Content Change Management: Problems for Web Systems

Note: This is a reprint of an article original © Springer Verlag 1999, from the proceedings of their 9th International Conference on SCM.

Susan Dart
Dart Technology Strategies

Behind the facade of a web site, lies the task of managing its infrastructure and content. This is driving the Internet economy into a Web crisis. The software community has experienced a similar crisis and knows that configuration management (CM) is a key tool in resolving it. Nine challenges facing web systems are presented. As the entire world becomes connected to the world wide web, content problems will be magnified. While traditional software CM provides a static solution (such as via a centralized development methodology creating batched, planned releases), content CM will provide a dynamic solution (via distributed, real-time updates) in response to user traffic monitoring. It is imperative that the lessons learned from CM be applied to web tools. Otherwise, the Web community is doomed to experience all the delivery, quality and complexity problems that have plagued the software community.

The World Wide Web is a unifying force bringing the world closer together. Regardless of race, color, creed, skills, educational background, computer platform, browser, nature of business, geographical location, and job position, we all *look* the same. Business is being transformed into E-commerce. Such revenue is expected to hit $220 billion by 2001 [1]. Behind the facade of e-commerce though, the Web Crisis is looming [2]. It is the exponential proliferation of web content created, and maintained, without any expertise in data management techniques—the proliferation of *hacked together* web-based systems developed without any rigorous approach and kept running via a continual stream of patches. Companies are desperate to put their business applications on the Internet. *Gold rush fever* is encouraging business start-ups centralized around the Web. With the advent of many low-cost publishing tools that are very easy to use, web system creation is now so simple that anyone without programming skills can create one.

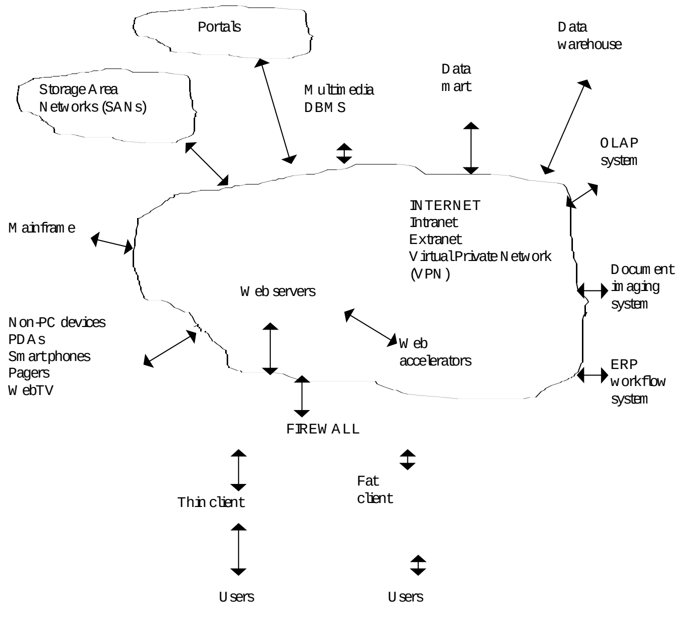
The demand for content creation and maintenance is escalating at an unmanageable rate. Some analysts have predicted that by the year 2002, the market revenue from content management tools will be around $5 billion [3]. And even when we have it under control, content has a multiplier or snowball effect, where we will further exploit new ways of using content. First-generation web systems have focused on providing access to any piece of information around the world. The next generation web systems will focus on knowledge management—managing the semantics, or concepts, of content rather than just the raw information.

For now, though, we see the shortcomings of first-generation web systems. There are problems with information being published on the web site at the wrong time and information that is inaccurate, top-secret, corrupt, inconsistent, unauthorized, unchecked, garbage, stale, or inappropriate. These can have devastating consequences for companies such as millions of dollars lost in lost revenue, lost customers, and lowered stock prices (such as with software crash [4]). The causes are easily linked to lack of well-defined processes, testing, cross-checking of information, authorized changes, security checking, or responsibility for coordinated changes. Essentially, the problems stem from poor CM practices. The first generation of web systems were crafted from immature tools and languages, and inexperienced staff. To properly provide Change Content Management (CCM)—CM for web systems—we will have to go beyond the capabilities traditionally provided by industrial-strength software CM tools because the challenges presented by the emerging web economy are exceptional.

This paper is designed to raise questions about CCM for the Web so that we can understand the new demands placed on companies by web systems. A web system is a generic term for an application that can be accessed via the Web. It fundamentally consists of content (its data, such as a document), application server (for executing actions on the data, such as updating document), access (its interface, such as the client's browser) and the web server (common ones include Apache, Internet Information Server and Enterprise Server). The biggest challenge for the Web community is how to build maintainable web systems that are highly responsive to immediate, high-volume change.

This paper defines the kinds of resources in the Web environment that are used in web systems, specifies the classes of web systems being developed, identifies the many challenges that companies are facing in their efforts to understand CCM, highlights capabilities provided by software CM and web CCM tools and ends with recommendations for approaching the solutions.

Figure 1. *The Web Environment*



The World Wide Web Environment

Web systems can be huge, with millions of pages, many interconnections, and incredibly high hit rates. Consider Figure 1 which highlights the many kinds of resources throughout the Web that can be components of web systems. It shows that users can be connected to the network via a thin client or a fat client. A thin client means application code is resident on the server, rather than on the client (fat client). A firewall determines the kind of access, encryption and security levels. Web servers provide much of the application code and can have accelerators for caching dynamic pages in order to improve user access time. The network can be specialized into an intranet, extranet or virtual private network (VPN). An Intranet is an internal network behind a firewall that allows only users within the company to access it. An Extranet allows outside partners to have access to the Intranet. A VPN is a secure and encrypted connection between two points across the Internet. It acts as an Intranet or Extranet except it uses the public Internet as the networking connection rather than a company's own wiring. This enables, for instance, a company's branch offices to be inexpensively connected via the Internet.

Attached to the network can be other types of networks such as storage area networks (SANs) and portals. SANs are networks that pool resources for centralized data storage. They may include multiple servers working against a centralized data store built with redundant hardware such as RAID (high-volume storage) devices. Portals (such as Yahoo!, AOL) are full-service hubs of e-commerce, mail, online communities, customized news, search engines and directories, all suited to the particular needs of an audience. Portals are evolving into corporate enterprise portals. Such portals, for instance, enhance corporate decision-making by integrating the company's applications, thereby removing barriers that exist between business units.

Other resources that can make up web systems are: Data Base Management Systems; workflow applications used for optimizing business processes, such as Enterprise Resource Planning tools (e.g., SAP, PeopleSoft, Baan); database applications such as OnLine Analytical Processing systems, which allow users to perform *multidimensional* analysis on data via their browsers; document management tools [5] for providing access into shared libraries of documents; imaging systems for optical character recognition of documents; data warehouses containing terabytes of data;[1] multimedia databases for holding archives of music, speech, videos; mainframes which contain approximately 70 percent of legacy data for large companies; data-marts, which are data warehouses with their own unique interpretation of business data to suit certain functional needs of a business unit; and, non-PC devices, such as pagers, personal digital assistants, WebTV, and smart phones.

Web systems are made up of various combinations of the resources shown in Figure 1. Each of the resources imply content that can be dynamically added, changed, deleted, accessed, manipulated, along with their relationships and hyperlinks. CCM will need to control the static content that goes into the web system along with the dynamic content that is created during execution of the web system. Different kinds of web systems are being developed which affect the nature of CCM.

Types of Web Systems

It is difficult to classify the types of web systems being built today as there is no universal blueprint for such systems, the design is still an immature art and the systems themselves are evolving fast. But for the purposes of opening up discussions about CCM, we need to understand the types of architecture of web systems with respect to content creation. In a broad sense, a web system which is visible via its web site, either acts as a provider of information or is an application. But the applications can be of different types.

From a content perspective, we are interested in types of web systems which have data points where data can be added, changed, deleted, accessed or accumulated. Once we understand the types of applications, we can then determine the nature of its CCM needs, its development processes, and types of tools needed to properly maintain it. A web system can be categorized as having the properties of one or more of the following classes:

Informational: information sites with read-only usage, commonly called "brochureware" e.g., information presented on a site that gives details about a company and its products. First-generation web systems are this type and are static.

Delivery system: download content to user or resource e.g., download upgrades or plug-ins

Customized access: access is via a customized interface or based on user's preferences e.g., my customized view of my Internet Service Provider's home page, or favorite portal

User-provided information: user provides content by filling in a form e.g., subscription to a magazine or registering for a company's seminar

Interactive: Two-way interaction between sites, users and resources e.g., business-to-business

File sharing: remote users collaborate on common files e.g., users coordinate schedules

Transaction-oriented: user buys something e.g., buys books or travel tickets

Service provider: rentable applications; user rents an application on a per user, per month basis e.g., virus scan program

Database access: user makes queries into a database e.g., supplier looks up catalog of parts

Document access: libraries of online documents are available e.g., view corporate standards

Workflow-oriented: a process has to be followed e.g., order entry automation

Automatic content generator: robots or agents automatically generate content e.g., "bots" scour the Web to bring back specific information such as best price on products.

Given these classes, it becomes obvious that content can essentially be created by anyone or any other resource: from the content designer, the webmaster, any user, another database or device, or other web system. From a CCM perspective, it is straightforward to capture content that makes up a released baseline since that is static content, but what about content that is created or changed dynamically? This raises four key questions:

(1)] What constitutes a configuration item for a baseline with static and dynamic objects?
(2) How can dynamic baselines be captured?
(3) Now that the user of the web system participates in the creation or changing of a baseline, how does that affect the definition of the CCM lifecycle?
(4) Are CCM requirements different for each class of web system?

These are some of the questions being asked by webmasters, developers and CM managers.

Enterprise Challenges for Web Systems

CCM is not a problem for small, static web systems managed by a few developers. It is a problem for medium and large, enterprise systems that involve many content developers creating many pages that will have a high hit rate involving high-volume database accesses and updates every minute. For instance, the NASDAQ stock exchange system [6], is a web system of types 1, 4, 5, 6,7, 9, 10 and 12, and was built to sustain 12 million hits per day with 8 web servers per database server. When the stock market goes *crazy*, the NASDAQ site gets 20 million hits per day. Its content must be completely accurate, and it changes within seconds. Boeing [7], with a web system of types 1, 2, 4, 5, 9, 10, and 11, has 1 million pages hosted by 2,300 Intranet sites on more than 1,000 web servers.

Developing and maintaining such large systems with large volumes of content offers many challenges to companies. These challenges span technical, people, process, and political issues. The major ones obvious today are the following, and are described in detail below.

1) the dynamic, active nature of content
2) variant explosion
3) the free-form style of development
4) the performance effect of content
5) scaleability of content
6) the urgency and frequency of change to content
7) the outsourcing and ownership of content
8) the immaturity of tools, techniques, standards and skills
9) corporate politics.

The Dynamic, Active Nature of Content

Web content is dynamic because it is created on-the-fly based on a user's or agent's request. It is active because programs are executed in response to the request and to the user's environment (browser and plug-ins on the client side). For instance, HTML is static but when combined with active controls (such as ActiveX), it becomes dynamic such as when the web site gives users feedback on the type of data they are supposed to enter to make sure the input complies. Content can be generated and changed in real-time such as with tables, forms, database queries, documents, and code.

Content is made up data objects, component libraries, and code. These can be static or dynamic, singular or a collection, compiled or interpreted, source or binary code. Objects include documents, images, streaming video and audio, files, or tables. Code can be active controls and scripts such as: ActiveX controls, Java, C++, VisualBasic, HTML, DHTML, XML, VRML, OLE controls, Active Server Pages, Java applets, VBScript, JavaScript, and ISAPI, CGI, and Perl scripts. Scripts or behaviours can be attached to web objects allowing, for instance, the user to alter attributes, such as color, positioning and font size on objects or execute applications. Component libraries are reusable code to be used as toolkits. Examples are JavaBeans, Microsoft Foundation Classes and Lotus' eSuite of business applets.

An applet or a control is a compiled binary file that a field in the HTML references. A script is executable code in a readable source language that can be embedded directly in the HTML tag. In essence, an object becomes a container for various pieces of content, all of which, need to be under CM control.

We are moving toward container-based, or a component bundling approach, to software development. This means CM techniques need to account for embedded scripts and customized components. Also, the executing environment needs to be taken into account. For example, if a browser does not support a certain scripting language, then the behavior of the web system will be different. Also, HTML files can be manually touched up. Scripts can easily be changed because they are interpreted whereas applets or controls typically are compiled. This assumes that the source code can be accessed, which is not the case when components are bought and reused in their binary form. Hence, changing or recompiling is not an option sometimes. Executing code may require a series of steps. For example, a Java file is compiled into platform-independent bytecodes; these are then processed by a Just In Time compiler to yield fast native instructions for a particular platform. All the above objects types, their relationships to intermediate forms, and all the tools, need to be tracked for good CM practices.

Web pages are dynamically created, which means that any CM control also must be of a dynamic nature. For example, an .asp file[2] is recognized; the VBScript is interpreted with the appropriate database or related files being accessed; the server creates the full HTML on-the-fly thereby dynamically generating the web page that is then displayed. How is all this data tracked for CM purposes? How is a dynamic baseline captured? To add complexity, hyperlinks can be created on-the-fly to point to documents. This changes the original baseline. Also, dynamically generated pages can be customized using ActiveX, CGI scripts, JavaScript and DHTML frames.

There is more. Content has meta-data associated with it that must be captured:

- The separation of content and format. Companies have stan dard templates into which content is published. These templates are part of the released baseline.
- External structure information, such as the hierarchy and relationship of web pages
- Internal structure information, such as embedded objects
- Hyperlinks to internal or external pages, static or dynamic
- Task objects that indicates some activity must happen to an object, such as updating the content
- Transaction, such as data involved in carrying out an e-commerce activity

- Security information attached to each objects
- Audit logs related to the activity on each object
- Tool compatibility information, such as the version of the browser for which this object is valid
- Bill of materials: the artifacts used to create the baseline (tools, tool options, data, files)
- Generated or converted files, such as a Word document that is converted into HTML
- Validation rules, such as a form requires input validation for each field
- Handler rules, such as a data base access request invokes certain tools and operations.

There are obviously many properties about content that need to be captured. Ideally, a company should have a well-defined CM data model that captures all the properties and relationships of content. With that, configuration items, baseline,s and releases can be defined.

Variant Explosion

Web systems imply a variant explosion problem. Consider that web systems are either created from scratch, are redesigned or merged web systems, or are web-enabled legacy applications. In many cases, companies must live with all these systems in parallel. Thus, a company could easily have a nightmarish number of versions of their latest baseline. For example, it has four variants of its main application available at all times:

- The demo version which is a partial web-enabled baseline of the original legacy code with a minimal set of functionality as this is the textual version
- a full version that is the same as the first where all functionality is available since it is the graphical version
- the true web version that is a completely redesigned form of the application ideally suited to the web rather thanmerely web-enabled legacy code, and
- the original legacy system for non-web use.

Each variant must work with two different browsers (Internet Explorer and Netscape Navigator), including the latest three versions of those browsers—and support five different languages for international use. Hence, we have (1 * 5) + (3 * 2 * 3 * 5) = 95 potential variants.[3] Most companies have different teams working on separate variants without much communication, reuse or change propagation across common code. With the variants, come all the complexity of parallel development support for simultaneous changes and concurrent baselines, along with significant change propagation to selected variants, thereby demanding change set support [8], more sophisticated change tracking along with help-desk support and much better release planning and change scheduling. The ramifications are dramatic. Variant management and change propagation have long plagued software companies.

The Free-Form Style of Development

Web system development is different from traditional software development. [9,10] This is due to the nature of the tools, languages, skills of the developers, and the dynamic nature of the Web environment. There is tremendous pressure on developers to code and publish. And the web tools support this free-form style of development. Also, the skill set of the developers is quite limited with typically no experience in software engineering. They are guided only by the capabilities of the tools and languages that, as we know from software engineering practices, cannot be adequate.

Scripting languages (such as JavaScript, Jscript, Tcl, VBScript) are changing the way that applications are developed. Most of these are interpretive languages or use Just In Time compilers. This leads to a style of change on the fly. There is no process between creating content and publishing it. Programming has gone from a process-oriented compiler-based approach, to combine components, mix in some new code and publish. Essentially, this squeezes the change cycle time dramatically because all sense of process is eliminated. This enables a faster rate of change that is a real benefit for web systems but provides greater opportunity for errors through lack of testing and content coordination and authorization of change. The question becomes how can testing, system integration, load testing and release management processes be inserted into the code-and-go paradigm to enable proper CM? Some companies use staging areas for testing before publishing to a live site, whereas many do not.

The complexity of web system development can be seen in Table 1. The major phases are highlighted along with who assumes responsibility for those steps. There are at least nine key steps involved in getting the web system functioning. At each point, CM issues come into play, such as, which release or version of the web system is being changed or published or tested or registered or validated for security purposes or being moni-

1. MAJOR ACTIVITY IN WEB SYSTEM	WHO DOES THE WORK
2. Design and creation	Web Team or IT Dept. or Outsourced
3. Infrastructure support: servers, network connections, databases	Outsourced to network management company, or hosted by IT Dept.
4. Testing e.g., compatibility of content, link accuracy, viewable by all kinds of browsers	Web Team or IT Dept.
5. Publishing of content	Business Units or Web Team or IT Dept.
6. Registering of sites on search engines	Web Team or IT Dept.
7. Security checking: access control, hacker analysis, virus detection	Web Team or IT Dept or Security Consultant
8. Monitoring: traffic performance: intelligent load balancing and web page redesign; replication; web accelerators/caching; traffic shaping capacity planning	Web Team or IT Dept.
9. Maintenance: content evolution via changes, enhancements, deletions, redesign	Content experts or Web Team or IT Dept.

Table 1. *Typical web system lifecycle phases*

tored for hits or performance improvements. Without CCM practices and tool support, these activities become fraught with errors. Automated workflow along with role-based activities must be supported in web tools

The Performance Effect on Content

Performance—particularly response time to a user's request—plays a major role in influencing content design. High performance web systems have continuous traffic monitoring. Users must have immediate access to quickly changing content under any load situations. If access times are not acceptable, a company makes a decision to either install web accelerators that enable caching to improve performance, or it redesigns the content for better access. For instance, at the Olympics site [11], traffic monitoring showed bottlenecks for users by having to navigate too many pages to get to the right content. The web site was redesigned on-the-fly to make access easier and speedier along with adding caches.

Web accelerators, or caches, are beginning to play bigger roles in performance enhancement, with content being designed to take into account caching techniques for accelerators. But a dependency results between the content baseline and the version of the caching algorithm and server that are used. Also, server crashes (such as with the E*trade brokerage site crashes that shut out users who lost money through lack of trading access) must be catered to in contingency plans. This means content must be replicated across servers that, in turn, means synchronization and distribution of real-time updates.

Scalability of Content

The Olympic and NASDAQ [12] web systems are huge in terms of number of pages, amount of traffic, and number of database and web servers. Millions of pages cannot be reasonably stored in a flat file system. Databases are obviously required for storage and are being redesigned to suit web access. Some database companies are redesigning their products so that web applications are stored directly in the database, such as Oracle's WebDB. This helps with scalability, reliability, and administration. It is likely that first-generation web systems will be redesigned to use web-enabled databases. This means that CM capabilities must be integrated and synchronized with database facilities.

The Urgency and Frequency of Change

The web enables the paradigm of change at the speed of thought. The mindset is typically: I see a problem and can, or need, to fix it immediately because it is globally visible. Corporate embarrassment or even worse, litigation, needs to be avoided. There may be no time to follow through a normal change life cycle (such as with a change request, Change Control Board, change authorization, edit, testing and re-release). Because the change can be done so easily, process is often bypassed. All the benefits of change tracking are lost. Repeatability will be a difficult benefit to achieve. Rollback of a site may be the only option for companies, but the corporate need of keeping the web site accurate takes top priority. There are changes that may need to be propagated across all pages of a web site, or just a few pages. For example, simply changing a copyright notice may involve changing each of the 1 million pages, whereas other changes may involve a select set of pages so that an incremental publishing capability is required, along with ways of organizing files into partitions to enable incremental updates. A company needs to define its classes and priority of changes and decide what process should be followed for each type of change.

Outsourcing

Outsourcing is a significant trend for industry, especially for web system creation, and sometimes maintenance. It is done for many reasons: to reduce operating costs, share risks with others, access leading-edge technology without having to purchase the infrastructure for it, use expertise not found in-house, do things more quickly, and to focus more on a company's core competencies. Outsourcing does require distributed management techniques along with doing CM with a third-party.

Easy-to-use web tools and specialized commercial-off-the-shelf tools (such as OLAP, ERP, document management) are helping to change the political infrastructure of companies. For instance, business units no longer are forced to rely on the Information Technology (IT) department in order to get things done. They buy the best tool that suits their need, bypassing IT. They can even rent the infrastructure for supporting the tools, and outsource its administration. This complicates issues of who has responsibility for what, how to maintain control and visibility over outsourced changes, and whether a business unit guarantees that a quality process was followed for the outsourced work.

Immaturity of Tools, Techniques, Standards, and Skills

Engineering techniques for web systems are in their infancy. Tools, standards, and skill sets are maturing, albeit slowly. Each month new tools and new versions of tools are released that support easier ways of building web systems. As a result, companies have to maintain different tool technologies in parallel. Standards (such as XML (eXtensible Markup Language) from World Wide Web Consortium, or WebDAV (from the Internet Engineering Task Force) are slowly being developed that in turn will affect the tools. There are many web technology tools that enable easy publishing of content without team coordination or process. Because of the many choices, large companies will end up having their business units using different tools. To get some control over how content is developed, and to ensure that quality processes are followed in publishing content, companies will have to define standards and guidelines. These standards will pertain to style templates, component libraries, tools, languages, servers, testing processes, and CM.

Web systems require developers and content experts for their creation and maintenance. Many web developers have little background in software engineering. Content creators can be human resources personnel, marketing people, accounting staff, etc. Their web skills are totally dependent on their knowledge gleaned from the web tool set and any training class they attended. This implies that the tools need to have interfaces that suit the content writer, yet have excellent CM processes embedded to compensate for the lack of software skills.

Corporate Politics

There is confusion in companies these days as to who should have the right to publish content on the web site. For instance, business units publish independently from the IT department. Essentially there is lack of control as to what goes up, when, and how it has been tested and whether it conforms to standards. This is particularly a problem when the web system has content that must be coordinated and validated as a whole with other departments (accounting, marketing, personnel, etc.) or with other applications. Who assumes responsibility for the information's accuracy on the web site? Who assures that quality control processes have been followed before information is published on the site? Who is responsible for making changes? Who assumes the cost of change? The IT department's role is changing dramatically—from an infrastructure provider to that of a strategic advisor and standards producer. Many traditional IT functions, such as network administration, are being outsourced. Outsourcing will significantly change the modus operandi of IT departments. Web creation is mostly outsourced these days. Companies face a delicate balancing act in trying to rein in the proliferation of web systems while leaving employees freedom to meet their business needs.

Software CM–Major Part of Content Change Management

Everything that the software community has learned about CM

GOAL	EXPLANATION
Identification	Uniquely identify parts of the content
Control	Version control of all objects including baselines
Status accounting	Tracking the status of all work on all objects
Audit and review	Keeping an audit trail, confirming all processes followed
Cost-effective production	Fast and quick builds of software releases
Quality automation	Ensuring all testing, notifications, signoffs, reviews are done
Teamwork optimization	Enabling teams to work in parallel effectively
Enabling change	Containing the explosion of changes

Table 2. *Goals of software configuration management*

can be applied to the CCM problems. Software CM spans a significant spectrum of activities and roles within a company [13, 14] and Table 2 highlights the main goals of CM. The software CM tool vendors are adding CCM capabilities to their tools.

Web tool vendors are beginning to realize that CM prac-

CM TOOL	VENDOR	WEBSITE
Continuus	Continuus	www.continuus.com
ClearCase	Rational	www.rational.com
Harvest	Platinum Technology	www.platinum.com
Perforce	Perforce Software	www.perforce.com
PVCS	Merant	www.merant.com
Source Integrity	MKS	www.mks.com
SourceSafe	Microsoft	www.microsoft.com
StarTeam	Starbase Corp.	www.starbase.com
Team Connection	IBM	www.ibm.com
TrueChange	True Software	www.truesoft.com

Table 3. *Some Commercial Configuration Management Tools*

tices must be incorporated into their tools Advice on good web design [15] is beginning to highlight the importance of CM but only in the sense of version control of files. However, according to Powell, web engineering advice completely ignores CM. Table 3 lists some of commercial software CM tools.

Software CM vendors are taking different approaches to CCM support in their tools. Some, such as StarTeam, are web-enabled and have purchased web technology companies with the intention of tool integration. Others, such as TrueChange, have decided to build a completely new software CM tool for CCM. Others such as Continuus and MKSIntegrity, have added on CCM support. The former's offering, WebSynergy and WebPT, provide a web front-end into all of its existing CM process-oriented capabilities as well as web-authoring tools with transparent access to files. The latter's offering, WebIntegrity, integrates its version control facilities with an authoring tool.

Web Technology Tools

Web tools are marketed for web authors or web developers. As to what constitutes a CCM tool, that is not totally clear and there is no consistency in functionality across the tools. Suitability for large-scale development seems to determine whether it is a CCM tool or an authoring tool. Tools are first generation ones (with respect to CCM support), with only one product (DynaBase) claiming that it provides configuration management facilities. Some tools are geared to large-scale web production although it is not yet clear how scaleable these tools are. Half a million components seems to be the maximum now. Table 4 lists some commercial CCM tools.

If there are any similarities or trends, they would be:

- support for web languages
- command line interfaces
- templates for separating content from formatting
- version control of files
- roll-back of complete sites
- minimal workflow support for publishing authorization
- audit logging, event triggers
- commercial database interfacing
- drag-and-drop component reuse (to minimize programming)
- role support for authorizations
- minimal change tracking
- concurrent site production (for multiple releases).

Some noteworthy features include:

- TeamSite provides visual differencing for examining two ver sions of content side by side.
- Tasks can be assigned to authors using notifications.
- Authors can be notified when content is published on the web.
- Content is moved to a staging area each time it is changed or receives approval to be published.
- Drumbeat gives developers guidance on targeting code to specific browsers thereby providing variant creation support.
- Raveler teams can be set up with pre-configured workflows.
- StoryServer supports static and dynamic versioning.

Overall, more CM support needs to be provided to support CCM needs.

Conclusion

The web environment provides the opportunity to connect

many different resources. Whilst the resultant web systems are easily created, they are complex systems offering many challenges for CCM. We need to understand the problems that companies are having with web systems in order to properly define their CCM requirements. We still need solutions to questions such as:
1) What are good content development and change processes for teams developing large-scale web systems?
2) Are there different processes depending on the type of web system, size of company, volume of web data?
3) Can the types of web systems be categorized into classes or architectures?
4) Will component libraries be indicative of these architectures?

CONTENT TOOL	VENDOR	WEB SITE
ArticleBase	Running Start	www.runningstart.com
DreamWeaver	Macromedia	www.dreamweaver.com
Drumbeat2000	Elemental Software	www.drumbeat.com
DynaBase	Inso	www.inso.com
Frontier	Userland	www.userland.com
FrontPage98	Microsoft	www.microsoft.com
Fusion	NetObjects	www.netobjects.com
Raveler	Platinum Technologies	www.raveler.com
StoryServer	Vignette corp.	www.vignette.com
Team Site	Interwoven	www.interwoven.com

Table 4. *Commercial Web Content Development Tools*

5) What factors affect the definition of the CM process and CM items?
6) Do we need system models, data models, architectures of web sites in order to fully capture the appropriate CM meta-information?

Second-generation web systems will focus on knowledge management and need sound engineering principles such as CM behind them. Given the many challenges, much of the solution will have to be embedded in the tools because the skill set of the developer cannot be guaranteed. This means that the CM processes will have to be implemented in the web tools rather than relying on manual procedures. Along with excellent variant support, change tracking, and change propagation (especially via change sets). CM is becoming an issue for all companies because in order to survive beyond the first decade of the new millenium, companies must place their applications on the World Wide Web.

References

1. International Data Corporation, 1999.
2. Murugasen, S., Deshpande, Y.: *Proceedings of ICSE99 Workshop on Web Engineering.* International Conference on Software Engineering, Los Angeles, USA (May 1999)
3. Merrill Lynch Co., 1999.
4. Bloomberg News: Net Shares Battered Amid Signals That Web's Expansion Is Slowing. *Wall Street Journal* (June 15, 1999)
5. Dart, S: The Dawn of Document Management. *Application Development Trends* (Aug. 1997)
6. Hutcheson, M.: The NT Application That Wouldn't Die (NASDAQ.COM). *Enterprise Development.* 1,1 (Dec. 1998)
7. Sliwa, C: Maverick Intranets: A Challenge for IT. *Computerworld* (March 15, 1999)
8. Dart, S.: To Change Or Not To Change. *Application Development Trends* (June 1997)
9. Gellerson, H. Gaedke, M.: Object-oriented Web Application Development. *IEEE Internet Computing* (Jan/Feb 1999) 60-68
10. Lockwood, L.: Taming Web Development. *Software Development Magazine* (April 1999)
11. Iyengar et al.: Techniques for Designing High-Performance Web Sites. *IBM Research* (March 1999) 17pp
12. Powell, T.: *Web Site Engineering.* Prentice Hall, NJ, (1998)
13. Dart, S.: The Agony and Ecstasy of CM. A half-day tutorial given at 8th International Workshop on Software CM, Brussels Belgium (July 20-21, 1998) http://www.cs.colorado.edu/~andre/SCM8/dart.html
14. Dart, S.: Not All Tools are Created Equal. *Application Development Trends* (Oct. 1996) 7pp http://www.adtmag.com/pub/oct96/fe1002.htm
15. Siegel, D: *Secrets of Successful Web Sites : Project Management on the World Wide Web.* Haydn Books, Indianapolis, Ind. (1997)

Notes

1. Data warehouses provide common interfaces to variant databases.
2. Active Server Page, a combination of static HTML and VBScript
3. Then add in variants for non-PC devices such as pagers, PDAs and smart phones that have "micro-browsers", and the number of variants escalates further.

About the Author

Susan Dart is President of Dart Technology Strategies Inc., an independent consulting firm that helps companies attain configuration management solutions. Dart has 23 years of experience with software tools, development environments, and technology adoption. She has several publications and seminars to her credit, including *Evaluating Configuration Management Tools* (Ovum, 1996), and was a member of the U.S. Federal Aviation Authority committee on developing CM for bomb detection systems. She is a member of the Program Committees for the International Conferences on Web Engineering and on CM.

Previously, Ms Dart was Vice President of Process Technology at Continuus Software Corporation, where she implemented deployment services to assist strategic customers in achieving the best possible, enterprise-wide CM solution. Before that, she spent seven years at the CMU SEI, creating models for CM and evaluating software development environments. Dart has also developed compilers at Tartan Inc. and telecommunications software and international standards for Telstra, Australia. She has a master's degree in software engineering from CMU and a bachelor's degree with distinction in domputer science from RMIT.

Dart Technology Strategies Incorporated.
1280 Bison, PMB 510.
Newport Beach, Calif. 92660. USA
Voice: 949-224-9929
Fax: 949-515-4442
E-mail: sdart@susandart.com
Internet: www.susandart.com