SOA (Service-Oriented Architecture)
Concepts et points de vue, infrastructure service-composants et outillage de développement Eclipse

Groupe SCOrWare

Alain Boulze, Mohammed El Jai, Claude Meynier, Adrian Mos, Vincent Zurczak

OW@INRIA – EBM Websourcing (Grenoble)

http://www.scorware.org

alain.boulze at inria.fr

mohammed.eljai at ebmwebsourcing.com

claude.meynier at ebmwebsourcing.com

adrian.mos at inria.fr

vincent.zurczak at ebmwebsourcing.com

Plan

- I SOA: présentations (matin)
- I-1Introduction à SOA
  - Concepts, éléments clé et standards
  - Point de vue ➔ conception de système d'information
  - Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI
- I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)
  - Introduction à SCA (et relations à JBI)
  - Relations à Fractal: Tinfi et la plateforme FraSCAti
  - Plateforme PEtALS / FraSCAti
- I-3 L'outillage de développement en environnement Eclipse
  - Le projet Eclipse STP
  - Focus sur les outils SCA
  - Interopérabilité entre outils: le modèle intermédiaire STP-IM
- II SOA: démonstrations (après-midi)
  - II-1 Cas d'étude « Voyage »
  - II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM
  - II-3 Mise en œuvre avec SCA
  - II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA

I-1 Introduction à SOA

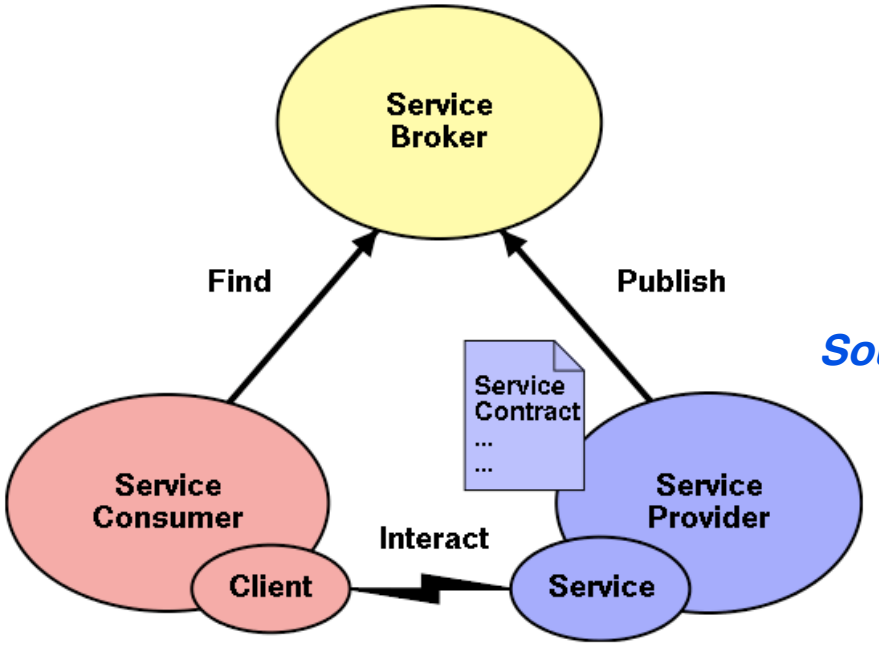
- I-1a Concepts, éléments clé et standards
  - SOA, paradigme (W3C) et approche
  - Concept de service
  - Eléments clé
    - WSDL, QoS et SLA, infrastructure de services, repository et gouvernance
    - Des standards: SCA, BPMN, WS-BPEL
    - Stacks SOA, marché et Open Source
- I-1b Point de vue conception du système d'information
  - Une démarche globale et agile
  - Collaboration entre fonctionnel et technique
  - Une collaboration avec le BPM (Business Process Modeling)
  - (cf. Démonstration II)
- I-1c Point de vue développement SCA autour d'une infrastructure ESB JBI
  - ESB
  - Standard JBI
  - (cf. Démonstration II)

Le paradigme SOA

- **Find, Publish (Register), Interact (Bind)**

W3C®

Service-Oriented Architecture



Source: w3c.org

SOA, une approche

- Une approche collaborative
  - Concevoir des solutions métier agiles et adaptées
- Orchestrer des services
  - Composer des processus
- Un mode de développement d'application
  - Réutiliser, construire, composer
- Un ensemble de principes architecturaux
  - Décrire la collaboration entre systèmes autonomes
  - SOA = « Interface-Oriented Architecture »
  - *OASIS SOA Reference Model*
- « Une abstraction qui encapsule des fonctions »

Abstraction logicielle, évolutions

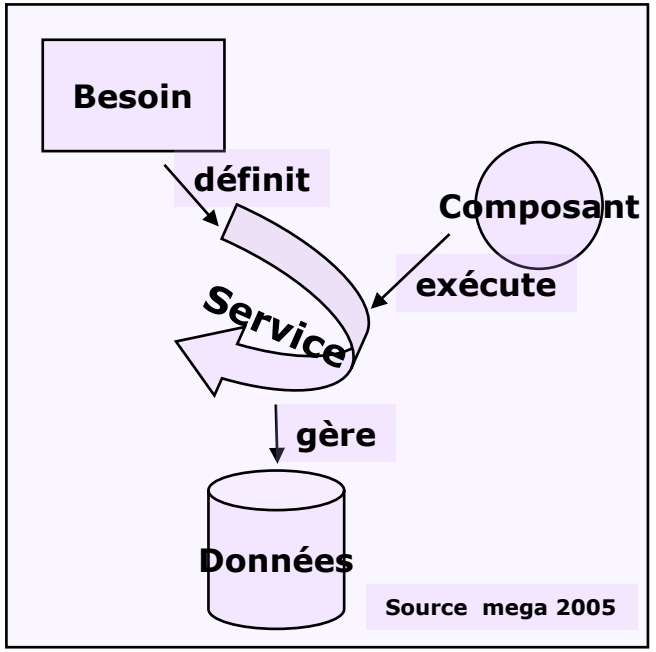
- Besoins de partage de concepts entre métier et IT
- Evolutions IT
  - Intégration
  - Adaptation
  - Configuration
- Evolutions de modèles de programmation
  - Fonctions et procédures
  - Modules
  - Objets
  - Composants et services

Concept de service

- **Abstraction / Encapsulation / Exposition**
  - Seule l'interface exposée d'un service est connue (publique)
    - Contrat / Interface
  - Ce que contient le service (connaissance, logique métier, implémentation) reste privée
  - Un service est réutilisable
- **Couplage faible**
  - Un service est orchestrable avec un autre service sans avoir besoin de connaître cet autre service
  - Echange avec son environnement via des messages
- **Orchestrable**
  - Un service est invoqué à travers une orchestration
  - Synchrone / asynchrone
  - Indépendant du protocole d'accès (SOAP/WS, JMS, MQ, Grid, …)

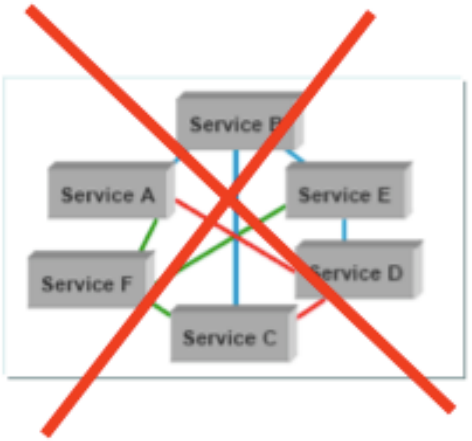
Service et réutilisation

- **Indépendant d'une technologie**
  - Java, C++, C#, Cobol
  - S'appuie sur des standards d'interopérabilité (WS-*, WSDL, …)
- **Réutilisable**
  - Registry, repository
  - Quelques standards (UDDI, ebXML)
  - Gouvernance SOA
    - Un service est défini avec l'intention d'être réutilisé



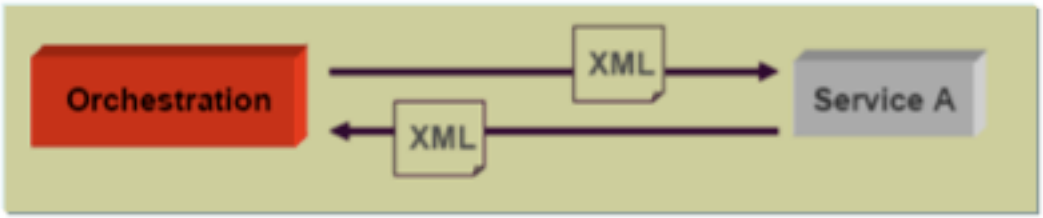
Service et couplage faible

- **Couplage faible**



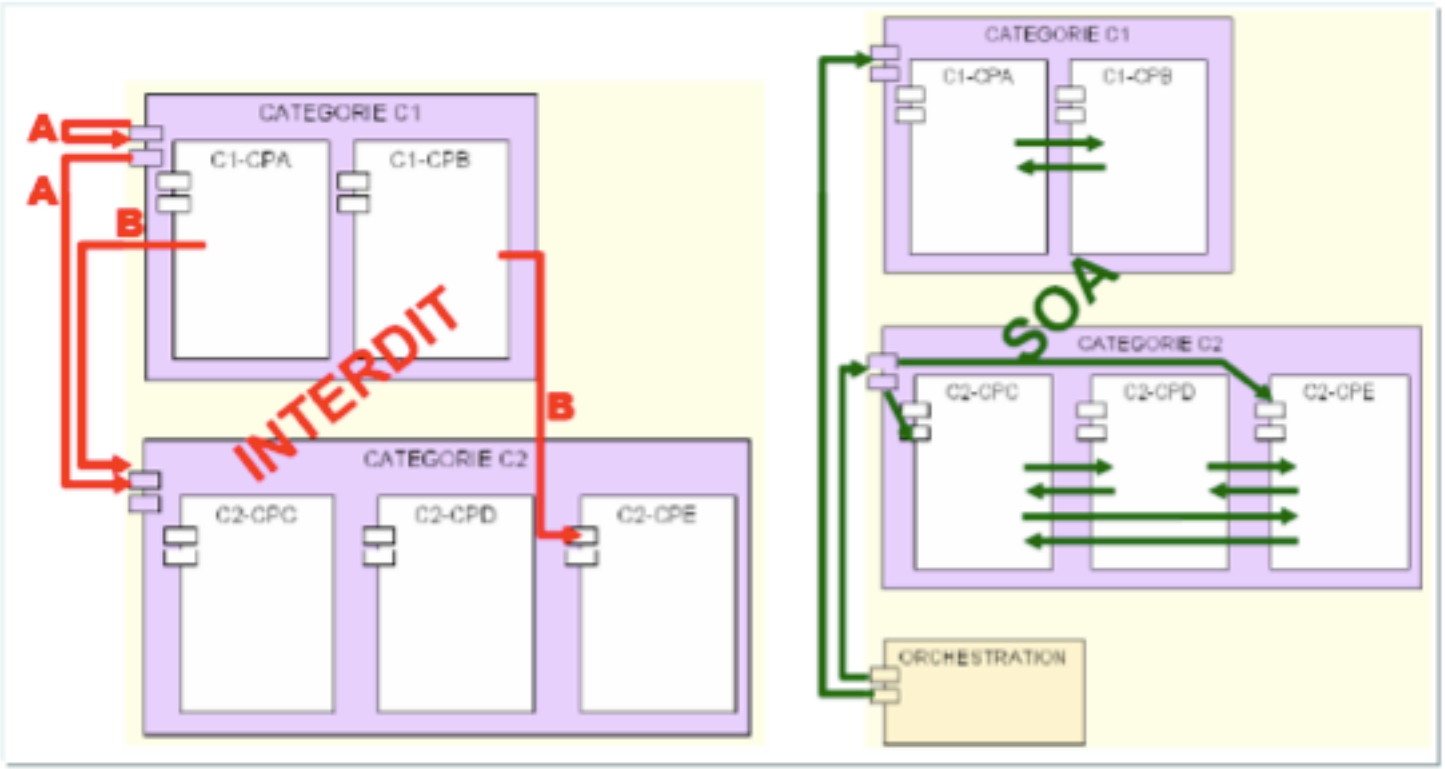
Not Good

Good



Service et orchestration

Orchestration, patron d'architecture

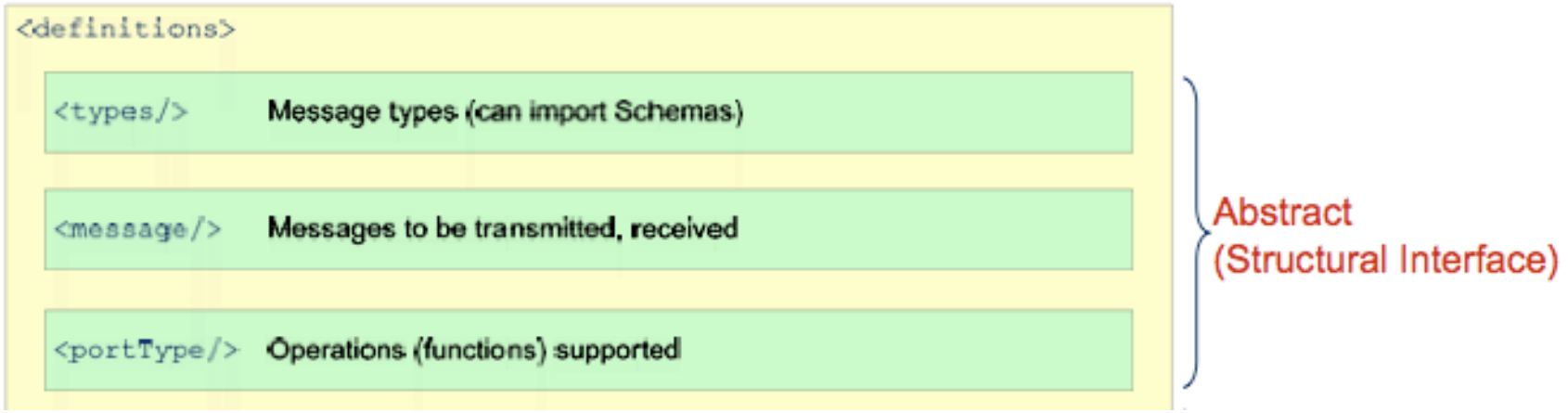


Eléments clé d'une SOA

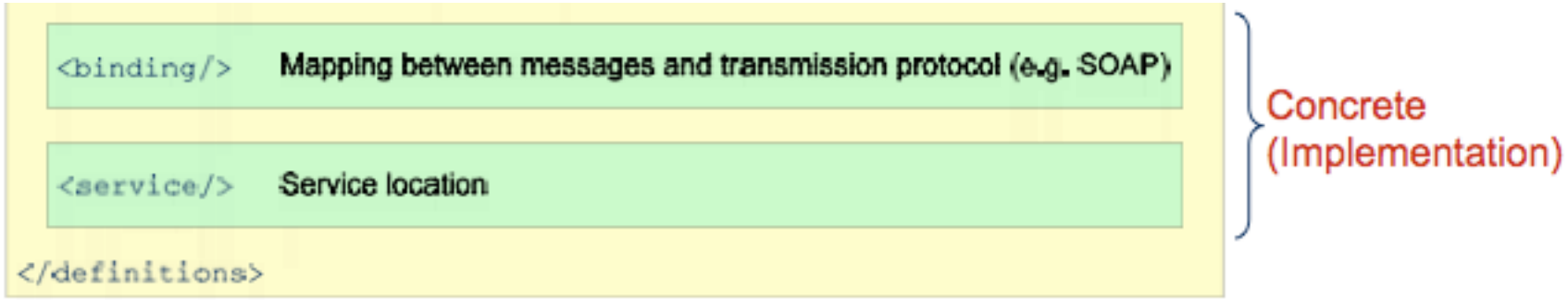
- **Interface et contrat**
  - WSDL (Web Service Description Language)
  - SLA (Service Level Agreement) / QoS (Quality of Service)
- **Des standards**
  - SCA (Service-Component Architecture)
  - BPMN (Business Process Modeling Notation)
  - WS-BPEL (Business Process Execution Language)
- **Référentiel de Services et Gouvernance SOA**
- **Stack SOA**
  - Offres propriétaires du marché
  - Open Source (projets en cours)

WSDL

- **Partie « abstraite »**



- **Partie « concrète »**

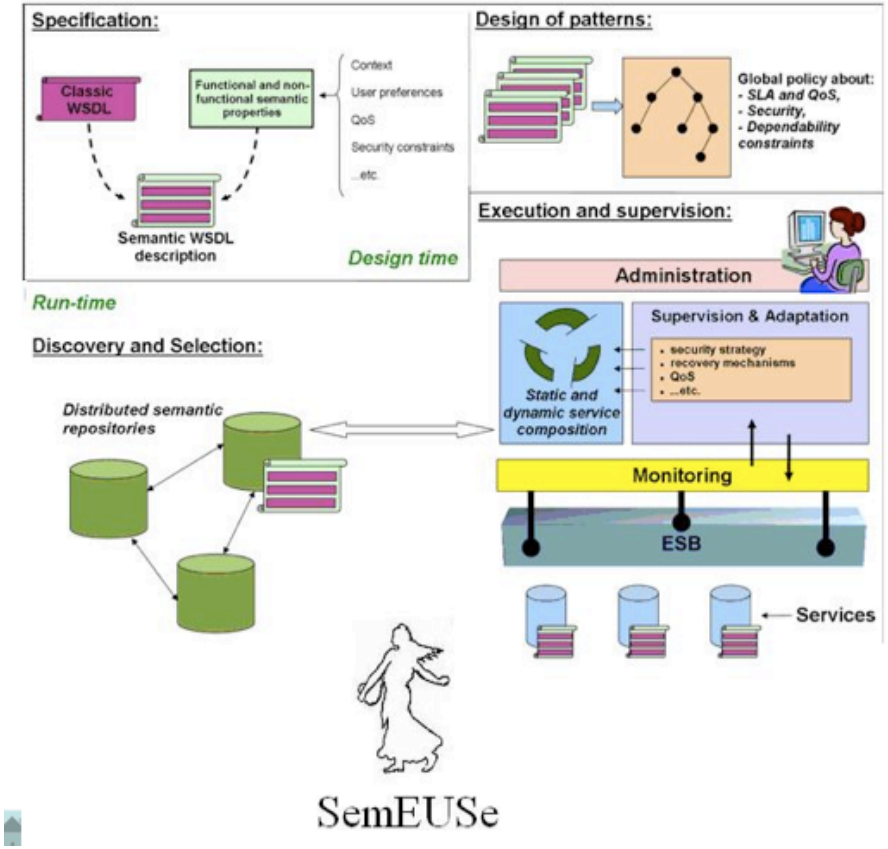


Gestion SLA (Service-Level Agreement) & QoS (Quality of Service)

- **Définition et négociation SLA au « design time »**
  - Spécification du contrat (XML, WS-Agreement)
  - SLOs (Service-Level Objectives) => exprime des niveaux de services observables et qualifiables
    - Ex: temps de réponse, disponibilité, performances
    - Fournisseur et consommateur
- **Supervision et application SLA/QoS au « run time »**
  - Superviser les ressources à différents niveaux d'abstraction (infrastructure, application, …)
  - Collecter des mesures, les agréger
  - Mécanismes d'application des propriétés QoS

SLA / QoS: exemple de mise en oeuvre

- **Projet ANR TL 2007 SemEUsE (2008-2010)**
  - **SEMantiquE pour bUS de sErvices**

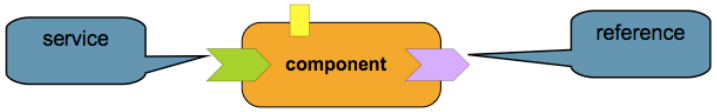


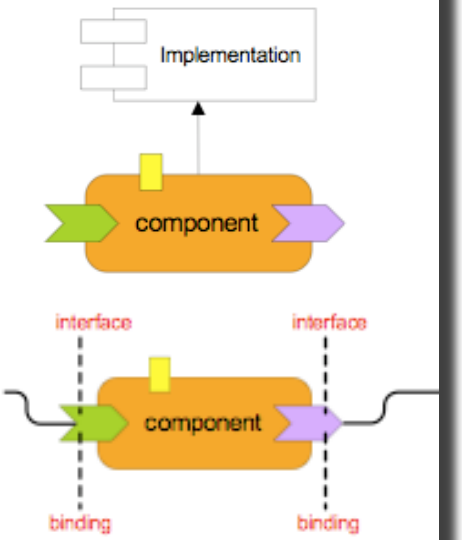
Standard SCA

- Initié en 2005 par une communauté industrielle
  - Open SOA (http://www.osoa.org)
  - Communauté élargie
- OASIS
  - http://www.oasis-open.org
  - SOA « technical work »
  - SCA « technical committees »
    - SCA-Assembly
    - SCA-Binding
    - SCA-BPEL, -C-C++, -J
    - SCA-Policy
  - Membres OASIS Open CSA
    - Composite Service Applications
- Présent aussi dans le JCP (Java Community Process)
  - http://www.jcp.org
  - JSR (Java Specification Request) 312 (JBI 2.0), JSR 316 (Java EE 6)

SCA (Service-Component Architecture)

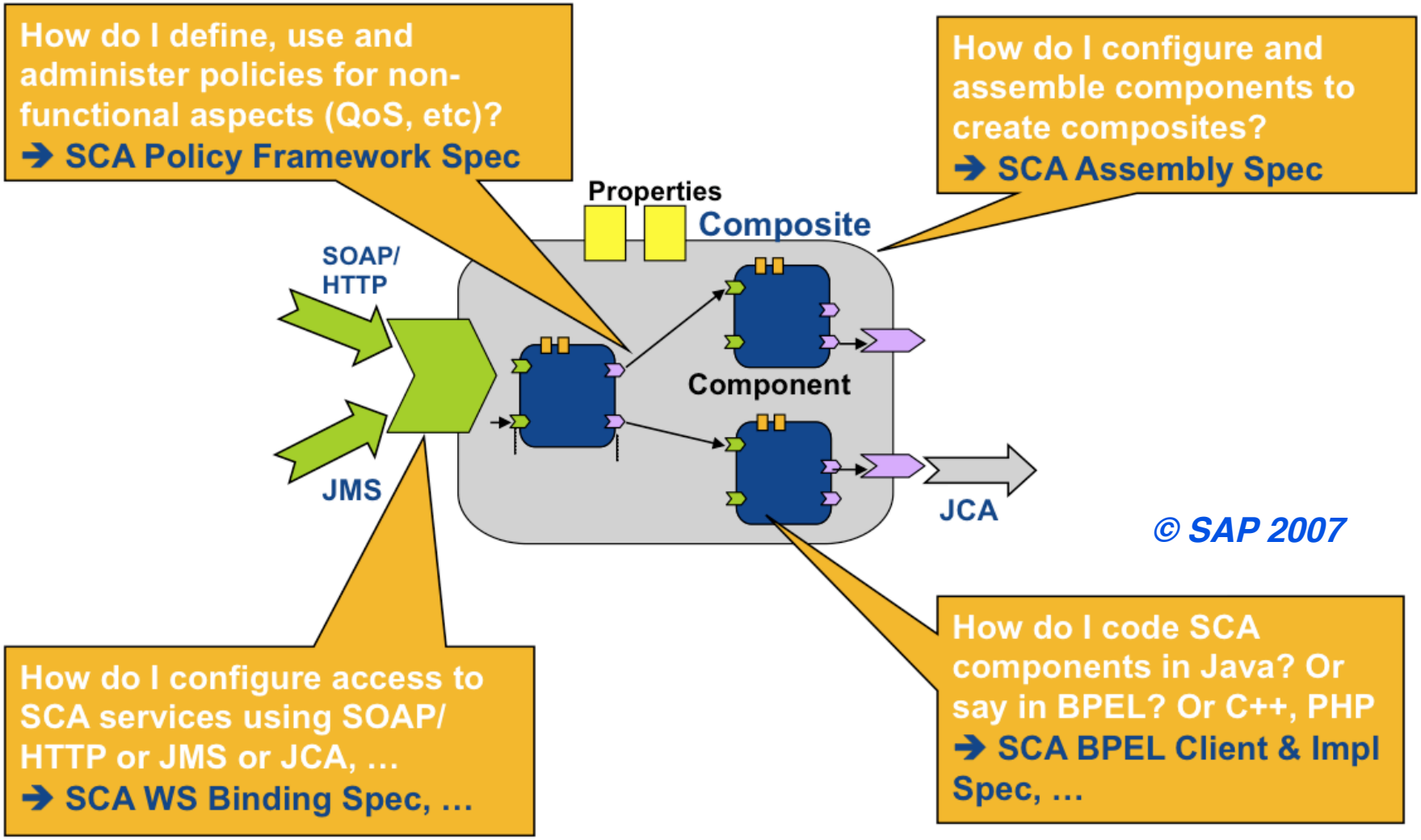
- **Modèle unifié déclaratif pour assembler des composants services dans des solutions métier**
- **Simplifie le développement SOA**
  - **Focus sur:**
    - ***services* offerts / utilisés par un composant**
    - ***references* utilisées par un composant**
  - **Technologies d'implémentation variées**
    - **Ex: EJBs, Java POJOs, BPEL, COBOL, C++, PHP, …**
  - **Technologies de communication variées**
    - **Ex: Web Services, Messaging/JMS, RMI-IIOP, ..**



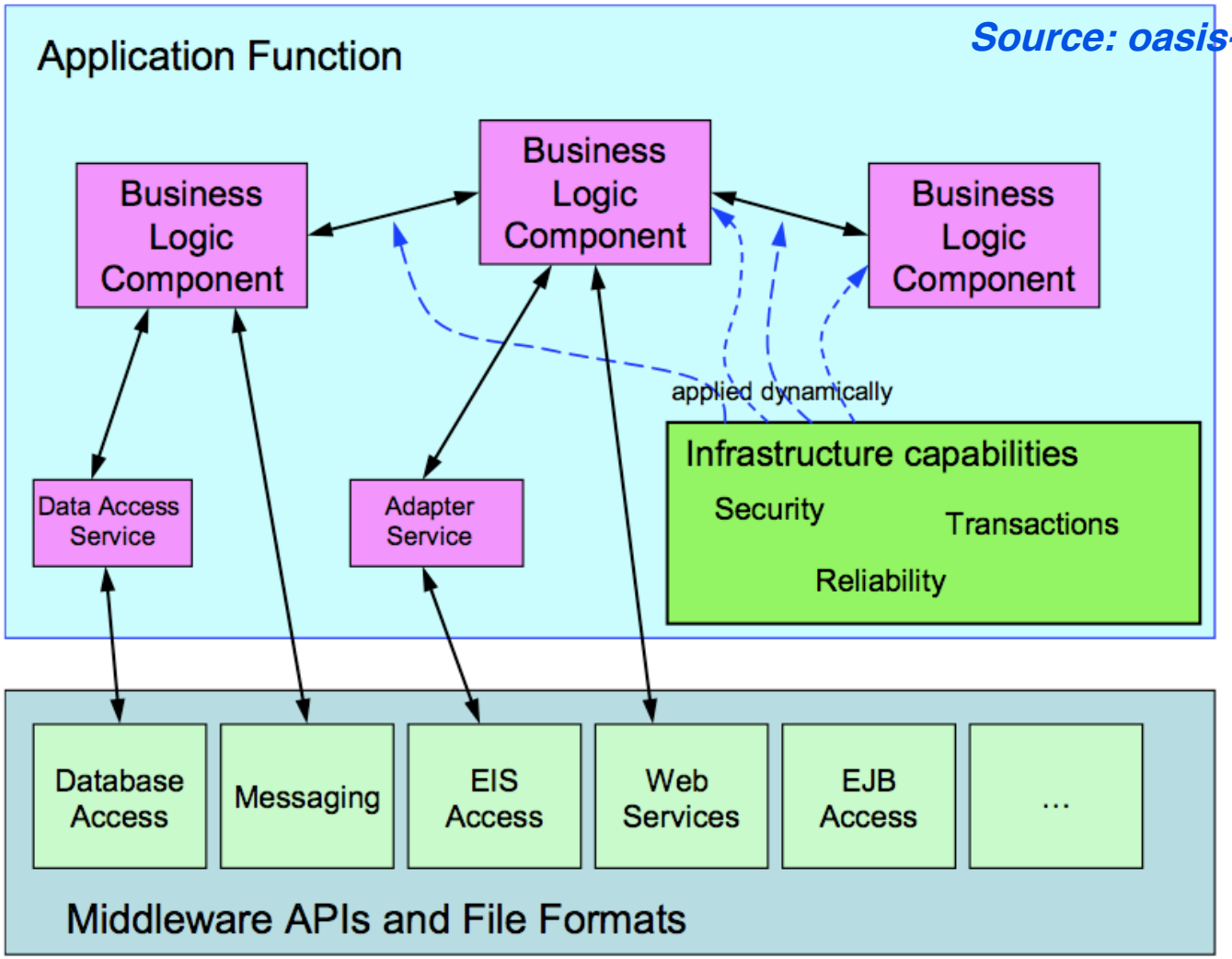


Sources: oasis-open.org, java.sun.com/javaone

SCA ➔ points clé



SCA, architecture applicative

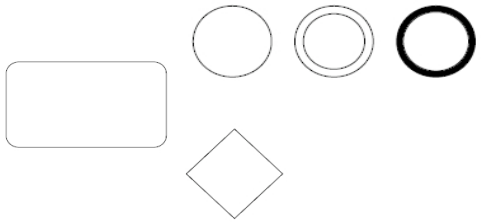


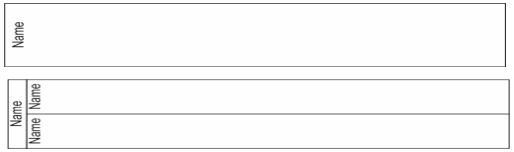
Standard BPMN

- BPMN = Business Process Modeling Notation
  - Standard OMG (Object Management Group)
  - La représentation est standard, la notation ne l'est pas
  - Modélisation autour de la notion de processus métier
  - Améliorer la communication entre les mondes “métier” et “technique”
- Modèle BPMN
  - Créer des modèles graphiques de processus métier
  - Modèle BPMN
    - Réseau d'objets graphiques où...
    - ... les objets représentent des activités...
    - ... qui interviennent dans le processus selon « l'agencement » représenté
- BPMN et UML
  - A l'origine, les diagrammes d'activité UML étaient utilisés
  - Pauvreté de ces diagrammes UML / métier ➔ BPMN
  - Similitudes dans les symboles

Objets BPMN

- **Objets BPMN**
  - Les « flow objects »
    - Les événements
    - Les activités
    - Les gateway (décisions, merge, etc...)
  - Les « swimlanes »
    - Les pools: participants dans un processus
    - Les lanes: organisent un pool
  - Les connexions
    - Les « sequence flow »: lient 2 « flow objects » dans un pool
    - Les « message flow »: lient 2 « flow objects » dans des pools différents
    - Les associations: attachent des éléments à des « flow objects »
  - Les artéfacts (éléments flexibles / celui qui modélise)

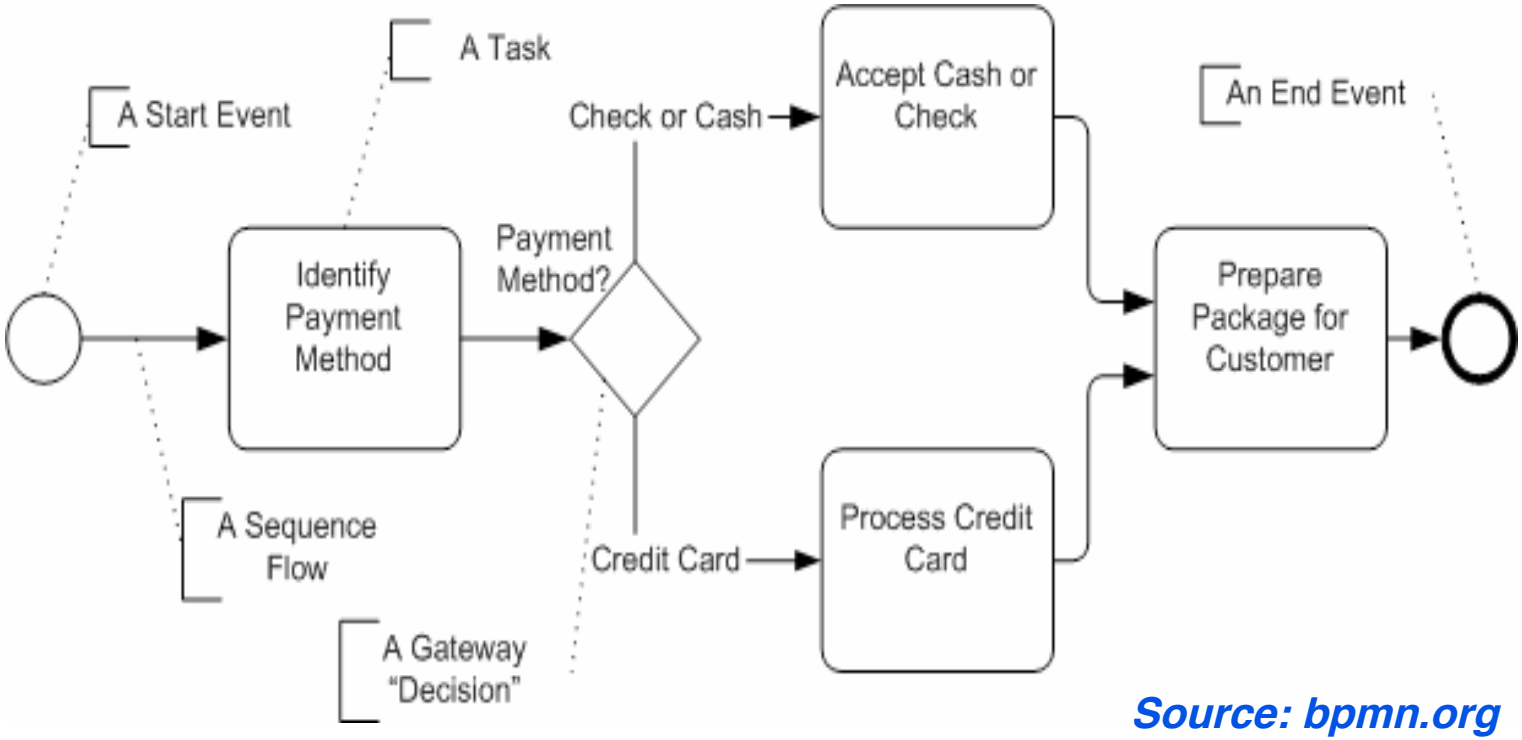




- **Evénements et gateways**
  - Distinction faite entre ce qui est « données » et « événements »
    - data-based / event-based

BPMN (Exemple)

- **Ex: « processus de paiement »**

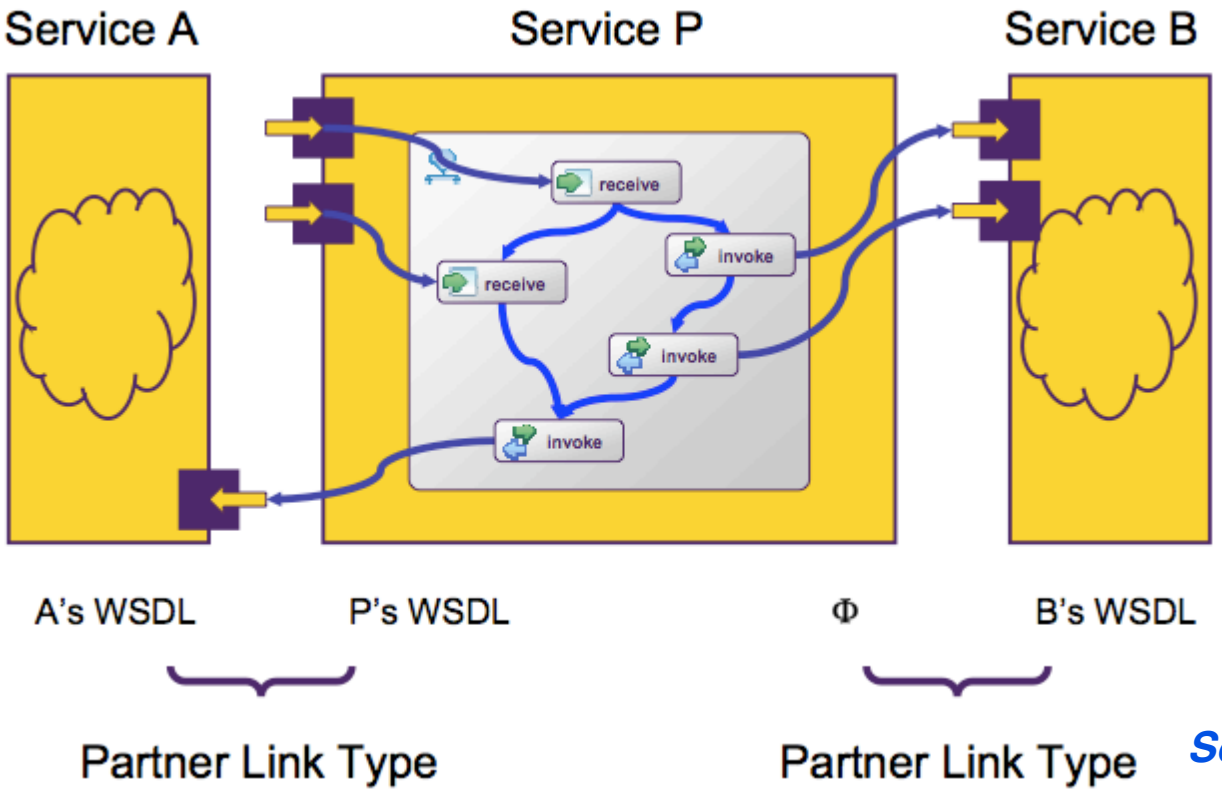


Source: bpmn.org

Standard WS-BPEL (2.0)

- **Standard OASIS**
  - http://www.oasis-open.org
- **Langage exécutable d'une séquence d'appels à des services**
  - Gère des interactions avec des services exposés (WSDL)
    - Composition / orchestration de services
  - Un BPEL est exposé lui-même en tant que service (WSDL)
  - Un langage de programmation utilisant des variables typées
    - Messages WSDL, éléments/types XML
  - Un langage de programmation
    - Assignation, while, if-then-else, exception handlers
  - Des fonctions spécifiques aux Web Services
    - Typage XML, Xpath/XSL, messages send/receive

WS-BPEL
Composition de Web Services



Source: oasis-open.org

WS-BPEL
Activités de base et structurées

Do a blocking wait for a matching message to arrive

Send a message in reply to a formerly received message

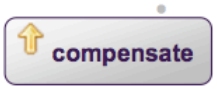
Invoke a one-way or request-response operation

Update the values of variables or partner links with new data

Generate a fault from inside the business process

Immediately terminate execution of a business process instance

Wait for a given time period or until a certain time has passed



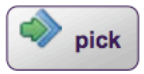
Invoke compensation on an inner scope that has already completed

Contained activities are executed in parallel

Contained activities are performed sequentially

Contained activity is repeated while a predicate holds

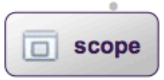
Contained activity is repeated until a predicate holds



Block and wait for a suitable message to arrive (or time out)

Contained activity is performed multiple times sequentially or concurrently

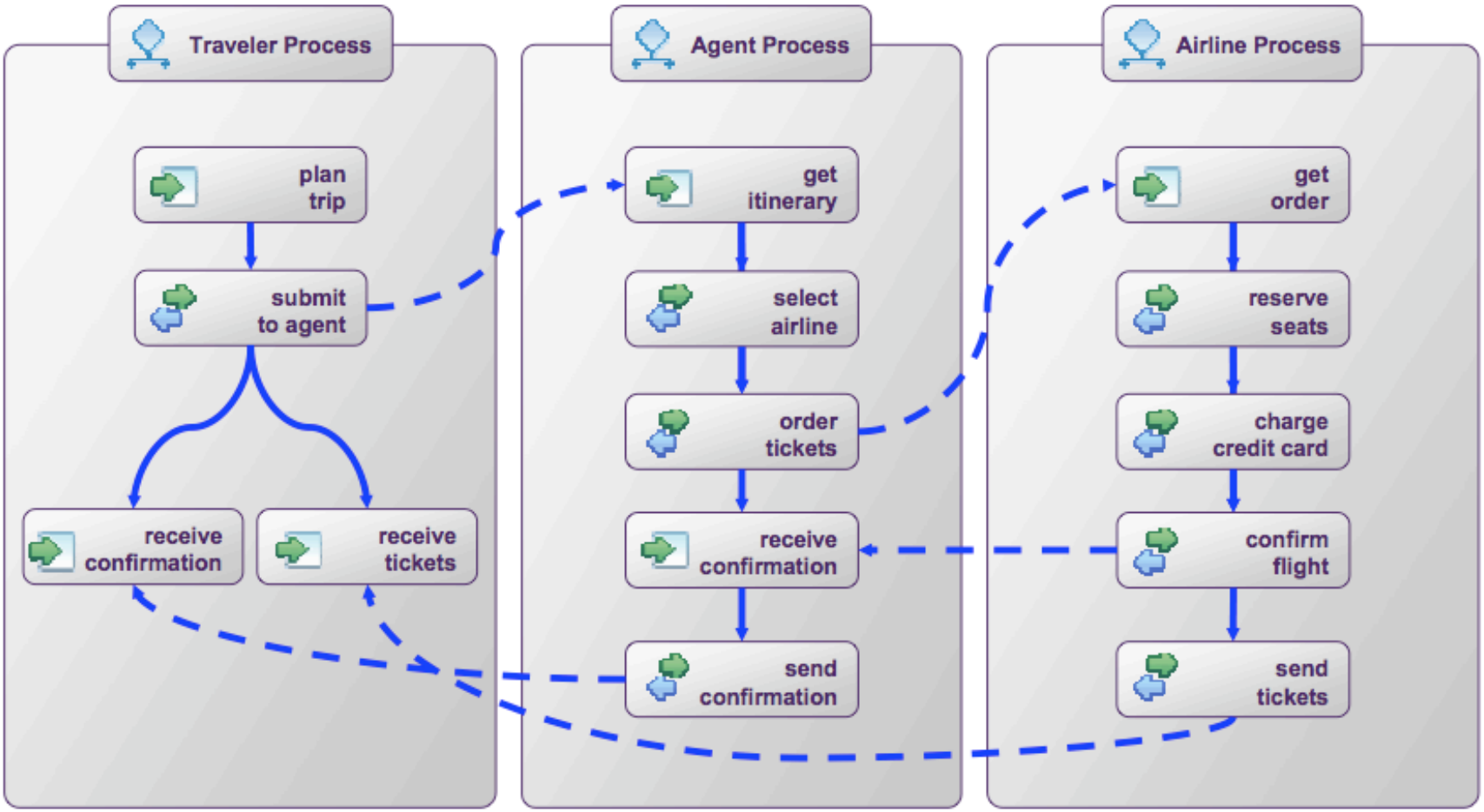
Select exactly one activity from a set of choices



Defines a block for declaring local variables, fault handlers, compensation handler, and event handlers

Source: oasis-open.org

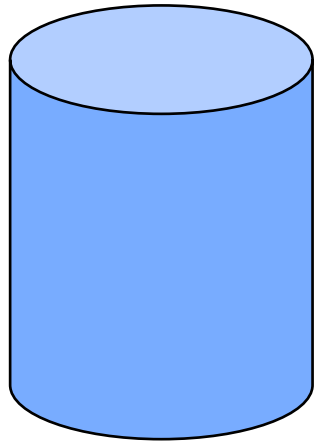
WS-BPEL
Exemple de processus exécutable



Source: oasis-open.org

Référentiel SOA

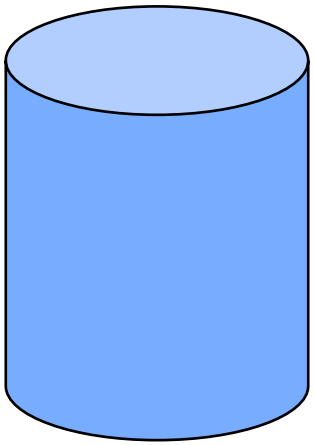
- Ensemble de métadonnées (élémentaires, relations)
- Reporting
  - Gestion des évolutions
  - Analyse des impacts
  - Usage des services
- Gestion des « actifs » SOA et versions
  - Modèles des orchestrations de services
  - Modèles de processus métier
  - Modéles informationnels
  - Schémas XML
  - Transformations (XSLT)
  - Applications composites



Source: software AG

Gouvernance SOA

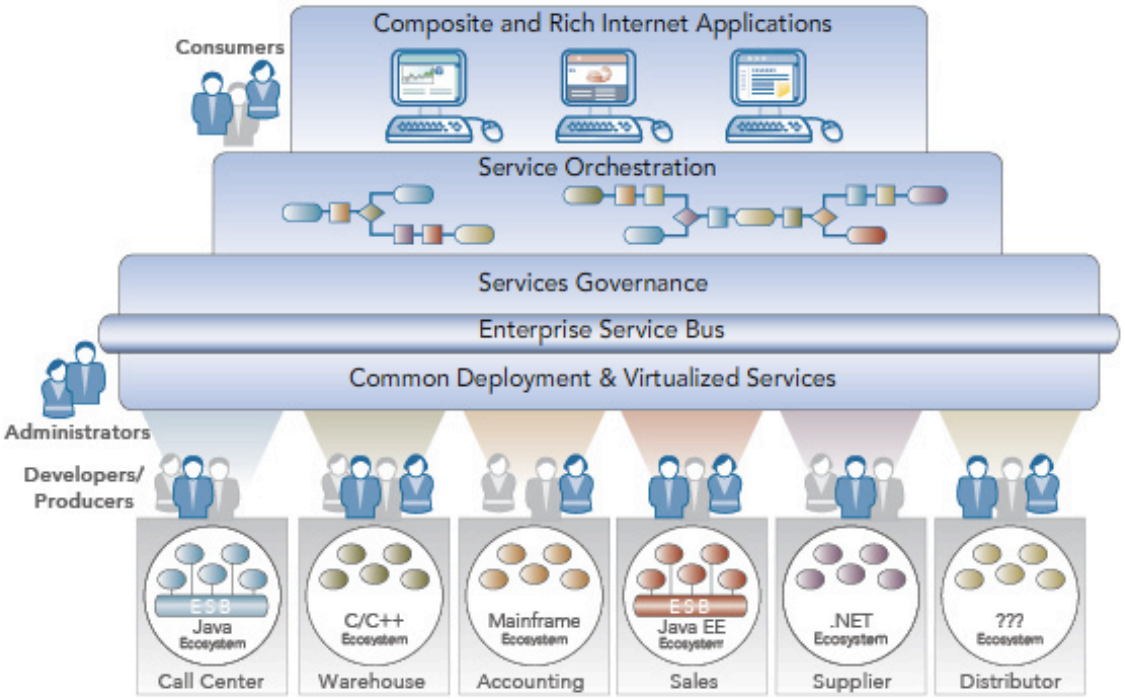
- **Gérer et capitaliser les « actifs » SOA**
  - Cataloguer, retrouver (« find ») les actifs SOA
  - Reporting sur l'usage des services
  - Réutilisation de fonctionnalité existante
  - Améliorer l'efficience SOA
- **Gérer les évolutions**
  - Support aux besoins métier
  - Analyser l'impact d'un changement
- **Gérer les résultats**
  - Analyser les investissements IT par rapport aux enjeux métier
  - Augmenter le ROI de « votre SOA »



Source: software AG

Stacks SOA
(Exemple sur le marché propriétaire)

- **TIBCO Active Matrix**



Source: tibco.com, 2007

- **IBM Websphere, BEA, Oracle**
- **SAP**

Stacks SOA
(Exemples dans le monde Open Source)

- **Projets français ANR**
  - JOnES (ESB/JBI)
  - SCOrWare (SCA), plateforme orientée service, à base de composant
  - SemEUsE, sémantique pour bus de service
- **Projets internationaux, communautés Open Source**
  - Plateforme SCA Java FraSCAti (forge Inria)
  - PEtALS (forge OW2)
  - Eclipse STP, outillage SOA pour développeurs (forge Eclipse)
  - Plateformes SCA Tuscany (forge Apache)
    - Fabric3 (forge Codehaus), MuleSCA (forge Mule)
  - Spagic (forge OW2), outillage ESB
  - Orchestra (forge OW2) , orchestration de Web Services
  - ProActive (forge OW2), plateforme GRID (évolution vers SCA)

Plan

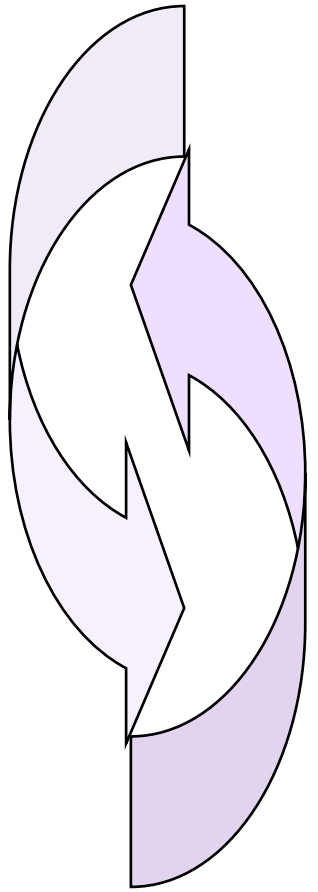
- **I SOA: présentations (matin)**
- **I-1Introduction à SOA**
  - *Concepts, éléments clé et standards*
  - **Point de vue ➔ conception de système d'information**
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- *I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)*
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - *Plateforme PEtALS / FraSCAti*
- *I-3 L'outillage de développement en environnement Eclipse*
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

Point de vue ➔ conception de système d'information

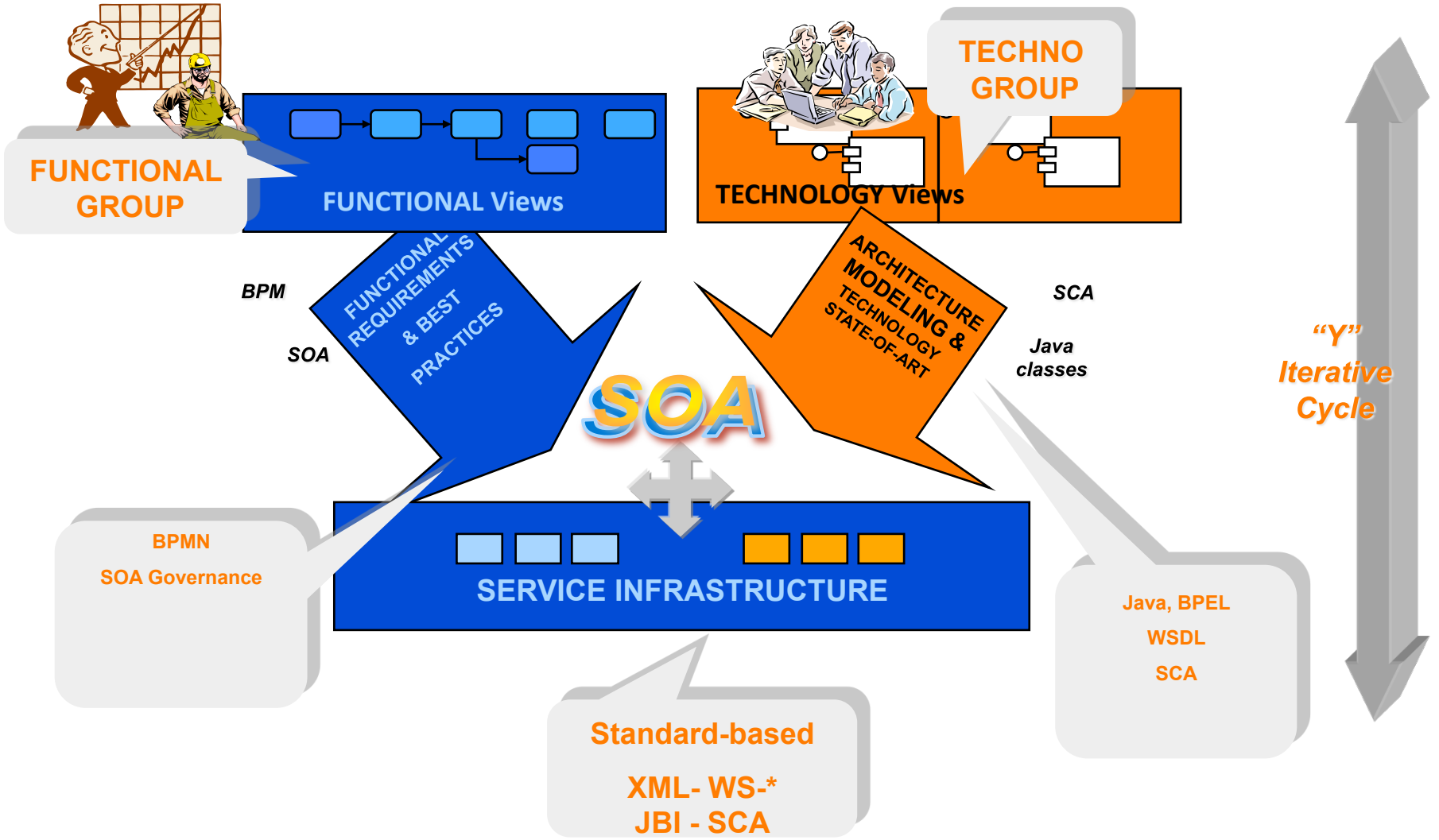
- Une démarche globale et agile
- Une approche collaborative
- Une vue « multi-couches »
  - Design et runtime
- Une collaboration avec le BPM (Business Process Modeling)
  - SOA et orchestration (proposition d'une approche innovante)
  - SOA et urbanisation (approche existante)
- Etapes de la collaboration
  - Du « métier » vers le technique
  - Selon une approche « top-down »
  - Pour plus de détails …
    - Présentations et démonstrations à venir (I-3, II)

Une démarche globale et agile

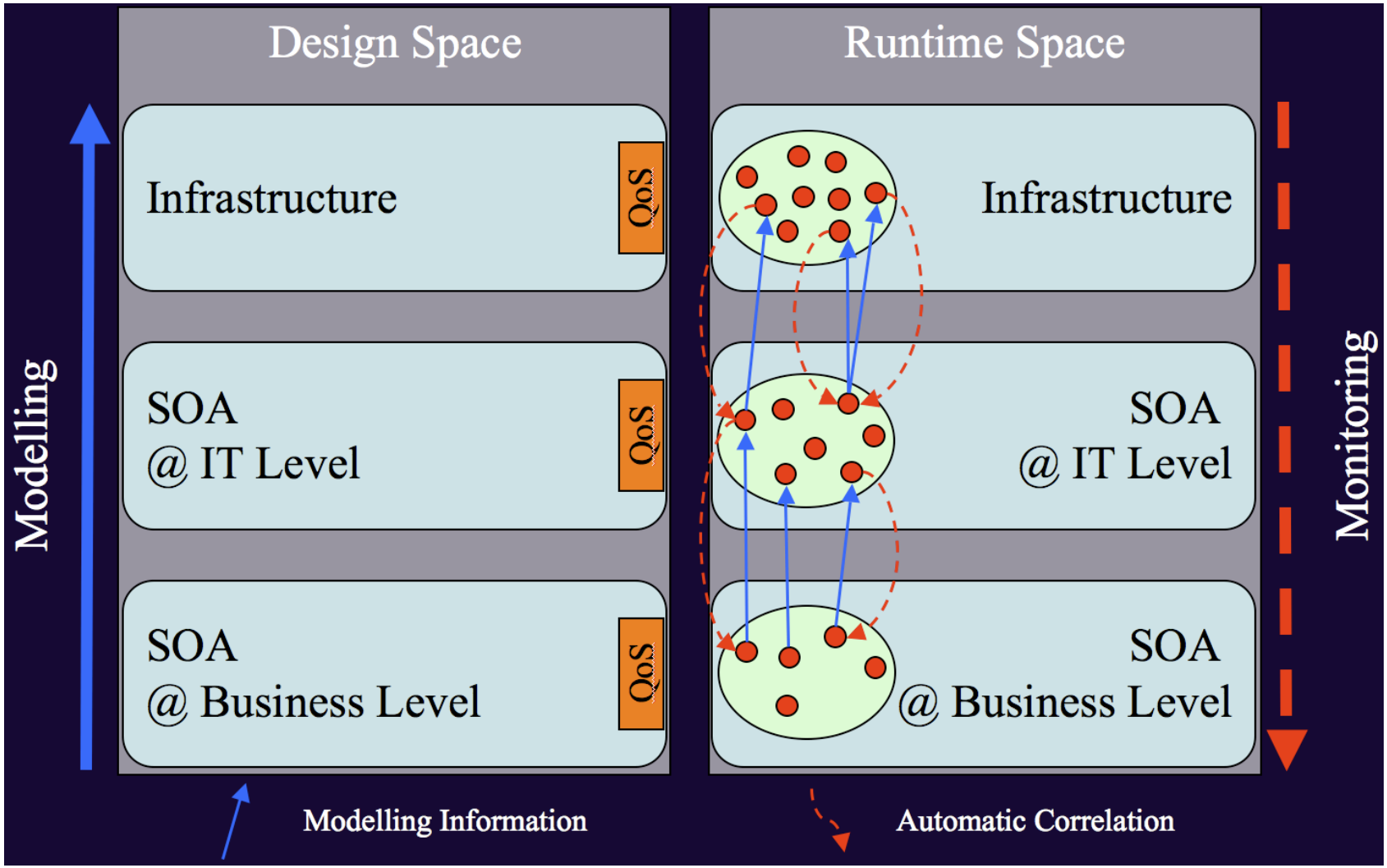
- Penser global et agir pragmatiquement
- Adaptée aux enjeux de l'entreprise
  - Coût, délai, qualité
- De la conception à la mise en œuvre
  - Recueillir et comprendre les besoins, modéliser
  - Orchestrer les services (logique)
  - Concevoir et assembler les services (composants)
  - Déployer la plate-forme (hébergement ou in situ)
- Livrer à date fixée
- Analyser le service rendu



SOA: vue collaborative et itérative



« *Multi-layer perspectives and spaces in SOA* »
Communication, SDSOA 2008
2nd Intl Workshop on Systems Development in SOA Env.

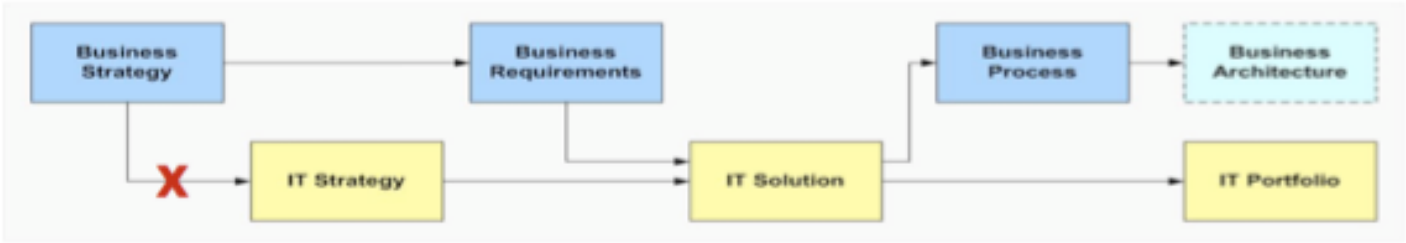


SOA et BPM

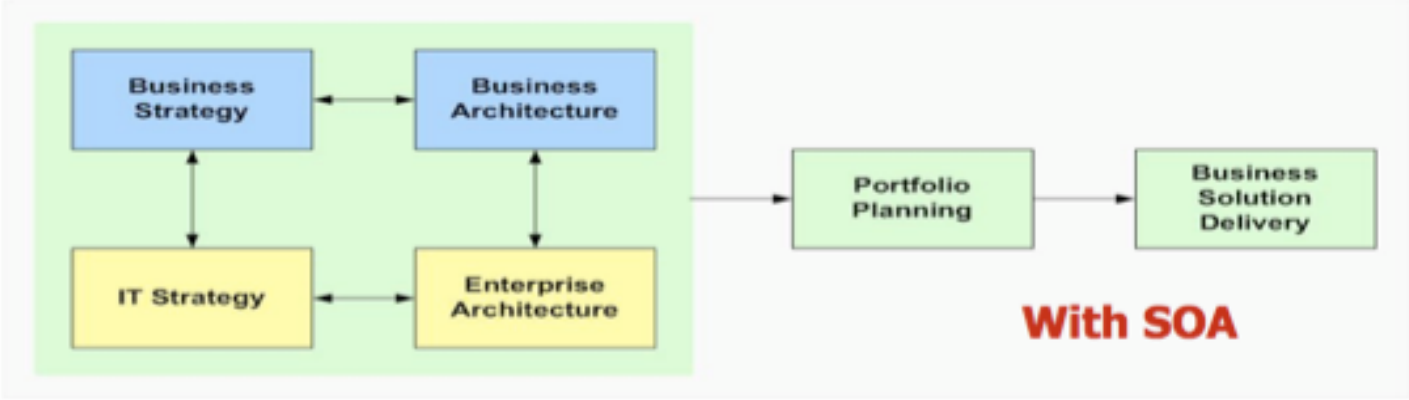
- **Développement du BPM (Business Process Modeling / Management) et du WfM (Workflow Management) (1995+)**
  - Fondations technologiques (process, workflow)
  - Donne les moyens aux organisation et entreprises de gérer, mettre sus contrôle et optimiser leurs processus métier
- **Emergence de SOA (2004+)**
  - Architectures techniques et de systèmes d'information
  - Donne les moyens d'une agilité et flexibilité IT / IS, et d'une mise en œuvre des processus métier et de leur évolution

« SOA and BPM are one strategy »
Richard Soley, Chairman & CEO OMG, Executive Director SOA Consortium

- **SOA est une réponse au besoin de collaboration IT / Métier**
  - **« Enterprise Architecture », OMG, SOA Consortium**

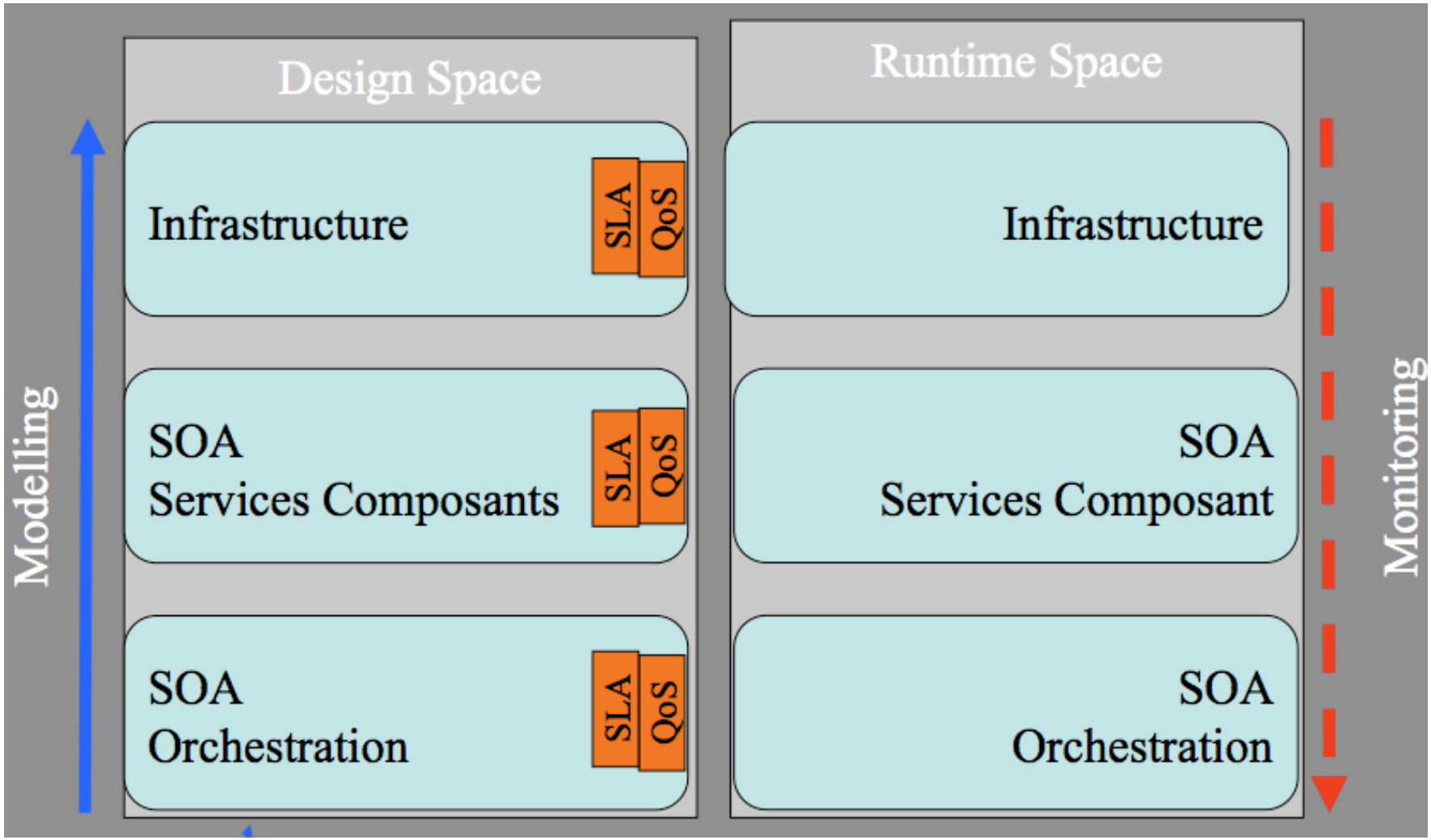


Before SOA

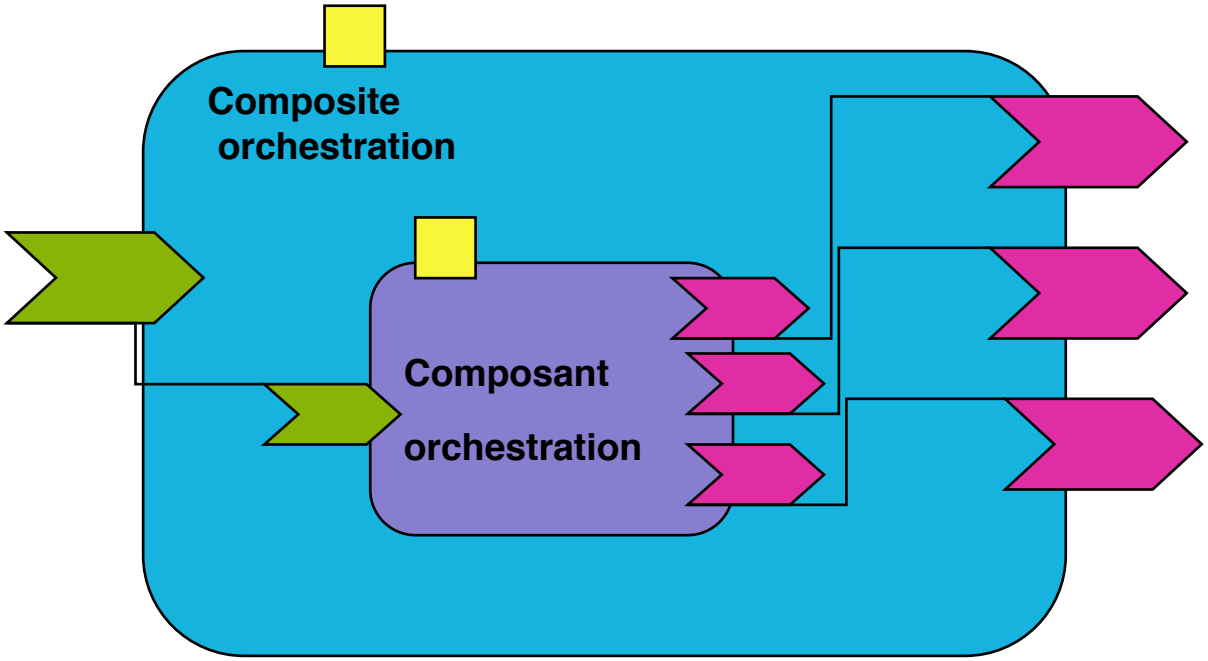


With SOA

SOA et orchestration
Problématique et proposition



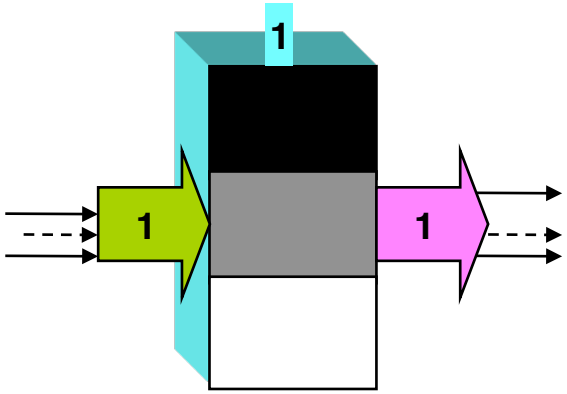
SOA ➔ Composite Orchestration



➔ Un composant d'orchestration associé à une liste de services

Boîtes …

- **Boîtes noires, blanches, grises**
  - **Interface**
  - **Implémentation**
  - **Déploiement**



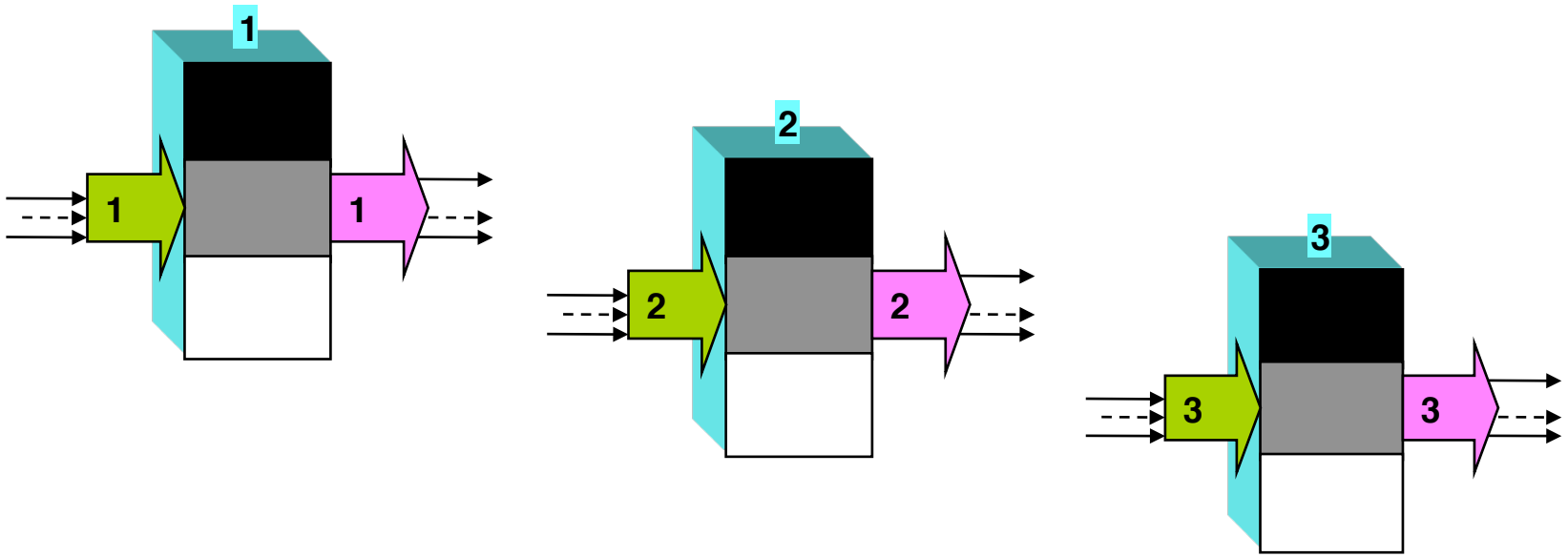
Niveau Abstrait

- Interface décrit une vue logique
  - Composant, service, composite, composant fractal..
- Contient a priori les différentes méthodes et les variables d'entrées et de sorties associées
- L'assemblage (composition, orchestration)
  - A un niveau logique
  - En enchaînant les éléments et en câblant les entrées sorties
- Informations Qos et SLA peuvent être prises en compte
- Informations logiques de monitoring et de propagation, les règles, plans d'actions aussi
  - Dans une sorte d'exécution logique virtuelle

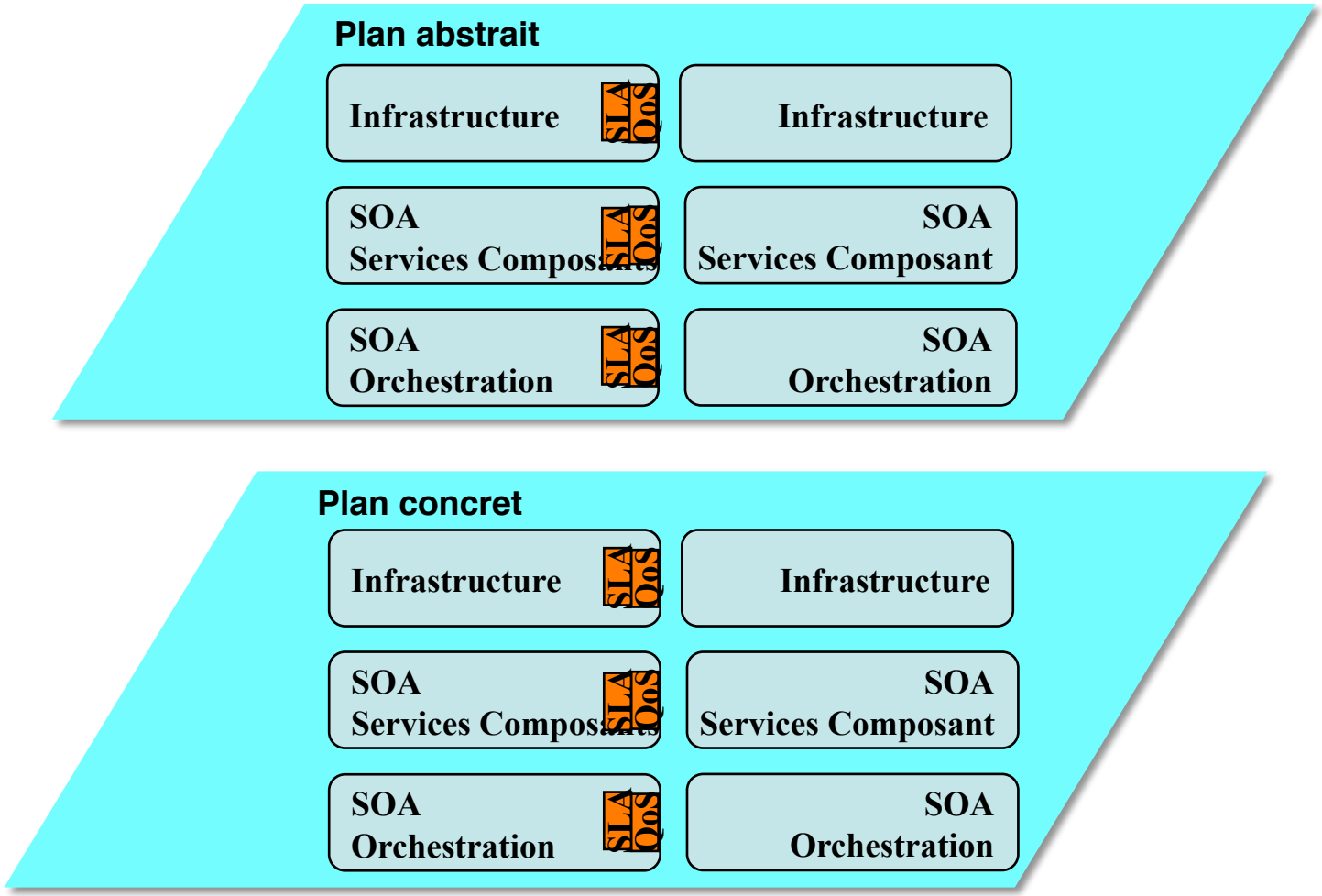
Projection Concrète

- **Projection de cette forme logique d'orchestration**
  - Sur un espace concret
  - Dans la mesure où les implémentations sont conformes aux interfaces
- **Binding des exécutions**
  - En fonction du déploiement physique des divers éléments
- **Monitoring de l'exécution finale**
  - Conformément au modèle design
  - Par remontée des diverses projections et bindings ….

Orchestration / assemblage des boîtes noires, grises, blanches



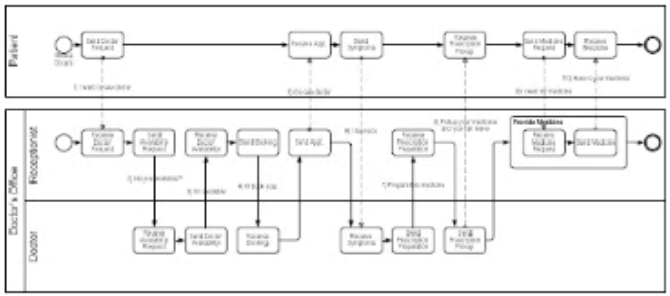
Multi-layer, espaces, et … plans



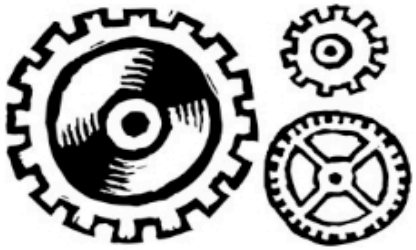
Mise en oeuvre dans une plate-forme de type BPMS

Plateforme BPMS

- = plateforme mettant en oeuvre une approche BPM



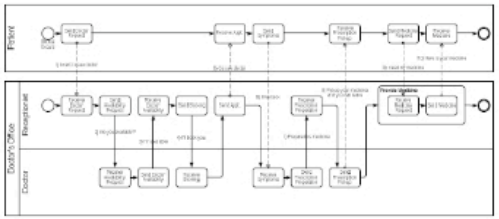
Processus d'entreprise
(BPMN)



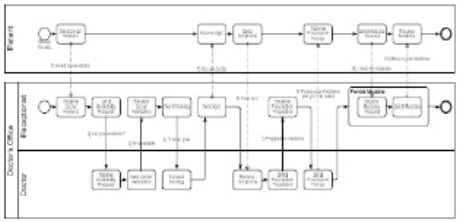
Système Informatique
(Services - BPEL)

Prise en main, problématique

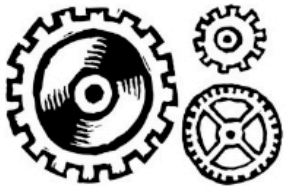
- **Problématique**
  - **étudier la mise en place et l'évolution d'une architecture SOA...**
  - **... dans le cadre d'une approche BPM**
- **Comparer deux approches**



Top-Down

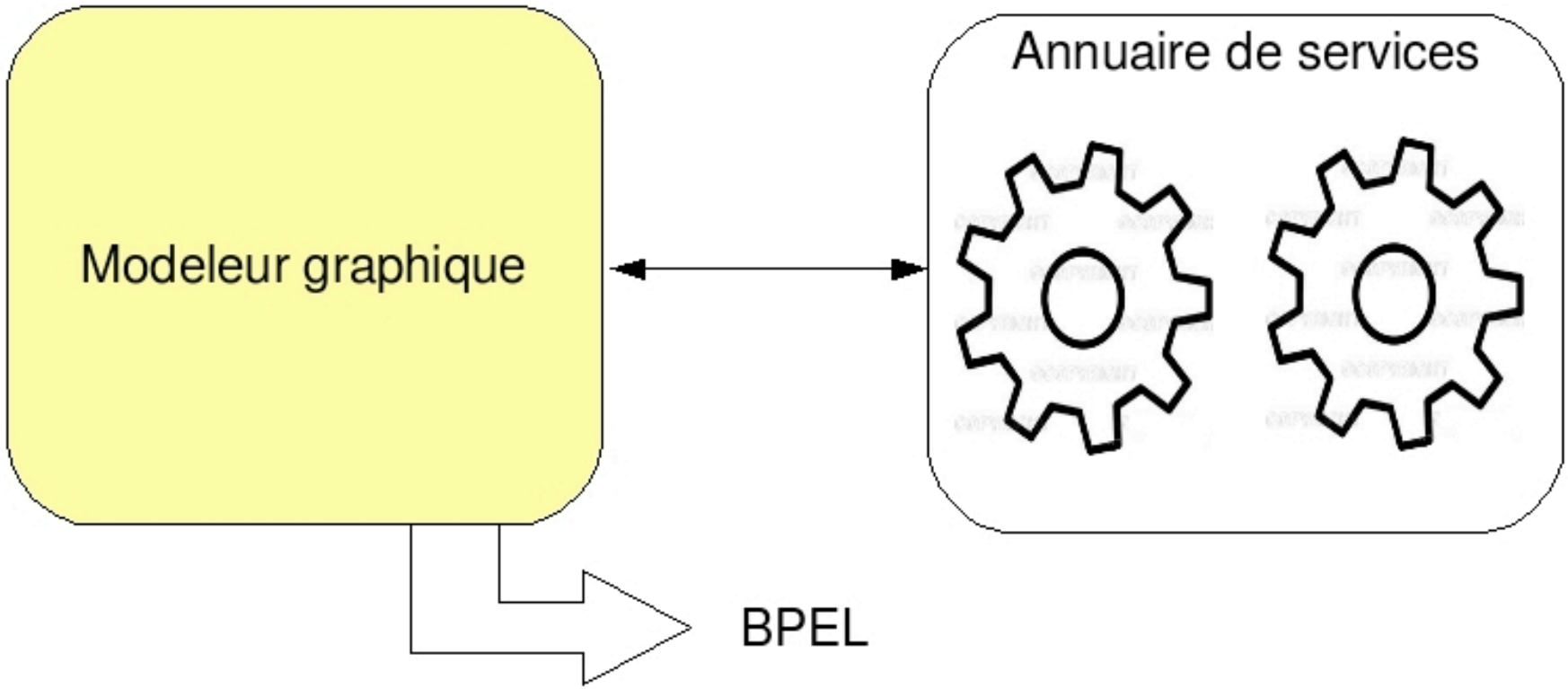


Bottom-Up



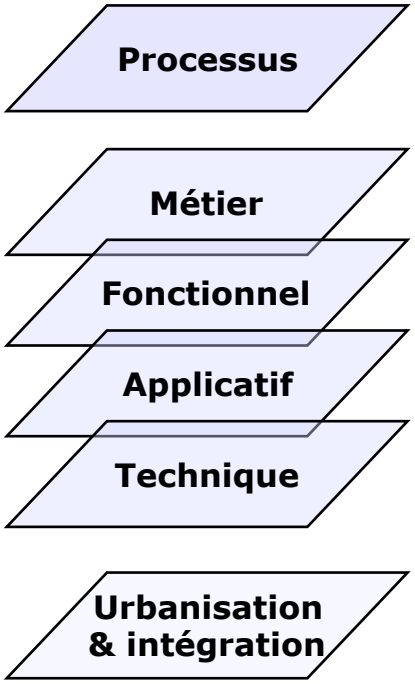
Approche bottom-up : mise en oeuvre

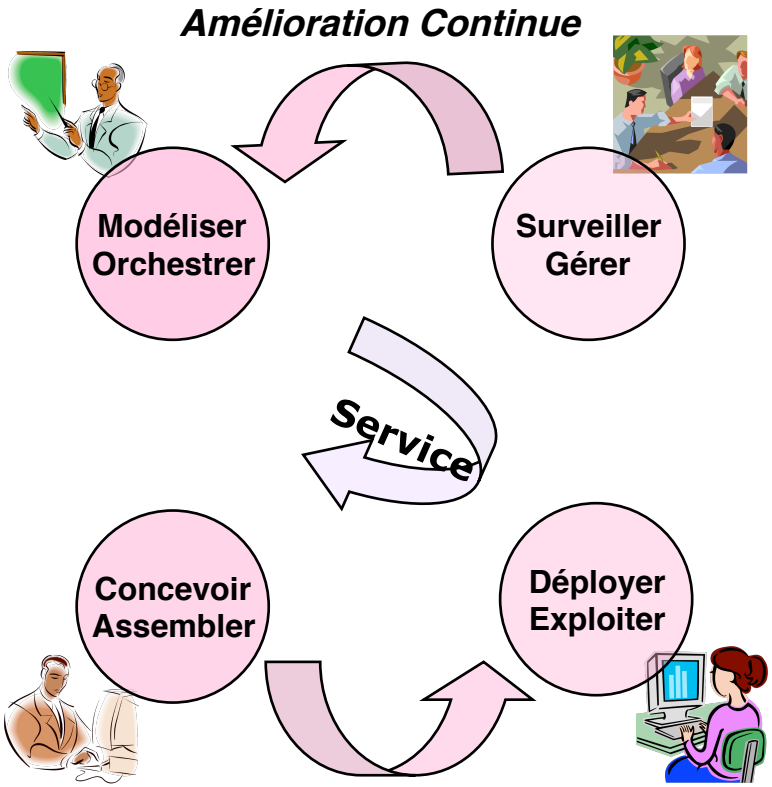
- **Point de départ ➔ processus et services**
  - **BPM / SOA**



SOA et urbanisation

Un processus itératif



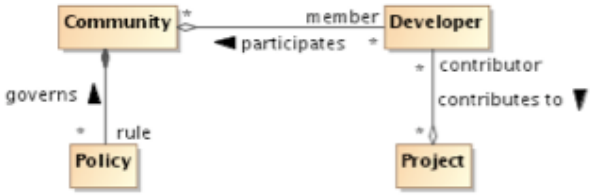


Etapes de la collaboration (1/5)

- **Analyse du domaine métier**
  - Architecte fonctionnel (BPM) / architecte SI (SOA)
  - Modélisation des processus métier, référencement
  - Modèle de composition SOA aligné sur les processus
  - Schéma d'urbanisation et modèle d'orchestration de services
  - Réalisation de « maquette SOA métier »
  - Référentiels de services et exposition de services
  - Outils
    - UML et modélisation processus métier
    - Gouvernance et exposition SOA

[Exemple SCOrWare]
T3.2-1) Analyse du domaine métier: une forge orientée service

- ✔ Etude et analyse de la problématique d'une nouvelle génération de forge orientée servi
  - « Open Forge Standard - Basic Specification » (IST FP6 QualiPSo) ➔ Spécification
  - n FOSDEM 2008 (Olivier Abdoun, Ciaran Bryce, Adrian Mos)



Communities

Lack of interoperability

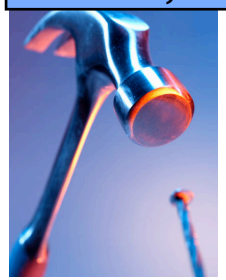
- data lock-in
- centralization
- tight coupling
- multiple identities

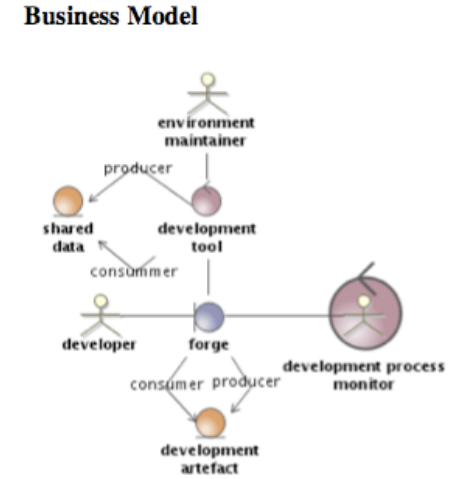
Businesses

Lack of development process support

- limited project management tools
- environment poorly adapted to project organization
- no process evaluation (BAM)

UML, BPMN PRM, …

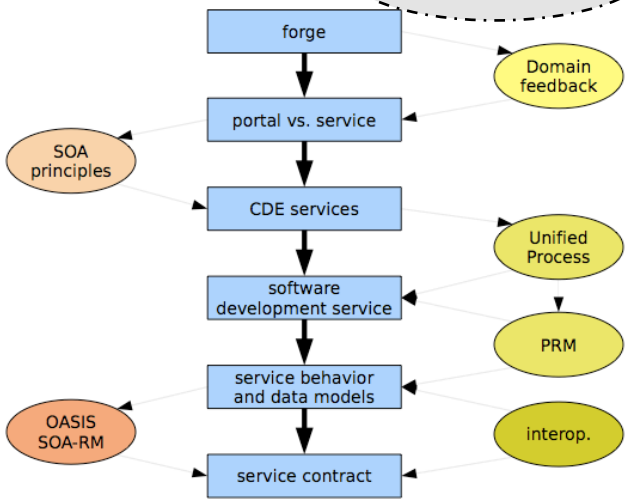




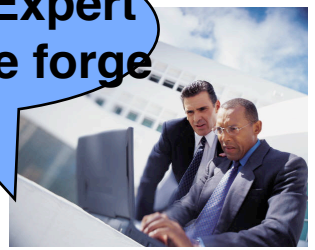
Business functionality

Abstraction · Autonomy · Cohesion and completeness · Compliance to standards · Composability · Discoverability · Internet based · Legacy encapsulation · Loose coupling · Optimization · Organization boundary crossing · Reusability · Service contract

Business SOA



Analyste / Expert du métier de forge



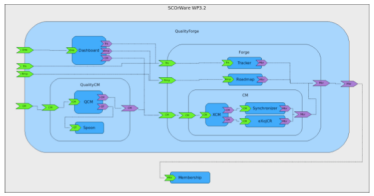
[Exemple SCOrWare]
T3.2-3) Analyse des services d'une forge NG

- Conception, définition et identification des services d'une forge NG (problématique SOA)
  - Spécification **Services and references**

Business

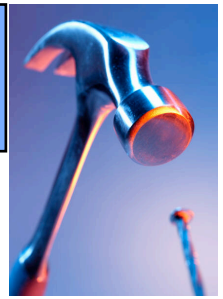
SOA

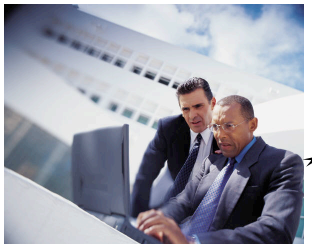
SCA



- **Dhb** (*Dashboard*): progress and quality dashboard specification and consultation interface
- **Trk** (*Tracker*): issue tracker browse and query interface
- **Rmp** (*Roadmap*): project planner browse and query interface
- **CM** (*Configuration Management*): configuration management and content access interface
- **QT** (*Quality Tool*): quality tool invocation interface
- **Mbr** (*Membership*): event publishing interface

SOA Repository





Analyste / Expert du métier de forge

Analyste / Expert SOA

Etapes de la collaboration (2/5)

- **Cas d'usage applicatif**
  - Architecte applicatif (SOA) / Architecte technique (SOA)
  - Design d'une architecture applicative
  - Réalisation de « maquette SOA applicative »
  - Conception normalisée et standardisée des composants applicatifs
  - Découpage applicatif et interopérabilité inter-applications
    - Standards et normes de conception
  - Outils
    - Modélisation
      - Utilisation « particulière » de SCA
      - BPMN
    - Plateformes SCA

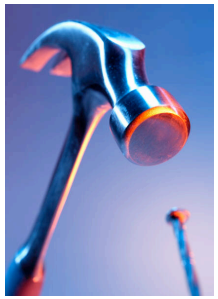
[Exemple SCOrWare]
T3.2-2) Identification & description de cas d'usage: QualityCheck, Dashboard

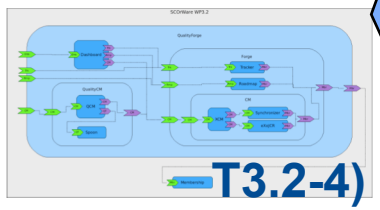
SCA
@ Business Level

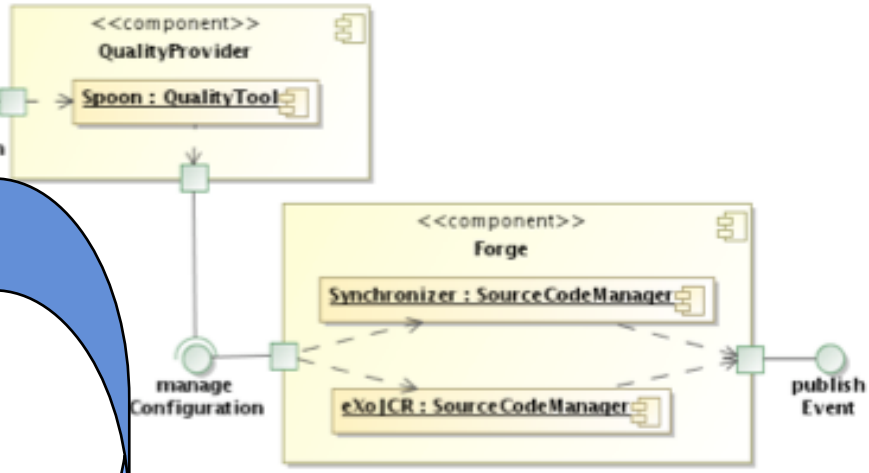
- ✔ Identification et description de 2 cas d'usage pour le démonstrateur SCA (solution) T3.2
  - Spécification **Quality check scenario realization**

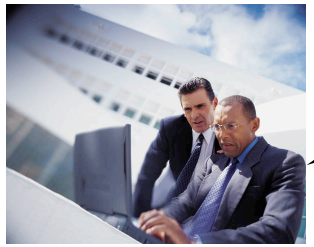
UML

BPMN









Analyste / Expert du métier de forge

T3.2-4) Design SCA

Etapes de la collaboration (3/5)

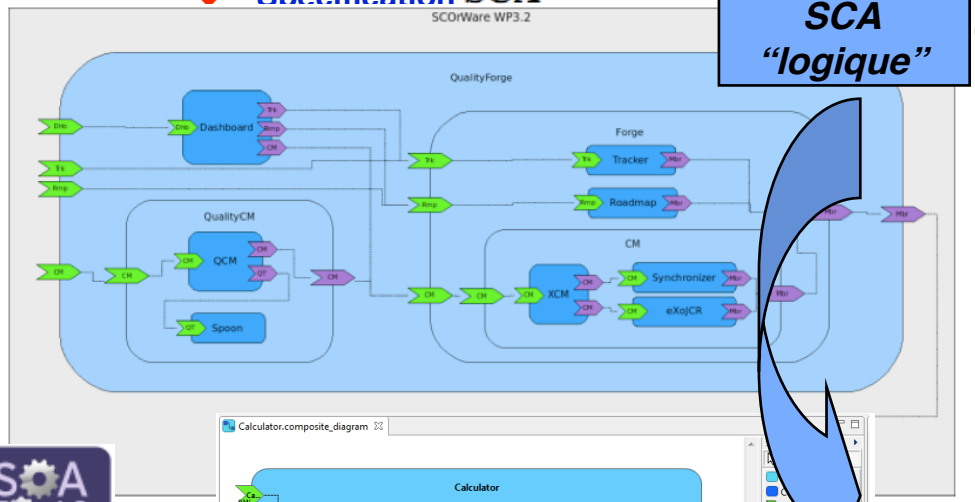
- **Architecture technique**
  - Architecte technique (SOA/SCA) / ingénieur développement (SCA, Java)
  - Définition d'une architecture distribuée
  - Définition d'une infrastructure commune de communication
  - Réalisation de « maquette de faisabilité technique »
  - Tests de charge & optimisation
  - Gestion de configuration
  - Outils
    - Java, XML, SCA, Eclipse
      - (Struts, Spring, Hibernate)
    - CVS, Subversion
      - (ClearCase)
    - Plateformes SCA
      - (J2EE, SCA, ESB)

[Exemple SCOrWare]
T3.2-4) Design itératif de l'architecture SCA

- Plusieurs itérations pour définir l'architecture SCA d'une solution de forge « Quality Chec

SCA
@ IT Level
WP3
WP2

- Spécification SCA



SCA "logique"

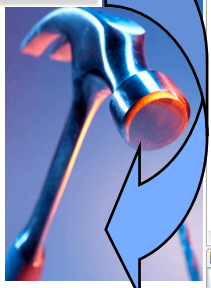
Components

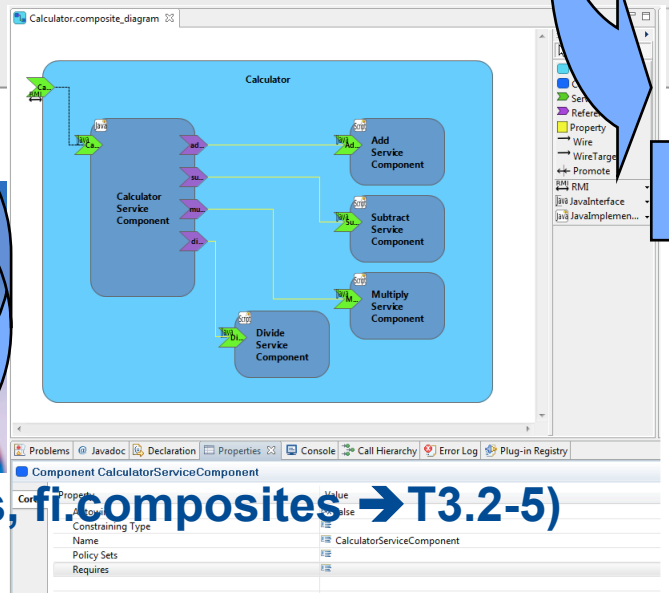
- **Dashboard**: compiles a project milestone's progress and quality assessment dashboard from roadmap, issue tracker and content manager
- **QCM**: composition service invoking quality tool prior to patch integration to content management
- **Spoon**: source code quality analyzer
- **Tracker**: LibreSource project issue tracking
- **Roadmap**: project planning service implementation (does not currently exist as such in LibreSource)
- **XCM**: source code management and content management is done through the unified CM service; this component dispatch CM requests to a component depending on the managed resource (source code to Synchronizer, content to eXoJCR)
- **Synchronizer**: LibreSource source code manager
- **eXoJCR**: eXo Platform implementation of JCR170
- **Membership**: team building and awareness component (part of LibreSource kernel)

Composites

- **Forge**: bundling of source code management, content management, issue tracking, project planning and software dependency management for collaborative software development
- **QualityCM**: facade checking code source quality prior to integration into another CM
- **QualityForge**: Forge integrating source code checking and exposing project dashboard service

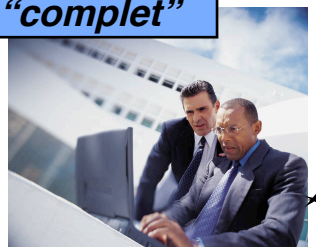
SOA TOOLS PLATFORM
SCA





Interfaces, fi.composites ➔T3.2-5)

SCA "complet"



Architecte SOA / SCA

Analyste / Expert SOA

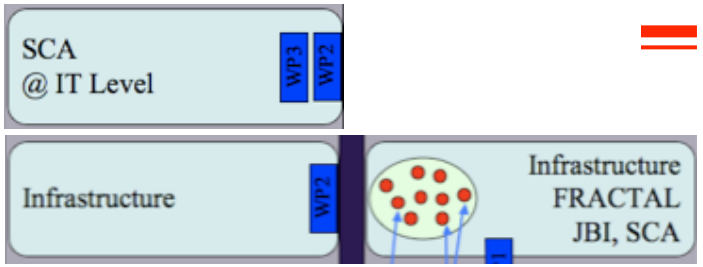
Expert technique composants

Etapes de la collaboration (4/5)

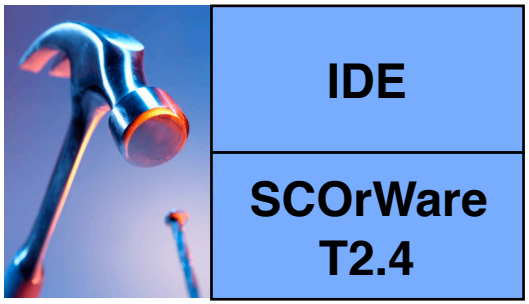
- **Implémentation et déploiement**
  - Ingénieur développement (SCA, Java) / Ingénieur infrastructure (SCA, ESB, réseaux, sécurité)
  - Développement et enrichissement des composants
    - Développement des IHMs
  - Mise en production et déploiement
  - Fiabilisation, sécurisation et garantie de fonctionnement des services hébergés
  - Outils
    - Java, XML, SCA, Eclipse
      - (Struts, Spring, Hibernate)
    - Plateformes SCA
      - (J2EE, .NET)
    - Firewalls, authentification, réseaux et systèmes, monitoring
      - (VPN, LDAP, PATROL)

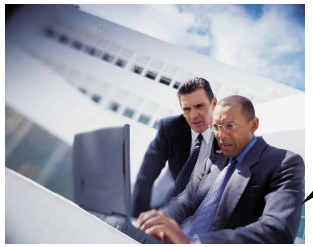
[Exemple SCOrWare]T3.2-5) Implémentation itérative

- Choix du langage, implémentation et test
- de la solution de forge « Quality Check »




```
scm> list
scm> status
add     foo
add     foo/Bar.java
add     foo/Foo.java
scm> checkin
!!! TestSuccessEvent : test success [VSuiteTest]
!!! NewContentEvent : new content [/]
!!! ConfigurationCheckinEvent : new source code revision [/default/path/./cfg/cfg2@bl3]
new version: bl2
scm> list
/foo
/foo/Bar.java
/foo/Foo.java
scm> status
scm> report
Status: success
Report:
<?xml version="1.0" encoding="UTF-8"?>
<report/>

scm> info
Provider [name, location]: [juliac.generated.javax.wvcm.WorkspaceProviderFcSR, /default/path/.]
Configuration: /default/path/./cfg/cfg1
Baseline [name, location]: [bl2, /default/path/./bh/blh1/bl2]
Path: /home/oabdoun/tmp/tinfi/examples/qualitycheck/src/test/resources
scm>
```





Développeur

“Déployeur”

Etapes de la collaboration (5/5)

- **Analyse existant et nouvelle itération d'amélioration**
  - Nouveaux besoins et enjeux métier ?
  - Nouvelles fonctionnalités ?
  - Evolutions technologiques ?
  - Montée en charge, fiabilisation et sécurisation ?
  - Montée en maturité des approches et solutions, industrialisation et généralisation ?
  - …

Pour plus de détails …

- Voir présentations et démonstrations à venir …
  - I-3 L'outillage de développement proposé par Eclipse STP (SOA Tooling Platform)
    - En particulier le modèle intermédiaire STP-IM
  - II-2 Architecture applicative et technique
    - Exemple du cas d'usage II-1

Plan

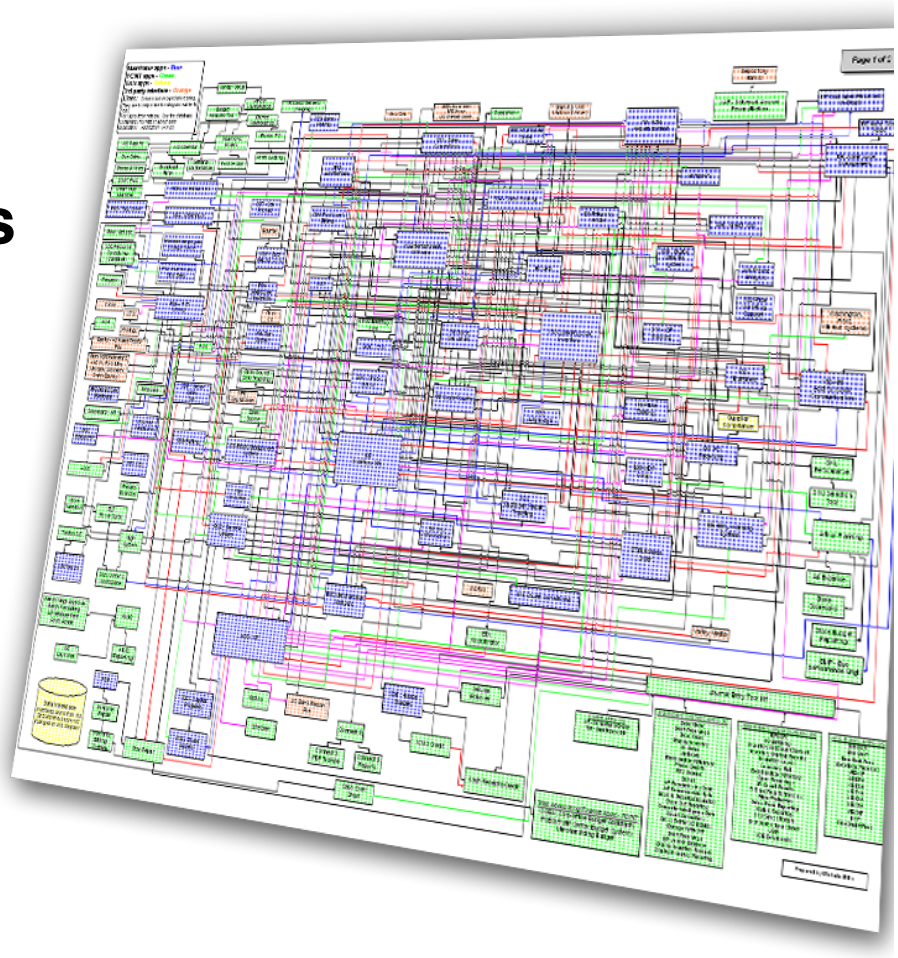
- **I SOA: présentations (matin)**
- **I-1Introduction à SOA**
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - **Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI**
- *I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)*
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - *Plateforme PEtALS / FraSCAti*
- *I-3 L'outillage de développement en environnement Eclipse*
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

Point de vue ➔ développement SCA autour d'une infrastructure ESB (JBI)

- Architecture d'intégration
- Infrastructure et développeur
  - ESB (Enterprise Service Bus)
  - Standard JBI (Java Business Integration)
  - Plateforme JBI PEtALS
- Pour plus de détails …
  - Voir les présentations et démonstrations à venir (I-2, I-3, II)

Constat IT …

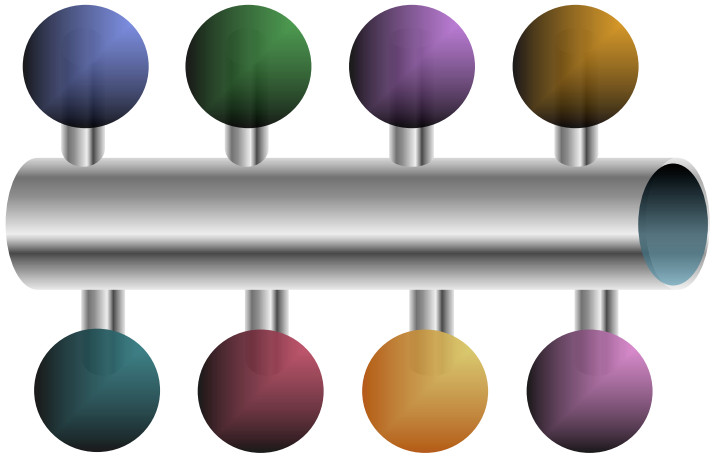
- Complexité
- Architectures rigides
- Evolutions peu aisées
- Et mal maîtrisées



Source: oasis-open.org

… cible IT

- **SOA est une des technologies clé pour réduire la complexité et améliorer la flexibilité**
  - **Orchestration flexible de services**
  - **Des interfaces bien définies**
  - **Des technologies et des outils standards**

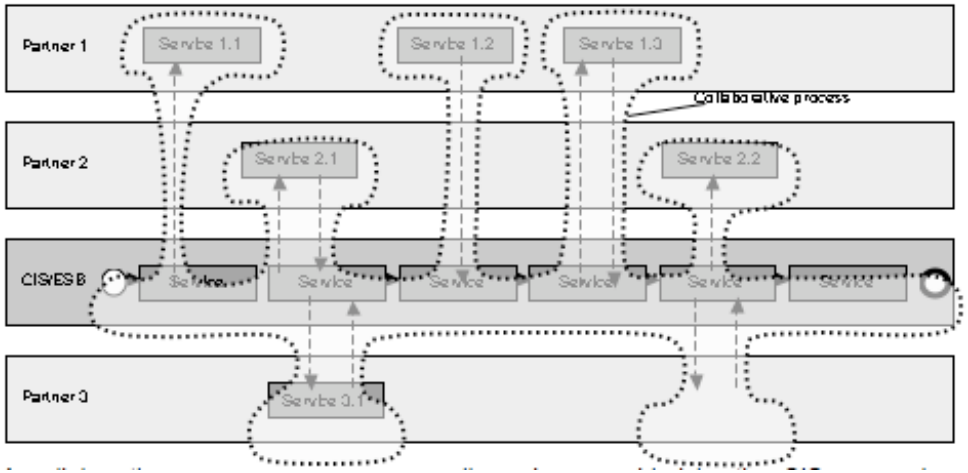


Source: oasis-open.org

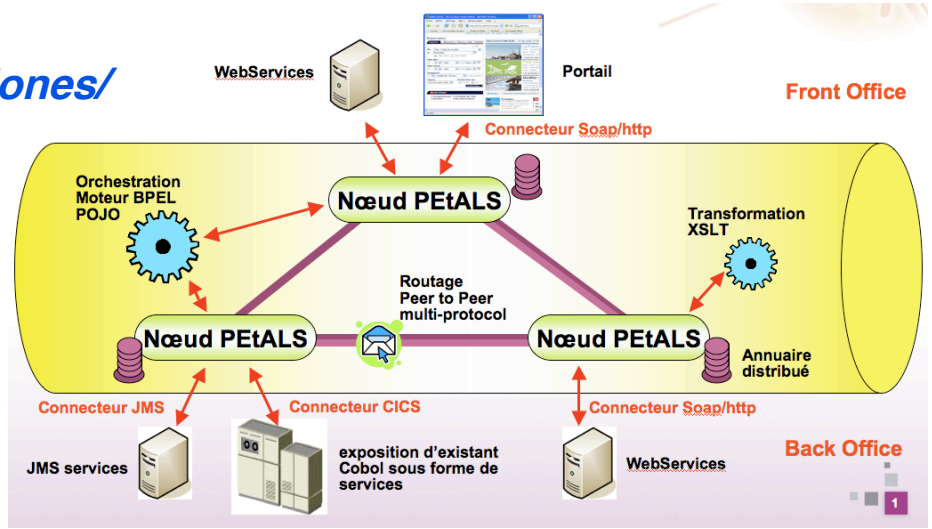
ESB (Enterprise Service Bus)

- Pas de définition universelle
- Bus de Service en tant que support à SOA
  - Un « médiateur »
- Un ESB fournit [tout ou partie de]
  - Routage de message
  - Transformation de message
  - Registre de service
  - Fonctions workflow et orchestration
  - Gestion de la sécurité
  - Gestion de transactions

ESB, « backbone » SOA

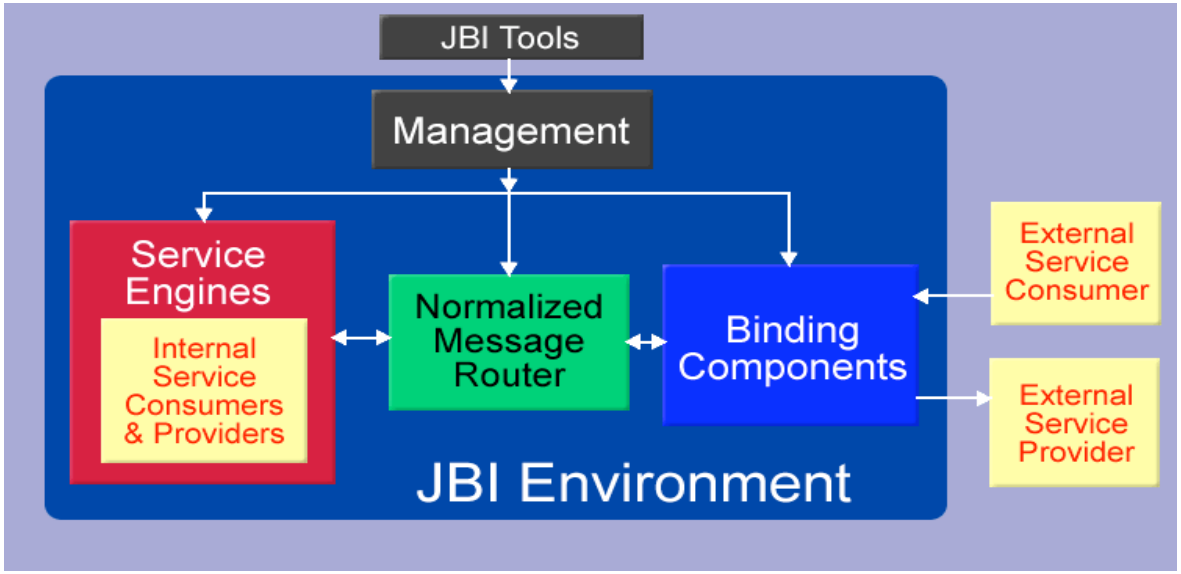


Source: https://ow.inrialpes.fr/projects/jones/



JBI, « Java Business Integration »
JCP, JSR-208

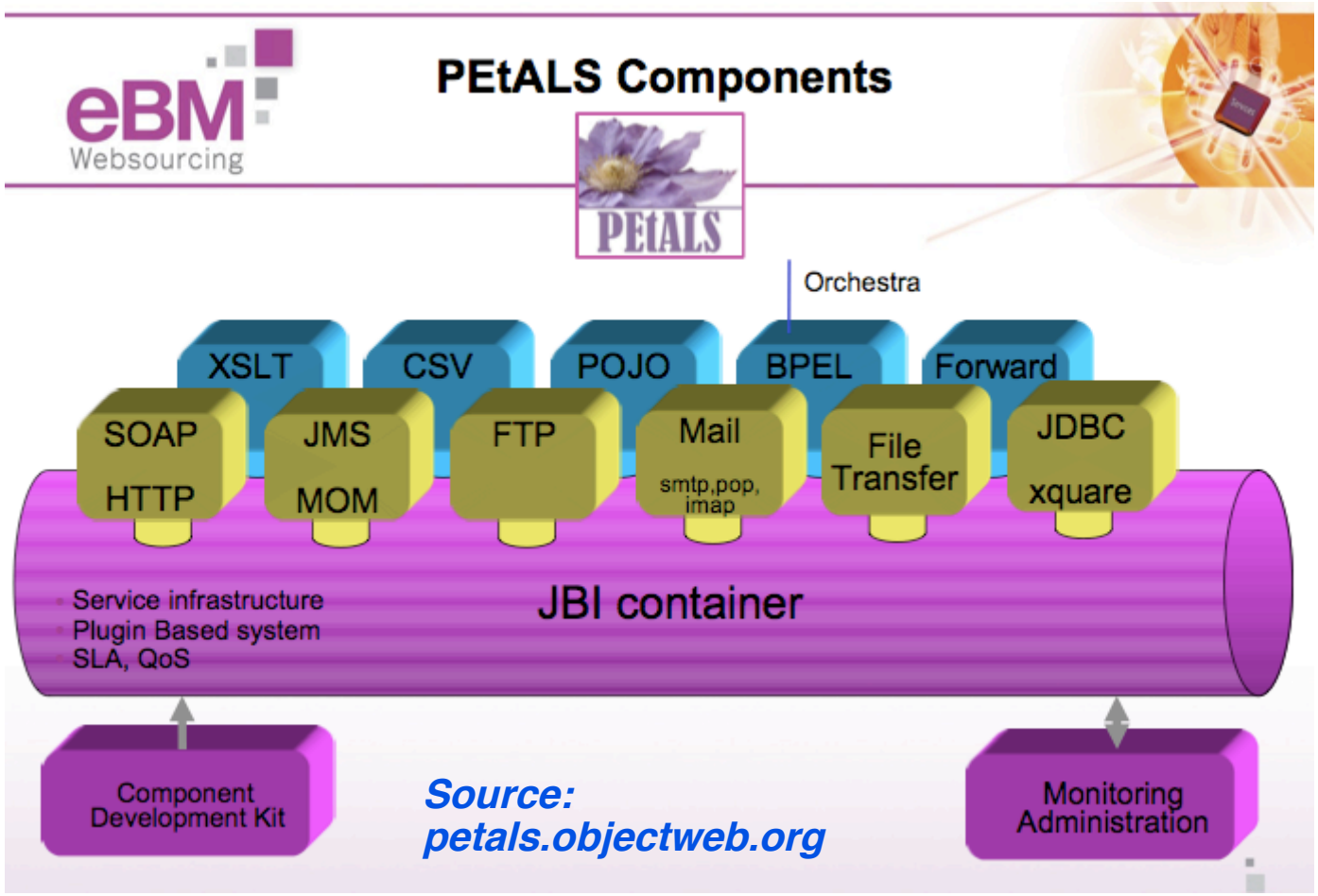
- **Architecture à composants, « pluggable »**
  - **Service Engine: fournir des fonctionnalités techniques**
  - **Binding Component: se connecter au monde extérieur**
- **Standard JBI en monde Java**



Source: java.sun.com

Un ESB conforme à JBI, PEtALS
EBM Websourcing, OW2 Consortium

Composants PEtALS



Pour plus de détails …

- Voir présentations et démonstrations à venir …
  - I-2 SCA et la plateforme d'exécution Open Source développée dans SCOrWare
  - I-3 L'outillage de développement proposé par Eclipse STP (SOA Tooling Platform)
    - En particulier les outils SCA
  - II-3, II-4 Mise en œuvre SCA et déploiement / supervision sur infrastructure JBI SCA
    - Exemple du cas d'usage II-1

Plan

- **I SOA: présentations (matin)**
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- **I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)**
  - **Introduction à SCA (et relations à JBI)**
  - **Relations à Fractal: Tinfi et la plateforme FraSCAti**
  - **Plateforme PEtALS / FraSCAti**
- *I-3 L'outillage de développement en environnement Eclipse*
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

I-2 SCA et la plateforme d'exécution Open Source (SCOrWare)

- I-2a Introduction à SCA
  - Présentation SCA et spécifications
  - Relations à JBI
- I-2b Relations à FRACTAL
  - Plateforme FraSCAti
  - Composant Tinfi
- I-2c Plateforme PEtALS / FraSCAti
  - Intégration FraSCAti dans PEtALS
  - La fonctionnalité apportée par PEtALS

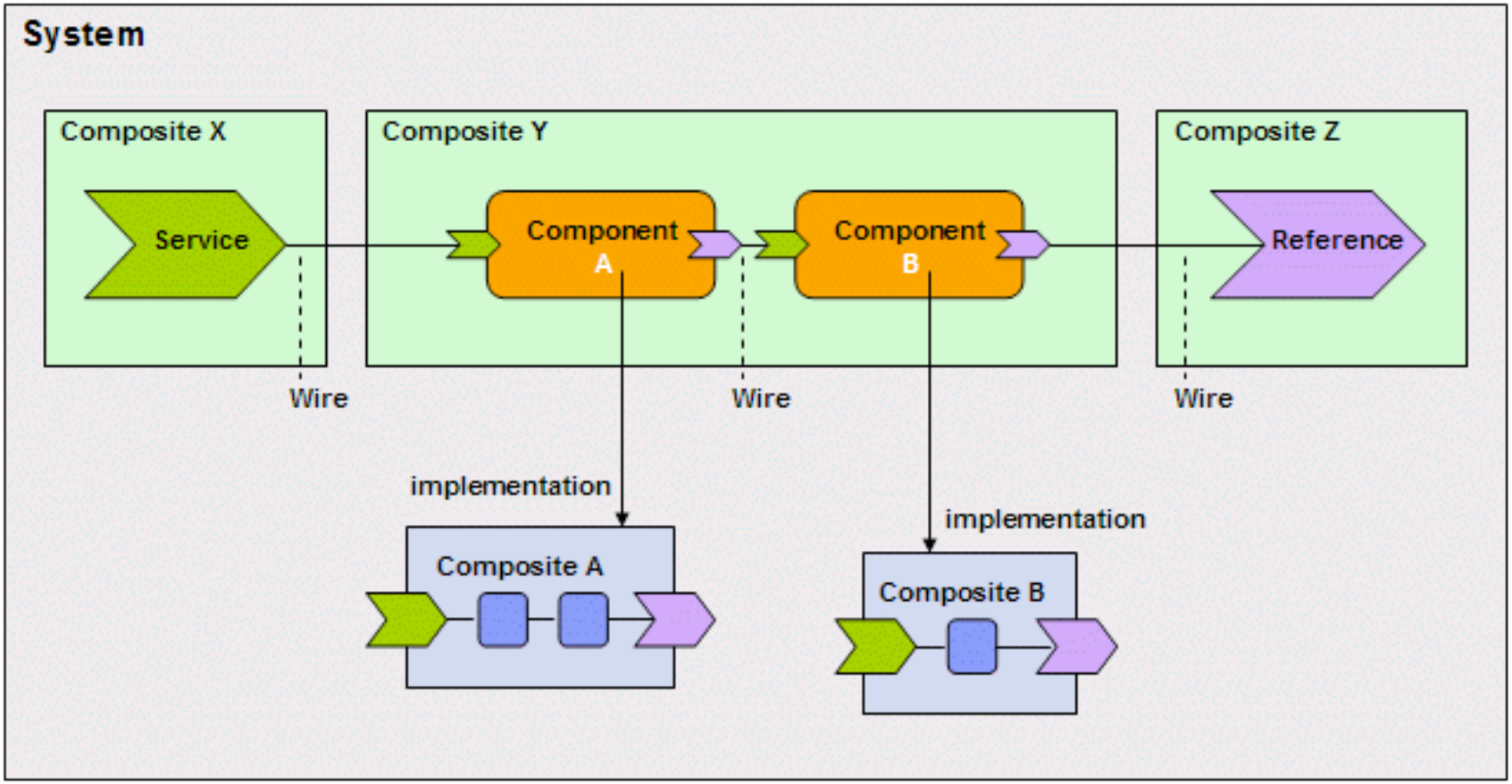
Service Component Architecture

- Component approach to SOA
  - Services are assembled together to create solutions
  - Composite applications contain new and legacy services
  - Separates architecture from implementation
  - Allows reuse at code and service level
- Standardisation by OASIS
- Most major industrial players are behind it
- Open-Source Implementations
  - Tuscany (Apache)
  - SCOrWare (French Consortium) - ANR Project
  - Fabric3
  - Newton
- Tooling available in Eclipse STP

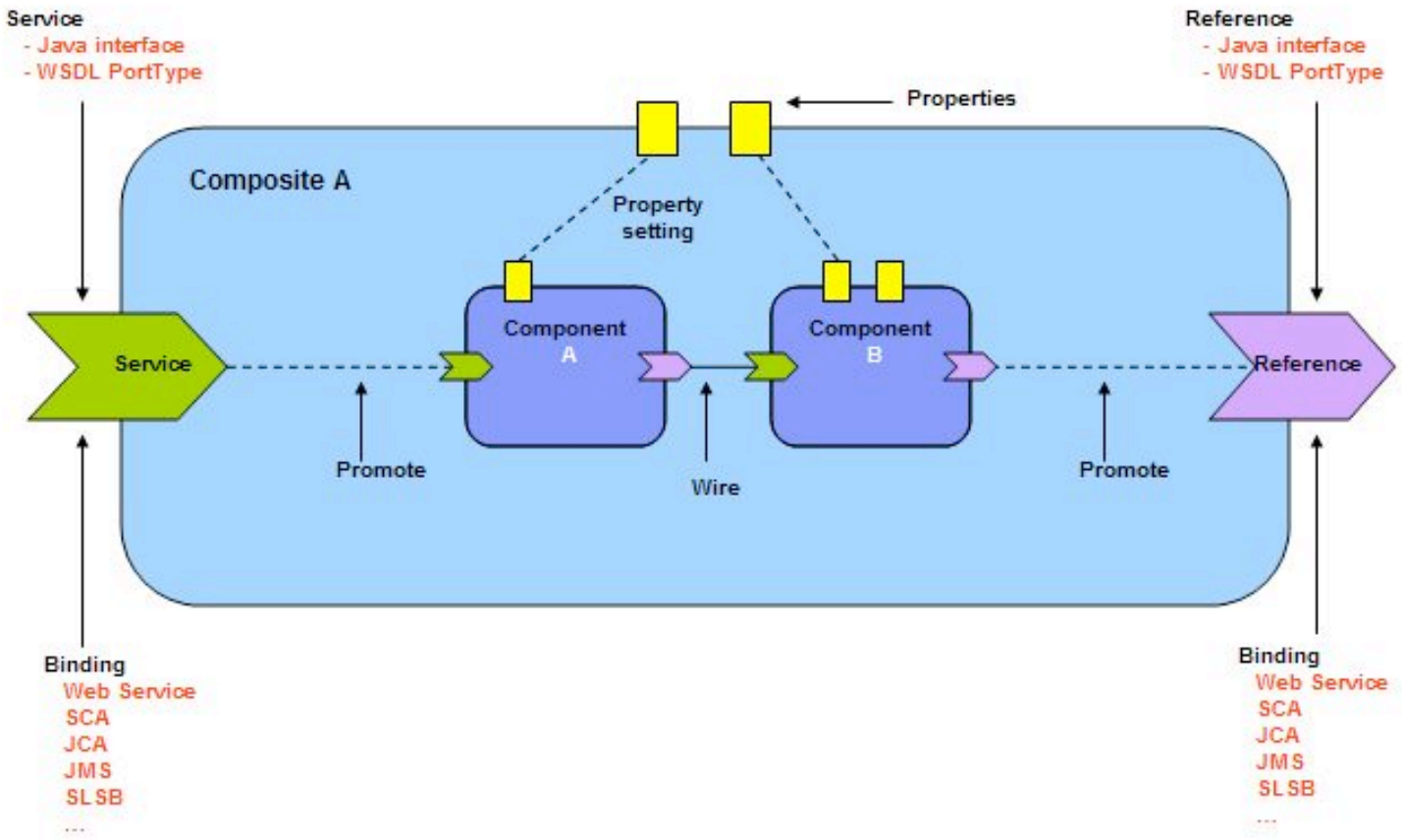
SCA Specification Parts

- The Assembly Model: architectural language + artefacts
  - Components, composites
  - Wires
  - Configuration properties
- The Policy Framework: non-functional requirements
  - Security, transactions
  - QoS
- Bindings: different transports and protocols
- Supports Synchronous and Asynchronous Invocations
- Java Annotations & APIs
- Implementations: Java, C++, BPEL, PHP
- Matching with Component Models: EJB, Spring

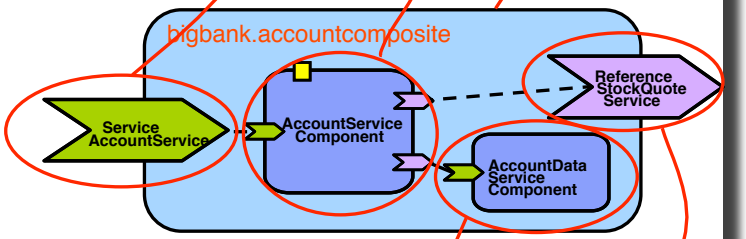
SCA: High-Level Diagram



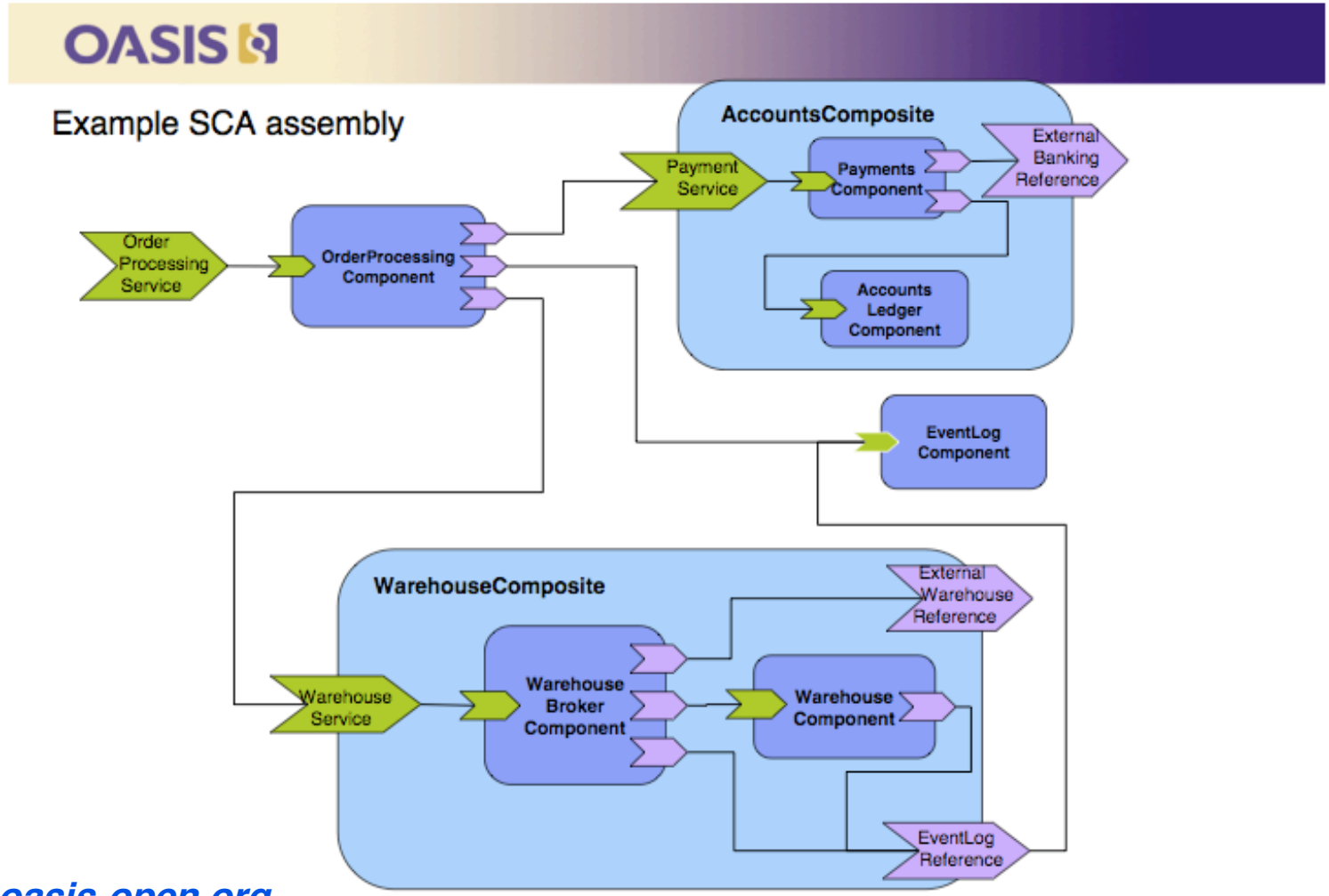
SCA Elements: A Closer Look



```
<composite xmlns="http://www.osoa.org/xmlns/sca/1.0"
           name="bigbank.accountcomposite" >

    <service name="AccountService" promote="AccountServiceComponent">
        <interface.java interface="services.account.AccountService"/>
        <binding.ws port="http://www.example.org/AccountService#
          wsdl.endpoint(AccountService/AccountServiceSOAP)"/>
    </service>
    <component name="AccountServiceComponent">
        <implementation.java class="services.account.AccountServiceImpl"/>
        <reference name="StockQuoteService"/>
        <reference name="AccountDataService"
                   target="AccountDataServiceComponent/AccountDataService"/>
        <property name="currency">EURO</property>
    </component>

    <component name="AccountDataServiceComponent">
        <implementation.bpel process="QName"/>
        <service name="AccountDataService">
            <interface.java interface="services.accountdata.AccountDataService"/>
        </service>
    </component>

    <reference name="StockQuoteService" promote="AccountServiceComponent/StockQuoteService">
        <interface.java interface="services.stockquote.StockQuoteService"/>
        <binding.ws port="http://example.org/StockQuoteService#
          wsdl.endpoint(StockQuoteService/StockQuoteServiceSOAP)"/>
    </reference>
</composite>
```



SCA assembly (example)

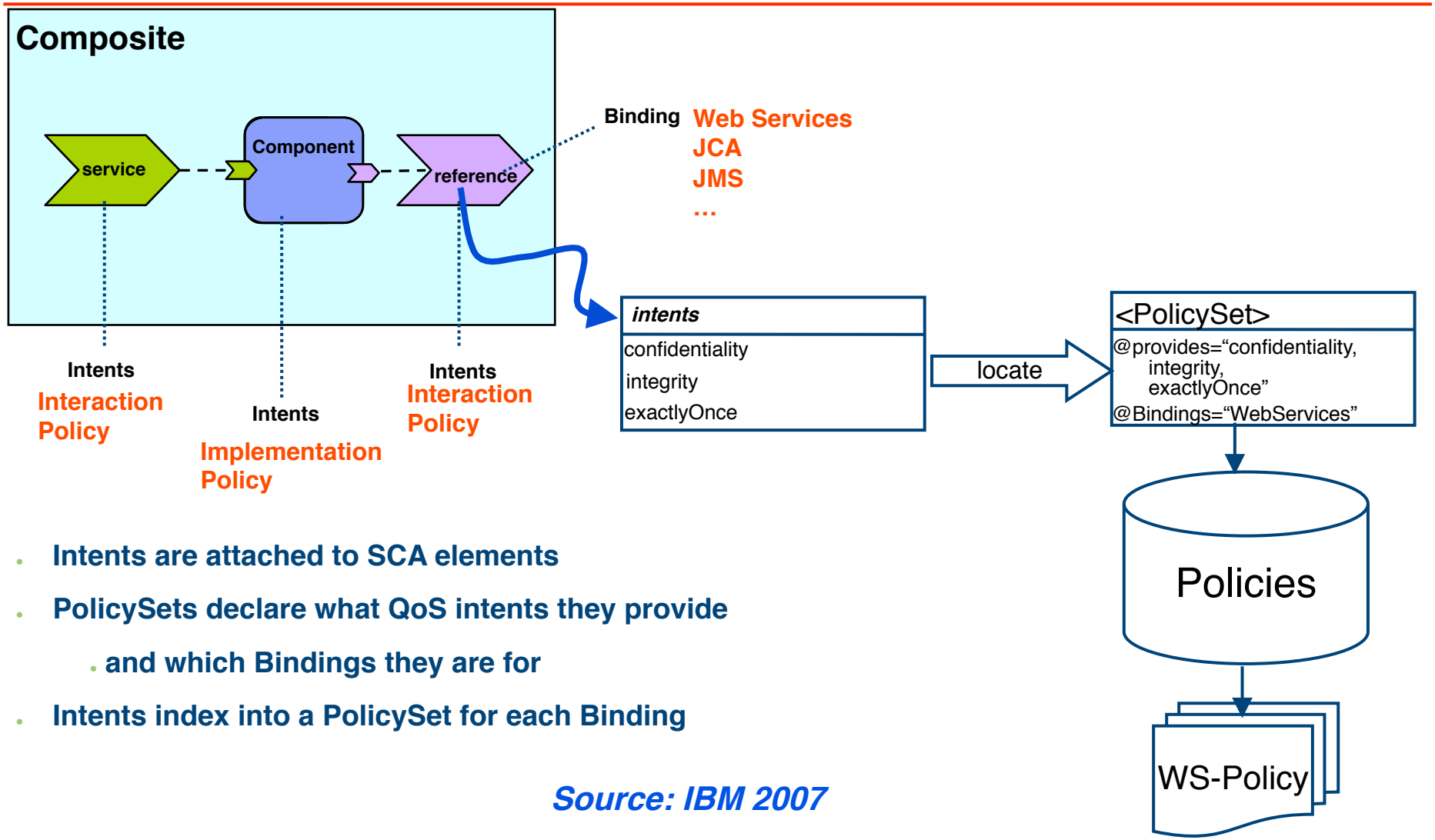


Source: oasis-open.org

Overview of the Policy Framework

- **Policies provide flexibility**
  - The same component can be used in different context with different QoS or Security requirements
  - No impact on the implementation
- **SCA Policy Intent**
  - Specify abstract QoS capabilities or requirements
  - Independent of their concrete realisation
- **SCA Policy Sets**
  - Define concrete collections of policies corresponding to intents
  - Typically apply to specific binding types or implementation types
- **Supports and promotes the use of WS-Policy**

Attaching Profiles and Mapping to PolicySets



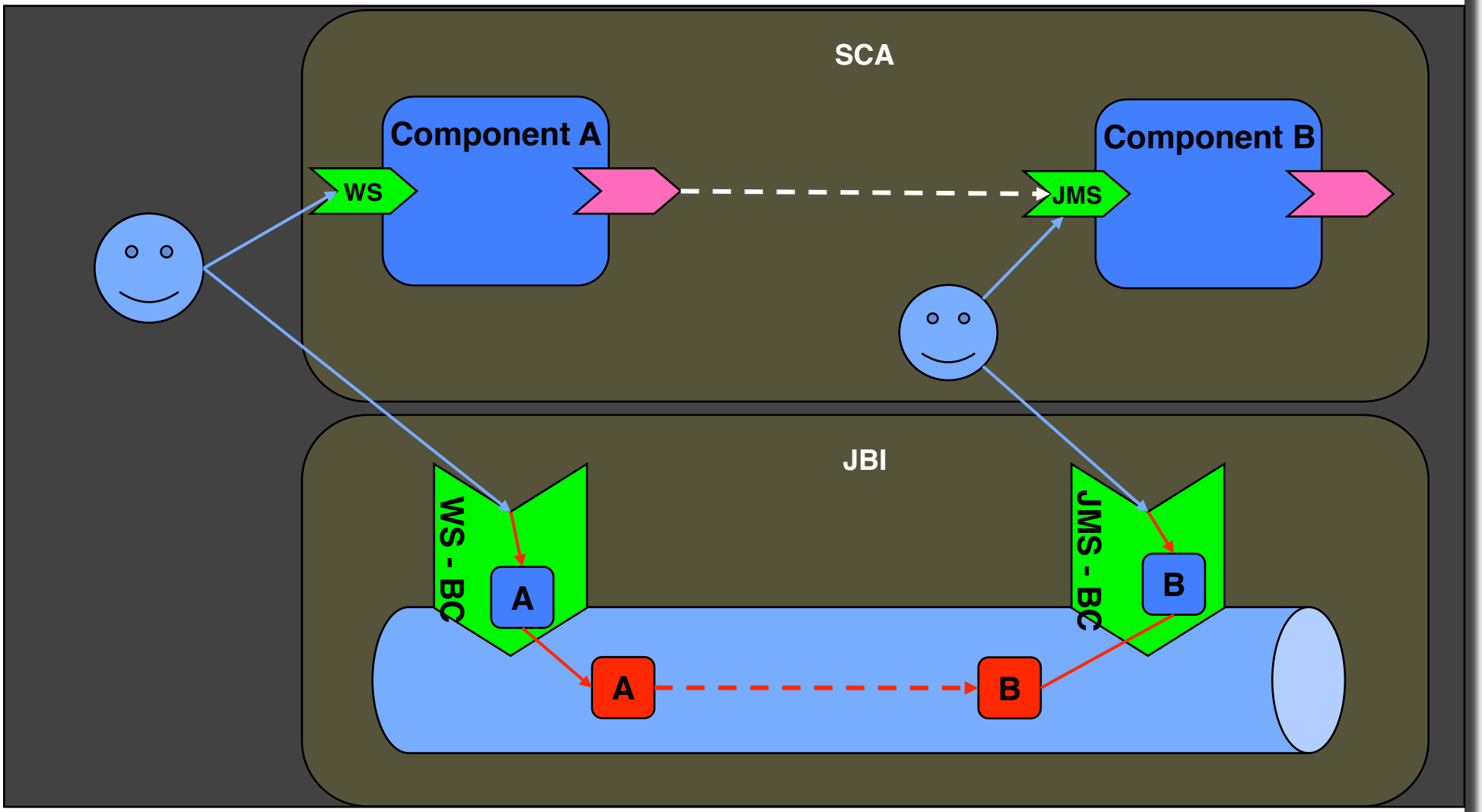
- Intents are attached to SCA elements
- PolicySets declare what QoS intents they provide
  - and which Bindings they are for
- Intents index into a PolicySet for each Binding

Source: IBM 2007

SCA <--> JBI Overview

- SCA and JBI are complementary
- SCA is about architecture - mainly aimed at architects
- JBI is about infrastructure - mainly aimed at infrastructure vendors
- Concepts
  - SCA Composites <--> JBI Service Assemblies
  - SCA Components <--> JBI Service Units
  - SCA Binding Types <--> JBI Binding Components
  - SCA Services <--> JBI endpoints
  - SCA Wires <--> JBI Runtime bindings (DeliveryCh / MessageEx)

SCA <--> JBI Mapping Example



SCA <--> JBI

- Plusieurs discussions à propos de cette relation
- Technologies complémentaires
- SCA pour l'assemblage des composants
- JBI plateforme d'exécution pour la gestion des bindings distants du composite
  - Appeler les références distantes via les Bindings Components
  - Exposer les services du composite comme des endpoints JBI

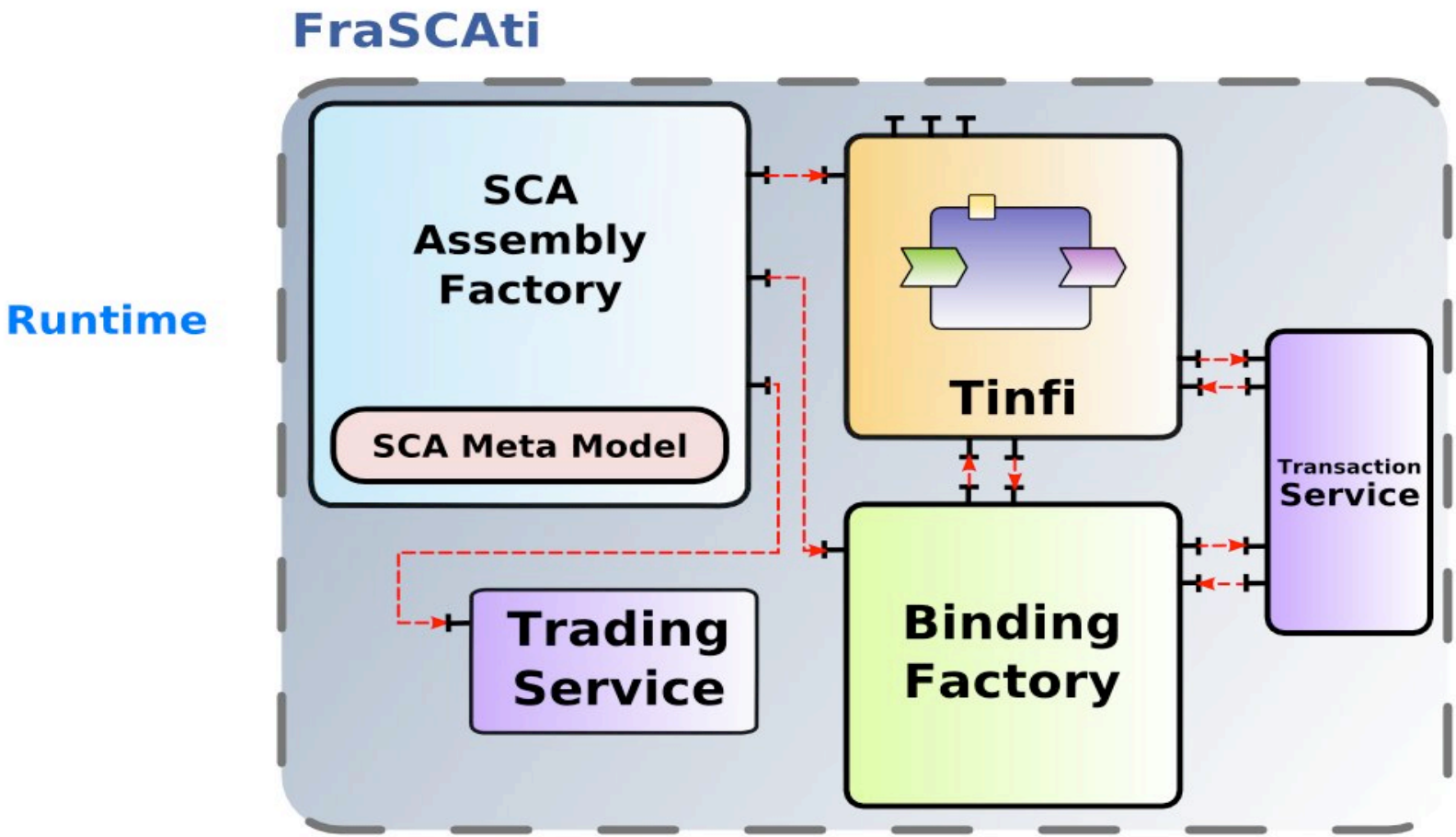
Plan

- **I SOA: présentations (matin)**
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- **I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)**
  - *Introduction à SCA (et relations à JBI)*
  - **Relations à Fractal: Tinfi et la plateforme FraSCAti**
  - *Plateforme PEtALS / FraSCAti*
- *I-3 L'outillage de développement en environnement Eclipse*
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

Plate-forme d'exécution Frascati (1/2)

- Développée dans le cadre du projet SCOrWare
- Vise à implémenter les aspects suivants de la specification SCA
  - Assembly Model
  - Java Common Annotations and APIs
  - Java Component Implementation
  - Web service Binding
  - JMS Binding
- Développement basé sur le modèle de composant FRACTAL

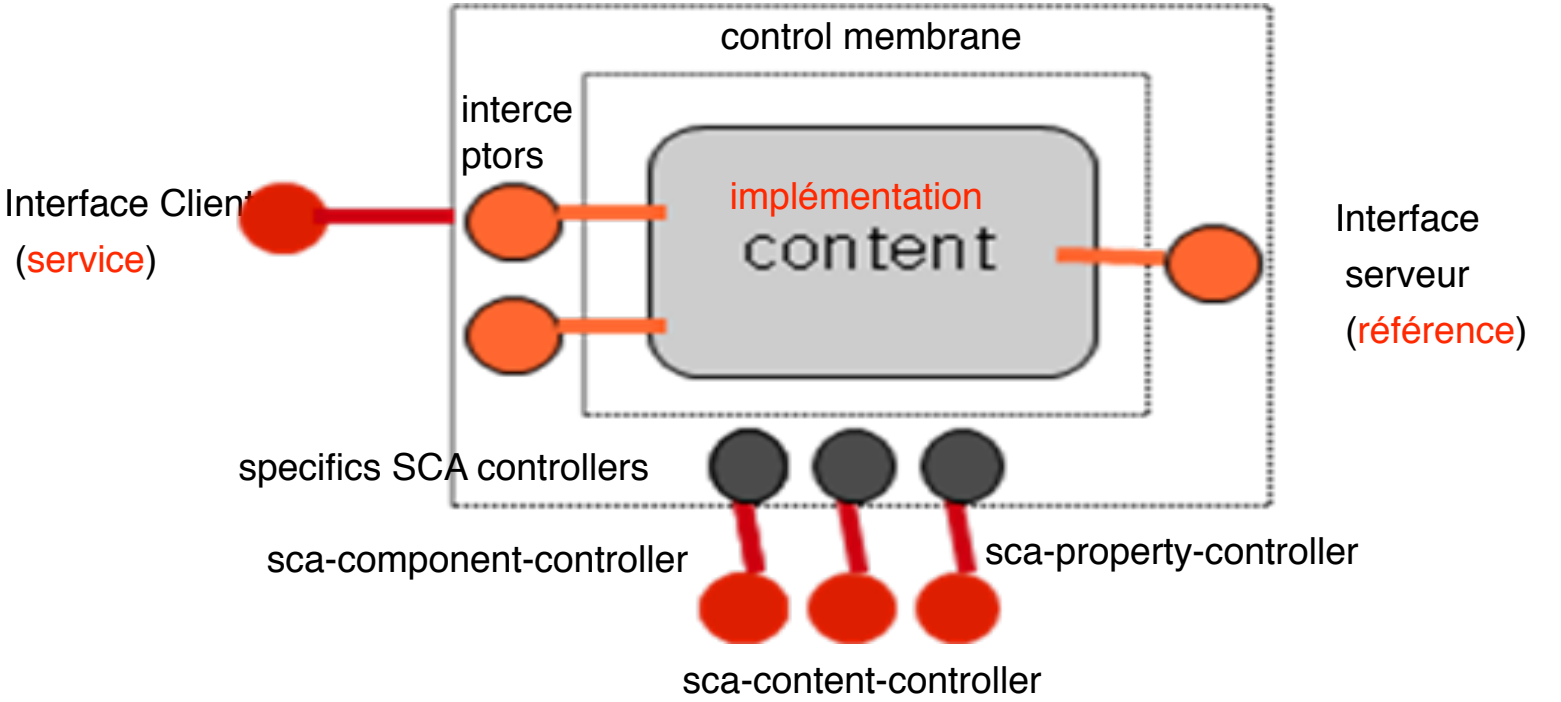
Plate-forme d'exécution Frascati (2/2)



Tinfi

- Tinfi est le noyau d'exécution des composants SCA
- Construit au dessus de FRACTAL
  - Personnalité du modèle de composant FRACTAL
  - Réutilise les contrôleurs standards de FRACTAL et Julia
  - Fournit des fonctionnalités en plus aux composants SCA
    - Reconfiguration
    - Introspection
    - Dynamicité
  - Composant Tinfi : bicéphale (SCA + Fractal)

Composant Tinfi



Plan

- **I SOA: présentations (matin)**
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- **I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)**
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - **Plateforme PEtALS / FraSCAti**
- *I-3 L'outillage de développement en environnement Eclipse*
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

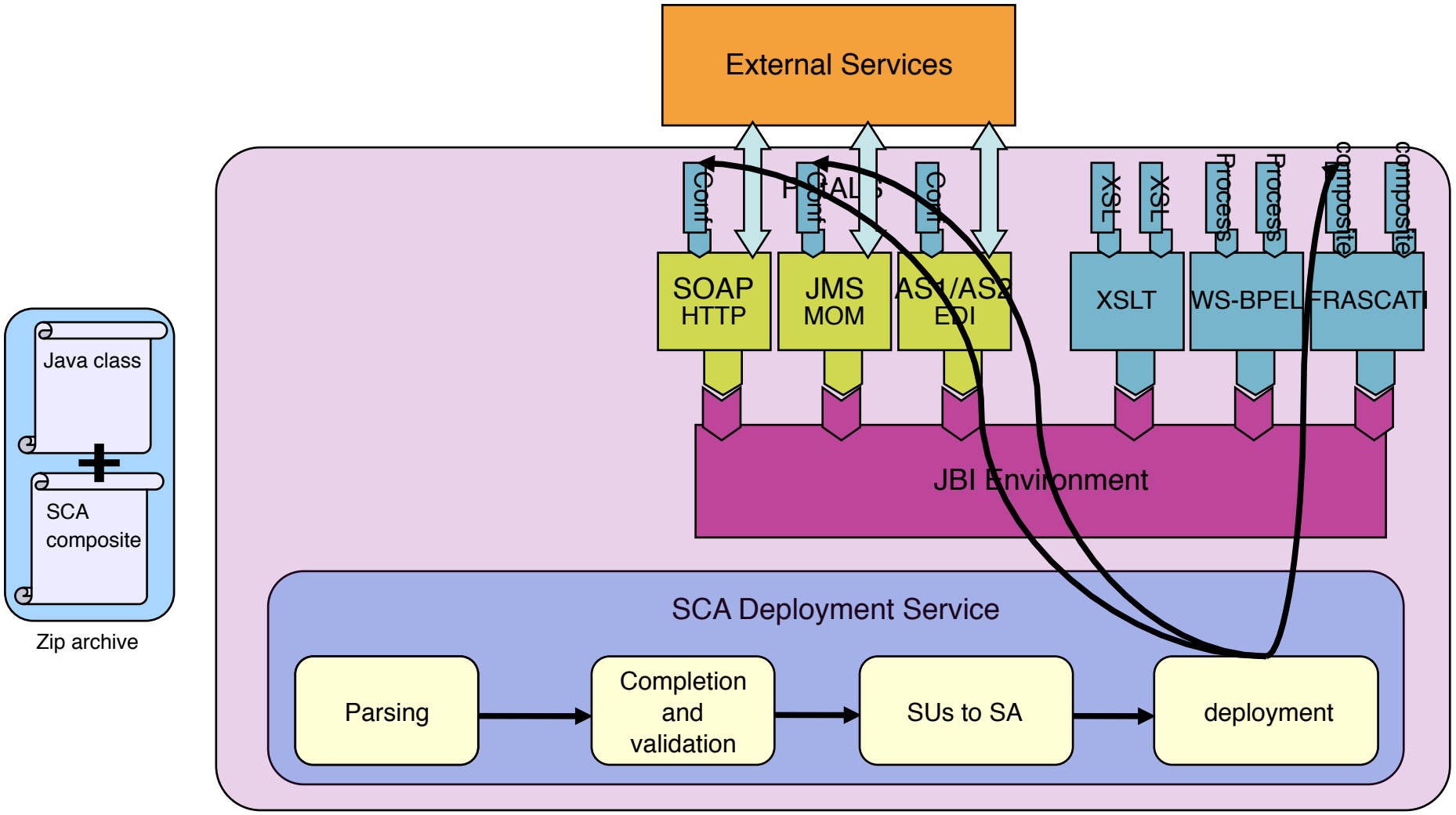
Plateforme PEtALS / Frascati (1/3)

- Frascati est intégré comme un Service Engine dans PEtALS
- Délègue à PEtALS les bindings distants du composant
  - Le composant s'exécute dans le SE FRASCATI
  - Les appels des services et des références distants sont gérés par les Binding Components PEtALS
- Offre au composite SCA la possibilité d'exposer ou de référencer un service dans PEtALS
- Réutilise les Binding Components PEtALS pour le support des bindings (JMS, WS, RMI)

Plateforme PEtALS / Frascati (2/3)

- Permet de reconfigurer les bindings à chaud
  - Une référence qui change d'URL
  - Exposition d'un service avec un nouveau type de binding
- Réutilise les fonctionnalités offertes par PEtALS
  - Monitoring via la console d'administration PEtALS
  - Politiques supportées par PEtALS (fiabilité, sécurité, transaction)

Plateforme PEtALS / Frascati (3/3)



Plan

- **I SOA: présentations (matin)**
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- *I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)*
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - *Plateforme PEtALS / FraSCAti*
- **I-3 L'outillage de développement en environnement Eclipse**
  - **Le projet Eclipse STP**
  - **Focus sur les outils SCA**
  - **Interopérabilité entre outils: le modèle intermédiaire STP-IM**
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

I-3 L'outillage de développement en environnement Eclipse

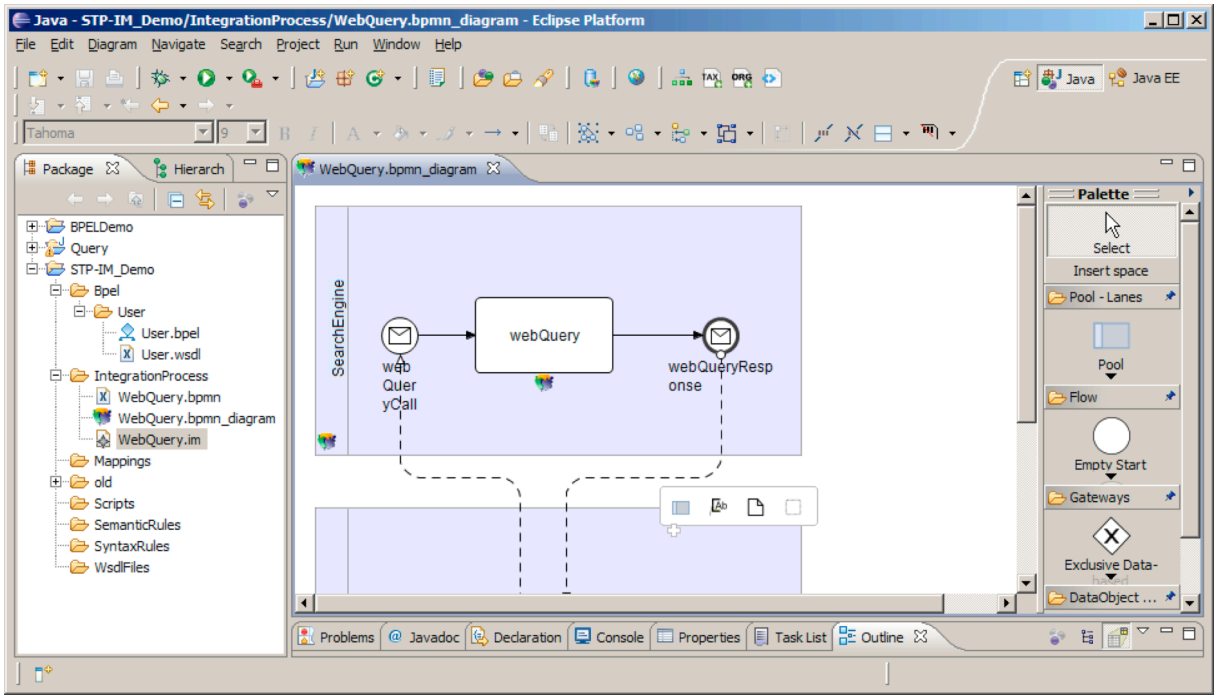
- I-3a Le projet Eclipse STP
  - Eclipse SOA Tooling Platform
  - Divers éditeurs et frameworks
    - Editeur BPMN
    - Enterprise Integration Designer
    - XML Framework
    - Policy Editor
    - Autres …
- I-3b Les outils Eclipse STP SCA
  - Méta-modèle SCA
  - Assistants de création et éditeur
  - Ecriture des implémentations
  - Validation et déploiement
  - Outils de test
  - Extensions des outils et transformations
- I-3c Le modèle intermédiaire STP-IM (Intermediate Model)
  - Intégration et interopérabilité
  - illustration
  - Statut du projet et évolutions
  - Comment contribuer ?!

Eclipse SOA Tools Platform

The mission of the SOA Tools Platform (STP) project is to build frameworks and exemplary extensible tools that enable the design, configuration, assembly, deployment, monitoring, and management of software designed around a Service Oriented Architecture (SOA).

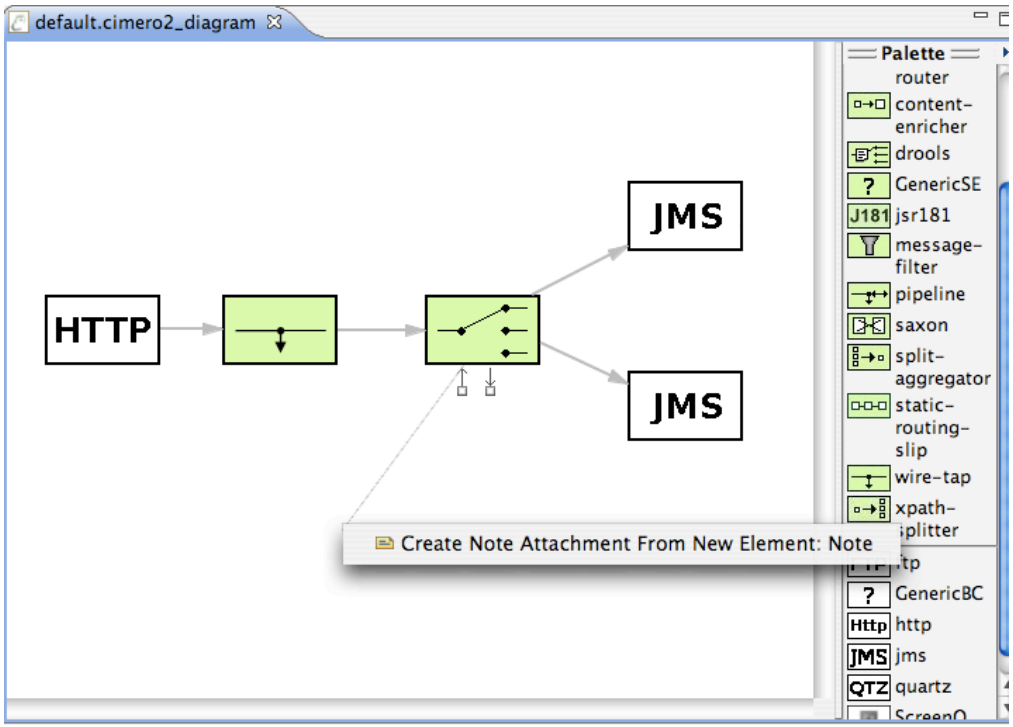
BPMN Editor

- Fully Featured BPMN Editor
- Extensive support for the standard
- Configurable & extensible



The Enterprise Integration Editor

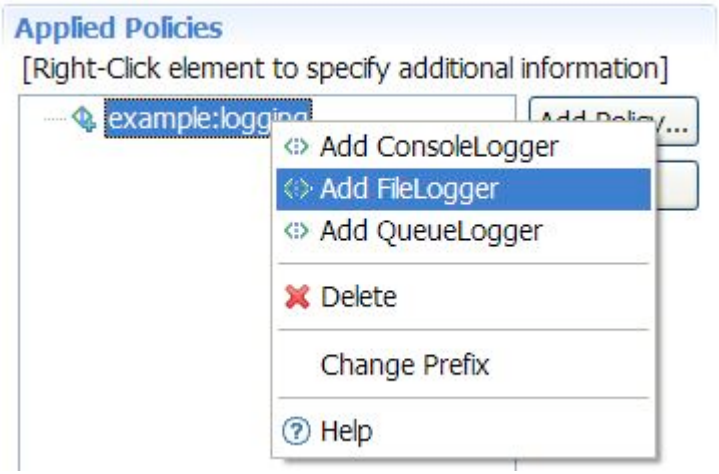
- Editor for integration solutions
- Based on the Enterprise Integration Patterns
- Aims to enable deployment to ESBs



XEF – XML Editing Framework

- **Generate XML editors from XML schemas (XSD)**

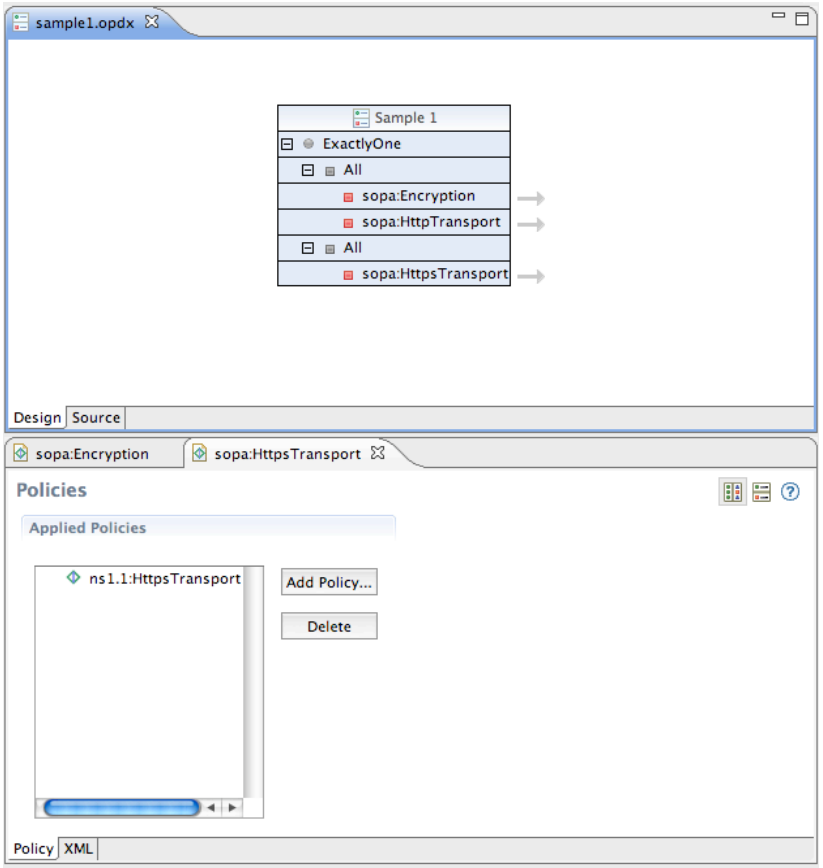
```
<xs:element name="logging">
  <xs:complexType>
    <xs:choice>
      <xs:element name="ConsoleLogger"/>
      <xs:element name="FileLogger"/>
      <xs:element name="QueueLogger"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```



- **Used by the Policy Editor**
- **1 XSD element <=> 1-* possible widgets**
  - **One by default**
  - **Can be changed using XEF annotations in the XSD files (appinfo)**

Policy Editor

- A set of editors to create and edit policies
  - Syntax: WS-Policy
  - Formal definition: WS-*
    - e.g. WS-Addressing
  - Associate policies with policy subjects: WS-PolicyAttachment
- The high level editor
  - Manipulate the structure of the policy
- The Details Editor
  - Edit details of WS-Policy assertions
  - Based on XEF
  - Generate the GUI from the policy XSD



Large Variety of SOA Tools and Platforms

- SOA Development involves different platforms
  - BPMN
  - BPEL
  - Policy
  - EID
  - JAX-WS
  - SCA
  - JBI
- Different roles use different editors / platforms
- Information duplication is inevitable when moving across editors

Plan

- **I SOA: présentations (matin)**
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- *I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)*
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - *Plateforme PEtALS / FraSCAti*
- **I-3 L'outillage de développement en environnement Eclipse**
  - *Le projet Eclipse STP*
  - **Focus sur les outils SCA**
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

SOA Tool Platform – Outillage SCA

- **Objectif : un outillage...**
  - Basé sur les spécifications SCA
  - Évolutif et personnalisable
  - En cohérence avec les diverses plateformes open-source
- **... pour...**
  - Construire
  - Développer
  - Tester
  - Et déployer des applications SCA

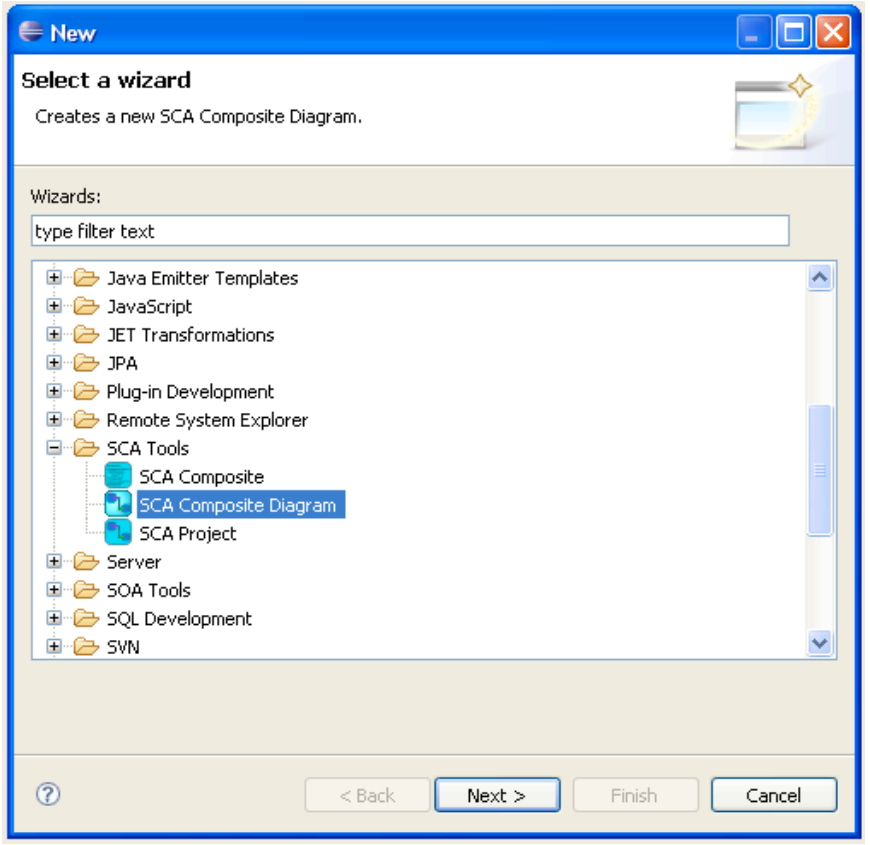
Outillage SCA – Le méta-modèle SCA

- Le cœur de l'outillage SCA
  - Basé sur les schémas XML des spécifications SCA
  - Enrichi au niveau des contraintes
  - Méta-modèle EMF
- Utilisé par
  - SCA Composite Designer
  - Validateur SCA
  - La plateforme SCA Frascati
  - STP-IM et JWT pour des transformations de modèle

Outillage SCA – Assistants de Création

Assistants de création

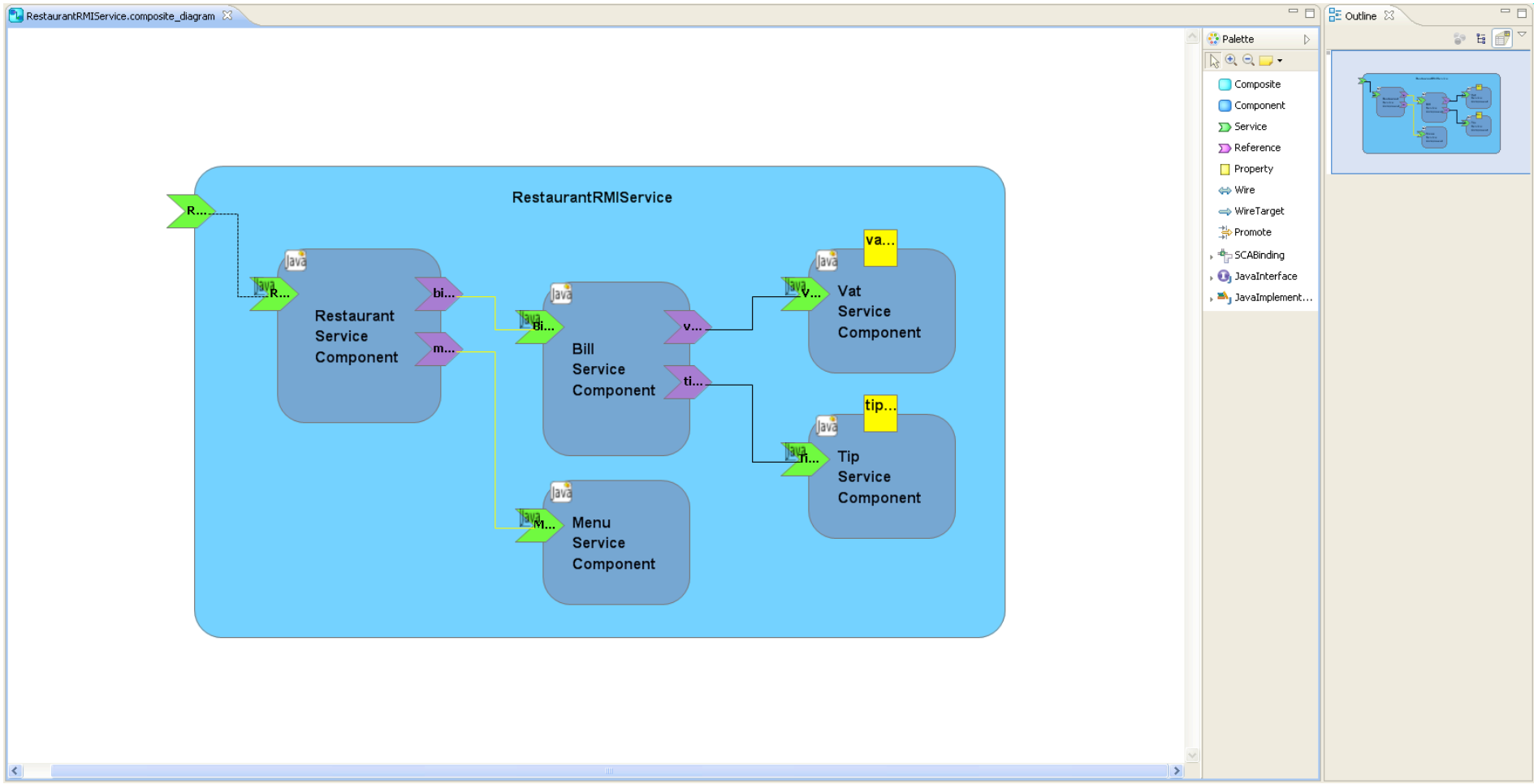
- Créer un projet SCA
  - Projet vide, de nature SCA (implémentation quelconque)
  - Projet Java, de nature SCA
- Créer un fichier *.composite (descripteur de l'assemblage SCA)
- Créer un diagramme SCA
- Créer un fichier *.componentType (« pochoir » de composite / composant)



Outillage SCA – Édition (1/5)

- **Spécifications SCA**
  - Notation XML (*.composite) ET représentation graphique
  - Contrairement à BPMN ou BPEL qui n'ont défini qu'un des deux
  - L'outillage d'édition gère les deux aspects
- **Le SCA Composite Designer**
  - Éditeur graphique pour SCA
  - Basé sur Eclipse GMF
  - Une zone de dessin + une palette pour manipuler les éléments SCA
  - Solution la plus pratique pour prendre en main et réaliser une application SCA
  - Écrire un fichier composite graphiquement

Outillage SCA – Édition (2/5)



Outillage SCA – Édition (3/5)

- Complété par deux éditeurs
- Éditeur XML pour *.composite...
  - Éditer manuellement des descripteurs d'assemblage SCA
- ... et pour *.componentType
  - Décrire le squelette d'un composant ou d'un composite
  - Syntaxe = sous-ensemble de celle d'un composite
- Fonctionnalités
  - Étend l'éditeur XML d'Eclipse
  - Aperçu personnalisé par rapport à l'éditeur XML classique
  - Auto-complétion
    - Noms et valeurs des balises et des attributs

Outillage SCA – Édition (4/5)

RestaurantRMIService.composite

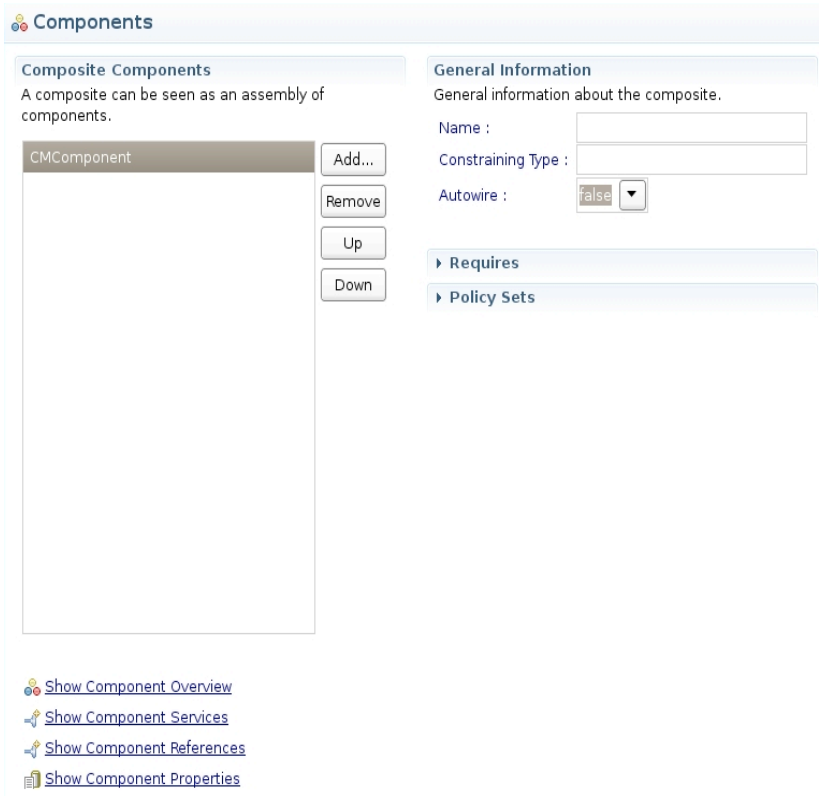
```
<sca:composite
    xmlns:sca="http://www.osoa.org/xmlns/sca/1.0"
    xmlns="http://www.osoa.org/xmlns/sca/1.0"
    xmlns:tuscany="http://tuscany.apache.org/xmlns/sca/1.0"
    name="RestaurantRMIService">

    <sca:component name="RestaurantServiceComponent">
        <sca:implementation.java
            class="restaurant_rmi_service.lib.RestaurantServiceImpl" />
        <sca:service name="RestaurantService">
            <sca:interface.java interface="restaurant_rmi_service.api.RestaurantService" />
            <binding.rmi host="localhost" port="8099" serviceName="RestaurantServiceRMI" />
        </sca:service>
        <sca:reference name="billService" target="BillServiceComponent/BillService" />
        <sca:reference name="menuService" target="MenuServiceComponent/MenuService" />
    </sca:component>

    <sca:service name="RestaurantService"
        promote="RestaurantServiceComponent/RestaurantService" />

    <sca:component name="MenuServiceComponent">
        <sca:implementation.java class="restaurant_rmi_service.lib.MenuServiceImpl" />
        <sca:service name="MenuService">
            <sca:interface.java interface="restaurant_rmi_service.api.MenuService" />
        </sca:service>
    </sca:component>

    <sca:component name="BillServiceComponent">
        <sca:implementation.java class="restaurant_rmi_service.lib.BillServiceImpl" />
        <sca:service name="BillService">
            <sca:interface.java interface="restaurant_rmi_service.api.BillService" />
        </sca:service>
        <sca:reference name="vatService" />
        <sca:reference name="tipService" />
    </sca:component>

    <sca:component name="TipServiceComponent">
        <sca:implementation.java class="restaurant_rmi_service.lib.TipServiceImpl" />
        <sca:service name="TipService">
            <sca:interface.java interface="restaurant_rmi_service.api.TipService" />
        </sca:service>
        <sca:property name="tipRate">15</sca:property>
    </sca:component>

    <sca:component name="VatServiceComponent">
        <sca:implementation.java class="restaurant_rmi_service.lib.VatServiceImpl" />
        <sca:service name="VatService">
```

Outline

- Restaurantrmiservice
  - Billservicecomponent
    - Billservice
      - interface.java
    - Tipservice
    - Vatservice
    - implementation.java
  - Menuservicecomponent
    - Menuservice
      - interface.java
    - implementation.java
  - Restaurantservicecomponent
    - Restaurantservice
      - binding.rmi
      - interface.java
    - Billservice
    - Menuservice
    - implementation.java
  - Tipservicecomponent
    - Tipservice
      - interface.java
    - Tiprate
    - implementation.java
  - Vatservicecomponent
    - Vatservice
      - interface.java
    - Vatrate
    - implementation.java
  - Restaurantservice
  - BillServiceComponent/tipService
  - BillServiceComponent/vatService

Outillage SCA – Édition (5/5)

- **Éditeur *Form* pour fichiers *.composite**
  - Éditer le descripteur de l'assemblage SCA
  - Éditeur multi-page
  - Interface de type « formulaire Web »
  - 1 « concept » = 1 page
  - Gestion des éléments sous forme de listes
    - Plus adapté aux composite avec de nombreux éléments



Outillage SCA – Écriture des Implémentations

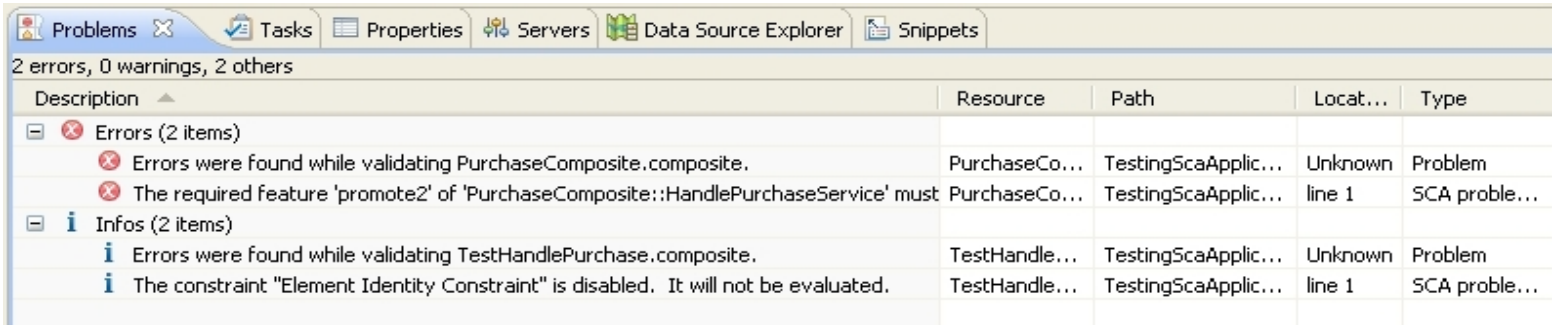
- Implémentations
  - Java, C/C++, BPEL...
  - Outillage déjà existant dans Eclipse
- Eclipse : environnement qui intègre de quoi écrire une application SCA complète
- Java : support des annotations SCA
  - Écriture des implémentations
  - Flexibilité sur l'écriture du composite
- Introspection de code (à l'étude)
  - Partir d'implémentations annotées et générer un fichier *.composite

Outillage SCA – Validation (1/2)

- Écriture d'un programme Java dans Eclipse
  - Validateur Java qui vérifie la correction syntaxique du code
- Validation SCA : même principe pour SCA
  - Valider une application SCA du point de vue de la spécification SCA
- Validateur basé sur le méta-modèle SCA
  - Application SCA = instance du méta-modèle SCA
  - Instance construite depuis le composite...
  - ... et depuis le code (introspection de code – à l'étude)
  - Valider cette instance par rapport au méta-modèle...
  - ... et par rapport à des règles externes
    - Typiquement, cohérence pour les mécanismes d'inclusion

Outillage SCA – Validation (2/2)

- **Erreurs trouvées => afficher des marqueurs d'erreurs**
  - **Dans les fichiers composite**
  - **Sur le diagramme (à l'étude)**

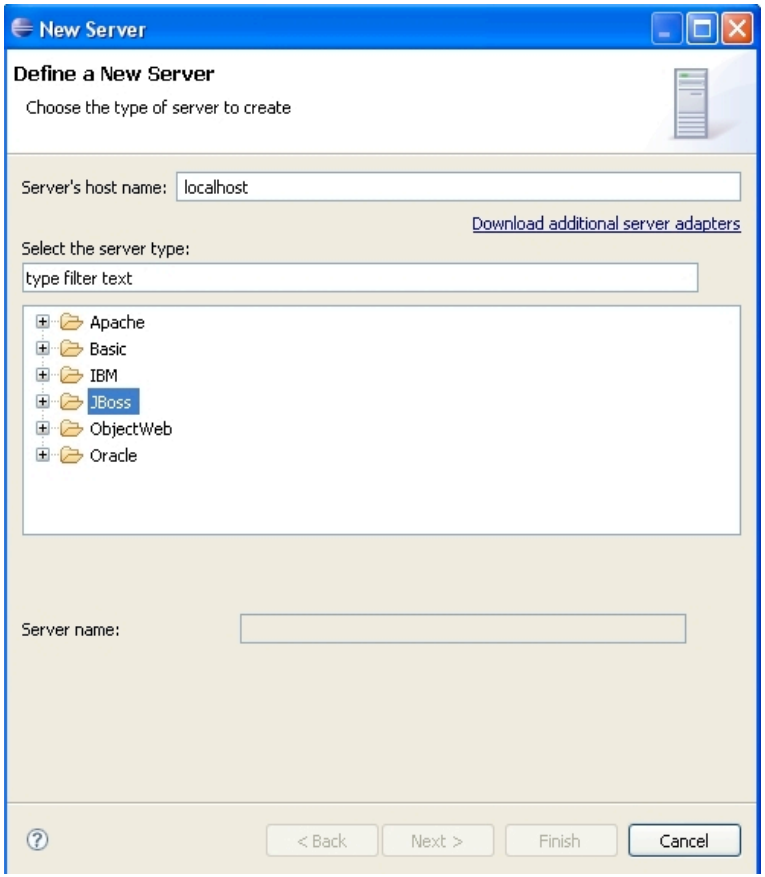


Outillage SCA – Déploiement (1/3)

- Plusieurs plateformes SCA open-source
  - Apache Tuscany
  - PEtALS – FraSCAti (projet ANR SCOrWare)
  - Fabric3
- Contacts répétés avec la communauté STP
- Supporter le déploiement d'applications SCA sur ces plateformes
- Deux aspects
  - Déployer et lancer des applications SCA depuis Eclipse (développement)
  - Créer des archives de déploiement / lanceurs SCA (production)

Outillage SCA – Déploiement (2/3)

- Déployer / lancer des applications SCA depuis Eclipse
  - Test durant le développement
- Vue « serveur »
  - Une plateforme = un type de serveur dans Eclipse
  - Déployer une application sur un serveur et interagir avec elle depuis Eclipse
- => serveur Tuscany, PEtALS et Fabric3



Outillage SCA – Déploiement (3/3)

- Créer des archives de déploiement / lanceurs SCA
  - Lancer simplement une application SCA
  - Tuscany – Fabric3 : plateformes « pur SCA »
  - FraSCAti : plateforme « pure SCA » basée sur Fractal
  - PEtALS – FraSCAti
    - ESB basé sur JBI avec...
    - ... un Service Engine SCA basé sur FraSCAti
- Déployer une application SCA sur PEtALS – FraSCAti
  - Empaqueter cette application dans un « Service Assembly »
  - Création d'un zip ayant une structure particulière...
  - ... et contenant l'application SCA et un descripteur JBI

Outillage SCA – Test (1/2)

- En cours d'étude
- Proposer des outils pour tester une application SCA
  - Test unitaire (tester un composant ou un composite)
  - Test d'intégration (des composants dans un composite)
- Test unitaire
  - Élément testé = composant OU composite
    - Ensemble de services (fonctionnalités offertes)
    - Ensemble de références (dépendances, fonctionnalités requises)
  - Tester les services, bouchonner les références et vérifier des assertions sur les résultats
  - Utilisation de JUnit et de EasyMock

Outillage SCA – Test (2/2)

- Test d'intégration
  - Même principes techniques que le test unitaire
  - Tester l'intégration progressive des références bouchonnées par des références effectives
- Principe des tests
  - Application SCA à tester
  - Écrire des tests = Générer une application SCA de test
  - L'utilisateur n'a que les TestCases à compléter
- A la base, inspiré par ce que Fabric3 a fait
- Tests lancés depuis Eclipse
  - Réflexion en cours pour embarquer ces tests dans Maven

Outillage SCA – Extensibilité des outils

- Méta-modèle SCA
  - Basé sur les spécifications SCA
- Les plateformes définissent de nouveaux éléments
  - Implémentations, bindings, interfaces
- Permettre aux utilisateurs d'étendre les outils
  - Rajouter des bindings, des implémentations, etc...
- Méta-modèle et SCA Composite Designer
  - Point d'extension en cours de développement
- Éditeurs XML et *form*
  - Page de préférence pour définir de nouveaux éléments

Outillage SCA – Transformations de Modèles

- Méta-modèle SCA
  - Réalisé avec Eclipse EMF
- Passer d'un modèle SCA à un autre modèle
  - Exemple : BPMN
- Transformations en cours de développement
  - Pont établi entre SCA et JWT (Java Workflow Tooling)
  - Pont avec les autres outils SOA via STP-IM
    - BPMN (en cours)
    - JBI
    - ...

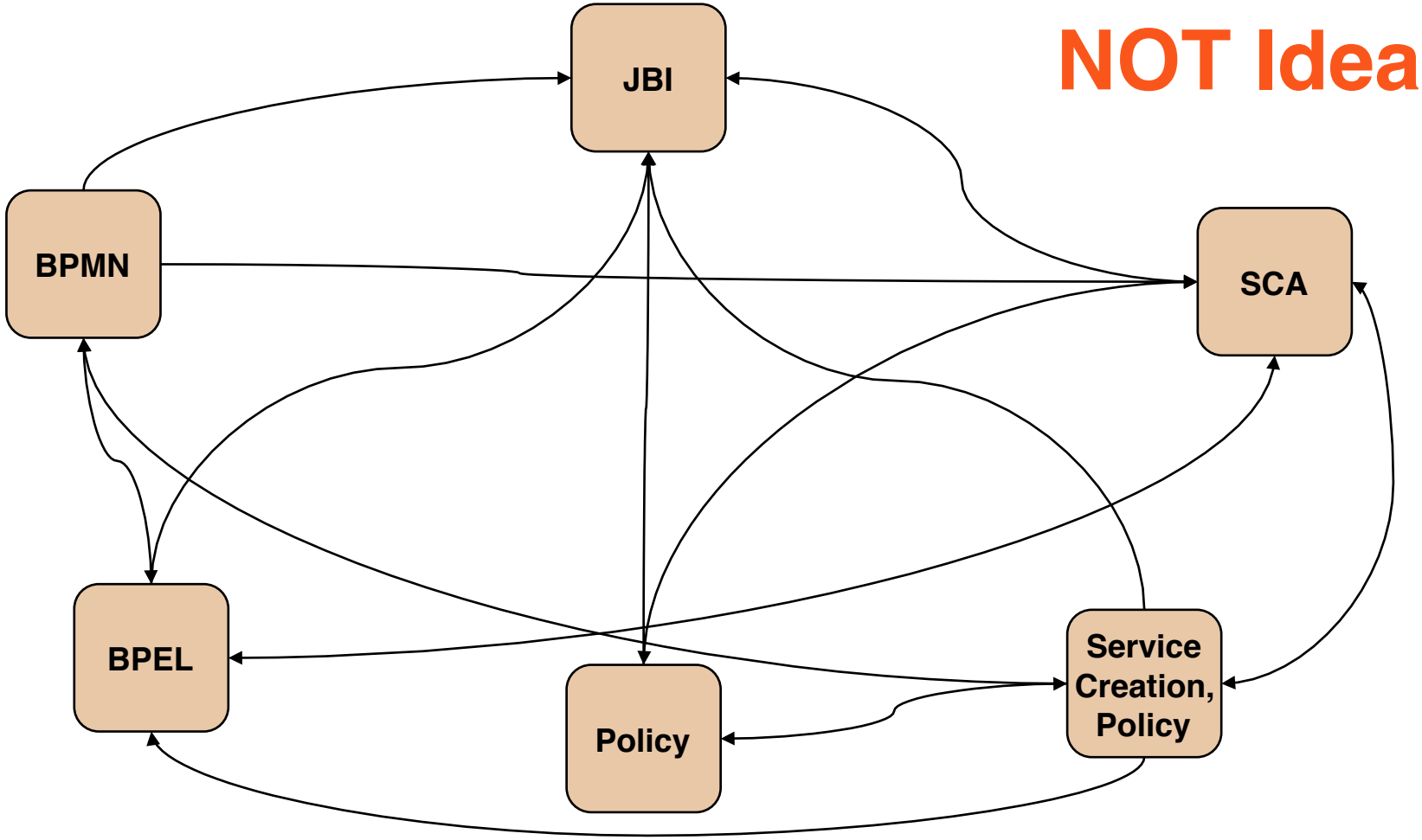
Plan

- **I SOA: présentations (matin)**
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- *I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)*
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - *Plateforme PEtALS / FraSCAti*
- **I-3 L'outillage de développement en environnement Eclipse**
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - **Interopérabilité entre outils: le modèle intermédiaire STP-IM**
- *II SOA: démonstrations (après-midi)*
  - *II-1 Cas d'étude « Voyage »*
  - *II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM*
  - *II-3 Mise en œuvre avec SCA*
  - *II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA*

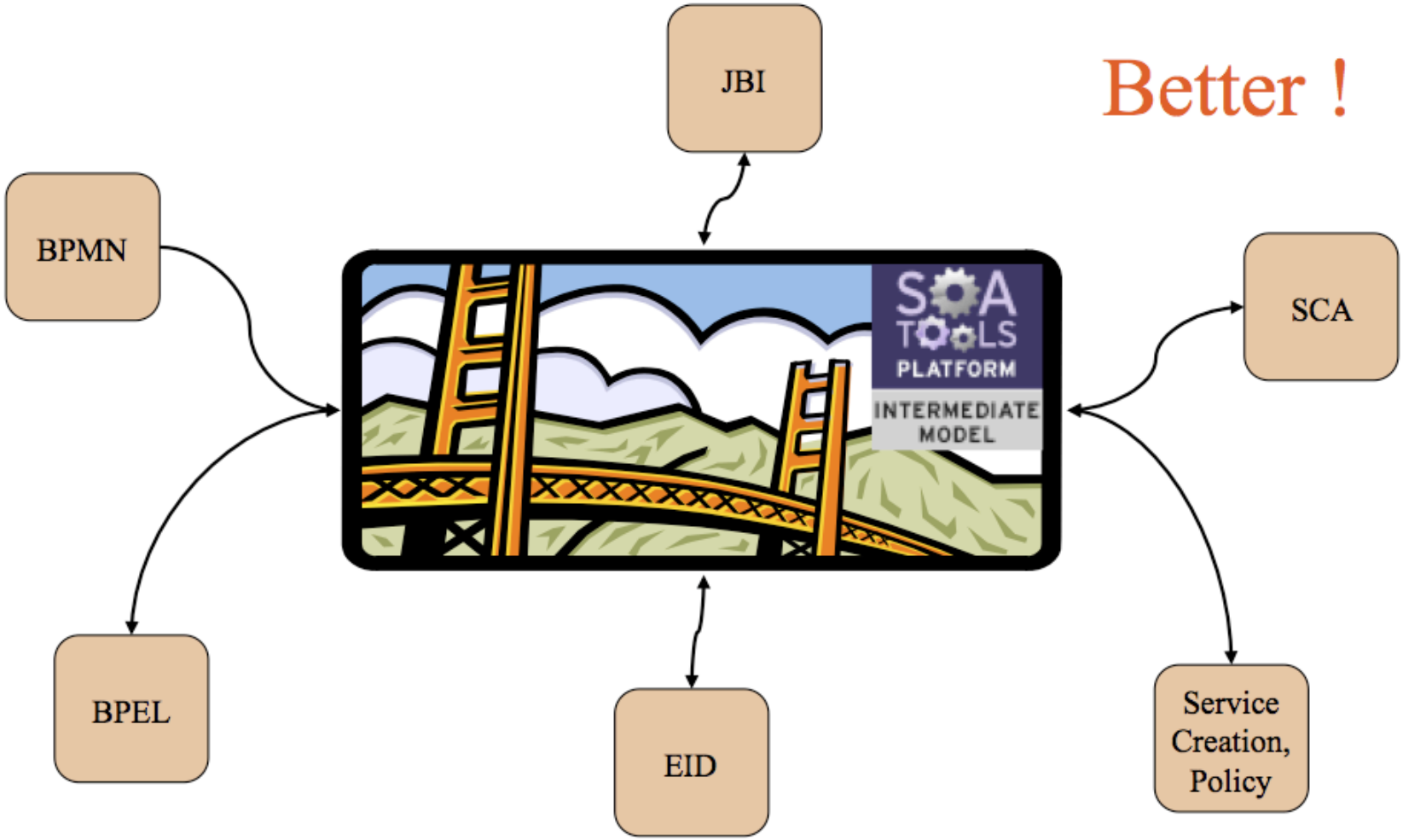
Intermediate Model Overview

- Bridges different SOA platforms in STP
  - Workflow / process: e.g. BPMN, BPEL
  - Architecture specification: e.g. SCA, EID, JBI
  - Service Creation: e.g. JAX-WS, Policy Specification
- Facilitates interoperability between editors
  - Provides a central SOA conceptual bridge
  - Avoids duplication of data
  - Minimizes amount of transformation code
  - Facilitates code generation from a variety of sources
- Initial Contribution: INRIA (FR) and Engineering (IT)
  - STP component: org.eclipse.stp.model
  - EMF model plugins + transformation plugins
  - Used in the Spagic 2.0 SOA Suite from Engineering

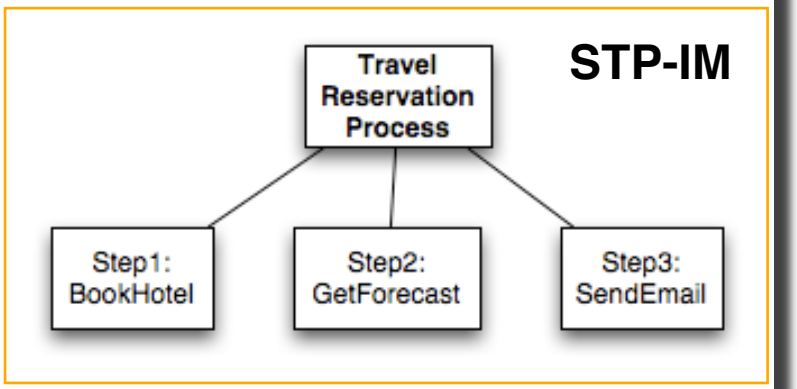
Integrating SOA Editors - A First Take



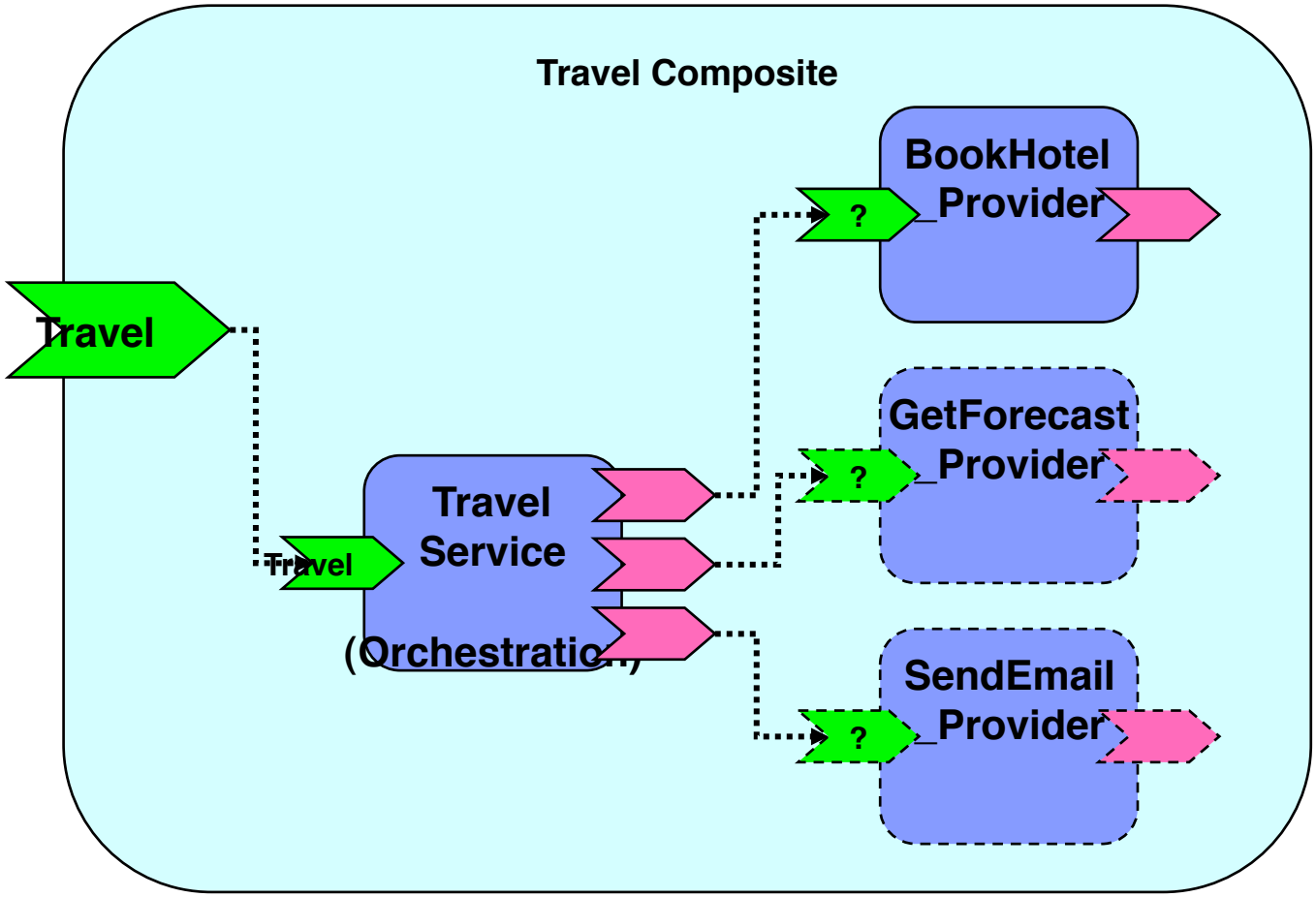
Bridging SOA Editors with STP-IM

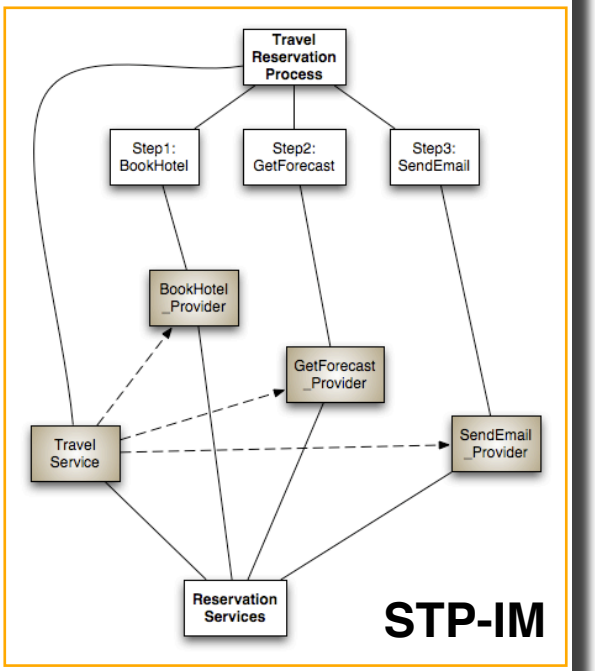


“Travel” Reservation Scenario

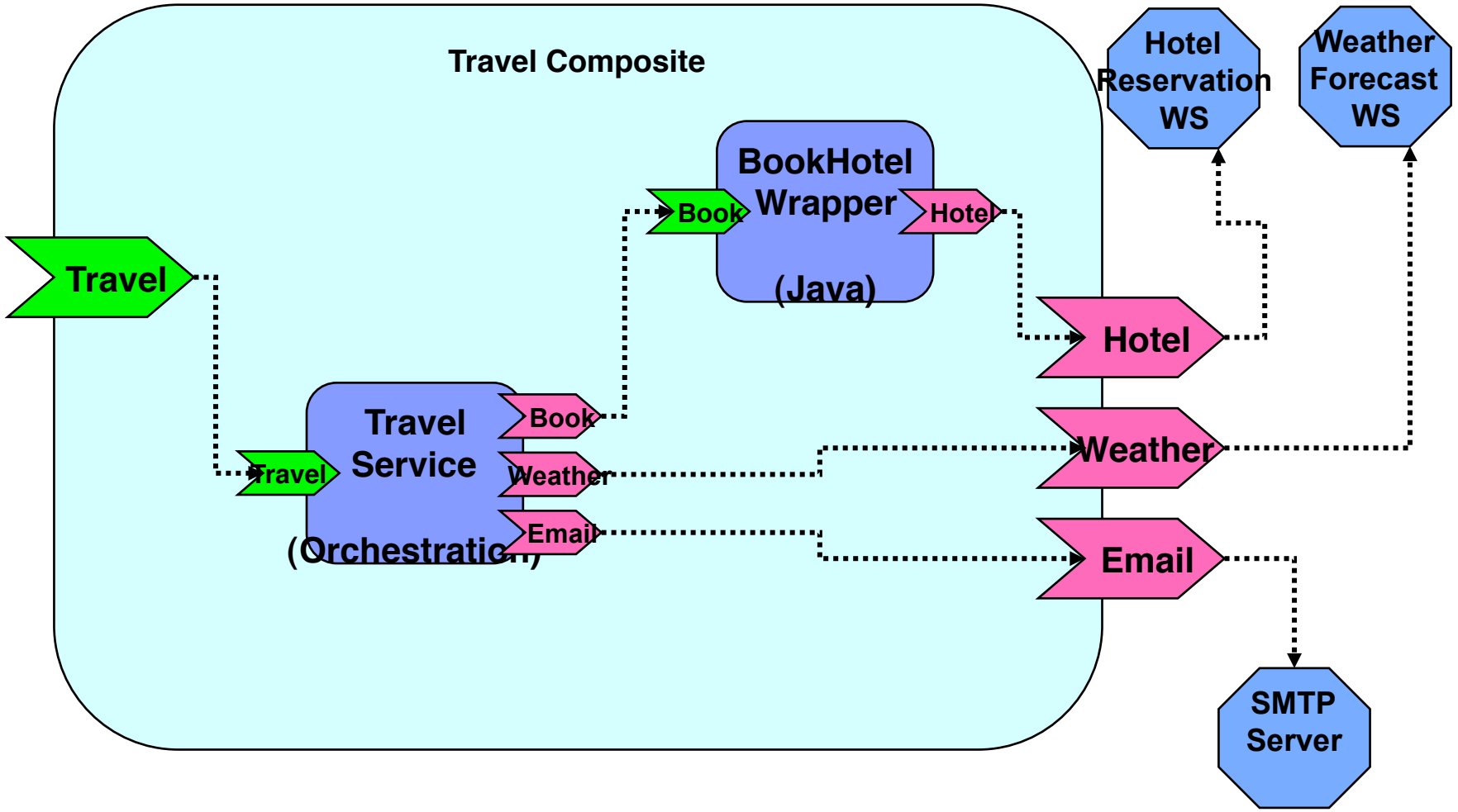


First SCA Diagram

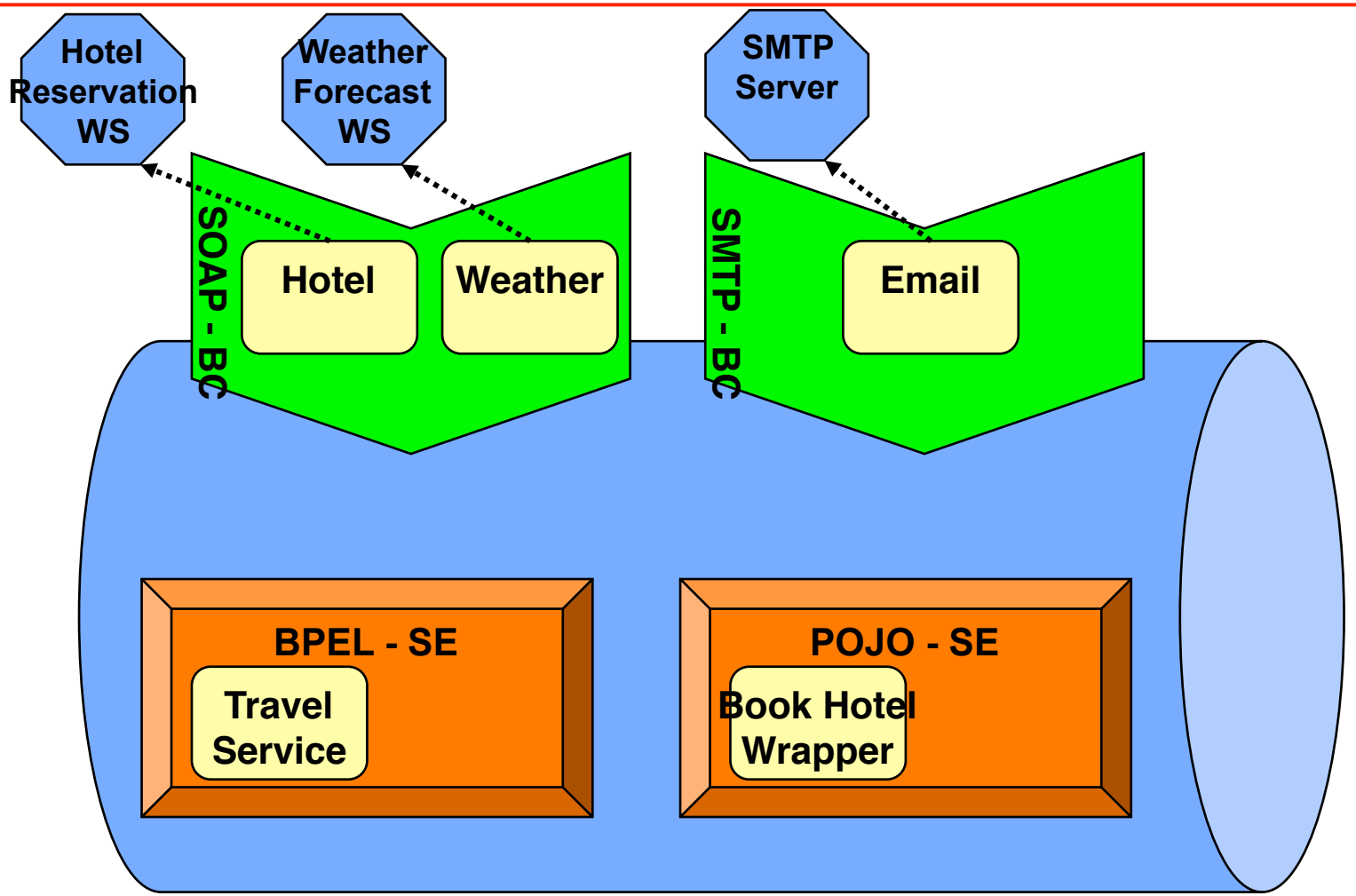




SCA Diagram 2



JBI



Current Status

- Improves the overall functionality of STP
- Available Transformations (in the repository)
  - BPMN to STP-IM
  - SCA to STP-IM (basic functionality)
  - STP-IM to SCA (basic functionality)
- Runtime extension capabilities
- Used in production in Spagic 2.0
  - Additional JBI support
  - BPEL support (to be transferred to STP)

Upcoming Contributions

- New transformation plug-ins
  - STP-IM to BPEL (immediate release)
  - STP-IM ←→ EID
  - STP-IM ←→ Service Creation
  - Eclipse JWT → STP-IM
- Improvements and Extensions to existing plug-ins
- Updated documentation: wiki and guides
- Google Summer of Code - Juan Cadavid
  - Funded project to contribute to STP-IM
  - Declarative Transformations: BPMN-BPEL-SCA-EID
  - A single builder-approach for better editor integration

Get and Contribute to STP-IM

- **Download Eclipse Ganymede!**
  - Use the Ganymede Update Site to get the full STP
- **Location:**
  - HTTP://www.eclipse.org/stp/im
  - SVN://…/stp/org.eclipse.stp.model
- **Plugin Structure:**
  - org.eclipse.stp.im (**STP-IM model**)
  - org.eclipse.stp.im.runtime.* (**e.g. bpel, jbi**)
  - org.eclipse.stp.im.in.* (**e.g. bpmn, sca**)
  - org.eclipse.stp.im.tool.in.* (**e.g. bpmneditor**)

Pour aller plus loin

- **Sites WEB & URLs diverses**
  - http://www.scorware.org
  - http://www.w3c.org
  - http://www.osoa.org
  - http://www.oasis-open.org
  - http://java.sun.com
    - http://java.sun.com/javaone
  - http://www.bpmn.org (OMG, http://www.omg.org)
  - http://www.soa-consortium.org (OMG, http://www.omg.org)
  - https://ow.inrialpes.fr/projects/jones/
  - http://petals.objectweb.org
  - http://www.eclipse.org/stp
    - http://wiki.eclipse.org/STP
  - http://tuscany.apache.org
  - http://fabric3.codehaus.org
  - http://mule.mulesource.org

Plan

- *I SOA: présentations (matin)*
- *I-1Introduction à SOA*
  - *Concepts, éléments clé et standards*
  - *Point de vue ➔ conception de système d'information*
  - *Point de vue ➔ développement SCA autour d'une infrastructure ESB JBI*
- *I-2 SCA et la plateforme d'exécution Open Source (projet SCOrWare)*
  - *Introduction à SCA (et relations à JBI)*
  - *Relations à Fractal: Tinfi et la plateforme FraSCAti*
  - *Plateforme PEtALS / FraSCAti*
- *I-3 L'outillage de développement en environnement Eclipse*
  - *Le projet Eclipse STP*
  - *Focus sur les outils SCA*
  - *Interopérabilité entre outils: le modèle intermédiaire STP-IM*
- **II SOA: démonstrations (après-midi)**
  - **II-1 Cas d'étude « Voyage »**
  - **II-2 Architecture applicative et projection sur architecture technique (BPMN / SCA) avec STP-IM**
  - **II-3 Mise en œuvre avec SCA**
  - **II-4 Déploiement, exécution et supervision sur une plateforme JBI SCA**