Routery Turris a jejich ekosystém I

Novinky ve Forisu, jak na pluginy

Štěpán Henek ▪ stepan.henek@nic.cz ▪ 2018–11–16

CZ.NIC | SPRÁVCE DOMÉNY CZ

První commit

```
commit c42a021a7f6249ed31fe1bc87d7e19aeb1ed8479
Author: Jan Čermák <jan.cermak@nic.cz>
Date:   Thu Oct 24 13:49:34 2013 +0200

    first commit of somewhat-stable code
    (see comment below)

    The previous development was very hectic and the
    code was not living in a standard version control
    repository. Now the changes are starting to be more
    meaningful (read: half of the files does not change
    within few minutes) and it makes sense to put it into
    a Git repository to track the changes. It still does
    contain a lot of development-related stuff, but most
    of the code should be "final".
```

Refaktorování

Foris před cca rokem a půl:

- python2
- bottle framework
- stpl šablony
- 2 web aplikace
- (n)uci backend
- pluginy ve složce

Refaktorování

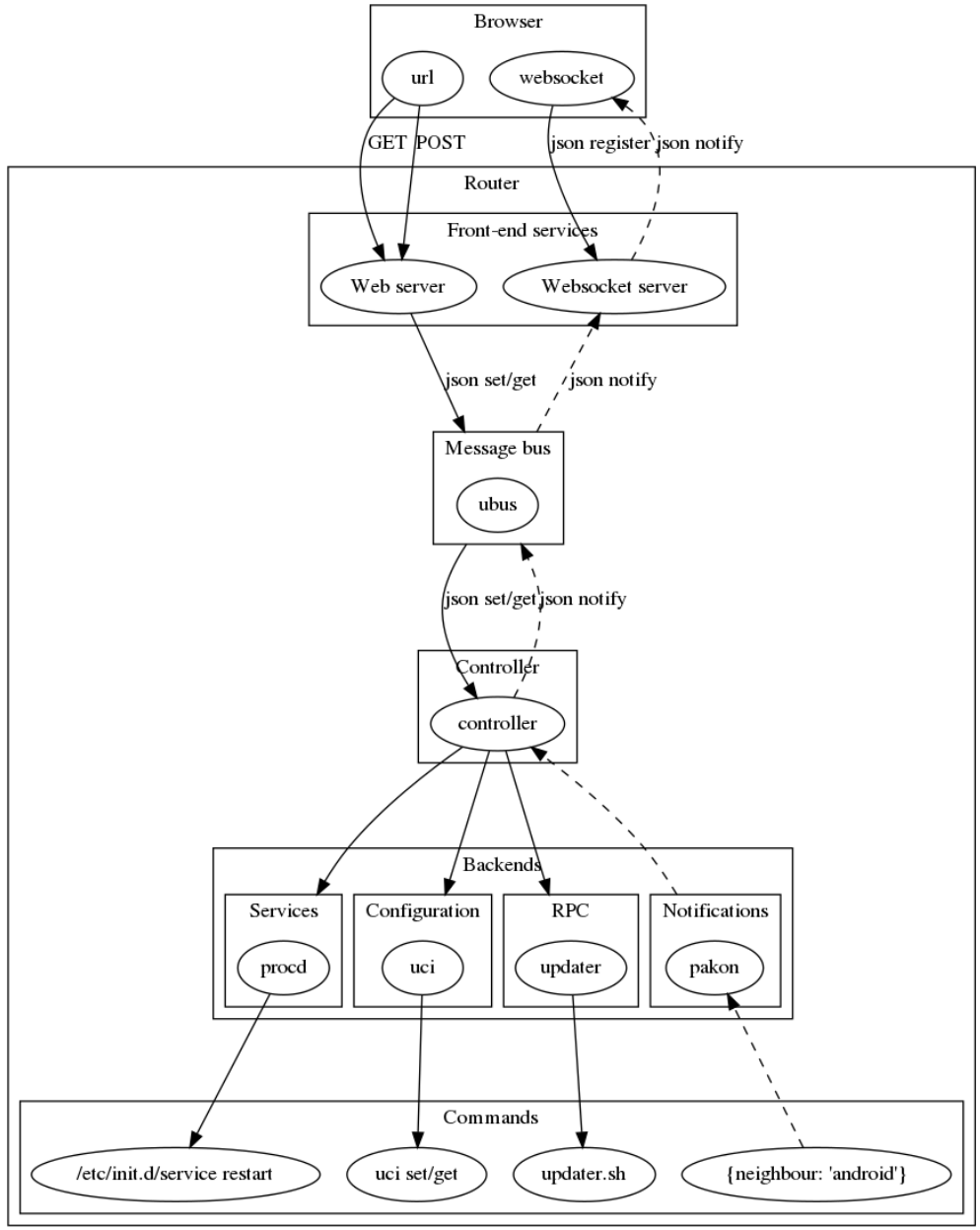
Foris nyní:

- ~~python2~~ python3
- bottle framework
- ~~stpl šablony~~ jinja2 šablony
- ~~2 web aplikace~~ 1 web aplikace
- ~~(n)uci backend~~ foris-controller backend
- ~~pluginy ve složce~~ pluginy jsou pythoní balíky
- integrace websocketů pro notifikace
- komunitní překlady přes weblate

Požadavky

- oddělení backendu a frontendu
- dobře definované API s kontrolou validity
- nahraditelnost komponent
- možnost běhu na více systémech (OpenWRT, Debian, . . .)
- možnost využití různých frontendu (web, mobilní aplikace)

Architektura



Architektura 1

Message bus

- předávání zpráv (požadavek, odpověď, notifikace)
- více implementací
  - ubus
  - unix-socket (spíš testovací)

Architektura 1

Message bus

- předávání zpráv (požadavek, odpověď, notifikace)
- více implementací
  - ubus
  - unix-socket (spíš testovací)

Controller

- přijímá požadavky na message busu
- posílá odpovědi do message busu
- posílá notifikace do message busu
- kontroluje schéma zpráv
- umí použít víc backendů
  - OpenWRT
  - mock (testovací backend)

Architektura 2

Websocket server

- přijímá notifikace z message busu
- přeposílá přijaté notifikace přes websockety klientům
- autentizace řešená přes session objekt na ubusu

Architektura 2

Websocket server

- přijímá notifikace z message busu
- přeposílá přijaté notifikace přes websockety klientům
- autentizace řešená přes session objekt na ubusu

Web server

- webová aplikace pod lighttpd
- používá framework bottle (views, middleware, ...)
- obsluhuje webové požadavky
- posílá požadavky do message busu
- čeká na odpovědi z message busu
- autentizace klientů přes sessiony z ubusu

Architektura 3

Prohlížeč

- posílá HTTP požadavky Webserveru
- připojuje se na Websocket server a čte z něj notifikace
- pamatuje si cookie, kterou posílá do požadavků

Architektura 3

Prohlížeč

- posílá HTTP požadavky Webserveru
- připojuje se na Websocket server a čte z něj notifikace
- pamatuje si cookie, kterou posílá do požadavků

Uživatel

- žasne jak to celé (ne)funguje
- stěžuje/chválí nás na fóru
- píše nám milé zprávy na podporu

Dlouho trvající požadavky

Jak to řešit?

- prohlížeč čeká
- javascript v prohlížeči čeká
- polling v javascriptu (Are we there yet?)

Dlouho trvající požadavky

Jak to dělá Foris:

Dlouho trvající požadavky

Jak to dělá Foris:

1. browser [HTTP ajax request] $\rightarrow$ webserver
2. webserver [API request] $\rightarrow$ controller
3. webserver $\leftarrow$ [syncID in API reply] controller
4. browser $\leftarrow$ [syncID in json] webserver

Dlouho trvající požadavky

Jak to dělá Foris:

1. browser [HTTP ajax request] $\rightarrow$ webserver
2. webserver [API request] $\rightarrow$ controller
3. webserver $\leftarrow$ [syncID in API reply] controller
4. browser $\leftarrow$ [syncID in json] webserver
5. ...

Dlouho trvající požadavky

Jak to dělá Foris:

1. browser [HTTP ajax request] $\rightarrow$ webserver
2. webserver [API request] $\rightarrow$ controller
3. webserver $\leftarrow$ [syncID in API reply] controller
4. browser $\leftarrow$ [syncID in json] webserver
5. ...
6. websocketserver $\leftarrow$ [syncID + progress] controller
7. browser $\leftarrow$ [syncID + progress] websocketserver

Dlouho trvající požadavky

Jak to dělá Foris:

1. browser [HTTP ajax request] $\rightarrow$ webserver
2. webserver [API request] $\rightarrow$ controller
3. webserver $\leftarrow$ [syncID in API reply] controller
4. browser $\leftarrow$ [syncID in json] webserver
5. ...
6. websocketserver $\leftarrow$ [syncID + progress] controller
7. browser $\leftarrow$ [syncID + progress] websocketserver
8. ...

Dlouho trvající požadavky

Jak to dělá Foris:

1. browser [HTTP ajax request] $\rightarrow$ webserver
2. webserver [API request] $\rightarrow$ controller
3. webserver $\leftarrow$ [syncID in API reply] controller
4. browser $\leftarrow$ [syncID in json] webserver
5. ...
6. websocketserver $\leftarrow$ [syncID + progress] controller
7. browser $\leftarrow$ [syncID + progress] websocketserver
8. ...
9. websocketserver $\leftarrow$ [syncID + finished] controller
10. browser $\leftarrow$ [syncID + finished] websocketserver

Vývoj pluginů

- dva projekty - backend a frontend část
- typické pojmenování:
  - foris-controller-*jmeno*-module (backend)
  - foris-*jmeno*-plugin (frontend)
- vzorové projekty:
  - https://gitlab.labs.nic.cz/turris/foris-controller/tree/master/doc/examples
  - https://gitlab.labs.nic.cz/turris/foris/tree/master/examples
- používat virtualenv
- python $<=$ 3.6.0

Modul do Controlleru

Skládá se z:

- pythoního kódu
- jsonschémata
- testů

Modul do Controlleru

Skládá se z:

- pythoního kódu
- jsonschémata
- testů

Funkce:

- přijímá požadavky, které se ho týkají
- sahá do systému (když je potřeba)
- odpovídá na požadavky
- posílá notifikace

Modul do Controlleru - Formát zpráv

```
{
  "module": "sample", "action": "get_slices",
  "type": "request"
}

{
  "module": "sample", "action": "get_slices",
  "type": "reply", "data": {...}
}

{
  "module": "sample", "action": "set_slices",
  "type": "notification", "data": {...}
}
```

Modul do Controlleru - Validace přes jsonschema

```
{
  "oneOf": [
    {
      "description": "Request to Get number of slices",
      "properties": {
        "module": {"enum": ["sample"]},
        "kind": {"enum": ["request"]},
        "action": {"enum": ["get_slices"]}
      },
      "additionalProperties": false
    },]
}
```

- https://json-schema.org/understanding-json-schema/
- https://gitlab.labs.nic.cz/turris/foris-schema

Modul do Controlleru - Rozcestník a Handlery

Rozcestník (hlavní modul)

- přijímá příchozí zprávy (jednotné API)
- přehazuje je jednotlivým handlerům

Modul do Controlleru - Rozcestník a Handlery

Rozcestník (hlavní modul)

- přijímá příchozí zprávy (jednotné API)
- přehazuje je jednotlivým handlerům

Handler

- existují různé handlery (např. pro OpenWRT)
- volají se z něj funkce, které něco dělají na systému
  - čtení/aktualizace nastavení (UCI)
  - start/restart/stop služeb
  - pouštění příkazů

Modul do Controlleru - Backend funkce

Dají se využít z ruzných handlerů.
Např. zpracování výstupu z příkazu uptime bude stejné jak na Debianu, tak na OpenWRT.

- *cmdline* - wrappery nad voláním příkazů, včetně async příkazů
- *uci* - wrappery nad čtení/nastavování uci
- *files* - wrappery nad čtením souborů
- *services* - wrappery správou služeb
- navíc má každý modul má svou vlastní sadu funkcí

Modul do Controlleru - Příklad Rozcestníku

```
class SampleModule(BaseModule):

    def action_get_slices(self, data):
        return {"slices": self.handler.get_slices()}

@wrap_required_functions([
    'get_slices',
])
class Handler(object):
    pass
```

Modul do Controlleru - Příklad Handleru

```
from .. import Handler

logger = logging.getLogger(__name__)


class OpenwrtSampleHandler(Handler, BaseOpenwrtHandler):

    uci = SampleUci()

    @logger_wrapper(logger)
    def get_slices(self):
        return OpenwrtSampleHandler.uci.get_slices()
```

Modul do Controlleru - Funkce backendu

```
from foris_controller_backends import uci

class SampleUci():
    def get_slices(self):
        with uci.UciBackend() as backend:
            data = backend.read("sample")
        return int(uci.get_option_named(
            data, "sample", "data", "slices"))
```

Plugin do Forise

Skládá se z:

- pythoního kódu
- statických/generovaných souborů (sass/css/js)
- jinja2 šablon (html/js)

Plugin do Forise

Skládá se z:

- pythoního kódu
- statických/generovaných souborů (sass/css/js)
- jinja2 šablon (html/js)

Co vlastně dělá:

- obsluhuje HTTP požadavky
- volá API backendu
- skládá dohromady HTML
- skládá dohromady javascript

Plugin do Forise - Python

- definice hlavního formuláře
- výběr šablony, která se použije
- pořadí v hlavním menu
- obsluhování použadavků (GET/POST)
  - hlavního formuláře
  - custom akcí
  - ajax volání
- volání backend příkazů
- include assetů

Plugin do Forise - Kód pluginu

```
class SamplePlugin(ForisPlugin):
  PLUGIN_NAME = "sample"  # also shown in the url
  DIRNAME = os.path.dirname(os.path.abspath(__file__))
  PLUGIN_STYLES = ["css/sample.css"]
  PLUGIN_STATIC_SCRIPTS = ["js/sample.js"]
  PLUGIN_DYNAMIC_SCRIPTS = ["sample.js"]
  def __init__(self, app):
    super().__init__(app)
    add_config_page(SamplePluginPage)
```

Plugin do Forise - Kód stránky

```
class SamplePluginPage(ConfigPageMixin, SamplePlugin...):
  slug = "sample"
  menu_order = 90
  template = "sample/sample"
  template_type = "jinja2"
  def save(self, *args, **kwargs):
    return super().save(*args, **kwargs)
  def render(self, **kwargs):
    kwargs['PLUGIN_NAME'] = SamplePlugin.PLUGIN_NAME
    kwargs['PLUGIN_STYLES'] = SamplePlugin.PLUGIN_STYLES
    kwargs['PLUGIN_STATIC_SCRIPTS'] = SamplePlugin.PL...
    kwargs['PLUGIN_DYNAMIC_SCRIPTS'] = SamplePlugin.P...
    return super().render(**kwargs)
```

Plugin do Forise - Kód formuláře

```
def get_form(self):
  data = backend.perform("sample", "get_slices")
  if self.data:
    data.update(self.data)
  form = fapi.ForisForm("sample", data)
  section = form.add_section(name="main", title=_("..."))
  section.add_field(Number, name="slices", label=_("..."),
    required=True, validators=validators.InRange(2, 15))
  def form_cb(data):
    res = current_state.backend.perform("sample",
      "set_slices", {"slices": int(data["slices"])})
    return "save_result", res
  form.add_callback(form_cb)
  return form
```

Plugin do Forise - Šablona

```
{% extends 'config/base.html.j2' %}
{% block config_base %}
<div id="page-sample-plugin" class="config-page">
  <form action="{{ request.fullpath }}" method="post" ...>
    <input type="hidden" name="csrf_token" value="...">
    {% for field in form.active_fields %}
      {% include '_field.html.j2' %}
    {% endfor %}
    <button type="submit" name="send">
    {% trans %}Update configuration{% endtrans %}
    </button>
  </form>
</div>
{% endblock %}
```

Výzvy

Výzvy

- nový bus (přístupný ze sítě, používání vzdálených frontendů)

Výzvy

- nový bus (přístupný ze sítě, používání vzdálených frontendů)
- lepší integrace Forise s podporovanými aplikacemi (nextcloud)

Výzvy

- nový bus (přístupný ze sítě, používání vzdálených frontendů)
- lepší integrace Forise s podporovanými aplikacemi (nextcloud)
- přepis Forise (nahrazení Bottlu za Flask, bootstrap)

Výzvy

- nový bus (přístupný ze sítě, používání vzdálených frontendů)
- lepší integrace Forise s podporovanými aplikacemi (nextcloud)
- přepis Forise (nahrazení Bottlu za Flask, bootstrap)
- testy pro Forise (controller jich má poměrně dost)

Výzvy

- nový bus (přístupný ze sítě, používání vzdálených frontendů)
- lepší integrace Forise s podporovanými aplikacemi (nextcloud)
- přepis Forise (nahrazení Bottlu za Flask, bootstrap)
- testy pro Forise (controller jich má poměrně dost)
- nezbláznit se z toho

Odkazy

- https://gitlab.labs.nic.cz/turris/foris-demo
- https://gitlab.labs.nic.cz/turris/foris-controller/tree/master/doc/examples
- https://gitlab.labs.nic.cz/turris/foris/tree/master/examples
- https://bottlepy.org/docs/dev/
- https://json-schema.org/understanding-json-schema/
- http://jinja.pocoo.org/docs/2.10/

Díky a pozornost

Otázky?

Štěpán Henek ▪ stepan.henek@nic.cz