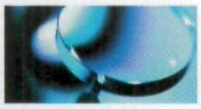

Interrelationship of ECM and SOA — Platonic Union or Wedded Bliss

By Lloyd Dugan

Given all of the hype lately about service-oriented architecture (SOA), I was surprised to find that discussion of SOA at the AIIM 2007 Conference was fairly muted relative to what I expected. It is not just that only a handful of SOA-related presentations were made (including my own), but even the enterprise content management (ECM) vendors on the exhibition floor had little to say.

In a general canvassing of the floor, I found only one ECM vendor showcasing an SOA-based version of its product, while others were only promoting the accessibility of their products' functionality via Web service (WS) calls and/or the capability to invoke external WSs as part of that functionality. It was also stunningly true that the pure business process management (BPM) vendors, the ECM vendor group most likely to have something meaningful to say about SOA, largely stayed at home this year.

Consequently (and perhaps prematurely), I concluded that the nexus of ECM and SOA about which I spoke at the conference was still forming into a critical mass of industry awareness and understanding. In an attempt to further this coalescing, I offer this brief treatment on the nature of the interrelationship between ECM and SOA, extending the metaphor of my earlier presentation by attempting to determine if this interrelationship is, or will be, more like a platonic union or wedded bliss.

A platonic union between ECM and SOA would be akin to two friends sharing a living space and some level of communal responsibilities, which in this case would be the application server (AS) and what it makes available. Whether it is a Microsoft .NET server or a standard Java 2 Enterprise Edition (J2EE) server, these infrastructure-level software platforms perform common functions and services that are used by ECM solutions and SOA technologies, including key application programming interfaces (APIs) for WSs and eXtended Markup Language (XML)-based messaging. With the platform vendors, particularly Oracle (with it acquisition of Stellent) and IBM (with its acquisition of FileNet), this aspect of the relationship can only get more cozy as acquired products become even more tightly integrated with the platforms. However, ECM and SOA remain roommates with different personalities and interests.

ECM is software that consists of functionality packaged into point solutions or integrated solution suites. ECM is primarily focused on supporting workflow patterns (WPs), such as synchronization or arbitrary cycles that accomplish human-centric or document-centric tasks. Documents are managed or referenced in native format or as binary large objects (BLOBs) as they are moved through the workflows and XML is used to manage the metadata. Native ECM Web clients that advance the workflow provide rich access to ECM server functions and robust process activity reporting capabilities, while customized Web clients require additional development, typically including API-level calls to the ECM server.

BPM products specialize in enabling such workflows. This type of ECM product typically offers a modeler tool that supports both some level of analysis of the processes being automated and the configuration or development of script for implementing the behavior of modeled components. The modeled design-time constructs convert into run-time elements, providing strong traceability between model and application.

SOA, on the other hand, is software that consists of technologies that can be configured to supply desired functionality. SOA is primarily focused on supporting Message Exchange Patterns (MEPs), such as request/response or fire-and-forget, that enables the integration between disparate systems, and the sharing and synchronization of data. Informational content is exchanged via XML payloads embedded in Simple Object Access Protocol (SOAP) messages. No native Web client exists, though any Web client that supports WSs can be used. Process activity reporting is essentially limited to what the SOA suite running on the AS can do natively.

Business logic for processes is implemented via server objects, such as Enterprise Java Beans (EJBs) or Business Process Execution Language (BPEL) processes, which require an integrated development environment (IDE) tool to support development. Though not really a modeling tool, an IDE, such as Eclipse, will use a diagramming metaphor to facilitate development and directly support conversion of design-time constructs into

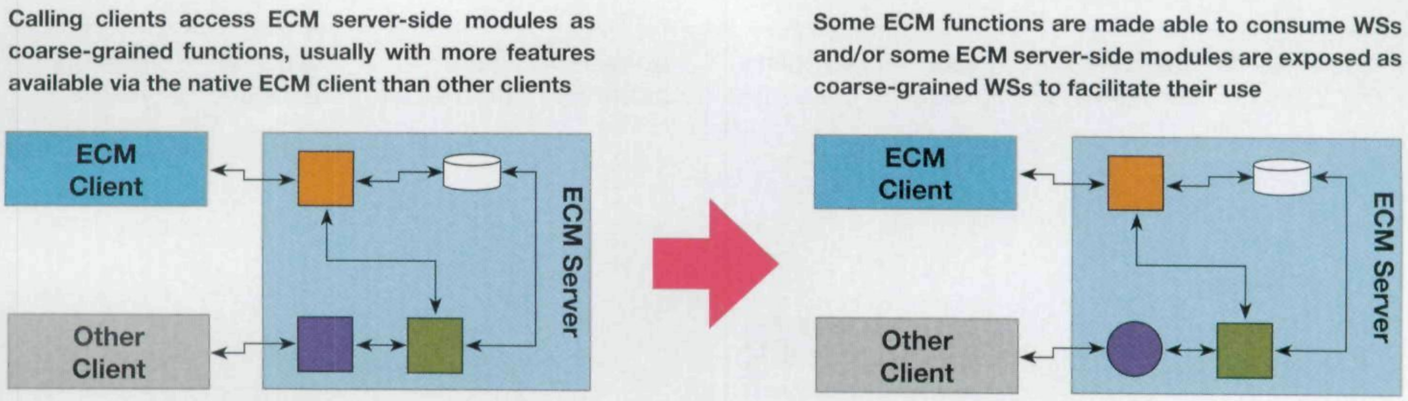
Calling clients access ECM server-side modules as coarse-grained functions, usually with more features available via the native ECM client than other clients
ECM Client
Other Client
ECM Server
Some ECM functions are made able to consume WSs and/or some ECM server-side modules are exposed as coarse-grained WSs to facilitate their use
ECM Client
Other Client
ECM Server

Figure 1. Micro-Evolution of ECM Solutions towards Using or Incorporating Web Services

run-time elements. However, with the advent of the enterprise service bus (ESB) and the business rules engine (BRE) as parts of the SOA suite, outright development of EJBs or BPEL processes is increasingly giving way to wizard-driven configuration of the ESB and BRE, with little if any scripting needed, as the means of effecting this conversion.

Wedded bliss between ECM and SOA would be akin to a married couple sharing various intimacies and a high level of communal responsibilities, which in this case would be the infrastructure layer and how it would be used. For example, ECM could use MEPs to enable WPs or WPs to orchestrate MEPs, all running at the infrastructure service layer. Principally, this means using and incorporating WSs, which, while not the only way to implement an SOA framework, are nonetheless the most adaptable and extensible approach, and are based on standards as opposed to relying on largely proprietary APIs.

As the myth goes, happily married couples start to look like each other over time. ECM solutions are still mostly distinguishable from SOA technologies. Before the ascendance of WSs, an ECM product was a self-contained system architecture that housed packaged functionality along with tightly controlled API-level access to those functions. Nowadays, most ECM products have been reengineered to be able to consume WSs and/or to expose some functionality via WSs rather than via traditional means of API-level calls. The former does not require much change in the system architecture other than the ability to call a Web Service Definition Language (WSDL) file, while the latter requires that a WSDL file be created for the exposed functionality, which is merely just another way to implement a remote procedure call (RPC). While being promoted as revolutionary by the ECM vendor community, this change is more accurately described as mildly evolutionary, and is illustrated in Figure 1 above.

For ECM and SOA to really start to look like each other, ECM must follow SOA into the infrastructure layer. This requires a radical re-thinking of the ECM system architecture, shifting away from packaged functionality towards reusable and extensible services. It requires that ECM server-side modules with coarse-grained functionality—meaning large, highly coupled functions that do not effectively decompose and are inaccessible except at limited service end-points—be examined for embedded, overlapping operations that can be redesigned as smaller, less coupled WSs with corresponding WSDL files. The risk here to the ECM vendor is that revealing the inner workings of the big "black box" as a series of exposed smaller "black boxes" may give away significant intellectual property (not to mention likely forcing a cannibalization of the installed base). In any event, for ECM to become more SOA-like, a major evolutionary change is needed, and is illustrated in Figure 2 below.

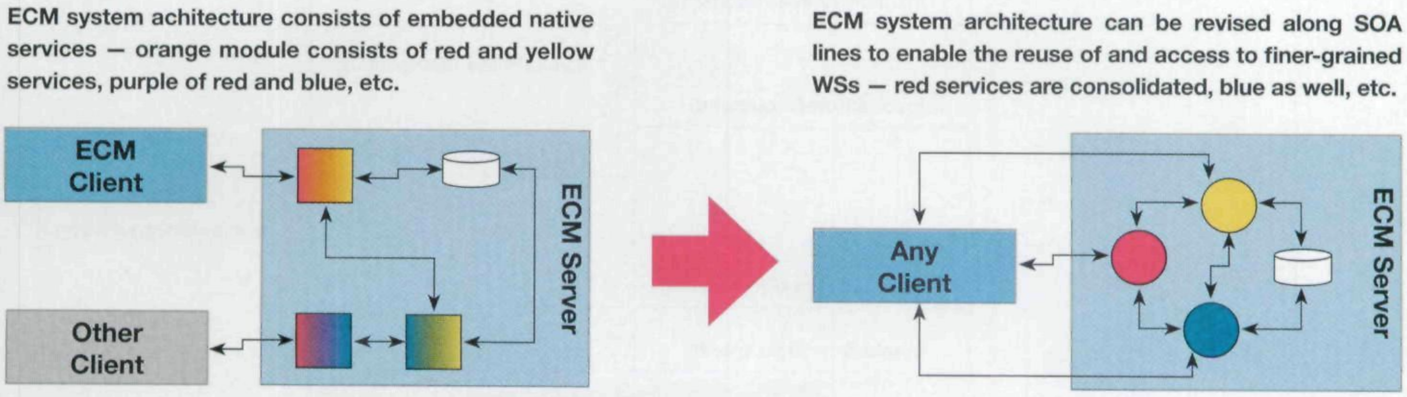
ECM system achitecture consists of embedded native services – orange module consists of red and yellow services, purple of red and blue, etc.
ECM Client
Other Client
ECM Server
ECM system architecture can be revised along SOA lines to enable the reuse of and access to finer-grained WSs – red services are consolidated, blue as well, etc.
Any Client
ECM Server

Figure 2. Macro-Evolution of ECM Solutions towards Tighter Integration with SOA

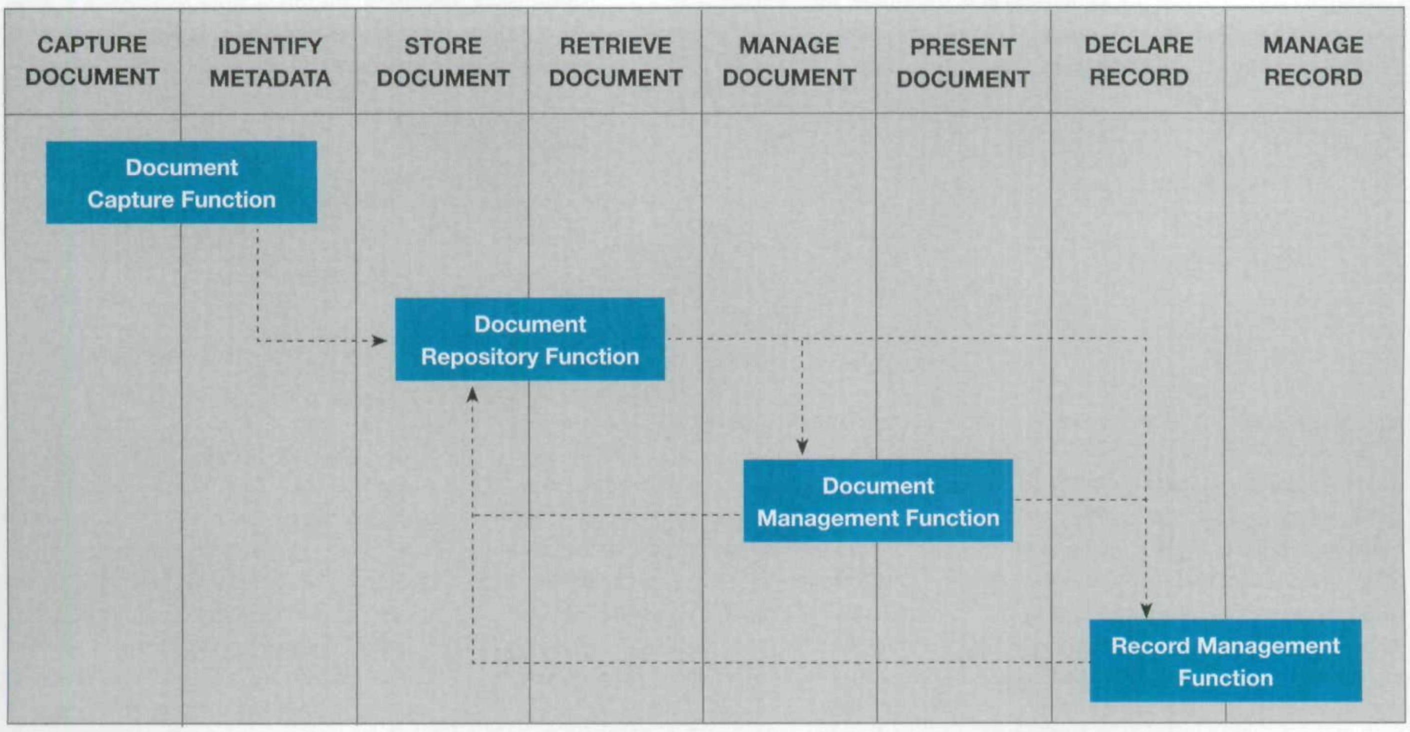

Figure 3. Packaged Functionality Model for ECM Solutions

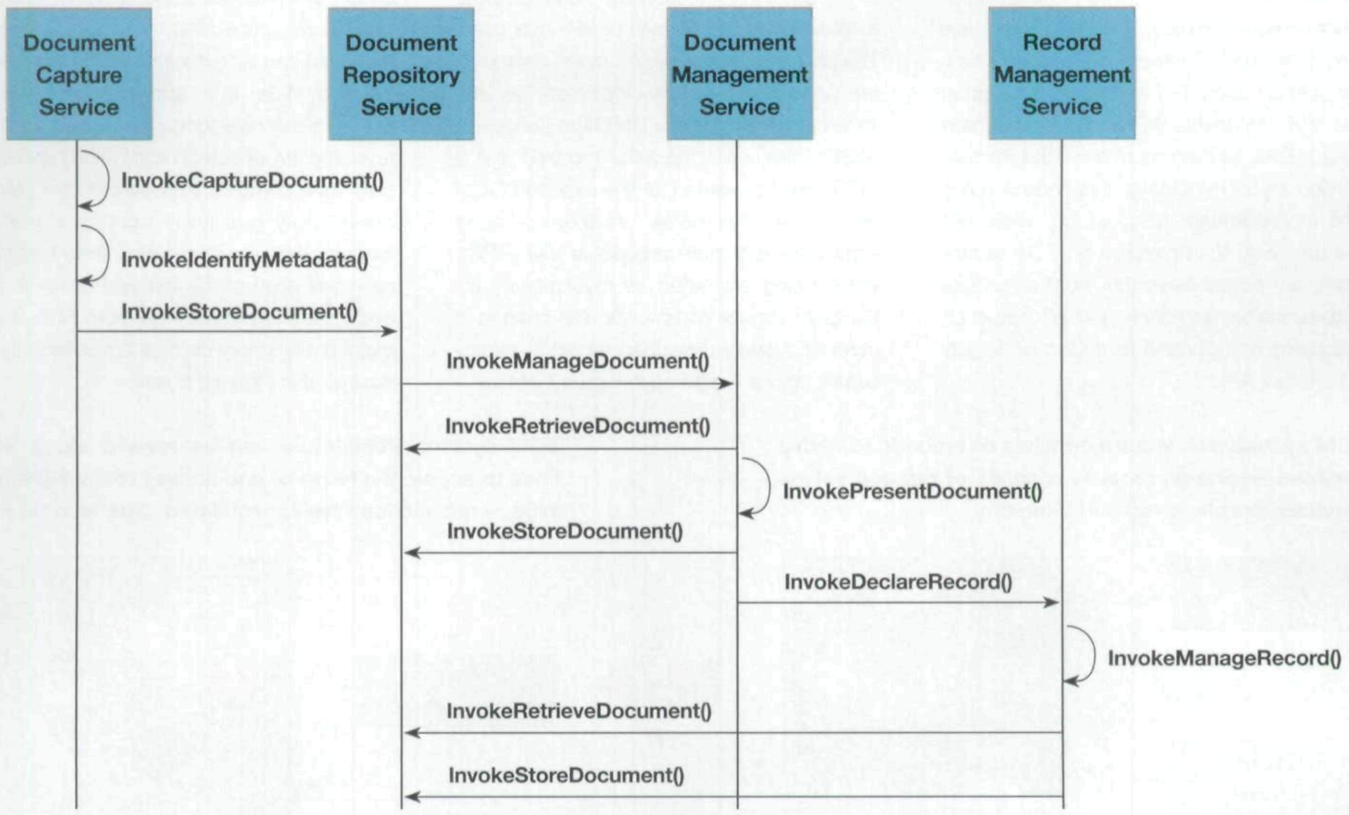

Figure 4. SOA-Based Model for ECM Solutions

Another way of looking at this shift is to see how the packaged functionality model and the SOA-based model provide a typical (and, for this example, simplified) set of ECM functions for supported processes. The packaged functionality model is illustrated in Figure 3 above, and shows how a suite of components from a single ECM vendor or components from different ECM vendors might be integrated together.

Now compare how this same set of functions might be provisioned instead as WSs with corresponding operations supporting the business processes, which can best be represented using Unified Modeling Language (UML) sequence diagrams. Each instantiated WS maps to a WSDL file and the listed operations map to specific functionality available within the WS. Service endpoints exist at multiple locations, such that any of these WSs are accessible and can be invoked by any authorized calling client. This SOA-based model for ECM is illustrated in Figure 4 above.

In conclusion, I would venture to say that, for now, ECM and SOA constitute a platonic union that grows increasingly more comfortable as time passes. Technology innovations and predatory market forces are combining to help move this relationship towards something potentially much more intimate. Perhaps we may yet see the day when ECM is really just another set of infrastructure services, which is where I think they rightly belong. On the other hand, if familiarity breeds contempt, we may find some ECM vendors resisting the moves to be more like SOA and remaining as software products rather than becoming services—in other words, ECM and SOA just remaining friends.

Lloyd Dugan (lloyd.dugan@ie-services.com) is senior project director/CTO of Information Engineering Services, Inc.

focus on...compliance *focus on*...financial services *focus on*...government *focus on*...healthcare

aiim *focus on...* **Compliance**

COMPLIANCE. Where do you start? What should you do first? How long do you need to retain paper and electronic information? Because existing regulations are revised and new ones created continuously, compliance is a moving target--a company is never finished with a compliance initiative.
www.aiim.org/compliance

aiim *focus on...* **Government**

Effective e-Government means state, local, and federal government agencies must provide for the creation, management, and storage of vast amounts of content and documents, often distributed across various agencies and departments, while meeting security, regulatory, and privacy standards.
www.aiim.org/government

aiim *focus on...* **Financial Services/ Insurance**

The use of forms (paper and electronic) in the financial services industry (banking, insurance, and investments) isn't likely to decrease anytime soon. Increasing the efficiency of how you process structured and unstructured content can greatly improve your business workflow and customer service.
www.aiim.org/finance

aiim *focus on...* **Healthcare**

As a healthcare information management (HIM) professional or practice management executive, you need better ways to control your practice's business content (paper documents and electronic information) to improve patient care. Electronic Medical Records (EMR) can help improve patient care thru streamlined processes.
www.aiim.org/healthcare

focus on...white papers *focus on*...case studies *focus on*...research *focus on*...news & Information