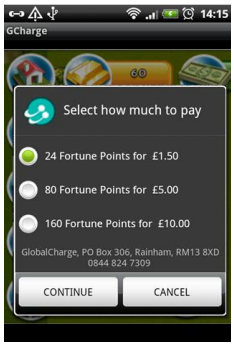
GlobalCharge In-App billing

Executive Summary

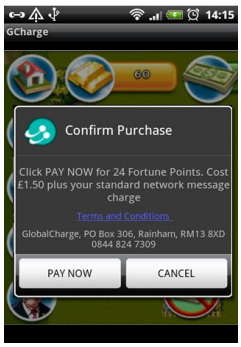
"Our aim is to create a cross-operator platform benefiting the entire eco-system through portable 'payment enabled' applications"

GlobalCharge provides a framework for operators, aggregators and developers to enable the sale of virtual goods and premium features inside Android apps using mobile operator billing and premium SMS.

Example In-App payment screens



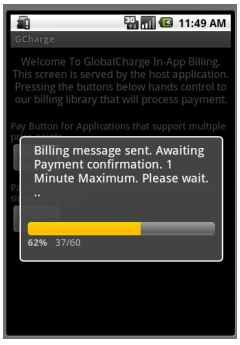
App hands payment request to billing library
Applicable price points presented with emphasis on regulatory compliance



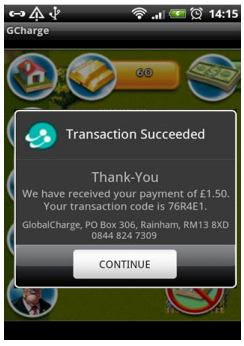
Payment screen confirms price option selected

Royalty Free

The GlobalCharge In-App billing solution is available on a royalty free basis for Operators selling their content. We provide our billing library to developers for inclusion in their apps, and server side APIs to operators and aggregators. (GlobalCharge have adopted an extended version of the GSMA's OneAPI V1.0 SMS and Payment via SOAP) These transactions do not touch, and are in no way dependent on the GlobalCharge network.



Transaction progress is displayed – carrier billing and PSMS supported



Confirmation message displayed and control then handed back to application

What is in it for GlobalCharge?

All we ask is that Operators and aggregators taking advantage of GlobalCharge allow us to provide billing services on normal commercial terms (through the same interfaces) to other GlobalCharge approved developers. In these cases, we act as a clearing house for transactions and retain a fee for the provision of this service.

For more details please contact Antony Redfern

antony.redfern@globalcharge.com

Tel: +44 208 596 5035

GlobalCharge In-App billing
Background and more detail

Apple success:

Standard Handsets	+	Quality Content	+	Seamless billing

Android Challenge:

Handset variations	+	Unvetted Content	+	Inconsistent Billing?

The GlobalCharge Proposition:

*Royalty free In-App billing solution for **Operators** selling their contracted content**
Operators** benefit from their contracted content sold through other **Operators
***Operators** benefit from approved third-party content sold over their network*
*One-build solution – attractive to the **Development** community*
** works independent of GlobalCharge Network – introduces no potential points of failure*

Background

Changed landscape

Operators have enjoyed many years of exclusive access to both handset and end-user. Those days are gone as premium content is increasingly embedded on handsets; available through OS/manufacturer based portals (Blackberry App World, Android Market) and downloaded from independent retailers such as Getjar and GoMobile. Walled gardens have gone the way of the dodo - WiFi availability has seen to that. So the options available to operators are becoming less attractive. They can embed games/content on handsets prior to distribution (expensive), deliver content through on device portals, over the air (cumbersome) or simply promote attractive options via their operator portals or by pushing marketing messages back to the handset.

None of these options have any chance of competing with the simple proposition presented through iTunes and, to some extent, Android Market.

Apple has changed perception of application downloads; once the domain of the geeks, now everyone with an iPhone or iPad provides developers on these platforms with an attractive marketplace for their wares. Apple's insistence on linking a credit card (or other pre-payment) to accounts works well in some markets, but is not ideal in others, for example only 15% of Germans say the credit card is their favoured form of payment but in the main, developers are happy with their 70% share.

Android is more challenging as Google Checkout is not a global solution, and like Apple, is predominantly a credit card offering. Carrier billing is on its way through Checkout (AT&T), but for developers with a requirement today, the only option is to strike deals on a country by country basis and deal with the considerable workload that creates.

The fact remains that the Operators are in the best position to capitalise on the micro-payment opportunity represented by the App marketplace; after all, they **STILL** own the primary billing relationships with the customers.

Why In-App?

What is the best way to charge for Apps and games? Apple maintains high quality standards for iTunes Apps and feedback from customers boosts confidence that purchases will represent value for money; paid before download. Refunds, when justified, are well managed. The Android Market is open and therefore less controlled; confidence in quality is lower. Refunds are a one-click process for the end-user so are problematic for developers. Wrappering does not work well in this situation.

The answer? Try before you buy, typically delivered via a wrapper, but now more seamlessly presented through In-App billing. If the end-users are happy with what they see, then the In-App solution provides a simple customer journey to acquire the game.

Business logic

Anyone with experience in the value added marketplace understands that access to a payment gateway is only part of the challenge. Business logic and the application of local regulatory requirements are key: preventing obvious duplicated transactions, velocity checking and limit enforcement, blacklisting, stats and reporting and all help to reduce end-user complaints and therefore regulatory issues – we all remember what happened when a certain frog went wild. This seriously damaged the mobile subscription industry – let's work to prevent history repeating itself.

The GlobalCharge solution

Radical? Perhaps...

We've created an In-App billing tool for Android by creating a JAR file that becomes a part of the application. It presents compliant payment screens to the end-user (Payforit approved) and handles multiple price points; carrier and PSMS billing from either mobile handset or WiFi enabled tablet, and we are giving it away! (under a conditional royalty free licence). Connection to the

Operator is through OneAPI soap interfaces with extensions to include some missing functions such as multiple price points.

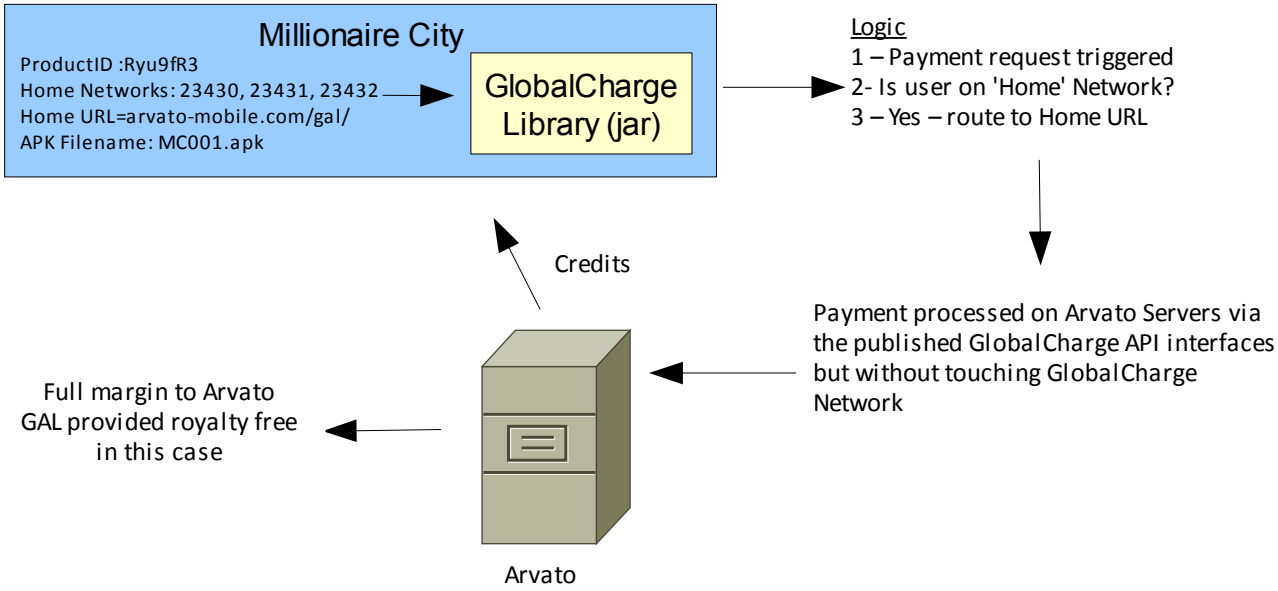
This is how it works:

Use Case 1

1. Operator adds subset of OneAPI interfaces as specified by GlobalCharge
2. Operator provides API endpoints to their contracted Developers
3. Developers add GlobalCharge jar to their Apps and deploy
4. Resulting transactions do not touch the GlobalCharge networks and are royalty free

Phone Network: T-Mobile
Client: Arvato
Publisher: Digital Chocolate

Game running on Android handset:

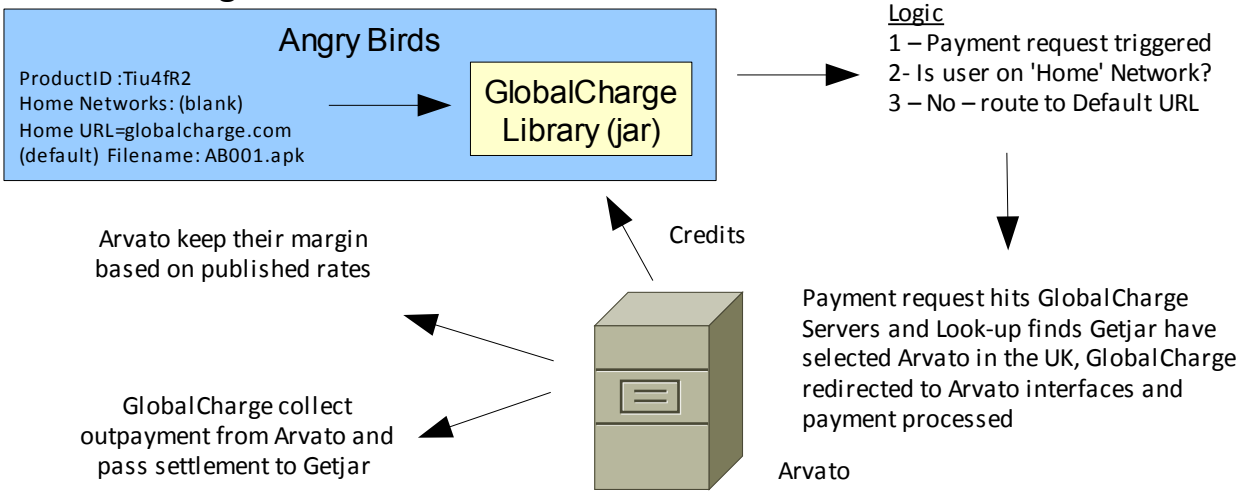


Use Case 2

1. Operator adds commercial rates to GlobalCharge outpayment database
2. Third party with approved content requests payment through host operator using the same OneAPI interfaces
3. Payment is collected by Operator and forwarded to GlobalCharge for onward settlement

Phone Network: T-Mobile
Client: GlobalCharge
Publisher: Getjar

Game running on Android handset:

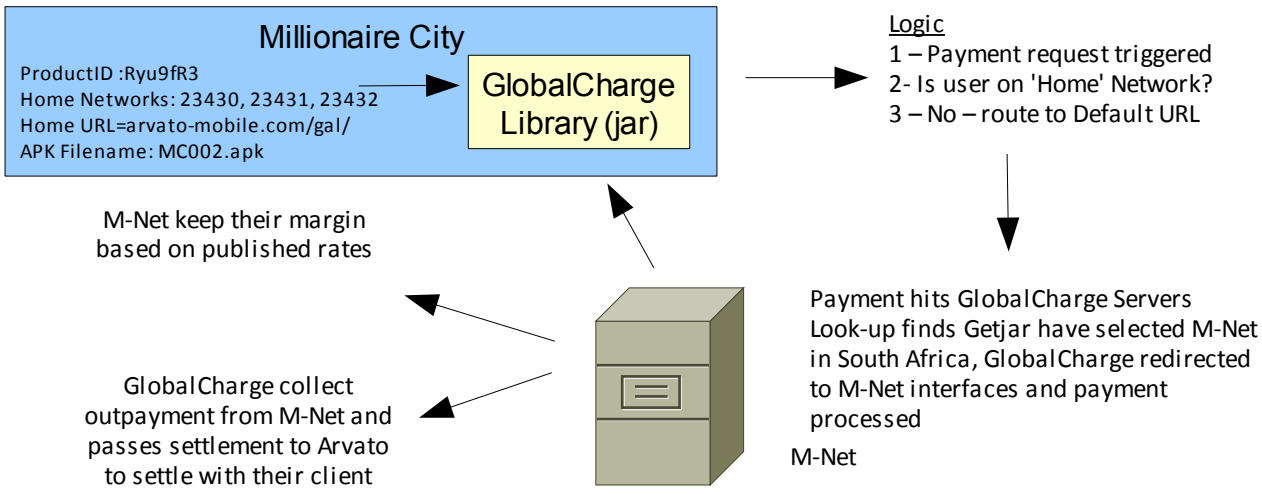


Use Case 3

1. Operator's contracted Developer' end-user on 'alien' Operators' Network request a payment through their OneAPI enabled interfaces
2. Alien Network completes transaction
3. Payment is collected by Alien Operator and forwarded to GlobalCharge for onward settlement to Operator with whom the developer contracted who, in this case, receives payment for transaction on another Operator network

Phone Network: M-Net
Client: Arvato
Publisher: Digital Chocolate

Game running on Android handset:



Developers want an easy life – the fewer the complexities the happier they are. So our 'one-build' global, carrier agnostic solutions is attractive.

The OneAPI initiative gives GlobalCharge a chance to communicate with carriers in a language they understand.

GlobalCharge act as a clearing house for In-App payments.

Operators, aggregators, portal providers etc. may use our library freely in respect of their customers. So, if Arvato sign up Digital Chocolate to use the GlobalCharge In-App billing library, the resulting transactions are handled completely independent of the GlobalCharge network. As a condition of the licence, Arvato will make these interfaces available for billing under their usual aggregator terms. So an Arvato customer who has downloaded an App from GlobalCharge client; Getjar, for example, will have access to In-App billing through the same interfaces as the Digital Chocolate customer, but in this case 'on-net' revenues are paid to GlobalCharge (and then onto Getjar). As a further twist, Arvato will receive revenue from customers of Digital Chocolate on other GlobalCharge active networks.

In this example, Arvato would post their rate offerings onto the GlobalCharge central server so that other users of the GlobalCharge In-App library can see what offering is available and select their preferred carrier by country.

Conclusion:

By offering the GlobalCharge In-App solution free to Operators and by choosing OneAPI standard interfaces we are making it easy for Operators to use our system to become the first cross operator In-app solution. This approach benefits all concerned, as the more operators that choose to use our solution, more developers will opt for this too. With wider coverage, their efforts will be more handsomely rewarded.