OPENUSSD DEVELOPMENT GUIDE

DATE	VERSION	CHANGE	AUTHOR
January 6th 2010	1.0	Initial Release	d3vnull

CONTENT

Introduction

OpenUSSD addresses a simple objective of giving users an **all-in-one Enterprise USSD Client platform** for integration into almost any USSD Gateway with support for protocols such as **SOAP,RAW-XML,XML-RPC,SMPP and HTTP POST**.

The problem we address with OpenUSSD is that of developers having to develop various interfaces or integration application which offers them connectivity into a USSD Gateway hosted by a Mobile Network operator.
The problem with such an approach includes but are not limited to:

1. **Monitoring:** Keeping an eye on multiple integration applications can be such a pain especially if they are developed by different programmers.
2. **Performance:** Since implementation are in various languages, the true test of performance is hardly determined as individual programmers hardly go through QA processes before deployment.
3. **Enhancement:** There is little or no time to revisit the application to enhance its performance as programmers tend to focus on other projects. The project virtually comes to a halt when the programmer deploys his application making it difficult for others to continue.
4. **Support and Continuity:** It becomes difficult for a support team to fully understand the function of the integration application making it much harder to troubleshoot when clients complain.

Our Solution. To solve the problems above we believe the best approach is Open Source Development that meets enterprise requirements and evolve in features over time.

Feature List

Central Monitoring:

OpenUSSD presents 3 ways to monitor nodes and their performances.

a) The CLI: We present a CLI which allows administrators and support teams to access information about each node their installation is connected to. Our design approach is central and in the next few chapters we explain our motivation for such an approach. Since most Linux administrators are used to working from CLI this provides a much more faster access to get statuses and statistics of individual connections.

b) The WebUI: Since our solution is 100% implemented in Python, we take advantage of the Django Framework to provide a very user friendly interface that makes it easier for support teams to even detect situations such as break in user sessions, loss of connectivity to host

c) The Logs: This is generic to almost every Linux based application. We provide administrators and support teams the chance to set debug level for core services including services to each Network Operator.

Link Monitoring:

We have a fully functional link monitoring feature which can be configured to send email alerts when connectivity to an MNO is lost. This status is presented as a graph on the WebUI and as an alert message on the CLI. The implementation of DBus in our solution keeps you constantly informed in other processes about events on OpenUSSD.

High Throughput Systems:

With implementation of high QA standards within the project, performance of the systems with various metrics are conducted and measured at the end of each day especially before a commit is done to the central repository.

Architecture

Below are 2 diagrams. Fig 1.1 shows what pertains in most organizations who have had to integrate into USSD gateways. Fig 1.2 shows how OpenUSSD solves the problem.

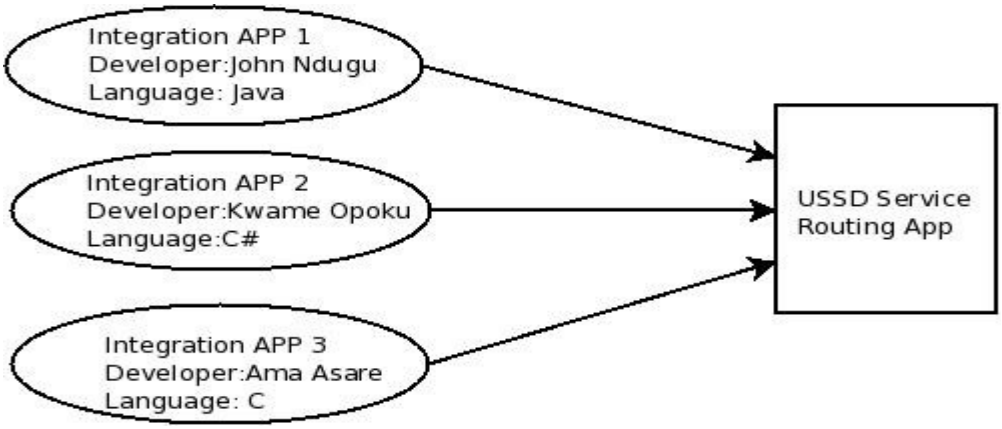

FIG 1.1

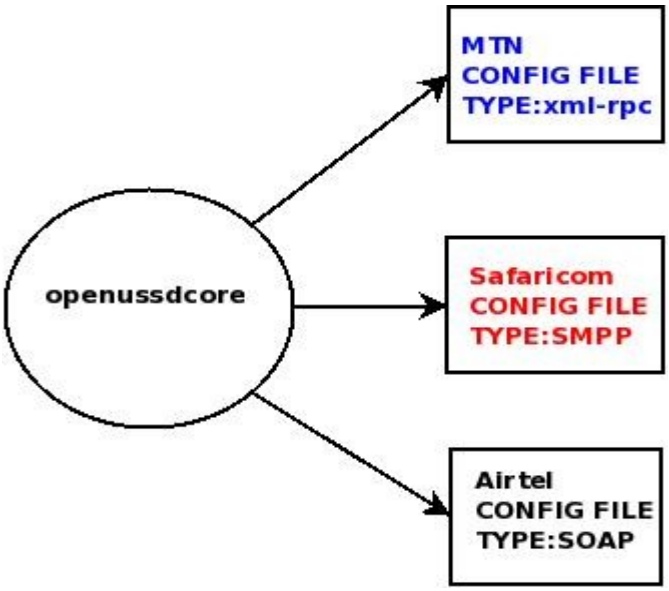

FIG 1.2

OpenUSSD uses an approach with the implementation of multiprocessing to provide high throughout services. It reads a single configuration that uses a key-value pair making its setup very simple. This eliminates the need to write and some times rewrite codes.

Libraries Required

PySMPP

ConfigParser