HUMAN ENGAGEMENT FRAMEWORK

1.4

First released OCT 2009

Introduction

The framework goes beyond just servicing the interface needs for humans to fully engage with mydigitalstructure based applications, it is about ensuring the framework allows the best people to be working on the support and construction of each layer of the application and it's interface –creating a great overall user experience.

eg:

- The best programmers are working on the core platform services – with deep understanding of what if takes to keep services and associated services healthy
- The best designers are working on the interface layer – with deep understanding of what it takes to keep humans happy / healthy engaged with mydigitalstructure based applications
- etc

Ensuring that not only are applications build within this framework highly engaging today but also into the future, with core framework philosophies of:

- Engaging
- Flexible
- Extendible (including right person at right time)
- Robust

A key concepts used within this framework are:

- **Layering**
  Creating layers of abstraction to allow people to work together with clear lines of responsibility.

- **Viewports**
  In conjunction with story boards, where viewports are used to contextualise information and bring it into focus at the right time for the needs of the human engaging with the application

[More human engagement, equals more subscriptions to the mydigitalstructure platform]

Layers

As used in many construction disciplines, layers are very important to allow teams of people to work together with maximum efficiency.

eg building a house, which brings together the layers of:
- Design (select for existing or bespoke)
- Approval
- Foundation
- House Frame
- Roofing
- etc

Within the mydigitalstructure framework the layers of abstractions are there to also **maximum reuse and flexibility**.

The layers and clear lines of abstraction make it much easier for mydigitalstructure application components to be **mashed into other applications**. eg allowing the contacts system to be mashed into a bespoke application either on the mydigitalstructure framework or off it.

The layers also allow the interface layer to exist in **other edge device frameworks** beyond the browser - still internet based but not just within the confines of the browser that we know, eg iPhone Application, Windows Mobile, Internet TV, in car dashboard. This mashing can also be done by people with the skills and expertise within each of these edge technology frameworks and associated marketplace.

Another key consideration for layers is **the economic advantages** – by being able to engage the right people into different layers. With strong protection as move down through the layers towards the core, means you can have less expensive 3rd parties fore filling some of the roles.

eg the highly skilled electrician, doesn't dig the ditches for the plumbing or do the painting.

With the use of the internet and exchange rates, resources can be used at the edge that are considerably cheaper at fore filling the needs of the humans engagement with the mydigitalstructure applications.

The use of layers also means they can **be executed on different physical devices** and thus the load on each layer can be minimised – most importantly the load on the core application layer and infrastructure layers can reduced and thus the economic value of them increased significantly eg the human engagement/$ can be increased significantly.   The overhead of the infrastructure can be leveraged much more greater, **leading to greater subscription margin**.

Another economic consideration is the **reduction of network traffic** via client side constructs.

More human engagement, equals more subscriptions to the mydigitalstructure platform.
Less load on infrastructure leads to greater subscription margin.

HUMAN ENGAGEMENT FRAMEWORK

Layers Overview

Ref	Layer	Description
	Human	Everyone different and constantly changing. Emotional.
1	Interface "view-controller"	Responsible for engaging the human and servicing their needs. Document and division (DIV) driven – managing viewports and mashing in other frameworks as required (eg Google mapping etc)
2	Webservices / Interface Endpoint "model"	Works on behalf of the core application layer to meet the needs of the interface layer – a "broker". Webservice REST endpoints servicing https GETs and POSTs. Responding with plain text, JSON or XML.
3	Core application layer	Responsible for managing the infrastructure and presenting to the Webservices layer. Core objects/classes.
	Infrastructure	Also the same. Non-emotional. Consistent if kept healthy and stress/load managed

Each layer uses different skills to construct and maintain.

Interface Layer

In terms of human engagement of course this layer is very significant. It's ability to be intuitive and flexible to reduce the difference between an interface that almost works and one that does.

This is not only significant in their level of engagement and emotional binding to it, but also in their willingness to pay for it. The last 5% that makes all the difference, in someone that wants to be engaged.

eg

an interface that meets 95% of their needs, some people will struggle to pay $50 for as can't see the value - fill in the last 5% and value becomes clear and present, $5000 is no longer not out of the question.

Layers within the Interface Layer by Responsibility / Technology

Ref	Layer	Description
1	User Interface Layout	XHTML
2	User Interface Presentation	Look CSS
3	User Interface Construction	jQuery
4	User Interaction	Javascript
5	Information Management	Search and Save, Persistence AJAX/onDemand

Layers within the Interface Layer by Typical Technology

Ref	Layer	Description
1	XHTML	Creates the document and elements, including divisions.
2	CSS	Manages the "look and feel" eg colours, images etc
3	Javascript/jQuery	Includes jQuery plugins and jQueryUI – works with CSS to create the more active components of the interface.
4	AJAX / onDemand	Communication with the webservices/end point layer.

HUMAN ENGAGEMENT FRAMEWORK

Viewports

Within the interface layer, viewports and their use to tell as story is critical to creating an interface that engages humans.

The use of cascading viewports is as import as the use of layers.

It allows a human to move through their information, bring into focus the right information and associated actions at the right time to forefill their needs.

Using a single document (XHTML) – information can be brought in and sent out as they view and work with it.

From a human perspective they are moving the red square around the storyboard (which is fixed)

From a technology perspective the red square (XHTML Document/Device) is fixed and the story is sliding underneath it – in the form of transforming divisions or in film/movie terms the frames [not HTML frames].
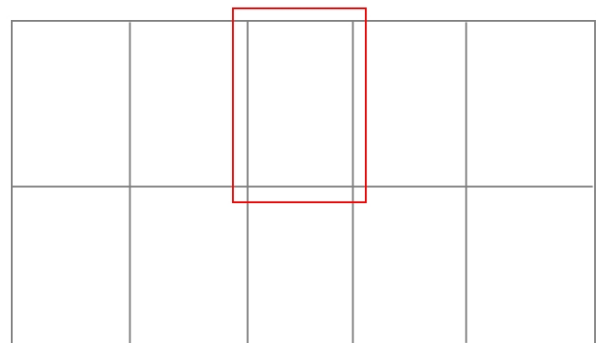
This is very similar to how gaming applications are constructed and by nature games are incredibly intuitive and built around engaging humans and telling a story.   Most people love a good story, especially one about them.

In the case of the mydigitalstructure based applications, the construction would use the concept of cascading viewports, starting with high-level context setting viewports and then cascading down to more needs based viewports.

eg

- Select the Contacts viewport with then enables viewports for say:
    o Person Search
    o Person View
    o Person Edit
    o Business Search etc

The actual construct of the viewports is held with in the appropriate layer.   Given that viewports are conceptual, where the layers and associated technologies and people do the execution.

Real Example Of Framework In Use

The position and existence of any one element is irrelevant to the majority of the layer logic.

Remembering the understanding of all layers is much more difficult than understanding of just one or layers which is the may the majority of people will ever engage with this framework.

Fundamentally at a technology level we already do everything that is discussed document. It is about creating a framework to consolidate this and create a consistent path forward.

Master Viewport Control Division (Header)
id=divInterfaceMasterViewportControl
class=interfaceMasterViewportControl

Viewport Control
Division
id=divInterfaceViewportControl
class=interfaceViewportControl

Main Division
id=divInterfaceMain
class=interfaceMain

Footer Division
id=divInterfaceFooter
class=interfaceFooter