

Course code: **ESB/M**

Course title: **Enterprise Service Bus – example of Mule ESB**

Days: 3

Description:

Course intended for:

The training is intended for Java programmers, system analysts and architects wanting to get familiar with the techniques of development of an ESB (Enterprise Service Bus).

Course objective:

The training objectives include:

- Getting the participants familiar with the basic terms and standards, associated with system integration and ESB bus,
- Getting familiar with integration patterns and methods of implementation of these patterns on the ESB,
- Presentation of good and bad practices of integration and the tools, available on the market,
- Acquiring of practical skills, associated with the Mule ESB product.

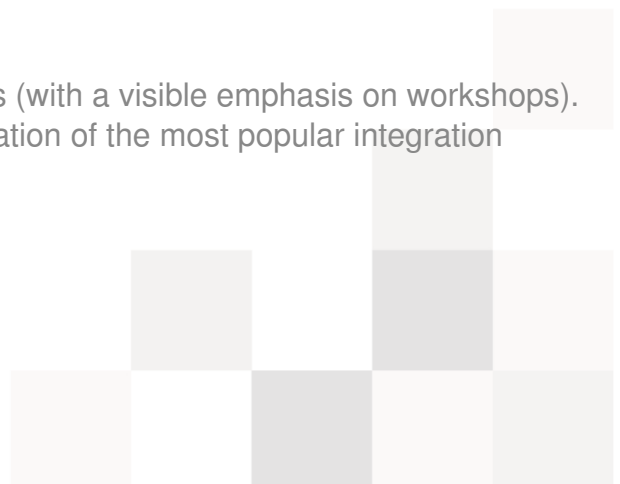
Requirements

The participants are required to have at least basic Java programming skills, to be familiar – at least on the basic level – with WebService and with the XML processing technology. Knowledge of basics of Java EE is also recommended.

Course parameters:

3*8 hours (3*7 net hours) of lectures and workshops (with a visible emphasis on workshops). During the workshop, examples that illustrate realization of the most popular integration patterns using Mule ESB are implemented.

Group size: no more than 8-10 participants.



1. The basics of integration of enterprise applications

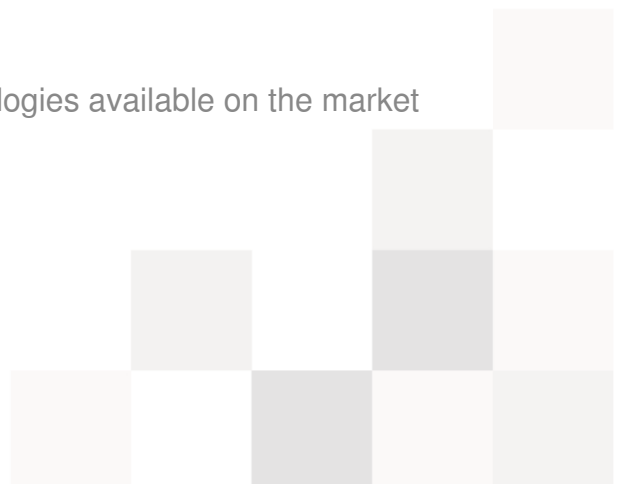
- Presentation of basic concepts and terms, associated with broadly understood integration: : silos, SOA, EIP, MEP, EAI, ESB, service bus, data bus, ETL, EDA, CMD, MOM, integration adapter and broker, orchestration, synchronous/asynchronous/offline integration, JBI, SCA, WS-* etc.
- Presentation of differences between the service bus, integration platform, integration broker, queue server etc.
- Pros and cos of implementation of an integration platform, the objective of such integration at a corporation, migration of existing systems to the "service bus"

2. Discussing of characteristics and functionalities provided by the ESB platform

- Location transparency
- Conversion of transport protocols
- Message transformation
- Message routing
- Message enhancement
- Service bus protection
- Monitoring and management
- Performance
- Interoperability
- Standardization

3. Review and comparison of tools and technologies available on the market

- Commercial and non-commercial
- Based on Java EE, .NET, other



- Mature and developing
- More and less popular
- According to other criteria

4. Discussion of Mule ESB platform

- Concepts and basic terminology for Mule ESB
 - Mule Application
 - Transport, Connector
 - Agent, Configuration,
 - Endpoint
 - Message Flow
 - Message Source, Message Processor
 - Component
 - Filter
 - Router
 - Transformer
 - Mule Configuration Pattern
 - MEL – Mule Expression Language
- Basic components and architecture of Mule ESB
 - Mule Framework – integration framework
 - Mule Manager – runtime environment
 - Mule SEDA (Staged Event-Driven Architecture)
 - Deployment in the Mule Server mode (standalone)
 - Deployment in the Mule Embedded mode (embedded in Java application)



- Deployment in CloudHub
- Ready-to-use components of Mule ESB
 - Components (CXF, Echo, Expression, HTTP Resource Handler, Logger, REST Component)
 - Routers (inbound, outbound, async-reply)
 - Transformers (Java, XML, XSLT, Script, Datamapper, ...)
 - Catch-all Strategies
 - Message Flow Processors (async, choice, aggregator, splitter, custom processor, idempotent filter, message filter, processor chain, recipient list, redelivery, request-reply, round robin, wire tap, ...)
 - Connectors (HTTP, File, FTP, JDBC, JMS, POP3, AMQP, Salesforce, ...)
 - Mule AnyPoint DataSens
- Creation of own components of Mule ESB on the basis of:
 - `org.mule.api.processor.MessageProcessor`
 - `org.mule.routing.outbound.AbstractOutboundRouter`
 - `org.mule.transport.*`
- The quality of performance of services by this platform (mechanisms: high accessibility, resistance to failures, load balancing, persistence, transactionability and security)
- Testing of solutions based on Mule ESB (Mule TCK – Test Compatibility Kit)
 - Functional tests (`org.mule.tck.junit4.FunctionalTestCase`)
 - Unit tests (`org.mule.tck.AbstractMuleTestCase`)
 - Performance tests (Mule Profiler Pack)
- Management and monitoring of the Mule environment via the Mule Management Console
- Similarities and differences in relation to other ESB platforms. Discussing of



the entire Anypoint Platform family of products, to which Mule ESB belongs

- Programming environment -Mule Studio / Anypoint Studio
- Discussing of differences between the commercial version (Mule ESB Enterprise Edition) and the freeware version (Mule ESB Community Edition)

5. The most frequently applied integration patterns (EIP – Enterprise Integration Patterns) and their mode of implementation using Mule ESB

- Channel
- Message
- Service
- filter
- Router
- Transformer
- Endpoint
- Discussing of other popular patterns

6. Working with message management

- Message structure
- Message types and formats
- Transformation and conversion of messages
- Validation of messages
- Persistence of messages

7. Working with services

- Structure of service
- Service types
- Service contract



- Service configuration

8. Arrangement of services and routing of messages on the bus

- Service registers and repositories
- Routing on the ESB
- CBR - Content Based Routing
- Notifications

9. Service performance quality

- Service replication
- Service level and transport protocol clustering
- Repeating of messages
- Service monitoring and management
- Implementation of changes (hot deployment)
- Other

10. Securing of services

- Secure services on Mule ESB
- Authentication and authorization
- Data encoding on the bus

11. Management of errors and exceptions

- Repeating
- Compensation
- Withdrawal
- Transaction processing

12. Performance



- Tuning of service parameters (transport protocol, number of threads etc.)
- Cache
- Launching environment parameter adjustment (Java, application server, queue server, database etc.)

13. Testing services on the ESB

- Methods and tools supporting service testing (automation)

14. Advanced services on ESB

- The difference between integration flow, orchestration of services (BPEL) and a business process (BPM)
- Support for business processes (BPM) and orchestration of services (BPEL)
- Support for business principles (BRMS)
- Support for event stream processing (CEP)

15. Review of the most popular integration adapters

16. Good and bad practices in building of integration solutions based on ESB

- Frequently encountered practices
- Recommended project patterns and anti-patterns to be avoided
- Recommended communication protocols

