Patrons d'Integració d'Aplicacions

- Patrons d'integració:
  - Decisions a prendre
  - Classificació: (File Transfer, Shared Database, Remote Procedure Call, Messaging)
- Elements de la integració amb missatges
- Exemples de patrons d'integració amb missatges
  - **Canal**: Point to Point, Publish-Subscribe, Datatype, Bus
  - **Missatge**: Command/Document/Event, Request-Reply
  - **Router**: Content-Based Router, Message Filter, Splitter/ Aggregator, Process manager, Broker
  - **Translator**: Content enricher, Cannonical Data Model
  - **Endpoint**: Message Gateway, Polling Consumer, Event-Driven consumer, Service Activator
- Bibliografia

Patrons Integració: Decisions a prendre

1. **Necessito integrar** dos o més aplicacions? No ho puc fer en una sola, independent que no hagi de col·laborar amb altres?
2. **Nivell d'acoblament**. La interfície per a la integració ha de ser prou específica com per implementar la funcionalitat útil, però prou general com per permetre canviar la implementació quan sigui necessari.
3. **Simplicitat**. Els desenvolupadors volen reduir al màxim canvis a la aplicació i reduir codi d'integració. Però, els canvis i el nou codi és necessari per proporcionar la funcionalitat d'una bona integració.
4. **Tecnologia**. Diferents tècniques d'integració requereixen una quantitat considerable de software especialitzat (middleware).
5. **Format de dades**. Aplicacions integrades han de estar d'acord en el format de les dades que es transmeten i tenir funcions per adaptar aquests formats.
6. **Puntualitat**. Dades transmeses han de sortir i arribar puntuals al destí. Paquets de transmissió petits. Avisos de dades disponibles.
7. **Dades o funcionalitats compartides**. Cal decidir si es comparteixen solament dades o funcionalitats entre aplicacions.
8. **Asincronicitat**. Cap aplicació espera a cap altre. Això afecta al disseny.

Patrons Integració: Classificació

- Existeixen diferents enfocaments d'integració. Cada un considera els anteriors criteris de forma diferent.
- Els enfocaments d'integració segueixen algun d'aquests quatre patrons o estils bàsics d'integració:

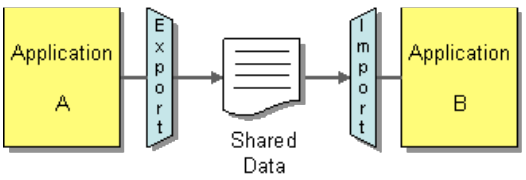
- **File Transfer**: Uns produeixen fitxers amb dades compartides que altres consumeixen, i a l'inrevés.
- **Shared Database**: les dades compartides es guarden a una base de dades comuna.
- **Remote Procedure Call**: aplicacions ofereixen els seus procediments per a ser invocats pels altres. Les dades i comportament es comparteixen al realitzar aquestes invocacions.
- **Messaging**; cada aplicació es connecta a un sistema de missatgeria comú i l'intercanvi de dades i de comportament es realitza per mitjà de missatges.

- Per a integrar aplicacions, els patrons específics a usar poden combinar aquests estils.

Patrons Integració: File Transfer

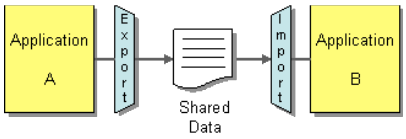
- Tenir un mecanisme comú de transferència de dades per diferents llenguatges de programació i plataformes amb poc software addicional.

- Les aplicacions produeixen fitxers que contenen les dades que altres aplicacions han de consumir. Integradors han de transformar aquests fitxers en diferents formats per a facilitar el seu consum. Producció de fitxers a intervals regular.



Patrons Integració: File Transfer

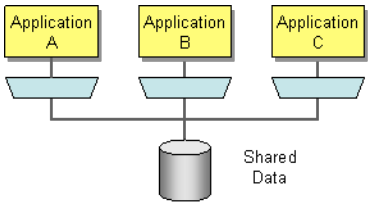
- **Format**: Diversitat de formats possibles. XML és una aposta d'unificació o d'estandardització de formats.
- **Interval producció**: Producció i transformació té un cost. Produir-los no sota demanda de consum, sinó amb periodicitat concreta segons necessitat de consum i flux de negoci (facilita asincronicitat).
- **Baix acoblament entre aplicacions**: Aplicacions i integradors sols coneixen dades i formats, i res sobre funcionament d'altres aplicacions.
- **Interfície pública**: Fitxers son la interfície pública de cada aplicació.
- **Gestió fitxers**: cal consensuar estil de nombrar fitxers, repositori, caducitat, estratègia d'esborrat (qui, quan, ...), gestió concurrència, actualització. ...
- **Actualitat**: Si dades canvien freqüentment, cal produir sovint noves versions dels fitxers, el que comporta un cost (potser prohibitiu).



Patrons Integració: Shared Database

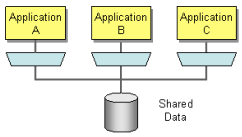
- Les dades compartides poden ser actualitzades amb freqüència i per totes les aplicacions. La compartició amb fitxers no por assegurar tenir la darrera versió de dades ni detectar inconsistències entre versions de fitxers.
- En cas de ambigüitat semàntica (diferent interpretació d'un concepte per les aplicacions), la compartició de dades via fitxers no resol aquesta ambigüitat.
- Cal un repositori de dades central, comú, consensuat i compartit entre totes les aplicacions on totes les aplicacions accedeixen.

- Integrar aplicacions que emmagatzemen les dades en una Base de Dades compartida amb un esquema comú que contempla les necesitats i interpretacions de totes les aplicacions.



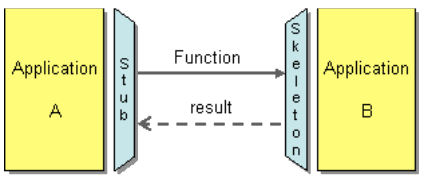
Patrons Integració: Shared Database

- **Actualització**: Les dades a la Base de Dades sempre estan actualitzades i son consistents.
- **Llenguatge i Formats**: Es disposa d'un únic format (esquema relacional) i un únic llenguatge de manipulació (SQL).
- **Ambigüitat semàntica**: Inexistent al ser un esquema consensuat i comú.
- **Esquema consensuat**: En certs casos dificultat de consensuar un esquema comú que sigui eficient i fàcil d'usar per cada aplicació.
- **Esquemes externs**: Necessitat de definir esquemes externs i les correspondències per no modificar aplicació internament.
- **Concurrència i bloqueigs**: Cal una gestió de concurrència per evitar colls d'ampolla i deadlocks. En el cas de bases de dades distribuïdes és més complex.



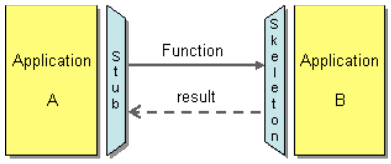
Patrons Integració: Remote Procedure Call (RPC)

- Canvis en les dades de la Base de Dades poden implicar la necessitat de reacció de les aplicacions. El poder de reacció de la BD és local a la mateixa, o com a molt, pot notificar el canvi a les aplicacions que la usen.
- Cal un mecanisme que permeti compartir funcionalitats entre aplicacions. Permetre a una aplicació accedir o modificar les dades o executar processos a altres aplicacions invocant funcions ofertes per l'aplicació propietària de les dades o processos.
- Aplicar el principi d'encapsulament per integrar aplicacions. Una aplicació passa les dades a compartir amb una altre aplicació, invocant-li la funció (remote procedure) que li indica com cal tractar les dades.



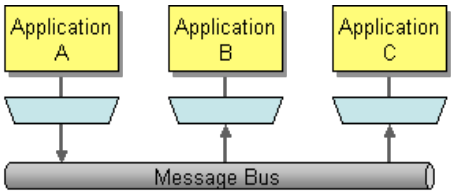
Patrons Integració: Remote Procedure Call (RPC)

- **Integritat interna**: Cada aplicació és responsable de les seves dades i processos, i és la única que les gestiona i actualitza.
- **Web-Services i XML**: Existeixen moltes plataformes que usen RPC (CORBA, COM, JAVA RMI, ...), però actualment els mecanismes més populars són: XML per passar dades i Web-Services per invocar funcions.
- **Ambigüitat semàntica**: diferents interfícies per accedir a les dades i per a invocar funcions de consulta i actualització. -> diferents punts de vista
- **Acoblament**: encapsulament redueix acoblament entre aplicacions respecte al coneixement del detall de les dades, però el acoblament és alt respecte a la crida de procediments (noms i paràmetres, seqüència de crides a fer, ...).
- **Sincronicitat**: RPCs entre aplicacions son lentes si moltes aplicacions integrades i produeixen alentiment general o aturada del sistema.



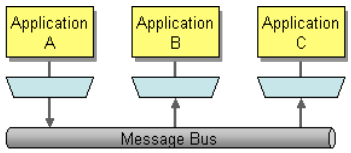
Patrons Integració: Messaging

- File Transfer i Shared Database permeten compartir dades, i Remote Procedure Call permet compartir funcionalitat però amb un cert acoblament.
- Cal un mecanisme que permeti crear paquets petits de dades i enviar-los asíncronament sense tenir que esperar la disponibilitat del receptor, i que el receptor sigui notificat quan el paquet està disponible per a ser consumit.
- La integració d'aplicacions es fa amb l'enviament de paquets de dades freqüentment, de manera fiable i de forma asíncrona, usant formats personalitzables.



Patrons Integració: Messaging

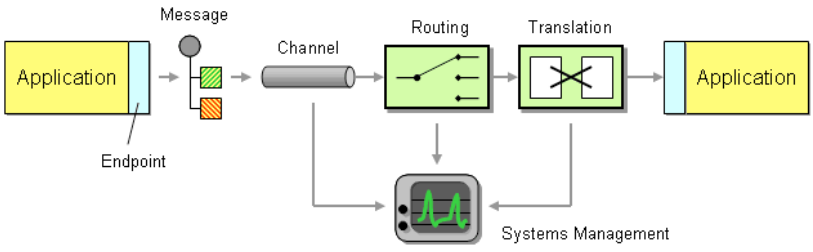
- **Asíncron**: l'emissor i receptors no han d'estar preparats per a comunicar-se.
- **Format dades**: L'enviament de dades amb missatges permet tenir mecanisme de transformació de formats durant la comunicació i alliberar a les aplicacions del consens en el format.
- **Encaminament**: els missatges s'envien al destinatari o a intermediaris que els re-direccionen convenientment.
- **Ambigüitat semàntica**: es delega a components (de tercers) addicionals al mecanisme de comunicació.
- **Missatges** petits i freqüents: amb l'enviament freqüent de missatges petits es potencia la col·laboració immediata i la compartició de dades.



Elements de la Integració mitjançant missatges

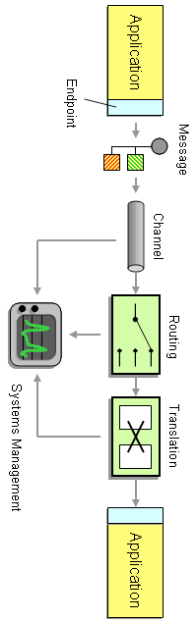
- La integració d'aplicacions es pot fer amb l'enviament de paquets de dades freqüentment, de manera fiable i de forma asíncrona, usant formats personalitzables

- La connexió entre dos aplicacions integrades es realitza usant elements que redueixen l'acoblament entre les dues aplicacions: el "middleware".



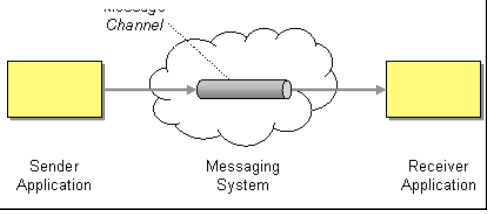
Elements de la Integració mitjançant missatges

- **Channel**: condueix la informació d'una aplicació a l'altre. És un conjunt de crides TCP/IP, una base de dades, un fitxer compartit ...
- **Message**: fragment de dades que viatge pel canal, el significat del qual està acordat per les dues aplicacions.
- **Translation**: etapa del procés en que es pot transformar el format de les dades del emissor en el format comprensible pel receptor. Es un transformador de formats de dades.
- **Routing**: component encarregat d'encaminar els missatges al/s destinatari/s adequat/s.
- **System Management**: degut a la diversitat de sistemes, plataformes, localitzacions, etc es requereix un component encarregat de monitoritzar la transmissió de dades, disponibilitat dels components i aplicacions.
- **Endpoint**: component encarregat d'adaptar i permetre a l'aplicació integrar-se al sistema de comunicació i esdevenir part del sistema integrat. És un Mediador o intermediari.



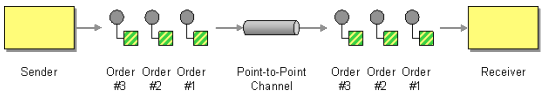
Messaging Channels

- El canal és usat per a connectar aplicacions.
- Recipient on una aplicació diposita la informació i d'on una altre l'obté.
- Canals específics per propòsits concrets o tipus d'informació.
- Decisions a prendre:
  - Nombre de canals fix (no es creen dinàmicament) i determinat per necessitats de les aplicacions
  - Canal unidireccional o bidireccional
  - Un-a-un o un-a-molts
  - Tipus de dades que viatgen pel canal
  - Tipus d'aplicacions proveïdores i consumidores de missatges
  - Canals per missatges erronis, morts, invàlids, ...
  - Persistents o no

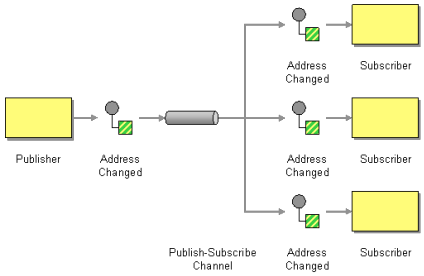


Messaging Channels: Patrons

- **Point-to-Point Channel**: assegura que solament un únic receptor rep el missatge, encara que hi hagi més d'un receptor potencial.
- Missatge consumit quan rebut i processat per un receptor.

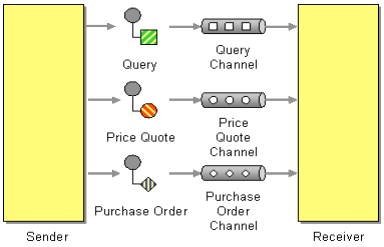


- **Publish-Subscriber Channel**: cada un dels receptors rep una còpia del missatge.
- Extensió del patró Observer.
- Missatge es consumit quan tots els receptors l'han rebut i processat.

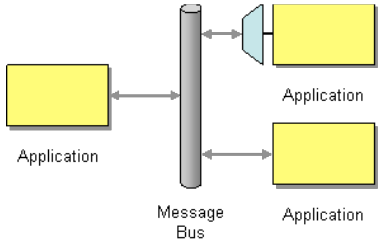


Messaging Channels: Patrons

- **Datatype Channel**: un canal específic per a cada tipus de missatge. Els missatge es deposita al canal segon el tipus de dades que conté.

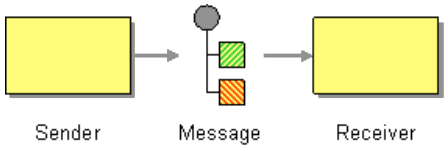


- **Bus Channel**: canal que permet a totes les aplicacions treballar i comunicar-se conjuntament.
- Infraestructura de comunicació comuna
- Adapters per a connectar-se al mecanisme de missatges
- Llenguatge de comandes comú



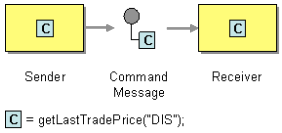
Message Construction

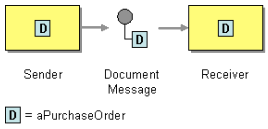
- Empaquetar la informació a transmetre en un missatge consistent en un registre de dades que es transmet a través del canal.
- Missatge té l'estructura següent:
  - Capçalera: indica dades que conté, origen, destí, etc.
  - Cos: dades, ignorades pel mecanisme de gestió de missatge i sols d'interès pel receptor
- Decisions a prendre:
  - Intenció pel que es genera el missatge
  - Tipus de contingut (comanda, document, esdeveniment, info control, ...)
  - Com es retorna la resposta (informació, identificador resposta, canal, ...)
  - Quantitat de dades a enviar (missatge simple, seqüencia missatges, ...)
  - Temps màxim espera per rebre resposta per si es perd



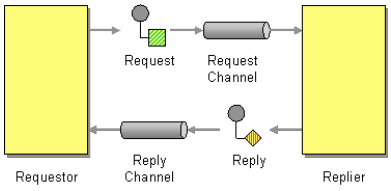
Message Construction: Patterns

- **Command/Document/Event Message**: el missatge conté un procediment/document/esdeveniment que el receptor ha d'executar/ processar/assabentar-se.



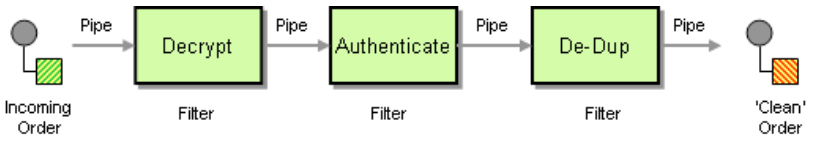


- **Request-Reply**: dos canals diferenciats per petició i resposta
- RPC i Confirmacions l'usen combinant:
  - Petició: Command o Event
  - Resposta: Document (resultat o ack)



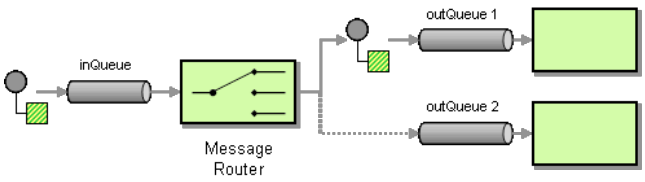
Message Pipes and Filters

- El patró arquitectònic Pipes and Filters s'usa per dividir una tasca llarga en subtasques més petites, executades en etapes independents (Filters) i connectades amb canals (Pipes).
- Etapes diferents de processament de missatges (filters) s'encadenen o es connectes via canals (filters).
- Permet organitzar el procés com a seqüències o com a threads executats en paral·lel.
- Pipe: rep un missatge pel canal d'entrada, el processa i el diposita al canal de sortida. No tenen estat. El processament d'un missatge és independents del següent i no cal esperar-se.
- Filter: fa de connector entre la sortida d'un pipe i l'entrada del següent.

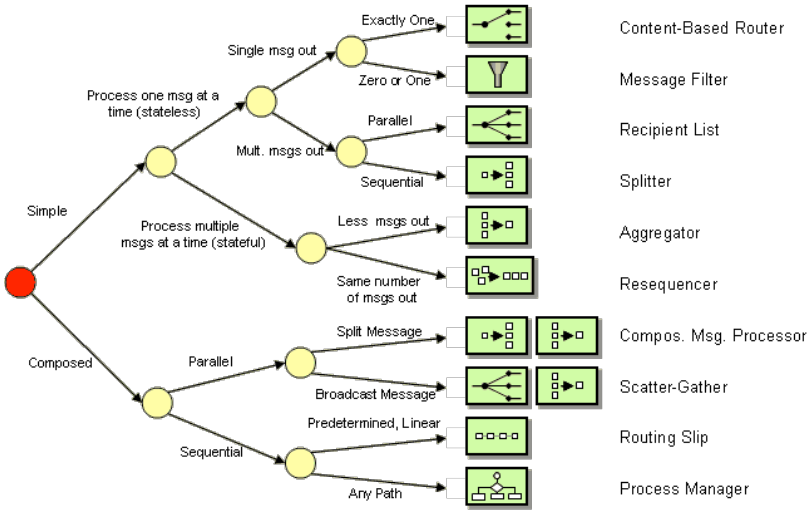


Message Routing

- El Router és un tipus especial de Filter que consumeix un missatge d'un canal (Pipe) d'entrada i el distribueix a un conjunt de canals (Pipes) de sortida. sota un conjunt de criteris o condicions que defineixen el comportament del Router
- Els criteris depenen de la informació del missatge:
  - Tipus dades que conté
  - Origen / Destinatari
  - ...
- El Router no modifica el missatge, solament l'encamina adequadament
- La lògica d'encaminament de missatges localitzat en un únic punt (Filter) en lloc de distribuït en diferents llocs (aplicacions, altres pipes, estructura de la xarxa amb molts canals especialitzats, ...)

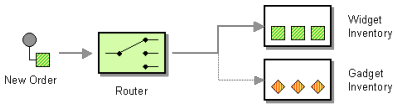


Message Routing:Patterns

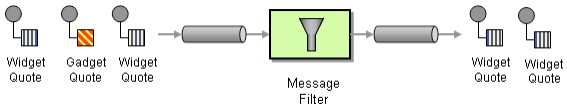


Message Routing:Patterns

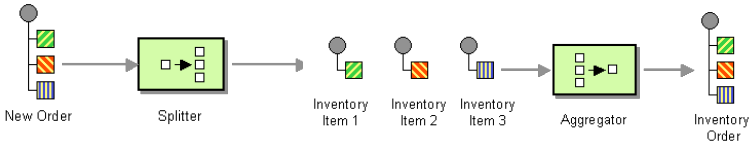
- **Content-based router**: el missatge s'encamina segons el seu contingut.



- **Message Filter**: s'eliminen els missatges segons un criteri definit

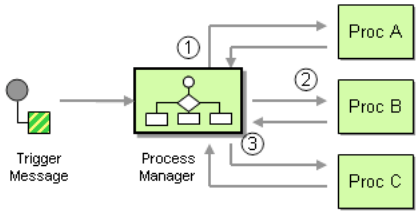


- **Splitter / Aggregator**: un missatge es descomposa en més missatges, o al revés, molts missatges s'agrupen en un de sol.

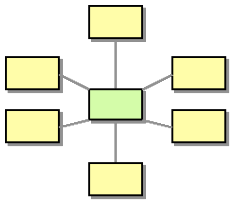


Message Routing:Patterns

- **Process Manager**: un missatge s'ha de encaminar a diferents destinataris determinats per un procés definit, però aquests depenen dels resultats intermedis de cada etapa del procés.

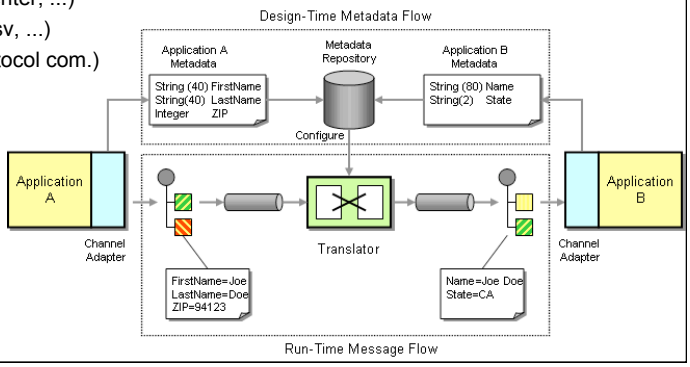


- **Message Broker**: el broker rep els missatges de diferents llocs i determina el destinatari, seleccionant el canal adequat per fer-li arribar.



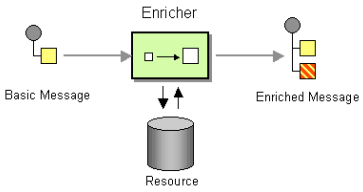
Message Transformation

- El Translator és un filter per a transformar el format de les dades del missatge a un format adequat al destinatari.
- La transformació ha de mantenir acoblament baix (evitar fer transformacions punt a punt).
- Transformació de dades:
  - Estructures (entitats, associacions, ...)
  - Tipus (string, enter, ...)
  - Sintaxi (xml, csv, ...)
  - Transport (protocol com.)

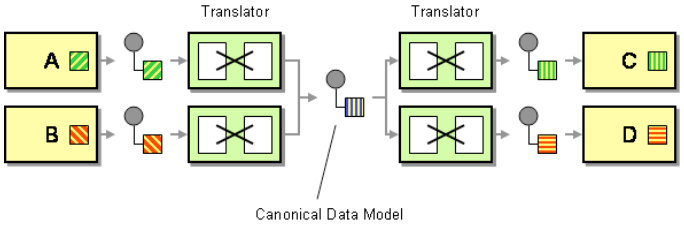


Message Transformation: Patterns

- **Content Enricher**: filter que pot accedir a una font externa o fer una transformació de les dades del missatge per tal d'afegir informació al mateix original requerida pel destinatari del missatge.

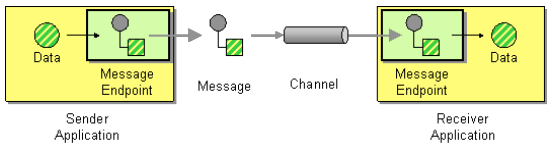


- **Canonical Data Model**: per minimitzar les dependències entre les aplicacions a integrar respecte al format de dades que comparteixen, aquest patró permet definir un format canònic de dades (independent de les aplicacions) i les transformacions de cada format específic de cada aplicació a aquest format canònic.



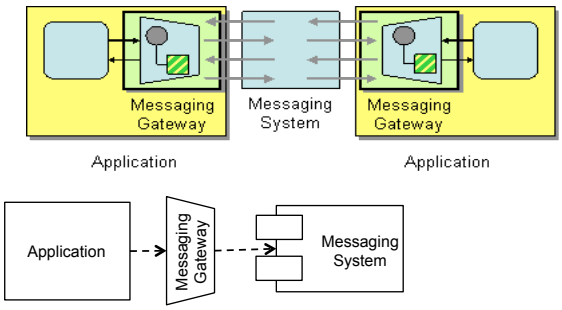
Messaging Endpoint

- Message Endpoint és un client del sistema de missatges que tota aplicació usa per enviar i rebre missatges.
- Es correspon a un Channel Adapter.
- Aquest component:
  - encapsula el sistema de missatgeria a la pròpia aplicació
  - transforma les dades de la aplicació al format de missatge o a l'inrevés
  - pot gestionar la recepció o emissió de missatges amb transaccions
  - pot gestionar els missatges de forma síncrona o asíncrona
  - els missatges es consumeixen/produeixen concurrentment o no
  - accepta tots els missatges o els filtra
  - com es consideren els missatges que arriben repetits
  - serveis de les aplicacions es poden invocar de forma síncrona o asíncrona



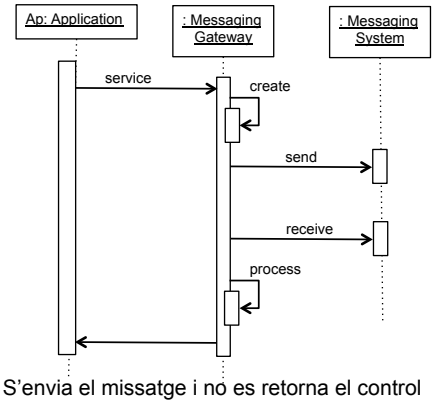
Messaging Endpoint: Patterns

- **Message Gateway**: encapsula el codi específic de generació, gestió i recepció de missatges i el separa de la resta de codi de l'aplicació.
- Ofereix funcionalitats de l'aplicació a la resta d'aplicacions a través d'una API.
- Permet que s'implementi el gateway usant la tecnologia d'integració que es vulgui (web-services, missatges, RPC, ...).



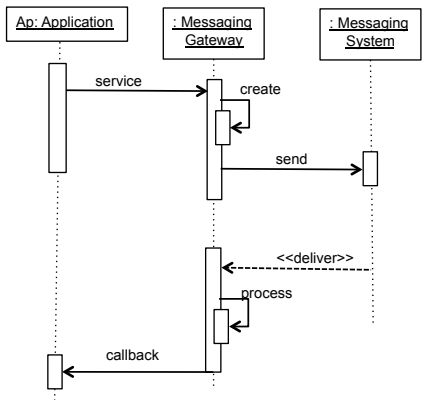
Messaging Endpoint: Patterns

Blocking Gateway



S'envia el missatge i no es retorna el control a l'aplicació fins que es rep el reply i es processa.
Encapsula l'asincronicitat gestionant la comunicació de forma síncrona.

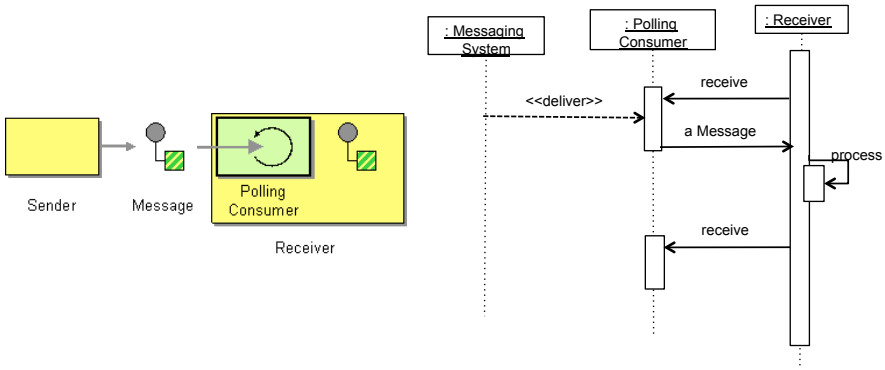
Event Driven



S'envia el missatge i es retorna el control a l'aplicació i es crida el callback quan es rep el reply i s'ha processat.
Exposa asincronicitat a l'aplicació.

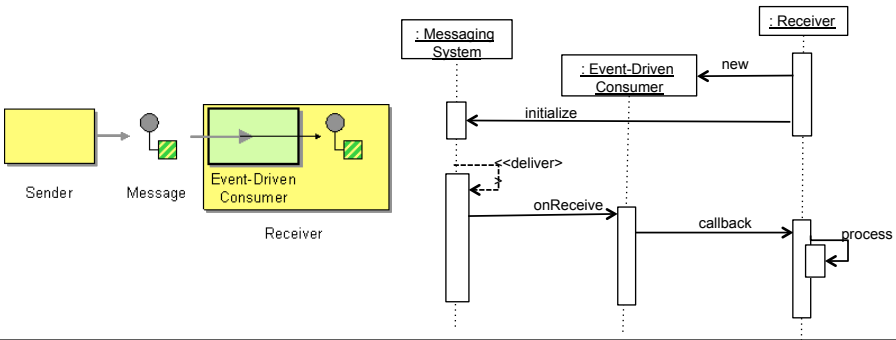
Messaging Endpoint: Patterns

- **Polling Consumer**: el consumidor pregunta constantment al sistema de missatges si hi ha missatges a processar.
- Quan el consumidor pot procesar un missatge, pregunta al sistema de missatges per si n'hi ha de disponibles. El processa i torna preguntar pel següent.



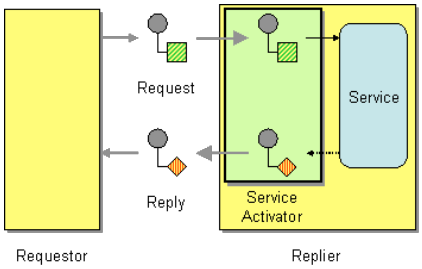
Messaging Endpoint: Patterns

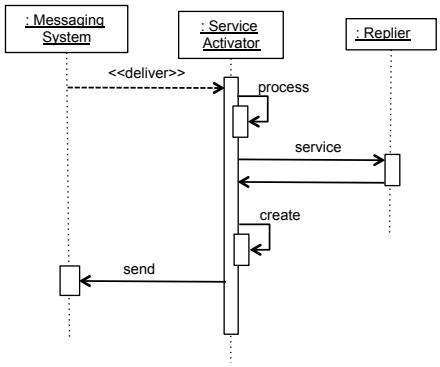
- **Event-Driven Consumer**: el consumidor rep una notificació del sistema de missatges quan té un missatge disponible per ser consumit.
- Inicialització: per a indicar que vol rebre notificacions de misstages disponibles.
- Consum: es rep el missatge i es passa a la aplicació perquè el processi.



Messaging Endpoint: Patterns

- **Service Activator**: connecta el sistema de missatgeria amb els serveis que han de ser invocats.
- Al rebre un missatge, el Service Activator, processa el missatge i invoca el servei de l'aplicació corresponent. La integració es fa via missatges, però la invocació del serveis és com si la fes directament el consumidor (requestor).





Bibliografia

- *Enterprise Integration Patterns:Designing, Building and Deploying Messaging Solutions*
  Hohpe, G.; Woolf, B.
  Addison-Wesley, *2003.*
- www.eaipatterns.com
- *The Making of Information Systems*
  Kurbel, K.E
  Springer Verlag, 2008, cap. 7.4