

UPSIDE-DOWN DESIGN

by John J Harris

What is upside-down design? It is when we put technology at the top and people at the bottom. It is when we try to do complicated things before simple things. A system is upside down when it ignores the great truth that people are more important than technology. So let us consider school portals.

Unfortunately upside-down design comes to us with the notion that complexity is a good thing. It could even be the reason why, years after the technology has been in place, most schools still do not have portals that perform the most basic of e-learning functions. We have tried to leverage teachers into unwieldy systems and we have failed to engage them or the students. We lost sight of the fact that portals can, if designed with the teachers in mind, be immensely powerful and inclusive — provided people actually use them. Engagement is everything and it is teachers who must be engaged first. Without teachers driving the system we really do not have a system at all. So let us look at the elements of engagement.

1. Learn one thing - Learn everything

The software should be designed in such a way that any teacher who manages one function should then have the skills to happily manage all the other functions. By learning one process we learn many. By learning the critical features we also learn the more creative features like reflections, e-portfolio management or wiki management.

For this to be achieved the system must be dogmatically systematic. It must follow the same path to different ends. It must focus on process rather than function.

2. Apply the 100% Rule

When we look at a teacher-driven application, we first need to consider just how many of the teaching staff are likely to use the system. Clearly, an attendance system requires 100% engagement because it is a critical task.

Right from the beginning we need 100% of our teachers using this part of the system. Let me stress this. Partial engagement is not an option. It is all or nothing.

The 100% rule does not just apply to critical tasks. It is the starting point for any system, but it must involve everyone, or no one at all.

3. Do the Simple things First

In our desire to create a massive technology revolution, we have missed the opportunity to achieve small things. How much more we would have achieved back in 2003 if we had implemented a system that did just a few useful things! A system that displays weekly homework tasks on the internet for example — just homework - nothing more. Instead we tried to change the world, and we lost five years in the process. Our aspirations have always been way ahead of our ability to deliver. In a world where technology promises so much, simply putting homework online hardly seems a big enough goal. We want more - and in the process we bit off more than we could chew and we have been suffering from indigestion ever since.

We know that a modern portal can do everything from pastoral care to ordering woodwork supplies. It is like a giant zorb ball made up of ambitious intentions. We ask our teachers to climb inside while someone from Silver City gives a heave. Some might find the experience fun. The rest of us get motion sickness or get thrown out altogether.

Better to encourage teachers to achieve small things first and build on that. We need to allow teachers to learn by their mistakes, to be encouraged by small victories, rather than pushing them off the top of the hill and hoping for the best.

First we engage the staff to achieve something simple, relevant and easy to achieve — and preferably something that we can all mess up without causing embarrassment or loss of confidence. In a portal, this might be an encouraging note to a class or an individual.

Once this is achieved, we can encourage teachers to engage with other more critical aspects of the system.

4. Put Process First

Upside-down design is function-based rather than process-based software. The preferred approach is to teach database fundamentals. By focusing on database principles we can understand how to control the portal, rather than training teachers to use a specific function.

A relational database separates data between different tables (files). We have student data in one table and our class data in another table, for example. Each student (in the student table) is then linked to a class in another table. If we understand where data is stored then we are more likely to understand what makes the database tick. For example; by focusing on a particular student, the relational database will show us everything it knows about that student — her parents, her classes, her academic progress and so on. So the first thing people need to understand is where our primary focus should be. And here is where many people get it wrong. A portal is student-based, so everything ties back to the student. But the student is not the starting point for all actions.

5. Strive for Invisibility

Software should work in the background. When we drive a car we do not have to think about where the clutch might be located every time we change gears. The software designer's role is to make the application disappear. To achieve this the user employs a chain of repeated steps that become second nature, like changing gears in a car. By repeating the same steps again and again the application becomes sub-cognitive. The mark of a good application is not so

much the way it looks, but the way it disappears. There is a delicious paradox here; the less people see of the system, the more they can use it.

6. More Steps Make Things Easier

Software designers who apply a truly consistent discipline understand that repetitive steps soon become second nature, thus allowing us to focus on the job at hand. What slows us down, what distracts us, is the process of thinking about what to do next. This takes our focus away from the job at hand. It is better to have seven simple steps to conclude an action than two carefully considered steps.

The simplest way to work in this type of environment is to focus on one thing and then another in a chain of actions called 'focus steps'. The more complex the application the more focus steps may be required, but the process need not be more complicated. I've heard software vendors boasting that you should only ever be two clicks away from your action, but I disagree. When it comes to clicks, more is less.

7. Focus + Focus + Focus

Let us look at focus steps. We have already mentioned primary focus. If we wish to post a homework item we must first go to the homework section. Likewise, if we are wanting to see all the students in a class then we must focus on the class section.

To view all matters concerning Cassie Harris, our primary focus must first be the section where students are kept :

- a. focus on the **student** section : click
- b. focus on Cassie Harris : click

We can now see if there is any data relating to Cassie Harris in the other tables. This might be her classes, teachers, parents or other. At this stage we do not need to see the content of the tables, we just need to know IF there is content in the other tables and this is where many systems turn upside-down. They clutter our screen with windows displaying data from other sections. They are now showing us too much information, thus diverting us from our focus.

Much better if we simply have alert flags that say: "*Look! Over here. Cassie Harris has something happening in this section*".

Let us look at the steps to view Cassie Harris' classes.

- a. focus on the **student** section : click
- b. focus on **Cassie Harris** : click
- c. focus on **other sections** '*Look! Over here . . .*
- d. focus on the **classes table** : click
- e. now view the six classes to which Cassie belongs (including Calculus 1.0)

That is five clicks, but the same logic can be used to locate any data in the system. You can find everything about a class, the students in the class, the teachers and so on.

We can also create actions using the same process. We will post Calculus homework to the student portal. The process uses the same logic, but adds actions and linking.

- f. focus on the **homework** table : click
- g. **create** a new homework item : click
- h. focus on the **class** table : click
- i. focus on **Calculus 1.0** : click
- j. now **link** Calculus 1.0 to the **homework** item : click
- k. **compile** the homework task : save
- l. **post** the homework task : click

You have just completed something that most schools have spent the past six years fretting about. And the best part is, these simple steps can drive any function in your portal.

Operating a portal has a lot in common with salsa dancing. Once you know the basic steps, you can easily learn a new turn or throw. Now that we have the user at the top and we have restored you to your proper place in the scheme of things, we might be able to catch up lost time. And students and parents will love you for it.