Home

+ I need a contractor

+ I need a bespoke software solution

About – Stuff about me

Services

+ Contracting services

+ Bespoke software development

Skills

Contact

>How will Lucidity deliver your software?

I recommend an Agile approach to software development. This is an alternative to the traditional Waterfall approach.

>What does the Agile software development involve?

Agile involves dividing the development process into sprints. Each sprint will typically be a couple of weeks long, but this can be tailored to what works best for you.

A set of features will be agreed at the start of the sprint, and a working, tested release will be delivered at the end of the sprint including these features. This release will then be demonstrated to you.

Once you are happy the software is providing a useful set of features, it can be released to your end users.

>Once the software is released, is that it?

No, the sprints can continue after public release, and updates can be released as frequently and for as long as you like.

>Will the updates cause errors in existing features?

No, all code is covered by automated testing. If any updates conflict with existing features, I will be alerted immediately, so bugs cannot make it into the final release.

>Why is this better than the traditional Waterfall model?

The Waterfall model involved specifying, designing, building and testing the entire system in one go. It would then be delivered to the client, sometimes weeks before the deadline. There would then be some final tweaks before releasing it to the client.

There are many problems with this approach.

Often the development process can take years. Sometimes by the time the software is released, the original specification is no longer relevant. The business model may have changed, or the original team that specified the project may have moved on.

Sometimes it only becomes clear at this stage that many of the specified features are no longer required, but time and money has already been spent on development.

Also, it may become clear that the specification for some features is now obsolete, or some required featured were never specified at all. These changes have to be squeezed into the 'final tweaks' stage, which is usually not sufficient.

>What sort of features can you deliver in a two week sprint?

>What does the sprint involves

Just in time design/Use cases/Class diagrams.

Use cases

<file:///C:/Users/Paul.LucidityPc/Downloads/1215437510.pdf>

Domain Modelling

Explain Domain Modelling

<http://martinfowler.com/eaaCatalog/domainModel.html>

<http://scaledagileframework.com/domain-modeling/>

<http://en.wikipedia.org/wiki/Domain_model>

One of my favoured approaches for software development in the Domain Model

I identify 'Entities' the main business elements

Give examples of entities. Inportant to establish a Ubiquitous Language, avoiding synonyms (many words, same meaning) and homonyms (same word, many meanings) for an entity and one word having many meanings(semantic diffusion)

<http://martinfowler.com/bliki/UbiquitousLanguage.html>

<http://martinfowler.com/bliki/SemanticDiffusion.html>

Mention that all this is within a Bounded Context

A bounded context is an area of the business that shares a Ubiqitous Language. The boundaries are drawn where the languge changes.

Bounded Context is sort of equal to a department – when I model, I start with a department as a BC and modify from there.

<http://martinfowler.com/bliki/BoundedContext.html>

http://www.sapiensworks.com/blog/post/2012/04/17/DDD-The-Bounded-Context-Explained.aspx

Customer/Client/Company – recording different info about a client

Next stage is to establish the relationship between them

The establish the attributes for each entity – the data we want to record for our business objects.

Then we need to establish the behaviour of our entities – This is the actions the business objects can perform, and the actions we can perform on them. Often these relate directly to the use cases, but sometimes use cases can comprise of multiple actions, and some actions can be shared between use cases. Examples.

Mention about keeping contexts small- 4x4 and 2x2 rubiks cube

Communicating between contexts:

<http://msdn.microsoft.com/en-us/library/jj591572.aspx>

Difference between DM and DDD.

Describe Repository

Example project:

Vehicle rental