Information Architecture 101: A crash course for the enterprise architect

Almost everything you ever wanted to know about information architecture

Thomas Myer

May 23, 2006

What is information architecture? What roles do information architects play in the enterprise space? Are information architects content experts, usability experts, database administrators (DBAs), Extensible Markup Language (XML) analysts? Or all of the above? How can understanding the concepts behind information architecture help your enterprise project succeed? Take this crash course on information architecture to find out.

What is information architecture?

Picture the scene: I'm mingling at an after-hours networking event at an IT conference, drink in hand. People ask me what my company does, and I tell them, "Web development, XML consulting, and information architecture." When they ask me what information architecture is, I'm a bit stumped.

Why? It's not that I don't know what I do; it's just that I've never met two information architects that share the same job description. A lot of information architects came up the way I did, through the world of journalism or technical publications, adding expertise in Standard Generalized Markup Language (SGML) or XML along the way. Others are experts in database design. Still others come from the usability camp. A new crop of information architects are emerging from schools of library science.

So, first things first: What is information architecture? The Asilomar Institute for Information Architecture (AlfIA), an information architecture (IA) industry body of practitioners, states that IA is "...the design of shared information environments," especially as it relates to Web sites, intranets, online communities, and software. Louis Rosenfeld, a chief practitioner of IA, talks about IA being the overlap of users, content, and context. But what do these definitions mean to you, the enterprise architect?

Typically, during any development project, there is an area where the needs of the users, the structure of the content, and the needs of the organization overlap. It's in this small but very critical
niche that an information architect makes their home. Within this niche, there is a great deal of differentiation; nevertheless, the niche is easy to identify once you know what you're looking for.

For example, say that you're leading an effort to build an intranet search engine for a company with 10,000 employees. The goal of the project is to allow easy access to the hundreds of thousands of documents, files, Web pages, and database records scattered across hundreds (if not thousands) of servers on the global corporate intranet.

Most projects of this scope end up on one of three paths: build from scratch, buy, or acquire/modify. These paths are meaningful not only to the developers on the project (they may be open source advocates who don't want a proprietary system), but also to business stakeholders (both business unit managers and the bean counters). However, all three of these paths are equal in the eyes of the information architect. Typically speaking, the information architect is not concerned with infrastructure, except when they know that the proposed solution won't meet the user's needs or achieve the goals of the business.

**What does the information architect care about?**

Instead of focusing on typical IT problems, the information architect comes to the project with a threefold focus:

1. **Users of the information.** A whole host of questions spring to mind when you consider users. There is no such thing as knowing too much about the user. For example, an information architect might ask: Who are they? What roles do they play in their organization? Are they experts or laymen? Decision makers or worker bees? Where do they work? What kinds of information are they seeking? How do they use that information? What formats are easiest for them to handle? Do they search or browse? What criteria are they searching on? What languages do they work in?

2. **The information itself.** Information architects organize, label, and chunk information. They construct navigation systems that make the information easy to find. They add metadata to the information so other systems can process and understand it. To accomplish these goals, they have to run both top-down and bottom-up discovery sessions. **Top-down discovery** involves getting a picture of the entire information space and working down into the details. **Bottom-up discovery** is all about figuring out metadata for each piece of content and working up toward the general.

3. **The business or organization.** My mentor in the world of information architecture had an accurate, if non-PC (non-politically correct), way of describing the role of an information architect in relation to the business. It is, simply put, to remind everyone that "technology is the handmaiden of policy." In other words, technology isn't just another cool toy that you spend money on just to say that you have it. It's there to serve the needs of the business, and the minute you forget that, the business suffers. I submit most of the 1990s as a case study.

In short, the information architect has to have good information about users, information, and business goals. If that information isn't present or is incomplete, the architect's job is to fill in the gaps in the team's knowledge. They accomplish this gap-filling with a variety of tools and approaches.
Why bother? According to the 10th edition of the annual CHAOS report from The Standish Group, almost 70 percent of all software projects fail. (See the Related topics section.) The two main causes of failure are lack of end-user adoption of the new software and unclear, ambiguous, uncommunicated, or nonexistent requirements. If you don't ask the users what they want and how they do their tasks, you end up building systems that don't work. If you don't get clear direction from the business stakeholders, you end up with cost overruns and failed projects. And who ends up taking the blame every time? The hapless development team.

What does an information architect do?

If you ever watch shows like CSI, you know that the crime scene investigators have one goal: Look at the forensic evidence at a crime scene and figure out not only "whodunit" but "howdoneit." During the course of their work, they employ lots of different tools, methodologies, and techniques. Sometimes they look through a microscope or compare fingerprints in a database, or use high-tech lasers and gases to identify a piece of trace evidence. Other times they talk to people, walk the area surrounding a crime scene, or observe patterns in movement.

Although it isn't quite the same with information architects (you wouldn't want them at a crime scene), one thing is true: They have an overarching goal of making an information space usable, and, like the forensic investigators, they have many different ways of figuring out how to go about doing that. I think of these techniques as fitting one of four main areas of inquiry, which I've labeled ask, observe, learn, and iterate.

Ask

The tools under ask involve asking questions either directly or indirectly. One of the best ways to organize information is to ask users to do a card sort. In a card sort, the information architect prepares a list of twenty to eighty concepts that relate to the information space. The information architect then randomizes the list and presents it to a group of users, one at a time. The users put like things together, label the groups, and thus provide the information architect with much valuable knowledge.

Information management products from IBM

DB2® Information Management is the IBM software solution for information management. At its core is a powerful family of relational database management system (RDBMS) servers. In addition, IBM has built information management software with DB2 as its "engine" for data warehousing, data analysis, data mining, media asset management, enterprise content management, and information integration. Explore the collection of IBM information management products and tools available to help you build an architecture that meets the information needs of your business.

What rules do the information architect follow during a card sort? It's bad form to argue with any participant in a card sort exercise. Encourage the participant to think out loud while they're doing the card sort, but never argue with them and never help them with the concepts, even if they ask. Just keep reminding them to do their best and that there are no right or wrong answers.

Once you've collected even a small sample of data, you can start cluster analysis. The point of cluster analysis is to calculate the distance between any two concepts. The result is a cluster graph that shows how closely concepts tie together. In the old days, you had to do this work by
hand. IBM's Ease of Use group created two tools, EZCalc and USort, which reduce the drudgery of creating card stacks and then doing the cluster analysis. This software is no longer available from IBM, but my company has agreed to make it available to the IA usability community.

Another important technique under *ask* is the survey, whether printed or online. A good survey, distributed to the right audience, can yield an incredible amount of information and insight about user needs, user-interaction models, and areas of interest or potential danger. It's not a bad idea to run a user satisfaction survey before and after your implementation to measure success rates. In the intranet search engine example, asking users how satisfied they are with search both before and after your project launch can provide valuable information for any project postmortem. If users are still unsatisfied, it's an opportunity for the team to figure out what went wrong and make improvements.

**Observe**

Observing is all about being unobtrusive, to sit in the background without drawing attention to yourself. Think of anthropologists on a field excursion to a totally alien culture deep in the jungles of South America. The anthropologists are not there to judge, offer advice, or correct behavior. They are there to observe, record, and experience.

Why is observation so important? Because you can *ask* users what they want in a new system or how they go about doing their jobs, but most people internalize their work and forget things. They forget that in between steps A and B they perform six other minor tasks outside the system. *A good information architect knows that there is always a gap between what users tell you they do and what they really do.* This isn't because users are maliciously trying to trick us; they just forget.

If you send out a survey, you get lots of great information. What you don't find out is what really happens -- that the network is always down, so people phone in their orders instead of going to the Web or that they always call in their requests for sales kits instead of using the online form because they like talking with the support staff. If you don't take the time to *observe*, you have a blind spot in your understanding.

**Learn**

Interacting with users by *asking* and *observing* is important, but so is doing lots of research. Sometimes the research is extremely directed. If your users' primary focus is on fast retrieval of information, is database A or database B your best bet? If you go with database A, do you have enough developers and administrators to support it? Furthermore, can database A support all the different metadata you need to allow all the different types of searches?

Other times, the information architect's research might range over a vast territory. For example, let's say you are involved with a project involving bilingual and bicultural audience members. General research topics may include how bilingual members of this population use online resources, which character sets and encoding need to be supported on the back end, what colors and layouts fall within cultural norms, and what labeling systems make the most sense.
Iterate

You can ask users all day long, you can observe them for weeks, you can even do your own research to learn, but, eventually, the only way to really achieve success is to iterate. This group of tasks involves building and refining an effective prototype. Why? Because users don't know what they want -- not really, anyway -- until they see it. Lack of prototyping on the front end almost always involves costly recoding at the end of the project.

Does iterating require full-blown prototypes built in Macromedia Flash or 3-D modeling software? You can certainly go that route -- it's referred to as high-fidelity prototyping. In other words, the prototype has a high fidelity to the real thing. However, the route that's better for everyone adheres to the core of iteration -- fast cycles of output and correction.

Think low fidelity. Think starting out in a room with a whiteboard and drawing simple boxes, circles, and arrows. Adjust it as you get feedback. Let the customer play too; if they start drawing boxes and arrows, they feel ownership. Eventually, turn that diagram into a Microsoft® Office Visio® wireframe, then into a medium-fidelity image created in Adobe Photoshop. Or, use a rapid-development framework, such as Ruby on Rails, to create working code in minutes. What's the difference? The wireframe shows a plain rectangle for that sidebar containing associated links. The Photoshop image shows that sidebar with that beautiful background color that marketing wants. The Ruby on Rails prototype actually lets users click and see the results of their interaction.

The more often you iterate, the quicker you get to the heart of the matter -- what users want. Eventually, you can put together a prototype that looks -- and behaves -- the way its users want. For example, when they perform a search on a new intranet search engine, users may want to see how many results would be returned for a query along with an opportunity to further narrow those results before seeing results displayed. You would almost never figure out this requirement if you simply asked users -- they either don't know or can't visualize it. But once they see something in front of them, they are able to say what they want.

Summary

You've covered a lot of territory in this article, and there's still a world of information out there about information architects and what they do. Now you have the necessary background to find out more. If you don't have an information architect on your enterprise IT project now -- get one. Their expertise and knowledge can help you keep an eye on what has traditionally been a huge blind spot for business: the convergence of users, information, and business needs.
Related topics

- Adaptive Path's [user experience case studies](#) provide insight into the process of IA.
- [BoxesAndArrows.com](#) features hundreds of articles on a wide variety of information architecture, usability, and user experience topics.
- "[Explore the different approaches to information management in SOA](#)" (developerWorks, June 2005) demonstrates how to leverage the power of information management for Service-Oriented Architecture (SOA)-based modeling, architecture, design, and implementation.
- [Louis Rosenfeld](#) is a top practitioner of IA and coauthor of the seminal Information Architecture for the World Wide Web published by O'Reilly (a.k.a. the Polar Bear book).
- "[Standish: Project Success Rates Improved Over 10 Years](#)" on SoftwareMag.com discusses the 10th edition of the annual CHAOS report from The Standish Group.
- "[Use Rational Data Architect to integrate data sources](#)" (developerWorks, March 2006) shows you how the IBM Rational Data Architect tool helps you gather a complete understanding of all data sources, understand how they relate to each other, and federate the data sources into a virtual schema.
- Download [a free trial version of IBM DB2 Universal Database™ Enterprise Server Edition](#), designed to meet the relational database server needs of mid- to large-size businesses.
- Download [IBM DB2 Express-C](#), a no-charge version of DB2 Universal Database Express Edition for the community. DB2 Express-C offers the same core data server features as DB2 Universal Database Express Edition and provides a solid base to build and deploy applications developed using C/C++, Java™, .NET®, PHP, and other programming languages.
- Download [WebSphere® Information Integrator](#), which provides the core capabilities needed by business integration and business intelligence applications -- access to diverse and distributed data as if it were a single data source.
- Download EZCalc and USort from the [Triple Dog Dare Media](#) Web site.
- The [tdog.blog](#) covers information architecture and usability.

© Copyright IBM Corporation 2006

[Trademarks](http://www.ibm.com/developerworks/ibm/trademarks/)