Apéndice A

Kannel box protocol description

Update stamp: 20041202

A.1. Current *C* structures

.

```
enum msg_type {
        heartbeat,
        admin,
        sms,
        ack,
        wdp_datagram,
        msg_type_count
};

typedef struct {
        enum msg_type type;

        struct heartbeat {
                long load;
        } heartbeat;

        struct admin {
                long command;
                Octstr *boxc_id;
        } admin;
```

```
struct sms {
        Octstr *sender;
        Octstr *receiver;
        Octstr *udhdata;
        Octstr *msgdata;
        long time;
        Octstr *smsc_id;
        Octstr *service;
        Octstr *account;
        uuid_t id;
        long sms_type;
        long mclass;
        long mwi;
        long coding;
        long compress;
        long validity;
        long deferred;
        long dlr_mask;
        Octstr *dlr_url;
        long pid;
        long alt_dcs;
        long rpi;
        Octstr *charset;
        Octstr *boxc_id;
        Octstr *binfo;
        long msg_left;
        void *split_parts;
        long priority;
} sms;

struct ack {
        long nack;
        long time;
        uuid_t id;
} ack;

struct wdp_datagram {
        Octstr *source_address;
        long source_port;
        Octstr *destination_address;
        long destination_port;
        Octstr *user_data;
} wdp_datagram;
```

```
} Msg;

struct split_parts {
    Msg *orig;
    Counter *parts_left;
    long status;
};

enum {
    mo = 0,
    mt_reply = 1,
    mt_push = 2,
    report_mo = 3,
    report_mt = 4
};

enum {
    cmd_shutdown = 0,
    cmd_suspend = 1,
    cmd_resume = 2,
    cmd_identify = 3,
    cmd_restart = 4
};

typedef enum {
    ack_success = 0,
    ack_failed = 1,
    ack_failed_tmp = 2,
    ack_buffered = 3
} ack_status_t;
```

A.2. Protocol Definition

.

```
Kannel Protocol Description
Packet description, draft v0.1
Description by:
        Oscar Medina Duarte
```

{moscar}[at]gmail.com

Protocol Description
This protocol has been designed with the purpose of maintaining a connection between kannel bearerbox and another box, typically an smsbox and/or a wapbox.

Kannel protocol session:
A session is initiated when a kannel box client requests to connect to a bearer box and sends an administration command requesting to be identified.And finishes when the bearer box requests the client box to shutdown or reset.

During the session, both parts can request heartbeats to each other to ensure that the link is up and to measure the round trip of packages, which is not yet handled by the current implementation of the protocol.

Kannel session to a smsbox
When a smsbox has established a session, it will send and receive sms packets which contain a short message, a destination (MT) address, a source (MO) address and an identification number, among other fields.

Kannel session to a wapbox
When a wap box has established a session, it will send and receive wdp datagrams trhough the connection containing, the source and destination's ports and addresses, and user data.

Protocol's Operations

Heart beating
Heart beating is not completely implemented in the current version of kannel. Heart beats are only sent by the smsbox and ignored at it. Future work for the implementation would be to have a complete specification of the protocol, and cover this issue correctly.

Sms boxing
The smsbox, can send and receive packets with the msg_type of the packet set to sms and ack.To keep track of sms messages, they are always sent with a UUID, wich is used later to acknowledge the message. All sms type messages should be

```
acknowledged.


Wap boxing
Not yet explored.
```

A.3. BNF of protocol packets

.

```
Packet BNF:

<message> := <lenght><type><packet>
<packet> := <heartbeat_packet> | <admin_packet> | <sms_packet> |
                <ack_packet> | <wdp_datagram_packet>

<heartbeat_packet> := <load>
<admin_packet> := <adm_comand><boxc_id>
<sms_packet> := <sender><receiver><udhdata><msgdata><time>\
                <smsc_id><service><account><uuid><sms_type>\
                <mclass><mwi><coding><compress><validity>\
                <deferred><dlr_mask><dlr_url><pid><alt_dcs>\
                <rpi><charset><boxc_id><binfo><msg_left><priority>
<ack_packet> := <nack><time><uuid>
<wdp_datagram_packet> := <source_address><source_port>\
                         <destination_address><destination_port>\
                         <user_data>

<lenght> := <INT>
<type> := <INT>
<load> := <INT>
<adm_comand> := <INT>
<boxc_id> := <STRING>
<sender> := <STRING>
<receiver> := <STRING>
<udhdata> := <STRING>
<msgdata> := <STRING>
<time> := <INT>
<smsc_id> := <STRING>
<service> := <STRING>
```

```
<account> := <STRING>
<sms_type> :=  <INT>
<mclass> := <INT>
<mwi> := <INT>
<coding> := <INT>
<compress> := <INT>
<validity> := <INT>
<deferred> := <INT>
<dlr_mask> := <INT>
<dlr_url> := <STRING>
<pid> := <INT>
<alt_dcs> := <INT>
<rpi> := <INT>
<charset> := <STRING>
<binfo> := <STRING>
<msg_left> := <INT>
<priority> := <INT>
<nack> := <INT>
<source_address> := <STRING>
<source_port> := <INT>
<destination_address> := <STRING>
<destination_port> := <INT>
<user_data> := <STRING>

<uuid> := <lenght><UUID>
<STRING> := <lenght><OctetString>

<UUID> := BYTE{16}
<INT> := BYTE{4}
<OctetString> := BYTE*
```

A.3.1. Protocol Glossary

.

```
message .- Is the unit of data transmitted over a link in a
        box to box session.
packet .- Is the biggest meaningful part transmitted by a box.
heartbeat_packet .- Is the kind of packet used for the heart
        beating operation.
```

```
admin_packet .- Is the kind of packet used to send commands
        among boxes.
sms_packet .- Is the kind of packet used to transfer SMS
        messages.
ack_packet .- Is the packet used to send acknowledges.
wdp_datagram_packet .- Is the packet to carry wdp datagrams.
lenght .- It is the number of bytes left plus one.
type .- Is the type of packet contained in a message.
        Package types:
                0 = heartbeat
                1 = admin
                2 = sms
                3 = ack
                4 = wdp_datagram
                5 = msg_type_count
load .-
adm_comand .- Is the command contained by an admin command.
        Admin commands:
                cmd_shutdown = 0
                cmd_suspend = 1
                cmd_resume = 2
                cmd_identify = 3
                cmd_restart = 4
boxc_id .- It is the client box ID
sender .- Is the address of the originating entity of an sms
        message.
receiver .- Is the address of the destination for an sms
        message.
udhdata .- User Data Header.
msgdata .- Is the data contained in a sms message, generally
        text.
time .- <Current time in milliseconds since the epoch>
smsc_id .- Is the internal name of the smsc link that received
        the sms message or the smsc link that should send the
        message.
service .- Must be a username defined in a sendsms-user group
        at the bearerbox.
account .- Account name or number to carry forward for billing
        purposes.
sms_type .- It indicates the type of sms message.
        SMS types:
            mo = 0,
            mt_reply = 1,
            mt_push = 2,
            report_mo = 3,
```

```
            report_mt = 4
mclass .- Sets the Message Class in DCS Field.
mwi .- Sets Message Waiting Indicator bits in DCS field.
coding .- Sets the coding scheme bits in DCS field.
compres .- Sets the Compression bit in DCS Field.
validity .- If given, Kannel will inform SMS Center that it
        should only try to send the message for this many
        minutes.
deferred .- If given, the SMS center will postpone the message
        to be delivered at now plus this many minutes.
dlr_mask .-
dlr_url .-
pid .- Sets the PID value. (See ETSI Documentation).
alt_dcs .- If unset, Kannel uses the alt-dcs defined on smsc
        configuration, or 0X per default. If equals to 1,
        uses FX. If equals to 2, force 0X.
rpi .- Sets the Return Path Indicator (RPI) value. (See ETSI
        Documentation).
charset .- It indicates the Charset used to encode the SMS
        message's data.
binfo .- Billing identifier/information proxy field used to
        pass arbitrary billing transaction IDs or information
        to the specific SMSC modules.
msg_left .-
priority .- Sets the Priority value (range 0-3 is allowed).
nack .- It indicates the ack status in a ack packet.
        Ack status:
                ack_success = 0,
                ack_failed = 1,
                ack_failed_tmp = 2,
                ack_buffered = 3
source_address .- It is the originating IP address of the
        datagram.
source_port .- Is is the originating port of the datagram.
destination_address .- It is the destination address for the
        datagram.
destination_port .- It is the destination port for the
        datagram.
user_data .- It is the data inside the datagram.
uuid .- UUID plus a 4 byte length heading the field.
STRING .- It is and octet string leaded by a 4 byte length
        field.
UUID .- Universal Unique IDentifier defined by the IETF
         (http://www1.ics.uci.edu/~ejw/authoring/uuid-guid
                                  /draft-leach-uuids-guids-01.txt)
```

```
INT .- It is a 4 byte signed integer.
OctetString .- It is an array of 8 bit bytes.
```