1. Bridge Configurable Entitlements

This section will outline the Deferred Entitlements feature (only available for OpenMAMA and OpenMAMA Enterprise) which will allow you to defer entitlement checking to a middleware bridge, on a per bridge basis, bypassing the entitlements checking in the core MAMA code.

The primary use case for this feature would be for deployments of MAMA with bridges supporting multiple platforms, where entitlements may be enforced at the bridge/middleware level or elsewhere in the platform.

An internal function has been added, mamaBridgeImpl_setReadOnlyProperty() (please refer to Example 1 for function prototype), to allow you to specify if entitlements should be deferred for a specific bridge. This should be called from within your bridge code if you wish to use the functionality.

The property must be set programmatically and cannot be set via your mama.properties file.

For example, to defer entitlement checking for a Wombat Middleware bridge you would need to call:

mamaBridgeImpl_setReadOnlyProperty (bridgeHandle, “mama.wmw.entitlements.deferred”, “true”);

On successfully deferring entitlements you can expect to see the following output as part of entitlements checking within mama_open() :

"Entitlements deferred on [middleware] bridge."

Alternatively for subscriptions, you can expect to see the following output as part of the subscription creation process within mamaSubscription_setupBasic():

“Entitlements checking at subscription creation deferred to [middleware] bridge [bridge pointer]"

Example 1: Function Prototype

mamaBridgeImpl_setReadOnlyProperty (mamaBridge bridgeImpl, const char property, const char* value);*

2. Mama Plugins

This section will outline the mamaPlugin feature (only available for OpenMAMA and OpenMAMA Enterprise) which will allow you to create and run your own code within 'hooks' into the MAMA codebase.

A mamaPlugin is essentially a shared object and you can use the template provided, see Example 2, to create your own plugin. The naming convention of the plugin will be of the form "mamaplugin<name>", for example on Linux your plugin would be called:

libmamaplugintemplate.so

Some hooks are passive and will simply notify you of an error that occurred whereas others will be more active and can result in MAMA functions failing.

Currently there are 4 hooks available (full prototypes provided in template examples):

Name	Required	Invoked	Action
MamaPlugin_initHook	Yes	Invoked during mama_loadPlugin(), (called from mama_initPlugins() within mama_open()) after functions are registered	Required function, the plugin will fail to load if not present
MamaPlugin_transportPostCreateHook	Optional	Invoked at the end of mamaTransport_create()	Will log an error if the hook fails
MamaPlugin_publisherPreSendHook	Optional	Invoked at the beginning of mamaPublisher_send()	mamaPublisher_send() will fail if the hook fails resulting in the message not being sent
MamaPlugin_shutdownHook	Yes	Invoked in mama_shutdownPlugins() which is called during mama_close()	Required function, the plugin will fail to load if not present

You must add the following property to your mama.properties to load your plugin(s)

mama.plugin.name_0=<name>
mama.plugin.name_1=<name>

If you follow the template (from Example 2) your plugin function names will be of the convention 'templateMamaPlugin_initHook' etc. and your mama.properties entry will look like this:

mama.plugin.name_0=template

Basically, the name of your plugin should be prefixed to each of the function names.

If you have verbose logging enabled you will expect to see the following output during the plugin initialisation phase:

2015-04-09 13:45:02: (f9018280) : mama_initPlugins(): Initialising [mama.plugin.name_0] template
2015-04-09 13:45:02: (f9018280) : mama_loadPlugin(): Successfully registered plugin functions for [template]
2015-04-09 13:45:02: (f9018280) : mama_loadPlugin(): Successfully run the init hook for mama plugin [template]

Where you will see if your plugin was successfully initialised or if it was missing any of the optional functions. If your plugin was missing a required function you can expect to see the following output:

2015-04-09 13:45:02: (f9018280) : mama_initPlugins(): Initialising [mama.plugin.name_0] template
2015-04-09 13:45:02: (f9018280) : mamaPlugin_registerFunctions(): Cannot load plugin, does not implement required function: [templateMamaPlugin_shutdownHook]
2015-04-09 13:45:02: (f9018280) : mama_loadPlugin(): Failed to register plugin functions for [template]

Example 2: Template for a basic mamaPlugin

```
/*
 * mamaPluginTemplate.c
 *
 *  Created on: 23 Mar 2015
 */

#include <stdio.h>
#include "mama/mama.h"
#include "mama/status.h"
#include "mama/types.h"
#include "mamaPluginTemplate.h"


mama_status
templateMamaPlugin_initHook (mamaPluginInfo* pluginInfo)
{
	mama_status status = MAMA_STATUS_OK;

	return status;
}

mama_status
templateMamaPlugin_transportPostCreateHook (mamaPluginInfo pluginInfo, mamaTransport transport)
{
	mama_status status = MAMA_STATUS_OK;

	return status;
}

mama_status
templateMamaPlugin_publisherPreSendHook (mamaPluginInfo pluginInfo, mamaPublisher publisher,
mamaMsg message)
{
	mama_status status = MAMA_STATUS_OK;

	return status;
```

```
}

mama_status
templateMamaPlugin_shutdownHook (mamaPluginInfo pluginInfo)
{
        mama_status status = MAMA_STATUS_OK;

        return status;
}
```

Example 3: Template header for a basic mamaPlugin

```
/*
 * mamaPluginTemplate.h
 *
 *  Created on: 23 Mar 2015
 */

#ifndef MAMAPLUGINHOOK_H_
#define MAMAPLUGINHOOK_H_

#include "mama/mama.h"
#include "mama/status.h"
#include "mama/types.h"

#if defined( WIN32 )
#   define PLUGINExpDLL __declspec( dllexport )
#else
#   define PLUGINExpDLL
#endif


PLUGINExpDLL
mama_status
templateMamaPlugin_initHook (mamaPluginInfo* pluginInfo);

PLUGINExpDLL
mama_status
templateMamaPlugin_transportPostCreateHook (mamaPluginInfo pluginInfo, mamaTransport transport);

PLUGINExpDLL
mama_status
templateMamaPlugin_publisherPreSendHook (mamaPluginInfo pluginInfo, mamaPublisher publisher,
mamaMsg message);

PLUGINExpDLL
mama_status
templateMamaPlugin_shutdownHook (mamaPluginInfo pluginInfo);

#endif /* MAMAPLUGINHOOK_H_ */
```