Introduction à MOBIDICK™

Modular But Integrated Application Framework

MOBIDICK™ 3 Standard Edition

Community & Enterprise

Principaux avantages et fonctionnalités

GECKO Software
http://consulting.byGecko.com
Email: Info@gecko.fr
Tél: (33) 04 42 26 06 08

Index

Avantages de MOBIDICK™

Les entreprises doivent composer avec de nombreux Frameworks Java changeant régulièrement. Cela occasionne une augmentation régulière des coûts d'intégration quand les besoins demeurent pourtant presque constants pour toutes les applications de gestion (en matière de sécurité, de monitoring, de gestion de contexte, d'interopérabilité etc…)

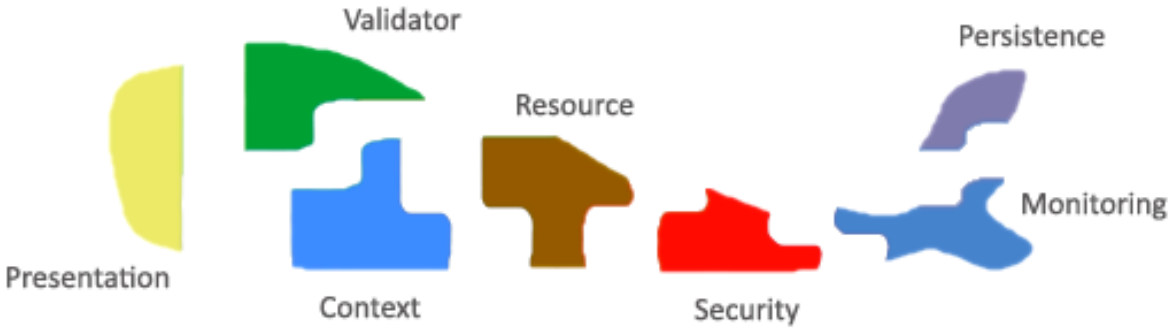
MO**dular **B**ut **I**ntegrate**D** appl**IC**ation framewor**K

MOBIDICK™ adresse principalement les besoins d'applications de gestion (de type « branch office » tels les banques, les assurances, le secteur des services). Le scope de fonctionnalités est donc limité à ce cadre mais adresse en revanche les vrais besoins de ce type d'entreprises dont l'inventaire est garanti par l'expérience d'experts et de contributeurs impliqués au sein d'un projet global OpenSource.

MOBIDICK™ est « ***M**odulaire* » pour une meilleure aptitude à l'évolution et pour une meilleure intégration avec les Framework déjà en place au sein des entreprises. Il est en effet possible à toute entreprise d'utiliser tout ou partie des modules **MOBIDICK™**.

MOBIDICK™ est « ***I**ntégré* », c'est-à-dire que toutes les fonctions présentées ci-après sont supportées en totalité et simultanément de telle manière qu'aucun ajout de Framework supplémentaire ne soit nécessaire au client final.

Les besoins des entreprises changent bien moins rapidement que les Frameworks Java, c'est pourquoi **MOBIDICK™** travaille à rendre possible la génération MDA pour chacun de ses modules.



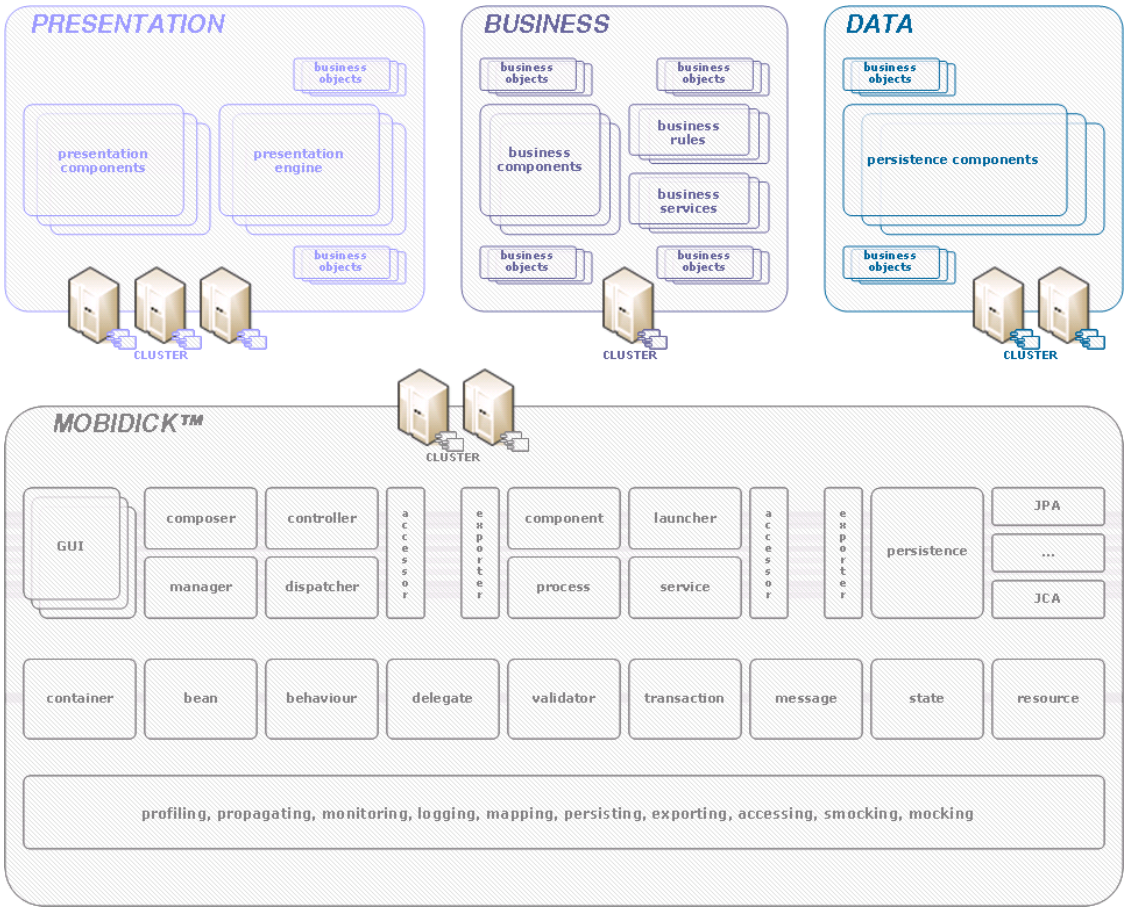
Fonctionnalités de MOBIDICK™

MOBIDICK™ concentre l'ensemble des problématiques techniques relatives aux architectures Java à connecter tant avec les systèmes modernes qu'avec les systèmes existants dit « Legacy ».

Prenez connaissance ci-après des fonctionnalités clés de **MOBIDICK™** (bientôt disponibles en version OpenSource *, déjà disponibles en version OpenSource *, et déjà disponibles en version entreprise), en partant des sujets les plus globaux pour parvenir aux points précis de *support d'une infrastructure*, d'un *service de données*, d'un *type d'IHM* etc… Il vous sera ainsi possible d'analyser notre solution au vu des problématiques qui vous sont propres.

Organisation d'une Application Métier

Sujet Global



** : contrats de services seulement. Veuillez nous contacter pour plus de détails.*

Fonctionnalités Supportées

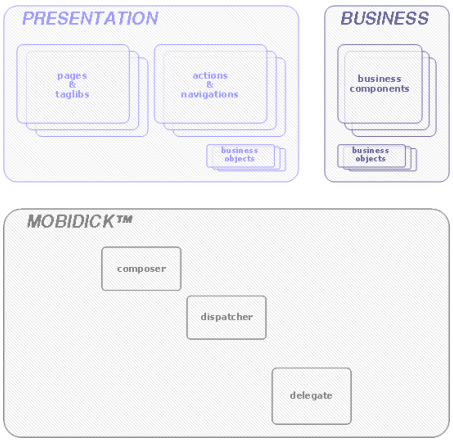
Factorisation en couches	- Identification des différentes couches de présentation, de règles métier, d'accès au données, d'accès aux systèmes connexes…	
Découpage logique des composants	- Définition d'interfaces techniques et de contrats de service pour assurer la délégation	
Autonomie des couches	- Urbanisation des composants sur différentes couches dans l'optique d'assurer une capacité totale d'évolution et cela sans effet de bord - Couches non contigües	
Découplage des composants applicatifs	- Implémentation des composants applicatifs respectant les patterns JEE et cela sans adhérence et sans perméabilité les uns par rapport aux autres - Isolation totale grâce à des mécanismes de mapping objet-objet et objet-relationnel - Anticipation d'API : test unitaire - Capacité à remplacer un type de service par un autre sans impact sur les développements avals et amonts - Capacité à réutiliser un service existant	
Banalisation des accès	- Support de composants techniques d'accès aux données (SGBD, MainFrame, GED, Mail / Fax, EAI, Ressources)	
Cadrage des composants	- Réduit les dépendances et le nombre de classes, et cadre les développements de composants métiers d'interface et de délégation (génériques, annotations, convention de nommage, injections des dépendances, objets bouchons)	
Répartition de charges des couches front et back office	- L'autonomie des couches et le découplage total des composants rendent possible la séparation physique des livrables sur différents serveurs	
Délégation entre couches par proxy	- Complexité des services masquée pour faciliter le travail du développeur	
Accès remoting et/ou local	- La définition de l'accès à un service est externalisée dans la configuration XML rendant transparent l'appel local ou distant pour le développeur	
Centralisation des déploiements	- L'externalisation, la factorisation et le découpage physique rendent possible la centralisation des livrables du framework et de certaines parties des composants applicatifs (contrats de service, objets de transfert)	
Découpage physique des livrables	- Découpage physique applicatif adapté à la modularité logique définit par l'urbanisation des composants - Séparation entre les livrables techniques et applicatifs	
Externalisation des composants	- Externalisation des composants techniques et applicatifs pour faciliter les phases d'intégration et de mise en production	

Factorisation des configurations	- Factorisation des configurations (properties, cache, datasource) pour faciliter les phases d'intégration et de mise en production	

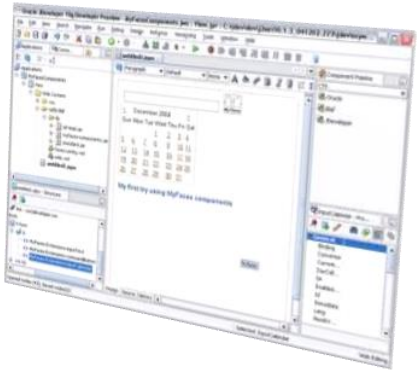
Interface Multi-Utilisateur

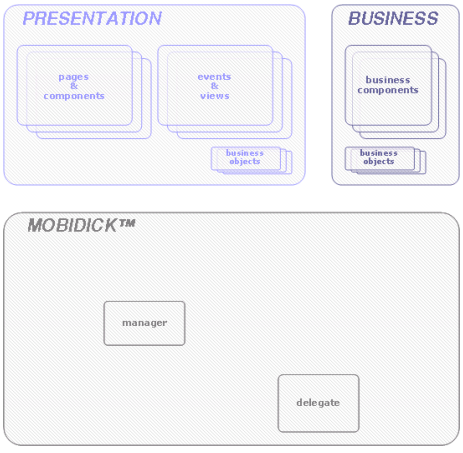
Sujet Global

Page Mode (MVC ZK)

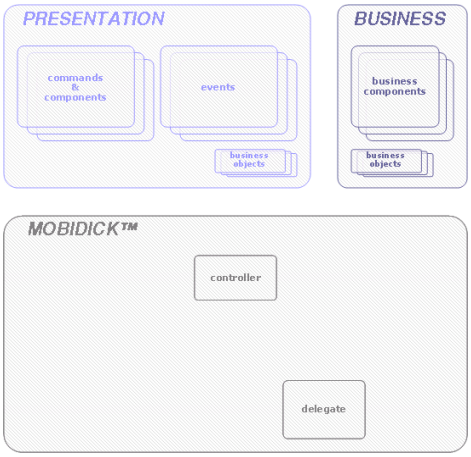


Component Mode (JSF)





Graphical Mode (FLEX)



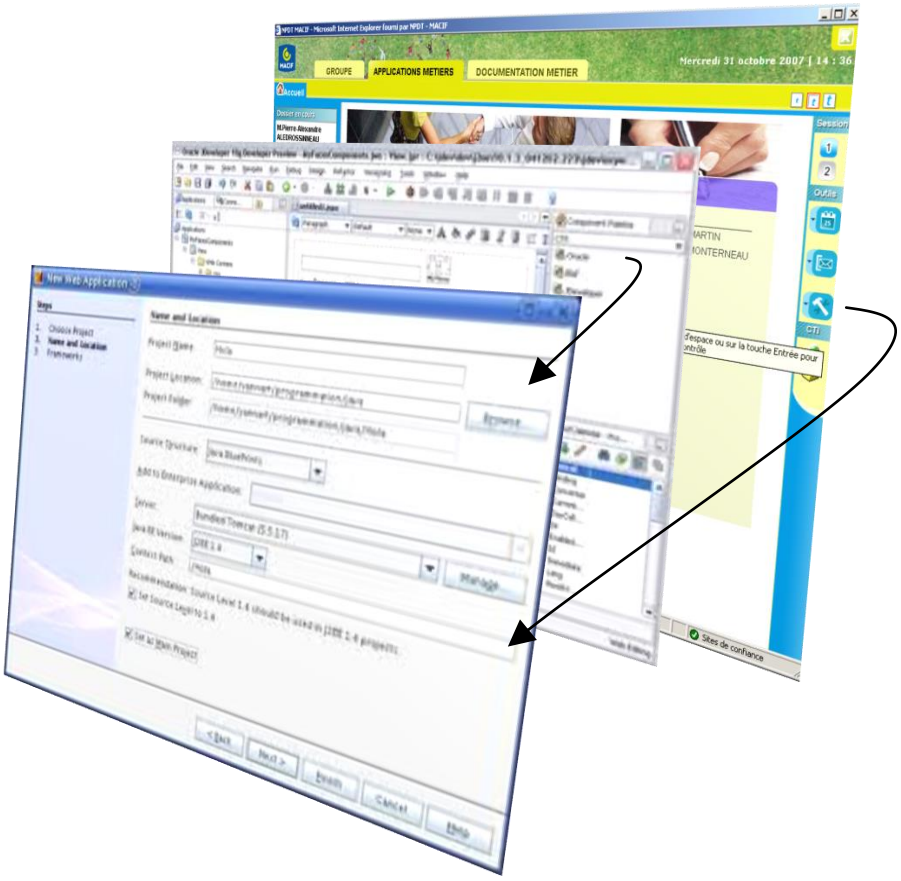
Fonctionnalités Supportées

Couche de présentation en mode Pages, Taglibs Ajax (MVC, ZK)	- Moteur MVC2 - Prise en charge de pages JSP agrémentées de taglibs ZK - Composants techniques pour l'écriture de contrôleurs applicatifs - Navigation Spring WebFlow	
Couche de présentation en mode Composants (JSF)	- Moteur JSF standard - Prise en charge de pages JSP agrémentées de taglibs JSF standard - Navigation JSF standard	
Couche de présentation en mode Graphique (FLEX)	- Prise en charge de Flex et Adobe AIR : connexion facilité avec les services existants Java en mode remoting et web messaging via BlazeDS - Haute performance de transfert de données pour des applications plus réactives - Push serveur sur HTTP standard disponible	
Intégration de couches propriétaires de présentation JSF	- Intégration des concepts de Mobidick au sein d'une couche propriétaire JSF : gestion de contexte, règles métier, débranchement entre applications, gestion des libellés, monitoring et sécurité - Mise à disposition de composants taglibs Mobidick compatibles JSF - Mise à disposition de composants techniques Java pour la création de contrôleurs JSF	
Intégration de couches propriétaires de présentation FLEX	- Intégration des concepts de Mobidick au sein d'une couche propriétaire FLEX: gestion de contexte, règles métier, débranchement entre applications, gestion des libellés, monitoring et sécurité - Mise à disposition de composants taglibs Mobidick compatibles FLEX - Mise à disposition de composants techniques en ActionScript pour la création de contrôleurs FLEX	

Intégration de couches propriétaires de présentation MVC	- Intégration des concepts de Mobidick au sein d'une couche propriétaire MVC: gestion de contexte, règles métier, débranchement entre applications, gestion des libellés, monitoring et sécurité - Mise à disposition de composants taglibs Mobidick compatibles MVC - Mise à disposition de composants techniques en Java pour la création de contrôleurs MVC	

Navigation visuelle entre Applications

Sujet Global



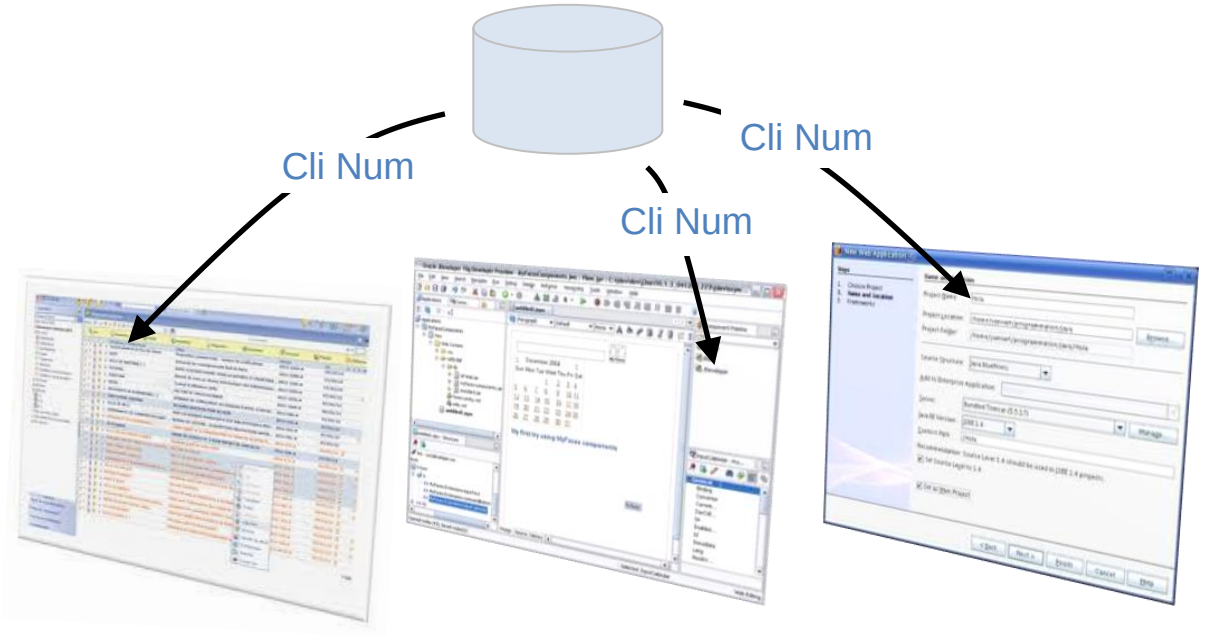
Fonctionnalités Supportées

Via Portail JSR 168	- Navigation entre applications (portlets) via le portail de type JSR 168 - Passage d'informations via le contexte	
Via Portail JSR 268	- Navigation entre applications (portlets) via le portail de type JSR 268 - Passage d'informations via le contexte	

Hors Portail Intra Applications	- Navigation intra application sans le portail - Facilite les développements et les tests d'intégration	
Hors Portail Inter Applications	- Navigation inter applications sans le portail - Facilite les développements et les tests d'intégration	
Compatibilité des deux modes (avec et sans Portail)	- Possibilité de déployer une application avec ou sans portail sans influer sur le code applicatif, la configuration et la navigation	

Gestion de Contexte

Sujet Global



Fonctionnalités Supportées

Niveaux de contexte	- Gestion du contexte utilisateur par niveaux (Poste, Session métier, Processus, Application) - Niveaux configurables - Blob supportés - Accès au contexte depuis toutes les couches applicatives - Contexte en base de données pour faciliter l'intégration avec d'autres systèmes - Accès contrôlé sur les différents niveaux	
Via Portail JSR 168	- Intégration des données contextuelles du portail JSR 168 au sein du contexte Mobidick	

Via Portail JSR 268	- Intégration des données contextuelles du portail JSR 268 au sein du contexte Mobidick	
Synchronisation de contextes externes	- Possibilité de synchronisation du contexte Mobidick avec d'autres systèmes - Synchronisation distante ou local par WebService, HTTP	
Saisie Portail JSR 168	- Mémorisation des saisies utilisateurs au sein des portlets - Passage de données saisies dans le contexte de façon automatique	
Saisie Portail JSR 268	- Mémorisation des saisies utilisateurs au sein des portlets - Passage de données saisies dans le contexte de façon automatique	
Pré contrôle taille de contexte	- Contrôle de la taille du contexte avant la mise à jour en base de données - Taille des contextes optimisée par niveau	
Haute disponibilité	- En cas de coupure du système, l'ensemble des informations du contexte de tous les utilisateurs est présent (cela rend possible les mécanismes de Fail over et ainsi la reprise des sessions utilisateurs)	

Contrôle des Règles de Gestion

Sujet Global

Fonctionnalités Supportées

Isolation des règles	- Mise en place du pattern Validator pour permettre l'isolation des règles métier dans des objets Java simple - Réutilisation possible des Validators sur différents types de composants (services, batch-process) - L'isolation de ces règles dans des objets métiers dédiés simplifie l'écriture des services qui ne contiennent alors que les enchaînements techniques d'appels aux données	
Isolation des messages	- Mécanismes simples de récupération de messages par clés pour l'affichage à l'utilisateur - Propagation des messages depuis les couches de services jusqu'aux couches de Présentation	
Criticité des messages	- Structure avancée de message pour définir sa criticité, son origine (technique ou fonctionnelle), un libellé technique, un libellé fonctionnel et une clé unique pour identifier cette structure	

Gestion des Libellés et des « Collections »

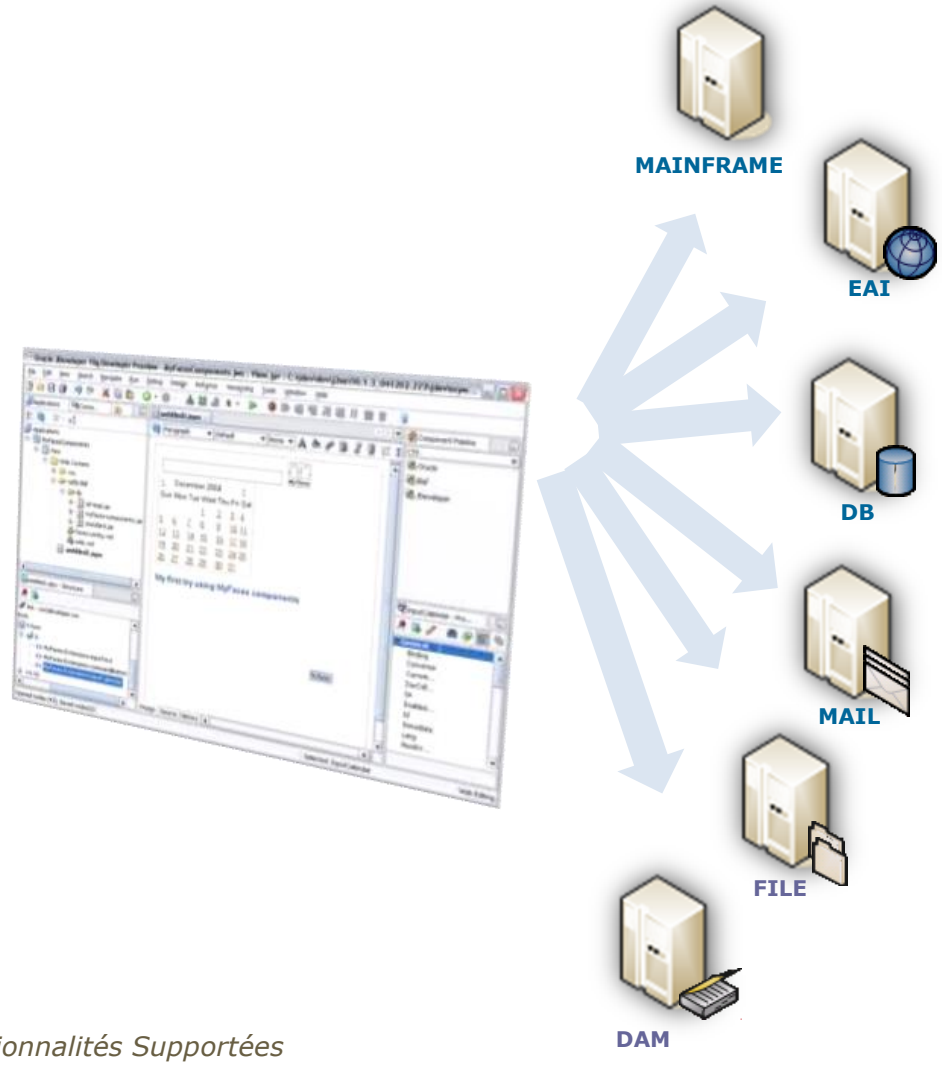
Sujet Global

Fonctionnalités Supportées

Externalisation	- Services dédiés à la récupération des libellés depuis une clé donnée	
Uniformisation	- Services uniformisés et accessibles depuis toutes les couches	
Entreprise	- Référentiel de données (clés / valeurs) spécifiques de l'entreprise bénéficiant des mécanismes d'accès de Mobidick	
Technical	- Référentiel de données (clés / valeurs) spécifiques de Mobidick	
Communautaire	- Référentiel de données (clés / valeurs) communautaires de Mobidick (pays, départements, régions, codes postaux…)	
Internationalisation	- Internationalisation des libellés applicatifs selon les normes i18	

Interopérabilité avec les Systèmes Externes

Sujet Global



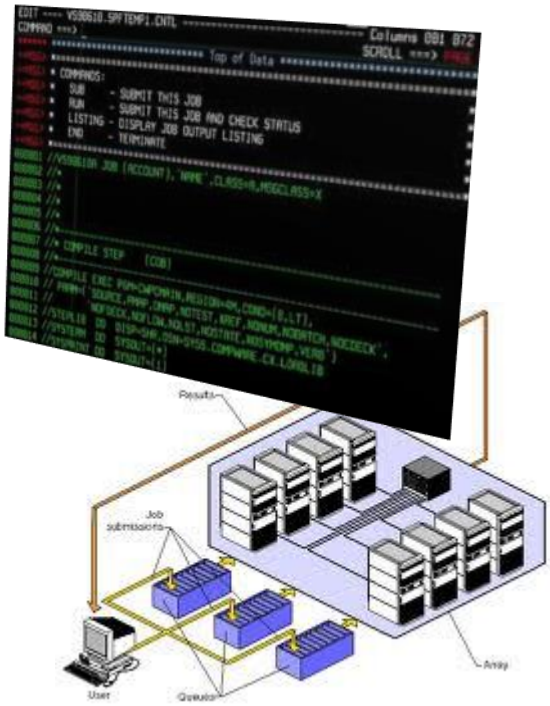
Fonctionnalités Supportées

Accès Orchestrateur (EAI)	- Support de communication en mode SOAP / HTTP / RMI	
Accès SGBD	- Mapping de tous les types de champs ANSI - Support du « caching » d'objets métiers (ehcache) - Fonctions avancées d'accès aux données (criteria)	
Accès Mainframe	- Prise en charge des accès via solution propriétaire Java d'accès au mainframe - Prise en charge des accès IMS, CICS, LU0, LU2 via JCOB (JCA)	
Accès Mail / Fax	- Connexion aux boîtes email (envoi / réception) - Connexion aux systèmes de Fax (envoi / réception)	
Accès GED	- Lecture / écriture de documents - Accès au workflow métier (activités, bannettes)	

Contexte transactionnel	- Support de l'intégrité transactionnelle (commit à deux phases)	
Accès Ressources	- Banalisation des accès aux ressources (via classpath, web, système, configuration, etc...) - Uniformisation de la manipulation, quel que soit le type de ressources (fichier, flux, binaire, etc...)	

Procédures Batch

Sujet Global

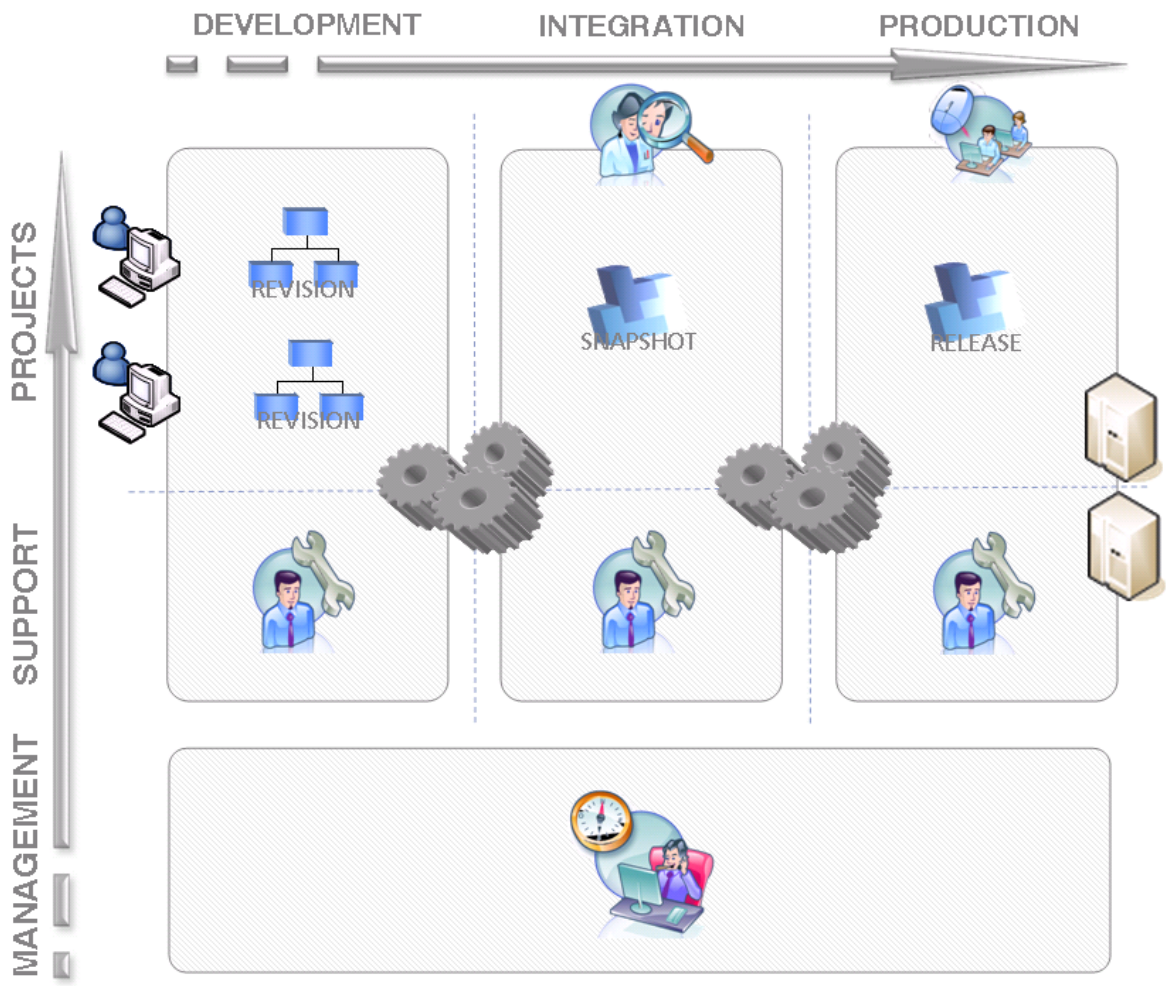


Fonctionnalités Supportées

Relance Job	- Gestion automatique ou manuelle de relance de job après échec	
Chunk processing	- Gestion de commit par période (chunk processing)	
Job par étapes	- Définition séquentielle des jobs par étapes	
Skip / rollback	- Traitement partiel des données (par exemple skip record i.e. sur rollback)	
Transaction	- Gestion transactionnelle globale des jobs	
Planification	- Exécution planifiée des jobs	
Non séquentiel	- Traitement non-séquentiel par étapes (conditionnal branching)	
Pause / reprise	- Gestion de pause / reprise sur l'exécution des jobs	
Reporting	- Reporting des tâches (compteurs, codes retours)	

Intégration Continue

Sujet Global



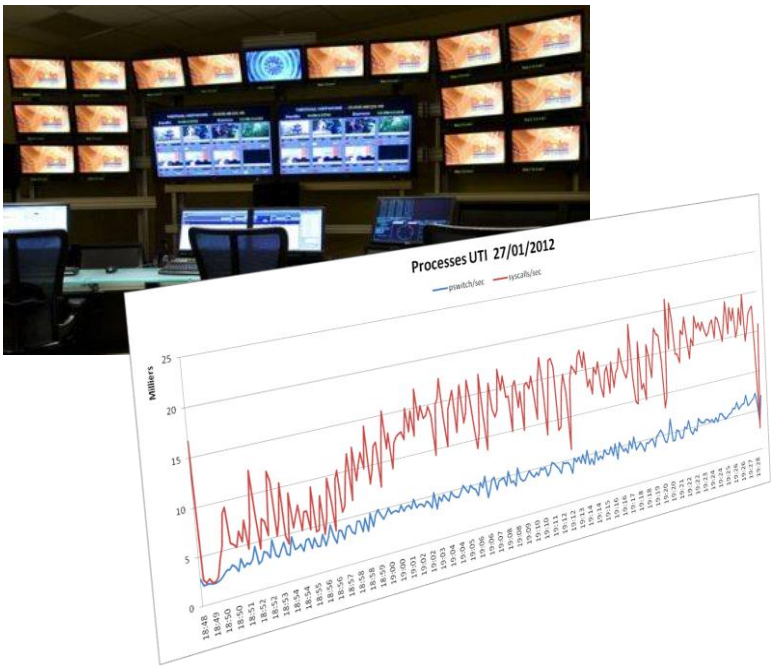
Fonctionnalités Supportées

Automatisation	- Automatisation des livraisons, distributions et publications - Accélération du cycle de production des livrables - Accélération des processus de mise en intégration / production - Utilisation de descripteur de publication pour les mises en intégration / production - Lancement automatisé des tests unitaires et d'intégration	
Référentiel sources	- Communication avec des référentiels de sources (CVS, SVN) - Suivi des modifications des applications	

Dépôts librairies	- Référentiels des livrables techniques et applicatifs (centralisation des livrables) - Mise à disposition rapide des composants aux développeur - Définition de modèles de livrables (ear, war, jar etc...)	
Tests / Intégration	- Composants techniques pour l'écriture de tests unitaires et d'intégration: bouchonnage de composant, validation des navigations, règles de gestion, contrôles de surface - Rapport détaillé des résultats d'exécution des tests - Estimation de la couverture des tests sur le code applicatif	
Configuration externalisée	- Externalisation des propriétés et des dépendances pour les applicatifs et Mobidick - Centralisation des configurations pour faciliter les déploiements - Facilite la maintenance des configurations	

Performance et Monitoring

Sujet Global



Fonctionnalités Supportées

Application Bench	- Mise à disposition d'une application déployable et pré configurée - Permet la validation de l'ensemble des chaînes de liaison techniques - Permet la vérification de la bonne répartition de charge selon les délégations entre couches (accès à un composant applicatif en mode remoting et/ou local)	
Métrologie / Surveillance	- Module graphique de surveillance des applications - Module graphique de métrologie des différents composants par chaines de liaison techniques	
Monitoring JMX	- Utilisation de la norme JMX pour surveiller chacun des composants applicatifs sur les toutes les couches	

Sécurité (SSO, ACL)

Sujet Global

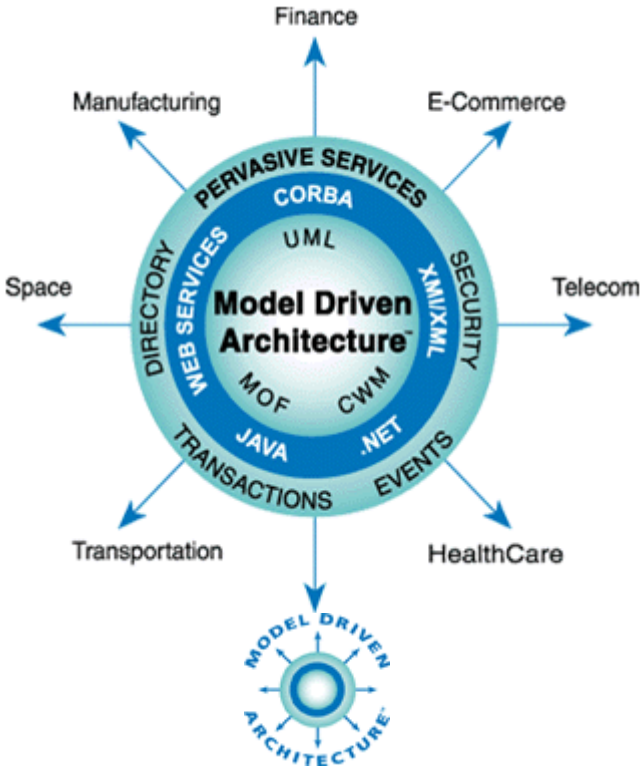
Fonctionnalités Supportées

SSO	- Support d'une affinité cluster - SSO cross-plateform - Gestion (non intrusive) de l'authentification J2EE	
Propagation	- Propagation de l'authentification vers des systèmes connexes (HOST, SGBD, Ldap, Xml, Properties) - Prise en compte de profils d'authentification externes - Via la gestion de contexte Mobidick	
Traçabilité	- Suivi configurable des actions des utilisateurs	
Utilisateur banalisé	- Possibilité de configurer des utilisateurs banalisés pour l'accès à différentes ressources (mainframe par exemple)	
Habilitations fines	- Possibilité de configurer des listes de contrôle d'accès (ACL) au niveau des classes mais aussi des méthodes	
Administration	- Fonctionnalités d'administration externalisée de la sécurité et des habilitations fines	
Haute disponibilité	- Les informations externalisées de la sécurité sont sauvegardées dans une base de données permettant la reprise du système sans perte de données (fail over)	

Fonctions connexes	- Mise à disposition de taglibs de sécurité facilitant le développement des interfaces utilisateurs - Mécanisme de "SwitchUser" pour changer d'utilisateur en cours de session - Mécanisme de gestion de session concurrente - "Mécanisme "RememberMe" - Mécanisme de filtrage des informations en fonction du profil de l'utilisateur - Fonctionnalité de cache des informations de sécurité pour optimiser les temps de réponse	

Approche MDA du besoin

Sujet Global



Fonctionnalités Supportées

Par rétro-ingénierie	- Générer du code à partir de la rétro-ingénierie (approche dite "bottom-up") d'une source de données telle que schéma de base de données, structure de message mainframe…	

Par modélisation	- Générer du code à partir de la modélisation d'une architecture logicielle telle que diagramme de classe, diagramme de cas d'utilisation, modèle relationnel…	
Par conception	- Générer du code à partir de la conception (approche dite "top-down") d'une interface utilisateur telle que composition d'écran, composant graphique…	

Prérequis de MOBIDICK™

Ci-dessous, les prérequis à l'utilisation de **MOBIDICK™** :

- ✓ JVM 6+
- ✓ Z/Os – AIX – Linux – Windows