THE ELEMENTS OF USER EXPERIENCE

USER-CENTERED DESIGN FOR THE WEB

Jesse James Garrett

AIGA

New Riders

chapter 2

Meet the Elements

The user experience development process is all about ensuring that no aspect of the user's experience with your site happens without your conscious, explicit intent. This means taking into account every possibility of every action the user is likely to take and understanding the user's expectations at every step of the way through that process. It sounds like a big job, and in some ways it is. But by breaking the job of crafting user experience down into its component elements, we can better understand the problem as a whole.

The Five Planes

Most people, at one time or another, have purchased a book over the Web. The experience is pretty much the same every time—you go to the site, you find the book you want (maybe by using a search engine or maybe by browsing a catalog), you give the site your credit card number and your address, and the site confirms that the book will be shipped to you.

That neat, tidy experience actually results from a whole set of decisions—some small, some large—about how the site looks, how it behaves, and what it allows you to do. These decisions build upon each other, informing and influencing all aspects of the user experience. If we peel away the layers of that experience, we can begin to understand how those decisions are made.

The Surface Plane

On the **surface** you see a series of Web pages, made up of images and text. Some of these images are things you can click on, performing some sort of function such as taking you to a shopping cart. Some of these images are just illustrations, such as a photograph of a book cover or the logo of the site itself.

The Skeleton Plane

Beneath that surface is the **skeleton** of the site: the placement of buttons, tabs, photos, and blocks of text. The skeleton is designed to optimize the arrangement of these elements for maximum effect and efficiency—so that you remember the logo and can find that shopping cart button when you need it.

The Structure Plane

The skeleton is a concrete expression of the more abstract **structure** of the site. The skeleton might define the placement of the interface elements on our checkout page; the structure would define how users got to that page and where they could go when they were finished there. The skeleton might define the arrangement of navigational items allowing the users to browse categories of books; the structure would define what those categories actually were.

The Scope Plane

The structure defines the way in which the various features and functions of the site fit together. Just what those features and functions are constitutes the **scope** of the site. Some sites that sell books offer a feature that enables users to save previously used addresses so they can be used again. The question of whether that feature—or any feature—is included on a site is a question of scope.

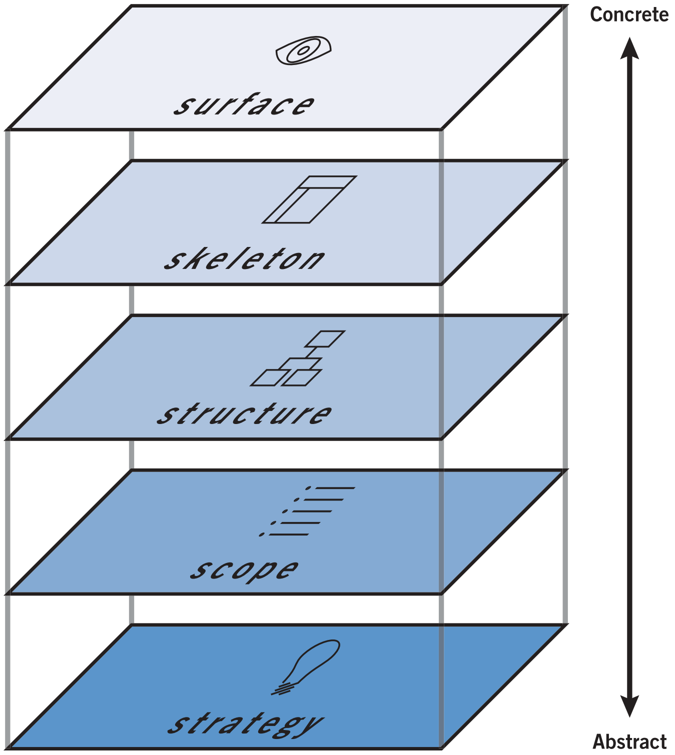
The Strategy Plane

The scope is fundamentally determined by the **strategy** of the site. This strategy incorporates not only what the people running the site want to get out of it but what the users want to get out of the site as well. In the case of our bookstore example, some of the strategic objectives are pretty obvious: Users want to buy books, and we want to sell them. Other objectives might not be so easy to articulate.

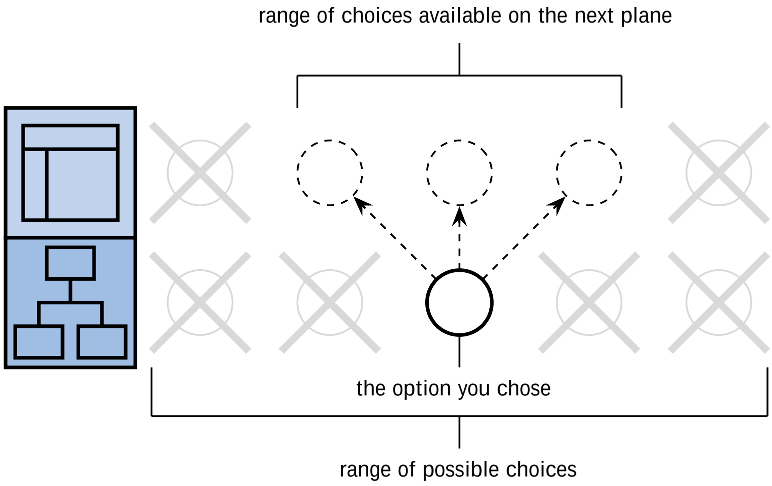
Building from Bottom to Top

These five planes—strategy, scope, structure, skeleton, and surface—provide a conceptual framework for talking about user experience problems and the tools we use to solve them.

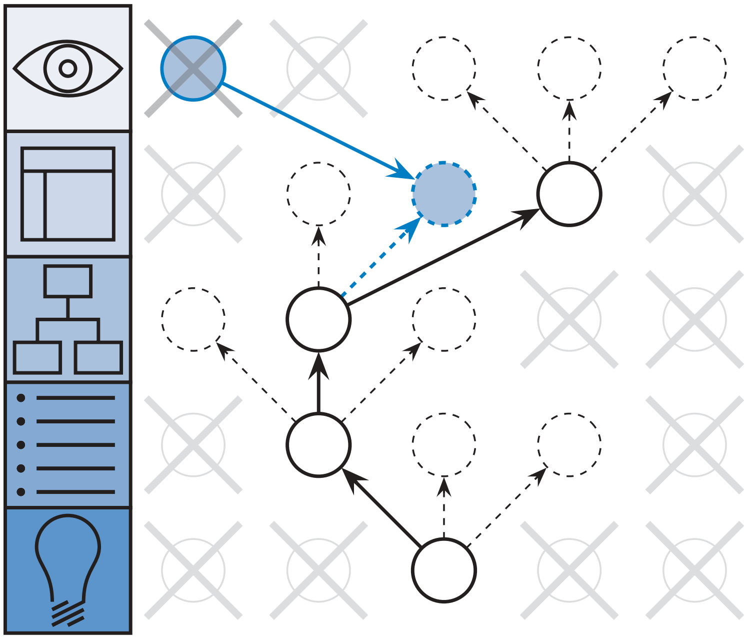
On each plane, the issues we must deal with become a little less abstract and a little more concrete. On the lowest plane, we are not concerned with the final shape of the site at all—we only care about how the site will fit into our strategy (while meeting the needs of our users). On the highest plane, we are only concerned with the most concrete details of the appearance of the site. Plane by plane, the decisions we have to make become a little more specific and involve finer levels of detail.



Each plane is dependent on the planes below it. So, the surface depends on the skeleton, which depends on the structure, which depends on the scope, which depends on the strategy. When the choices we make don't align with those above and below, projects often derail, deadlines are missed, and costs begin to skyrocket as the development team tries to piece together components that don't naturally fit. Even worse, when the site finally does launch, the users will hate it. This dependence means that decisions on the strategy plane will have a sort of "ripple effect" all the way up the chain. Conversely, the choices available to us on each plane are constrained by the decisions we make about issues on the planes below it.


range of choices available on the next plane
the option you chose
range of possible choices

The choices you make on each plane affect the choices available to you on the next plane above it.

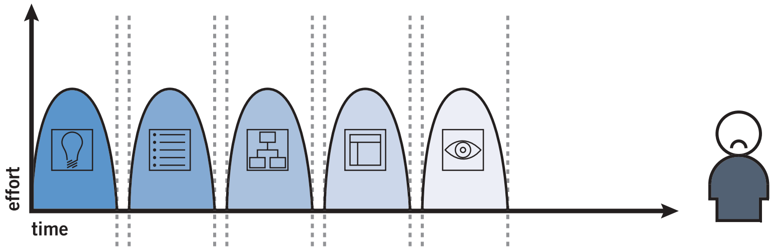


This ripple effect means that choosing an "out of bounds" option on an upper plane will require rethinking decisions on lower planes.

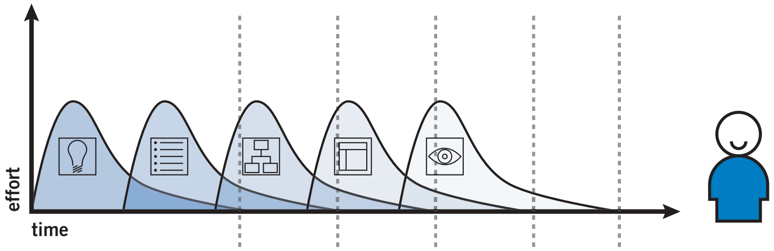
That does not mean, however, that every decision about the lower plane must be made before the upper plane can be addressed. Dependencies run in both directions, with decisions made on upper planes sometimes forcing a reevaluation (or an evaluation made for the first time!) of decisions on lower planes. At each level, we make decisions according to what the competition is doing, industry best practices, and plain old common sense. These decisions can have a ripple effect in both directions.

If you consider your decisions on lower planes to be set in stone before you take on your decisions on higher planes, you will almost certainly be throwing your project schedule at the very least—and possibly the success of your final product—into jeopardy.

Instead, you should plan your project so that work on any plane cannot *finish* before work on lower planes has finished. The important consideration here is not to build the roof of the house before we know the shape of its foundation.



Requiring work on each plane to **finish** before work on the next can **start** leads to unsatisfactory results for you and your users.



A better approach is to have work on each plane **finish** before work on the next can **finish**.

A Basic Duality

Of course, there are more than just five elements of user experience, and as with any specialized field, this one has evolved a vocabulary all its own. To someone encountering the field for the first time, user experience can appear to be a complicated business. All these seemingly identical terms are thrown around: interaction design, information design, information architecture. What do they mean? Anything? Or are they just more meaningless industry buzzwords?

To further complicate matters, people will use the same terms in different ways. One person might use “information design” to refer to what another knows as “information architecture.” And what’s the difference between “interface design” and “interaction design?” Is there one?

Fortunately, the field of user experience seems to be moving out of this Babel-like state. Consistency is gradually creeping into our discussions of these issues. To understand the terms themselves, however, we should look at where they came from.

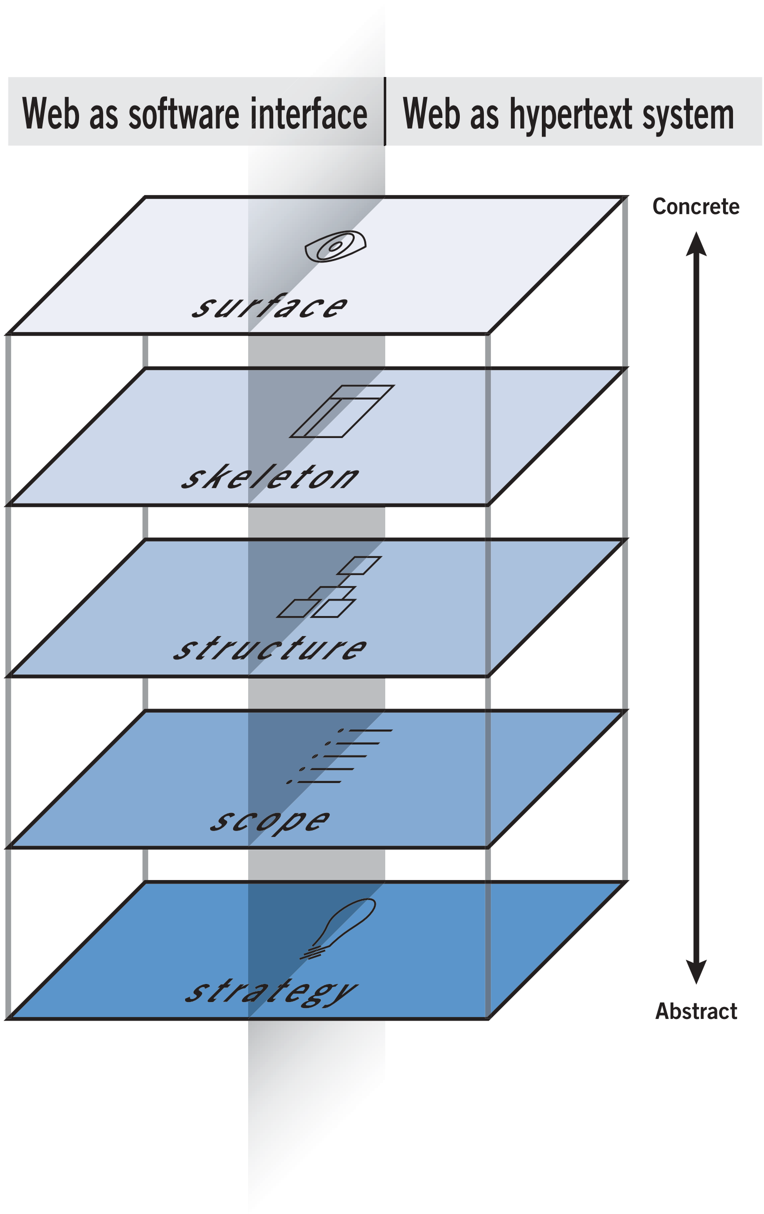
When the Web started, it was just about hypertext. People could create documents, and they could link them to other documents. Tim Berners-Lee, the inventor of the Web, created it as a way for researchers in the high-energy physics community, who were spread out all over the world, to share and refer to each other’s findings. He knew the Web had the potential to be much more than that, but few others really understood how great its potential was.

People originally seized on the Web as a new publishing medium, but as technology advanced and new features were added to Web browsers and Web servers alike, the Web took on new capabilities. After the Web began to catch on in the larger Internet community, it developed a more complex and robust feature set that would enable Web sites not only to distribute information but to collect and manipulate it as well. With this, the Web became more interactive, responding to the input of users in ways that were very much like traditional desktop applications.

With the advent of commercial interests on the Web, this application functionality found a wide range of uses, such as electronic commerce, community forums, and online banking, among others. Meanwhile, the Web continued to flourish as a publishing medium, with countless newspaper and magazine sites augmenting the wave of Web-only "e-zines" being published. Technology continued to advance on both fronts as all kinds of sites made the transition from static collections of information that changed infrequently to dynamic, database-driven sites that were constantly evolving.

When the Web user experience community started to form, its members spoke two different languages. One group saw every problem as an application design problem, and applied problem-solving approaches from the traditional desktop and mainframe software worlds. (These, in turn, were rooted in common practices applied to creating all kinds of products, from cars to running shoes.) The other group saw the Web in terms of information distribution and retrieval, and applied problem-solving approaches from the traditional worlds of publishing, media, and information science.

This became quite a stumbling block. Very little progress could be made when the community could not even agree on basic terminology. The waters were further muddied by the fact that many Web sites could not be neatly categorized as either applications or hypertext information spaces—a huge number seemed to be a sort of hybrid, incorporating qualities from each world.


Web as software interface
Web as hypertext system
Concrete
surface
skeleton
structure
scope
strategy
Abstract

To address this basic duality in the nature of the Web, let's split our five planes down the middle. On the left, we'll put those elements specific to using the Web as a **software interface**. On the right, we'll put the elements specific to **hypertext information spaces**.

On the software side, we are mainly concerned with **tasks**—the steps involved in a process and how people think about completing them. Here, we consider the site as a tool or set of tools that the user employs to accomplish one or more tasks.

On the hypertext side, our concern is **information**—what information the site offers and what it means to our users. Hypertext is about creating an information space that users can move through.

The Elements of User Experience

Now we can map that whole confusing array of terms into the model. By breaking each plane down into its component elements, we'll be able to take a closer look at how all the pieces fit together to create the whole user experience.

The Strategy Plane

The same strategic concerns come into play for both software products and information spaces. **User needs** are the goals for the site that come from outside our organization—specifically from the people who will use our site. We must understand what our audience wants from us and how that fits in with other goals it has.

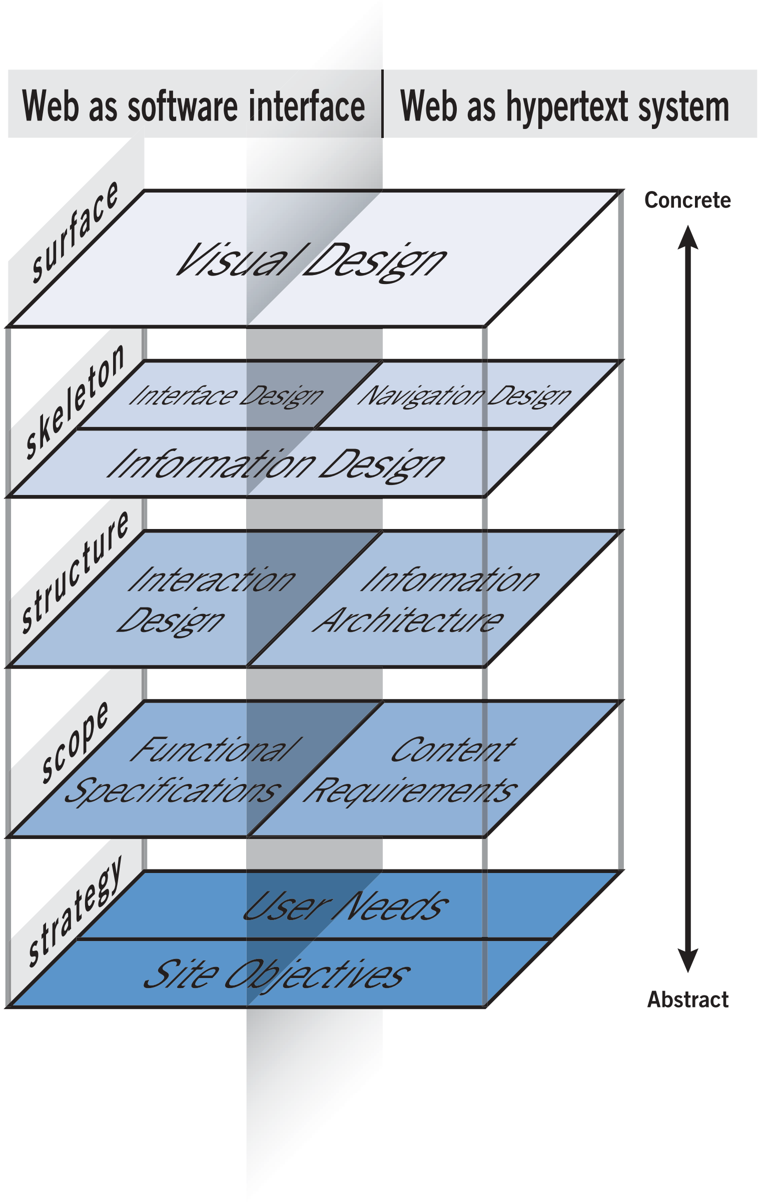
Balanced against user needs are our own objectives for the site. These **site objectives** can be business goals ("Make $1 million in sales over the Web this year") or other kinds of goals ("Inform voters about the candidates in the next election"). In Chapter 3 we'll go into more detail about these elements.

The Scope Plane

On the software side, the strategy is translated into scope through the creation of **functional specifications**: a detailed description of the "feature set" of the product. On the information space side, scope takes the form of **content requirements**: a description of the various content elements that will be required. Chapter 4 will cover the scope elements.

The Structure Plane

The scope is given structure on the software side through **interaction design**, in which we define how the system behaves in response to the user. For information spaces, the structure is the **information architecture**: the arrangement of content elements within the information space. You'll find more details on these in Chapter 5.


Web as software interface
Web as hypertext system
surface
Visual Design
skeleton
Interface Design
Navigation Design
Information Design
structure
Interaction Design
Information Architecture
scope
Functional Specifications
Content Requirements
strategy
User Needs
Site Objectives
Concrete
Abstract

The Skeleton Plane

The skeleton plane breaks down into three components. On both sides, we must address **information design**: the presentation of information in a way that facilitates understanding. For software products, the skeleton also includes **interface design**, or arranging interface elements to enable users to interact with the functionality of the system. The interface for an information space is its **navigation design**: the set of screen elements that allow the user to move through the information architecture. There's more about the skeleton plane in Chapter 6.

The Surface Plane

Finally, we have the surface. Regardless of whether we are dealing with a software product or an information space, our concern here is the same: the **visual design**, or the look of the finished product. It's trickier than it sounds; you can find out all about it in Chapter 7.

Using the Elements

Few sites fall exclusively on one side of this model or the other. Within each plane, the elements must work together to accomplish that plane's goals. For example, information design, navigation design, and interface design jointly define the skeleton of a site. The effects of decisions you make about one element from all other elements on the plane is very difficult. All the elements on every plane have a common function—in this example, defining the site's skeleton—even if they perform that function in different ways.

This model, divided up into neat boxes and planes, is a convenient way to think about user experience problems. In reality, however, the lines between these areas are not so clearly drawn. Frequently, it can be difficult to identify whether a particular user experience problem is best solved through attention to one element instead of another. Can a change to the visual design do the trick, or will the underlying navigation design have to be reworked? Some problems require attention in several areas at once, and some seem to straddle the borders identified in this model.

The way organizations often delegate responsibility for user experience issues only complicates matters further. In some organizations, you will encounter people with job titles like information architect or interface designer. Don't be confused by this. These people generally have expertise spanning many of the elements of user experience, not just the specialty indicated by their title. It's not necessary to have a member of your team who is a specialist in each of these areas; instead, you only have to ensure that someone is responsible for thinking about each of these issues.

A couple of additional factors go into shaping the final user experience that you won't find covered in detail here. The first of these is **content**. The old saying (well, old in Web years) is that "content is king" on the Web. This is absolutely true—the single most important thing most Web sites can offer to their users is content that those users will find valuable.

Users don't visit Web sites to experience the joy of navigation. The content that is available to you (or that you have resources to obtain and manage) will play a huge role in shaping your site. In the case of our bookstore site example, we might decide that we want the users to be able to see cover images of all the books we sell. If we can get them, will we have a way to catalog them, keep track of them, and keep them up to date? And what if we can't get photos of the book covers at all? These content questions are essential to the ultimate user experience of the site.

Second, **technology** can be just as important as content in creating a successful user experience. In many cases, the nature of the experience you can provide your users is largely determined by technology. In the early days of the Web, the tools to connect Web sites to databases were fairly primitive and limited. As the technology has advanced, however, databases have become more widely used to drive Web sites. This in turn has enabled more and more sophisticated user experience approaches, such as dynamic navigation systems that change in response to the way users move through the site. Technology is always changing, and the field of user experience always has to adapt to it. Nevertheless, the fundamental elements of user experience remain the same.

The rest of this book looks at the elements, plane by plane, in greater detail. We'll take a closer look at some of the tools and techniques commonly used to address each element. We'll see what the elements on each plane have in common, what makes each one different, and how they affect each other to create the total user experience.