WebSphere®

IBM®

Une publication IBM avec la contribution de Gartner

Juillet 2008

Business Event Processing ou le traitement des événements opérationnels .2

Tutorial for EDA and How It Relates to SOA : 20

Business Event Processing ou le traitement des événements opérationnels

Featuring research from Gartner®

Business Event Processing ou le traitement des événements opérationnels

Avertissement

Un certain nombre de concepts introduits dans le présent document, sont internationalement connus, sous des dénominations anglo-saxonnes. J'ai essayé, là où cela s'avérait nécessaire à la compréhension, de traduire ces vocables en français, en essayant de rester fidèle à l'idée véhiculée. J'ai pris donc le principe de traduire systématiquement le mot « **business** », essentiel dans la compréhension de ce document par « **opérations** » ou « **opérationnel** » qui m'a paru plus approprié que la traduction « **métier** » généralement employée et qui ne veut pas dire grand-chose. Par opération ou opérationnel, il faut donc comprendre dans la suite de ce document, **opérations de l'Entreprise**, et non pas opérations informatiques, qui n'en sont, après tout, qu'un cas particulier.

Hubert Lalanne - Distinguished Engineer - IBM Software Group

La vie est une succession d'événements…

Que ce soit dans la vie professionnelle ou la vie privée, on passe son temps à réagir à des événements.

Traiter ou réagir à des événements est donc un comportement très familier à la nature humaine. Ce comportement suit toujours à peu près les mêmes étapes :

- Détection d'un ou plusieurs **événements**,
- Mise en évidence d'une **alerte** potentielle,
- **Diffusion de l'information**, ou **notification** des intéressés,
- **Action** ou suite **d'actions correctives ou réactives**, ou déroulement d'un **processus de traitement ou d'analyse** de l'alerte

Bien souvent, on ne réagit pas un événement mais à une conjonction d'événements (« je suis sorti sans parapluie et il pleut ») ou d'événements et de non événements (« il faut que je paye mes fournisseurs mais mes clients n'ont toujours pas acquitté leurs factures, je vais avoir des problèmes de trésorerie »). De même les événements peuvent être fortuits ou planifiés.

Une approche par événements est tout aussi naturelle dans la gestion d'une entreprise. Les opérationnels (managers, professionnels ou analystes) connaissent très bien les événements importants pour l'entreprise. Au travers d'une détection d'une combinaison d'événements, ils peuvent identifier une situation importante pour l'Entreprise, c'est-à-dire une menace ou une opportunité, et réagir en conséquence.

Ce peut être pour :

- Etre plus réactif aux besoins des clients, par exemple en s'assurant qu'on atteint les niveaux de service et de ce fait éviter pénalités ou poursuites
- Utiliser plus efficacement les ressources de l'entreprise, par exemple en étant capable de prendre des décisions proactives pour réduire les impacts de retard de livraison des fournisseurs
- D'une manière plus générale, être capable de mieux gérer des chaînes de valeur complexes, dont on ne peut appréhender tous les comportements

Un des intérêts de l'approche par événement est qu'elle est justement itérative et évolue par auto-apprentissage :

Elle peut commencer par la détection d'événements simples mais d'importance majeure et la simple notification des intéressés puis évoluer vers la détection de combinaison d'événements plus complexes et l'automatisation progressive des processus de traitement ou d'analyse.

La question qui se pose est de savoir comment et avec quels moyens technologiques mettre en œuvre cette approche au niveau des Systèmes d'Information des entreprises.

En d'autre termes, il s'agit de fournir **aux opérationnels** les moyens de « relier les pointillés » d'événements disparates pour identifier la « signature » d'une situation opérationnelle pour l'entreprise. C'est la finalité des approches de **traitement des événements** ou **Event Processing**

L'apport d'une gestion efficace des événements

Qu'est ce qu'un événement au sens informatique du terme ?

On définira un **événement** comme étant tout signal électronique indiquant que quelque chose d'important s'est produit, se produit ou va se produire. Les exemples sont infinis.

- Une puce RFID passe sous un portique
- Quelqu'un consulte une base de données
- Un véhicule est localisé par GPS
- Une commande d'un montant xx est reçue
- On anticipe un délai moyen de livraison supérieur à xx
- Le serveur Web est saturé
- Un client sur le Web ajoute un produit à son chariot
- Un client change son mot de passe sur le site de banque en ligne
- Le baril de pétrole passe la barre des 200 US$
- …

Pourquoi est-ce important de pouvoir traiter les événements ?

Prenons un exemple pour situer la problématique de traitement des événements dans un contexte opérationnel : la livraison de colis. Le processus de base est assez simple : l'expéditeur demande l'enlèvement de son colis par téléphone ou par Internet. Le transporteur organise le transport, répartit les envois dans les camions et planifie les livraisons. Le centre de transferts trie les colis et organise les livraisons. Ce processus peut être assez aisément en grande partie automatisé En effet, avec des technologies comme le GPS, le RFID, ou plus simplement, les lecteurs de codes barre fixes ou portables, il est possible de détecter les **événements de passage** des colis à des endroits précis de la chaîne de distribution et donc de suivre virtuellement chaque étape du processus, et permettre ainsi la supervision, la gestion, l'analyse et l'optimisation du processus nominal.

Maintenant, il y a d'autres **événements ou non événements** à prendre en compte : des enlèvements peuvent être manqués, des livraisons être en retard ou des colis être perdus. Ces événements, importants ou bénins peuvent survenir et devenir le grain de sable dans le rouage bien huilé du processus le mieux conçu et le mieux géré, ce qui peut avoir un impact direct sur les opérations et l'activité de l'entreprise. Il faut donc aussi pouvoir détecter ces événements et contribuer, progressivement, par leur traitement, à rendre beaucoup plus efficace le processus, capable maintenant de prendre en compte des situations exceptionnelles.

Ce raisonnement peut s'étendre à tous les secteurs d'activités et la plupart des activités considérées peuvent donc tirer parti d'un traitement efficace des événements.

De la livraison de colis, au traitement d'ordres complexes de marché ou la fabrication de pièces dans une usine, tout peut être envisagé comme une succession d'événements, planifiés ou non.

Quelques exemples

Toutes les entreprises sont tenues de mettre en place des contraintes ou des exigences légales et de pouvoir superviser ou auditer leur application. L'Event Processing peut fournir aux responsables en charge cette mise en place afin de tracer les exceptions ou les déviances par rapport à ces exigences, en minimisant l'investissement en développement spécifique.

Dans le domaine de la santé, on peut concevoir un monitoring personnalisé des malades Par exemple, le malade est branché à de multiples outils de monitoring (à l'hôpital ou à son domicile). Le médecin peut définir et faire évoluer, lui-même, en se

basant sur des sources multiples et sur l'historique du patient, les règles d'envoi d'alertes, ainsi que les destinataires concernés.

Exemple de règles :

- Si la pression sanguine est trop élevée de 10 %, constamment pendant 2 jours ou plus, alerter le médecin
- si la fièvre est supérieure à 40° et la pression sanguine en hausse par rapport à la veille, appeler l'infirmière

Dans la banque, on peut imaginer des scénarios de détection de fraude au travers de règles de corrélation d'événements comme par exemple :

- Niveau 1 de suspicion quand un client effectue :
  - 3 tentatives ou plus de connexion infructueuse au niveau du site de banque en ligne
  - Retrait important depuis un distributeur dans les 24 heures suivant le changement du code de la carte
- 3 événements « Niveau 1 de suspicion » déclenchent un événement « Niveau 2 de suspicion » (indépendamment de ce qui a pu causer l'événement original)
- 3 événements « Niveau 2 de suspicion » signifient une probabilité élevée de fraude
- Action : déclencher une enquête sur la fraude supposée en alertant le professionnel en charge du compte

Dans la distribution, un responsable de magasin pourrait mettre en place des étagères « intelligentes » dans son magasin.

Exemple de règles de traitement d'événements :

- Alertes au niveau d'un rayon en particulier :
  - Plus de 25% de ceux qui ont pris un produit donné dans ce rayon l'y ont reposé dans les 2 dernières heures.
  - Plus de 30 % d'un produit a été enlevé dans les 15 dernières minutes
  - Un produit a été positionné par erreur dans ce rayon
- Au niveau de l'activité globale du magasin :
  - Un produit donné n'a été pris dans aucun rayon dans les 2 dernières heures
  - Un objet a été pris dans un rayon mais n'a pas été enregistré à la caisse dans les 2 heures qui ont suivi

Positionnement de l'Event Processing dans le Système d'Information

Les Systèmes d'Information modernes des entreprises visent à plus d'intégration et permettent d'automatiser un certain nombre de processus opérationnels. Pour que ces entreprises soient plus réactives, il faut pouvoir mettre en œuvre des moyens de détection des événements opérationnels (**business events**), des règles opérationnelles de filtrage et de corrélations, et faire le lien avec **la ou les actions** visant à traiter ou analyser l'événement.

Divers concepts adressent de manière plus ou moins complète ce besoin. On peut citer par exemple le Business Activity Monitoring (BAM), le Complex Event Processing (CEP), l'Event Stream Processing (ESP), l'Event Driven architecture (EDA) et d'autres.

Bien que parfois les entreprises font du traitement d'événements au quotidien, ces efforts sont souvent improvisés, partiels et sans cohérence globale au niveau du Système d'information.

Pourquoi maintenant ?

L'Event Processing n'est pas nouveau. Il est déjà utilisé dans les outils de gestion et de supervision des infrastructures ou dans les systèmes de sécurité. Les systèmes de messageries inter-applicatifs mettent déjà en œuvre des systèmes de publication/souscription qui s'apparentent à de l'Event Processing.

L'Event Processing contribue à la ***tendance générale qui vise à améliorer la réactivité du Système d'Information de l'Entreprise***. Il contribue à diminuer le temps de réaction entre

- La mise en évidence d'une situation opérationnelle
- La prise de décision par rapport à cette situation
- L'exécution des actions liées à cette décision

De nombreux facteurs opérationnels ont accéléré le besoin en moyens d'Event Processing comme la validation en temps réel de la conformité (ou non conformité) aux contraintes légales, les exigences de réduction de coûts induisant une plus grande automatisation des opérations.

L'évolution technologique rend aussi possible aujourd'hui la détection d'un très grand nombre d'événements opérationnels divers et variés, et la mise en place des moyens de traitement puissants. On peut citer par exemple l'apparition des nouveautés technologiques comme le RFID.

L'évolution de l'architecture des Systèmes d'information au travers d'approche comme le SOA (Service Oriented Architecture) vise à une meilleure adaptation des moyens informatiques à l'évolution des besoins tactiques ou stratégiques de l'Entreprise (« reduce the gap between business and IT »). Ils ont pour but d'améliorer la réactivité et la visibilité sur le fonctionnement opérationnel de l'Entreprise. L'Event Processing peut compléter utilement ces approches.

Informatique, informaticiens et traitement des événements

Comment le logiciel prend-il en compte les événements ?

L'informatique a toujours pris en compte le traitement d'événements. Mais les paradigmes logiciels et les langages sont étroitement liés aux paradigmes matériels : les événements sont considérés comme des **exceptions**, comme par exemple une division par zéro.

Toutefois certains systèmes informatiques sont, par nature, complètement basés sur des approches événementielles. On peut par exemple évoquer les systèmes temps réels (systèmes de contrôle de processus industriels ou de ligne de fabrication), ou les systèmes embarqués, les systèmes de surveillance et de sécurité, les systèmes de simulation comme les bancs de tests.

Les systèmes informatiques supportant les réseaux d'échange de transactions boursières sont, par nature, complètement basés sur la gestion d'événements.

Ces systèmes sont mis en œuvre en utilisant ou non des produits logiciels d'Event Processing « spécialisés ». Comme illustration, dans le domaine industriel, on peut évoquer **l'Event Correlation Engine** de **l'Offre Manufacturing Integration Framework** d'IBM. Dans le domaine de la supervision des processus industriels ou de la supervision des infrastructures industrielles, les solutions **d'historisation de données (Data Historian)**, intègrent également des moyens de traitement d'événements.

Au niveau des Systèmes d'Information de gestion plus classiques, l'Event Processing a fait son entrée il y a quelques années, par les systèmes de gestion et de supervision des réseaux et des infrastructures. Des logiciels IBM comme **Tivoli Enterprise Console** ou **Netcool Impact**, sont clairement des solutions d'Event Processing, spécialisés sur des événements liés aux infrastructures, et à l'usage des opérationnels des centres informatiques.

Plus récemment, l'introduction de l'e-business dans les Systèmes d'Informations et l'ouverture de ces systèmes via Internet, a induit la mise en place de systèmes de sécurité et de détection des intrusions et des fraudes. Là encore, on a fait appel à des technologies d'Event Processing. La solution IBM, **Tivoli Compliance Insight Manager** est un bon exemple.

Business Event Processing ou le traitement des événements opérationnels

Le traitement des événements est-il naturel pour les informaticiens ?

Cependant, la plupart des ingénieurs logiciels préfèrent dans leur conception une approche déterministe, un peu à l'encontre des principes de l'Event Processing. De ce fait ils n'utilisent ces concepts qu'à la marge (traitement d'exception) et sont même assez peu familiers avec de simples modèles de conception basés sur l'occurrence d'un événement, la publication d'un document ou la diffusion d'un message.

Une des explications peut-être que l'approche événementielle et essentiellement non déterministe c'est-à-dire qu'on réagit, on corrige des situations ce qui permet de faire évoluer et de rendre plus efficace le système, plutôt que d'essayer d'anticiper toutes les situations possibles. Or la démarche de conception des informaticiens est essentiellement déterministe : il faut anticiper tous les cas de figure au travers de l'algorithme du programme, la logique du processus ou l'intelligence d'une machine à états. Pas vraiment de mettre en place un système qui puisse évoluer par auto-apprentissage.

De ce fait, ils n'utilisent pas l'Event Processing dans des scénarios ou il s'avérerait pourtant bien utile.

Déjà assez peu sensibles au concept, fréquemment, ils comprennent encore moins comment ils pourraient l'appliquer au contexte opérationnel de l'Entreprise, c'est-à-dire savoir quels événements et quelles combinaisons d'événements sont importants, et quel traitement leur appliquer. Ceci relève du savoir-faire des opérationnels ?

En conséquence, l'Event Processing a été, jusqu'à une période récente largement confiné à des applications spécialisées, telles qu'évoquées ci-dessus.

Les logiciels spécifiques d'Event Processing

Dans un passé récent, sont apparus des logiciels commerciaux de traitement plus générique d'événements.

On parle maintenant de **Complex Event Processing** (CEP) ou **traitement d'événements complexes**. Le CEP adresse en premier lieu la problématique de traitement d'événements complexes ou de flots complexes d'événements (en volume ou en fréquence ou en contraintes de temps de réaction) ou de règles complexes de corrélation d'événements.

Les solutions de CEP proposent des moteurs capables de traiter en masse des événements capturés au niveau du nuage d'événements (« **event cloud** ») dans le but d'identifier les événements signifiants. Le CEP a développé un certain nombre de techniques particulières qui sont développées plus loin dans ce document. Elles sont liées à la détection, à la virtualisation, au filtrage, au routage à l'enrichissement et à la corrélation des événements. Ces règles de corrélations peuvent s'appuyer sur des contraintes de temps de causalité ou d'appartenance à tel ou tel domaine opérationnel.

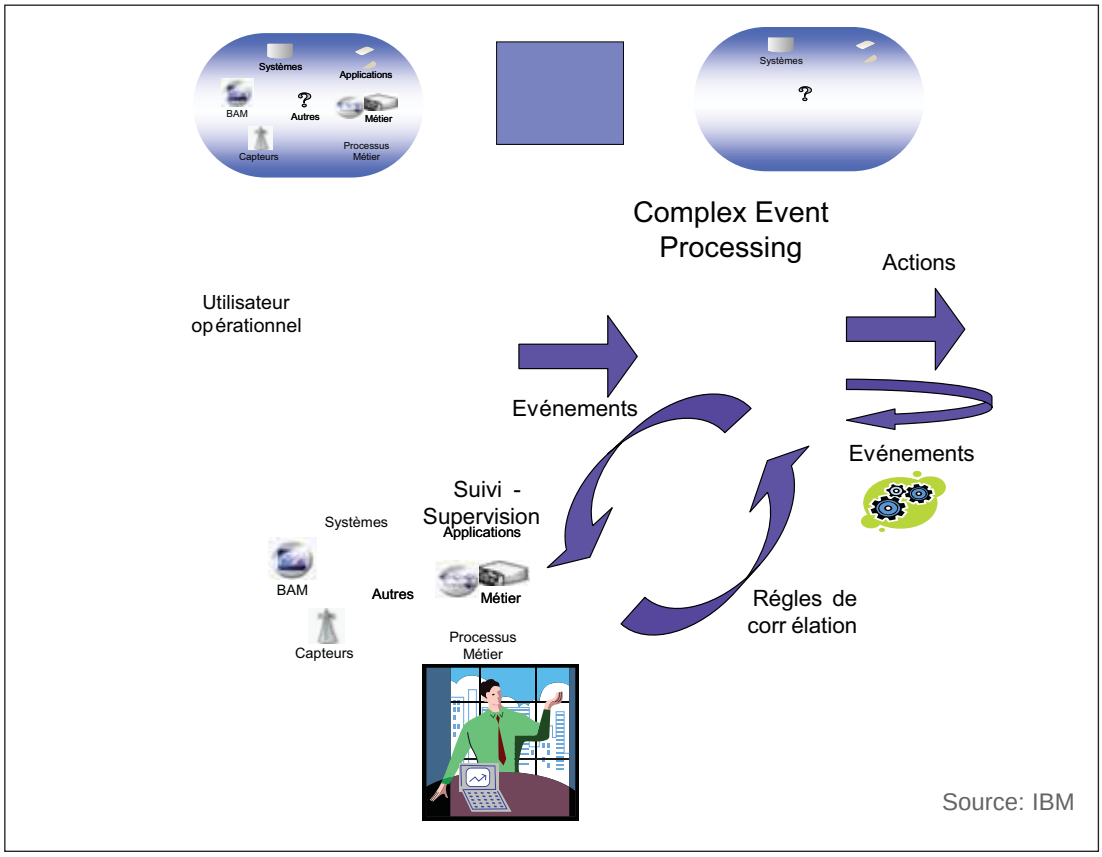
Qu'est ce qu'il manque ?

Le problème est que la plupart des outils d'Event Processing ou de Complex Event Processing sont écrits en utilisant des langages divers et variés, mais toujours basés sur des extensions des paradigmes de programmation traditionnels (SQL pour l'Event Stream Processing, langages de scripting spécifiques, règles d'inférence, ….).

Le développeur type « d'applications » d'Event Processing reste l'informaticien. Comme nous l'avons vu plus haut, ça ne correspond pas forcément à sa manière naturelle de concevoir les choses.

Par contre les fonctionnels sont plus portés vers un comportement moins déterministe. C'est-à-dire qu'ils comprennent très bien ce que sont les événements opérationnels qui importent pour l'Entreprise, parce qu'ils ont à les gérer au quotidien. Cependant, ils connaissent peu ou pas la valeur qu'une approche événementielle pourrait apporter à leur processus et combi leurs applications seraient plus efficaces si elles étaient capables de tirer parti de cette approche. De plus, il n'est pas toujours facile pour eux de s'extraire de leur tâches quotidiennes, pour formaliser leur manière de travailler, l'expliquer aux informaticiens et contribuer à l'amélioration à long-terme de ces applications.

Ce manque d'expérience des opérationnels sur les coûts et les bénéfices liés à l'Event Processing complique aussi le calcul de retour sur



investissement des ces solutions et par-là même les décisions d'investissement dans les technologies correspondantes.

Pour que l'Event Processing s'impose dans l'Entreprise, il faut donc que les opérationnels s'approprient le sujet. C'est-à-dire qu'ils en apprennent les concepts et les finalités afin de pouvoir prendre les bonnes décisions quant à sa mise en œuvre, et, également, que l'Informatique leur fournisse les moyens d'y contribuer activement.

C'est la finalité du ***Business Event Processing***.

Qu'est ce que le Business Event Processing ?

Définition et principaux composants

Le ***Business Event Processing*** vise à fournir la possibilité de capter les signaux électroniques caractérisant l'occurrence d'une situation opérationnelle nécessitant une action, et de fournir aux opérationnels les moyens pour définir et coordonner la réponse appropriée avec le délai de réaction adéquat.

La figure ci-dessous résume les principaux composants d'une architecture de Business Event Processing :

Le composant clef est l'outillage proposé aux opérationnels de l'Entreprise pour pouvoir interagir avec l'infrastructure de Complex Event Processing (CEP), en charge de dérouler les opérations sur les événements.

Le CEP, à partir d'un flot d'événements opérationnels, provenant de sources diverses et variées, va détecter des situations opérationnelles importantes (menaces ou opportunités) et déclencher des actions ou générer de nouveaux événements.

L'utilisateur opérationnel doit pouvoir être à même de lui indiquer quand agir, savoir quoi faire et d'évaluer l'impact des actions. C'est-à-dire qu'il faut lui donner des moyens puissants d'interaction avec le CEP.

Business Event Processing ou le traitement des événements opérationnels

Et il doit pouvoir le faire, sans avoir à se former à des concepts informatiques complexes…

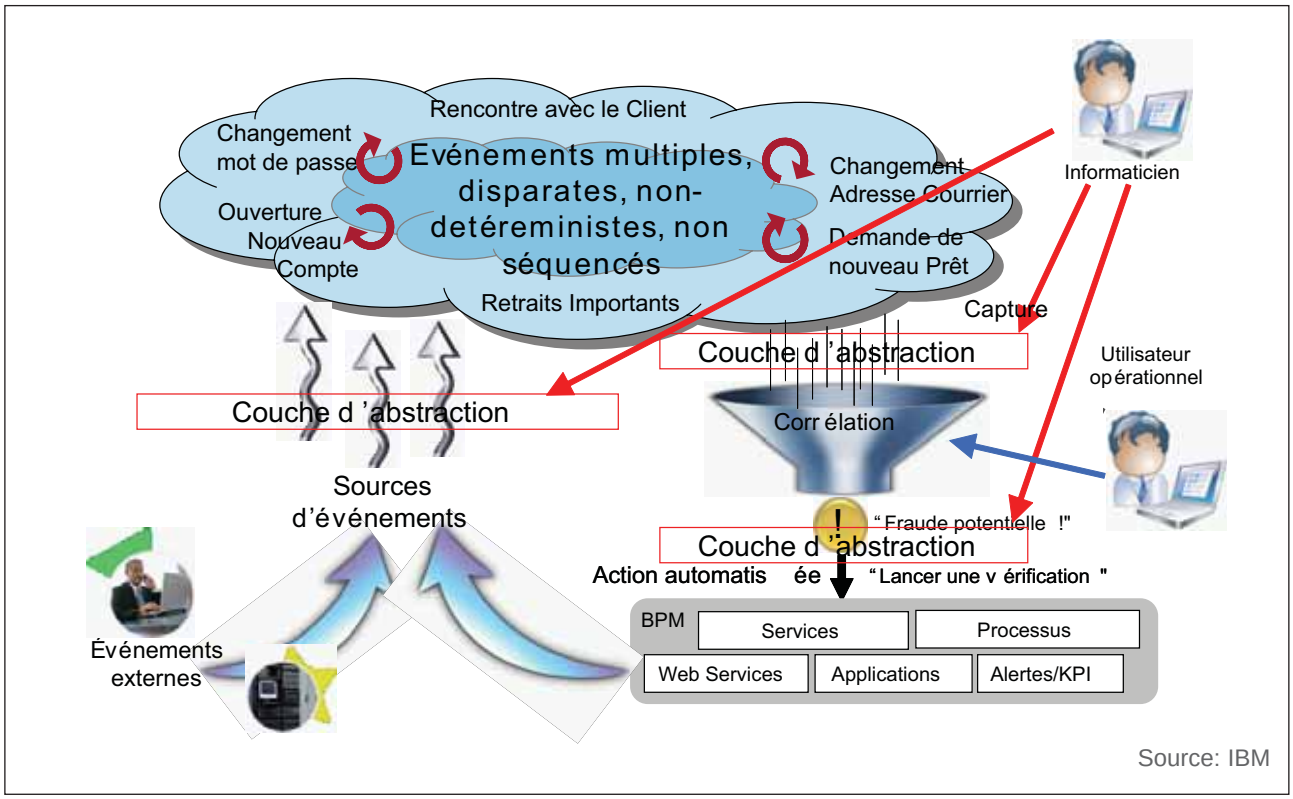
- ***manipuler*** des événements de sources opérationnelles multiples (le site de vente en ligne, la comptabilité, l'opérateur logistique ABC, ….) ;
- ***traiter*** ces événements ;
- ***identifier*** des combinaisons caractéristiques d'événements et d'information véhiculées par les événements, caractérisant des situations opérationnelles ;
- ***mettre en œuvre des mécanismes de détection*** de ces situations opérationnelles ;
- ***générer des événements dérivés*** par agrégation, enrichissement etc. et ou « programmer » les actions de traitement de la situation détectée ;
- ***superviser*** l'ensemble des événements - détection et traitement - action, encore une fois d'un point de vue opérationnel.

Comment mettre en œuvre une architecture de Business Event Processing ?

Il serait illusoire de penser que les utilisateurs opérationnels peuvent être à même d'appréhender toute la complexité de la chaîne de traitement d'événements. Beaucoup d'éléments de cette chaîne restent liés à des paradigmes informatiques et nécessitent le savoir-faire, voire l'expertise des informaticiens.

Le principe de base serait donc le cloisonnement des problèmes (***separation of concerns***) :

- Aux informaticiens, incombe la tâche de gérer la ***partie déterministe*** de l'édifice, c'est-à-dire prendre en compte la complexité technique d'accès aux sources d'événements et d'interaction avec les consommateurs d'événements. Leur revient, également et naturellement, la responsabilité d'assurer le niveau de service adéquat pour l'ensemble de la chaîne.
- Aux opérationnels, revient la tâche de gérer la ***partie non-déterministe*** de l'édifice et de faire



Source: IBM

évoluer le système par auto apprentissage, à partir des éléments mis à disposition par les informaticiens.

- Entre les deux, il est important de mettre en place une couche d'***abstraction*** ou de ***virtualisation*** des concepts (événements, sources d'événements, consommateurs d'événements, modèles d'information, actions ….) qui les rendent intelligibles aux deux populations d'une part, et entre les différents métiers opérationnels d'autre part.

- Enfin il faut pouvoir gérer le cycle de vie de ces éléments d'abstraction.

La figure ci-dessous résume l'organisation globale de l'architecture de Business Event Processing

Plusieurs remarques sont à faire sur ce modèle :

L'approche d'abstraction des événements est très similaire à ce que la SOA a introduit au niveau de l'abstraction des services. En étant schématique, l'évolution majeure dans l'intégration entre applications, à partir d'approches plus anciennes comme l'Enterprise Application Integration (EAI) vers le SOA, a consisté à mettre en œuvre cette couche d'abstraction (« les services ») entre opérationnels et informaticiens.

On peut poursuivre le parallèle en disant que le Busines Event Processing se positionne en regard du ***Business Process Management (BPM)*** ou gestion des processus opérationnels. Le BPM fournit aux opérationnels (professionnels, managers ou analystes) les moyens de gérer, de piloter et de faire évoluer directement par l'automatisation, certains processus opérationnels.

On pourrait avoir les niveaux de correspondance suivants :

- EAI et CEP

- Services et événements

- SOA et Event Processing (ou EDA)

- BPM et BEP

Les relations entre Business Event Processing et Service Oriented Architecture sont développées plus loin dans ce document, au paragraphe « ***Lien entre Business Event Processing et SOA*** ».

Cela induit aussi de nouveaux moyens, de nouvelles techniques, de nouvelles méthodes et de nouveaux standards.

Tout d'abord, les moteurs de CEP doivent être à même de traiter de gros volumes d'événements similaires, mais aussi de traiter des combinaisons sophistiquées d'événements disparates et ce, à la demande. Ce point est loin d'être anodin : que se passerait-il si un opérationnel (qui, à priori, ne comprend rien aux contraintes opérationnelles informatiques) imaginait un scénario tarabiscoté de corrélation d'événements capable de dégrader la performance d'ensemble de l'infrastructure de CEP (garantie par l'informaticien) ? Comment faire donc pour isoler les flux d'événements, distribuer la charge, mettre en place des classes de services, valider les règles de corrélation avant la mise en production.

En terme d'outils, il faut non seulement pouvoir proposer un outillage adapté aux opérationnels mais aussi gérer en commun les niveaux d'abstraction du BEP. Le SOA a positionné le rôle clef de l'annuaire ou plus exactement du ***référentiel de Services*** dans la gestion commune du cycle de vie des services. Un composant similaire devient nécessaire dans le monde du BEP. Il faut donc pouvoir construire et faire vivre cette architecture autour d'un ***référentiel d'Evénements***.

De même, dans le modèle BEP, un événement est un événement opérationnel, pas un événement technique. Un outillage méthodologique d'identification et de définition d'événements, comparable à ce que le SOA a mis en place pour les services, est à imaginer.

Reste le problème des standards qui est à prendre à 2 niveaux : au niveau de la technique, pour faciliter l'interopérabilité « événementielle » des différents composants techniques de l'architecture, et au niveau opérationnel, pour pouvoir standardiser des événements entre constituants de l'Entreprise.

Les fonctions du traitement des événements

Nous allons aborder dans ce paragraphe les moyens élémentaires de traitement qui peuvent être utilisés et combinés par les opérationnels pour corréler les événements. La liste ci-dessous n'est pas exhaustive, mais résume les plus significatifs :

Business Event Processing ou le traitement des événements opérationnels

- Le filtrage
- Le routage
- La transformation
- L'enrichissement
- La corrélation

Filtrage d'événements

Un Filtre est une fonction sans état (stateless) qui filtre les événements en fonction de leur contenu, c'est-à-dire l'information véhiculée par le message généré par l'occurrence de l'événement.

Exemple : filtrer les requêtes faites par nos clients « platinum »

Routage d'événements

Les formes de ***routage*** d'événements sont multiples, comme par exemple

Routage basé sur un itinéraire pré-établi

Exemple : envoi au spécialiste de la conformité légale

Routage basé sur les souscriptions

Exemple : envoi à toutes les applications abonnées

Routage « intelligent »

Exemple : parcourir un arbre de décision pour déterminer l'adresse du ou des destinataires

Transformation d'événements

Les transformations peuvent être multiples. On peut citer :

Traduction, comme par exemple : utiliser un convertisseur de monnaie pour changer tous les prix en €

Agrégation, comme par exemple : analyser la profondeur de la file d'attente et générer un événement lié au nombre moyen ou maximum d'événements en attente

Eclatement, comme par exemple: envoyer une notification suite à l'envoi d'une facture au CRM et à la compta avec des contenus partiellement communs

Enrichissement

Le but est d'enrichir le contenu de l'événement avec des données provenant de base de données, de tableurs, d'e-mails, de fichiers etc..

Exemple : ajouter à une commande le type de client et la limite de crédit autorisé, lus dans un référentiel Client

Corrélation d'événements

Il s'agit de détecter des combinaisons significatives d'événements multiples

Exemple : 3 dépôts de plus de 5000 € sur le même compte en moins de 10 jours

La règle de corrélation produit en sortie est un ***événement dérivé***, représentant la situation caractérisée par la combinaison d'événements

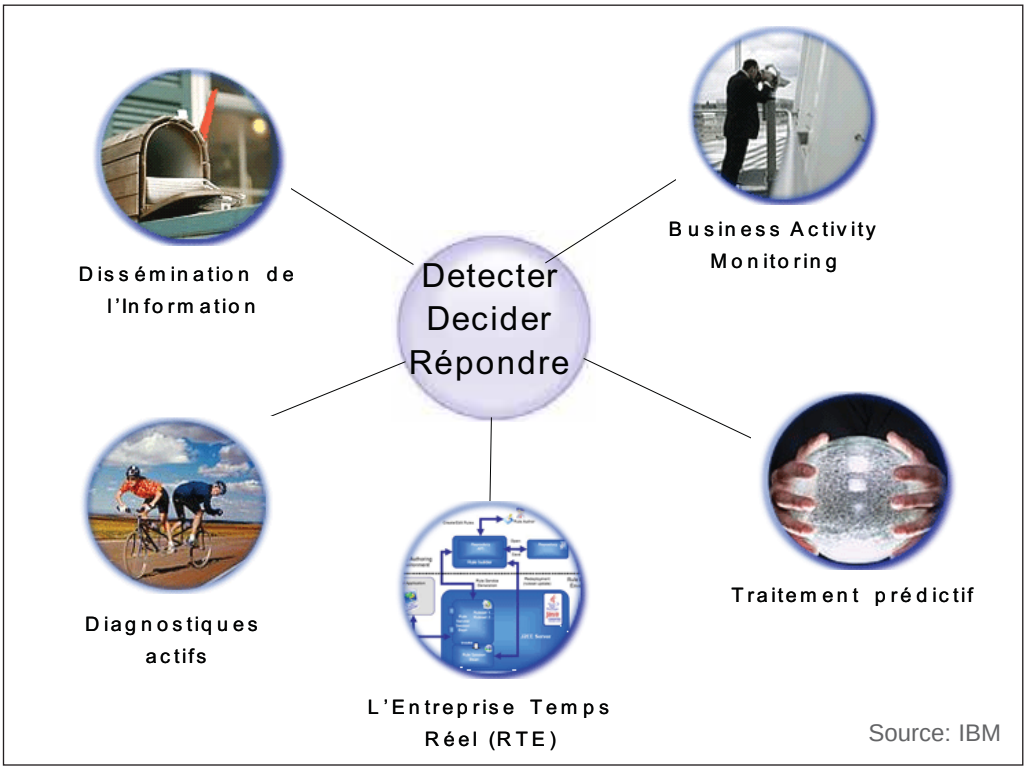
Domaines d'utilisation de l'Event Processing

Le Software Group d'IBM a identifié 5 domaines majeurs d'utilisation de l'Event Processing. Chacun de ces domaines correspond à un segment d'utilisation, qui peut être plus adapté à un secteur d'activité qu'à un autre, ou une activité plutôt qu'à une autre à l'intérieur de l'Entreprise. Chaque domaine a donc ses spécificités. Il peut y avoir également des éditeurs spécialisés dans tel ou tel de ces domaines.

Ces 5 domaines sont résumés sur la figure suivante :

Les caractéristiques propres à chaque domaine sont les suivantes :

- ***Supervision de l'Activité Opérationnelle*** ou **Business Activity Monitoring (BAM)** : Cela concerne l'observation en temps réel de situations opérationnelles exceptionnelles et la notification des intéressés. On parle également d'EPBO ou ***Event Processing Business Observation)***. En termes de fonctionnalités, ce domaine de solution doit pouvoir implémenter,



outre les mécanismes de base de détection et de réponse, le support des métriques opérationnelles (seuil, KPI,….) de réseaux de dépendances (Dependency Nets).

- ***Traitement prédictif ou Event Processing Predictive Processing (EPPS)*** : Il s'agit d'atténuer ou d'éliminer l'effet de certains problèmes en les anticipant (exemple rupture de stock, conséquences du retard sur un vol …..). En termes de fonctionnalités, ce domaine de solution doit pouvoir implémenter des systèmes de prédiction et de dérivation prédictive intégrant des capacités d'analyse des événements, ainsi que la capacité de gérer des actions anticipées.

- ***L'Entreprise Temps Réel ou Real-time Enterprise (RTE) ou Event Processing Business Logic (EPBL)***; le cœur du fonctionnement du Système d'Information et la gestion des transactions opérationnelles sont complètement basés sur la réaction à des événements. En termes de fonctionnalités, ce domaine de solution doit pouvoir implémenter des capacités transactionnelles et de haute disponibilité.

- ***Diagnostiques actifs ou Active Diagnostics*** (AP) ou ***Event Processing Problem Determination (EPPD)*** : Il s'agit de diagnostiquer les problèmes à partir de l'analyse automatisée ou partiellement automatisée de leurs symptômes et d'automatiser leur résolution. En termes de fonctionnalités, ce domaine de solution doit pouvoir implémenter des mécanismes d'analyse de cause première (***Root Cause Analysis***) ou d'analyse d'impact (***Impact Analysis***)

- ***Dissémination de l'Information*** ou ***Event Processing Information Dissemination (EPID ou Situation Awareness)*** : cela consiste à fournir la bonne information, avec la bonne précision, dans le bon format, par le bon canal, à la bonne personne, au bon moment. En termes de fonctionnalités, ce domaine de solution doit pouvoir implémenter de gestion de la distribution.

Un sixième domaine est également à l'étude. Il concerne la ***collaboration entre personnes à base d'événements (Event Processing Human Collaboration, ou EPHC)***

Business Event Processing ou le traitement des événements opérationnels

Chaque segment apporte une valeur ajoutée particulière mais il est probable que, dans une organisation d'une certaine importance, plusieurs de ces domaines de solution sont ou seront présents. D'autre part certaines des fonctionnalités décrites ci-dessus ne sont pas particulières au monde de l'Event Processing. On peut penser, par exemple à la gestion des Key Performance Indicators (KPI) que l'on retrouve également au niveau du BPM ou de tout système décisionnel.

Ce qui signifie que si on veut optimiser les approches de conception et minimiser et mutualiser les couches techniques à mettre en œuvre et à gérer.

On s'achemine donc vers une combinaison de solutions à base d'événements et de services. C'est ce qui va être développé dans le chapitre suivant.

Lien entre Business Event Processing et SOA

Les similitudes entre l'approche BEP et le SOA ont été traitées ci-dessus dans le chapitre précédent. La question qui se pose maintenant est de savoir si Event Processing et SOA sont deux mondes parallèles ou complémentaires ou plus exactement si le BEP n'est pas le « chaînon manquant » dans la vision SOA.

Les deux modèles sont complémentaires

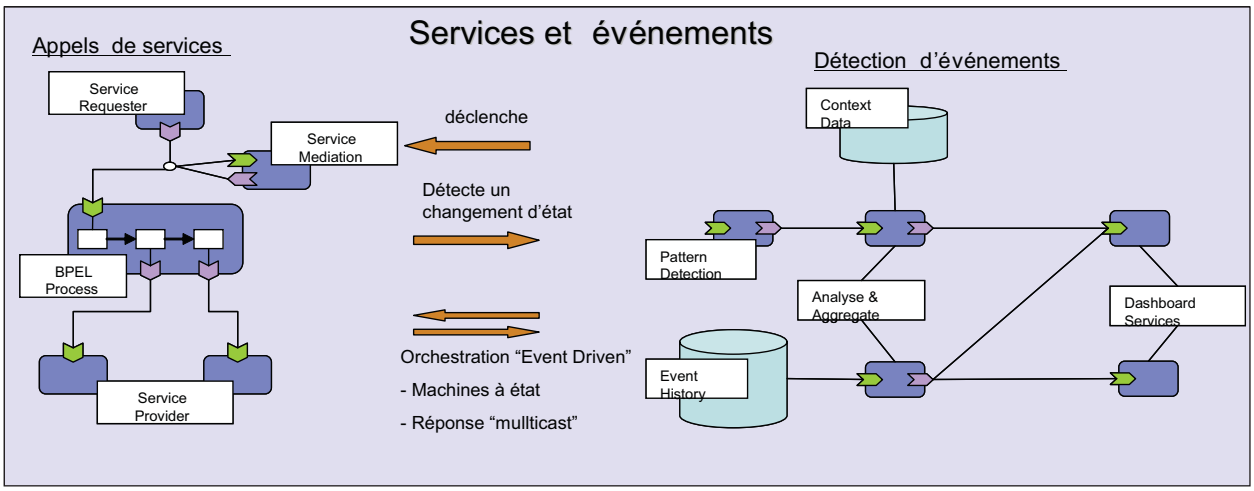
Si on se réfère au principe général de l'Event Processing décrit plus haut, la chaîne de traitement d'un événement est la suivante :

- Détection d'un ou plusieurs **événements,**
- Mise en évidence d'une **alerte** potentielle,
- Déclenchement d'un **module de traitement** de cette alerte
- Ce **module de traitement** va alors déclencher une **action** ou suite **d'actions** correctives ou réactives,

En fonction du niveau de réutilisabilité des services considérés, il serait intéressant que ces modules de traitement d'alerte utilisent des services identifiés pour déclencher les actions envisagées.

Si on va plus loin, quand on doit chaîner l'appel à plusieurs de ces services pour réagir à une alerte, il en résulte la mise en place d'un véritable processus opérationnel (business process) au sens BPM du terme, pour gérer l'alerte.

A l'inverse, un événement opérationnel matérialise un changement d'état opérationnel dans le fonctionnement de l'entreprise, pouvant nécessiter un traitement spécifique. Les exemples sont multiples : le prix d'un produit a été changé, un vol a été retardé, un client a résilié son abonnement. Dans de nombreux cas, l'émission de cet événement est liée à



Source: IBM

l'exécution d'un service. Par exemple, l'événement « un client a résilié son abonnement » à été généré par l'exécution du service « résiliation d'un abonnement », par le client lui-même au travers du portail de services en ligne de l'Entreprise.

Un service ou par extension un processus opérationnel peut donc générer des événements.

Les approches d'Event Processing et de SOA sont donc très complémentaires.

La figure ci-dessous liste un certain nombre d'interactions entre les deux mondes :

Les avantages de cette complémentarité sont les suivants :

- L'approche top-down SOA permet l'identification des domaines où le traitement des événements peut s'avérer utile
- Dans l'infrastructure SOA, l'Enterprise Service Bus peut être le support des traitements spécifiques aux événements
- Le SOA génère des événements dérivés, plus explicites et mieux alignés par rapport au contexte opérationnel
- L'Event Processing fournit de nouveaux déclencheurs aux business processes et rend ainsi le SOA plus réactif et plus dynamique
- Le Business Event Processing permet au SOA d'être plus en phase avec les évolutions du Système d'Information et/ou de l'organisation de l'Entreprise

Mais il y a des différences à ne pas oublier …

Toutefois, cela ne doit pas conduire à fusionner complètement les deux modèles, ou à tenter de gommer leurs spécificités, dans la façon dont on va mettre en œuvre cette approche de SOA avancé. Même si dans les deux cas, il va s'agir de mettre en place des éléments de logiciel présentant des interfaces formalisées (les « abstractions » décrites plus haut), il faut garder à l'esprit quelques différences fondamentales :

- Le modèle SOA est un modèle ***à couplage lâche (loosely coupling)*** entre consommateur et producteur de services. Le modèle Event Processing est un modèle ***sans couplage*** entre producteur et consommateur de l'événement, l'événement perçu par le consommateur ne pouvant avoir que peu de rapport avec l'événement émis, en dehors d'un lien de causalité.
- Le modèle SOA reste principalement un modèle « ***pull*** », c'est-à-dire ***tiré*** par ***un consommateur du service***, et se matérialise par des interactions de type requête/réponse. Il y a une unité d'œuvre, au sens opérationnel du terme, avec un début et une fin. Par exemple, un client d'une banque souscrit un prêt au travers d'un site de banque en ligne.
- Le modèle Event Processing est un modèle « push », c'est-à-dire poussé par plusieurs générateurs d'événements. Il n'y a pas interaction, mais émission en cascade d'une succession d'événements pouvant déclencher au passage des appels de services. Il n'y a ni début ni fin et une succession d'unités d'œuvres indépendantes. Exemple, un avion est en retard, donc certains passagers vont rater leur correspondance, ça va poser des problèmes au niveau des bagages enregistrés…. Pour d'autres il va falloir faire attendre les vols. Les places de parking à l'aéroport vont être occupées plus longtemps que prévu, donc peut-être occasionner d'autres retards sur des vols qui devaient occuper ces places à leur arrivée et ainsi de suite
- Comme dit plus haut, un processus opérationnel (business process) est mis en œuvre dans le modèle SOA, d'une manière qui reste déterministe. C'est moins vrai quand on combine l'approche événementielle dans la mise en œuvre de ce processus global.
- Enfin, reste à prendre en compte le niveau de réutilisation d'un même service, selon qu'il est utilisé de manière requête/réponse ou événementielle.

Il faut donc enrichir le modèle SOA mais bien distinguer la gestion des événements, la gestion des services et les interactions entre flux et services.

Business Event Processing ou le traitement des événements opérationnels

En résumé, une architecture de Business Event Processing va s'appuyer sur une infrastructure SOA complétée et enrichie avec des moyens propres au traitement des événements. De même les approches méthodologiques du SOA, devront être étendues pour prendre en compte la dimension événementielle.

La stratégie et l'offre d'IBM en matière d'Event Processing

Les principes généraux

La stratégie d'IBM est clairement de fournir les moyens du Business Event Processing.

Ces moyens doivent permettre la mise en œuvre et le support d'infrastructures couvrant les 5 domaines listés plus haut (voir le paragraphe « ***Domaines d'utilisation de l'Event Processing*** » ainsi que des combinaisons.

Ces moyens doivent être un modèle d'architecture intégrée, un même modèle de programmation et un jeu de composants permettant une réutilisation maximum.

Enfin, ces moyens viennent enrichir le modèle d'architecture SOA d'IBM

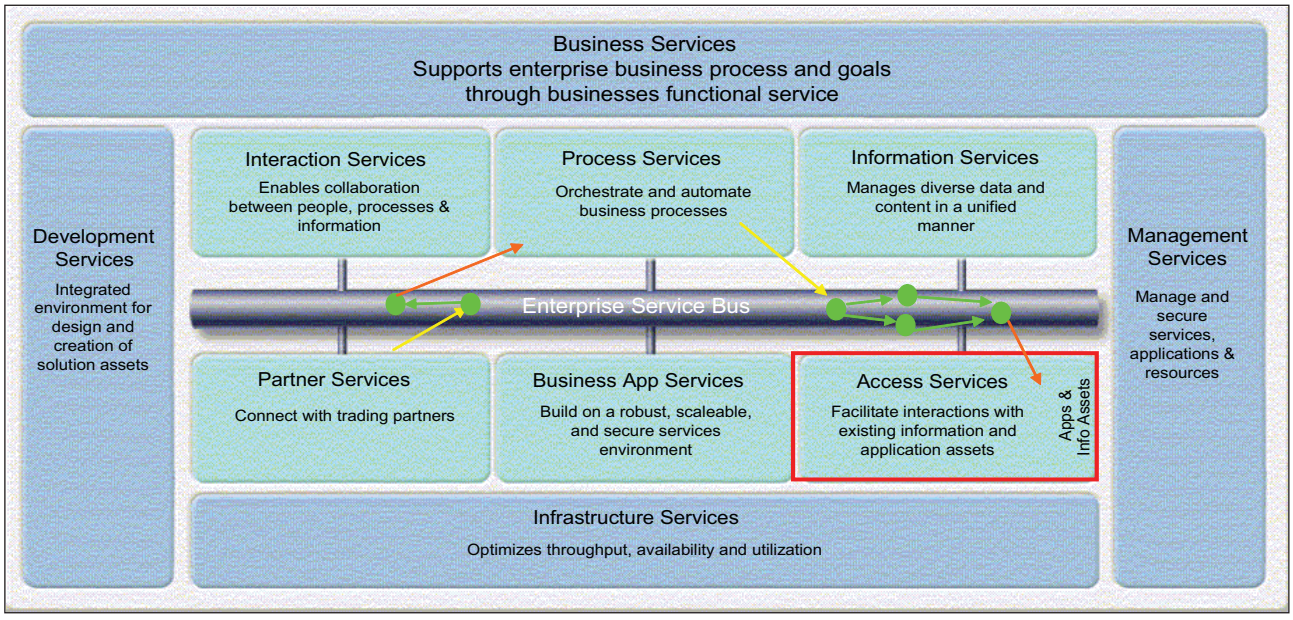
Event Processing et modèle d'architecture SOA d'IBM

La figure ci-dessus positionne le Business Event Processing dans le modèle d'architecture SOA d'IBM

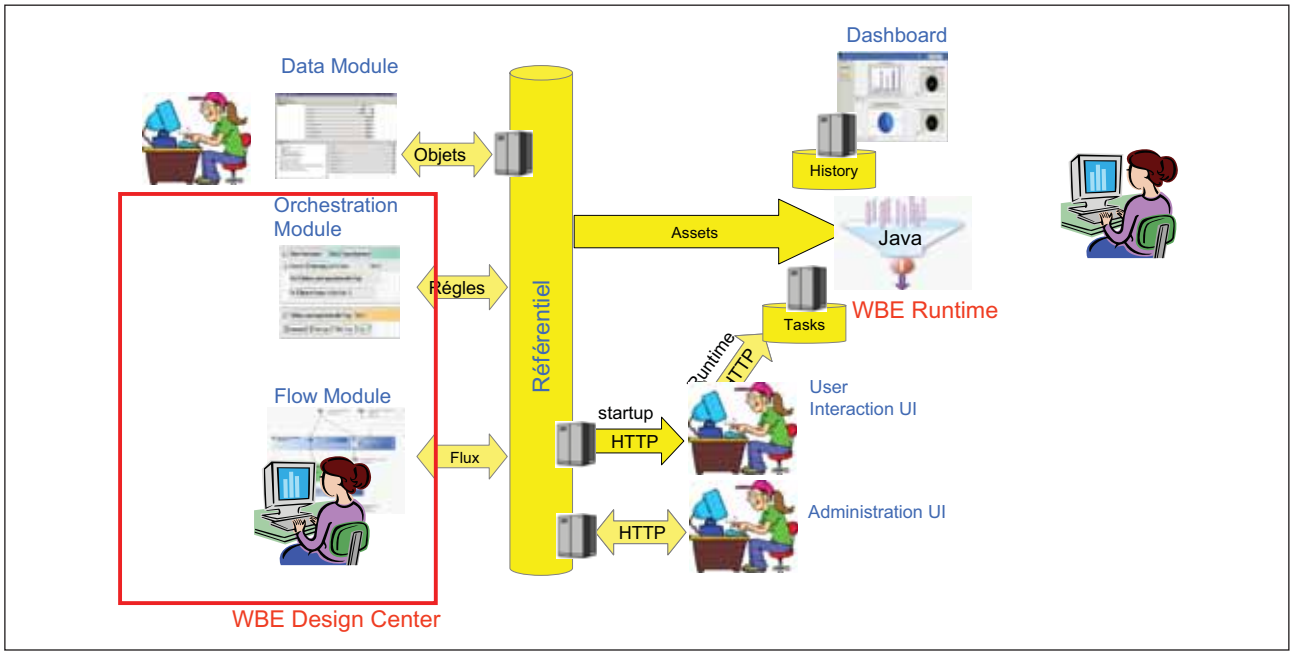
- Production d'événements (flèches jaunes) : toute application ou système ou source de données ou processus, interne ou externe, peut produire des événements, directement au travers des connecteurs. Les événements sont produits de manière continue, périodique ou à la demande.

- Traitement des événements (flèches vertes) : l'Enterprise Service Bus (ESB) a un triple rôle. Il achemine les flux d'événements, met en œuvre les moteurs de traitement d'événements (CEP décrit plus haut), et distribue les événements induits et les alertes, et déclenche les services supportant les actions.

- Consommation d'événements (flèches rouges) : les consommateurs d'événements peuvent être des applications ou des systèmes ou des sources de données ou des processus, interne ou externe. Ils reçoivent notification des événements auxquels ils sont abonnés via l'ESB. Ils déclenchent ou orchestrent les actions nécessaires au travers de services.



Source: IBM



Source: IBM

L'outillage nécessaire à la mise en œuvre du BEP est pris en compte au niveau des Development et Management Services (pour les informaticiens) et Business Services (pour les analystes opérationnels).

Une première étape WebSphere Business Events

Le produit WebSphere Business Events (WBE) annoncé en avril 2008 et disponible en juin 2008 constitue la première annonce majeure de l'offre Business Event Processing d'IBM.

Organisation générale du produit

La figure ci-dessous résume la structure générale du produit.

WBE s'articule autour d'un référentiel regroupant les objets manipulés par le produit (événements, sources d'événements, actions, règles de traitement des événements, flux d'événements, indicateurs …).

Ce référentiel est partagé entre :

- L'environnement d'exécution ou **WBE runtime** en charge de traiter les flux d'événements sur base J2EE
- Les outils proposés aux informaticiens **(WBE Data Manager)** et aux analystes opérationnels **(WBE Design Center)**
- D'une interface Web d'administration du produit
- D'interfaces Web à l'intention des utilisateurs (notifications, tableaux de bord ….)

La mise en place d'une solution de traitement d'événements se fait en suivant les étapes suivantes

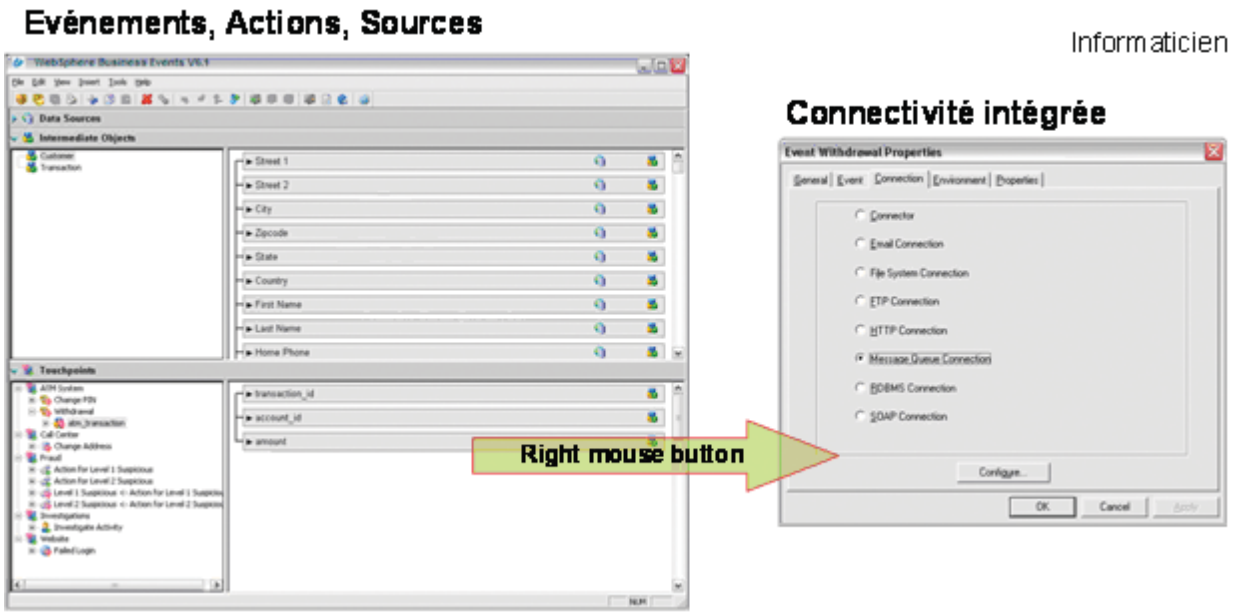
- Identification des artefacts base (building blocks)
- Création des règles de traitement des événements, mise en place de flux d'événements
- Lien entre événements induits et actions
- Déploiement du flux d'événements sur le runtime
- Exécution et supervision

Identification des artefacts de base

La figure ci-dessous donne un aperçu de l'interface proposée à l'informaticien.

Cette étape correspond à l'étape d'***abstraction*** du modèle BEP décrit plus haut. Elle revient à

Business Event Processing ou le traitement des événements opérationnels



Source: IBM

l'informaticien à l'aide de l'outil de Data Management et va consister à mettre en place toute la connectivité avec les sources de données et à mettre à disposition dans le référentiel partagé les composants de base que sont les sources des événements, les modèles de données véhiculées par ces événements ainsi que les actions, ainsi que les modèles de données liées à ces actions.

Création des flux d'événements

Cette étape revient à l'analyste opérationnel. Elle correspond à la mise en place des modules de traitement des événements du modèle BEP décrit plus haut.

L'outil Design Center lui fournit un outil puissant et convivial (voir figures ci-dessous) pour lui permettre de :

- Définir les règles élémentaires de traitement des événements
- Lier les événements aux actions
- Lier ces règles dans des flux d'événements qui vont corréler ces événements

Tout ce travail se fait à partir du référentiel déjà alimenté par l'informaticien. L'analyste n'a à avoir aucune connaissance de la nature technique des sources d'événements, de la manière dont sont récupérés ces événements, ou de l'artifice technique de déclenchement des actions.

Cette étape peut être menée indépendamment de la précédente si les artefacts considérés sont déjà présents dans le référentiel.

Le déploiement se fait à chaud sur le runtime, par publication dans le référentiel des flux d'événements.

Intérêt du produit et étapes suivantes

Par rapport au modèle précédemment décrit, WebSphere Business Events apporte les bases d'une approche de Business Event Processing, en couvrant les points suivants :

- Pour les informaticiens, il fournit les outils pour mettre en œuvre les couches d'***abstraction*** vis-à-vis des événements, des sources d'événements et des actions.
- Il propose aux ***opérationnels***, ou tout au moins aux ***analystes*** un outillage puissant. Cet outillage leur permet de créer et de faire évoluer eux-mêmes les ***modules de traitement des***

événements, pour détecter de l'occurrence de situations opérationnelles, de coordonner la réponse adaptée à ces situations et d'évaluer le résultat de ces actions

- C'est un environnement générique pouvant adresser n'importe quel type d'événements opérationnels, qui peut être utilisé pour couvrir des besoins transverses à l'organisation et donc servir de base à la mise en place du BEP au niveau de l'Entreprise.

- Il s'appuie sur un environnement d'exécution puissant (CEP), tirant parti des technologies et des infrastructures J2EE, et assurant ainsi la performance et la capacité à monter en charge. Il permet en outre le déploiement incrémental et le déploiement à chaud des solutions de BEP. Tout cela conduit à une optimisation du coût global opérationnel.

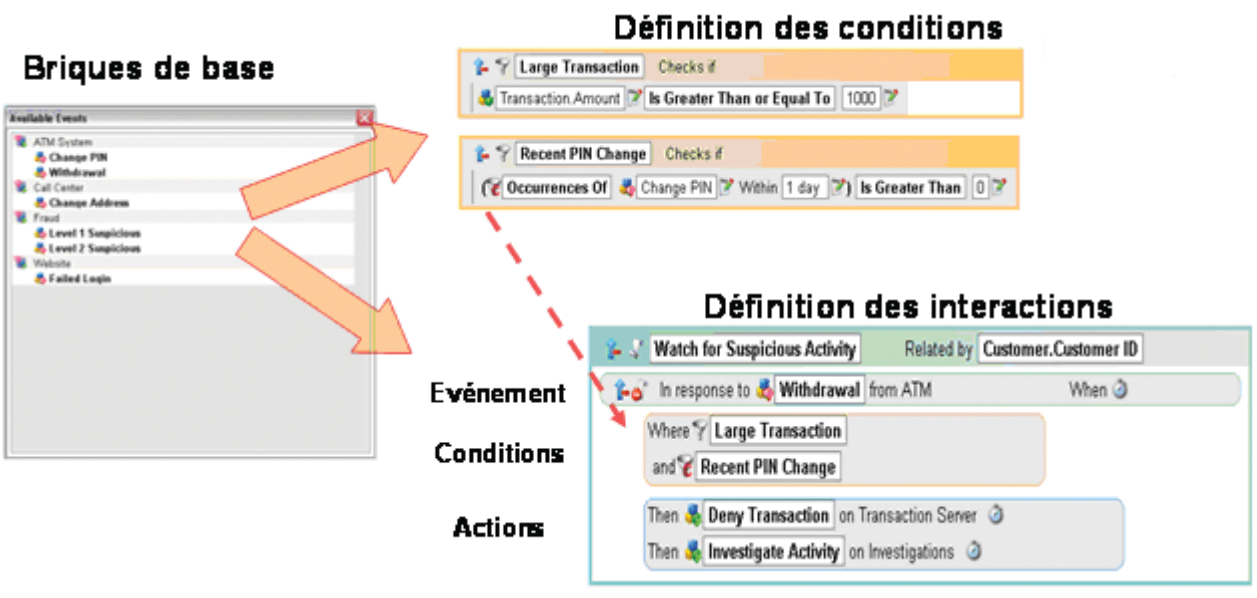
Au delà de cette première étape qui pose les bases du BEP, l'offre IBM va évoluer pour adresser les points évoqués plus haut concernant :

- L'amélioration constante de la capacité de l'environnement d'exécution de pouvoir supporter tout type de solutions de BEP, tout en garantissant à chacune d'entre elles le niveau de qualité de service adéquat, en tirant parti des toutes dernières évolutions du serveur J2EE WebSphere.

- Une plus grande intégration avec les autres éléments du modèle SOA, notamment en ce qui concerne le référentiel d'artefacts, les moyens de supervision et de monitoring, l'outillage pour les opérationnels, ou l'utilisation de l'ESB pour étendre les capacités de connectique.

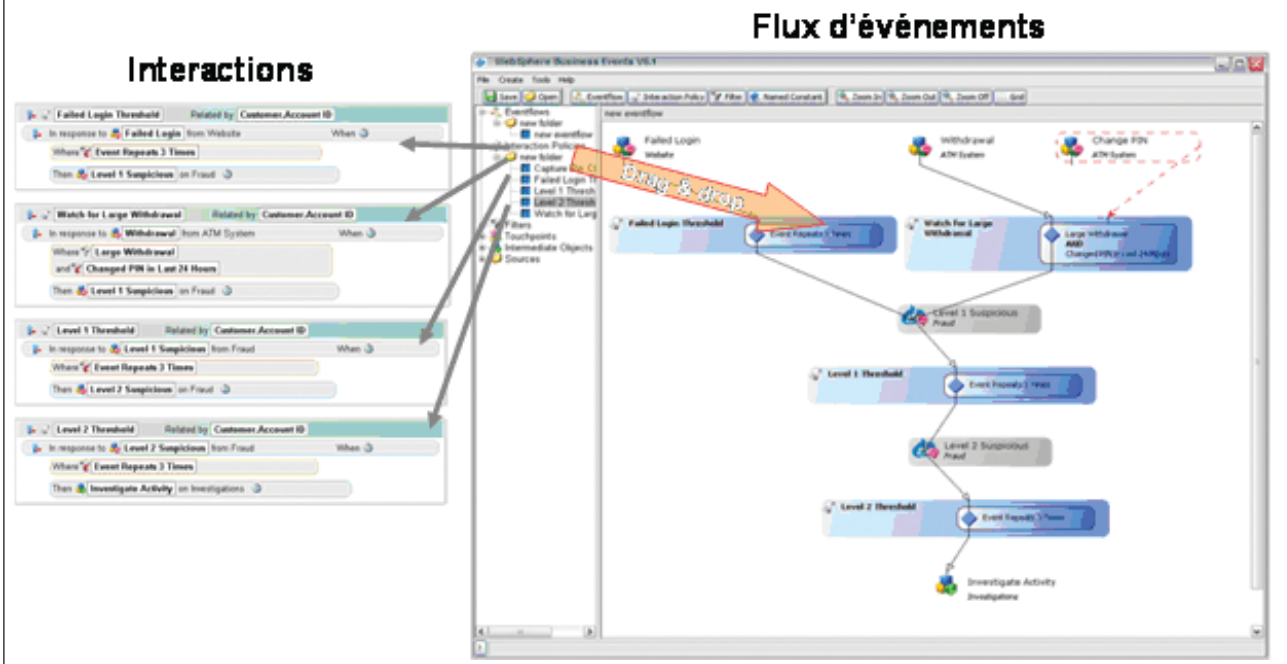
- L'utilisation des standards émergeants en matière de traitement d'événements comme par exemple WS-Notification.

- La corrélation entre les événements opérationnels et les événements purement informatiques, tels que ceux géré par Tivoli Monitoring ou autres logiciels de gestion de système.



Source: IBM

Business Event Processing ou le traitement des événements opérationnels



Source: IBM

Au-delà, on parle d'***Intelligent Event Processing***, ou la combinaison de techniques de BEP et de Business Intelligence permettra de fournir aux opérationnels des moyens encore plus avancés de détection et de traitement des situations opérationnelles.

Conclusion

En complément du SOA, le Business Event Processing doit permettre une meilleure adaptation des moyens informatiques à l'évolution des besoins tactiques ou stratégiques de l'Entreprise (« ***reduce the gap between business and IT*** »).

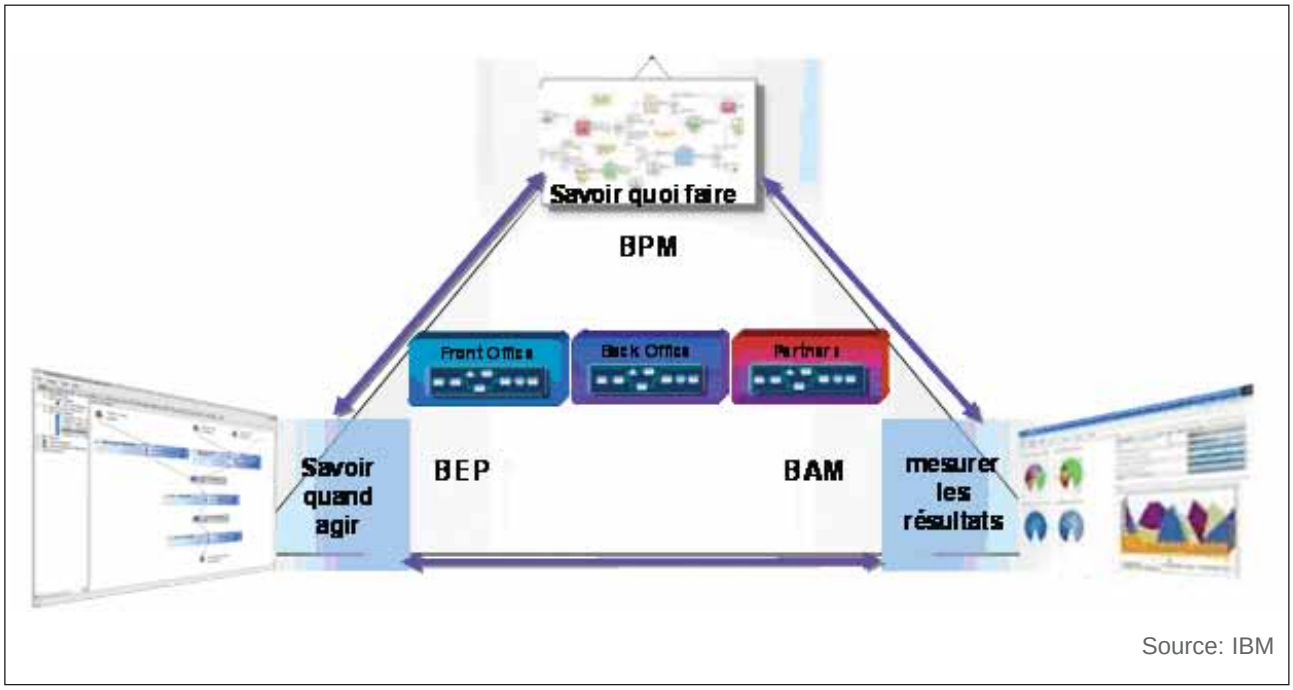
Business Event Processing, Business Process Management et Business Activity Monitoring sont les 3 piliers de la performance opérationnelle de l'Entreprise.

Le schéma ci-dessous peut être une conclusion à ce document.

Le SOA vise déjà à accroître la contribution directe des opérationnels (professionnels, managers et analystes) dans la mise en œuvre et le contrôle de ces moyens.

Le **Business Process Management (BPM)** doit leur permettre de modéliser certains processus de l'entreprise, d'une part afin de mieux les comprendre, de les automatiser dans la mesure du possible et de les optimiser. C'est le « Savoir quoi faire » du dessin ci-dessus.

Le **Business Activity Monitoring (BAM)** vient compléter le BPM en facilitant la supervision et le pilotage des activités opérationnelles de l'Entreprise au travers des éléments qui sont automatisés par l'Informatique. C'est le « **Mesurer les résultats** » du dessin ci-dessus.



Source: IBM

Le Business Event Processing, déjà défini précédemment fournit la troisième dimension de l'édifice, en leur fournissant les moyens de détecter et de traiter des situations opérationnelles critiques pour l'Entreprise. C'est le «**Savoir quand agir** » du dessin ci-dessus.

Par exemple, un utilisateur opérationnel va détecter une anomalie de fonctionnement au travers d'un outil de BAM parce qu'un indicateur a dépassé un seuil critique, et ou parce qu'un certain nombre de messages d'alerte lui ont été remontés. Avec un analyste, il va déterminer la conjonction d'événements conduisant à cette anomalie, et la suite d'actions correctrices à mettre en place pour y remédier. Il se servira alors du BEP, pour mettre en place le module de traitement des événements, et du BPM pour la mise en place de la logique d'orchestration des actions correctives, ainsi que des éléments d'observation des résultats de cette correction.

Source: IBM

Tutorial for EDA and How It Relates to SOA

This research unit defines event-driven architecture (EDA) and explains how it relates to service-oriented architecture (SOA). Architects, developers and business analysts who are designing EDA applications must understand where and how to use SOA principles with EDA.

Key Findings

- SOA and EDA are compatible and complementary concepts. They can be used in the same system at the same time because they refer to different aspects of design.
- SOA is an architectural style in which systems are modular and their components are distributable, have defined interfaces and are loosely coupled and shareable.
- EDA is an architectural style in which one or more components of a system execute in response to receiving one or more event notifications.
- Not all event processing is EDA. Event objects also may be processed in batch, query or update modes.

Recommendations

- Most new applications that use EDA with business events should be implemented using the five principles of SOA (modularity, distributability, defined interface, separation of interface from implementation and shareability). Such applications employ SOA and EDA simultaneously in event-driven SOA interfaces.
- When EDA is used in fine-grained or local (nondistributed) systems, SOA usually is not relevant. Such systems use EDA but not SOA.
- EDA applications must implement these three principles. First, notifications are pushed by the event source, not pulled by the event consumer. Second, the arrival of a notification causes the event consumer to act immediately. Third, a notification does not specify what action the event consumer will perform.

WHAT YOU NEED TO KNOW

This document is an updated version of the document published on 14 February 2005.

The principles of service-oriented architecture (SOA) are useful in most distributed applications to clarify the application structure, facilitate data and code sharing, and enable incremental maintenance and enhancements (see "The Business Impact of Service-Oriented Architecture"). Most large SOA systems should use event-driven architecture (EDA) for some asynchronous aspects of their processing.

ANALYSIS

Context

Most IT professionals understand SOA's basic characteristics, but many are less familiar with the nature of EDA. It is relatively straightforward to position EDA with respect to SOA. This research starts by defining the two terms, then explores where the concepts are complementary and where they are independent of each other.

SOA Defined

SOA is an architectural style in which systems are modular and their components are distributable, have defined interfaces and are loosely coupled and shareable. Any business application that adheres to these five principles is an SOA implementation:

1. Modular: The system has two or more components (usually dozens), including at least one component that acts as a service consumer and another that acts as a service provider.
2. Distributable: The components can run on disparate computers and can communicate with each other by sending messages over a network at runtime. SOA relies on program-to-program communication.
3. Defined interfaces: Component interfaces are documented using metadata that specifies an explicit contract between consumers and providers. This metadata describes the messages that are exchanged and other characteristics of the agreement among the components.

4. Loosely coupled: A provider component can be swapped out for another component that supplies the same service without changing or recompiling the consumer (or consumers), because the interface is separate from the service provider's implementation (the provider component's internal code and data).

5. Shareable: A service provider component can be used successively by disparate consumer components (sometimes called "reuse").

SOA is a set of best practices for distributed applications. It is a logical evolution of the concept of component software. Generally, an SOA interface should map directly to a business concept, which makes sense inside and outside the IT department. For example, "look up customer's credit history" is a reasonable SOA service because it is meaningful to business analysts and end-user businesspeople, and is of a scale (level of granularity) that is practical to encapsulate in a discrete software package (being neither too large nor too small to warrant its own packaging).

An SOA interface may use a client/server (C/S) interaction (generally, a request/reply message pair), an EDA event notification (a one-way message) or a conversational message exchange pattern (a sequence of any number of messages where state is maintained on both sides). We address this more closely in "SOA Applications Should Mix Client/Server, EDA and Conversational Patterns." Most large SOA applications use a mix of C/S and EDA communication patterns. The conversational pattern is used less often but sometimes is appropriate. This research addresses only the EDA style.

EDA Defined

EDA is centered on the concept of events. An event is anything that happens or is thought to happen. Examples of events vary widely and include bank transactions, stock trades, customer orders, address changes, shipment deliveries, birthdays, moon landings, the Great Depression and anything that occurs in a simulation or dream. Events may be instantaneous, or they may happen over time.

Data about an event can be recorded in an event object (an event object may be called just an "event," which overloads the term and sometimes leads to confusion). Event objects exist in many forms, such as an XML document containing a customer order, an e-mail confirmation of an airline reservation, a database row containing a new customer address, data from a radio frequency identification (RFID) sensor reading or a financial data feed message that reports an equity trade. An event object generally includes some attributes of the event, one or more time stamps and sometimes an identifier of the event source and related context data. If the event is instantaneous, then there may be one time stamp noting when it happened; otherwise, there may be separate time stamps for the beginning and end of the event. There may be yet another time stamp for when the event object was created or sent. When an event object is transmitted in the form of a message, the combination is called an "event notification" or "notification."

EDA is defined as an architectural style in which one or more components of a system execute in response to receiving one or more event notifications. The flow of work through the system is determined, in part, by transmitting notifications. EDA applications must implement these three principles:

1. Notifications are pushed by the event source, not pulled by the event consumer. In other words, the event source determines when the message that contains the event object is sent.

2. The arrival of a notification causes the event consumer to act immediately. However, in some cases, the action is merely to save the event object for subsequent processing. The consumer is waiting for a notification and is driven to do something by its arrival.

3. A notification does not specify what action the event consumer will perform; it is a report, not a request. The consumer contains the logic that determines how it will respond. In this respect, notifications are fundamentally different from procedure calls or method invocations, which explicitly specify the particular functions that the message recipient will perform.

Tutorial for EDA and How It Relates to SOA

Analysis

Event Processing Without EDA

Not all event processing is EDA. Some event processing uses passive event objects. For example, a business system could store thousands of event objects, such as sales records, in rows in a database table in the course of a day. Then:

- At night, a scheduled batch application could read through the database and do follow-up processing.
- The next day an ad hoc business intelligence (BI) query could be run against the database.
- An online application could retrieve a sales record, modify it in response to a new customer request and put it back in the database (many event-processing systems would not allow this because their event objects are unchangeable, but some event-processing systems allow changes to event objects).

These are event-processing applications because they consume (read) event objects and perform computations on them, but they are not EDA. The event objects they use are events "at rest" in the form of database rows rather than events "in motion" in the form of notifications (messages). Moreover, the timing of when the applications (the batch job, ad hoc query and update request) run is not determined by the arrival of a notification (an event object in a message).The batch job executes according to a timer event, and the BI query and the update transactions execute in response to human requests. All are event-driven, in a real-world sense, but they are not event-object-driven in the EDA sense. It is likely that the schema of the event objects in the database would be different from the schema of an event notification, although this does not affect the definitional issue of whether the application is EDA.

In casual use, people often use the term "EDA" as a synonym for event processing. In many cases, this loose wording doesn't cause problems. However, architects who are designing real systems must distinguish EDA from passive event processing.

Outside the IT realm, the term "event driven" describes behavior that occurs in response to unplanned or unpredictable external stimuli. For example, a department may be criticized for being event-driven rather than being guided by a systematic consistent plan of action. Or people may bemoan that their work day is event-driven, because they are unable to find time to carry out a long-term project. Such behaviors are event-driven but are not EDA. EDA is limited to software systems that use notification messages. This is analogous to the use of the terms "service oriented" and "SOA." Business organizations may be service-oriented but not SOA, because the term SOA is limited to software systems.

Key Facts

Using SOA With EDA

As a general principle, architectural styles often are composable. In other words, multiple architectural styles can apply simultaneously to different aspects of one system. For example, an application may use relational data architecture, object-oriented architecture and model-driven architecture all at once because these styles are not mutually exclusive.

This principle applies because SOA and EDA are complementary and composable. Most new applications that use EDA with business events should be implemented as SOA simply because SOA is helpful in most distributed software systems. The key qualifier is the term "business." A business event is an event that is relevant to the business: It is a meaningful change in something in the company or related to its activities. Examples of business events include bank transactions, stock trades, customer orders, address changes, shipment deliveries and hiring employees. Business events typically are of the right level of granularity to be implemented as SOA service interfaces. By contrast, technical events often are too fine-grained to be implemented using SOA (see the Using EDA Without SOA section below).

An EDA application is an SOA application when it implements all five principles of SOA listed in the definition of SOA:

1. Modular: An EDA system always has two or more components (usually dozens), including at least one component that acts as an event source and another that acts as an event consumer.

2. Distributable: EDA components often are designed to run on disparate computers and communicate with each other by sending notifications over a network at runtime. However, not all EDA systems are distributable; thus, not all EDA systems use SOA.

3. Defined interfaces: The schema of event objects and other aspects of the notification communication should be documented using interface metadata to facilitate application maintenance and expansion (for the same reason that C/S SOA applications use interface metadata). However, when EDA systems do not have formal interface metadata, they are not implementing the SOA style.

4. Loosely coupled: An event source, which is the provider component in an EDA relationship, can be swapped out for another event source implementation, because notifications are defined separately from the event source and from the component's internals.

5. Shareable: Event sources are usable by multiple, disparate event consumers in the sense that each emitted event notification can be delivered to multiple disparate event consumers (for example, using publish-and-subscribe middleware).

Using EDA Without SOA

Systems that use EDA for local (nondistributed) or fine-grained events may not get benefits by implementing the SOA style, so they should not use SOA. For example, EDA often is used with technical (nonbusiness) events for purposes such as dispatching work or responding to an input/output interrupt in an operating system. SOA is not appropriate for these purposes because the event source and consumer are designed in concert with each other, and formal metadata for the interface would be superfluous. Moreover, the event source and consumer in this and some other EDA systems may be optimized to take advantage of the fact that they are running in the same computer. Local event processing does not use SOA, because event source and consumer components are not distributable.

Consider the example of an RFID system that uses EDA for sending raw sensor readings to an RFID edge controller. The individual readings from the RFID tags are fine-grained events. They are not business events because they are not directly meaningful to a business person. This part of the system probably would not be implemented as SOA, because the sensor communication may not need formal metadata. However, once the edge controller has filtered multiple (often redundant) sensor readings, it will derive (compute) a higher level, abstracted event that indicates the presence of a particular package at a particular location and time. This derived event is called a "complex" event, but it simplifies the information because it condenses and summarizes the significance of multiple incoming raw events. The derived event is a business event because it means something to a businessperson. The notification that conveys the derived event is likely to be consumed by a business application (that is, a supply chain management system or business activity monitoring dashboard), and, in most cases, it should be treated using the principles of SOA.

Gartner RAS Core Research Note G00155163,
Roy W. Schulte,14 February 2008

Architectures orientées services :
http://www-306.ibm.com/software/fr/soa/?ca=homepage=w&met=websphere

BPM :
http://www-306.ibm.com/software/fr/soa/launch/bpmsoa.html

IBM Software TV :
http://www.softwaretv.com/web/guest/home

WebSphere®