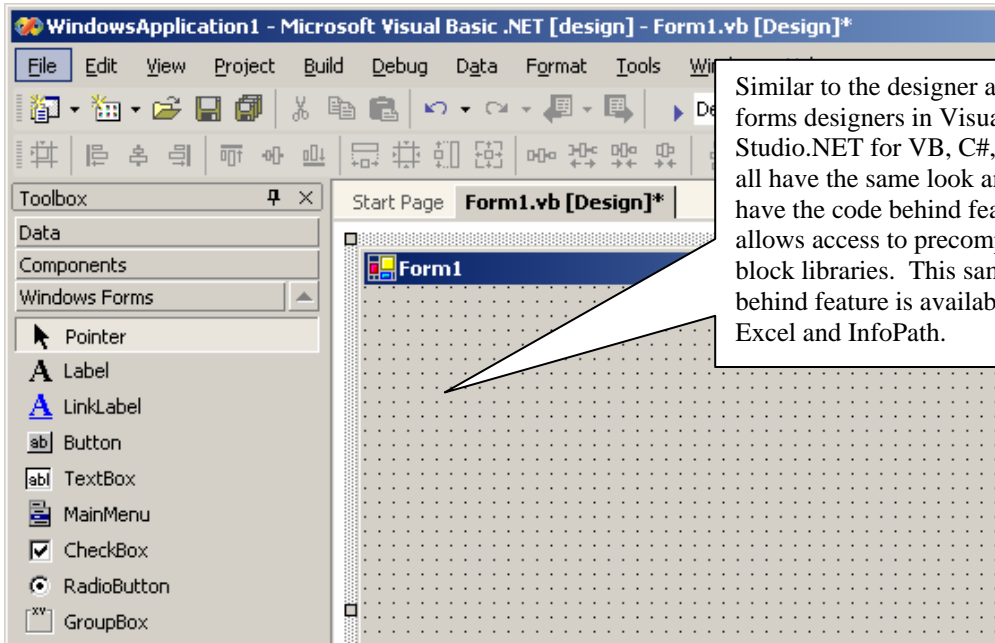
MetaPro is based on the International Organization for Standardization/International Electrotechnical Commission (ISO) 11179 which describes the specification of a registry for managing metadata to ensure that the representation of relevant characteristics is consistent and accurate.

MetaPro's database design is based upon a version of the metamodel for shareable data which was set forth in the American National Standards Institute (ANSI) X 3.285 standard.

The envisioned scenario could be developed by bolting together XML functional pieces of Microsoft Access, Excel, Word and FrontPage (without programming). Because the forms design tool in Visual Basic for Applications (Excel and Word) is basically the same as the forms design tools for Visual Studio.Net (Visual Basic.Net, C#, J#, C++, etc) very little knowledge of actual programming is needed to develop sophisticated applications. Following are screen shots of various designers showing the similarities.

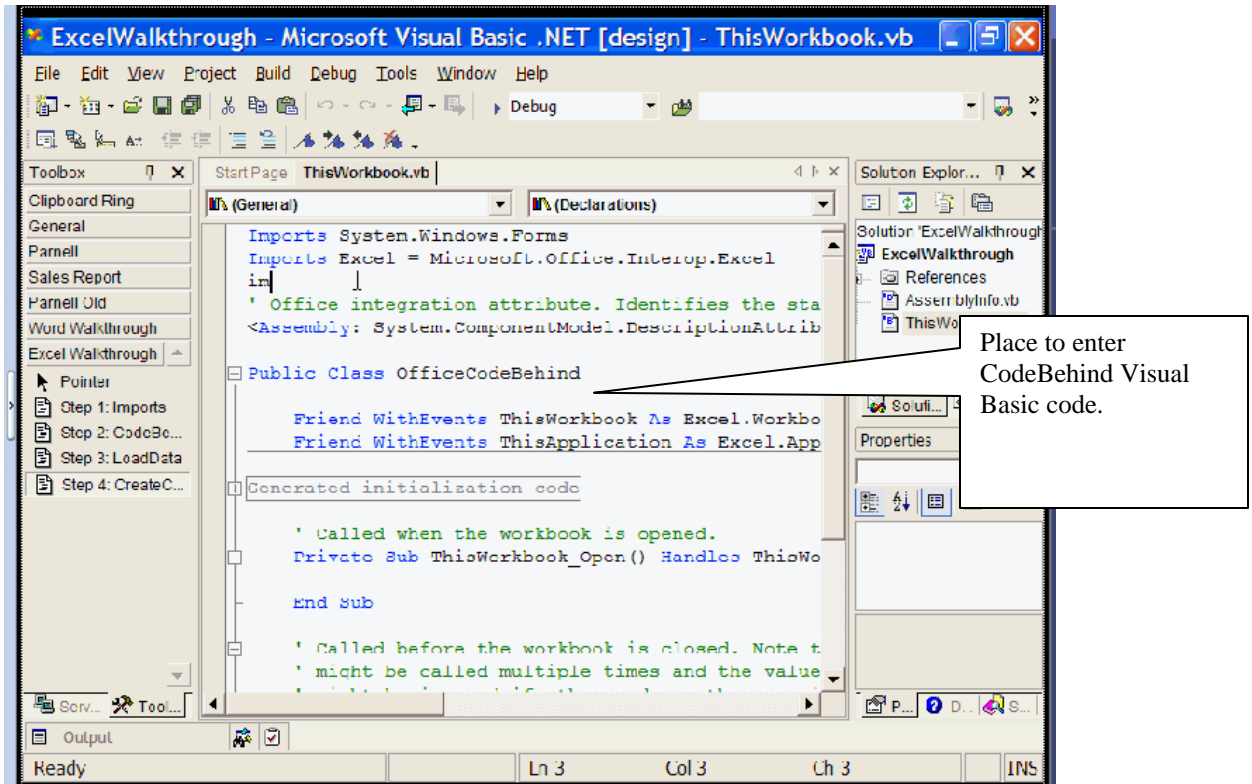


Word and Excel Forms designers are identical. This screen print for the Word forms designer looks identical to that for Excel. Only the project objects (seen in the task panel on the left) are explicit to Word or Excel.



Similar to the designer above, the forms designers in Visual Studio.NET for VB, C#, J# and C++ all have the same look and feel. All have the code behind feature which allows access to precompiled code block libraries. This same code behind feature is available for Word, Excel and InfoPath.

Visual Studio.Net is the framework that now unifies the Microsoft Office System. New Office tools now enable developers to directly use the framework from within Visual Studio. Using the Office Tools System Development Kit (SDK) it is now possible to select a specific "code behind" template for an Excel Workbook, Word Document or Word Template. In addition, an SDK has been released specifically for InfoPath. This example taken from an Excel WalkThru Video shows the code behind feature.



**Question: What is a “forms-centric” architecture?**

"Forms-centric" is a term that I attached to ideas which come out of a paper entitled, *Automated Forms: Putting the Customer First Through Intelligent Object-Oriented Chunking of Information and Technology* by Owen Ambur. Although the article was written prior to E-forms becoming XML enabled, the ideas incorporated in the paper foreshadowed the current XML environment.

Currently, Owen Ambur is the Chief XML Strategist for the U.S. Department of the Interior (DOI) in the Office of the Chief Information Officer (OCIO). He is also responsible for managing the DOI Enterprise Architecture Repository. He co-founded and co-chairs the CIO Council's XML Working Group (<http://xml.gov>). He is a former vice chair of the board of directors of the Federal Information and Records Managers Council (FIRM), a voluntary association of Federal information and records managers.

Ambur's paper was written as an answer to the question, what is public-service agency to do? in response to the Information Technology Management Reform Act (ITMRA) of 1996. That act embraced the notion of shared data and systems, and extended it beyond individual departments and agencies – to treat the entire Federal Government as a *single* enterprise for purposes of interagency information technology (IT) investments. Ambur posits that object-oriented programming (OOP) provides a pertinent model to meet the challenge.

OOP is based upon the encapsulation of properties, events, and methods together in logical, interoperating sets – called ‘objects’ – that convey meaning and/or ‘act’ appropriately when called upon to do so. In OOP terms, objects are "chunks" of larger systems in which some form of intelligence is embedded.

Ambur says, “While OOP has gained great credence among IT specialists, the concept of ‘objects’ is still too technical and ill-defined to capture much mindshare among the average user of information systems. The average user continues to think in terms of, and to work with, document, forms, and perhaps data. In some cases, they have made the transition to electronic representation of documents, forms, and data. However, few, if any would feel comfortable with the thought that they work with ‘objects’ (much less that they themselves are ‘objects’). Thus, if a customer-focused approach is to be applied, it would be appropriate for systems developers to think in terms of documents, rather than objects or perhaps even data. ...Forms are a particular kind of document used to gather highly structured information (i.e., ‘data’) in a highly structured way.”

Drawing upon the work of other experts, Ambur points out that “detailed data elements can be grouped into classes and that data classes are the underpinnings of business processes. Conceptually, at a high level of technical abstraction but in terms readily understood by the average person, all types of information can be circumscribed by only three categories:

- Informal, unstructured
- Formal, relatively unstructured
- Formal, highly structured

Different means are applicable to the process of information in each category, i.e., different applications software is best suited to each. Informal, unstructured information is most appropriately processed and shared via E-mail, voice mail, and real-time voice communications, including one-on-one conversations and group meeting. Formal, highly structured information – commonly called data – is most appropriately gathered and shared via forms. ...For highly

structured data, the single application needed is electronic forms automation (E-forms). ... The forms automation approach can also help to instill in users, in a non-confrontational way, the necessary discipline and incentive to participate in establishing and maintaining enterprisewide data dictionaries. Each time the need for the collection of highly structured data occurs, the E-forms system would be available to facilitate the effort, by and for anyone in the organization.”

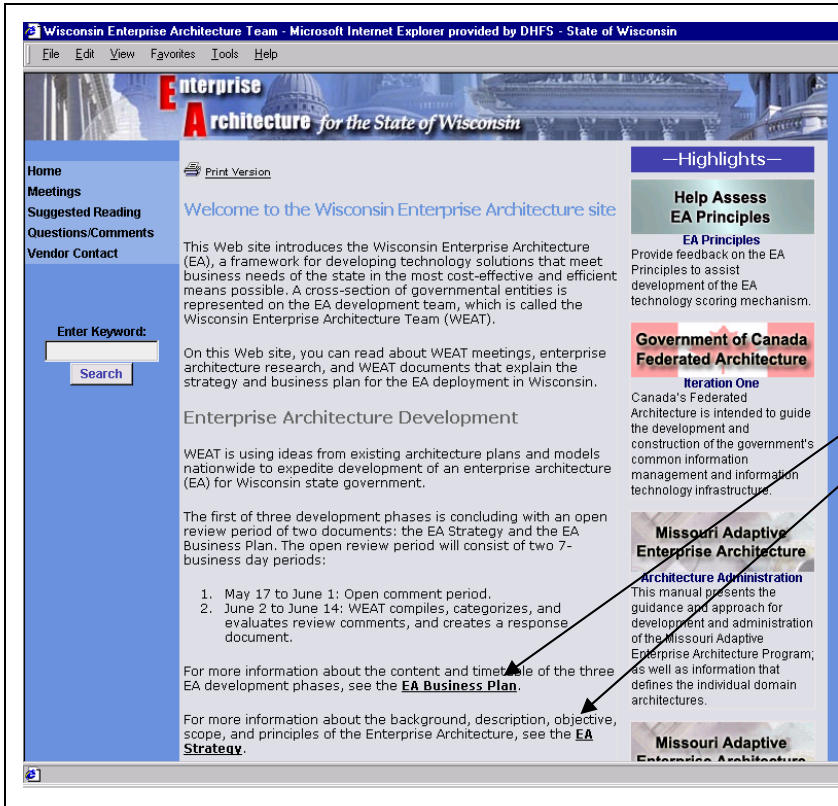
In his conclusion, Ambur writes, “The answer to these questions is obvious and so too is the need for a new approach to IT systems development. This is a better alternative to the traditional database-focused Systems Development Life Cycle (SDLC) approach, which so inevitably leads to stovepipe systems. There is an approach that is far more likely to lead to a rational enterprise-wide information management solution. That approach is to deliver to the desktop of the average user the basic set of *applications* they need to process all three categories of information – informal/unstructured, formal/relatively unstructured, and formal/highly structured.”

Microsoft Office 2003 will bring XML-enabled applications to the desktop of all (average and advanced) users that will give them the power to process all three categories of information. If DHFS (as a department) were to attach XML tags to data elements on all forms and match the elements (as appropriate) with common core data definitions, the result would be a comprehensive dictionary of current common core data and a listing of named data elements to be defined in subsequent steps. The chunking principle that Ambur champions, is that projects should: “Be implemented in phased, successive chunks as narrow in scope and brief in duration as practicable, each of which solves a specific part of an overall mission problem and delivers a measurable net benefit independent of future chunks...” “*Forms-centric*” is a term that describes an architecture where the primary “chunk” of an enterprise-wide information management solution is based upon naming and defining data elements used on forms. XML provides the syntax that makes such an endeavor feasible.

**Question: Would a DHFS "forms-centric" architecture conflict with the emerging DOA Enterprise Architecture?**

Indications are that a "forms-centric" architecture as described in the DMT 25 Use Case scenario, not only would be consistent with the emerging Wisconsin Enterprise Architecture (EA) that is underdevelopment, but would provide the State with a working model of a number of core principles that were enunciated in the May 12, 2004 draft *EA Strategy* document, developed by

the Wisconsin Enterprise Architecture Team (WEAT).



WEAT has divided Enterprise Architecture (EA) development into three phases. The first has concluded with an open review period for the EA Business Plan and the EA Strategy documents. A response document to comments is currently under development.

According to the business plan, each phase is anticipated to take approximately 90 business days for completion.

Phase 1 - Which is now concluding, was to establish a strategy for implementing an Enterprise Architecture (EA) within the extended enterprise. The major deliverable was the creation of an EA strategy for the extended enterprise which included a conceptual architecture that identifies a set of core principles which are the foundation of the enterprise architecture with the State of Wisconsin. (Approximate end date - June 4, 2004)

Phase 2 - The objectives of this phase are to develop processes for evaluation and selecting technology solutions to support an EA Life Cycle. The Wisconsin Enterprise Architecture Team (WEAT) has selected a balanced scorecard methodology as a quantitative approach to the selection of technologies within the extended enterprise. (Approximate end date - Sept 24, 2004)

Phase 3 - The objectives of this phase include (among others) establishment of a framework and model for the development of EA reference models and beginning development of the reference models. (Approximate end date - Jan 27, 2005)

WEAT says that the development of the EA reference models will be an on-going, top down, effort that, when completed, will define the State's business and technical environments. The team plans to develop the following reference models:

1. Business Reference Model (BRM)
2. Technical Reference Model (TRM)
3. Service Component Reference Model (SRM)

While all models are relevant to the "forms-centric" approach, it is the Data and Information Reference Model (DRM) that is most relevant. The description is a direct quote from the EA Business Plan:

4. **Data and Information Reference Model (DRM)**

This model will describe, at an aggregate level, the data and information that support program and business line operations. The model will aid in describing the types of interaction and exchanges that occur between the State of Wisconsin Government and its various customers, constituencies, and business partners. **The DRM will categorize the government's information along general content areas and decompose those content areas into greater levels of detail.** The DRM establishes a commonly understood classification for state data and leads to the identification of duplicative data resources. A common data model will streamline the processes associated with information exchange both within the state government between state agencies and its external stakeholders.

5. Performance Reference Model (PRM)

The Federal Government has been working on a Federal Enterprise Architecture (FEA) having similar reference models. The Federal DRM is still under development but will be built on principles that include, among others:

- Will heavily leverage XML and interoperability principles
- Classifications of data will form the basis for the definition of business driven XML Schemas
- Will leverage industry vocabularies
- XML Schemas will be stored within a central repository

The screenshot shows a web page with a navigation menu on the left and a main content area. The main content area features a diagram titled "Business Areas / Functions" with a "Conceptual" label. The diagram consists of four vertical bars representing business areas: Social Services, Consumer Safety, Public Health, and Trade Import / Export. Horizontal bars represent functions: Food / Merchandise Inspection, Benefits, Tariffs, Quotas, Immunizations, Vaccinations, Disease Tracking / Monitoring, and News, Events. A list of principles is on the left, and a callout box is at the bottom left.

2003 Conference Proceedings: Egov OpenSource and SecurE-biz Executive Summit [Conference Topic Map](#)

2003 SecurE-Biz Executive Summit [First page](#) [Back](#) [Continue](#) [Last page](#) [Overview](#)

**The draft FEA Data Reference Model (DRM) is envisioned to classification of data across horizontal and vertical business**

Browse by Topic Category: [Keyword](#) [Author](#) [Presentation](#) [Organization](#) [City](#) [State](#) [Country](#)

Browse: [XTM Format](#)

**Business Areas / Functions**

- Will heavily leverage XML and interoperability principles
- Classifications of data will form the basis for the definition of business-driven XML Schemas
- Will leverage industry vocabularies
- XML Schemas will be stored within a central repository (e.g., XML.Gov, FEAMS)
- Security and data privacy are TOP priorities, records management
- State and local government collaboration is essential

*Conceptual*

Food / Merchandise Inspection

Benefits, Tariffs, Quotas

Immunizations, Vaccinations

Disease Tracking / Monitoring

News, Events

Social Services Consumer Safety Public Health Trade Import / Export

Internet

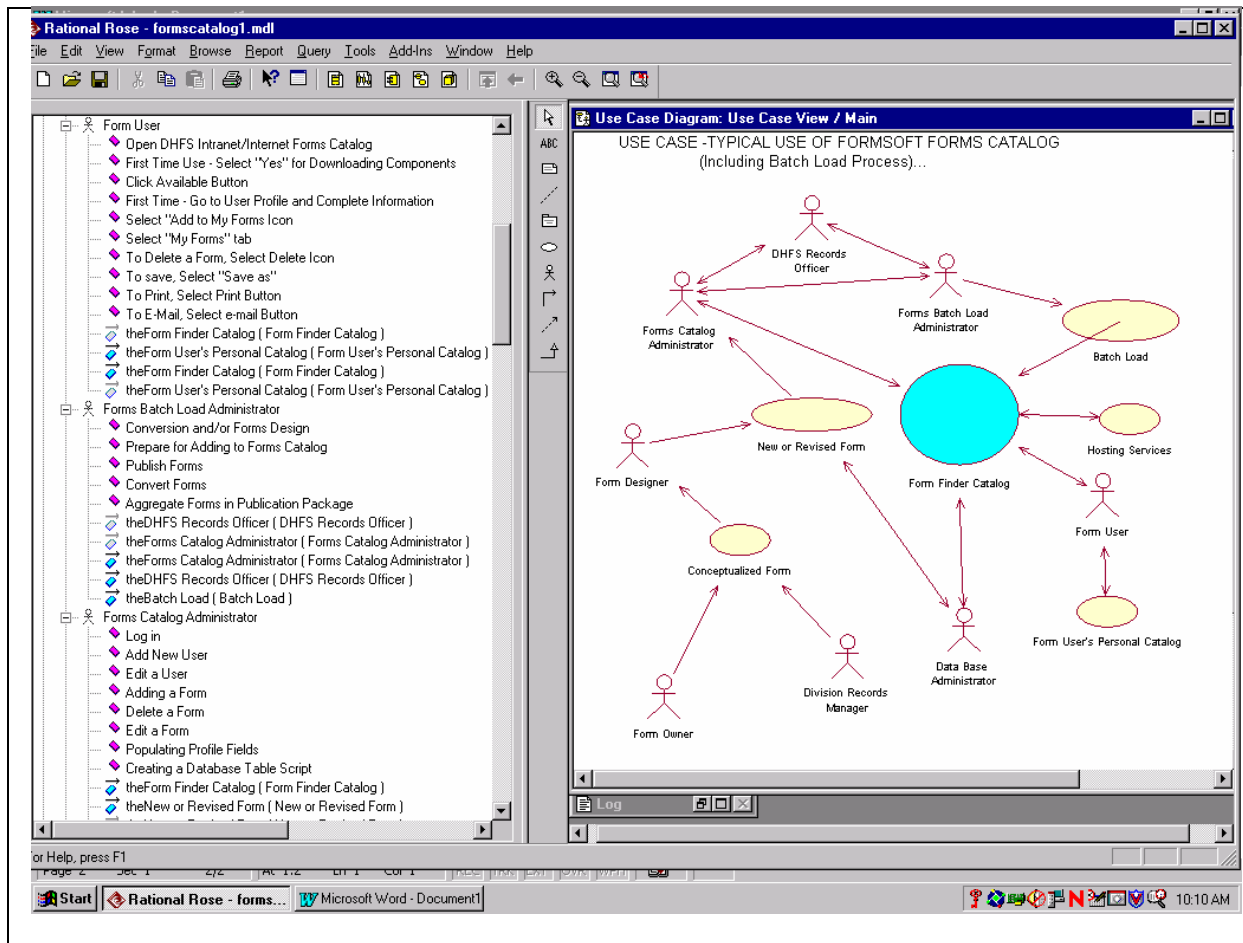
For the Federal Government, conceptual planning documents break out examples of Business Areas / Functions as:

- Social Services
- Consumer Safety
- Public Health
- Trade Import / Export

Assuming the Wisconsin Enterprise Architectural Team (WEAT) follows a similar conceptual framework to identify business areas and functions, it is likely that classifications similar to *Social Services* and *Public Health* will also be identified. Because WEAT will be following standards and principles using a top down approach that are similar to those that DHFS followed

in a bottom up approach in defining Common Core Data Standards (which would anchor the forms centric approach) the architectures would likely not only be consistent but actually would be views of the same architecture but from different levels. Decomposition of government's information along general content areas into greater levels of detail (at least in theory) should match up with composition of details into general content areas of government's information (as it pertains to the Department of Health and Family Services).

My first assignment as E-forms Coordinator was to describe an E-forms architecture. The following graphic is a screen shot of a use case depicting the FormFinder Catalog as the electronic forms repository. Conceptually, the graphic depicts a typical use case for any forms repository such as Microsoft's SharePoint forms library and the roles that are played by the various actors. Essentially, the depicted architecture a bottom-up approach using a process of composition of details.



DHFS already has a number of other detailed “objects” or “chunks” that have been developed according to standards. The DHFS Common Core Data Standards were developed consistent with ISO 11179 – Standardization of Data Elements and used a process consistent with ANSI X 3.285 – Metamodel for the Management of Shared Data wherein the metamodel regions are identified as:

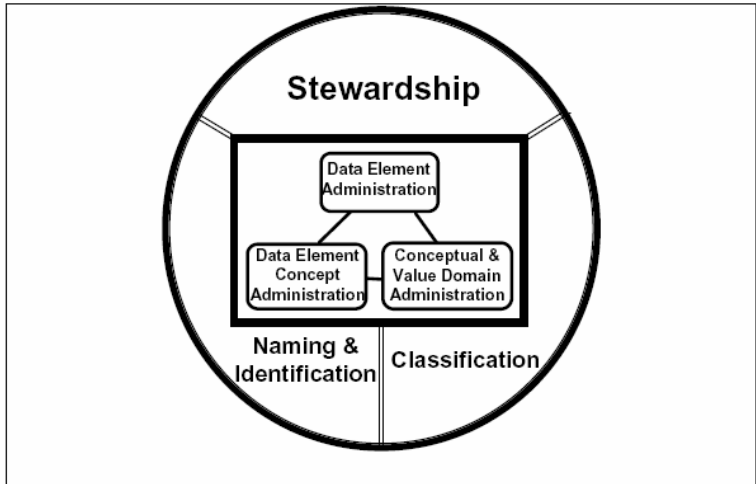


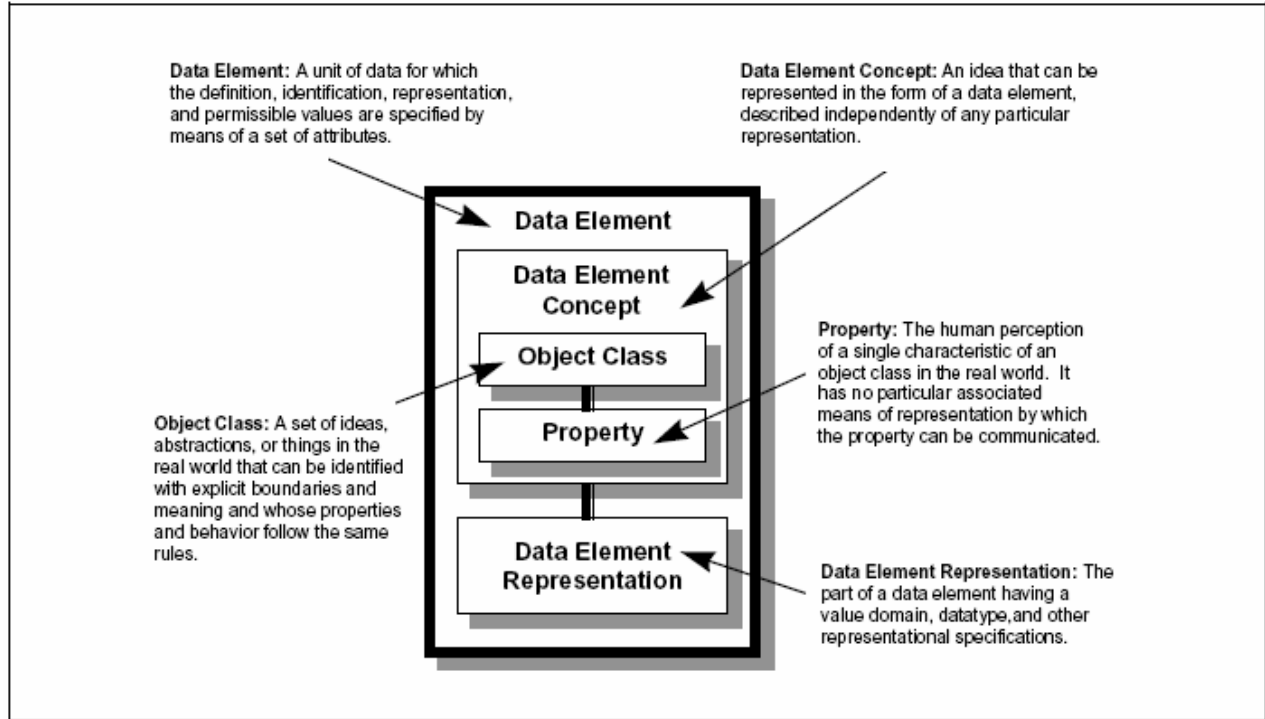
Figure 2 – Metamodel regions

- Stewardship
- Naming and Identification
- Classification

The DHFS Data Stewardship Committee was the principle vehicle for developing the *DHFS Common Core Data Standards*.

As has been mentioned earlier in this paper, the beta version MetaPro, a distributable metadata

registry built with Access 97 was used to register all data elements and is proposed to be extended to include XML tags associated with the data element names and definitions. The graphic below provides an overview of the more important components of the data registry, each of which is described as an object.



The data management principles included in the EA Strategy draft document are similar to those that guided development of the Common Core Data Standards under sponsorship of the DHFS Data Stewardship Committee. Further, designation of CUSP as an IT laboratory site would be in keeping with the principles identified under Center of Excellence Principles.



### **Data Management Principles**

1. Each individual data item has a single steward or authoritative source, clearly defined locations, and is accessible. Authoritative data must be accessible and available for reuse by any entitled systems or business processes
2. Data stored in information repositories within the extended enterprise should be widely available and accessible by all entities within the state extended enterprise, by federal agencies, and by other appropriate partners and entities.
3. Data is an asset that must be managed for the benefit of the extended enterprise. Data must be shared to the maximum degree possible, without jeopardizing security and confidentiality.
4. Data is collected, protected, and maintained in accordance with appropriate standards and guidelines.
5. Records in electronic format must be preserved and maintained, and remain accessible for their designated retention period.

Also included in the EA Strategy document is a conceptual description of Centers of Excellence:

### **Center of Excellence Principles**

1. The EA will promote organizational diversity and virtual organizations such as centers of excellence
2. Facilitate change, encourage IT research and development, and promote integrating new technologies within the extended enterprise.

Conceptually, extending the notion of a Forms / Publications Center to hosting electronic as well as housing physical (paper based) forms might fit the idea of a virtual organization.

The first (essential) step of developing an XML tagged vocabulary associated with common core data elements and their definitions could be accomplished, before Office 2003 is deployed, under sponsorship of the DHFS Data Stewardship Council.

**Question: What would be the steps in developing an XML based “forms-centric” information architecture?**

Step 1. Develop an XML tagged vocabulary

Step 2. Apply the XML vocabulary to all forms

Step 3. Use a “forms design” tool to reverse engineer and construct XML schemas for each form

Step 4. Analyze common syntactical patterns of the schemas

Step 5. Decompose and classify common patterns of the schemas as objects

Step 6. Develop a DHFS ontology and object library of form objects and schema parts