



Logosworld.com



Lean SOA

Loose coupling, MQ, EDA

Axel Angeli

Logos! Informatik GmbH

Axel.jax2009@logosworld.de

Logosworld.com



The path to lean SOA is through a proper design that allows execution of code bubbles locally with communication at endpoints only-

Lean SOA: Process Locally, Communicate Globally

A clear OO design allows you to jump-start SOA



Most SOA code concentrate s on service protocols rather than service alorithm. Challenge is to reduce app to SOA communicat ion to one single canonical format

How does SOA work today?

- ✓ Client connects to middleware
- ✓ Middleware translates request
 - ✓ Request is sent to server

Logosworld.com



Applications can concentrate on algorithm and drop/peek messages only in canonical formats.

Concentrate on application

- ✓ You forgot about device drivers?
- ✓ Now forget protocols etc.

Logosworld.com



Practically it means that you need to encapsulate every I/O in a separate proxy (object) and communicate via this proxy alone.

The path to lean SOA

- ✓ Create identical local and remote proxies
- ✓ Develop against local proxy
- ✓ Let proxies care of communication

Logosworld.com

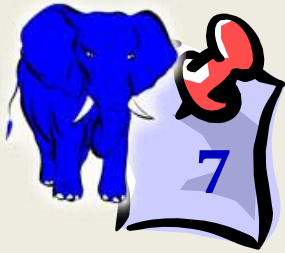


Communication is handled via synchronizing the queues only. No data fetch from/to apps across domains or limits of ecosystem.

Requests are handled in queues

- ✓ Requests go to local queue
 - ✓ Queues are auto-sync!
- ✓ Processing against queue which is local to process

Logosworld.com



The decision which proxy can handle the request is taken late. In fact multiple handling is (broadcast) easy then.

Active Proxy picks the request

**Whether Remote or local:
Requester is completely
agnostic to it**

Logosworld.com



**Instead of a new connection
for every remote action**

- ✓ Only local DB (=MQ) operations with wait!
- ✓ That is lean!

Logosworld.com



**Only MQs communicate over
the network**

**This keeps SOA
development lean and
easy**



Loose coupling = late binding

- ✓ Assign of handler takes place „late“ during execution time

Logosworld.com



MQ processing naturally leads to loose coupling

- ✓ A message can be picked up by any service
- ✓ Governance checks if a service is entitled to



It is the old MVC paradigm

- ✓ Model-View-Controller
 - ✓ Segregation of Duties
- ✓ Canonical data transport

Logosworld.com



The consequence



Loose coupling will keep layers necessarily separate. Since the sender cannot know the device it needs to stay in canonical format

- **No longer writing of AJAX or Flash Code**
 - Front end handler receives XML message
 - Front end decides whether JS or FLX etc

Logosworld.com



The whole communication is „outsourced“ from the application. Apps only talk to channels and the channels are serviced by special handling agents.

Communication is reduced to MQ



■ Apps address logical channels!

- No programming against devices
- Data sent to channels (MQ) only
- Queues are treated by handling agents
- App dev is complete protocol agnostic
 - Not SOAP, XMLRPC etc, just SEND/RECEIVE
- Async/Sync handling is a duty of queue only



Summary

- ✓ Loose coupling is a side-effect of asynchronous MQ processing
 - ✓ It is prerequisite for clouds
- ✓ It is the difference between SOA and simple networked programming



Loose coupling with MQ = Lean SOA

- ✓ Dhanyawada galu!
 - ✓ Sukria!
 - ✓ Nandri!
- ✓ Thank You!
- ✓ Danke schön!