ARC90 White Paper

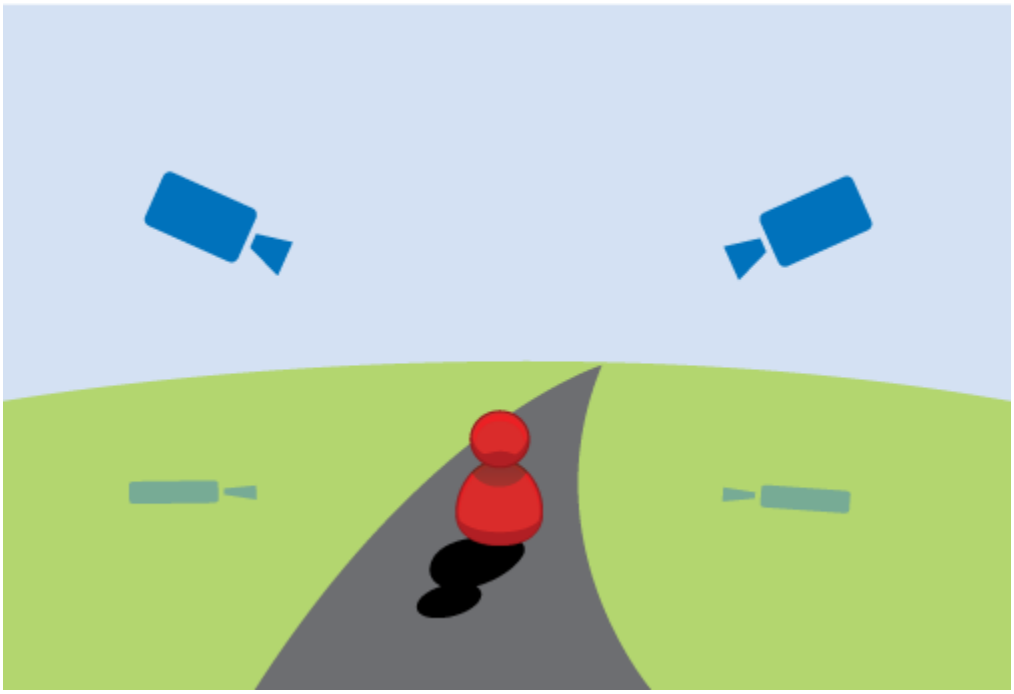
Information Objects

Applying Cognitive Load Theory & Object-Oriented Thinking to Information Design

Richard Ziade, Matthew Kittredge

1. Background

Interface design principals today are dominated by what is commonly termed “user-centered design” (“UCD”). The idea behind UCD is that systems should be designed with the experience of the user in mind. Users are profiled, their habits are monitored and storyboards are often drawn. The goal is to create an experience that is locked in to how users work and hopefully how users think. There is no doubt that UCD is a justifiably effective approach of interface design today. By designing to the way users think and behave rather than how systems work, a more intuitive, and sometimes more satisfying experience is delivered. This often leads to a higher level of user loyalty and improved good will.



While UCD speaks to an important aspect of sound design, it does not address many of the more basic characteristics of human-machine interaction. This paper is an attempt to apply some of the fundamental tenets of how we perceive and deal with objects in the real world to information architecture and interface design. It is by no means a suggested substitute for UCD. Instead, it is an attempt to augment and support the practices of UCD by applying some basic principles.

2. A Look Back

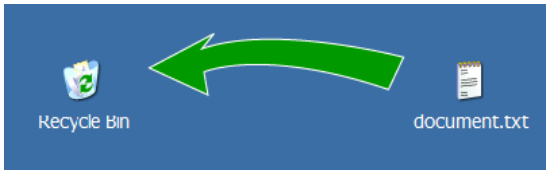
2.1. The Big Bang

If we take a few steps back from the timeline of computer interfaces, there lies one significant milestone that yanked computers from the exclusive domain of the technically savvy and hurled them into everyone else’s everyday world.

Visually speaking, it was when we went from this:

```
C:\>delete c:\desktop\MyOldDocument.doc
```

To this:



The *desktop metaphor* succeeded in introducing the power of computers to a far wider population because it mimicked behaviors and experiences that we are already accustomed to in the real world. Rather than wait for users to make the cognitive leap to understand how these complex machines work, the machines themselves made the leap towards how human beings think.

The underlying premise behind the desktop metaphor is that users have already hard-wired numerous conventions that drive how they perceive and interact with the everyday physical world. The desktop interface felt immediately intuitive to many because it presented visual representations (documents, folders, pictures, trash cans) and actions (cutting, pasting, viewing, editing) that we are already familiar with.

2.2. A Dirty Trick

And so, this remarkable leap in the evolution in human-computer interaction is nothing more than…a trick. The inventors of the desktop metaphor created this simulated world that was more akin to the world to which we are accustomed. Of course, just beneath the veneer of the desktop and its icons still lived the same complex systems. The only difference now was that users were not required to know as much about these systems to tap into their power. Despite the additional processing demands the desktop metaphor required, it was still a worthwhile step in that it delivers the power of computers to the masses.

And so, while interface designers today extol the virtues of sitting down with and listening to users (as they should), some of the most significant leaps in interface design involved no user observation or discussions at all. They simply leveraged what we already knew and understood about the physical world.

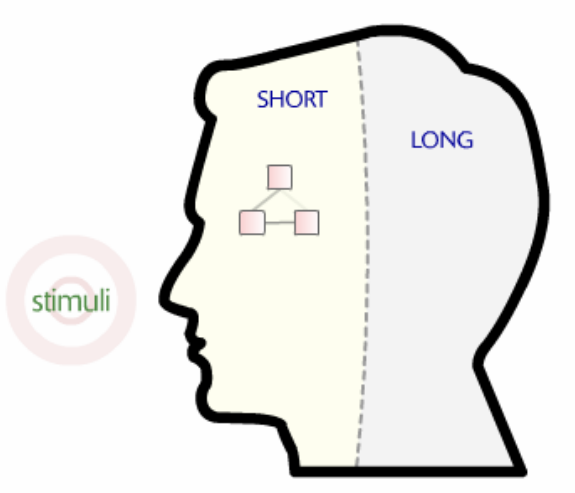
3. Cognitive Load Theory

When we learn to execute the task above, deleting a file on a desktop, our minds engage in the most basic of cognitive processes. In each instance I can tell you how to delete the file. For the older method I would say, “type ‘C’, colon, backslash, greater than delimiter, the word “delete”, all lowercase, space, lowercase ‘c’, colon, backslash, “desktop”, backslash, ‘MyOldDocument’ (all case sensitive), period, ‘doc’”. In the second instance I would say, “click on the piece of paper, hold down the mouse, drag it into the bin and let go”. Which process takes more mental work? Which is more likely to be remembered the next time the user has to execute the operation? This work, this mental grappling, is what Cognitive Psychologists call “cognitive load”. Simply put, cognitive load is the amount of work the mind must do to integrate a recently learned task into an established skill. As designers, it is our job to make this learning process as easy as possible for users to go from new task to mastered task.

Regardless of how simple or complex a task is, how does this process work and how can we learn from it to make better designs?

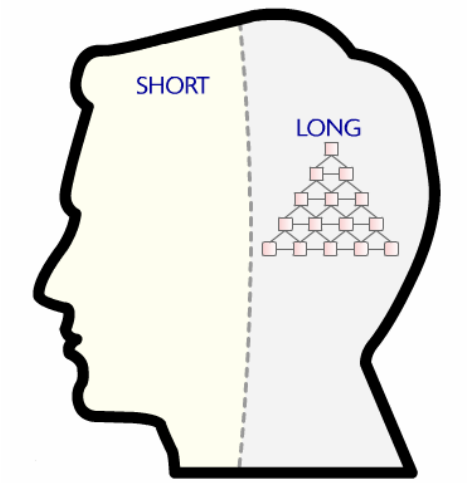
3.1. Short Term v. Long Term Memory

The very same Cognitive Psychologists that offered this theory of cognitive load base this theory on established tenets of Memory. Memory, they suggest, is a two tiered strata divided into short term and long term.



3.1.1. Short Term Memory

Short term memory or working memory is much akin to our immediate consciousness, but it is often transitory, disorganized and suffers from inefficient recall. As stimuli enters our consciousness, our mind works to make sense of what our sensory faculties are providing it.



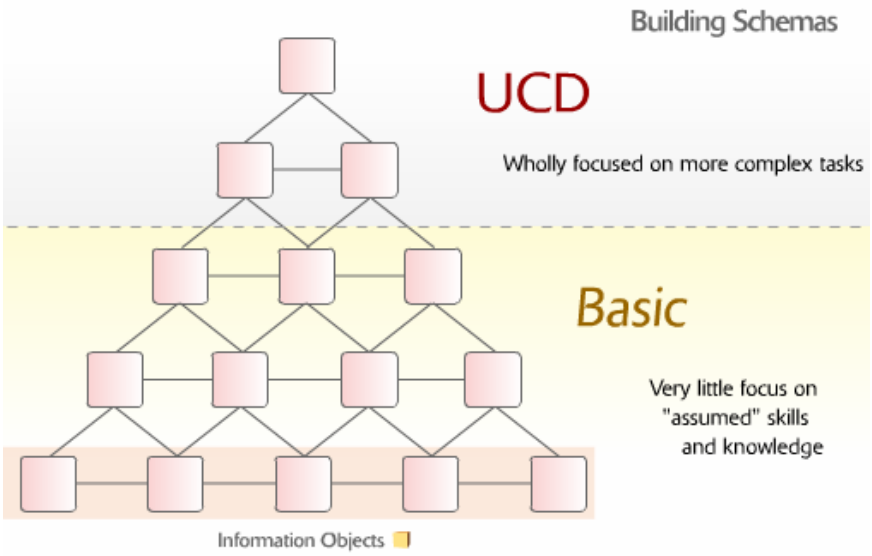
3.1.2. Long Term Memory

Long term memory is much the opposite. It is generally much more stable; more organized and is recalled more efficiently. In long term memory, each bit of information is structured in a complex yet highly organized manner. Each bit of information tends to have a more fixed relationship with everything else in a person's long term memory.

3.2. From Short to Long Term Memory – Building Schemas

The amount of processing the mind must do to capture these fleeting and disorganized bits of information and move them into the established structure of long term memory is again the cognitive load. In school we may remember being asked to use pneumonic devices to remember new or complex information. My 7th grade Algebra teacher burned the order of mathematical operations into my head not by rote repetition, but by asking me to "please excuse my dear Aunt Sally". The order of mathematical operations? Parentheses, exponents, multiply, divide, add and subtract. This simple phrase was easy to remember. The order of operations, not as easy. Attaching the order of mathematical operations to the phrase about "Aunt Sally" connected a complex operation with a simple phrase and thus made the memory permanent.

As we can see, a major component of moving new information into our long term memory is attaching that which is new to structures that already exist. The structure of the desktop/recycle bin metaphor existed for most of us long before our first contact with computers. At some point a parent or a teacher taught us that when it is time to "get rid of something" disposable, we picked it up and tossed it in the trashcan. But even this required a few basic skills: picking up objects, moving them, placing them. The use of the desktop metaphor worked because it built upon our most basic understanding of the physical world.



Herein lies a basic principle of this "Theory of Information Objects" as it relates to "user centered design: User Centered Design often fails to build upon the most basic aspects of our relationship to the physical world and instead starts with more complex tasks thus slowing the integration of new processes into our existing schemas of knowledge and skills.

4. Information Design Today

4.1. Information Overload

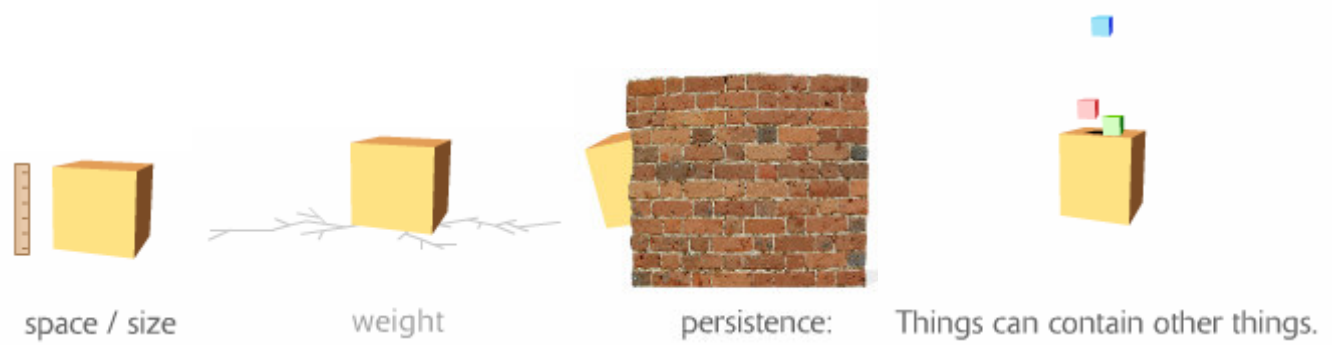
With the Internet explosion and the increasing complexity of software applications came a world where massive amounts of information, often intricately related, began dominating the computer experience. From the endless articles on the Web, to the endless products in e-commerce shops, to the boggling complexity of today's desktop applications, the benefits of the desktop metaphor have worn a bit thin. Graphical icons and the mouse can only take us so far. The challenge today is less about *how* we interact with computers and more about the mountains of information before us and how to present it and make it easy to manipulate.

5. The "*Information object*"

5.1. What's That "Thing" Called?

The world around us today, as experienced through our senses, is incredibly rich. We are constantly confronted with an amazing amount of "data" and yet we navigate through it with relative ease. In fact, most of us have a harder time grasping abstract concepts than the tangible world and its artifacts.

From a very young age, we quickly internalize the notion of "things" - actual things that exist (not ideas and the like). What do we know about things that don't need to be explained to us?



- Things can take up space.
- Things can have weight.
- Things persist even when they're not in our field of view.
- Things can contain other things.
- Things can sometimes perform certain actions if prompted or controlled in certain ways.

From a cognitive behavioral perspective, we were not born with these suppositions. Instead, we constructed them through our early experiences and stored them away into long-term memory. As we experience more about the world, we use these very basic building blocks to build upon our understanding of things and how they work.

From a design point-of-view, the less we rely on these building blocks, the more we demand of the user to understand and construct these building blocks on their own (i.e. the more "cognitive load" we impose on the user). The simpler the building blocks, like those listed above, the more universally a particular interface will be understood and accepted.

One of the most flattering compliments an interface designer can receive is that an interface is intuitive. Such a statement insinuates that the designer introduced something that, regardless of experience or exposure, required almost no thinking. Its usability is somehow instinctive. In actuality, the designer obviously didn't tap into the user's mind and read it. Instead, they leveraged the building blocks described above. To the user, it appeared intuitive because almost no work was required.

5.2. Information Objects

Information (in its most general sense) is far less tangible than things. A timesheet or an invoice is a thing, but it can exist in all sorts of incarnations in today's information age. Beyond timesheets and invoices, more complex types of information can exist in even more convoluted forms. Despite the fact that a "sales effort" or a "customer review" may be comprised of amorphous blobs of information that may be spread across many forms and locations, we persist in our drive to perceive these collections of information in tangible ways.

The suggestions outlined below are founded on the principle that the more we *objectify* information, the easier it will be for people to understand it and understand how they can use it. Many of the suggestions put forth may appear obvious. Designers consciously and subconsciously apply many of these concepts regularly. The guidelines set forth below attempt to more clearly define various techniques for objectifying information.

One final point of clarification, this approach does not assume or suggest that *information objects* must have a real-world counterpart. Unlike the desktop metaphor, which attempts to simulate real world objects and their mechanisms, *information objects* can simply be information – any information – that takes on object-like characteristics.

5.3. Defining The Objects

A key exercise before embarking upon organizing your information into *information objects* is to define what those objects are. There is no hard and fast rule to this process. Some objects jump out at you because they have real-world counterparts: a timesheet, a financial report, a plane ticket. The more challenging *information objects* arise less from metaphor and more from a clear need to emphasize their objectiveness in order to make them easier to perceive and use. These less obvious *information objects* are based more upon collections of concepts and commonalities but have no real-world sibling. For example, a "report builder" or a "work queue" can be nicely represented as *information objects*.

5.4. Key Guidelines

There are some key guidelines that should be adhered to in deciding when and how much to objectify information.

5.4.1. Object Complexity & Richness

The richer and more complex an object is, the more clearly it should be defined. Objects that possess detailed and intricate information or more involved and complex controls should have clearer object-like characteristics.

For example, if a headline in a list of headlines is nothing more than a link to an external article, then almost none of the methods described below are necessary. However, if the object is an insurance policy with all the layers of information and controls that come with it, then we are serving users well by doing as much as possible to clearly convey that this is an *information object*.

5.4.2. Information Object Interplay

As an extension to the rule described above, it is also important to convey as clearly as possible *how* objects relate to one another. Is an *information object* comprised of numerous other objects? Are a collection of *information objects* peers? Simple conventions can help convey how *information objects* relate to one another.

5.4.3. Some Examples

To help illustrate how to define *information objects*, some examples follow. The below are by no means exhaustive lists of objects for the below applications, but rather helpful starting points:

- **Timesheet Application.** Contains *timesheet*, *consultant*, *manager* and *project information objects*.
- **RSS Aggregator.** Contains *channel, channel group, feed* and *feed entry information objects*.
- **E-Commerce Application.** Contains *product*, *product category*, *shopping cart* and *invoice information objects*.

5.5. Conveying Objectiveness

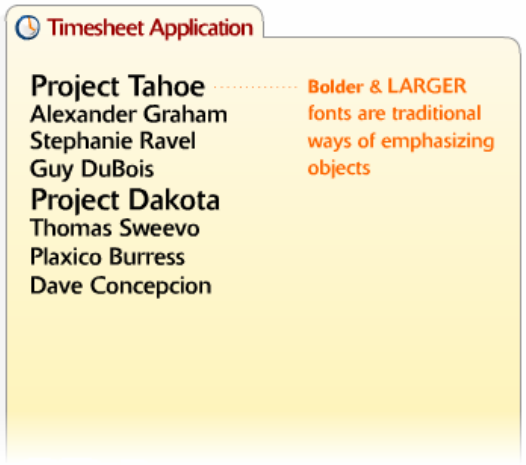
Once our objects are defined, we have at our disposal an arsenal of techniques that hint to the user that they are interacting with objects. What follows is an outline of these techniques. This list is by no means exhaustive, but many of them will appear obvious. Nevertheless, systems today are rife with poor information design so while they may be obvious, they are often neglected.

5.5.1. Delineation

The most effective method for making information easier to understand and interact with is delineation. Delineation is the practice of providing some visual hints that some information is discrete and stands on its own relative to other information.

We delineate information all the time. Look at your typical restaurant menu and you'll most likely find the dishes available to be categorized and grouped: appetizers, salads, entrees and dessert. The *information object* in that context is *meal category*.

5.5.1.1. Headlining



One of the easiest (and most obvious) ways to objectify information is to title it. By making a blurb of text larger and more prominent, we're able to hint that the information below (and it's always the information below) *falls within* the domain of the headline, thus alluding to the existence of an *information object*.

Headlining becomes even more valuable when numerous objects of a similar type are displayed in a single instance.

5.5.1.2. Spacing

A less obvious but commonly used device for objectifying information is spacing. Put simply, by taking a set of information and clearly demarcating an area of space around it, we imply that the set of information that resides within the information "zone" is related.

Beyond its effectiveness to objectify information, space is effective in emphasizing information in a clean rather than loud manner. Whereas you can cause a user to pay attention to something with large text and bright colors, generous use of spacing can prove just as effective, if not more dramatic, in emphasizing an *information object*.

5.5.1.3. Wrapping

An extremely effective method of emphasizing the objectiveness of a set of information is to visually wrap that information in some way. This is typically done by simply drawing a line around the information. Optionally, lines that run horizontally or vertically along the objects border are nearly as effective.

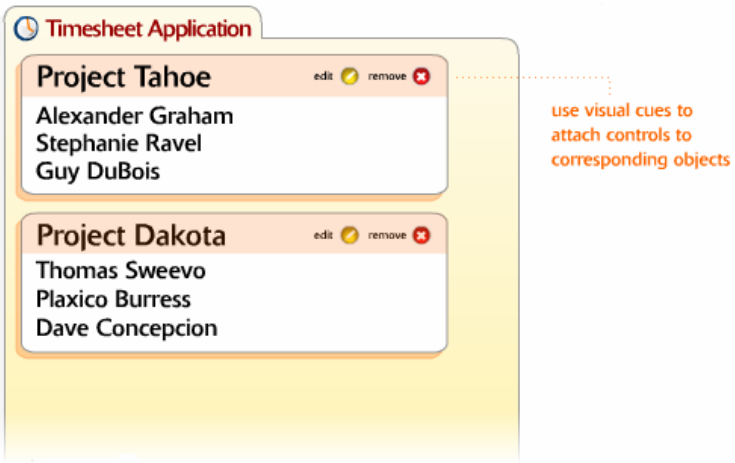
Wrapping is a method of delineation that hints that the object has some perceived volume to it. It is the most effective means of combating the perception that information is merely floating in space. The "thing" now has physical dimensions, like objects in the real world. The object's borders clearly intimate that that which is within them is the object and the object alone, and everything else not within these borders is not of the object. Wrapping is one of the most powerful mechanisms for objectifying information.

A powerful technique that combines headlining and wrapping can effectively declare and establish that an *information object* exists and what exactly it is.

5.5.2. Information object Controls

5.5.2.1. Buttons & Levers

The above are effective techniques for emphasizing information in the form of objects that users can examine and study. While these methods are adequate for one-way mediums like charts, presentations and diagrams, much of the principles of *information objects* apply to interactive systems as well.



Many objects in the real world – radios, air conditioners, hair dryers – possess "controls" that allow people to utilize them. Knobs, switches, levers, buttons and the like are part of the makeup of such devices. It is obvious to anyone that the knob on the toaster will operate the toaster and not the television set in the other room. This is obvious because the controls *are part of* the object itself. They are, in effect, one and the same.

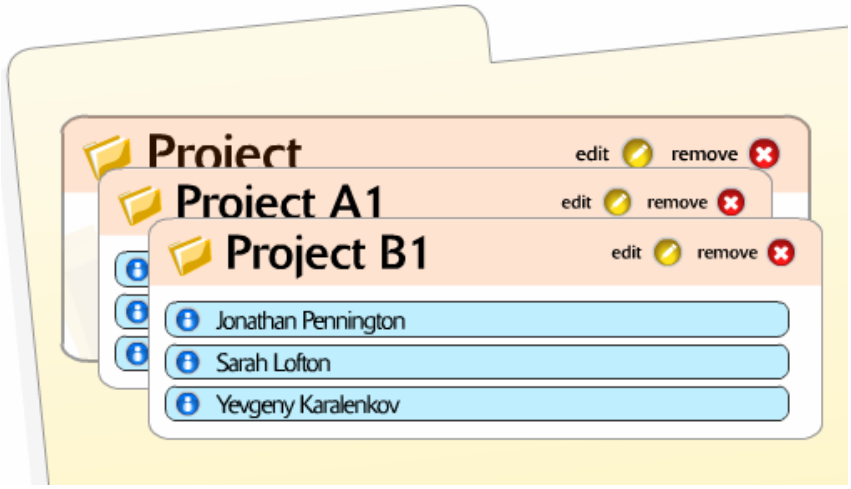
5.5.2.2. Knobs Applied

Many *information objects* can be deleted, copied, forwarded, marked as important and the like. Like the real world, we can *do things* to these *information objects*. If a user is confronted with a screen that contains an amalgamation of various *information objects* (some of which live inside other *information objects*), it is important to use visual cues to *attach* the controls available to that object.

This can be done in numerous ways. One of the simplest is to make certain that controls reside within that *information object* and placed in a location that indicates that the consequences of using these controls apply to the *whole* object. An obvious location is alongside the headline of the object.

5.5.3. "He Looks Just Like His Father": Object Instances

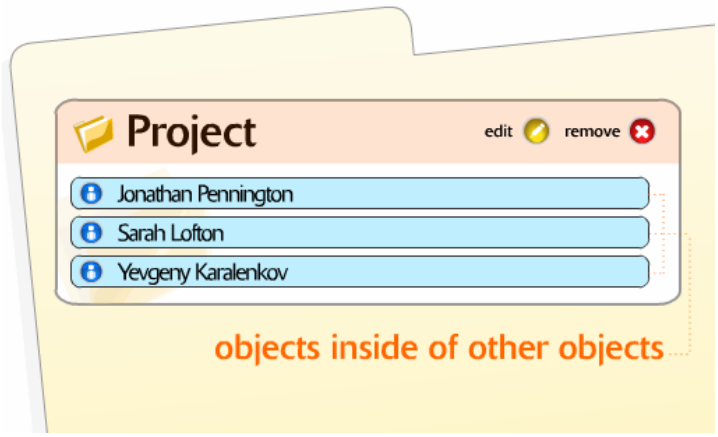
Very often, users manipulating *information objects* are working with *instances* of a particular object type. For example, a human resources system provides information and controls around the Accounting Department, the Sales Department and the IT Department. All of these *information objects* are instances of the *Department Object* type. Once *information objects* are defined, it is worthwhile to attempt to lay out which visual cues will be used to coax the user into perceiving an object *type* vs. the particular *instance* of an object.



Various techniques can be used to cue the user that a particular object is merely an instance of a particular object type. One of the most effective means is to establish a clear icon vocabulary that represents the various object types that reside within a system. Color is also an effective means of signaling that an object is of a particular object type.

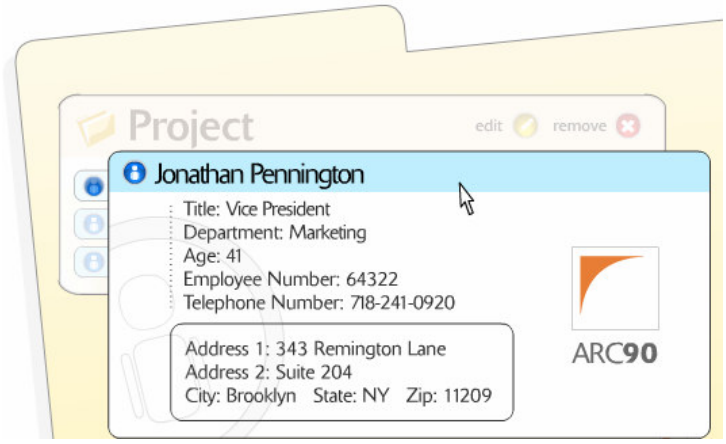
5.5.4. Objects in Objects : Composition

Another common scenario is the existence of one or more objects within another object. A simple example is the shopping cart as a container object. Within your typical shopping cart are numerous products.



5.5.5. Motion

In the real world, physical objects persist. That is, they do not magically disappear and reappear elsewhere. Their movements or motion are fluid. Thankfully, this makes it a lot easier for us to understand and interact with these objects. In other words, our world is richly animated. Rather than having things pop up or blink in and out of our field of view, they move in a continuous motion.



While computers are fully capable of animation and motion, they are seldom used effectively to convey the persistent state of *information objects*. This is an often overlooked technique but a critical one. When objects on a computer or device disappear even momentarily (whether due to our own actions or some other events), the user is forced to "reconnect" the consequences of what just happened. The more significant the change, the more work the user has to do to fully understand the implications of what happened and re-orient himself.

5.5.5.1. Smooth Operators

To mitigate the above scenario, it is useful to use transitional animations wherever possible. By doing so, even complex changes are more easily digested by the user without much explanation. An example of this can be found in Apple's interfaces (in both their desktop and iPod products). When a user selects an option in an iPod, the screen does not flash to another screen. Instead, it simply slides off to reveal the next level. Similarly, on the Mac operating systems, when a program

window is minimized, it doesn't merely disappear. Instead, a "slurping" effect is shown, effectively associating the application with its sibling representation in the task bar below.

5.5.5.2. The Web

When it comes to applications, there is no worse software platform than the World Wide Web. In many ways, the web was a step back from the richer desktop applications we became accustomed to. Users tolerated this because the web introduced far greater reach than your conventional desktop applications.

Nevertheless, any web application with even a semblance of complexity is notorious for constantly jarring the user out of their experience every time a new page is loaded. Most events that require communication back to the server are handled by effectively flashing the application white and re-drawing it with the new changes in place. This is the root of one of the greatest failings of the web as an application platform: the user is constantly forced to re-evaluate where they are and what has happened when a page reloads. As stated earlier, real world objects don't just disappear and reappear, they move in continuous fluid motions. In contrast, the web presents these frozen snapshots in time. The user is left to fill in the gaps.

As a result, the web application experience is less perceived as an exercise in manipulating objects than as a trip through a timeline of moments. The true experience – that of manipulating and working with these objects – is replaced by an artificial one: traveling back and forth through a collection of pages. The notorious reliance on the browser *BACK* button is testament to this. The web browser imposes one significant artificial *information object*: the web page.

5.5.5.3. Drag & Drop

An excellent interface mechanism that has also fallen by the wayside in light of the popularity of the web is the ability to drag and drop items. This mechanism is especially powerful because it reinforced the illusion of volume and mass that *information objects* are perceived to possess.

5.5.5.4. Zooming

Another excellent use of motion is the ability to "zoom" in on an object. This is especially useful when a user is presented with a line-item view of many objects of a similar type. Today, when a user clicks on a line item, they are typically taken to a "detailed view" of that object without much transition. This view often presents more detailed information or controls about the particular object.

A more effective way of connecting these different views is to simulate the action of "zooming in" on a particular object. Rather than flashing to another screen, the object is literally lifted from the line-item view and enlarged. To reinforce, the line item view (now in the background) can be grayed out or blurred, thus providing context but forcing the user to focus on the object that the user has brought closer.

6. Object-Oriented Thinking

Much of what has been put forth here may sound familiar to those versed in object-oriented thinking. Object-oriented concepts have been around for a number of years in the software architecture and programming world. Object-oriented thinking demands that programmers take the problem domain before them and describe the objects that are involved. Once the objects are established, their characteristics and capabilities, along with how they interact with one another, are defined and implemented in programming code.

This paper does not suggest that the object-oriented principles should simply be applied to interface design wholesale. Rather, the motivation behind thinking about abstract systems in terms of a set of objects is the same. The guidelines put forth above - delineation (encapsulation), composition, controls (methods) and inheritance – all bear similarities in both rationale and benefits to the object-oriented programming world.

7. User-Centered Design Revisited

The *information objects* approach is by no means a suggested replacement for user-centered design (or any other design discipline for that matter). It is instead a new "weapon" to add to the interface designer's arsenal. Whereas user-centered focuses on the specialized skills and prior knowledge of existing users today (often within some work environment context), information objects focuses on some of the more basic yet universal "skills" that most of us attain possess.

8. Conclusion

The technology explosion brought many innovations and efficiencies to the world. It also introduced new challenges in understanding how best to bring that power to the users without forcing them to cope with the intricacies of the systems that serve them. The more technology can do, the greater the challenge of bringing that power to users without overwhelming them. The primary driver behind *information objects* is to deliver that power in as familiar a manner as possible. By forcing machines to interact with us on our terms, we're able to bring the benefits of technology to a broader audience more successfully. The result are products that are perceived to be "locked in" to how people think without requiring too much mental labor on the part of the users. Along with user-centered design, the concepts and guidelines outlined above provide a powerful arsenal in designing intuitive user interfaces.

9. Bibliography

Miller, George. "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information." The Psychological Review, 1956, vol. 63, pp. 81-97.

Baddeley, A.D., Hitch, G.J. (1974). Working Memory, In G.A. Bower (Ed.), *Recent advances in learning and motivation (Vol. 8, pp. 47-90), New York: Academic Press.*

Sweller, J. "Cognitive load during problem solving: Effects on learning." Cognitive Science, 12, 257-285 (1988).

Sweller (1999). *Instructional design in technical areas*, Camberwell, Australia: Australian Council for Educational Research.

http://java.sun.com/docs/books/tutorial/java/concepts/index.html

http://en.wikipedia.org/wiki/Desktop_metaphor

http://www.sitepoint.com/print/real-history-gui