



# **Title : Technical and functional analysis of TOMP-API**

**Author : Christophe Duquesne, [christophe.duquesne@aurigetech.com](mailto:christophe.duquesne@aurigetech.com)**

**Meeting : SWG4.3 - Multimodal data**

**Date : March 2023**

# Outline

1. Objectives and methods
2. Presentation of TOMP-API
3. System architecture
4. Links with GBFS
5. Links with GTFS
6. Positioning towards OJP
7. Positioning towards NeTEx
8. Positioning towards SIRI
9. Replies to specific points
10. Other activities to consider

# Analysis objectives

- Questions that have been addressed:
  - Can TOMP-API be used for the **new mobility modes as well as for the Public transport and the planning of multimodal trips?**
  - Can TOMP-API be used for : **planning, booking, purchasing, payment ?**
  - What is the link between the « **planning** » as described in TOMP-API, and in **TC278/WG3/SG9 and in TC278/WG3/SG8 (for service provision)?**
  - TOMP-API, supposed to be standards-agnostic, can it ensure interoperability between:
    - data (traveller info) provided in a CEN NeTEx format and other data related to planning, booking and/or purchasing a transport or mobility service?
    - data related to the CEN OJP standard and other data related to the reservation and/or purchase of a transport or mobility service?
  - to what extent can TOMP-API manage dynamic data in the sense of SIRI services?

# Analysis objectives

- Essentially technical analysis and as factual as possible
- Functional analysis
- No organizational or legal element

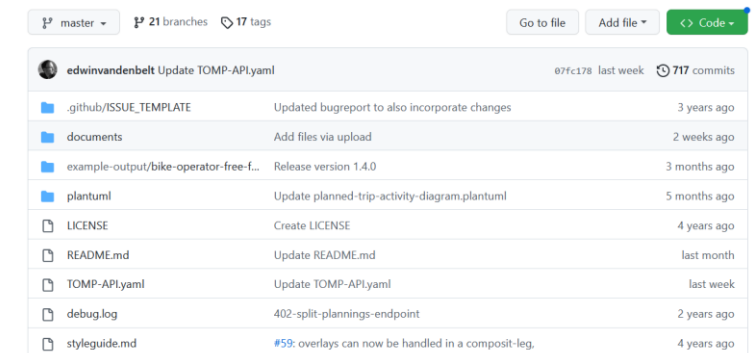
*I don't claim to be a TOMP expert, I just analyzed as many documents as possible, exchanged with other experts, and above all analyzed the API itself*

# TOMP-API Presentation

- **TOMP-API** allows to **inquire prices, confirm availability of offers and to book**
- **TO-MP API** = **T**ransport **O**perator to **MaaS** **P**rovider **A**PI

- **Github repository** (<https://github.com/TOMP-WG/TOMP-API> )

- **TOMP-API.yaml** : Swagger defining the API
- **UML (plantuml)** : diagrams of sequences
- **Many documents** (not always recent), including summaries of the working group
- **Some examples (4)**
- **Note** : versions are developed in dedicated branches and identified by tags



edwinvandenbelt Update TOMP-API.yaml 07fc178 last week 717 commits		
.github/ISSUE_TEMPLATE	Updated bugreport to also incorporate changes	3 years ago
documents	Add files via upload	2 weeks ago
example-output/bike-operator-free-f...	Release version 1.4.0	3 months ago
plantuml	Update planned-trip-activity-diagram.plantuml	5 months ago
LICENSE	Create LICENSE	4 years ago
README.md	Update README.md	last month
TOMP-API.yaml	Update TOMP-API.yaml	last week
debug.log	402-split-plannings-endpoint	2 years ago
styleguide.md	#59: overlays can now be handled in a composi...	4 years ago

# TOMP-API Presentation

## ■ API

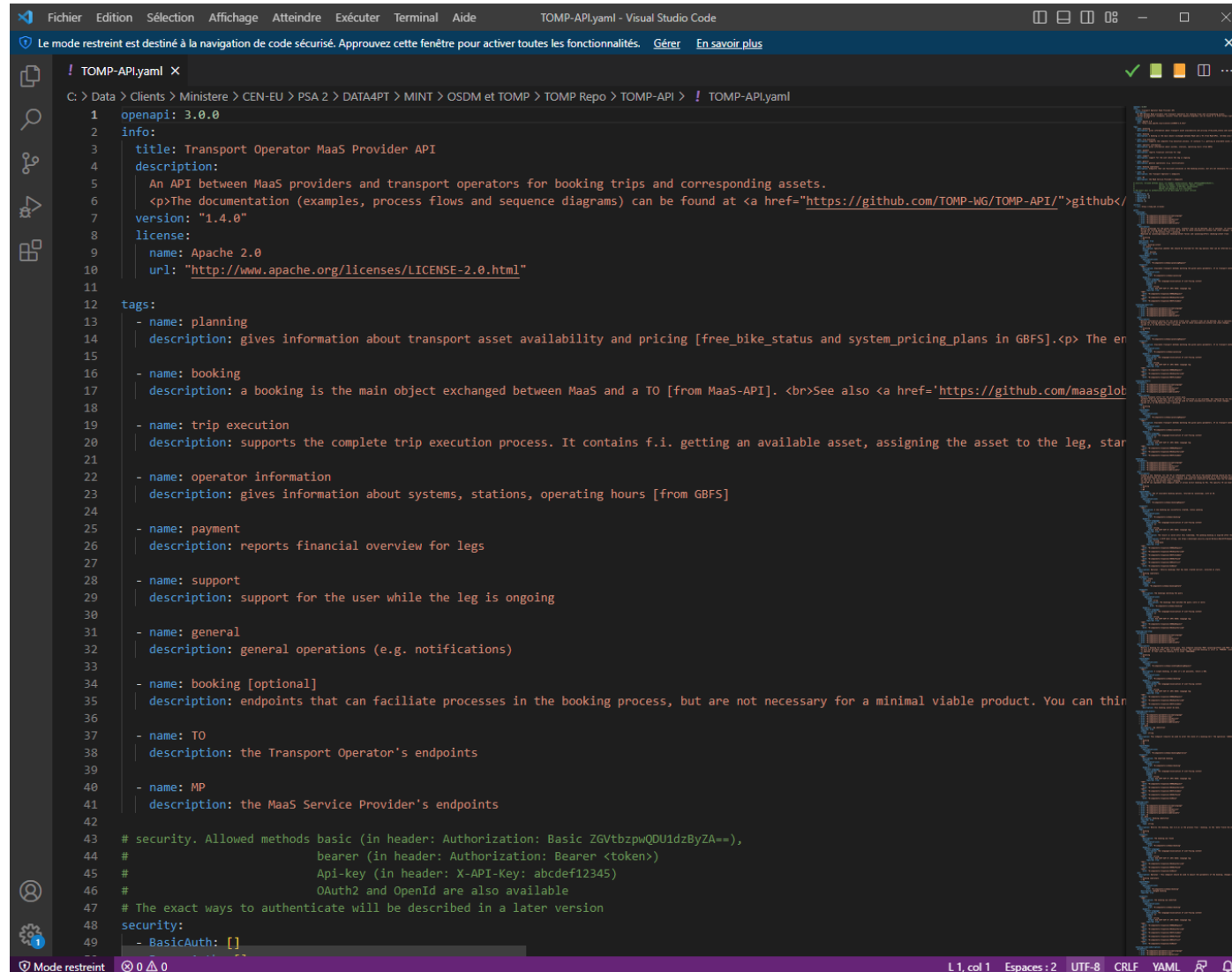
### ■ Open-API 3.0 (<https://swagger.io/specification> )

- Programming Interface Description Standard (HTTP REST)
- Swagger (or « specification OpenAPI ») in YAML format (and not JSON) - authorises comments (not much used in the end)
- Actual Version 1.4 of TOMP API (used for this analysis)
- This is the heart of the technical implementation of TOMP-API
- Online visualization tool

[https://app.swaggerhub.com/apis/TOMP-API-WG/transport-operator\\_maas\\_provider\\_api/1.4.0](https://app.swaggerhub.com/apis/TOMP-API-WG/transport-operator_maas_provider_api/1.4.0)

# TOMP-API Presentation

## ■ Code YAML



```
1 openapi: 3.0.0
2 info:
3   title: Transport Operator MaaS Provider API
4   description:
5     An API between MaaS providers and transport operators for booking trips and corresponding assets.
6     <p>The documentation (examples, process flows and sequence diagrams) can be found at <a href="https://github.com/TOMP-WG/TOMP-API">github</a>
7   version: "1.4.0"
8   license:
9     name: Apache 2.0
10    url: "http://www.apache.org/licenses/LICENSE-2.0.html"
11
12 tags:
13   - name: planning
14     description: gives information about transport asset availability and pricing [free_bike_status and system_pricing_plans in GBFS].<p> The en
15
16   - name: booking
17     description: a booking is the main object exchanged between MaaS and a TO [from MaaS-API]. <br>See also <a href="https://github.com/maasglob
18
19   - name: trip execution
20     description: supports the complete trip execution process. It contains f.i. getting an available asset, assigning the asset to the leg, star
21
22   - name: operator information
23     description: gives information about systems, stations, operating hours [from GBFS]
24
25   - name: payment
26     description: reports financial overview for legs
27
28   - name: support
29     description: support for the user while the leg is ongoing
30
31   - name: general
32     description: general operations (e.g. notifications)
33
34   - name: booking [optional]
35     description: endpoints that can facilitate processes in the booking process, but are not necessary for a minimal viable product. You can thin
36
37   - name: TO
38     description: the Transport Operator's endpoints
39
40   - name: MP
41     description: the MaaS Service Provider's endpoints
42
43 # security. Allowed methods basic (in header: Authorization: Basic ZGVtbzpwQU01dzByZA==),
44 # bearer (in header: Authorization: Bearer <token>)
45 # Api-key (in header: X-API-Key: abcdef12345)
46 # OAuth2 and OpenId are also available
47 # The exact ways to authenticate will be described in a later version
48 security:
49   - BasicAuth: []
```

# TOMP-API Presentation

## ■ Visualisation on app.swaggerhub.com

SMARTBEAR  
SwaggerHub

transport-operator\_maas\_provider\_api 1.4.0

Info  
Tags  
Servers  
Search  
planning  
booking  
trip execution

```
1 openapi: 3.0.0
2 info:
3   title: Transport Operator MaaS Provider API
4   description:
5     An API between MaaS providers and transport operators for booking
6     trips and corresponding assets.
7     <p>The documentation (examples, process flows and sequence
8     diagrams) can be found at <a href="https://github.com/TOMP-WG
9     /TOMP-API/">github</a>.
10  version: "1.4.0"
11  license:
12    name: Apache 2.0
13    url: "http://www.apache.org/licenses/LICENSE-2.0.html"
14  tags:
15    - name: planning
16      description: gives information about transport asset availability
17        and pricing [free_bike_status and system_pricing_plans in GBFS]
18        .<p> The endpoints in this part can give information about the
19        availability of assets (or assetTypes) and can provide
20        information to take the next step - the booking part.
21    - name: booking
22      description: a booking is the main object exchanged between MaaS
23        and a TO [from MaaS-API]. <br>See also <a href="https://github
24        .com/maasglobal/maas-tsp-api/blob/master/specs/Booking.md"
25        >Booking.md</a><p>This section contains functionality to book a
26        leg (part of a trip) for one asset (or assetType), including
27        the non-happy paths (cancel, expire etc).
28    - name: trip execution
29      description: supports the complete trip execution process. It
30        contains f.i. getting an available asset, assigning the asset
31        to the leg, starting, pausing, finishing a leg (all by using
32        the POST /legs/{id}/events) or updating an execution (not the
33        state!).
34    - name: operator information
35      description: gives information about systems, stations, operating
```

POST /planning/inquiries

POST /planning/offers

POST /bookings/one-stop

POST /bookings

a booking is the main object exchanged between MaaS and a TO [from MaaS-API]. See also Booking.md

booking

POST /bookings

POST /bookings/{id}/events

GET /bookings/{id}

trip execution

supports the complete trip execution process. It contains f.i. getting an available asset, assigning the asset to the leg, starting, pausing, finishing a leg (all by using the POST /legs/{id}/events) or updating an execution (not the state!).

VALID

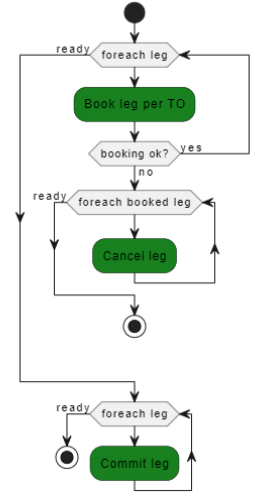


# TOMP-API Presentation

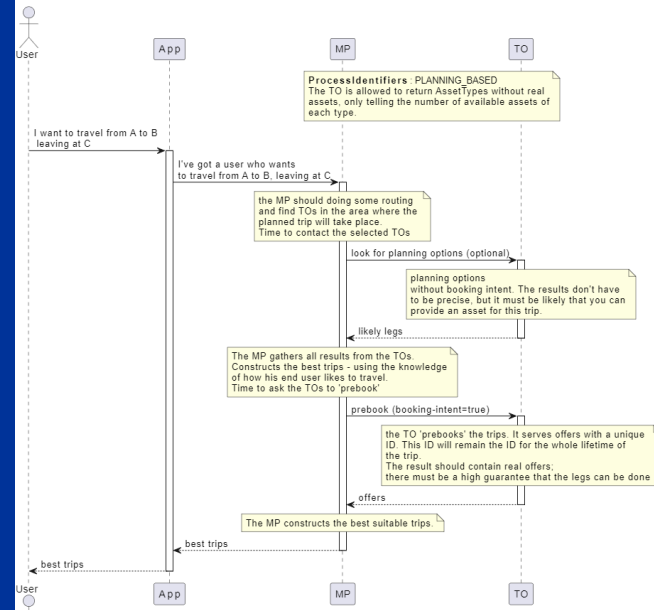
## ■ Interaction diagram PlantUML

- It is possible to define the action sequences for the selected use cases, and therefore the main TOMP API services
- Doesn't seem to be regularly updated
- Incompatible with CEN tools

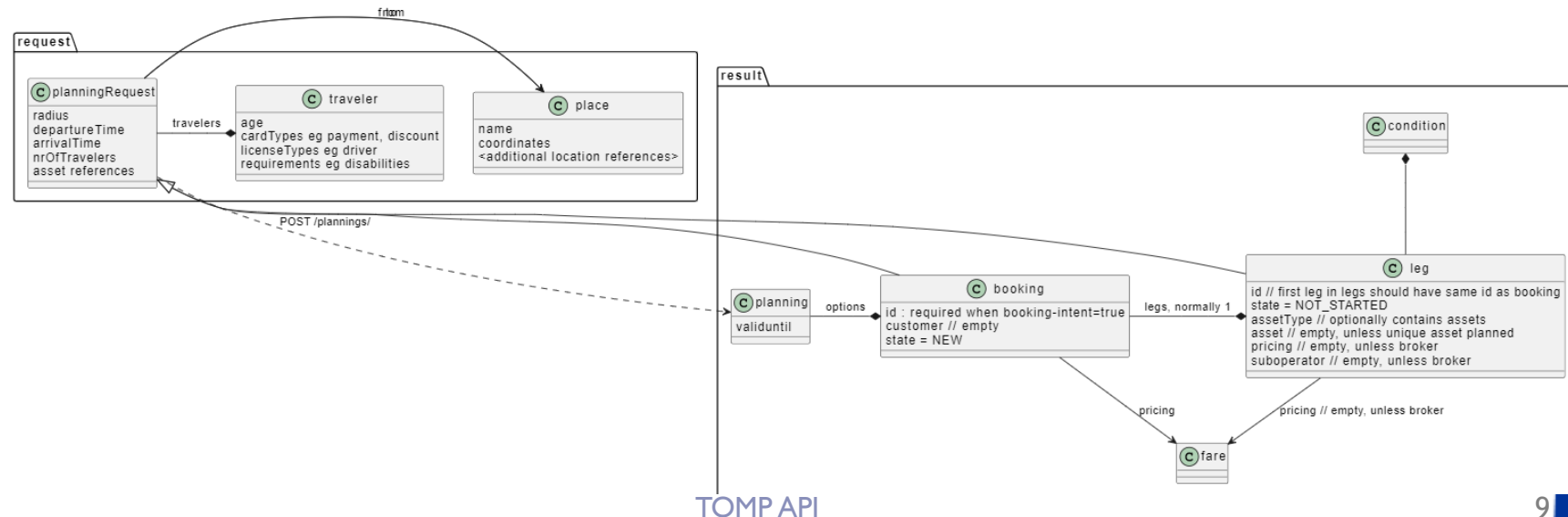
TOMP API workflow - booking process



Planning Scenario 'planned trip' - V1.0.0

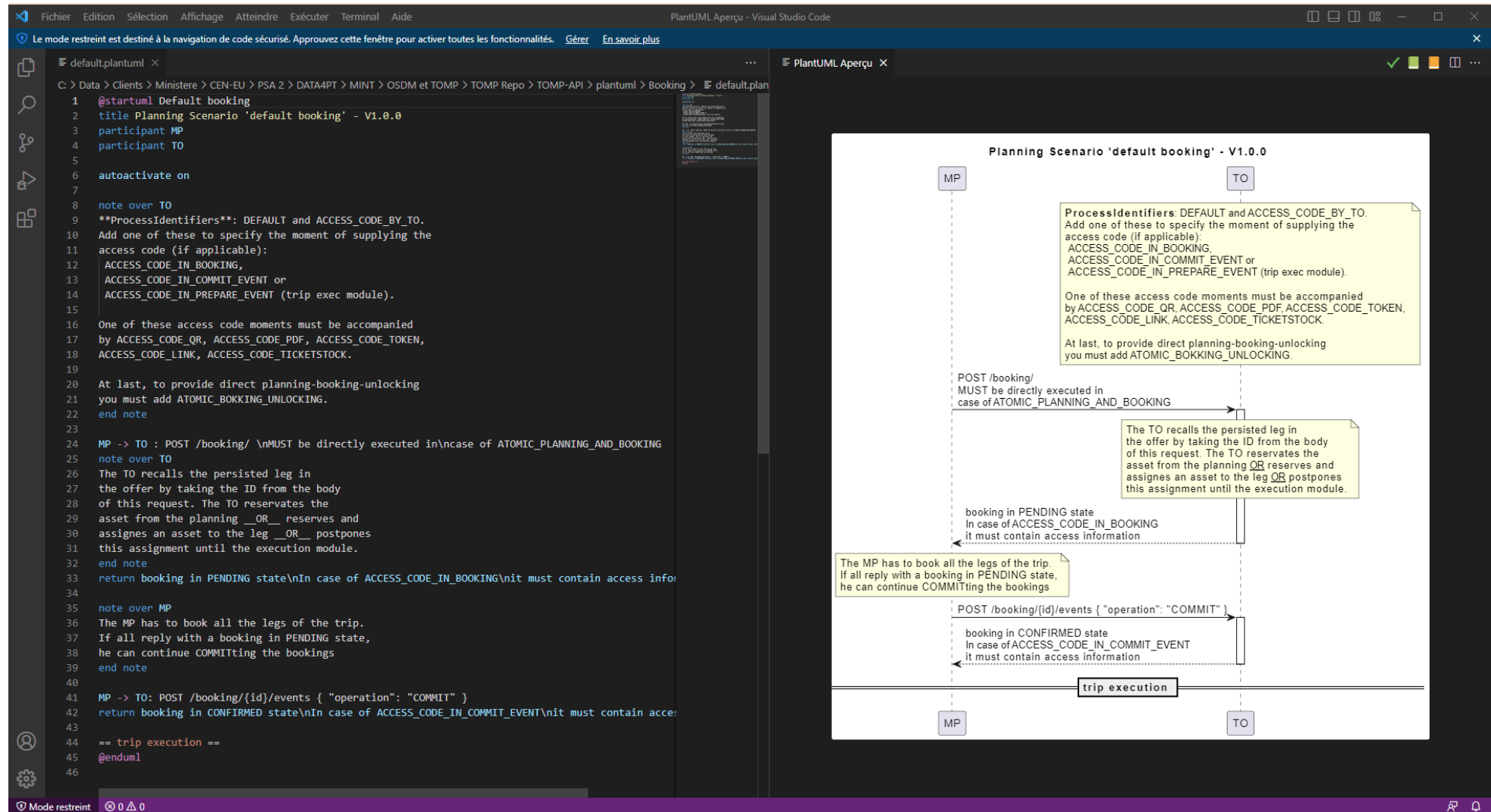


Planning phase



# TOMP-API Presentation

## ■ Interaction diagram PlantUML



# Documentation

## ■ The YAML is « self-documented »

- But this is not enough to answer all the questions (API documentation, no model or interaction)
- It also gives a very low level of vision which sometimes lacks a bit of hindsight.

1.4.1 TOMP-API / documents /

This branch is 32 commits behind master.

bonbakermans and edwinvandenbelt Add files via upload	811e6c0 2 weeks ago	History
..		
presentations	Add files via upload	3 years ago
proposals	Timestamp 159 0.6.0 (#170)	3 years ago
working group reports	Add files via upload	2 weeks ago
191021 - Blueprint for a TO-MP API v1.1.pdf	Add files via upload	4 years ago
200301 - Blueprint for a TOMP API v1.2.pdf	Add files via upload	3 years ago
200715 - Blueprint for a TOMP-API version Dragonfly.pdf	Add files via upload	3 years ago
201005 - Blueprint for the TOMP-API-Version Dragonfl...	Add files via upload	3 years ago
CROW passenger characteristics V2.0.xlsx	Add files via upload	2 years ago
CROW passenger characteristics.xlsx	Add files via upload	2 years ago
CROW-traveler_characteristics_en.json	Add files via upload	2 years ago
Connecting a MSP to the MaaS-NL-Router, version 1.0 ...	Add files via upload	4 years ago
Information document for testing parties within MaaS ...	Add files via upload	4 years ago
List of links and resources	Rename List of links to List of links and resources	4 years ago
MaaS functionalities and support of TOMP.xlsx	Add files via upload	3 years ago
Strategical committee.docx	378-via-possibility	2 years ago
TOMP-API Glossary V1.0.docx	Add files via upload	3 years ago
TOMP-API_Payment Proposal_20191211.pdf	Add files via upload	3 years ago
Woordenboek Reizigerskenmerken CROW.pdf	Add files via upload	4 years ago

# Documentation

- <https://github.com/TOMP-WG/TOMP-API/wiki>



## Operator information

Information about the operator. Where does it operate, what are its opening hours. Most of these endpoints contain static data, except for the available-assets. [more...](#)

## Planning phase

The planning phase actually consists out of 1 endpoint but has 2 modes: one for the routing (returns options, without registering anything) and one for getting offers (creates offers that can be booked, with a generated id for the complete life cycle). Completely controlled by one query parameter: booking-intent.

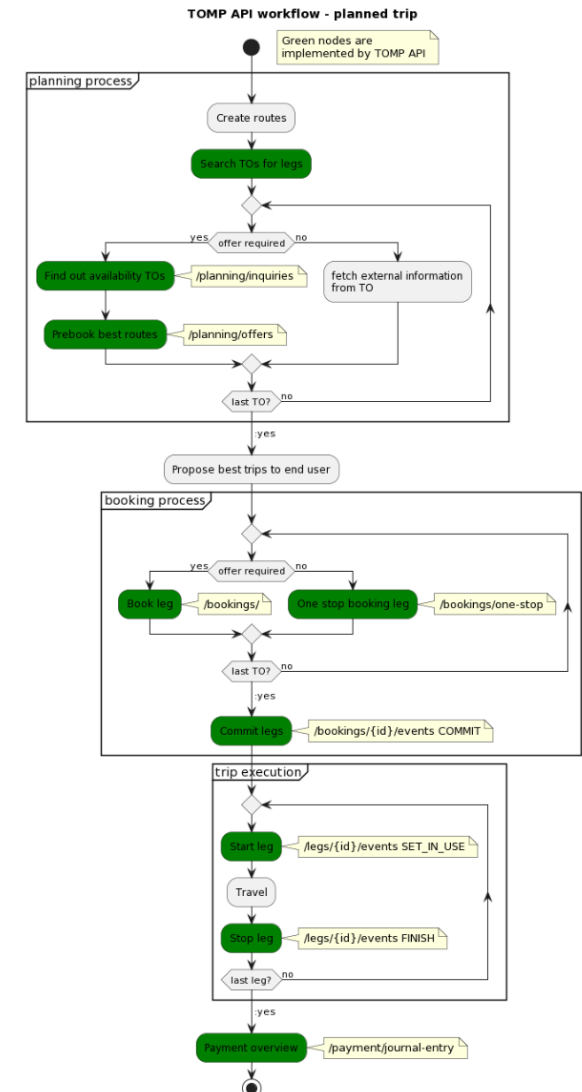
- In version 1.3 this is split into the endpoints 'inquiry' and 'offer'.
- In version 1.4 the planning phase can be skipped when all booking information is already provided in 'static' information (using Operator Information Module or external data source, like GBFS, NeTeX or alike) [more...](#)

## Booking phase

The booking is a transactional process. The booking scenarios are:

- up to version 1.3: one of the offers of the planning phase is booked (using the generated id).
- after 1.3.0: the 'one-stop-booking' endpoint can be used as well, if offers are not needed (e.g. 'I want bike no. 18').

After all the TOs have responded positively to the bookings, each TO has to be confirmed by a commit. (Even this can be bypassed using the [process identifiers](#)). [more...](#)



# TOMP Services

## The structure of TOMP-API



Source *Skedo*

# TOMP Services

## Operator Information

Note: NeTEx scope

get (all) /operator/ping  
/operator/meta  
/operator/stations  
/operator/available-assets  
/operator/alerts  
/operator/operating-calendar  
/operator/operating-hours  
/operator/information  
/operator/pricing-plans  
/operator/regions

This is a healthcheck endpoint to see if the TO is up and running  
Describes the running implementations  
Describes all available stations => NeTEx  
Returns a list of available assets (note: peu clair pour transport public) => NeTEx  
informs customers about changes outside of normal operations => SIRI  
describes the operating calendar for a system (issu de GBFS) => NeTEx  
describes the system hours of operation (issu de GBFS) => NeTEx  
Describes the system (operator, location, URLs, contact info...) => NeTEx  
Describes pricing of systems or assets [from GBFS] => NeTEx  
describes regions for a system that is broken up by region => NeTEx

## Planning

Note: OJP scope

post /planning/inquiries  
post /planning/offers  
post /bookings/one-stop

Returns **informative** options for the given travel plan (for O/D)  
Returns **bookable** offers for the given travel plan.  
Returns **bookable** offers for the given travel plan.

## Booking

post /bookings  
post /bookings/{id}/events  
get /bookings/{id}

### Optional:

get/bookings  
put/bookings/{id}  
post/bookings/{id}/subscription  
delete/bookings/{id}/subscription

Returns bookings that has been created earlier  
Adjust the parameters of the booking  
Subscribe to a specific booking (=leg & (type of) asset).  
Delete subscription

# TOMP Services

## Trip Execution

Note: SIRI scope

get /legs/{id}/available-assets	Returns a list of available assets for the given leg (Ticket ou course..?)
get /legs/{id}	Retrieves the latest summary of the leg => OJP
put /legs/{id}	Updates the leg with new information (from TO...)
post /legs/{id}/ancillaries/{category}/{number}	A new ancillary is added to the leg. => NeTEx/SIRI (ancillary=equipement?)
delete /legs/{id}/ancillaries/{category}/{number}	An ancillary (or amount) is removed to the leg. => SIRI
post/legs/{id}/events	Alter the state of a leg (ASSIGN_ASSET, TIME_POSTPONE, FINISH, etc.)
get/legs/{id}/progress	Monitors the current location of the asset and duration & distance of the leg (TO)
post/legs/{id}/progress	Monitors the current location of the asset and duration & distance of the leg (MP)
post/legs/{id}/confirmation	The TO can request confirmation for certain actions from the MP. ({id}=leg)

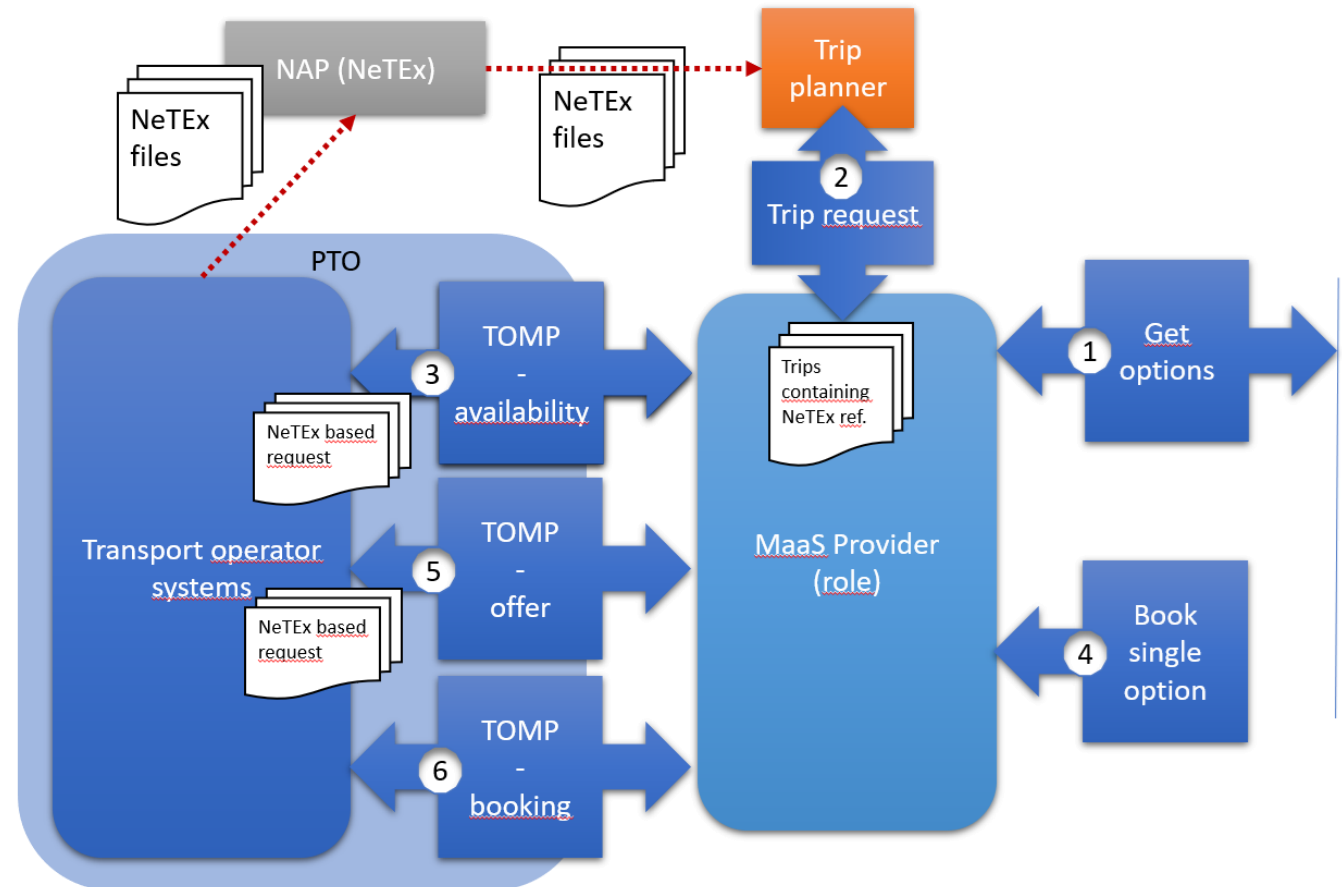
## Support

post/support/	Creates a request for support from end user via MP
get/support/{id}/status	Gets the status report of the support request.

## Payment

get/payment/journal-entry	Returns all the journal entries that should be paid per leg
post/payment/{id}/claim-extra-costs	Extra costs that the TO has to charge to the MP or vice versa. (?? not to user ??)

- **A relatively simplified overall architecture** (view with possible integration of NeTEx)

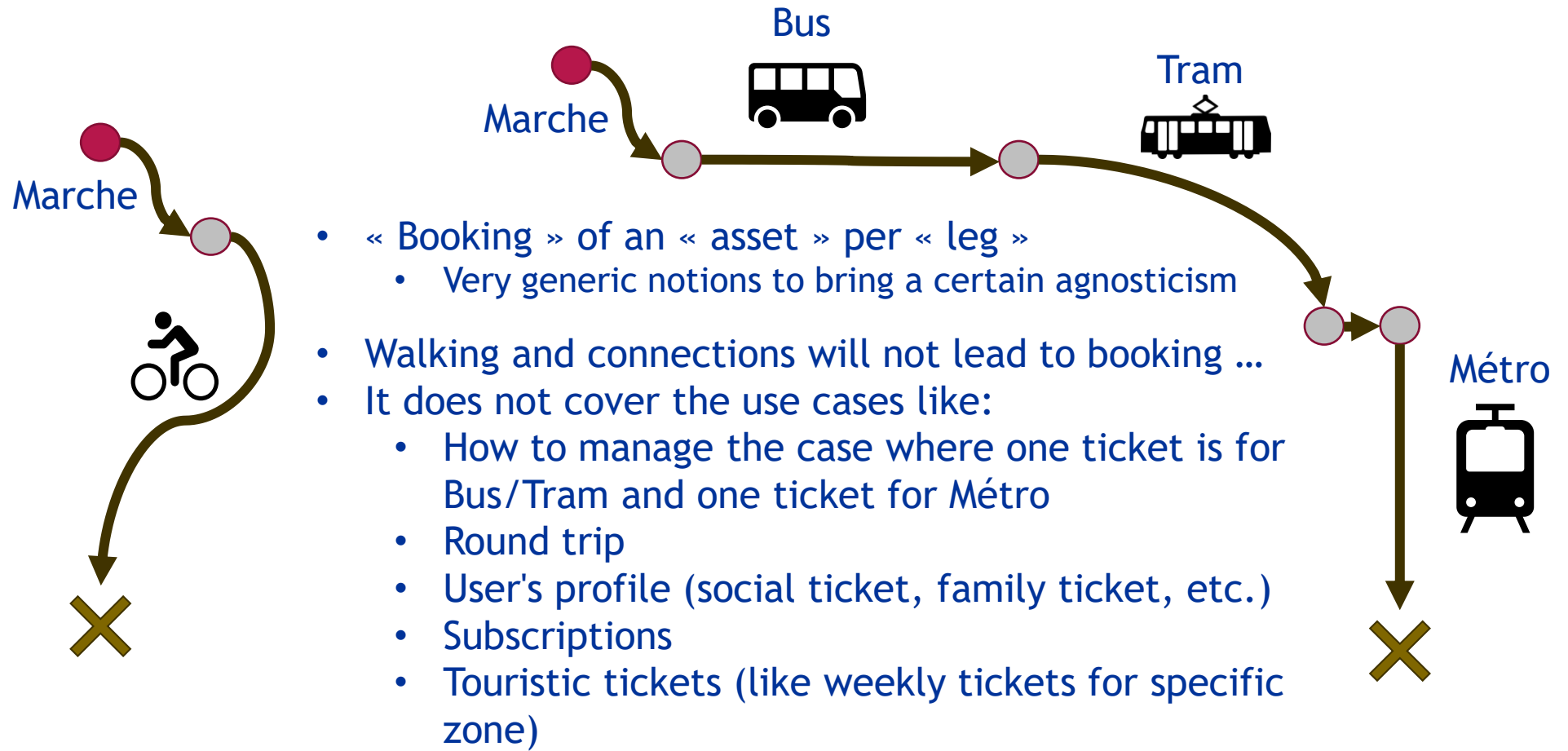
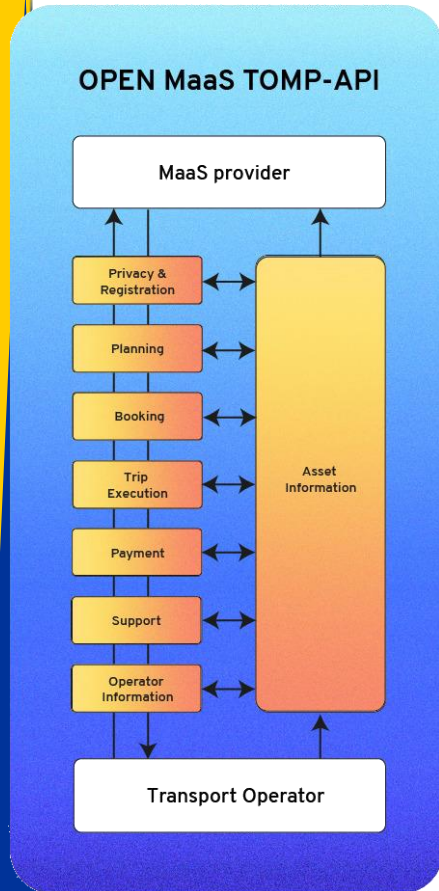


Mars 2023



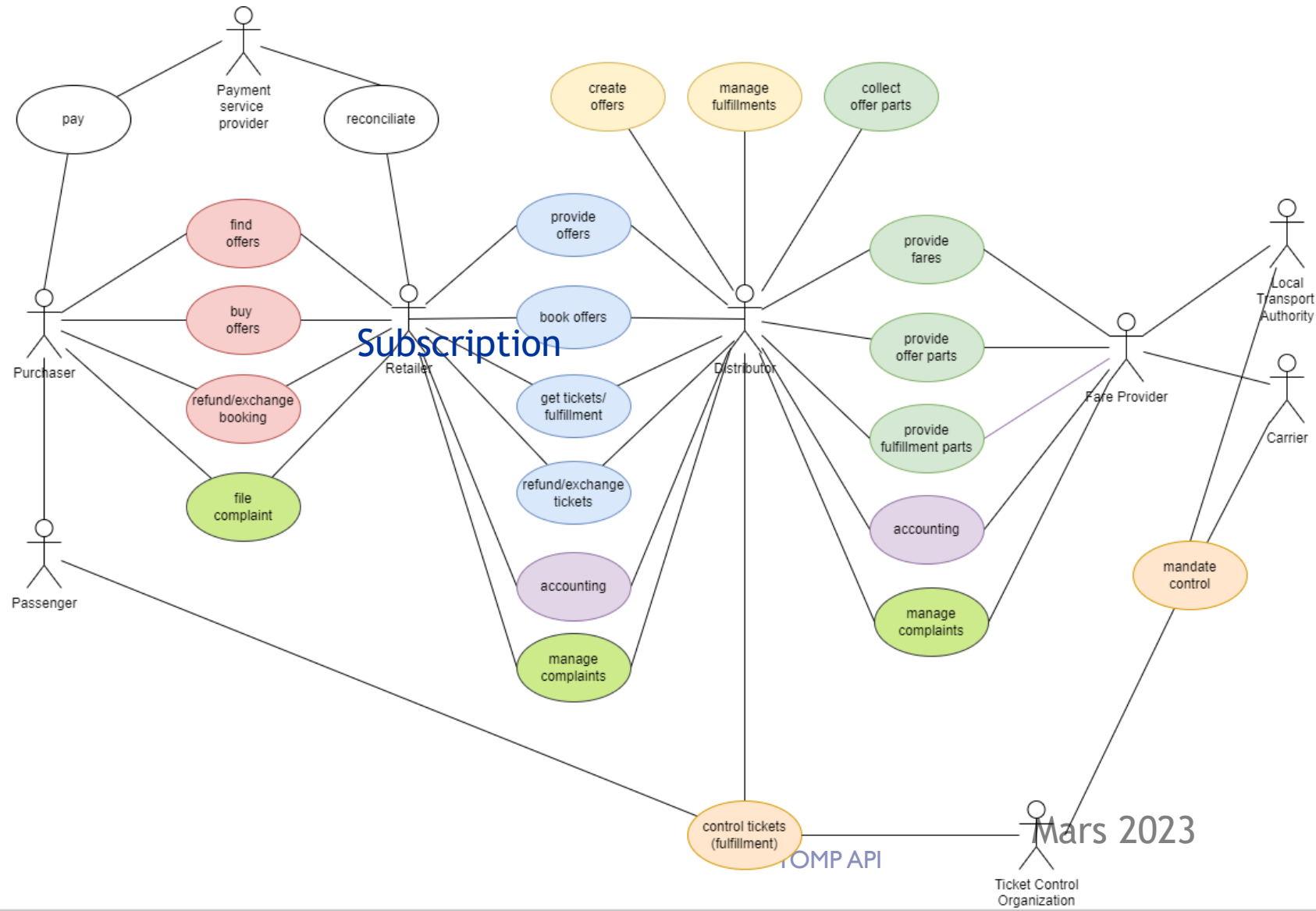
# Architecture

- A structure completely built on the route sections (leg)



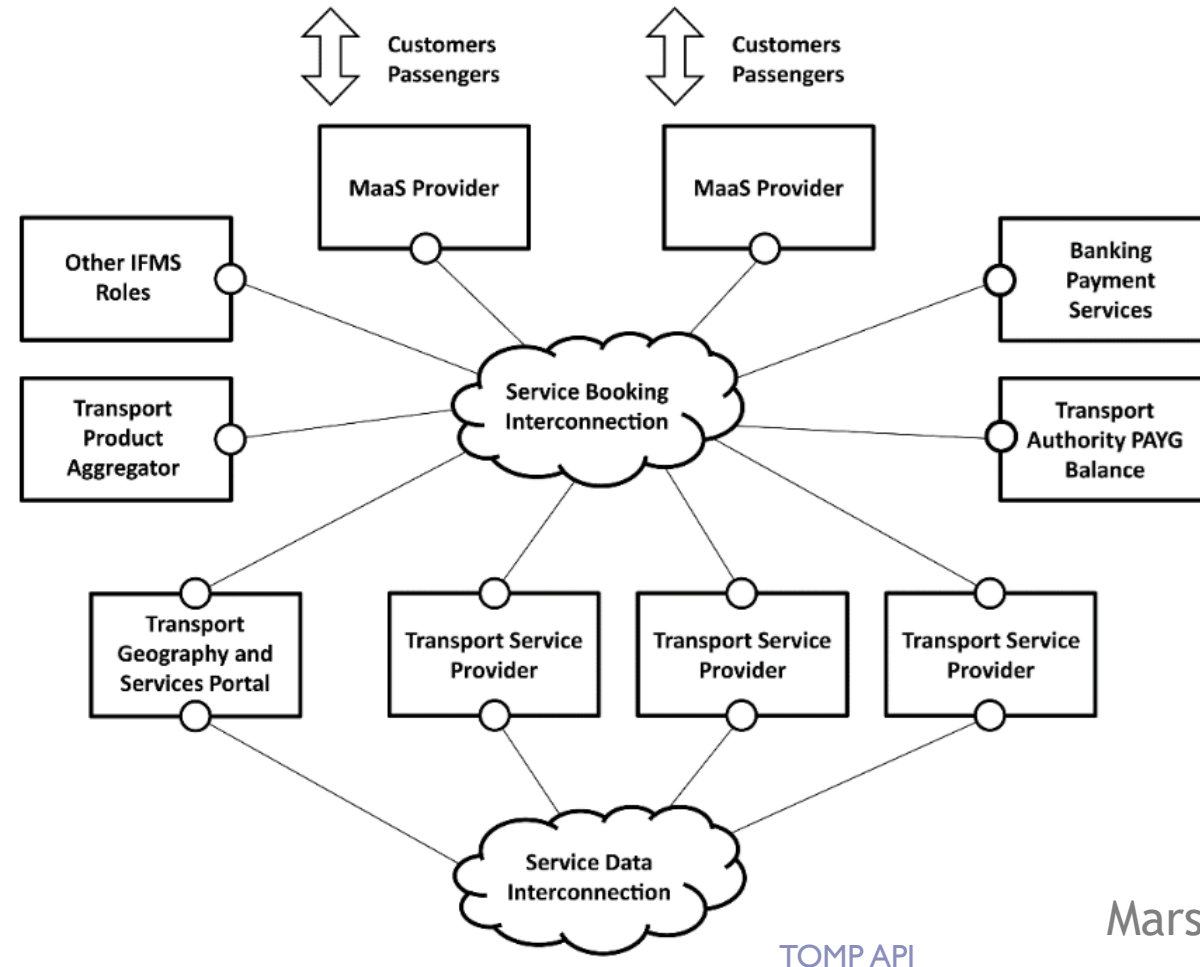
# System architecture

## ■ Comparison - OSDM: roles and actors (for comparison)



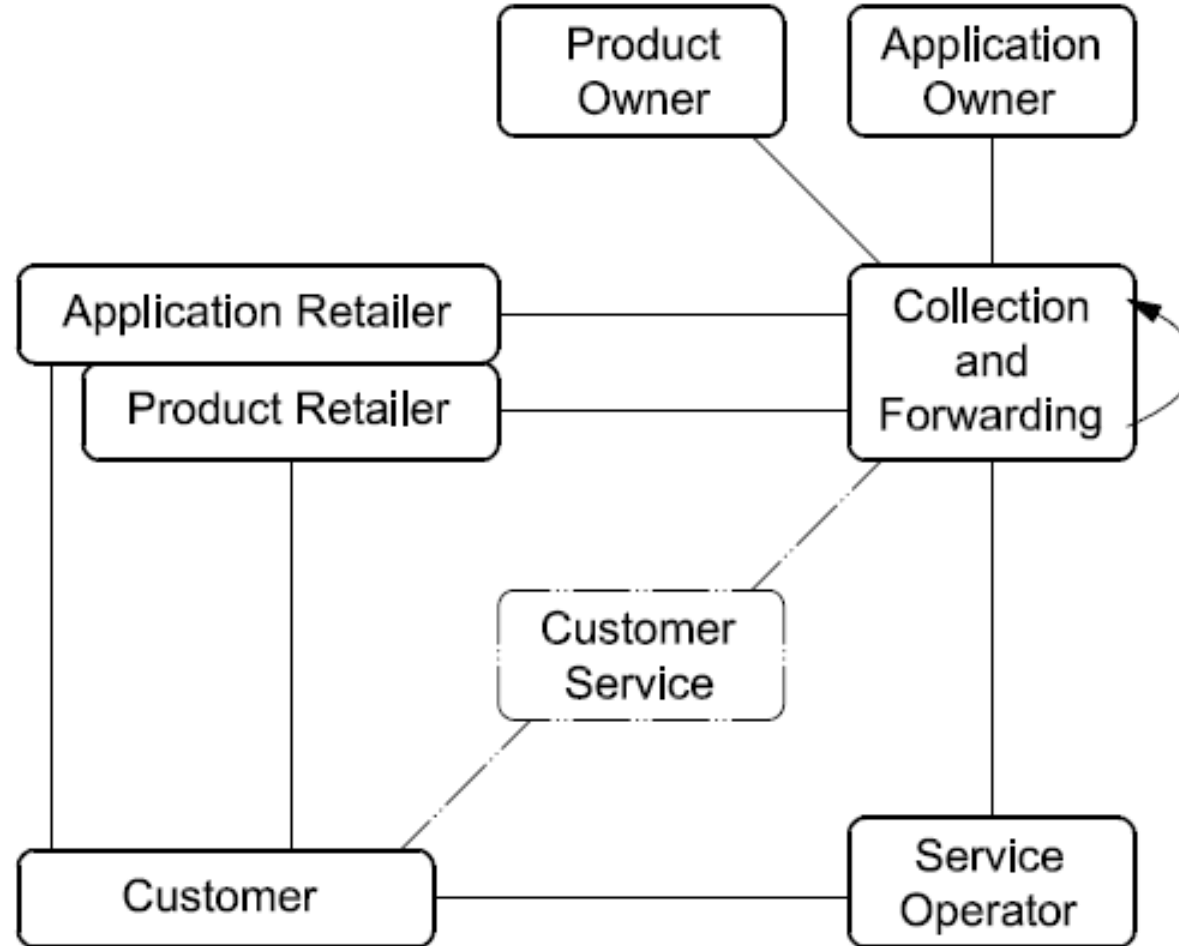
# System architecture

## ■ Comparaison - Transport distribution architecture – TR Distribution APIs of CEN/TC278 WG3



# System architecture

## ■ Comparison – Roles in IFM



# Relations with GBFS

- **GBFS (and bike sharing) is the main/initial use case TOMP-API was built for**
  - The example on **Github** concerne bike sharing
  - The notion *asset* aims to be generic but remains quite linked with the vehicle to book
  - It is deeply inside the **API** *(check next slide)*

# Relations with GBFS

```
asset ▾ {  
  id* > [...]  
  isReserved > [...]  
  isReservedFrom > [...]  
  isReservedTo > [...]  
  isDisabled > [...]  
  availableUntil > [...]  
  rentalUrl > [...]  
  rentalUrlAndroid > [...]  
  rentalUrlIOS > [...]  
  mileage > [...]  
  stateOfCharge > [...]  
  maxRange > [...]  
  licensePlate > [...]  
  stationId > [...]  
  homeStationId > [...]  
  overriddenProperties assetProperties ▾ { ◀
```

```
assetProperties ▾ { ◀  
  description: Aspects of an asset or assetType. Most aspects are optional and should  
               only be used when applicable.  
  
  name > [...]  
  location place > {...} ◀  
  brand > [...]  
  model > [...]  
  buildingYear > [...]  
  colour > [...]  
  maxSpeed > [...]  
  wheelCount > [...]  
  image > [...]  
  icon > [...]  
  accessMethods > [...]  
  fuel > [...]  
  propulsion > [...]  
  energyLabel > [...]  
  ecolabel > [...]  
  co2PerKm > [...]  
  gears > [...]  
  gearbox > [...]  
  airConditioning > [...]  
  cabrio > [...]
```

# Relations with GTFS

- The relations with planned data of public transport have been considered
  - We find them in Swagger (especially for the identification of stops )

```
3520
3521 - stopReference:
3522   type: object
3523   description: reference to a stop (can be nation specific). This can help to specific pinpoint a
3524   stop is not supplied; you should find it elsewhere.
3525   required:
3526     - type
3527     - id
3528     - country
3529   properties:
3530     type:
3531       type: string
3532       description: type of external reference (GTFS, CHB).
3533       enum:
3534         [
3535           GTFS_STOP_ID,
3536           GTFS_STOP_CODE,
3537           GTFS_AREA_ID,
3538           CHB_STOP_PLACE_CODE,
3539           CHB_QUAY_CODE,
3540           NS_CODE,
3541         ]
3542     id:
3543       type: string
3544       description: this field should contain the complete ID. E.g. NL:S:13121110 or BE:S:79640040
3545     country:
3546       $ref: "#/components/schemas/country"
```

```
stopReference v {
  description: reference to a stop (can be nation specific). This can help to specific pinpoint a (bus) stop.
  Extra information about the stop is not supplied; you should find it elsewhere.

  type* string
  type of external reference (GTFS, CHB).

  Enum:
    v [ GTFS_STOP_ID, GTFS_STOP_CODE, GTFS_AREA_ID, CHB_STOP_PLACE_CODE, CHB_QUAY_CODE, NS_CODE ]

  id* string
  this field should contain the complete ID. E.g. NL:S:13121110 or BE:S:79640040

  country* country string
  maxLength: 2
  minLength: 2
  example: NL
  two-letter country codes according to ISO 3166-1
```

Note the absence of NeTEx and the explicit reference to the Dutch national stop reference

# Relations with GTFS

- Found in Swagger (especially for identification of stops)

```
endpoint {
  description: a formal description of an endpoint.
  method*    > [...]
  path*      > [...]
  eventType  > [...]
  status*    > [...]
  supportsPaging > [...]
  maxPageSize > [...]
  externalType string
              this field must be used when adressng other standards for exchanging 'static' data (Level 1 MaaS)
  Enum:
    ▾ [ GBFS, GTFS, NeTEx, OSDM_Offline, IXSI5, APDS ]
  useAssetTypes > [...]
  useAssets     > [...]
}
```



# TOMP and the "classic" public transport

<https://github.com/TOMP-WG/TOMP-API/wiki/How-do-I-implement-a-public-transport-operator>

- **Identical scenario for Bus, Metro, Train...**
- **But no “composite-leg” currently in the YAML**
- **And this means that the journey planner must do the pre-division to be consistent with the tarification (example of a Bus+Tram+Metro trip in Paris)**
- **The reservation seems to refer to the transporter's services**



## How do I implement a public transport operator

Edwin van den Belt edited this page on Nov 16, 2020 · 7 revisions

[home](#) > [Modalities](#) > [Public transport](#)

Q: How to implement a Public transport operator?

A: There are a few specific items to take into account implementing a train TO. Per module:

Planning phase

- you have to use composite-legs when planning overlaps.
- scenario: reserve chairs is not yet implemented
- whenever you have to provide tokens, you probably have to request birthday, name etc. Therefore add the condition require-booking-data-condition: birthday to the response of your planning options.

Booking phase

- In case you can deliver an access token (QR code or other), you can put it into the booking response in

```
"token": {
  "startTime": "2020-06-28T14:55:00+02:00",
  "endTime": "2020-06-29T00:00:00+02:00",
  "meta": [
    {
      "QR": "base64 string containing the image"
    }
  ]
},
```

Of course it can be an azztec image.

Trip execution

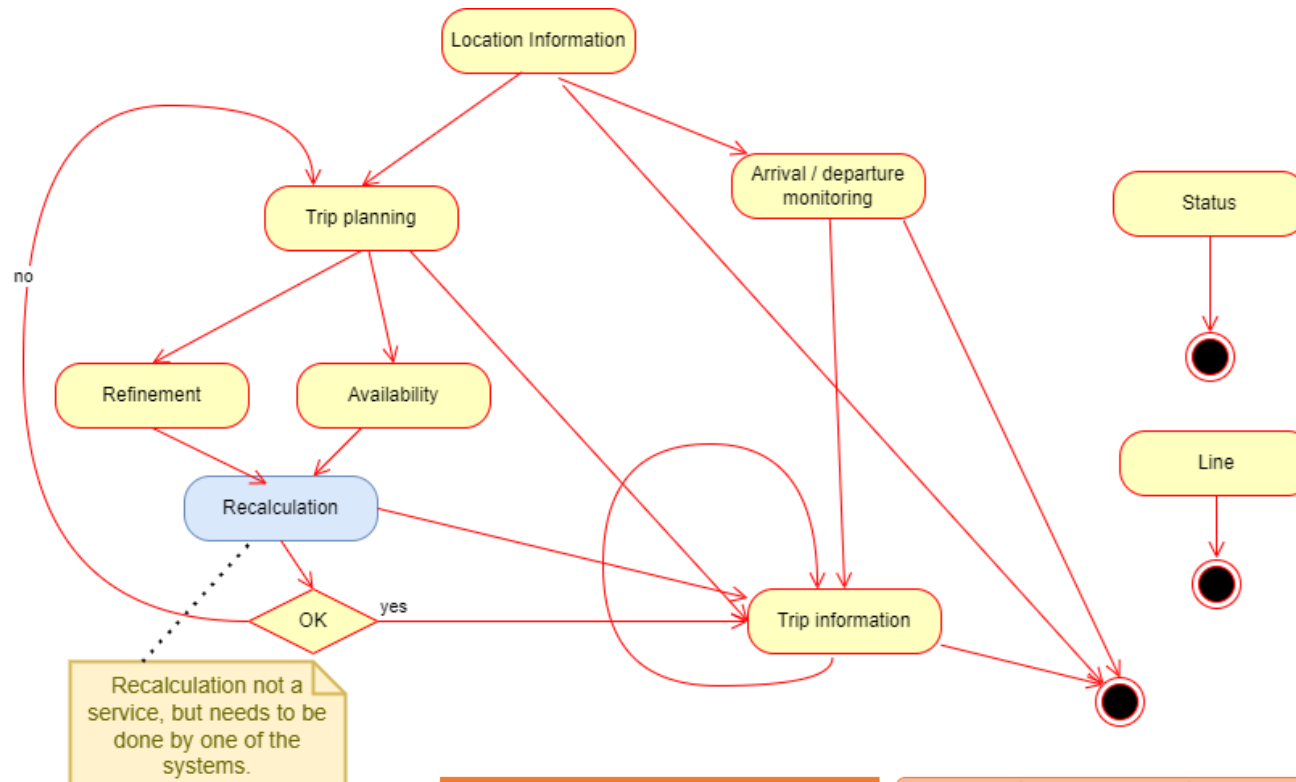
\* when you cannot create an access token at the time of booking, it should be retrieved before starting the leg using /executions/{id}/events, requesting a PREPARE. In the response there is

```
"assetAccessData": {
  "startTime": "2020-06-28T14:55:00+02:00",
  "endTime": "2020-06-29T00:00:00+02:00",
  "meta": [
    {
      "QR": "base64 string containing the image"
    }
  ]
},
```

In both cases the start and end time are used to specify the time window of validity. \* In case of required birthday etc, the MP must supply the required fields.

# Position comparing with OJP

- OJP covers (much) more than the Planning service of TOMP
  - OJP Services and utilisation sequence



Recalculation not a service, but needs to be done by one of the systems.

Planning  
Note: OJP scope

`post /planning/inquiries`  
`post /planning/offers`  
`post /bookings/one-stop`

# Position comparing with OJP

- **OJP covers (much) more than the Planning service of TOMP**
  - **OJP Services and utilisation sequence**

<b>Location information</b>	matching text input against possible origin and destination locations
<b>Exchange points</b>	Connexion point for Distributed journey planning
<b>Trip request</b>	intermodal trip information from an origin location to a destination taking various user preferences into account
<b>Distributed journey planning</b>	Distributed journey planning
<b>Departure/arrival board</b>	provides information on arrivals and/or departures of public transport services from stops for a requested time or period of time
<b>Trip/Vehicle information</b>	information on a single trip (service pattern, real-time status, vehicle facilities etc .).
<b>Ticket price calculation</b>	general, stop-specific and trip-specific fare information
<b>Availability information</b>	informs about the availability of a MOBILITY SERVICE, a VEHICLE, SERVICE JOURNEY, or SINGLE JOURNEY
<b>Line information</b>	informs about about a specific line
<b>Refinement of trips</b>	additional or updated information about a known, previously retrieved trip

Note: OJP makes several references to TOMP and consistency with TOMP has been taken into account

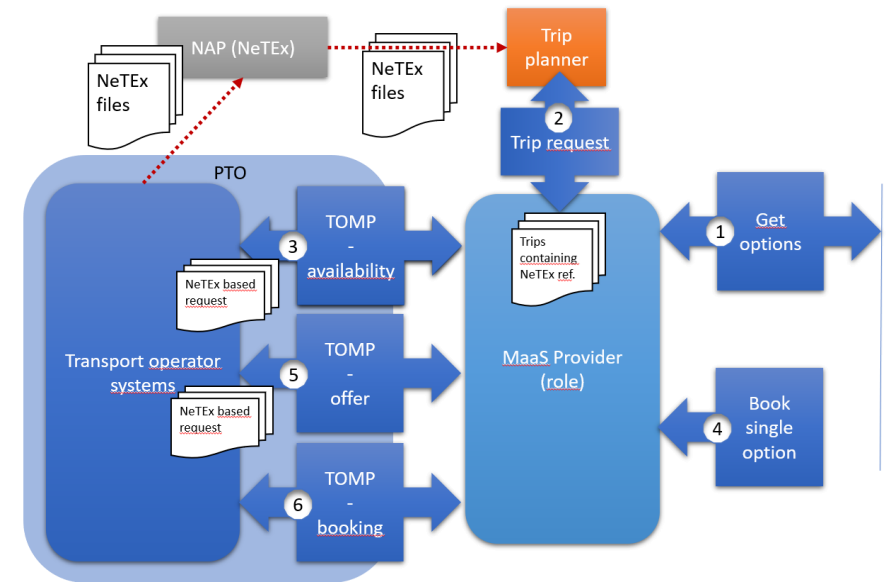
# Position comparing with NeTEx

- NeTEx covers all the "Operator Information" part and much more
- - For all the modes considered... and more
- Only identifiers are exchanged (exchangeable)
- This ensures agnosticism ... in a limited way

- It has many TOMP element that could advantageously reuse Transmodel Model and Implementation NeTEx

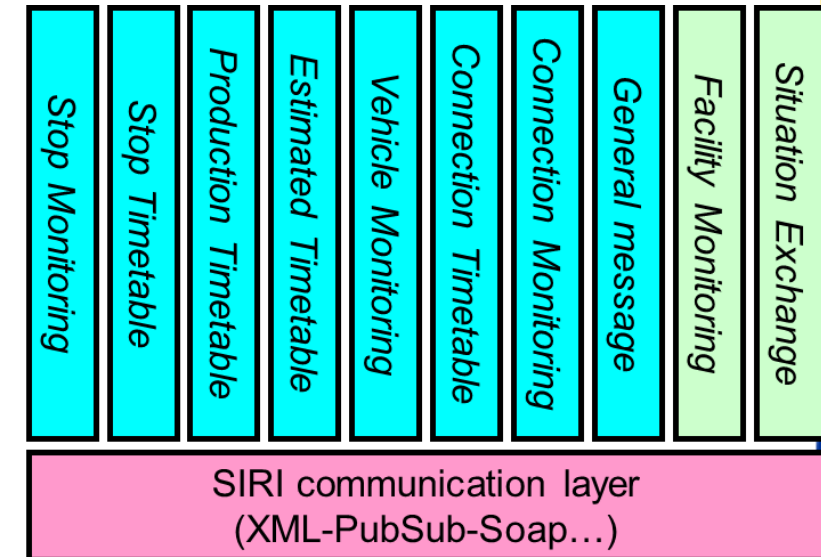
- Leg, Operator, Address, Place, Stop Place, Day & Day Types, etc. etc.

- The concept of FARE PRODUCT (tariff product) would have been useful!



# Position comparing with SIRI

- SIRI Covers all of TOMP's Trip Execution needs (and much more)
- 
- The main difference is the fact that TOMP focuses on "Legs" and SIRI on trips
- 
- If NeTEx appears in a certain number of presentations and documents related to TOMP, SIRI has been largely forgotten
- 
- Also note that SIRI has set up a "Communication Layer" shared with OJP and NeTEx, and which could have been reused
- 



## Trip Execution

Note: SIRI scope

get /legs/{id}/available-assets	Returns a list of available assets for the given leg (Ticket ou course..?)
get /legs/{id}	Retrieves the latest summary of the leg => OJP
put /legs/{id}	Updates the leg with new information (from TO...)
post /legs/{id}/ancillaries/{category}/{number}	A new ancillary is added to the leg. => NeTEx/SIRI (ancillary=equipement?)
delete /legs/{id}/ancillaries/{category}/{number}	An ancillary (or amount) is removed to the leg. => SIRI
post/legs/{id}/events	Alter the state of a leg (ASSIGN_ASSET, TIME_POSTPONE, FINISH, etc.)
get/legs/{id}/progress	Monitors the current location of the asset and duration & distance of the leg (TO)
post/legs/{id}/progress	Monitors the current location of the asset and duration & distance of the leg (MP)
post/legs/{id}/confirmation	The TO can request confirmation for certain actions from the MP. ({id}=leg)

# Replying to key questions

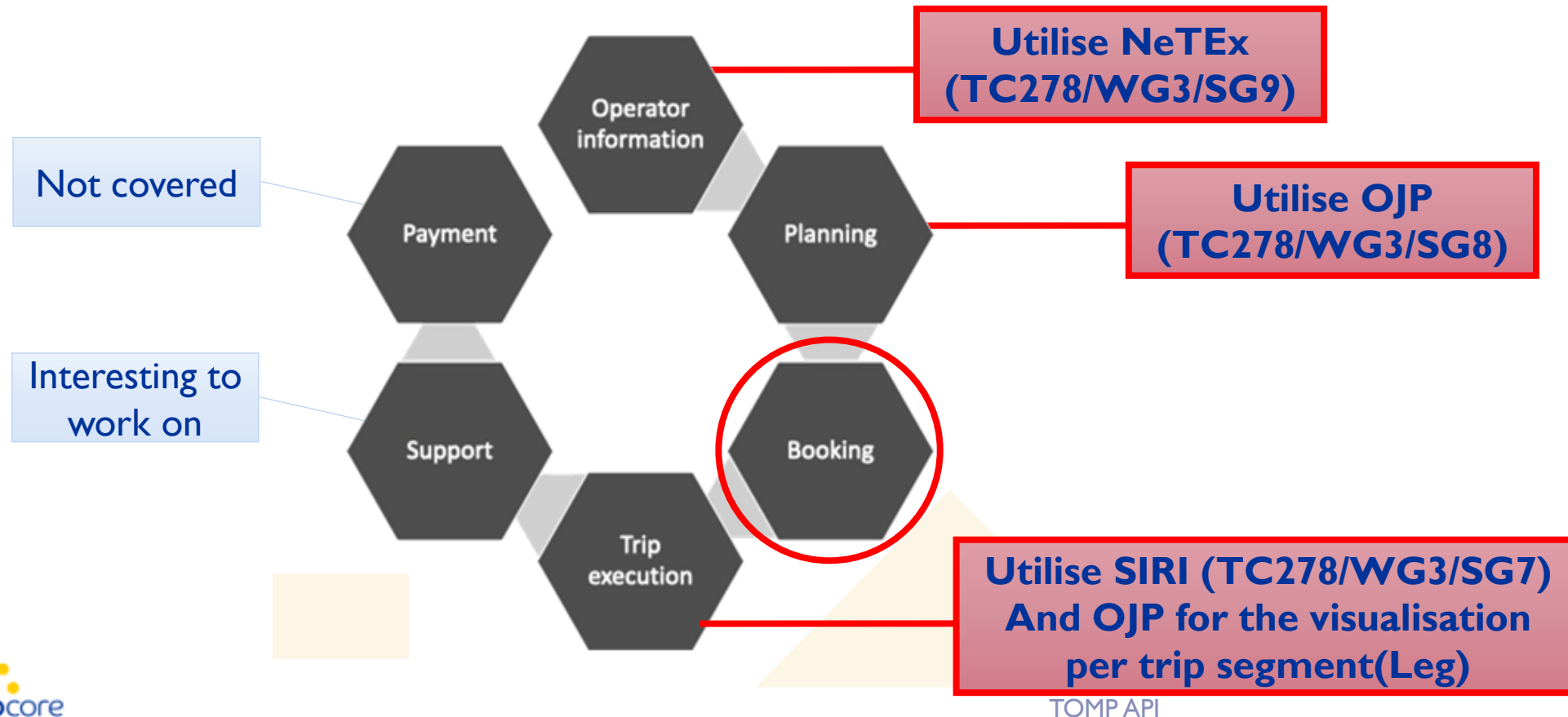
- **Can TOMP-API be used for new mobility modes but also for public transport and the planning of multimodal journeys?**
- **TOMP does not provide the planning itself (it is an interface relay)**
  - **OJP is much more complete**
- **The reservation part (booking) is suitable but only for some particular use case... starting from a route search and with routes where each section can lead to the reservation of an independent “asset”**

# Replying to key questions

- **TOMP-API can be used for: "planning ", booking, purchasing, payment...?**
- **The focal point is really the reservation**
- **Purchase and payment are not supported**
- **Planning remains simple (and relevant to target modes), but OJP is more comprehensive**

# Replying to key questions

- What is the link between the "planning", as prevented by TOMP-API, and the theoretical info included in TC278/WG3/SG9 and the service included in TC278/WG3/SG8 ?





# Replying to key questions

- **TOMP-API, supposed to be standards-agnostic, can it ensure interoperability between:**
  - data (traveller info) provided in a **CEN NeTEx** format and other data related to planning, booking and/or purchasing a transport or mobility service?
  - data related to the **CEN OJP** standard and other data related to the reservation and/or purchase of a transport or mobility service?
- **In a very limited way (covers only a very small subset of the functional scope)**

# Replying to key questions

- **At which level TOMP-API can handle dynamic data with regard to SIRI services?**
- **Very partially**
- **With a view by route section or by equipment**
- **The view by trip segment can be provided by OJP**

# Conclusion

- A good protocol but with relatively limited use cases
- The itinerary structure with the “Booking” of an “asset” by “leg” only covers part of the needs
- But offers a first option for booking management
- It cannot be replaced in any way or offer an access interface to OJP, NeTEx and SIRI
- The evolution to fully take into account the Transmodel ecosystem would lead to many "acrobatics" and will be tricky because of the fairly deep interweaving of bike sharing in the API
- The payment still remains uncovered

# Other projects to have in mind

- **OSDM**

- <https://osdm.io/spec/getting-started>

- **CoRoM projet (CEN)**

- CEN project funded by European Commission

- **European Commission, DGMOVE => MMTIS/MDMS**

- **MMTIS – Delegated Regulation (UE) 2017/1926**

Ongoing revision

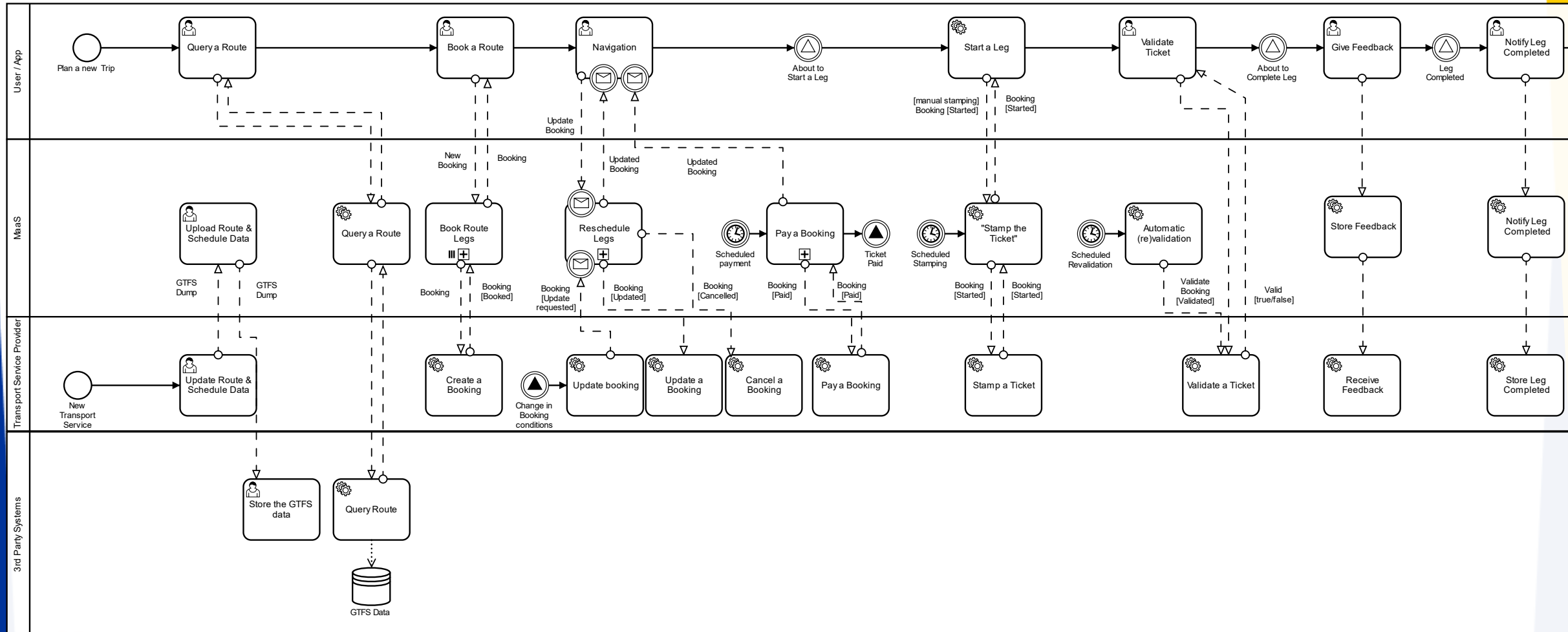
- **MDMS Multimodal Digital Mobility Services**

*MDMS means a service providing information on traffic and travel data such as location of transport facilities, schedules, availability or fares for more than one transport mode, which may include **features enabling the making of reservations, bookings or payments or the issuing of tickets.***

# Other projects to have in mind

- **Whim API** ([source GitHub maasglobal/maas-tsp-api](https://github.com/maasglobal/maas-tsp-api) référencé par le Swagger TOMP)

Voir aussi <https://maasglobal.github.io/maas-tsp-api/redoc.html#section/Updating-a-booking>



# Other projects to have in mind

- **BoB (Samtrafiken)**

<https://samtrafiken.atlassian.net/wiki/spaces/BOB/overview>

- **SKI+ (Suisse)**

<https://transportdatamanagement.ch/de/standards/#standards-SKI+>  
<https://transportdatamanagement.ch/content/uploads/2023/01/TOMP-API-SKIProfil-0.5.pdf>  
[https://github.com/openTdataCH/ojpch/tree/main/doc/ojpfare2tomp\\_analysis](https://github.com/openTdataCH/ojpch/tree/main/doc/ojpfare2tomp_analysis)

- **Oslo (Belgium)**

<https://www.its.be/maas/technical-harmonisation/oslo>

- **eHub - eHUBS - Smart Shared Green Mobility Hubs (Interreg) – TOMP**

Based

[https://www.nweurope.eu/media/12785/d63\\_api\\_standard\\_for\\_information.pdf](https://www.nweurope.eu/media/12785/d63_api_standard_for_information.pdf)

- **MaaS Global (Finland)**

<https://whimapp.com/about-us/>  
<https://github.com/maasglobal/>

# Other projects to have in mind

- **Kasia Bourée et Edwin van den Belt**

<https://www.linkedin.com/pulse/best-both-netex-tomp-api-edwin-van-den-belt/>

- **Matthias Guenter**

<https://transportdatamanagement.ch/fr/standards/>

- **Skedo**

<https://skedgo.com/introducing-the-tomp-api/>

- **TOMP WG**

- **MaaS Alliance**

- **OJP**

- **OSDM**

- **CEN**

# Thank you for your attention