Technologies distribuées

Java EE 5

1

Introduction

- Plan
  - Présentation générale
    - L'entreprise et le Java EE
    - Définition
  - Le Java EE
    - L'architecture multicouche
    - Présentation des composants
    - Le serveur d'application & les composants
  - Les composants et leur clients
    - Les composants de la logique de présentation
    - Les composants métier de type EJB
    - Les clients EJB
    - Les composants de type services web
    - Les clients WS
  - Exemple & mise en œuvre

2

Introduction

- Compétences acquises
  - Compréhension
  - Interventions
  - Affranchir des contraintes
- Profil
  - POO, Design Pattern
- Intérêts
  - Professionnalisation
  - Démystifier

3

Plan

- **Présentation générale**
  - **L'entreprise et le Java EE**
  - Définition
- Le Java EE
  - L'architecture multicouche
  - Présentation des composants
  - Le serveur d'application & les composants
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - Les clients EJB
  - Les composants de type services web
  - Les clients WS
- Exemple & mise en œuvre

4

L'entreprise et le Java EE

- Editeurs
- Service (encore) public
- Banques
- Assurance
- Santé
- …

5

Plan

- **Présentation générale**
  - L'entreprise et le Java EE
  - **Définition**
- Le Java EE
  - L'architecture multicouche
  - Présentation des composants
  - Le serveur d'application & les composants
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - Les clients EJB
  - Les composants de type services web
  - Les clients WS
- Exemple & mise en œuvre

6

Définition

- Java Enterprise Edition
  - Norme Sun
  - Plateforme Java EE
    - Infrastructure d'exécution (Serveur d'applications)
    - Ensemble de services :
      - Transaction, localisation, concurrence, sécurité, persistance, connectivité …

Plan

L'architecture J2EE

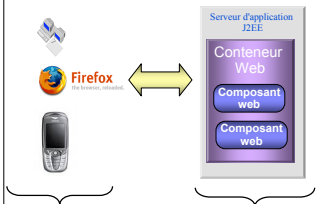
- Architecture multi niveaux
  - Interface utilisateur
  - Logique de présentation & Services Web
  - Logique métier
  - Service d'infrastructure (Transverse)
  - Données

Interface utilisateur

Couche 1
Client

- Application de bureau
- Navigateur WAP
- WEB Browser
- Tiers applications
- …

Logique de présentation & Services web

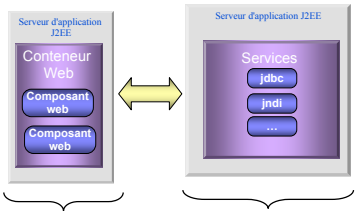


Couche 1
Client

Couche 2
Logique de présentation
Services Web

- Logique présentation
  - Règles d'affichage
  - Traitement des requêtes
- Services Web
  - Echange de données entre applications

Services d'infrastructure

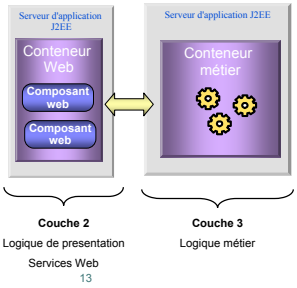


Couche 2
Logique de presentation
Services Web

Couche Transverse
Couche transverse de services internes

- Fonctionnalités fournies aux composants

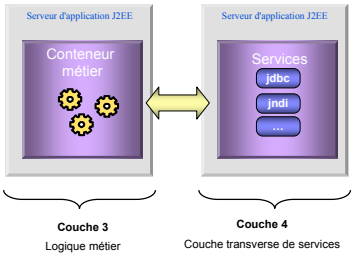
Logique métier



- Règles métiers de l'entreprise

13

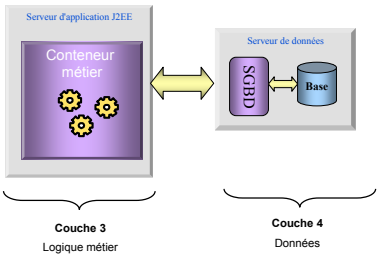
Services d'infrastructure



- Fonctionnalités fournies aux composants

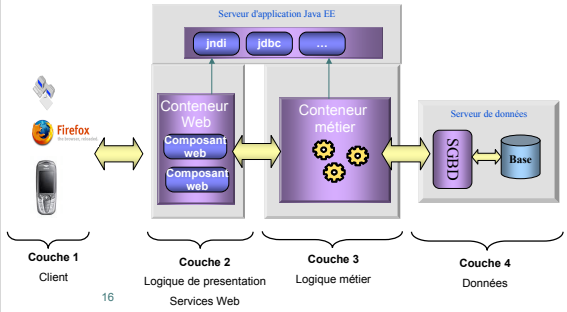
14

Données



15

Architecture multi couches MVC



16

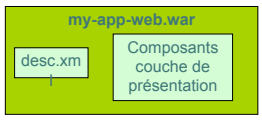
Plan

- Présentation générale
- **Le Java EE**
  - L'architecture multicouche
  - **Présentation des composants**
  - Le serveur d'application & les composants
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - Les clients EJB
  - Les composants de type services web
  - Les clients WS
- Exemple & mise en œuvre

17

Présentation des composants

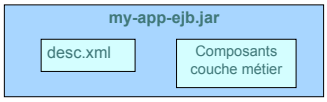
- Les composants web
  - Rôle :
    - Logique de présentation
      - Génération du code IHM client
      - Traitement des requêtes client
      - Appel du métier
    - Service Web
      - Traitement des requêtes tiers d'application
      - Appel du métier
  - Présenté par un descripteur XML
  - Archive WAR



18

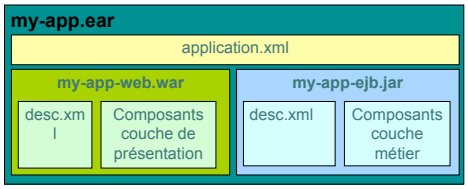
Présentation des composants

- Les composants métiers
  - Rôle :
    - Chargé des traitements
    - Factorisation
  - Présenté par un descripteur XML
  - Archive JAR



Présentation des composants

- Une application Java EE ? Composée de composants
  - Archive entreprise (EAR)
    - Archive Web
    - Archive Métier
    - Descripteur XML : application.xml



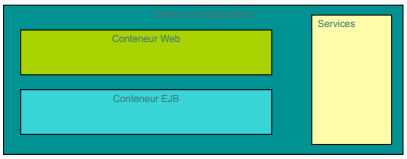
Plan

- Présentation générale
- **Le Java EE**
  - L'architecture multicouche
  - Présentation des composants
  - **Le serveur d'application & les composants**
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - Les clients EJB
  - Les composants de type services web
  - Les clients WS
- Étude d'un exemple & mise en œuvre

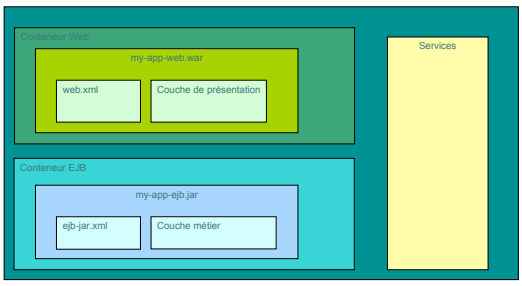
Le serveur d'application & les composants

Plateforme J2EE

- Infrastructure d'exécution
  - Conteneur WEB
  - Conteneur EJB
  - Services d'infrastructures et de communication



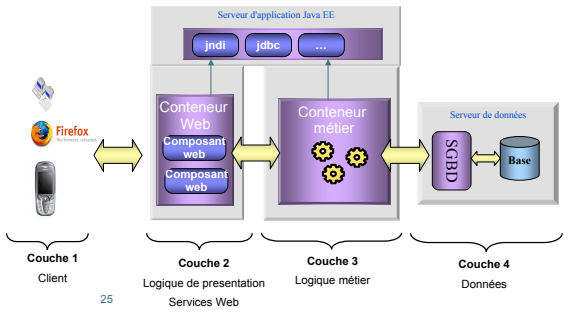
Le serveur d'application & les composants



Plan

- Présentation générale
- Le Java EE
- Les composants et leur clients
  - **Les composants de la logique de présentation**
  - Les composants métier de type EJB
  - Les clients EJB
  - Les composants de type services web
  - Les clients WS
- Exemple & mise en œuvre

Architecture multi couches MVC



Étude des composants web

La logique de présentation

- Fonctionnement : MVC
- Le modèle :
  - Les données
  - Les procédures de contrôle de validité des données
  - Il ne connaît pas les vues
- La vue : Représente la visualisation des données
  - Représente les données pour l'utilisateur
  - Passe les actions de l'utilisateur au contrôleur
- Le contrôleur :
  - Traite les actions des utilisateurs
  - Gère la cinématique
  - Appel aux procédures « métiers »

Conteneur Web
Composant web

Plan

- Présentation générale
- Étude du J2EE
- Les composants et leur clients
  - Les composants de la logique de présentation
  - **Les composants métier de type EJB**
  - Les clients EJB
  - Les composants de type services web
  - Les clients WS
- Étude d'un exemple & mise en œuvre

Conteneur métier

Étude des composants de la couche métier

- Composant métier (rappel) :
  - Encapsule la logique métier
- Implémentation : EJB
- Avantages
  - Simplification des développements
    - Services du conteneur
  - Décharge le client de la logique métier
  - Factorisation

Conteneur métier

EJB, le composant métier

- Composition d'un EJB
  - L'interface REMOTE ou LOCAL
  - L'implémentation
- Les types d'EJB
  - EJB Session
    - Rend un service
  - EJB Entity
    - Représente une entité métier
  - Message-Driven
    - Listener de fil

Conteneur métier

EJB Session

- Avec état
  - Relation client : 1-1.
- Sans état
  - Relation client : 1-N.

Conteneur métier

L'interface REMOTE

```
package contact.book.interfaces;

import contact.book.entity.Contact;

public interface ContactServiceRemote {

    public static String JNDI = "ContactBook-Core/ContactServiceImpl/remote";

    public Contact addContact(Contact c);
}
```

31

L'implémentation

```
package contact.book.services;

import javax.annotation.EJB;

@Stateless
@Remote(ContactServiceRemote.class)
public class ContactServiceImpl implements ContactServiceRemote{

    @EJB
    private ContactDao contactDao;

    public Contact addContact(Contact c) {
        Contact contact = contactDao.addContact(c);
        return contact;
    }
}
```

32

EJB Entité

- Représente un Objet persistant
  - Prise en charge de la persistance :
    - Par l'EJB Entité lui-même
    - Par le conteneur

33

```
package contact.book.entity;

import java.io.Serializable;

@Entity
@Table(name = "contact")
public class Contact implements Serializable{

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int idContact;
    private String firstName;
    private String lastName;
    private String email;

    @ManyToOne
    @JoinColumn (name = "countryName")
    private Country country;

    public Country getCountry() {
        return country;
    }

    public void setCountry(Country country) {
        this.country = country;
    }

    public String getEmail() {
```

34

Rappel EJB Session

```
package contact.book.services;

import javax.annotation.EJB;

@Stateless
@Remote(ContactServiceRemote.class)
public class ContactServiceImpl implements ContactServiceRemote{

    @EJB
    private ContactDao contactDao;

    public Contact addContact(Contact c) {
        Contact contact = contactDao.addContact(c);
        return contact;
    }
}
```

35

Object d'accès aux données

```
package contact.book.dao.sql;

import javax.ejb.Local;

@Stateless
@Local(ContactDao.class)
public class ContactSqlDao extends AbstractSqlDao implements ContactDao {

    @PersistenceContext(unitName="ContactBookPU")
    protected EntityManager em;

    public Contact addContact(Contact c) {
        em.persist(c);
        return c;
    }
}
```

36

Plan

- Présentation générale
- Le J2EE
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - **Les EJB clients**
  - Les composants de type services web
  - Les clients WS.
- Exemple & mise en œuvre

EJB Client

- JNDI
- Contexte

```
public ContactServiceRemote getContactService(){
    try {
        ContactServiceRemote service = (ContactServiceRemote) ctx.lookup(ContactServiceRemote.JNDI);
        return service;
    } catch (NamingException e) {
        return null;
    }
}
```


EJB Client

```
public void test1ContactServiceEJBAddContact() throws Exception {
    ServiceLocator serviceLocator = ServiceLocator.getInstance();
    Contact contact = new Contact();
    contact.setFirstName("Jack");
    contact.setLastName("Kerouack");
    contact.setEmail("jack.kerouac@okiwi.org");

    Country country = serviceLocator.getCountryServiceRemote().findByName("France");
    contact.setCountry(country);

    contact = serviceLocator.getContactServiceRemote().addContact(contact);
    assertTrue(contact.getIdContact() > 0);
}
```


Plan

- Présentation générale
- Étude du J2EE
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - Les clients EJB
  - **Les composants de type services web**
  - Les clients WS
- Étude d'un exemple & mise en œuvre

Étude des composants web

Les services web

- Introduction aux web services
  - Echange de données inter-applicatif
- Qu'est ce qu'un Web service
  - Composant enveloppée d'XML
  - Fournis des fonctionnalités basiques

Étude des composants web

Les services web

- Le concept de Web service
  - Non propriétaire
  - Multiplateforme
  - SOAP / WSDL / UDDI
- Pourquoi faire ?
  - Collaboration de plateformes hétérogènes

```java
package contact.book.services;

import javax.annotation.EJB;[]

@WebService(serviceName = "ContactBookService")
@Stateless
@Remote(ContactServiceRemote.class)
public class ContactServiceImpl implements ContactServiceRemote{

    @EJB
    private ContactDao contactDao;
    @EJB
    private CountryServiceRemote countryServiceRemote;

    public Contact addContact(Contact c) {
        Contact contact = contactDao.addContact(c);
        return contact;
    }

    @WebMethod
    public int addContact(String firstName, String lastName, String email, String country) {
        Country c = countryServiceRemote.findByName(country);
        Contact contact = new Contact();
        contact.setFirstName(firstName);
        contact.setLastName(lastName);
        contact.setEmail(email);
        contact.setCountry(c);
        contact = this.addContact(contact);
        return contact.getIdContact();
    }
```

Conteneur Web
Composant web


```
- <definitions name="ContactBookService" targetNamespace="http://services.book.contact/jaws">
  - <types>
    - <schema elementFormDefault="qualified" targetNamespace="http://services.book.contact/jaws">
      - <complexType name="addContact">
        - <sequence>
            <element name="String_1" nillable="true" type="string"/>
            <element name="String_2" nillable="true" type="string"/>
            <element name="String_3" nillable="true" type="string"/>
            <element name="String_4" nillable="true" type="string"/>
          </sequence>
        </complexType>
      - <complexType name="addContactResponse">
        - <sequence>
            <element name="result" type="int"/>
          </sequence>
        </complexType>
        <element name="addContact" type="tns:addContact"/>
        <element name="addContactResponse" type="tns:addContactResponse"/>
      </schema>
    </types>
```


Plan

- Présentation générale
- Le J2EE
- Les composants et leur clients
  - Les composants de la logique de présentation
  - Les composants métier de type EJB
  - Les clients EJB
  - Les composants de type services web
  - **Les clients WS**
- Exemple & mise en œuvre

Web Service Client

- Génération des classes cliente via la WSDL

```
<property name="remote.wsdl" value="http://localhost:8080/ContactBookEjbServices/ContactServiceImpl?WSDL"/>

   <axis-wsdl2java
     output="${generated.dir}"
     testcase="false"
     verbose="true"
     serverside="true"
     skeletondeploy="true"
     url="${remote.wsdl}" >
     </axis-wsdl2java>
```


Web service client

```java
public void test1ContactServiceImplPortAddContact() throws Exception {
    contact.book.services.jaws.ContactServiceImplBindingStub binding;
    try {
        binding = (contact.book.services.jaws.ContactServiceImplBindingStub)
                new contact.book.services.jaws.ContactBookServiceLocator().getContactServiceImplPort();
    }
    catch (javax.xml.rpc.ServiceException jre) {
        if(jre.getLinkedCause()!=null)
            jre.getLinkedCause().printStackTrace();
        throw new junit.framework.AssertionFailedError("JAX-RPC ServiceException caught: " + jre);
    }
    assertNotNull("binding is null", binding);

    // Time out after a minute
    binding.setTimeout(60000);

    // Test operation
    int value = -3;
    value = binding.addContact("Web", "Services", "Web@Service.com", "France");
    assertTrue(value > 0);
    // TBD - validate results
}
```


Plan

- Présentation générale
- Le J2EE
- Les composants et de leur collaboration
- ***Exemple & mise en œuvre***

Conclusion

- Résumé
  - L'entreprise et le Java EE
  - Définition : Norme
  - L'architecture multicouches
  - Composants Web & Métier
  - Services d'infrastructure & de communications
  - Complémentarité EJB & Web Services
- Critique
  - Gros système
  - Forte utilisation du 1.4

Sources d'informations

- http://www.labo-sun.com
- http://penserenjava.free.fr/
- http://java.sun.com/reference/blueprints/

- http://struts.apache.org/
- http://www.springframework.org/
- http://www.hibernate.org/
- http://maven.apache.org/

THAT'S ALL FOLKS !!

- Questions ?
- Commentaires

- Contact
  - samir.hanna@okiwi.org