XebiCon 2013

A DSL for generating mobile applications

by Marc Grol

http://bit.ly/xc_a

Speeding up app development

Using a model-driven approach

Content

- Explain approach
- Show example

Introduction

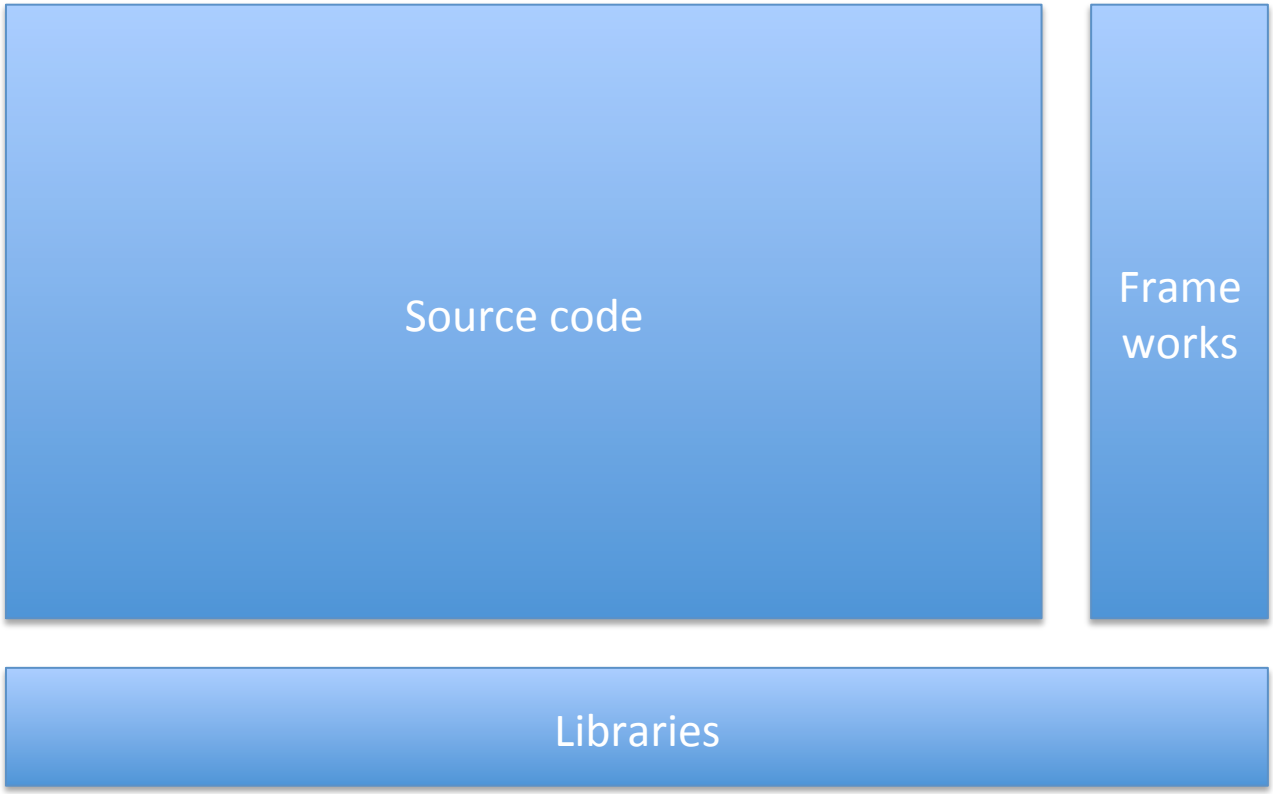
App development is expensive:

- Multiple platforms (IOS, Android, etc)

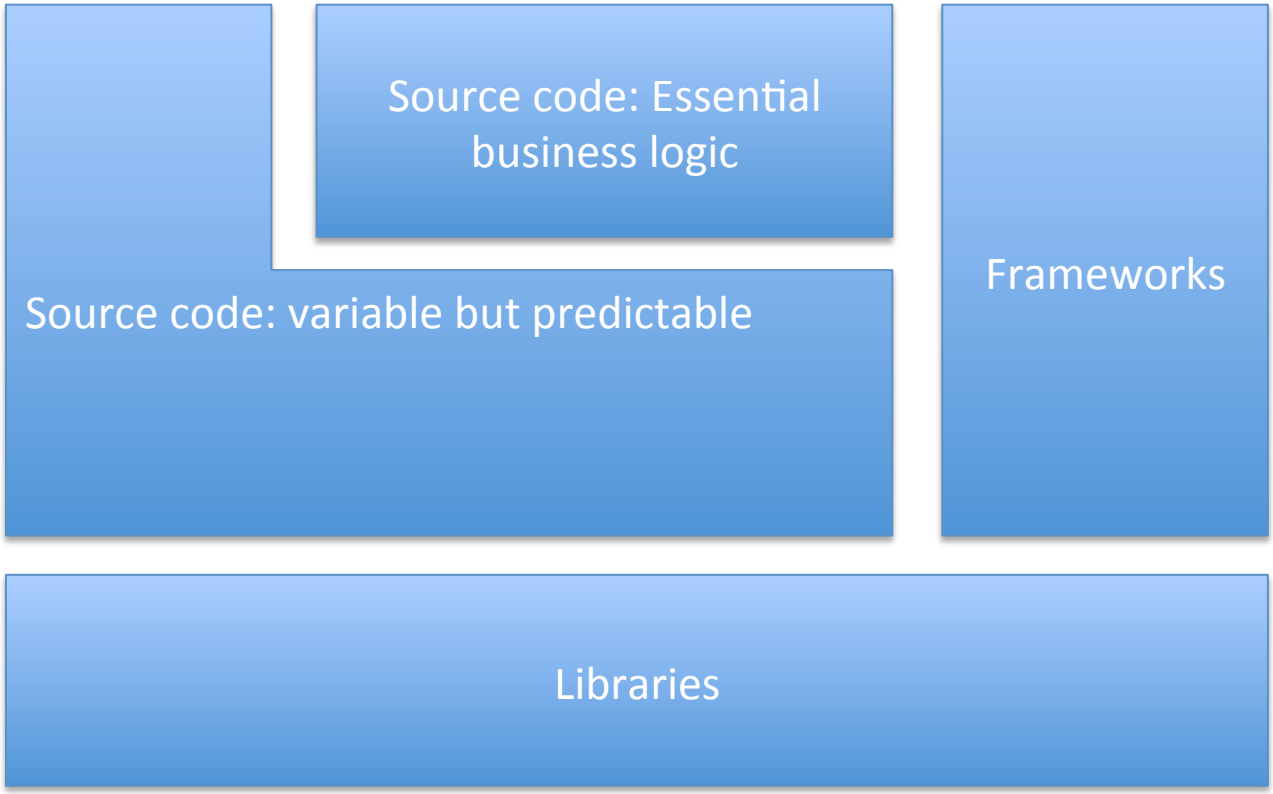
Cost reduction:

- Hybrid/web-based approach
- Alternative “model-driven” approach:
  - Do NOT compromize on usability

App: First attempt

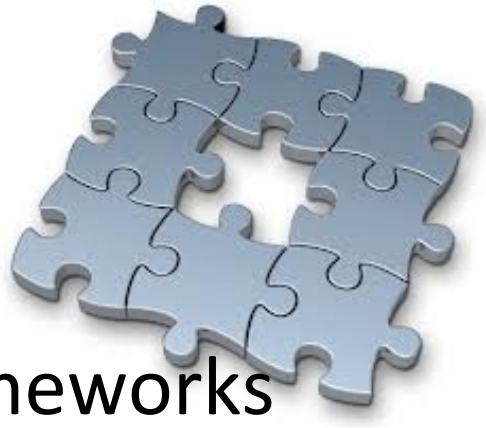


App: After some practice



Experience

- Apps are about user-interface
- Data model looks like “screens”
- Business logic lives in backends



- Using common libraries and frameworks
- Still very repetitive “recipe”

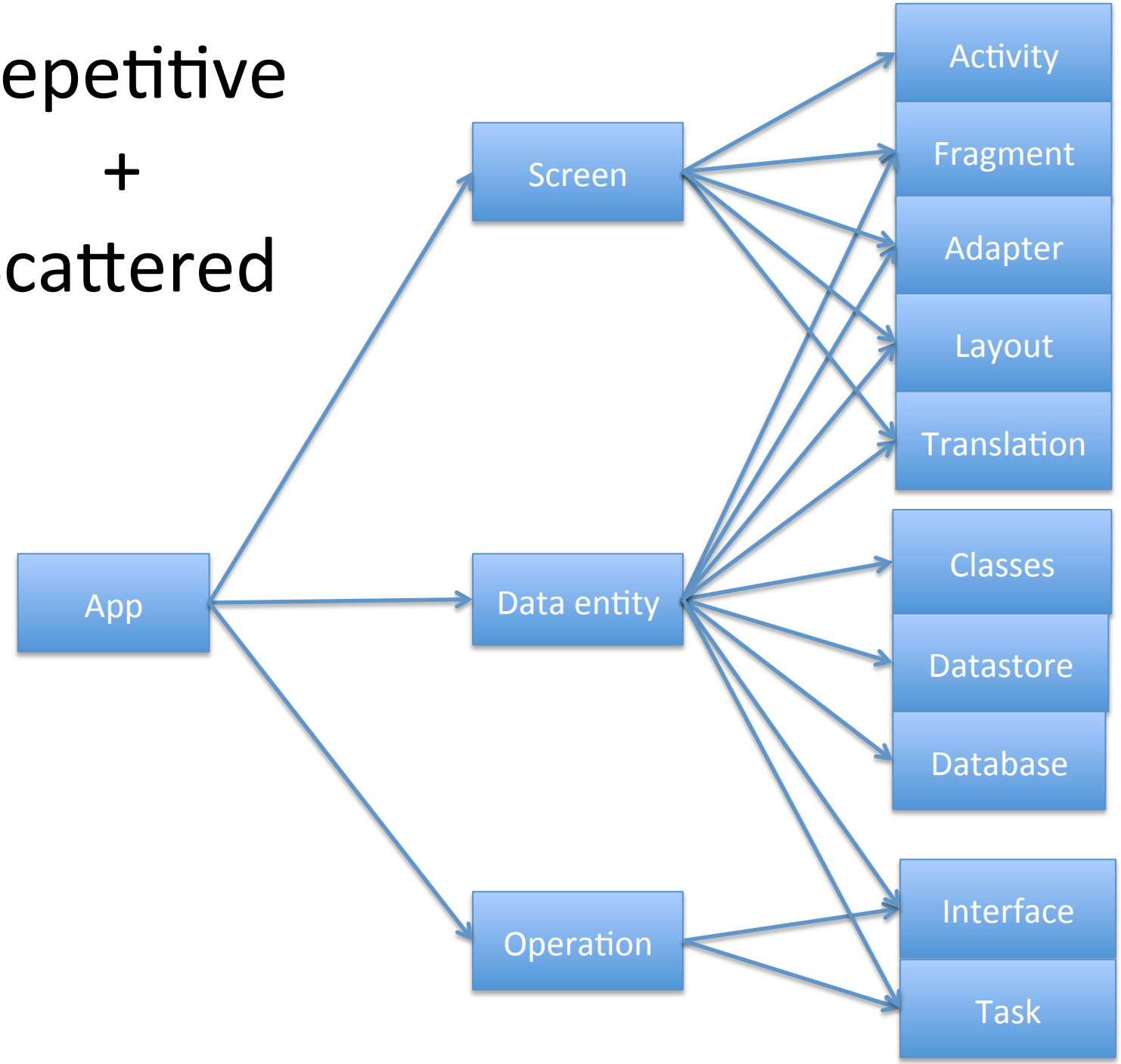
Observations

- Repetitive:
  - A single “Entity” always ends up in:
    - data (class, datastoreI, database)
    - screen (view, presenter, layout, translations, etc)
- Scattered:
  - Single concept ends up in many places
- But predictable:
  - For “data-driven” apps (like banking)

Repetitive

+

Scattered



Goal

- Focus on the essentials

-> Deliver quality fast

-> Keep delivering fast

How?

- Automate the predictable part:
  - Entity, Screen, Operation

- Risk:
  - Loose flexibility?

Automation

Model-driven approach:

1. Meta-model for apps:
   - Entity, Screen, Operation
2. Use a DSL to “describe” your app
3. Generate code from model
4. Manually add and refine

DSL

- Domain Specific Language
- Very readable and expressive
- For specific problem only

Styles:

- Internal DSL:
  - Fluent interface
- External DSL:
  - Excel, XML, Language workbench (XTEXT)

External DSL (XTEXT)

```
Enumeration FuelType {
        diesel
        petrol
        lpg
}

Structure Car {
        prim String       [1]     brand
        prim String       [1]     type
        prim Integer      [1]     yearBuild
        prim Image        [0..1] photo
        enum FuelType     [1]     fuleType
 }

InputScreen CarForm {
      dataType: Car
      fields: brand type yearBuild color
      submitButton: Save
      button: AddPhoto
}
```

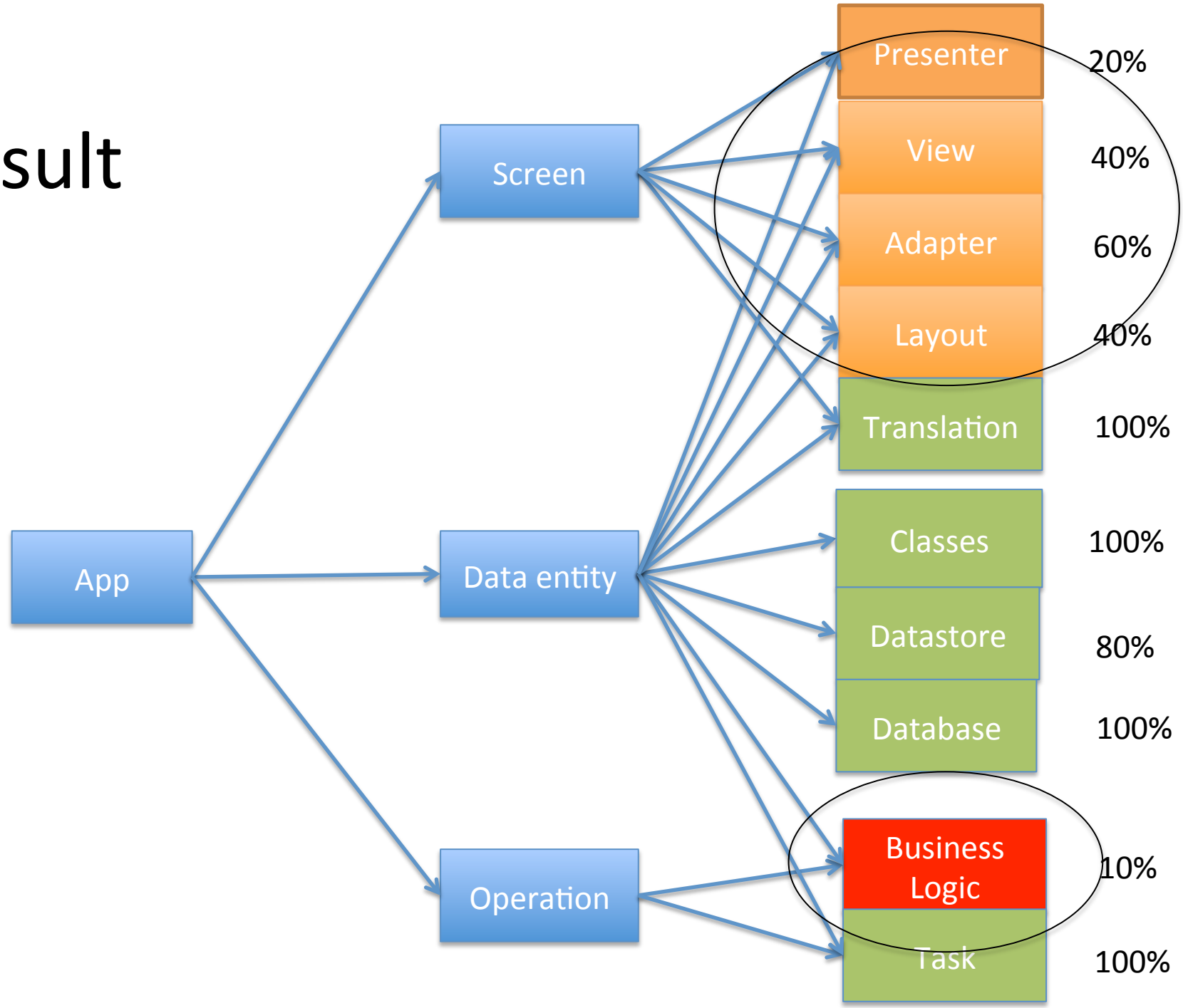
Internal DSL (fluent Java)

```
Enumeration fuelType = register(
  enumeration(”FuelType")
     .with( literal(”diesel") )
     .with( literal(”petrol") )
     .with( literal(”lpg") )
  );

Structure car = register(
  structure("Car”)
     .with( string( "brand", mandatory) )
     .with( string( "type", mandatory ) )
     .with( integer("yearBuild", mandatory ) )
     .with( image( "photo", optional) )
     .with( enumeration( ”fuelTYpe", mandatory , fuelType ) )
);

Screen carFormScreen = register(
   inputScreen("CarForm", car)
        .forFields("brand", "type", “yearBuild”, ”fuelType")
         .withSubmitButton(button(”AddPhoto"))
         .withButton(button("Save"))
);
```

Code generation

- Clear target architecture
- Separate generated and hand-written
- Understandable and testable
- Be able to ignore, refine, extend the generated part

Result


App
Screen
Data entity
Operation
Presenter
View
Adapter
Layout
Translation
Classes
Datastore
Database
Business Logic
Task
20%
40%
60%
40%
100%
100%
80%
100%
10%
100%

Refine by hand

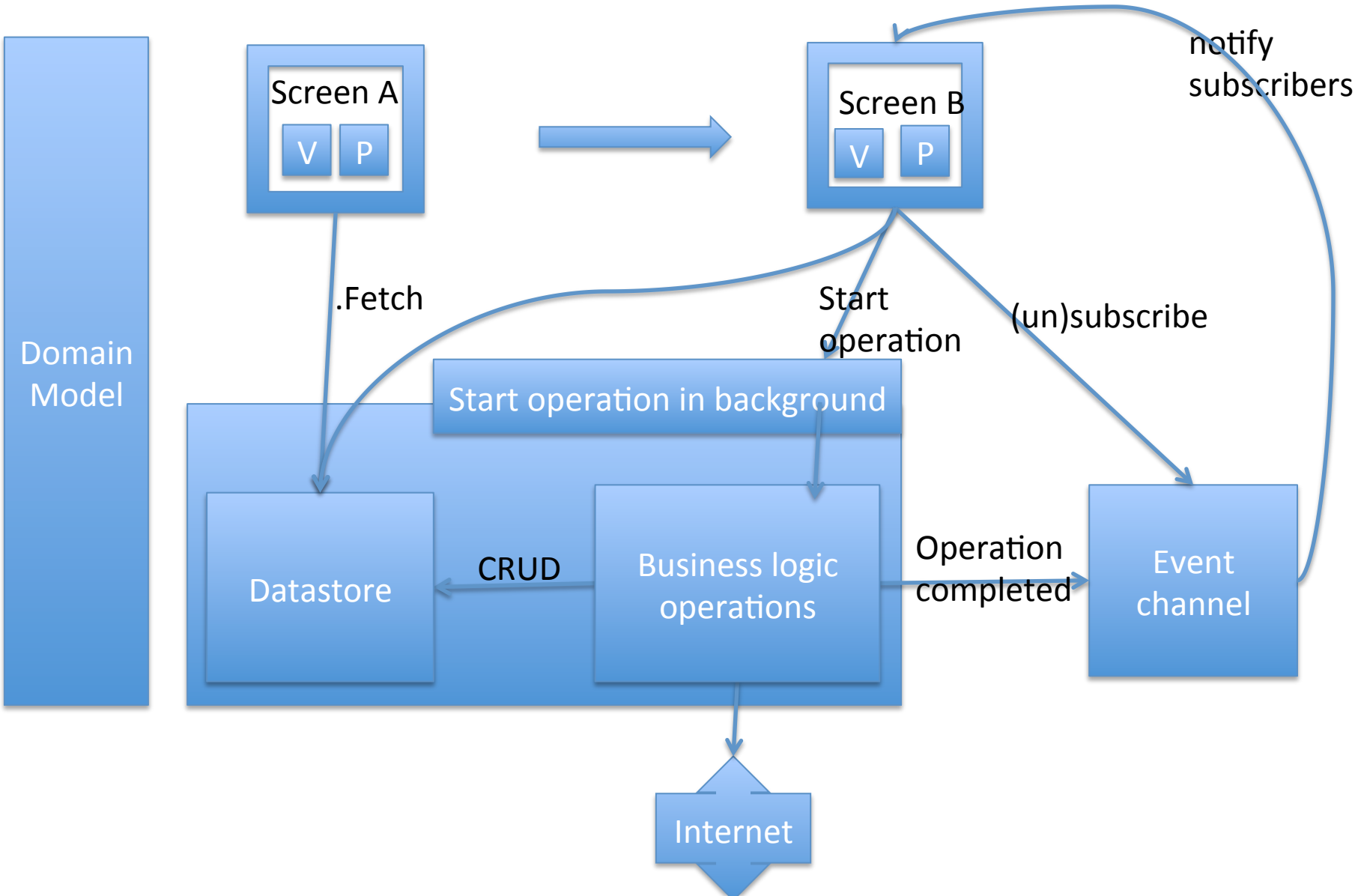
Non “predictable” stuff:

- Screens:
  - Logic
  - Polishing views
  - “Stylesheet”
  - “Exceptional” functionality
- Operations:
  - Business logic, remote communication
- DataStore:
  - Non standard queries

Example: XebiconApp

- Example project:
  - Focus on internals: No fancy UI (yet)
- DSL:
  - Low tech: fluent java + StringTemplate
  - Extendable language (add MapScreen or Money-primitive)
- Code generator:
  - Selective overriding and disabling
- Architecture:
  - Clear programing model (MVP + Async)
- Source code:
  - Separate generated and manual
  - Overridable and extendable
  - Testable

Architecture



Rough results (1:25)

Example project XebiconApp:

- DSL:
  - (Xtext: 80 SLOC)
  - Java: 130 SLOC
- Android:
  - Generated:
    - Java: 2600 SLOC
    - XML: 900 lines
  - Hand-written:
    - Java: 800 SLOC

Conclusion

- Approach suitable for most apps
- Introduces clear vocabulary
- Consistent code
- Productive:
  - 1 line of DSL => 25 lines of source

Questions?

XebiCon 2013

A DSL for generating mobile applications

by Marc Grol

http://bit.ly/xc_a

Remember

1:25