# Collaborative data API

## C++ interface

The following structure must be used to provide collaboration data to RTFPPL. The structure contains data from one mobile device

struct CollaborationData

{

int64\_t timestamp; /\*\*< unix time [ms] \*/

double lattitude; /\*\*< lattitude [deg] [-90..+90]\*/

double longitude; /\*\*< longitude [deg] [-180..+180] \*/

double cov\_ne[2][2]; /\*\*< local NE-coordinates covariance matrix [m^2]\*/

bool is\_position\_valid; /\*\*< position validity flag \*/

double floor\_number; /\*\*< discrete physical floor number\*/

double floor\_std; /\*\*< floor standard deviation [floor], the field is unavailable if it is negative \*/

bool is\_floor\_valid; /\*\*< position validity flag \*/

double distance; /\*\*< distance to collaborative unit [m] \*/

double distance\_uncertainty; /\*\*< distance uncertainty [m], the field is unavailable if it is negative \*/

bool is\_distance\_valid; /\*\*< position validity flag \*/

uint64\_t BSSID; /\*\*< device ID \*/

};

To provide multi-device data, the structures from different devices must be combined in STL vector

std::vector <Fppe:: CollaborationData>

Use the following function to input collaboration data into RTFPPL:

/\*\*

\* pushes collaboration data into the processing pipeline

\* \param[in] collaboration position

\*/

void processCollaborationData(std::vector <Fppe::CollaborationData> &collaboration\_data);

To enable/disable this data for collaboration update in particle filter, use the following function:

/\*\* enable collaboration pf updater

\* \param[in] enable status

\*/

void setUpdateCollaboration(const bool enable);

To enable/disable this data for positioning start, use the following function:

/\*\* enable collaboration data usage for positioning start

\* \param[in] enable status

\*/

void enableCollaborationForStart(const bool enable);

***NOTE: this function is not implemented yet***

To enable/disable this data for positioning reacquisition, use the following function:

/\*\* enable collaboration data usage for pf injection (reacquisition)

\* \param[in] enable status

\*/

void enableCollaborationForInjection(const bool enable);

***NOTE: this function is not implemented yet***

## C interface

The following structure must be used to provide collaboration data to RTFPPL. The structure contains data from one mobile device.

typedef struct CICollaborationDataTag

{

int64\_t timestamp; /\*\*< unix time [ms] \*/

double lattitude; /\*\*< lattitude [deg] [-90..+90]\*/

double longitude; /\*\*< longitude [deg] [-180..+180] \*/

double cov\_ne[2][2]; /\*\*< local NE-coordinates covariance matrix [m^2]\*/

bool is\_position\_valid; /\*\*< position validity flag \*/

double floor\_number; /\*\*< discrete physical floor number\*/

double floor\_std; /\*\*< floor standard deviation [floor], the field is unavailable if it is negative \*/

bool is\_floor\_valid; /\*\*< position validity flag \*/

double distance; /\*\*< distance to collaborative unit [m] \*/

double distance\_uncertainty; /\*\*< distance uncertainty [m], the field is unavailable if it is negative \*/

bool is\_distance\_valid; /\*\*< position validity flag \*/

uint64\_t BSSID; /\*\*< device ID \*/

} CICollaborationData;

To provide multi-device data, the structures from different devices must be combined in array:

CICollaborationData collaboration\_position\_array[];

Use the following function to input collaboration data into RTFPPL:

DLL\_EXPORT void FPEngine\_processInputCollaboration(CIFPEngine\* fpEngine, CICollaborationData\* collaboration\_position, size\_t size);

size\_t size is the collaboration position array size.

To enable/disable this data for collaboration update in particle filter, use the following function:

DLL\_EXPORT void FPEngine\_setUpdateCollaboration(CIFPEngine\* fpEngine, bool enable);

## Special notes about collaboration timestamps.

Timestamps of all collaboration data provided in RTFPPL must be converted into the same time scale as other data provided (wifi, BLE and TPN data).

Time tags of peer positions can differ from transmission and receiving times. The following picture illustrates this.



Position in i-th peer device is calculated in some time prior to transmission. Initially the position accompanied with time tag related with tame scale of i-th peer device. There is a delay between time of position calculation and transmission time. Finally, all position time tags must be converted in time scale of receiving peer device. The following equation can be used to compensate transmission delays: