How to deal with poorly informed customer choices [closed]

Asked 16 years, 3 months ago Modified 12 years, 7 months ago

Viewed 440 times

6

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Update the question so it can be answered with facts and citations by editing this post.

Closed 5 years ago.

Improve this question

Here's a scenario I'm sure you're all familiar with.

1. You have a fairly "hands off" customer, who really doesn't want to get too involved in the decision making despite your best efforts.
2. An experienced development team spend hours discussing the pros and cons of a particular approach to a problem and come up with an elegant solution which avoids the pitfalls of the more obvious approaches.

3. The customer casually mentions after a quick glance that they want it changed. They have no understanding of all the usability / consistency issues you were trying to avoid in your very carefully thought out approach.
4. Despite explanations, customer isn't interested, they just want it changed.
5. You sigh and do what they ask, knowing full well what will happen next...
6. 3 weeks later, customer says it isn't working well this way, could you change it? You suggest again your original solution, and they seize on it with enthusiasm. They invariably seem to have had a form of selective amnesia and blocked out their role in messing this up in the first place.

I'm sure many of you have gone through this. The thing which gets me is always when we know the time and effort that reasonably bright and able people have put in to really understanding the problem and trying to come up with a good solution. The frustration comes in contrasting this with the knowledge that the customer's choice is made in 3 minutes in a casual glance (or worse, by their managers who often don't even know what the project is really about). The icing on the cake is that it's usually made very late in the day.

I know that the agile methodologies are designed to solve exactly this kind of problem, but it requires a level of customer buy in that certain types of customers (people

spending other peoples money usually) are just not willing to give.

Anyone any clever insight into how you deal with this?

EDIT: Oops - by the way, I'm not talking about any current or recent customer in this. It's purely hypothetical...

methodology

Share

Improve this question

Follow

edited Apr 30, 2012 at 7:03

codingbadger

43.9k ● 13 ● 98 ● 112

asked Sep 10, 2008 at 8:30

reefnet_alex

9,745 ● 5 ● 34 ● 32

6 Answers

Sorted by: Highest score (default)

9

Make your customer pay by the effort you are putting into designing and developing the solution to their problem.

The more you work, the more you get. The customer will have to pay for his mistakes.

Customer will eventually learn to appreciate your experience and insight in the programming field.

Share Improve this answer

answered Sep 10, 2008 at 8:33

Follow

Niyaz
54.8k • 56 • 152 • 183

2

Niyaz is correct, unfortunately getting a customer buy-in is difficult until they have been burned like this once before.

Additionally describe to the customer the scenario above and state how much extra it would cost if you went three or four weeks down the line and had to rewrite it due to a change and then let them use the prototype. It may take a few days to put one together so they can see both options (theirs [the wrong way], and yours [the right way]). Remember they are paying you not only for your ability to program but also your experience and knowledge of the issues which crop up.

Whatever the decision the customer makes, ensure that you get it documented, update your risks register for the project with the risks that the chosen implementation will incurr and speak to the project manager (if its not you) about the mitigation plans for them.

Share Improve this answer

Follow

answered Sep 10, 2008 at 9:23

Mauro
4,511 • 3 • 31 • 59

2

I agree with Niyaz. However at the time the customer suggests the change you should work out what the impact of the change will be, and how likely that impact is

to happen. Then ask whomever is responsible (it's not always that customer) for the deliverable if they approve the change.

Making the impact clear (more cost, lower reliability, longer delivery time etc) is very important to helping the customer to make a decision. It's very important to describe the impacts on the project or their business in a factual way, and assess how likely that impact is to occur. "Maybes" and "i feel" are very ignorable.

After that as long as the right people approve the change and as long as they pay for it.. well you did give them what they wanted :)

Share Improve this answer

Follow

answered Sep 10, 2008 at 9:35

Mark Nold

5,698 ● 7 ● 33 ● 34

1

One thing we have done with some success in the past in these kinds of situations is to hand the issues over to the client.

> "OK, you want to change it - this is what will happen if you do that. These are the issues involved. You have a think about how you'd like it to work and then get back to us".

This approach doesn't tend to yield good solutions (unsurprisingly) but does tend to let the client see that it's

not a "gut feeling", wild stab in the dark kind of question.

And failing that, it usually makes them stop asking you to change it!

Share Improve this answer

Follow

answered Sep 11, 2008 at 12:57

reefnet_alex

9,745 ● 5 ● 34 ● 32

1

Usually a scenario like this is caused by 2 things. The ones that are supposed to give you the requirement specifications are either don't put their hearts into the project because they have no interest in it, or because they really have no idea what they want.

Agile programming is one of the best ways, but there are other ways to do this. Personally I usually use a classic waterfall method, so spiral and agile methods are out of the questions. But this doesn't mean that you **can't** use prototypes.

As a matter of fact, using a prototype would probably be the most helpful tool to use. Think about the iceberg effect. The secret is that People Who Aren't Programmers Do Not Understand This. http://img134.imageshack.us/my.php?image=icebergbelowwater.jpg

> "You know how an iceberg is 90% underwater? Well, most software is like that too -- there's a pretty user interface that takes about 10% of the

> work, and then 90% of the programming work is under the covers...." - Joel Spolsky

Generating the prototype takes time and effort but it is the most effective way to gather requirements. What my project team did was, the UI designer was the one that made the prototypes. If you give the users a prototype (at least a working interface of what the application is going to look and feel like) then you will get lots of criticism which can lead to desires and requirements. It can look like comments on YouTube but it's a start.

Second issue:

> The customer casually mentions after a quick glance that they want it changed. They have no understanding of all the usability / consistency issues you were trying to avoid in your very carefully thought out approach.

Generate another prototype. The key here are results that the users would like to *see* instead of advice that they have to *listen* to.

But if all else fails you can always list the pros and cons of why you implemented the solution, whether or not the particular solution they like is not the one you insisted on. Make that part of the documentation as readable as possible. For example:

Problem:

The park is where all the good looking women jog to stay in shape. Johnny Bravo loves enjoying "mother nature's beauty", so he's lookin to blend in... you know... lookin all buff and do a little jogging while chasing tail.

Alternative Solutions:

1) Put on black suede shoes to look as stylish as you can.

2) Put on a pair of Nike's. Essential shoes for running. Try the latest styles.

Implemented Solution:

Black suede shoes were top choice because... well because hot mommies dig black suede shoes.

Share Improve this answer

Follow

answered Jun 3, 2009 at 8:15

Ludwi

437 ● 4 ● 8

0

Or else, *if they won't pay for the effort*, just avoid putting that much resources into the solution of the problem, and just give them exactly what they've asked for and then think about it after the three weeks have passed.

Somewhat frustrating, yes, but that's the way it'll always be with that kind of customers. At least you won't be losing money.

Share Improve this answer

Follow

answered Sep 10, 2008 at 9:27

Vinko Vrsalovic

340k ● 55 ● 340 ● 373