

<http://membres-liglab.imag.fr/donsez>

Message Oriented Middleware (MOM)

MOM & JMS, Didier Donsez, 1998-2012 – F. Baudé 2014

Merci à Didier DONSEZ
*Université Joseph Fourier (Grenoble
1)*

PolyTech Grenoble – LIG ERODS

`Didier.Donsez@imag.fr`

`Didier.Donsez@ieee.org`

27/04/2022

MOM & JMS, Didier Donsez, 1998-2012

1

1

27/04/2022

Motivations

■ Modèle Client-Serveur

- requêtage synchrone
- RPC DCE et DCOM, CORBA, RMI
- inconvénient : connexion permanente des 2 parties
- Problème des pannes/connexions transitoires
- Delay-Tolerant Networks

■ Une alternative : la messagerie « Messaging » inter-application

- les messages (qui peuvent être des requêtes et leurs réponses) sont envoyés quand la connexion est ouverte.

■ Style architectural du *Store-and-Forward*

- voir <http://www.eaipatterns.com/MessagingComponentsIntro.html>

■ Ne pas confondre avec le Message Passing

- Ex MPI, PVM: surcouches aux sockets pour applications parallèles facilitant l'échange de données sans notion explicite de boîte à lettres²



MOM & JMS, Didier Donsez, 1998-2012 – F. Baudé 2014

2

27/04/2022

Motivations

- Applications (passage à très grande échelle)
 - Diffusion d'information (push)
 - news, stock quote, weather forecast ...
 - Messagerie inter-bancaire, workflow, ERP, ...
 - Synchronisation de BD nomades et réplicat asynchrone (hot standby)
 - EAI (Enterprise Application Integration), B2B
 - ESB (Enterprise Service Bus)
 - Data Warehouse (*ETL : Extract Transform Load*)
 - Collecte des données (journaux Firewall, mesures réseaux de capteurs, ...)
 - Déploiement grande échelle de logiciels (antivirus, ...)
 -

MOM & JMS, Didier Donsez, 1998-2012

3

3

27/04/2022

Principe

- Messagerie inter-application
 - Asynchrone
 - Non temps réel – Offline (Not online)
 - s'oppose aux ORBs synchrones (Corba, DCOM, RMI)
- Fichiers de Messages (Message Queueing)
 - les messages sont mis dans une file d'attente persistante (i.e. sur disque) avant d'être relayés vers l'application: guaranteed delivery
 - Partage d'une file par plusieurs applications
 - Priorité des messages
 - Filtrage des messages à la réception
- Avantages
 - Insensible aux partitions de réseaux (sans fil, satellite, WLAN, ...)
 - Insensible aux applications non disponibles (temporairement) ou latence

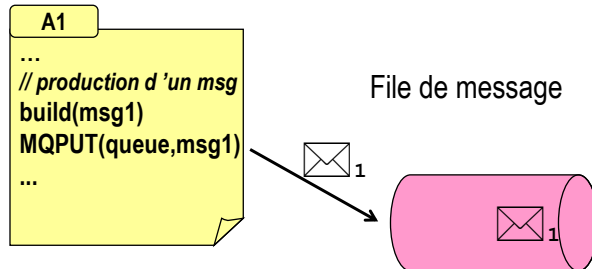
MOM & JMS, Didier Donsez, 1998-2012

4

4

27/04/2022

Principe des Files de Messages (i)



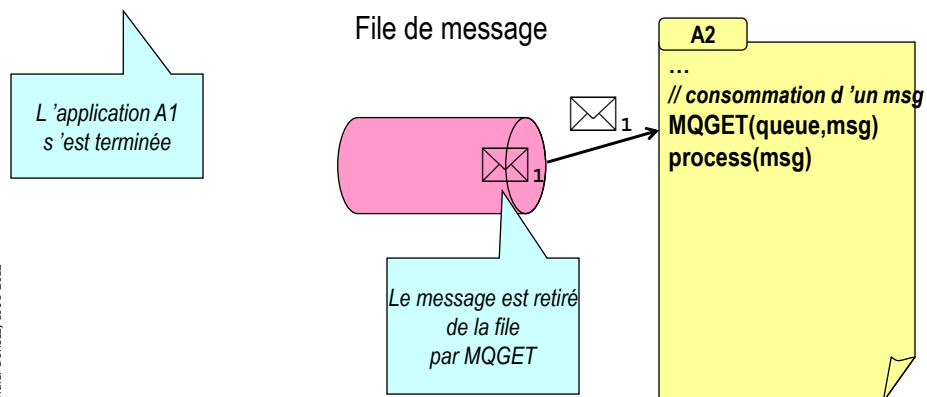
MQM & JMS, Didier Donsez, 1998-2012

5

5

27/04/2022

Principe des Files de Messages (ii)



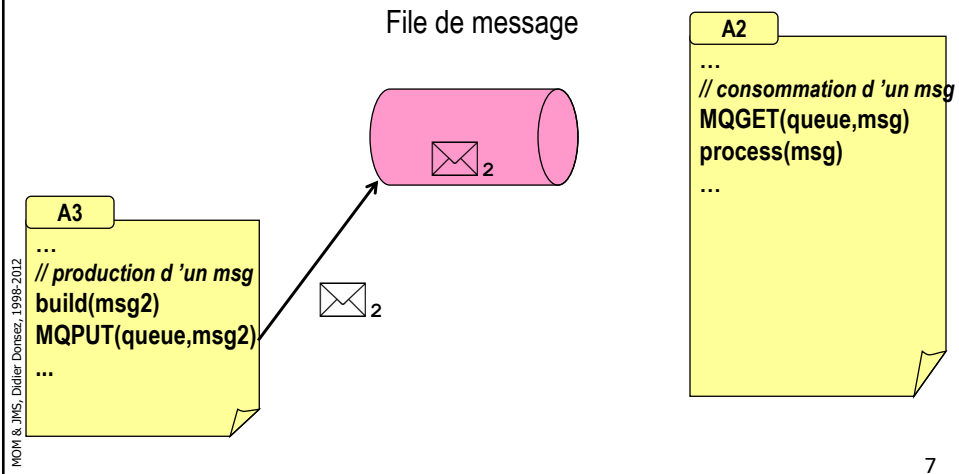
MQM & JMS, Didier Donsez, 1998-2012

6

6

27/04/2022

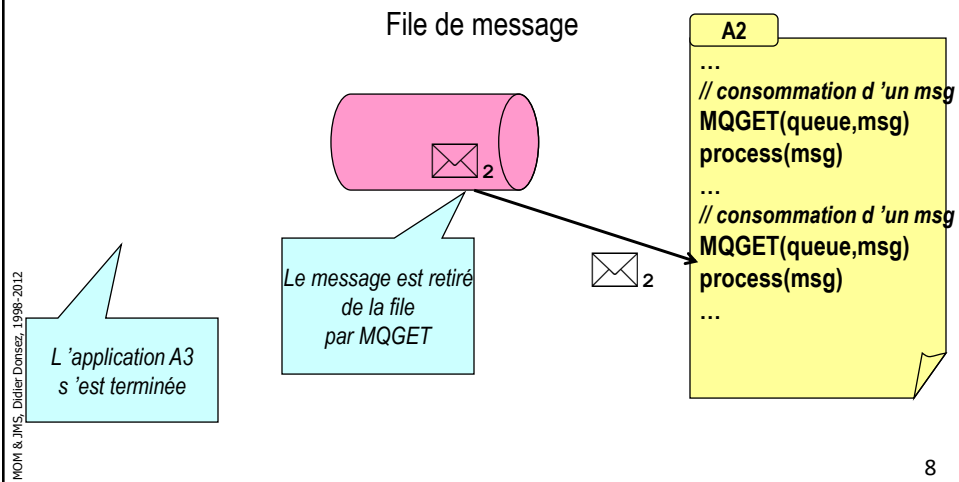
Principe des Files de Messages (iii)



7

27/04/2022

Principe des Files de Messages (iv)



8

8

27/04/2022

Modèles de messageries

■ Routage de Message

- par l'identité de l'application
- par le contenu du message
- chaque application consommateur définit un critère sur les messages à consommer
- le critère peut être 1 expression booléenne sur les valeurs de champs du message

■ Modèles

■ Message Queue

- un message envoyé (produit) est consommé par un **seul** client

■ Publication-Souscription

- un message publié est diffusé à **tous** les souscripteurs
- Publication-Souscription par le contenu (content based publish-subscribe)
- un message publié est diffusé à tous les souscripteurs par rapport au contenu du message (IBM' Gryphon, U. Colorado' Siena, ...)
- Requête-Réponse
- Client-Serveur asynchrone basé sur des queues pour ces messages là

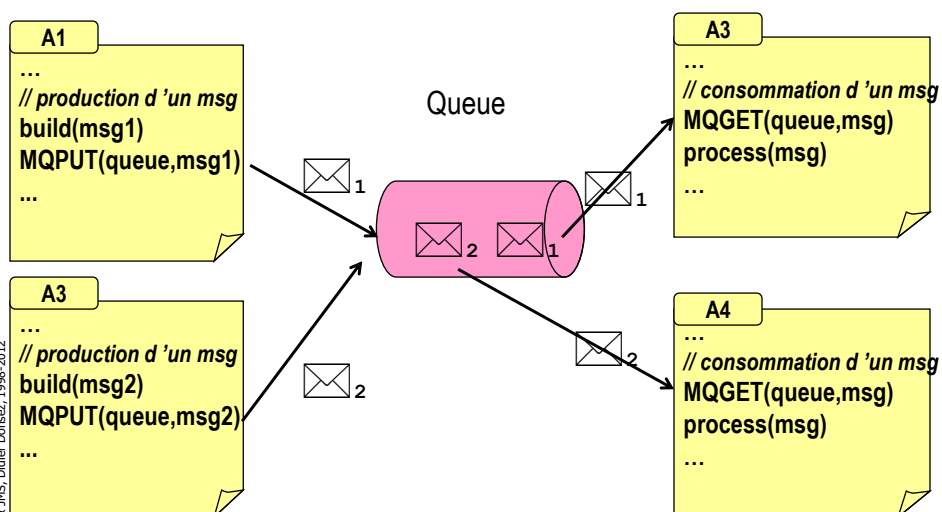
MOM & JMS, Didier Domez, 1998-2012

9

9

27/04/2022

Modèle des Message Queues



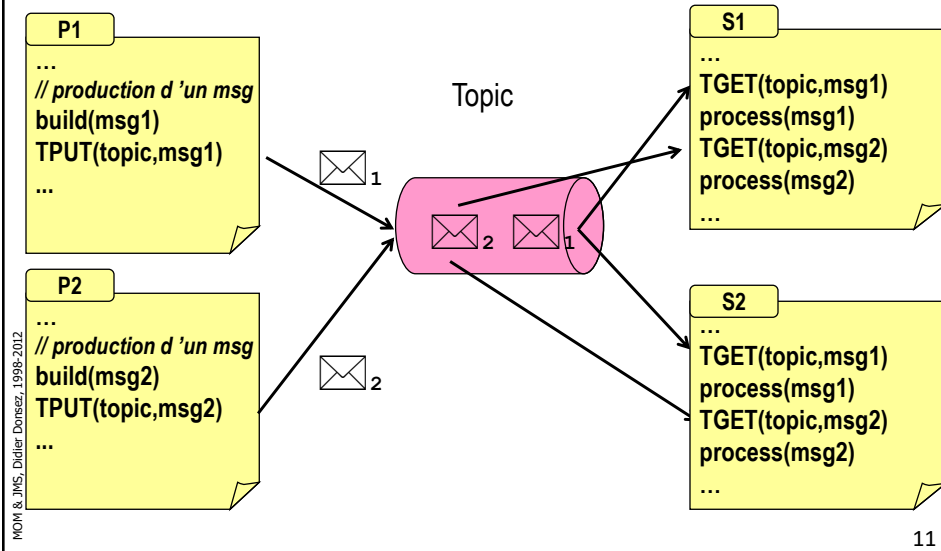
MOM & JMS, Didier Domez, 1998-2012

10

10

27/04/2022

Modèle Publication-Souscription

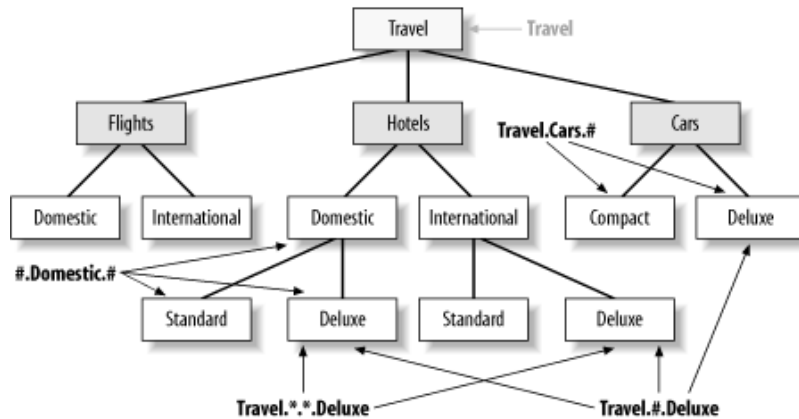


11

11

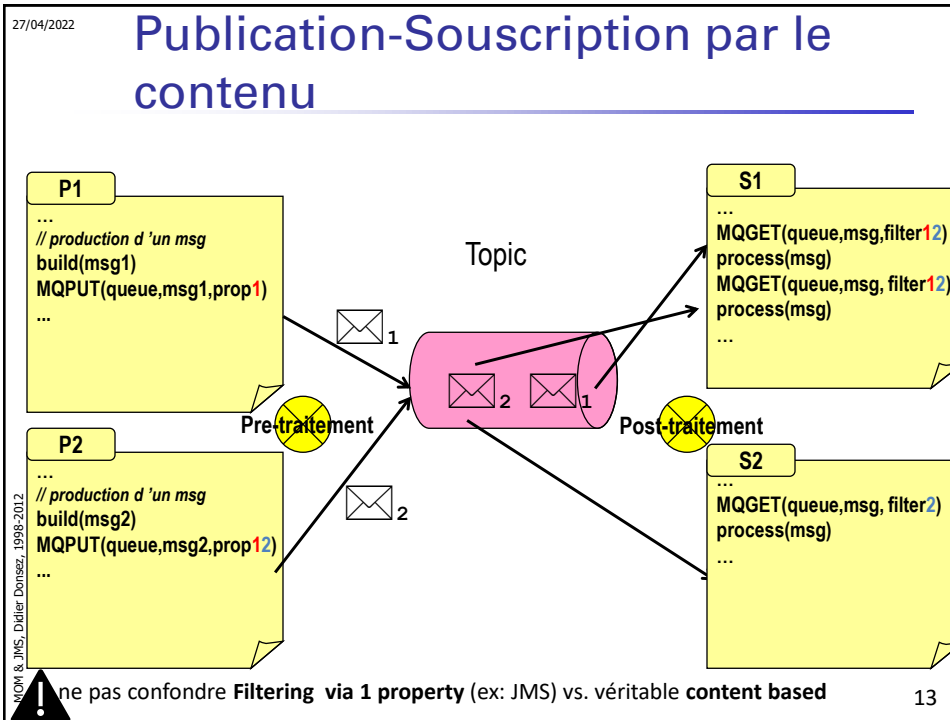
27/04/2022

Publication-Souscription sur des *topics* hiérarchiques

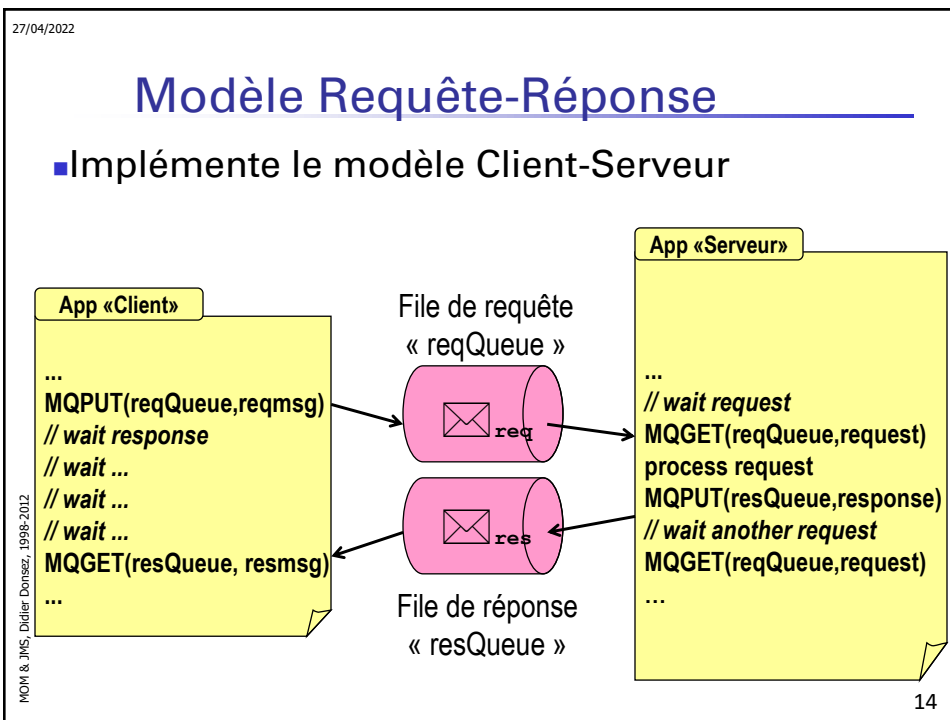


D'après Dave Chappel

12



13



14

27/04/2022

Architecture d'un MOM

- **Client MOM**
 - relié de manière permanente à un serveur MOM
 - **envoie** et **reçoit** des messages
- **Serveurs MOM**
 - reliés entre eux de manière épisodique
 - réseau mobile, réseau WAN sur lignes dédiés, ...
 - maintiennent des copies des messages
 - réplication (serveurs primaires, serveurs secondaires)
- **Administrateur/Contrôleur du MOM**
 - crée et surveille les files
 - définit la topologie des interconnexions entre serveurs
 - définit les politiques de connexion (période, ...)

MOM & JMS, Didier Donsez, 1998-2012

16

16

27/04/2022

Implémentation

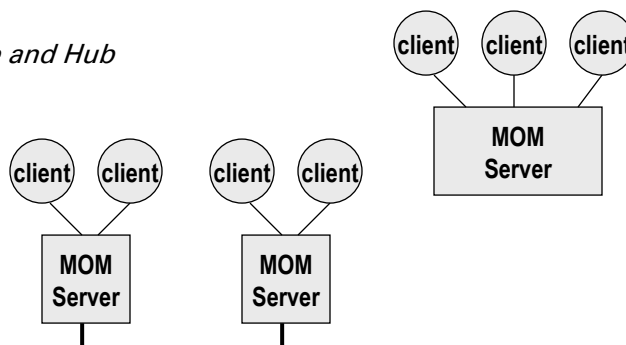
- **Architecture**
 - Centralisée : *Spoke and Hub*

- Distribuée : *Bus*

- Pair à Pair : *Snowflake*

- **QoS**

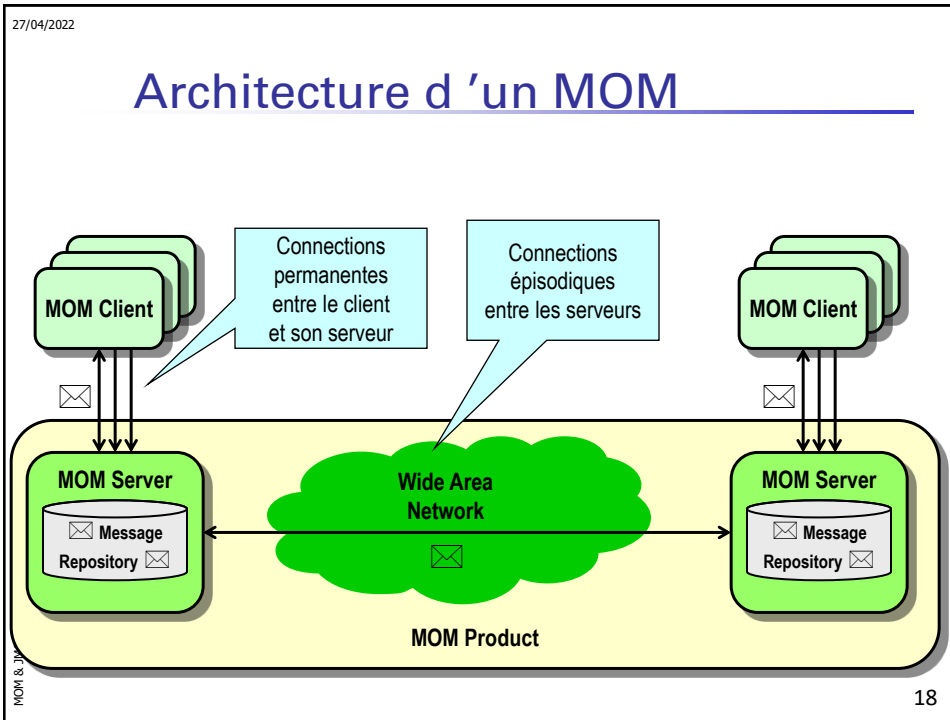
- Disponibilité (du MOM), Causalité des msgs délivrés respectée, Fiabilité (perte possible msgs?), Passage à l'échelle (perfs), Sécurité, ...



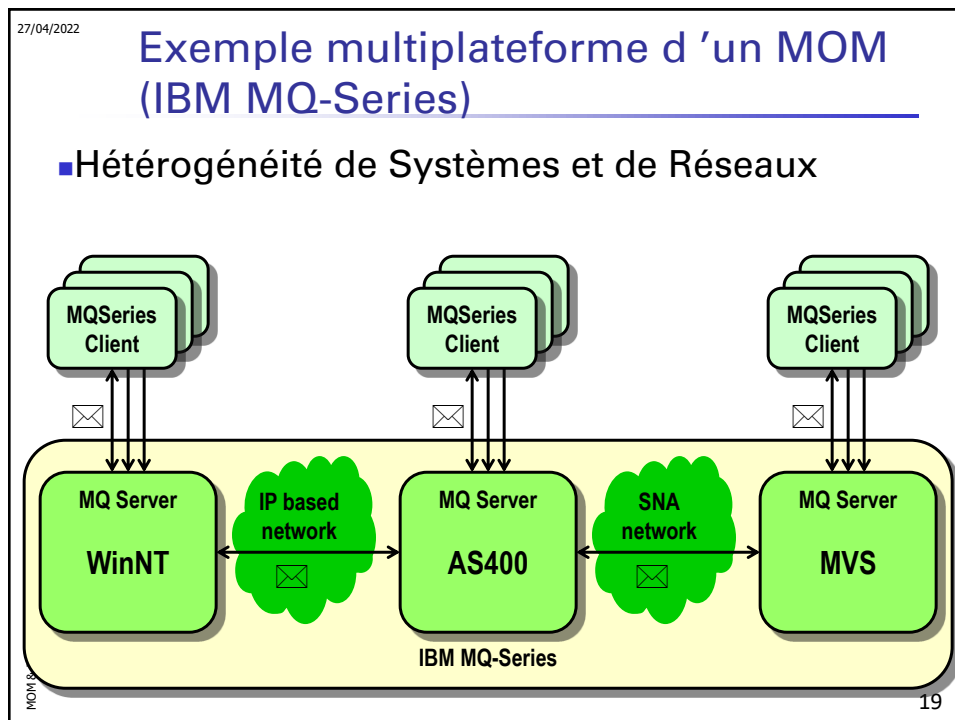
MOM & JMS, Didier Donsez, 1998-2012

17

17



18



19

27/04/2022

Interopérabilité entre MOMs ?

- Difficulté de faire interopérer des MOMs
- Pas de standardisation entre les MOMs, de base
 - Certains efforts plus récents pour définir un protocole de Messagerie Interoperable : AMQP
- Des tentatives (historiques) pour l'interopérabilité
 - CORBA 3.0
 - introduction de la notion de messages asynchrones, notification service
 - J2EE
 - JMS javax.jms
 - API Java permettant à des clients d'envoyer/recevoir des messages avec des serveurs implémentant des JMS SPI (service provider interface) sans pour autant être des implémentations du protocole de messagerie JMS
 - EJB : Message-Driven Bean

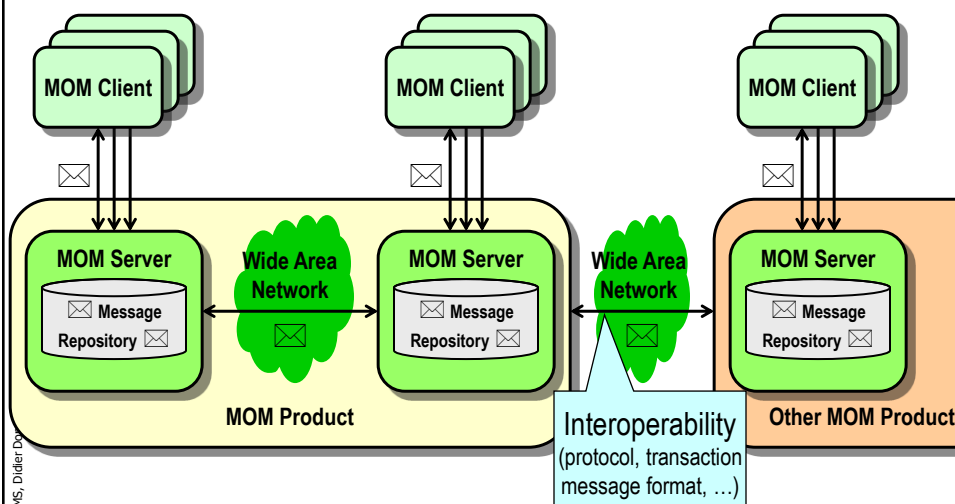
MOM & JMS, Didier Donsez, 1998-2012

20

20

27/04/2022

Interopérabilité entre MOMs



MOM & JMS, Didier Donsez, 1998-2012

- Issues : End-to-End Transactional delivery ?


21

21

27/04/2022

Exemples d' « intergiciels » PubSub

Spécifications/protocoles

- OSGi Event Admin : Wire Admin
- CORBA Data Distribution Service (DDS)
- AMQP: an OASIS open Internet (or “wire”) Protocol standard for message-queuing communications
- STOMP: Simple/Streaming Text Oriented Messaging Protocol
- MQTT: Message Queue Telemetry Transport
- XMPP PubSub: publish subscribe in the context of instant messaging
- ROS (Robot Operating System)
- UPnP GENA: General Event Notification Architecture.
- PubSubHubbub (hubs publics hébergés chez Google, webhooks (callbacks sur subscribers, étend RSS/Atom) )
- CoAP CORE: Constrained Application Protocol (CoAP), a RESTful protocol

Quelques solutions de brokers (plus dans prochains slides :=))

- *Siena*: Scalable Internet Event Notification Architectures, content based, recherche (EPFL)
- *Apache Kafka*: distributed publish-subscribe messaging system designed as a replicated extended commit log service

MOM & JMS, Didier Donsez, 1998-2012 – F. Baude 2014

22

22

27/04/2022

Opérateurs Cloud PubSub-as-a-Service

Xively

Axeda

Open.sen.se

Thingworx

SKYNET.im

ClearBlade

2lemetry

AirVantage

WSO2 MB

...

Twitter dans une certaine mesure (Hashtag=topic).

MOM & JMS, Didier Donsez, 1998-2012

23

23

27/04/2022

AMQP

Advanced Message Queuing Protocol

- <http://amqp.org/>
- Red Hat, Cisco Systems, IONA, iMatix, ...
- Standardiser l'échange de messages entre serveurs de message (standard OASIS) hétérogènes
<http://docs.oasis-open.org/amqp/core/v1.0/amqp-core-complete-v1.0.pdf>
- Support des transactions XA (two-phases commit)

Transaction (répartie ou pas): l'émetteur peut englober un paquet de messages dans une transaction, ces messages ne pourront être lus que lorsque l'émetteur les aura acquittés.
- Implémentations
 - Red Hat Enterprise MRG, IONA, ...
 - OpenAMQ, Apache QPid, ActiveMQ plus récemment ...
 - JORAM, RabbitMQ: *a solid [multiprotocol, polyglot broker](#): it can send STOMP, MQTT, or AMQP in and get one of the other ones out.*

MOM & JMS, Didier Donsez, 1998-2012

24

24

AMQP versus JMS

- la spéc. JMS est le vrai concurrent d'AMQP même si encore une fois les deux peuvent vivre ensemble. Un adaptateur JMS pour AMQP peut sembler intéressant pour des projets qui migrent vers un nouveau broker, mais il semble plus pertinent pour un nouveau projet de commencer directement dans ce nouveau standard. Par exemple l'utilisation de la librairie Java de RabbitMQ est vraiment très proche de l'API JMS et la vitesse d'apprentissage semble assez rapide pour un développeur Java. Car un des problèmes de JMS c'est justement son manque d'interopérabilité avec d'autres langages.
- <http://www.wmrichards.com/amqp.pdf> : Consider the case where you want to send a message from a Java message producer to a Ruby message consumer. Since Ruby can't use JMS, you need a message broker that can bridge the two platforms and transform the protocol and message structure used by each platform. Since the most popular choice for Ruby is the STOMP protocol you would need a message broker that can support both STOMP and JMS at the same time. You can use ActiveMQ but ... you would be locked into one specific vendor solution (or in some cases only a few vendor choices) due to the built-in message bridge. **Whereas JMS provides a standard messaging API for the Java Platform, AMQP provides a standard messaging protocol across all platforms. AMQP has taken the broker-agnostic benefits of JMS within the Java Platform and escalated that concept to all platforms**=> autant prendre une plateforme qui est AMQP si le but est de supporter l'hétérogénéité des langages/plateformes
- The routing model of AMQP essentially separates the transport model from the queuing model

MOM & JMS, Didier Donsez, 1998-2012 – F. Baude 2014

25

OMG Data Distribution Service

- The **Data Distribution Service** for Real-Time Systems (**DDS**) is an (OMG) **standard** for a machine to machine (M2M) middleware
 - Topic publish subscribe model, data transfer oriented (data=sample) « data-broker »
 - No queues but a shared data space (DB table with rows), dynamic topic-based discovery of dataflows (matching pub with sub), interoperable
 - Applications never need information about the other participating applications, including their existence or location: DDS takes care of determining where recipients are located
 - Since DDS discovery is spontaneous, the topics can dynamically change over the lifetime of a deployed distributed system based on DDS, without any administrative impact.
 - ≠ JMS: Static destinations are discovered via JNDI APIs, which bind logical destination names to destination objects. The static destinations accessible this way must have been previously configured in the JMS middleware (server) using vendor supplied administrative tool
 - De + en + d'implémentations: cibles IoT, M2M, =>pas de transactions
 - Eg: used for Dutch rail network management
 - <http://www.dds-foundation.org/who-is-using-dds-2/>
- http://portals.omg.org/dds/sites/default/files/Comparison_of_DDS_and_JMS.pdf

MOM & JMS, Didier Denez, 1998-2012 – F. Baude 2014

26

MQ Telemetry Transport (MQTT)

<http://mqtt.org/> (IBM defined)

- Protocole léger de type Publish-Subscribe (no queue) pour M2M
 - Hiérarchie de « topics » : /buildingF/sensors/s11/#
- Support de connectivité (TCP/IP) intermittente ou couteuse
 - Satellite, WSN, ...
- Faible overhead par paquet (2 octets)
- 3 niveaux de QoS pour livraison (fire-and-forget, fire-and-confirm)
 - 0 (At Most Once), 1 (At Least Once) and 2 (Exactly Once)
- Retained messages
- Sécurité par certificat (SSL/TLS) + user-password
- Nombreuses implémentations de clients et de serveurs concises
 - C, C++, Arduino, Java, Python, JS (Node.JS), Lua, ...
 - 80 KB pour l'implémentation de référence (IBM)
- Standardisation OASIS (5.0, 2018)
- Clients, Servers, MaaS (« metal as a service »)
 - Mosquitto, Eclipse Paho (m2m.eclipse.org), RabbitMQ, JORAM

MOM & JMS, Didier Denez, 1998-2014

27

27

27/04/2022

Apache Kafka

- <http://kafka.apache.org/intro>
- Sorte de modèle Queue&Pub/sub en + général!
 - Répartit msgs d'un topic sur instances consos/groupe
 - Données gardées même si consommées (go in the past!)
- Les données sont stockées dans des fichiers de « log », sous forme de partitions qui peuvent être dupliquées et distribuées sur cluster Kafka
 - Protocole de maintien de la cohérence des copies
 - Commit protocol(s) à la Paxos , avec propriété « FIFO sent order » sur chaque partition d'un topic
 - Utilise pour cela le système Zookeeper (service nommage réparti)
 - Possibilité de manipuler les données reçues avant renvoi dans un autre stream (Dist. Stream Proc. System)

28

28

27/04/2022

RabbitMQ

- <http://www.rabbitmq.com/>
- Basé sur Erlang
- Distribué
- Aussi général que JMS /ActiveMQ,
 - Tous les modèles de conso. de messages
- supporte plus d'interopérabilité de protocoles (implémente AMQP), est multi langages,
- <https://stackshare.io/stackups/activemq-vs-kafka-vs-rabbitmq>

29

29

27/04/2022

Comparaison de protocoles (supposant qu'ils ont été implantés)

	DDS	MQTT	AMQP	JMS	REST
Abstraction	Pub/Sub	Pub/Sub	Pub/Sub	Pub/Sub	Request/Reply
Architecture Style	Global Data Space	Brokered	P2P or Brokered	Brokered	P2P
QoS	22	3	3	3	Provided by transport e.g. TCP
Interoperability	Yes	Partial	Yes	No	Yes
Performance	10s of 1000s of messages per second. Massive fan-out performance	Typically 100s to 1000+ messages per second per broker	Typically 100s to 1000+ messages per second per broker	Typically 100s to 1000+ messages per second per broker	Typically 100s of message per second
Real-time	Yes	No	No	No	No
Transports	UDP by default but other transports such as TCP can also be used	TCP	TCP	Not specified but typically TCP	TCP
Subscription Control	Partitions, Topics with message filtering	Topics with hierarchical matching	Exchanges, Queues and bindings in v0.9.1 standard, undefined in latest v1.0 standard	Topics and Queues with message filtering	N/A
Data Serialization	CDR	Undefined	AMQP type system or user defined	Undefined	No

Messaging Technologies , A Comparison Between DDS, AMQP, MQTT, JMS and REST , PrismTech/ADLINK
Andrew Foster ,

MOM & JMS, Didier Domez, 1998-2012

30

30

27/04/2022

Comparaison (supposant qu'ils ont été implantés)

	DDS	MQTT	AMQP	JMS	REST
Standards	OMG's RTPS and DDSI standards	Proposed OASIS standard MQTT	OASIS AMQP	JCP JMS standard	Is an architectural style rather than a standard
Encoding	Binary	Binary	Binary	Binary	Plain Text
Licensing Model	Open Source & Commercially Licensed	Open Source & Commercially Licensed	Open Source & Commercially Licensed	Open Source & Commercially Licensed	HTTP available for free on most platforms
Dynamic Discovery	Yes	No	No	No	No
Mobile OS	Yes	Yes	Yes	Dependent on JAVA capabilities of the OS	Yes
Multi-phase Transactions	No	No	Yes	Yes	No
Security	Vendor specific but typically based on SSL or TLS with proprietary access control	Simple Username/Password Authentication, SSL for data encryption	SASL authentication, TLS for data encryption	Vendor specific but typically based on SSL or TLS. Commonly used with JAAS API	Typically based on SSL or TLS

Messaging Technologies , A Comparison Between DDS, AMQP, MQTT, JMS and REST , PrismTech/ADLINK
Andrew Foster ,

MOM & JMS, Didier Domez, 1998-2012

31

31

27/04/2022

Le Transactionnel

- La consommation et la production de messages peuvent être des actions recouvrables
 - une file des messages est considérée comme une ressource recouvrable
- Elles ne sont effectives qu'à la validation d'une transaction
 - tous les messages produits sont envoyés au moment de la validation, et ceux consommés avec succès (par receive() ou onMessage()) sont retirés de la file
 - en cas d'abandon de la transaction, les messages produits sont abandonnés (=non envoyés) et aucun message consommé n'est acquitté ce qui implique qu'il reste/"est remis dans la file", et sera donc re-délivré plus tard
- La transaction peut être distribuée (cad inclut d'autres partenaires externes au MOM)
- Moniteur transactionnel (XA – j2ee, MTS – msoft , ...)

MOM & JMS, Didier Donsez, 1998-2012 – F. Baude 2014

32

32

27/04/2022

Conséquences du Transactionnel

- L'ordre de consommation des messages peut être différent de l'ordre de production


```

begin T1
  T1 produit M1
  T1 produit M2
  commit T1 => M1 et M2 sont bien dans la file

begin T2
  T2 consomme M1 => M1 est retiré de la file
begin T3
  T3 consomme M2 => M2, msg suivant est retiré de la
  file
  abort T2 => M1 est remis dans la file
  commit T3 => confirmation que M2 est bien retiré
begin T4
  T4 consomme M1 => M1 est donc consommé après que M2
  l'ait été
  commit T4
      
```

MOM & JMS, Didier Donsez, 1998-2012 – F. Baude 2014

33

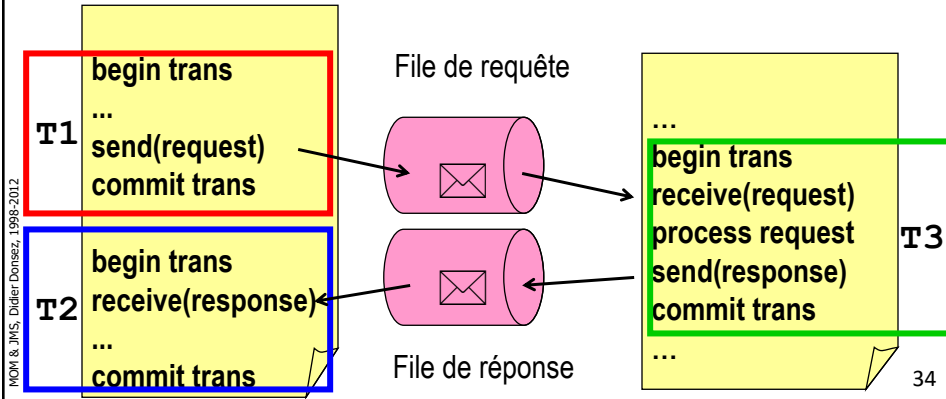
33

27/04/2022

Conséquences du Transactionnel

■ Conception de Requête-Réponse transactionnel

- l'envoi de la requête et la réception de la réponse sont forcément dans 2 transactions successives : Pourquoi ??



34

Messaging Transactionnel: ex de JMS

<http://middleware.smile.fr/Concepts-des-moms-et-jms/Caracteristiques-principales-des-mom>

Il y a de nombreux usages des transactions JMS, d'une manière générale pour assurer la *cohérence*:

- Une application peut par exemple émettre 10 messages et être assurée que soit tous seront bien émis, soit aucun ne le sera.
- Une application qui jouerait un rôle de relais pourra ainsi lire un message sur une queue, le traiter, et écrire un message résultant sur une queue en aval, tout cela au sein d'une transaction, et donc avec la garantie de ne pas perdre de message si elle est interrompue entre la lecture et l'écriture.
- Enfin, de la même manière, une application qui doit réceptionner plusieurs messages avant d'effectuer un traitement, peut réunir ces lectures en une même transaction. Si le traitement réussit, le programme client exécute un *commit*, sinon, il demande un *rollback*, c'est-à-dire qu'il ordonne au broker de messages de tout annuler.

MOM & JMS, Didier Donsez, 1998-2012 - F. Baude 2014

35

27/04/2022

Acteurs et Produits MOM « incontournables » pour l'entreprise

- BEA MessageQ
- IBM - MQ Series
 - 25 plateformes
- MicroSoft - MSMQ (Message Queue Server)
 - essentiellement NT
- Level 8 Systems - Falcom MQ
 - passerelle vers MSMQ et MQ Series
- Sybase - DBQ
 - Adaptive Serveur
- Tibco - TIB/RendezVous
 - accord avec Oracle pour Oracle 8
- JSR Java Messaging Service (v 2.0 en mai 2013)
 - API pour les MQ, implem. très nombreuses: J2EE, Glassfish OpenMQ,...
- Apache ActiveMQ/Apollo
 - support de AMQP, STOMP, groupes de msgs avec délivrance ordonnée

MOM & JMS, Didier Donsez, 1998-2012 - F. Baudé 2014

38

27/04/2022

Enterprise Service Bus (ESB)

- Cible l'EAI:
 - intégration d'applications orientées service, hétérogènes, nécessitant de la transformation des messages
- Event Driven SOA: ED-SOA
 - Messaging asynchrone entre applications
- API de l'EAI
 - Ex: JBI (Java Business Integration)
- Plateformes ESB
 - PetalsLink Petals, Apache ServiceMix, Mulesoft Mule, Apache Camel, OpenESB, jBoss ESB...
 - Reposent sur des MOMs sous jacents, pour la fonction ED-SOA
 - Comment: « Apache ActiveMQ is a JMS provider. By using Apache Camel you get a one-stop-shopping solution for message oriented middleware (MOM) solutions: use the camel-jms connector to connect to JMS compliant broker. »

MOM & JMS, Didier Donsez, 1998-2012

39

39

27/04/2022

IBM MQSeries/WebSphere MQ

- Leader du marché (66% du marché)
- Plates-formes
 - >20 plates-formes
 - 5 protocoles réseaux
 - langages (C++, C, Cobol, Java, PL/1, ...)
- Nombreux modules
 - Publish/Subscribe, Workflow, ...
 - assured one-time delivery of messages across a wide variety of platforms
 - It implements the [Java Message Service](#) (JMS) standard API, and also has its own proprietary API, known as the Message Queuing Interface (MQI)

MQI & JMS, Didier Donsez, 1998-2012 - F.

40

40

27/04/2022

MSMQ (MicroSoft Message Queue)

- Plates-formes NT/2000 (v2) et XP (v3)
 - Réseaux IP et IPX
 - IP Multicast (avec PGM pour la tolérance aux pertes) (v3)
 - Transport sur HTTP/HTTPS et message à enveloppe SOAP (v3)
- Modèles (v3)
 - One-To-One, One-To-Many
 - Distribution Lists
 - Real-Time Messaging Multicast
 - Message Queuing Triggers
 - (activation d'une méthode d'un objet COM sur reception)
 - SDK MSMQ pour C, C++, ActiveX, MSMQ Explorer
 - API MSMQ dans .NET

MQI & JMS, Didier Donsez, 1998-2012

41

41

27/04/2022

MSMQ (MicroSoft Message Queue)

- **Serveur (v2)**
 - 4 types de serveur
 - PEC pour Primary Enterprise Controller
 - informations sur la topologie (sites, liaisons entre sites et RC)
 - PSC pour Primary Site Controller
 - informations sur les sites (serveurs, clients et files d'attente)
 - BSC pour Backup Site Controller
 - secours et équilibrage de charge de PSC
 - RS pour Routing Server
 - MSMQ Information Store (MQIS)
 - référentiel (utilise SQL Server ou Active Directory)
 - Dépôt transactionnel de message (MTS)
 - 2 Go par file (v2), 1 To par queue (v3)
- **Client**
 - Windows CE, Win9x, ...

MOM & JMS, Didier Donsez, 1998-2012

42

42

27/04/2022

MSMQ (MicroSoft Message Queue) et WCF

- **MSMQ est un Binding possible pour transport des messages de WCF**
 - Choisir: **NetMsmqBinding**
 - Assure la délivrance des messages WCF: ordonnée, transactionnelle, etc, et leur persistance
 - "The key differentiator is that messages are placed into queues to await processing, rather than sent directly into the WCF runtime."
<http://www.devx.com/architect/Article/41058>
 - Autre choix de binding possible dans WCF:
 - Demander utilisation d'un ReliableSessionBinding qui implémente la spécification WS-ReliableMessaging
 - "WS-RM is designed to control reliable delivery of single SOAP messages or sequences of SOAP messages between two endpoints, irrespective of how these endpoints are connected"
 - Reproduit en quelque sorte le fonctionnement de TCP en multi-hop->session
 - "WS-ReliableMessaging is a transfer protocol and it says nothing about what should happen to the message after it is successfully received by the other side. It does not provide durability to messages"
https://msdn.microsoft.com/en-us/library/aa480191.aspx#introtowcfreliablemessaging_topic5

MOM & JMS, Didier Donsez, 1998-2012 – Flaudie 2014

43

43

27/04/2022

Exemple d 'ASP utilisant MSMQ et MTS

```
<%@ TRANSACTION=REQUIRED LANGUAGE=JScript %>
<HTML><HEAD><TITLE>Envoi transactionnel par MSMQ</TITLE></HEAD><BODY>
<h1>Envoi transactionnel par MSMQ</h1><hr>
<%
QueueInfo = Server.CreateObject("MSMQ.MSMQueueInfo")
QueueInfo.pathname = ".\MS_SDK_TRANSACTIONED";
Queue = QueueInfo.Open(2, 0);
Msg = Server.CreateObject("MSMQ.MSMQueueMessage");
Msg.body = "Corps du Message";    Msg.Label = "Label du Message";
Msg.Delivery = 1;                // recouvrable : résiste au crash et au shutdown
Msg.PrivLevel = 1;               // chiffré
Msg.Send(Queue);
Queue.Close();
%>
</BODY></HTML>
<%
function OnTransactionCommit() {
    Response.Write("<p>La transaction est validée et le message MSMQ est envoyé."); }

function OnTransactionAbort() {
    Response.Write("<p>La transaction est abandonnée");
    Response.Write("<p>et le message MSMQ n 'a pas été envoyé."); } %>
```

44

44

27/04/2022

JoramMQ (ObjectWeb & Scalagent (Grenoble))

- MOM à la spec. JMS implémenté p.ex sur AMQP
- Destination (au sens JMS) : PtoP (Queue) et PubSub (Topic)
- Architecture Multi-Serveurs
- Open Source
- Intégré à ESB JONAS
- Disponibilité sur OSGi pour déployer des bundles OSGi
- Version kJORAM pour KVM
- Administration par des MBeans (Console JMX)
- Utilisation
 - Kelkoo (remontée de log)
 - Schneider Electric (remontée de mesures de capteurs)
 - ...

45

45

27/04/2022

JORAM (ObjectWeb & Scalagent)

■ Architecture Multi-Serveurs

- Une destination par serveur
 - La ConnectionFactory JMS est connecté au serveur
 - Equilibrage de charge (*Load Balancing*)
 - La Destination est répliquée sur R serveurs (pair à pair)
 - Connections: TCP, HTTP, SSL, ...
 - Privilégie la consommation locale des messages
 - Pas d'ordre global des messages
 - Ordre local
- Haute disponibilité (*High Availability*)
 - Serveur maître répliquant (avec JGroup) ses queues/topics sur S serveurs esclaves ($S > 0$)
 - La ConnectionFactory du client JMS peut basculer du serveur maître vers un des serveurs esclaves

MOM & JMS, Didier Donsez, 1998-2012

46

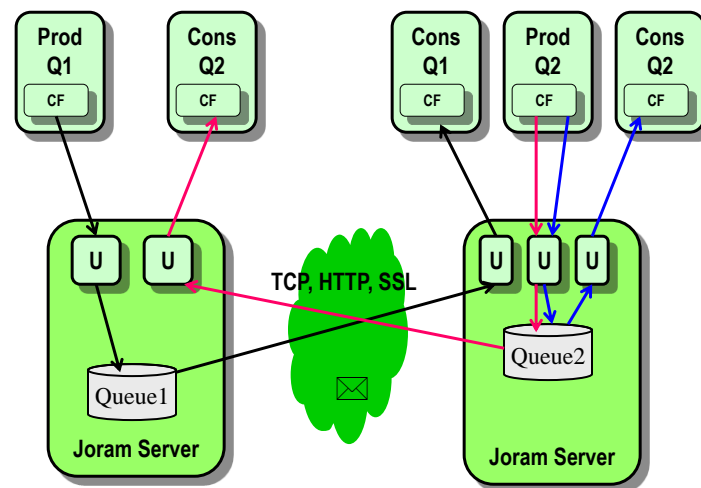
46

27/04/2022

JORAM (ObjectWeb & Scalagent)

Architecture Multi-Serveurs

■ Une destination par serveur



MOM & JMS, Didier Donsez, 1998-2012

47

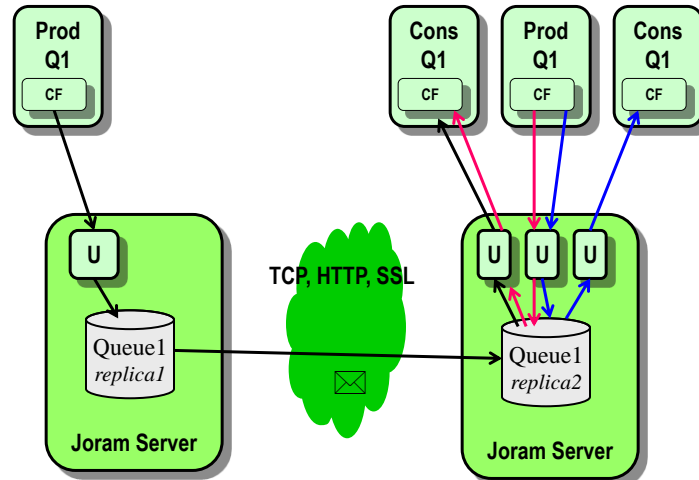
47

27/04/2022

JORAM (ObjectWeb & Scalagent) Architecture Multi-Serveurs

- Equilibrage de la charge telle que perçue par clients

MOM & JMS, Didier Donsez, 1998-2012



48

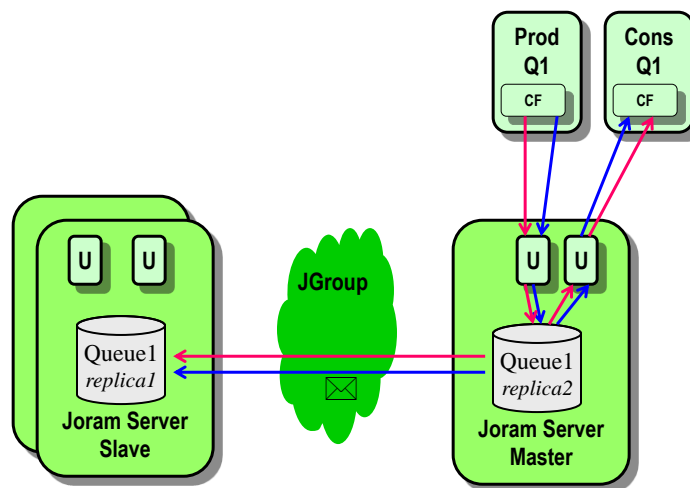
48

27/04/2022

JORAM (ObjectWeb & Scalagent) Architecture Multi-Serveurs

- Haute disponibilité (1)

MOM & JMS, Didier Donsez, 1998-2012



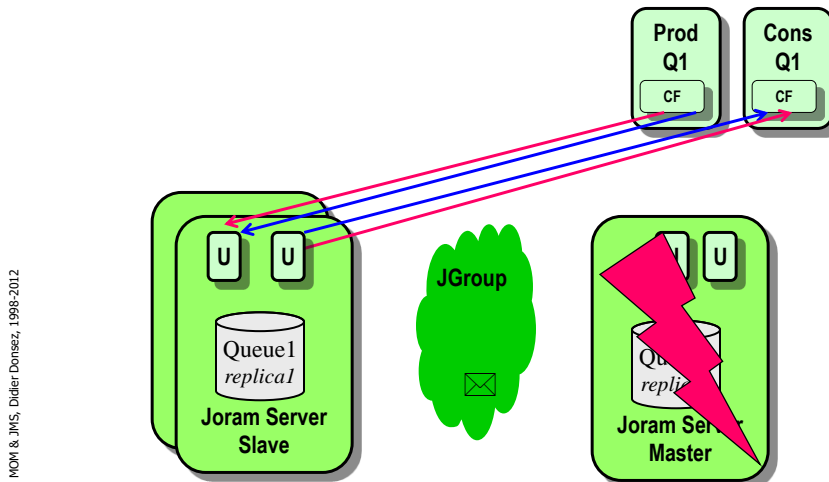
49

49

27/04/2022

JORAM (ObjectWeb & Scalagent) Architecture Multi-Serveurs

■ Haute disponibilité (2)



50

50

27/04/2022

MOM et Composants

■ Motivations

- Fournir la couche de communication pour le paradigme Événement (=message/event reçu en asynchrone) dans des modèles à composants (qui le supportent)

■ Modèles

- CORBA CCM (Corba Component Model)
- .NET Asynchronous [OneWay] calls
- J2EE/EJB Message Driven Beans (pas de typage des msg)

MOM & JMS, Didier Donsez, 1998-2012

54

54

27/04/2022

Bibliographie

- Gregor Hohpe, Enterprise Integration Patterns, <http://www.enterpriseintegrationpatterns.com>
- Très bon livre traitant de l'utilisation des MOMs
- <http://middleware.smile.fr/Middleware-orientes-messages>
- Site intéressant

MOM & JMS, Didier Donsez, 1998-2012

55