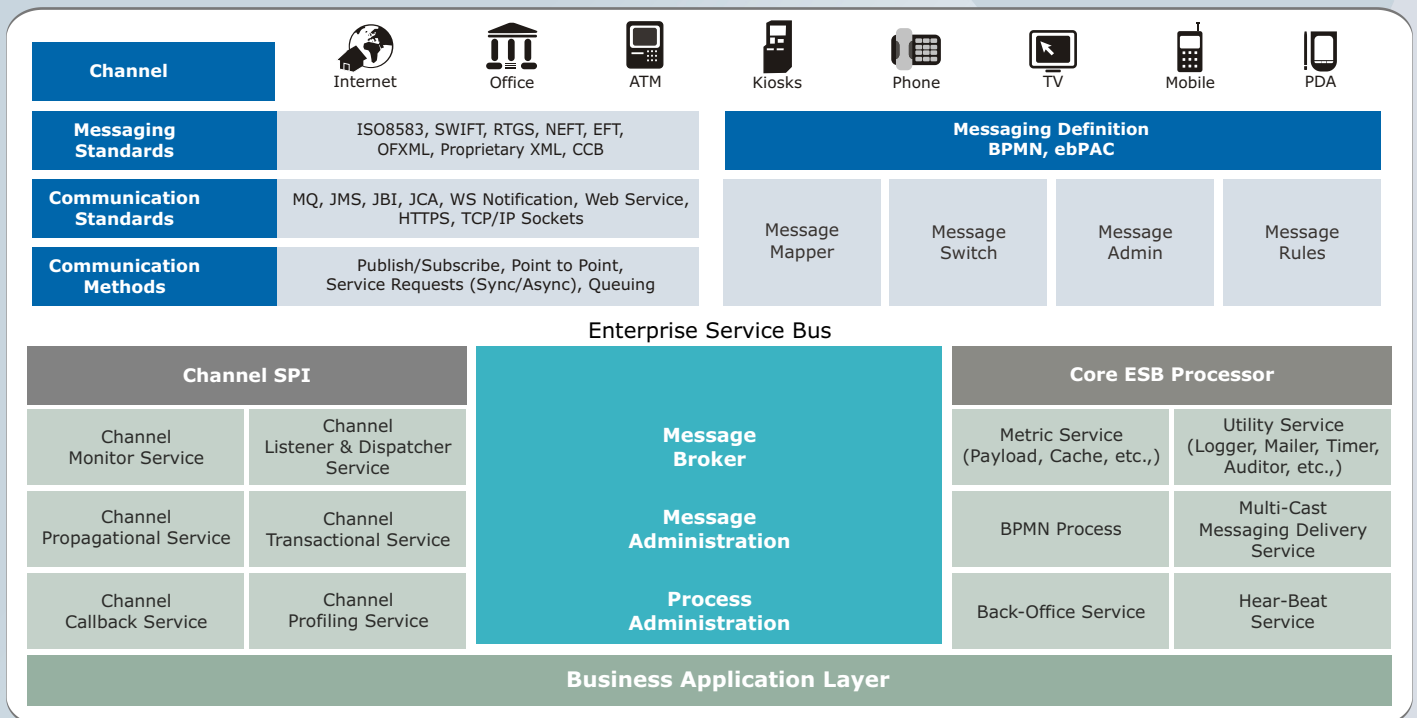


DESIGNED TO STAND THE TEST OF TOMORROW



Modern business enterprises comprise multiple legacy systems built along functional rather than business process oriented lines. Such systems, which were never designed to share information cross-functionally, create information silos, preventing optimal monitoring and coordination between enterprise functions. Use of application-specific adapters to bridge these systems creates brittle integrations with multiple points of failure. Hence a new architecture is required which is scalable and adaptable and provides high performance for mission critical tasks.



One such architecture is Enterprise Service Bus. The ESB architecture envisions brokering all interactions between applications through enterprise middleware which encapsulates application functionality and removes application specific dependencies.

While the idea is simple, the execution is far more complex. Technological requirements include:

- Addressing and routing capabilities to provide location transparency.
- A mechanism for defining and administering global application interfaces and the functionality that they will provide (services).
- Support for all messaging paradigms e.g. Publish/Subscribe, Point to Point, Service Requests (Sync/Async), Queuing, One to Many, Many to One etc to pass information and requests between applications.
- Support for transformation and mapping logic to transform existing application interfaces into standard form.
- Support for multiple communication protocols: MQ, JMS, JBI, JCA, WS Notification, Web Service, TCP/IP Sockets etc.

In addition to these basic requirements, desirable features include:

- Support for advanced routing and logic capabilities to transform message payload and delivery location en route.
- Support for multiple data standards: e.g. ISO8583, SWIFT, RTGS, NEFT, EFT, OFXML, Proprietary XML, CCB
- Reliable message delivery mechanisms to ensure that messages are delivered once and only once, in the correct order
- The ability to maintain quality of service to ensure delivery of high priority messages in a time-sensitive manner.
- Robust error and fault handling mechanisms to allow smooth recovery from system and message errors.
- Sophisticated management and reporting mechanisms to provide visibility into the running systems and automatic timely notification of exceptional conditions.

The Solution

Cuecent ESB is a standards-based intelligent Enterprise Service Bus which provides a platform for integrating enterprise applications in a scalable, flexible and reliable manner. It is implemented within Cuecent BPMS, so it is highly configurable and extensible. All service definitions and message transformations within the Cuecent ESB are defined as BPMN processes. In addition, enterprise integrations defined in Cuecent ESB may be exposed as services which can be consumed by Cuecent BPMS workflows to deliver required business functionality.

It is based on a scalable architecture and provides support for multiple communications protocols and data format standards. It is extremely flexible in terms of message transformations - allowing messages to be transformed pre-transport, during transport, or pre-delivery, and based upon source, destination or content. It supports reliable message delivery and prioritization.

Cuecent ESB includes the following components:

ESB Message Builder: To configure various messaging standards and message format adaptors.

ESB Flow Builder: To configure message flows, message transformations and access controls.

ESB Engine: To translate, switch, and orchestrate messaging services enabling SOA.

ESB Monitor: To administer message loads, cache, and reporting.

Features

Cuecent ESB supports:

- Inter-converting message formats
- Routing rules
- Multi host interactions
- Validation rules
- Message storage into database
- Audit trail
- Host System Adapters
- Transformation rules
- Run-time modification of messaging processes
- Changes to message formats using message configuration component
- Supports online monitoring of service endpoints
- Evaluation of event rules by capturing fault occurrences and determination of usage patterns

