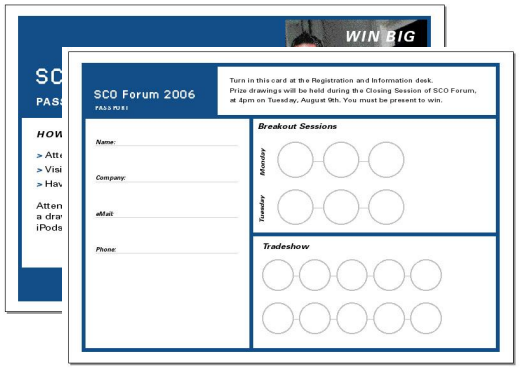
SCO Forum 2006

MOBILITY EVERYWHERE >

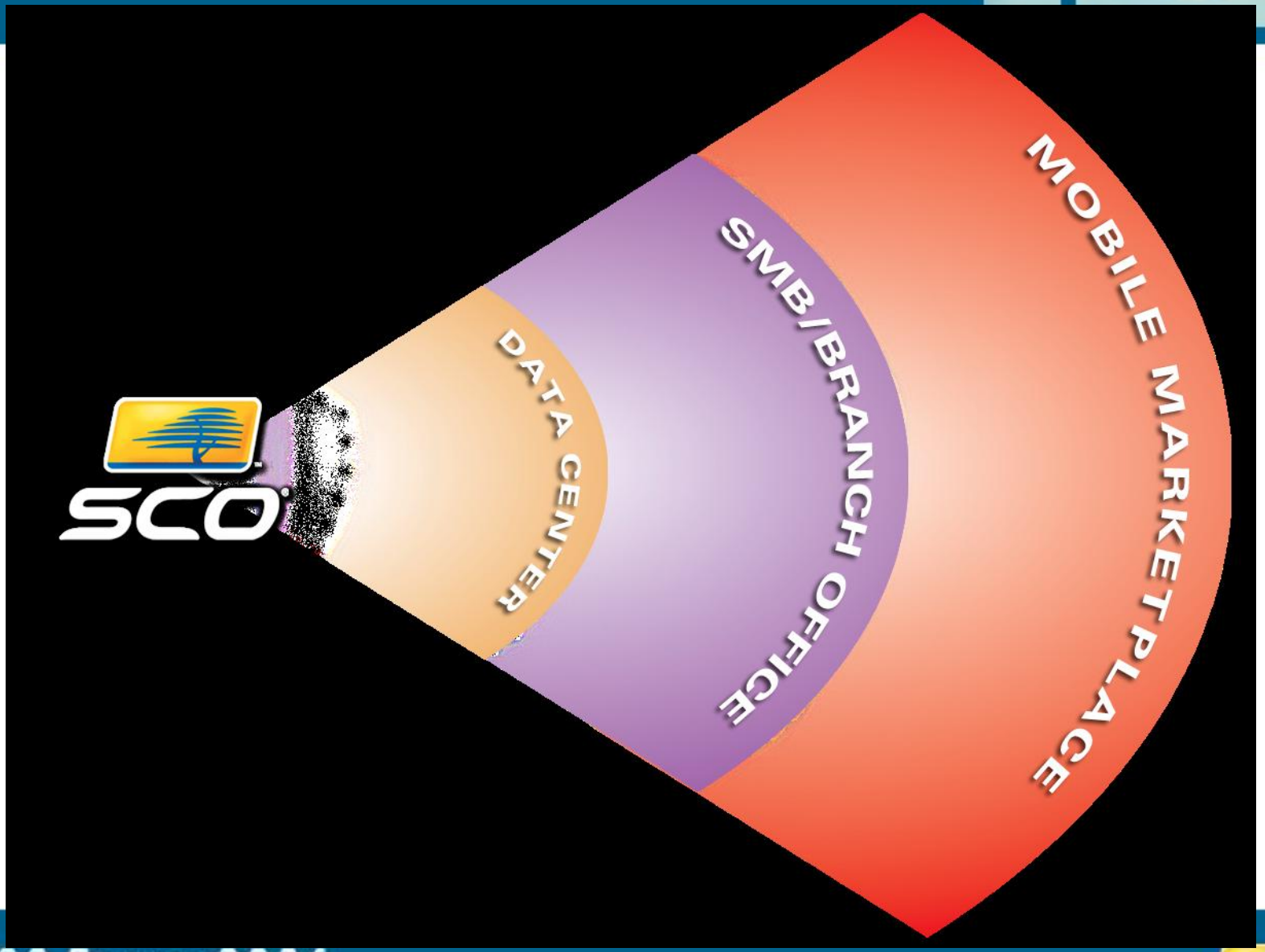
Presentation Title: EdgeClick Services Revealed
Presenter's Name: Alexander Sack
Session ID: 119

Get Your Passport Stamped

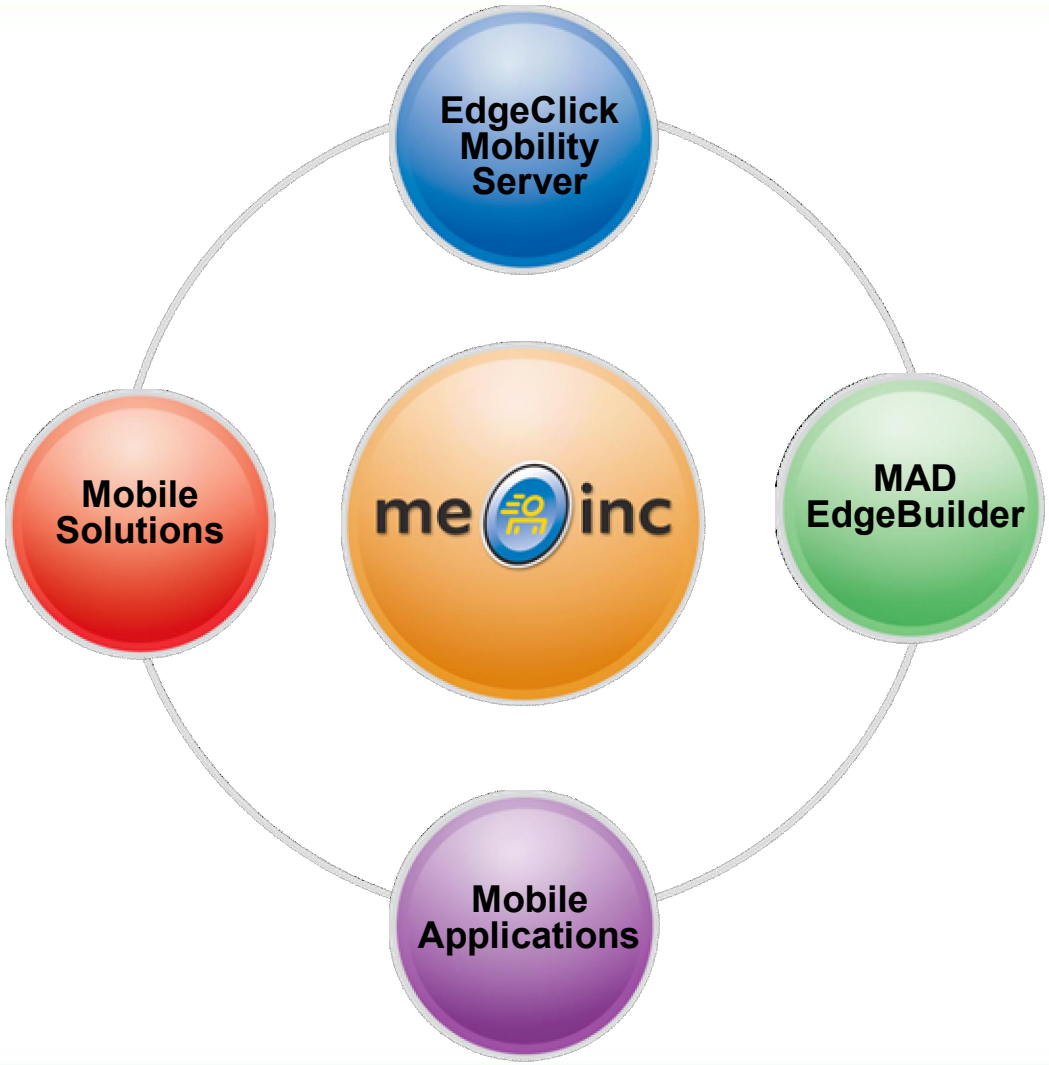
- Be sure to get your Passport stamped.
  - Get your passport stamped
    - By breakout session instructors
    - By exhibitors in the exhibit hall
  - Turn in your Passport
    - After the last breakout session on Wednesday
    - Drawing for great prizes for Wrap-up Session
- Remember to complete the breakout session evaluation form, too



SCO Automates Transactions



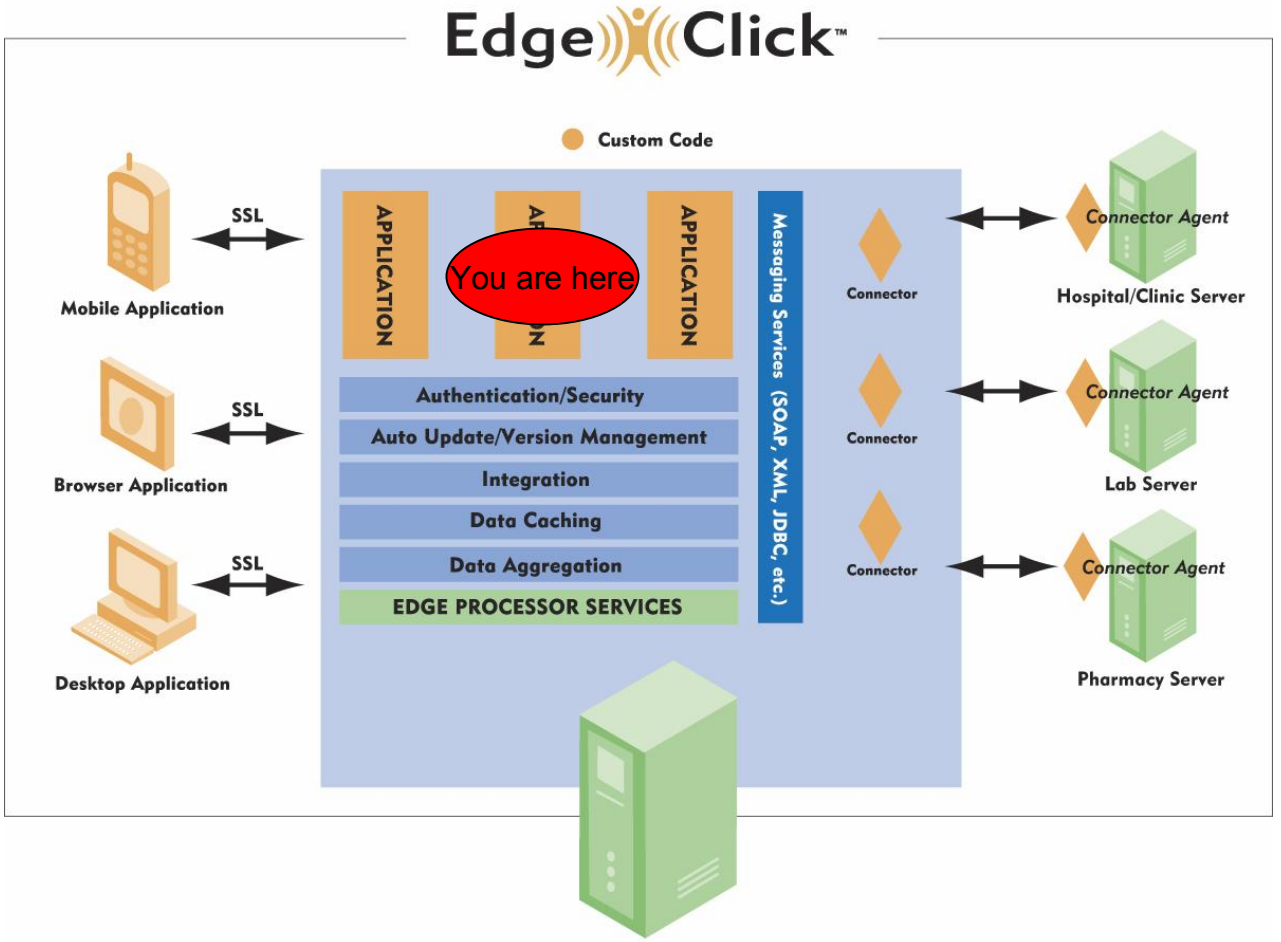
Huh?


EdgeClick Mobility Server
Mobile Solutions
me inc
MAD EdgeBuilder
Mobile Applications

Agenda

- What, Why and How?
- Dissecting an EdgeClick Service
- Some Do's and Don'ts
- EdgeClick Mobility Server Core Services
- EdgeBuilder SDK Development Tools
- Future Directions

"Where are we?"


Edge Click™
Custom Code
SSL
Mobile Application
SSL
Browser Application
SSL
Desktop Application
APPLICATION
APPLICATION
You are here
Authentication/Security
Auto Update/Version Management
Integration
Data Caching
Data Aggregation
EDGE PROCESSOR SERVICES
Messaging Services (SOAP, XML, JDBC, etc.)
Connector
Connector
Connector
Connector Agent
Hospital/Clinic Server
Connector Agent
Lab Server
Connector Agent
Pharmacy Server

What is an EdgeClick Service?

- Contains the main processing of the application
- Reuses existing services provided by the ECP
- Consumes application specific requests made by the mobile clients on behalf of subscribers
- Hot deployable
- Managed by the ECP Admin site

Why do I need EdgeClick Services?

- Mobile clients have a limited runtime environment for heavy business intensive processing
  - Limited remoting (RMI, WebServices, etc.)
  - Small memory and CPU power
  - Idiosyncratic runtimes depending on manufacture
  - Security constraints
- No direct access to the enterprise
  - The ECP is an edge-of-network server that provides a secure façade into the enterprise for mobile clients
  - Your server code can orchestrate data across multiple enterprises via connectors and agents (see Session 124)
- Leverage existing services provided on the ECP

Why do I need EdgeClick Services (cont'd)

- "Okay, do I really need to write an EdgeClick Service to mobilize my application or solution?"
  - Do you have direct access to the enterprise?
  - Do you need to manage different accounts and subscribers?
  - Do you need to orchestrate or federate data from various sources to the mobile device?
  - Does your application involve complicated business logic?
  - Does your target mobile device have a sufficient runtime environment for your solution?

How do I write EdgeClick Services?

1. Implement the IProcessor interface and extend the BaseProcessor class
2. Define your IProcessor class implementation as @Stateless
3. Register the object in JNDI space
4. Set your service code, version, and name
5. Implement client specific commands
6. Return a Result object with a return code

How do I write EdgeClick Services?

7. Build and package any number of services in an EAR
8. Deploy your EAR
9. Manage service using the ECP admin site
10. Test service using emulator, mobile client, or static web page

Dissecting an EdgeClick Service

- Let's take a look at the HelloService within the EdgeBuilder SDK to learn the anatomy of a service!
- %EC_HOME%\sdk\service\samples\HelloService
- Project can be built with Apache Ant from the command line OR use your favorite IDE (Eclipse, NetBeans, etc.) to import the project and build

Dissecting an EdgeClick Service (cont'd)

- Steps 1-3:

```
@Stateless
@LocalBinding (jndiBinding =
     edgeclick/services/HelloWorldService/local" )
public class HelloProcessor extends BaseProcessor implements
     IProcessor {
```

- HelloProcessor class is a Stateless Session Bean
  - Service will be managed by the EJB3 container
  - Annotations WILL BE REPLACED by custom annotations in next release
- Registered in the right JNDI namespace
  - ECP searches over "edgeclick/services/*/local" to find appropriate service
- Extends abstract BaseProcessor class
  - Inherits common methods and fields

Dissecting an EdgeClick Service (cont'd)

- Step 4:

```
public HelloProcessor() {
      setAppCode(_applicationCode);
      setAppVersion("_test_");
      setProcessorName("HelloService");
 }
```

- Application code is locally defined
  - Currently codes are fabricated by the service writer
- "_test_" special version
  - By passes application partitioning tests
  - Use for unit testing your service
- Processor name is what is displayed by the ECP Admin site

Dissecting an EdgeClickService (cont'd)

- Step 7:

```
edgeclick-service-hello.jar
     META-INF/MANIFEST.MF
     com/sco/edgeclick/HelloService/HelloProcessor.class
edgeclick-service-hello.ear
     META-INF/MANIFEST.MF
     META-INF/application.xml
     edgeclick-service-hello.jar
```

- The edgeclick-service-hello.jar contains your service
  - If you need to use third-party libraries as part of your service that are not included with the ECP, use the "Class-Path" manifest entry and package them relative to your EAR
- The application.xml file specifies what modules are included in the EAR and how to deploy them
  - More advanced examples could include a Web Archive (WAR) that includes a web front for your service

Dissecting an EdgeClick Service (cont'd)

- Step 8:

```
$ ant deploy
Buildfile: build.xml
deploy:
     [copy] Copying 1 file to …
Build Successful
Total time: 1 second
```

- The HelloService EAR file is copied to the appropriate deploy directory under the ECP
  - The app server's deployer will automatically see a new file and deploy the EAR
  - If you already deployed the service, the deployer will redeploy if there are any changes

Dissecting an EdgeClick Service (cont'd)

- Step 9:
  - Login into ECP Admin site using EdgeClickAdmin account
  - Click "Admin Portal"
  - Click "Manage Applications"
  - Verify your application is listed
    - If your application is NOT listed, make sure you set an application code, version, and name
  - Add subscribers and/or groups to the application
    - If your application version is "_test_", this is not necessary as all subscribers have access to the service

Dissecting an EdgeClick Service (cont'd)

- Step 10:
  - For the HelloService, there is a static test page that emulates a client
  - %EC_HOME%\sdk\service\samples\HelloService\test\sample-test.html
- Let's edit the sample-test.html file and try to send it different name/value pairs
- Watch the console print the name/value pairs that are sent as well as the response back to the client

Do's and Don'ts

- Don't try to save STATE across mobile client sessions
  - Services are stateless and pooled
- Don't forget to set application code, version and name
  - Use "_test_" when developing new services
- Do use the "edgeclick-service-<yourname>" naming convention
  - The deployer is alphabetical and if you have intra-service dependencies, you need to be sensitive to deployment order
- Do reuse other services provided by the ECP
- Do give feedback to EdgeBuilder Developer's Program about the APIs and documentation

EdgeClick Processor Core Services

- IPhoenixCommons
  - API to get access to information about Accounts, Subscribers, Groups through the ITeam interface
  - Also provides a way to get access to other services such as the PostOffice service (SMS)
- IRoadrunnerApi
  - Set of APIs that represent common useful utility functions that the ECP and services can use
  - Overtime this API will be expanded to provide more and more useful internal functionality
- Services can be directly injected using the @EJB annotation for now

EdgeBuilder SDK Development Tools

- EdgeBuilder SDK comes with both NetBeans 5.5 Beta 2 and Eclipse 3.1.2
  - Eclipse has a dedicated JBoss IDE plugin that can be used to start and stop JBoss automatically
  - NetBeans can be configured to also work with JBoss and a plugin is in the works (announced at JavaOne)
- Apache Ant 1.6.5
  - Equivalent to "make" in C/C++
  - Can be used instead if you prefer to work outside of an IDE

Future directions

- EdgeClick Annotations
  - Abstract the EJB specification to make service writing more intuitive
- Web Services based interfaces
  - Services can be written in any language
- Custom deployer
  - Solves intraservice dependency issues
- Automatic registration
  - App codes will be better defined to void collisions
- Configurable lifecycle management functions
- EdgeClick IDE Plugins

SCO Forum 2006

MOBILITY EVERYWHERE >

Q & A