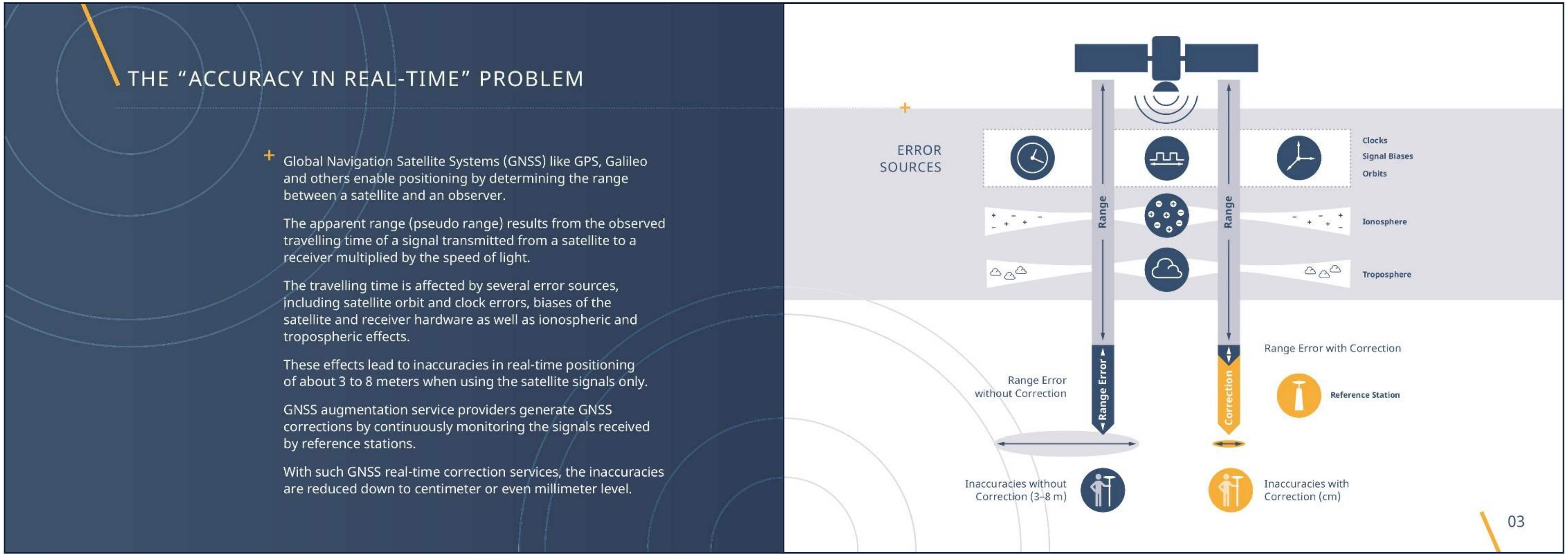
Introduction and Goals

Client Software estimates GNSS (Global Navigation Satellite Systems) corrections based on network observations. These corrections are transmitted from a service provider to the users by different communication technologies and formats.

The following slides give an introduction to GNSS corrections.



OSR AND SSR – TWO QUALITIES OF CORRECTIONS

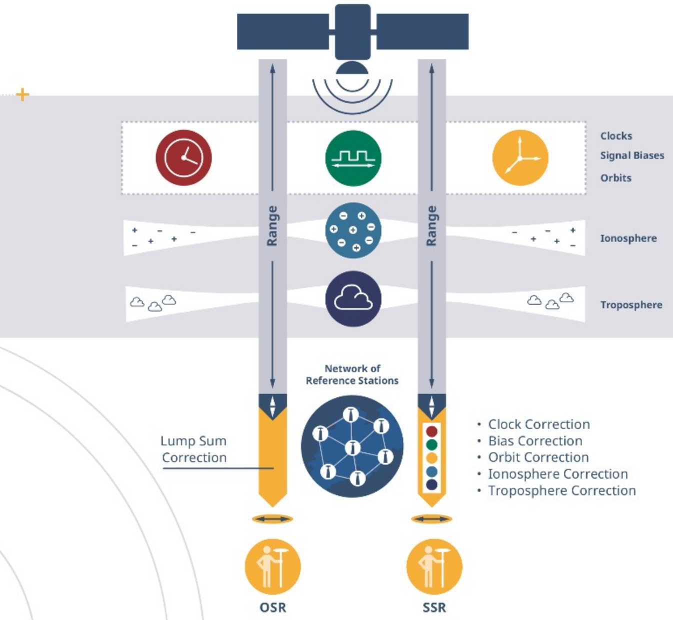
+ **Observation Space Representation (OSR)**
In conventional Real-Time-Kinematic (RTK) services the lump sum of all these errors is observed by a network of reference stations and provided to the rover as range corrections for each supported combination of satellite, frequency and signal. OSR requires the processing of the same signals on each reference station (homogeneous network) and the support of these signals by the user.

+ **State Space Representation (SSR)**
A network of reference stations is used to decorrelate and estimate the different GNSS error components (states):

- Satellite Clocks
- Satellite Orbits
- Satellite Signal Biases
- Ionospheric Delay/Advance
- Tropospheric Delay

With **SSR** users can compute GNSS corrections valid for their position.

Additionally, statistical accuracy information can be transmitted to support the rover algorithm.



SCALABLE SERVICES – VARIOUS APPLICATIONS

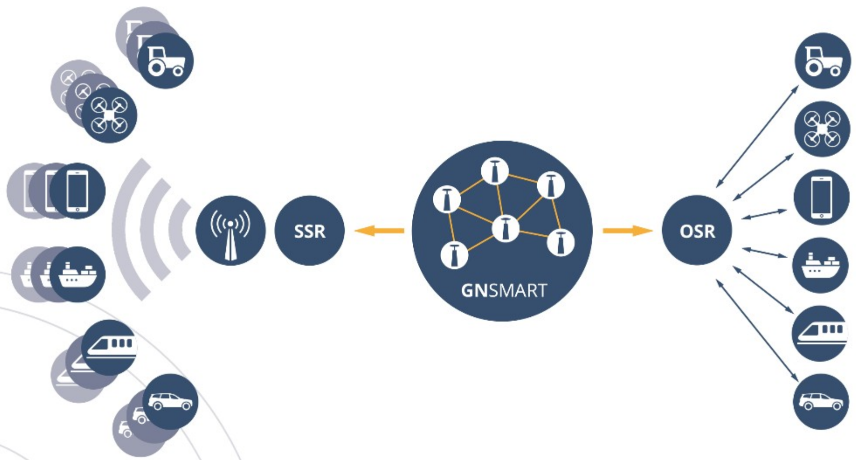
+ **OSR** corrections are exchanged via duplex communication media since the user needs to transmit its approximate position to the service provider to generate the corrections.

SSR based network-RTK enables the use of simplex communication media (satellite and ground based broadcast) using only a single data stream for all users. Thereby, an SSR service can in principle supply an unlimited number of users at the same time, making it ideally suited for future mass market applications as e.g. autonomous cars or drones.

The structure of **SSR** allows scalable RTK services with respect to accuracy and with respect to various specific applications in numerous GNSS market segments, including:

- Agriculture
- Aviation
- Location Based Services
- Mapping and Surveying
- Maritime
- Rail
- Road

GNSMART2
enables the consistent and seamless estimation of GNSS errors (rigorous solution) and provides both **OSR** and **SSR** corrections.



The SAPA Format is an open format realization from Sapcorda to transmit SSR corrections. The SAPA format content is based on the State State Representation (SSR) model with basic SSR parameters orbit, clock, biases, and atmospheric corrections for the ionosphere and troposphere, respectively.

Client's goal is to support as many as possible correction formats. For this purpose we need an **encoder/decoder library** for the SAPA format.

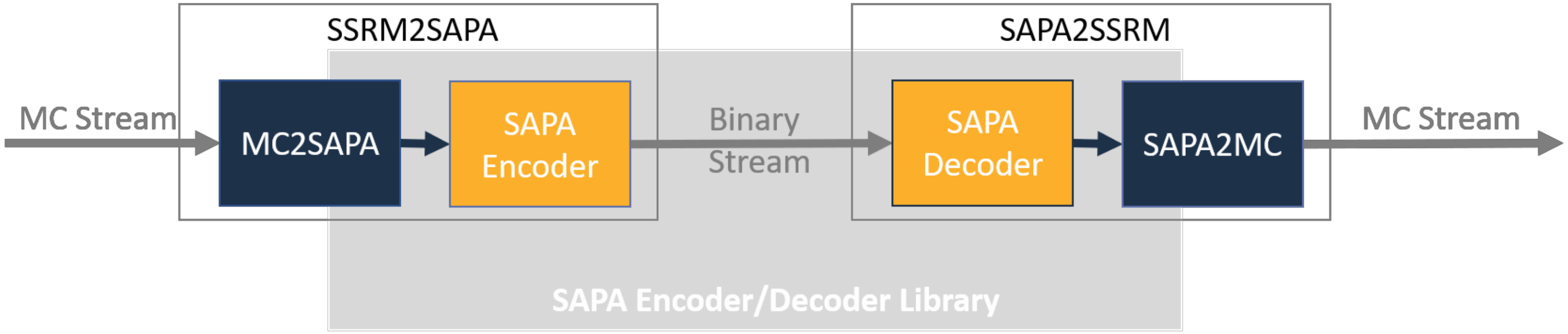
So far, the SAPA format consists of three message types OCB, HPAC and Area definition (see SAPA ICD 1.5). The OCB part has been already implemented and can be used as a kind of blue print for the remaining messages types.

Requirements Overview

The SAPA format will be supported by a program SSRM2SAPA (and counter part SAPA2SSRM). This program consists of two main parts MC2SAPA (SAPA2MC) and SAPA Encoder (SAPA Decoder), respectively.

In SSRM2SAPA the first part receives data from the internally used SSR data stream (referred as "MC stream") and maps these data to SAPA structures (declared in `gpp_sapa.h`). The mapping is done by Client. The second part encodes data from the SAPA structures to binary according to the SAPA ICD.

At the client's side the SAPA Decoder decodes the binary stream to SAPA structures which will be mapped afterwards to the internally used MC stream by the SAPA2MC functions.



The requested Client SAPA library should contain:

- encoder and decoder functions for SAPA messages defined in the latest SAPA ICD
- functions to add data to SAPA structures

Client provides structure and prototype declarations as well as an example to the SAPA OCB implementation that should be used implement/extent the SAPA Encoder/Decoder library.

Architecture Constrains

- programming language: C
- Watcom 11 Compiler (version provided by Client)
- Windows is preferred as Client provides a dll