



## Documentation Cannot Compensate for Poor Design

In 1996 I was preparing Release Notes for the latest version of a high-profile software development environment. The Release Notes were being published in the form of a Help file with document and code attachments. As the publication date grew closer, I experienced what I am sure most technical communicators have experienced---the request to include a last minute warning to users.

The Support team discovered that customers who installed the new release would lose critical data and that customers would have issues restoring the data from standard backups. Would I please include a warning in the Release Notes advising customers to write a special backup function and save the affected data before installing the new release.

Back in 1996, my only response was to lobby the Support team to write the backup function and to link the function to the warning in the Help file.

Today, I would say that you can't solve the problem of a poorly designed solution with words. The risk is that the message won't get through and that the design's flaws will prevail.

Twice during the past year, I've been reminded of my Release Notes experience and of what user advocates can do to ensure "good" design.

## I Want My Refund

Some time ago, I went to buy some workshop materials from a large office supply store. The city where the store is located had recently introduced fees for street parking. As a result, everyone wanted to park in the store's free car park, taking up spaces reserved for the store's own customers. The store implemented a *get-refund-if-you-buy* system. You paid for your parking, handed the parking receipt in when you made your purchase and received a token that you took to the refund machine.

The first time I used the machine I remember having to put the token in several times but finally received my refund. On my next visit to the store, I went over to the refund machine only to find a man banging on the machine and shouting loudly. The poor fellow was almost beside himself with rage... over a \$2 refund!

I assumed that the machine wasn't working and was going to forget about the refund when one of the assistants rushed over and pointed out that the man was "doing it wrong". The man completed the task correctly, received his refund, and walked off. I asked the assistant whether the scene I'd witnessed happened very often. "All the time. It's terrible. Customers get very angry and swear at *US*. It's not our fault. I have to stop what I'm doing and dash over before they break the machine."

## Why All the Angst and Disruption



Easy. The machine was badly designed. You had to press the button before inserting the token -- the complete reverse of what you would automatically do.

Someone had obviously recognised the problem and decided to solve it with words.

1. A 3-step procedure
2. Each part of the interface numbered to relate action to object

The *documentation* could not have been clearer... only... people don't read instructions for such a simple task.

What do they have in their hands as they approach the machine? The token.

What do they expect to do first? Insert the token.

You can't correct a bad design with words, even when you anticipate, as this writer did, the flaw in the design.

The store must have realised that words wouldn't stop customer rage because I visited the store recently, and there it was, a new machine.

They have words all over the interface that customers still won't read but that doesn't matter because customers approaching with their tokens

- put the token in the slot
- receive the refund

No button to press. No conflict with their natural behaviour. I asked the assistant how the new machine was going. "Wonderful. No one has a problem now."



## Let Me In! Let Me Out!

Late last year, my niece gave birth to her first child prematurely at 27 months gestation. The little baby was taken immediately to the Neo-natal Intensive Care Unit, the NICU. Amazingly, visits are allowed at the NICU so we gathered outside the Unit, waiting for our one-by-one visit to the tiny babe.



As I stood outside the door waiting, I became aware of the signs, not one sign, not two signs, four signs - all with the same message.

1. "Press button to open doors" [original perspective, sign mounted on the wall outside the door recess]
2. "Please use the BIG RED BUTTON [Finger pointer image] to open the door" [printed, red paper sign beside the button]
3. "PLEASE DO NOT push or pull these doors or they will break" [printed orange and red paper signs, beside each other in the middle of the door near the door handle!]

I just laughed.

My nephew took me in to see the baby--such a little creature, skin almost transparent. My heart sank. How could I have imagined then that the little baby would survive to be the healthy baby she is now? I left the humidicrib area with my nephew, in tears, so upset.

We paused for a moment, then went out.

I didn't notice "the exit signs" that night but several days later, when I visited the NICU again, I saw the signs as I walked out. So many signs!



1. Two original, perspective signs - "Press button to open door"
2. Red and orange signs - "PLEASE DO NOT push or pull these doors or they will break"



3. Red and orange signs - "Please use the BIG RED BUTTON [Finger pointer image] To open the door"
4. A yellow sign near the red button - "PRESS THE RED BUTTON". (Obviously, a lot of people didn't see the BIG RED BUTTON.)

I took my camera on my next visit. A nursing administrator came up to me as I was taking the picture of the exit signs. I explained that I was interested in communication and that I was amazed at the number of signs. Her reply was "they STILL keep breaking the door."

### **Words are Only Part of Successful Communication**

I thought back to the emotion of that first night. Imagine if you were a parent who wanted to get in to see your child. You'd go straight to the door, not to a red button on a wall outside the door recess, and you'd push and pull as hard as you could to get to your child.

Imagine if you were a parent leaving behind a tiny little baby fighting for life. You would just want to get out. Would you look for a BIG RED BUTTON or READ signs scattered in rainbow colours around the door area? Hardly.

The documentation on the refund machine and at the NICU presented the required information in clear, easy-to-understand language.

What can be clearer than "*PLEASE DO NOT push or pull these doors or they will break*"?

The way people behave and will always behave around the NICU is totally at odds with the door opening design and the handles on both sides of the door invite the visitor to pull or push. No amount of documentation will ever reduce damage to the doors.

### **But It's Not Easy to Raise Design Issues...**

Good designs take into account user contexts and the user agendas, and writers are in the unique situation of being able to assess design against these dynamic parameters...if someone will listen.

When I shared the refund machine and NICU anecdotes with technical communicator friends, they indicated that they often have reservations about aspects of product design but that team members and project management won't accept user behaviour as a show stopper.

My friends had their own anecdotes of no-win situations when they were unable to convince their teams that documentation wouldn't be enough to stop users failing.

As writers and user advocates, we face three 'mountains'.

1. How do you demonstrate the RISK that bad design brings to product and project success?
2. How do you convince your peers and management that YOUR CONTRIBUTION can increase the quality AND viability of a design?
3. How can you contribute user insights EARLY ENOUGH in a product, service, or process life cycle to head off bad design decisions?

### **First, Convince Project Members that Users Determine Design Success**

Convincing peers about unwelcome truths is not easy given the chemistry of teams, relationships, and egos. However, we have to try. We need to challenge colleagues every time we are asked to document products, services, or processes that have design flaws.

The anecdotes in this article form the basis of a collaborative session that I will be leading at the October conference of the Australian Society of Technical Communication (NSW). We plan to develop the presentation into a tool that technical communicators can use to push the "good design" story.<sup>1</sup>

Get a copy of the presentation and build up a visual record of your own anecdotes. Use your library of anecdotes to open discussion about bad design and product failure.

Dramatic but humorous examples of design flaws make the core issue less confrontational and should help teams come to accept that good designs mean successful users AND successful products.

---

<sup>1</sup> Part 1 of the presentation is already available. It deals with the refund machine and the NICU door signs.



## **Second, Promote Your Ability to Identify Bad Designs**

Once teams understand the impact of users on the usefulness of a design, it is up to the technical communicator to show the contribution THEY can make to getting design right.

What enables a technical communicator to test an early design and identify a flaw?

Early in a product development cycle, we identify users, their roles, their objectives and their task requirements. Much of this comes from what users tell us - it's the "I" talking.

"I" information won't arm us with sufficient insights to identify the drivers that give us successful or angry or frustrated users. We need to extend our analysis to what is driving user engagement, that is, the agenda side of each task.

And that's where we can establish our real value for validating designs - we can develop tools to focus on user contexts and agendas.

As in the design of the NICU door or the design of the refund machine, the user's real objective is not to perform the task but to achieve their outcome, in the timeframe that their context allows, and given the limitations that priorities, expectations and assumptions impose on what they see, how they act and how they react.

We have to build our credibility as insightful user analysts progressively - by bringing up agenda and context issues whenever we can. Once you see your engineers using YOUR speak about agenda and context, and their relation to the simplest design issue such as an interface element, you KNOW you are establishing your credentials.

## **Third, Act *Early* in the Project Cycle**

During the 1990s, technical communicators became automatic members of development teams. The driver for writer involvement was originally quality management. As a member of the team, writers were able to track design changes and produce documentation that was fully synchronised with product releases.

Most of today's development teams also involve their writers in the development of interface text and error messages, and invite their writers to give feedback on interface design.

Writers typically operate in this after-the-fact space so design flaws are often irreversible by the time a writer starts reviewing or documenting them. Even user testing often occurs too late in the cycle for significant changes to be made.

### **Apply user validation to the functional specification.**

We have to apply user validation to design as early as possible and the functional specification provides the opportunity to act as user advocate.

Functional specifications may frequently mention users but they generally document atomic system "usage scenarios", not user tasks. User tasks are part of the real-world, not the system.

The user advocate needs

- to present a virtual user in context, with all their conscious and subconscious drivers - their objectives, as well as their likely priorities, assumptions, pushbacks, and automatic behaviours
- to map system "user cases" to usage scenarios that describe how people "do their business", and where and how the system enables or frustrates their tasks

### **Is this really possible? Yes it is. Here's an example.**

When I moved into the dot.com world, I joined a project mid-stream.

- I used the functional specification to identify how much of the user's domain the system was trying to support.
- I talked to people in our own company who would be typical users to understand the tasks, and task contexts and agendas within the performance domain covered by the system.

All the discussions pointed to unexpressed, subliminal task 'drivers' - to protect employees from bad decisions and to protect the company from exposure to tax penalties.

Validated against the user's context and agendas, the system looked like it would be a very useful tool.



I was well into planning the user documentation when the design group published an updated functional specification - rather late because the project team was hard-pressed. Anyway, no one believed that the changes would have an impact. Nothing had changed; they had simply removed some functionality.

What had been removed? Information lookups that enabled the user to make the right decision on behalf of the employee and the company. Why remove information lookups? Because costs demanded shrinking project scope and the information lookups *had no system dependencies*.

I protested the risk of the "design changes", and the project architect and project manager accepted the scenarios I painted. However, I failed to **sell the risk** and development was "too far down the track".

Unfortunately, the users' inability to look up the information they needed before making a decision changed the value of the system from everything to nothing. Instead of providing a low-overhead solution to a business problem, the system would increase the workload of business users *and* increase the risk of incurring substantial tax penalties through uninformed decisions.

Our project was Phase 1 of the system. Phase 1 was delivered with its documentation and user-tested interface but could not be sold as a viable business solution. Phase 2 was cancelled.

## A practical way forward

I frequently look back at the dot-com project and its unnecessary failure. The solution failed not because it wasn't engineered well but because it could not overcome the roadblock of user resistance.

Today, I am more successful than not in demonstrating the risk of poor design, whether it's a website, a system or a document, and that's because

- I have amassed "my" collection of anecdotes so I can convince clients and team members of the impact of poor design.
- I talk agenda and context constantly, but in practical terms that clients and team members understand.
- I get involved early in every project so I have an opportunity to identify the "user risks" before change has too great an impact on project costs.

There is no easy road to building credibility but it comes down to meeting the challenge.

How do you prevent an innocent shop assistant from being strangled by an enraged customer seeking a refund? How do you stop the doors to an intensive care unit being broken down? How do you stop a company failing because customers won't buy its Phase 1 product?

1. Use anecdotes and presentations to convince your teams that users determine the success of design.
2. Constantly demonstrate your ability to inform design from the perspective of task context and agenda.
3. Provide input early in the project cycle, as soon as the first designs surface. For example, at the release of the first functional specification.