Technical Manual of the `Universal Server`

Organisation: Copyright (C) 2019-2021 Olivier Boudeville

Contact: about (dash) universal-server (at) esperide (dot) com

Creation date: Saturday, May 2, 2020

Lastly updated: Monday, February 8, 2021

Status: Work in progress

Version: 0.0.3

Dedication: Users and maintainers of the `Universal Server`.

Abstract: The Universal Server, part of the umbrella project of the same name, is a multi-service daemon in charge of the automation (monitoring, scheduling and performing) of various computer-based tasks, such as the proper management of the server itself or, in the future, of house automation.

We present here a short overview of these services, to introduce them to newcomers.

The next level of information is to read the corresponding source files, which are intensely commented and generally straightforward.

Table of Contents

Overview

We present here a short overview of these services, to introduce them to newcomers.

The next level of information is to read the corresponding source files, which are intensely commented and generally straightforward.

The project repository is located here.

Layer Stack

From the highest level to the lowest, as summarised here, a software stack involving the Universal Server usually is like:

- the *Universal Server* services themselves (i.e. this us-main layer)
- [optional] the *Universal Webserver*, i.e. US-Web (for web interaction)
- US-Common (for US base facilities)
- Ceylan-Traces (for advanced runtime traces)
- Ceylan-WOOPER (for OOP)
- Ceylan-Myriad (as an Erlang toolbox)
- Erlang (for the compiler and runtime)
- GNU/Linux

The shorthand for `Universal Server` is `us`.

Licence

The `Universal Server` is licensed by its author (Olivier Boudeville) under the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of this license, or (at your option) any later version.

This allows the use of the Universal Server code in a wide a variety of software projects, while still maintaining copyleft on this code, ensuring improvements are shared.

We hope indeed that enhancements will be back-contributed (ex: thanks to merge requests), so that everyone will be able to benefit from them.

Current Stable Version & Download

As mentioned, the single, the single mandatory prerequisite of the Universal Server is US-Common, which relies on Ceylan-Traces, which implies in turn Ceylan-WOOPER, then Ceylan-Myriad and Erlang.

We prefer using GNU/Linux, sticking to the latest stable release of Erlang (refer to the corresponding Myriad prerequisite section for more precise guidelines), and building the Universal Server from sources, thanks to GNU `make`.

We recommend, for all Erlang-related software, to rely on rebar3.

One wanting to be able to operate on the source code of these dependencies may define appropriate symbolic links in a `_checkouts` directory created at the root of `us-main`, these links pointing to relevant GIT clones.

Using Cutting-Edge GIT

This is the installation method that we use and recommend; the Universal Server `master` branch is meant to stick to the latest stable version: we try to ensure that this main line always stays functional (sorry for the pun). Evolutions are to take place in feature branches and to be merged only when ready.

Once Erlang, Cowboy and possibly Awstats are available, it should be just a matter of executing:

```
$ git clone https://github.com/Olivier-Boudeville/Ceylan-Myriad myriad
$ cd myriad && make all && cd ..

$ git clone https://github.com/Olivier-Boudeville/Ceylan-WOOPER wooper
$ cd wooper && make all && cd ..

$ git clone https://github.com/Olivier-Boudeville/Ceylan-Traces traces
$ cd traces && make all && cd ..

$ git clone https://github.com/Olivier-Boudeville/us-common
$ cd us-common && make all

$ git clone https://github.com/Olivier-Boudeville/us-main
$ cd us-main && make all
```

Running a corresponding test just then boils down to:

```
$ make debug
```

Using OTP-Related Build/Runtime Conventions

As discussed in these sections of Myriad, WOOPER, Traces and US-Common, we added the (optional) possibility of generating a Universal Server *OTP application* out of the build tree, ready to result directly in an *(OTP) release*. For that we rely on rebar3, relx and hex.

Then we benefit from a standalone, complete Universal Server.

As for Myriad, WOOPER, Traces and US-Common, most versions of the Universal Server are also published as Hex packages.

For more details, one may have a look at:

- rebar.config.template, the general rebar configuration file used when generating the Universal Server OTP application and release (implying the automatic management of Myriad and WOOPER)
- rebar-for-hex.config.template, to generate a corresponding Hex package for Universal Server (whose structure and conventions is quite different from the previous OTP elements)

Support

Bugs, questions, remarks, patches, requests for enhancements, etc. are to be reported to the project interface (typically issues) or directly at the email address mentioned at the beginning of this document.

Please React!

If you have information more detailed or more recent than those presented in this document, if you noticed errors, neglects or points insufficiently discussed, drop us a line! (for that, follow the Support guidelines).

Ending Word

Have fun with the Universal Server!