

my take on things

{ by david linsin }

November 23, 2006

Extreme Prototyping

- Reshaping IT Project Delivery Through Extreme Prototyping

OnJava.com is featuring an article about *Extreme Prototyping*, which is described as follows:

Extreme Prototyping is an architectural process for developing applications, especially web applications, in terms of increasingly functional prototypes. At a high level, it breaks down web development into three distinct phases. The first phase is the static prototype, consisting of HTML pages and possibly a logical data model supporting those pages. The second phase is a coding process in your chosen web framework whereby the screens are fully functional using a simulated services layer. The third phase is where the services are implemented.

The article goes on in outlining the complete development process:


- Static Prototype phase
 - Static Prototype
 - Master or background pages
 - CSS, JavaScript
 - Business rules, use-cases
- Extended Static Prototype phase
 - All of the above
 - Logical data model to support the screens
- Dynamic prototype (or Extreme Prototype phase)
 - UI recoded/adjusted for the chosen web framework
 - Working executable code
 - Field validations work

- Navigation of screens will work
- Service signatures solidified
- A complete working UI with no implementation behind
- Service Implementation Phase
 - API document
 - Each service implemented by calling databases or other resources
 - Integration

I must admit that I used to be very skeptical about this kind of approach. I used to think that customers would expect a feature to be fully implemented when it's click able. I came to learn that this is only a misunderstanding. If you point out clearly that you'd like a customer to give early feedback on a *prototype*, there won't be any wrong expectations.

Using this early feedback is very valuable in developing and shaping the use cases and thus the interfaces to the application's business logic. I like this top-down approach, first developing the ui, then the service interfaces. Defining interfaces upfront with only a vague idea of an use case and of how they will be used in ui make you change them a lot, which can be quite painful. Nevertheless another team might already start implementing business logic or integration work, based in preliminary analysis.

This approach might not be useful or reasonable in all cases, but I think it's very helpful when developing small/mid size web applications.

 **Technorati** development, software development, agile, prototyping, extrem prototyping

labels: development

0 comments:

Post a Comment

[<< previous](#)

[| home |](#)

[next >>](#)

com_channels

my_links

- [mail\(dlinsin@gmail.com\)](mailto:dlinsin@gmail.com)
- [jabber\(dlinsin@gmail.com\)](jabber:dlinsin@gmail.com)
- [skype\(dlinsin\)](#)

- [about](#)
- [furryfishApps](#)
- [code](#)
- [app store](#)
- [search](#)
- [jug-karlsruhe](#)
- [archives](#)

recent_postings

- [ShareKit Pimping](#)
- [UI Test Automation with Instruments](#)
- [5 Star Rating](#)
- [Word Buzz Lite](#)
- [Why Word Buzz doesn't support iOS 3.1](#)



[imprint](#)
