The Rise and Fall…and Rise Again of Information Architecture: An Interested Observer's Perspective

Presented by Bob Royce, President, The Understanding Group LLC
September 23, 2011 at the Euro IA Summit VII in Prague, Czech Republic

Good Morning, today I plan to give you my perspective on the history and future of information architecture, not as an inside practitioner, but rather as an outsider. Although my whole career has been spent in disciplines with strong analogs to IA, and I was in the first digital library program at the University of Michigan with Lou Rosenfeld, Peter Morville, and Joe Janes that birthed the field as we now know it, never-the-less I chose paths that kept me outside the mainstream until this past March when I co-founded an information architecture consulting practice called The Understanding Group with Dan Klyn. While my journey may give me an interesting perspective, I recognize that it's likely that much of what I'm going to say may be controversial, and is probably at best, only partially right. Thus my goal here is not to pontificate, but rather to provoke discussion around topics that I think add value to the field. I welcome feedback and even pushback.

Context

To begin, I want to start by establishing some context for what I mean by information architecture. I understand that this is a prickly topic, this idea of defining things, but as an outsider I think it's important to clarify how I've come to see Information Architecture from the outside looking in.

Let's start with information. James Gleick's recent book provides a fascinating overview of the history of information. He describes it as coming in three waves: Historically, for thousands of years, people created and exchanged information but paid little attention to measuring it; Then Claude Shannon came a long and created mathematical theories about information, which then opened the door to the flood, in which we now live.

The historical parts of Gleick's book are entertaining and interesting, but for our purposes I want to focus on the work of Claude Shannon, pictured here. Fundamental to Shannon's theory of information is the idea that, at least as far as information engineering is concerned, "meaning is irrelevant." Shannon dealt primarily with issues of encoding and signal processing. I think this idea of "meaningless" information has caused confusion regarding how information architecture has been defined. While information architects, as we know them today, deal intimately with meaning, the first use of term "information architecture" arose in the 1960s within the context of Shannon's "meaningless" information processing. Because of this, we tend to dismiss these early efforts as something of a homonym—the same words, different meaning. Yet, as I'll discuss later, there were other streams within the disciplines of computer science which were operating in the spirit of what we practice today, but they weren't called information architecture, so they don't appear in any "history of IA" that I'm aware of.
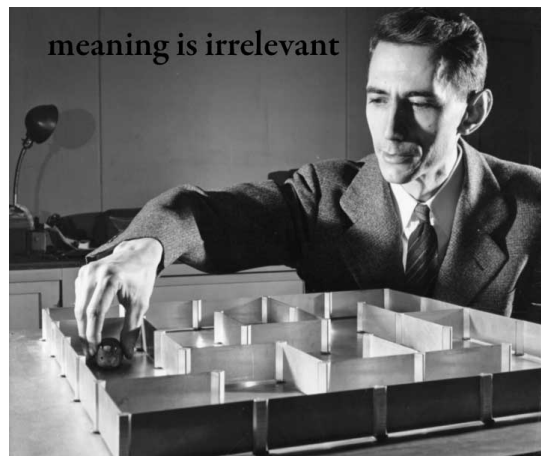
So while computer scientists tend to deal with data issues in the vein of Shannon, people that use the systems they create certainly do care about the meaning of the data. Which brings us to the flood, the deluge of information we now face.

Meaning, of course, is important to those receiving the information, but we live in a world much like Jorge Louis Borges' Library of Babel that contained all the knowledge of the world...somewhere. We often despair of finding meaning in the midst the deluge of information, or rather, in the deluge of data parading as information that comes our way every day.


"Ours is a world about which we pretend to have more and more *information* but which seems to us increasingly devoid of meaning."

Left: *Erik Desmazieres: La Salle des planetes, from his sieries of illustrations for Jorge Luis Borge's "the Library of Babel' 1997–2001.*

Jean-Pierre Dupuy

So while it's taken me a while to get where I am today professionally, I am quite glad to have started an information architecture company. Although until recently I've been an outsider to the field, I've always cared quite a bit about the same kind of things an information architect cares about. In particular, my hope in starting "The Understanding Group" (aka TUG) is that I can help people navigate the world of information and help them structure information to deliver real value.

To TUG the idea of IA is rather broad: Like building architects did for the industrial age, creating plans and blueprints for turning raw material into structures which enable us to work, play and live, information architects work in the "information age" and design, develop and plan structures and blueprints for digital spaces. We distill the essence of IA down to three core concepts:
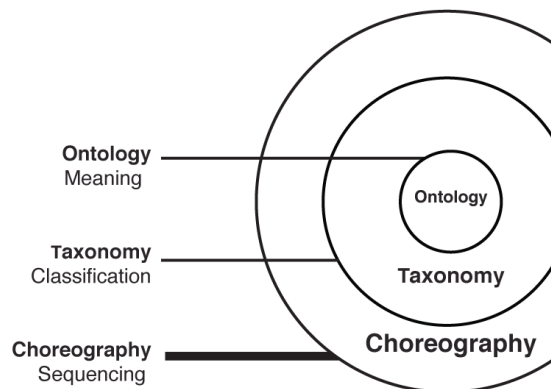
Increasingly, we've started talking about IA less as a thing we do, and more as way we see things, a lens applied to all that we do.

How Did We Get Here?!

With this framework of definitions, I want to now walk through how I've processed what's happened to IA historically, and why I'm so optimistic about it's future, which brings me to the genesis of this talk.

A month after starting TUG, Dan Klyn and I went to Denver for the IA Summit. It was a great event. The highlight was Michael Atherton's talk, Beyond the Polar Bear, which I was inspirational to this talk in how it applied the concept of domain modeling to information architecture.

The low point of the summit came toward the end as I walked into the discussion on plans for 2012. I arrived in the middle of a discussion about what to call the summit. There was serious discussion that the name should be changed to the UX Summit. I found this to be very confusing. At this point, as an outsider, I had heard very little about User Experience as a discipline. I was familiar with the concepts back in the 1990's when Donald Norman first talked about them, but I couldn't figure out why you'd name a summit after it.

What had happened to the idea of "information architecture"? What is this UX thing? Did I just start the wrong kind of company? After I calmed down, I stepped back and did some research, talked to people in the field, and started formulating the hypothesis I'm about to present.

Let me say to start that as an outsider I feel there is unnecessary confusion around the terms. To me, at least, it's quite simple: IA and UX are two fundamentally different disciplines that address the same issue from different perspectives. Both are working to facilitate a relationship between two parties, two users, if you will: the website, and the people behind the website, on the one hand, and the person using the website (or mobile device or book or what have you) on the other. User Experience is focused on the point where the two parties meet, while IA is focused on helping the "people behind the website" organize their thoughts, so to speak, so they have the best chance possible to build the kind of relationship they want with the people coming to the site.

Both disciplines are essential for success—*but they're not the same thing.*

I think there is great opportunity for us to broaden our area of focus in IA. We see this starting to happen in the writings of Peter Morville and his Ambient Findability and Resmini and Rosati's Pervasive Information Architecture, but today I'm going to propose another step which fuses together the three streams of IA that have emerged over the past 60 years or so.

To help frame what I mean by this, let's take a look at how Resmini and Rosati frame the history of IA. Here we see the three streams identified as information design, referring to the work at Xerox and also folding in Wurman, the information systems approach, which tended to look at things more along the lines of Claude Shannon's "meaningless" data and then Rosenfeld and

Morville's librarian approach. They then extend the timeline, referring to Rosenfeld and Moreville as "Classic IA" and propose a new extension called Pervasive and Ubiquitous IA.
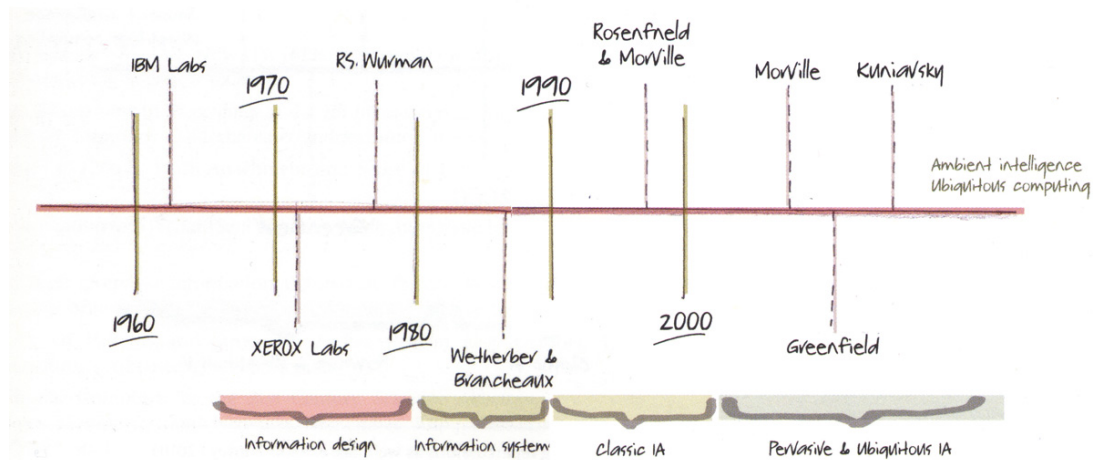
The History of IA adapted from Resmini and Rosati's "Pervasive Information Architecture"

I basically agree with the right side of their timeline, but I want to propose an alternate perspective on what came before which I think might broaden the field of history from which we can draw ideas.

What I propose is that the most significant contribution that the Xerox team made to IA was related to their role in developing a new way of programming computers and that in hindsight, this new programming technique, called object oriented programming, shared many concepts with what Richard Saul Wurman was trying to accomplish. To make this connection, I also want to propose that what Wurman was doing was not "information design." Rather, what both Wurman and the Xerox team were trying to accomplish, each with the palette of tools they had available to them, was the same thing Lou and Peter set out to create, again with the tools available within their field at the time.

What I hope to show then, is that we are now at the point where all three sets of tools are available to us as information architects with the digital realm and it behooves us to work to combine all three into our practice.

To do that, let me take a short step back and provide you a quick thumbnail of my background so you'll see a little of what is feeding into these ideas. To the extent that IA is about organizing, classifying, and labeling, I've been an IA since I was about 7 or 8 when I started collecting rocks. Baseball cards, stamps and coins soon followed. I loved to collect things. Then in high school, I was editor of my Yearbook where I learned about layout, wireframes, and press type. A couple of years after heading to college, Apple introduced the Macintosh computer. I fell in love and quickly began to apply what I learned in high school to producing publications using MacPaint and MacDraw at first, and eventually graduating to PageMaker, Quark Xpress, Illustrator and Photoshop. Though I earned a degree in Education, I only taught for a year, and went on to work a series of jobs combining graphic design, technical writing and marketing skills.

Of particular relevance to my talk today was work I did for Apple computer writing course materials for educating software developers on the workings of the Macintosh. Apple used object-oriented techniques both in its operating system and in its MacApp development tool. This was my first exposure to this powerful way to view the world of software. I was also exposed to one of the earliest hypertext systems, Hypercard, helping to publish a newsletter focused on this innovative new tool.
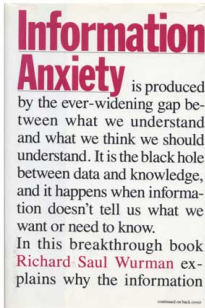
Then in 1989 I picked up a copy of Richard Saul Wurman's Information Anxiety. Up to this point, most of my work had been for technology companies where I was always the "explainer", the one whose job it was to try and distill what we did for others, whether that was through course materials, technical manuals or marketing materials, so I really fell in love with the concepts RSW put in his book. I resonated with the idea that you could use the tools of typography and layout, not primarily to "design" some to look good, but rather to explain something, to promote understanding, or in the words of Wurman to make it "be good."

At the same time, the world of desktop publishing was maturing and turning into a commodity skill and I realized that I was in a limited career. So I started casting about, looking for a career change, one that combined my background in education, computing and design. I learned an important lesson at this point, one that Mr. Wurman mentioned in his curmudgeonly talk to the IA Summit in Phoenix—as technology advances, your expertise in tools is ephemeral. Either the technology changes, or just as likely, it gets to be so easy to use that no one needs an expert. So If your love of IA has to do "how" you do your job, whether it be using a particular technique or a technology, then you should be prepared to experience disappointment—your job is already on the verge of commodification.

Fortunately for me, around 1992-1993, James Duderstadt, the president of the University of Michigan thought this Internet thing was going somewhere and ran an experiment. He asked Dan Atkins, the Dean of the College of Engineering, to become Dean of the university's Library School, renamed the School of Information. So in the fall of 1993 I was in a new masters program focused on Digital Library program with Lou Rosenfeld, who was the enrolled in the PhD program, and taking classes from his future business partner, Joe Janes. It was an exciting time. The World Wide Web was in its infancy and Michigan was a great place to study and dream about what this new network could become.

I remember the early tensions that existed between the two "schools" of thought that were percolating there. For thousands of years, Librarians had developed tools and techniques for managing large amounts of information: cataloging and archiving systems, systems of authority, expertise in abstracting and labeling. The computer scientists, meanwhile, were a bit arrogant. "What's the big deal? All this problem of millions of documents needs is a good relational database. Problem solved. It's only data." Eventually, though, even the computer scientists realized that a search that returns a list of 600 documents is pretty useless if you can't figure out how to make the first 30 or so items the most relevant to the query, Something they didn't know how to do at the time. Why? Because relevance implies meaning, and few of them had spent much time learning about how to manage and extract meaning. You can't blame them. Even today it's hard to do and requires a lot of computing power.

It was also the case that few people paid much attention to Wurman's ideas about creating understanding by architecting information on a page. The visual palette was just too sparse to do much and it was further hindered by the sacrosanct concept of separating structure from display that was embedded in HTML. This separation made sense and it simplified the computing issues to allow some great things to be done, but in hindsight it made it easy to ignore some pretty valuable concepts.

The Rise of Information Architecture

It was out of this milieu that Argus was formed and the Polar Bear book was written. Lou and Peter did a great job of distilling down the fundamental concepts of this "new" field of

Information Architecture. It was big hit and they sold a lot of books. After all, you really can't build a usable website without giving thought to these issues of organization, labeling, navigation and structure. At the same time, the Web was blooming and people were talking about a new paradigm that would change how businesses operate forever. Brick and mortar were dead... long live the Net...yada, yada, yada. It's important to note here the things Lou and Peter pointed to as NOT IA in the 2nd edition of their book. In particular: It's not graphic design, and it's is NOT Software programming. Both of these boundaries made sense at the time, and more or less still do, but I think the box they put around IA through these limitations needs re-examining.
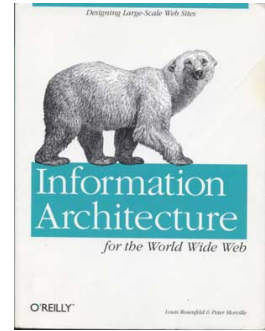
Now I have a confession to make—I dropped out of the Digital Library program about 2/3 of the way through. I loved the program, but to pay my bills I had taken a job with a custom software development shop. By the end of my third semester, I was Vice President of Marketing for Arbor Intelligent Systems (AIS), which had grown from five of us to 15 people, and I figured I had made the career transition I was looking for.

At AIS I witnessed first hand the "Inmates Running the Asylum" that Alan Cooper wrote about. *Homo Logicus*, the data focused programmer, will always focus primarily on the data, ignoring the meaning. Reading the book as I prepared for this talk, I was struck by how real the problems still are. I was also struck by how he described the solution. In describing what he calls the role of the interaction designer, the language he uses evokes that of an architect.
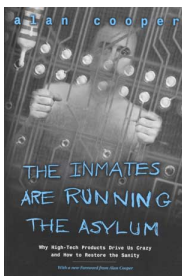
"Architecture— the human design part of programming, in which users are studied, use scenarios are described, interaction is designed, form is determined, and behavior is described..."

Now, recall how Lou and Peter had explicitly said the IA is NOT software programming, yet here Cooper describes a set of tasks which sounds a lot like what information architects do.
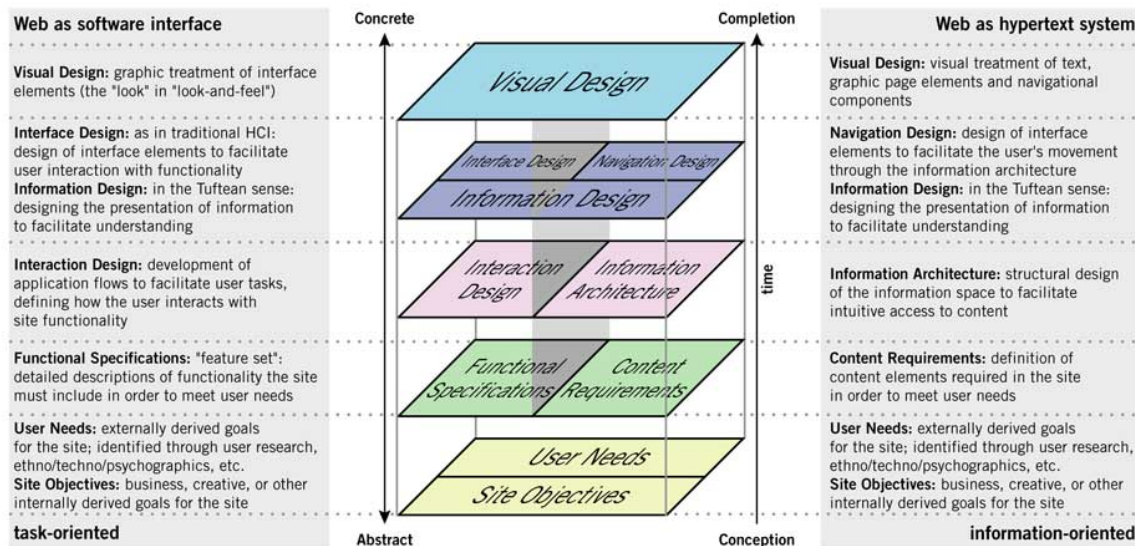
The Decline of Information Architecture

The schism probably made sense at the time. In fact, Jesse James Garrett highlights the schism, while also relating the two sides together in his well-known diagram "The Elements of User Experience."

The Elements of User Experience

Jesse James Garrett
jjg@jjg.net
30 March 2000

A basic duality: The Web was originally conceived as a hypertextual information space; but the development of increasingly sophisticated front- and back-end technologies has fostered its use as a remote software interface. This dual nature has led to much confusion, as user experience practitioners have attempted to adapt their terminology to cases beyond the scope of its original application. The goal of this document is to define some of these terms within their appropriate contexts, and to clarify the underlying relationships among these various elements.

Web as software interface		Web as hypertext system
Visual Design: graphic treatment of interface elements (the "look" in "look-and-feel")	Visual Design	**Visual Design:** visual treatment of text, graphic page elements and navigational components
Interface Design: as in traditional HCI: design of interface elements to facilitate user interaction with functionality **Information Design:** in the Tuftean sense: designing the presentation of information to facilitate understanding	Interface Design / Navigation Design / Information Design	**Navigation Design:** design of interface elements to facilitate the user's movement through the information architecture **Information Design:** in the Tuftean sense: designing the presentation of information to facilitate understanding
Interaction Design: development of application flows to facilitate user tasks, defining how the user interacts with site functionality	Interaction Design / Information Architecture	**Information Architecture:** structural design of the information space to facilitate intuitive access to content
Functional Specifications: "feature set": detailed descriptions of functionality the site must include in order to meet user needs	Functional Specifications / Content Requirements	**Content Requirements:** definition of content elements required in the site in order to meet user needs
User Needs: externally derived goals for the site; identified through user research, ethno/techno/psychographics, etc. **Site Objectives:** business, creative, or other internally derived goals for the site	User Needs / Site Objectives	**User Needs:** externally derived goals for the site; identified through user research, ethno/techno/psychographics, etc. **Site Objectives:** business, creative, or other internally derived goals for the site

Concrete / Completion / Abstract / Conception / time

task-oriented — information-oriented

This picture is incomplete: The model outlined here does not account for secondary considerations (such as those arising during technical or content development) that may influence decisions during user experience development. Also, this model does not describe a development process, nor does it define roles within a user experience development team. Rather, it seeks to define the key considerations that go into the development of user experience on the Web today.

© 2000 Jesse James Garrett

http://www.jjg.net/ia/

Here is where I think IA began to "fall" so to speak. By the time the dot.com explosion was starting to implode, just a year after this diagram was introduced, people were increasingly trying to use the web as more than a hypertext system. And while all the problems that IAs are good at solving still exist in this new paradigm, they paled in comparison to the challenge of trying to build decent interaction into a website. At this point the focus began to shift away from Web as hypertext to Web as software, and IA, as it was practiced at the time, started to diminish in prominence, if not importance.

By putting a box around IA saying NO graphic design and NO programming, it was only natural that it begin to lose its luster as focused shifted to other problems. Note, I'm not saying IA was less important. Rather, IA had pretty decent tools to solve the problems at hand but the visual and programming issues with the web were a complete mess, so of course that's where attention was paid.

In hindsight, we can look back to this and see two ways in which IA might have played a stronger role... and I do this, not to lament what could have been, but rather to point to what still can be. First, note how Mr. Garrett focuses on Tufte as the poster boy for information design. I like Tufte, but in hindsight, I think if people had recognized how Wurman used the tools of information

design to do information architecture, then the role of IAs might have been seen differently. With the improved tools we now have with CSS and such, I think it's reasonable to reconsider what role Wurman's ideas can have—not that we should become visual designers—but rather so we can have a stronger voice in how visual design is accomplished by providing input to it through the lens of IA.

Second, and this is where I'll spend most of the rest of my talk, I look back and see that the concepts underlying information architecture were playing a large role in the development of the web and of software development in general, only it wasn't recognized as such because it didn't use those words, and it was only practiced by part of the programmer community.

IA Before there was IA

Now to back this up, I need to step back through a history of computing that ran parallel to that which stemmed more directly from Shannon's theories.


The Memex
"...an enlarged supplement to his memory"
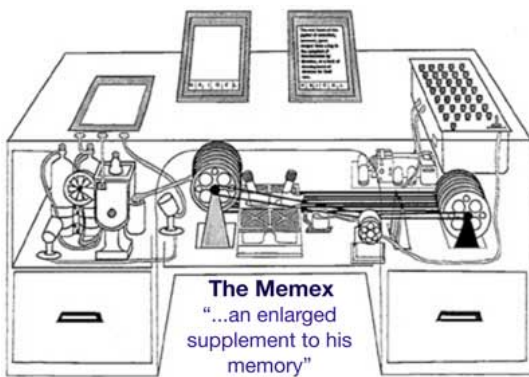
Here we see a picture of Vannevar Bush's Memex macine, what I consider the first information architecture machine. Reading again his article "As We May Think" in the 1945 Atlantic Monthly, it's impressive how prescient he was to what computing would become. It's an important reminder that while much of computing is still focused on replacing the first computers, who were literally humans calculating complex equations, there have been a cadre of scientists focused on developing computing technology designed to leverage and extend human intellect.

I would contend that this is where we should look to for our heritage—men like Ivan Sutherland who developed the first sketchpad in 1962 the first interactive CAD, or Doug Engelbart, inventor of the mouse and the first interactive personal networked computer workstation, who contributed significantly to the idea of networked computing being a tool to improve humanity, not just "do work." Of particular interest to Information Architects is Alan Kay and his team at Xerox, which later included Adele Goldberg *(both pictured right)*.


While some histories of IA rightfully point to Xerox as a place where the architecture of information systems began to blossom, I believe the biggest contribution was not in their development of document management systems and the graphical user interface, which is what people typically point to. Rather, I think we can find the beginnings of today's field of information architecture in the new form of programming called Object Oriented Programming (OOP) that Kay and his team were developing.

A key driver behind this development was the belief that the fundamental economics behind computing would change over time. Throughout the early history of computing the largest expense, and thus the thing to optimize, was the computing
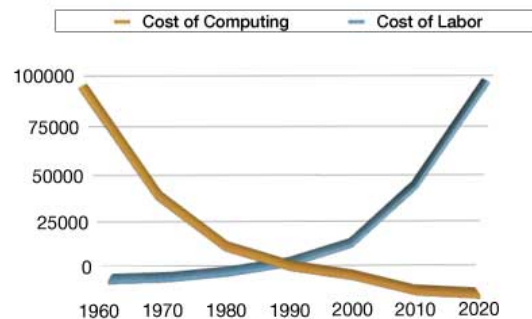
infrastructure. The Xerox team rightly believed that at some point, the cost of computing would drop to well below the cost of human capital, which was increasing while the cost of computing was decreasing. To take advantage of this phenomenon they developed languages that were designed to optimize human understanding of the system being built v. computational efficiency.

Unlike traditional programming, which focuses on moving bits, OOP focuses heavily on developing an abstract model of the how the system your working with actually works in the real world. I don't have time get into all the details of OOP, but these diagrams give you a taste for its power. The Use Case *(below left)* and the class diagram *(below right)* model certain aspects of a software system involving travel. Whether this is eventually coded as a website or a desktop app or a mobile application is irrelevant at this point. What is important for the models to capture are the responsibilities of the actors in the system, the information and behaviors they need to accomplish their tasks, and a representation of other "objects" it needs to interact with.
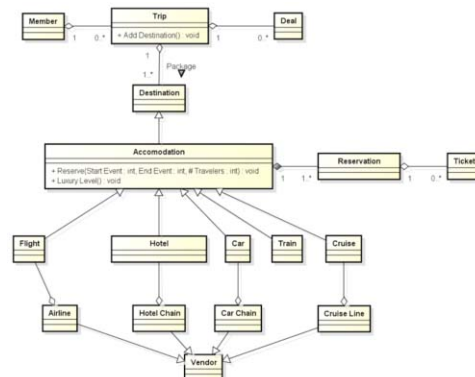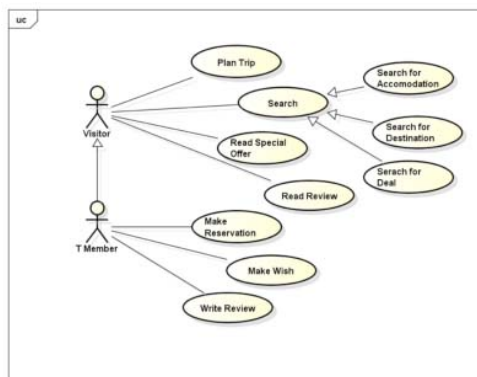
So, much like an information architect seeks to understand and then capture in a website ways to facilitate the mental models and social constructs of the groups interacting with the site, OO architects have been working to capture the same things for software. One of the reasons I left the UM School of Information was because at AIS we were doing the kind of thing I went back to school to do. Using OO tools and methodologies we were architecting information systems, not in the 1960's systems way, but in a way quite analogous to what Lou and Peter were describing in the Polar Bear book. We used the Smalltalk language developed at Xerox, and later adding Java to our toolkit. We also experimented with new project management paradigms and participated on the Chrysler payroll project where Kent Beck worked out his Extreme Programming methodology, the precursor to today's Agile methodology.

In 1998 we were bought by AppNet, an eCommerce roll-up, and we grew our team to 250 programmers primarily building web based software for large corporations. As I was preparing for this talk, I was talking to Tom Meloche, an old friend who I worked with throughout that time. Unlike me, Tom really is an object architect and programmer and he pointed out to me that as far as he's concerned, that AppNet team had something like 30–40 people who could
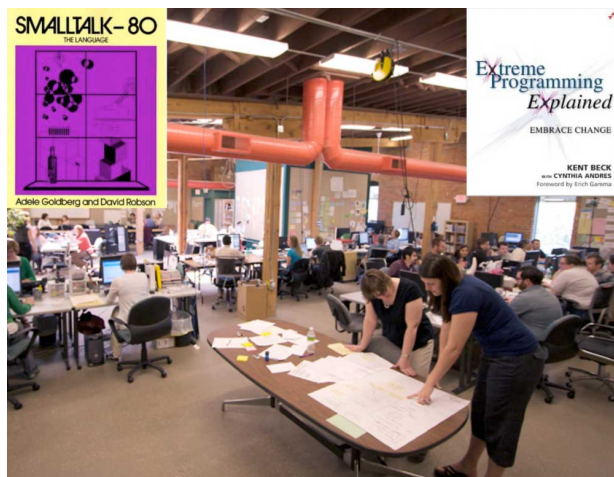
The "software factory" at Menlo Innovations, founded by three former AIS alumni including Tom Meloche.

be reasonably called information architects—non-programmers focused on developing the ontological meaning of the core elements of the system, structuring them in a kind of taxonomy, and describing their choreography using models that others on the team could use to write the programs.

With this background, what can we learn from our cousins in the OO world who have been building "software" using IA techniques for over a decade?

There's Great Power in Models

The fundamental thing that OOP brings to software development and by extension to information architecture is this idea of creating models that depict entire systems from the human perspective. Use cases, like the one shown here, are only one way of showing this. The personas that we develop as IAs are powerful tools, and at TUG we combine personas with scenarios, but OO use cases take it a step further to formally model each of the various interactions that personas, or actors as they're called here, have with the system.

Use Cases are just one of many modeling tools OO architects use. The general take away for IA, I think, is that there are existing modeling tools which can help us capture more than semantic relationship. They also have the ability to capture behavior and interactive relationships. As we think about how to apply IA to broader contexts such as social and mobile and ubiquitous computing in general, this kind of power to model could be highly relevant.

It's worth considering that there is a robust, formal language that's been developed for creating these models called the Unified Modeling Language, which was birthed in the furnace of the "object wars" of the 1990's. UML is not perfect, but it's decent and broadly accepted and there is a mature ecosystem of trainers and the like to help you get up to speed on the tools and techniques.
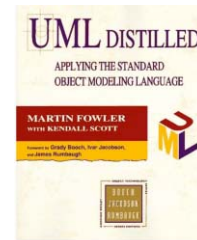
Model/Architect First

Now eventually, *homo logicus* will need to model what happens "under the hood," but that should come later, after the actors and processes, both physical and digital are modeled. One of the things that still surprises me when I hear how people structure web development projects is the fact that modeling the system to be built isn't a key deliverable at the very beginning of the project. Not a low-level system architecture, not even necessarily wireframes, but some type of high-level representation of all the primary activities and relationships to be accomplished by the site. Once buy-in occurs, then you can start rendering structural diagrams and wireframes. Or as my colleague Abby Covert says in a talk she gives to agencies...

Wire frame that first. Doing things in this way reduces risk, because questions are asked and answered early in the process where they are far less expensive to fix.

This kind of thinking is almost second nature to anyone doing complex software development, but not necessarily to web developers. Earlier this week I read a contract from a web development firm that said just the opposite: "we think it's cheaper to fix problems later on v. performing lots of QA up front." This may, or may not, be true for simple web development projects, but it is absolutely broken for complex projects.

Don't Present Data. Present Answers.

This get's at the crux of what makes IA's special. We're trained to look at content and think about what it means. *Homo Logicus* is extremely adept at serving up loads of data to the users screen, all arranged in nice tables, with all the bells and whistles called for by the visual designers. IA's are trained to ask "so what." This is where our "librarian" heritage has something to offer the OO world. As my business partner Dan often notes, one of the most lasting things he took away from his studies at library school had to do with the interview techniques learned by reference librarians. People rarely ask for what they want. It sounds strange, but is quite true... we tend to ask for what we think we need. The skilled reference librarian knows to look behind the question and probe into the "why" we need the information. We ask for data even though we really want answers. And answers require effort to understand user requirements, understand what the data fundamentally means (aka ontology), understand the relationships that exist within the data and so forth. You need a data mining expert to build the cubes, but may I suggest that you should use an information architect to drive the process.

UX is useless if you don't transform data into answers

A fundamental problem with the web circa 2000 to 2005 was the inability to present reasonable interfaces. It was a real problem and it made sense for the industry to pay a lot of attention to the problem. We owe a huge debt of gratitude to the folks at Adaptive Path for introducing AJAX and to all the others who developed various asynchronous tools that allow a website to behave like a real software application. But that problem is now basically solved. Today, it's not special to have a usable interface...it's expected. Today I believe the bigger challenge is delivering valuable information, because without valuable information, the most lovely user experience falls flat. I think even Jesse James Garret might agree with this. In fact, during the Q&A after his keynote at the Midwest UX conference in Columbus Ohio last April someone asked him something to the effect of what would he do differently if we could turn back the clock, what did he miss. He answered simply that he would have given more importance to information architecture.

In the end, we want as little UX as possible.
It's not about the computer, it's about leveraging our intellect.

When all is said and done, our goal should be for the users' interactions with the computer to be virtually invisible...something they don't need to think about. To be sure, you don't get there without exceptional user experience skills on your team, but you also don't get there if you haven't put a great deal of effort into understanding and then to model how the world your client and their users operate in. For example: Who needs to get what from whom? Not just within the computer but among all the participants in the system. Information architects already know how to do this at one level, and I think the opportunity is there for us to apply these skills more broadly. Not everywhere, with every kind of system. Let me be clear that I'm not suggesting the IAs morph into OO architects. There are certainly classes of software where the structuring of information spaces is not a major task, but there are plenty of applications where, at the least, we ought to be tightly partnering with the lead software architect to help model the worlds and relationships we're trying to facilitate.

To circle back to this issue of IA v. UX. I think we've tended to focus our attention the wrong "why." A positive user experience is clearly the end goal, but I'd suggest as information architects, we refine our efforts to focus on something more specific...

understanding

Many disciplines contribute to creating a good user experience. Look at anyone's taxonomy of skills and you see plenty of overlap around which "titles" do which tasks. But there's really only one that contains within it this idea of creating understanding as primary output of our efforts—information architects.

If we're to help people navigate in their Library of Babel, then we need to recognize this unique aspect to our craft and foster it. And make no mistake, we've only begun to see how important the world of data is to become.

Data is becoming the new "Intel Inside" as companies scramble to gather and then monetize the data that flows through their systems. It's been coming for a while now, driven by a number of factors ranging from the increasing number of Internet users to the fact that everything from cell phones to refrigerators are being connected to the network. In the 1990's Sun Microsystem had the slogan "The Network is the Computer." It was beginning to be true back then, thought they paid the price for being on the bleeding edge (which goes to prove the adage that "people tend to over estimate what will happen in the near term and under estimate what will happen in the long run"). What this eventually means for data, only time will tell, but you can see signs that the past predictions about it are starting to come true.

One example is a new program at the information school at Syracuse University in New York where their developing a program focused on dealing with "Big data" problems in science. Another is seen in the focus on "the data layer" at the upcoming Web 2.0 conference. That conference is put on in part by John Battelle, one of my favorite technology journalist. He was part of the team that formed Wired magazine back when it was interesting, and in 2003 he wrote about the web as a database of intentions. What people do as they interact will signal their intentions, opening up lots of opportunity to help them achieve their goals. Depending on how this plays out it could be very creepy, or quite nice. But either way, we're starting to seeing this become a reality.

Now a major hindrance to this happening is the tendency by the major players to try and lock up their data within their own walled garden. But history has shown that walled gardens rarely survive an attack by a good open ecosystem. As Battelle writes:

"A generation from now our industry's approach to data collection will seem outdated and laughable. The most valuable digital services and companies will be rewarded for what they do with openly shareable data, not by how much data they hoard and control."

Flipboard, shown at right is but one example of this. As information architects we ought to be drooling at the possibilities.

"There are all sorts of people who don't call themselves journalists now who traffic in information, as governments put out data, companies put out raw information and people in the course of their business gather and share information. That's the way Arrington saw TechCrunch."
Jeff Jarvis from the CUNY Graduate School of Journalism

Another example are entities like the Huffington Post or Tech Crunch. Is Michael Arrington, founder of Tech Crunch, an information architect of sorts? He took a collection of disparate information feeds, developed a taxonomy and a content strategy and built a profitable enterprise around it.

Another key driver for the growth of IA is the emergence of the social channel. Antonio Lucio, the CMO of Visa (pictured left) has challenged his all the marketers in his organization with these three principles of social media shown left.

Think about this from an information architecture perspective. Do I create a new set of structures and classifications for social? Or do I take a step back and make sure my existing information architecture is robust enough to help guide decisions about what content to push where and when? What tools do I use to model this type of cross-channel effort? This is a complicated problem, and I'd suggest that our cousins in the software space have gifts to share that we'd be wise to consider.

Bringing it all together
In summary, let me say that the ups and downs of our field are to be expected for a young discipline. We shouldn't be afraid to figure out just who we are and what we want to be. I think that information architecture is poised to explode in importance if:

- We double down on what we're good at, and
- We draw upon our rich heritage of "understanding" and modeling of systems.

Thank You.