Real-time charging for OpenSIPS 2.1 using CGRateS

Dan Christian Bogos
dan.bogos@itsyscom.com

OpenSIPS Summit Amsterdam, May 2015

Our Background

Located in Bavaria/Germany, over 8 years of experience with architecting server side solutions in VoIP environment

Platform implementations covering both wholesale and retail business categories

Responsibly understanding real-time processing constrains and the seriousness of live system outages

About CGRateS

Charging/Billing engine
Plug-able into existing billing infrastructure
Accommodate new components into ISP/ITSP network (eg: add new VoIP switch, SMS Service, Data stream)

Modular architecture
Easy to enhance by rewriting specific components
JSON/HTTP/GOB RPC API

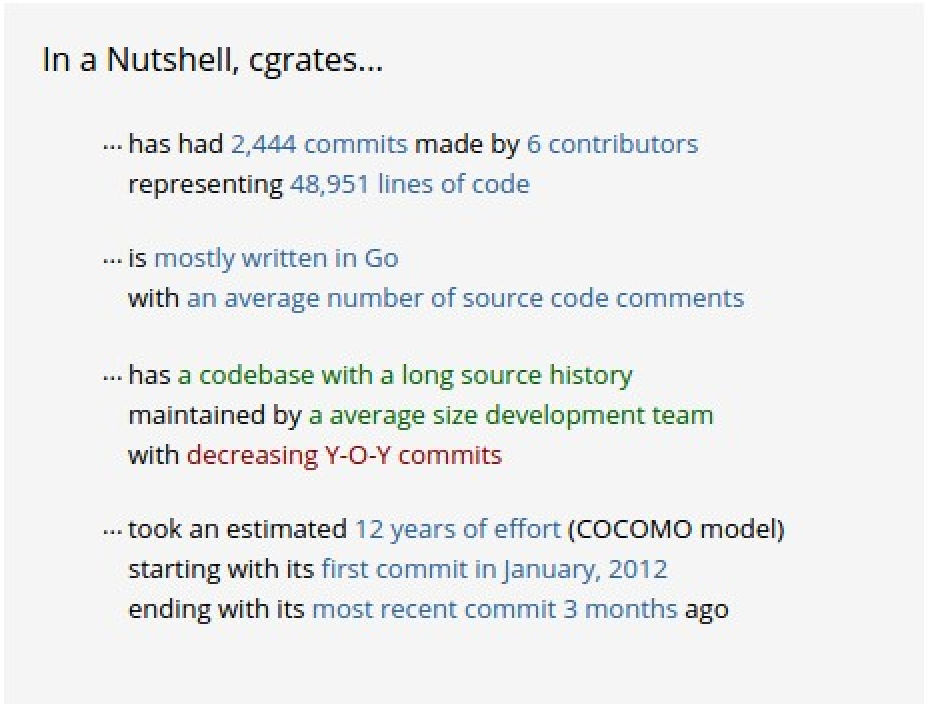
Performance Oriented
Built-in transactional cache system (data ageing, live counters)
Asynchronous processing with micro-threads

Feature-rich
Multi-tenancy, derived charging, account bundles, LCR, CDRStats, rates history, etc

Test driven development
Aprox. 900 tests as part of the build system

CGRATES
carrier grade realtime charging

About CGRateS (2)



In a Nutshell, cgrates...

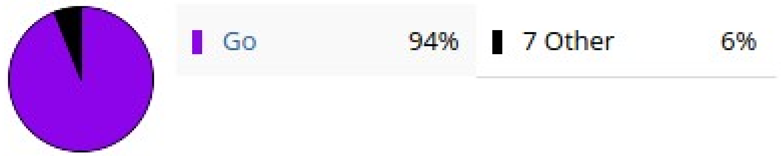
... has had 2,444 commits made by 6 contributors representing 48,951 lines of code

... is mostly written in Go with an average number of source code comments

... has a codebase with a long source history maintained by a average size development team with decreasing Y-O-Y commits

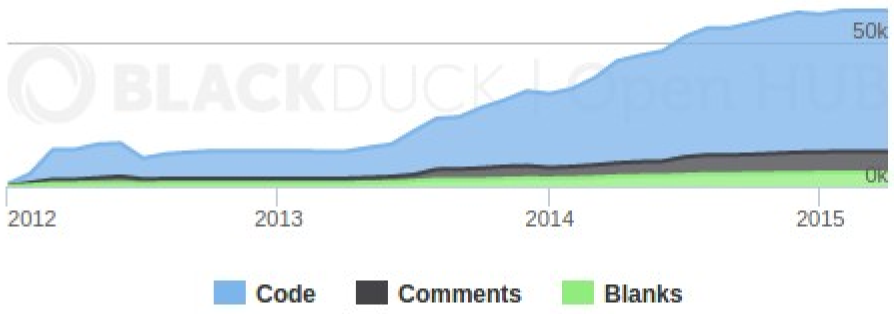
... took an estimated 12 years of effort (COCOMO model) starting with its first commit in January, 2012 ending with its most recent commit 3 months ago

Languages



Go	94%
7 Other	6%

Lines of Code

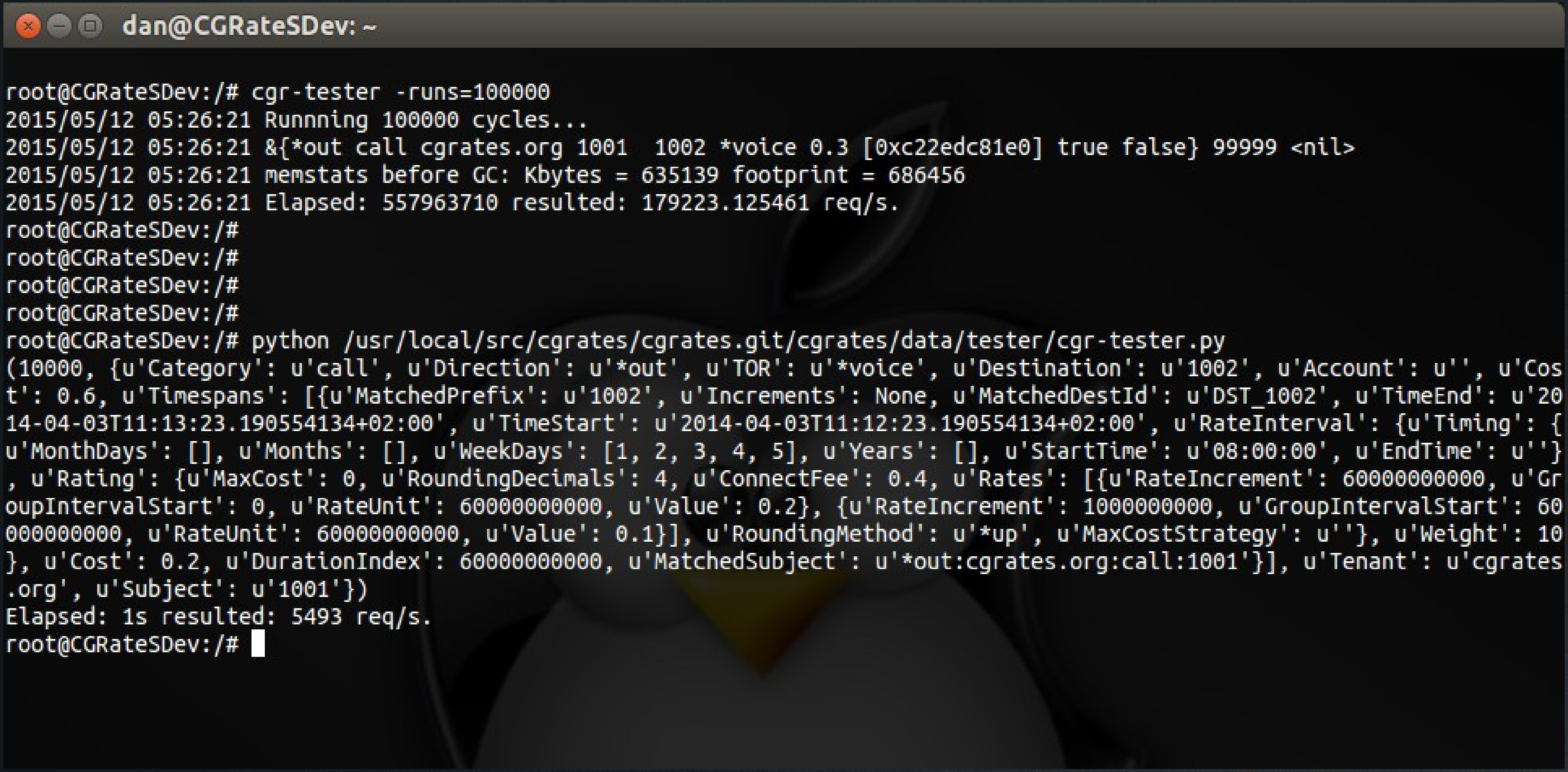


Actively maintained

*stats provided by openhub.net

carrier grade realtime charging

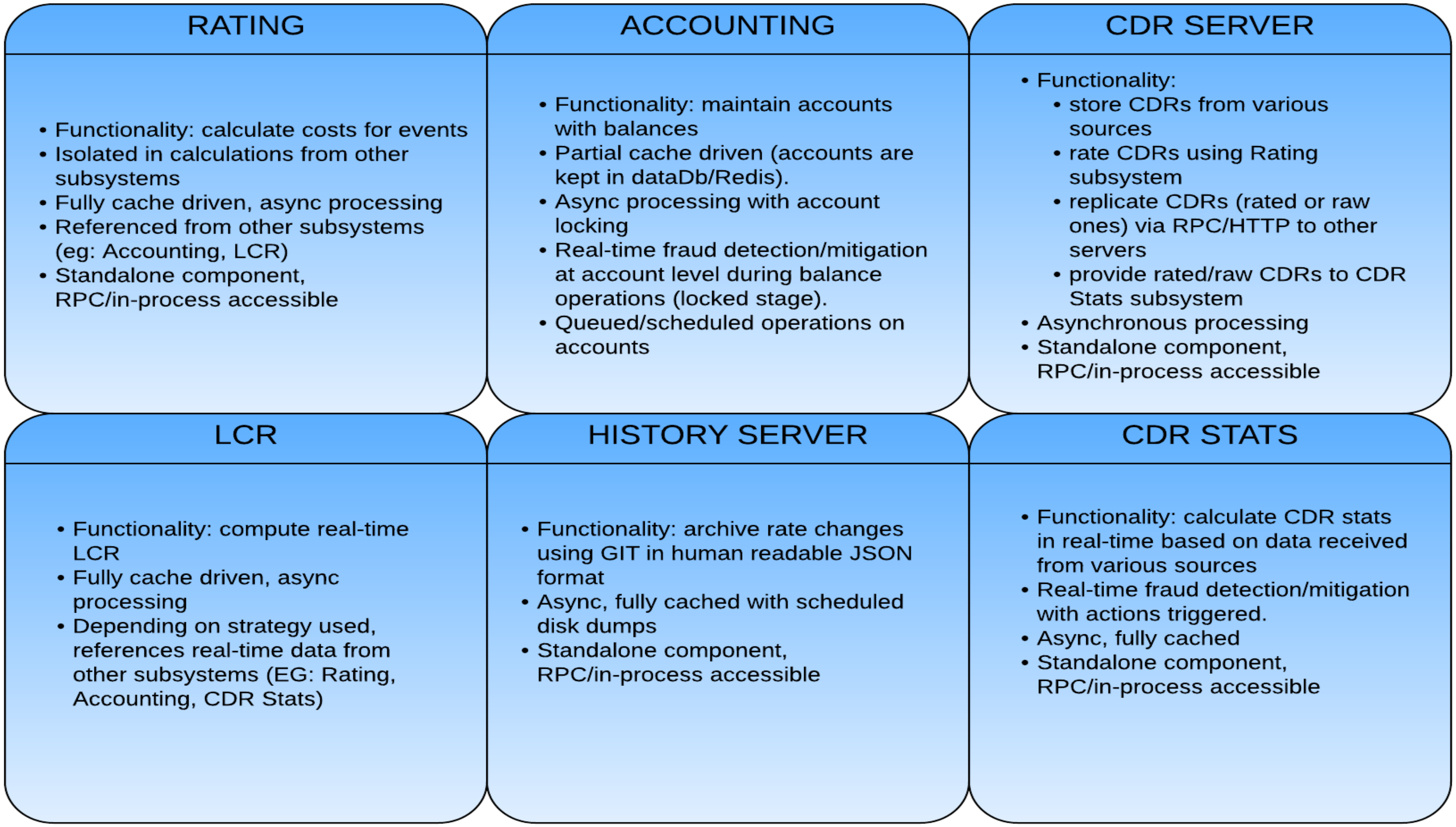
About CGRateS (3)



```
root@CGRateSDev:/# cgr-tester -runs=100000
2015/05/12 05:26:21 Runnning 100000 cycles...
2015/05/12 05:26:21 &{*out call cgrates.org 1001  1002 *voice 0.3 [0xc22edc81e0] true false} 99999 <nil>
2015/05/12 05:26:21 memstats before GC: Kbytes = 635139 footprint = 686456
2015/05/12 05:26:21 Elapsed: 557963710 resulted: 179223.125461 req/s.
root@CGRateSDev:/#
root@CGRateSDev:/#
root@CGRateSDev:/#
root@CGRateSDev:/#
root@CGRateSDev:/# python /usr/local/src/cgrates/cgrates.git/cgrates/data/tester/cgr-tester.py
(10000, {u'Category': u'call', u'Direction': u'*out', u'TOR': u'*voice', u'Destination': u'1002', u'Account': u'', u'Cos
t': 0.6, u'Timespans': [{u'MatchedPrefix': u'1002', u'Increments': None, u'MatchedDestId': u'DST_1002', u'TimeEnd': u'20
14-04-03T11:13:23.190554134+02:00', u'TimeStart': u'2014-04-03T11:12:23.190554134+02:00', u'RateInterval': {u'Timing': {
u'MonthDays': [], u'Months': [], u'WeekDays': [1, 2, 3, 4, 5], u'Years': [], u'StartTime': u'08:00:00', u'EndTime': u''}
, u'Rating': {u'MaxCost': 0, u'RoundingDecimals': 4, u'ConnectFee': 0.4, u'Rates': [{u'RateIncrement': 60000000000, u'Gr
oupIntervalStart': 0, u'RateUnit': 60000000000, u'Value': 0.2}, {u'RateIncrement': 1000000000, u'GroupIntervalStart': 60
000000000, u'RateUnit': 60000000000, u'Value': 0.1}], u'RoundingMethod': u'*up', u'MaxCostStrategy': u''}, u'Weight': 10
}, u'Cost': 0.2, u'DurationIndex': 60000000000, u'MatchedSubject': u'*out:cgrates.org:call:1001'}], u'Tenant': u'cgrates
.org', u'Subject': u'1001'})
Elapsed: 1s resulted: 5493 req/s.
root@CGRateSDev:/#
```

Fast and … very fast

CGRATES
carrier grade realtime charging



RATING

- Functionality: calculate costs for events
- Isolated in calculations from other subsystems
- Fully cache driven, async processing
- Referenced from other subsystems (eg: Accounting, LCR)
- Standalone component, RPC/in-process accessible

ACCOUNTING

- Functionality: maintain accounts with balances
- Partial cache driven (accounts are kept in dataDb/Redis).
- Async processing with account locking
- Real-time fraud detection/mitigation at account level during balance operations (locked stage).
- Queued/scheduled operations on accounts

CDR SERVER

- Functionality:
  - store CDRs from various sources
  - rate CDRs using Rating subsystem
  - replicate CDRs (rated or raw ones) via RPC/HTTP to other servers
  - provide rated/raw CDRs to CDR Stats subsystem
- Asynchronous processing
- Standalone component, RPC/in-process accessible

LCR

- Functionality: compute real-time LCR
- Fully cache driven, async processing
- Depending on strategy used, references real-time data from other subsystems (EG: Rating, Accounting, CDR Stats)

HISTORY SERVER

- Functionality: archive rate changes using GIT in human readable JSON format
- Async, fully cached with scheduled disk dumps
- Standalone component, RPC/in-process accessible

CDR STATS

- Functionality: calculate CDR stats in real-time based on data received from various sources
- Real-time fraud detection/mitigation with actions triggered.
- Async, fully cached
- Standalone component, RPC/in-process accessible

CGRateS subsystems

CGRATES
carrier grade realtime charging

RATING

Highly configurable rating

Connect fees, rate units, rate increments, rates grouping, various rounding methods, configurable decimals in costs, maximum cost per destination with hit strategy

Performance oriented

Fully cached
Asynchronous processing

Rating profile scheduling

Derived Charging

Reseller/distributors chaining or inbound/outbound traffic charging

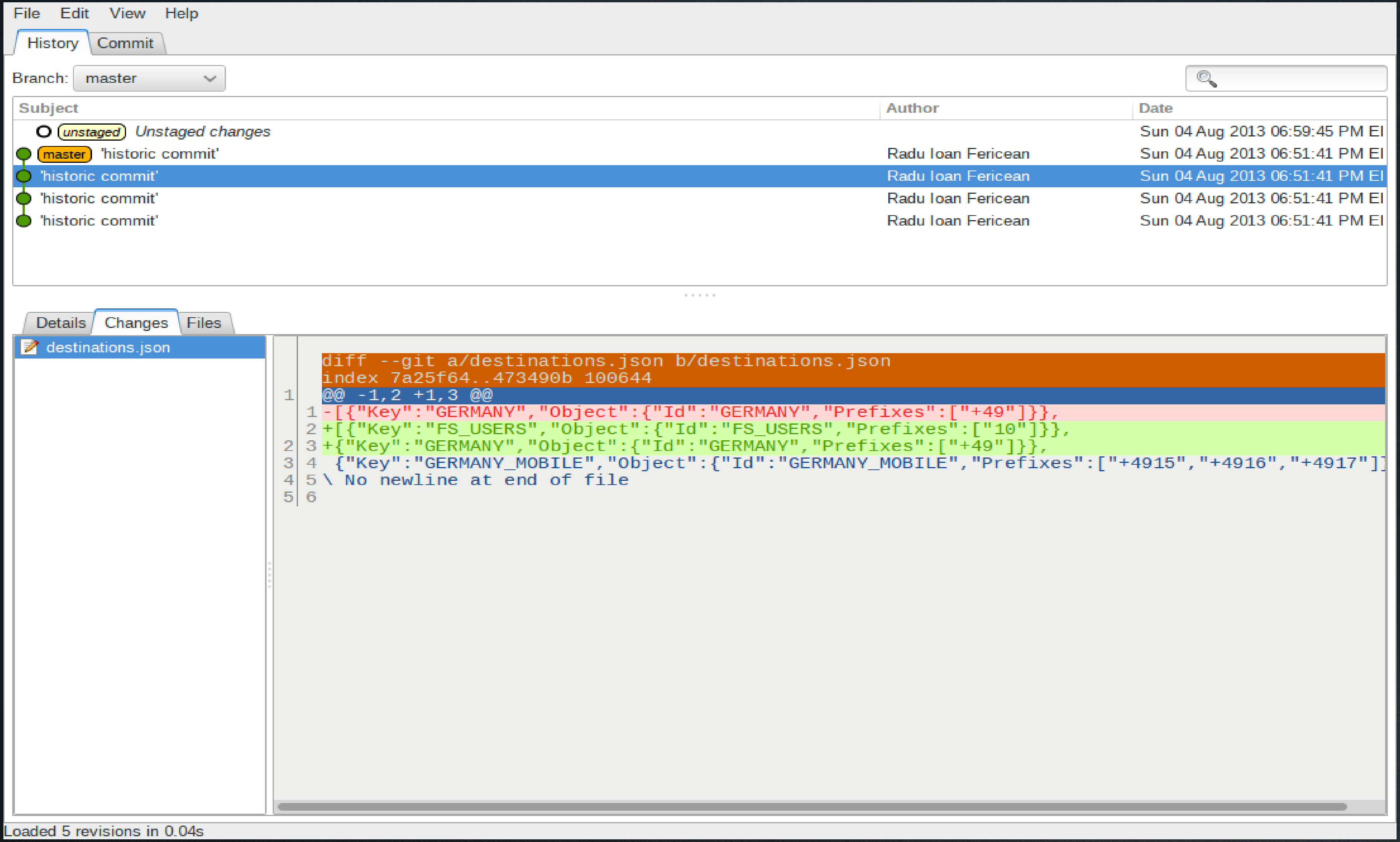
RATING (2)

Multiple TypeOfRecord support
(eg: *voice, *data, *sms)

Multiple Category filters for same TOR
(eg: calls, premium_calls, inbound_calls)

Multiple rating subjects with fallback
(useful for example with roaming CDRs)

Rating Aliases



Subject	Author	Date
unstaged *Unstaged changes*		Sun 04 Aug 2013 06:59:45 PM EI
master 'historic commit'	Radu Ioan Fericean	Sun 04 Aug 2013 06:51:41 PM EI
'historic commit'	Radu Ioan Fericean	Sun 04 Aug 2013 06:51:41 PM EI
'historic commit'	Radu Ioan Fericean	Sun 04 Aug 2013 06:51:41 PM EI
'historic commit'	Radu Ioan Fericean	Sun 04 Aug 2013 06:51:41 PM EI

```
diff --git a/destinations.json b/destinations.json
index 7a25f64..473490b 100644
@@ -1,2 +1,3 @@
-[{"Key":"GERMANY","Object":{"Id":"GERMANY","Prefixes":["+49"]}},
+[{"Key":"FS_USERS","Object":{"Id":"FS_USERS","Prefixes":["10"]}},
+{"Key":"GERMANY","Object":{"Id":"GERMANY","Prefixes":["+49"]}},
 {"Key":"GERMANY_MOBILE","Object":{"Id":"GERMANY_MOBILE","Prefixes":["+4915","+4916","+4917"]]
\ No newline at end of file
```

Loaded 5 revisions in 0.04s

Git powered History Server

CGRATES
carrier grade realtime charging

ACCOUNTING

Prepaid, Postpaid, Pseudo-prepaid controller

Account Monitors through ActionTriggers
Balance Monitors (min/max)
Counters Monitors (min/max) – eg: Usage per Destination
Synchronous and Asynchronous Actions triggered

Session emulation
Through DerivedCharging

Accounts Aliases

ACCOUNTING (2)

Unlimited Balances per Account

*voice, *data, *sms, *monetary
Balance selection prioritisation through weights
Various bundle combinations

Shared/Group Balances

Balance lifetime controls

Eg: balance expires or balance is active on specific time intervals

Concurrent sessions per account

Balance reservation in chunks of debit interval
Balance refunds
Debit sleep when needed

ACCOUNTING - Fraud Detection

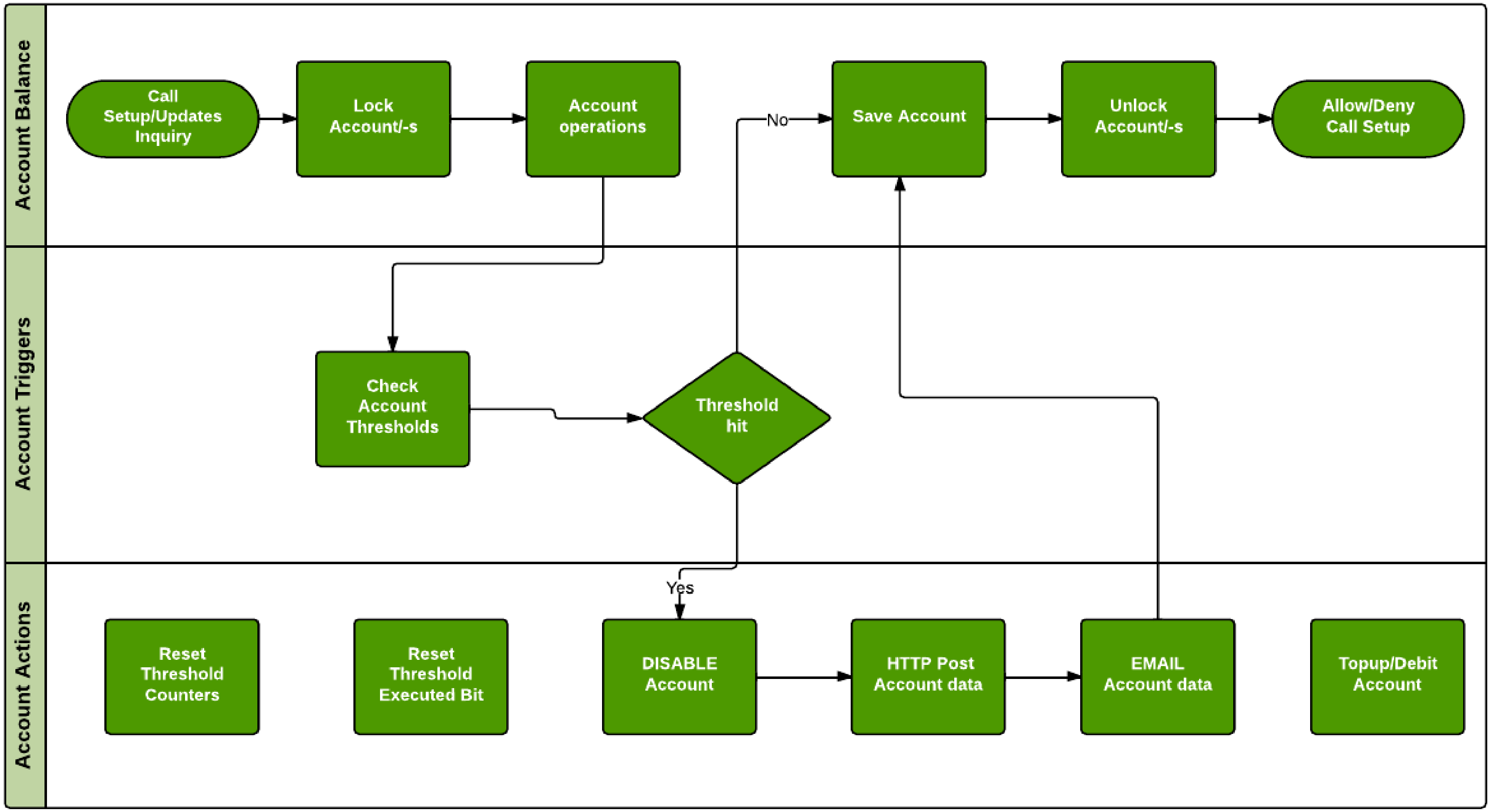
Part of Accounting Subsystem
Tightly integrated, balance operations cannot avoid it

Balances monitoring
Minimum & maximum balance monitors

Counters monitoring
Minimum & maximum counter monitors

Scheduler integration
One-time, recurrent triggers

Synchronous & Asynchronous Actions



Account handling logic

carrier grade realtime charging

CDR SERVER

Realtime CDR Server
Accessible Internal, GOB, JSON, HTTP-JSON, HTTP-REST interfaces

Offline CDR Import (eg: csv format)
Automated via Linux inotify or scheduled
Simultaneous folders monitored with multiple import templates per folder

Zero configuration CDR Sources
FreeSWITCH
Kamailio
OpenSIPS

carrier grade realtime charging

CDR SERVER (2)

Derived Charging support

Real-time CDR replication
Raw or Rated CDRs

CDR Exporter
CSV, Fixed Length Fields, Combined
Export templates

CDR STATS

Standalone component

Internally or remotely accessible
Performance oriented

RawCDR and RatedCDR sources

Multiple Stats Queues

Per server and individually configurable stat queues for same CDR

Highly configurable Stats Queues

QueueLength, TimeWindow, Metrics
CDR Field Filters

Individually configured ActionTriggers

One-time, recurrent triggers
Synchronous & Asynchronous Actions executed
Part of the Fraud Detection mechanism

carrier grade realtime charging

LCR

Core component logic
Internally or remotely accessible through APIer or RATER components
Performance oriented, fully cached

Advanced profile selection mechanism
Filter on Direction,Tenant,Category,Account,Subject,Destination
Weight based prioritization

Extended functionality through multiple strategies
*static, *least_cost, *highest_cost, *qos_thresholds, *qos
Flexible strategy parameters

Tightly coupled with ACCOUNTING subsystem
Provide LCR over bundles

Integrate traffic patterns
Compute LCR for specific call duration

CGRates Peripherals

APIer (RPC server)

Tariff plan and Account management
Export commands form internal components (Eg: get_cdrs, export_cdrs, etc)
Partial and full rates/accounts reload without restarts

Console

Interactive and non-interactive
History
Help
Command auto-completion

Loader

CSV Imports

Tester

OpenSIPS Integration

Multiple integration mechanisms

Based on traffic profile

Shared data through pseudovariables

REST_CLIENT for call authorization, LCR

HTTP-JSON RPC Request/Answer

EVI ACC_ACCOUNTING

*prepaid, *pseudoprepaid, *postpaid, *rated

EVI E_ACC_CDR

*pseudoprepaid, *postpaid, *rated

CDR.csv processing

*pseudoprepaid, *postpaid, *rated

OpenSIPS Real-time Prepaid

Call Authorization
Async/sync support through the user of rest_client
Sets maximum call duration through dialog timeout

Call disconnect
Executed through mi_datagram by CGR SessionManager

Call Start
Out of E_ACC_EVENT via event_datagram to CGR-SM
Starts debit loop in case of prepaid calls – real or emulated ones

Call Stop/Missed
Out of E_ACC_EVENT/E_ACC_MISSED_EVENT via event_datagram to CGR-SM
Stops debit loop
Writes CDRs

OpenSIPS Real-time Prepaid - DEMO

Simple call handling

Explanation of opensips.cfg
Monitor traffic exchanged between OpenSIPS and CGRateS
Call auth and LCR processing
CDR Export via cgr-console

Advanced call handling

Simultaneous calls out of same account
Fraud detection with automatic mitigation example

Where to go from here

Website
http://www.cgrates.org

Documentation
http://cgrates.readthedocs.org

Code + issues tracker
https://github.com/cgrates/cgrates

Support
Google group: CGRateS
IRC Freenode: #cgrates

CGRATES
carrier grade realtime charging

Thank you!

Questions?

CGRATES
carrier grade realtime charging