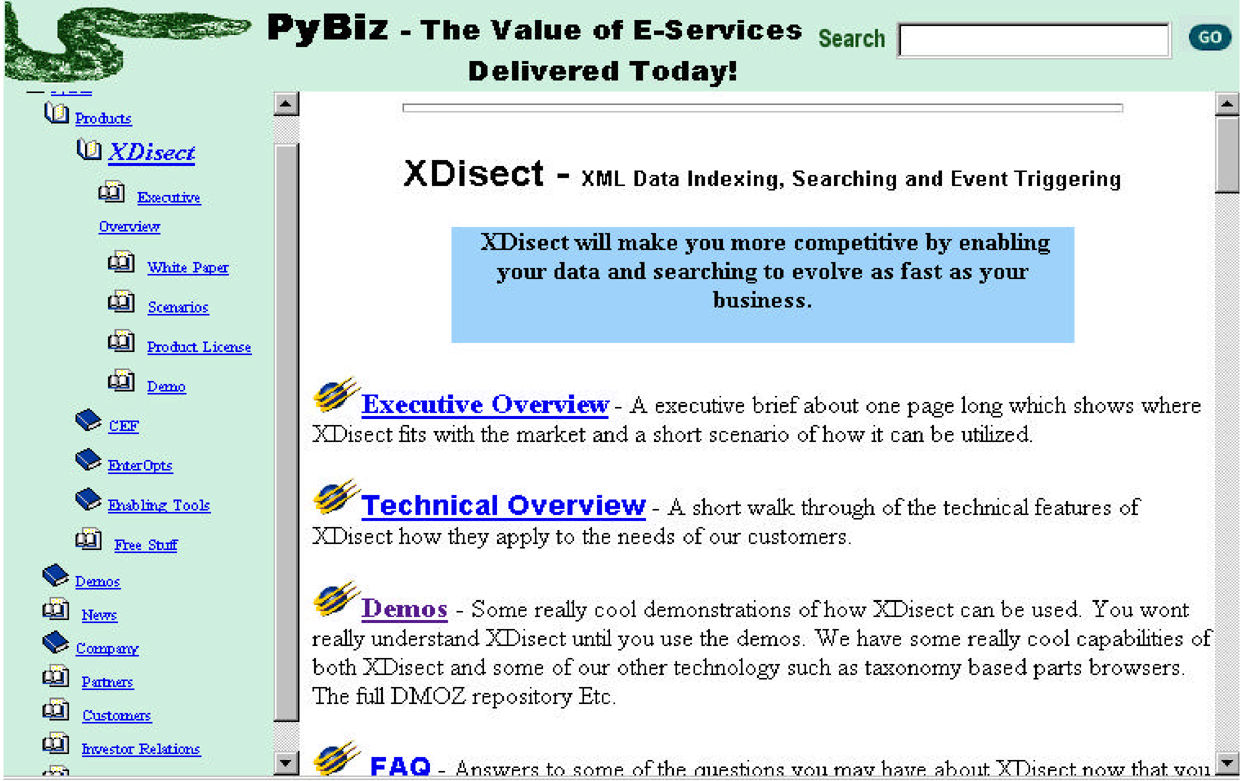
EContentMgr (Content Management & Web Publishing)

By Joe Ellsworth. © PyBiz, Inc April-2000
See the Last Page for Table of Contents.

EContentMgr turns web site navigation and content management into a natural real-time experience. Because it directly reads your normal document repository, your documents appear on your site as soon as you commit them to the repository. Since EContentMgr automatically updates the web site, you are able to see the site grow to encompass new content as fast as it can be produced and approved for release.

Specialized site creation tools have a high overhead in terms of time. Time spent buying, installing, and learning software; time spent managing the navigation of the site, uploading new versions of the site and planning new releases of the content. EContentMgr turns this into a low-cost no stress process while automatically providing features that companies normally spend many thousands of dollars to obtain.

You can think of EContentMgr as having the ability to create an automatic N-depth table of contents that is browsable, hyper-linked, searchable and expandable for nice presentation to end-users. EContentMgr allows a web site to be built directly from the data provided in a directory structure or version control repository. **Below is an image of a site that could be generated using EContentMgr.**



XDisect - XML Data Indexing, Searching and Event Triggering

XDisect will make you more competitive by enabling your data and searching to evolve as fast as your business.

Executive Overview - A executive brief about one page long which shows where XDisect fits with the market and a short scenario of how it can be utilized.

Technical Overview - A short walk through of the technical features of XDisect how they apply to the needs of our customers.

Demos - Some really cool demonstrations of how XDisect can be used. You wont really understand XDisect until you use the demos. We have some really cool capabilities of both XDisect and some of our other technology such as taxonomy based parts browsers. The full DMOZ repository Etc.

FAQ - Answers to some of the questions you may have about XDisect now that you

Our Scenario

Imagine you have a company web site where you are publishing your list of open jobs. You get a new job opening and the respective manager is able to use his word processor to publish a new description for the job. A small category definition is made for this new job file saying that the listing should appear on the web site under technical jobs but it also should appear in a couple of other places; under the picture of the manager's team and near the description of the project the new employee would be working on.

What if the non-technical manager could publish the new job opening on the web site in the proper places in less than 10 minutes without being required to go find the web design company and without having to post an entirely new version of the site?

Now imagine that this problem is not just job-listing files but technical documents, white papers, sales documents, and many others. Even with hundreds of people changing their content, the site continues to be robust and up-to-date with current content within minutes.

And here is the kicker! Imagine that the navigation to every section looks the same and that the site has a consistently applied style unless somebody explicitly decides on a new artistic style for a given page. The individual documents such as the job offer may vary, but until then everything has the same clear consistent style that is unique for your company.

What if the system that enabled all of this was absolutely free? What if the individual contributors didn't have to install any new software to make it work? What if they were able to use their normal file manger version control system or file share and they used all the normal tools with which they were already familiar? What if the only required configuration was a very simple one and establishing the templates for the company look-and-feel only needed to be built once and were very easy to edit? Wouldn’t that be cool?

Unbelievably, this even works for individual groups in large Fortune 100 companies. Large companies probably don't want to allow individual employees to affect the top-level company site pages for which they pay web artists tens of thousands of dollars. These same companies will benefit from allowing this at the departmental level and quite often do allow it. But generally each department is doing it in their own way and adding their own desired artistry to their portion of the web site.

In today's average company this scenario would have involved three meetings to decide where the content should appear and then three or four people would have worked on it for a combined time of 1/2 - 3 engineering-days. The manager would be lucky if the job opening made it to the site in less than 3 days. Our experience has been that this can require months. Our company cannot afford this and feel that there are a large number of other companies that cannot afford it either.

✍✍ *TODO: Add High Level Data Flow Picture Here Showing The Scenario*

✍✍EContentMgr makes publishing new and changed document a natural process, minimizing the work required to maintain and update web sites over time. What business wouldn’t want a lot of additional value for very little investment?

Value Proposition & Basic Differentiation.

✍✍ Makes the publishing of a content rich always up to date site a natural result of the teams existing day-to-day activities with less effort than currently invested to accomplish less.

✍✍ Provides maximum value with minimal changes to existing processes.

✍✍ Start at no or very low start cost.

?? Very low start up overhead.

?? Install in 30 minutes and start publishing.

?? Minimal learning needed to get started.

?? The Technology / product is free.

?? Commercial support, training and customization are available.

?? Basic GUI customization is as simple as some bit map images.

?? Expand to make more personalized as it makes business sense.

?? Expand to use more advanced features as it makes business sense.

✍✍ Allows the use of the teams existing version control processes and technology.

✍✍ Allows the site to evolve new navigation dynamically as content evolves.

✍✍ Allows a broader cross section of the team to specify where they want their content to appear on the site.

✍✍ Facilitates sites that are up to date with the content as soon as it changes.

✍✍ Facilitates consistent visual navigation with minimal restrictions.

✍✍ Supports participation by very non-technical users.

✍✍ Lots of very advanced capabilities which don't cost you any effort up front and are easy to grow in to as your needs evolve.

?? Personalized filtering of content displayed.

?? Extensible meta data for all types of documents.

?? Automatic notification of changes in areas of interest.

Goal

EContentMgr will build a dynamic web site directly from content that is already contained in the existing version control systems. It will allow sufficient control over the navigation of the site while not requiring centralized planning. It will copy content that has been released directly to the web site and make that content visible in the navigation structure of the web site without necessitating meetings or other overhead.

EContentMgr is fully html template driven so while the core business and publishing logic remains the same the entire look and feel of the site can be customized to end users / company desired view. At a minimum our users want to customize their site with their own logo bar and in some instances they customize the entire site to the point where we won't even recognize it.

EContentMgr maximizes the use of existing knowledge and processes for managing content, especially for software development teams. It adds the absolute minimum of technology and processes that are needed to allow the team to build and maintain a web site that is professional and always up-to-date.

EContentMgr is built on the premise that empowered people should be able to make new content visible on flexible web sites automatically with an absolute minimum of fuss. They should be able to use their normal tools and version control techniques for managing the files on their own computer systems.

EContentMgr is built on the belief that documentation for software should stay as close to the software as reasonably possible so that it is closely associated as a bundle that is managed together. This creates a problem of extracting the documentation from the source code and publishing it on the web site. EContentMgr makes this not only possible but also cost effective.

EContentMgr works on the premise that people already have version control systems like CVS that are more than adequate for storing and controlling access to content files with read, write and create privileges. Furthermore, with the very strong and powerful version control systems already available it would be a folly to introduce new technology products just to version control web site content.

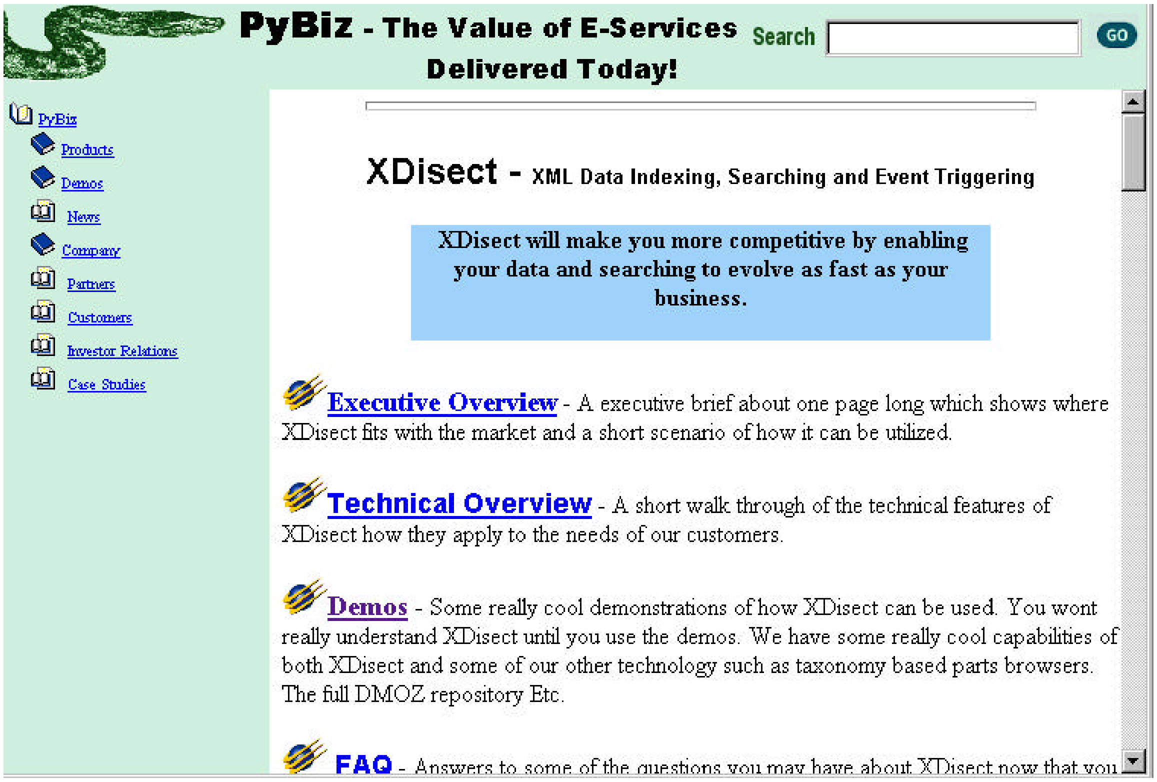
EContentMgr also provides some advanced capabilities such as:

- Allowing a user to register for notification whenever a document they have accessed has changed or when new documents within specific categories have been added.
- Allowing the site to filter content and branches based on a users privileges and preferences.
- The ability to add external content to the site using nothing but a web browser and the ability to place that content at multiple locations in the site taxonomy.
- The ability to add complex meta data for each document in the site and then use that meta data both for event delivery and for filtering the users view of the data.
- The ability for users to create local book marks of content they have viewed and organize it in their own taxonomy structures.

A Visual Overview of the Product

A view of a EContentMgr generated menu page.

This is our default view of course this can be customized to contain any visual format desired by our customers. At the very least we would expect our customers to replace the top banner with their own.



XDisect - XML Data Indexing, Searching and Event Triggering

XDisect will make you more competitive by enabling your data and searching to evolve as fast as your business.

Executive Overview - A executive brief about one page long which shows where XDisect fits with the market and a short scenario of how it can be utilized.

Technical Overview - A short walk through of the technical features of XDisect how they apply to the needs of our customers.

Demos - Some really cool demonstrations of how XDisect can be used. You wont really understand XDisect until you use the demos. We have some really cool capabilities of both XDisect and some of our other technology such as taxonomy based parts browsers. The full DMOZ repository Etc.

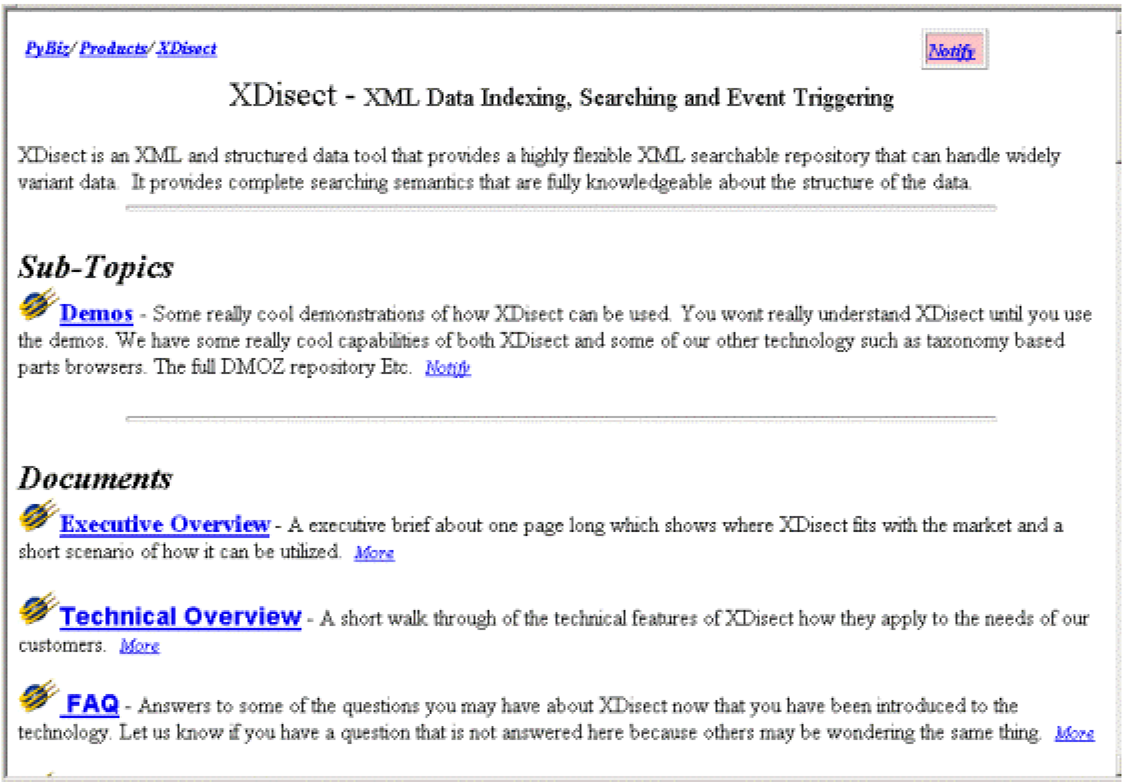
FAQ - Answers to some of the questions you may have about XDisect now that you

Things to notice on this Page:

1) The taxonomy is always available on the left.
2) A title and description for the current node appears at the top on right.
3) The documents and the nodes appear on the right.
4) Each document and sub category has a title which is the hyperlink and a long description.
5) The search window is always available on the top right and always searches inside of the currently active document nodes.
6) It runs just find in standard browsers such as Netscape 4.5+ and IE 5+.
7) It can handle an unlimited number of documents and nodes in the taxonomy tree without performance degradation.

Topic View Screen

This is the screen that will be rendered whenever you click on a specific top for a specific windows.



PyBiz/ Products/ XDisect

XDisect - XML Data Indexing, Searching and Event Triggering

XDisect is an XML and structured data tool that provides a highly flexible XML searchable repository that can handle widely variant data. It provides complete searching semantics that are fully knowledgeable about the structure of the data.

Sub-Topics

Demos - Some really cool demonstrations of how XDisect can be used. You wont really understand XDisect until you use the demos. We have some really cool capabilities of both XDisect and some of our other technology such as taxonomy based parts browsers. The full DMOZ repository Etc. *Notify*

Documents

Executive Overview - A executive brief about one page long which shows where XDisect fits with the market and a short scenario of how it can be utilized. *More*

Technical Overview - A short walk through of the technical features of XDisect how they apply to the needs of our customers. *More*

FAQ - Answers to some of the questions you may have about XDisect now that you have been introduced to the technology. Let us know if you have a question that is not answered here because others may be wondering the same thing. *More*

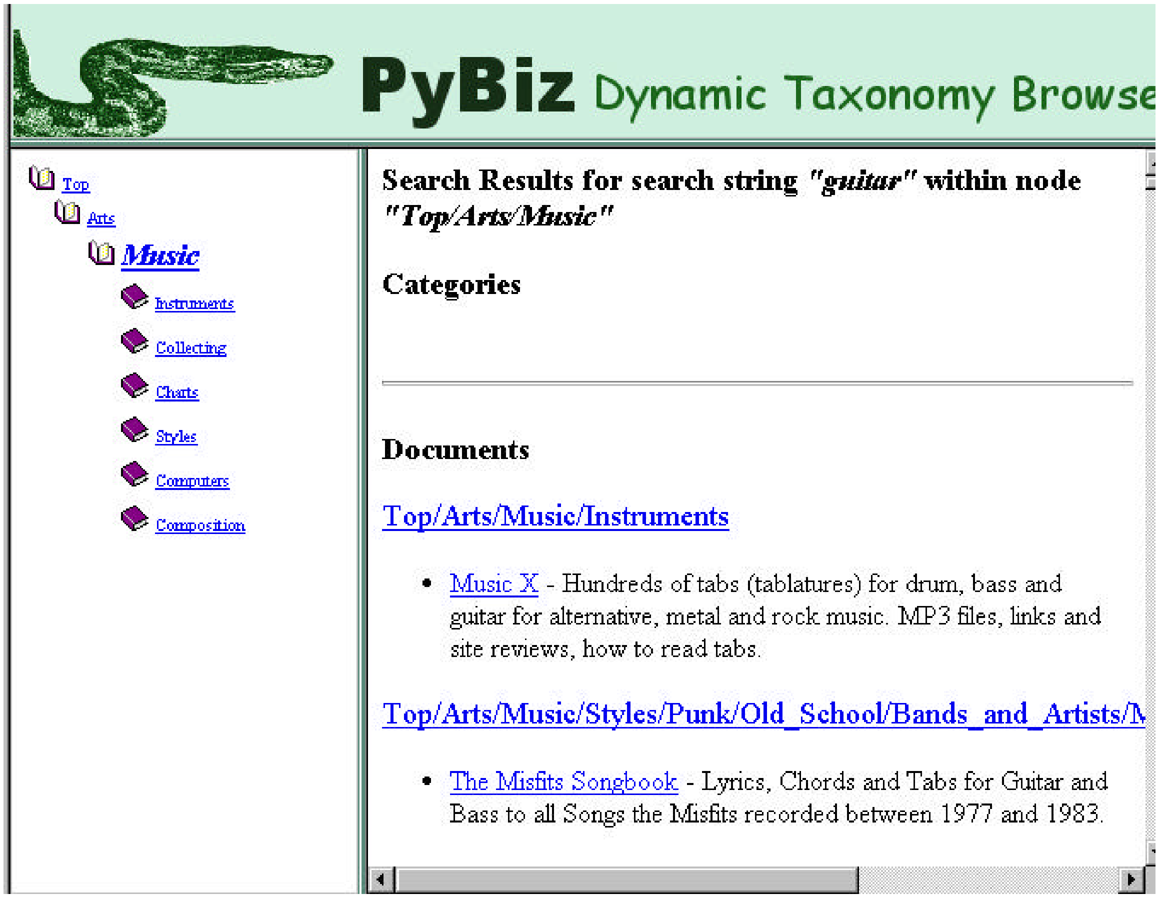
- ***TODO: Show list of people registered (Optional) based on site policy.***
- ***TODO: Remove the notify link for sub topics. If people want to get notification for a sub topic they can first drill down in to that topic and then click on the notify there.***

Things to Notice:

- Clicking on the topics at the top left will take you to the parent topic.
- Clicking on Notify will take you to the topic change registration screen where you can register to receive notifications documents which change in this topic.
- Clicking on one of the listed sub topics such as demos will cause you to display to topics page for that sub-topic. The navigation tree on the left (frames only) will also be repositioned to make that topic the active.
- Clicking on the [more] link for any of the documents will display a document summary screen where you can register for notification of that document, view the history of the document, etc.

- Clicking on the document title for any document will cause that document to be displayed.

Search Result Screen



Search Results for search string *"guitar"* within node *"Top/Arts/Music"*

Categories

Documents

Top/Arts/Music/Instruments

- Music X - Hundreds of tabs (tablatures) for drum, bass and guitar for alternative, metal and rock music. MP3 files, links and site reviews, how to read tabs.

Top/Arts/Music/Styles/Punk/Old_School/Bands_and_Artists/M

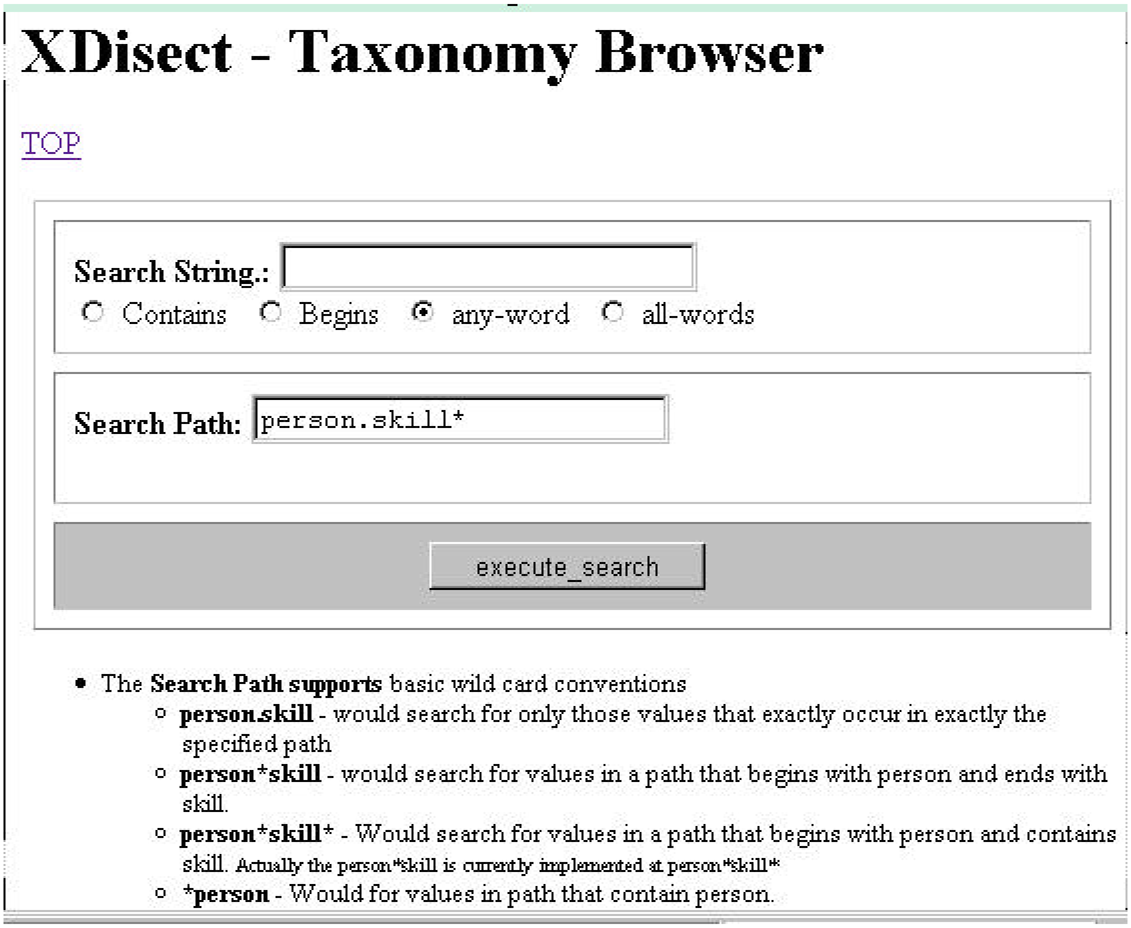
- The Misfits Songbook - Lyrics, Chords and Tabs for Guitar and Bass to all Songs the Misfits recorded between 1977 and 1983.

Things to Notice.:

- ✍✍ The search was conducted on the DMOZ data in music category on the search string "guitar".
- ✍✍ Categories located if any are listed first and then the documents are included organized by category.
- ✍✍ Clicking on the document will spawn a new window showing the document selected.
- ✍✍ Clicking on the category where the document is located will reposition the tree on the left to that category and will
- ✍✍ The document title and descriptive text is show for each document shown.

✍✍ Everything on this page can be customize to meet our customer's expectations.

Advanced Search Screen



XDisect - Taxonomy Browser

TOP

Search String.:

○ Contains ○ Begins ● any-word ○ all-words

Search Path: person.skill*

- The **Search Path supports** basic wild card conventions
  - **person.skill** - would search for only those values that exactly occur in exactly the specified path
  - **person*skill** - would search for values in a path that begins with person and ends with skill.
  - **person*skill*** - Would search for values in a path that begins with person and contains skill. Actually the person*skill is currently implemented at person*skill*
  - ***person** - Would for values in path that contain person.

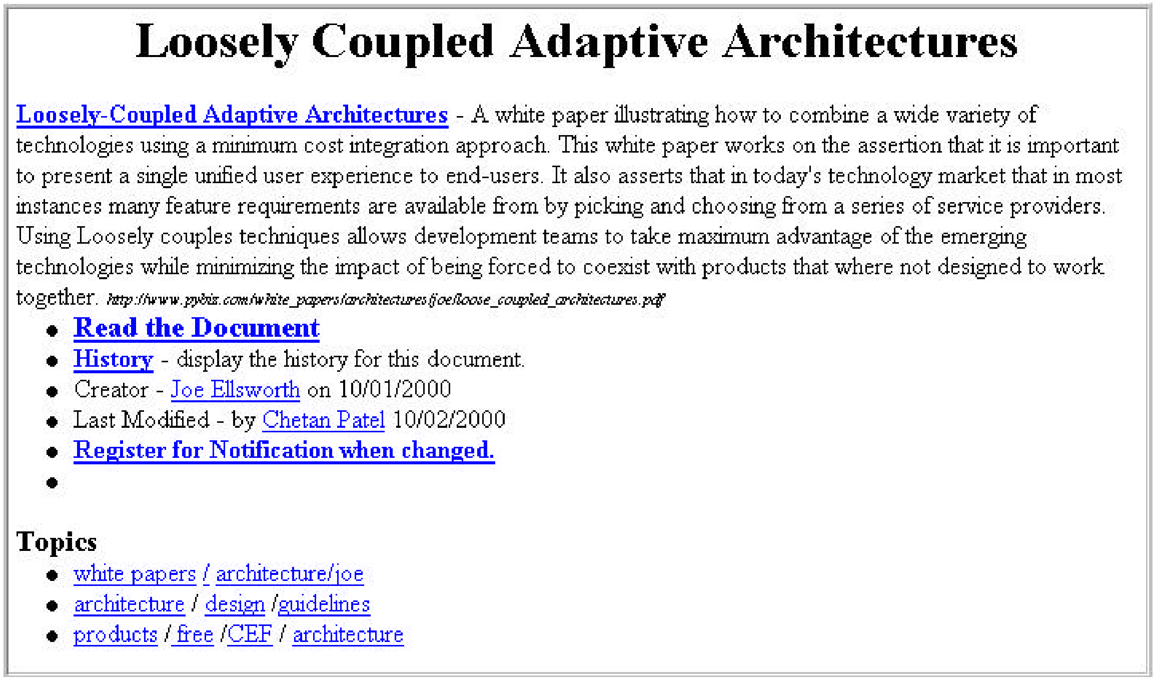
Things to Notice:

- ✍✍ There are two screens the advanced and the very advanced.
- ✍✍ The meta data field name is fully customizable and can handle almost an infinite variety of meta data.
- ✍✍ The string being searched for is quite flexible.
- ✍✍ The meta wild cards for the field names allows very interesting searches to be conducted.

✍✍ ***TODO: Add Second screen to show the standard Meta Data driven search document for Given documents. E.G. Subject, Title, Author, Etc. Show it with the standard includes, words, all words, begins for each of these as a more basic interface.***

Document Summary View

The document summary view is used to provide a central jumping off navigation point for any single document. It is the central action point where notifications for a document can be requested. This screen is generally reached from the topic view and search results windows by clicking on the [more] link for any document.



Loosely Coupled Adaptive Architectures

Loosely-Coupled Adaptive Architectures - A white paper illustrating how to combine a wide variety of technologies using a minimum cost integration approach. This white paper works on the assertion that it is important to present a single unified user experience to end-users. It also asserts that in today's technology market that in most instances many feature requirements are available from by picking and choosing from a series of service providers. Using Loosely couples techniques allows development teams to take maximum advantage of the emerging technologies while minimizing the impact of being forced to coexist with products that where not designed to work together. *http://www.pybiz.com/white_papers/architectures/joe/loose_coupled_architectures.pdf*

- **Read the Document**
- **History** - display the history for this document.
- Creator - Joe Ellsworth on 10/01/2000
- Last Modified - by Chetan Patel 10/02/2000
- **Register for Notification when changed.**
-

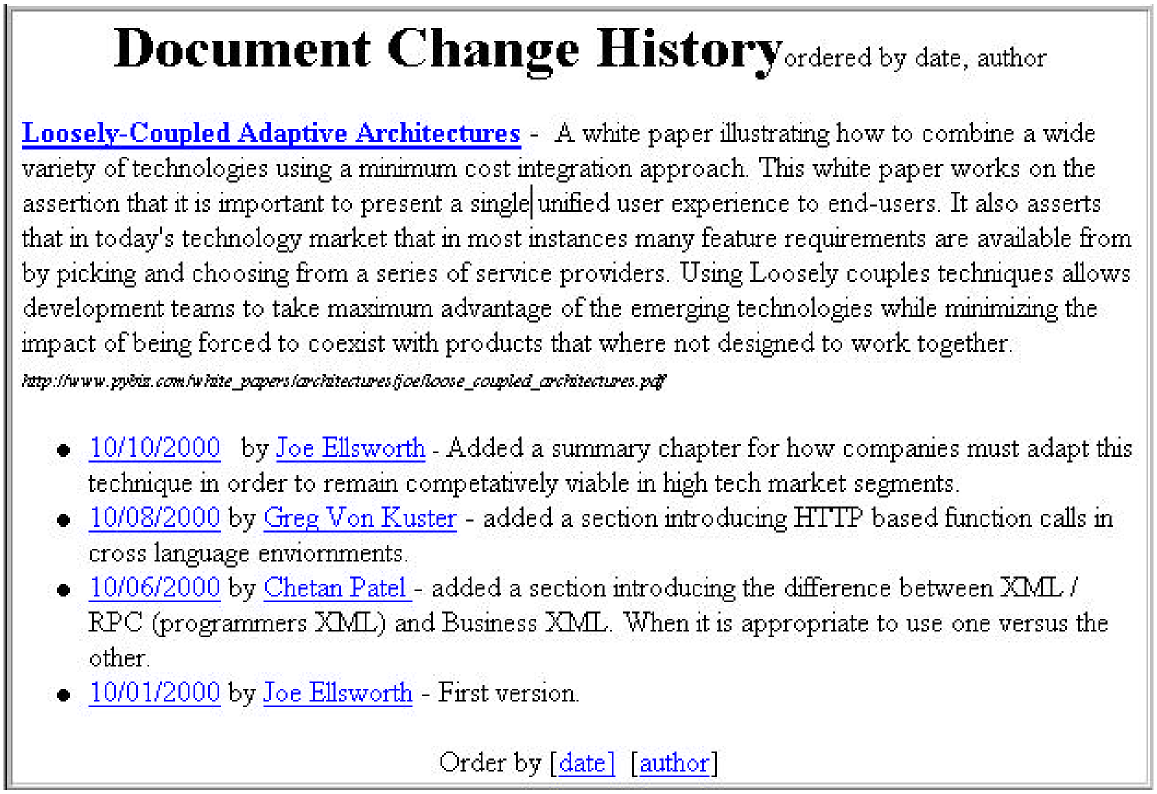
Topics

- white papers / architecture/joe
- architecture / design /guidelines
- products / free /CEF / architecture

Things to Notice:

- ✍✍ Clicking on the document title or [read then document] links will display the actual document.
- ✍✍ Clicking on the history link will display the document history view for that document.
- ✍✍ Clicking on either the creator or last modified people links will display the authors page.
- ✍✍

Document History View



Document Change History ordered by date, author

Loosely-Coupled Adaptive Architectures - A white paper illustrating how to combine a wide variety of technologies using a minimum cost integration approach. This white paper works on the assertion that it is important to present a single unified user experience to end-users. It also asserts that in today's technology market that in most instances many feature requirements are available from by picking and choosing from a series of service providers. Using Loosely couples techniques allows development teams to take maximum advantage of the emerging technologies while minimizing the impact of being forced to coexist with products that where not designed to work together.

http://www.pybiz.com/white_papers/architectures/joe/loose_coupled_architectures.pdf

- 10/10/2000 by Joe Ellsworth - Added a summary chapter for how companies must adapt this technique in order to remain competatively viable in high tech market segments.
- 10/08/2000 by Greg Von Kuster - added a section introducing HTTP based function calls in cross language enviornments.
- 10/06/2000 by Chetan Patel - added a section introducing the difference between XML / RPC (programmers XML) and Business XML. When it is appropriate to use one versus the other.
- 10/01/2000 by Joe Ellsworth - First version.

Order by [date] [author]

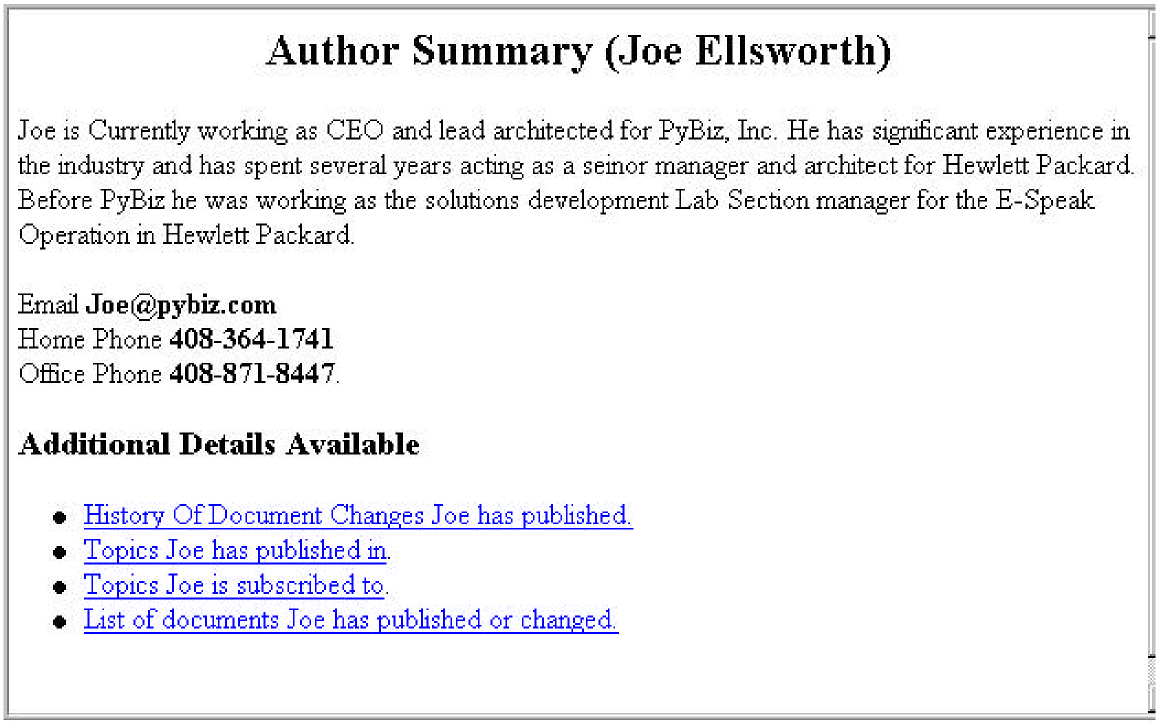
Things to Notice:

- Clicking on the document title will retrieve the document for viewing.
- Clicking on the one of the dates in the history will show all changes for all documents occurring on that day sorted by document Title. The display will be centered on the changes for this document for that day.
- Clicking on the author for any change will display all of the changes that author has made for that document. Or It will display the author summary screen.

Note: Only the history items that have historical log entries prefixed with #* will be included in the history log file.

Author / User Summary Screen

The author summary screen shows all of the publicly visible information for an author. An author is defined as a person who has either published new documents or posted changes to existing documents. It provides primary jump off navigation options to find additional details about the person in the context of this system.



Author Summary (Joe Ellsworth)

Joe is Currently working as CEO and lead architected for PyBiz, Inc. He has significant experience in the industry and has spent several years acting as a seinor manager and architect for Hewlett Packard. Before PyBiz he was working as the solutions development Lab Section manager for the E-Speak Operation in Hewlett Packard.

Email **Joe@pybiz.com**
Home Phone **408-364-1741**
Office Phone **408-871-8447**.

Additional Details Available

- History Of Document Changes Joe has published.
- Topics Joe has published in.
- Topics Joe is subscribed to.
- List of documents Joe has published or changed.

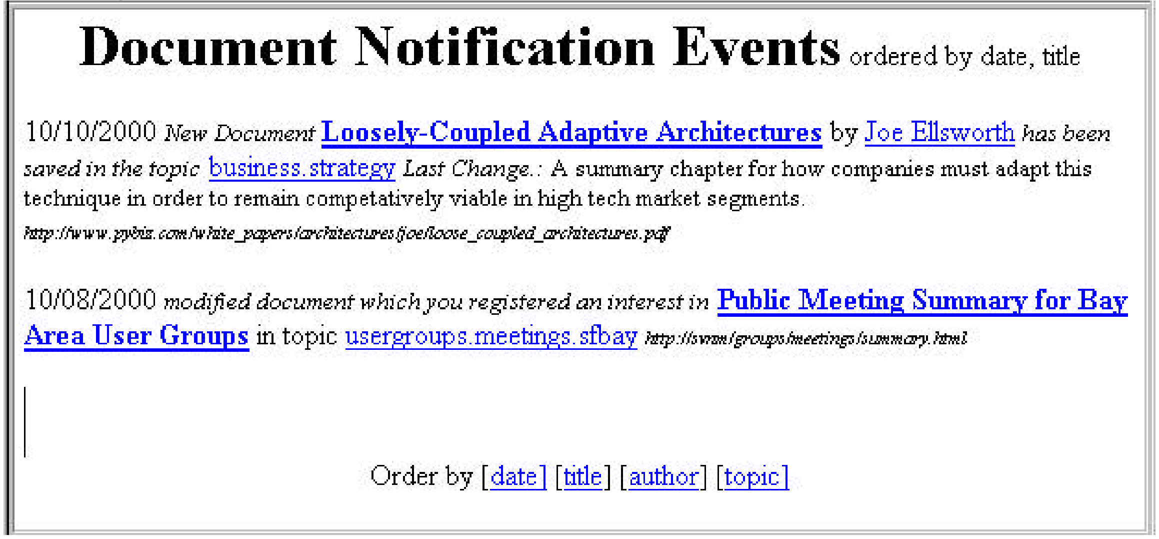
Things to Notice:

- The history of document changes will list all the documents which the author has changed. They will be listed in reverse chronological order first although there is an option to list sorted by title and by title.
- The topics the author has published to will be the total list of topics and sub topics that contain a document which the author has created or changed.
- The topics the author is subscribed to is the list of topics and sub topics which the author will receive notifications on.
- The list of documents the author has published contains links to each document the author has created or modified.

> - *TODO: Need additional detail for the detailed attributes and other related records for this person. This may be an opportunity for a taxonomy browser / query which brings up everything that has this persons id as a reference item.*

Document Event List View

This is a screen showing the list of changes to documents in the system. Only those changes for documents and topics that you have registered an interest in will be shown. Each time you download a document you may be given a chance to register for notification of changes to that document. If the author has included a history item for the last change then the detail from the last change will be shown as well.



Document Notification Events ordered by date, title

10/10/2000 *New Document* **Loosely-Coupled Adaptive Architectures** by Joe Ellsworth *has been saved in the topic* business.strategy *Last Change.:* A summary chapter for how companies must adapt this technique in order to remain competatively viable in high tech market segments. *http://www.pybiz.com/white_papers/architectures/joe/loose_coupled_architectures.pdf*

10/08/2000 *modified document which you registered an interest in* **Public Meeting Summary for Bay Area User Groups** in topic usergroups.meetings.sfbay *http://swm/groups/meetings/summary.html*

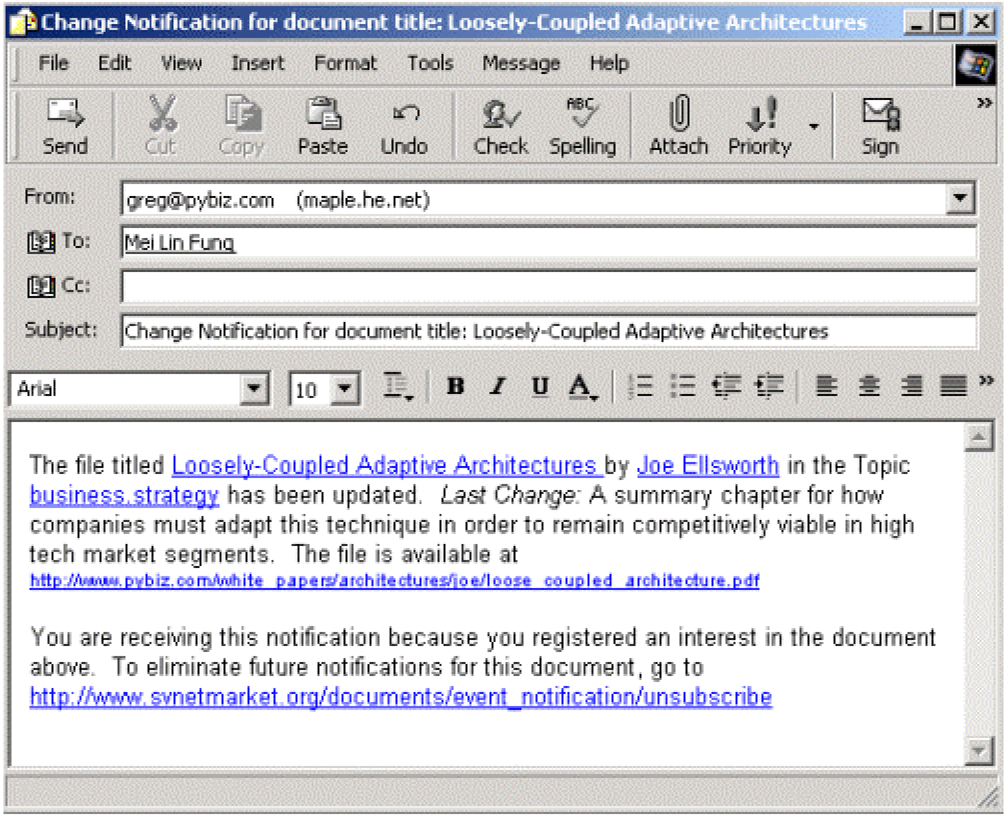
Order by [date] [title] [author] [topic]

Things to Notice:

- ✍✍ The list can be grouped by the data, title, author or topic.
- ✍✍ Clicking on the title will bring up the document which caused the event.
- ✍✍ Clicking on the topic will take you to the topic view where you will be able to see the documents in that topic and the recent events for that topic.
- ✍✍ Clicking on the author will take you to the bio for that author and can optionally show you other events and or documents published by that author.

Document Change E-Mail Notification

This is a sample of the email you would receive if you register interest in receiving update notifications for specific document. Notifications can also be requested on a topic level.



From: greg@pybiz.com (maple.he.net)

To: Mei Lin Fung

Cc:

Subject: Change Notification for document title: Loosely-Coupled Adaptive Architectures

The file titled Loosely-Coupled Adaptive Architectures by Joe Ellsworth in the Topic business.strategy has been updated. *Last Change:* A summary chapter for how companies must adapt this technique in order to remain competitively viable in high tech market segments. The file is available at http://www.pybiz.com/white_papers/architectures/joe/loose_coupled_architecture.pdf

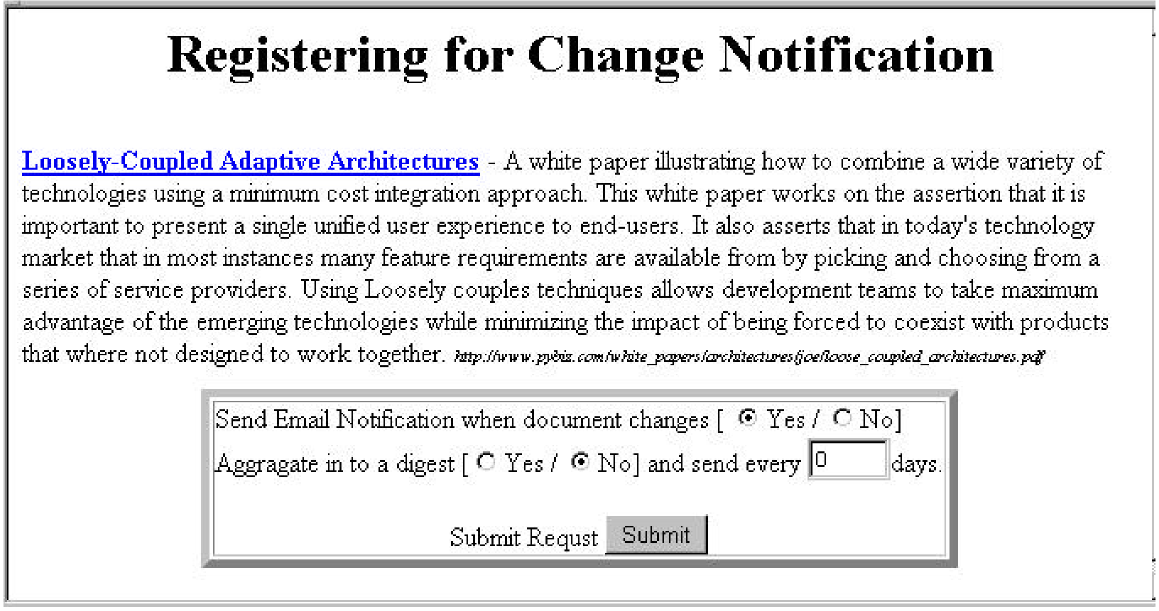
You are receiving this notification because you registered an interest in the document above. To eliminate future notifications for this document, go to http://www.svnetmarket.org/documents/event_notification/unsubscribe

Things to Notice:

- A single message is sent for each item notification item.
- Optionally you could have requested a daily or weekly digest of changes.
- Clicking on the unsubscribe will unsubscribe you for receiving notifications of this type for this document.
- Clicking on the Author "Joe Ellsworth" would take you to the web page displaying Joe's biography, document list, etc.
-

Registering for Notification of Document Changes

Registering for notification of change for a document is easy. Any time you can click on the document to download and read it you can also click on the link that will allow you register your interest. See the [Notify] link on pages that are listing documents.



Registering for Change Notification

Loosely-Coupled Adaptive Architectures - A white paper illustrating how to combine a wide variety of technologies using a minimum cost integration approach. This white paper works on the assertion that it is important to present a single unified user experience to end-users. It also asserts that in today's technology market that in most instances many feature requirements are available from by picking and choosing from a series of service providers. Using Loosely couples techniques allows development teams to take maximum advantage of the emerging technologies while minimizing the impact of being forced to coexist with products that where not designed to work together. *http://www.pybiz.com/white_papers/architectures/joe/loose_coupled_architectures.pdf*

Send Email Notification when document changes [● Yes / ○ No]

Aggregate in to a digest [○ Yes / ● No] and send every 0 days.

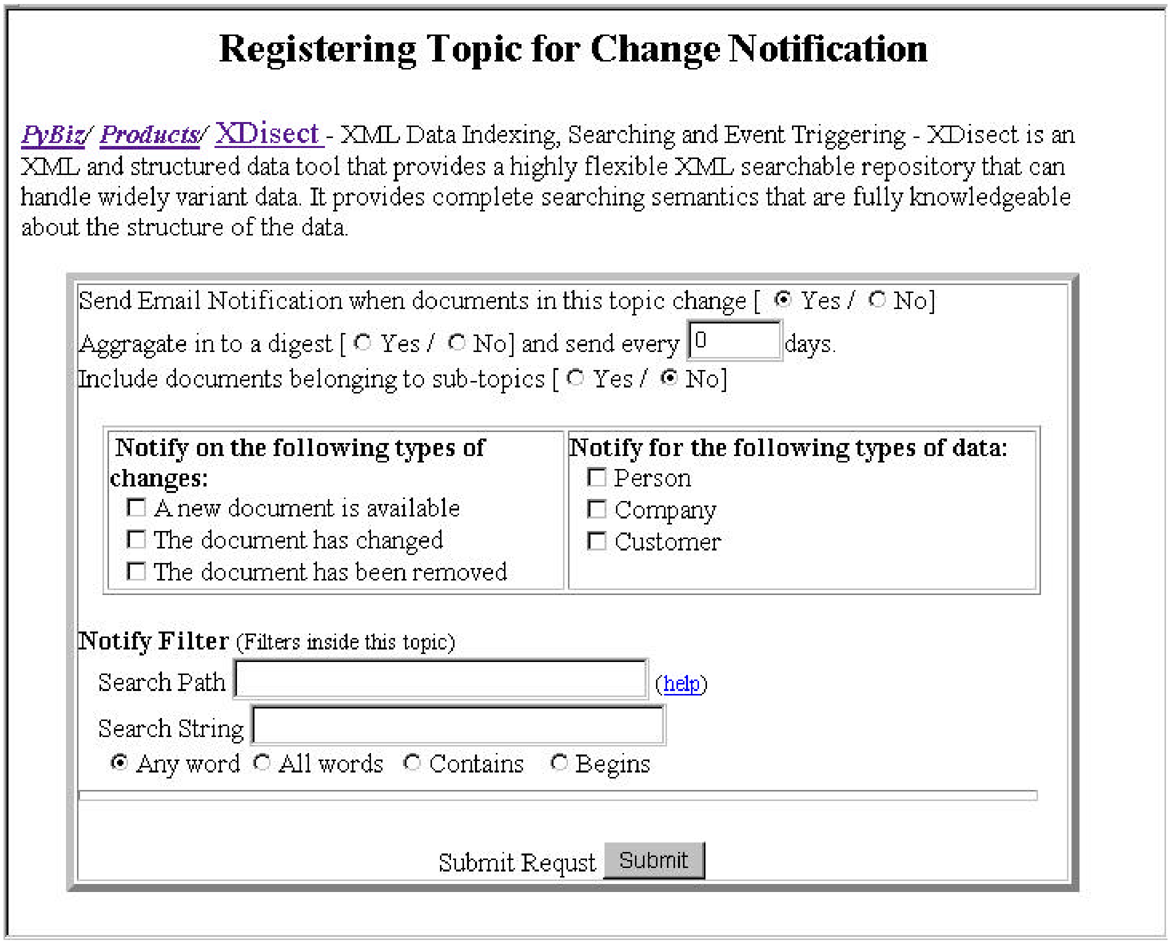
Submit Requst

Things to Notice:

- ✍✍ Cool isn't it.

Registering a Interest in a Topic

This screen allows the user to be notified whenever a document in a given topic has been changed. It is generally reached by clicking on the [notify] tag in the topic window.



Registering Topic for Change Notification

PyBiz/ Products/ XDisect - XML Data Indexing, Searching and Event Triggering - XDisect is an XML and structured data tool that provides a highly flexible XML searchable repository that can handle widely variant data. It provides complete searching semantics that are fully knowledgeable about the structure of the data.

Send Email Notification when documents in this topic change [● Yes / ○ No]

Aggragate in to a digest [○ Yes / ○ No] and send every 0 days.

Include documents belonging to sub-topics [○ Yes / ● No]

Notify on the following types of changes:	Notify for the following types of data:
□ A new document is available □ The document has changed □ The document has been removed	□ Person □ Company □ Customer

Notify Filter (Filters inside this topic)

Search Path (help)

Search String

● Any word ○ All words ○ Contains ○ Begins

Submit Requst

Things to Notice:

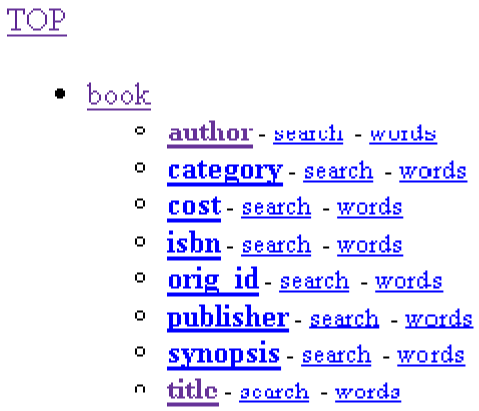
- The filter will search for items in the existing taxonomy that match the search specification. All other items will be filtered out. The types of data must also match or the items will be filtered. If not supplied with a search string the system will assume no filtering is required and will return all items in the topic.
- The help tag in the notify filter section will supply help on the specification of searching criteria.
- The aggregation feature causes the system to hold notification event until the specified number of days has elapsed since the last notification. If aggregation button is not set "yes" then the number of days to aggregate is ignored.

✍✍ *TODO: Add a test button that allows the user to see how their query would respond against current documents.*

✍✍ *TODO: Add a browse button for the search path button that would call up another small window and allow the user to drill down to the path they are looking for and then auto transfer the path specification back to the search path field in this window.*

Meta Data Browser

The meta data browser view of the allows the use of evolving meta data structures. It helps users explore what meta data is present and include it in their search and filter criteria with little or no effort on behalf of the end users.

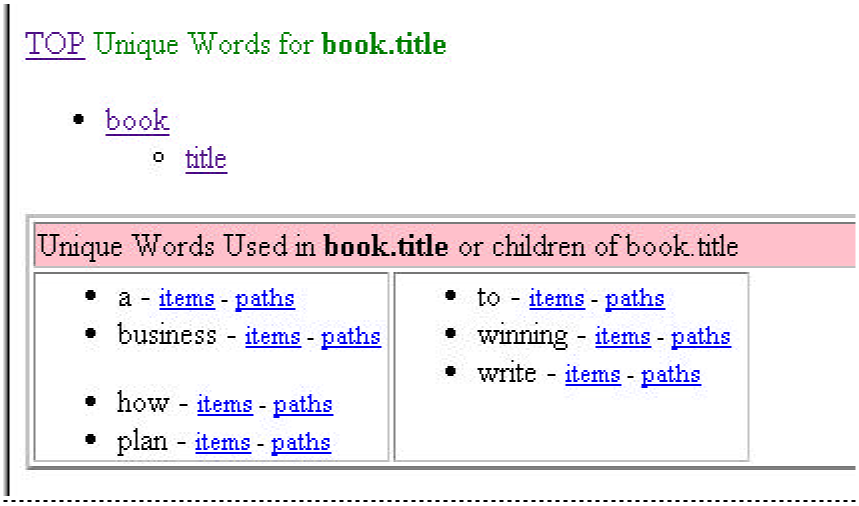


TOP

- book
  - **author** - search - words
  - **category** - search - words
  - **cost** - search - words
  - **isbn** - search - words
  - **orig_id** - search - words
  - **publisher** - search - words
  - **synopsis** - search - words
  - **title** - search - words

Things to Notice:

- We already clicked on the Book. Listed now are the next level of attributes available for a book.
- If we click on the attribute name we will see all the values for that attribute and any nested attributes that this particular item contains will be displayed.
- If we click on the words next to any attribute then all the unique words utilized by in this attribute or it's child attributes will be displayed..
- This was the type of meta data stored for a book.
- This screen is used to help very advanced users get a feel for the meta structure available at the site. They can browse down the direct taxonomy structures. We believe that once we have a better user interface worked out that many of these advanced capabilities can be made safe and understandable for novice users.



TOP Unique Words for **book.title**

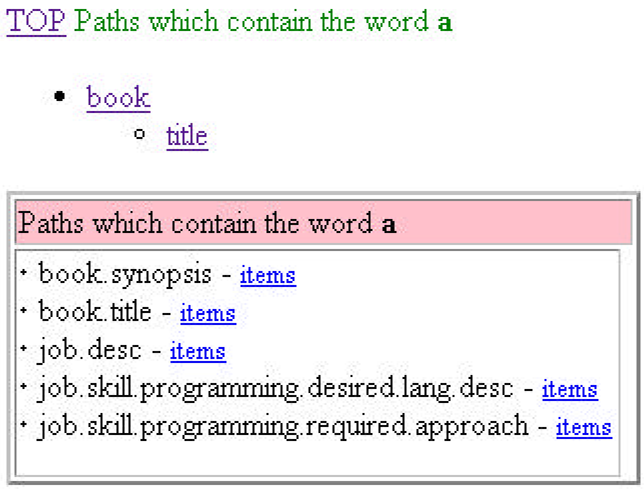
- book
  - title

Unique Words Used in **book.title** or children of book.title

- a - items - paths
- business - items - paths
- how - items - paths
- plan - items - paths
- to - items - paths
- winning - items - paths
- write - items - paths

Things to Notice:

- We reached here by clicking on the words item.
- By clicking on the paths link we can be shown the set of unique places in the meta data where that word occurs.
- By clicking on the items link we will be shown the items that contain that word for this path. These will be formatted as the typical search result screen.
- Information from this screen is Often utilized to fill out advanced search criteria in advanced search screen.



TOP Paths which contain the word **a**

- book
  - title

Paths which contain the word **a**

- book.synopsis - items
- book.title - items
- job.desc - items
- job.skill.programming.desired.lang.desc - items
- job.skill.programming.required.approach - items

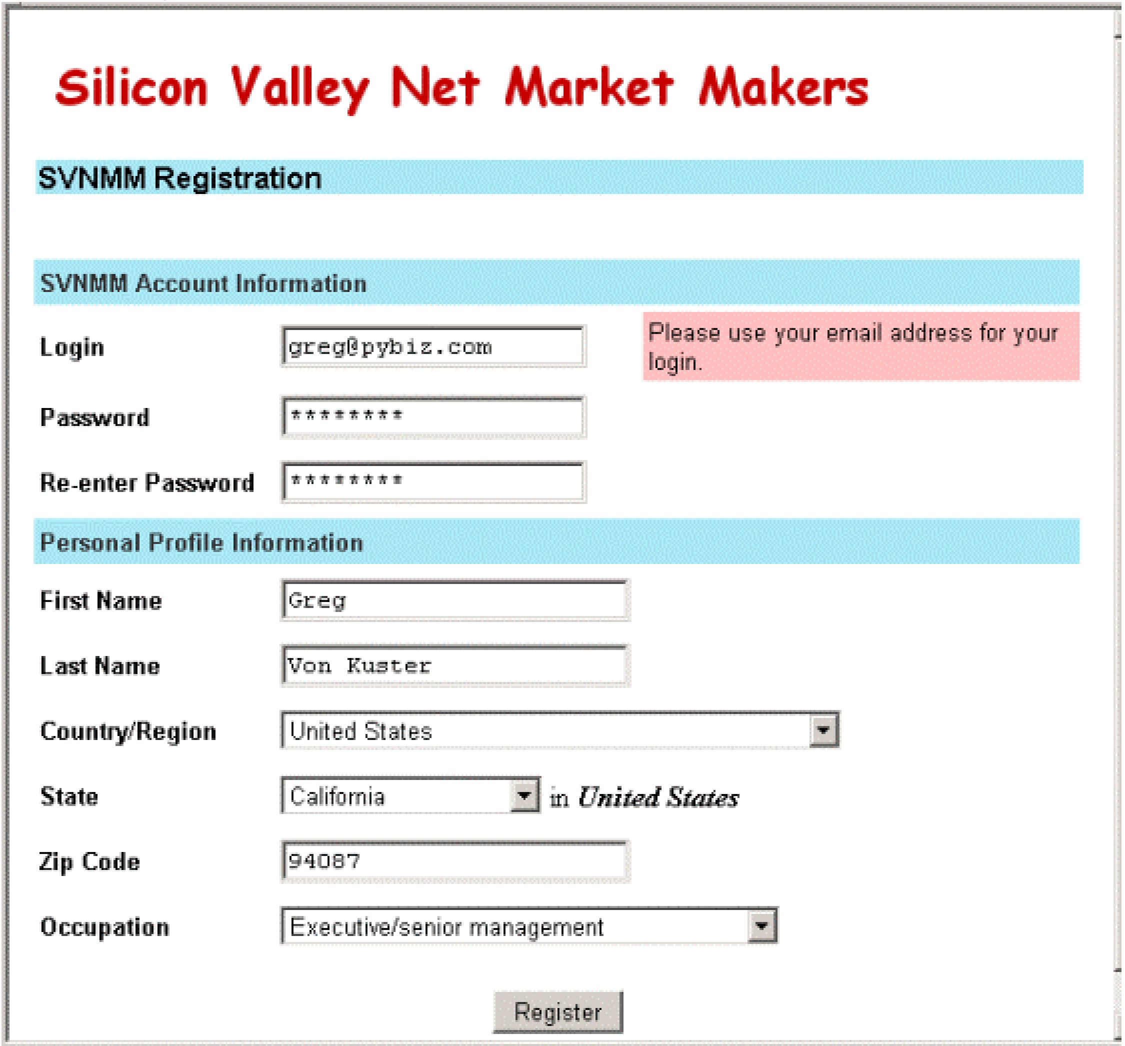
Things to Notice:

- We reached here by clicking on the path link from the words display.
- Each unique place the word occurs in the meta data structure is shown.

✍✍ Clicking on the items list will display the items in a way similar to the search result screen.

✍✍

User Registration Screen



Silicon Valley Net Market Makers

SVNMM Registration

SVNMM Account Information

Login	greg@pybiz.com	Please use your email address for your login.
Password	********	
Re-enter Password	********	

Personal Profile Information

First Name	Greg
Last Name	Von Kuster
Country/Region	United States
State	California in *United States*
Zip Code	94087
Occupation	Executive/senior management

Things to Notice:

✍✍ .

✍✍

User Preferences Screen

✍✍ ***TODO: Put in a Screen Shot***

✍✍ ***Should include the core user information.***

- ***Should include a list of topics the user is interested in***
-

Things to Notice:

- .
-

Publishing Assistant Screens

Publish a new file Assistant Screen

Things to Notice:

✍✍ .

Change a Files Meta Information Assistant Screen

✍✍ *TODO: Put in a Screen Shot*

Things to Notice:

✍✍ .

Publishing A document to Multiple Categories.

✍✍ *TODO: Put in a Screen Shot*

Things to Notice:

✍✍ .

Publishing A document to Multiple Sites.

✍✍ *TODO: Put in a Screen Shot*

Things to Notice:

✍✍ .

Configuring the Publishing Agent

✍✍ *TODO: Put in a Screen Shot*

Things to Notice:

✍✍ .

Configuring the Site Server

✍✍ **TODO: Put in a Screen Shot**

Things to Notice:

✍✍ .

Configuring the Mirroring Agent

✍✍ **TODO: Put in a Screen Shot**

Things to Notice:

✍✍ .

Configuring the Site Monitor

✍✍ **TODO: Put in a Screen Shot**

Things to Notice:

✍✍ .

What Motivated EContentMgr

The Problem

EContentMgr is the result of a problem that has reoccurred for our teams in the past and for which there does not appear to be a solution that fits the working style of our teams.

The size of our teams has generally been from 3 to 50 people who are very busy. Each engineer and architect will produce between three and five documents a month that must be maintained during the life of the team. A subset of these documents will be published on our web sites.

We have tried hiring professional technical writers who also managed the process of moving the content from our version control systems to the web site and have found that they can never keep up with the document flow and their positioning of the documents within the site is usually inadequate.

We have found that due to the way different people use our sites, a single document must appear in multiple places in the presentation tree and that the tree itself seems to have different organization based on who is currently navigating the site.

In short, we found that even though our senior technical and business people were able to produce new content to meet customer needs, it required weeks and sometimes months for the new content to be published. It was almost impossible to get this content published in multiple locations on the web site and the cost of changing the navigation tree was so extreme that we avoided it and published massive change lists once or twice a quarter. This was expensive and required the time of many senior contributors, web authors and other individual contributors. As a result, we never had the web site we really wanted. It was always “just good enough”.

Our company is composed of technical people who are comfortable with computers and standard file browsers, editors and command-line tools. We work in a highly decentralized fashion and as such find that centralized planning and release of web content just doesn’t work. When we have tried it our productivity is unacceptably decreased. We made the decision to empower individuals to determine which of their new and changed documents should be published and where they should be positioned on the web site.

To a certain extent, we feel that the depth and diversity of content is worth more than perfectly managed and presented content.

Solutions We’ve Tried.

There are lots of systems that attempt to solve this problem:

- ✍✍ The simplest fall into the class of ® Microsoft's Front Page. You have a complete site project file that contains a very small group of files and the entire site file is loaded into the tool. New content and links are added to the tree manually and the site is re-published. This runs into a problem when you have many people publishing new content and wanting to move content around in the tree. The project files tend to have an upper size limit in scale and when you start running into having documents in multiple places in the tree, it seems to almost completely break down. You could say that at a certain size and dynamic change rate, this approach collapses under it's own weight, resulting in long lead-times, massive planning activities and cost, just to do what should have been a relatively easy task.

- The more complex solutions like Documentum. This seemed to work for very large and very structured organizations, especially those that need to support multi-step approvals for the publishing process. Our teams typically were somewhat large but also required the speed and flexibility of smaller teams. We found cost to be a major issue because by the time we purchased, configured and installed the system, we had spent more than we could afford and we inevitably wound up with a person dedicated to doing nothing but managing the tool. This was something we also could not afford. We found that the cost of acquisition was high, the cost of planning implementation was high, the cost of maintenance was high and it still didn't yield the up-to-date, evolving web navigation structure we desired.

EcontentMgr Overview

Content management refers to the process of managing static files (documents) that are intended for publishing on a web site. It includes the version control, organization, placement and presentation of that content. It also includes the processes for moving the content from its development state to the web sites where it can be navigated. Site content often includes various types of documents such as Acrobat, Word, RTF, and others.

Simple is Sometimes Better

Many sophisticated and full-featured solutions for content management are available on the web today. We have listed many of these below. In most instances these are very sophisticated, full-featured and expensive both to acquire and manage.

Our problem with these sophisticated systems is that they enforce a greater degree of complexity and administrative overhead than we are willing to deal with. Their cost can also be prohibitive especially for early start-up companies and smaller teams.

EContentMgr grew out of an internal system that is not nearly as sophisticated as some of the existing solutions, but is quite competent to support the scenario above. It completely sidesteps the issues we found with the typical site management tools such as FrontPage.

EContentMgr works on an assumption that CVS is great for version control and with WinCVS it makes an adequate collaborative document control system. We only had to add the missing features while minimizing any intrusion in the use of existing tools that the team members already know and like. EContentMgr works just as well with other version control systems such as Visual Source Safe.

EContentMgr uses the XDisect engine to address the issues mentioned above. This allows for a minimum of complexity above that of CVS. Users are able to use their own tools such as Windows Explorer, Notepad, Word etc, to manage the actual content. By using CVS it allows multiple people to easily update the central repository.

Our system is based on the concept of dynamically evolving web sites that change incrementally. New content as well as changes to the existing content should become visible as rapidly as possible. This is opposed to many approaches that expect monolithic pre-planed publishing of a new version of an existing site.

You could successfully argue that EContentMgr works best to professionally publish business and technical documentation for complex projects. It is probably not the right tool for the top two levels of sophisticated company sites but it is more than adequate to use for the detailed levels of those sites.

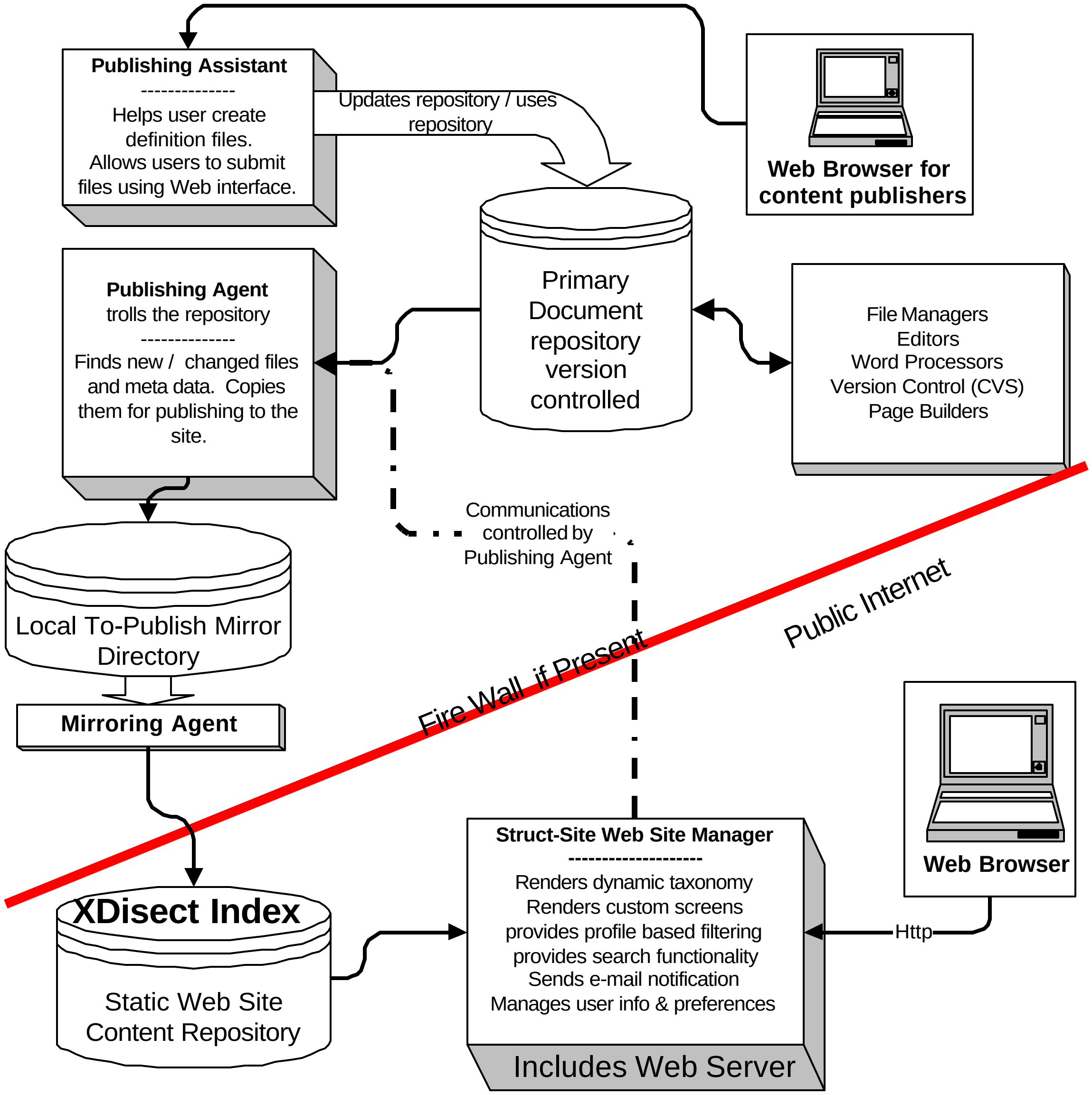
EContentMgr removes the need for centrally coordinated and massively planned site releases. In some instances local business processes require this kind of centralized planning. EContentMgr will not prevent such planning but it will not add as much value in such environments.

What EContentMgr Adds to Existing Version Control Systems

- The ability to detect documents that are ready to publish and publish them automatically to the site.

- The ability to provide a tree based navigation hierarchy.
- The ability to automatically retrieve the documents from the internal repository and place them on the published web site. We actually support publishing to different locations on different sites as well.
- The ability to retrieve a document from the CVS directory tree and place it in the correct navigation location for the web site.
- The ability to merge those documents automatically in a visually attractive drill-down menu system.
- The ability to provide a tree-based navigation hierarchy.
- The ability for a document to appear in multiple places in both the tree and drill-down menu systems.
- A simple mechanism to add metadata such as title and description to non-HTML files.
- The ability to filter which documents are shown based on user preferences.
- The ability to search on any combination of meta-attributes, data values and words.
- The ability to notify users when new content with specific attributes shows up or when existing content is changed.
- The ability to target content to specific sets of users with specific attributes.
- A user profile management subsystem.
- A user group profile management system.
- The ability to automatically notify users of document changes.
- The ability to automatically notify users of new documents showing up in specific areas of the site.
- The ability to display the change summary for any file. This can either be defined in the CVS change log or it can be added to the meta-definition for the file.
- The ability to browse files by change history rather than the traditional taxonomy.

Architectural View


Publishing Assistant

Helps user create definition files.
Allows users to submit files using Web interface.
Updates repository / uses repository
Web Browser for content publishers
Primary Document repository version controlled
Publishing Agent
trolls the repository

Finds new / changed files and meta data. Copies them for publishing to the site.
File Managers
Editors
Word Processors
Version Control (CVS)
Page Builders
Communications controlled by Publishing Agent
Local To-Publish Mirror Directory
Mirroring Agent
Fire Wall if Present
Public Internet
Web Browser
XDisect Index
Static Web Site Content Repository
Struct-Site Web Site Manager

Renders dynamic taxonomy
Renders custom screens
provides profile based filtering
provides search functionality
Sends e-mail notification
Manages user info & preferences
Includes Web Server
Http

✍✍ TODO: Show the publishing process the site display, the ftp process. Show the local monitoring site and the specialized web site functionality which can return a list of all files on the site, their sequence # and their hash code. Make sure it is clear that we use our existing map for the site until we have good reason to believe it has changed but should not be more than necessary because if it

contains thousands of files it could become cost prohibitive to walk read and re-hash the files. Shouldn't really have to re-hash anyway since it only has to read files that do not have a last change data identical or which do not exist in the tree.

The Publishing Process

We assume that users that are using a version control tool which results in a copy of a repository files which is kept up to date. This may be a standard NT file share that everybody edits using standard tools. This copy of the version repository is generally used only by the publishing agent but others can access it as local security policy allows. For those who are using a normal file share set up the system point directly at that share. (This holds true for RCS).

The documents are created and added to the version control system as necessary. The .DEF file for each document that needs to be published is added. This file will be the same name as the publishable file but will have a .DEF file extension. There will be a simple utility that will create the .DEF file automatically for some files that do not already have them. This will be extended to have a web page that can create a basic .DEF file from a simple user interface. Note: The .DEF files must be checked into version control just like any other source file.

The monitor agent will copy all files that have changed to a new location defined during configuration of the subsystem.

The web site agent will be running the run time version of the system and will provide an API where it can respond with an XML file containing the time, size and hash code for all of the files in it's a subset of it's current directory system. The files are listed relative to the configured root for the site. If any files are new, different sized or yield a different hash code then they will be moved to the relative directory in the output as defined by the .def files.

> ✍✍ *TODO: user screens for publishing a document. This should include a displayed taxonomy from which to select and filter the documents. It will also include a form where the user can enter the primary meta-attributes for documents.*

Any new .DEF files will be re-indexed in the system.

EcontentMgr uses an FTP agent directory configured with instructions to pick up anything that appears in that directory and copy it to a remote directory on a specified web server via FTP. There will be a subdirectory structure inside of the agent directory and any files placed in this directory will be copied to the same relative directory in the remote machine and will then be deleted from the host machine.

Publishing to multiple web sites is supported byconfiguring the publishing agent to make multiple copies of the files to be mirrored in different directory structures and running a multiple mirroring agents each configured to pick the files up from a unique location.

Simple New File

1) User writes a document
2) User saves document in repository.
3) User commits the document in the repository.
4) User creates a .DEF file for the new file.

5) User commits the new .DEF file to repository
6) Publishing agent finds the new .DEF and document file.
7) Publishing agent copied new file and .DEF file from the repository location to the mirroring directory.
8) The mirroring agent copies the files from the local mirroring directory to the web site and removes the files from the mirroring directory.
9) The local CGI task recognizes the new .DEF file and updates it's local index with the information in that definition file.
10) The information is now available for access by end users.

Modified A File

- User modifies a document
- User saves document in repository.
- User commits the document in the repository.
- The publisher agent's view of repository is updated.
- The .DEF file already exists to the changed document is automatically detected. The user could optionally have set the <release>no</release> flag on the document which would cause the system to ignore the file until they where ready to publish a new version.
- Publishing agent detects the new document file.
- Publishing agent copied new file from the repository location to the mirroring directory.
- The mirroring agent copies the files from the local mirroring directory to the web site and removes the files from the mirroring directory.
- The local CGI task recognizes the new .DEF file and updates it's local index with the information in that definition file.
- The information is now available for access by end users.
- Any necessary notification events are sent to people registered for those events.

Note.: When using CVS the publishing agent would never see the changed document until the user commits it. This means that no modifications to the .DEF file would be needed to hold publishing unless multiple interim non-publishable versions are intended to be published. This means that it becomes easy to publish a version of a file and keep on editing since the user can change the <publish>yes</publish> flag check the document in and then go on changing their local copy. They can change the flag back to <no> after the system has had time to detect and publish the interim version.

New File published with Publishing Assistant

- User writes a document
- User saves document on local hard disk.
- User brings up the Web-Browser and goes to publishing assistant URL.
- User uses web browser file selector to choose file to submit from local hard disk.

- The user enters the meta data such as Title, Subject, Keywords, etc to the screen.
- The user selects the categories the file belongs to.
- The user selects where in the physical directory structure to save the specified file.
- The user submits the request.
- The system automatically adds the user's user-id, name, etc as the author section of the system.
- The agent saves the file in the requested location in the repository. Moreover, commits it to the version control if using CVS.
- The system writes an appropriate .DEF file for the submission based on the users entered attributes.
- Publishing agent finds the new .DEF and document file.
- Publishing agent copies the new file and .DEF file from the repository location to the mirroring directory.
- The mirroring agent copies the files from the local mirroring directory to the web site and removes the files from the mirroring directory.
- The local CGI task recognizes the new .DEF file and updates it's local index with the information in that definition file.
- The information is now available for access by end users.
-

Note.: When CVS is utilized the supplied files are actually added and committed to the repository ideally under the user id of the person logged in to the publishing agent.
NOTE.: The file published can be a ZIP file. One of the meta attributes supplied to the assistant is a specification on expansion of ZIP files. Then the file is unzipped in the specified location on the system. This works on the assumption that .DEF files already exist on the system.

Publishing a File to Multiple Web Sites

This is the same as the agent step above. The only difference is that the .DEF file contains multiple destination web sites.

- *TODO: Hyperlink to the .DEF file for this activity*
- *TODO:Hyperlink to the publishing screen showing*

TODO: Note.: There is a design decision to make here whether the categories are the same across multiple sites or whether each site would have their own categories or whether there are a set of shared categories and a set of categories specific to that site.

Note:

Publishing a File to multiple categories.

This is the same as the normal publishing process it only requires a modification to the .DEF file to support multiple taxonomy category nodes.

- *TODO: Hyperlink to the .DEF file for this activity*

Defining a Category

Defining a category is as simple as naming it for use in a document. The category will automatically show up in the navigation structure just by using it. In most instances, users will want to add some descriptive text and possibly some HTML to introduce the category.

- User uses the category as a reference from at least one document
- The user creates the .DEF file for this category that will contain the title, the category navigation position, some descriptive text and a pointer to any HTML to display as part of the page.
  - ?? The HTML defaults to a system default if it is not specified.
- The user commits the .DEF file in for the category.
- The publishing agent detects the new or modified category file.
- The publishing agent copies the category file to the proper mirroring locations.
- The mirroring agent moves the file to the web publisher.
- The web publisher detects the new file and adds it to its local index.
- The category is now rendered in the navigation hierarchy as long as it has at least one document that refers to it.

Note.: By convention all category files have the semantic of cateogryname.category.def.xml
Note.: The Publishing Assistant can be used to generate category definition files.
Note.: The root category for the top of the system is defined like any other category.
Note.: Category files can be stored anyplace in the repository hierarchy but generally would be stored in a site/sitename/categories sub directory.

- *TODO: Hyperlink to the screen supporting this activity.*
- *TODO: Hyperlink to the .DEF file for this activity Make sure you define the where the category file is stored and how you define the category it is rendering. Do not forget external html page pointer that can be included in addition to the descriptive text.*

The Rendering Process

Taxonomy top level node

✍✍ Fill this in

Top level Navigation page

✍✍ Fill this in

Searching

✍✍ Fill this in

A Document is published in area monitored for Notification

✍✍ Fill this in

A document is changed in an area monitored for notification

✍✍ Fill this in

Configuration

Set up Process

A repository of files is set up. In general it is expected that this repository is controlled for write access by CVS or other version control system.

EContentMgr is configured to point at the base directory for that repository of files as it's source. It is also configured for a named web site(s) where the document can be published.

For users that are using a version control set up a user and repository instance to be used onlyby the publishing agent. For those who are using a normal file share set up the system point directly at that share. (This holds true for RCS).

Test Site for Pre-production Testing

In some instances, all content must be tested with the new navigation structures prior to exposing these to end-users. This can be accomplished by defining a test site and a production site in the publishing agent. Leave the mirroring agent running on the test site and turn the mirroring agent for the production site off. Once the

Two web sites will be required; the publishing web site and the execution / run time web site. Both of these web sites are based on Python and running the XDisect engine. The publishing web site allows the actions needed to defined and update the list of files to be included and provides automated assistance when creating .DEF files. It can also give a list of needed .DEF files. This one runs on the Intranet and has direct access to the user repository. The execution web site is the one accessed by the end users.

Site Configuration

Template for page header

Template for page footer

Company level configuration

All company-level information such as a company name, contact information, etc.

Source Configuration

Repository Configuration

HTML Templates

- ✍✍ *TODO: Introduce the templates. Including the variable interpolation. Including the local environment and user level variable interpolation.*
- ✍✍ *Delineate simple versus complex templates.*
- ✍✍ *TODO: introduce dynamically selected templates.*
- ✍✍ *TODO: introduce dynamic It is in templates for different includes.*
- ✍✍ *TODO: Introduce access to a given users profile attributes in the templates.*
- ✍✍

Page Template

```
<html>
   $$$header$$$
   $$$body$$$
   $$$footer$$$
```

```
</html>
```

Page Header

Page Footer

Page Body

Short Node Template

Long Node Template

Frames Template

Tree Node Template

Tree Node Page Template

Customization

Struct site can be customized in a variety of ways. We use a onion mechanism for customization. The simplest form of customization is brute force simple with more sophisticated customization requirements also requiring some additional work.

This is illustrated best by the fact that our standard banners and footers are stored such that they can be easily replaced with our customer GIF images.

This continues up the chain to the point where the customers can change the rendering logic itself.

A Page Template is changed

> ✍✍ *TODO: Describe how the changed page template is written tested and then posted to the production web site. NOTE: this may mean that we need to replicate to a testing server in addition to a production server.*

How EContentMgr Uses XDisect

- EContentMgr uses the taxonomy browsing capability built into XDisect to provide the categorized taxonomy browsing.
- Each navigation page is constructed dynamically by the CGI application that uses the XDisect engine to dynamically retrieve the next level of items to display in the navigation hierarchy. These items are fetched from the XDisect engine.
- The XDisect engine is the reason documents can have new metadata added to them with little impact on the overall application. This new metadata can be included in the rendering templates simply by referring to it in the interpolation strings in the templates.
- The local XDisect engine provides searching by metadata.
- The event notification service of XDisect is utilized to provide notification of document changes and new documents in specific category nodes for new users.

✍✍

Sample Data Structures

For the purposes of discussion the following XML structure will be needed as a reference.

Location	*A file definition specification.*

The Actual data structure

```
<publish_spec>
  <id>product/xdisect/struct_web_arch.pdf</id>
  <parent>Pybiz.Products.XDisect.Demo</parent>
  <parent>Pybiz.Demos</parent>
  <status>working</status>
  <site_path>product/xdisect</site_path>
  <title>XDisect Demo Script</title>
  <desc>This is the 2 liner</desc>
  <render>
    <long>
      <template> </template>
      <external_agent>  </external_agent>
    </long>
  </render>
</publish_spec>
```

Shorthand `Struct_web_arch.pdf`

- Id = unique identifier for the record. Must be globally unique for the entire system. For this reason it is commonly a URL or a sequential counter.
- Status = working => do not copy the file. Ignore and leave previous version.
- Status = Released => copy the file and add to index if necessary.
- Parent = the dot separated path to the categories where this document will be displayed.
- External_template = an HTML template that will be used to render this node. If not specified, use default.
- <long>
- Render_agent = the display name of a registered display agent. This would be a registered instance of a class that would be utilized to display the node.
- <site_path> = the location of the file in the web site. This will be the location where we copy this file relative to the site base. If omitted it will default to the same relative location of the file in the repository base.

Version Control Systems

CVS

This solution takes advantage of the fact that CVS already exists, so it does not attempt to provide version control for the repository. However, it will work equally well with other version control systems such as Source Safe. CVS is used by more development teams than any other document control and versioning system worldwide and is arguably the default choice for source management.

In the future, we may actually use the CVS version tags to retrieve site versions. In that instance, other version control systems will require custom configuration to work.

CVS is an open source product that participates in the same market segment as Microsoft's Visual Source Safe product. We find CVS is more robust and does not crash the way we found Visual Source Safe does. In our experience, CVS definitely scales to handle large complex projects better than Visual Source Safe.

CVS is free for an unlimited number of users. There are a large number of GUI front ends that can access it and it scales for very large, very complex projects.

CVS has been used for just about every large open source project including Perl, Python and Apache.

CVS is a distributed client-server application that supports distributed collaborative teams. For those people who do not want to install the client portion of CVS, it also provides a powerful web interface.

See the following for information on CVS:

?? http://www.sourcegear.com/CVS

?? http://www.wincvs.org/

?? http://dir.yahoo.com/Computers_and_Internet/Software/Programming_Tools/Software_Engineering/Configuration_Management/CVS___Concurrent_Versions_System/

?? http://sourceforge.net/docs/site/

??

We have spent money on Microsoft's Source Safe product and on much more expensive products like Rationale's version control system, but we would rather use CVS and some creative Python scripts than any of the commercial products. CVS just flat works reliably.

RCS

RCS is the ancestor of CVS. It is simpler and more limited than CVS but it is also fool proof to set up and administer.

RCS is not a distributed tool. The versions for each file are kept in an RCS directory contained in the same directory as the master.

RCS is a great system to use when multiple people are editing the same directory tree simultaneously. In this case, there is a high probability of two people changing the same file at the same time, thereby causing conflicts. RCS ensures that this will not occur.

We have found that RCS is great for two or three programmers who are simultaneously editing the same repository of files on a common Unix platform. Since all files stay in the repository, it is very easy to break each other's code, but for a normal web site, this is not a problem.

RCS's biggest advantage is that it takes minutes to set up and less than that to add brand new projects.

In my opinion, the things that make RCS not work well for development teams are likely to be beneficial for web site development teams. For this reason it should be given reasonable consideration.

✍✍ *TODO: Add Visual Source Safe, Perforce, Visual RCS, Rational hyperlinks here.*

Other Content Management & Publishing Systems

There are many document and content management systems available on the market. They range from multi million dollar systems with multiple year implementation cycles to plug and play systems.

Judge for yourself, but we don't feel that any of them meet our segment needs as well as eContentMgr.

Static Content Systems

Static content management systems are those

- **Interwoven Team Site** - Interwoven appears to be the closes in mindset and direct competition for eContentMgr. Interwoven like eContentMgr uses a hybrid repository / flat file approach to maximize the use of existing tools. It does not go as far as eContentMgr in allowing all of the content to be managed in standard version control systems and automatically republished.
- **Team Site** seems to be predominantly focused on controlling the and facilitating the publishing of front end web sites for very large and well funded projects where they are controlling the site from the perspective of the top several pages. EcontentMgr can be utilized in the same market segment effectively and at a lower total cost however the design point for EcontentMgr is the departmental group which has a large amount of evolving content which must be published for consumers elsewhere in the company and possibly by external customers. Where Team Site projects are large and focused on themselves EContentMgr is an enabling tool to help existing teams keep their content fully published and readily available with a very capable navigation structure. EcontentMgr includes features that are important for documentation sites that may not be important for the typical sales oriented sites which Team Site appears to focus on. One of these special features from EContentMgr is that it supports sending out notifications on document changes to people who have requested such notification all without any programming or configuration overhead on the part of the end user.

  It appears from an initial review that Interwoven is a very sophisticated tool which will require some non-trivial investment in time and enter to adopt. EContentMgr has many of the same features although it is more powerfully aimed at teams which already have version control and document production processes under control and do not want to change those processes just to meet their web publishing requirements.

  In many ways the interwoven solution has many of the similar limitations as Intermedia. Interwoven does not provide adequate material from their web site to determine the next level similarity in features so we where in some instances forced to extrapolate some items. We where unable to find the actual list price for deploying Interwoven.

So far we have not been able to find pricing for Team Site and as such do not have a complete competitive position.

?? Interwoven uses a traditional relational database for meta data storage. As a result it is easily accessible by a wide range of tools on the other hand more complex meta data requires additional administrative and programming overhead.

?? Team sites templating techniques are oriented towards changing the look and feel of the actual documents or enabling the content of those documents to be repurposed while EcontentMgr utilizes templating predominantly to control the navigation and look & feel for the site. PyBiz generally feels that introducing templating too early in the process forces people to abandon their favorite tools such as Word and will reduce the team wide participation in the site. In most instances the re-targeting of data is better done by manually preparing separate documents for the new form factor since the flow through the documents also changes dramatically when purposed for a smaller device.

?? Team Site seems to impose it's own works flow and versioning system. PyBiz utilizes existing version control systems to minimize impact on existing processes.

?? http://www.interwoven.com/developer/products/hybrid_arch.html

?? http://www.interwoven.com/developer/library/analysts/forrester.html

?? http://www.interwoven.com/developer/expertise/

?? http://www.interwoven.com/developer/expertise/general/moving1.html

It is clear that the choice to use EcontentMgr versus Team Site would be one of investment willingness and the nature of the project. EcontentMgr will generally be more applicable for teams with very limited budgets for time and money to contribute as well as some of it's advanced event, notification and dynamic site structuring will better fit highly empowered distributed teams which already have their own version control processes in place. EcontentMgr also has an unfair advantage by utilizing XDisect which allows it to embrace the use of much more sophisticated forms of meta data than Team site can handle.

✍✍ **Documentum** - is the high-end document management tool. Content management is also a major subsystem of these packages. Document can be considered the Cadillac of systems in the document & content management space.

?? Documentum supports a wide rage of advanced features the most impressive being a work flow based approval for publishing of new content. This would be the primary differentiator between EContentMgr and Documentum.

?? It is generally considered a high cost, high value solution where the cost of acquisition is high and the cost of setup, configuration and maintenance is also high.

?? Documentum will require intrusive changes to the document management. It seems to be the primary favorite for very large organizations with rather convoluted processes for managing and approving the publishing of new content.

?? Documentum provides tagging capabilities for content however it does not seem to be as robust as that provided by eContentMgr.

?? http://www.documentum.com/products/content/internet-4i/doc4i.html

✍✍ http://www.easysite.com/esfrontpage/ - Many of the capabilities of a site oriented package such as front page made available as a Web Service. EasySite is a Web-based site creation tool that lets you build your own site quickly, easily and affordably. Pre-built templates let you fill out simple online worksheets to create your site - with no programming or software to buy or learn. A wide range of pre-designed Web site styles and page layouts to put together an attractive Web site in just minutes. It's Affordable and there is nothing to buy and no setup fees - and hosting costs less than $20 per month. It's Professional - Get the look and features of a professionally built site!. You can Update your site anytime! Easy-to-use Web interface, you can update or add to your Web site anytime, anywhere - as many times you want.

Other Data Management Systems

Data-cleaning and categorization products. Some data management about data management and catalogue aggregation tools.

- Saqqara
- Cardonet
- Cohera
- Requisite
- Aspect
- Poet

Content Management & Publishing Embedded as a Component in Larger Systems

- Vignette,
- Broadvision
- ATG

Search Tools

- Mercado
- Verity

Uncategorized

- Wiznet
- versifi.com
- agentics.com
- profilesys.com
- pharostech.com
- http://www.flexisite.com/flexistar.nsf?Open

Community & List Management

http://www.greatcircle.com/majordomo/majordomo-faq.html - A Perl based list management with moderation, approved list membership, etc.

?? http://www.redhat.com/support/docs/howto/RH-postfix-HOWTO/c480.html

TODO:

- Add a orange box bullet to each page which does not have a picture.
- Consider allowing the use of both INI and XML for DEF files of course the .INI form would only support limited functionality. The reason is that INI form files are easier for users to edit.
- Decide whether to use PSXL or simple substitution templates. The simple substitution requires more inline CGI knowledge of the GUI but will probably result in a operational product faster while the PSXL would be much more flexible and would ultimately market the product much easier to maintain and extend. Chetan Thinks we should be using straight XSL which would be the longest in the design and construction phase but would also be easier to sell for partner extension.

Random Ideas to be Categorized.

- In the short term can just re-index but in the longer term will be able modify document detention files in the XDisect using the update with a unique key of id.
- Meta data enabled screen templates for both query, rendering and update will be needed.
- Different screen templates may be needed to collect different meta data based on document time.
- For HTML files, the title and description fields can be specified in the HTML file. The respective fields in the associated .DEF file will be automatically updated from the meta-fields in the HTML file. If no Meta field is present then the one in the .DEF file is used.
- When the site is created, the monitor agent should be configured with a shell command that can be executed periodically to refresh its version of the repository.
- We need to support specification of a node. This would be the root node for a given non-terminal point in the taxonomy.   This would allow us to override the rendering template for that level in the taxonomy.
- A query system should load the entire document taxonomy section into a list. The user would select a taxonomy location and then put in the value to search. The taxonomy location would be loaded directly from the XDisect repository.
- The publishing process would allow the person to browse a taxonomy structure in the bottom left frame of the Browser. The user would

continue to browse and then select the items in this list. When they selected the item from the list, that item would be added to the metadata parent for that item and shown in the main data frame.

- When the users add additional metadata, we will optionally display that data by name in the templates for the documents. This means that we can allow new meta-structures that we have never heard of before to be added to the rendering.
- When rendering we should allow a specification / configuration item showing custom rendering logic for a path. For example, “*person.skill.technical*”. A loaded configuration file would allow “*person.skill.technical = modulexy2.py.class49.method*”.
- We should be using the PSP/PXSL modules for interpolation stuff. That will allow ultimate flexibility in the template-based drivers for the system. In particular, the IF statements will be necessary as well as the FOR items.
- Ability for customer's portal users to access this site using an SST Like session management and session transfer capability
- Users should be able to specify a HTML file that contains overridden text to be utilized for rendering the long description field.
- We need to be able to specify what portions of the site will be done in a Site Directory. I think this can just be by creating a site directory and putting site-specific stuff there. With the right META tags they can control the whole directory.
- Need a directory level meta file that allows a person to override default behavior E.G. If a person wants to publish all files in a directory without requiring .DEF files then they should be able to do so.
- Need to allow HTML files to contain META tags. And if they do then use those in lieu of a .DEF file.
- Ability to make a static site like this protected by putting up the apache API content protector module. This will instantly protect their site(although, I think this is a module that most content management and web site tools have these days)

Will eventually need a agent converter that can convert PDF files automatically from .DOC, etc. Probably actually needs a conversion request. That is if the source document .DOC changes after the .PDF or other output is generated then we should know who to request a conversion step from. This could be extended to an automatic infrastructure in the future.

Table of Contents