T-Mobile m-platba (MAMI)
Implementation manual for Merchants

Version: 03 (2010/02/04)

Contents

1 Introduction

This is implementation manual for T-Mobile **m-platba** payment interface (also called MAMI).

1.1 Revision history

Version	Date	Changes
04	2011/03/21	• Information about limited character sets in transaction description • Description for result code -1 in MerchantNotification

2 What is m-platba?

M-platba allows users to make payments on the internet – in e-shops, on-line gaming sites, sport betting companies and so on. Payment is cleared to user's next mobile phone bill (in case of post-paid customer) or substracted from user's credit (pre-paid, credit user). Merchant is receiving transaction details in real-time – T-Mobile confirms the transaction to Merchant instantly.

Main features of m-platba payment system:

- **Anonymity** – merchant doesn't know user's identity (MSISDN, T-Zones name, ...) – if you need that information from user, you have to ask the user for it.
- **Security for merchant** – communication between MAMI and Merchant is done in secured way, user can't change transferred information; T-Mobile confirms the transaction to merchant instantly and merchant can be sure for receiving paid amount.
- **Security for user** – user enters his/her username and password on T-Mobile's secure payment portal. No personal information / transaction authorization is entered on merchant's web site.

There are two basic types of payments in m-platba: **standard payment** and **subscription**.

2.1 Standard payment

Standard payment is like **bank wire transfer** from one account to another. Customer confirms transfer of amount X to merchant's account Y.

Standard payment scenario:
1) Merchant create transaction request in m-platba system
2) Merchant redirects user's browser to T-Mobile's secure payment portal
3) Customer signs on the payment portal and confirms the payment
4) Payment portal redirects user back to merchant's web site
5) Merchant finalize the transaction in m-platba

2.2 Subscription

Merchant can create subscription in M-Platba defined by maximal transaction amount, maximal month summary amount and expiration date, and redirect user to Payment Portal. User confirms the subscription on Payment Portal and returns to merchant's web site.
Merchant can request an payment (within confirmed subscription parameters) subsequently, without further interaction with user. It is simple: merchant calls WebServices interface and payment is done – with no delay, no user confirmation, no logged-in user on merchant's web.

Scenarios that can be implemented with subscriptions (samples):

- **Common "subscription"**: User authorize the merchant to transfer 300 CZK monthly for one year. Merchant then call "pay from subscription" WebService at first day of every month for this user and top up the user's account on game server.

- **Easy micropayment for books, MP3s etc:** User authorize the merchant to transfer 500 CZK per month for one year and goes shopping in merchant's e-shop with e-books, MP3s, videos and so on. When user clicks on "I want this" icon, merchant will call "pay from subscription" WebService - so transfer is done quickly and without inconvenience to user – and serves the requested commodity. (This scenario is similar to shopping experience at Amazon – user has to register his credit card and then user can buy e-book by clicking on it from Kindle reader without any confirmation.)

There are three use-cases with subscription:

- **creation** of subscription (with confirmation from user)
- **payment** from subscription (without further interaction with user; more than one payments can be done from one subscription)
- **termination** of subscription

Creation of subscription is the similar process as simple payment:
1) Merchant create subscription request in m-platba system
2) Merchant redirects user's browser to T-Mobile's secure payment portal
3) Customer signs on the payment portal and confirms the subscription
4) Payment portal redirects user back to merchant's web site
5) Merchant check the subscription status in m-platba

Payment from subscription is really simple: merchant calls WebServices interface "pay from subscription" and instantly receive the result (whether the transaction has been successful or not). There is no interaction with customer, no need of further confirmation, no need of user's presence on merchant's web site. Customer is **notified** by SMS message with transaction details and subscription expiration information.

Subscription **ends automatically** on its expiration date. Subscription **can be cancelled** by merchant (using WebServices) or by user (in T-Zones self-care).

3 M-mplatba interfaces and tools available for Merchants

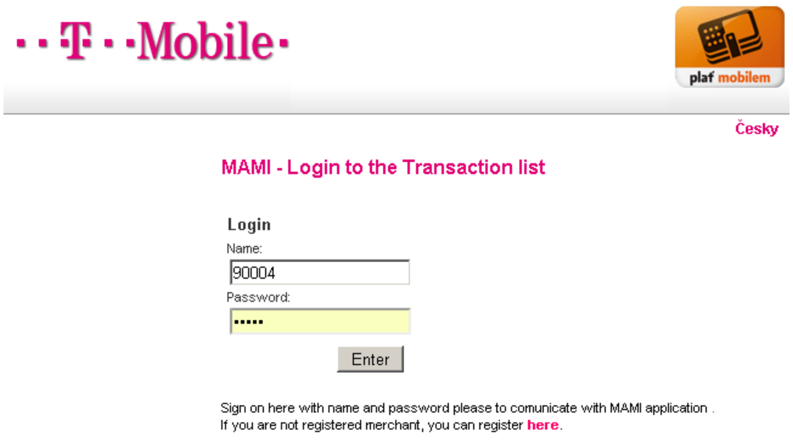
M-platba offers the following tools and interfaces for Merchants:

- WebServices interface for both standard payments and subscriptions
- Simple payment interface ("payment button") for the easiest integration of m-platba into merchant's web site
- Machine-readable transaction list
- "Merchant View" web application – you can get transaction list and set some options here
- E-mail notification about successful payments

4 “Merchant View” – web access for merchants

“Merchant View” application allows you to list transactions and set some options.

“Merchant View” application can be found on this address:
https://m-platba.t-mobile.cz:9447/MerchantView/brana1.1.1eng.html
You need merchant identification number (5-digits number) and password for sign-in.



Česky

MAMI - Login to the Transaction list

Login

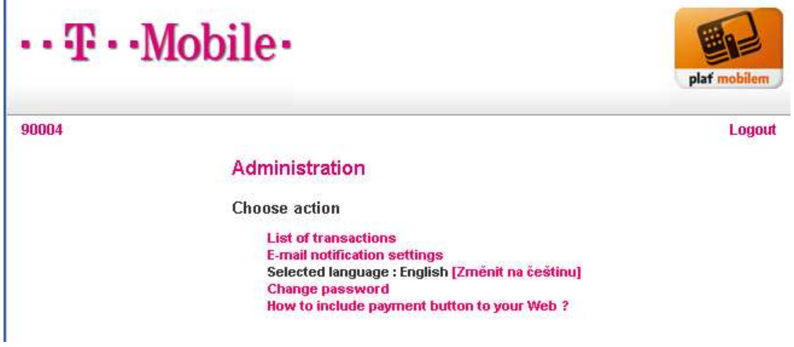
Name: 90004

Password: •••••

Enter

Sign on here with name and password please to comunicate with MAMI application .
If you are not registered merchant, you can register here.

Main menu is shown after successful login:



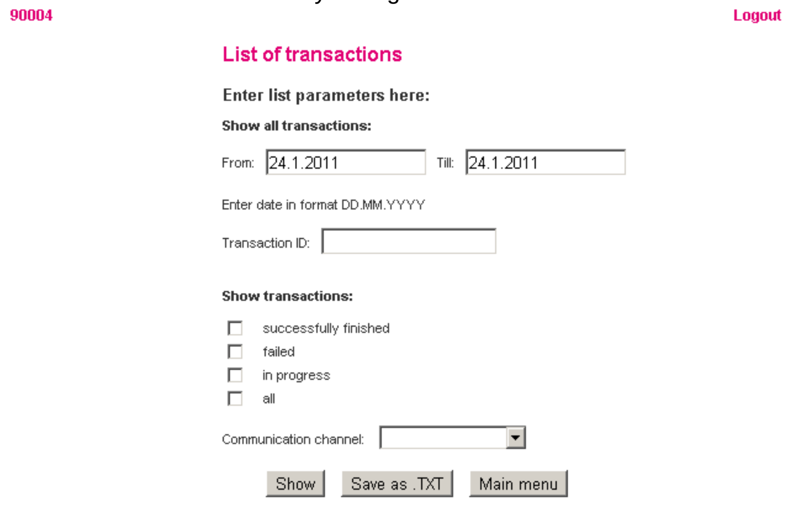
90004 Logout

Administration

Choose action

List of transactions
E-mail notification settings
Selected language : English [Změnit na češtinu]
Change password
How to include payment button to your Web ?

“**List of transactions**” allows you to get filtered list of transactions:



90004 Logout

List of transactions

Enter list parameters here:

Show all transactions:

From: 24.1.2011 Till: 24.1.2011

Enter date in format DD.MM.YYYY

Transaction ID:

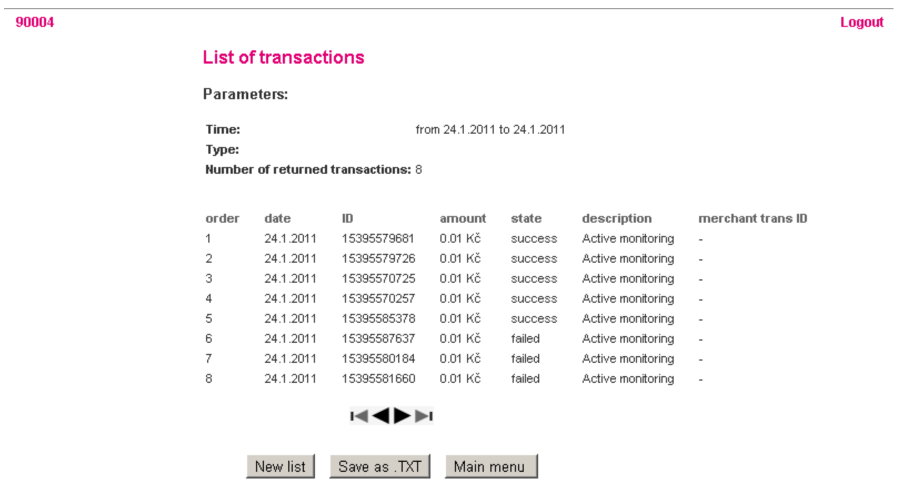
Show transactions:

- ☐ successfully finished
- ☐ failed
- ☐ in progress
- ☐ all

Communication channel:

Show | Save as .TXT | Main menu

„**Save as .TXT**“ exports the filtered transaction list into text file; “**Show**” will show the transaction list in web page:



90004 Logout

List of transactions

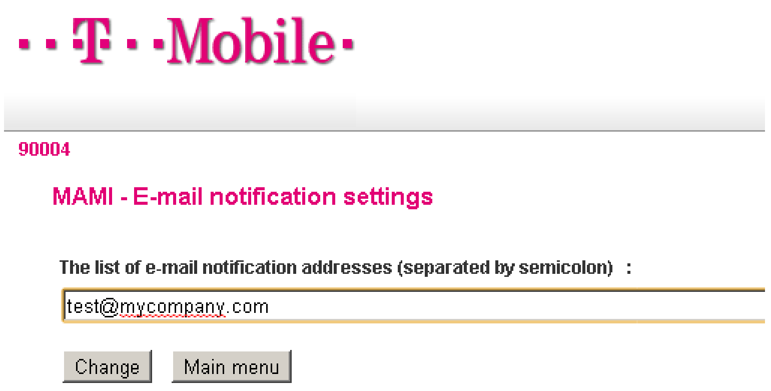
Parameters:

Time: from 24.1.2011 to 24.1.2011
Type:
Number of returned transactions: 8

order	date	ID	amount	state	description	merchant trans ID
1	24.1.2011	15395579681	0.01 Kč	success	Active monitoring	-
2	24.1.2011	15395579726	0.01 Kč	success	Active monitoring	-
3	24.1.2011	15395570725	0.01 Kč	success	Active monitoring	-
4	24.1.2011	15395570257	0.01 Kč	success	Active monitoring	-
5	24.1.2011	15395585378	0.01 Kč	success	Active monitoring	-
6	24.1.2011	15395587637	0.01 Kč	failed	Active monitoring	-
7	24.1.2011	15395580184	0.01 Kč	failed	Active monitoring	-
8	24.1.2011	15395581660	0.01 Kč	failed	Active monitoring	-

(Notice: there is neither MSISDN nor any other identification of customer)

If you want to receive an e-mail after every successful transaction, you can set this in “**E-mail notification settings**” item in main menu:



90004

MAMI - E-mail notification settings

The list of e-mail notification addresses (separated by semicolon) :

test@mycompany.com

5 Machine-readable transaction list

Do you want to read transaction list programmatically? Do you want to do reconsolidation between your transaction list and transaction list on M-Platba side? You can use **machine-readable transaction list**.

It is simple – just send a HTTP GET request for defined URL and you'll receive plain text file with transaction list in defined format.

You can request one of four **time filters**:

- today's transactions: https://m-platba.t-mobile.cz:9447/MerchantView/TransList/List.jsp?relative=0
- yesterday's transactions: https://m-platba.t-mobile.cz:9447/MerchantView/TransList/List.jsp?relative=1
- transactions from date YYYYMMDD to date YYYYMMDD: https://m-platba.t-mobile.cz:9447/MerchantView/TransList/List.jsp?from=20100901&to=20101231)
- transactions from the last N minutes (120 used as sample): https://m-platba.t-mobile.cz:9447/MerchantView/TransList/List.jsp?last=120

HTTP Basic authentication is required – with the same username/password you're using in Merchant View.

Transaction list has the following structure:

```
FROM:2010-09-01 00:00:00;TO:2010-12-31 23:59:59
15397593133;2010-09-01 15:02:05;failed;17,0;www;-;test
15397599560;2010-11-19 11:34:46;success;10,35;www;001;item1
15397599687;2010-11-19 12:23:51;failed;10,0;subscription;-;testovaci_predplatne1
TOTAL:3
```

First line contains information about time filter used; last line displays count of returned records.

The body of report contains one transaction per line. Decimal separator is comma (,), item separator is semicolon (;).

Items within one transaction record:

- Transaction ID in M-Platba
- Timestamp YYYY-MM-DD hh24:mi:ss
- Status: failed / success
- Amount
- Channel used by user - www, wap, subscription
- Transaction ID on merchant's side, or dash (-) if not specified by merchant
- Transaction description

6 How to connect our system to m-platba? Which interface should I use?

M-platba offers you two different interfaces – WebServices and “simple HTML button”. WebServices are much more powerful and much more secure, but they can be a little difficult to implement, especially if your web is done in PHP or old ASP. “Simple HTML button” is (as the name suggest) really simple to use, but it doesn’t offer full scale of m-platba services.

The following table is here to help you with the technology selection:

What I want to do?	**How to do it?**
I want to extend web site of our charity (civic association, foundation and so on) with simple “*donate us some money*” button.	Insert “**simple HTML payment button**” into your web page with fixed payment amount (alt. more buttons with different payment amounts or button with editable amount). As the “target page” for successful transaction give the URL of “thank you for donation” page. Set up *e-mail notification about payment* in “Merchant View” application so you’ll be notified about received transactions. The whole implementation is done **without any programing**.
I have a small e-shop and want to allow customers to pay by m-platba. I have a little number of transactions monthly, so I preffer a **simple** (and cheap) **implementation**.	Treat with m-platba in the same way as with money transfer. Add a new payment method in your e-shop – T-Mobile m-platba. Use “**simple HTML payment button**”. Fill the transaction amount with amount the customer has to pay; insert order ID into “MerchantTrans” field. Use URL of “your payment has been received, thank you” page as “ReturnUrlOk” field. Set up *e-mail notification about payment* in “Merchant View” application so you’ll be notified about received transactions. When a new transaction came in, look into transaction list in “Merchant View”. Get an order ID from transaction list (the last column, “merchant trans ID”). Check the transaction amount. If everything is all right, mark the payment in your e-shop as “paid” and proceed the order (send goods, provide services...). Implementation is done with **minimal effort**. You only have to insert payment button on “payment method selection” page with transaction amount and order ID.
I have a small e-shop and want to allow customers to pay by m-platba. Payment processing should work **automatically**, with no user interaction on my side. But: • My e-shop is written in PHP (or plain ASP, or any other technology with limited features). or • I have no experiences with WebServices. or • My e-shop software doesn’t support WebServices	Use “**simple HTML payment button**”. Fill the transaction amount with amount the customer has to pay; insert order ID into “MerchantTrans” field. Create new page which will be used as landing page for paid transactions (“ReturnUrlOk” field in payment button). This page will receive m-platba transaction ID as one of parameters. It will call the **machine-readable transaction list** service (for “last 20 minutes” list) and search the result for the given transaction ID. It will check the transaction amount, status (success/failure) and order ID from your e-shop. If everything is OK, it will mark the order as paid and deliver the goods/services. Implementation is done with a little effort and with only the basic programming skills, without usage of “rocket-science” technologies.
My e-shop is written in **.Net** or **Java** environment. I use **WebServices** routinelly.	Use the WebServices payment interface. The payment process will be fully automatic.
I want to use **subscriptions**.	User **WebServices** interface. No other interface is available for subscriptions.

7 Simple HTML payment button - implementation

If you want to use the simple payment button, just enter the following code into your web page:

```
<form action="https://m-platba.t-mobile.cz/spp/simple/pay.jsp" method="post">
<input type="hidden" name="IdMerchant" value="90004"/>
<input type="hidden" name="Description" value="Donation for charity ABCDEF"/>
<input type="hidden" name="Amount" value="50.0"/>
<input type="hidden" name="ReturnUrlOk" value="http://www.ABCDEF.cz/pay_simple_ok.php"/>
<input type="hidden" name="ReturnUrlErr" value="http://www.ABCDEF.cz/pay_simple_err.php"/>
<input type="hidden" name="MerchantTrans" value="-">
<input type="hidden" name="origin" value="www.ABCDEF.cz"/>
<input type="hidden" name="lang" value="cz"/>
<input type="submit" value="Donate 50 CZK via T-Mobile m-platba"/>
</form>
```

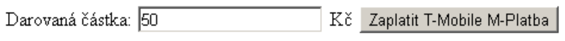
Fields description:

Field	**Comment**
IdMerchant	Your merchant’s ID in m-platba.
Description	Transaction description – will be displayed to customer
Amount	Transaction amount, in CZK. Decimal **dot** is used.
MerchantTrans	Payment identification in your system – order ID, customer ID etc. If you have no such ID for transaction (in case of donations etc), insert dash (-). Will not be displayed to customer. Will be inserted into transaction list.
ReturnUrlOk	URL of page the user will be directed after successful payment. M-platba transaction ID and your transaction ID (from MerchTrans item) is passed to this URL as parameters, so if you set ReturnUrlOk=http://www.testmerchant.cz/pay_simple_ok.jsp the user will be returned to URL like: http://www.testmerchant.cz/pay_simple_ok.jsp**?MerchantTrans=123456&IdTrans=1539759051490004078l4**
ReturnUrlErr	URL of page the user will be directed after **unsuccessful** payment. MerchTrans and IdTrans will be passed to page as parameters in the same way as described in ReturnUrlOk.
origin	Main (entry) URL of server (e-shop) where the payment is made.
lang	Language to be used on payment portal for login page. “cz” and “eng” are the only allowed values.

Want to allow user to change transaction amount? Change the input type for “Amount” item from “hidden” to “text” (or “select” or whatever):

```
<form action="https://m-platba.t-mobile.cz/spp/simple/pay.jsp" method="post">
<input type="hidden" name="IdMerchant" value="90004"/>
<input type="hidden" name="Description" value="item1"/>
Darovaná částka: <input type="text" name="Amount" value="50"/> Kč
<input type="hidden" name="ReturnUrlOk" value="http://www.testmerchant.cz/pay_simple_ok.php"/>
<input type="hidden" name="ReturnUrlErr" value="http://www.testmerchant.cz/pay_simple_err.php"/>
<input type="hidden" name="MerchantTrans" value="123456">
<input type="hidden" name="origin" value="www.testmerchant.cz"/>
<input type="hidden" name="lang" value="cz"/>
<input type="submit" value="Zaplatit T-Mobile M-Platba"/>
</form>
```

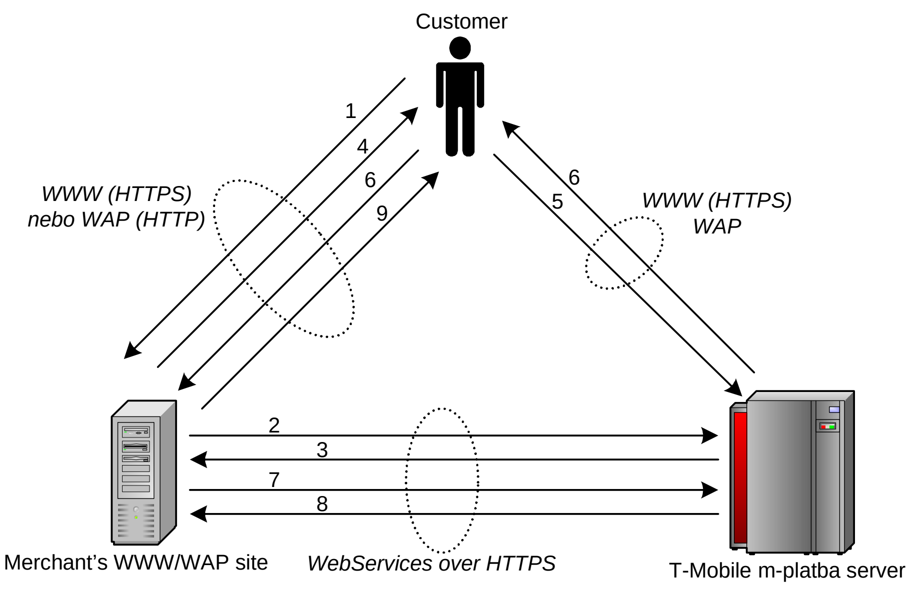
so payment button can look in the following way:



Warning: Payment details are transferred through customer’s web browser so they are vulnerable – can easily be modified by customer. You have to check transaction properties (amount and MerchantTrans) after customer is transferred back to your web site (after payment is done) using machine-readable transaction list or by using Merchant View manually.

8 WebServices interface for standard payment - implementation

8.1 WebServices payment scenario



1. Customer fills the basket in e-shop with goods/services and press the "Pay with T-Mobile m-platba" button during check-out.
2. Merchant's application make WebServices call for **transaction creation (M411_PaymentRequest)**. This request carries all information about transaction – description, amount, return URL's and so on.
   - M-Platba server create a transaction record and assign unique Transaction ID for it. Transaction ID is returned to Merchant.
3. *Transaction ID is returned from m-platba to merchant's application*
4. Merchant **redirects the customer to Secure payment portal**. Only transaction ID (and user language) is passed through the customer's browser, so user can not alter the payment.
5. Customer log in to payment portal using t-zones login/password or using one-time SMS password, and **confirm the transaction**. There is no communication between user and merchant (nor m-platba and merchant) in this step.
6. Customer is **redirected back to merchant's site**, onto URL written in transaction details. If transaction is not successful, customer is directed on different URL.
7. Merchant's application calls the "finalize transaction" WebService (**M411_MerchantNotification**).
   - This is the point the transaction is actually done. Money is transferred, SMS message is sent to user. Transaction is closed.
8. Merchant's application receive the transaction status from m-platba and deliver the service/goods to customer.

8.2 Transaction creation – PaymentRequest service

PaymentRequest service create the transaction in m-platba and return Transaction ID back to caller. Customer should be redirected to Payment Portal then.

There are two functions for transaction creation:

- **M411_PaymentRequest** – standard transaction.
- **M411_PaymentRequestMSISDN** – transaction that can be paid by customer with specific MSISDN only (this is for legal purposes – for example if your customers has to be older than 18 years to use your services and you checked user's age, you can make sure that this user only is allowed to pay)

8.2.1 Parameters

M411_PaymentRequest(	M411_PaymentRequestMSISDN(

int idmerch, string ipaddress, string item, string origin, double price, string urierr, string uriok, string merchTrans, string hash)	int idmerch, string ipaddress, string item, string origin, double price, string urierr, string uriok, string merchTrans, string hash, string msisdn)

Name	**Description**	**Formát**
idmerch	Your merchant identification number in m-platba.	Five digit number
ipaddress	IP address of your server, as seen from internet.	Character string, maximal length 15 chars.
item	Transaction description. Will be displayed to user.	Character string, maximal length 100 chars, standard 7-bit ASCII only (no national characters), with limited characters only – see bellow.
origin	Main (entry) URL of your web site.	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
price	Price (transaction amount)	Double.
urierr	URL for successful transaction – the user will be redirected there after payment is confirmed on payment portal	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
uriok	URL for unsuccessful transaction – the user will be redirected there after an error on payment portal	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
hash	Security hash with shared secret, as described in [10.4.3].	Character string
msisdn	Phone number which the transaction must be paid from. Full international format with plus on the beginning - „+420603123456“. If you create transaction with M411_PaymentRequestMSISDN and other than specified user will log-on to Payment portal, transaction will be cancelled.	Character string, maximal length 15 chars.

You can use limited set of characters only in „**item**“ parameter: alphanumeric characters a-z, A-Z, 0-9, space, hyphen „-„ and underscore „_“.

8.2.2 Return value

Both functions returns **M411_PaymentRequestAnswer** structure. It contains the following items:

Name	**Description**
errcode	Return code. 0 = OK other value = error code
errmessage	Error description in case errcode <> 0.
idtrans	Transaction ID in case of success (errcode==0)

8.2.3 Sample

Usage of **M411_PaymentRequest** function can be found in **samples_cs\sample_PaymentRequest.cs**.
Usage of **M411_PaymentRequestMSISDN** function can be found in **samples_cs\sample_PaymentRequestMSISDN.cs**.

8.3 Customer's redirection to Secure Payment Portal

If M411_PaymentRequest / M411_PaymentRequestMSISDN returns OK (and transaction ID), you should redirect customer to Secure Payment Portal.
This is described in [10.3].

8.4 Transaction finaliyation– Merchant Notification

When customer confirms the transaction on Payment Portal, he/she is redirected back to your web site to "**uriok**" URL with appended Transaction ID (in form "?IdTrans=153972696101234607 32").
You have to finalize the transaction by calling **M411_MerchantNotification** function now. The transaction is not done until you call this!

8.4.1 Parameters

```
M411_MerchantNotification(
        int idmerch,
        string idtrans,
        string hash,
        string ipaddress )
```

Name	Description	Formát
idmerch	Your merchant identification number in m-platba.	Five digit number
idtrans	Transaction ID (in „long" form - 21 characters)	String, 21 chars.
ipaddress	IP address of your server, as seen from internet.	Character string, maximal length 15 chars.
hash	Security hash with shared secret, as described in [10.4.3].	Character string

8.4.2 Return value

The function returns **M411_MerchantNotificationAnswer** structure with the following fields:

Name	Description
errcode	Return code. 20000 = OK, transaction successfully done 20001 = OK, transaction successfully done before. You'll receive this code if you call MerchantNotification repeatedly for the same transaction ID – see [8.4.2.1] 24000 = Error, transaction is modified by another thread – see [8.4.2.1] -1 = timeout within internal communication – see [8.4.2.2] other value = error code
errmessage	Error description in case errcode <> 20000, 20001.
idtrans	Transaction ID
merchauthcode	Transaction security code. Not displayed in transaction list. Please keep this code.
merchTrans	Order ID on your side – value entered in PaymentRequest in MerchTrans field

8.4.2.1 Fixing race conditions on MerchantNotifications

If your application calls the MerchantNotification function directly from landing page (uriok), there is a potential problem – if user reloads the page, MerchantNotification will be called again. M-platba can handle it – but **your application should be prepared also**.

Two scenarios can happen:

1) If the first call of MerchantNotification is already processed on back-end servers, transaction is fully settled. Repeated call of MerchantNotification will return code 20001 "transaction successfully done before". You can handle code 20001 in the same way as 20000 – transaction is OK. But you should check your side of transaction – isn't the transaction already proceed in your system also? Check the transaction ID / order ID in your system before processing.

2) The previous call of MerchantNotification is just in processing on back-end servers. You'll receive error code 24000 "transaction is processed by another thread" in this case. The best action for this case is to call MerchantNotification repeatedly (in – for example – 2 seconds interval; up to 60-90 seconds total time) until you get other code than 24000.

8.4.2.2 Internal server timeout (return code -1)

Error code **-1** is returned in case of timeout between M-Platba web server and back-end server. This code means that request has been transferred to back-end server, but no response has been received in reasonable time. Transaction **may** be done correctly. You can solve this situation by calling MerchantNotification again (see [8.4.2.1]) or by resolving the transaction status from transaction list within reconsolidation (see [12 Daily/monthly transaction reconciliation~~12 Daily/monthly transaction reconciliation~~]).

8.4.3 Sample

Usage of **M411_MerchantNotification** function can be found in **samples_cs\sample_MerchantNotification.cs**.

8.5 Failed transaction

If there is any trouble within transaction processing on Payment portal (user don't confirm the payment, user is over quota, user has m-platba disabled or so on), user is redirected to URL specified in **urierr** field of Paymentrequest.
Please **don't call MerchantNotification in this case**.

9 WebServices for Subscriptions - implementation

9.1 Creation of subscription

You have to call **CR2_SetupSubscription** for creating a subscription. You'll receive Subscription ID and you have to redirect the customer to Payment Portal with this Subscription ID.

9.1.1 Parameters

```
CR2_SetupSubscription(
        int idmerch,
        string description,
        double maxmonthmoney,
        double maxtransmoney,
        string validTo,
        string origin,
        string uriok,
        string urierr,
        string ipaddress,
        string hash  )
```

Name	**Description**	**Formát**
idmerch	Your merchant identification number in m-platba.	Five digit number
description	Description of subscription	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters). Limited characters only – see bellow.
origin	Main (entry) URL of your web site	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
maxmonthmoney	Maximal summary amount that can be paid from this subscription monthly	Double.
maxtransmoney	Maximal amount of one transaction	Double
validTo	Subscription validity. Max. one year, if you specify longer validity, it will be truncated to one year.	Character string in YYYYMMDDhhmmss format
urierr	Return URL from Payment portal for failed subscriptions.	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
uriok	Return URL from Payment portal for confirmed subscriptions.	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
ipaddress	IP address of your server, as seen from internet.	Character string, maximal length 15 chars.
hash	Security hash with shared secret, as described in [10.4.3].	Character string

You can use limited set of characters only in „**description**" parameter: alphanumeric characters a-z, A-Z, 0-9, space, hyphen „-„ and underscore „_".

9.1.2 Return value

The function returns **MR2_SubsPayAnswer** structure with the following fields:

Name	**Description**

errcode	Return code. 0 = OK other value = error code
errmessage	Error description in case errcode <> 0.
idtrans	Transaction ID for payment portal – this ID you should use when redirecting customer to Payment Portal
idSubscription	Subscription ID

9.1.3 Sample

Usage of **CR2_SetupSubscription** function can be found in **samples_cs\sample_SetupSubscription.cs**.

9.2 Check of subscription status

By calling **CR2_GetSubscription** you can get information about subscription status.

Warning! Even if the **CR2_GetSubscription** return that subscription is valid, it doesn't mean payment will be successfull. Payment from subscription may fail on many problems, for example on user's low credit balance.

9.2.1 Parameters

```
CR2_GetSubscription(
        int idmerch,
        long idsubscription,
        string hash)
```

Name	Description	Formát
idmerch	Your merchant identification number in m-platba.	Five digit number
idsubscription	Subscription ID	Number, 11 digits
hash	Security hash with shared secret, as described in [10.4.3].	Character string

9.2.2 Return value

The function returns **MR2_SubsAdmAnswer** structure with the following fields:

Name	Description
errcode	Return code. 0 = OK other value = error code
errmessage	Error description in case errcode <> 0.
subsAdmArray	Array of **MR2_SubsArray** objects. There will be 0 or 1 returned objects in array.

If user has no such subscription, error code 16002 will be returned.

MR2_SubsArray object contains the following fields:

Name	Description
idSubscription	Subscription ID
description	Subscription description
idMerchant	Merchant ID
msisdn	Empty value – not filled in
createTime	Date of subscription creation
validTo	Subscription validity
confirmTime	Date of subscription confirmation by user (id confirmed by user)
cancelTime	Datum of subscription cancellation (if cancelled)
maxMonthMoney	Maximal monthly summary amount
maxTransMoney	Maximal amount in one transaction
currMonth	Amount used in current month
cancelled	Status of subscription: Y = subscription is CANCELLED (or expired) and can't be used

	N = subscription is valid and can be used
merchantName	Merchant name

Use “cancelled” item for decision if the subscription is valid or not. Value of **“N”** means the subscription **is valid**.

9.2.3 Sample

Usage of **CR2_GetSubscription** function can be found in **samples_cs\sample_GetSubscription.cs**.

9.3 Payment from subscription

Function **CR2_PaySubscription** makes the payment from subscription.

9.3.1 Parameters

```
CR2_PaySubscription(
        int idmerch,
        long idsubscription,
        double subscription,
        string description,
        string origin,
        string uriok,
        string urierr,
        string ipaddress,
        string hash )
```

Name	**Description**	**Formát**
idmerch	Your merchant identification number in m-platba.	Five digit number
idsubscription	Subscription ID	Number, 11 digits
subscription	Paid amount	Double
description	Transaction description	Character string, maximal length 100 chars, standard 7-bit ASCII only (no national characters). Limited characters only – see bellow.
origin	Main (entry) URL of your web site	Character string, maximal length 255 chars, standard 7-bit ASCII only (no national characters)
urierr	Empty string („“)	Character string, maximal length 255 chars.
uriok	Empty string („“)	Character string, maximal length 255 chars.
ipaddress	IP address of your server, as seen from internet.	Character string, maximal length 15 chars.
hash	Security hash with shared secret, as described in [10.4.3].	Character string

You can use limited set of characters only in „**description**“ parameter: alphanumeric characters a-z, A-Z, 0-9, space, hyphen „-„ and underscore „_“.

9.3.2 Return value

The function returns **MR2_SubsPayAnswer** structure with the following fields:

Name	**Description**
errcode	Return code. 0 = OK, transaction has been done successfully other value = error code, no payment done
errmessage	Error description in case errcode <> 0.
idtrans	Transaction ID

9.3.3 The most common errcodes for CR2_PaySubscription

Kód	**Description**

16001	Subscription with this ID is not valid for payment: ID is not valid; subscription has been created by other merchant; has already been cancelled. Recommended reaction: this subscription is no longer available, make it as invalid.
16003	You have reached limit for this subscription (transaction amount or total amount monthly). Recommended reaction: Try again next month OR try again with smaller amount. Subscription is valid.
11031, 11005	Customer has m-platba service disabled. Recommended reaction: Treat as temporary problem. Service may be enabled for this user in future.
11006	Customer doesn’t exists. Recommended reaction: Customer’s account has been deleted. treat subscription ID as invalid, it will not be longer available.
11004, 19610, 19600	Not enough credit on customer’s account. Recommended reaction: Try again later; hope that user will top-up his credit.

9.3.4 Sample

Usage of **CR2_PaySubscription** function can be found in **samples_cs\sample_PaySubscription.cs**.

9.4 Subscription cancellation

You can cancel the subscription by calling **CR2_CancelSubscription** function.

9.4.1 Parameters

```
CR2_CancelSubscription(
        int idmerch,
        long idsubscription,
        string hash)
```

Name	**Description**	**Formát**
idmerch	Your merchant identification number in m-platba.	Five digit number
idsubscription	Subscription ID	Number, 11 digits
hash	Security hash with shared secret, as described in [10.4.3].	Character string

9.4.2 Return value

The function returns **MR2_SubsAdmAnswer** structure with the following fields:

Name	**Description**
errcode	Return code. 0 = OK, subscription cancelled other value = error code
errmessage	Error description in case errcode <> 0.

9.4.3 Sample

Usage of **CR2_CancelSubscription** function can be found in **samples_cs\sample_CancelSubscription.cs**.

10 WebServices for both Standard payments and Subscriptions – common information

10.1 Transaction identification number (Transaction ID)

Transaction ID is 21-digit decimal number.
Only the first 11 digits should be displayed to user; only the first 11 digits is shown in transaction list and on customer’s invoice. The rest 10 digits is used for security.

You have to use full 21-digits variant for calling the WebServices and for user redirection to Payment Portal.

10.2 Subscription identification number (Subscription ID)

Subscription ID is 11-digits decimal number.

10.3 Secure Payment Portal – how to redirect the user to Payment Portal?

10.3.1 Standard redirect

Use the following address for redirecting users to payment portal:
    http://tmo.cz/data/mplatba?idTrans=<Transaction ID>&lang=<Language>
for example:
    http://tmo.cz/data/mplatba?idTrans=153972696101234607329&lang=eng

Correct Payment Portal version (WAP, WWW for desktop browsers, WWW for mobile browsers) will be selected automatically. Language ("lang" item) can be **cz** or **eng**. This language will be used for Payment portal's login page. Customer's language settings from t-zones will be used after successful login.

10.3.2 Pre-filling MSISDS for SMS login

In case your application know customer's phone number you can pre-fill the phone number in login form and force the SMS one-time-password to be sent – so user doesn't have to fill his login name nor phone number, user is asked for one time password directly. The payment workflow will be little shorter.

If you want to pre-fill the MSISDN and send the SMS password, add "msisdn" parameter to the end of redirection URL:
    http://tmo.cz/data/mplatba?idTrans=153972696101234607329&lang=eng**&msisdn=603123456**

Phone number should be used in national format (9 digits).

If you use the "msisdn" parameter, **you disable the user from logging via t-zones name/password**! User can't change log-in method, he is allowed to use SMS one-time-password only. User **can't even change the phone number** – so he can't pay from other account than your application know.

10.4 WebServices security

Security of WebServices are built from the following components:
1. Merchant's SSL certificate
2. Server's SSL certificate
3. Security has with „shared secret“
4. Merchant's IP address information

10.4.1 Merchant's SSL certificate (WebServices only!)

Merchant has unique SSL certificate. Every WebServices call have to be transferred within HTTPS request with this client certificate. No other certificate will be allowed.

Certificate name is "appNNNNN", where NNNNN = merchant identification number. For example "app90004" for merchant 90004.

10.4.2 Server's SSL certificate

M-platba server has correct SSL certificate **m-platba.t-mobile.cz**, signed by one of trusted authorities (Verisign, at this time). Every HTTPS communication uses the certificate.
Your application have to check the SSL certificate validity – at least it should check:
- certificate has the correct name
- certificate is valid (not before and not after the certificate validity)
- certificate is signed by trusted authority

Both Java and .net environments can do that verifications automatically.

10.4.3 Security has with „shared secret“

“Shared secret hash” is used for improved security – as a single “electronic signature” of request.
Both parties know the “shared secret” – secure password for communication (in case of m-platba every merchant has different shared secret).
Sender compute SHA1 hash of request and secret password. Then send the request and hash code through the communication channel (secret password is NOT sent). Recipient compute the hash again from request data and secret password. If both hashes are the same, request is valid; if hashes are different, request has been tampered somewhere and can’t be used.

Detailed description of “shared secret hash” procedure

1. Sender want to send three request properties – P1, P2, P3.
2. For example:
   - P1 = string „ahoj“
   - P2 = number 10.15
   - P3 = string „test“
3. Sender make string from all the properties, without delimiters
   - Integer numbers without the leading zeros nor plus sign.
   - Floating point number without leading zero nor plus sign, with decimal dot, with two decimal digits after the decimal dot. If number is less then 1, it should be written as 0.xx (with leading zero).
4. So string „ahoj10.15test“ will be created in our case.
5. Sender will append the shared secret after the string. For example shared secret = “PASSWORD”.
6. Resulting string is „ahoj10.115testPASSWORD“
7. Sender compute the hash from this string.
8. Sender sends P1, P2, P3 items and computed hash code. No shared secret password is sent!
9. Recipient make the same computation as described in points 3-7, and check if the computed hash is the same as received hash.
10. If hashes are the same, message is valid and can be processed.

We are using SHA1 hash algorithm in m-platba. Result (20 byte) is transferred as text – as a string of hexadecimal numbers, ith lower-case letters. For example:

    5a3559da2151bd1fdd0a6021184333a156ce33b9

Items used for hash computation for every WebServices calls:

PaymentRequest, PaymentRequestMSISDN:

    <idmerch><ipaddress><item><origin><price><urierr><uriok><shared_secret>

MerchantNotification:

    <idmerch><idtrans><ipaddress><shared_secret>

SetupSubscription

    <idmerch><description><uriOk><shared_secret>

PaySubscription

    <idmerch><description><částka><shared_secret>

GetSubscription

    <idmerch><subscriptionid><shared_secret>

CancelSubscription

    <idmerch><subscriptionid><shared_secret>

where „<“ and „>“ characters shows properties placement – should not be used for has computation!

No national characters (czech character with diacritics marks) **should be used in Description** field to avoid codepage-conversion problems. Convert all texts to 7-bit ASCII, please!

Sample implementation of hash can be found in **samples_cs/sha1hash.cs**.

10.4.4 Merchant’s IP address

You have to fill your IP address into WebServices request. Use the address that is seen from internet – so if your server is after NAT, enter the “public” address of NAT.
Request with bad IP address (different from IP address the request came from) will not be accepted – they’ll fail with “security alert” SOAP failure.

10.5 Commented simple samples of WebService calls

You can find commented and really simple samples in **samples_cs/** directory.
There is one sample per every function m-platba provides – it displays how to fill the parameters, compute hash and parse the result. Samples are written in C#, but they are simple to read even for Java-only programmers.
Samples are named **sample_<function name>.cs**. In the same directory you'll find WebService proxy client (MamiService.cs), tool for computing shared-secret hash (sha1hash.cs) and a configuration file (config.cs).

Samples can be run directly from command-line (without compilation, without the MS Visual Studio) by using **cs-script** tool (http://www.csscript.net/). Cs-script is scripting environment based on C# language. Install the cs-script and then you start the samples by the following command:

```
cscs sample_<function name>.cs
```

for example:

```
cscs sample_PaymentRequest.cs
```

For successful run of samples you have to:

- save your public certificate appNNNNN.cer into this directory
- import your private key (appNNNNN.pfx) to Windows certificate store - see chapter 11.1 for details
- modify config.cs file with valid configuration information

10.6 More complex sample application – Java

There is more complex sample application in **samples_java/** directory – there is a full application which do subscriptions via WebServices and standard payments with both WebServices and simple HTML button.
Application is stored in **Ukazkova_aplikace_java** subdirectory. Application is written for IBM WebSphere Application Server; project files are for IBM RAD 7.5.
In **Ukazkova_aplikaceEAR** directory you can find project for EAR-creation from application above. Click on this project and select Export -> EAR.

10.7 WSDL - WebService definition

WSDL can be found in wsdl/

10.8 WebService endpoint, ports used

WebService is running on this URL:

https://m-platba.t-mobile.cz:9443/C7_ProxyTransaction2/services/Transaction

Your firewall have to be configured so your application can connect to **m-platba.t-mobile.cz** port **9443/tcp**.

11 FAQ and other implementation details

11.1 How to use HTTPS with client-certificate from ASP.NET (and .Net in common)

Standard .Net implementation of SSL client-certificate is not really straightforward. You have to supply public certificate (.cer file) from your application – and private key is fetched from Windows cryptography store.

Many developers make mistakes here – for example if you import private key into user's store (and not into machine's store), application will work from commandline, but it will not work within IIS. If you didn't supply private key in correct way, no error will be recorded from key management – you'll receive **403 Forbidden** from m-platba server instead.

Correct way of private key usage is described here:
http://support.microsoft.com/kb/901183/en-us?fr=1

11.2 How can I test the implementation? Is there some TEST environment?

M-platba has no public TEST environment. You can test directly on PRODUCTION system. You can do tests with amounts like 0.01 CZK.

12 Daily/monthly transaction reconciliation

You can encounter troubles in some cases of non-standard customer's actions (as many page-reloads and so on) – transaction is treaten as successful on m-platba side, but you see it as unsuccessful. This is described for example in chapter 8.4.2.1.

To automatically detect that cases, we recommend you to do daily/monthly reconciliation – to compare list of transactions in your system with transaction list you can receive from machine-readable transaction list [5 Machine-readable transaction list~~5 Machine-readable transaction list~~].

Notice: timestamp in transaction list is:

- for failed transactions – time when transaction has been created (PaymentRequest webservice has been called)
- for successful transactions – time when transaction has been approved in T-Mobile billing system (this equals to time when customer press "Yes, do the transaction" on Payment Portal)

This timestamp may be different from transaction timestamp in your system. So you have to carefully handle transactions on boundaries – around midnight. Some transactions can be found in previous (or following) day than in your system.