

RBC DEXIA Investor Services

Adoption et mise en place de Ruby on Rails

Sylvain Perez

sylvain.perez@rbcdexia-is.net

Paris on Rails - lundi 10 décembre 2007



RBC DEXIA
INVESTOR SERVICES

Déroulement de la présentation



RBC DEXIA
INVESTOR SERVICES

- RBC Dexia Investor Services
- Pourquoi Ruby on Rails ?
- Des besoins métiers aux patterns de développement
- Intégration au système d'information
- Retour d'expérience
- Le futur
- Conclusion

RBC Dexia Investor Services



- Qui sommes nous ?
 - Nouvelle banque issue de la joint venture des services d'investissement institutionnel de Dexia Group et de RBC Financial Group
 - Classé 1^{er} dépositaire global pendant 3 années consécutives (2005 → 2007)
 - Classé 1^{er} pour la qualité de son service client et des technologies utilisées (2005 et 2006)
 - Présent dans 15 pays et sur 4 continents : <http://www.rbcdexia-is.com/>
- Nos points forts
 - Solutions multi-marchés pour les fonds d'investissements
 - Expérience reconnue dans l'administration de fonds d'investissements
 - Offre de services groupés incluant la distribution et les activités de clientèle
- IT à Luxembourg
 - 190 informaticiens dont 50 externes (35% des effectifs au niveau du groupe)
 - Budget IT en croissance régulière depuis 2005 (15 à 20% du budget de la banque)



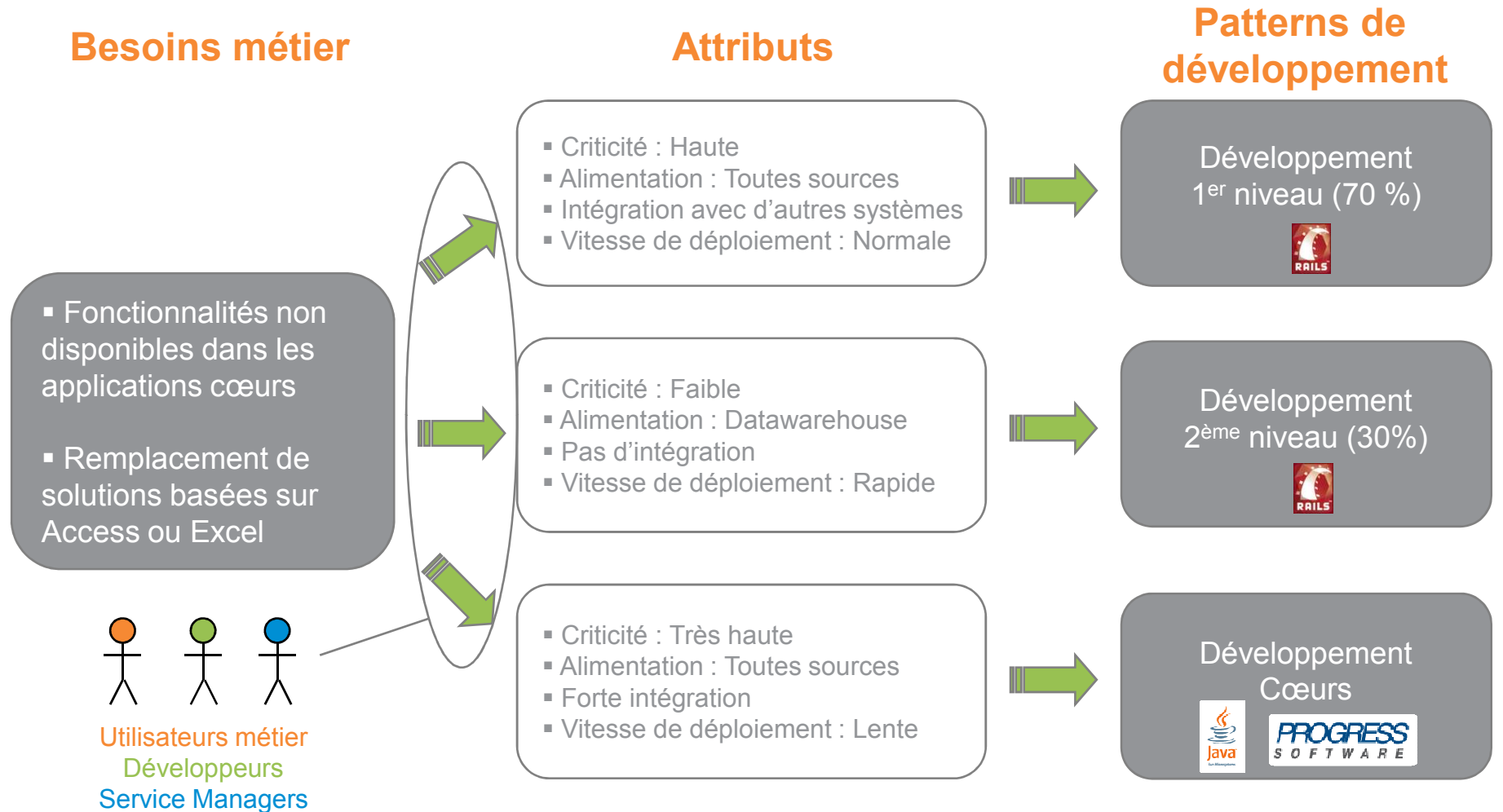
Pourquoi Ruby on Rails ?

- La situation
 - Nouvelle organisation IT suite à la joint-venture entre RBC et Dexia
 - Besoin d'une IT réactive pour des besoins métiers limités (Excel®, Access®, ...)
 - Beaucoup de demandes, de demandeurs et peu de temps
 - Sujets très variés
- Pourquoi a-t-on choisi Ruby on Rails (juin 2006) ?
 - Pas de compétence dans la solution WebDev® héritée de Dexia
 - Pas de technologie propriétaire et pas de Java !
 - Évaluation des frameworks agiles Symfony et Ruby on Rails
- Choix de Ruby on Rails
 - Simplicité et puissance du framework
 - Réduction des temps de développement
 - Ouverture et importance de sa communauté
 - Développement du « WebUploader » qui sert aussi de template

Des besoins métiers aux patterns de développement



RBC DEXIA
INVESTOR SERVICES

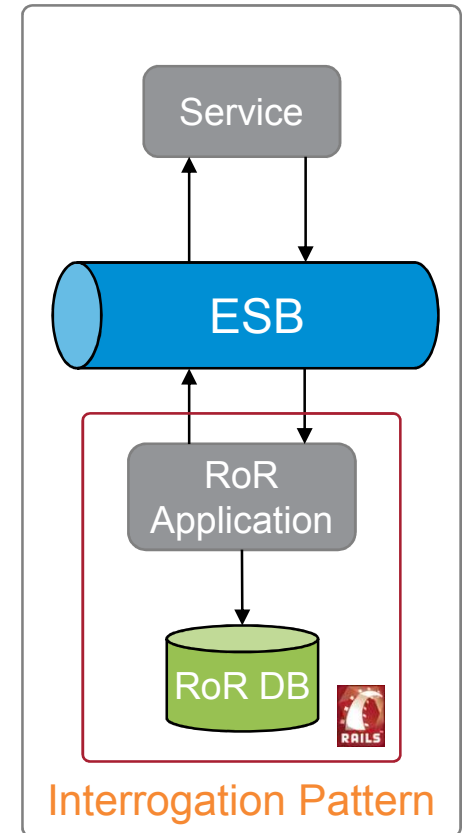
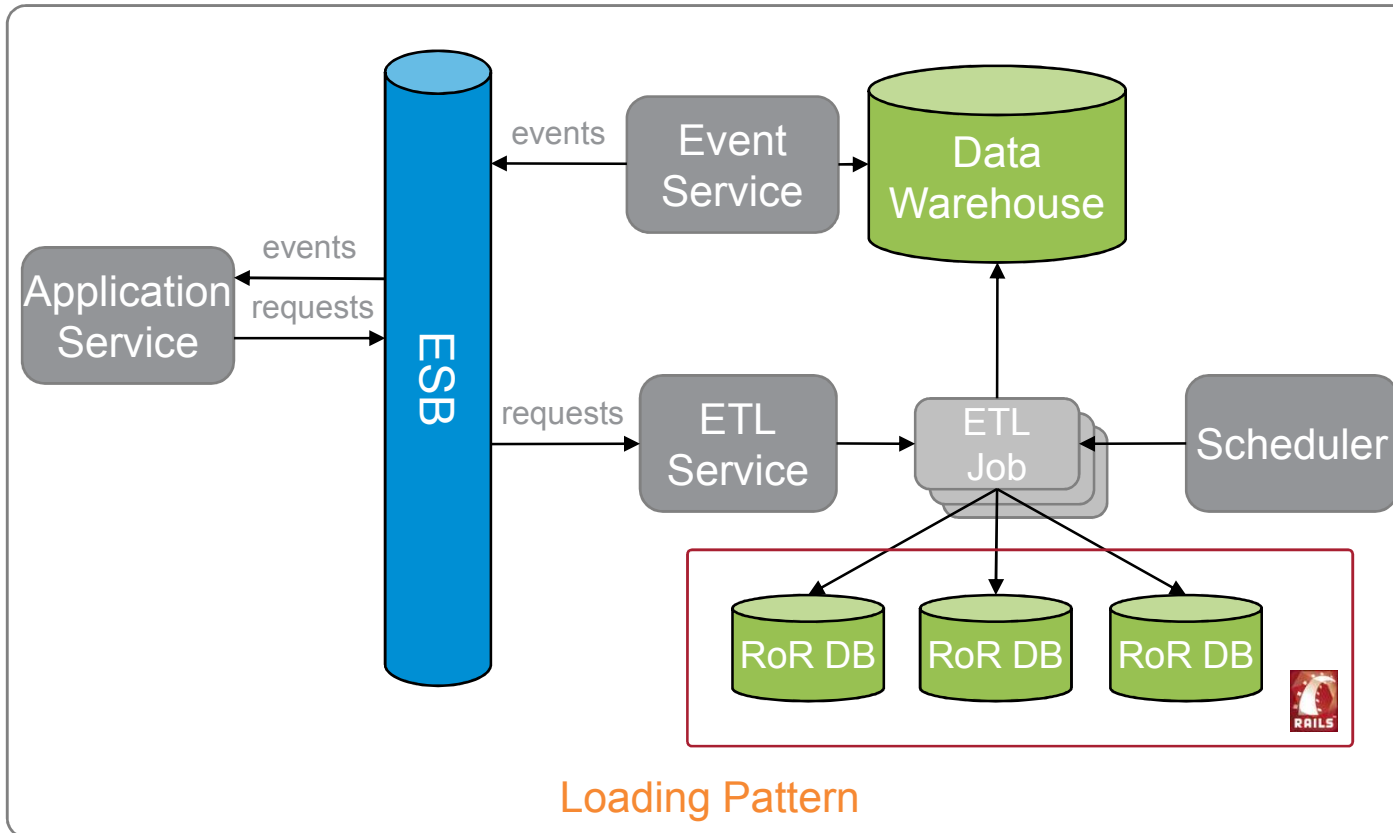


Patterns de développement

1^{er} niveau – Solution typique



RBC DEXIA
INVESTOR SERVICES

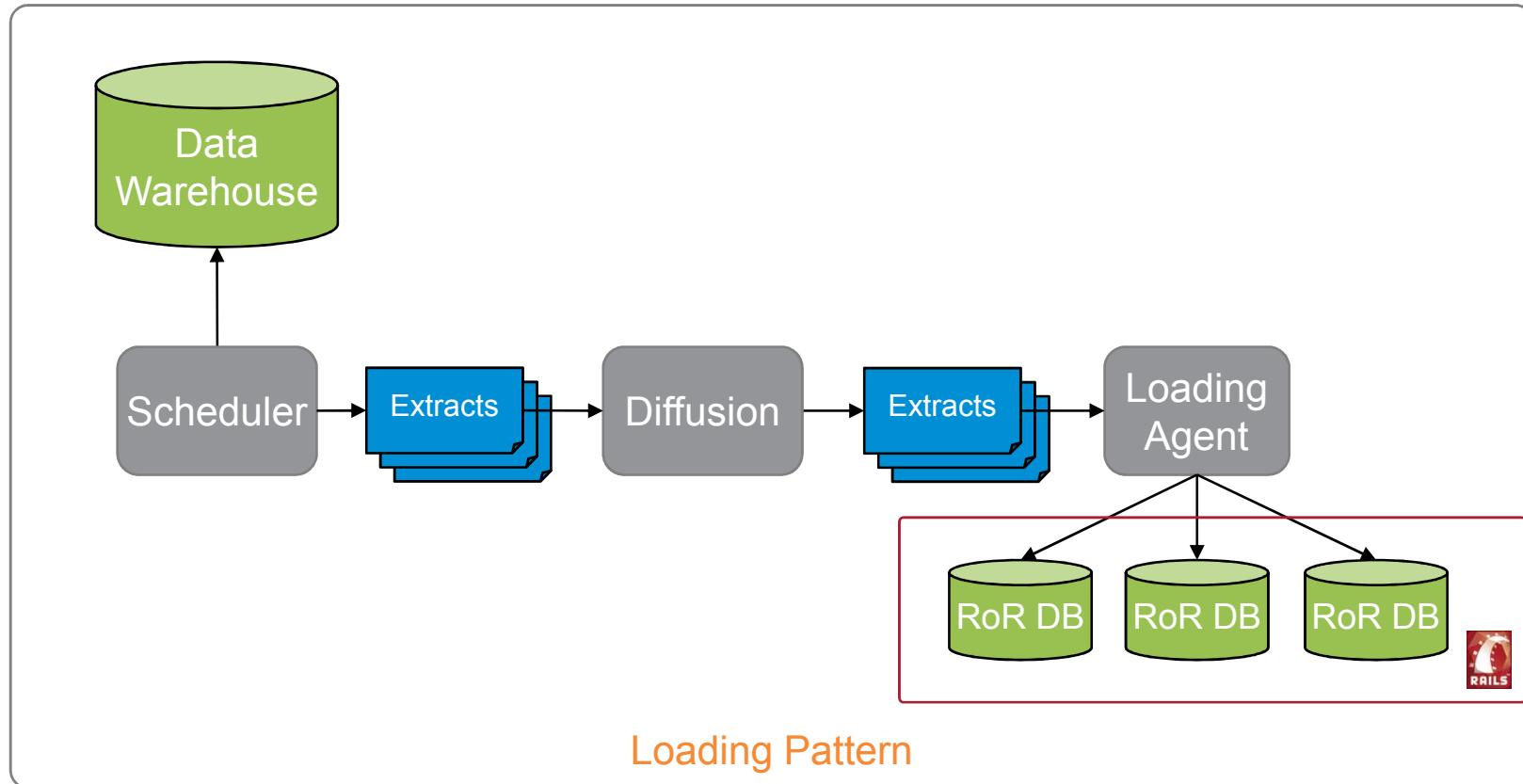


- 1 Datawarehouse → Event Service → Application Service
- 2 Application Service → BODI Service → Datawarehouse → RoR DB
- 3 TNG Scheduler → BODI Job



Patterns de développement

2^{ème} niveau – Solution typique



- 1 Datawarehouse → Scheduler → Extracts
- 2 Diffusion → Extracts → Loading Agent
- 3 Loading Agent → RoR DB

Intégration au système d'information

De l'isolé vers l'intégré



- Apparition progressive d'un besoin d'intégration avec les autres applications du système IT
- Initiative en cours de construction d'une architecture orientée service avec des produits IBM (WebSphere Message Broker et WebSphere Application Server)
- Mandat donné à IBM pour la construction d'un framework d'intégration appelé MOF (Message Object Framework) en Java
- Adoption des standards SOAP et WS-Addressing pour le format des messages échangés entre les applications



Intégration au SI

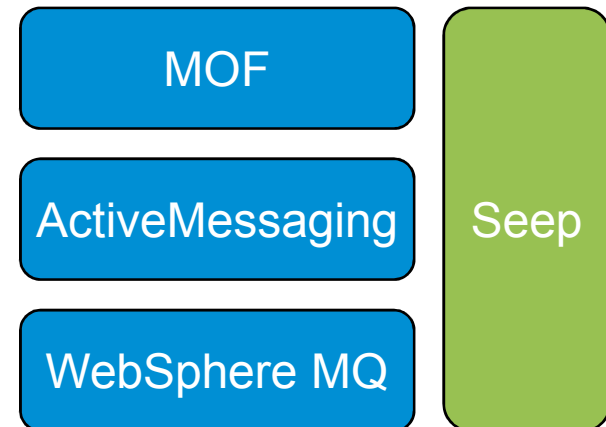
Le MOF pour Ruby on Rails



RBC DEXIA
INVESTOR SERVICES

Technologies

- MOF (plug-in Rails) pour une intégration via messages basés sur les standards RBC Dexia
- ActiveMessaging (plug-in Rails) équivalent JMS du monde Java
- WebSphere MQ pour la couche transport
- Seep (injection de dépendance) assure la configuration du MOF via policy



Fonctionnalités de la version 1.0

- Communications asynchrones
- Support des message JMS et non JMS
- Support du format propriétaire « DovReq »
- API simplifiée pour l'envoi et la réception de messages
- Basé sur une policy sous forme de fichier XML
- Générateurs Rails pour les MessageReceiver

Intégration au SI

Exemples MOF pour Ruby on Rails



RBC DEXIA
INVESTOR SERVICES

▪ Envoyer un message

```
message = MOFMessageFactory.create_message(1)
message.content = "Hello World !"
MOF.send(message)
```

« 1 » est la policy utilisée pour envoyer le message

▪ Recevoir un message

```
class MyMessageReceiver < MOFMessageReceiver
  def on_message(message)
    puts "Message received : #{message.content}"
  end
end
```

« on_message » est la méthode appelée par MOF quand il reçoit un message

```
C:\WINNT\system32\cmd.exe - ruby script/poller run
C:\LOCALAPP\Specific\rails_apps\echo>ruby script/poller run
MOF starting...
MOF Policy #1 : C:/LOCALAPP/Specific/rails_apps/echo/config/mof_policy_echo.xml loaded successfully !
MOF started successfully !
Loading C:/LOCALAPP/Specific/rails_apps/echo/app/processors/application.rb
Loading C:/LOCALAPP/Specific/rails_apps/echo/app/processors/mof_message_processor.rb
=> Subscribing to TO.ECHO (processed by MOFMessageProcessor)
Message received : Hello World !
```

Retour d'expérience



RBC DEXIA
INVESTOR SERVICES

- Technologie
 - Cadrage de la technologie pour ne pas tomber dans du tout Ruby on Rails
 - Définition et mise en place progressive des patterns
- Organisation
 - Coexistence des développements agiles (Rails) et traditionnels (Java & Progress)
 - Évolution de l'organisation pour supporter les bonnes pratiques inhérentes à la technologie (« release early, release often », environnement tests unitaires, ...)
 - Réunions bimensuelles autour des problèmes rencontrés par les développeurs et la définition de bonnes pratiques (méthodologies, plug-ins, librairies, ...)
 - Estimation des capacités hardware / software en début de mise en place
- Ruby on Rails chez RBC Dexia Investor Services... un succès !
 - Utilisateurs comme développeurs satisfaits (peu d'incidents, réactivité élevée, ...)
 - Plus de 20 applications développées depuis septembre 2006

Le futur

- SOA
 - JRuby pour utiliser le MOF Java
 - Montée en charges des services Java
- Des migrations, pourquoi ?
 - Rails 1.1.6 → 1.2.6 → 2.0
 - WAS à la place d'Apache pour les développements 1^{er} niveau
 - Mongrel à la place d'Apache pour les développements 2^{ème} niveau
 - RadRails 0.7.2 → NetBeans 6.0 ou Aptana Studio 1.0
- Organisation
 - Généralisation dans nos filiales (Singapour, Italie, ...)
 - Test d'intégration
- Continuer à répondre aux besoins des utilisateurs !

Conclusion



RBC DEXIA
INVESTOR SERVICES

- Ruby on Rails
 - Dans l'entreprise ça marche !
 - Ruby on Rails c'est bien mais pas pour tout
 - Ruby on Rails est intégrable à une architecture d'entreprise
- Comment nous avons..
 - Choisi et implémenté Ruby on Rails dans le cas d'applications isolées
 - Abordé l'intégration de Ruby on Rails à notre architecture

Document placé sous licence Creative Commons



RBC DEXIA
INVESTOR SERVICES



▪ Vous êtes libres

- Reproduire, distribuer et communiquer cette présentation au public



▪ Selon les conditions suivantes

▪ Paternité

- Vous devez citer le nom de l'auteur original de la manière indiquée par l'auteur de l'œuvre ou le titulaire des droits qui vous confère cette autorisation (mais pas d'une manière qui suggérerait qu'ils vous soutiennent ou approuvent votre utilisation de l'œuvre)



▪ Pas d'utilisation commerciale

- Vous n'avez pas le droit d'utiliser cette création à des fins commerciales



▪ Pas de modification

- Vous n'avez pas le droit de modifier, de transformer ou d'adapter cette création

▪ Résumé explicatif disponible sur <http://creativecommons.org/licenses/by-nc-nd/2.0/fr/>

▪ Les logos et images utilisés dans cette présentation (celui de Paris on Rails, de RBC Dexia Investor Services, d'IBM, de Java, de Rails, du W3C et de Progress) appartiennent à leurs propriétaires respectifs ; il ne vous est pas permis de les réutiliser indépendamment de ce document, sans autorisation explicite écrite des ayants-droits