

Ouroboros

Embedded Web Server Automation

Team Cobra
Jim Kuglics
Andrew Lyne
Gabriel Marcano
Jared Smith

R·I·T

Academic Year
2014 - 2015



Software Engineering
Rochester Institute of Technology

Coach: Rick Weil



Sponsor: Nathan Ransom

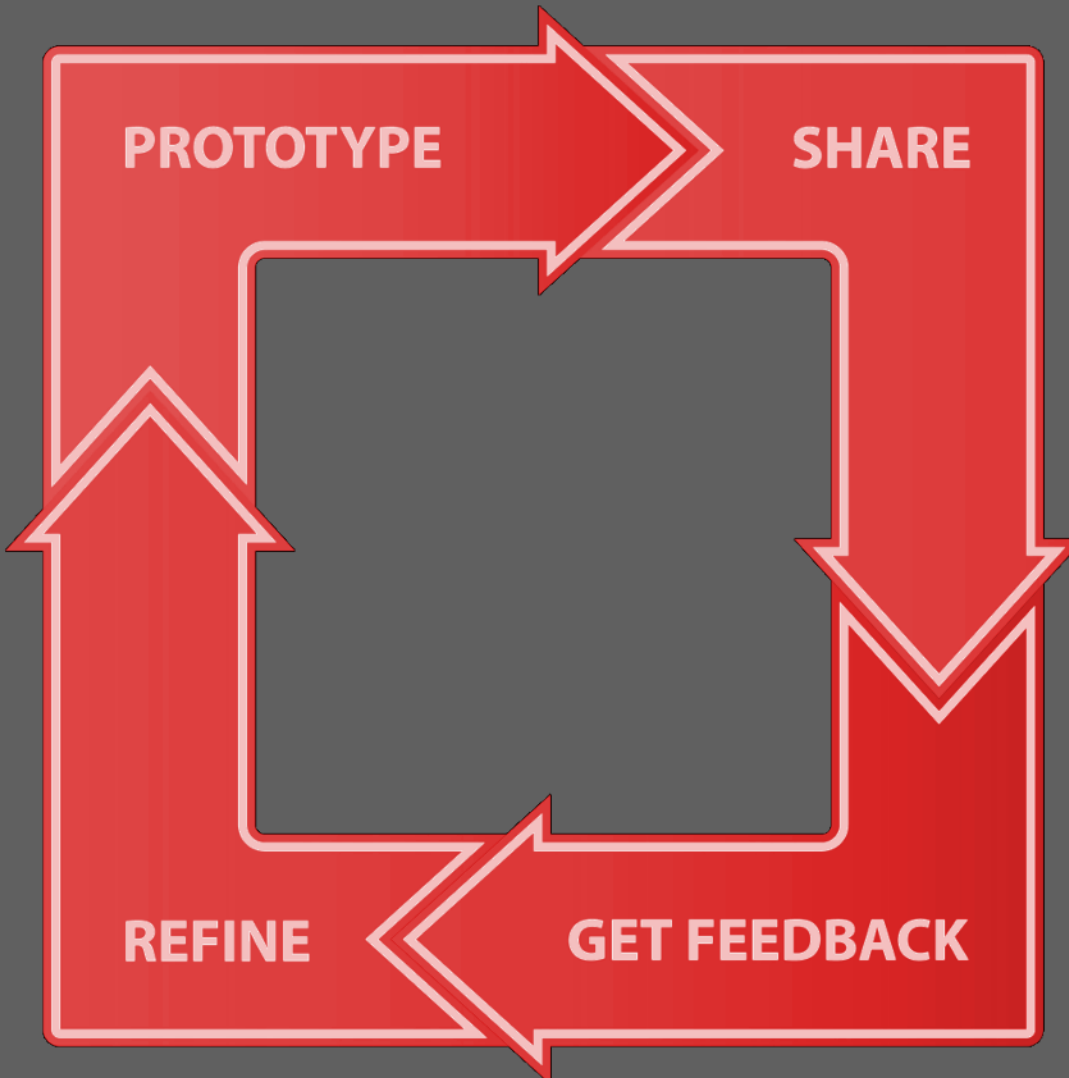
Problem

The process for testing networked embedded devices can be complicated. Developers must spend time setting up a web/data server on their target platform before development and testing may commence, adding time and complexity to the development process.

Ouroboros addresses this problem by offering a platform to generate custom web servers for embedded devices. Given an input configuration file, Ouroboros generates the code for the server which can then be compiled and run on the target platform.

Iterative Process

- Stable requirements
- Focus on customer feedback
- Continuous intergration testing
- Working build easily accessible
- Gradually build final product



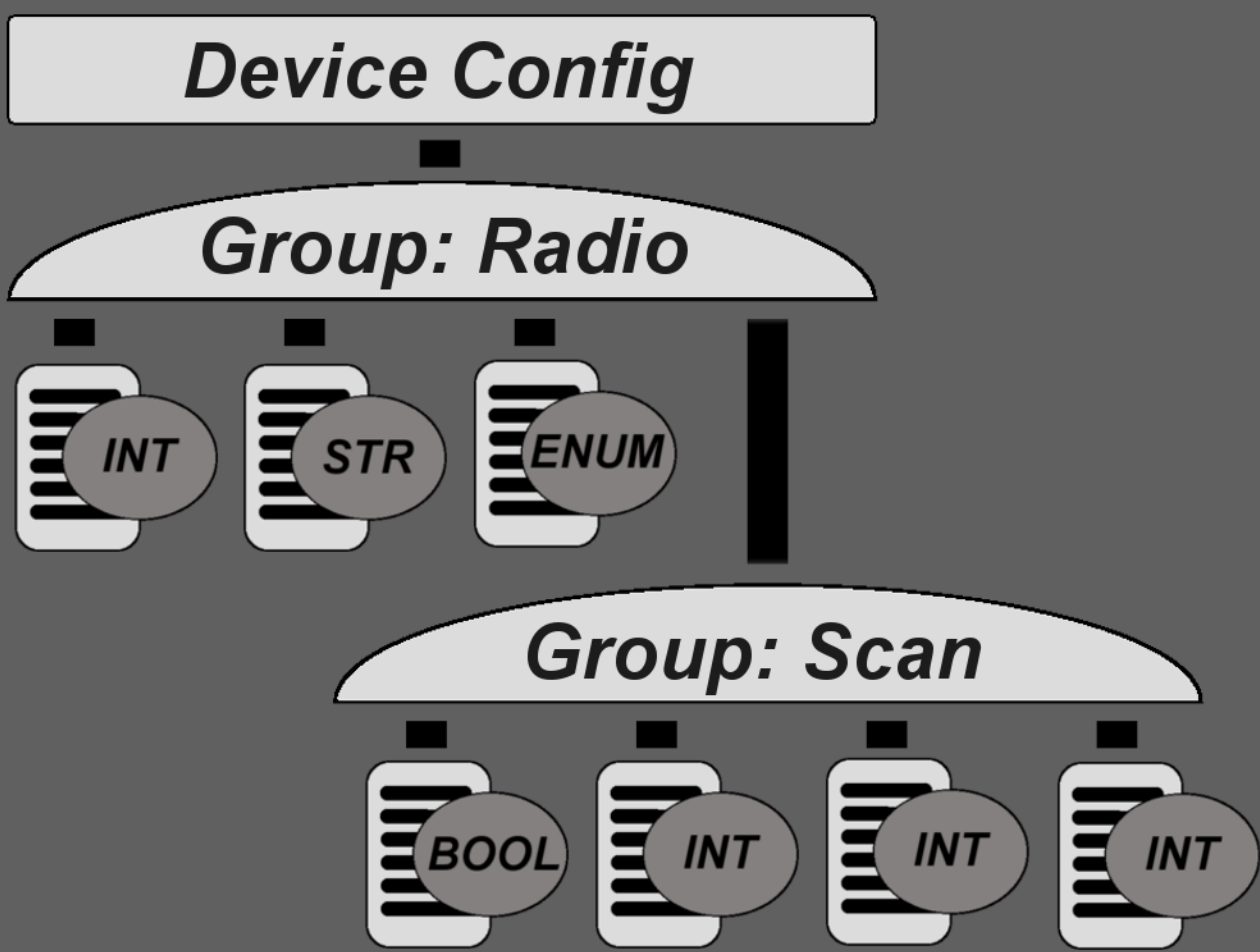
1. Configuration

Data organized into tree structure using XML

Groups Contain multiple data fields and other groups.

Fields Several pre-defined field types such as int, string, boolean and enum.

Custom Fields Can be created by user for more flexibility

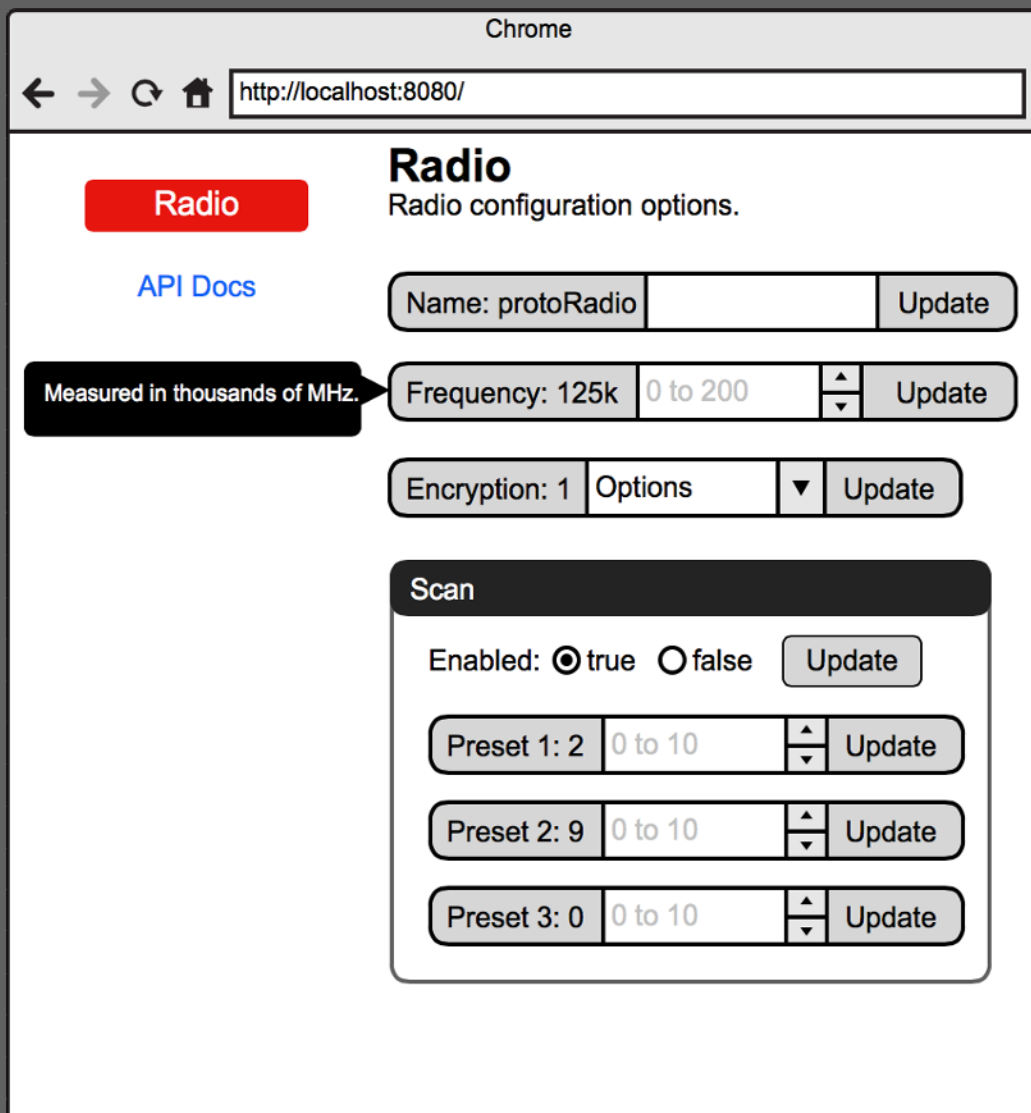


4. Interaction

The interfaces share much of the same functionality, with the exception of the Web UI which has an additional API Docs page and does not allow for registering of callbacks.

The shared functionality includes:

- Read/Write attributes of device's internal state
- Invoking function calls defined in input file.



2. Generation

- XML is validated against schema and checked for logical errors

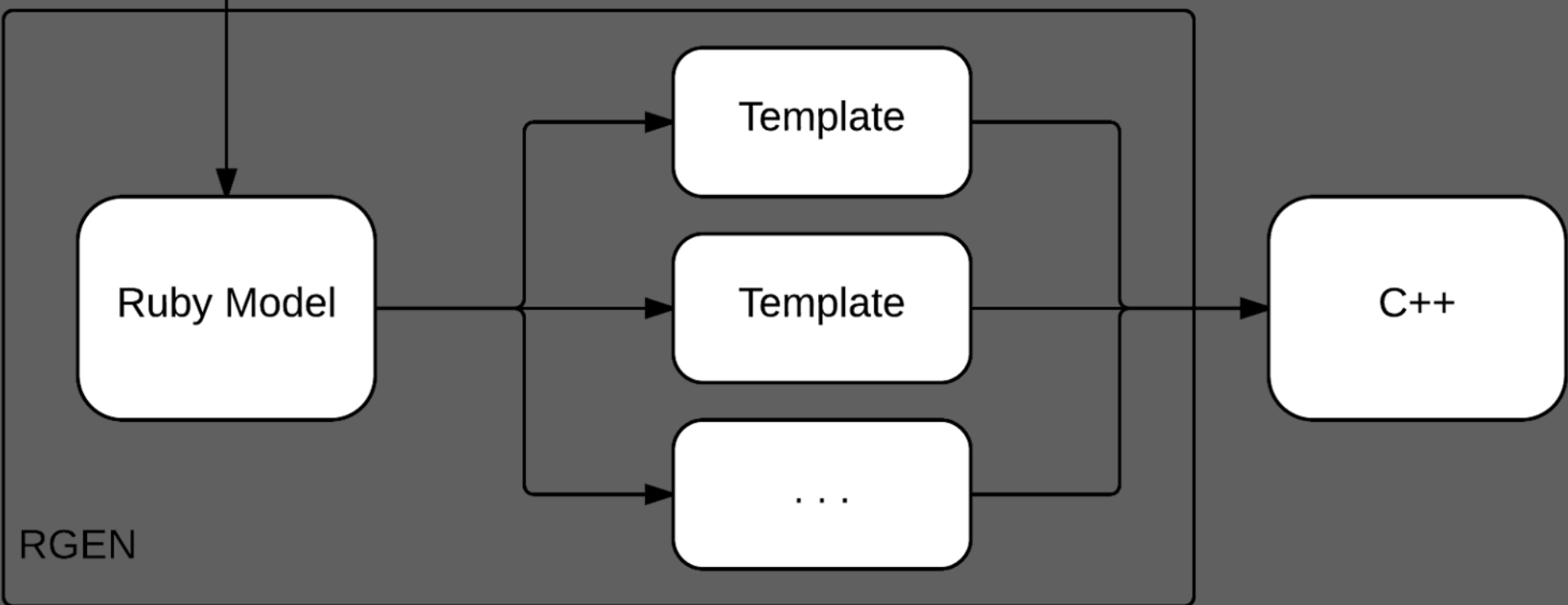
- Ruby model is built from XML

- Model is passed to templates

- C++ code is generated from templates

Data Flow:

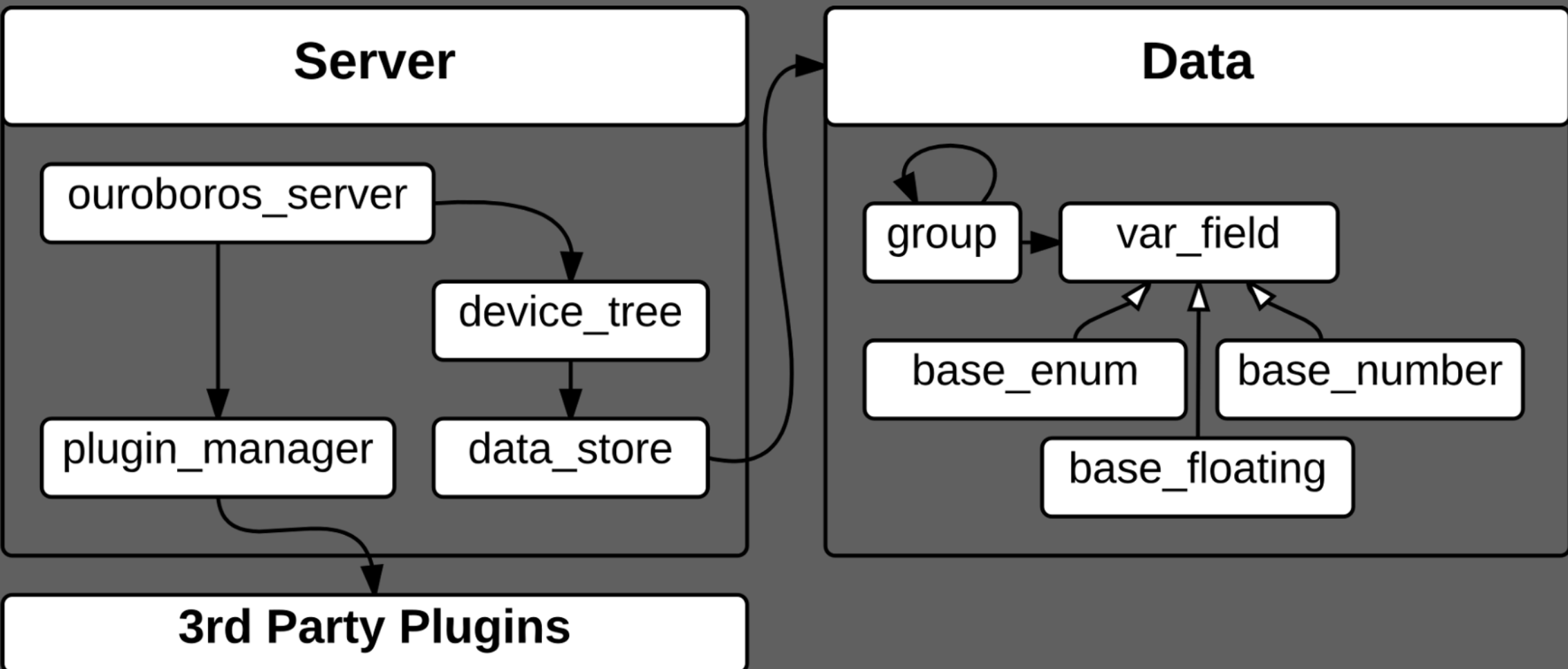
C++ code output is used to build the server



3. Execution

- Based on embedded Mongoose webserver
- C++98/03 compatible
- Accessible via C++ and REST APIs, and HTTP

Radio Server Runtime Component Model:



Technologies

