**SNAAPP use cases (draft)**

* Typical use case scenarios:
	+ **Gaming**
	An end user is playing a time-sensitive game communicating with a game provider's server. The end user wants to have a high-quality and low-latency communication for better service experience, so the game server (AF or API invoker) try to invoke the QoS API to change the end user's QoS according to the request from the application on the end user's UE. In order to let the game server invoke the QoS API, the end user is asked to confirm that the server may do so. The end user chooses "Yes" when the UE shows a message saying whether the game server may change the user's QoS. After this authorization, the game server sends the service API invocation request to the API exposing function. This is an example of real-time or near real-time request of authorization, but the game server may also use the authorization information given by the resource owner in the past authorization procedure.
	+ **Supporting UE applications as API invokers**
	A tracking application is based on enabling the user on UE Y to track the location of a user on UE X. An API Provider AP provides location APIs via CAPIF, with current functionality allowing the corresponding AS/AF to act as API Invoker and provide the tracking functionality.

Consent is obtained from the end user on UE X via an API allowing “user on UE Y" to invoke the location API for UE X exposed via CAPIF by AP (UE-originated API invocation as shown in 3GPP TR 23.700-95 Figure 4.1.1-1). When the tracking goes on for a long time, the corresponding AS/AF maintains the tracking of UE X while UE Y may be asleep (AF-originated API invocation as shown in 3GPPTR 23.700-95 Figure 4.2.1-1).

**(The scenario below is still under discussion)**

Optionally, the UE X end user may want to be able to provide consent contextually and does not want to provide the authorization equally to the server or to other users. The UE X end user may also want to be able to provide consent per information user (end user of UE Y) rather than per application.

For example, the UE X end user chooses "every day during work hours only” to message requesting consent for UE Y to track its location. This consent however is not given to an application server, so if UE Y goes to sleep, an application server cannot continue tracking UE X on its behalf. In addition, such consent can be provided independent of application, i.e. UE Y can track UE X during this time via another app. (This description does not exclude the ability to provide authorization per-application and to application servers, but those aspects are assumed already provided for). This consent configuration applies for a specific duration and can be renewed, i.e. does not need to be provided at each API invocation.

* **Location service for a group of UEs (under discussion)**

In the same example, UE X end user can choose to provide consent for an entire G1 group of UEs (e.g. involved in a cooperative game), only if the invoker group member is located within a certain geographic area. When a group member makes the request from outside the area, the UE X end user wants to be triggered to provide consent on a case-by-case basis. This consent configuration applies for a specific duration and can be renewed.

* + **Supporting 3rd party API provider**

SA6 discusses what the functional model should be when the features described above are provided by the 3rd party API provider. (related to the bullet 2-ii of the LS)

* SNAAPP allows the end users of a particular UE to give authorization to the API invoker before API invocation relating to that UE. The UE is considered to have a user interface that can display the authorization dialogue. The process should differentiate between authorizing an application server and another UE (see SA1 requirements below)
* “Resource owner” is the end user, and the UE shows certain user interfaces to let the end user authorize or deny the API invocation.
* Unlike the user consent in UDM/UDR, SNAAPP aims to realize at least near real-time (Solution #4) authorization; (real-time authorization is also a possibility (Solution #3)).
* As stated in the bullet 4 of the LS (S6-221771), SA6 wants SA3 to provide feedback on whether the resource owner client (application on the UE) can provide the detailed configuration of authorization for each API. This configuration may be renewed periodically and can include settings such as “always allow/deny”, “conditionally allow/deny”, “ask for consent at each API invocation”, etc.
* Two different API invocation scenarios (corresponding to SA1 requirements):
	+ UE-originated API invocation (Key Issue #1, TR 23.700-95)
		- [SA1 requirement: *The 5G system shall be able to provide a UE with secure access to APIs (e.g. triggered by an application that is not visible to the 5G system), by authenticating and authorizing the UE.*]
	+ AF-originated API invocation (Key Issue #2, TR 23.700-95)
		- [SA1 requirement: *The 5G system shall be able to provide a third-party with secure access to APIs (e.g. triggered by an application that is visible to the 5G system), by authenticating and authorizing both the third-party and the UE using the third-party's service.*]
* End user wants to permit a certain application to change its QoS, obtain user location, etc., but does not permit others to do so.