People like Dieter, Holger, Daniel and I have invested lots of our spare time in the early years. So funding for that came from other paid work, and Osmocom projects were a hobby. It was a lot of fun, but it’s not a sustainable basis for the scope of the projects at this time.

Today, our development is driven by

- what people contribute patches for, and/or
- what companies and consultants in the project get development contracts for (e.g. sysmocom customers)

This means in general that very little will happen, unless somebody commits to putting resources at it!
Major Roadmap Areas

- Splitting up the NITB
- Code Quality
- Testing
- Integration between 2G and 3G
- External Interfaces
OsmoNITB

- has been at the heart of most Osmocom based cellular networks
- is generally doing a good job, but suffers from some issues
- NITB includes BSC, but BSC code also used to build OsmoBSC
  - No way to test OsmoBSC without a proprietary MSC :(
- NITB includes sqlite3 based HLR
  - synchronous/blocking access from single-threaded OsmoNITB
  - sqlite3 (particularly via DBI) scalability issues
RIP NITB; Long Live OsmoMSC

- New **OsmoMSC** = **OsmoNITB** - **BSC** - **HLR**
- New OsmoHLR (GSUP interface only)
- 3GPP A-over-IP for OsmoBSC + OsmoMSC
  - New SCCP+M3UA stack, libosmo-sigtran
- M3UA based IuCS/IuPS in HNB-GW, MSC, SGSN
  - used to be SUA, as it was simpler initially
- Single code base/branch for 2G and 3G

→ typical 2G setup will become **OsmoBSC + OsmoMSC + OsmoHLR**
OsmoSTP

Signal Transfer Point, part of libosmo-sccp

- M3UA + SUA support
- Connectionless and Connection Oriented SCCP
- Used primarily for RAN-CN interface (A, Iu) so far
osmo_fsm + osmo_prim

osmo_fsm

structured approach to state machines

osmo_prim

structured approach towards primitives at SAP between layers
osmo_fsm

Generalized finite state machine (FSM) abstraction

- pure C language, no pre-processor / code-generator
- description of states, permitted events in a state, permitted output states
- built-in timer support. Default action on expiry: Destroy FSM
- parent/children relationship, allows hierarchy of FSMs

Much more maintainable and deterministic than the many implicit or non-existing state machines in older code
osmo_fsm so far

- Ericsson OM2000 managed objects
- Connection-Oriented SCCP
- M3UA/SUA ASP + AS state machine
- All VLR FSMs (LU, Auth, ...) in OsmoMSC
- AMR DTX in OsmoBTS
osmo_fsm candidates

- LAPD + LAPDm code
- A-bis OML managed objects
- GSM Call Control (maybe even CAMEL-like FSMs?)

More osmo_fsm Would greatly improve code quality, testability and maintainability

- but who wants to invest in that? Any volunteers?
Testing

- unit test coverage of most code is poor, needs attention
  - coverage of new code much better than old code
- osmo_fsm state introspection via CTRL enables better testing
- end-to-end system testing software: osmo-gsm-tester
osmo-gsm-tester

- python language for managing BTSs + Modems
  - supports different BTS models
  - can execute suites of test cases on each software version for all BTS models
- RF cabling between BTSs and Modems to avoid RF interference

→ GOAL: Daily functional testing on all supported platforms/configs
osmo-gsm-tester

- core python infrastructure working
- jenkins integration for testing new builds working
- modem hardware being replaced due to poor ofono support for SL8082

TODO:

- modem integration using better supported hardware
- more actual test cases beyond SMS + LU
- 3G support
External Interfaces

- **VTY** is a human interface, not intended for consumption by software
  - text output syntax not guaranteed stable
  - inefficient
  - commands get added during development to help developers

- **CTRL** interface is the programmatic interface
  - but developers don’t need it during development
  - all commands so far added due to user request/requirement
    - let us know what you need exposed!
  - only way to fundamentally change this is to automatically export things without extra effort

⇒ Roadmap is to put more effort into CTRL completeness
Grand Unified Config Theory

Big Wishlist item:

- Unified configuration store / MIB with
  - automatic VTY printer/parser generation
  - automatic CTRL interface exposure
  - telnet/VTY process + MIB daemon outside actual applications

But who will fund this / put resources at it?
Integration between 2G and 3G

- so far, 2G and 3G live separate to each other
- we want integration
  - cross-advertisement of neighbor cells
  - inter-RAT mobility
  - inter-RAT hand-over
- combined PS + CS attach
- SMS over packet switched
Operation / Maintenance

- generation of Alarms in BTS + PCU (done)
- report via CTRL TRAP as well as A-bis OML (done)
- centralized reporting / collections of Alarms (tbd)

→ we need to generate more alarms in abnormal situations

- generation of more stat, counters / KPIs
  - lots of work spent in 2016 on this already

→ we need aggregation/interpretation/analysis of that data
Further Wishlist

- move SMSC out of OsmoMSC
- GSUP to MAP gateway
- billing interfaces
- LTE MME/S-GW-/P-GW
  - biggest issue is lack of good FOSS asn.1 tools for C
  - we either have to ditch C, or invest lots of time in tools :/
End of File