QCon London, March 2011

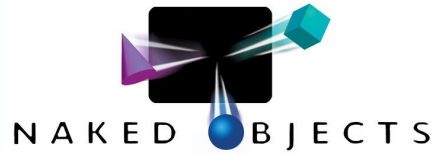
Case Study:
Large-scale pure OO at the Irish Government

Richard Pawson – rpawson@nakedobjects.net
Naked Objects Group
www.nakedobjects.net

What makes this an interesting story:

- Repeated successful delivery of very large scale projects *in the public sector*
- Extraordinary business domain complexity
- Possibly the purest example of OO design for a large-scale transactional business application, worldwide
- One of the largest-scale applications of agile-development within the public sector, worldwide
- A rich user interface, to a core transactional business system
- First, and still the largest, deployment of the naked objects pattern

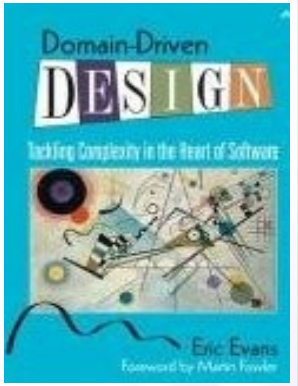
The Department of Social ~~Welfare~~, ~~Community,~~ ~~and~~ ~~Family~~ ~~Affairs~~ Protection (www.welfare.ie)



- As part of a huge Service Delivery Modernisation (SDM) programme, initiated 10 years ago, the Department recognised the need to replace its aging mainframe-based benefit administration systems
-
- The primary requirement: ‘agility’
  - Technical agility
  - Strategic business agility
  - Operational business agility
  -
- The chosen solution:
  - An agile architecture
  - Agile development methodology
  - A Microsoft-based platform

Agile design principle 1: Domain-driven design

NAKED OBJECTS



- Aim: Define an ubiquitous language (the 'BOM') - a common language between business stakeholders and software developers
- Rationale: It represents the (slowly-changing) essence of the business, rather than the (rapidly-changing) current practices
- New requirements are expressed in BOM terms
  - Violation: Process-driven design
  - Violation: Screen-driven design
- Evidence: Business stakeholders are involved directly in the BOM design
- Evidence: Almost 100% the BOM can be understood by the business
- Constraint: BOM should be agnostic of the technical platform
  - but *is* fundamentally represented in code

Partial extract of .NET classes from the Common BOM

Customer Maintenance
Abstract Customer
- Customer
- Customer Awaiting PPSN
Alternate ID
Basis Match
Customer Details Form
Customer Email Address
Life Event
- Birth
- Death
- Marriage
Abstract Name
- Name
- Usage Name
Previous Claim
Relationship
- Primary Relationship
- Reciprocal Relationship
Telephone Number

Address Maintenance
Abstract Address
- CRS Address
  - Household Address
  - Correspondence Address
  - Previous Address
- Generic Foreign Address
- Irish Address
- UK Address
- USA Address

Scheme administration
Benefit
- Component
  - Exception Payment Generator
  - Split Payment Generator For Customer
  - Split Payment Generator For Other Party
  - Overlap Correction
- Scheme
Entitlement
Entitlement Period
Hypothetical Entitlement
Means Element
- Capital Means
- Weekly Income Means
Means Assessment
Rate Table

Customer Communication
Address
- Email Address
- Postal Address
Communication
- Form Communication
  - Customer Communication
  - Scheme Communication
- Letter Communication
Communication Template
- Form Communication Template
- Letter Communication Template
Communication Template Translation
- Form Communication Template Translation
- Letter Communication Template Translation
Postal Address Line

Paying Benefits
Book
Book Renewal Cycle
Overlap
Overpayment
Payment
- Cheque Payment
- EFT Payment
- EIT Payment
- PPO Payment Voucher
Payment Statement

Payment Method Maintenance
Agent
- Individual Agent
- Institutional Agent
Bank
- Credit Union
- Foreign Bank Details
- Irish Bank
- UK Bank
Payment Method
- Cheque Payment Method
- EFT Payment Method
- Post Office Payment Method
  - PPO Payment Method
  - EIT Payment Method
Post Office

Organisational Maintenance
Actor
- Officer
- Org Unit
Actor Email Address
CRS User
Grade
Local Office

Work Management
Abstract Case Content
- Case Note
- Imported File
- Imported Mail
- Scanned Document
Bring Back Details
Customer Case
- Maintenance Case
- Scheme Case

Quality Control
Abstract Recorded Action
- Decision
- Recorded Action
  - Customer Case _ Recorded Action
Recorded Action Type

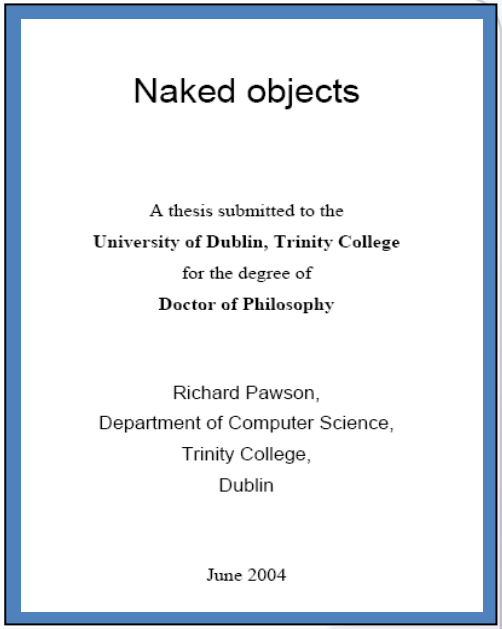
Agile design principle 2: Behaviourally-rich objects

NAKED OBJECTS

- Aim: 100% of business logic encapsulated as methods on (persistent) domain entities
  - Violation: Introduction of 'view', 'controller', or 'process' objects
  - Violation: Decomposition of core business functionality into services
- Rationale:
  - Strategic business agility
  - Very high-reuse between different applications (from polymorphism)
- Rarely, business logic may be encapsulated on non-persistent 'transient' objects
  - Example: (Contribution History) TakeOnSheet
- The BOM makes extensive use of inheritance and polymorphism
  - Violation: dictating BOM design by the convenience of database representation
- Services play a distinct, and secondary, role to domain objects
  - As a bridge to other domains, such as technical domains or external applications
  - To provide logic that *cannot* be associated with an object, such as the start points featured on the user's desktop
  - To provide logic needed by multiple objects, that have no common superclass

Agile design principle 3:
The naked objects pattern

- All user interfaces (UI) for internal use are a direct reflection of the BOM
- Rationale:
  - eliminates the need to develop/maintain the 'view' and 'controller' layers
  - easier to adhere to the principles of DDD
  - facilitates a very pure style of OO
  - facilitates agile development practices
  - creates a more empowering style of UI
- Violation: Introduction of any custom UI code (for an internal facing application)
- Violation: Code Generation



Naked objects

A thesis submitted to the
University of Dublin, Trinity College
for the degree of
Doctor of Philosophy

Richard Pawson,
Department of Computer Science,
Trinity College,
Dublin

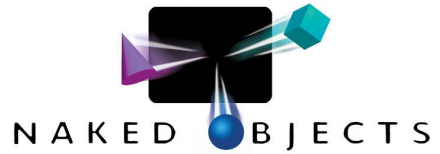
June 2004

Agile design principle 4:
No such thing as an ‘application’

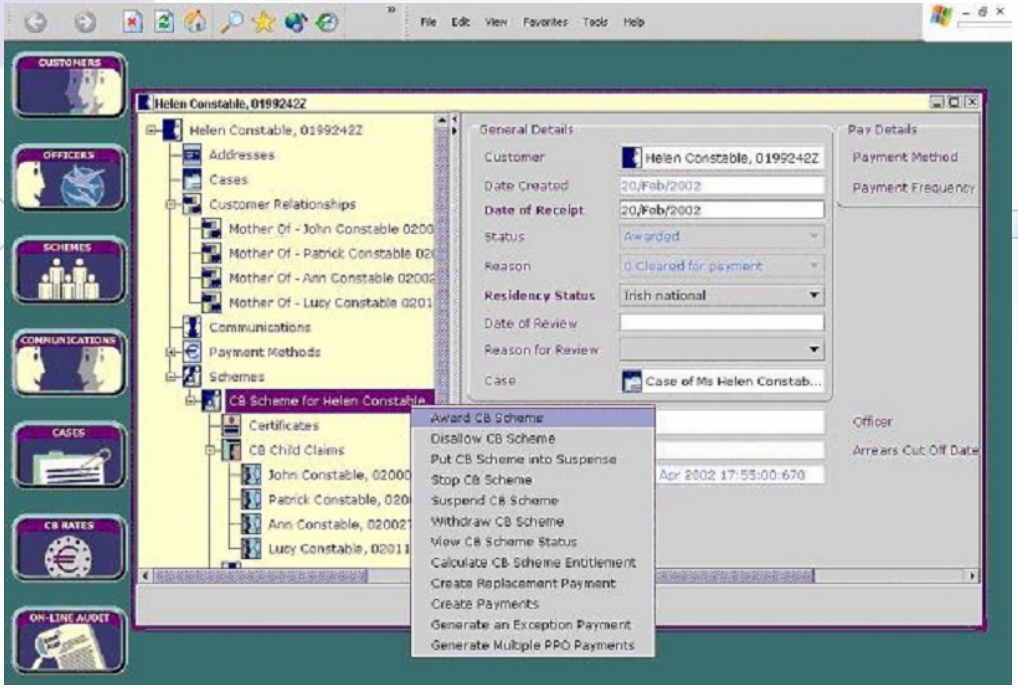
NAKED OBJECTS

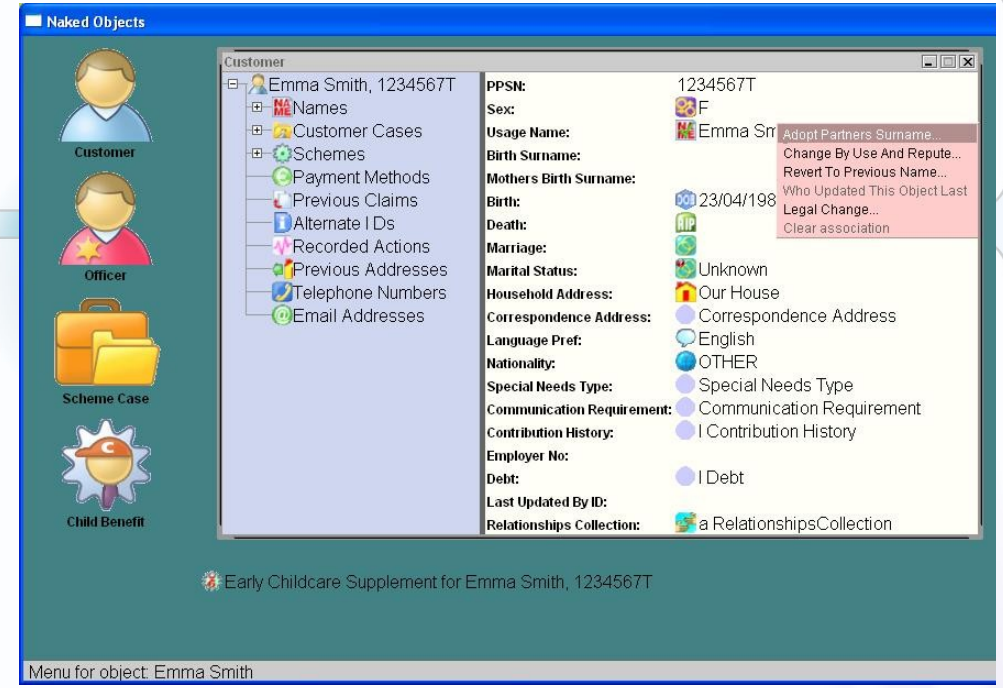
- All Officers are logging on to a single system, with role-based permissions for accessing object types, properties and actions
  - Violation: a separate log-on from the generic SDM log-on
- The separation of Common and Specific BOMs is for project management and testing purposes
  - The specific BOMs do *not* constitute ‘applications’
- Requirements for new specific objects, should, wherever possible, be implemented in abstract form, in the Common BOM

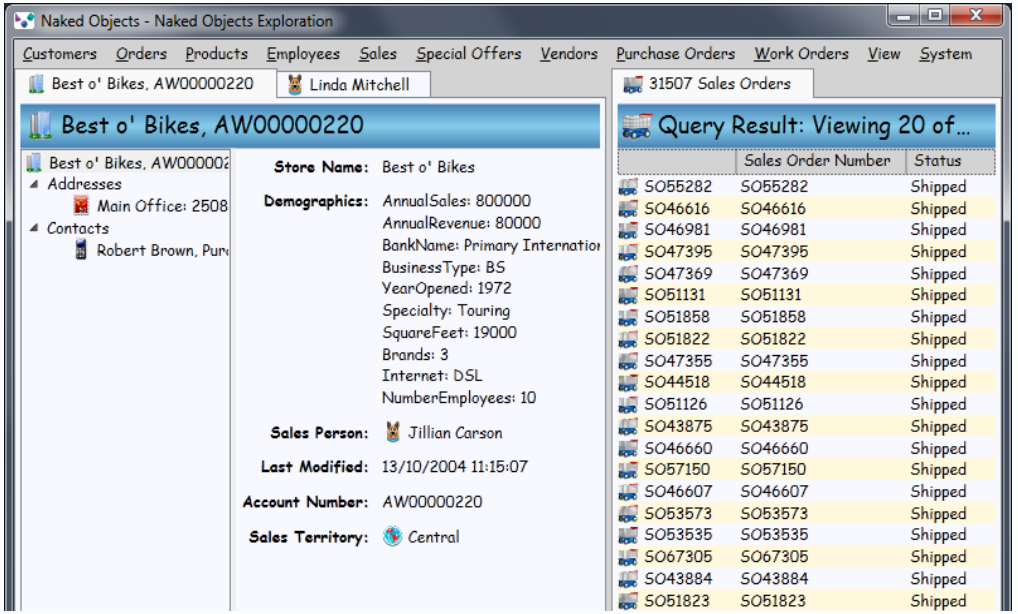
The technical platform

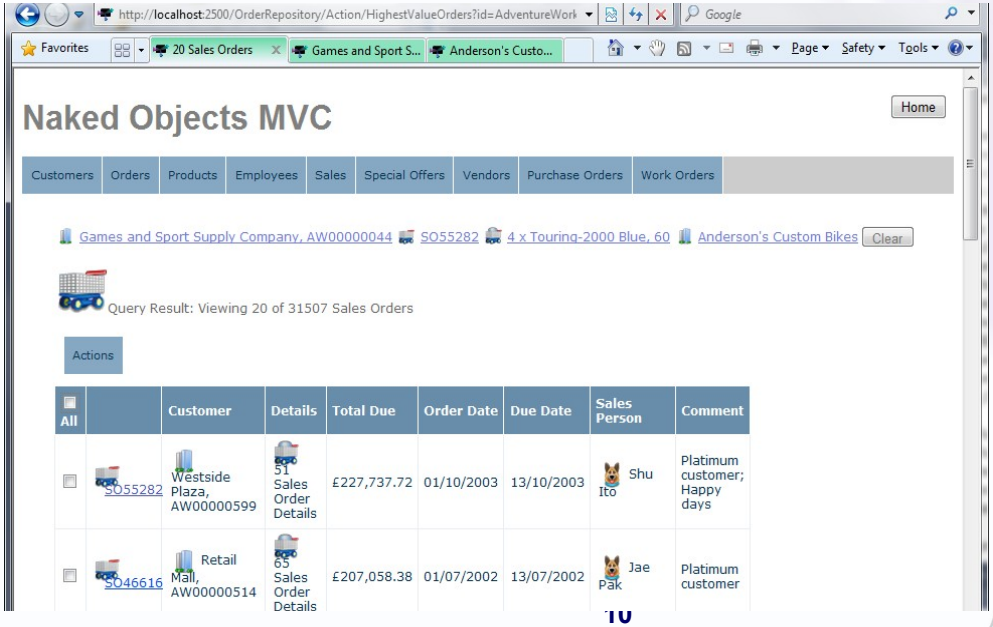


- Domain objects written as POCO .NET classes
- A framework to implement the Naked Objects Pattern
  - Phase 1: a bespoke framework written by Fujitsu
  - Phase 2: replaced by a port of an early version of (Java) Naked Objects (now Apache Isis incubator.apache.org/isis/)
  - Phase 4: will move to Naked Objects WPF & Naked Objects MVC
- Bespoke automated test framework (built on top of NUnit)
  - eXecutable Application Tests (XAT) and System Tests (XST)
- Persistence on RDBMS
  - Phase 1: Data Access Objects
  - Phase 2 & 3: Custom ORM
  - Phase 4: Entity Framework
- BizTalk for Publish & Subscribe
- Open source components: Spring.NET, CruiseControl.NET, Apache FOP …

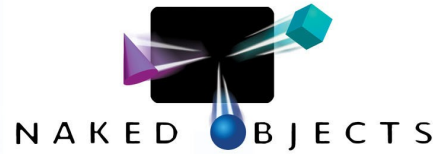

File Edit View Favorites Tools Help
CUSTOMERS
OFFICERS
SCHEMES
COMMUNICATIONS
CASES
CB RATES
ON-LINE AUDIT
Helen Constable, 01992422
Addresses
Cases
Customer Relationships
Communications
Payment Methods
Schemes
CB Scheme for Helen Constable
Certificates
CB Child Claims
General Details
Customer
Helen Constable, 01992422
Date Created
20/Feb/2002
Date of Receipt
20/Feb/2002
Status
Awarded
Reason
0 Cleared for payment
Residency Status
Irish national
Date of Review
Reason for Review
Case
Case of Ms Helen Constab...
Pay Details
Payment Method
Payment Frequency
Officer
Arrears Cut Off Date
Award CB Scheme
Disallow CB Scheme
Put CB Scheme into Suspense
Stop CB Scheme
Suspend CB Scheme
Withdraw CB Scheme
View CB Scheme Status
Calculate CB Scheme Entitlement
Create Replacement Payment
Create Payments
Generate an Exception Payment
Generate Multiple PPO Payments


Naked Objects
Customer
Officer
Scheme Case
Child Benefit
Customer
Emma Smith, 1234567T
Names
Customer Cases
Schemes
Payment Methods
Previous Claims
Alternate I Ds
Recorded Actions
Previous Addresses
Telephone Numbers
Email Addresses
PPSN: 1234567T
Sex: F
Usage Name: Emma Sm
Birth Surname:
Mothers Birth Surname:
Birth: 23/04/198
Death:
Marriage:
Marital Status: Unknown
Household Address: Our House
Correspondence Address: Correspondence Address
Language Pref: English
Nationality: OTHER
Special Needs Type: Special Needs Type
Communication Requirement: Communication Requirement
Contribution History: I Contribution History
Employer No:
Debt: I Debt
Last Updated By ID:
Relationships Collection: a RelationshipsCollection
Adopt Partners Surname...
Change By Use And Repute...
Revert To Previous Name...
Who Updated This Object Last
Legal Change...
Clear association
Early Childcare Supplement for Emma Smith, 1234567T
Menu for object: Emma Smith


Naked Objects - Naked Objects Exploration
Customers Orders Products Employees Sales Special Offers Vendors Purchase Orders Work Orders View System
Best o' Bikes, AW00000220
Linda Mitchell
31507 Sales Orders
Best o' Bikes, AW00000220
Addresses
Main Office: 2508
Contacts
Store Name: Best o' Bikes
Demographics: AnnualSales: 800000
AnnualRevenue: 80000
BankName: Primary Internatio
BusinessType: BS
YearOpened: 1972
Specialty: Touring
SquareFeet: 19000
Brands: 3
Internet: DSL
NumberEmployees: 10
Sales Person: Jillian Carson
Last Modified: 13/10/2004 11:15:07
Account Number: AW00000220
Sales Territory: Central
Query Result: Viewing 20 of...
Sales Order Number
Status
SO55282 SO55282 Shipped
SO46616 SO46616 Shipped
SO46981 SO46981 Shipped
SO47395 SO47395 Shipped
SO47369 SO47369 Shipped
SO51131 SO51131 Shipped
SO51858 SO51858 Shipped
SO51822 SO51822 Shipped
SO47355 SO47355 Shipped
SO44518 SO44518 Shipped
SO51126 SO51126 Shipped
SO43875 SO43875 Shipped
SO46660 SO46660 Shipped
SO57150 SO57150 Shipped
SO46607 SO46607 Shipped
SO53573 SO53573 Shipped
SO53535 SO53535 Shipped
SO67305 SO67305 Shipped
SO43884 SO43884 Shipped
SO51823 SO51823 Shipped

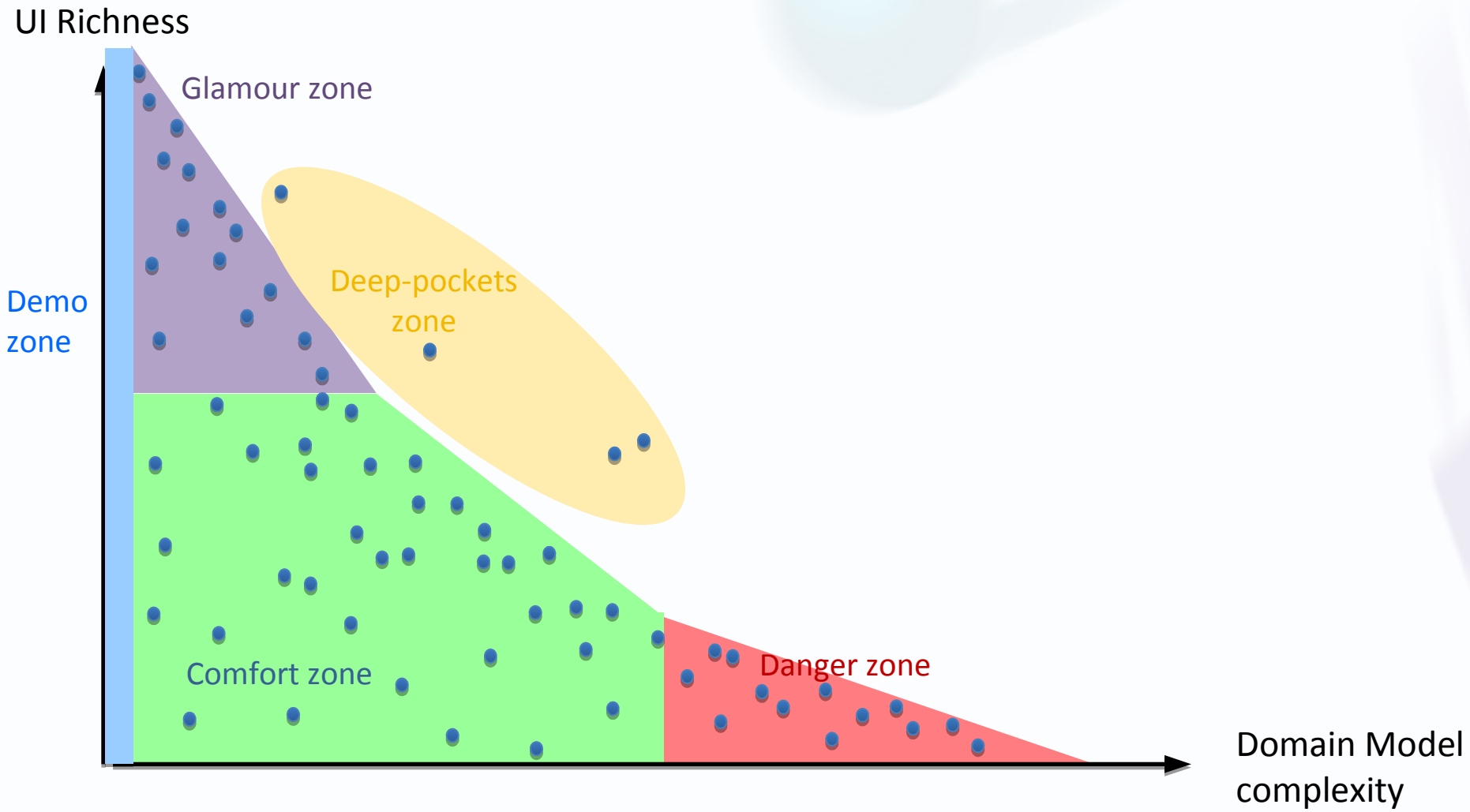

http://localhost:2500/OrderRepository/Action/HighestValueOrders?id=AdventureWork
Favorites
20 Sales Orders
Games and Sport S...
Anderson's Custo...
Naked Objects MVC
Home
Customers Orders Products Employees Sales Special Offers Vendors Purchase Orders Work Orders
Games and Sport Supply Company, AW00000044 SO55282 4 x Touring-2000 Blue, 60 Anderson's Custom Bikes Clear
Query Result: Viewing 20 of 31507 Sales Orders
Actions
All
Customer
Details
Total Due
Order Date
Due Date
Sales Person
Comment
SO55282 Westside Plaza, AW00000599 51 Sales Order Details £227,737.72 01/10/2003 13/10/2003 Shu Ito Platimum customer; Happy days
SO46616 Retail Mall, AW00000514 65 Sales Order Details £207,058.38 01/07/2002 13/07/2002 Jae Pak Platimum customer

Extraordinary hurdles



- Procurement constraints
- Technical constraints
  - bridging SQLServer and Oracle RDB
- Immaturity of the technologies (at the time)
  - Naked Objects framework
  - Visual Studio (in fact, the whole .NET dev. Environment!)
  - Lack of a viable ORM
- Business complexity

Two (out of several) dimensions of systems development projects



Biggest surprises (positive)

NAKED OBJECTS

- That the Department went ahead with the idea at all
- That all (but one) of the projects has been a complete success
- Performance (until very recently)
- Business willing to engage in object modelling
  - and willingness to lose the process focus
- ‘Extreme re-use’
- Extensibility in unforeseen ways
- User reaction
  - though not 100%

Metaphor: The Incredible Machine

NAKED OBJECTS

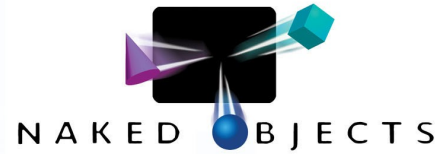
User reaction to the ‘problem-solving’ user interface has been positive

NAKED OBJECTS

“The new system permits me to better deal with the needs of individual customers”

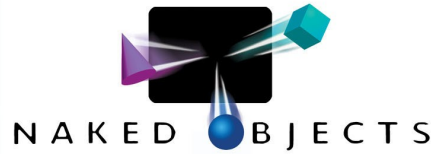
“The new system contributes positively to my sense of job satisfaction”

Biggest surprises (negative)



- How poor the Microsoft tools and frameworks were in 2004
  - though much improved since
- Exposing the range of developer skills and motivation
- Poor fit with traditional (development) job roles
  - Testers, BAs, Batch, DBAs
- The single project failure

What could have been done better?



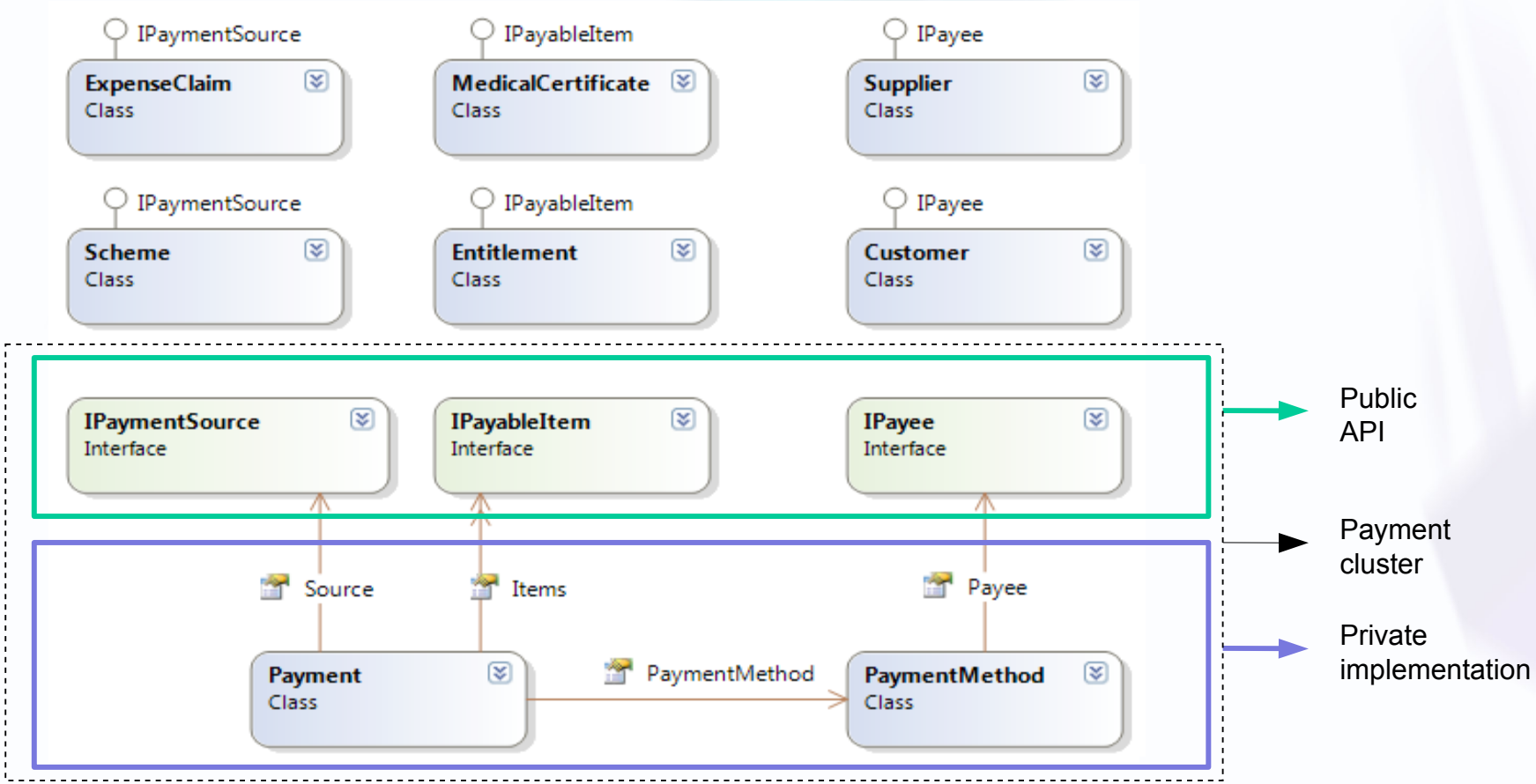
- Too much coupling in the BOM
- On-going investment in the infrastructure
- Batch processing
- Insufficiently aggressive about the 80:20 rule
- Not adopting progressive roll-out
- A more imaginative procurement process
-

Where next?

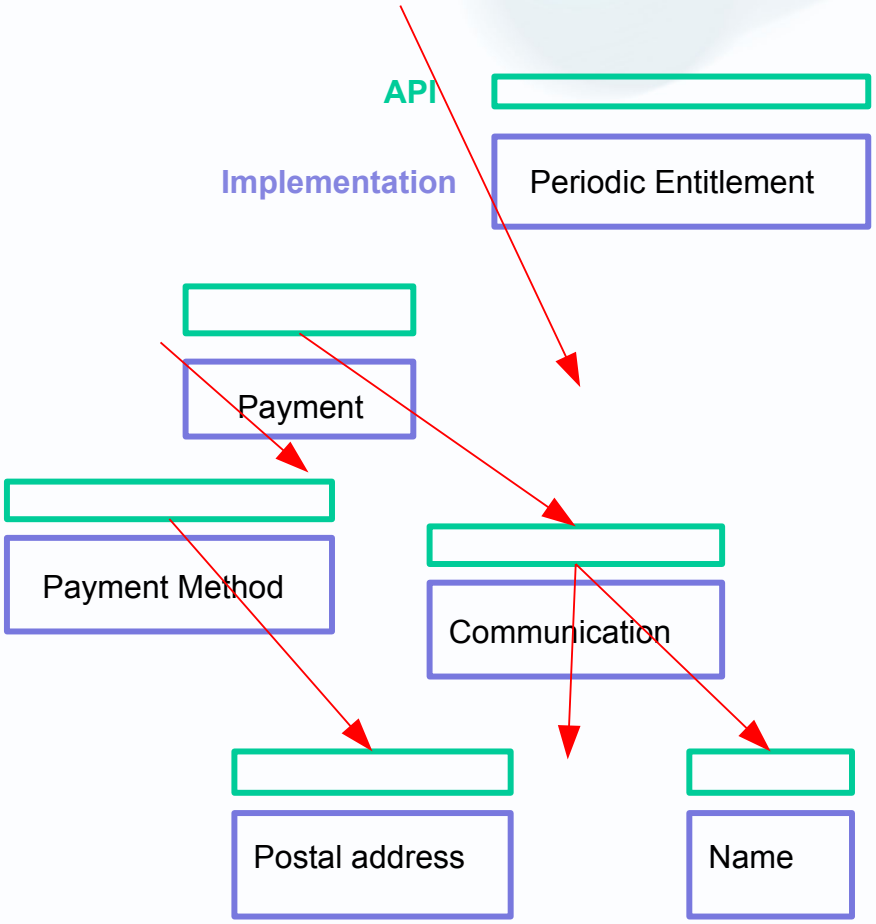
NAKED OBJECTS

- Continued aggressive development of new functionality due to:
  - Heavy business pressures from the economy
  - Massive increase in role & responsibility of DSP
  - Pressure to port the remaining mainframe systems
- Migration to the second generation platform
  - New Technical Architecture
    - Naked Objects MVC (based on ASP.NET MVC)
    - Entity Framework
    - WIF security
  - Porting and refactoring of the BOM using the ‘cluster’ pattern

The cluster model
Example: Payments cluster (partial view)



A cluster may depend on other clusters *where this reflects a clear business dependency*



A cluster is defined by an API project

NAKED OBJECTS

1. API is defined by .NET Interfaces, of three broad kinds:
   - A 'Role interface' is intended to be implemented by objects in other clusters e.g. `IPayableItem, ICorrespondenceHolder`
   - A 'Result interface' provides a restricted view of a class defined inside the cluster e.g. `ICustomer, ICountry`
   - A 'Service interface' provides a programmatic view of a service e.g. `ICustomerRepository`
2. API may also contain static (shared) functionality e.g. Enums, Modules
3. API must *not* contain instantiable classes
   - Because all knowledge of persistence resides inside the cluster
4. API is pure POCO
   - no references to Naked Objects Framework (though App Lib is allowed for annotations)
5. API project may reference other API projects, where this reflects a business reality
   - `PaymentAPI.IPayableParty` inherits `CommunicationAPI.ICommunicableParty`

The ‘cluster model’: How
A cluster is defined by an API project

NAKED OBJECTS

1. API is defined by .NET Interfaces, of three broad kinds:
   - A ‘Role interface’ is intended to be implemented by objects in other clusters e.g.
     `IPayableItem, ICorrespondenceHolder`
   - A ‘Result interface’ provides a restricted view of a class defined inside the cluster e.g.
     `ICustomer, ICountry`
   - A ‘Service interface’ provides a programmatic view of a service e.g.
     `ICustomerRepository`
2. API may also contain static (shared) functionality e.g. Enums, Modules
3. API must *not* contain instantiable classes
   - Because all knowledge of persistence resides inside the cluster
4. API is pure POCO
   - no references to Naked Objects Framework (though App Lib is allowed for annotations)
5. API project may reference other API projects, where this reflects a business reality
   - `PaymentAPI.IPayableParty` inherits `CommunicationAPI.ICommunicableParty`

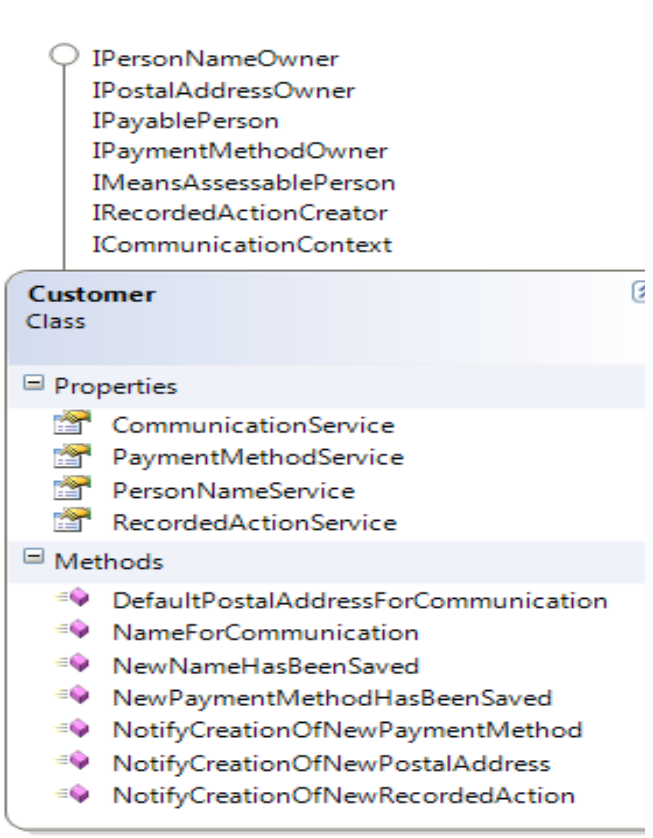
New application classes may delegate most of their functionality to these injected services

The list of implemented role interfaces.

(Provides a readable summary of the object's intent.)

Properties (hidden from user) into which required services are injected.

Lightweight methods required by the implementation of various interfaces.



Some additional subtleties of the Cluster Model

NAKED OBJECTS

- Yes, the cluster model does bear some resemblance to an SOA, but
  - The API is specified as object types (interfaces) which define the business functionality that they provide, not just their structure
- All relationships across clusters must be defined as 'interface associations' (a.k.a. 'any' or 'polymorphic' associations)
  - DBAs won't like this
  - Microsoft Entity Framework does not support this yet!
  - Naked Objects supplies its own implementation
  - One huge upside is that clusters can be moved between databases with zero impact on the code
- Cluster functionality my be invoked by means of:
  - Dependency injection of a cluster service e.g. IPaymentService
  - Aspects
  - Naked Objects: 'Contributed Actions'