IBM Unica Marketing Operations
Version 8.6
25 mai 2012

Module d'intégration

Important

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à la section «Remarques», à la page 33.

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. Les informations qui y sont fournies sont susceptibles d'être modifiées avant que les produits décrits ne deviennent eux-mêmes disponibles. En outre, il peut contenir des informations ou des références concernant certains produits, logiciels ou services non annoncés dans ce pays. Cela ne signifie cependant pas qu'ils y seront annoncés.

Pour plus de détails, pour toute demande d'ordre technique, ou pour obtenir des exemplaires de documents IBM, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial.

Vous pouvez également consulter les serveurs Internet suivants :

- http://www.fr.ibm.com (serveur IBM en France)
- http://www.can.ibm.com (serveur IBM au Canada)
- http://www.ibm.com (serveur IBM aux Etats-Unis)

Compagnie IBM France
Direction Qualité
17, avenue de l'Europe
92275 Bois-Colombes Cedex

Table des matières

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.

OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
(Pos1)		Home
Fin	Fin	End
(PgAr)		PgUp
(PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
(Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

Chapitre 1. Présentation de Marketing Operations Integration Services

Marketing Operations Integration Services se compose des éléments suivants :

- **Service Web Marketing Operations Integration**

  La fonction Integration Services permet aux clients, aux partenaires et aux services IBM® Professional Services d'IBM Unica Marketing Operations d'intégrer Marketing Operations à d'autres applications s'exécutant dans leur environnement.
- **Procédures et API Marketing Operations**

  Vous pouvez définir des procédures personnalisées dans Marketing Operations pour étendre la logique métier Marketing Operations de façon arbitraire. Une fois définies, ces procédures peuvent être les cibles des appels de service Web Integration Services provenant d'autres applications. Il est également possible de définir des procédures pour envoyer des messages à d'autres applications.
- **Déclencheurs Marketing Operations**

  Les déclencheurs peuvent être associés à des événement et à des procédures dans Marketing Operations. Lorsque ce type d'événement se produit, le déclencheur associé est exécuté.

Conditions requises pour Marketing Operations Integration Services

La fonction Marketing Operations Integration Services doit :

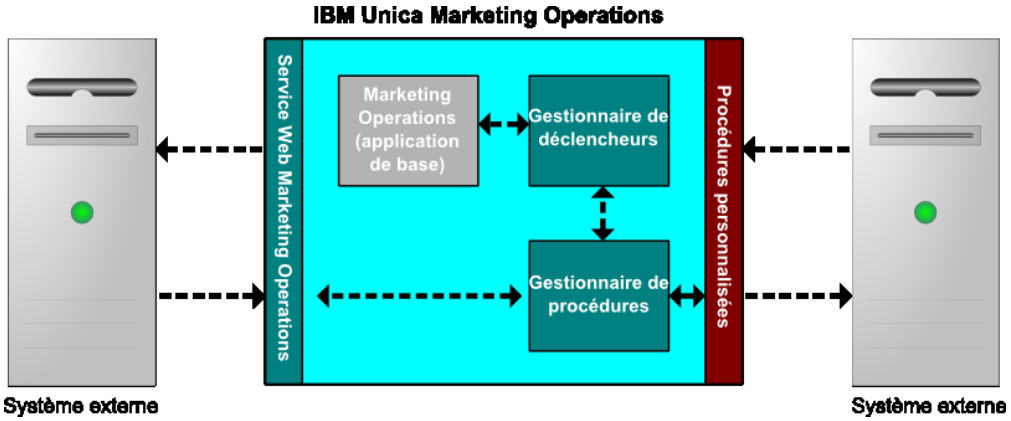
- Coupler de façon souple l'intégration du système
- Fournir un mécanisme permettant aux applications du client d'affecter Marketing Operations via des appels de service Web
- Fournir un mécanisme permettant aux applications du client d'être averties de certains événements dans Marketing Operations
- Fournir un modèle de programmation simple, facile à comprendre et à utiliser
- Etre robuste en cas de reprise sur incident
- Garantir l'intégrité des données
- S'intégrer aux clients Marketing Operations basés sur l'interface graphique existants et réduire les effets sur ces derniers
- Fournir un accès à granularité fine aux composants Marketing Operations tout en isolant les programmateurs des détails d'implémentation sous-jacents

Initiation à IBM Unica Marketing Operations Integration Services

La fonctionnalité IBM Unica Marketing Operations Integration Services permet de créer des procédures personnalisées. Vous pouvez utiliser ces procédures pour déclencher des événements externes lorsque certains événements se produisent dans Marketing Operations. Vous pouvez également utiliser ces procédures pour exécuter des fonctions Marketing Operations à partir de systèmes ou de programmes externes.

Vous utilisez l'API comme interface avec IBM Unica Marketing Operations au niveau du programme, de la même façon que vous utilisez l'interface graphique comme interface avec Marketing Operations au niveau utilisateur. L'API vous permet de construire des procédures. Ces dernières vous permettent d'établir une communication entre Marketing Operations et les systèmes externes. Le service Web Marketing Operations est l'objet conteneur de ces procédures, de l'API et des déclencheurs.

L'architecture de Marketing Operations Integration Services vous est présentée ici.

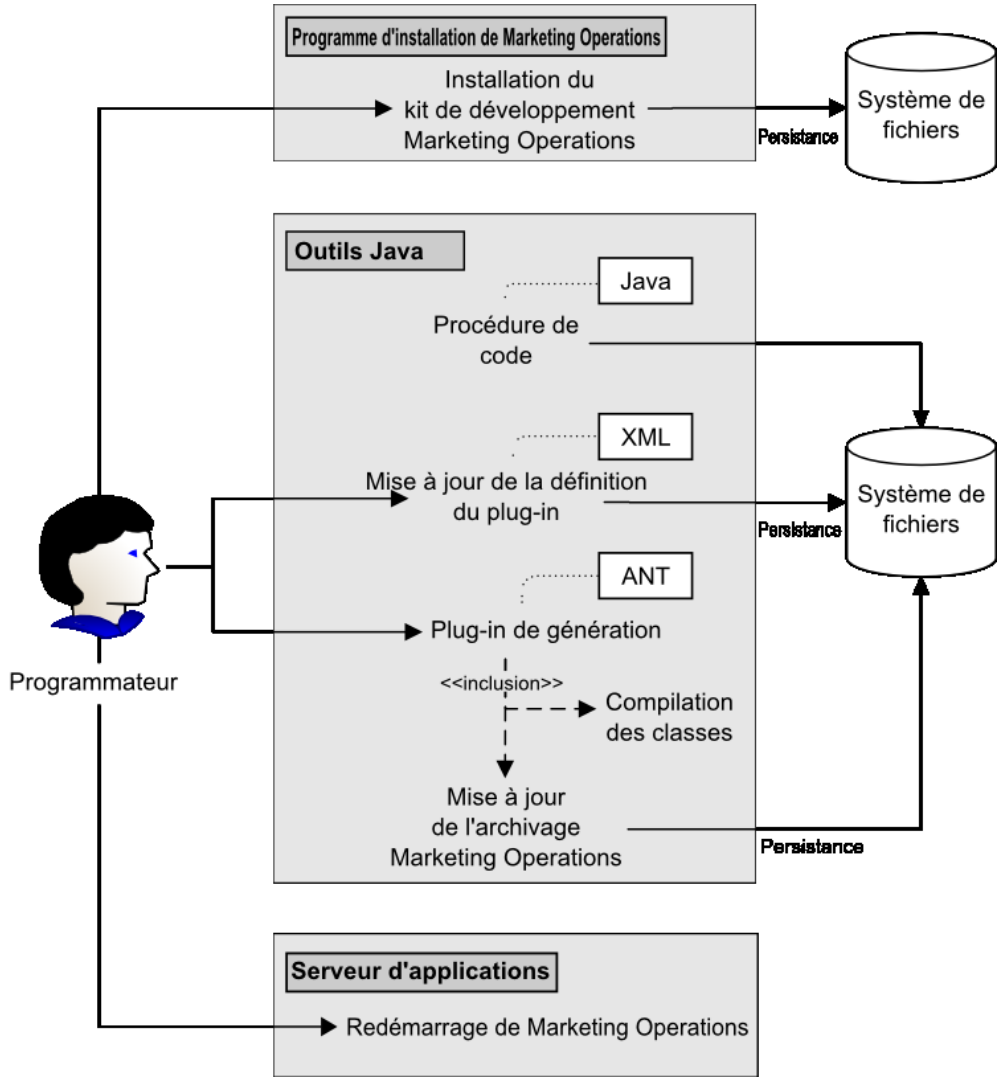


Voici les principaux composants des services Integration Services :

- Marketing Operations Procedure Manager : étend la logique métier en interagissant avec Marketing Operations via l'API.
- Marketing Operations Trigger Manager : associe une condition (par exemple, le changement d'état d'un projet marketing) à une action (une procédure à exécuter lorsque la condition associée au déclencheur est remplie).

Méthodologie

Vous utilisez les composants d'IBM Unica Marketing Operations Integration Services pour développer des procédures personnalisées, comme l'indique ce diagramme :



Après avoir utilisé le programme d'installation de Marketing Operations pour installer le kit de développement, vous effectuez les étapes de base suivantes :

1. Codification de la procédure personnalisée. Actuellement, vous devez utiliser Java.
2. Mise à jour de la définition du plug-in dans le fichier de définition XML.
3. Génération du plug-in :
   a. Compilation des classes nécessaires.
   b. Mise à jour de l'archive Marketing Operations (fichier WAR).
4. Redémarrage de Marketing Operations.

Exemple de base de communication entre IBM Unica Marketing Operations et l'API

Cette section présente un exemple de base d'établissement d'une communication entre l'API et Marketing Operations. Il ne s'agit pas d'un travail utile mais d'un aller-retour entre Marketing Operations et Integration Services.

Cette section utilise des portions des exemples de procédures fournis avec le kit de développement de Marketing Operations Integration Services. Vous pouvez trouver le code référencé ici dans les fichiers suivants :

- PlanClientFacade.java
- PlanWSNOOPTestCase.java

La méthode `noop` est un appel de service Web en direction de Marketing Operations. Elle est définie dans la classe PlanClientFacade et transmet des valeurs nulles dans une matrice.

```
public ProcedureResponse noop(String jobId)
  throws RemoteException, ServiceException {
  NameValueArrays parameters =
    new NameValueArrays(null, null, null, null, null, null, null, null);
  return _serviceBinding.executeProcedure("uapNOOPProcedure", jobId, parameters);
}
```

La procédure testExecuteProcedure appelle la méthode `noop` à partir de PlanClientFacade pour établir un aller-retour avec l'application Marketing Operations.

```
public void testExecuteProcedure() throws Exception {
  // Time out after a minute
  int timeout = 60000;
  PlanClientFacade clientFacade = new PlanClientFacade(urlWebService, timeout);
  System.out.println("noop w/no parameters");
  long startTime = new Date().getTime();
  ProcedureResponse response = clientFacade.noop("junit-jobid");
  long duration = new Date().getTime() - startTime;

  // zero or positive status => success
  System.out.println("Status: " + response.getStatus());
  System.out.println("Duration: " + duration + " ms");
  assertTrue(response.getStatus() >= 0);
  System.out.println("Done.");
}
```

Pour plus d'informations sur NameValueArrays, ProcedureResponse et les autres méthodes et types de données listés, reportez-vous aux sections spécifiques de ce guide et aux fichiers JavaDoc.

Fichiers JavaDoc hébergés

Pour obtenir des informations spécifiques sur les méthodes API publiques, reportez-vus à la classe iPlanAPI dans les fichiers de documentation API JavaDoc. Pour accéder à ces fichiers, connectez-vous à Marketing Operations et sélectionnez **Aide > Documentation produit** dans n'importe quelle page, puis téléchargez le fichier `<version>PublicAPI.zip`.

Chapitre 2. A propos du service Web Marketing Operations Integration

Le service Web fournit une vue client de Marketing Operations Integration Services, qui fait partie du déploiement du serveur IBM Unica Marketing Operations. Le service est conçu pour être utilisé en même temps que les utilisateurs Web de Marketing Operations.

Le service Web prend en charge un appel API, executeProcedure.

C'est un client qui effectue directement cet appel de service Web.

Authentification

L'authentification n'est pas requise ; tous les clients sont associés à un utilisateur IBM Unica Marketing Operations connu appelé PlanAPIUser. On suppose que les fonctions de sécurité de cet utilisateur spécial sont configurées par un administrateur système Marketing Operations en fonction des besoins de tous les clients du service Web.

Une future version du service pourrait fournir un mécanisme plus général de sécurisation de l'authentification client.

Langue

La seule langue prise en charge est la langue actuellement configurée pour l'instance de système IBM Unica Marketing Operations. Toutes les données dépendantes de la langue accessibles via le service (messages, devises, etc.) sont supposées être dans la langue du système.

Une future version du service pourrait fournir un mécanisme permettant au client d'indiquer à Marketing Operations la langue à utiliser.

Gestion d'état

Le service Web est *sans état* ; aucune information par client n'est sauvegardée par l'implémentation du service au fil des appels API. Cette fonction permet de bénéficier d'une implémentation de service plus efficace et simplifie la prise en charge du cluster.

Transactions de base de données

Le service Web n'expose pas les transactions de base de données et les verrous d'édition au client. Toutefois, il garantit l'effet *atomique* de toute exécution de procédure. Cela signifie que la procédure aboutit ou échoue ; un échec laisse la base de données dans le même état que si aucun appel API n'avait été émis.

A propos des types de données du service Web Marketing Operations Integration

Cette section définit les types de données utilisés par le service Web, indépendamment d'une liaison de service ou d'une implémentation de programme particulière.

La notation suivante est utilisée :

- <*type*> : <*type definition*> définit un type de données simple. Par exemple : Handle: string
- <*type*>: [<*type definition*>] définit un type de données complexe ou une structure de données.
- <*type*>: { <*type definition*> } définit un type de données complexe ou une structure de données.

Les éléments de type complexe et les paramètres API peuvent utiliser ces types pour déclarer des matrices. Par exemple :

```
Handle [] handles
```

Le type, handles, est une matrice de types Handle.

Types primitifs

Les types primitifs sont limités aux types définis dans la table qui suit pour simplifier la prise en charge des liaisons SOAP 1.1. Tous les types peuvent être déclarés sous forme de matrices, par exemple, **String [ ]**. Fondamentalement, les types de données binaires tels que **long** peuvent être représentés sous forme de chaînes par une liaison de protocole (par exemple, SOAP). Cependant, cette représentation n'a aucun effet sur la sémantique du type, les valeurs admises, etc., tels qu'ils sont vus par le client.

Tableau 1. Types primitifs

Type API	Description	Type SOAP	Type Java
booléen	Valeur booléenne : **true** ou **false**	xsd:boolean	booléen
dateHeure	Valeur de date/heure	xsd:datetime	Date
décimal	Valeur décimale à précision arbitraire	xsd:decimal	java.math.BigDecimal
double	Valeur décimale signée à double précision	xsd:double	double
int	Valeur de type entier signée 32 bits	xsd:int	int
entier	Valeur de type entier signée à précision arbitraire	xsd:integer	java.math.BigInteger
long	Valeur de type entier signée 64 bits	xsd:long	long
string (chaîne)	Chaîne de caractères Unicode	xsd:string	java.lang.String

MessageTypeEnum

```
MessageTypeEnum: { INFORMATION, WARNING, ERROR }
```

MessageTypeEnum est un type énuméré qui définit tous les types de message possibles.

- INFORMATION : message d'information
- WARNING : message d'avertissement
- ERROR : message d'erreur

Message

```
Message: [MessageTypeEnum type, string code, string localizedText, string logDetail]
```

Message est une structure de données qui définit le résultat d'un appel API de service Web. Il fournit des zones facultatives pour le code non localisé, le texte localisé et le détail du journal. Actuellement, tous les textes localisés utilisent la langue définie pour l'instance de serveur IBM Unica Marketing Operations.

Tableau 2. Paramètres du message

Paramètre	Description
type	MessageTypeEnum définissant le type du message.
code	Code facultatif au format chaîne (string) pour le message.
localizedText	Chaîne de texte facultative à associer au message.
logDetail	Message de trace de pile facultatif.

NameValue

```
NameValue: [string name, int sequence]
```

NameValue est un type complexe de base qui définit une paire nom-valeur. Il définit également une séquence facultative utilisée par le service pour construire les matrices de valeur nécessaires (les séquences sont de base zéro).

Tous les NameValues portant le même nom mais possédant des numéros de séquence différents sont convertis dans une matrice de valeurs et associés au nom commun.

La taille de la matrice dépend du numéro de séquence maximal ; les éléments de matrice non spécifiés possèdent la valeur NULL. Les numéros de séquence de matrice doivent être uniques. La valeur et son type sont fournis par le type étendu.

Tableau 3. Paramètres de NameValue

Paramètre	Description
name	Chaîne qui définit le nom d'un type NameValue.
sequence	Entier de base zéro qui définit le numéro de séquence de la valeur NameValue concernée.

Les types NameValue étendus sont définis pour chaque type primitif, comme suit :

Tableau 4. Types NameValue étendus

Type étendu	Description
BigDecimalNameValue: NameValue [decimal value]	Type NameValue dont la valeur est un nombre décimal à précision arbitraire.

Tableau 4. Types NameValue étendus (suite)

Type étendu	Description
BigIntegerNameValue: NameValue [integer value]	Type NameValue dont la valeur est un entier signée de façon arbitraire.
BooleanNameValue: NameValue [boolean value]	Type NameValue dont la valeur est un booléen.
CurrencyNameValue: NameValue [string locale, decimal value]	Type NameValue convenant pour représenter les devises dans un langue spécifique. La langue est représentée par un code de langue ISO, c'est-à-dire un code à deux lettres en minuscules, tel que défini par la norme ISO-639. Actuellement, la langue doit correspondre à la langue définie dans l'instance de serveur IBM Unica Marketing Operations.
DateNameValue: NameValue [datetime value]	Type NameValue dont la valeur est une date.
DecimalNameValue: NameValue [double value]	Type NameValue dont la valeur est un nombre décimal à double précision.
IntegerNameValue: NameValue [long value]	Type NameValue dont la valeur est un entier de 64 bits.
String NameValue: NameValue [string value]	Type NameValue dont la valeur est une chaîne.

Une matrice des types NameValue étendus est définie afin d'être utilisée lorsque vous avez besoin de définir un jeu de NameValues de différents types.

```
     NameValueArrays: [
BooleanNameValue[]    booleanValues,
StringNameValue[]     stringValues,
IntegerNameValue[]    integerValues,
BigIntegerNameValue[] bigIntegooleanNameValue,
DecimalNameValue[]    decimalValues,
BigDecimalNameValue[] bigDecimalValues
DateNameValue[]       dateNameValues
CurrencyNameValue[]   currencyValues
     ]
```

executeProcedure

Syntaxe

```
executeProcedure(string key, string jobid, NameValueArrays paramArray)
```

Retour

```
int: status
Message[]: messages
```

Description

Cette méthode appelle la procédure spécifiée avec un ensemble de paramètres facultatifs. L'appel s'exécute de façon synchrone, c'est-à-dire qu'il bloque le client et renvoie le résultat à la fin de l'exécution.

Paramètres

Tableau 5. Paramètres executeProcedure

Nom	Description
key	Clé unique de la procédure à exécuter. Une erreur *RemoteException* est renvoyée si aucune procédure n'est liée à **key**.
jobid	Chaîne facultative qui identifie le travail associé à l'exécution de la procédure. Cette chaîne est un élément passe-système mais elle peut être utilisée pour lier des travaux client à l'exécution d'une procédure particulière.
paramArray	Ensemble de paramètres à transmettre à la procédure. Un état et un message d'erreur sont renvoyés si un ou plusieurs des paramètres sont non valides (type non valide, valeur non autorisée, etc.). C'est au client qu'il revient de déterminer les paramètres, leur type et le nombre de valeurs requises par la procédure.

Paramètres de retour

Tableau 6. Paramètres de retour de executeProcedure

Nom	Description
status (état)	Code entier : • 0 indique que l'exécution de la procédure a abouti • un entier indique une erreur Les procédures peuvent utiliser l'état pour indiquer différents niveaux d'erreur.
messages	Ensemble de zéro ou plusieurs structures de données de message. Si **status** a pour valeur 0, cet ensemble ne contient pas de messages d'ERREUR mais peut contenir des messages d'INFORMATION et d'AVERTISSEMENT. Si **status** est différent de zéro, les messages peuvent contenir un mélange de messages d'ERREUR, d'INFORMATION et d'AVERTISSEMENT.

Langage WSDL associé à Marketing Operations Integration Services

Cette rubrique définit le langage WSDL (Web Services Definition Language) associé à Marketing Operations Integration Services. La langage WSDL a été défini manuellement et constitue le point final de la définition du service Web.

Axis

Cette version du service Web utilise Axis2 1.5.2 pour générer les classes côté serveur qui constituent l'implémentation du service Web à partir du fichier WSDL. Les utilisateurs peuvent employer n'importe quelle version de Axis ou une technique autre que Axis, pour créer une implémentation côté client permettant une intégration avec l'API à partir du WSDL fourni.

Version du protocole

La version du protocole est explicitement liée au WSDL :

- Dans le nom WSDL, par exemple, `PlanIntegrationService1.0.wsdl`

- Dans le targetNamespace WSDL, par exemple, `xmlns:tns="http://webservices.unica.com /MktOps/services/PlanIntegrationServices1.0?wsdl"`

WSDL

Un fichier WSDL est fourni avec IBM Unica Marketing Operations Integration Services : `PlanIntegrationServices1.0.wsdl`. Ce fichier WSDL est situé dans le répertoire `integration/examples/soap/plan`. L'exemple de script de génération utilise ce fichier pour générer les modules de remplacement côté client appropriés à connecter au service Web.

Chapitre 3. Procédures IBM Unica Marketing Operations

Une *procédure* est une classe Java personnalisée ou standard, hébergée par IBM Unica Marketing Operations, qui exécute une unité de travail. Les procédures permettent aux clients, aux partenaires et aux services IBM Unica Professional Services d'étendre la logique métier Marketing Operations de façon arbitraire.

Les procédures suivent un modèle de programmation simple, en utilisant une API bien définie pour affecter des composants gérés par Marketing Operations. La reconnaissance des procédures s'effectue via un mécanisme de recherche simple et un fichier de définition XML. Marketing Operations exécute les procédures en fonction des besoins de ses "clients" (par exemple, en réponse à une demande d'intégration (entrante) ou à l'action d'un déclencheur (interne ou sortant)).

Les procédures s'exécutent de façon synchronisée par rapport à leur client ; les résultats sont directement mis à la disposition du client via un mécanisme d'audit persistant. L'exécution d'une procédure peut également provoquer d'autres événements et déclencheurs dans Marketing Operations.

Les procédures doivent être écrites en Java.

Hypothèses

Notez les hypothèses suivantes concernant les procédures.

- Les classes d'implémentation de procédure sont regroupées dans une arborescence de classes différente ou dans un autre fichier jar et sont mises à la disposition d'IBM Unica Marketing Operations via un chemin URL. Le gestionnaire d'exécution de procédure utilise un chargeur de classe indépendant pour charger ces classes en fonction des besoins. Par défaut, Marketing Operations recherche dans le répertoire suivant :

  `<Marketing Operations_home>/devkits/integration/examples/classes`

  Pour modifier ce paramètre par défaut, définissez le paramètre **integrationProcedureClasspathURL** sous **Paramètres > Configuration > Marketing Operations > umoConfiguration > integrationServices**.
- Le nom de la classe d'implémentation de procédure obéit aux conventions de dénomination Java acceptées, afin d'éviter des collisions de package avec “unica” et avec les classes des autres fournisseurs. Les clients ne doivent pas placer des procédures dans l'arborescence de packages **com.unica** ou **com.unicacorp**.
- L'implémentation de procédure est codée dans la version Java Runtime utilisée par IBM Unica Marketing Operations sur le serveur d'applications (au minimum JRE 1.5.10).
- IBM Unica Marketing Operations fournit un certain nombre de bibliothèques Open Source et tierces ; les serveurs d'applications utilisent également différentes versions de ces bibliothèques. En général, cette liste varie d'une version à une autre.

  Remarque : Afin d'éviter tout problème de compatibilité, les procédures ne doivent pas utiliser de bibliothèques Open Source, tierces ou propres à un serveur d'applications.

  Toutefois, si ces packages sont utilisés par une procédure ou par les classes secondaires importées par la procédure, leur utilisation doit être totalement

conformes aux packages fournis par Marketing Operations et/ou par le serveur d'applications. Notez que, dans ce cas, il peut s'avérer nécessaire de retravailler votre code de procédure si une version ultérieure de Marketing Operations met à niveau ou abandonne une bibliothèque.

- La classe d'implémentation de procédure est chargée par la règle de chargement de classe normalement utilisée par IBM Unica Marketing Operations (généralement **parent-last**). Le serveur d'applications peut fournir des outils et options de développement pour recharger les classes qui pourraient s'appliquer aux procédures Marketing Operations, mais cela n'est pas obligatoire.
- La procédure doit autoriser les unités d'exécution multiples concernant son propre état ; cela signifie que sa méthode d'exécution ne peut pas dépendre des changements d'état internes d'un appel à un autre.
- Une procédure ne peut pas créer des unités d'exécution par elle-même.

Conception

La conception doit se focaliser sur la production d'une unité de travail unique qui doit être exécutée de façon atomique. Idéalement, la procédure exécute des séries de tâches qui peuvent être planifiées de façon asynchrone pour être exécutées ultérieurement ; ce modèle d'intégration de type "lancer et oublier" permet d'obtenir une charge minimale sur chaque système.

La classe d'implémentation de procédure utilise l'API IBM Unica Marketing Operations pour lire et mettre à jour les composants Marketing Operations, les services d'appel, etc. Vous pouvez utiliser d'autres packages Java pour effectuer d'autres tâches.

Remarque : Seuls les classes et les méthodes documentées seront prises en charge dans les futures versions de Marketing Operations. Toutes les autres classes et méthodes de Marketing Operations doivent être considérées comme non autorisées.

Une fois codées et compilées, les classes d'implémentation de procédure doivent être mises à la disposition de Marketing Operations. Les scripts de génération fournis avec la fonction Marketing Operations Integration Services placent les procédures compilées dans l'emplacement par défaut. L'étape de développement final consiste à mettre à jour le fichier de définition du plug-in de procédure personnalisée utilisé par Marketing Operations pour reconnaître les procédures personnalisées.

La procédure doit implémenter l'interface **com.unica.publicapi.plan.plugin.procedure.IProcedure** et comporter un constructeur dans paramètre (modèle JavaBean habituel). La codification et la compilation de chaque procédure est effectuée dans un outil Java choisi par le client (par exemple, Eclipse, Borland JBuilder, Idea, etc.). Un exemple de code est fourni avec IBM Unica Marketing Operations sous forme de kits d'outils de développement dans l'emplacement suivant :

<Marketing Operations-home>/devkits/integration/examples/src/procedure

Configuration

Utilisez les paramètres sous **Paramètres > Configuration > Marketing Operations > umoConfiguration > integrationServices** pour configurer la fonction Marketing Operations Integration Module.

Pour plus d'informations, voir le *Guide d'installation de Marketing Operations*.

Cycle de vie de la procédure

Le cycle de vie d'exécution d'une procédure est le suivant :

1. Reconnaissance et initialisation
2. Sélection (facultatif)
3. Exécution
4. Destruction

Reconnaissance et initialisation

IBM Unica Marketing Operations doit être informé de toutes les procédures personnalisées et standard disponibles pour une instance d'installation particulière. Ce processus s'appelle la reconnaissance.

Remarque : Les procédures standard (procédures définies par l'équipe d'ingénierie Marketing Operations) sont connues implicitement et ne nécessitent donc pas de reconnaissance.

Les procédures personnalisées sont définies dans le fichier de définition du plug-in de procédure. Le gestionnaire du plug-in Marketing Operations lit ce fichier lors de l'initialisation. Pour chaque procédure détectée, le gestionnaire de plug-in effectue les tâches suivantes :

1. Instanciation de la procédure ; passage de son état à INSTANCIEE.
2. Création d'un nouvel enregistrement d'audit de procédure.
3. Si la procédure a pu être instanciée, sa méthode **`initialize()`** est appelée avec tout paramètre d'initialisation trouvé dans son fichier de description de plug-in. Si cette méthode émet une exception, le statut est consigné et la procédure est abandonnée. Dans le cas contraire, la procédure passe à l'état INITIALISEE. Elle est alors prête à être exécutée.
4. Création d'un nouvel enregistrement d'audit de procédure.
5. Si la procédure a pu être initialisée, sa méthode **`getKey()`** est appelée pour identifier la clé utilisée par les clients pour référencer la procédure. Cette clé est associée à l'instance et sauvegardée pour une recherche ultérieure.

Sélection

De temps en temps, il peut arriver que IBM Unica Marketing Operations présente une liste des procédures disponibles aux utilisateurs, par exemple, pour permettre aux administrateurs de définir un déclencheur. Cela ne peut se produire qu'une fois la procédure initialisée. Ce sont les méthodes **`getDisplayName()`** et **`getDescription()`** de la procédure qui sont utilisées à cet effet.

Exécution

Une fois la procédure initialisée, IBM Unica Marketing Operations reçoit une demande d'exécution de la procédure. Cela peut se produire en même temps que pour d'autres procédures (ou pour la même) s'exécutant sur d'autres unités d'exécution.

Au moment de l'exécution, le gestionnaire d'exécution de procédure effectue les tâches suivantes :

1. Démarrage d'une transaction de base de données.
2. Définition de l'état de la procédure sur EN COURS D'EXECUTION.
3. Création d'un nouvel enregistrement d'audit de procédure.
4. Appel de la méthode **execute()** de la procédure avec un contexte d'exécution et tout paramètre d'exécution fourni par le client. L'implémentation de la méthode utilise l'API Marketing Operations si nécessaire, en acquérant les verrous d'édition et en transmettant le contexte d'exécution. Si la méthode d'exécution émet une exception, le gestionnaire d'exécution marque la transaction comme devant être annulée.
5. Validation ou annulation de la transaction en fonction des résultats de l'exécution ; définition de l'état de la procédure sur EXECUTEE.
6. Libération de tout verrou d'édition en suspens.
7. Création d'un nouvel enregistrement d'audit de procédure.

Remarque : La méthode **execute()** ne doit pas modifier les données d'instance de la procédure.

Destruction

A l'arrêt d'IBM Unica Marketing Operations, le gestionnaire du plug-in de procédure passe en revue toutes les procédures chargées. Pour chaque procédure détectée, il effectue les tâches suivantes :

1. Appel de la méthode destroy() de la procédure afin de permettre à cette dernière d'effectuer un nettoyage avant la destruction de l'instance.
2. Passage de l'état de la procédure à FINALISEE (elle ne peut pas être exécutée).
3. Création d'un nouvel enregistrement d'audit de procédure.
4. Destruction de l'instance de la procédure.

Verrouillage des données

IBM Unica Marketing Operations utilise un schéma de verrouillage d'édition pessimiste, c'est-à-dire qu'à un moment donnée, un seul utilisateur possède des droits de mise à jour sur le instances du composant. Pour l'utilisateur de l'interface graphique, ce verrouillage est effectué au niveau visuel de l'onglet. Dans certains cas, cela représente un sous-ensemble d'instance (par exemple, un onglet de récapitulatif de projet) et dans d'autres cas, cela représente plusieurs instances (l'onglet de flux de travaux). Une fois qu'un utilisateur a acquis un verrou, tous les autres utilisateurs ne possèdent plus qu'un accès en lecture seule aux données concernées.

Remarque : Les verrous d'édition sont différents des transactions de base de données.

Afin d'éviter que les modifications apportées par une procédure à une instance de composant ou à un groupe d'instances ne soient écrasées par inadvertance par un autre utilisateur, une procédure doit acquérir les verrous appropriés avant la mise à jour des données du composant. C'est l'objet contexte d'exécution transmis à la méthode **execute()** de la procédure qui est utilisé pour accomplir cette tâche.

Avant de mettre à jour des données, la procédure doit appeler la méthode **acquireLock()** du contexte pour chaque verrou dont elle a besoins. Par exemple, si une procédure doit mettre à jour un projet et le flux de travaux associé, elle doit acquérir des verrous pour ces deux éléments.

Si un autre utilisateur possède déjà un verrou, la méthode **acquireLock()** émet immédiatement une exception **LockInUseException**. Afin de réduire le nombre de collisions, la procédure doit libérer le verrou dès qu'elle met à jour l'objet.

Le gestionnaire d'exécution libère automatiquement tout verrou en suspens lorsque la méthode d'exécution revient. Dans tous les cas, les verrous ne sont détenus qu'à concurrence de la durée de vie de la transaction de base de données. Les verrous expirent si le délai de la transaction propre à la base de données a été dépassé.

Transactions de procédure

Le gestionnaire d'exécution de procédure effectue automatiquement en boucle l'exécution de la procédure avec une transaction de base de données, en la validant ou en l'annulant en fonction du résultat de l'exécution de la procédure. Cela permet d'être sûr que les mises à jour de la base de données IBM Unica Marketing Operations ne sont pas visibles des autres utilisateurs tant qu'elles ne sont pas validées et que les mises à jour sont atomiques.

L'auteur de la procédure doit cependant acquérir les verrous d'édition nécessaires afin d'être sûr que les autres utilisateurs ne puissent pas copier des modifications dans la base de données avant la fin de l'exécution de la procédure.

Communication des résultats

La méthode **execute()** d'une procédure renvoie un code d'état d'entier et zéro ou plusieurs messages qui sont consignés et conservés dans la table d'audit de procédure IBM Unica Marketing Operations. Le client peut également communiquer les informations d'état d'une autre manière.

Consignation des procédures

IBM Unica Marketing Operations possède un fichier journal distinct pour les procédures.

`<Marketing Operations-home>\logs\procedure.log`

Le gestionnaire d'exécution de procédure consigne le cycle de vie de chaque procédure et crée des enregistrements d'audit.

- **logInfo()** : écrit un message d'information dans le journal des procédures
- **logWarning()** : écrit un message d'avertissement dans le journal des procédures
- **logError()** : écrit un message d'erreur dans le journal des procédures
- **logException()** : vide la trace de pile pour l'exception dans le journal des procédures

Principales classes Java

Le kit de développement d'intégration fourni contient une série de fichiers Javadoc pour l'API IBM Unica Marketing Operations publique et pour les classes de support. Les plus importants sont répertoriés ci-après :

- IProcedure (com.unica.publicapi.plan.plugin.procedure.IProcedure) : interface que toutes les procédures doivent implémenter. Les procédures ont un cycle de vie bien défini et accèdent à l'API Marketing Operations pour effectuer un travail.
- ITriggerProcedure (com.unica.publicapi.plan.plugin.procedure.ITriggerProcedure) : interface que toutes les procédures de déclencheur doivent implémenter (interface de marqueur).
- IExecutionContext (com.unica.publicapi.plan.plugin.procedure.IExecutionContext) : interface d'objet contextuel opaque transmis à la procédure par le gestionnaire d'exécution. Cet objet comporte des méthodes publiques pour la consignation et la gestion des verrous d'édition. La procédure transmet également cet objet à tous les appels PlanAPI.
- IPlanAPI (com.unica.publicapi.plan.api.IPlanAPI) : interface vers l'API Marketing Operations. Le contexte d'exécution fournit une méthode **getPlanAPI()** qui extrait l'implémentation appropriée.

Exemple de procédure

Cet exemple présente une procédure standard permettant de modifier l'état d'un projet à partir d'un service Web d'intégration ou d'un déclencheur.

Remarque : Ne modifiez pas les exemples de procédure et les définitions XML associées car ces exemples sont modifiés lors de la mise à niveau d'IBM Unica Marketing Operations. Vous perdriez donc toutes vos modifications. Vous devez créer et modifier toutes les procédures personnalisées dans un autre répertoire.

```
// ProjectStateChangeProcedure
// (c) Copyright 2006 by Unica Corporation. All rights reserved.

package com.unica.uap.plugin.procedure.standard;

import java.util.Iterator;
import java.util.List;
import java.util.Locale;
import java.util.Map;

import com.unica.publicapi.plan.api.Handle;
import com.unica.publicapi.plan.api.IExecutionContext;
import com.unica.publicapi.plan.api.IPlanAPI;
import com.unica.publicapi.plan.api.LockInUseException;
import com.unica.publicapi.plan.api.ProjectHandle;
import com.unica.publicapi.plan.api.ProjectStateEnum;
import com.unica.publicapi.plan.plugin.PluginVersion;
import com.unica.publicapi.plan.plugin.procedure.IProcedure;
import com.unica.publicapi.plan.plugin.procedure.ProcedureExecutionException;
import com.unica.publicapi.plan.plugin.procedure.ProcedureInitializationException;
import com.unica.publicapi.plan.plugin.procedure.ProcedureMessage;
import com.unica.publicapi.plan.plugin.procedure.ProcedureMessageTypeEnum;
import com.unica.publicapi.plan.plugin.procedure.ProcedureResult;

/**
 * <b>ProjectStateChangeProcedure</b> est une procédure standard Marketing Operations
 * qui tente de modifier
 * l'état d'un projet.
```

```
 * <p>
 * Les paramètres d'initialisation attendus sont les suivants :
 * <ul>
 *   <li>debug : objet booléen, <tt>true</tt> ou <tt>false</tt>, indiquant
 *   si la trace de débogage est activée ou pas</li>
 * </ul>
 *
 * <p>
 * Les paramètres d'exécution attendus sont les suivants :
 * <ul>
 *   <li>hProject : formulaire de tableau de chaînes du descripteur du projet. ex :
 *     "http://mymachine:7001/MktOps/affiniumplan.jsp?
 *         cat=projecttabs&projectid=12"</li>
 *   <li>uapState : formulaire de tableau de chaînes de l'état du nouveau projet. ex :
 *      "COMPLETED". La casse est prise en compte !</li>
 * </ul>
 *
 */
public final class ProjectStateChangeProcedure implements IProcedure {
// paramètres d'initialisation
private final static String DEBUG_INITPARAMETER_NAME = "debug";
  // paramètres d'exécution
  private final static String HPROJECT_PARAMETER_NAME = "hProject";
  private final static String STATE_PARAMETER_NAME
    = IPlanAPI.PROJECT_ATTRIBUTE_STATEENUM; // identique au nom d'attribut

  // nos codes d'état
  private final static int STATUS_SUCCESS = 0;

    // Propriété debug. Définit le paramètre d'initialisation "debug"
  //   de la procédure sur true pour activer la trace de débogage
    private boolean _debug = false;
    private boolean isDebug() { return _debug; }

  // Le nom simple est un nom de classe non qualifié
  public String getName() {
        return "uapProjectStateChangeProcedure";
    }

  // Le nom d'affichage est toujours
  public String getDisplayName(Locale locale) {
        // uniquement en anglais pour l'instant
        return getName();
    }

  // description toujours en anglais
    public String getDescription(Locale locale) {
        // uniquement en anglais pour l'instant
          return "A procedure to transition the state of a project.";
    }

  // version codée également : 1.0.0 pour l'instant
  public PluginVersion getVersion() {
    return new PluginVersion(1,0,0);
  }

  // initialisation de l'instance à partir des paramètres d'initialisation
    public void initialize(Map initParameters)
      throws ProcedureInitializationException {
    // Le seul paramètre d'initialisation dont nous disposons est : debug, Boolean
        if (initParameters.containsKey(DEBUG_INITPARAMETER_NAME)) {
            try {
                _debug = ((Boolean)initParameters.get(DEBUG_INITPARAMETER_NAME)).
                  booleanValue();
            } catch (Exception exception) {
                throw new ProcedureInitializationException("Problem using "
                                  + DEBUG_INITPARAMETER_NAME
```

```
                                        + " init parameter: "
                                        + exception.getMessage());
            }
            }
        }

      // execute : hProject et state enum attendus
        public ProcedureResult execute(IExecutionContext context, Map parameters)
          throws ProcedureExecutionException {
            // extraction des paramètres d'exécution : deux sont attendus :
            // - hProject: string[] pour le descripteur de projet
            // - uapState: string[] pour ProjectStateEnum
            ProjectHandle hProject = null;
        if (parameters.containsKey(HPROJECT_PARAMETER_NAME)) {
          try {
            hProject = (ProjectHandle)
    Handle.makeHandle(((String[])parameters.get(HPROJECT_PARAMETER_NAME))[0]);
          } catch (Exception exception) {
            throw new ProcedureExecutionException("Problem using "
                                + HPROJECT_PARAMETER_NAME
                                + " parameter: "
                                + exception.getMessage());
          }
        } else throw new ProcedureExecutionException(HPROJECT_PARAMETER_NAME
                                + " parameter must be provided.");

            ProjectStateEnum stateEnum = null;
        if (parameters.containsKey(STATE_PARAMETER_NAME)) {
          try {
            stateEnum =
              ProjectStateEnum.valueOf(((String[])parameters.
                get(STATE_PARAMETER_NAME))[0]);
          } catch (Exception exception) {
            throw new ProcedureExecutionException("Problem using "
                                + STATE_PARAMETER_NAME
                                + " parameter: "
                                + exception.getMessage());
          }
        } else throw new ProcedureExecutionException(STATE_PARAMETER_NAME
                                + " parameter must be provided.");

        int status = -1;
        ProcedureMessage[] messages = null;
            try {
               // tentative d'acquisition d'un verrou d'édition pour le projet
                context.acquireLock(hProject, IExecutionContext.LOCK_ALL_FIELDS);

                // utilisation de PlanAPIImpl pour mettre à jour l'état
                IPlanAPI planAPI = context.getPlanAPI();
                planAPI.updateAttribute(context, hProject, STATE_PARAMETER_NAME,
                            new ProjectStateEnum[]{stateEnum});

              // réussite
              status = STATUS_SUCCESS;

            } catch (Exception exception) {
              // écriture de la pile de trace en cas de débogage
                if (isDebug()) {
                    context.logError(getName(), exception);
                }
                throw new ProcedureExecutionException(exception);
            } finally {
              // libération du verrou
              try {
                context.releaseAllLocks();
              } catch (Exception exception) { /* ignored */ }
            }
```

```
        return new ProcedureResult(status, messages);
    }

    public void destroy( ){
         // aucune action n'est nécessaire
    }
}
```

Fichier de définition du plug-in de procédure

Cette rubrique décrit le fichier de définition du plug-in de procédure. Ce fichier définit la classe d'implémentation, les métadonnées et d'autres informations relatives aux procédures personnalisées à héberger dans IBM Unica Marketing Operations. Par défaut, la définition du plug-in de procédure est dans le répertoire suivant :

```
<Marketing Operations-home>/devkits/integration/examples/src/
  procedures/procedure-plugins.xml
```

Ce fichier est un document XML qui contient les informations présentées ci-après.

Procedures : liste de zéro ou plusieurs éléments **Procedure**.

Procedure : élément qui définit une procédure. Chaque procédure contient les éléments suivants :

- **key** (facultatif) : chaîne définissant la clé de consultation de la procédure. Cette clé doit être unique pour toutes les procédures standard (fournies par IBM) et personnalisées hébergées par une instance Marketing Operations particulière. Si elle n'est pas définie, elle prend par défaut la valeur de la version qualifiée complète de l'élément **className**. Les noms commençant par la chaîne "uap" sont réservés à IBM Unica Marketing Operations.
- **className** (obligatoire) : nom de package qualifié complet de la classe de procédure. Cette classe doit implémenter la classe IProcedure (com.unica.public.plan.plugin.procedure.IProcedure).
- **initParameters** (facultatif) : liste de zéro ou plusieurs éléments initParameter.

  initParameter (facultatif) : paramètre à transmettre à la méthode initialize() de la procédure. Cet élément inclut le nom du paramètre imbriqué, son type et les éléments de valeur.
  - name : chaîne définissant le nom du paramètre
  - type : nom de classe facultatif de la classe d'encapsuleur Java qui définit le type de la valeur du paramètre. Il doit s'agir de l'un des types suivants :
    - java.lang.String (valeur par défaut)
    - java.lang.Integer
    - java.lang.Double
    - java.lang.Calendar
    - java.lang.Boolean
  - value : forme de la chaîne associée à la valeur d'attribut en fonction de son type

Chapitre 4. A propos de l'API IBM Unica Marketing Operations

L'API IBM Unica Marketing Operations est une façade qui offre une vue client d'une instance de Marketing Operations en cours d'exécution. Nous n'exposerons qu'une petite partie des possibilités de Marketing Operations. L'API est conçue pour être utilisée simultanément par des utilisateurs Web de Marketing Operations et par les demandes et les déclencheurs Marketing Operations Integration Services WebService SOAP. Cette API prend en charge les types suivants d'opérations :

- Création et suppression de composant
- Reconnaissance (par type de composant, valeur d'attribut, etc.)
- Inspection de composant (via ses attributs, des liens spécialisés, etc.)
- Modification de composant

Gestion des versions et compatibilité amont

Les futures versions de cette API seront compatibles avec les versions antérieures et toutes les éditions secondaires et de maintenance qui partagent le même numéro de version. Toutefois, IBM se réserve le droit d'abandonner la compatibilité avec la version antérieure pour les éditions principales "point zéro" (x.0) si le script commercial ou technique le justifie.

Le numéro de version principal de cette API sera incrémenté si l'une des modifications suivantes est apportée :

- modification de l'interprétation des données ;
- modification de la logique métier (par exemple, modification de la fonctionnalité de méthode de service) ;
- modification des paramètres de méthode et/ou des types de retour.

Le numéro d'édition de l'API sera incrémenté si l'une des modifications suivantes est apportée (notez que ces modifications sont par définition compatibles en amont) :

- ajout d'une nouvelle méthode ;
- ajout d'un nouveau type de données et restriction de son utilisation à une nouvelle méthode ;
- ajout d'un nouvel élément à un type énuméré ;
- définition d'une nouvelle version d'interface avec un suffixe de version.

Sécurité utilisateur

L'authentification est supposée être effectuée par le gestionnaire d'exécution de la procédure et par les informations utilisateur authentifié liées au contexte d'utilisation utilisé par toutes les API. L'API n'expose pas l'utilisateur authentifié mais le transmet à IBM Unica Marketing Operations afin qu'il l'utilise si nécessaire.

Toutefois, l'utilisateur authentifié peut ne pas être autorisé à effectuer toutes les opérations exposées par l'API. Dans ce cas, la méthode d'API émet un élément **`AuthorizationException`**.

Langue

Les seules langues prises en charge par cette version sont celles actuellement configurées pour l'instance de serveur IBM Unica Marketing Operations. Toutes les données dépendantes de la langue accessibles via l'API (messages, devise, etc.) sont supposées être dans la langue du système.

Gestion d'état

Cette API est sans état, ce qui signifie qu'aucune information par client n'est enregistrée par l'API au fil des appels.

Notez cependant que certains appels API peuvent modifier l'état des instances de composant sous-jacentes gérées par IBM Unica Marketing Operations, et ces changements d'état peuvent être conservés dans la base de données.

Transactions de base de données

Cette API n'expose pas les transactions de base de données au client, mais utilise ces informations si elles sont incluses dans le contexte d'exécution. Si une transaction est démarrée, l'effet de tous les appels API au sein d'une procédure particulière peut être atomique. Les autres utilisateurs d'IBM Unica Marketing Operations ne verront pas les modifications tant que la procédure n'aura pas validé la transaction.

Les appels API qui mettent à jour la base de données doivent tout d'abord acquérir un verrou d'édition afin d'empêcher les autres utilisateurs de Marketing Operations de modifier les données sous-jacentes durant l'exécution du ou des appels API. Les autres utilisateurs ne pourront alors pas mettre à jour les composants verrouillés tant que l'appel API n'aura pas terminé son exécution ; de même, un autre utilisateur de Marketing Operations ou un autre client de l'API peut avoir acquis le verrou sur les données concernées, cela empêchant l'appel API de s'exécuter.

Traitement des événements

Les opérations effectuées sur les composants IBM Unica Marketing Operations via cette API génèrent les mêmes événements que si l'opération avait été effectuée par un utilisateur Web de Marketing Operations. En particulier, les déclencheurs attendant certains événements se déclencheront finalement dans les deux cas. Les utilisateurs qui ont souscrit à certaines notifications (par exemple, lors du changement d'état d'un projet) seront avertis des changements d'état résultant des appels API et des actions des utilisateurs Web.

A propos des types de données d'API

L'API IBM Unica Marketing Operations contient les types de données publiques suivants :

- **ApprovalMethodEnum** : type énuméré pour les méthodes d'approbation.
- **ApprovalStateEnum** : type énuméré pour les états d'approbation.
- **AssetLibraryStateEnum** : type énuméré pour les état de bibliothèque de ressources.
- **AssetStateEnum** : type énuméré pour les état de ressource.
- **AttachmentTypeEnum** : type énuméré pour les pièces jointes.
- **AttributeMap** : mappe Java contenant des attributs et leur valeur.

- **BudgetPeriodEnum** : type énuméré pour les types de période budgétaire.
- **BudgetTypeEnum** : type énuméré pour les types de budget.
- **Handle (descripteur)** : identifie une instance de composant dans une instance Marketing Operations particulière.
- **InvoiceStateEnum** : type énuméré pour les état de facture.
- **MonthEnum** : type énuméré pour les mois dans une année calendaire.
- **ProjectCopyTypeEnum** : type énuméré pour les options de copie de projet.
- **ProjectStateEnum** : type énuméré pour les états de projet.
- **TaskStateEnum** : type énuméré pour les état de tâche.
- **QuarterEnum** : type énuméré pour les trimestres dans une année calendaire.
- **WeekEnum** : type énuméré pour les semaines dans une année calendaire.

Les valeurs possibles pour chaque type de données sont mentionnées ci-après.

Types énumérés

ApprovalMethodEnum

ApprovalMethodEnum est un type énuméré qui définit toutes les méthodes d'approbation possibles. Les valeurs possibles sont :

- SIMULTANE
- SEQUENTIEL

ApprovalStateEnum

ApprovalStateEnum est un type énuméré qui définit tous les états d'approbation possibles. Les valeurs possibles sont :

- PAS D'ETAT (NOT_STATED)
- EN ATTENTE (ON_HOLD)
- EN COURS (IN_PROGRESS)
- TERMINE
- ANNULE

AssetLibraryStateEnum

AssetLibraryStateEnum est un type énuméré qui définit tous les états de bibliothèque de ressources possibles. Les valeurs possibles sont :

- ACTIVE
- DESACTIVE

AssetStateEnum

AssetStateEnum est un type énuméré qui définit tous les états possibles d'une ressource. Les valeurs possibles sont :

- BROUILLON
- VERROUILLE
- FINALISE
- ARCHIVE

AttachmentTypeEnum

AttachmentTypeEnum est un type énuméré qui définit tous les types possibles de pièce jointe. Les valeurs possibles sont :

- URL
- FICHIER
- RESSOURCE

BudgetPeriodEnum

BudgetPeriodEnum est un type énuméré qui définit toutes les périodes budgétaires possibles. Les valeurs possibles sont :

- TRIMESTRIEL
- MENSUEL
- HEBDOMADAIRE
- ANNUEL
- TOUT

BudgetTypeEnum

BudgetTypeEnum est un type énuméré qui définit toutes les types de budget possibles. Les valeurs possibles sont :

- TOTAL
- PREVU
- ENGAGE
- EFFECTIF
- ALLOUE

InvoiceStateEnum

InvoiceStateEnum est un type énuméré qui définit tous les états possibles d'une facture. Les valeurs possibles sont :

- BROUILLON
- PAYABLE
- PAYE
- ANNULE

MonthEnum

MonthEnum est un type énuméré qui définit toutes les valeurs possibles pour un mois.

ProjectCopyTypeEnum

ProjectCopyTypeEnum est un type énuméré qui définit tous les types possibles d'une copie de projet. Les valeurs possibles sont :

- COPY_USING_PROJECT_METRICS
- COPY_USING_TEMMPLATE_METRICS

Pour plus d'informations sur les états des projets et des tâches, voir le *Guide d'utilisation Marketing Operations*.

ProjectStateEnum

ProjectStateEnum est un type énuméré qui définit tous les états possibles d'un projet ou d'une requête. Les valeurs possibles sont :

- NON DEMARRE (NOT_STARTED)
- EN COURS (IN_PROGRESS)
- EN ATTENTE (ON_HOLD)
- EN RAPPROCHEMENT (IN_RECONCILIATION)
- EN RETARD (LATE) : indique que le projet n'a pas démarré à la date de début prévue.
- DEPASSE (OVERDUE) : indique que le projet ne s'est pas terminé avant la date de fin prévue.
- BROUILLON
- SOUMIS
- RENVOYE
- ACCEPTE
- TERMINE
- ANNULE
- SUPPRIME

TaskStateEnum

TaskStateEnum est un type énuméré qui définit tous les états possibles d'une tâche de flux de travaux. Les valeurs possibles sont :

- EN ATTENTE
- ACTIF
- TERMINE
- IGNORE
- DESACTIVE

QuarterEnum

QuarterEnum est un type énuméré qui définit toutes les valeurs possibles pour un trimestre.

WeekEnum

WeekEnum est un type énuméré qui définit toutes les valeurs possibles pour une semaine.

Descripteur

Un descripteur (Handle) est un objet URL spécial qui fait référence à une instance de composant dans une instance Marketing Operations particulière. Les descripteurs incluent le type de composant, l'identificateur des données interne, une URL de base d'instance, etc. Les descripteurs utilisés ou générés par l'API peuvent être externalisés vers une URL complète qui peut être utilisée pour naviguer vers une vue du composant dans l'interface graphique Marketing Operations, copiée-collée, envoyée dans des courriers électroniques, utilisée dans une autre procédure sous forme de paramètre, etc.

Les descripteurs sont uniquement valides pour une instance de service ou une instance en cluster Marketing Operations particulière mais leur validité est garantie pour toute la durée de vie du service déployé. Par conséquent, les descripteurs peuvent être sauvegardés dans un fichier pour une référence ultérieure, mais ils ne peuvent pas être utilisés pour accéder aux composants sur une autre instance Marketing Operations (même si elle se trouve sur la même machine). Toutefois, Marketing Operations ne fournit pas de mécanisme de mappage de différentes URL de base à l'instance en cours en vue d'accueillir le déplacement d'une instance sur une autre machine (par exemple, si la machine d'origine présente un dysfonctionnement).

Les descripteurs sont indépendants du client. Par exemple, un déclencheur peut transmettre un descripteur à une procédure, qui l'utilise ensuite en tant que paramètre dans un appel SOAP en direction d'un système tiers, qui renvoie une demande SOAP à Marketing Operations afin d'appeler une procédure qui met à jour un attribut.

La classe Handle comporte des méthodes de fabrique destinées à créer des descripteurs pour divers types d'URL.

Objet approbation :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=approvaldetail&approvalid=101
```

Objet AssetLibrary (bibliothèque de ressources) :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=library&id=101
```

Objet dossier de ressource :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=folder&id=101
```

Objet ressource :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=asset&assetMode=VIEW_ASSET
  &assetid=101
```

Objet pièce jointe :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=attachmentview&
  attachid=101&parentObjectId=101&parentObjectType=project
```

Objet compte financier :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=accountdetails&accountid=101
```

Objet ligne de facture :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=invoicedetails&
  invoiceid=134&line_item_id=101
```

Objet facture :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=invoicedetails&invoiceid=134
```

Objet marketing :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=componenttabs&
  componentid=creatives &componentinstid=1234"
```

Grille d'objet marketing :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=componenttabs&
  componentid=creatives &componentinstid=1234&gridid=grid"
```

Ligne de grille d'objet marketing :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=componenttabs&
  componentid=creatives &componentinstid=1234&gridid=grid&gridrowid=101"
```

Projet :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&projectid=1234"
```

Grille de projet :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&
  projectid=1234&gridid=grid"
```

Ligne de grille de projet :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&
  projectid=1234&gridid=grid &gridrowid=101"
```

Objet ligne de projet :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projecttabs&
  projectid=1234&projectlineitemid=123&projectlineitemisversionfinal=false"
```

Objet équipe :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=teamdetails&
  func=edit&teamid=100001
```

Objet utilisateur :

```
http://mymachine:7001/MktOps/affiniumplan.jsp?cat=adminuserpermissions&
  func=edit&userId=101
```

Tâche de flux de travaux :

```
"http://mymachine:7001/MktOps/affiniumplan.jsp?cat=projectworkflow&
  projectid=1234&taskid=5678"
```

Mappe d'attribut

Une mappe d'attribut (AttributeMap) est une mappe qui contient uniquement des attributs. Le *nom* d'attribut est la clé d'entrée de mappe et le a matrice des *valeurs* d'attribut (notez le pluriel) est la valeur de l'entrée de mappe.

AttributeMap inclut les zones suivantes :

- *Nom* : nom défini par programme de l'attribut. Ce nom sert de clé unique pour accéder à l'attribut dans l'instance de composant où il apparaît.

  Remarque : Le *nom* n'est pas obligatoirement le nom d'affichage présenté à l'utilisateur dans l'interface graphique. Pour les composants créés avec des modèles (tels que les projets ou les tâches de flux de travail), le nom d'attribut est spécifié par la définition d'élément du modèle et doit être unique. Pour les autres composants, le nom d'attribut est généralement dérivé par voie de programme de l'instance de composant côté serveur (par exemple, via l'introspection Java).

  Remarque : Par convention, les attributs personnalisés incluent le nom du formulaire dans lequel la version éditable est définie : <nom_form>.<nom_attribut>.

- *valeurs* : matrice d'objet Java, contenant zéro valeurs d'attribut ou plus. Le type de chaque valeur doit être identique et en accord avec le type de l'attribut défini dans Marketing Operations. Seul l'encapsuleur Java et les types Marketing Operations suivants sont pris en charge :
  - AssetLibraryStateEnum : valeur AssetLibraryStateEnum de type énuméré.
  - AssetStateEnum : valeur AssetStateEnum de type énuméré.
  - AttachmentTypeEnum : valeur AttachmentTypeEnum de type énuméré.
  - AttributeMap : mappe qui contient des attributs.
  - BudgetPeriodEnum : valeur BudgetPeriodEnum de type énuméré.
  - BudgetTypeEnum : valeur BudgetTypeEnum de type énuméré.
  - Handle : référence à une instance de composant, une ligne de grille, un attribut, etc.
  - InvoiceStateEnum : valeur InvoiceStateEnum de type énuméré.
  - java.io.File : représentation d'un fichier.
  - java.lang.Boolean : valeur booléenne (True ou False)
  - java.lang.Double : valeur de nombre décimal à double précision.
  - java.lang.Float : valeur de nombre décimal à simple précision.
  - java.lang.Integer : valeur de type entier 32 bits
  - java.lang.Long : valeur de type entier 64 bits
  - java.lang.Object : objet Java générique
  - java.lang.String : chaîne comprenant zéro ou plusieurs caractères Unicode
  - java.math.BigDecimal : valeur de nombre décimal signée à précision arbitraire. Convient pour les devises ; l'interprétation de la valeur dépend de la langue utilisée pour les devises pour le client.
  - java.math.BigInteger : valeur de type entier à précision arbitraire.
  - java.net.URL : objet URL.
  - java.util.ArrayList : liste d'objets.
  - java.util.Calendar : valeur date-heure pour une langue particulière.
  - java.util.Date : valeur date-heure. Ce type est obsolète. Utilisez à la place java.util.Calendar ou java.util.GregorianCalendar.
  - java.util.GregorianCalendar : GregorianCalendar est une sous-classe concrète de java.util.Calendar et fournit le système calendaire standard utilisé dans la plupart des pays du monde.
  - MonthEnum : valeur MonthEnum de type énuméré.
  - ProjectStateEnum : valeur ProjectStateEnum de type énuméré.
  - QuarterEnum : valeur QuarterEnum de type énuméré.
  - TaskStateEnum : valeur TaskStateEnum de type énuméré.
  - WeekEnum : valeur WeekEnum de type énuméré.

Les métadonnées d'un attribut (telles que le nom d'affichage localisé et la description associée) sont définies par le modèle associé à l'attribut et à son instance d'objet parent. Les attributs fournissent un mécanisme simple et extensible pour exposer les attributs d'instance d'objet facultatifs et obligatoires, tels que le nom du projet, le code et la date de début.

Remarque : Pour mettre en oeuvre la date, les utilisateurs peuvent utiliser java.util.Calendar ou java.util.GregorianCalendar.

Exceptions courantes

Cette rubrique décrit certaines exceptions courantes émises par l'API.

- AuthorizationException : L'utilisateur associé au contexte d'exécution du client n'est pas autorisé à effectuer l'opération demandée. Cette exception peut être émise par n'importe quelle méthode API, elle n'est donc pas déclarée.
- DataException : Une exception s'est produite dans la couche de base de données sous-jacente dans IBM Unica Marketing Operations. Pour plus d'informations, voir le journal SQL.
- InvalidExecutionContextException : Un problème lié au contexte d'exécution transmis à une méthode API s'est produit (par exemple, la méthode n'a pas été correctement initialisée). Cette exception peut être émise par n'importe quelle méthode API, elle n'est donc pas déclarée.
- NotLockedException : Tentative de mettre à jour des données de composant sans acquisition préalable du verrou requis. Voir la méthode acquireLock() de IExecutionContext.

Méthodes API

Pour obtenir des informations spécifiques sur les méthodes API publiques, reportez-vus à la classe iPlanAPI dans les fichiers de documentation API JavaDoc. Pour accéder à ces fichiers, connectez-vous à Marketing Operations et sélectionnez **Aide > Documentation produit** dans n'importe quelle page, puis téléchargez le fichier `<version>PublicAPI.zip`.

Coordonnées du support technique d'IBM Unica

Si vous rencontrez un problème que vous ne parvenez pas à résoudre à l'aide de la documentation, le responsable désigné dans votre société peut contacter le support technique d'IBM Unica. Utilisez les informations de cette section pour garantir la résolution efficace de votre problème.

Si vous ne faites pas partie des personnes autorisées dans votre société à appeler le support technique, contactez votre administrateur IBM Unica qui vous donnera toutes les informations nécessaires.

Informations à rassembler

Avant de contacter le support technique d'IBM Unica, rassemblez les informations suivantes :

- une brève description de la nature du problème,
- les messages d'erreur détaillés qui apparaissent lorsque l'erreur se produit,
- la liste détaillée des étapes permettant de reproduire l'erreur,
- les fichiers journaux, les fichiers de la session, les fichiers de configuration et les fichiers de données appropriés,
- les informations sur l'environnement de votre système et de votre produit, que vous pouvez obtenir en procédant comme indiqué dans la section Informations sur le système.

Informations système

Lorsque vous appelez le support technique d'IBM Unica, vous pouvez être invité à fournir des informations sur votre environnement.

Si vous pouvez vous connecter à votre application, la plupart de ces informations sont disponibles dans la page À propos, qui affiche des informations sur l'application IBM Unica installée.

Vous pouvez accéder à la page A propos de en sélectionnant **Aide > A propos de**. Si la page A propos de est inaccessible, il est possible d'obtenir le numéro de version de chaque application IBM Unica en consultant le fichier `version.txt` situé dans le répertoire d'installation de chaque application.

Coordonnées du support technique d'IBM Unica

Pour savoir comment contacter le support technique d'IBM Unica, consultez le site Web du support technique des produits IBM Unica : (http://www.unica.com/about/product-technical-support.htm).

Remarques

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Pour le Canada, veuillez adresser votre courrier à :

IBM Director of Commercial Relations
IBM Canada Ltd.
3600 Steeles Avenue East
Markham, Ontario
L3R 9Z7
Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues par écrit à l'adresse suivante :

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales. LE PRESENT DOCUMENT EST LIVRE "EN L'ETAT" SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation
170 Tracer Lane
Waltham, MA 02451
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM conformément aux dispositions de l'ICA, des Conditions internationales d'utilisation des logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Tous les tarifs indiqués sont les prix de vente actuels suggérés par IBM et sont susceptibles d'être modifiés sans préavis. Les tarifs appliqués peuvent varier selon les revendeurs.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes. Les exemples de programme sont fournis "en l'état", sans garantie d'aucune sorte. IBM ne sera en aucun cas responsable des dommages liés à l'utilisation de ces programmes.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Marques

IBM, le logo IBM et ibm.com sont des marques d'International Business Machines Corp., dans de nombreux pays. Les autres noms de produits et de services peuvent être des marques d'IBM ou d'autres sociétés. La liste actualisée de toutes les marques d'IBM est disponible sur la page Web «Copyright and trademark information» à l'adresse suivante : www.ibm.com/legal/copytrade.shtml.