Metaphor-Based Interfaces

Hayden Young
University of Auckland
Department of Computer Science
hyou030@ec.auckland.ac.nz

ABSTRACT
This paper looks at what metaphor is in user interfaces and how they affect user's performance. The definition of metaphor is analysed in light detail then four interfaces that are based on metaphors are also analysed but in heavier detail. The interfaces analysed are Fold-and-Drop, Boomerang, DocPlayer, and BumpTop. Both strengths and flaws given to the interface because of the metaphors used in these systems are discussed based on what Marcus and Hamilton say about metaphors.

Author Keywords
Metaphor, user interface

INTRODUCTION
Graphical User Interfaces are now a big thing in computing and getting new users used to them is one of the challengers the designer has to overcome. This is where basing an interface from a metaphor comes in. To explain what metaphor is and what the intention is when used to design user interfaces, sources from Marcus and Hamilton will be used.

Four attempts to use metaphor in user interfaces will be looked at. The first is Fold-and-Drop, a technique developed to make drag-and-drop operations easier by using a paper metaphor [3]. The second is Boomerang; another technique developed to make drag-and-drop operations easier and improves on Fold-and-Drop [7]. The third is DocPlayer. Whilst it doesn't use a physical metaphor it achieves its goal of document management by being based off a metaphor of media-players [6]. The last is BumpTop, a 3D desktop that uses a number of metaphors like piling and physics to organize files and make the work space look more like a real desktop [1].

This paper is split into nine parts: what metaphors are, the purpose of metaphor in user interfaces, what motivated the developers to develop their interface, Marcus's three guidelines for metaphor design, how the four interfaces differ from the metaphor and if it's a bad difference, the flaws in the interface caused by the metaphor, a summary, and then finally shortcomings for metaphor-based-interfaces.

WHAT A METAPHOR IS
When asked "what is a metaphor?" you might give the answer that gets drilled into children's heads when they are in a high school English class. That is "a metaphor is when you say something is something else". In fact "gets drilled into our heads" is a metaphor in itself. Obviously our English teachers never took a drill to our heads and inserted the meaning of metaphors into them but it was mentioned so much it's kind of the same. In fact this is the mismatch that Hamilton et al. talk about which alerts people to look for similarities between the things mentioned [4].

The above definition is the definition if you want to use metaphor as a figure of speech. So what does a figure of speech have to do with user interfaces? Perhaps a more elaborate answer to "what is a metaphor?" would be to say "they utilize well-understood concepts of attributes from one domain to make points or provide insights about another" [5] or they "are the fundamental concepts, terms, and images by which and through which information is easily recognized, understood, and remembered" [4]. For a history of metaphor I refer the interested reader onto Hamilton [4].

PURPOSE OF METAPHOR
In designing user interfaces it is argued that metaphors help quickly turn a novice user into an expert user by showing complex things as something more familiar [5].

With a metaphor the user is expected to instantly know what certain pieces of a user interface is, how it operates, and what its purpose is. Of course it is possible that metaphor can have the reverse of the intended effect or may be so strange to users that it gets confusing. Hamilton describes this with Macintosh's "trashcan" metaphor, which Marcus says was probably used because of some programmer's culture speaking of "garbage in, garbage out" [5]. User's can drag discarded files into the trashcan and are able to retrieve them so long as the trashcan hasn't been "emptied". The trashcan can also be used to eject floppy disks when the appropriate icon is moved to the trash can. This "broke" the metaphor in a troubling way since it seems

like you are discarding all the work that is on the floppy disk says Hamilton [4].

Marcus warns that should a metaphor be inappropriate (by being unbelievable or too foreign to the user) then "the user will become confused, disinterested, distracted, bored, and antagonistic to the message carried by the metaphor."[5] Being confused is certainly what most users would have felt with the trashcan being able to eject floppy disks.

MOTIVATION FOR DEVELOPMENT

The metaphors that are developed in the four interfaces mentioned are all aimed around making it easier for a user to perform common operations. Dragicevic [3] and Kobayashi and Igarashi [7] aim at making drag-and-drop operations easier, while Agarawala and Balakrisnan [1] and McGee and Foo [6] attempt to improve on file management.

In the case of the drag and drop operations Dragicevic [3] tells of how drag and dropping an object between windows can be a problem when the target window is partially or totally hidden. Current techniques that may be employed to get around this include dropping the object on the visible part of the target window, using the cut and paste operation, rearranging windows so that they are both visible, and using the Windows's alt-tab during the drag operation.

Dropping the object on visible parts of the target window relies on the assumption that the target window can be identified, and the target is not totally hidden. Rearranging windows requires more effort than really needed. Using cut and paste requires giving the target window focus, and as mentioned by Kobayashi and Igarashi [7] cut and paste involves either keyboard short cuts or context menus, which is not desirable on small-screen devices. Using Windows's alt-tab is said to be intricate.

After Dragicevic developed Fold-and-Drop Kobayashi and Igarashi [7] identified the same problems for drag-and-drop operations. He also said that Fold-and-Drop is insufficient as users may need to scroll a window to a distant target. Opening a sub folder or change an active tab is also not possible to do while dragging an object. Therefore Boomerang was developed to overcome difficulties of the drag-and-drop operations.

McGee and Foo [6] recognise that keepings lots of files is typical for a typical user and that "'the need for more intuitive, flexible, and effective management-systems is becoming more and more evident". Files become hard to manage because users are forced to categorize their files into folders and there are limited tools aid file management. Storing and retrieving files can be difficult should the file be able to fit into two different categories. An example of this is given by having two folders, one for horses and one for people, and deciding which folder a file about people on horses should go into. Therefore they have developed a solution called DocPlayer, "inspired by the control-metaphor of media-players" [6].

Agarawala and Balakrisnan [1] identify that the "casual organisation, prevalent in the real world, differs greatly from the GUI desktop which forces users to immediately file their documents into a rigid hierarchy". Because of all the effort put into filing it's more likely that a user will keep old files that have low value. Therefore they developed BumpTop, a new way of presenting the desktop metaphor interface by making a 3D interface with a physics engine. Also BumpTop looks more like a real desktop unlike the current look of desktops.

THE INTERFACE AND THEIR METAPHORS

The four interfaces will be introduced separately so it is easier to see the mapping between the metaphor and the interface.

Fold-and-Drop

Fold-and-Drop uses a paper metaphor. Imagine you have a pile of paper and on the top sheet you have a post-it note. Now you've decided that the post-it note doesn't belong on the top page but rather on some page further down in the stack. So what do you do? You flick through and lift the pages until you find the target page and you stick the post-it note there. The same thing applies to fold-and-drop. The windows represent paper, and being on top of each other represents their position in the pile of paper. Now you have an object you want to be put into some other window, you'll flick through the windows like they are paper until you find the right window and drop the object there.

The interface has many features to it mapping to the paper metaphor.

When first searching through a pile of paper people tend to pick up the corner of the paper before deciding whether or not to remove it from the pile. This is represented by something called a *transient fold* [3], which becomes visible when the cursor leaves the window for a short time before it folds back (see Figure 1).

When someone decides they want to take a page off the pile or lift up part of it they would take the corner and continue lifting. In the fold-and-drop technique this is done when the cursor is put near the transient fold and only when the transient fold is visible. After all, it's not possible to lift a piece of paper with your fingers without holding onto it first. Orientation of the mouse gesture determines how the fold folds (see Figure 2).

Figure 1. Fold-and-Drop: transient fold (from Dragicevic, 2004) [3]

Figure 2. Fold-and-Drop: confirming and pushing folds (from Dragicevic, 2004) [3]

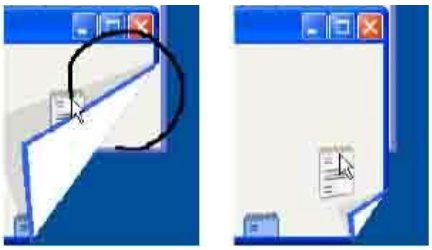

Figure 4. Fold-and-Drop: unfolding a page by moving the mouse around the fold (from Dragicevic, 2004) [3]

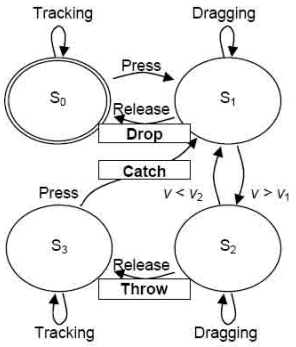

Figure 6. Boomerang state diagram of the actions performed in the technique (from Kobayashi and Igarashi, 2007) [7]

Just like when someone lifts off a page from the pile, the user can make a window disappear if they keep going with folding the window (see Figure 3).

Should someone decide they have lifted too many pages they would just remove their finger out from under the lifted pages. The same with the interface, if the user moves the cursor out from the fold and back onto an unfolded portion of the window then the window will unfold (see Figure 4).

Being able to fold mass amounts of paper in the pile is doable in a physical pile and so such thing is doable in the interface. Hence the user can fold multiple windows all at once (see Figure 5).

Boomerang

The most recognised type of boomerang is the returning boomerang, a throwing stick that that, when thrown correctly, returns to where it was thrown. During the time you've thrown it you can do other things like throw another boomerang. This is the action of a boomerang, and the Boomerang interface uses this idea for throwing objects in this manner instead of using drag-and-drop.

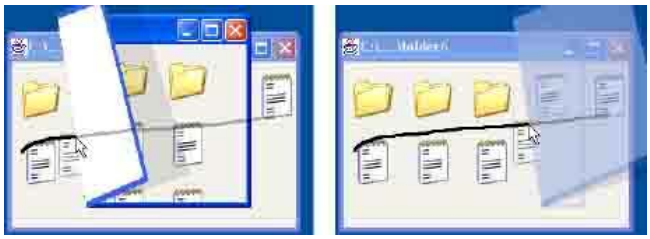

Figure 3. Fold-and-Drop: discarding a window by lifting it off (from Dragicevic, 2004)

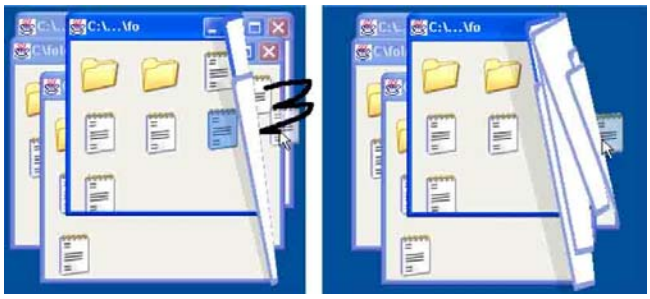

Figure 5. Fold-and-Drop: manipulating multiple folds, (from Dragicevic, 2004) [3]

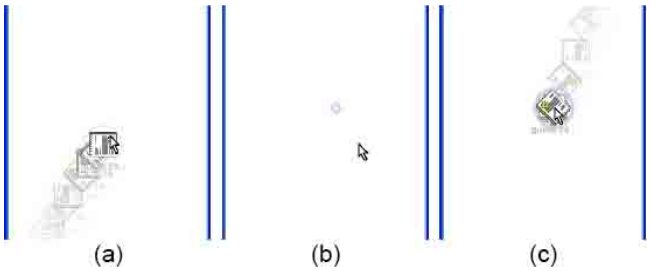

Figure 7. Boomerang: an example of throwing and catching an object (from Kobayashi and Igarashi, 2007) [7]

In reference to the Boomerang state diagram (see Figure 6) S_0 is active when nothing is happening and the user hasn't picked up a file. Then transition S_0 to S_1 represents the time when the user picks up a file. At this point it's the same as someone picking up a Boomerang. In the implementation when a user picks up a file the file is animated as spinning.

To throw a Boomerang one must first move their hand at a certain speed, if they don't then the Boomerang won't go anywhere. Moving the hand fast is represented by the transition S_1 to S_2 should the user move the mouse faster than a user defined threshold. If the user changes their mind and decides to slow how fast they are moving the mouse then the active state will become S_1 again.

When holding a Boomerang and moving your hand fast you need to let go of the Boomerang or it won't go anywhere. The same with the Boomerang interface, if you don't let go of the mouse button when the mouse is moving faster than the threshold then the file will not be thrown. Throwing the file is represented by the transition between S_2 and S_3. Should the threshold not be met then the file will just drop where it is at the time (see Figure 7).

Figure 8. Boomerang: throwing multiple objects (from Kobayashi and Igarashi, 2007) [7]

Figure 10. Boomerang: prompts for copy and deletion (from Kobayashi and Igarashi, 2007) [7]

Figure 12. DocPlayer: a partial view of a media player (from McGee and Foo, 2003) [6]

When the file is thrown a translucent growing and shrinking circle will appear at the point of where the file was thrown. Whilst the file is thrown the user can do other things like finding the target folder. To bring back the file the user just has to move the mouse to the circle. This is the same as when someone stands where they threw the Boomerang and it comes back to them. Multiple files can be thrown in this manner whilst other files have been thrown (see Figure 8).

There are some advanced features to this interface. The first is grouping. Users can create groups by dropping objects onto the representative circle of another thrown object. Groups can be controlled by grabbing the circle at the centre and objects can be removed from the group if dragged out separately (see Figure 9).

Figure 9. Boomerang: adding objects to groups (from Kobayashi and Igarashi, 2007) [7]

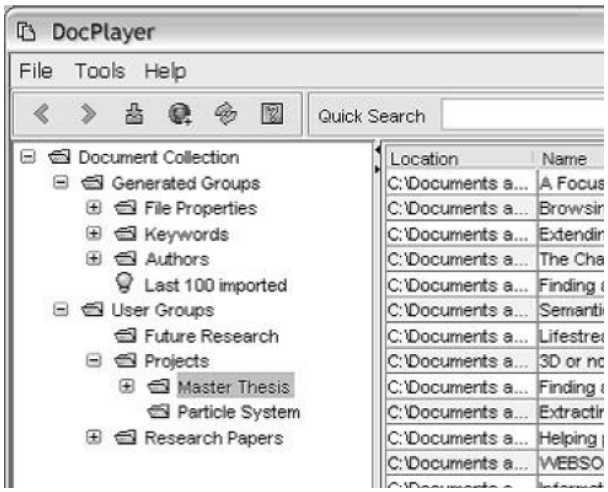

Figure 11. DocPlayer: a partial view of the interface (from McGee and Foo, 2003) [6]

Two visual cues will appear at the upper and lower centre of the screen when holding an object. The top one represents copy, the bottom represents deleting. Throwing things up will copy and object and the original will go back its original place, and throwing down gives a prompt for deletion (see Figure 10).

DocPlayer

DocPlayer is a very simple interface. With a media player you can add media files to it and group them into certain groups, play them, and remove them from play lists. DocPlayer uses much the same concept in that you can import documents into a document collection where users can manage the collections and open and view documents in the collection.

With media players such as iTunes a user can create play-lists and smart play-lists. A similar thing can be done in DocPlayer where users make groups and smart groups. Groups have the same function as play-lists, where the user can add/remove documents to groups the same way they can add media files to play-lists. Smart groups are also implemented in DocPlayer. A smart group is a subgroup of groups which contain documents that match a specific query. Quite like how Smart Play-lists work in iTunes where a user can filter media files that only contain certain properties such as artist or album [2].

Quick search can be done to enable queries of different sorts to be performed [6].

Figure 13. BumpTop: a view of icons on the BumpTop interface Here we see a pile of photos (bottom left) and (fromasually arranged (top left) and crumpled up (top right) windows (from Agarawala and Balakrisnan, 2006)

BumpTop

BumpTop takes the desktop metaphor to a more true-to-the-metaphor level. It's a 3D desktop that looks like a cubicle that uses a pen as its main input device. The interface has shadows and a 2½D view of a 25° desktop and objects are scattered all over the place or in piles (see Figure 13).

On a physical desktop people can have their books, folders, and papers scattered all over or in nice/messy piles. This sort of behaviour is captured in BumpTop by having icons scattered around the workspace which the user can throw around. Objects are represented as squashed cubes that are texture mapped. The texture mapping allows for quick browsing when documents are stacked up on top of each other, similar to when someone piles a whole lot of books on top of each other and scans the spine to see what's in the pile. Icons can be dragged and thrown around and when they collide they displace one another. Windows can be crumpled up like paper, and both windows and icons can have their size changed consequently making them heavier and less mobile when thrown.

People can stack paper and books in different orientations on purpose to show the document's importance. This

Figure 14. BumpTop: Pile with items rotated and pulled out for emphasis. (a) In the real world (b) In BumpTop (from Agarawala and Balakrisnan, 2006) [1]

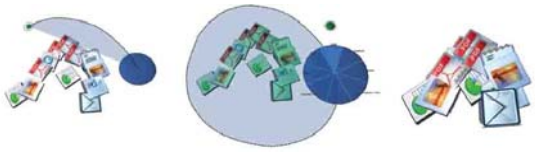

Figure 15. BumpTop: the lasso tool grouping together some objects with a menu that appears when the lasso is complete (from Agarawala and Balakrisnan, 2006) [1]

behaviour is mapped into the interface by allowing users to rotate or out objects slightly from piles (see Figure 14). Piles can be created with the lasso tool which also has additional menus for additional functions (see Figure 15).

Much like how users can pile their papers into certain piles, users can do the same with objects. Grouping objects like this is done with the lasso tool, a drawing technique used to encapsulate various objects. After creating the group the user can interpolate between a messy and tidy pile and see the contents of the pile with a grid view, fish eye view or fan layout. The user can also "leaf through" a group as well as compressing higher items in a pile to view the ones beneath (see Figure 16).

If a person can't remember where they put a specific document but sort of know the location of the pile it's in then they could dump the pile onto the floor and sift through the documents. In BumpTop there is a technique called *Exploding piles* [1] which takes a pile and explodes it into a hovering grid view.

A feature called *PressureLock* [1] allows for objects to be

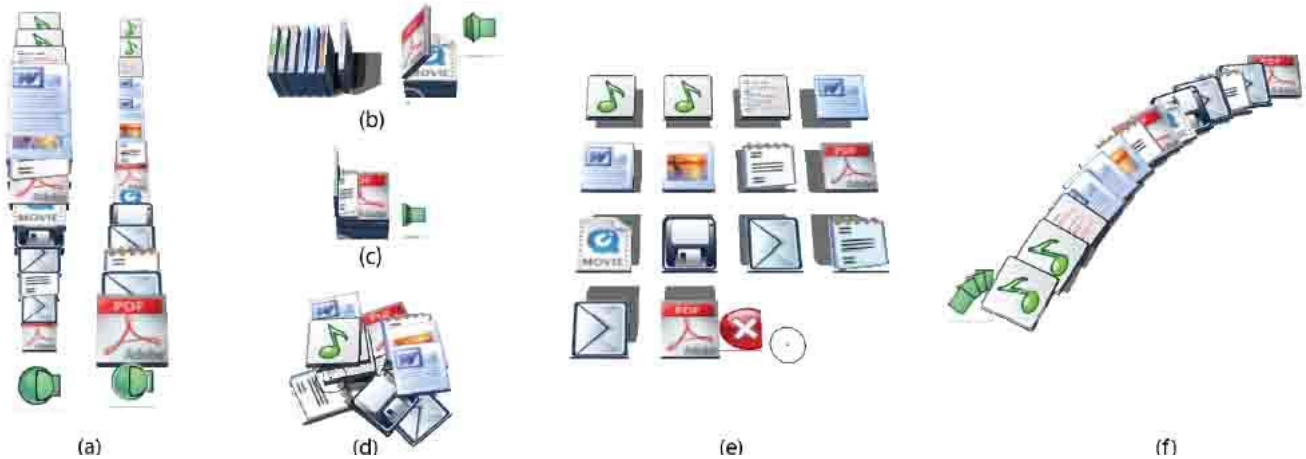

Figure 16. BumpTop: pile browsing is triggered by six different widgets. (a) Fisheye. (b) Leafing through like pages of a book,. (c) Compression Browsing higher items to view items below. (d) Interpolating between messy and tidy piles. (e) Grid browse, locked down for further manipulation with PressureLock, (f) Fan out on a user drawn path (from Agarawala and Balakrisnan,

Figure 17. BumpTop: Axis alignment to enforce a tidier appearance. The 'shelf' (left) was made by pinning up a rotation-locked item (from Agarawala and Balakrisnan,

held in place when a user applies enough pressure with the pen. This also allows for objects to be pinned up to the walls and rotations to be locked. Locking an object's rotation allows for them to constantly stand up (even when bumped by other objects) (see Figure 17). PressureLock is shown with a pressure circle with an inner circle that increases in size as the pressure does.

REVIEW OF METAPHORS

Marcus says that "achieving the right mixture of metaphorical references in a complex user interface is a design task." He also says that three proven effective principles for visual communication can be used to assist in designing metaphors. The three principles he outlines are organisation, economy, and communication [5].

For the metaphor to be organised it must be simple, clear, and consistent [5]. Simplicity is definitely present in the fold-and-drop, Boomerang, and DocPlayer interfaces as they are only based on one metaphor and the metaphors chosen are very simple concepts. BumpTop is based mainly on a desktop metaphor, but it also adds in a few other metaphors with the lasso tool and paper-like objects. Yet it is simple enough for its intentional purpose. Clarity between the interfaces and their metaphor are easy to see. Both DocPlayer and its metaphor have been programmed on the computer so it's easy to see the mappings. The other three interfaces have more complex metaphors but they also map clearly onto it. Seeing windows as paper and objects as Boomerangs or physical entities on your desk gives a clear and easy to see metaphor in the respective interfaces. The implementation of metaphors is consistent throughout the interfaces other than the time that the physics engine is turned off in BumpTop.

The economy of the interfaces is achieved if the interface maximises the effectiveness of a minimal set of metaphors [5]. In other words, there shouldn't be too many metaphors and the metaphors that are used had better portray the intended meaning well. This is clearly achieved by all four interfaces as none of them have any pointless metaphors when trying to achieve their goals. Fold-and-Drop, Boomerang, and DocPlayer are all based on one metaphor and they are very simple. BumpTop being a more complex interface requires more metaphors, but nonetheless all metaphors are relevant to the desktop.

Good communication is achieved by matching the metaphors to the user's capabilities such as their needs, desires, education, and social habits [5]. Effective communication of the metaphor to the interface is achieved well in all four interfaces. This is because the interfaces are mapped well to their metaphors. Most people will be able to look at these interfaces and realise exactly what they can do (with the few exceptions if they are new to computers and don't quite understand all the extra features in Boomerang or understand how media players work). Most people would have come across paper before computers and most people would know what a desk is or at least how physical objects interact with each other.

HOW THE METAPHOR IS BROKEN

When designing the metaphor new concepts might be introduced if the metaphor is not enough to capture all functionality for the interface. If the interface at all breaks the metaphor then the designer risks getting undesired responses from users. The example used for both Marcus [5] and Hamilton [4] is the Macintosh's trash can explained earlier in this paper. Using a metaphor which the user "trusts" results in the rethinking certain actions [4].

Fold-and-Drop does not break the metaphor as it contains no features that differ from paper. Everything you can do with a post-it and pile of paper you can do in Fold-and-Drop. Unlike Fold-and-Drop, on one extreme Boomerang which adds features to the metaphor with the copy and delete prompts and being able to add objects to make groups when you've already thrown an object. Such concept is not within a real Boomerang because if one was to throw a boomerang up then it is not expected a second copy would come back down, and throwing it down one would not expect the boomerang to vanish. Nonetheless such features do not appear to hinder the user's understanding of the interface according to the initial feedback. On the other end of the extreme DocPlayer ignores parts of the metaphor it's based upon — and for good reason. DocPlayer only copies the play-list management from media players and how to import files. Should it contain all the controls of a media player then it would also have a volume control and play, fast-forward, rewind, slow forward, slow rewind, and skip buttons. Such things make no sense to contain in an interface for document management (what does it meant to rewind or fast-forward a document?). Reaching into both ends of the extreme is BumpTop, which adds functions such as the widgets for browsing piles and ignores certain parts of the metaphor such as turning off the physics engine. This too is also done on purpose and works well. If the widgets for browsing piles were not there then it would take a lot more effort to sift and search through piles and it would get frustrating. Should the physics engine always be on then in a messy or tidy pile perfectly touching objects would cause

all sorts of unnecessary movements. Also it would be possible to knock over piles by throwing objects at them and the user would probably get frustrated if they had to keep putting objects back into a pile [1].

FLAWS CAUSED BY THE METAPHORS

Not everything is perfect and these interfaces do not go without flaws caused by the metaphors. Marcus [5] and Hamilton [4] say a metaphor can restrict the actions of the user. But it is also apparent that the metaphor can restrict the functions of the interface or get distracting.

Fold-and-Drop has the flaw pointed out by Kobayashi and Igarashi [7] and that is the user can not perform other tasks while still holding a file to be moved like renaming a few folders. This is a fault by the metaphor as people can't just start whiting out some words and write over it while holding a post-it note.

The initial feedback from Boomerang mentions that the spinning of the objects gets distracting [7]. The spinning is mapped from boomerangs spinning when being thrown. Also if a user lets go of the object at a place where they need the mouse to perform a job then the objects continuously coming back every time the mouse hits the circle could get annoying (although this can be fixed by letting the mouse go in a more efficient location).

DocPlayer does not appear to have any particular flaws caused by the metaphor. Even if a user is new to computers and has never seen a media player the concept of creating groups and smart groups is easy to pick up.

BumpTop's pile metaphor is its biggest flaw. Piles do not scale up very well and BumpTop can get overcrowded pretty fast. Also icons do not have names so it's impossible to distinguish between multiple files of the same type. Developers and animators will generate millions of files for a single project. Most users have thousands of media files and the widgets aren't particularly suited for thousands of files. Even if this interface is useful for new computer uses (as implied by the user evaluation) this interface would not do well in the industry or for people who use the computer a lot for their own things.

SUMMARY

Metaphor-based interaction has its strengths and its flaws. Whilst metaphors will help new users adapt to new systems a metaphor might not be able to capture all the features that the interface can use or it might get in the way and prevent possible functions or events. Four interfaces that rely heavily on a metaphor have been analysed against guidelines by Marcus and the purpose of the metaphor has been achieved. Two of the interfaces aimed at making drag-and-drop operations easier (Fold-and-Drop and Boomerang), the other two aimed at making document management easier (DocPlayer and BumpTop) and BumpTop also aimed to make the virtual desktop an easier interface to work with. Metaphors should be used lightly in designing interfaces (as Fold-and-Drop isn't the best solution to drag-and-drop operations) since new concepts or functions will inevitably have to be introduced for complex tasks (shown by Boomerang, and BumpTop).

SHORTCOMINGS OF METAPHORS IN USER INTERFACES

Metaphor-based interfaces might be a good way to introduce new ideas but sticking strictly to it can cause problems. As demonstrated by Fold-and-Drop sticking strictly to a metaphor may not completely solve the problem. Sometimes only having part of a metaphor like in DocPlayer may be better than being true to the whole metaphor. Nonetheless, having an interface based off from a metaphor will most likely have the designer introducing new functions that the user has never seen before (such as copy and delete in Boomerang and widgets in BumpTop). Being based on a metaphor will restrict the user into rethinking certain actions or thinking that something isn't possible when it is. It still isn't clear if metaphors should be avoided so they shouldn't be heavily relied upon.

REFERENCES

1. Agarawala A., Balakrishnan R, (2006). Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles, and the Pen. Proceedings of the SIGCHI conference on Human Factors in computing systems. 1283 – 1292.
2. Apple iTunes, (2001). www.apple.com/itunes/
3. Dragicevic P, (2004). Combining Crossing-Based and Paper-Based Interaction Paradigms for Dragging and Dropping Between Overlapping Windows. Proceedings of the 17th annual ACM symposium on User interface software and technology. 193 – 196.
4. Hamilton A, (2000) Metaphor in Theory and Practice: the Influence of Metaphors on Expectations. ACM Journal of Computer Documentation (JCD), 24(4). 237 – 253.
5. Marcus A, (1994). Managing Metaphors for Advanced User Interfaces. Proceedings of the workshop on Advanced visual interfaces. 12 – 18.
6. McGee K., Foo J, (2003). DocPlayer: why doesn't your desktop play this way? Linköpings universitet http://citeseer.ist.psu.edu/738493.html
7. Kobayashi M., Igarashi T, (2007). Boomerang: Suspendable Drag-and-Drop Interactions Based on a Throw-and-Catch Metaphor . Proceedings of the 20th annual ACM symposium on User interface software and technology. 187 – 190.