Solutions pour environnements d'exploitation à la demande
Document de présentation technique

Migration vers une architecture orientée services

par Kishore Channabasavaiah et Kerrie Holley, IBM Global Services, et Edward M. Tuggle, Jr., IBM Software Group

Avril 2004

Table des matières

2 Introduction : arguments en faveur du développement d'une architecture de services
3 Problème n°1 : la complexité
4 Problème n° 2 : une programmation redondante et ne pouvant être réutilisée
4 Problème n° 3 : des interfaces multiples
5 Et l'avenir ?
7 Exigences concernant une architecture orientée services
8 Une architecture orientée services : au-delà des services Web
10 Nature d'un service
12 Réponses aux problèmes anciens
14 Exigences d'intégration au sein de l'architecture
17 Avantages du déploiement d'une architecture orientée services
19 L'avenir : de nouveaux modèles, de nouvelles exigences
21 Résumé
21 Pour plus d'informations

Introduction : arguments en faveur du développement d'une architecture orientée services

Au cours des quarante dernières années, les systèmes informatiques se sont développés de manière exponentielle, sous forme d'architectures logicielles de plus en plus complexes et difficiles à gérer pour les entreprises. Les architectures traditionnelles ont atteint leurs limites en termes de capacité, alors que les besoins traditionnels des organisations informatiques demeurent. Les départements informatiques doivent toujours répondre rapidement aux nouvelles exigences des entreprises, tout en réduisant constamment le coût informatique de l'entreprise et en absorbant et en intégrant de manière transparente de nouveaux partenaires et clients commerciaux. Le secteur d'activité des logiciels a connu de nombreuses architectures conçues pour permettre un traitement entièrement distribué, des langages de programmation destinés à n'importe quelle plate-forme et une myriade de produits de connectivité conçus pour permettre une intégration plus rapide et plus efficace des applications, et pour réduire considérablement les temps de mise en oeuvre. Toutefois, une solution globale ne semble être encore qu'une utopie.

De nos jours, les architectures orientées services sont présentées comme l'étape suivante de l'évolution qui permet d'aider les organisations informatiques à relever des défis toujours plus complexes. Cependant, une question se pose : les architectures orientées services sont-elles réelles ? Si elles peuvent être conçues et décrites, peuvent-elles être mises en oeuvre ? Ce document de présentation technique démontre que les possibilités offertes par les architectures orientées services ne sont pas uniquement des promesses. Une fois la publicité et toutes les merveilleuses promesses concernant ces produits passées, les organisations informatiques comprendront que les architectures orientées services fournissent la meilleure base d'intégration de nouveaux systèmes d'applications, tout en permettant de tirer profit du parc informatique existant. Ce document de présentation technique est le premier d'une série conçue pour vous permettre de comprendre la vraie valeur d'une architecture orientée services et vous aider à développer un programme réaliste d'évaluation de votre infrastructure actuelle et à effectuer une migration vers une architecture orientée services.

Depuis quelques temps déjà, l'existence de technologies de services Web a provoqué de nombreux débats au sujet des architectures orientées services. Le sujet n'est pas nouveau, le concept a été développé depuis plus d'une dizaine d'années, depuis que la technologie CORBA a permis de penser que l'intégration d'applications sur des plates-formes hétérogènes était possible.

Les problèmes d'intégration d'applications diverses se sont posés, car souvent des modèles nombreux et très différents (ne prenant pas en charge la norme CORBA) sont devenus à la mode. De nombreux architectes et ingénieurs se sont donc trouvés tellement submergés de problèmes technologiques à résoudre que le développement d'une architecture plus robuste permettant une intégration simple, rapide et hautement sécurisée des systèmes et des applications n'a pu avoir lieu. Malheureusement, le problème persiste et devient de plus en plus complexe chaque année. Les besoins fondamentaux de votre entreprise doivent vous conduire à rechercher une meilleure solution. Des besoins tels que la réduction des coûts, la réduction des cycles, l'intégration des systèmes dans l'ensemble de l'entreprise, l'intégration de systèmes interentreprises et orientés client, un retour sur investissement plus rapide et la création d'un modèle plus réactif et plus adaptatif. Mais, il est de plus en plus fréquent que les solutions point à point ne résolvent pas les problèmes de base. Il manque un cadre architectural cohérent qui permette le développement, l'intégration et la réutilisation rapides des applications. Plus important encore, vous avez besoin d'un cadre architectural qui permette de rassembler les composants et les services afin de fournir des solutions dynamiques à mesure que votre entreprise évolue. Ce document de présentation technique va au-delà du débat sur l'efficacité sans conteste des technologies telles que les services Web. Il propose une vision architecturale sans limite par technologie.
Vous devriez tout d'abord prendre en compte certains problèmes fondamentaux qui sous-tendent votre recherche d'une meilleure base informatique. La manière dont vous allez répondre à ces problèmes va déterminer votre niveau de réussite.

Problème n°1 : la complexité

Votre organisation informatique rencontre toujours les mêmes problèmes. La direction pousse à une meilleure utilisation des ressources informatiques, un meilleur retour sur investissement, l'intégration de systèmes historiquement distincts et la mise en oeuvre plus rapide de nouveaux systèmes. Toutefois, aujourd'hui, certains aspects informatiques ont changé. Les environnements sont plus complexes. Les contraintes budgétaires et les exigences d'efficacité requièrent la réutilisation et non le remplacement des systèmes déjà en place. L'accès à faible coût et omniprésent à Internet a créé la possibilité de modèles d'entreprises entièrement nouveaux que vous devez évaluer pour rester compétitifs. Les fusions et les acquisitions d'entreprises sont devenues monnaie courante, de telle sorte que des organisations, des applications et des infrastructures informatiques entières doivent être intégrées et absorbées. Dans un environnement d'une telle complexité, les solutions point à point ne font qu'exacerber les problèmes et ne relèvent jamais véritablement les défis. Vous devez développer des systèmes qui incorporent le caractère hétérogène des organisations, en tant que composant essentiel de votre environnement informatique, de sorte que ces systèmes puissent prendre en charge une gamme diverse et

D'après Aberdeen Group, des enquêtes auprès de responsables des technologies de l'information de Global 2000 montrent à chaque fois que les coûts, la complexité et le temps d'intégration des applications d'entreprise, ainsi que l'intégration inter-entreprises, sont des priorités absolues. Même en cas de réduction budgétaire et de réduction des marges de bénéfices, les avantages d'une stratégie d'intégration solide sont tels que ces responsables prévoient de dépenser entre 35 et 60 % de leurs budgets dans des projets d'intégration.[1]

sans limite de matériel, de systèmes d'exploitation, de logiciels intermédiaires, de langages et de stockage de données. Des décennies de croissance et d'évolution en termes informatiques ont créé la complexité à laquelle vous devez faire face aujourd'hui. Au vu de tous les défis que l'informatique doit relever, il n'est pas étonnant que l'intégration d'applications soit la priorité absolue de beaucoup de responsables des technologies de l'information.

Problème n° 2 : une programmation redondante et ne pouvant être réutilisée

Comme dans beaucoup de sociétés, l'éventail de vos applications a peut-être augmenté en raison de fusions ou d'acquisitions d'autres entreprises. Le résultat est qu'aujourd'hui vous devez peut-être faire face à des applications redondantes ou à des applications dont les fonctions ne peuvent être facilement réutilisées. Il se peut que chaque unité de votre organisation ait travaillé indépendamment des autres, ce qui freine l'effort de coordination visant à créer un parc ou des services informatiques fonctionnels réutilisables. Cette redondance augmente les coûts et le temps de mise sur le marché pour le déploiement de nouveaux produits et services, car les modifications doivent être effectuées pour chaque application ou système concerné. Le fait de ne pouvoir réutiliser certaines fonctions requiert, en fin de compte, plus de ressources, et souvent plus de temps, pour livrer de nouvelles applications.

Problème 3 : des interfaces multiples

Pensez au problème d'intégration *n*(*n-1*). Toutes les organisations doivent faire face à des problèmes d'intégration, quelle que soit leur nature, en raison d'une fusion, d'un nouveau partenariat commercial ou simplement d'un besoin d'interconnexion entre des systèmes existants. Si n systèmes d'applications doivent être directement interconnectés, le processus va créer n(n-1) connexions ou interfaces. Dans la figure 1, chaque pointe de flèche représente une interface.

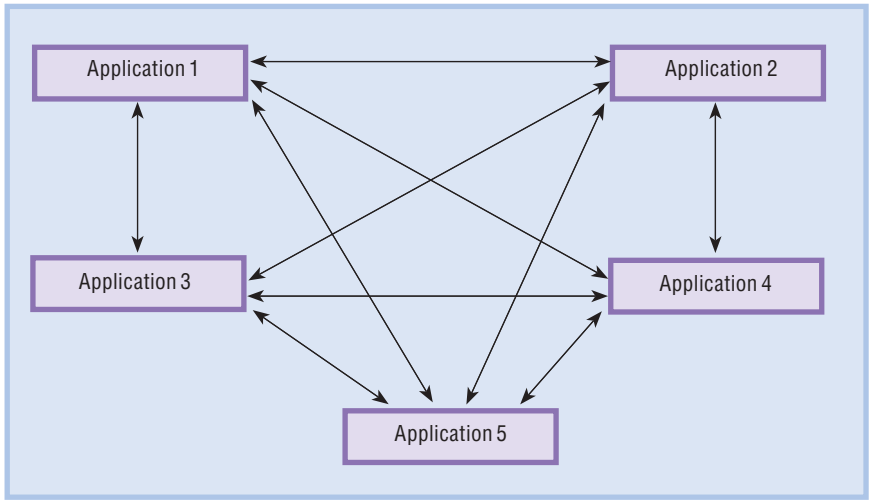

Figure 1. Le problème d'intégration n(n-1).

Par conséquent, si vous avez besoin d'intégrer un autre système d'applications A(*n+1*), vous devrez créer, documenter, tester et maintenir *2n* interfaces nouvelles. Dans la figure 1, l'ensemble de cinq applications requiert 20 interfaces directes. L'ajout d'une sixième application supposerait dix nouvelles interfaces. Et pour augmenter la complexité de l'opération, vous devez également modifier le code de chaque application existante pour inclure les nouvelles interfaces : vous générez ainsi des coûts de test substantiels. Pour réduire les coûts et la complexité de l'intégration, vous avez besoin d'une solution optimale qui produise un nombre minimal d'interfaces *n* pour *n* applications, avec une seule nouvelle interface pour chaque ajout de système. Toutefois, cette opération ne peut être effectuée par connexion directe.

Et l'avenir ?

Au cours des quarante dernières années, le développement logiciel a connu plusieurs modèles de programmation différents. Chaque changement devait répondre à des niveaux de complexité logicielle plus élevés et aux demandes d'architectes souhaitant rassembler les applications par parties, composants ou services. Récemment, la technologie Java™ a permis une programmation indépendante de la plate-forme utilisée. Le langage XML a fourni des données auto-descriptives et indépendantes de la plate-forme utilisée. De nos jours, les services Web ont supprimé un obstacle supplémentaire en permettant aux applications de se connecter entre elles indépendamment des modèles d'objets. Par exemple, grâce au schéma de message simple XML, les applications Java peuvent appeler des applications Microsoft® .NET ou prenant en charge la norme CORBA, ou même le langage COBOL. Les transactions IBM CICS ou IBM IMS sur un mainframe à Singapour peuvent donc être sollicitées par une application .NET qui peut à son tour être appelée par un agent exécutant un serveur IBM Lotus Domino à Munich. Le plus gros avantage est que l'application appelante n'a pas besoin de connaître l'emplacement d'exécution de la transaction, le langage dans lequel elle a été écrite ou le chemin entrepris par le message. Un service est demandé et une réponse est envoyée.

Il était plus probable que les services Web soient adoptés en tant que norme de facto permettant de fournir une interaction efficace, fiable, évolutive et extensible entre les ordinateurs, au détriment de n'importe quel autre de leurs prédécesseurs. En effet, l'apparition opportune de plusieurs technologies et conditions préalables culturelles nécessaires ont contribué à cette adoption, par exemple :

- *Une infrastructure de réseau à faible coût, basée sur des normes ouvertes et omniprésente et des technologies offrant un environnement distribué qui incitent beaucoup plus à adopter des services Web plutôt que des environnements CORBA et des environnements ETCD.*
- *Un niveau d'acceptation et de stabilité technologique qui fonctionne au sein d'un environnement centré autour d'un réseau requérant un interfonctionnement pour atteindre des objectifs d'entreprise fondamentaux, tels qu'une collaboration distribuée.*
- *Un consensus selon lequel l'interfonctionnement à faible coût est obtenu à l'aide de normes basées sur Internet et autres technologies similaires.*
- *La stabilité des technologies de réseau (telles que TCP/IP), des ensembles d'outils (environnements de développement intégrés [IDE] et UML (Unified Modeling Language), des plates-formes (telles que Java 2 Platform, Enterprise Edition [J2EE]) et d'autres technologies semblables (telles que la technologie et les services OO (orientés objet)), qui fournissent l'infrastructure nécessaire à l'amélioration des interactions entre ordinateurs à couplage lâche et à interfonctionnement (état bien plus avancé que celui des utilisateurs CORBA).*

Une architecture orientée services peut être à la fois une architecture et un modèle de programmation, une manière de concevoir la création logicielle. Une architecture orientée services permet de concevoir des systèmes logiciels fournissant des services à d'autres applications à l'aide d'interfaces publiées et découvrables, et où les services peuvent être sollicités via un réseau. Lorsque vous mettez en oeuvre une architecture orientée services à l'aide de technologies de services Web, vous créez une nouvelle méthode de constitution d'applications au sein d'un modèle de programmation plus puissant et plus flexible. Vous pouvez réduire les coûts de développement et de propriété, ainsi que les risques de mise en oeuvre.

Mais, les avantages à long terme sont encore plus significatifs. Tout d'abord, le calcul distribué, qui représente bien plus que l'application de millions d'instructions par seconde (MIPS) pour mettre en place une solution informatique. Le calcul distribué fournit également un cadre permettant de localiser, de relocaliser, d'équilibrer et de gérer de manière dynamique d'un nombre considérable de services afin que vous puissiez garantir que les applications nécessaires sont toujours disponibles et sécurisées, quelle que soit la charge du système.

Ce cadre donne naissance à son tour au concept d'informatique à la demande, qui pourrait être mis en oeuvre dans n'importe quelle configuration, qu'il s'agisse d'un simple groupe de serveurs ou d'un réseau de systèmes IBM SP2 à 1024 noeuds. Lorsqu'un utilisateur doit résoudre un problème et a besoin d'appliquer les ressources informatiques appropriées à ce problème (ni plus, ni moins), vous avez la possibilité de ne payer que les ressources utilisées. L'utilisation efficace de ces nouvelles capacités requiert la restructuration de nombreuses applications existantes. Des applications monolithiques existantes peuvent être exécutées dans ces environnements, sans jamais utiliser les ressources disponibles de manière optimale. Dans ces circonstances et étant donné les problèmes cités, votre infrastructure doit subir un changement radical : la conversion à une architecture orientée services.

Exigences concernant une architecture orientée services
Au vu des problèmes débattus précédemment, il devrait être évident qu'il est important de développer une architecture qui satisfasse toutes vos exigences. Ces exigences devraient inclure les éléments suivants :

- Exploitation du parc informatique existant.
  Il s'agit de la condition la plus importante. Il est rare que des systèmes existants puissent être éliminés. Ils contiennent certainement des données de grande importance pour votre entreprise. D'un point de vue stratégique, l'objectif est de constituer une nouvelle architecture dont le rendement corresponde à ce que vous souhaitez. Mais d'un point de vue tactique, les systèmes existants doivent être intégrés de sorte qu'à terme, ils puissent être décomposés et remplacés par des projets gérables et incrémentiels.

- Prise en charge de tous les types d'intégrations requis.
  Cela comprend l'interaction avec l'utilisateur (pour fournir un fonctionnement unique et interactif), la connectivité des applications (pour créer un modèle de communication qui sous-tend toute l'architecture), l'intégration des processus (pour organiser les applications et les services), l'intégration des informations (pour rassembler et déplacer les données de l'entreprise) et la possibilité d'effectuer de futures intégrations (pour créer et déployer de nouveaux services et applications).

- Possibilité de mises en oeuvre incrémentielles et de migration du parc.
  Si cette exigence est satisfaite, l'un des aspects essentiels du développement de l'architecture pourra être mis en place : la possibilité de produire un retour sur investissement différentiel. D'innombrables projets d'intégration ont échoué en raison de leur complexité, de leur coût et de temps de mise en oeuvre impossibles à réaliser.

- Constitution dans un cadre de composants standard.
  Vous devez inclure un environnement de développement créé autour d'un cadre de composants standard pour promouvoir une meilleure réutilisation des modules et des systèmes, permettre au parc informatique existant de migrer vers le cadre en question et permettre une mise en oeuvre opportune des nouvelles technologies.

- Possibilité de mise en oeuvre de nouveaux modèles informatiques.
  Des exemples spécifiques incluent les nouveaux modèles client de portails, le calcul distribué et l'informatique à la demande.

Une architecture orientée services : au-delà des services Web

L'avènement des services Web a précipité le changement fondamental dans le développement, le déploiement et la gestion des infrastructures informatiques. La réussite de nombreux projets de services Web a montré que la technologie permettant de mettre en oeuvre une véritable architecture orientée services existe. Elle vous permet de prendre du recul pour examiner votre architecture d'applications et les problèmes de base que vous essayez de résoudre. D'un point de vue de l'entreprise, le problème n'est plus uniquement un problème technologique. Il s'agit de développer une architecture et un cadre d'applications au sein desquels vous pourrez définir des problèmes et mettre en oeuvre des solutions que vous pourrez réutiliser de manière cohérente.

Toutefois, il est important d'abord de comprendre qu'une architecture orientée services ne se limite pas aux services Web. Les services Web constituent un ensemble de technologies, y compris les technologies XML, SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) et UDDI (Universal Description, Discover and Integration), qui vous permet de créer des solutions de programmation pour des problèmes de messagerie et d'intégration d'applications spécifiques. Ces technologies se stabiliseront à terme et seront finalement remplacées par de meilleures technologies plus efficaces et plus robustes. Mais pour l'instant, les technologies existantes sont suffisantes et ont déjà démontré que vous pouvez dès à présent mettre en oeuvre une architecture orientée services.

Quels sont les éléments constitutifs d'une architecture orientée services ? Une architecture orientée services porte bien son nom car il s'agit en effet d'une architecture. Ce n'est pas uniquement un ensemble spécifique de technologies, telles que les services Web. Elle transcende ces technologies et, en théorie, en est complètement indépendante. Au sein d'un environnement d'entreprise, une définition purement architecturale d'une architecture orientée services pourrait être la suivante : architecture d'applications au sein de laquelle toutes les fonctions sont définies en tant que services indépendants, comprenant des interfaces bien définies qui peuvent être sollicitées dans des séquences définies pour créer des processus d'entreprise. Remarquez les composants de la définition suivante :

- *Toutes les fonctions sont définies comme des services. Ceci comprend les fonctions purement commerciales (telles que la création d'une application d'hypothèques ou de bons de commande), des transactions composées de fonctions de niveau inférieur (telles que les rapports de crédit ou la vérification de l'embauche) et les fonctions de services système (telles que la validation de l'identification ou l'obtention d'un profil utilisateur). Cet élément pose la question de la granularité qui sera analysée ultérieurement.*
- *Tous les services sont indépendants. Ils fonctionnent comme des " boîtes noires ", des composants externes qui ne savent pas et ne se soucient pas de la méthode d'exécution de la fonction et ne font que fournir le résultat attendu.*
- *Les interfaces peuvent être appelées, au sens le plus large : au niveau architectural, le fait que ces interfaces soient locales (au sein du système) ou distantes (externes au système immédiat) n'a aucune importance. Le schéma d'interconnexion ou le protocole utilisés pour l'appel, ou les composants d'infrastructure requis pour la connexion importent peu. Le service peut se trouver au sein de la même application, à une adresse différente dans un multiprocesseur asymétrique, sur un système complètement différent sur un intranet d'entreprise ou sur une application du système d'un partenaire utilisée dans une configuration interentreprises.*

L'élément clé et le centre d'attention de l'application appelante d'une architecture orientée services est l'interface. Elle définit les paramètres requis et la nature du résultat, et, par conséquent, la nature du service et non la technologie utilisée pour la mise en oeuvre de ce dernier. Le système doit effectuer et gérer la sollicitation du service et non l'application appelante. Cette fonction permet la réalisation de deux caractéristiques essentielles : tout d'abord, l'indépendance véritable des services et deuxièmement, la possibilité de gérer ces derniers. La gestion comprend plusieurs fonctions :

- *La sécurité, l'autorisation des requêtes, le cryptage et le décryptage des données, selon les instructions données, et la validation des informations.*
- *Le déploiement permet au service d'être déplacé au sein du réseau pour maximiser les performances et supprimer la redondance afin que la disponibilité des applications soit maximale.*
- *La journalisation propose des fonctions d'audit informatique et de taxation.*
- *Le reroutage dynamique fournit des fonctions de reprise sur incident et d'équilibrage de la charge.*
- *La maintenance permet de gérer de nouvelles versions du service.*

Nature d'un service

Qu'est-ce qu'un service ? Comme nous l'avons déjà indiqué, un service au sein d'un environnement d'entreprise constitue en général une fonction simple (telle que getStockQuote, getCustomerAddress ou checkCreditRating), une transaction plus complexe (telle que commitInventory, sellCoveredOption ou scheduleDelivery) ou un service système (tel que logMessageIn ou authenticateUser). D'un point de vue de l'application, ces fonctions sont des fonctions non système atomiques. Les transactions peuvent sembler n'être que de simples fonctions pour l'application appelante, mais elles peuvent être mises en oeuvre en tant que fonctions composites masquées par leur contexte transactionnel propre. Elles impliquent parfois de multiples fonctions de niveau inférieur transparentes pour l'appelant. Les fonctions système sont des fonctions généralisées qui peuvent être extraites d'une plate-forme spécifique, telle que Microsoft Windows® ou Linux.

Ceci n'est pas une simple distinction artificielle entre les différents services. D'un point de vue de l'application, tous les services sont atomiques, mais le fait qu'il s'agisse de services d'entreprise ou système n'a aucune importance. Cette distinction ne sert qu'à introduire le concept important de granularité. La décomposition des applications d'entreprise en services ne constitue pas uniquement un processus abstrait, elle implique également des opérations très pratiques. Les services peuvent être composés de fonctions de niveau inférieur (granularité fine) ou de niveau supérieur complexe (granularité grossière). Il existe de véritables compromis de performance, de flexibilité, de possibilités de maintenance et de réutilisation basés sur les définitions de ces fonctions. Le niveau de granularité indique la richesse fonctionnelle d'un service. Par exemple, plus la granularité d'un service est grossière, plus la fonction offerte par le service est riche. Les services sont en général composés de fonctions commerciales à granularité grossière, telles que openAccount, car ces opérations peuvent supposer l'exécution de multiples opérations à granularité plus fine, telles que verifyCustomerIdentity et createCustomerAccount. Ce

processus de définition de services est en général effectué dans un cadre plus large, le cadre d'applications. C'est ce cadre qui doit être mis en place. Il s'agit de développer un cadre d'applications de composants dans lequel les services sont définis en tant qu'ensemble de composants réutilisables pour créer de nouvelles applications ou intégrer un parc logiciel existant.

Ce type de cadre existe aujourd'hui sous de multiples formes ; chez IBM, de nombreux cadres, tels que EWA (Enterprise Workframe Architecture), JADE[2] et Struts (de Jakarta), sont utilisés lors d'intégrations client. Prenons par exemple EWA, de l'équipe IBM Software Group Advanced Technology Solutions. A un niveau très élevé, le cadre ressemble à celui de la figure 2. Au sein de ce cadre, une configuration définit une application. Il décrit également les composants de l'application, ainsi que la séquence et la méthode d'appel. Les données en entrée sont reçues et envoyées à l'application indépendamment de la source. Par exemple, l'ajout d'une connexion Internet à une application bancaire disposant d'un accès ATM existant est donc transparent pour la logique de l'application et ce, grâce au périphérique frontal et aux programmes de gestion des protocoles. Les services au niveau système sont fournis par des fonctions de base et des composants d'accès spécifique permettent une connexion aux applications principales, de sorte qu'elles puissent rester sur place ou être migrées ultérieurement. La technologie EWA peut entièrement prendre en charge la technologie J2EE et se connecter à des systèmes de composants externes DCOM ou CORBA.

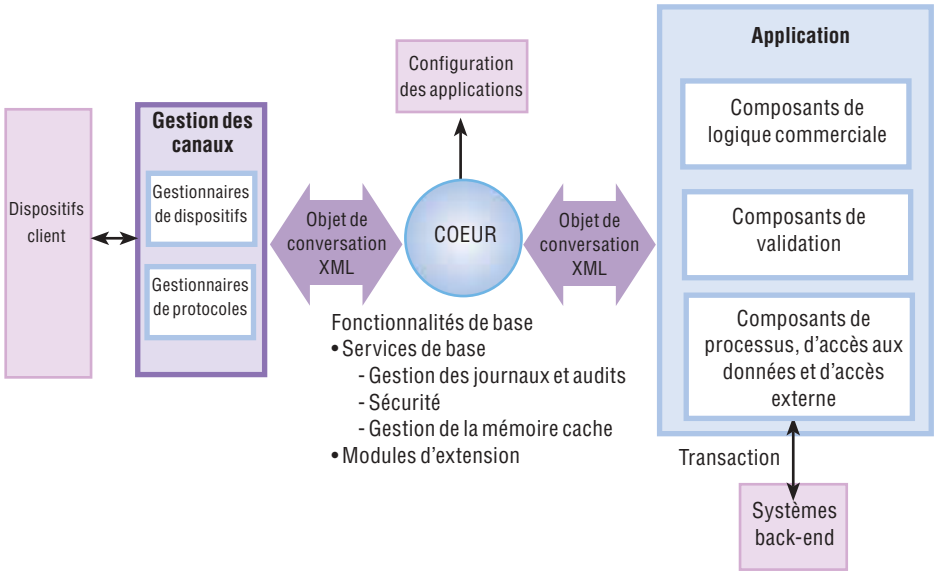

Figure 2. Le cadre EWA

De nos jours, le cadre EWA comprend plus de 1500 composants généraux et spécialisés, réduisant ainsi considérablement la quantité de code à écrire lors de la création d'une nouvelle application.

Réponses aux problèmes anciens

Si nous revenons à présent au premier exemple d'intégration mentionné, que pensez-vous d'un schéma qui minimise le nombre d'interfaces requises, tel que celui présenté à la figure 3 ?

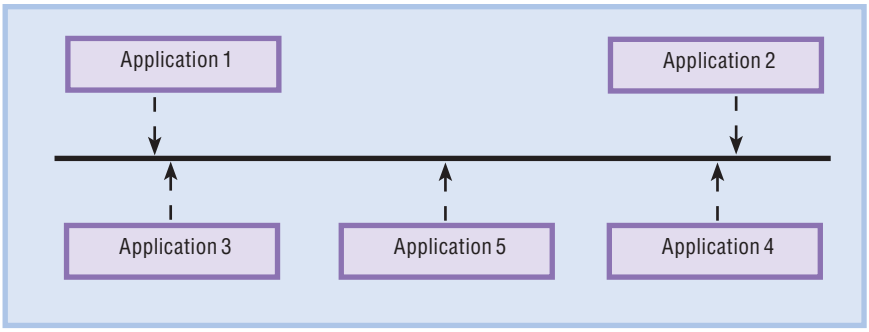

Figure 3. Intégration des applications

La figure 3 peut paraître trop simpliste, mais elle illustre bien le point de départ de l'intégration au sein d'un cadre tel que le cadre EWA. Vous pouvez ajouter le concept architectural de bus de services, représenté à la figure 4 par la ligne épaisse au centre, et un gestionnaire de services ou de flux pour connecter les services et fournir un chemin aux requêtes de services. Le gestionnaire de flux traite une séquence d'exécution définie, ou *flux de services*, qui sollicite les services requis dans la séquence adéquate pour produire le résultat final. Le langage BPEL (Business Process Execution Language) est un exemple de cette technologie de définition d'un processus en tant qu'ensemble d'appels de services.

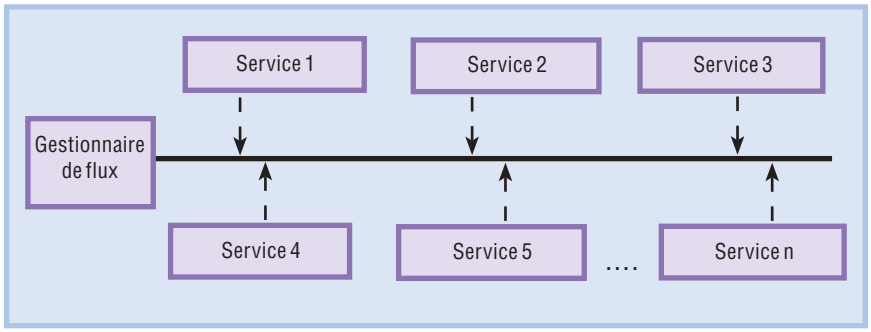

Figure 4. Intégration des services

A ce stade, vous devez déterminer la méthode de sollicitation des services afin d'ajouter la configuration des applications. Puis, vous devez virtualiser les données en entrée et les résultats. Enfin, il vous faut fournir une connectivité aux processus principaux. Il en résulte un cadre global qui permet aux processus d'être exécutés tels quels et qui offre la possibilité d'une migration future. A présent, comme l'indique la figure 5, l'ensemble de haut niveau est complet du point de vue structurel.

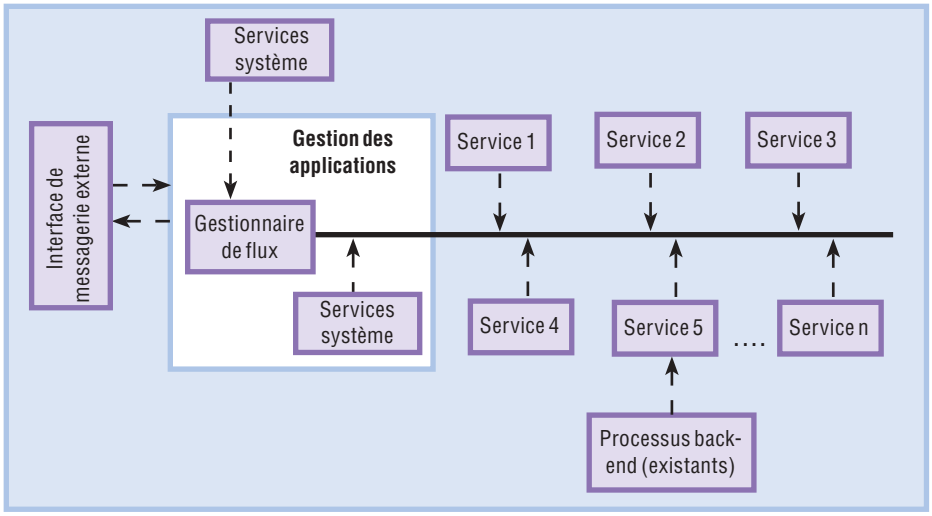

Figure 5. L'infrastructure complète

Ne vous étonnez pas si la figure 5 ressemble à un diagramme de blocs EWA. Au plus haut niveau, tout cadre d'applications robuste doit proposer ces fonctions. A présent, le véritable travail commence : la constitution de 1500 composants pour matérialiser ce cadre. Le processus de décomposition des applications existantes en composants du cadre suppose un certain travail. Inutile donc de réinventer tous les autres composants généraux et système nécessaires. De nombreux architectes choisissent de mettre en oeuvre ces composants au sein d'un cadre existant. Quelle que soit votre approche, vous devez mettre en oeuvre cette architecture en utilisant des technologies et des cadres qui existent de nos jours, ce qui nous ramène au point de départ : l'analyse des problèmes que vous souhaitez résoudre. Vous devez trouver des solutions à ces problèmes avec la certitude que votre architecture pourra être mise en oeuvre.

Exigences d'intégration au sein de l'architecture

Jusqu'à présent, le débat sur l'intégration s'est limité à l'intégration d'applications via des services de composants, mais l'intégration est un sujet beaucoup plus vaste. Lorsque vous évaluez vos exigences en ce qui concerne une architecture, vous devez prendre en compte plusieurs types d'intégrations. Vous devez penser non seulement à l'intégration d'applications, mais également à celle de l'interface de l'utilisateur final, à la connectivité des applications, à l'intégration des processus, à l'intégration des informations et au modèle de développement prêt à l'intégration.

L'intégration au niveau de l'interface de l'utilisateur final correspond à la manière dont l'ensemble complet d'applications et de services auquel un utilisateur donné accède est intégré afin de fournir une interface utilisable, efficace et cohérente. Ce sujet est en évolution constante et les nouveaux développements, à court terme, traiteront essentiellement des avancées dans l'utilisation des serveurs de portails. Alors que les portlets peuvent d'ores et déjà solliciter des composants de services locaux à l'aide de services Web, de nouvelles technologies, telles que les services Web pour portlets distants, activeront des fournisseurs de contenu et d'applications pour créer des services interactifs prêts à être utilisés avec des portails via Internet et offriront donc de nombreuses possibilités nouvelles d'intégration.

La connectivité d'applications est un type d'intégration relatif à tous les types de connectivités devant être pris en charge par l'architecture. A un certain niveau, cette intégration traite des communications synchrones et asynchrones, du routage, des transformations, de la distribution rapide des données, et des convertisseurs de passerelles et de protocoles. A un autre niveau, elle s'occupe également de la virtualisation des données en entrée et des résultats, ou des sources et des récepteurs, tels que les programmes de gestion de canaux et de protocoles de la figure 2. Le problème réside dans la manière fondamentale dont les données entrent, sortent et se déplacent dans le cadre qui met en oeuvre l'architecture.

L'intégration des processus traite du développement des processus de calcul qui fournissent et appliquent des solutions pour les processus d'entreprise, l'intégration des processus d'application et l'intégration des processus à d'autres processus. La première exigence en ce qui concerne l'architecture peut sembler évidente : l'architecture doit permettre la mise en place d'un environnement au sein duquel les problèmes de base de l'entreprise peuvent être mis sous formes de modèles. Toutefois, une analyse insuffisante à ce niveau peut présenter des défis significatifs pour toute mise en oeuvre d'architecture, quelle que soit la complexité technique. L'intégration d'applications fait partie des processus et peut inclure des applications de l'entreprise ou l'appel d'applications ou de services sur des systèmes distants, par exemple, ceux d'un partenaire commercial. L'intégration au niveau des processus peut ainsi comprendre l'intégration de processus entiers, outre celle de services spécifiques, à partir de sources externes, telles que la gestion de la chaîne logistique ou les services financiers qui couvrent de très nombreuses organisations. Pour de tels besoins d'intégration d'applications et de processus, vous pouvez utiliser des technologies telles que le langage BPEL pour services Web (BPEL4WS). Le cadre d'applications peut également utiliser un schéma de configuration de programme, tel que le cadre EWA mentionné auparavant. Un schéma de configuration de niveau supérieur peut être constitué à l'aide de la technologie BPEL4WS à un niveau inférieur, puis commandé par un moteur qui ne fournit pas uniquement des fonctions de gestion des flux. Avant la constitution de ce schéma, vous devez tout d'abord comprendre vos exigences architecturales afin de mettre en place l'infrastructure appropriée.

L'intégration des informations est un processus qui fournit un accès cohérent à toutes les données de l'entreprise pour toutes les applications, quels que soient la manière dont ces données doivent être envoyées, leur format, leur source ou leur emplacement. Lors de la mise en oeuvre, cette exigence peut supposer la simple utilisation d'un logiciel adaptateur et d'un moteur de transformation. Toutefois, en général, le processus est plus complexe. Le concept clé est souvent la virtualisation des données qui peut inclure le développement d'un bus de données à partir duquel toutes les applications de l'entreprise peuvent envoyer des requêtes de données via des services ou des interfaces standard. Les données peuvent ainsi être envoyées à l'application qu'elles proviennent d'une feuille de calcul, d'un fichier natif, d'une base de données SQL (Structure Query Language), d'un autre type de base de données ou d'un stockage de données en mémoire. Il existe également la possibilité que l'application ne connaisse pas le format de stockage permanent des données. L'application ne connaît pas le système d'exploitation qui gère les données ; les fichiers natifs de systèmes IBM AIX ou Linux sont donc accessibles de la même manière que des fichiers Windows, IBM z/OS ou de presque n'importe quel autre système. L'emplacement des données est aussi transparent car il est fourni par un service commun. La récupération locale ou distante des données, ainsi que la présentation de ces dernières au format requis, incombe au service d'accès et non à l'application.

Enfin, l'une des exigences vis-à-vis d'un environnement de développement d'applications doit être la suivante : cet environnement doit prendre en compte tous les types et niveaux d'intégration pouvant être mis en oeuvre dans l'entreprise et fournir le développement et le déploiement nécessaires. Pour être véritablement robuste, l'environnement de développement doit inclure et appliquer une méthodologie qui indique clairement la manière dont les services et les composants sont conçus et constitués, qui facilite la réutilisation, supprime la redondance et simplifie le test, le déploiement et la maintenance.

Tous les types d'intégrations mentionnés ci-avant seront représentés au sein de l'entreprise, même si dans certains cas, ils seront simplifiés ou ne seront pas clairement définis. Ainsi, tous les types d'intégrations doivent être pris en compte lors de la transformation du cadre architectural de l'entreprise. Votre environnement informatique peut ne disposer que d'un nombre restreint de types de sources de données, l'intégration des informations n'en sera que plus simple. Il se peut également que la portée de la connectivité d'applications soit limitée. Les fonctions

d'intégration du cadre doivent quand même être fournies par des services et non par des applications ad hoc, si vous souhaitez que le cadre puisse croître et changer à mesure que votre entreprise évolue.

Avantages du déploiement d'une architecture orientée services

Une architecture orientée services peut souvent évoluer à partir de systèmes existants et ne requiert donc pas de réécriture système complète. Les organisations qui concentrent leurs efforts de développement sur la création de services, en utilisant des technologies existantes associées à une approche basée sur les composants vis-à-vis du développement logiciel, profiteront de nombreux avantages.

- Exploitation du parc informatique existant
  Cet avantage est le premier et le plus important de toutes les exigences mentionnées précédemment. Vous pouvez constituer un service en ajoutant des composants existants à l'aide d'un cadre d'architecture orientée services adapté et mis à la disposition de votre entreprise. L'utilisation de ce nouveau service requiert uniquement la connaissance de l'interface et du nom du service. Les spécificités de mise en oeuvre du service (l'architecture des composants, par exemple) ou les composants fonctionnels discrets, ainsi que la complexité du flux de données au sein des composants du service, sont transparents pour les appelants. L'anonymat des composants permet aux organisations de tirer profit des systèmes actuels, de créer des services à partir d'un ensemble de composants installés sur différents ordinateurs exécutant différents systèmes d'exploitation et développés dans des langages de programmation différents. Les systèmes existants peuvent être encapsulés et leur accès est possible grâce aux interfaces des services Web. Plus important encore, ils peuvent être transformés et prendre ainsi de la valeur à mesure que leur fonctionnalité est transformée en services.

- L'infrastructure comme matière première
  Le développement et le déploiement de l'infrastructure deviennent de plus en plus cohérents sur les diverses applications de l'entreprise. Les composants existants, les nouveaux composants et les composants achetés auprès de différents fournisseurs peuvent être consolidés au sein d'un cadre d'architecture orientée services bien défini. Un tel ensemble de composants est déployé sous forme de services dans l'infrastructure existante. L'infrastructure sous-jacente devient alors une matière première. Ensuite, à mesure que le couplage des services et du matériel prenant ces derniers en charge devient plus lâche, vous pouvez optimiser le matériel car le programme d'assemblage de services ne dépend plus de l'environnement matériel dans lequel le service fonctionne au moment de l'exécution.

- Temps de mise sur le marché plus court
  Les bibliothèques d'organisation de services Web deviennent les atouts essentiels de votre organisation en faisant partie du cadre d'architecture orientée services. La création et le déploiement de services à l'aide de ces bibliothèques de services Web réduit considérablement le temps de mise sur le marché, car les nouvelles initiatives réutilisent les services et les composants existants, réduisent le temps de conception, de développement, de test et de déploiement du processus. A mesure que les services atteignent un volume critique dans votre organisation ou sur le réseau sécurisé, un écosystème plus vaste fait son apparition et vous permet d'assembler des applications composites utilisant des services, au lieu de développer des applications personnalisées.

- Réduction des coûts
  A mesure que de nouvelles exigences apparaissent et que celles des entreprises évoluent, le coût de l'amélioration et de la création de nouveaux services diminue grâce à l'adaptation du cadre d'architecture orientée services et de la bibliothèque de services pour les applications nouvelles et existantes. Le temps d'apprentissage de l'équipe de développement est également réduit car ses membres sont déjà familiarisés avec les composants existants.

- Atténuation des risques
  La réutilisation de composants existants réduit le risque d'introduction de nouveaux échecs dans le processus d'amélioration ou de création de nouveaux services. Le risque de maintenance et de gestion de l'infrastructure prenant en charge les services diminue également.

- Amélioration permanente des processus d'entreprise
  Une architecture orientée services *permet une représentation claire des flux de processus identifiés par l'ordre des composants utilisés pour un service spécifique. Elle fournit aussi aux utilisateurs un environnement idéal pour le contrôle des opérations de l'entreprise. Les modèles de processus se retrouvent au niveau des services. La manipulation des processus est possible grâce à la réorganisation des éléments (composants d'un service) selon un schéma. Cette fonction permet de changer les flux de processus tout en contrôlant les résultats afin de faciliter l'amélioration permanente des processus.*

- Architecture centrée autour des processus
  Les modèles et les pratiques d'architectures existants se concentrent souvent sur les processus. Les applications sont développées pour convenir aux programmeurs. La connaissance des processus est souvent dispersée parmi les composants. L'application fonctionne comme une boîte noire, sans aucune granularité hors de l'application. La réutilisation suppose la copie du code, l'incorporation des bibliothèques partagées ou l'héritage des objets. Dans une architecture centrée autour des processus, l'application est développée pour le processus. Ce dernier est décomposé en une série d'étapes qui représentent chacune un service. En réalité, chaque fonction de service ou de composant constitue une sous-application. Ces sous-applications peuvent être assemblées pour créer un flux de processus capable de satisfaire les besoins de l'entreprise. Cette granularité permet de tirer profit des processus et de réutiliser chaque sous-application dans toute l'organisation.

L'avenir : de nouveaux modèles, de nouvelles exigences

Jusqu'à présent, ce document de présentation technique s'est concentré sur le besoin d'augmenter la vitesse des changements dans l'entreprise et d'améliorer la performance et l'efficacité de cette dernière. Ces exigences requièrent un ensemble d'impératifs informatiques en termes de flexibilité et l'architecture orientée services devient alors un élément dynamisant clé. Mais, que se passe-t-il si un modèle de développement d'application entièrement nouveau apparaît ? Est-ce que la notion d'architecture orientée services aura toujours un sens ou sera toujours une nécessité ? La réponse est sans le moindre doute oui. Deux nouveaux concepts émergeants commencent à être mis en oeuvre : le calcul distribué et l'informatique à la demande. Même si ces modèles sont différents et ont été développés séparément, ils sont intimement liés et chacun rend encore plus nécessaire l'évolution vers une architecture orientée services. La représentation de toutes les applications, ressources ou possibilités d'entreprise sous forme de service doté d'une interface normalisée vous permet d'associer rapidement les applications nouvelles et existantes pour répondre à l'évolution des besoins et améliorer l'efficacité opérationnelle de l'entreprise, ce qui caractérise l'architecture orientée services. Par conséquent, l'architecture orientée services devient la base même du calcul distribué et de l'informatique à la demande.

Calcul distribué
Le présent document de présentation technique n'a pas pour objet de débattre en profondeur du calcul distribué, mais il est tout de même nécessaire de mentionner quelques points. Tout d'abord, le calcul distribué représente bien plus que l'application de millions d'instructions par seconde (MIPS) pour mettre en place une solution informatique à un problème complexe. Il vous permet de diviser les ressources en de multiples environnements d'exécution en appliquant un ou plusieurs concepts, tels que le partitionnement matériel ou logiciel, l'exploitation en temps partagé, la simulation et l'émulation d'ordinateurs, et la qualité de service. Cette virtualisation, ou déploiement à la demande, de toutes les ressources informatiques distribuées vous permet de vous servir de ces dernières quels que soient l'emplacement et la manière dont elles doivent être utilisées dans l'environnement distribué. La virtualisation est une méthode simple de gestion des ressources de périphériques, de stockage, d'applications, de services ou d'objets de données. Ainsi, l'application d'une architecture orientée services vous aide à maximiser l'utilisation des ressources dans un environnement distribué. Vous avez la possibilité de déployer et de migrer un écosystème de services sur les noeuds adéquats de l'environnement distribué pour répondre efficacement aux changements survenus dans l'environnement interne et externe de votre entreprise.

Informatique à la demande
Le présent document de présentation technique n'a pas pour objet de débattre en profondeur de l'informatique à la demande. Toutefois, l'architecture orientée services peut constituer une condition préalable à l'informatique à la demande. Il s'agit d'une architecture dynamisante pour applications à la demande. Les applications doivent donc fonctionner au sein d'une architecture orientée services pour que vous puissiez tirer profit des avantages de l'informatique à la demande. Les services Web constituent une technologie dynamisante pour architectures orientées services.

En tant que sous-ensemble de l'informatique à la demande, les services Web à la demande représentent simplement des services d'entreprise exposés lors de l'utilisation de normes de services Web. L'informatique à la demande peut couvrir une gamme étendue. Une partie de cette gamme se concentre sur l'environnement d'applications, une autre partie sur l'environnement d'exploitation qui comprend des éléments tels que l'infrastructure et le calcul autonome. La transformation de votre entreprise suppose l'exploitation des environnements d'applications et d'exploitation en vue de créer une entreprise à la demande. Au cœur de votre entreprise à la demande se trouvent les services à la demande où les services au niveau des applications peuvent être découverts, reconfigurés, assemblés et livrés à la demande, avec des fonctions d'intégration juste à temps.

Ce qu'offrent les services Web, en tant que technologie dynamisante, c'est une augmentation de la valeur de votre entreprise grâce à des possibilités telles que les services à la demande et, à terme, une transformation de la méthode de développement logiciel des organisations informatiques. Ils pourraient même transformer la méthode de gestion de l'entreprise et la manière dont vous proposez vos produits et services sur le Web. Que se passerait-il si toutes vos applications partageaient le même protocole de transport ? Si elles utilisaient toutes la même interface ? Si elles pouvaient participer et utiliser le même modèle transactionnel ? Que se passerait-il si cela se produisait également chez vos partenaires ? Vous disposeriez alors d'applications et d'une infrastructure pour prendre en charge un environnement commercial en constante évolution et vous deviendriez une entreprise à la demande. Les services Web et l'architecture orientée services peuvent réaliser de telles transformations au niveau des applications.

Résumé
L'architecture orientée services constitue la prochaine vague de développement d'applications. Les services Web et les architectures orientées services sont sur le point de concevoir et de créer des systèmes à l'aide de composants logiciels hétérogènes et adressables par le réseau. L'architecture orientée services comprend des propriétés spécifiques, des composants et des interconnexions qui mettent l'accent sur l'interfonctionnement et la transparence d'emplacement. Elle peut souvent évoluer à partir de systèmes existants et ne requiert donc pas de réécriture système complète. Elle tire avantage des systèmes existants de votre organisation en profitant des ressources actuelles, y compris des développeurs, des langages logiciels, des plates-formes matérielles, des bases de données et des applications, et permet de réduire les coûts et les risques tout en stimulant la productivité. Cette architecture adaptable et flexible est la base d'une réduction du temps de mise sur le marché, ainsi que des coûts et des risques de développement et de maintenance. Les services Web constituent un ensemble de technologies dynamisantes pour les architectures orientées services qui deviennent l'architecture de choix pour le développement de nouvelles applications réactives et adaptatives.

Pour plus d'informations
Pour en savoir plus sur les architectures orientées services et sur les propositions d'IBM pour la création d'une telle architecture dans votre entreprise, veuillez consulter les sites suivants :

ibm.com/software/info/openenvironment/soa/

ou

ibm.com/services

Compagnie IBM France
Tour Descartes - La Défense 5
2, avenue Gambetta
92066 - Paris-La Défense Cedex

IBM Belgium
Avenue du Bourget/Bourgetlaan, 42
B - 1130 Brussels

La page d'accueil d'IBM peut être consultée à l'adresse suivante
ibm.com

Produit aux USA 04-04

[1] Tom Dwyer, Using Composite Applications to Lower Integration Costs, Aberdeen Group, (April 2003).

[2] Jade fournit l'infrastructure d'applications de base pour une application JSP/Servlet. Cette infrastructure consiste en un modèle de programmation basé sur des centaines d'engagements et de bonnes pratiques d'IBM, ainsi que sur un ensemble d'utilitaires Java et de pratiques éprouvées de création d'applications Web.

e business on demand

G224-7298-00-FR