AN ALTERNATIVE ALT.NET

Jimmy Nilsson

factor10
About Jimmy Nilsson

Primarily a developer/architect

Blog: jimmynilsson.com/blog/
Twitter: twitter.com/jimmynilsson

Author of "Applying Domain-Driven Design and Patterns" and ".NET Enterprise Design"

Co-founder and CEO of factor10
The MSDN way

One Big Vendor

"Buy in to the whole thing, and nothing else..."

1990
ALT.NET

OS Java Agile Ruby Community XP Rails

Revolution

Pretty large generate frameworks

2005
An alternative
ALT. WET

Evolution

Tiny custom frameworks

Examples

2010
1990

Wizards

Datasets

Drag till you drop

Webforms

~ 11 layers
Custom large generic frameworks

Crash

2005

DDD TDD Craftsmanship

How

Elegance Quality

O/RM MVC

What

Great code

Why

$ ~ 5 layers

~ 5 layers
BDD
REST
Real DDD

2010

DSL
No, Not Only
Tiny!
Great code

3 layers
BDD helps with a red thread

• The step definitions points out which unit test to write now
• That unit test points out what real code to write now
• When done, check that BDD test is green too
What is DDD?

Domain-Driven Design is an approach to the development of complex software in which we:

• Focus on the **core domain**

• Explore *models* in a creative collaboration of domain practitioners and software practitioners

• Speak a *ubiquitous language* within an explicitly **bounded context**
Value Objects

Customer -> Address -> Rental -> Price -> Discount

FilmTitle
Like this?

```csharp
var stinasÄgg = 7 * 12;
var pellesÄgg = 3 * 20;
var antaletlÄggkorgen = stinasÄgg + pellesÄgg;
Assert.AreEqual(144, antaletlÄggkorgen);
```
Or more like this?

```javascript
var stinasÄgg = Antal.Dussin(7);
var pellesÄgg = Antal.Tjog(3);
var antaletlÄggkorgen = stinasÄgg + pellesÄgg;
Assert.AreEqual(144, antaletlÄggkorgen.IStyck);
```
Examples
Domain experts can read and write!

- Workflow
- Filespec
- Checklist

Tiny languages

Domain experts can read.

The documentation is the program.
Workflow example

Order registered

Check

Contact customer

Prepare product family A
And so on...

Prepare product family B
And so on...

Prepare product family C
And so on...

Order registered
Check
Contact customer
Prepare product family A
And so on...
Prepare product family B
And so on...
Prepare product family C
And so on...
Filespec example

Filetype ABC
Matches files that contain the text nd ABC
The file uses tab as field separator and the first row should be skipped

Fields in the file:
Administrator is text
Department is text
PlanId is text
ExportDate is date of format yyyy-MM-dd with default 9999-12-31
CustomerGroup is text
SSNPOH is text
Product is text
Amount is decimal with typeconverter DecimalConverter
< cut for demo... >
CustomerNumber is text
InsuranceCompany is text
FirstName is text
FamilyName is text
< cut for demo... >

The identity of a row is decided by the fields SSNPOH, Product, PlanId

Mappings:
FirstName is mapped to FirstName on Person
FamilyName is mapped to FamilyName on Person
SSNPOH is mapped if it has value to SSN on Person
CustomerNumber is mapped if it has value to SSN on Person
< cut for demo... >

Administrator is mapped to FromAdministrator on Insurance
Department is mapped to Department on Insurance
PlanId is mapped to PlanId on Insurance
ExportDate is mapped to ExportDate on Insurance
CustomerGroup is mapped to CustomerGroup on Insurance
SSNPOH is mapped to SSNPOH on Insurance
SSNPOH is mapped if it has value to SSNinsured on Insurance
Checklist example

Create checklist

Name: Distribution execution 2010

Header: 2010-04-20 Preparations
  Do manually: Check that backup has been taken
  Do manually: Contact Helpdesk and ask them to execute "Rebuild Index" and shut down antivirus protection McAfee
  Run command: CLEARXML Mission RNFIL

Header: 2010-04-21 More preparations
  Do manually: XYZ

Header: 2010-04-23 More preparations again
  Run command: AGRBOKF2 Mission XSEND with parameters Poll=AgrSndTrn

Header: 2010-04-24 And so on...
UI matters. A lot.
Great for discussions from day 1.
Often has the worst and most code.
Need a look and feel refresh every now & then.

Declarative UI

NoJS
Semantic templating
Metadata driven

Checklist again

Origin of hypermedia in DM
NoJS example

Title
{:info.title:}

ISBN
{:info.isbn:}

Description
{:info.description:}
Checklist example again

Distribution execution 2010

2010-04-20 Preparations

- Check that backup has been taken
  
  Start: ✓ Name: Jimmy Date: 2011-10-10 00:18:16 Comments: 
  Done: ✓ Name: Jimmy Date: 2011-10-10 00:18:15 Comments: They had already prepared it...

  Save  Refresh

- Contact Helpdesk and ask them to execute "Rebuild Index" and shut down antivirus protection McAfee
  
  Start: □ Name:  Date:  Comments: 
  Done: □ Name:  Date:  Comments: 

  Save  Refresh

Kommando

  CLEARXML Parameters:

  Ordered: x Started:  Executed:  Problem: 
  Done: □ Name:  Date:  Comments: 

  Save  Refresh

2010-04-21 More preparations
Not all problems are relational... :-)

But strongly, Oracle/SQL Server is almost mandatory

Custom storage

Avoids "global db"
- Code disappears
- True aggregates
- Perf increases
- Operations says O(1)
- Good for big graphs
- Protobuf
- Migrations are relational
Custom storage example

```csharp
var projects = new Projects("Data Source=.;Initial Catalog=TimeReg; Integrated Security=True");
var project = projects.GetById(new Guid("A29E1A0B-2B7E-4F45-B8E4-0B304366C690"));
Console.WriteLine(project);
```
public Project GetById(Guid id)
{
    return storageMetadata.GetById(id);
}

public IEnumerable<ProjectForListing> ThatHasCustomerWithName(string customerName)
{
    return storageMetadata.GetListByCriteria<ProjectForListing>
        ("Customer_Name", customerName);
}

projects.GetCreateScript();
Summary

ALT.NET was important
But the potential is way bigger
Let’s unleash it!