Internet+ Technical description KITv3.5 (single one-off payment) w-HA

Summary :	The purpose of the document is to describe the single one-off payment using w-HA payment system. Following points are get onto: » Process principle of w-HA system » Technical description of w-HA system (content provider side) » Implementation of w-HA payment system

VALIDATION

	Name	Department	Date
Validation	DIAKONOFF	Relation Client	04/09/2009

REVISION HISTORY

Version	Date	Modifications
1.10	04/09/2009	Creation
1.11	19/10/2009	Update of w-HA nodes + refund ($2.2.1)

REFERENCED DOCUMENTS

Title	Typologie

TABLE DES MATIERES

1. TRANSACTIONS' GENERAL CONCEPTS

1.1. Vocabulary

The aim of this paragraph is to define "protagonists" involved in a transaction with w-HA, and to explain different flows between protagonists.

w-HA
The word "w-HA" can point, depending on the context:
- w-HA company
- w-HA platform

Content provider
The « Merchant » (or content provider) is the seller of service accessible on the net.

The ISP (Internet Servcie Provider) of the client
The ISP has commercial and financial connection with the internet user and has a partnership with w-HA.

Internet user
The internet user is the final client who buys, on the net, to a product or service using w-HA payment solution.

1.2. Principle scheme (commercial and marketing aspect)

1.2.1. Transaction process with the client point of view

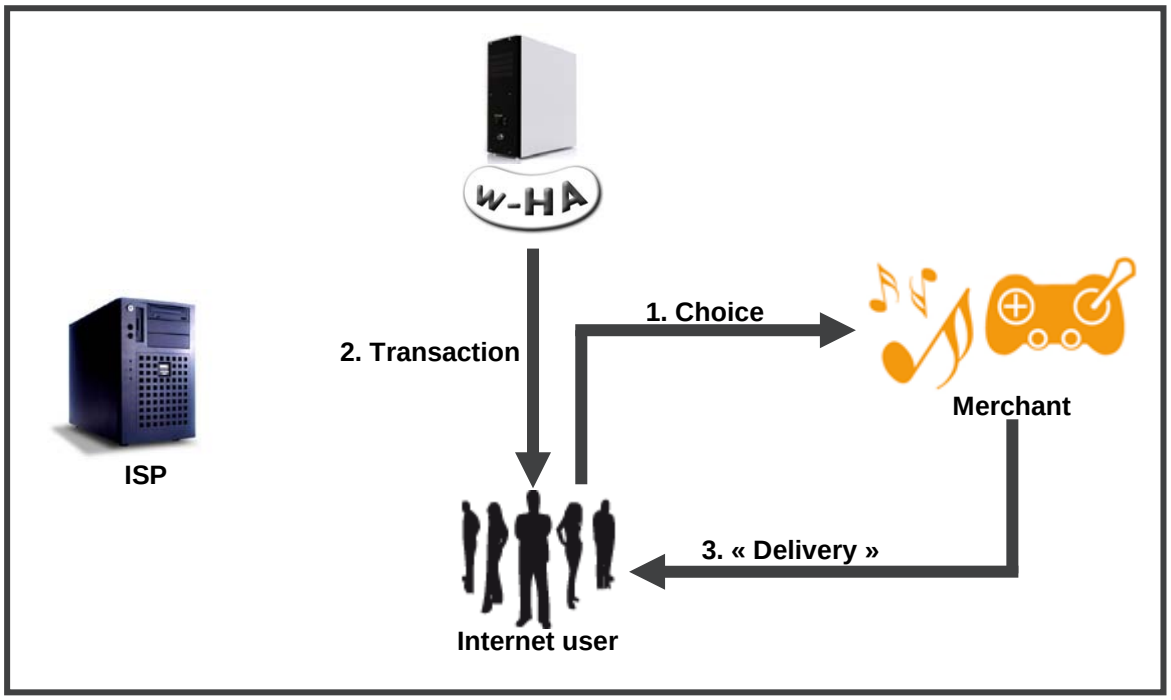


fig 1 : transaction process (internet user point of view)

The internet user will see pages bellow:

1. « merchand » screen:	The internet user chooses a transaction on the merchant web site.
2. « w-HA » screen:	w-HA displays the billing panel on which the internet user will confirm his transaction.
3. « merchand » screen:	After w-HA authorization the internet user is redirected to the merchant web page to access to the service he bought.

Warning :
Only web page seen by the internet user are presented here. In reality, there are different information flows between the protagonists. Details of thoses flows are given in §1.3.

1.2.2. Financial flows (deferred)

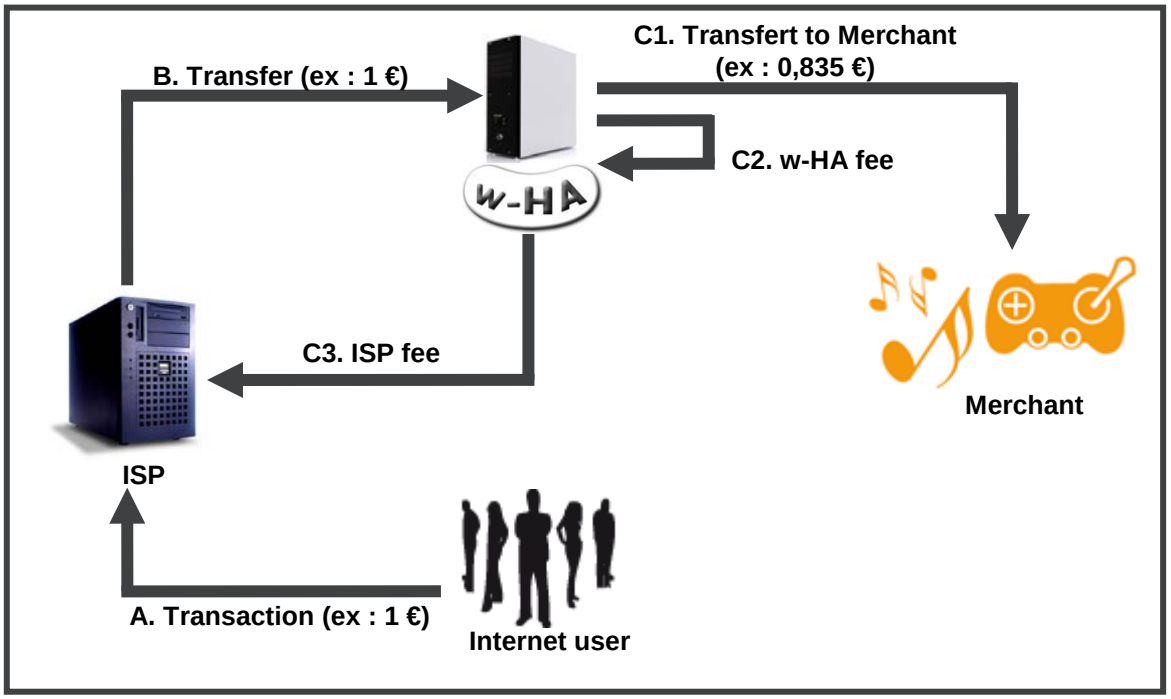


fig 2 : financials flows between all protagonists is the transaction

Financials flows between all protagonists are the transactions (deferred) are:

Example for a 1€ transaction:

A	The internet user is billed by its ISP
B	The ISP transfer 100% off the payment to w-HA
C1	w-HA transfers a part of the payment to the Merchant (ex : 0,835€)
C2	w-HA conserves a part of the payment
C3	w-HA transfers a part of the payment to the ISP

1.3. Transaction cinematic with w-HA payment system (technical aspects)

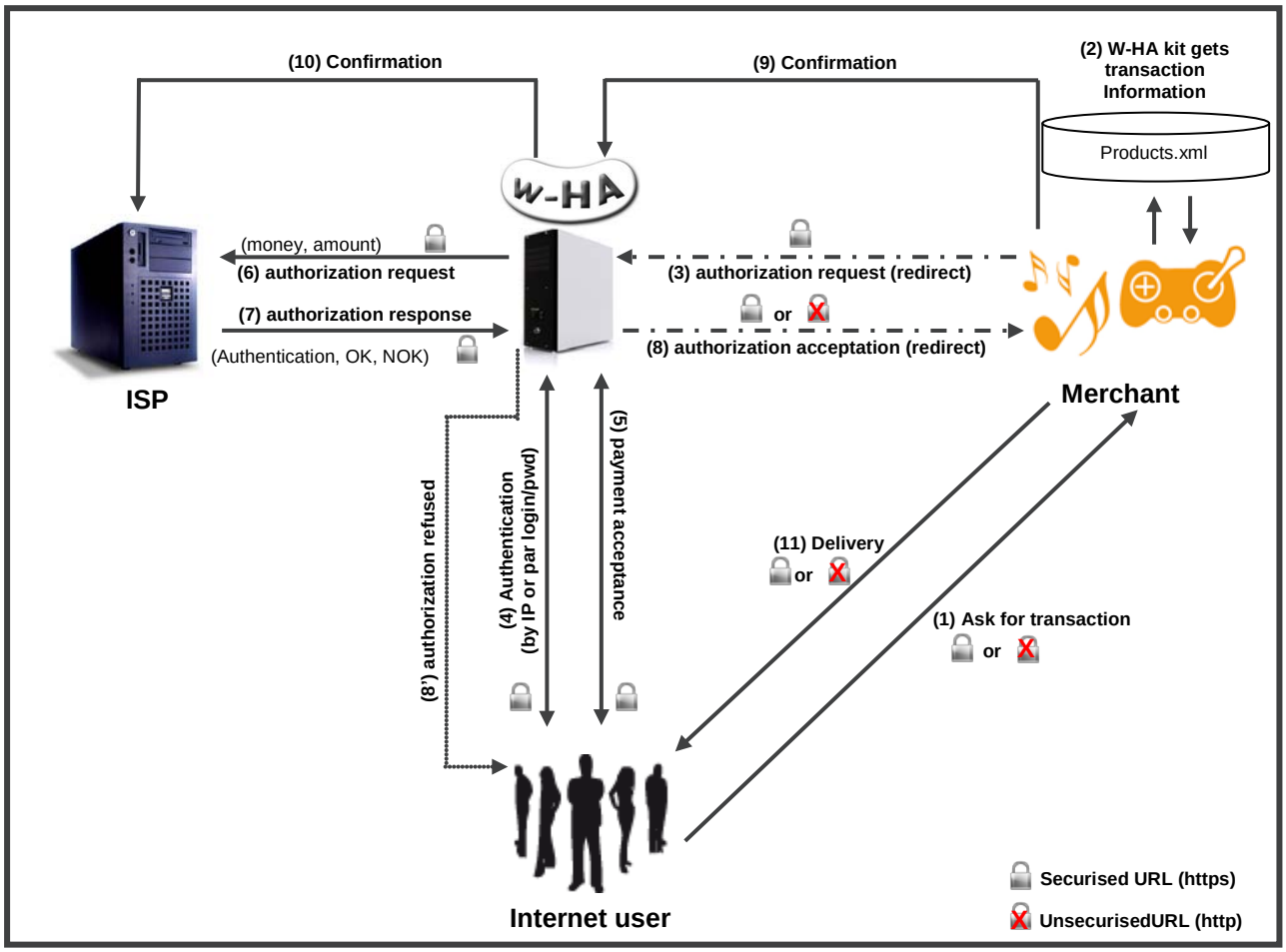


fig 3 : details of information flows all protagonists is the transaction

1.4. Internet user Registration, Authentication and Payment

1.4.1. Inscription

w-HA ISP partners
In general, the internet user **registration** of the w-HA payment system at its ISP is **automatic** (optin) when he creates his internet access. The internet user can disable this option later if he wants to. He can also accept or not “nomad payment” (see § bellow).

Others ISP
If the internet user subscribed to internet to an ISP, which is **not w-HA partner**, or if he wishes to pay with an other payment system, he can open an **Orbeo** account with his **bankcard** on http://www.w-ha.com.

1.4.2. Authentication

Internet connexion with ISP partner
If the internet user uses an **ISP partner** connexion when he buys, he is **automatically identified by his IP address** and he is immediately redirected to the « billing panel » (designed by his ISP).

Internet connexion with an other ISP
If the internet user use an ISP, which is **not internet+ partner** (or a company internet access) when he buys, he is redirected to the « panel of ISP choice ».

If he accepted « **nomad payment** », he can **choose his ISP and authenticate him self (login/pwd)** then, he will be redirected to the « **billing panel** ».

He can also decide to buy with his **Orbeo account**, if the merchant allow this type of payment. The internet user has to **authenticate him self (login/pwd)** when he opens his Orbeo account.

1.4.3. Payment

When the internet user is authenticated (automatically by his IP address or by login/pwd), he is redirected to the billing panel.

This billing panel is designed by his ISP and display information bellow:
- merchant name
- merchant logo
- product description
- transaction amount

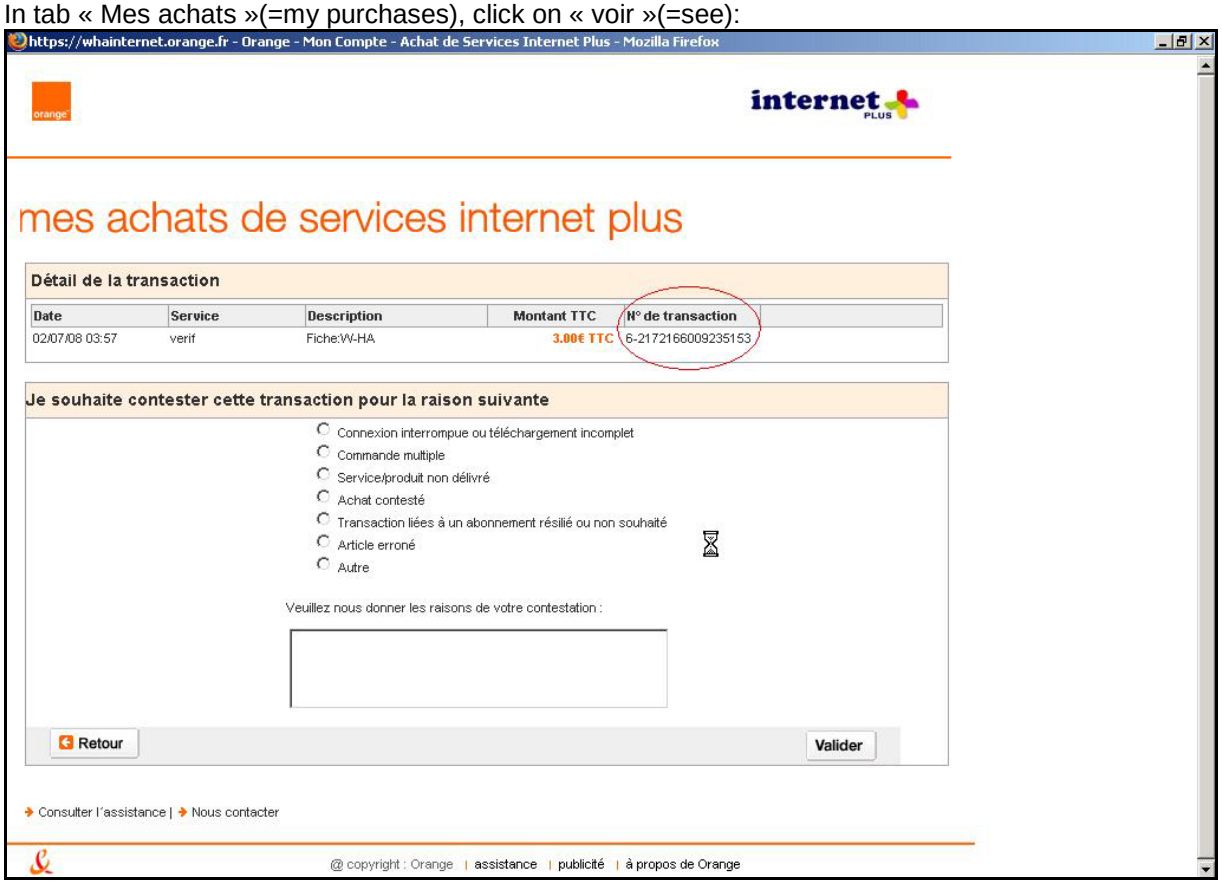
The internet user has to **clic just one more time to buy** (confirmation button). Thanks to its system, w-HA perform **one of the best transformation rates of the market!**

Example: Orange billing panel

1.4.4. Transaction refund

An internet user Orange (for instance) may on the web, the refund of a transaction:
www.orange.fr > espace client > conso internet (voir le détail) > Achats de service internet plus (voir le détail) > Consulter votre relevé détaillé
(direct access: https://wanadoo.w-ha.com/app-am/node)

In tab « Mes achats »(=my purchases), click on « voir »(=see):



Note: a transaction ID (for an Orange client) is like: 6-XXXXXXXXXXXXXXX

The demand is transmitted to the internet access provider who decides (or not) the refund.

2. FUNCTIONING DETAILS OF EACH FUNCTIONALITY / SERVLET

2.1. Servlets with redirection of the internet user to the w-HA platform

The functionality “single one-off payment” lead the internet user (via his browser) to the redirection to the w-HA platform.

Caution !

The « pos_init » sevlet must be accessible from the public internet (port 80).

2.1.1. Transaction demand / « pos_init?action=authorize »

2.1.1.1 Calling the servlet

When the content provider wishes to start the transaction process, he calls the “**pos_init**” servlet with the parameter « **action** » at « **authorize** ».

A button on the offer presentation page of the content provider calls this servlet.

2.1.1.2 Calling example of the servlet

http://*wha.marchand.com*/bundle/pos_init?
action=authorize
&pid=P1
&wha_desc2=current
&ParametresSupplementaires=abc

In red: mandatory parameters
In blue: optional parameters (merchant properties)

2.1.1.3 Calling parameters of the servlet

To start the transaction process, the “pos_init” servlet is called with the following settings:

Mandatory parametrers:

action=authorize: to indicate to the servlet that this is a transaction request

pid=*product identifier*

Optional parametrers:
Additional parameters of the content provider (or “merchant properties” (mp)):

Those parameters are not theoretically obligatory. In general, the content provider uses at least one parameters type: “user ID”. This parameters allows him to associate the userID in his database to the w-HA transaction ID (trxid).

Merchant properties (mp) are in the transaction notification Url (**fulfillmentUrl**) which must be an executable (servlet, cgi, asp, php, ...)

Caution !

The additional parameter "**wha_desc2=current**" is obligatory if the billing apnel is displayed in current page.

Caution !

For ticketFun, the URL to be parametered is: **https://route.w-ha.com/app-authorization/node**

2.1.1.4 Servlet functioning

When it is called using the setting action=authorize, the "pos_init" servlet performs the following actions:

- Gets information connected to the shop (merchantId, keyId, keyValue, URLs…) in the "web.xml".
- Gets information connected to the product (productId…) in the "products.xml" file.
- Creates a secure and signed URL (https) calling the w-HA platform and containing this information as well as the settings (merchant properties) added by the merchant
- Redirects the internet user to this URL.

2.1.1.5 URL example created by servlet

The internet user is rederected to an URL like the one bellow (encoded URL):

https%3A%2F%2Froute.w-ha.com%2Fapp-authorization%2Fnode%3Fm%3Dh%3D8c2a104518b1252f92eef510545a8fa7%3Bp%3D515%3Bk%3D515%3Bv%3D2%3A%7Bc%3DAuthorizeReq%3Bv%3D%7Bps%3D2%3Bamt%3D0%3Bmp%3D%7Bpid%3DP1%3B_ap_wha_desc2%3Dcurrent%3Bts%3D2009-08-25%2011%3A50%3A32.921%3B_ap_ParametresSupplementaires%3Dabc%3B%7D%3Bpi%3DP1%3Bpg%3D0%3BmUrl%3Dhttp%3A%2F%2Flocalhost%3A8080%2Fbundle%2Fpos_init%3Bpd%3DProduit%201%3Bpc%3DImage%3Bhr%3D1%3Bcur%3DEUR%3Bcl%3D-1%3B%7D%7D%26lg%3Dfr%0A

Decoded URL:

https://route.w-ha.com/app-authorization/node?
m=h=8c2a104518b1252f92eef510545a8fa7;
p=5XXX;
k=5XXX;
v=2:{
c=AuthorizeReq;
v={
ps=2;
amt=0;
mp={pid=P1;_ap_wha_desc2=current;ts=2009-08-25
11:50:32.921;_ap_ParametresSupplementaires=abc;};
pi=P1;
pg=0;
mUrl=http://*wha.marchand.com*/bundle/pos_init;
pd=Produit 1;
pc=Image;
hr=1;
cur=EUR;
cl=-1;}}&lg=fr

This url is securised (https) and signed).

In fact it is a redirection (directive redirect) of the internet user, via his browser to the w-HA platform.

2.1.1.6 *w-HA response to the servlet*

When the user is redirected to the w-HA platform for a transaction request (see previous paragraph), a payment panel is displayed:

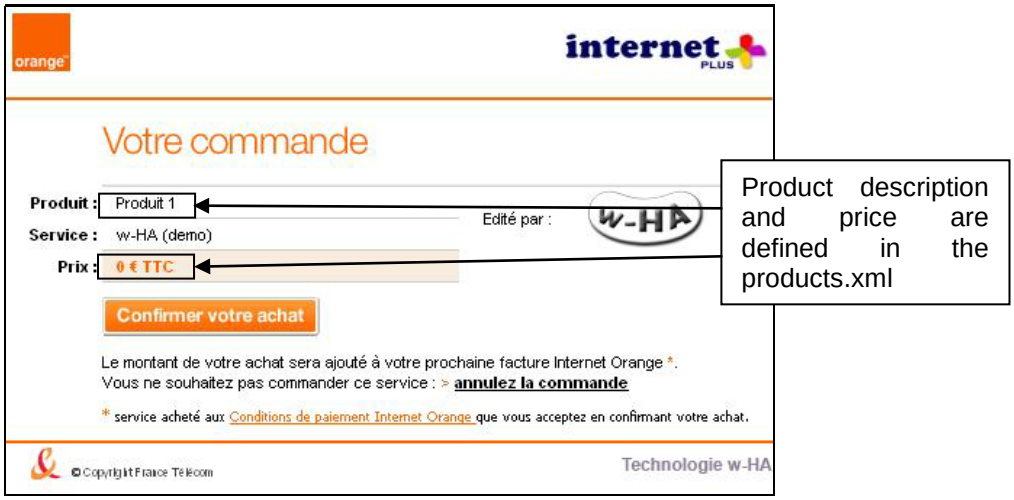


Figure 5 : Orange billing panel

The user can either, accept the transaction or refuse.

1st case: user cancels the transaction

If the user clicks on the "cancel" link, the transaction is not validated on the w-HA platform: no transaction identifier is generated.

- The user is redirected to the Content Provider "pos_init" servlet (merchantUrl) with a "c=AuthorizeCancel" message.

URL example (pos_init) to which the user is redirected in case of cancellation from the payment panel:

```
http://localhost:8080/bundle/pos_init?m=h%3D6c9a10d076c5c141f334469704b1545c%3Bp%3D515
%3Bk%3D515%3Bv%3D3%3A%7Bc%3DAuthorizeCancel%3Bv%3D%7Bpid%3DP1%3B_ap_wha_d
esc2%3Dcurrent%3Bts%3D2009-08-
25+13%3A11%3A23.796%3B_ap_ParametresSupplementaires%3Dabc%3B%7D%7D
```

Decoded URL:

```
http://wha.marchand.com/bundle/pos_init?
m=h=6c9a10d076c5c141f334469704b1545c;
p=5XXX;
k=5XXX;
v=3:{
c=AuthorizeCancel;
v={pid=P1;_ap_wha_desc2=current;ts=2009-08-25
13:11:23.796;_ap_ParametresSupplementaires=abc;}}
```

This URL is secure (https) or not (http) and signed.

Information included in this URL are:
» an hmac to ensure message integrity,
» "c=AuthorizeCancel" message
» Additional Content Provider settings (mp) (with the prefix : _ap_)

- The "pos_init" servlet redirects the user towards the "trxCancelFromPaymentPanelUrl" set in the "web.xml" file.

URL example (trxCancelFromPaymentPanelUrl) to which the user is redirected in case of cancellation from the payment panel:

```
http://wha.marchand.com/demo/bundle/html/panel_cancel.html?
ParametresSupplementaires=abc
&wha_desc2=current
&hmac=9b79ef06de7305d54fc97986fb05554f
```

This URL is secure (https) or not (http) and signed.

Information included in this URL are:
» an hmac to ensure message integrity
» merchant properties (cur + Content Provider's own settings)

2nd case: the ISP cancels the transaction (via w-HA platform)

If the internet user clicks on the "confirmer votre achat" button, but the transaction is refused (monthly spending limit reached …) no transaction identifier is created by w-Ha platform.

An error message is displayed on the billing panel.

There is no call of the "pos_init" servlet on the content provider platform to precise the error reason.

3rd case: user transaction acceptance

If the user clicks on the "Confirm your purchase" button and the transaction is accepted by the w-HA platform, a transaction identifier is created with an "authorized" status.

- The user is redirected to the Content Provider "pos_init" server (merchantUrl) with a "c=AuthorizeSuccess" message.

URL example to which the user is redirected in case of cancellation from the payment panel:

http%3A%2F%2Flocalhost%3A8080%2Fbundle%2Fpos_init%3Fm%3Dh%3Dc5b6f2ea0b77196357ab6c5ab5fe07c3%3Bp%3D515%3Bk%3D515%3Bv%3D3%3A%7Bc%3DAuthorizeSuccess%3Bv%3D%7Bst%3DFR%3Bv%3D86400%3Bmp%3D%7Bpid%3DP1%3B_ap_wha_desc2%3Dcurrent%3Bts%3D2009-08-25%2011%3A50%3A32.921%3B_ap_ParametresSupplementaires%3Dabc%3B%7D%3BtId%3D6-5150049942659815%3Bz%3D75000%3Bci%3DParis%3Brt%3Dhttps%3A%2F%2Fwanadoo.w-ha.com%2Fapp-node-mct%2Fresponder%3Bco%3DFR%3B%7D%7D%0A

Decoded URL

http://*wha.marchand.com*/bundle/pos_init?
m=h=c5b6f2ea0b77196357ab6c5ab5fe07c3;
p=5XXX;
k=5XXX;
v=3:{
c=AuthorizeSuccess;
v={
st=FR;
v=86400;
mp={pid=P1;_ap_wha_desc2=current;ts=2009-08-25
11:50:32.921;_ap_ParametresSupplementaires=abc;};
tId=6-5150049942659815;
z=75000;
ci=Paris;
rt=https://wanadoo.w-ha.com/app-node-mct/responder;
co=FR;}}

This URL is signed.

In fact it is a redirection (directive redirect) of the internet user, via his browser to the w-HA platform.

Information included in this URL are:
- an hmac to ensure message integrity
- "AuthorizeSuccess" (c=)
- Information relative to the product
- geographical information on the buyer
- additional Content Provider settings (mp=)
- the URL to confirm the transaction (rt=)
- the transaction identifier (trxid=)

The "pos_init" servlet role is to:
- Check the authorization massage sent buy w-HA platform
- Send the confirmation request to w-HA platform

2.1.1.7 Confirmation request

Upon receipt of a "c=AuthorizeSuccess" message, the "pos_init" servlet sends **automatically** and immediately the transaction confirmation request.

As for the w-HA status of the transaction goes to "confirmed".

Example of confirmation request sent by the "pos_init" servlet:

https://ticketnet.w-ha.com/app-node-mct/responder?m=h%3D33b3c2b73eb7d47944fa29cffa6d94f1%3B%0Ap%3D505%3B%0Ak%3D505%3B%0Av%3D2%3A%0A%7Bc%3DConfirm%3B%0Av%3D%7B%0As_rate%3D0%3B%0An_amt%3D1.00%3B%0Ag_amt%3D1.00%3B%0Av_amt%3D0%3B%0AtrxId%3D6-5150049942659815%3B%0Av_rate%3D0%3B%0As_amt%3D0%3B%0Acur%3DEUR%3B%7D%7D

Decoded URL:

https://ticketnet.w-ha.com/app-node-mct/responder?
m=h=33b3c2b73eb7d47944fa29cffa6d94f1;
p=5XXX;
k=5XXX;
v=2:{
c=Confirm;
v={
s_rate=0;
n_amt=0;
g_amt=0;
v_amt=0;
trxId=6-5150049942659815;
v_rate=0;
s_amt=0;
cur=EUR;}}

This URL is secure (https) and signed.

This is a **server-to-server request (https)**, so the content provider **proxy** and **firewalls** have to be configured to allow https communications.

Information included in this URL are:
- an hmac to ensure message integrity
- the debit confirmation: "confirm" (c=)
- the product amount (n_amt, g_amt)
- the transaction identifier (trxid=)

The "pos_init" servlet role is to:
- Confirm the debit (start billing process)
- Redirect the client to the delivery page (fulfillmentUrl)
- Update the "authorization.txt" file
- Update the "logs.txt" file

Caution:
The **confirmation** is done **before** the **delivery** to the client.

If the confirmation is **OK** (the "pos_init" servlet receives an acknowledgment "c=ack"):
- The **debit** is **effective**
- The client is redirected to the delivery page: **fulfillmentUrl**

If the confirmation is **not OK**:
- There is **no debit**
- The client is redirected to the page: **messageUrl**

2.1.1.8 Delivery to the client

When the “pos_init” servlet receives an acknowledgment “c=ack” form the w-HA platform, it redirects the user to the delivery page (“fulfillmentUrl”) set in the “products.xml” file.

Example of delivery URL, created by the “pos_init” servlet:

```
http(s)//wha.marchand.com/...../cgi-bin/paiement_ok.cgi?
ParametresSupplementaires=abc
&wha_desc2=current
&hmac=9b79ef06de7305d54fc97986fb05554f
&trxId=30-5150049942659815
```

This URL is secure (https) or not (http) and signed.

This URL is a program (servlet, php, asp, coldfusion …) which receives (GET method) following parameters:

- a hmac to ensure message integrity (h=),
- the transaction identifier (trxid=)
- additional Content Provider settings (mp)

2.2. Servlet for server-to-server requests

Functionalities bellow work with server-to-server requests https:

- transaction refund (“pos_request” servlet : CP → w-HA) *(CP = Content Provider)*

Warning!

“responder” and “**pos_request**” servlets functionalities **must NOT be avialable from public internet (port 80).**

Warning!

Because those requests are **https**, **network equipments** need to be correctly configured for **requests form/to w-HA platform**.

2.2.1. Transaction refund / « pos_request ?action=refund »

The content provider can refund a transaction with a server to server request to the w-HA platform.

2.2.1.1 Calling the servlet

When an internet user wishes to terminate his transaction, he can:

- Contact the Orange customer service (Internet)
- Contact the content provider customer service

The content provider can send to w-HA platform a refund request (with the transaction identifier (trxid) to be refund, and comments).

2.2.1.2 Example of calling the servlet

To refund a transaction, the Content Provider will call the servlet via his HTTP client (to be developed) by transmitting the necessary settings.

In order to do this he will use a URL:

http://*wha.marchand.com*/bundle/pos_request?
action=refund
&mid=5XXX
&trxid=6-17141248844587211
&url=https://wanadoo.w-ha.com/app-node-mct/responder
&userComment=commentaire client
&adminComment=commentaire editeur
&reasonCode=110

2.2.1.3 Calling parameters of the servlet

For refund request, the “pos_request” servlet is called with the following settings:

action=refund: to indicate to the servlet that this is a refund request.

mId=*shop identifier*: shop identifier affected by the refund request; the keyId values and the matching keyValue are collected by the servlet in the “merchants.xml” file.

trxid=*the transaction identifier*: transaction identifier to be refund.

url=*Url of the w-HA node (responder)*: the w-HA url node which will receive the refund request.
(ex: Orange (Internet) node is: https://wanadoo.w-ha.com/app-node-mct/responder)

userComment: internet user comment on the reason of his refund demand to the content provider.

adminComment: content provider comment on the reason of his refund demand to w-HA.

reasonCode:

Code (r=)	Comment
110	internet user refund demand (else)
111	internet user refund demand (multiple command)
112	internet user refund demand (multiple billing)
113	internet user refund demand (undelivery service)
114	internet user refund demand (transaction contested)
115	internet user refund demand (child transaction)
116	internet user refund demand (transaction due to a terminated subscription or an unwanted one)
117	Subscription terminated: I would like to be refund
118	The service does not correspond to my expectations
119	I subscribed many times to this single offer

2.2.1.4 Servlet functioning

When the servlet « pos_request » is called with the parameter action=refund, it does actions bellow:

- Gets information of the merchant in the « web.xml » file (merchantId, URLs, …)
- Gets parameters of the service (merchantId, keyId, keyValue) in the « merchants.xml » file
- Creates a securised URL (https) and signed, containing the transaction ID (trxid) to be refund, to the w-HA platform
- Sends an https request (in background) to the w-HA platform.

2.2.1.5 URL example created by the servlet

URL example of a refund request, created by the « pos_request » servlet:

```
https%3A%2F%2Fwanadoo.w-ha.com%2Fapp-node-
mct%2Fresponder%3Fm%3Dh%3Dfd4685944ae52d91601e343f7f561d3c%3Bp%3D13%3Bk%3D13
%3Bv%3D3%3A%7Bc%3Dm_fullRefund%3Bv%3D%7Badmincom%3DCommentaire%20editeur%3B
rid%3Drq004442%3BtrxId%3D6-
17141248844587211%3Breason%3D110%3Bd%3D0%3Busercom%3DCommentaire%20client%3B
%7D%7D
```

Decoded URL:

```
https://wanadoo.w-ha.com/app-node-mct/responder?
m=h=fd4685944ae52d91601e343f7f561d3c;
p=5XXX;
k=5XXX;
v=3:{
c=m_fullRefund;
v={
admincom=Commentaire editeur;
rid=rq004442;
trxId=6-17141248844587211;
reason=110;
d=0;
usercom=Commentaire client;}}
```

This URL is securised (https) and signed.

Information included in this URL are:

- a hmac to ensure message integrity (h=),
- the “m_fullRefund” (c=) command
- the transactionID to be refuned (trxId=)

2.2.1.6 *w-HA platform response to deferred confirmation request*

When the transaction identifier (**trxid**) **exists** and the transaction has a **status** which **allows its refund**, the w-HA platform returns:
The refund demand identifier: trxid=RR1234567891234567

If an error occurs, the platform returns a specific message based on the following nomenclature:

Message	Comment
h=a0e198418d3a05ed657c1cfce78c0964;p=5XXX;k=5XXX;v=2:{c=ex;v={m=INVALID_MERCHANT_INFO;t=com.ipin.core.api.node.pos.MerchantTransactionRefundManager$RefundException;c=2;}}	Refund fail
e=3	Wrong hmac
e=15	Error in the transmitted parameters (example : "trx**i**d=" instead of "trx**I**d=")

Example :
h=a0e198418d3a05ed657c1cfce78c0964;p=5XXX;k=5XXX;v=2:{c=ex;v={m=INVALID_MERCHANT_INFO;t=com.ipin.core.api.node.pos.MerchantTransactionRefundManager$RefundException;c=2;}}

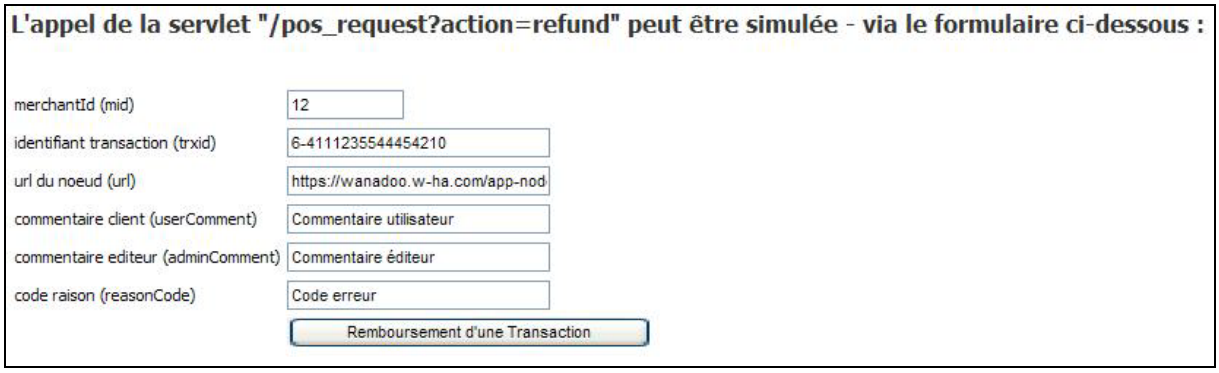
Note: If you try to refund **many times the same transaction**:
The w-HA platform is **not idempotent for refund requests**, for the first request the w-HA Kit will return the **refund demand identifier** (trxid=RR1234567891234567) then for followings request the w-HA platform will return:
"**h=a0e198418d3a05ed657c1cfce78c0964;p=5XXX;k=5XXX;v=2:{c=ex;v={m=INVALID_MERCHANT_INFO;t=com.ipin.core.api.node.pos.MerchantTransactionRefundManager$RefundException;c=2;}}**" (the refund is already effective).

2.2.1.7 *Example of using the servlet: web interface*

A web interface, which uses the “pos_request” servlet (action=refund) is also available to the Content Provider.

It enables the Content Provider:
- to properly understand how the servlet works
- to perform access controls manually if necessary

This web interface can be found a the following URL:
http://*wha.merchant.com*/demo/bundle/html/remboursement_transaction.html (*):
(*) replace wha.merchant.com by the IP adress or public domain name of the server on which the w-HA application is installed.



Warning!

Access to this page must be retricted to the content provider and **must NOT be avialable to public internet users.**

Otherwise they could refund transactions via this « demo » page !

3. WHAT THE MERCHANT MUST IMPLEMENT TO USE KIT V3.5

To integrate w-HA payment system to his platform, the content provider does not need complex knowledge in Java.
The content provider has to parameter the Kit and creates web pages.

3.1. "*.xml" configuration file settings

The Content Provider will have to set the configuration files of the « pos_init » servlet

web.xml
The "web.xml" file should only have to be set once by the Content Provider before going into production.

Products.xml
The merchant for each product addition, modification or deletion will modify the "products.xml" file.

Note: For each **modification** carried out in the "web.xml" or "products.xml" files (or even "server.xml"), it is necessary to stop and **restart the Servlet engine** in order to acknowledge the change.

3.2. Web pages

The content provider has to create web pages to present the product to the internet user. This page will call the pos_init servlet for the client to buy this product.

The content provider has also to manage specific cases:
For example when the internet user clics on cancel « annuler » on the billing panel or when the products is not present in the "products.xml" file.

When one of these events happens, the internet user is redirected to url set in the "web.xml" file:

```
<param-name>merchantHomeUrl</param-name>
<param-name>trxCancelFromPaymentPanelUrl</param-name>
<param-name>productUnavailableUrl</param-name>
```

So the content provider must create theses web pages set in the "web.xml".

3.3. Display ticketFun web pages in “current page”

Content provider side:

Due to browser evolution (Internet Explorer, Firefox, Google Chrome), and protection services (antivirus, pop-up killer, ….) more and more efficient, you must display Internet+ web pages (choice of the Internet Service Provider, authentication, billing panel) in “current page”.

From the billing panel, on your web site, the html code you have to use to call the billing servlet w-HA (pos_init) becomes:

```
<a href="/bundle/pos_bundle?action=authorizeOffer&oid=O1&wha_desc2=current">
<img src="../images/bt_validate_gd.jpg" border="0">          (1)
</a>
```

A classical href link is used.

The following html code could also be used:

```
<input type="image" src="../button_payment_internetplus.gif”
OnClick="window.location.href='
/bundle/pos_init?action=authorize&pid=P1&wha_desc2=current"> (1)
```

The method location.href allow the opening of the billing panel w-HA in the current page.
However, in that implementation, a javascript code is used to display screens (function OnClick and location.href).

The Kit redirects the internet user to"route.w-ha.com" with "**wha_desc2=current**" as additional parameter. Those parameters appear in merchant properties (mp) and "_ap_" is added before the parameter so we can read in mp: "**_ap_wha_desc2=current**"

Example:
https://route.w-ha.com/app-authorization/node?m=h=1ad42bde9f9812b062a2d11b79c00e83;p=5XXX;k=5XXX;v=2:{c=AuthorizeReq;v={ps=2;amt=0;mp={_ap_userId=fmoreau;pid=P1;**_ap_wha_desc2=current;**};pi=P1;pg=0;mUrl=http://localhost:8080/bundle/pos_init;pd=Produit#1;pc=Image;hr=1;cur=EUR;cl=-1;}}

w-HA platform side:

The w-HA server does a control on the parameter **wha_desc2=current**, to optimize the billing cinematic in « current page » mode.

So the merchant must associate the “wha_desc2=current” parameter to a billing cinematic in current page.

Warning:

You have to check that the new parameter "wha_desc2=current" is used in the hmac calculation when the client is redirected to :

- the **fulfillmenturl** (url parametered in tomcat/webapps/bundle/WEB-INF/prodcuts.xml)
- the **trxCancelFromPaymentPanelUrl** (url parametered in tomcat/webapps/bundle/WEB-INF/web.xml)

(1) The file « ../images/bouton_paiement_internetplus.gif » contains the picture of the internet+ billing button, which should be displayed on the merchant web site.
All usable buttons are available on www.internetplus.fr > « tool box »

Example of the Orange billing panel displayed in current page:

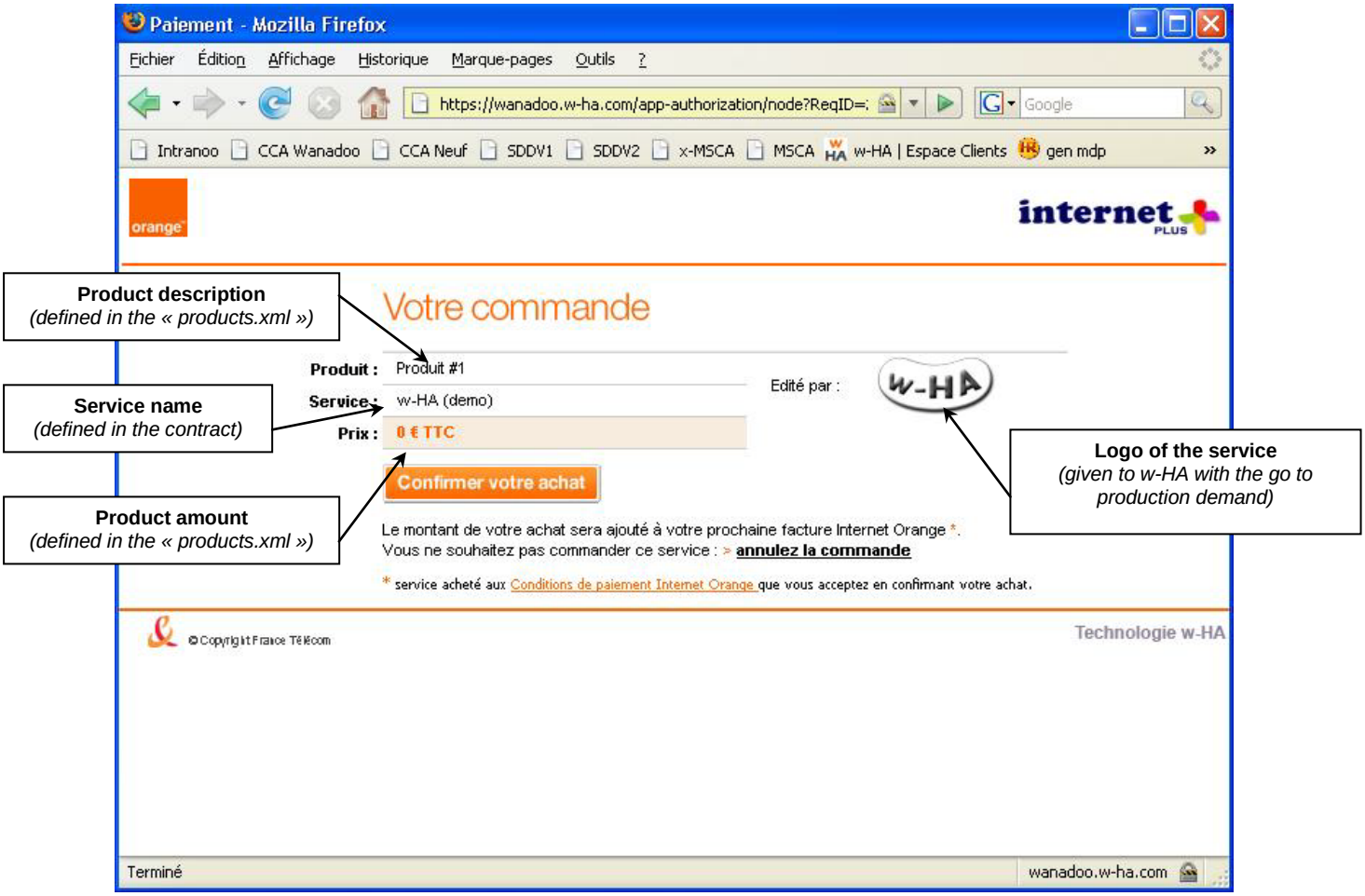


figure: display of the Orange billing panel

3.4. Content protection (using the hmac)

All information exchanged between the service Content Provider platform and the w-HA will be **signed** (HMAC-MD5) **which will guarantee platform authentification and the integrity of exchanged information.**

The electronic signature is based on a **shared symmetrical crypt encryption key** by the merchant platform and that of the w-HA: the **mctKey** (or **KeyValue**).

Once this information is sent, the "A" platform uses an algorithm to create a MAC (Message Authentification Code) based on the symmetrical key and its information, which will then be transmitted simultaneously to platform "B" in the URL.

Upon receipt of this information and the MAC, platform "B" uses the same algorithm to created another MAC based on the same symmetrical key and its information.

If it is identical to the one read, no changes to the information has been made: the transaction is authorized.
If it is different to the one read, the information has been modified: the transaction is denied.

To prevent the Internet User from using a service for free and accessing it directly through its URL (fulfillmentUrl), the following must be checked:
- he integrity of additional settings (calculating a HMAC)
- the value of one or several additional settings (ex: session identifier)

The HMAC is created using a HMAC MD5 algorithm with the following settings:
- the settings collected in the fulfillmentUrl (as a unique "string"),
- the value of the "mctKey" (or KeyValue) setting for the "merchants.xml" (as a unique "string"),

For example, if the return to the end URL (fulfillment Url) is done in the following manner:

http://marchand.com/payant/paiement_ok.php?sessionId=1234&commandeId=abcd&userId=toto&hmac=891284e23faa662c033a41dd9905cc10&trsId=6-5151292477228013

The program (here "paiement_ok.php") must check the integrity of settings by calculating the HMAC relative to these settings prior to checking their value and displaying the purchased product/service (or by refusing access if necessary).

If the "mctKey" setting value is "a1b2c3d4e5", then the HMAC calculation id done in the following manner:

my-hmac = H-MAC ("commandeId=abcd&sessionId=1234&trxId=6-5151292477228013&userId=toto","a1b2c3d4e5")

Caution 1:
The additional settings (or "merchant properties" or "mp") sent to the w-HA servlet must not contain any accents or special characters.

The URL encoding of these characters will result in an error in the HMAC calculation and in this case the product/service delivery will not be carried out even though the end user will be debited.

Caution 2:
The order of settings for consideration for the HMAC calculation by the w-HA application is an alphabetical order.

Caution 3:
The transaction identifier (trxid), collected on the fulfillementUrl, must also be taken into account when re-calculating the hmac.

A (basic) calculation example for HMAC in PHP:

If the fulfillment URL is:
http://marchand.com/payant/paiement_ok.php?
sessionId=1234&commandeId=abcd&userId=toto
&hmac=891284e23faa 662c033a41dd9905cc10&trxid=6-U5151292477228013

The hmac has to be calculated this way:

```
$hmac_verif =
bin2hex(mhash(MHASH_MD5,"commandeId=abcd&sessionId=1234&trxid=6-
U5151292477228013&userId=toto","a1b2c3d4e5f");
        if ($hmac_verif == $hmac)
        print "ok"; else
        print "no";
```

Check of the value of one (or many) additional parameters.

In order to prevent internet user to use many times the fulfillment (ffl) (and doing it access freely to the product/service), **the content provider should check that the "delivery" of a specific transaction has not been done yet.**

To do it the content provider can use the transaction identifier (trxid) which is unique or an additional parameter of his choice (calling the servlet for transaction authorization demand, see § 2.1.1.1) to check the delivery URL (ffl) uniqueness.

When the ffl is called the content provider can check that unique parameter (trxid or one parameter of his choice) defines a "delivery which was already done or not.

For the first call of the URL, the content provider redirects the client to the ffl.
If there are other calls, the content provider blocks the access to the ffl and display an error message.

4. ANNEXE I : TECHNICAL PRE-REQUIREMENTS FOR OPERATING KIT V3.5

Implementing the w-HA solution requires the installation of a Java "bundle" application on the merchant's server.

The following diagram illustrates the relationships between the different software and material components necessary for operation the w-HA "bundle" application.

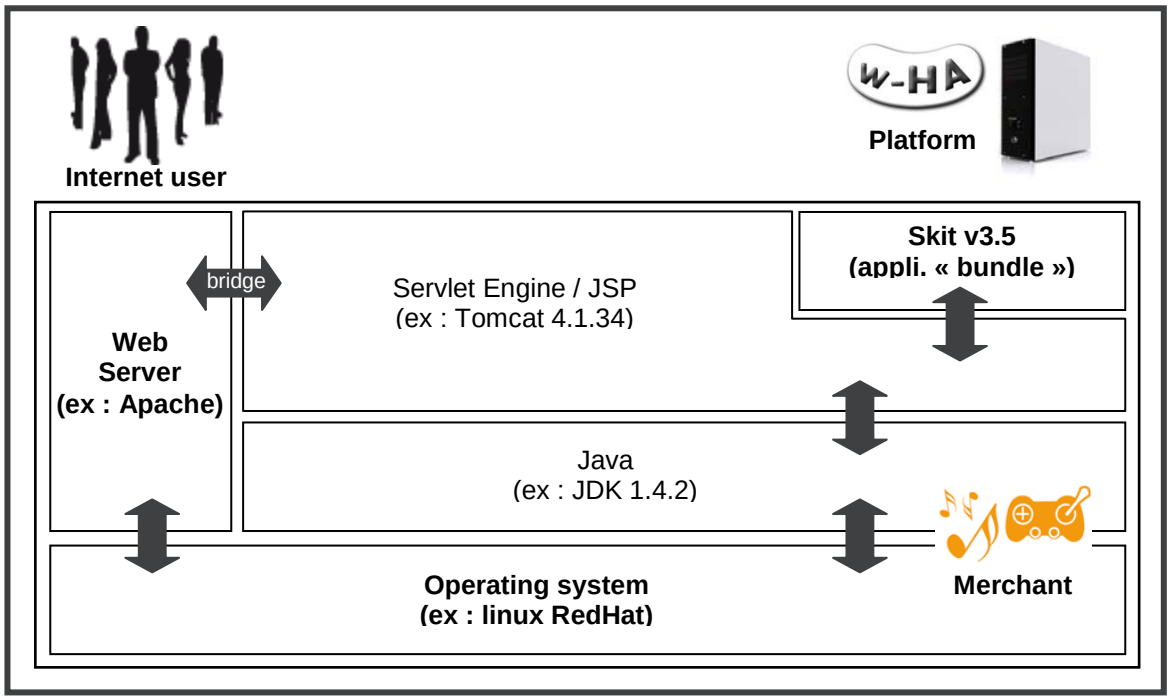


Figure 6 : Components necessary to the « bundle » application

The w-HA payment system operation requires installation of the "w-HA" (Java application) application on the service Content Provider hosting platform.

4.1. Hosting platform

The technical environment (operating system and web server) of the Content Provider's website hosting platform must be compatible with the "w-HA" application.

This application relies on the use of Java Servlets enabling compatibility with most platforms on the market.

4.1.1. Operating system

W-HA guarantees that Kit v3.5 will work properly and ensure technical support for the environments described below.

If the operating system is:

Sun Solaris version 2.6, version 2.7 or above, w-HA will ensure technical support for the web servers:

» Apache Web Server version 1.3.9 or above (with support for DSO modules)
If Apache is not installed, install a recent version of 1.3.x or 2.x.

Linux Kernel version 2.2.x (glibc/libc6 must be installed) w-HA will ensure technical support for the web servers:

» Apache Web Server version 1.3.9 or above (with support for DSO modules)
If Apache is not installed, install a recent version of 1.3.x or 2.x.

Windows NT version 4.0 (Service Pack 5 or above must be installed) w-HA will ensure technical support for the web servers:

» Apache Web Server version 1.3.9 or above (with support for DSO modules)
If Apache is not installed, install a recent version of 1.3.x or 2.x.
» Internet Information Server version 4.x (IIS4)

Windows 2000, w-Ha will ensure technical support for the web servers:

» Apache Web Server version 1.3.9 or above (with support for DSO modules)
If Apache is not installed, install a recent version of 1.3.x or 2.x.
» Internet Information Server version 5.x (IIS5)

Windows 2003, w-Ha will ensure technical support for the web servers:

» Apache Web Server version 1.3.9 or above (with support for DSO modules)
If Apache is not installed, install a recent version of 1.3.x or 2.x.
» Internet Information Server version 6.x (IIS6)

For other environments (other Operating System / Web Server couples), operating the w-HA application is quite possible, but in this case the installation of Java J2SDK 1.3.x version (or above) and the creation of the “bridge” between the web server and the Servlet engine will be performed by the merchant and will be his sole responsibility.

4.1.2. Java Virtual Machine

Java 2 SDK version 1.4.x or above must be installed prior to or during the integration.

W-HA recommends installing version “j2sdk1.4.x” (for example: j2sdk1.4.2).

Note:
If the platform is a Linux/UNIX type, the patches required for each component must be installed (for example, when installing j2sdk1.3.x on Solaris 7, the installation document specifies that patches No. xxxxx must be installed).

4.1.3. SSL protocol management module : “JSSE”

JSSE version 1.0.2 must be installed prior to or during the integration.

Note:
JSSE is included by default in J2SDK Java versions 1.4.x and above.
Installation is therefore not necessary.

4.1.4. Servlet engine

A Servlets/JSP engine complying with the following SUN requirements must be installed prior to or during the integration:

- JavaServlet 2.2
- JavaServlets Pages (JSP) 1.1

W-HA recommends the Servlets/JSP engine "Tomcat 4.1.x" (for example: Tomcat 4.1.34).

If the Servlets/JSP engine is Tomcat, the Web server must be one of the following:

- Apache Web Server version 1.3.9 or above (with support for DSO modules). If Apache is not installed, install a recent version of 1.3.x or 2.x
- Internet Information Server 4 - (IIS4)
- Internet Information Server 5 - (IIS5)
- **For all other Web Servers, or if the merchant's Servlet engine is not Jakarta-Tomcat**, operating the w-HA application is quite possible on the condition that the Servlet's engine complies with the SUN requirements cited above.

In this case, integrating the application within the Servlet engine is the merchant's sole responsibility. W-HA ensures technical support for the w-HA application operation and settings.

4.2. Network Considerations

For Kit v.35 to work, some network equipment settings must be considered (port opening, firewalls, proxies)

4.2.1. During the w-HA application installation

The w-HA application-hosting platform must be:

- **accessible** from the outside by the **existing Web Server port** (by default: port 80)
- if possible, accessible from the outside by the Servlet Engine port (by default: port 8080)
- **Can initiate a SSL communication (on port 443) with the w-HA production platform**
- The different network equipments (Firewalls, Proxies,…) must therefore be properly configured prior to installation to allow access as mentioned above.

4.2.2. In production

The w-HA application-hosting platform must be:

- accessible from the outside by the existing Web Server port (by default: port 80)
- Can initiate a SSL communication (on port 443) with w-HA platform nodes:

Domain Name	Operator	IP	Port	Request type
https://wanadoo.w-ha.com	Orange Internet	193.28.205.2	443	Out going
https://free.w-ha.com	Alice/Free	193.28.205.52	443	Out going
https://sfr.w-ha.com	SFR	193.28.205.17	443	Out going
https://cb.w-ha.com	Orbeo	193.28.205.21	443	Out going
https://qualif-marchand.w-ha.com	w-HA tests web	193.164.148.75	443	Out going

The different network equipments (Firewalls, Proxies, ...) must therefore be properly configured prior to installation to allow access as mentioned above.

Note:
To know the IP address of a domain name with windows (in start>execute>cmd) you can use the nslookup commande (ex: nslookup wanadoo.w-ha.com)

4.3. Other pre-requisites for integration

4.3.1. Decompression Utility program

As the Kit v3.5 application is delivered in "zip" or ".tar.gz" form, a decompression utility program such as WinZip, unzip or tar is required.

4.3.2. Web Server re-boot

Application installation requires a modification of the server(s) configuration (Http server and Servlet Engine).
These modifications require a re-boot of the server(s), or even of the machine in some cases.

4.3.3. System administrator presence

During the application installation and the Web server restart, the hosting platform system administrator's presence ("root" rights) is required.
(1/2 to 1 Day.)

5. ANNEXE II : COMPONENTS OF THE KIT V3.5

Kit v3.5 uses the following components for transaction payments:

Application « bundle »	
Servlet « pos_init »	Manages: - transaction requests on the w-HA platform - transaction request replies - transaction confirmation
Configuration file « web.xml »	Contains overall and default settings for Kit v3.5
Configuration file « products.xml »	Contains the Product Catalogue
Log file « authorization.txt »	Writes the information regarding transaction requests
Log file « logs.txt »	Writes information on transactions validated by the w-HA platform

Application « demo »	
/demo/acte/html/index.html	Enables to check if the "pos_init" servlet works for a transaction.

Note: When the content provider completes **settings in ".xml"** files, he can **add only one parameter in urls defined**.

Here is the list of urls to be set in ".xml" files :

» in tomcat/wepapps/bundle/WEB-INF/web.xml :
- trxCancelFromPaymentPanelUrl
- merchantHomeUrl
- productUnavailableUrl

» in tomcat/wepapps/bundle/WEB-INF/products.xml :
- fulfillmentUrl

Note:
If the url contains 0 parameter:
(example of an url set in an ".xml" file :
http://wha.marchand.com/page.php)
Then at the time of the redirection to the set url, parameters are added after a "?" and are separated by "&".

If the url contains 1 parameter:
(example of an url set in an ".xml" file :
http://wha.marchand.com/page.php?parameter1=abc)
Then at the time of the redirection to the set url, parameters are added after a "&" and are separated by "&".

If the url contains 2 parameters or more:
(example of an url set in an ".xml" file :
http://wha.marchand.com/page.php?parameter1=abc&parameter2=cba)
Then there is an error 500 (IllegalArgumentException) because the Kit can not read the url.

In order to avoid theses error due to settings, content providers should add **additional parameters when they call the Kit** and **not in urls set in ".xml" files**.

5.1. Configuration file “web.xml”

In the configuration file “web.xml”, elements used for the single one-off payment are defined between tags:

```
<servlet>
        <servlet-name>servlet_pos_init</servlet-name>
        <servlet-class>com.ipin.core.merchant.skit.WebAuthorizationServlet</servlet-class>
        …
</servlet>
```

5.1.1. Structure of the "web.xml" file : servlet "pos_init"

Parameter	Comment
merchantLogDir	**access path to authorizations.txt and logs.txt (do not modify)** « authorization.txt » logs events regarding authorization debit demands « logs.txt » logs events regarding debit or «Confirm»
MerchantId	**Shop identifier used by default (supplied by w-HA).** It enables the w-HA platform to identify the shop. There is a mctid for the test platform (qualify-marchand.w-ha) and a mctid for the production platform (route.w-ha.com)
KeyId	**Secret key ID (supplied by w-HA)** The key value is shared between Kit v3.5 and the w-HA platform. It is connected to the "mctKey" and ensures information integrity There is a keyid for the test platform (qualify-marchand.w-ha) and a keyed for the production platform (route.w-ha.com)
KeyValue	**Secret key value (supplied by w-HA)** The key value is shared between Kit v3.5 and the w-HA platform. It is connected to the "mctKeyId" and ensures information integrity. There is a keyvalue for the test platform (qualify-marchand.w-ha) and a keyvalue for the production platform (route.w-ha.com)
NodeAuthorizationUrl	**URL of the w-HA platform node (supplied by w-HA)** Test URL: https://qualif-marchand.w-ha.com/app-authorization/node Production URL: https://route.w-ha.com/app-authorization/node
MerchantLanguage	**Content provider language: fr (do not modifiy)**
MerchantCurrency	**Merchant currenty: EUR (do not modify)** The Euro is the only currency accepted on the w-HA platform for Internet+. The Content Provider currency must therefore by set to the EUR value. The offer amount is displayed in Euros on the w-HA payment panel.
MessageUrl	**Access path to JSP that manages the html format error messages (do not modify)** The different error cases are sent to the JSP "message.jsp" in the "bundle" application.
	For the 5 parameters bellow : If http requests are get by Tomcat, you must precise the port: (example : http://www.marchand.com:8080/acte/pos_init) If http requests are get by Apach or IIS, you must not precise the port: (example : http://www.marchand.com/acte/pos_init)
MerchantUrl	**URL where the "pos_init" servlet is located on the merchant server (to be modified)** URL towards which the user is redirected after accepting the transaction on the w-HA payment panel. Adapt with the public IP address or the server public domain name where the "bundle" application is installed.
MerchantHomeUrl	**Content provider home Page URL (to be modified)** In case of error, the JSP « message.jsp » redirects the internet user to « merchantHomeUrl »
TrxCancelFrom PaymentPanelUrl	**URL to redirect the client if he clicks on the cancel button on the billing panel.**
ProductUnavailableUrl	**Backward URL if the product in not available (to be modified)** If the product (pid) is not defined in the « products.xml » file, the internet user is redirected to this URL.
XmlProductDatabase	**Access path to the "products.xml" file (do not modify)** The "productbundle.xml" file contains the Content Provider's offer.
TimestampDelay	**Maximum delay for offer confirmation (in milliseconds, do not modify)** Maximum delay between the transaction request and the transaction confirmation.
ContentProtectionUrl	NOT USED (do not modify)
ContentProtectionKey	NOT USED (do not modify)
ServiceUnavailableUrl	NOT USED (do not modify)

5.1.2. Example of a "web.xml" file

```
<servlet>

        <servlet-name>servlet_pos_init</servlet-name>
        <servlet-class>com.ipin.core.merchant.skit.WebAuthorizationServlet</servlet-class>
        <init-param>
                <param-name>merchantLogDir</param-name>
                <param-value>c:\Tomcat\webapps\bundle\WEB-INF\logs\</param-value>
        <description></description>
        </init-param>
        <init-param>
                <param-name>merchantId</param-name>
                <param-value>5XXX</param-value>
                <description>CUSTOMIZE: Provide the correct merchant id (numeric).</description>
        </init-param>
        <init-param>
                <param-name>keyId</param-name>
                <param-value>5XXX</param-value>
                <description>CUSTOMIZE: Provide the correct keyId (numeric).</description>
        </init-param>
        <init-param>
                <param-name>keyValue</param-name>
                <param-value>abcdefghijklmnopqrstuvwxyz123456</param-value>
                <description>CUSTOMIZE: Provide the correct key.</description>
        </init-param>
        <init-param>
                <param-name>nodeAuthorizationUrl</param-name>
                <param-value>https://route.w-ha.com/app-authorization/node</param-value>
                <description>CUSTOMIZE: Provide the correct host name.</description>
        </init-param>
        <init-param>
                <param-name>merchantLanguage</param-name>
                <param-value>fr</param-value>
        </init-param>
        <init-param>
                <param-name>merchantCurrency</param-name>
                <param-value>EUR</param-value>
        </init-param>
        <init-param>
                <param-name>merchantUrl</param-name>
                <param-value>http://wha.marchand.com/bundle/pos_init</param-value>
                <description>CUSTOMIZE: Provide the correct host name.</description>
        </init-param>
        <init-param>
                <param-name>messageUrl</param-name>
                <param-value>/bundle/jsp/message.jsp</param-value>
        </init-param>
        <init-param>
                <param-name>merchantHomeUrl</param-name>
                <param-value>http://www.marchand.com/demo/bundle/html/index.html</param-value>
                <description>CUSTOMIZE: Provide the correct host name.</description>
        </init-param>
        <init-param>
<param-name>trxCancelFromPaymentPanelUrl</param-name>
<param-value>http://www.marchand.com/demo/bundle/html/panel_cancel.html</param-value>
<description>CUSTOMIZE: Provide the correct host name.</description>
        </init-param>
```

```
        <init-param>
<param-name>productUnavailableUrl</param-name>
<param-value>http://www.marchand.com/demo/bundle/html/productUnavailable.html</param-value>
<description>CUSTOMIZE: Provide the correct host name.</description>
        </init-param>
        <init-param>
                <param-name>xmlProductDatabase</param-name>
                <param-value>/WEB-INF/products.xml</param-value>
        </init-param>
        <init-param>
                <param-name>timestampDelay</param-name>
                <param-value>300000</param-value>
        </init-param>
        <init-param>
                <param-name>serviceUnavailableUrl</param-name>
                <param-value>NON UTILISE - NOT USED</param-value>
                <description>NON UTILISE / NOT USED</description>
        </init-param>
        <init-param>
                <param-name>contentProtectionUrl</param-name>
                <param-value>NON UTILISE - NOT USED</param-value>
                <description>NON UTILISE / NOT USED</description>
        </init-param>
        <init-param>
                <param-name>contentProtectionKey</param-name>
                <param-value>NON UTILISE - NOT USED</param-value>
                <description>NON UTILISE / NOT USED</description>
        </init-param>

        <load-on-startup></load-on-startup>

</servlet>

<servlet-mapping>
        <servlet-name>servlet_pos_init</servlet-name>
        <url-pattern>/pos_init</url-pattern>
</servlet-mapping>
```

5.2. Configuration file “products.xml”

All products sold by the content provider are listed in the “products.xml” file.

The “pos_init” servlet gets information of the product in the “products.xml” file thanks to the “productId”.

The **number of products** in the “products.xml” file is **not limited**.

5.2.1. Structure of the "products.xml" file

Parameter	Comment
ProductId	**Product identifier (alphanumerical, maximum 50 characters, unique for each product)** Unique product identifier, transmitted as a parameter to the Javascript function « authorize » on the merchant web page.
Amount	**Product price (in euros, the decimal separator is a dot, for example : 1.00)** The amount of the product is displayed on the billing panel.
Description	**Product description (maximum 200 characters, no special characters)** The product description is displayed on the billing panel.
Category	**Product category (maximum 50 characters, no special characters)** Can be used by the merchant to do statistics. This information appears in logs.
Class	**w-HA product class (-1=generalist, 1=adult, 2=money games)** Used by internet user to forbide them selves access to specific web pages.
FulfillmentUrl	**URL for transaction acknowledgement (to be modified)** The Content Provider URL towards which the client is redirected after confirmed the transaction.
PaymentGuaranteed	**Payment guaranteed : false (do not modify – contractual clause)** The merchant accepts all internet users.
IPINHandleRefund	**Refund requests managed by w-HA : true (do not modify – contractual clause)** w-HA, and not the merchant, decides to refund or not a client.
AutoConfirm	**Confirm (debit request) automatic : true (do not modify)** The kit automatically does the "Confim" (effective debit request) as soon as it receives the authorization request.
paymentService	**Payment service type accepted by the merchant : 2 (do not modify – contractual clause)** NOT USED

5.2.2. Example of a "products.xml" file

```
<product-list>

<product>
        <productId>P1</productId>
        <amount>1.00</amount>
        <description>Produit #1</description>
        <category>Image</category>
        <class>-1</class>
        <fulfillmentUrl>http://www.marchand.com/demo/bundle/html/produit1.html</fulfillmentUrl>
        <paymentGuaranteed>false</paymentGuaranteed>
        <iPINHandleRefund>true</iPINHandleRefund>
        <autoConfirm>true</autoConfirm>
        <paymentService>2</paymentService>
</product>

</product-list>
```

5.3. Logs file

5.3.1. File “authorization.txt”

Access path to the log file “authorization.txt” is defined in the “web.xml” file of the Kit: **merchantLogDir**

It is an absolute path.
By default: [TOMCAT_HOME]/webapps/bundle/WEB-INF/authorization.txt

It is a text file (*.txt) which is updated (a new line is added) for each authorization request received by the “pos_init” servlet.

Caution !

If many requests are received, the size of this file may be important (several MB) and it might lower the performances’ application.
So the setting up of rotating logs advised.

So this log file is updated each time an internet user on the billing panel clicks:
- Either on the “confirmer votre achat” button (OK)
- Either on the “annuler” button (cancel)

5.3.2. Structure of the “authorization.txt” file

Each line is composed by the following fields:

Fields	Description	Type / Format
Transaction ID	Unique transaction identifier	X-XXXXXXXXXXXXXXX where X are numbers
Status	Transaction status	AUTHORIZED=1 CANCELLED=2 CONFIRMED=3 REFUNDED=5 REFUND DENIED=7
Reply URL	w-HA platform URL used for confirm and cancel.	URL type https://XXXXX.w-ha.com/app-node-mct/responder
Time stamp	Purchase date and time	Format : yyyy-mm-dd hh:mm:ss
Description	Product description	(see “products.xml” file)
Merchant ID	Merchant unique identifier	(see “web.xml” file)
Currency	Product currency	(see “products.xml” file) ISO code of 3 lettres of the currency. You can only have one currency type in a "products.xml" file.
Merchant Amount	Product amount	(see “products.xml” file)
User City	City of the client	Alphanumeric
User State	State of the client	Code : 2 or 3 letters of the state
User Zip Code	Zip code of the client	Numeric
User Country	Country of the client	Alphanumeric
Key ID	NOT USED	NOT USED

5.3.3. Example of the "authorization.txt" file

Each new transaction request adds a new line containing different fields separated by the "Pipe" character ('|'):

6-7672821718212150|3|https://wanadoo.w-ha.com/app-node-mct/responder|86400|2008-11-06 16:38:45|Produit #1|505|EUR|0.0|PARIS|FR|75007|FR|505
6-587456287965542|3|https:// wanadoo.w-ha.com/app-node-mct/responder|86400|2008-11-06 16:39:07|Produit#3|505|EUR|0.0|PARIS|FR|75007|FR|505
8-452596245485771|3|https://club-internet.w-ha.com/app-node-mct/responder|86400|2008-11-19 17:25:33|Produit #1|505|EUR|0.0|PARIS|FR|75007|FR|505

5.3.4. "logs.txt" file

Access path to the log file "logs.txt" is defined in the "web.xml" file of the Kit: **merchantLogDir**

It is an absolute path.
By default: [TOMCAT_HOME]/webapps/bundle/WEB-INF/log_response_XXX.txt

It is a text file (*.txt) which is updated (a new line is added) for each confirmation request sent by the "pos_init" servlet. Additional parameters have the prefix "_ap_" in "logs.txt".

Caution !

If many requests are received, the size of this file may be important (several MB) and it might lower the performances' application.
So the setting up of rotating logs advised.

5.3.5. Structure of the "logs.txt" file

Each line is composed by the following fields:

Fields	Description	Type / Format
Time stamp	Purchase date and time	Format : yyyy-mm-dd hh:mm:ss
Status	Transaction status	AUTHORIZED CANCELLED CONFIRMED REFUNDED REFUND DENIED
Transaction ID	Unique transaction identifier	X-XXXXXXXXXXXXXXX where X are numbers
User State	State of the client	Code : 2 or 3 letters of the state
User Zip Code	Zip code of the client	Numeric
User Country	Country of the client	Alphanumeric
User City	City of the client	Alphanumeric
_ap_XXX	Merchant additional parameter and its value	XXX is the name of the additional parameter
pid=	Unique product identifier	(see "products.xml" file)
ts=	Purchase date and time	Format : yyyy-mm-dd hh:mm:ss.ccc

5.3.6. Example of the “logs.txt” file

Each new transaction request adds a new line containing different fields separated by the "Pipe" character (‘|’):

Wed Feb 06 16:38:45 CET 2008|Confirm|6-7672821718212150|86400||75007|FR|PARIS|pid=P1|ts=2008-11-06 16:38:35.865
Wed Feb 06 16:39:07 CET 2008|Confirm|6-8541452112494723|86400||75007|FR|PARIS|pid=P3|ts=2008-11-06 16:38:58.648
Tue Feb 19 17:25:33 CET 2008|Confirm|6-5634298321514357|86400||75007|FR|PARIS|_ap_V=1D|_ap_URI=/pages/cuisine/recette.cfm?ID=278|pid=P1|_ap_Nb=2|ts=2008-11-19 17:24:16.956