Introducing MOBIDICK™

Modular But Integrated Application Framework

MOBIDICK™ 3 Standard Edition
Community & Enterprise
Key Benefits and Features

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

Contents

MOBIDICK™ Key Benefits

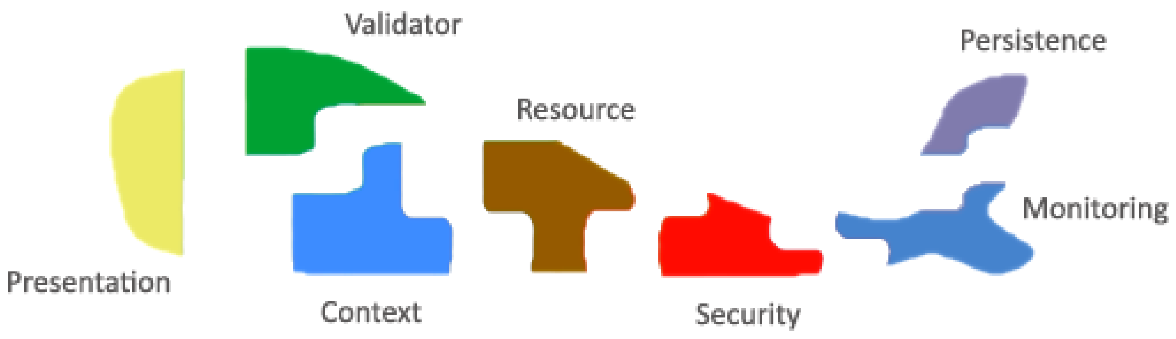
Companies have to deal with a lot of different Java Frameworks which change frequently causing integration's cost to increase over and over although needs are very similar for all business application (considering security, monitoring, context management, interoperability, etc..).

MOBIDICK™ is dedicated to business application mainly (branch office like, as banks, insurances, services) so the scope of features is limited but match the real needs who are granted by the professional experience of the project contributors and experts.

MOBIDICK™ is « ***MO****dular* » for best progressive evolution and for easier integration with companies' specifics framework. Any company may use one or more **MOBIDICK™**'s modules.

MOBIDICK™ is « ***I****ntegrate****D*** », all the features shown in the appendix are supported so that the Company does not have to add any other Java Framework.

Companies' needs change much slower than Java framework's do, so **MOBIDICK™** will have to allow MDA generation of these needs on selected Java modules.



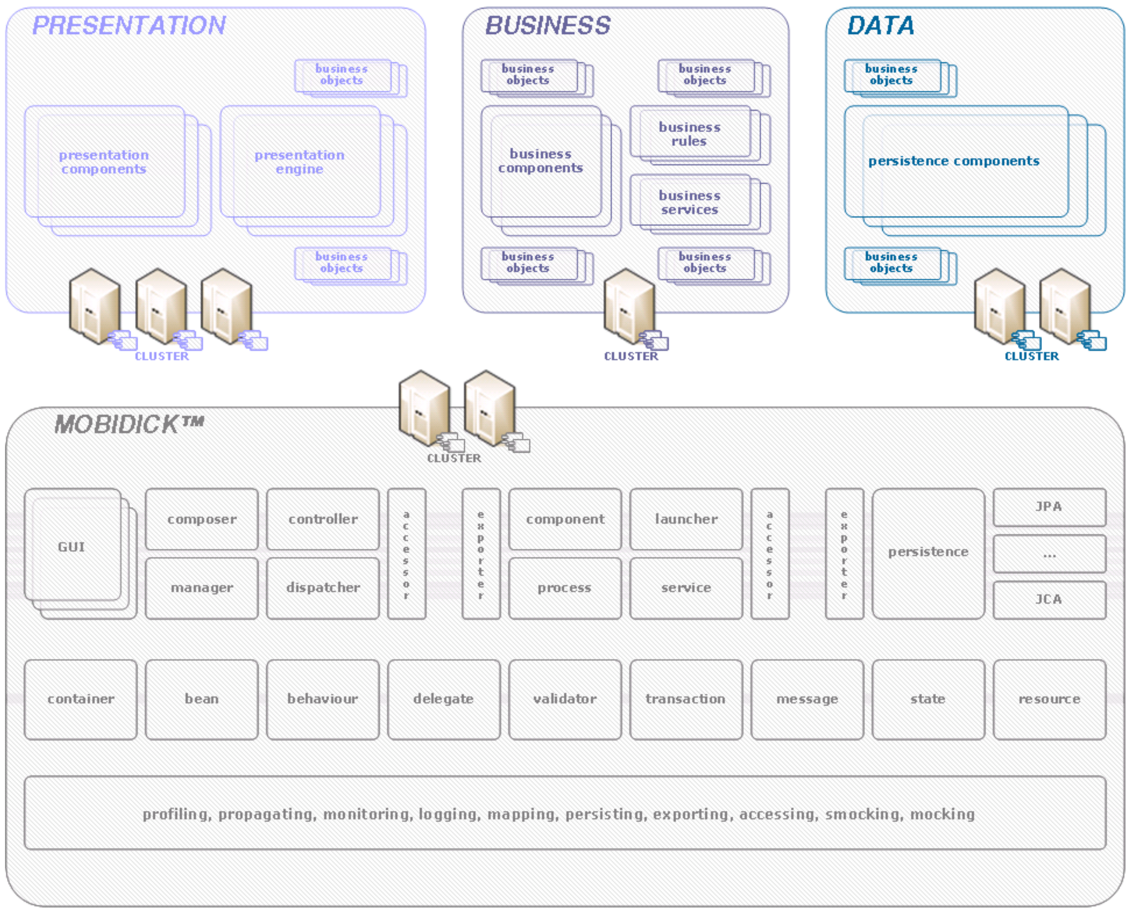
MOBIDICK™ Key Features

MOBIDICK™ is a concentrate of technical concerns gathered from real Java architectures connecting modern layers and legacy systems together.

Take a quick tour of **MOBIDICK™**'s key features (soon available in community distribution*, already available in community distribution*, and already available in enterprise distribution), from the general purpose to the native *support of infrastructure*, *data services*, *particular GUI* etc… to allow you to focus on your business issues.

Design of Business Applications

Global Scope



**: services contracts only. Please contact us for more details.*

Supported Features

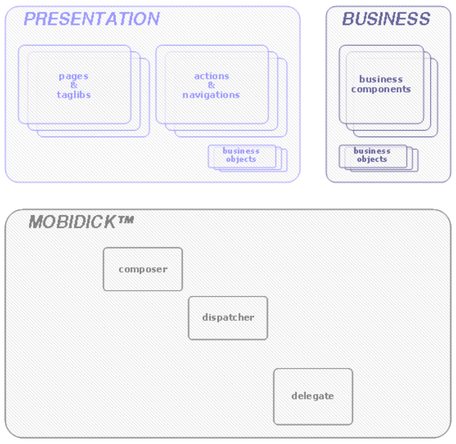
Factorisation layered	- Identification of different layers like Presentation layer, Business rules layer, Data access layer and subsidiaries systems layer	🟩
Logical organization of components	- Definition of technical interfaces and business service api's to ensure delegation	🟩
Autonomy layers	- Urbanization of components on different layers in order to ensure a total capacity of evolution and without side effect - Non-contiguous layers	🟩
Total decoupling of applicative components	- Use of JEE patterns to implement application's components without adhesion and permeability relative to each other - Total Isolation through object-object and object-relationnal mapping - Anticipation of API: unit testing - Ability to replace one type of service by another with no impact on upstream and downstream developments - Ability to reuse an existing service	🟩
Trivilization of all access	- Support for data access technical component (DBMS, Mainframe, EDM, Mail / Fax, EAI, Resources)	🟩
Directive components	- Reduces dependency and the number of classes, and drive development of business components interface and delegation (generics, annotations, naming convention over configuration, dependency injection, mock objects)	🟩
Front and back office layers load balancing	- Autonomy of the layers and the decoupling of components make possible the physical separation of the deliverables on different servers	🟩
Delegation between layers by proxy	- Hidden complexity of services to facilitate software development	🟩
Remoting and/or local access	- The definition of a service access is outsourced in the XML configuration making the local or remote call transparent for the developper	🟩
Centralized deployment	- The outsourcing, factoring and physics organization make it possible to centralize framework deliverables and parts of the application's components (service contracts, transfer objects)	🟩
Cutting physical deliverables	- Applicative physical cutting adapted to the logic modularity defined by the component's urbanization - Separation between technicals and applicatives librairies	🟩
Outsourcing of components	- Outsourcing of technical and applicative components to facilitate the integration and production phases	🟩

Factorization of configurations	- Factorization of configurations (properties, cache, datasource) to facilitate the integration and production phases	🟩

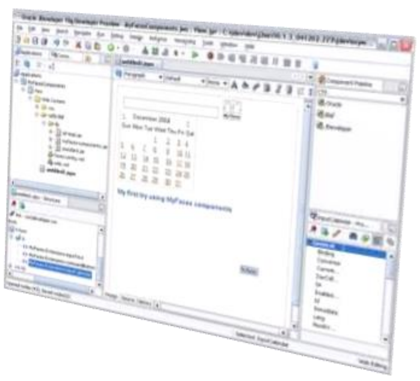
Multi-User Interface aspects

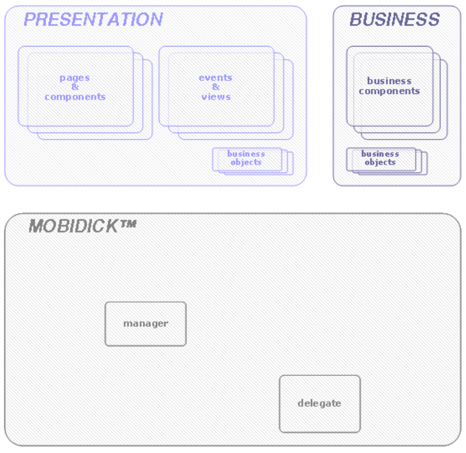
Global Scope

Page Mode (MVC ZK)

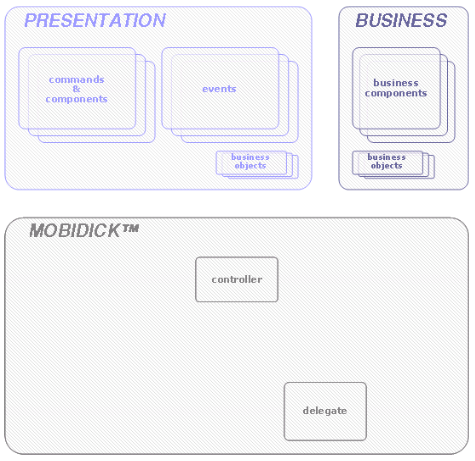


Component Mode (JSF)





Graphical Mode (FLEX)



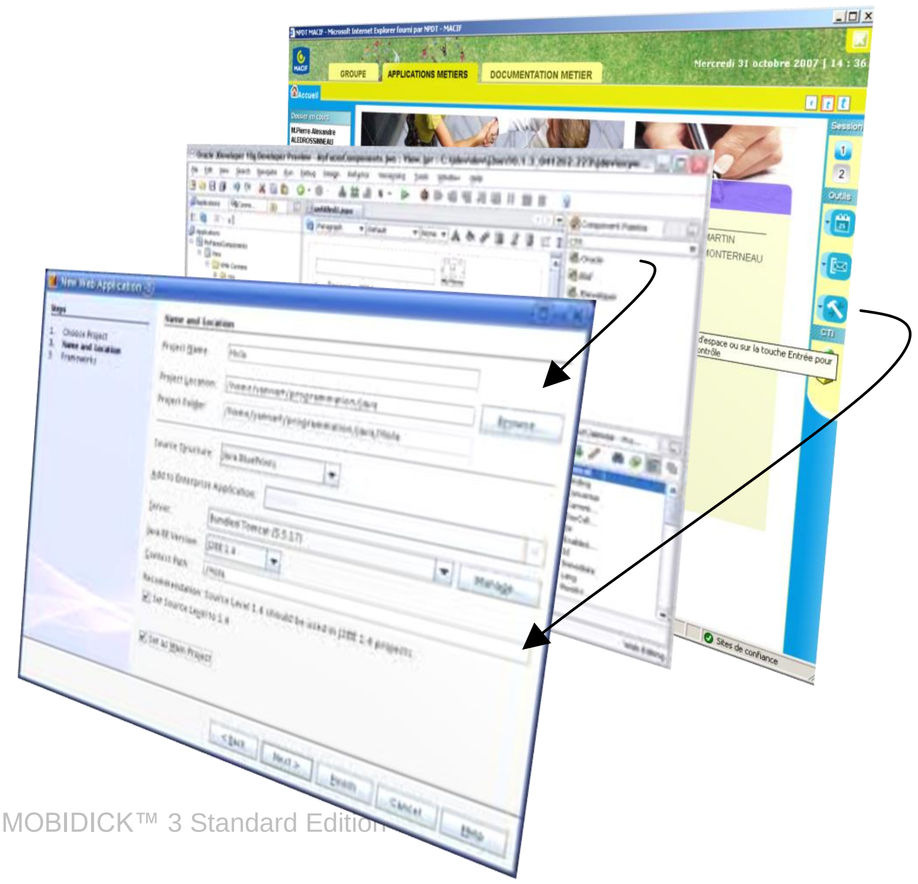
Supported Features

Presentation layer in Page mode, Taglibs Ajax (MVC, ZK)	- MVC2 engine - Support for JSPs decorated with ZK taglibs - Technical components for writing applications controller - Spring WebFlow Navigation	🟩
Presentation layer in Component mode (JSF)	- Standard JSF Engine - Support for JSPs decorated with standard JSF taglibs - JSF standard navigation	🟨
Presentation layer in Graphical mode (FLEX)	- Support for Adobe AIR and Flex: easy connection with existing Java services by remoting and web messaging mode via BlazeDS - High performance data transfer for more responsive applications - Server push over standard HTTP available	🟨
Integration of JSF presentation layers owners	- Integration of Mobidick's concepts within a JSF layer owner: context management, business rules, dispatching between applications, labels, monitoring and security - Mobidick compatible JSF taglibs available - Technical Java components to create JSF controllers available	🟦

Integration of FLEX presentation layers owners	- Integration of Mobidick's concepts within a FLEX layer owner: context management, business rules, dispatching between applications, labels, monitoring and security - Mobidick compatible FLEX taglibs available - Technical ActionScript components to create FLEX controllers available	🟦
Integration of MVC presentation layers owners	- Integration of Mobidick's concepts within a MVC layer owner: context management, business rules, dispatching between applications, labels, monitoring and security - Mobidick compatible MVC taglibs available - Technical Java components to create MVC controllers available	🟦

Visual Dispatching between Applications

Global Scope

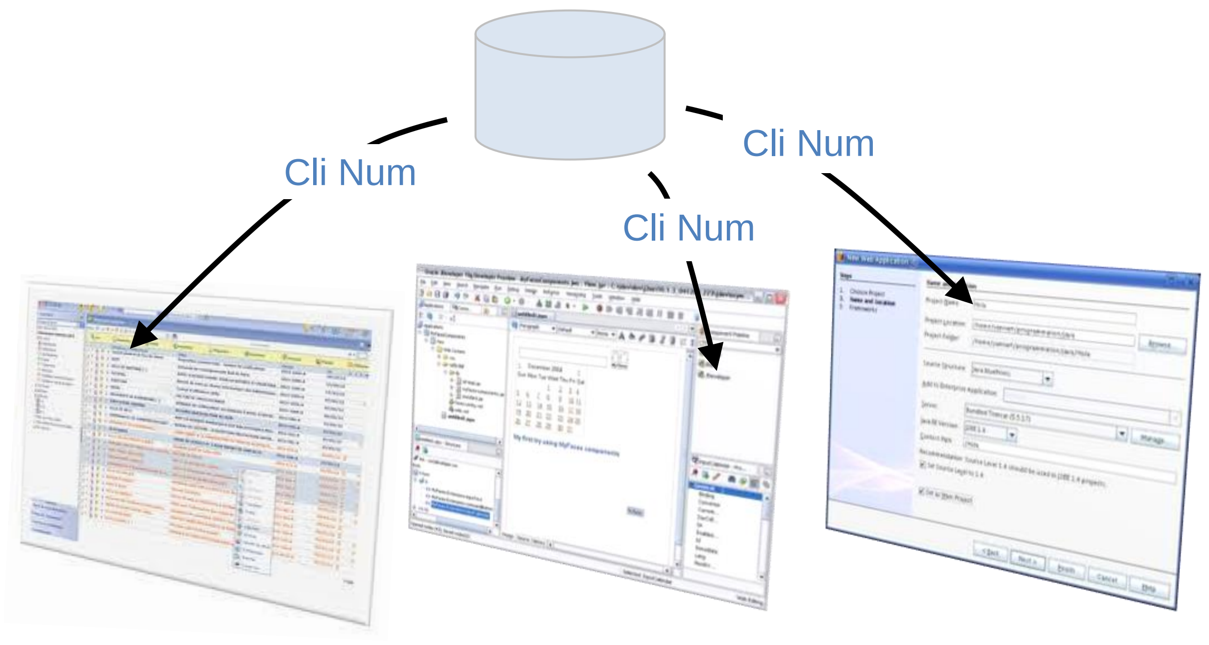


Supported Features

Through JSR 168 Portal	- Navigation between applications (portlets) via JSR 168 portal - Transfer of information via the context	🟩
Through JSR 268 Portal	- Navigation between applications (portlets) via JSR 268 portal - Transfer of information via the context	🟨
Off Portal Intra Applications	- Navigation within application without portal - Facilitates the development and integration tests	🟩
Off Portal Inter Applications	- Navigation between applications without portal - Facilitates the development and integration tests	🟩
Compatibility of the two modes (with or without Portal)	- Ability to deploy an application with or without portal without affecting the application code, configuration and navigation	🟩

Context Management

Global Scope



Supported Features

Context levels	- Management of user's context by levels (Computer, Business session, Process, Application) - Configurable Levels - Blob supported - Access to the context from all application layers - Context saved in database for easy integration with other systems - Controlled access to the various levels	🟩
Through JSR 168 Portal	- Integration of portal JSR 168 contextual information within the Mobidick context	🟩
Through JSR 268 Portal	- Integration of portal JSR 268 contextual information within the Mobidick context	🟨
External context synchronization	- Ability to synchronize Mobidick context with other systems - Remote or local synchronization by WebService, HTTP	🟦
Inputs through JSR 168 Portal	- Users inputs memorisation within portlets - Automatically transfer input data to the context	🟦
Inputs through JSR 268 Portal	- Users inputs memorisation within portlets - Automatically transfer input data to the context	🟦
Pre control of context size	- Control the size of the context before database update - Size optimized contexts by levels	🟩
High Availability	- If the system is down, all information in the context of all users is present (this makes fail over mechanisms possible and though the recovery of user sessions)	🟩

Business Rules Control

Global Scope

Supported Features

Rules isolation	- Setting up the Validator pattern to allow isolation of business rules in simple Java objects - Possible reuse of Validators on different types of components (services, batch-process) - Isolation of these rules in dedicated business objects simplifies writing services that do not contain sequences while technical calls to data	🟩
Message isolation	- Simple key recovery system to display messages to the use - Propagation of messages from the service layers to presentation layers	🟩
Criticality of messages	- Advanced message structure to determine its criticality, its origin (technical or functional), a technical label, a functional label and a unique key to identify this structure	🟨

Labels and Collections Management

Global Scope

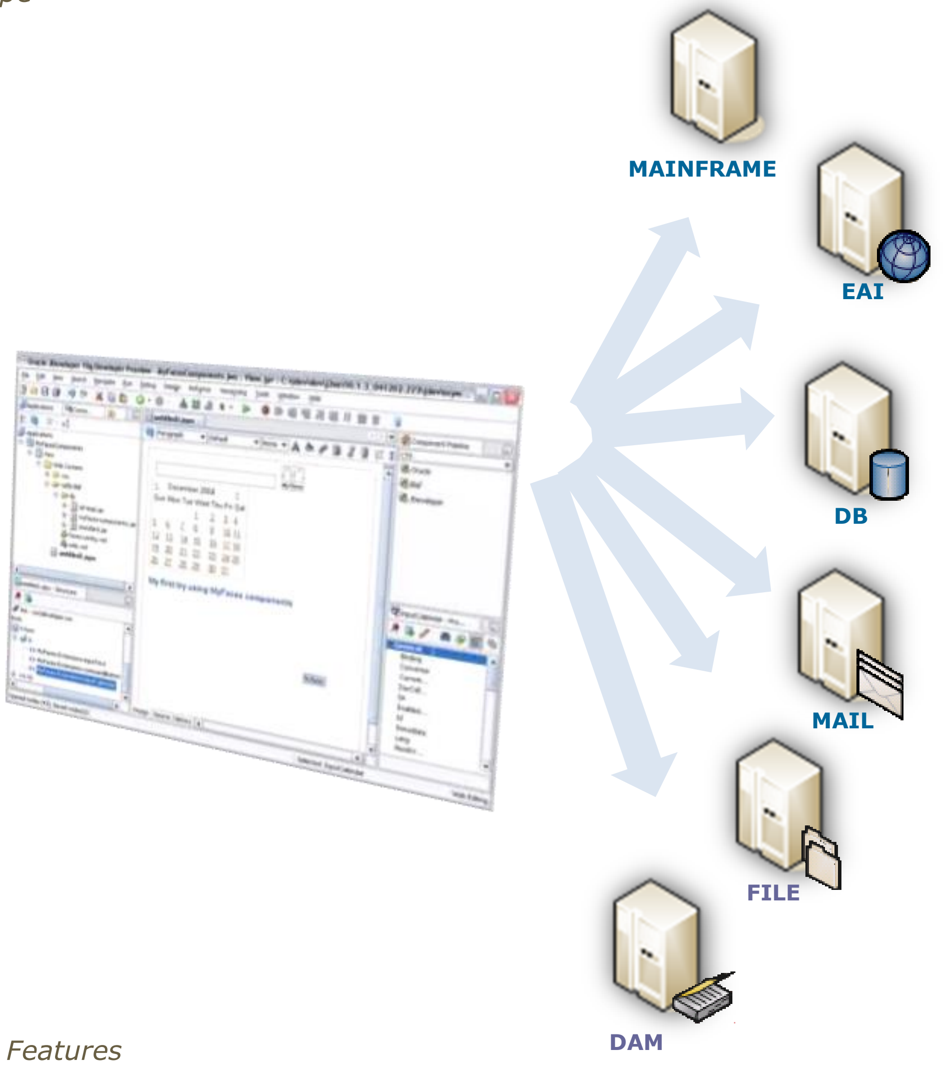
Countries	
United States	us
Unknown	ip
Great Britain	gb
France	fr
Australia	au
European country	eu
Belgium	be
China	cn
Netherlands	nl
Greece	gr
Germany	de
Canada	ca
Turkey	tr
Denmark	dk
New Zealand	nz
Spain	es
Italy	it
South Africa	za
Argentina	ar
Portugal	pt
Bosnia-Herzegovina	ba
Russian Federation	ru
Norway	no
Mexico	mx
Romania	ro

Supported Features

Outsourcing	- Dedicated services for labels recovery from a given key	🟩
Uniformity	- Uniform and accessible services from all layers	🟩
Enterprise	- Enterprise specific data reference (keys / values) with Mobidick access mechanisms	🟨
Technical	- Mobidick specific data reference (keys / values)	🟩
Community	- Community data reference (keys / values) with Mobidick access mechanisms	🟨
Internationalization	- Label internationalization based on i18 rules	🟩

Interoperable subsidiaries Systems

Global Scope



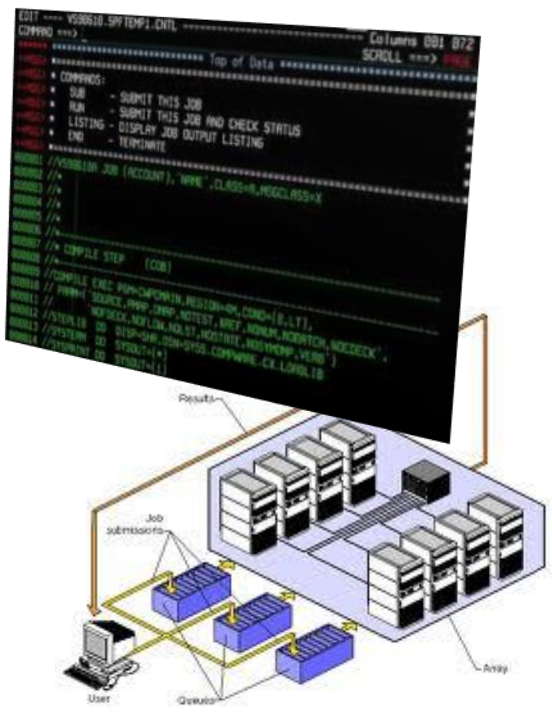
Supported Features

Orchestrator Access (EAI)	- Communication support for SOAP / HTTP / RMI modes	🟩
DBMS access	- Mapping of all types of ANSI fields - Support for "caching" business objects (ehcache) - Advanced data access functions (criteria)	🟩
Mainframe access	- Supports access via proprietary Java solution for access to mainframe - Supports access IMS, CICS, LU0, LU2 via JCOB (JCA)	🟦
Mail / Fax access	- Connection to mail boxes (send / receive) - Connection to fax systems (send / receive)	🟩
EDM access	- Read / write documents - Access to business workflow (activities, baskets)	🟨
Transactionnal context	- Supports transactional integrity (two phase commit)	🟩

Resources access	- Standard access to resources (via classpath, web, system, configuration, etc ...) - Standardization of the handling, whatever the type of resource (file, stream, binary, etc ...)	🟩

Batch Process

Global Scope



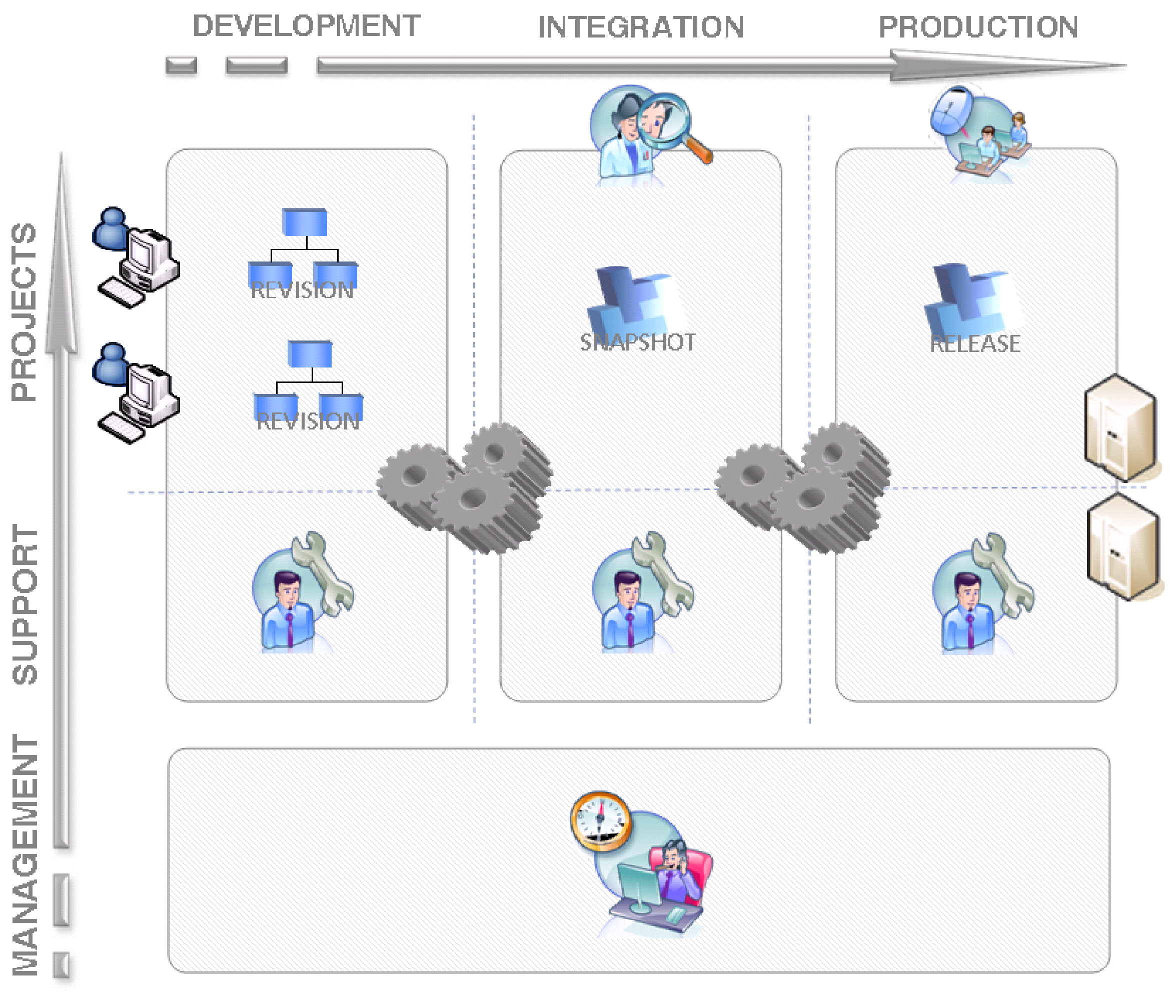
```
EDIT ---- VS98610.SPFTEMP1.CNTL ------------------------- Columns 001 072
COMMAND ===>                                              SCROLL ===> PAGE
****** ******************** Top of Data ******************************
       * COMMANDS:
       *   SUB     - SUBMIT THIS JOB
       *   RUN     - SUBMIT THIS JOB AND CHECK STATUS
       *   LISTING - DISPLAY JOB OUTPUT LISTING
       *   END     - TERMINATE
       ***************************************************************
//VS98610A JOB (ACCOUNT),'NAME',CLASS=A,MSGCLASS=X
//*
//*
//*
//*
//*-------------------------------------------------------------
//* COMPILE STEP    (COB)
//*-------------------------------------------------------------
//COMPILE EXEC PGM=CWPCMAIN,REGION=4M,COND=(0,LT),
// PARM=('SOURCE,MAP,DMAP,NOTEST,XREF,NONUM,NOBATCH,NOCDECK',
//       NOFDECK,NOFLOW,NOLST,NOSTATE,NOSYMDMP,VERB')
//STEPLIB  DD  DISP=SHR,DSN=SYS5.COMPWARE.CX.LOADLIB
//SYSTERM  DD  SYSOUT=(*)
//SYSPRINT DD  SYSOUT=(1)
```

Supported Features

Restart Job	- Automatic or manual recovery job after failure	🟩
Chunk processing	- Commit management by period (chunk processing)	🟩
Step organized job	- Definition of jobs by sequential steps	🟩
Skip / rollback	- Treatment of partial data (eg skip record on rollback)	🟩
Transaction	- Global transactional management of jobs	🟩
Planning	- Scheduled execution of jobs	🟦
Non-sequential	- Support for non-sequential steps treatments (conditional branching)	🟦
Pause / resume	- Manage pause / resume on job's execution	🟦
Reporting	- Reporting (counters, exit codes)	🟦

Continuous Integration

Global Scope



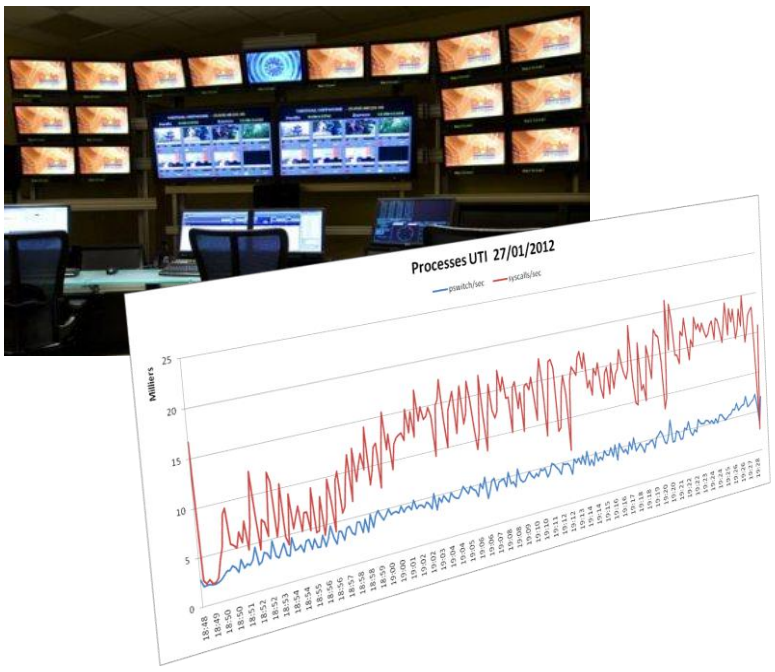
Supported Features

Automation	- Automated deliveries, distributions and publications - Acceleration of the release cycle of deliverables - Acceleration of validation / release process - Using publication descriptors for validation / release - Automated launch of unit and integration testing	🟩
Source repositories	- Communication with source repositories (CVS, SVN) - Tracking changes of applications	🟩

Library repositories	- Repositories of technical and applicatives deliverables (centralization of librairies) - Rapid provision of components to developer - Definition of delivery models (ear, war, jar etc ...)	🟩
Testing / Integration	- Technical components for writing unit and integration tests: capping component validation of navigations, business rules, input controls - Detailed report of the results of test's executions - Estimation of the test's coverage of on the code of applications	🟩
Externalised configuration	- Outsourcing the properties and dependencies for applications and Mobidick - Centralization of configurations to facilitate deployment - Facilitates the maintenance of configurations	🟩

Performance and Monitoring

Global Scope



Supported Features

Bench Application	- Provision of a pre-configured and deployable application - Allows validation of all technical channels - Allows verification of the proper load distribution according to the delegations between layers (access to a component in a remoting and / or local mode)	🟩
Metrics / Monitoring	- Graphical surveillance of applications - Graphical metrology of components by technical chains	🟨
JMX Monitoring	- Using the standard JMX to monitor each components of the application on all layers	🟩

Security (SSO, ACL)

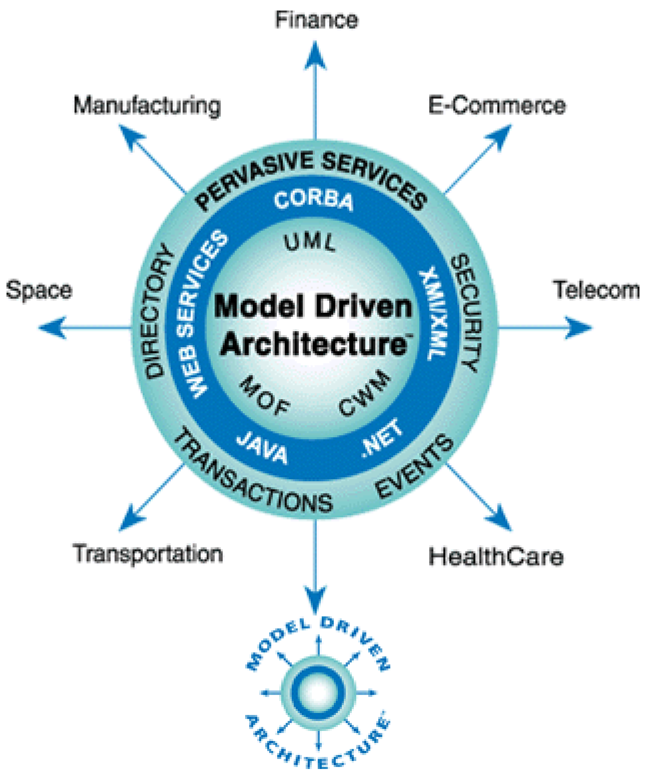
Global Scope

Supported Features

SSO	- Support for cluster affinity - Cross-platform SSO - Management (non intrusive) authentication of J2EE	🟨
Propagation	- Propagation of authentication to subsidiaries systems (HOST, RDBMS, LDAP, XML, Properties) - Support for external authentication profiles - Through Mobidick's context management	🟩
Traceability	- Configurable user's actions tracking	🟩
User unmarked	- Ability to configure unmarked users for access to various resources (eg mainframe)	🟩
Fine grained accreditation	- Ability to configure access control lists (ACL) at the class level but also method level	🟦
Administration	- Features of external administration of security and fine grained accreditation	🟨
High Availability	- External security informations are stored in a database allowing the recovery of the system without losing data (fail over)	🟩
Related Functions	- Provision of security taglibs facilitating the development of user interfaces - "SwitchUser" mechanism to change the current user during the same session - Mechanism for concurrent session handling - "RememberMe" mechanism - Filtering mechanism for information based on user profile - Cache system for security information to optimize the response time	🟨

MDA Needs Approach

Global Scope



Supported Features

By reverse-engineering	- Generate code from the reverse-engineering (approach so-called "bottom-up") of a data source such as database schema, mainframe message structure…	🟦
By modeling	- Generate code from the modeling of a software architecture such as class diagram, use case diagram, relational model…	🟦
By design	- Generate code from the design (approach so-called "top-down") of an user interface such as screen composition, graphical component…	🟦

MOBIDICK™ Prerequisites

Here are the prerequisites of **MOBIDICK™**:

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