

Branded Software Experiences* – Next Generation Brand Advertising for the Web

*A White Paper for Brand Advertisers
and Agency Professionals*

Jackson Fish Market
Handcrafted Software Experiences
www.jacksonfish.com
whitepaper@jacksonfish.com

**a.k.a. Useful Advertising, Branded Utilities,
Branded Interactive, Branded Destinations, etc.*

This paper is for brand advertisers. By brand advertisers we mean the marketing leaders at companies with key consumer brand assets as well as the talent at the agencies and media buyers that work with the marketers helping them get their message out. Since we wrote the first version of this white paper the notion that brand advertisers would sponsor deep and empathetic software experiences, essentially “useful advertising” was uncharted territory. Now it’s the logical conclusion of marketers who are seeing that merely translating old media tactics to the internet are expensive and ineffective relative to engaging audiences with value and utility.

The evolution of brand advertising on the web – how did we get from banner ad to branded widget to branded web app?

It didn’t take long after advent of the internet for marketers to look at it as a medium for advertising. And as most creators do when they see a new medium, they create for that new medium using the techniques and approaches that have served them well in more established mediums. In our specific example this means that marketers essentially converted billboards into banner ads and 30 second TV commercials into viral videos (or into 15 second TV commercials run uninterrupted before other people’s viral videos).

As happens with any new medium, after an initial phase of exuberance and random exploration, the creators start to learn the intricacies of the medium. And the internet, due to its fundamental identity as an enormous piece of distributed, networked, software has intricacies aplenty. They’re relatively well known but worth recounting:

- creation is cheap to free, and open to all
- distribution is cheap to free, and open to all
- interactivity is the default

These properties are being understood by marketers more and more every day. But our collective understanding is far from complete. In many cases marketers understand how sharing and social media can amplify their messages, but still end up delivering their messages in the form of billboards and television spots adapted for the internet. And to a certain extent the properties on the web are to blame.

Online businesses whose revenue comes from advertising look for standardization and scale in order to grow. Standardization and scale (as in most mediums) can't help but lead to some degree of homogeneity and lack of authenticity. Filling rectangles on the web with banners touting your product has long been suspected as not the most effective means of communicating with an online audience. The logical next step is to add interactivity to those banners. However shooting ducks in a banner ad is not exactly earth-shaking innovation.

The next innovation was seemingly innocuous, but in fact much more powerful than most people realized. The innovation was the ability for the user to take this "interactive banner ad" and put it wherever *they* chose. In other words, not just anywhere on the page the advertiser chose, but rather on any page that the user controlled. In effect, this is what we now call widgets.

Why was user control over placement of the advertiser-created content so critical? Because at the moment that advertisers started relying on users to place the ads on their own web pages, they had to make those ad units actually useful. And this is the moment of conception for what we expect to be an eventual avalanche of advertiser-created/sponsored useful "content" on the web.

We believe that the evolution of brand advertising continues at breakneck speed. It's true that the widget is an evolution of the banner ad (once you've added interactivity and the ability for users to place them in convenient locations). But what is the destination of this journey? Widgets are mini-applications that users embed in the web page of their choice. Their constrained screen real estate makes them ideal for embedding. That said, brand marketers who are deploying widgets successfully are already realizing that those same constraints that make widgets consumable also limit the amount of engagement they can generate with an audience. Why not create entire useful web applications that engage audiences exponentially relative to their widget siblings? In fact, these new web apps can have their own fleet of widgets that bring new users back to the "mother ship".

Once marketers understand that this new generation of advertising is not just banner ads made large (and larger) but has a fundamentally different contract with the audience, both marketers and the general public will start to truly get the benefits of this new advertising medium.

For online brand advertising to evolve, the worlds of content and software must merge. (And it will be ugly for awhile.)

We've discussed the evolution from banner ads to branded widgets to full blown branded web applications. Many marketers are starting to understand the value of delivering useful "content" to their audiences as a means of engaging them. However, it's worth spending a moment to examine the word "content" and how it's used. It's critical that we share a common understanding so we can take advantage of the new medium.

The following assertions are somewhat generalized but still useful to understand for our purposes. In the software industry content usually refers to stuff made by writers, (as well as photographers, musicians, videographers, directors, and producers). Content is the stuff that software presents, manipulates, finds, edits, etc. And it's considered "easy". In the content world, if software is even given any consideration, it's considered... well... content. And more typically in the content world, software is not given any more thought than the film on which a movie is printed. Software is infrastructure.

Now imagine a world where software and content have to merge in a seamless fashion to create experiences that delight and engage audiences whose attention is at a premium given how many things are vying for it. In Hollywood, studios like Pixar are already creating breeds of hybrids who understand how to use the tech at its best but also know that all the tech in the world won't help a bad story.

Marketing professionals and ad folks are just starting to grow these skillsets. They're hiring the geeks. They're learning the ropes. But there's still a long way to go. For this next generation of ads it's true that the tech (and the brand for that matter) should be subservient to delivering something that the user actually **wants**. But creative people still need to understand the subtleties and possibilities of the technology in order to create said experience. Interactivity is not just an extra feature like THX sound or even 3D. Interactivity completely changes the dynamic. People are not looking for interactive movies. Whether they know it or not, they're looking for great software. And the best software mixes seamlessly with content to create things that users want to come back to again and again.

And this really is the test. Great software makes the user want to come back on their own again and again, essentially incorporating the experience into the routine of their lives. The distractions that many marketers are creating for the web today are nothing more than banner ads on steroids. And they're definitely not anything that most people incorporate into the routine of their lives. If you do get a repeat visit it's because they liked the distraction enough to inflict it on their friends and co-workers.

Consider that for a moment. For many of today's interactive marketers, success is getting people to bug their friends.

Today however, the bulk of interactive content is does not meet the criteria above. These sites often have one or both of the following characteristics that come with their own drawbacks:

- They are dependent on produced content. Producing high quality content is expensive. And even if you can get away with cheaper content, it often gets stale quickly. The production costs, which can be high, become excessive when they turn into recurring costs.
- When they try to go beyond passive enjoyment of content and engage the user to extend their time with the site, the interactivity can often be relatively shallow taking the form of things like poorly executed flash games and the like.

There is nothing wrong with these techniques per se. But given some of the drawbacks listed above, there are alternatives that are worthy to explore (and to be clear, a handful of marketers are already heading down these new paths).

User-generated content is the key alternative to produced content. In addition to being much less costly to produce, not to mention potentially engaging, it can (when a result of a critical mass generating enough breadth) keep the experience fresh and new. Much has been written about user-generated content and clearly distribution and getting to critical mass are key components that enable a UGC ecosystem.

What's truly new is a focus on what we'll call "software" as opposed to mere interactivity. While an excellent complement to a UGC ecosystem, having real software *or* advanced interactivity is a potential source for serious engagement of an audience. We define software as going beyond what commodity sites provide. Don't just enable users to upload their own videos, but create video

editing tools that let users do interesting things they wouldn't be able to do anywhere else. Don't just let users post journals from their trips to exotic places, but let them plot their trips visually using photography and pins on a map of their journey.

Unfortunately, it can be difficult to get this software built. Most of today's interactive agencies (or interactive units at advertising agencies) while producing excellent work, do not always have the technical talent in-house to deliver the kind of advanced software experiences that result in extended minutes of engagement. And often (though not always) traditional contract software development houses don't have the design expertise and understanding to deliver experiences that make an appropriate emotional impact on an audience.

How to identify the right online software vehicle for your brand message.

Being creative is critical throughout all phases of this process, but perhaps nowhere is it more essential than in identifying the right piece of software to express your brand's message. There are several strategies for identifying the right solution. Most important is to distinguish between expensive content creation and much more cost-effective software creation. Software can span the gamut (and some software is actually more content-oriented like videogames).

Identifying the right software to express your brand is no different than choosing which television show to sponsor or which sporting event to support. You are looking for some type of direct or indirect linkage with some combination of your product and your brand. You can have a very specific relationship such as some sort of product placement, or you can have a more abstract relationship like sponsoring a cultural event that conveys the values you want your brand to be associated with in the public's mind.

Let's take some very obvious and simple examples to illustrate. Imagine a Financial Services company like Fidelity sponsoring Microsoft Excel. Imagine Windex sponsoring an Anti-Virus package for your PC – "It disinfects your Windows". Ha ha, but you get the idea. But there are even more interesting examples. How about Amazon sponsoring Delicious Library (delicious-monster.com)? Not to put direct sales links in it, but to connect their brand to people who love their collections of media so much

that they spend time cataloging them. How about United Airlines sponsoring an app to keep track of all your frequent flier miles across multiple airlines and what you can spend them on? Imagine United being so secure in their brand that they made it easy to track your frequent flier miles on other airlines? These examples just scratch the surface though and don't even get into the aspirational aspect of the brands. They are just straight obvious connections.

Here are more abstract (but in my opinion) potentially stronger relationships between brands and software. Think of a car company with a minivan to promote sponsoring an application that gives parents resources to help their kids with their homework. The car company cares about the growth and success of your family no matter what you drive. Think of a beer company putting out its own fantasy football software. Think of Disney putting out family photo album software brought to you by Disneyland. As a user of that software you would know that Disney wants your family to create fond memories even when you're not visiting one of their parks.

It may seem counter-intuitive, but in many ways relationships with brands are like relationships with people. The less obvious connections often show a better grasp of the subtle values of the brand and the underlying needs of the customer. Just like people, brands can seem insecure. And just like people, brands that are confident in who they are, are attractive. In the midst of conceptualizing one of these brand/software relationships you may be tempted to jam your product into the software experience. And in some cases there may be a way to do it that makes sense. But in many more it won't make sense. The product's presence will be obvious and awkward and detract from any value you got from sponsoring the experience in the first place. Lincoln put car ads on their My Dream site [mydream.tv] over and above their strong brand presence. Aside from the problems with the site's design and focus, the car ads just detracted from the premise that Lincoln had built the site to help its potential customers achieve their dreams. Did the presence of the ad sell even a single additional car for Lincoln? I doubt it. Did it take away some of the genuineness that users might have attributed to Lincoln for putting the site there in the first place? I think so. There are places where product placement not only makes sense but actually enhances the experience. (The Price is Right being my favorite example.) But those are in the minority so tread carefully.

When creating a concept for a software vehicle for your brand, ask yourself these questions:

1. Who am I trying to reach?
2. What values are important to me?
3. How can we use technology to make that person's life easier and reinforce that we share their values?
4. How are we making them more productive, effective, connected instead of just entertaining or informing them (not that those can't be key components of a branded software experience)?
5. How will we get them to come back to the experience again and again? (To get the most bang for your buck, why not invest in an experience that becomes a recurring part of the lives of your target audience?)

These questions will lead you down the path of finding the right software that will ultimately accrue positive value to your brand.

How to build an online software vehicle for your brand message.

Despite all the technology, the creation of software is not a science. It's not necessarily an art either. It's more of a craft. And, more often than not, it's a craft where the design and construction of the product are intermingled in the same phase. Anyone who has built anything knows that if you are designing as you go (instead of building from an exact specification) then you lose a significant degree of predictability over the project's timeline (and cost). Even modern buildings with incredibly detailed architectural specifications can go well over budget in terms of time and cost. Most modern software (unless it's commissioned by the Department of Defense) has no such detailed specifications. And the specs that are in place are more "vague wish list" than true detailed design document. I believe that the only perfect spec for code is the code itself.

Before you dismiss this as a copout, there are plenty of ways to mitigate this problem. It's true that you could spend the time to spec out your software properly, but the deadlines of almost any modern advertising campaign wouldn't afford nearly enough time (or budget) to make that happen. The best approach is to keep things small, do things one at a time, have items on your wish list you're willing to cut, don't be afraid to rewrite the software if

necessary, and include budget for multiple iterations as you will learn with each successive release of the application. Careful planning and not making mistakes is one approach, but it's likely not the approach you will end up taking.

Even at the most advanced and successful software companies in the world, careful long-term (think a year) planning, when it works, usually results in so much risk mitigation that there's not much actual product left when the plan is ready for execution. More often than not, major long-term software planning efforts result in projects that go nowhere with even the time spent planning them being a waste. And just think, instead of working with the premier software companies in the world, you'll likely be working with small contract software development shops. Certainly size is no indicator of quality in this case, and many of these small companies can run rings around the big guys. But keep in mind, no matter the size of the company, software is not as simple as it may seem.

Creating software is very different than building most of the content and collateral that agencies have typically produced for their campaigns. Agencies with in-house interactive teams already understand some of this, but deeper software experiences require deeper technical expertise than agencies usually have. Here are some keys to moving quickly, and getting what you want in a reasonable timeframe, as well as some of the differences between developing software vs. developing more traditional brand advertising vehicles.

- Define the smallest (and small is definitely not the same as insubstantial) experience you possibly can, and then refine it to make it even smaller. This is your bottom line.
- Make a list in stackrank priority order of the features you want in your software. The features above the bottom line are the minimum you need to ship v1. Anything below that line can come later. (The most annoying thing about software is also its coolest feature – it's never done!) Your users will tell you very quickly which five features (of the twenty you had to cut to make your v1 dates) you should implement next.
- Your feature list should contain the bulk of the text used to describe the software you're building. Everything else should be in screenshot form. The process of creating software is one of translating ideas into code. Usually text and bitmaps are interim steps. And good ones as they are

often much easier to change than code. Over time the feature list and bitmaps will fall by the wayside and the only representation of what you're trying to build will be the code itself and your list of bugs/suggestions/feature requests.

- Code early. Code often. You don't want to wait until just before launch time to see the actual software the team has produced. Demand weekly if not daily "builds" of your software that you get to see, touch, and play with. No amount of specification or screen design can prepare you for how software actually feels when you use it. Trying it out is an essential part of the process and the only way to refine it for human consumption. Additionally, seeing the software evolve every few days can give you a much better idea of the realities of the software development process and what kind of a trajectory your project is on.
- If you do look at the early versions of the software on a regular basis make sure to give the development team some latitude as the creation of software can be somewhat like the creation of sausage. Kind of messy to make, but tasty in the end.
- And finally, understand that setting dates at the beginning of the project is almost always an arbitrary process. You can wish all you want that your software will be ready by a certain date. Nobody can guarantee dates. Anyone who does has either already written the software or is lying to you (or is talking about a date so far in the future as to be irrelevant). The only semi-reliable way to hit a date is to cut features.

Note: Count the opinions in a room full of software professionals on the best way to write software and those opinions will outnumber the people in the room. The list above is by no means meant to be the only way to get software projects written, or a comprehensive philosophy. It is however intended to highlight for brand advertising professionals some of the realities of building software experiences, and how they may differ from other elements (print, video, etc.) that have been created for your campaigns in the past.

There are a couple of other things to look for when hiring software development talent:

- Attention to detail. Look at past projects from these folks. Do they exhibit a strong attention to detail in their work?

Ultimately it would be best to work with a technology partner that has both the technical and emotional skills to understand how to deliver software with a strong positive emotional experience that is well-engineered.

Unfortunately, it's not often that you find that mix of right and left brain characteristics. Finding out how well they've worked with strong designers in the past can also be helpful.

- Do they understand scalability? While our definition of software spans online and client offerings, it is unlikely in today's consumer landscape that any branded software experience you specify won't be primarily (or more likely exclusively) an online experience. And that means that the more successful and popular your software is, the more computing power and bandwidth you'll need to run it on the back end. Do the folks you're working with have experience at scaling software for large audiences? Can they handle the spike of interest that will likely be caused by the promotional campaign for your new service? Will they have to take the site down to do maintenance on it every time they expand its capabilities? These are all important questions, and any modern software development talent needs to be able to answer these from experience. You should also note that as the usage of your service increases so will the costs of the computing resources required to run it. If your site is delivering the brand's message effectively, this is a nice problem to have, but it's important to understand that even if you've stopped adding features, there are likely many ongoing costs that you'll need to be aware of and build into your budget.

Countless books have been written about both how to develop software on your own as well as how to manage software development projects effectively. Hopefully this short article will give brand advertising professionals a taste of what's involved in actually building a software experience as a tool for getting out their message.

How to measure your results and improve the performance of your branded online experience.

The first step in understanding the benefits you're receiving from your new branded software experience is to understand the actual cost of the effort. Your time sunk as well as the cost of the vendors

you used to build the experience obviously need to be calculated as a one-time cost. And as we mentioned in the previous post, the costs of operating the experience on an ongoing basis need to be factored in. But there's a final component you need to factor into your budgeting – distribution.

In the days of packaged software, distribution meant putting boxed software on retail shelves or sending catalogs through the mail. On the Internet, it means driving traffic to your site, and there's a whole different methodology to measuring success and return on investment. Normally online services advertise once they've built an audience that is big enough to be attractive to advertisers. But once their site has built an audience, they maximize your revenue by offering scalable advertising inventory to multiple advertisers, which makes you just another company competing for eyeballs. The benefit of the branded software experience, versus a typical online service, is that in the best case, you are the only sponsor and thus you accrue all the value to your own brand. Of course, the cost of getting in on the ground floor is that you have to help drive traffic to the site. This is not a difficult challenge, but it should be part of your budgeting. And if the experience is done well, the money you set aside for distribution should shrink over time as the site becomes more popular and builds a self-sustaining audience. If the experience *isn't* done well, people won't spend time on the site, return to the site, or recommend it to their friends, which is why it's important to take to heart the advice we've given you in the other posts.

If you're a large brand, you likely already have a significant audience on your product's website, at least a portion of which you could redirect to your branded experience, and that is a good place to be in. Above and beyond that, there are many traditional avenues for getting the word out and acquiring users for your new service (both online and offline advertising being the main methods). But as mentioned above, if the site resonates with users both from a utilitarian and emotional point of view, then it's possible (and even likely) that your distribution budget is merely a catalyst to get the word out to early adopters and influentials. The rest will follow. And of course, building viral and social capabilities into the experience that you can't find on your typical content site can increase the pace at which usage of the service spreads.

In the best case you're able to measure whether you've succeeded or not by your ability to reduce or eliminate your budget for

advertising the site itself. In the middle case, you may need to continue driving traffic to the site, but the engagement of the site amplifies the time you get to spend with your audience. This is obviously not as cost-effective as the site growing organically on its own, but can still be a cost-effective way for your audience to interact with your brand in an extended fashion.

If traffic to and time on your site is completely dependent on your continued (and possibly increasing) advertising, then the site has failed to engage the audience. No number of viral features will help a site catch on if nobody wants to show it to their friends in the first place. And beyond just eliminating your advertising, if your site is significantly compelling, it should not only retain and attract traffic, but grow its audience over time. Organic distribution is your ultimate goal. If your service is not (to a large extent) selling itself, then all you've made is a fancy billboard.

It's good to set a high bar as we've done above, but we should acknowledge that in this increasingly "noisy" online world you may need to invest in driving distribution on an ongoing basis. But in that situation your ability to retain a percentage of your audience for multiple visits had better be very good. More on that later.

Growing traffic on the Internet is like growing an audience for any product. News items and promotional pushes spike the audience for a time, but you only see the real impact after the first blush of excitement has faded and you find out how much of that audience you've retained. To the extent that you're able to constantly grow the baseline (even if you fade a bit from the high of each spike in awareness) then you're on the right track. Then the question becomes how comfortable are you with the pace of growth, and if you're impatient, how much are you willing to spend to increase it. (And just one note on "spikes", related to a previous post. If you didn't think scalability from day one was important before, you will the day that a news item comes out about your site, causing people to flock to it, only to see that the site is down. These people will never ever return, no matter how good your site eventually becomes.)

There is of course another level of detail that's critical in understanding the mechanics of what your users are doing once they arrive at the site (and thus, how effective your site is). Some key metrics:

- How many unique visitors are you getting per day?

- How many page views are you getting per day?
- How many of your visitors each day are return visitors vs. new visitors?
- How many visitors come once and never come again?
- How many people get to the site and then leave without doing anything there?
- What is the average length of time someone spends on the site? (What does this distribution curve look like? – are 50% leaving immediately, but the other 50% are spending 10 minutes on average at the site?)
- If there's a user contribution component (posting stories or comments about your product, for example), what percentage of users are contributing?
- When users stay on the site, where do they spend their time?
- When users leave the site, where do they leave from?
- What is the usage pattern in terms of times of the day and days of the week?
- What is the geographical distribution of people using the experience?
- If you have viral features, how many of your users are inviting others to the experience? How many are they inviting? How many accept?

These are all relatively standard metrics for anyone operating an Internet site/experience. But you'd be surprised at how many people do not take these numbers as seriously as they should—and often even when advertising professionals know to review these numbers it's not clear they have a deep sense of what the results mean. I have some numbers from existing branded software experiences that I could offer as a baseline, but honestly given the immaturity of this space I'm not sure the numbers would make for meaningful comparisons. Until we have audited and consistently measured results across a broad range of branded software experiences, I recommend these tactics for evaluating each number.

- If your goal is to engage your users on an ongoing basis the single most important statistic is how many of your visitors come back vs. how many show up once never to return. The data on how many people you get each day is simply a function of how effective your pr/advertising is. But knowing how many people see the site and make it a part of their life is showing you whether spending all that money getting traffic was worth it. What's the point of driving

people to a site they abandon after one visit (or even two for that matter)?

- Once you've established that you are getting people to the site and they are coming back more than once, it's time to tune your experience to make their time with your software as engaging and positive as possible. The key metrics helping us answer this question are a) how much time your users spend on the site, b) what they do while they're using it, and c) where (and why) they leave. You should have a sense of how much time a normal person might want to spend in your experience if you've built it properly. Then you need to gauge whether the average time spent is below or above your expectations.

The mechanisms for gathering these metrics should be built into the software from the start, so make sure that the people building the software have an instrumentation plan. Looking at these key metrics, updating your experience, watching the results, and then updating again based on your impact is the essence of how you turn your "great start" into a long-lasting, and incredibly cost-effective, brand messaging monster. Shipping and promoting the experience is not the end of the process, it is the beginning. You must plan for going through this iterative loop multiple times. And you must be disciplined as you do it. Small changes (moving a button a few pixels to the left, making a link bigger or smaller) can have incredible positive or negative effects on the numbers above. You have to make small changes, and watch their effects over time (days not hours) to get a real sense. You are running clinical trials on your branded software experience: Think like a clinician.

Companies such as Amazon who ship sites highly tuned to users habits often run controlled, comparative experiments to see what people like the best. You can do things like offering two versions of your site at the same time and randomly directing users to one or the other so you can compare the results of your iterative changes to your baseline. (These are called "parallel flights.") Patience, and willingness to think outside the box (some positive changes can be very counter-intuitive) are the essential ingredients to success here.

And finally, there are a host of traditional tools at your disposal that can be migrated from traditional brand advertising mediums to the software experience: e.g. sampling your audience to understand demographics, surveys measuring brand awareness and attribution,

and customer satisfaction studies. These tools all translate into the world of software and interactivity quite well.

In terms of understanding how cost-effective your branded software experience is versus your other initiatives, it's critical to define a consistent way to measure across all mediums. Software will likely be only one component of your strategy, as your audience hasn't entirely (or even mostly) migrated to the online world. Calculating the reach and effectiveness you have per dollar spent in each medium, and then comparing the two will also start to give you an idea of whether you're getting acceptable results. Our speculation is that the world of software can be so exponentially effective when done properly, that once you see the results of a branded software experience you'll start counting the days until you can move the bulk of your efforts to the digital world.

Of course, we are in fairly unexplored territory. Our job now is not only to build great branded software experiences but to measure their effectiveness religiously and consistently so that we can together start to understand the dimensions of this new world, so please look at this guidance as a starting point. We may find out that some of these assumptions are incorrect or just off-base. That's a fine result as long as we do our best to understand the reality, and adapt our techniques to take advantage of our learnings. Over the next weeks, months, and years, we will endeavor to do exactly that.

Additional Resources

Some quality branded software experiences:

- Change Everything (www.changeeverything.ca), sponsored by VanCity Credit Union
- Layer Tennis (www.layertennis.com), sponsored by Adobe
- Nike Plus (www.nikeplus.com), sponsored by Nike

Branded software experiences from Jackson Fish Market:

- They're Beautiful! (www.theyrebeautiful.com), sponsored by Vosges Haut Chocolate
- Invitastic (www.invitastic.com), sponsored by Dry Soda
- Carbon Grove (www.carbongrove.com), sponsored by Microsoft Windows Internet Explorer

More experiences and thoughts at the Branded Software Experience Index (www.jacksonfish.com/bsei) and the Jackson Fish Market blog (www.jacksonfish.com).