European Collaborative Innovation Centres for Broadband Media Service

YOUR ELECTRONIC ASSISTANT™

e-Me

Prototype – e-Me Core

EUROPEAN UNION
European Regional
Development Fund

Investing in the future by working together for a sustainable and competitive region

The Interreg IVB
North Sea Region
Programme

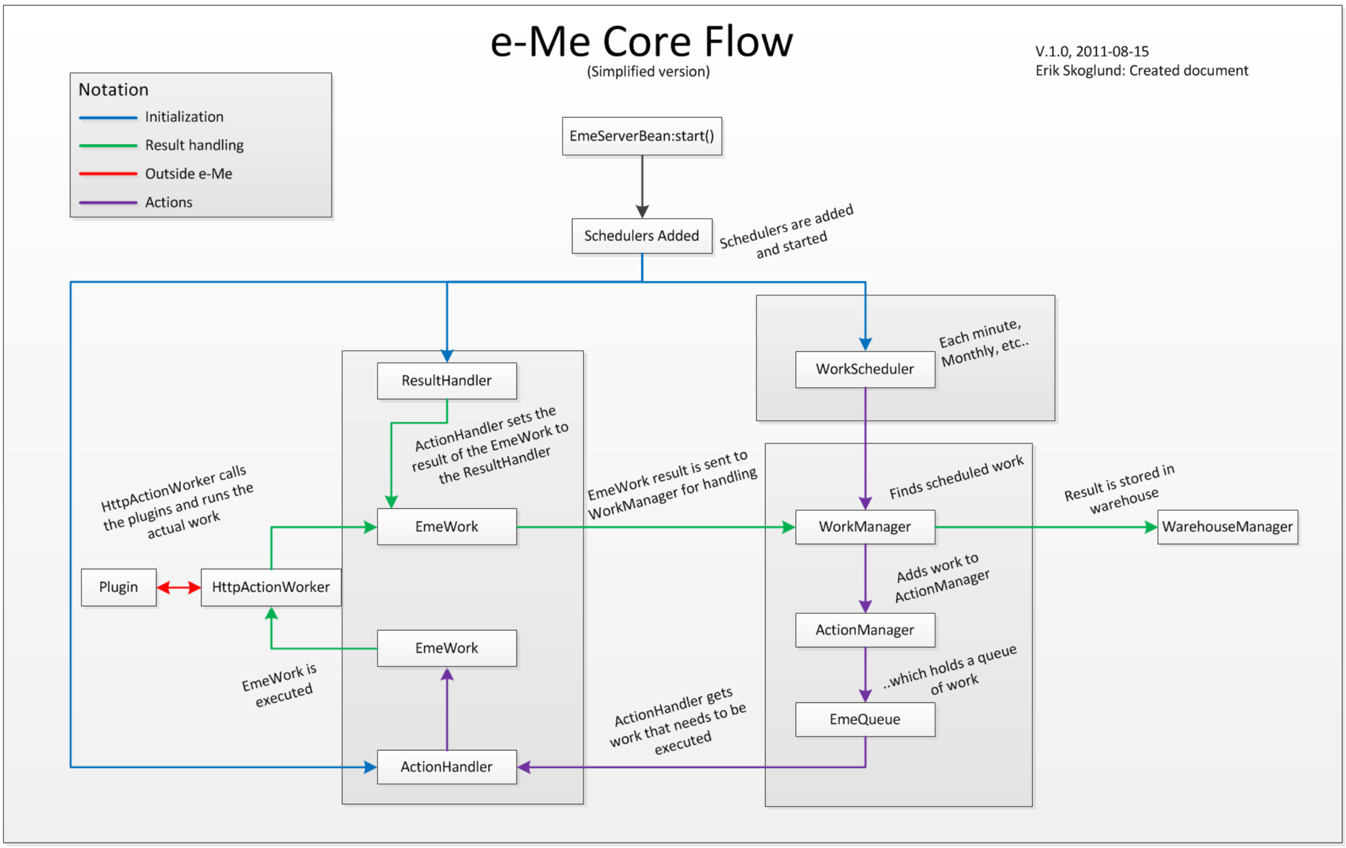
1 Core Prototype

1.1 Core idea

The basic idea behind e-Me Core is an engine that runs and performs abstract work. Meaning that Core itself won’t have any idea of what work it is performing. The actual work is delegated to plugins. This way e-Me Core can be heavily modified by pulling and putting plugins into the engine. So in general e-Me Core is an engine that plans, creates and delegates work.

1.2 Core System architecture

e-Me consists of two separate systems, *Web* and *Core*. This report focuses on *Core*. Core is a system that runs in the background of e-Me. This system is a collection of techniques brought together to an architecture that looks like this image.



e-Me Core Flow
(Simplified version)
V.1.0, 2011-08-15
Erik Skoglund: Created document
Notation
Initialization
Result handling
Outside e-Me
Actions
EmeServerBean:start()
Schedulers Added
Schedulers are added and started
ResultHandler
WorkScheduler
Each minute, Monthly, etc..
ActionHandler sets the result of the EmeWork to the ResultHandler
HttpActionWorker calls the plugins and runs the actual work
EmeWork
EmeWork result is sent to WorkManager for handling
Finds scheduled work
WorkManager
Result is stored in warehouse
WarehouseManager
Plugin
HttpActionWorker
Adds work to ActionManager
ActionManager
EmeWork
EmeWork is executed
..which holds a queue of work
EmeQueue
ActionHandler gets work that needs to be executed
ActionHandler

This image shows the flow of Core. Core is basically a machine running scheduled tasks, but it also relies on many plugins. These plugins are completely separate from Core itself and are deployed freely on the same server running Core.

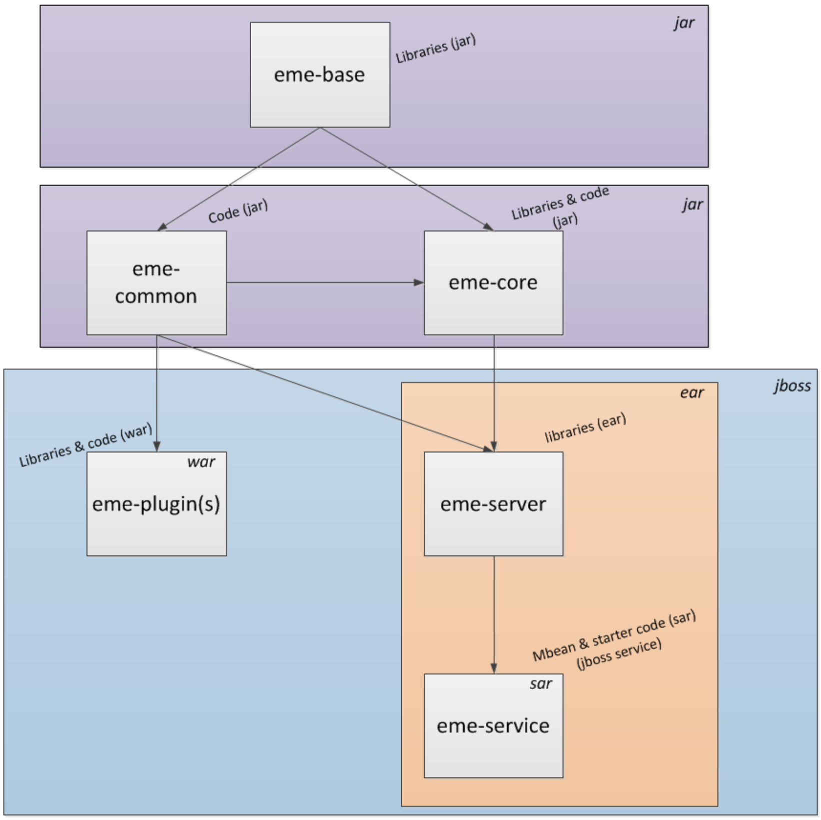
As previously mentioned, Core is based on schedules. These schedules are run on a range from every month to as often as every minute. It uses what is called a “Work scheduler” to find work that is then added to a “Work manager”. These processes handles the tasks at hand and performs different kinds of work depending on what they are handed. Work is then executed by an “Action worker”. This is where the plugin comes into the picture. The plugin is started and required information is sent to it. The plugin does its work and returns a result that is later handled by Core.

This separation of core and the plugins allows a very dynamic way of adding functionality to e-Me.

1.3 Core File architecture

On the more technical side of Core, it is based on three tiers of abstraction. On the first level we have “eme-base” this is a jar-file with libraries that are needed both by e-Me Core itself and by plugins of e-Me Core. The second level consists of “eme-common” and “eme-core”. Common contains classes and libraries that plugins and eme-server needs to run. It also contains eme-base. Eme-core is the actual "core" of the system. It has all the logic of the system and the entire engine that runs it. Core also has a relation to eme-common.

The final level consists of two parts. One is eme-plugin, which is a .war file. These files are the actual plugins and can be pretty much anything. The pink rectangle containing eme-server and eme-service is an ear file. This is the application for Core that is run on a server.



1.4 Software

e-Me Core is a Java EE application that runs on a JBoss Application Server. This makes e-Me platform independent and can run on any platform that supports JBoss AS. It uses maven to handle dependencies and as the previous image showed it is divided into three tiers. Persistence is handled with a mysql database.

Apart from this, Core uses a lot of libraries and open source software, some of them are:

- Hibernate
- JBoss AS
- JUnit
- Spring
- ical4j
- Jackson JSON processor
- Tuckey urlrewritefilter
- Maven

1.5 Hardware

Since e-Me Core runs on a JBoss AS, it can be run on many different platforms including Windows, Mac OSX and Linux.