Voltumna Linux: A CUSTOM DISTRIBUTION
FOR (EMBEDDED) SYSTEMS

L. Pivetta∗, A. I. Bogani, G. Scalamera, Elettra Sincrotrone Trieste, Trieste, Italy

Abstract
In the last years a thorough approach has been adopted to address the ageing and the variability of control system platforms at Elettra Sincrotrone Trieste. The second generation of an in-house built operating system, named Voltumna Linux, which is based on an immutable image approach, is now ready for production, supporting a number of commercial-off-the-shelf embedded systems. Moreover, the same approach is perfectly suitable for rack-mount servers, with large memory support, that often require the inclusion of third party or closed source packages. Being entirely based on Git for revision control, Voltumna Linux brings in a number of advantages, such as reproducibility of the product, ease of upgrading or downgrading complete systems, centralised management and deployment of the user software to name a few.

INTRODUCTION
In recent years the number of front-end machines used within Elettra control systems increased considerably. Legacy systems, such as VME single board computers based on MC680x0 and PowerPC microprocessors, were joined by x86 systems in standard 19-inch form factor, like rack-mount servers, and smaller form factor such as NUC, UPBoard, Jetway and MinnowBoard. Moreover, also embedded boards based on ARM microprocessors, such as the Beaglebone, and system-on-chip boards based on FPGA, e.g. De10-Nano, Sockit, Dinet and Arria10 have been adopted.

At Elettra, control systems are mostly based on GNU/Linux distributions, introduced over the years, many times featuring hard real-time extensions such as RTAI [1] or Xenomai [2]. Keeping the same approach for new platforms, over the time, would lead to an even more heterogeneous install base, with additional GNU/Linux distributions or versions. Different subsystems make the administration difficult; typical examples are the init-manager, available in SystemV, Upstart or systemd flavours, or the network management stack that can be based on systemd-networkd, NetworkManager etc. Many versions of system libraries would make it difficult to develop the same application on different targets, restricting the replacement of a system with a new one based on a different architecture.

Moreover, together with the new platforms, existing systems have to be maintained much longer than the typical commercial distribution support, which several times is limited to security updates.

Novel technical solutions introduced in FERMI and the prototypes developed for Elettra 2.0 required new control platforms, specifically designed; GNU/Linux is the natural operating system choice, stated the very good know-how available in house. A common solution to deal with operating system fragmentation is to adopt automation and configuration management tools, such as Ansible or Puppet. Although valid, this approach does not cover all the requirements and cannot support all the legacy platforms in use.

REQUIREMENTS
Based on the experience integrating and supporting control system platforms and the use foreseen for the incoming installations, the requirements to be satisfied by the operating system and software stack support for the new platforms have been identified:

1. Allow the adoption of specific versions of system components.
2. Allow to integrate third-party software, when source code is available.
3. Provide multiple levels of customization (kernel, drivers, libraries) by patching or revision control.
4. Optimise the operating system for each hardware.
5. Guarantee reproducibility, for both the operating system and the BIOS/firmware of motherboard and adapters.
6. Build, whenever possible, system configurations first.
7. Encourage software reuse making it available from initial releases.
8. Minimise platform dissimilarity, with special attention to the operating system and low level software stack.
9. Provide separate images for development and production systems.
10. Simplify working with new or low performance platforms supporting cross-compiling.

Yocto / OpenEmbedded
The Yocto/OpenEmbedded Project [3] is an open source collaboration that provides a flexible set of tools to create custom GNU/Linux based systems for embedded hardware, regardless of the architecture. Established in 2010, it involves many hardware manufacturers, including AMD, ARM, Intel, Texas Instruments to name a few, open-source operating system vendors and electronics companies.

Within the boards in use at Elettra, Yocto/OpenEmbedded is supported by Terasic for the Sockit system-on-chip FPGA based board and by Texas Instruments for the Beaglebone.

Therefore Yocto/OpenEmbedded was the natural, more convenient, choice to base a new, custom, GNU/Linux distribution to be developed in-house.

Software
Software Architecture & Technology Evolution
**Voltumna Linux**

The prototype, named Flop [4], demonstrated not being flexible enough for all the use cases, mostly because the support for all platforms was based on a single distribution. Voltumna [5], hence, is based on the concept of *meta-distribution*, defining a base system that can be kept as is or customised in additional steps. This is done properly aggregating several, pre-existing or custom built, layers. Each layer defines a collection of text files, used to generate or modify a software package or its configuration, named *recipes*. As an example, the recipe written for a library is listed below.

```bash
DESCRIPTIOn = "Library for Gaussian fitting"
LICENSE = "GPL-3.0-only"
LIC_FILES_CHKSUM = "file://LICENSE;md5=6f9f2aeef846d428f2b35d46d9c5ebe2"

do_install() {
    oe_runmake PREFIX=${D}${prefix} install
}

SRC_URI = "https://gitlab.elettra.eu/cs/lib/libfit/-/archive/\n  $PV/libfit-$PV.tar.bz2"
SRC_URI[sha256sum] = "2b279235c1e1de02e10a650336bb6c2\nbabf501353f1f79800f57cfbd2c6767924"
```

The following Voltumna distributions have been built at date:

- *ec*: Equipment Controller (EC) [9],
- *ccd*: Charge-Coupled Device (CCD) [10],
- *ebpm*: Elettra Beam Position Monitor [11],
- *imrf*: Interlock Module for Radio Frequency [12],
- *a2720*: A2720 power supply [13].

This approach allowed to use the same hardware, a power supply designed by Elettra, and integrate the support for different environments: the layer *meta-elettra* for use at Elettra and the layer *meta-ess* for use at ESS. All the underlying layers, which constitute the larger part of the system, are exactly the same.

Voltumna images, shown in Fig. 1, are available for download from the organisation page [14]. Within Yocto/OpenEmbedded each platform is represented by the *MACHINE* file abstraction. Voltumna defines its own *MACHINE* file for each specific target; this allows, for instance, to build the same distribution for different platforms. As visible in Fig. 1, the *ec* distribution [9] is available for three different boards made by Artesyn, five different boards made by BeagleBoard.org and a couple of additional boards made by Jetway and Aaeon; the details are shown in Table 1.

Furthermore, *MACHINE* file customisation allows to use all the instruction set of a microprocessor, rather than a common subset, and to include specific tools, such as tools to update the BIOS/firmware of the motherboard as well as of any pluggable adapter, together with the BIOS/firmware updates as distributed by the manufacturer. Typical use cases are network adapters (NIC), frame grabbers or serial line adapters, that, under heavy load, often show buggy or erratic behavior when the BIOS/firmware version is not up-to-date.
Using custom *MACHINE* files enables pushing the optimisation to the limit, exploiting techniques like *isolcpus*, *cgroups*, *vfio* and *DPDK* [15] for hardware partitioning to reserve adapters and/or CPU cores for the execution of hard real-time tasks [16].

Voltumna Linux distribution images are available in two flavors, based on the intended use, and a software development kit for each target.

The Software Runtime Environment (SRE) image is finalised for deployment on production hosts, where system resilience and reproducibility are fundamental. With these goals in mind, the SRE is built with essential components, reduced to the minimum, and totally lacks any development tool or documentation. Most of the SRE systems in production at Elettra are deployed in diskless configuration.
and network boot; shared filesystems are mounted read-only from a NFS server. The root user is disabled and nominal users can access the SRE systems by ssh key.

The Software Development Environment (SDE) image provides a development environment, complete with all development and debugging tools, built to run on the target platform. Native target development is essential to develop low-level system components, such as firmware, device drivers or kernel modules.

The Software Development Kit (SDK) is an installable package, with the cross-compiler and the development environment proper to the target system. The Voltumna SDK is available for guest hosts running Linux Ubuntu 18.04. Beyond making the development environment available detached from the hardware, the SDK enables working on high performance desktop or server hosts, shortening build times as compared to compiling on the target systems, that often are low performance embedded systems.

All middleware, libraries and application software developed in-house for Elettra and FERMI control systems are structured in such a way that can be built either on the target platform or with the SDK without changes.

**DEPLOYMENT**

As already mentioned, at Elettra the typical deployment for front-end computers is based on network boot and diskless operation. Each facility deserves at least one server for network services, one for shared filesystem, based on NFS, one for the Tango database, all deployed as virtual machines, plus several physical and virtual hosts running the control system devices.

Yocto/OpenEmbedded allowed to adapt software packages for the selected targets, enabling network boot where needed and making each target boot process as similar as possible. The result is that all targets share the same boot process starting from `initramfs` step.

As noted, Voltumna Linux is based on immutable image approach, regardless the target is booting over the network or from local storage: the main system folder, where Voltumna is stored, is mounted read-only. When in diskless operation, target systems mount a private root filesystem in read-write mode, but everything under `/usr` is shared, reducing storage requirements on the NFS server as side benefit.

The Software Development Kits for Voltumna Linux distributions, supporting the target systems, have been integrated into Elettra's automatic build and installation system, named INAU [17], and are available to the developers like legacy development environments.

**ACKNOWLEDGEMENTS**

The authors would like to thank the Controls group for the stimulating discussions.

**REFERENCES**


\(^1\) A real use-case at Elettra is the firmware for the Programmable Real-time Unit Subsystem (PRUSS) used within the A2720 power-supply.