What is a Composite Smart Client?
Contact Center Agent Desktop

Current Session: John Smith - (425) 533-3335

Customer: John Smith - (425) 533-3335
First Name: John
Last Name: Smith
Phone Number: 4255333335

Open Applications: Billing, CRM, Current Promotions, Current Application: CRM

Update Customer Profile

Note: * indicates required field.

Personal Details
First Name: John
Middle Initial: R
Last Name: Smith
Gender: Male

Contact Details
Address: North Avenue
City: Seattle
Zip Code: 98007
State: PENNSYLVANIA
Email: JohnSmith@hotmail.com
Date of Birth: 01/11/1963
Contact Number1: 4255333335
Contact Number2: 4545454545

Billing Information
Billing Address: North Avenue
Billing State: PENNSYLVANIA
Billing City: Seattle
Billing Zip Code: 98101

Bank/SSN Information

On call with John Smith (425) 533-3335 [Connected] 11:03 AM

July 24, 2005
Portland Code Camp 1.0
Technology
Dependency Injection

*The "old way":* Components have intimate knowledge of services

```java
public class Component {
    private Service svc = new Service();
}
```

Component

Service
Dependency Injection

*In the right direction:* Component uses services by interfaces and factories

```csharp
public class Component {
    private IService svc = ServiceFactory.GetService();
}
```
Dependency Injection

The "CAB way": Dependencies are injected at runtime instead of compile time

```csharp
public class Component
{
    [ServiceDependency]
    private IService svc;
}
```
Event Broker

The "old way": Events are wired up at compile time

```csharp
public class Sender {
    public event EventHandler evt;
}

public class Listener {
    mySender.evt += new EventHandler(myHandler);
}
```
Event Broker

The "CAB way": Events are wired up at runtime

```csharp
public class Sender {
    [EventPublication("topic://Sender/event1")]
    public event EventHandler evt;
}

public class Listener {
    [EventSubscription("topic://Sender/event1")]
    private void MyHandler(object sender, EventArgs e)
}
```
Event Broker

Multiple subscribers

Listener 1

Listener 2

Sender
Event Broker

Multiple publishers
Against multiple subscribers, too

Listener 1 -> Sender 1
  topic://sender/event

Listener 2
  topic://sender/event

Sender 2
Demo

Event Broker
Command Dispatching

- De-couples UI elements from command handlers
- Architecture similar to Event Broker
Demo

Command Dispatching
Modules

- Isolated, reusable functionality
- Loosely coupled teams
- Role-based modules
Demo

Modules
CAB Application Architecture

Application Host
Shell

Module A
WorkItem A
- Services
- Monitors
- State
- Controller A
- View A
- Controller B
- View B

Module B
WorkItem C
- Services
- Monitors
- State
- Controller A
- View A
- Controller B
- View B
CAB Architecture

- Composite UI (WinForms)
- Composite UI
- Component Model
CAB Architecture

Composite UI (WinForms)  Composite UI (Avalon)  Composite UI (3rd party)

Composite UI

Component Model
Views & Controllers

- Views present model data to the user
- Controllers contain the business logic
- Automatically wired up by the container
- A view has one and only one controller
- A controller may control many views
Work Item

- Represents a use case in the system
- Specialized container for
  - Model Data
  - Views
  - Controllers
  - Workspaces
  - Services
- Key point of reusability
Workspaces

- Workspaces are canvases onto which you can paint your views

- For Windows Forms, we ship:
  - WindowWorkspace
  - MdiWorkspace
  - ZoneWorkspace
  - TabWorkspace
  - DeckWorkspace
Demo

Bank Teller QuickStart
Q & A