Themyscira Wireless: Osmocom CNI meets USA PSTN

Themyscira Wireless is a rebel GSM network, operated in the southwestern corner of USA as part of underground resistance against 5G. Built and run by a trans woman, it is named after the fictional home of Wonder Woman, a source of inspiration to many women in LGBTQ+ and neopagan subcultures.

This technical presentation is a case study in interconnecting an Osmocom-based GSM network to USA PSTN in a production manner, showing the problems that needed to be solved and the technical solutions that have been implemented.

Presented by Mychaela Nadezhda Falconia,
Mother of FreeCalypso,
Dame of the Order of 2G,
Champion of Published Source Code

Problem I had to solve

A private GSM network has been built, using Osmocom CNI sw stack. This network needs to be interconnected with USA PSTN, meaning that:

* Every ThemWi GSM user gets a real 10-digit phone number in the North American Numbering Plan (NANP), chosen from an NPA-NXX prefix that geolocates to our home town.

* Every ThemWi user needs to be able to call other ThemWi users and the outside world, and needs to be able to receive calls from other local users and from the outside world.

* Same deal with SMS: we need to be able to SMS-text among each other, but also to and from the outside world.

* The entire user experience of a ThemWi GSM user needs to be exactly the same as the UX on a standard commercial cellular phone operator in USA!

All dialing formats that are considered standard for cellphones in USA must be supported; no "extension" numbers, no phone number NAT, no other weird hacks allowed!

What I mean by PSTN

When I say PSTN (worldwide), I mean the total set of all telephone destinations in the world, irrespective of technology, that can be reached by dialing an E.164 number and paying the appropriate toll costs, domestic or international.

When I say USA PSTN, I mean that portion of the worldwide PSTN which is physically or logically geolocated to the country of USA, operates with E.164 numbers in the USA portion of NANP (+1 country code), and follows USA telecom culture practices for interconnecting with other countries, originating and receiving international calls.

I do NOT use the term PSTN in the sense of analog lines only! If I use a SIP trunk provider to connect to USA PSTN, the Kantian thing-in-itself to which I am connecting is still USA PSTN; the SIP trunk is merely an access method.

Telecom landscape in USA

USA phone numbers (10-digit numbers in USA area codes) are available at insanely cheap prices when obtained by way of a SIP trunk provider. For voice services only, without SMS, TNs (telephone numbers) can be as cheap as 6 cents per number per month!

With numbers this cheap, there is absolutely no point in deploying any scheme other than giving a real NANP number to every GSM subscriber, including test numbers for lab use etc.

Large nationwide SIP trunk providers serve all of USA on a flat-rate basis. A customer located anywhere in USA, connecting over public Internet, can get TNs that geolocate to anywhere in the country.

Access to SMS (exchanging P2P SMS between your allocated TNs and the outside world, without being forcibly misclassified as A2P) is much more difficult: the only "white market" option I found (Bandwidth.com) requires minimum MONTHLY spend of $2500! All others only provide A2P SMS, which is useless when the users are human beings rather than business applications.

Community comes to the rescue

Denver Gingerich, who runs JMP.chat and is a member of Osmocom and FreeCalypso communities via mailing lists, came to my rescue.

Denver explained the basics of VoIP/SIP provider landscape to me back in the summer of 2022, pointed me to BulkVS for voice interconnection, and more recently set up an arrangement whereby his company resells a little bit of their P2P SMS service from Bandwidth to ThemWi.

JMP.chat is a gateway service that allows people who like XMPP to get USA or Canadian phone numbers and use XMPP to access SMS/MMS and voice. P2P-classified access to the worldwide SMS/MMS network is required for this feat to work, and Denver was able to find enough people who like XMPP to meet the steep volume requirements of the sole available mega-provider.

I personally have no interest in XMPP (I like GSM!), but I am very thankful to Denver for setting up a special arrangement whereby ThemWi can access BW's SMPP server directly, without extra translations and without getting into XMPP.

ThemWi setup uses hetero routing: BulkVS for voice, JMP for SMS.

Interconnection to USA PSTN in the voice plane

Origination provider: the company from whom you rent phone numbers and which handles (delivers to you via SIP) all incoming calls to these numbers, no matter where they come from.

Termination provider(s): one or more companies to whom you send your outbound call traffic. One can use multiple termination providers in the same deployment - in particular, it sometimes makes sense to use different termination providers for domestic vs international outgoing calls.

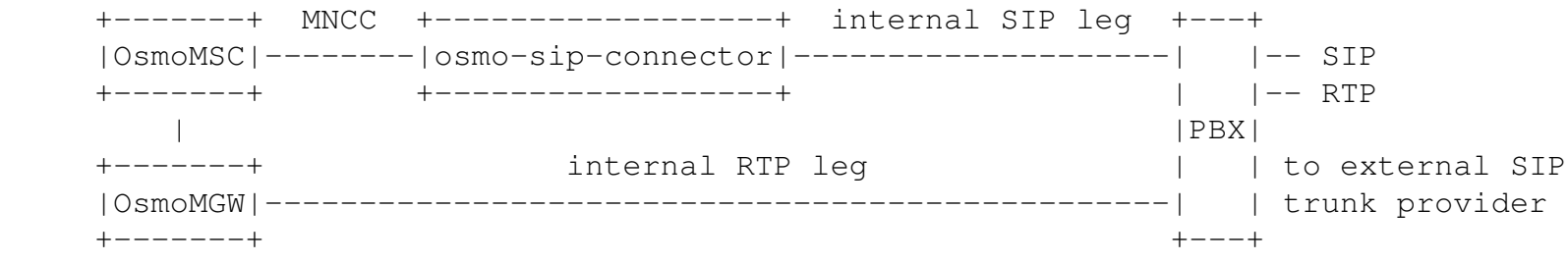
My current experience is with: BulkVS for origination and NANP termination, AnveoDirect for termination to international destinations outside of NANP.

Voice codecs supported by these providers: PCMU, PCMA and G729. No GSM codecs and no AMR! Transcoding is thus always required in order to interconnect an Osmocom GSM network to USA PSTN via SIP.

ThemWi design/policy decision: we only use PCMU and PCMA codecs (G.711), no G729. The objective is to deliver the same QoS as traditional commercial GSM networks, and those always operated with G.711 backhaul, not transcoding from FR/EFR/AMR to another non-GSM lossy speech codec.

Osmocom interconnection: the official way

The method familiar to most developers and officially recommended by Osmocom is to use osmo-sip-connector:



Required non-Osmocom external component: PBX, a piece of software of spaceship complexity - first major difficulty.

But even if I went off to write my own PBX, a more fundamental problem still remains: the internal SIP leg in the above diagram is an unnecessary and unwanted lossy translator from MNCC.

My desired alternative architecture

My principal idea is to eliminate the internal SIP leg:

The big box in the above block diagram includes 3 functions:

* Local call switching from one local GSM subscriber to another;

* Inbound gateway: calls from the outside world to phone numbers on the local network, routing to GSM subscribers;

* Outbound gateway: calls from local GSM subscribers to the outside world.

More detail: reasons for not wanting the internal SIP leg

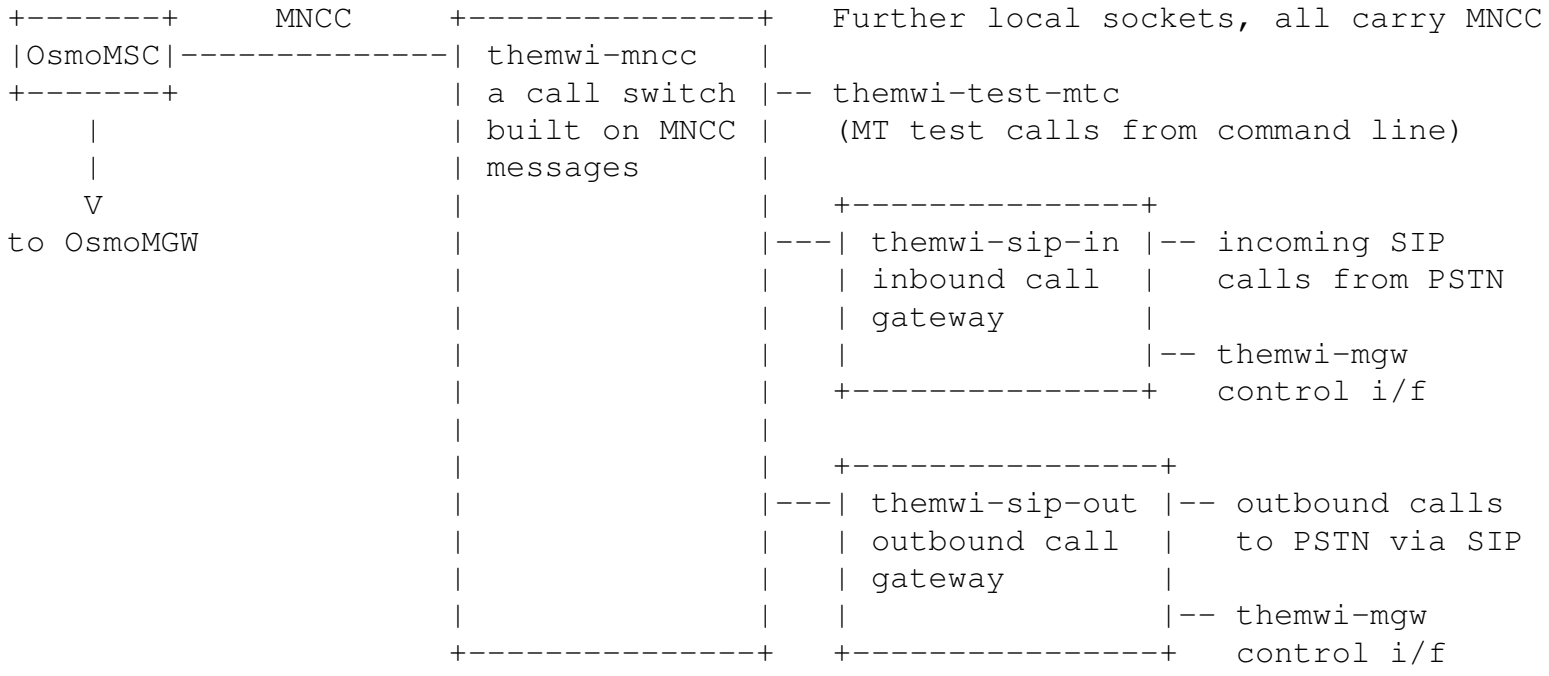
Local call switching: if one local GSM subscriber calls another, I wish to keep this call native in GSM CC/MNCC land, without translating to SIP and back. Preserve all native GSM signaling, native GSM causes for all possible error conditions, bearer capability pass-through for CSD, User-to-User information elements, etc.

For gatewayed calls (inbound and outbound), I would rather manage two animals (GSM CC/MNCC and external SIP) than three (those two plus the extra internal SIP leg).

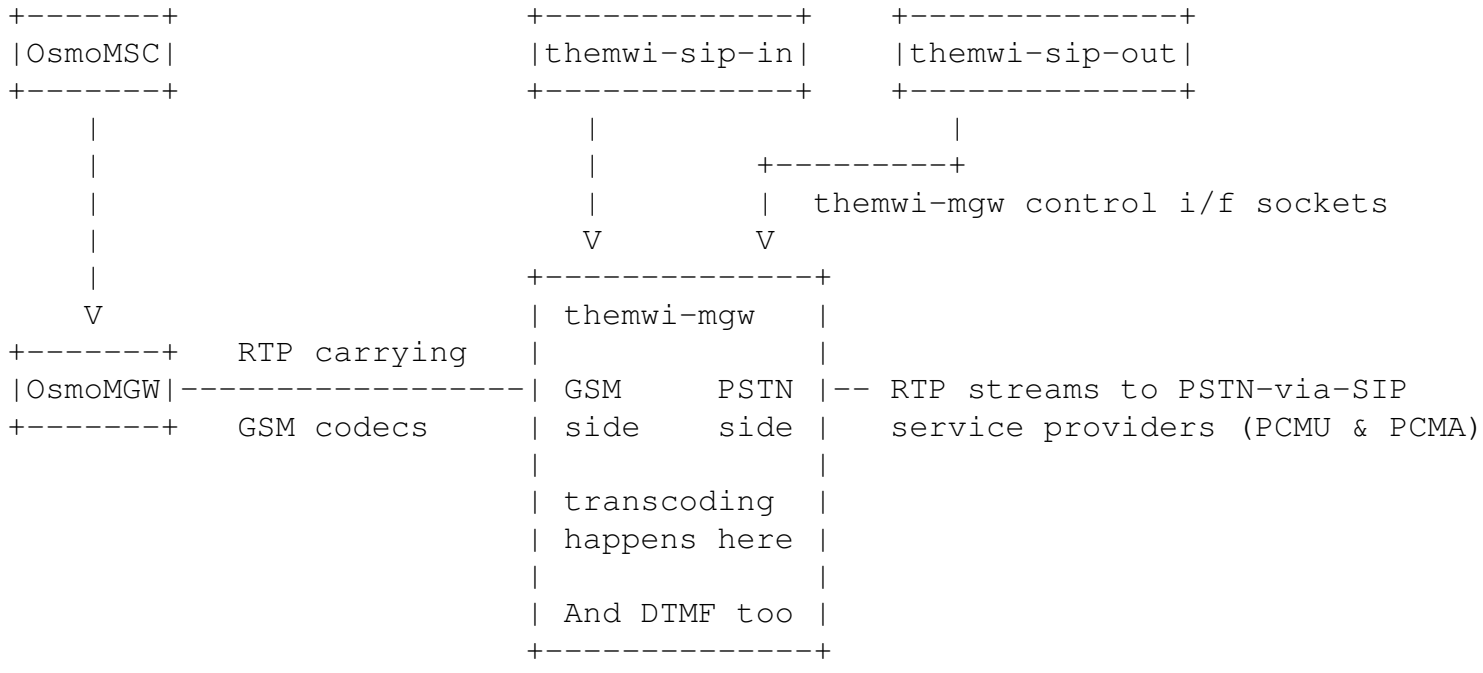
Signaling events and error conditions occuring inside my gateway: signal them natively in the GSM way, instead of coming up with SIP representations that won't go anywhere other than to osmo-sip-connector, to be translated into MNCC anyway.

DTMF: when a Start DTMF message comes from the MS, we have no way of knowing what the duration will be - the MS will send Stop DTMF when the user lifts her finger from the key. But the internal SIP leg of osmo-sip-connector requires a fixed duration to be set upfront for each DTMF.

Detailed design of ThemWi call handling: control plane



Detailed design of ThemWi call handling: RTP streams



How themwi-system-sw components are implemented

The SIP implementation in themwi-sip-in and themwi-sip-out was written from scratch by me, in plain C, parsing and all, NO outside libraries were used, zero dependencies.

The transcoding function of themwi-mgw supports FRv1 and EFR on the GSM side, no HRv1 or AMR currently. Themyscira GSM codec libraries (libgsmefr and libgsmfrp) are used, see previous presentation for more details. The PSTN side is PCMU or PCMA only, implemented via look-up tables.

Current version of themwi-system-sw does not use or link with Osmocom libraries, hence the long-running processes lack vty control. This limitation needs to be fixed for production use - but I am not sure yet which path would be easier: rewriting the sw to use Osmocom libraries or implementing some very primitive vty-like mechanism in my usual low-level manner.

I very recently figured out a way to link with Osmocom libs using only static hand-written Makefiles (with baked-in paths for my installation), matching my 1980s UNIX programming paradigm - I am using this approach for SMSC work.

Use of SDP in ThemWi system architecture

SDP is used in ThemWi only on the PSTN side, as part of SIP INVITE requests and responses describing PSTN calls. SDP is used there to describe PCMU and PCMA codecs and select between these two.

Because the codecs on GSM and PSTN sides of themwi-mgw are always completely different, the use of SDP on the SIP-to-PSTN side does not in any way translate to or impose on the GSM side - there is absolutely no need to use SDP to describe GSM codecs on the RTP leg between themwi-mgw and the Osmocom network.

All of this software was developed prior to the merging of SDP-fication patches in OsmoMSC, using classic MNCC interface without SDP. struct gsm_mncc_rtp members addr, payload_type and payload_msg_type are perfectly sufficient for communicating Osmo-network-selected codec and RTP info to themwi-mgw, no need for SDP - and binary structures are easier to work with.

There is no code in ThemWi to parse or construct SDP descriptions of GSM codecs - only PCMU and PCMA on the PSTN side.

How codec selection happens

Codec selection flow in Osmocom components prior to SDP-fication:

- OsmoMSC takes the speech version list from the MS (bearer cap IE), translates it to GSM 08.08 Channel Type and passes the ball to OsmoBSC. The MS-indicated order of preference is preserved in this step.

- OsmoBSC ignores the order of preference between MSC-indicated codecs and uses its own vty-configured order of preference instead - but it does heed the MSC's instruction (based on MS capabilities) as to which codecs are permitted or not permitted.

- The codec selection made by OsmoBSC is returned to OsmoMSC and becomes final.

What happens in ThemWi production setup:

- OsmoBSC is configured with:

  codec-list fr2 fr1

- If the MS supports EFR, then EFR gets selected, otherwise FRv1 - no other possibilities.

Why it works so well in practice

If the call is to or from the outside world, then there is only one GSM call leg, the call is always transcoded to PCMU or PCMA in themwi-mgw, and there is no possibility of codec mismatch - there is nothing to mismatch against.

Only calls from one local MS to another local MS (themwi-mncc switches them exactly like OsmoMSC's internal MNCC, using a copy of that code) can potentially be subject to codec mismatch problems - but:

- To get a codec mismatch in ThemWi, one MS would have to get EFR while the other got FRv1;

- In order for an MS to get FRv1 and not EFR, it must be non-EFR-capable;

- Outside of artificial tinkering with the transmitted speech version list on FreeCalypso MS, non-EFR-capable GSM phones are extremely rare - I have only one phone in my collection which I suspect to be non-EFR-capable, but I haven't got it to work at all yet, and I won't be able to get it to work until I progress much further along with SIMsniff - a topic for another day.

End result: no codec mismatches in practice, as all phones use EFR on ThemWi.

Need for future improvement in codec matching

The current situation works for now, but of course it is not correct: an FRv1-only phone MUST be able to call and receive calls not only to/from the outside world (transcoded), but also to/from other local subscribers with newer phones who would otherwise get EFR.

If a call is to be connected between an FRv1 phone and another MS that normally gets EFR, the correct course is to switch the EFR-preferring leg to FRv1. The other alternative of transcoding would be worse than FRv1 end to end.

But for other networks with AMR-HR, and for me in the future if and when I get capacity demands and have to start using AMR-HR: transcoding in corner cases may not be such a bad idea! Suppose Alice has a phone that supports AMR-HR, but Bob has an older phone that only goes up to EFR - switching Alice's call leg to FR/EFR to match Bob would increase radio resource usage, whereas transcoding in this rare corner case would allow Alice's call leg to remain a TCH/H.

Even with AMR-FR the situation isn't immediately obvious. AMR-FR handles poor C/I better than FR/EFR. If Alice's phone supports AMR but Bob's phone doesn't, which is better: transcoding or downgrading Alice's call leg to EFR?

Need for dynamic channel reassignment, switching codecs

In previous Osmocom discussions, different strategies were considered for MO leg to MT leg codec matching. One considered option was to perform a later channel reassignment if the initial channel assignment came out with a codec that later turns out to be incompatible end to end.

This reassignment ability needs to be implemented, as it is the only way to support codec switching in call waiting scenarios:

1: Alice and Bob are in a call, doing EFR end to end without transcoding.

2: Charlie calls Alice – but Charlie's phone supports FRv1 only.

3: If Alice decides to accept Charlie's call, putting Bob on hold, then her TCH mode will need to be switched dynamically in order to talk FRv1 with Charlie.

4: When Alice switches back to Bob (retrieves call from hold), her TCH will need to be switched back to EFR, as that's what Bob's call leg expects.

5: If Alice keeps pressing the "switch calls" button on her phone, going back and forth between Bob and Charlie, her TCH will need to switch codecs each time!

Entering subscriber MSISDNs into OsmoHLR

For networks that give each subscriber a real phone number in the host country's PSTN numbering plan, as opposed to a private extension, there is no clear guidance in Osmocom on exactly how MSISDNs should be entered into OsmoHLR:

* Should it be the full international E.164 number beginning with the country code, with only the '+' character stripped off?

* Should it be a national number, such as 10 digits NPANXXXXXX (without the leading 1) for USA, Canada and other NANP countries?

* In NANP territory we have it easy because '1' is both the country code and the trunk prefix for domestically dialed calls - but what about the rest of the world?

* As an example, should globally-routable (not private extension) MSISDNs in Mexico be entered into OsmoHLR as 52xxxxxxxxxx or in whatever dialing format is used domestically inside Mexico?

Solution adopted in ThemWi: our globally-routable NANP numbers are entered into OsmoHLR as 11-digit 1NPANXXXXXX.

Additional database of owned phone numbers in ThemWi

Any type of local phone switch aka end office (not just mobile) needs to answer this fundamental question when handling a locally dialed call: does the dialed number belong to another local subscriber, or does it belong somewhere else? The answer to this question drives the first switching decision: are we switching to another local subscriber loop in our town, or are we switching to a trunk line on the backhaul link connecting our town to some upstream?

In the old days this question was answered by the structure of the numbering plan in use: each local exchange would have its own range of numbers, and every dialed number would be immediately known as local or non-local based on the prefix. But this simple logic no longer works today in USA:

* Local number portability (LNP) means that people bring arbitrary NANP phone numbers around with them, even from different area codes, signifying different "home" location, without any numerical relation to the serving carrier.

* ThemWi being a tiny operator, the set of phone numbers we rent from BulkVS is very small, nowhere close to a full exchange range of olden days.

Thus we need our own database of locally owned numbers, answering the question "Is this number one of ours?"

Additional database of owned phone numbers in ThemWi (continued)

Why is OsmoHLR subscriber database not enough, why do we need an extra database of TNs (telephone numbers) that are currently rented from BulkVS (or other providers) to ThemWi?

The idea is to keep TN rental/allocation (from BulkVS to ThemWi) logically separate from subsequent assignment of these TNs to GSM subscribers as in IMSIs and programmed SIM cards.

When themwi-mncc processes a MO call setup from OsmoMSC, it begins with these two checks on the dialed number:

1) Is it NANP or international to some other country code? Calls to non-NANP numbers go to the outbound gateway (themwi-sip-out).

2) If it's NANP, look up in the owned number database - is the number one of ours? Based on the answer, switch back to OsmoMSC or to themwi-sip-out.

The inbound gateway (themwi-sip-in) requires the destination number to be one of ours per the database - otherwise the SIP INVITE is statelessly rejected upfront.

Additional complexity: dialing formats

There is only one way to dial international outbound calls to non-NANP countries: enter the '+' prefix on the handset, causing the MS to transmit the called party BCD number with TON=1, signifying a full international number.

However, for dialing calls to NANP destinations, USA telecom culture supports 3 different dialing formats for cellphones:

* Just the 10 digits of NPANXXXXXX, without '1' prefix;

* 11-digit number beginning with '1', but no '+' prefix, hence TON=0;

* Full international format beginning with '+', MS transmits TON=1.

(In the past there was also 7-digit dialing within the local NPA code,
but this method no longer needs to be supported, at least not for cellular.)

ThemWi must support all these methods, in order to provide the same UX as major commercial cellular operators! themwi-mncc implements these rules as part of called number preening, and as part of deciding whether the called number is NANP or not.

Sharing my work with others

Contrary to perception, I _DO NOT WANT_ to be "the only one in the world" - it's a horribly lonely place to be in! Thus if I really am the FIRST person in the world to interconnect an Osmocom GSM network to the regular phone network in my country-of-residence in the manner just presented, then I don't want to be the last.

If there already exist other people who run Osmocom GSM networks with full interconnection to their respective countries' regular PSTN, with each GSM subscriber on the Osmocom network being a full-fledged participant in the host country's regular public phone network, I would like to hear from them - how did you do it, how did you address all of the presumably-similar issues to the ones I had to solve.

If there are other people who haven't done so yet, but who would like to replicate what I did, especially if your motivation is similar to mine (major commercial operators taking away GSM/2G service) - you are my primary target audience here! I would love to see more small indie GSM networks around the world, copying what I did in ThemWi.

Sharing my work with others (continued)

The source code is public:

https://www.freecalypso.org/hg/themwi-system-sw/

If you are anywhere in USA or Canada or one of the smaller NANP countries, if you can get cheap NANP phone numbers from BulkVS or similar providers, and if you wish to interconnect your GSM network to PSTN in this manner, then you can use themwi-system-sw as-is to achieve that goal.

Adapting this sw for use anywhere else in the world will require some changes, to be preceded by discussion on what is the right way to handle the telecom landscape outside of NANP:

* The code is peppered with assumptions that all local numbers are NANP, and that any non-NANP number must be an outbound international call.

* The dialing format handling code in themwi-mncc is specific to NANP and to USA telecom culture, in terms of which dialing formats need to be supported.

* The database of locally owned numbers is a critical piece, required for the sw to work, and its format is specific to NANP.

Adding SMS capability

OsmoMSC's built-in SMSC is not really suitable for the needs of ThemWi:

* Just like with calls, the SMSC needs to be smart enough to understand that each NANP subscriber can be addressed by 3 different number formats.

* Upon receiving a MO SM or a SM entered through some other channel, the SMSC needs to look up in the database of locally owned numbers to determine if a given NANP number belongs locally or in the outside world, and route accordingly.

* OsmoMSC implements SMPP in the role of a server. However, in order to exchange SMs with the outside world, ThemWi SMSC will need to connect to our upstream provider's SMSC in the role of an SMPP client. (SMPP is an asymmetric client-server protocol, despite the name reading "peer to peer".)

Solution: implement a proper separate SMSC, connecting to OsmoHLR via GSUP. The necessary OsmoHLR patches are currently being worked out in Gerrit, to be followed by some OsmoMSC patches - see OS#6135.

Timeline

I am working under severe time pressure:

* T-Mobile USA said they will shut down GSM/2G service on 2024-04-02 - and it's a service which I critically rely upon for personal communication.

* The service quality of T-Mobile GSM is already so degraded that I started using ThemWi instead of TMO when I am at home, and actually relying on ThemWi for important personal and business communication - thus it is now a production network for me, not just a toy any more.

* There is still a ton of work to be done, and I need to complete this work before TMO kill their network.

This under-the-gun mode of work, plus me having to _already_ rely rather critically on my current ThemWi network despite it being very incomplete, is the reason for my current inability to test newer versions of Osmocom CNI suite than 2023-02 stable release.

Newer Osmocom CNI developments

To Osmocom developers whom I offended by my negative attitude toward recent SDP-fication developments and my seeming unwillingness to even test newer code: I apologize for my abrasiveness, but please understand that I am working under a gun to my head, with TMO about to kill a service which I critically rely on.

In order to test newer OsmoCNI code, I will need to set up a separate test network, separate from the production ThemWi network which I rely on for my personal communication. Because of digraph dependencies on other parts of the work, it may be some months before I can get to this test network setup. It may or may not happen before TMO shutdown date.

But eventually, in a few months to under-a-year at most, the time will come when I will be free to set up a separate network just for OsmoCNI sw testing, and I will then thoroughly test whatever the current Osmocom master will be on that day. I will then be able to join Osmocom development properly, working on the current master tip.