Dragonfly

Blueprint for an API

From Transport Operator to MaaS Provider

30/09/2020

TOMP WG

MaaS
Mobility as a Service

Blueprint for an

Application Programming Interface (API)

from Transport Operator to MaaS Provider

---

a technical milestone towards Mobility as a Service

Version Dragonfly (1.0)
30-09-2020

Dragonfly

Table of Contents

Introduction

With Mobility as a Service (MaaS) travellers can plan, book, execute their trips using any available transport mode and pay for all of them via integrated apps. For MaaS to be successful, transport operators are required to share their transport services and availability of their assets in a digital form. To facilitate MaaS providers and thus enable the deployment of MaaS services, transport operators are also required to standardize the digital form to facilitate access to their information. The TOMP-API (Transport Operator to MaaS Provider - Application Programming Interface) is a standardized and technical interface between MaaS providers and transport operators.

Figure 1 below depicts the concept of having a standard-based Application Programming Interface (API) from Transport Operators (TO) to or from MaaS Providers (MP). It allows all participating companies to communicate about planning, booking, execution, support, general information and payments of multimodal, end-user specific trips. Using the TOMP-API enhances the interoperability between parties in the MaaS ecosystem.

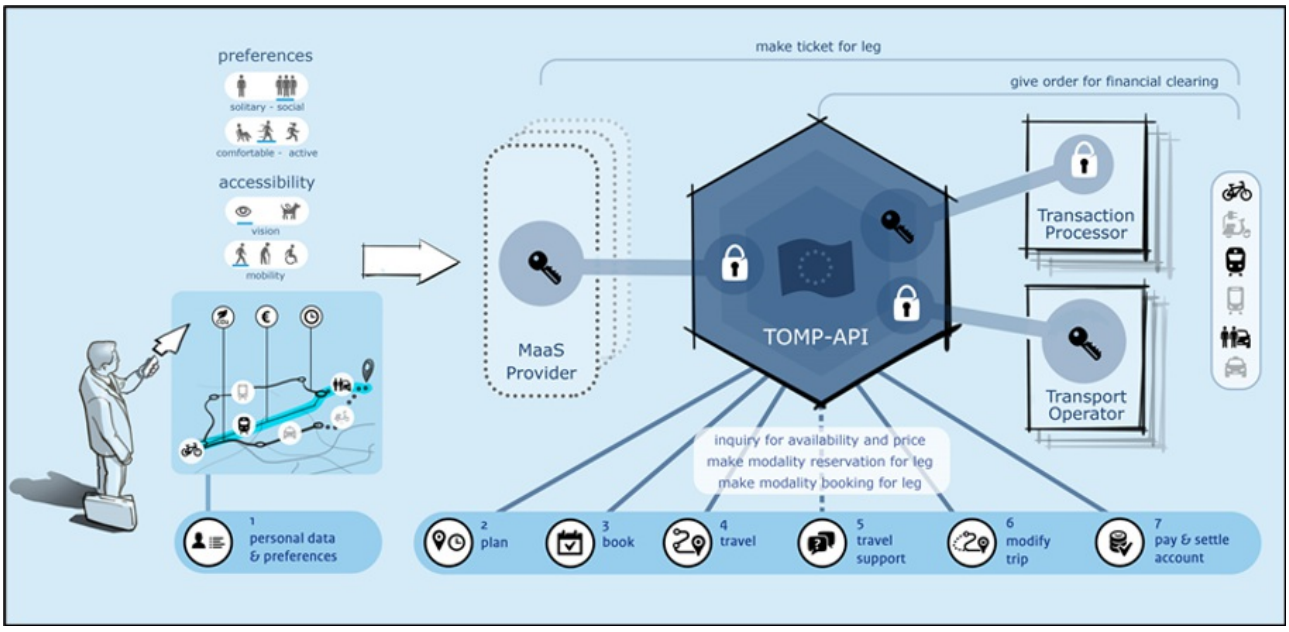


Fig. 1: The standard-based API for Transport Operators to/from MaaS providers (Source: MaaS program of the Dutch Ministry of Infrastructure and Water Management)

Goal of the blueprint

In this Blueprint for an API for TOs and MPs (the TOMP-API) we look into the necessary functional requirements for the interoperability between transport operators and MaaS Providers. The goal of this document is to:

- Define the necessary scope for full interoperability between TOs for the deployment of MaaS services, always keeping the customer journey in mind to determine which API calls are needed between TOs and MPs.
- Define the necessary parameters and values to fulfil this scope.
- Define the available parameters in various already available APIs and propose amendments where applicable.

Who is involved?

This document has been written to consolidate the work of the Transport Operators and MaaS Providers - Working Group (TOMP-WG). The TOMP-WG is an initiative started in the Netherlands by the *Ministry of Infrastructure and Water Management* in 2018. The goal of the group is to provide standardised APIs to facilitate the development of the MaaS ecosystem. TOMP is used as standard for the seven MaaS pilots in The Netherlands. Since 2020 the TOMP-WG has been moved to become an open-source foundation with an international scope. A list with all collaborators, companies and stakeholders involved in the current design and development of the TOMP-API is provided in Appendix A.5

What is in this version?

The first implementations of the TOMP-API took place in the last couple of months which allowed us to improve the API based on the lessons learned. The API is now capable of describing a full MaaS journey. Only minor changes have been added to this Dragonfly version.

These results are especially reflected in a simplified object model in the planning phase (flattened the object structure of the leg) and a new endpoint with self-describing facilities was created. This last one is needed for (inter)national scale-up, to be informed of what the addressed TO is capable of.

The digital version of the API is available for consultation at Swaggerhub : https://app.swaggerhub.com/apis/TOMP-API-WG/transport-operator_maas_provider_api/.

Scope of the TOMP-API

In 2018 the Dutch Ministry of I&W (Infrastructure and Water management) initiated 7 MaaS pilots. As part of the agreement for facilitating these pilots, participating parties are required to cooperate in a standardization process.
The goal of this standardization process is to create an ecosystem where Transport Operators and MaaS providers can interact in a standardized way. To maintain the standard an open source community was established (TOMP WG).

TOMP facilitates the MaaS ecosystem by describing how the communication can be done between the Transport Operators and MaaS Providers, but it’s not a full ecosystem. In a working ecosystem several other items are required. These are the services that are at least required:

- a discovery service to find out which TOs are operating in a certain area
- an authentication service for the parties using the ecosystem
- at least one clearing house (an independent party guaranteeing payment)

Other items could come in handy, but are not required:

- a travel right stock (to facilitate exchange of validation/access information),
- governance organizations (to control legislation)
- personal data stores

These ecosystem facilities are not yet in place, but will be there in the (near) future. The discovery service is under investigation for development now, the authentication is a needed and the TOMP parties are trying to agree on one way of authentication. In Belgium the personal data stores are under development nowadays. In other words, in the upcoming months the first contours of the ecosystem will show.

EU legislation requires its countries to implement a National Access Point, the discovery service is a part of it. Setting up other services is left to the market.

Versioning and releases

Changes in the API are inevitable since we are exploring a new field and knowledge and experience improves. These changes are controlled using milestones and semantic versioning. First, the WG defines functional milestones for the API. The milestones refer to new capabilities of the API at a point in time. The most recent version is TOMP Dragonfly and contains all functional aspects of a MaaS journey, (operator information, planning, booking, travel, payment, support). Milestones have animal names, in alphabetic order. Secondly, for developers and implementors of TOMP, semantic versioning is used. Semantic versioning means that by looking at our version number, you can quickly identify what has changed and how much work goes into changing your own implementation. Table 1 below shows the different TOMP versions and the major updates - (https://github.com/TOMP-WG/TOMP-API/wiki#updates).

Version	Release date	API content
TOMP 0.0.3	November 2018	User stories defined and first description of planning module, based on GBFS
TOMP 0.0.7	January 2019	Modules for planning and booking added
TOMP 0.1.1	October 2019	Trip execution added and API was adjusted to a consistent and uniform REST format
TOMP 0.1.2	March 2020	Added license and first description of payment module
TOMP 0.5.0	May 2020	All modules for a complete MaaS journey including support and payment
TOMP 0.9.0 (prerelease Dragonfly)	July 2020	Added simplified object model in planning phase and a self-describing endpoint
TOMP 1.0.0 (Dragonfly)	September 2020	Minor changes and feedback from first full implementations

Table 1. Overview of all TOMP-API releases

Process Flows

Together with the eMaaS project team from the University of Twente, process flows for the customer journey have been defined. This helps to scope the necessary functions required in the API building blocks. The goal is to accommodate different business models and variations of transportation within these functional flows. Asset information can be shared for both free-floating systems (bike sharing, car sharing, ride sharing, taxi) and (virtual)station- or fixed-route- based systems (public transport, (virtual)mobility hubs, or station-based transportation) through the functional descriptions provided in this chapter.

Functional Blocks

The TOMP-API is composed of 8 functional blocks. Fig. 2 below aims at giving a general overview of the different functional modules within the TOMP-API.

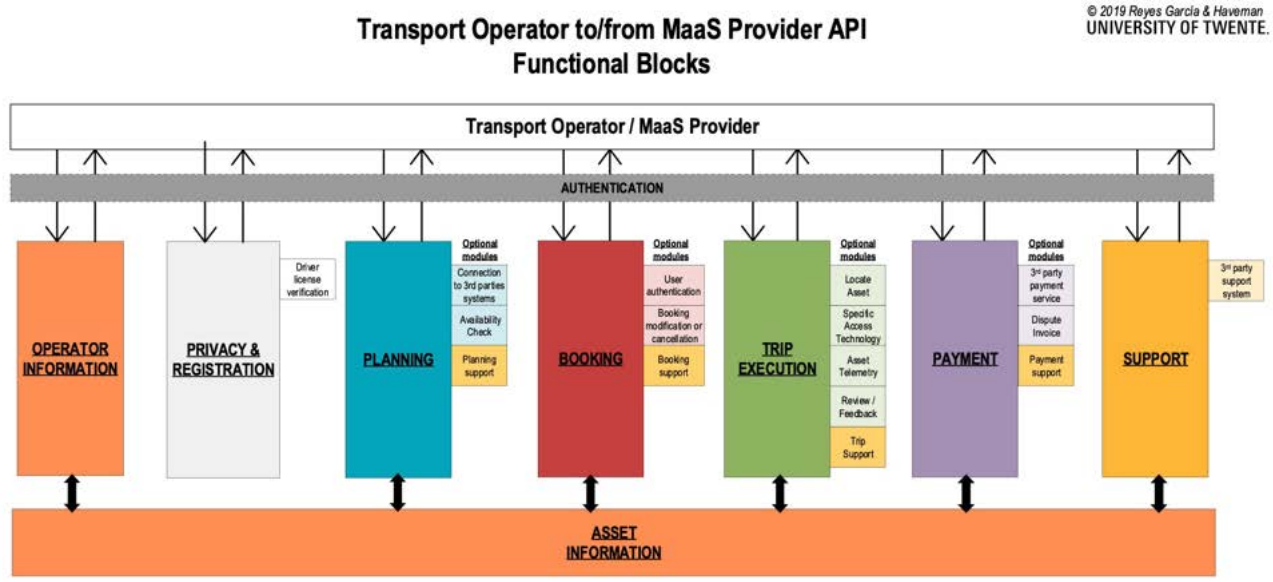


Fig. 2: Functional blocks of the TOMP-API

The different functions for the interface between MPs and TOs are described as follows:

- **Operator Information / General Information**: Gives static information on the operator according to the General Bikeshare Feed Specification+ (GBFS+) standard.
- **Privacy and Registration**: Although the focus of the TOMP-API is not on this block because it impacts not only TOs and MPs but the complete MaaS ecosystem, it is included here as a future point for investigation and possible integration in this API.
- **Planning**: Gives information about availability, estimated travel time and costs.
- **Booking**: Allows reservation of specific assets for a specific place, time and date.
- **Trip Execution**: Allows access to the asset(s) and travel during the booked period.
- **Payment**: Allows settlement between TOs and MPs. Supports different business models (i.e., pay-as-you-go or subscription-based).
- **Support**: Assists users in the solution of operational troubles encountered during any part of the process. Connects with optional support modules.
- **Asset Information**: Is defined as a separate module that can be used by other modules to supplement API calls with specific asset information where applicable. Assets can be vehicles or for example infrastructural assets.
- **Optional modules**: The more dynamic functional blocks have additional optional modules which are used for the execution of sub-processes derived from the main functions which might not be desired or required

depending on the scope of the MaaS implementation and Business Models.

API Authentication

The TOMP-WG is currently exploring different forms of authentication, the final standard has **not** been decided yet.

Fig. 3 below shows that the API features authentication for each call to allow secure communication and exchange of information between MPs and TOs.

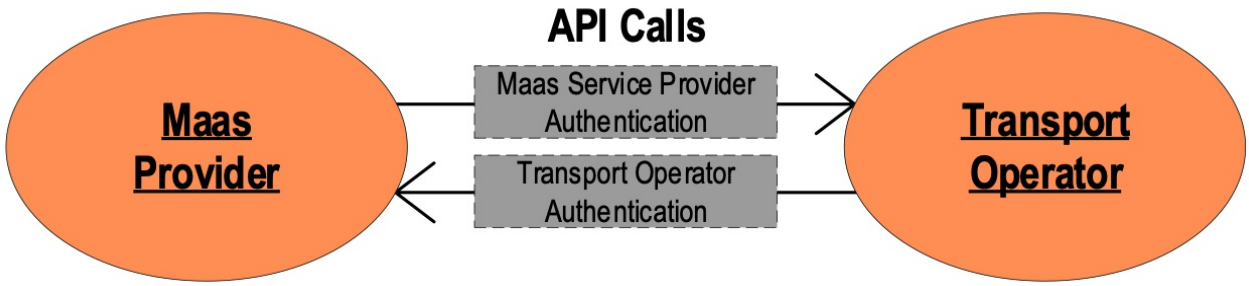


Fig. 3: API calls and authentication

MaaS Provider authentication and authorization should take place following the process below:

- Authorization code – The most common flow, mostly used for server-side and mobile web applications. This flow is similar to how users sign up into a web application using their Facebook or Google account.
- Resource owner password credentials (or just password) – Requires logging in with a username and password. Since in that case the credentials will be a part of the request, this flow is suitable only for trusted clients (for example, official applications released by the API provider).

A Transport Operator might require authentication to communicate with a MaaS Provider,
for example to manage (update/cancel) a booking or to send a call-back request. That makes bidirectional authentication necessary.

Operational Information

As a summary of the data exchanged provided by the TOMP-API calls, Fig. 4 shows an overview of the data sets' blocks and units exchanged between the TOs and MPs.

The operator information endpoint should contain all necessary information about the operator, this information can be summarized in the following categories :

- Regions
- Stations
- Available assets
- Alerts
- Calendar / opening hours
- General information
- Pricing plans, these are extended with TOMP objects (to represent conditions like max fares, scales etc)

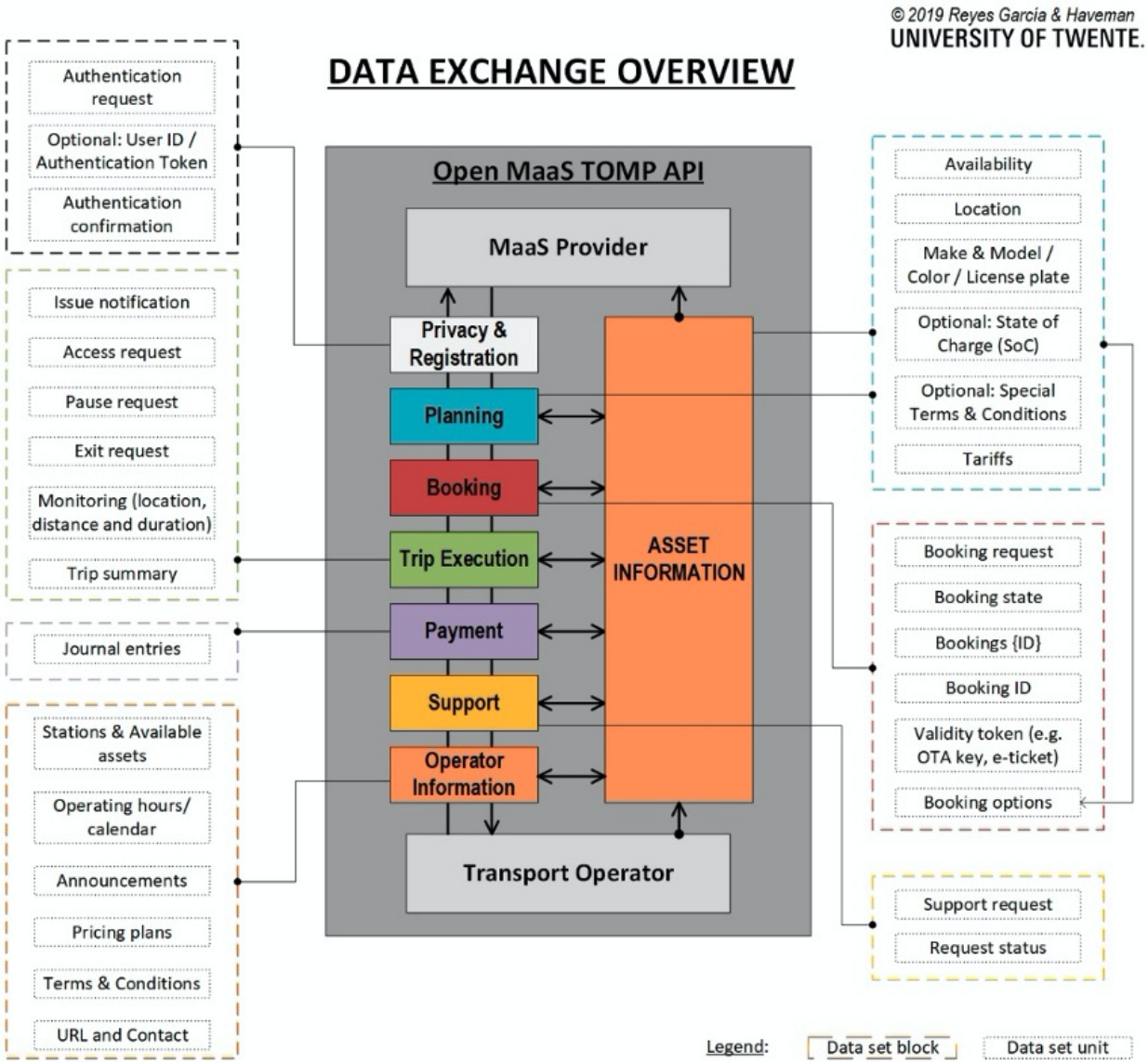


Fig. 4: Data exchange overview of TOMP-API

Privacy and Registration

The first operational block in the TOMP-API is the Privacy & Registration or deregistration block. This block offers the possibility for users to either delete, sign-up or log-in into their account with the MP. The TOMP-API would enable the possibility to use the customer account with a specific TO to log-in into the MP system.

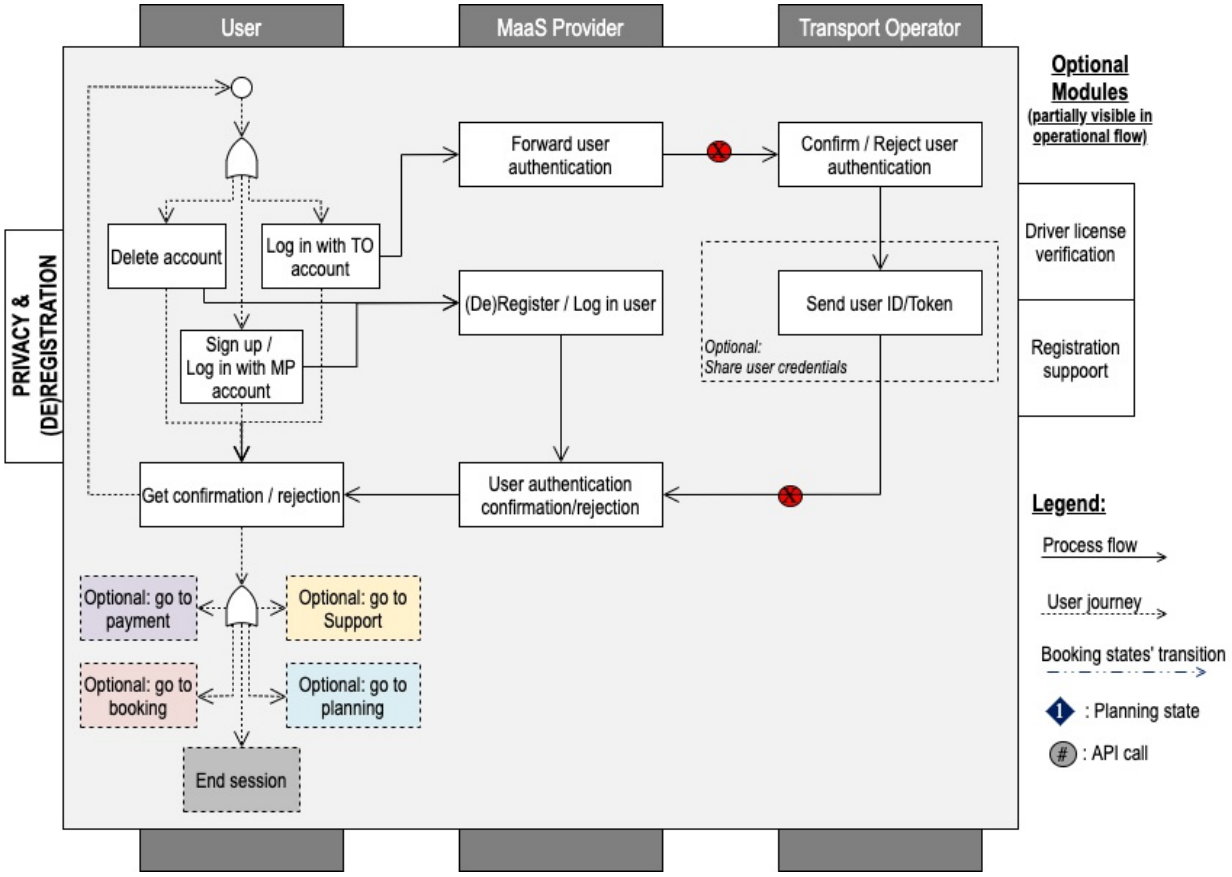


Fig. 5a: Operational view of the Registration & Planning modules

Planning

Planning forms the exploration phase of a trip, where options are explored by the User through the MP. The MP has an archive of (semi-)static general information which is periodically retrieved from the TO. Thus, the MP can check real-time availability of assets to give different travel options to the User. Table 2 presents the functions between the MP and TOs within the planning process, which relate to the user stories presented earlier in §4 and to available API calls from similar API specifications.

Function	User Story; see Appendix A.4	Reference
Update static operator information > provide static operator information	1.2; 1.6; 2.1; 2.2; 2.3; 2.4; 3.4	General Information [from GBFS]
Check availability of trips > Verify availability and temporarily reserve asset	1.1; 1.2; 2.1; 2.2; 2.3; 3.2	Asset availability and competences [from GBFS and amended]

Table 2. Functions between the MaaS Provider and Transport Operator within the planning process

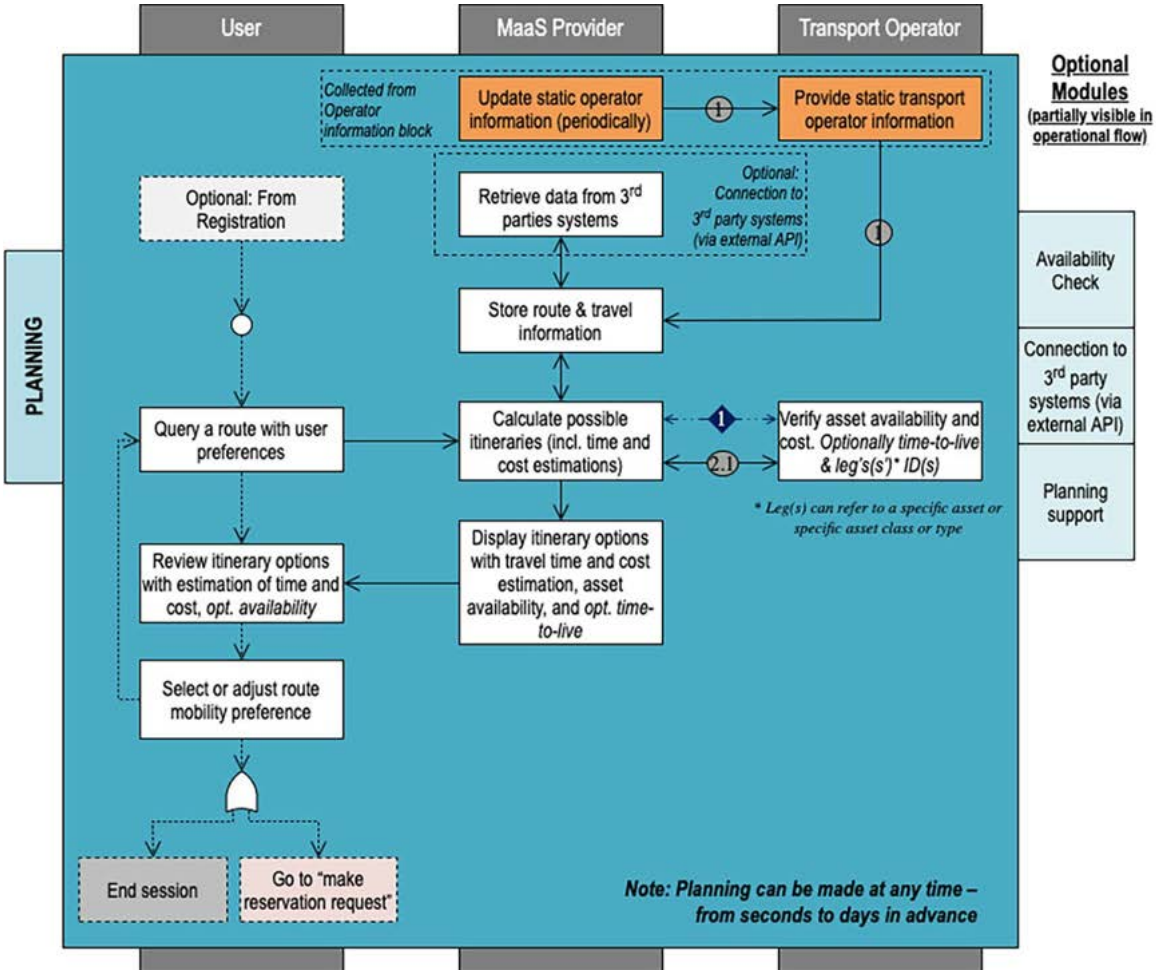


Fig. 5b: Operational view of the Registration & Planning modules

Calls

The planning phase is quite simple with respect to the API: the Transport Operators (TOs) can be asked what assets they have available. This can be done using a request with fields like: from location to location, timestamp to start, timestamp to end, the number of travellers and their (dis)abilities.

The TO should return the available asset types and per asset type the physical assets (if applicable).
Some examples:

- you (TO) have 3 bikes. There is a request for 8 persons in the area you're serving. You can or deliver 3 results (if they are all available) or deliver none. That's up to you.
- you (TO) have an awful lot of cars. There is a request for 5 persons to travel. You can decide to deliver per available car, even if they don't have the capacity to transport 5 persons. You let the MP (MaaS Provider) create the best possible solution for the customer. The only thing you've got to take into account is the (dis)ability list.

The planning comes in 2 flavours: with or without booking intent. If you (MP) are just investigating the options the TO can deliver, use the 'without booking intent' way. When you (MP) have constructed a trip (consisting out of multiple legs), you have to "prereserve" the legs by calling the 'planning with booking intent'. The TO has to respond with a leg they can fulfil (otherwise none), including an ID that can be used throughout the process.

Routing

The MP (MaaS Provider) can (optional!) call this endpoint on the TO's side to ask for the available assets during the routing phase:

```
POST https://exampleTO.com/plannings/?booking-intent=false
{
  "from": { "coordinates": { "lng": 6.169639,"lat": 52.253279} },
  "radius": 100,
  "to": { "coordinates": {"lng": 6.169639,"lat": 52.253279} },
  "startTime": "2020-06-24T07:12:03.000Z",
  "endTime": "2020-06-24T07:20:03.000Z",
  "travellers": 1,
  "users": [ ... ]
}
```

'Users' contains information about specific travellers:

```
"users": [
    {
      "age": 22,
      "licenses": [
        {
          "cardType": "DRIVING LICENSE",
          "country": "NL",
          "assetType": "B"
        }
      ],
      "cards": [
        {
          "cardType": "NS",
          "country": "NL",
          "assetType": "TRAIN"
        }
      ],
      "requirements": []
    }
```

This information can be used to a) give more accurate prices (age), b) return only the assets that are appropriate for the travellers (license-check or (discount) cards). The requirements can be provided per user.

On the other hand, if a TO requires personal information (like driver license) and it's not provided in the planning call, the TO can put in the result a condition: 'conditionRequireBookingData'. The MP should provide the requested data with the booking, otherwise the TO can reject the booking.

The 'planning-options' call can be made to all known TOs. They should respond very quickly on this request. Therefore it's advisable to implement this with simple conditions (like operating area, opening times).
This is an example response. It says that there is one bike available for the requested leg:

```
{
  "conditions": [ ... ],
  "legOptions": [
    {
      "startTime": "2020-06-28T14:55:00+02:00",
      "endTime": "2020-06-28T14:25:00+02:00",
      "from": {
        "coordinates": {
          "lng": 6.169639,
          "lat": 52.253279 }
      },
      "to": {
        "coordinates": {
          "lng": 6.169639,
          "lat": 52.253279 }
      },
      "assetType": {
        "typeId": "bike",
        "name": "bike",
        "travelAbroad": true,
        "assetClass": "bike",
        "brand": "MyBike",
        "cabrio": false,
        "gears": 3,
        "gearbox": "MANUAL",
        "infantSeat": false,
        "persons": 1,
        "propulsion": "MUSCLE",
        "meta": [
          {
            "rating": "4.5"
          }
        ]
      },
      "pricing": {
        "parts": [
          {
            "amount": 9.96,
            "currencyCode": "EUR",
            "taxRate": 21,
            "type": "FLEX",
            "unitType": "HOUR",
            "units": 1
          }
        ]
      },
      "conditions": []
    }
  ]
}
```

If the TO is offering specific assets, the physical asset should be returned. Or if the startTime (or endTime) is in the near future, in that case it would also be wise to give physical assets directly (if applicable, it's not usual to return scheduled assets like busses or trains).
In case for an 'upfront planning' the result normally will be containing 'asset types'. The physical asset can be attached (if needed/wanted) later on in the process.

Conditions

In the response are 'conditions'. These are conditions the TO is requesting. These conditions can like be 'conditionReturnArea', 'conditionPostponedCommit', 'conditionDeposit', … In the results there can be references to these conditions (so they don't need to be repeated).

Fare

We've chosen to add a Fare construction. Instead of just supplying an exact or estimated amount of money, we supply the 'calculation rules'. This can be a fixed price, but also scaled prices.

Just before giving options to the end-user

After the MP has constructed a route, possibly containing multiple legs, the MP must call the TOs once again to provide id's per option. These IDs can be used to communicate between MP and TO:

```
POST https://exampleTO.com/plannings/?bookingIntent=true
{
  "from": { "coordinates": { "lng": 6.169639,"lat": 52.253279} },
  "radius": 100,
  "to": { "coordinates": {"lng": 6.169639,"lat": 52.253279} },
  "startTime": "2020-06-24T07:12:03.000Z",
  "endTime": "2020-06-24T07:20:03.000Z",
  "travellers": 1,
  "users": [ ... ]
}
```

The result will contain id's now:

```
{
  "legOptions": [
    {
      **"id": "746ac-48792bb-746dac3",**
      ...
    }
  ]
}
```

The MP can now give the routes to the end-user. The end-user can simply select a route and the MP can book the legs in the route for the end-user, just by booking a provided trip using it's provided id, but that's in the next phase.

Booking

Booking is the phase where the User will commit to a certain travel option offered by the MaaS Provider (MP). This can be a result of the Planning phase, or in case Users know exactly which ticket or booking they want, the result of a new booking request directly. Table 3 presents the functions between MPs and TOs within this process, which relate to the user stories presented earlier in §4 and to available API calls from similar API specifications.

Category	Function	User Story See Appendix A.4	Reference
Booking	Make booking request > Process booking	1.6; 2.5; 3.1; 3.2	Booking > POST/bookings/
Booking	Provide User Authentication > Request User Authentication	3.3; 3.4; 3.6	Components/securityschemes [from MaaS Alliance API]
Booking	Cancel / Modify Booking	1.5; 2.9; 3.8	Booking > PUT /bookings/{id}

Table 3. Functions between the MaaS Provider and Transport Operator within the booking process

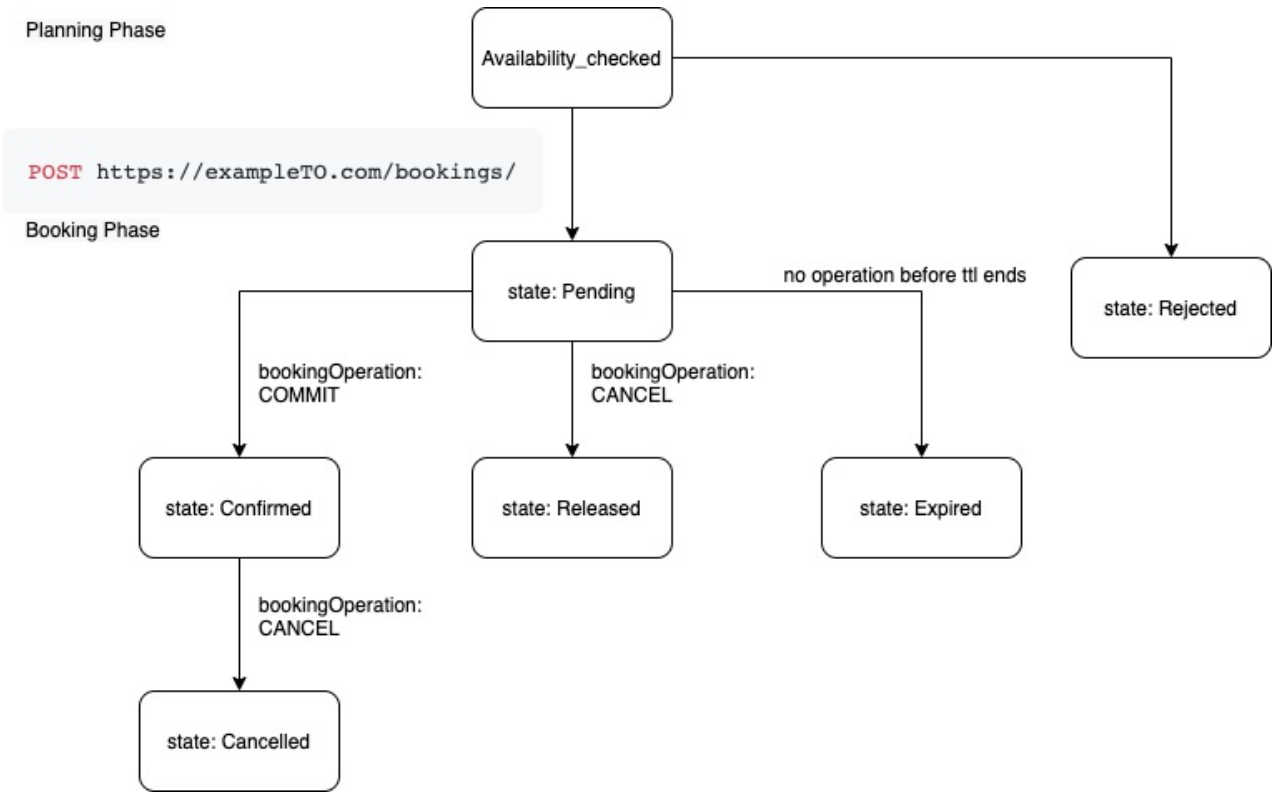
In addition, Table 4 describes the transition states that take place during the Booking process. All these states are helpful to understand the steps and actions within the process of making a reservation. The Booking states are also indicated in the operational flow presented in Fig. 6.

Phase	State	
Planning	Availability check	In the planning phase, a MP can check the real-time **availability of assets** from a TO. In this way, a MP can offer their Users an overview of which assets and options are currently available following the User's request (for a specific mode, a specific location or other User conditions). A time-to-live can optionally be added to the response to show the User how long the information will be valid for. Just before presenting the results to the user add `booking-intent = true` to get booking ids.
Booking	Pending	Once the User has narrowed down their selection, the MP can send a booking request to the TO for a specific asset (or asset type) selection, using the id provided in the previous step. This creates a booking with the state **PENDING** and temporarily 'freezes' an asset while the User is finalizing the selection (i.e., while the User is having to choose multiple options for multiple legs of a journey). A time-to-live in the availability confirmation response is mandatory. This phase is also known as 'Prereserving'.
	Released	If a User decides to go for other options than the one(s) narrowed down, the PENDING state can be cancelled by the MP. The Booking State is changed to **RELEASED**. This frees up the asset for other Users.
	Expired	If the expiry time for the PENDING state is reached (as defined in the time-to-live in the availability confirmation), because the User has not (yet) made a selection, the booking state changes to **EXPIRED** and the corresponding asset(s) are no longer 'frozen' for the specific request and the asset is released for other Users.
	Confirmed	If a User confirms the selection of a given option, the asset (or asset type) is requested from the MP to the TO and the Booking State changes to either CONFIRMED (in case the "authentication" and payment conditions

		are met) or to REJECTED (in case the “authentication” and/or “payment” conditions haven’t been met). In case of ‘POSTPONED-COMMIT’ scenarios the TO cannot confirm immediately and the ‘Conditional-confirmed’ can be replied instead of 'confirmed’.
	Rejected	The conditions for a booking have not been met and the TO rejects the booking that has been conditional-confirmed state. This is the result of a deny-event.
Trip Execution	Started	Once the confirmed asset is in use, the Booking State is changed to **STARTED**.
	Finished	Once the asset is returned, the leg is considered completed and the booking state is changed to **FINISHED**.
Additional States	Cancelled	If the asset confirmation is cancelled by the MP (which could also happen upon request from the User), the Booking State changes to **CANCELLED**, and the corresponding terms and conditions for cancellations between TOs and MPs apply. If the asset confirmation is cancelled by the TO (in case of a broken-down vehicle, late return etc.), the booking state changes to **CANCELLED**, and the corresponding terms and conditions for cancellations between TOs and MPs apply.
	Conditional_Confirmed	Booking state for TOs that cannot immediately confirm a booking (due to e.g. planning issues). The **CONDITIONAL-CONFIRMED** state can be set by the TO to inform that a reservation it’s not yet completely confirmed. Whenever the subcontractor confirms, the booking state will change to CONFIRMED. The **CONDITIONAL_CONFIRMED** state is also limited by a time-to-postponed-commitment, if the time has expired, the booking state will become EXPIRED.

Table 4. Transition states of the Booking process

Booking operations



The first step in making a booking is for the MP to post a call to the */bookings-options* end point to let the TO's know that he's intending to book this asset for the requested leg.

If the TO requested extra data (using the conditionRequireBookingData), this data should also be included in the posted fields, like the customer data in the example below.

```
POST https://exampleTO.com/bookings/
{
  "id": "746ac-48792bb-746dac3", ** this is the ID provided in the planning
phase **
  "customer": {
    "id": 123456,
    "firstName": "John",
    "lastName": "Doe",
    "phone": "tring"
  }
}
```

The example result below is a booking object in the state PENDING:

```
{
  "id": "746ac-48792bb-746dac3",
  "customer": {
    "id": 123456,
    "firstName": "John",
    "lastName": "Doe",
    "phone": "string"
  },
  "state": "PENDING", ** this is a new booking in state pending **
  "terms": "some text",
  "token": {
    "validityDuration": {
      "from": 1546336800,
      "to": 1546336800,
      "meta": []
    }
  },
  "meta": []
}
```

After sending the request to all required TOs - and they all responded with a confirmation, the MP will have to send a POST request with the operation ‘COMMIT’ in it to change the booking-option into a real booking.

```
POST https://exampleTO.com/booking/{id}/events/
{
  "operation": "COMMIT"
}
```

The result will be a booking object, in state COMFIRMED. The webhook in the booking object can be used to communicate from TO to MP about the asset or booked leg.

```
{
  "id": "746ac-48792bb-746dac3",
  "customer": {
    "id": 123456,
    "firstName": "John",
    "lastName": "Doe",
    "phone": "string"
  },
  "state": "CONFIRMED", ** this is a confirmed booking **
  "terms": "some text",
  "token": {
    "validityDuration": {
      "from": 1546336800,
      "to": 1546336800,
      "meta": []
    }
  },
  "meta": []
}
```

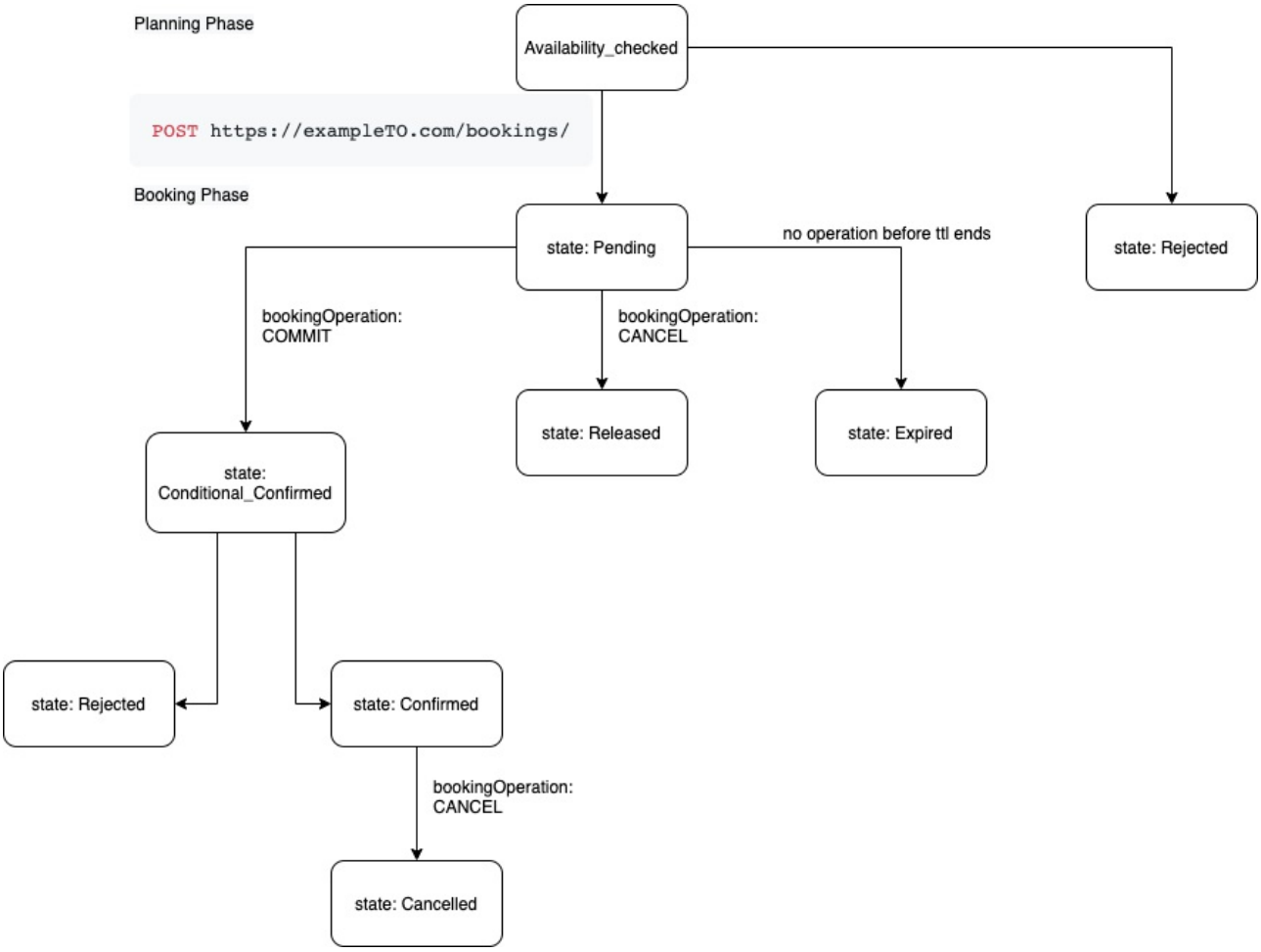
In case of assets that will be opened offline, the ‘token’ should provide the information to open the asset.

So, the route is booked, the user can be informed everything is OK. The trip can be made. Look at the next section: trip execution.

In case the user is changing her/his mind, the booking can be cancelled by sending a cancel-event to https://exampleTO.com/bookings/{id}/events/.

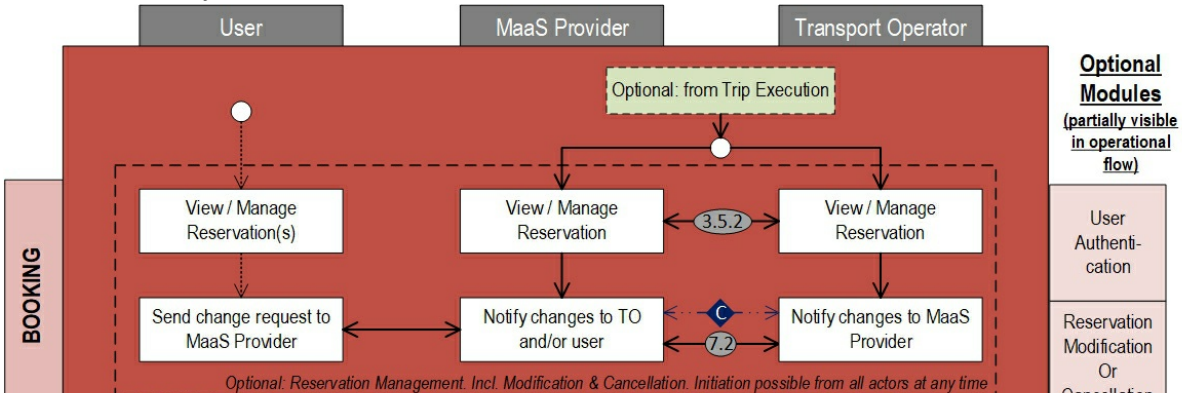
Postponed commit

During the planning phase the TO can send a 'conditionPostponedCommit'. The MP should keep this in mind as the process flow of these legs is quite different. The returned booked leg is in state 'CONDITIONAL_CONFIRMED' and will later on be committed or denied, using the /booking/{id}/events endpoint, when the conditions for the leg have either been met or are rejected.



Booking module overview

For reference only



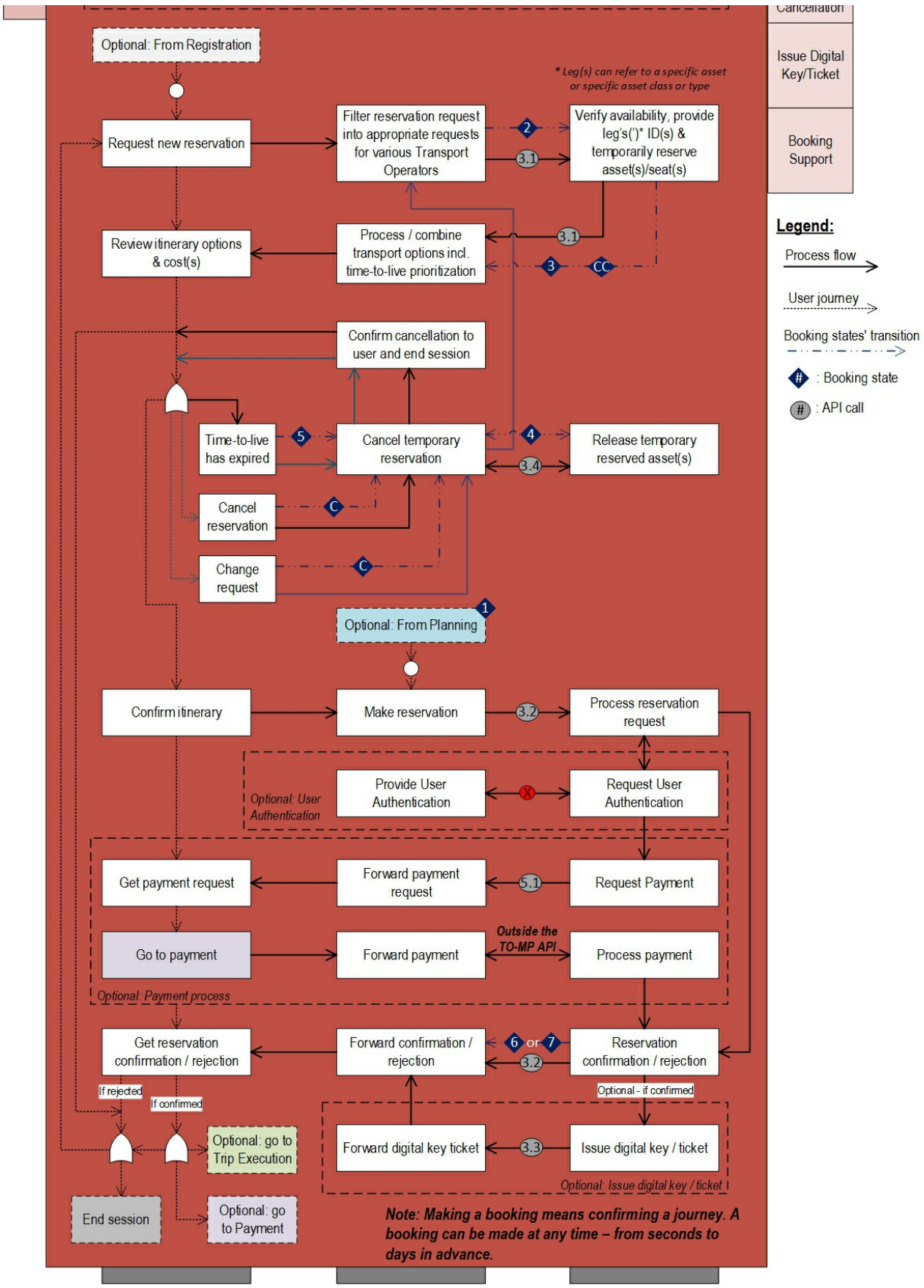


Fig. 6: Operational view of the Booking module

Trip Execution

The Trip Execution module offers all functionalities for the User during the trip. This includes breakdown into different legs, access to the asset, ending a leg and monitoring a trip. When all legs are concluded, summaries of the specific legs are exchanged to offer the User a complete overview of the executed trip. Table 5 presents the functions between the MPs and TOs within this process, which relate to the User stories presented earlier in §4.

Function	User Story (See Appendix A.4)	Reference
Forward location request > provide location	1.1; 2.1	Asset availability and competences > GET /legs/{id}/available-assets
Forward access request > grant / reject access	2.6; 3.6	PUT /legs/{id}/events - PREPARE and SET_IN_USE
Monitor trip <> monitor use of asset	2.7	POST /legs/{id}/progress
Forward exit request > grant / reject exit	2.6	PUT /legs/{id}/events - FINISH
Generate Trip Summary > Provide Leg Summary	1.3	GET /legs/{id}
Manage Review / Feedback <> Review / Feedback with respect to user	1.4; 2.8; 3.5	POST /bookings/{id}/notifications (TO can post this) or GET /payment/journal-entry (not realtime)
Trip support (optional)	2.8; 3.7	POST /support

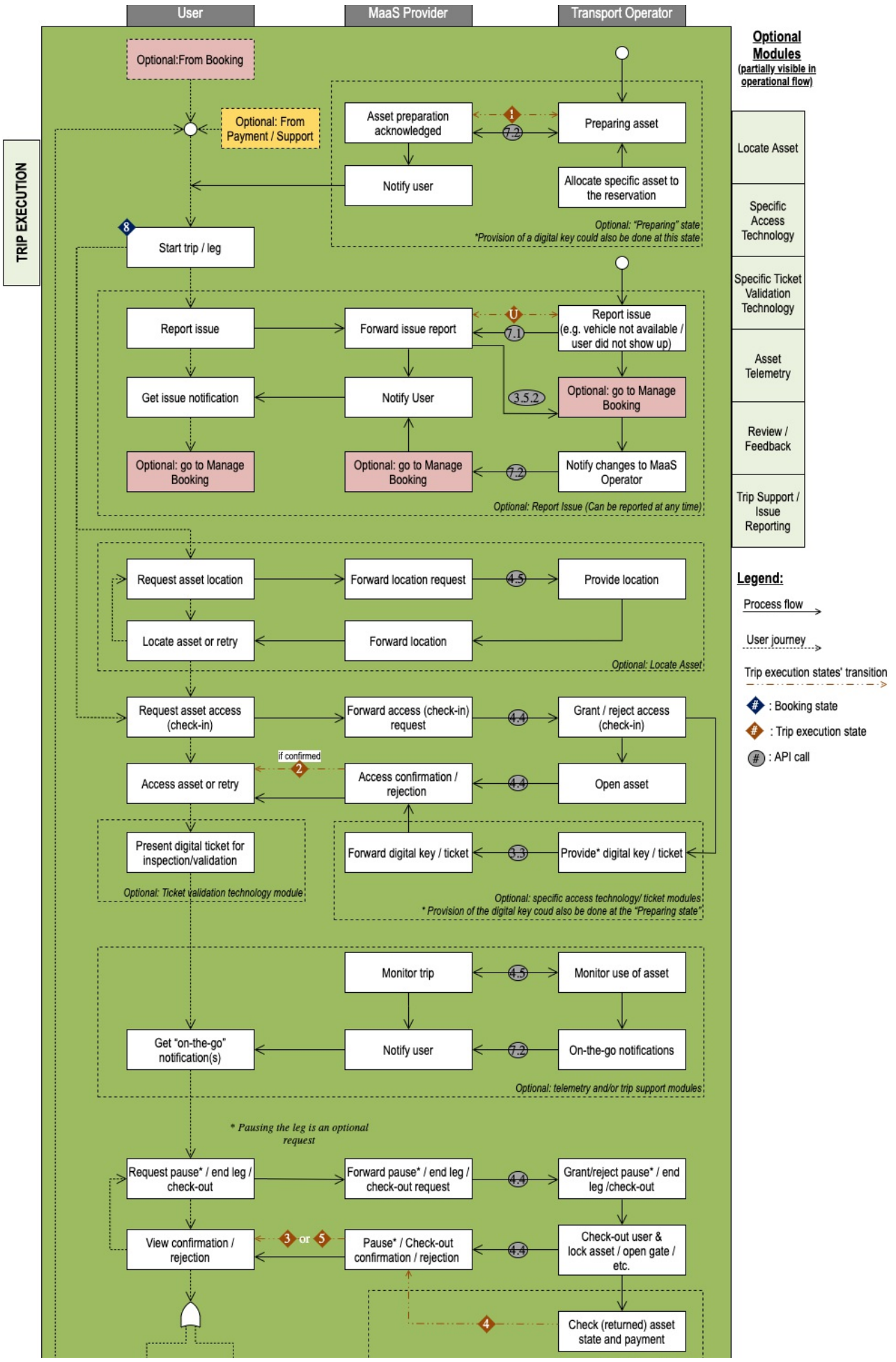
Table 5. Functions between the MaaS Provider and Transport Operators within the Trip Execution process

In addition, Table 6 describes the transition states that take place during the Trip Execution process. All these states are helpful to understand the steps and actions within the process of executing a trip. The Trip Execution states are also indicated in the operational flow presented in **Fig. 7.**

Trip Execution states

#	State	Description
1	Preparing	When an asset is not being used yet by the user, but is being prepared (e.g., a taxi is coming towards the user, or a rental car is being cleaned before start of the rental). Also used to request access information.
2	In use	The user has started to use the asset. This can be acknowledged or confirmed either by the TO or MP, depending on the type of asset.
3	Paused	If possible, an asset that is in use can be paused in order to apply a lower rate (e.g., when parked).
4	Finishing	When the asset is no longer being used by the user, but the Trip execution is not yet finished (e.g., during verification of damages, cleaning of asset, payment check). At this time the user could have continued with another leg of their trip.
5	Finished	The asset has been returned and the trip/leg is confirmed to be finished.
U	Issue	An issue has arisen during the trip execution, reported by the user through the MP to the TO.

Table 6. Transition states of the Trip Execution process


User
MaaS Provider
Transport Operator
TRIP EXECUTION
Optional:From Booking
Optional: From Payment / Support
Asset preparation acknowledged
Preparing asset
Notify user
Allocate specific asset to the reservation
Optional: "Preparing" state
*Provision of a digital key could also be done at this state
Start trip / leg
Report issue
Forward issue report
Report issue (e.g. vehicle not available / user did not show up)
Get issue notification
Notify User
Optional: go to Manage Booking
Optional: go to Manage Booking
Optional: go to Manage Booking
Notify changes to MaaS Operator
Optional: Report Issue (Can be reported at any time)
Request asset location
Forward location request
Provide location
Locate asset or retry
Forward location
Optional: Locate Asset
Request asset access (check-in)
Forward access (check-in) request
Grant / reject access (check-in)
if confirmed
Access asset or retry
Access confirmation / rejection
Open asset
Present digital ticket for inspection/validation
Forward digital key / ticket
Provide* digital key / ticket
Optional: Ticket validation technology module
Optional: specific access technology/ ticket modules
* Provision of the digital key coud also be done at the "Preparing state"
Monitor trip
Monitor use of asset
Get "on-the-go" notification(s)
Notify user
On-the-go notifications
Optional: telemetry and/or trip support modules
* Pausing the leg is an optional request
Request pause* / end leg / check-out
Forward pause* / end leg / check-out request
Grant/reject pause* / end leg /check-out
View confirmation / rejection
or
Pause* / Check-out confirmation / rejection
Check-out user & lock asset / open gate / etc.
Check (returned) asset state and payment
Optional Modules
(partially visible in operational flow)
Locate Asset
Specific Access Technology
Specific Ticket Validation Technology
Asset Telemetry
Review / Feedback
Trip Support / Issue Reporting
Legend:
Process flow
User journey
Trip execution states' transition
: Booking state
: Trip execution state
: API call

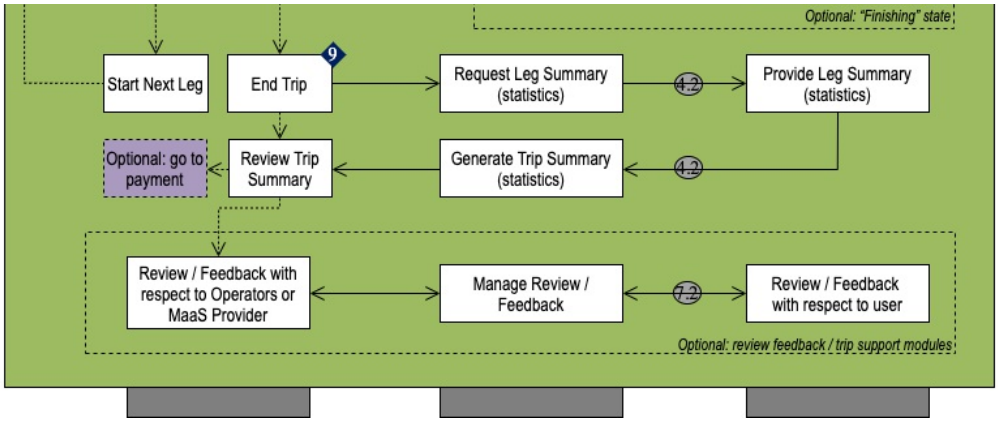


Fig. 7: Operational view of the Trip Execution module

Payment module

The scope of the Payment module is limited to the communication between TOs and MPs concerning settlement and clearing, not about ticketing or the actual payment process. The Payment module offers two alternative payment models that can also be used in conjunction: a prepayment model and a post-payment model. A prepayment model can be used to exchange payment information regarding fares for the legs booked, deposit, subscriptions, etc. A post-payment model can be used to exchange payment information after a trip has been completed regarding fares for the legs travelled, reimbursements, fines, etc. Table 7 presents the functions between MPs and TOs within this process, which relate to the User stories presented earlier in §4.

Currently, the payment module supports only reporting and requesting payments. The TO can enlist all the trip costs and 'other costs', like fines, extra usages etc. The MP can request the 'journal items' to find out how much has to be paid to the TO. In the journal items, there is also a precise description of the executed leg: distance, time etc. All different scenarios (prepaid, postpaid, subscription, deposits, fines, etc) can be implemented with the current setup.

Functions between MaaS-provider and Transport Operator

Category	Function	User Story (See Appendix A.4)
Payment/PrePay	Request / receive payment <> Request / receive payment	1.2; 1.3
Payment/PostPay	Request / receive payment <> Request / receive payment	1.2; 1.3

Table 7. Functions between the MPs and TOs within the Payment process

In addition, Table 8 describes the transition states that take place during the Payment process. These states are helpful to understand the steps and actions within the process of making a reservation. The booking states are also indicated in the operational flow presented in Fig. 8.

Payment states

#	State	Description
T	To invoice	TO requests payment from MP
I	Invoiced	TO has confirmed payment from MP

Table 8. Transition states of the Payment process

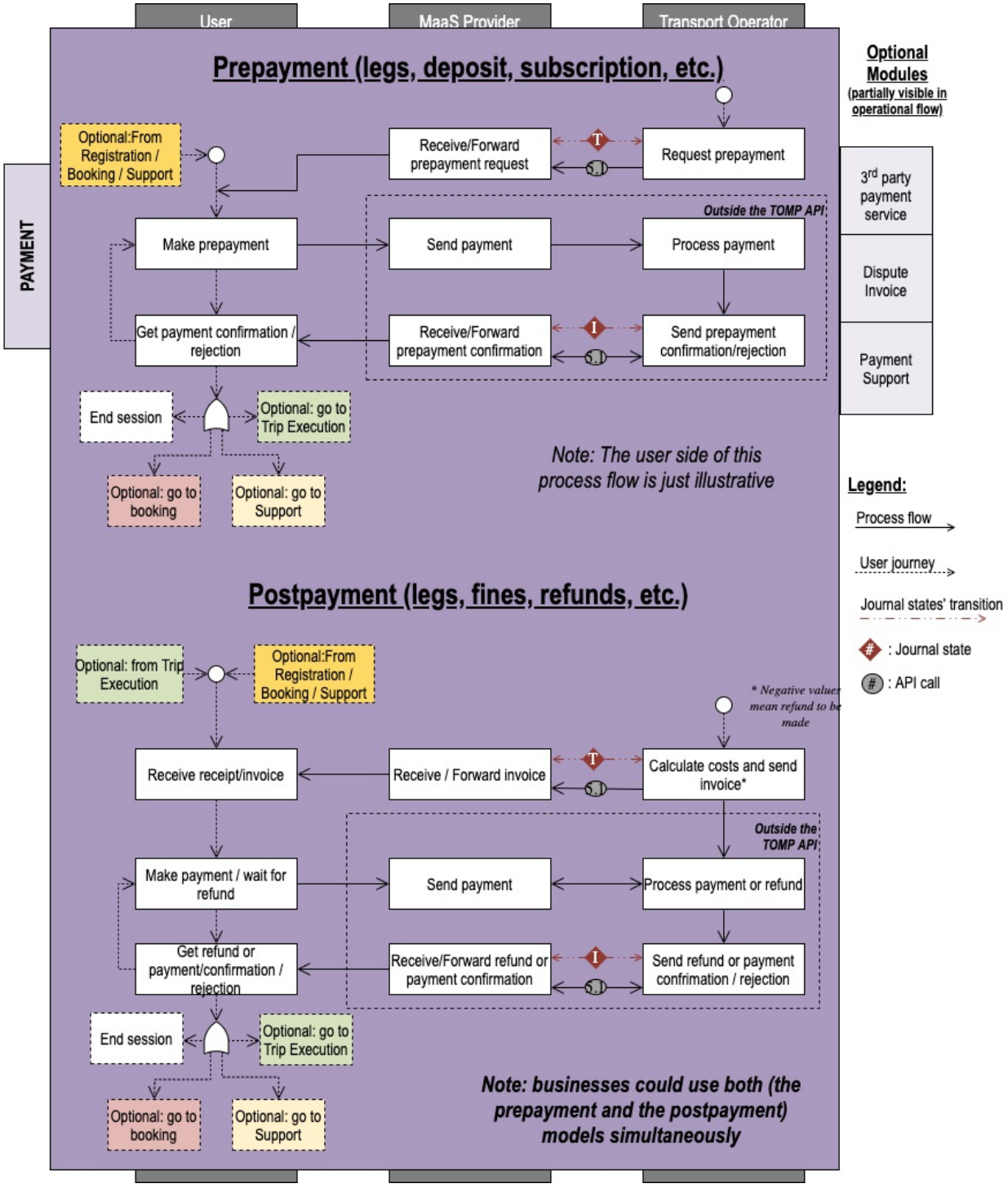


Fig. 8: Operational view of the Payment module

In the near future usable clearing houses will be constructed in the ecosystem (by the market). TOs and MPs can use these clearing houses for frequent clearing of the fares and other payments.

Support

The support module offers functional blocks that refer to the technical assistance to the User in case of an issue experienced during any of the other modules. Within this module, optionally, 3rd party systems could be used to solve the User problems. Fig. 9 shows the process flow of the Support module

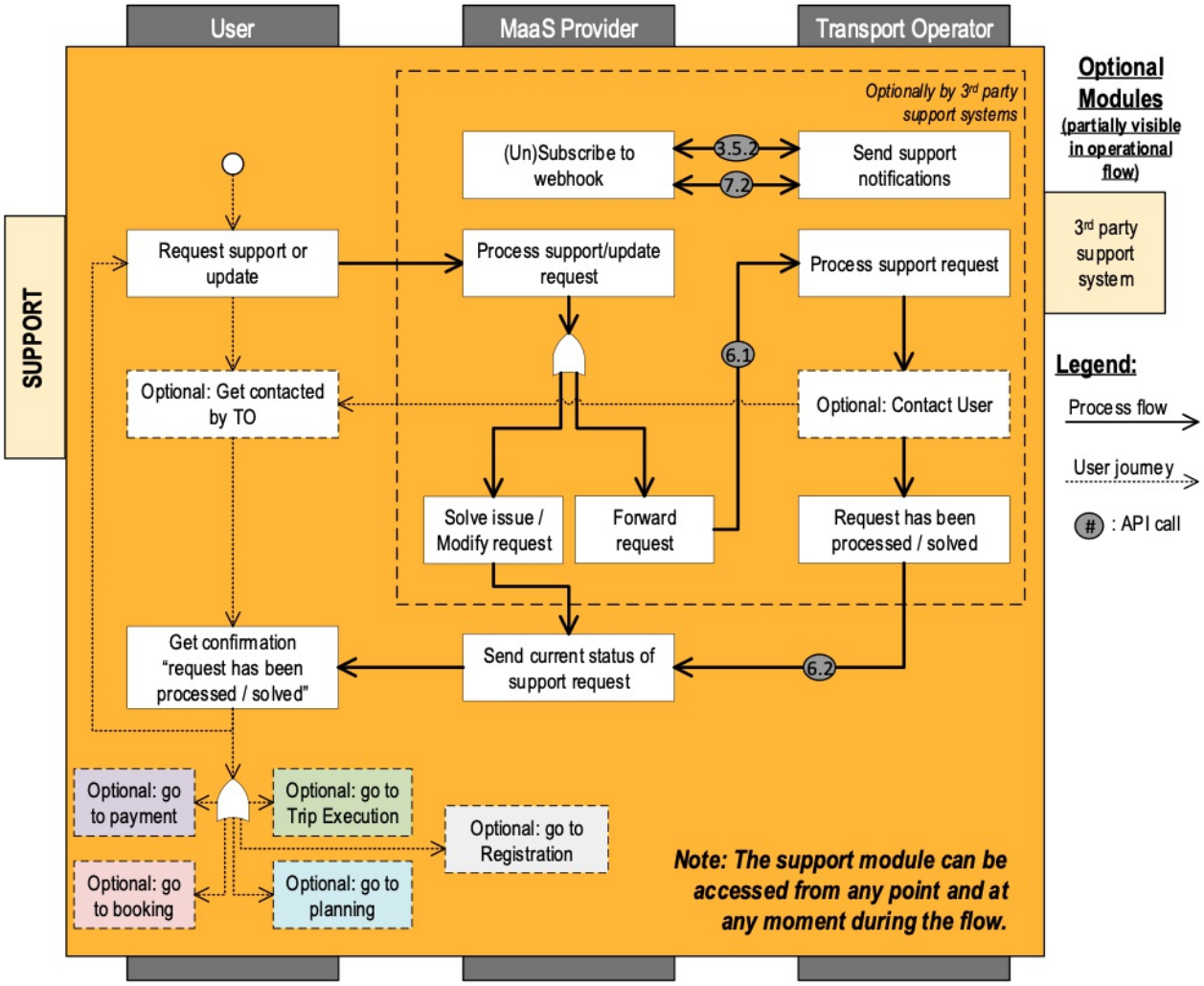


Fig. 9: Operational view of the Support module

Meta Information

The TOMP-API contains a self-describing endpoint (/operator/meta). This endpoint contains information about the endpoints that are implemented and a series of 'process identifiers', describing how the process flow around these endpoints should be. Alternatives like HATEOAS don't cover the required functionality. These Process Identifiers are described in the wiki on Github (https://github.com/TOMP-WG/TOMP-API/wiki/ProcessIdentifiers).

For example, in the planning module, there is the process identifier 'ASSET_BASED', telling that the TO provides nearby assets to end-users in the near future. It supports the pick-up-and-go scenario. The MP cannot reserve assets from this TO as long they don't provide 'PLANNING_BASED' as well.

But in combination with 'ASSET_BASED', the MP must supply a reference to the asset. When the TO requires the identifier 'QR_SCAN', the reference to the asset can be gained by the MP using a QR scan. There are quite a few process identifiers, making it possible to describe the process of the TO. The list of process identifiers on Github is leading, but new identifiers will be introduced in the future, together with new releases of the swagger file.

This is an example response:

```
[
  {
    "version": "0.9.0",
    "baseUrl": "https://dummy-bikes.org/",
    "endpoints": [
      {"method": "POST","path": "/planning-options/","status": "IMPLEMENTED"},
      {"method": "POST","path": "/bookings/","status": "IMPLEMENTED"},
      {"method": "POST","path": "/bookings/{id}/events","status":
"IMPLEMENTED"},
      {"method": "GET","path": "/bookings/{id}","status": "NOT_IMPLEMENTED"},
      {"method": "PUT","path": "/bookings/{id}","status": "NOT_IMPLEMENTED"},
      {"method": "GET","path": "/bookings/?state=","status": "NOT_IMPLEMENTED"},
       .....
     ],
    "scenarios": [
      "POSTPONED_COMMIT"
    ],
    "processIdentifiers": {
      "planning": [ "ASSET_BASED", "QR_SCAN" ],
      "booking": [ "ACCESS_CODE_DEEPLINK", "ACCESS_CODE_IN_COMMIT_EVENT"],
      .....
     ]
   }
  }
]
```

Reference implementations

To facilitate a quick and smooth implementation, several examples, explanations and step-by-step guidelines are provided in the TOMP-API Wiki page (https://github.com/TOMP-WG/TOMP-API/wiki).

As a result, during the past few months, several parties (both TOs and MPs) have started to implement the TOMP-API. These implementations however have been in most cases limited in scope and/or adaptations of the intended standard. On the other hand, none of the implementers has encountered major impediments with the implementation of the TOMP-API.

To have an overview of the latest implementations, the TOMP Working Group conducted an internal survey to explore the level of implementation among members of the group. A summary of the results of this survey is presented in table 9 below:

What	Type	Amount
Type of implementer	Bike Rental	3
	Car Rental	4
	Public Transport	3
	MaaS Provider	4
MP vs TO	Nr. of TOs	7
	Nr. of MPs	4
Type of project	Commercial Pilot	70%
	Reference Implementation	20%
	Other	10%
Multiple TOs behind a single API		40%
Communication	Peer-to-peer	90%
	Router	30%
	Unclear	10%
Security	Custom	70%
	None	30%

Table 9. Implementation survey results.

Technical Specifications

The technical working group suggests implementing this interface using REST-APIs.
Other quality specifications are:

Criteria	Value
Time To Live	Max. 30 seconds
Reliability	95%
API-call max radius around asset	500 meters
API-call min radius around asset	10 meters
Pagination of API-responses	t.b.d. after testing of v. 1.1/1.2
Rate limiting	t.b.d. after testing of v. 1.1/1.2

Table 10. Quality specifications, specific for the TOMP-API

To dos and risks

- Opening and closing of assets can vary greatly between different transport operators. Some regard this technology as their own intellectual property and are not willing to offer external access. This is a risk for common API development and might require further harmonization in the future.
- Which service/helpdesk functions are required for the User?
- Options for ticketing and payment of legs/trips
- Central registry of available stations/hubs/operators and assets, with unique ID’s

A.1 List of terms and definitions

This appendix presents the terms and definitions that served as a reference for the development of the functionalities covered by the TOMP-API.

TERM	DEFINITION	SOURCE
Availability	The ability of an asset to perform a required function under given conditions at a given instant in time, or over a given time interval, assuming that the required external resources are provided.	Adapted from UNISIG (2016)
Booking	The process of making a reservation for space on a means of transport for the movement of people or goods.	Adapted from EC 1305/2014
Booking Process	The process involving those steps necessary to make a reservation, possibly including: - Query of route - Select preferred option - Request reservation - Accept terms and conditions (incl. payment) - Get reservation confirmation	TOMP WG (2019)
Booking State	The situation at a particular time during the booking process.	TOMP WG (2019)
	Started: User requested the usage or reservation of an asset(s) or a seat(s).	TOMP WG (2019)
	Pending: The requested seat(s) or asset(s) is/are temporary reserved for the user. Reservation is pending for payment.	TOMP WG (2019)
	Released: If a User decides to go for other options than the one(s) narrowed down, the PENDING state can be cancelled by the MP. Then the booking state is changed to RELEASED.	TOMP WG (2019)
	Confirmed: Reservation has been paid and the seat(s) or vehicle(s) has/have been granted for the user	TOMP WG (2019)
	Cancelled: The reservations have been cancelled by one of the parties involved	TOMP WG (2019)
	Changed: If a reservation needs to be changed after it has been CONFIRMED by the User or TO (e.g., a different asset has been assigned, different starting time), the MP will indicate it to the other party and the booking state will change to CHANGED.	TOMP WG (2019)
	Finished: Reservation period has ended and the utilization of the asset or seat is no longer valid.	TOMP WG (2019)
(passenger) Journey	A collection of segments which satisfies transportation of a passenger for a given origin and destination.	IATA (2018)
Mass transit	Large-scale public transportation with high carrying capacities, such as buses, subways, and trains.	Byars, M., Wei, A., & Handy, S. (2017)
Motor vehicle	A road vehicle propelled by an engine or motor (internal combustion engine, or electric motor, or some combination of the two) and used for the	

	transportation of passengers, property, or freight	
Multi-modal travel	Travel using more than one travel mode.	
Multimodal access	A system that meets the needs of bicyclists, pedestrians, transit users, passenger vehicles, and other motor vehicle users. A system providing multimodal access integrates different transportation modes to allow co-existence and easy switching between modes	California State Bicycle and Pedestrian Plan in Byars et al. (2017)
Multimodal connectivity	The ease with which people can switch between modes on the same trip. For example, pedestrian and bicycling access to transit stops and stations	Byars et. al (2017)
OSLO	Open Standards of Linking Organisations, a Flanders' initiative to enhance the exchange and traceability of data.	Blumauer, A. (2012)
Passenger vehicle	A motor vehicle with at least four wheels, used for the transport of passengers, and comprising no more than eight seats in addition to the driver's seat.	Organisation Internationale des Constructeurs d'Automobiles (OICA)
Private transportation	Transport services owned and operated by private entities, such as privately-owned shuttles	Adapted from Byars, M., Wei, A., & Handy, S. (2017)
Public transportation	Transport services owned and operated by state, regional, or local public agencies.	
Rebooking	A change of reservation and/or other changes which do not require ticket issuance or exchange	IATA (2018)
Reservation	The allotment in advance of seating or sleeping accommodation for a passenger or of space or weight capacity for baggage, cargo or mail. This term is also applied to hotel, car and other types of travel services.	IATA (2018)
Rideshare	When a driver, or a passenger, shares an open seat(s) in a vehicle with one or more passengers that have similar travel paths and schedules. Traditional forms of ridesharing include carpooling and vanpooling and current use includes sharing space in a ride sourced vehicle.	Byars et. al (2017)
Ride sourcing	A rideshare service that connects passengers to drivers, typically through a digital application and typically for a fee. Drivers and companies work for-profit and typically offer rides that are not incidental to their own trips.	
Shared Mobility	When a transportation mode, such as an automobile or bicycle, is used by more than one person either for moving a person or personal goods. Mode-usage typically occurs at the same time, but may also refer to sequential use, i.e. a leasing a shared bicycle. Although it can reduce miles travelled per person, it may or may not be efficient in terms of the mode used or emissions per person. This includes public transit options, car sharing; personal vehicle sharing (peer-to-peer car-sharing and fractional ownership); car-pooling; van-pooling; ride-splitting, bike-sharing; scooter sharing; shuttle services; micro-transit; ridesharing; e-Hail (taxis); shuttle services; neighbourhood jitneys; ride-sourcing; transportation network companies; ride-hailing; paratransit; and	

	more. It can also include courier network services or flexible goods delivery, which provide for-hire delivery services using an online application or platform (such as a website or smartphone app) to connect couriers using their personal vehicles, bicycles, or scooters with freight (e.g., packages, food), and commercial delivery vehicles providing flexible goods movement.	
Station	Location or facility where air or surface transportation originates, stops and/or terminates, and where passengers and/or cargo can be taken on or off.	
Traffic	The vehicles, pedestrians, ships, or planes moving through an area or along a route.	
Transport	Take or carry (people or goods) from one place to another by means of a vehicle, aircraft, or ship.	Oxford Dictionary
Transportation	The action of transporting someone or something or the process of being transported	
Transit	Public or private transportation service that moves passengers in mass and usually has fixed routes, stops, and fares. Operates within cities or regions rather than between cities or regions.	Byars et. al (2017)
Travel	The action of going from one location to the other, from origin to destination.	
Travel mode	The means by which travel is done. Common travel modes for people include passenger car (driving alone or shared ride), public transit (bus, subway, or train), walking, and bicycling. Common travel modes for freight include land (road, rail, and pipelines), maritime, and air transportation.	
Vehicle sharing	Provides short-term, on-demand access to a transportation mode without sole, direct ownership, thus reducing the overall number of vehicles including automobiles, bicycles, and scooters.	

References

Reference	Description	Source
Blumauer, A (2012)	Linked Open Data: The Essentials – A quick start guide for decision-makers	Retrieved from https://semantic-web.com/2012/01/20/linked-open-data-the-essentials-a-quick-start-guide-for-decision-makers/
Byars, M., Wei, A., & Handy, S. (2017)	Sustainable Transportation Terms: A Glossary	Retrieved from https://itspubs.ucdavis.edu/wp-content/themes/ucdavis/pubs/download_pdf.php?id=2759
EC 1305/2014	COMMISSION REGULATION (EU) No 1305/2014 – Annex II, Glossary	Retrieved from https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32014R1305&from=EN
EC 62/2006	COMMISSION REGULATION (EU) No 1305/2014 – Annex B, Glossary	Retrieved from https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32006R0062&from=EN
IATA (2007)	International Air Transport Association (IATA) — Ticketing Handbook 39th Ed.	Retrieved from https://www.travelready.org/PDF%20Files/IATA%20-%20Ticketing%20Handbook.pdf
IATA (2018)	International Air Transport Association (IATA) — Passenger Glossary of Terms	Retrieved from https://www.iata.org/whatwedo/passenger/Documents/IATA-Passenger-Glossary-of-Terms.xlsx
OICA	OICA statistics web page	Retrieved from http://oica.net/wp-content/uploads/stats-

		definition1.pdf
Oxford Dictionary	Online	https://www.lexico.com. Accessed on 30 July 2019
TOMP WG	Dutch working group for a Transport Operator to MaaS Provider	https://www.linkedin.com/company/tomp-wg
UNISIG (2016)	Glossary of Terms and Definitions - SUBSET-023 v.3.3.0	Retrieved from https://www.era.europa.eu/filebrowser/download/1091982_en

A.2 Passenger characteristics dictionary

This appendix presents the main classification of the corresponding codes for passenger characteristics as defined in the (Dutch) dictionary of passenger characteristics (woordenboek reizigerskenmerken) by the Traffic and Transport Knowledge Platform (CROW-KpVV, 2019). By using these codes, it is possible to clearly establish what are the passengers' needs to successfully complete a (multi-leg) journey. For a full description of the codes and terms please consult the original source at:

https://github.com/efel85/TOMP-API/blob/master/documents/Woordenboek%20Reizigerskenmerken%20CROW.pdf

Category	Code	Name
Passenger's assistance tool	HR-01	Standard wheelchair
	HR-02	Electric wheelchair
	HR-03	Foldable wheelchair
	HR-04	Wheelchair - not securely fixed
	HR-05	Wheelchair - self-balancing two-wheeler
	HR-06	Wheelchair - Different
	HR-07	Rollator – or walker
	HR-08	Rollator - different
	HR-09	Mobility scooter - Standard
	HR-10	Mobility Scooter - Different
	HR-11	Variable assistance tool
	HR-12	Dog
Vehicle tool	HV-01	Belt – exemption seatbelt duty
	HV-02	Belt - extension
	HV-03	Seat - booster seat
	HV-04	Seat - child seat category 1
Additional requirement - transport	AV-01	Transportation – individual
	AV-02	Transportation - in (wheelchair) bus
	AV-03	Transportation - last in / first out
	AV-04	Transport – if passenger car, then in the front
	AV-05	Transport – in passenger car
	AV-06	Transport - in the front and in passenger car

	AV-07	Transport - in the front, regardless of type vehicle
	AV-08	Transport – low entry
	AV-09	Transport - combi with others except …
	AV-10	Transportation - no combi other target group
	AV-11	Transport – shortened travel time
Additional requirement - guidance	AB-01	Guidance - room-room transfer
	AB-02	Guidance - door-to-door
	AB-03	Counselling - necessary / medical counsellor
	AB-04	Guidance - variable companion
Additional requirement - extra passenger	AER-01	Shared travel – for free
	AER-02	Shared travel – reduced fee
	AER-03	Housemate
Characteristics	K-01	Characteristic - blind / visually impaired
	K-02	Characteristic - deaf / poor hearing
	K-03	Characteristic – cognitively limited
(Guided) Independent travel	ZR-01	Public transport advice
	ZR-02	Public transport stop at max 100 meters
	ZR-03	Public transport stop at max 250 meters
	ZR-04	Public transport stop at max 500 meters
	ZR-05	Public transport stop at max 1000 meters
	ZR-06	public transport stop at (variable) meters
	ZR-07	Public transport stop: required - motorically accessible
	ZR-08	Public transport stop: required - non-visual
	ZR-09	Transfers: max 0 times
	ZR-10	Transfers: max 1 times
	ZR-11	Transfers: max 2 times
	ZR-12	Transfers: max 3 times
	ZR-13	Multimodal trip
	ZR-14	Night-blind
	ZR-15	Use of bicycle - partly
	ZR-16	Use of bicycle - fully
	ZR-17	Use of train - partly
	ZR-18	Use of train - fully
	ZR-19	Use of bus - partly
	ZR-20	Use of bus - fully
	ZR-21	Use of own transport - partly
	ZR-22	Use of own transport - fully
	ZR-23	Use of the boarding place/platform
Travel rights	RR-01	Kilometre budget
	RR-02	Mobility budget

	RR-03	Transboundary transport - Valys

A.3 APIs available on the transportation ecosystem

This appendix provides an overview of available commercial and non-commercial APIs on the market.

Name	Website	Service	License
BoMaaS / FLOU.io	https://tapahtumat.tekes.fi/event/bomaas2310 https://app.swaggerhub.com/apis/FLOU	Ticket sales (example) Service registry catalogue	Creative commons 4.0
SUTI	http://www.suti.se/	Exchange of demand-responsive traffic information between clients and providers	Membership
GTFS	General Transit Feed Specification https://developers.google.com/transit/gtfs/	Public transportation schedules and associated geographic information	Google - Apache 2.0
GBFS	General Bikeshare Feed Specification https://github.com/NABSA/gbfs	Bike sharing system, service and status information	Open standard, a community on Github
MaaS-API	http://www.maas-api.org/	Booking and listing	MIT license / Alliance Membership
Uber API	https://developer.uber.com/docs/riders/ride-requests/introduction	Uber ride requests	Developer dashboard membership
IPSI	Interoperable Product Service Interface https://oepnv.eticket-deutschland.de/en/fachpublikationen/themenportal-ipsi/	Mobile ticketing, ticket purchase, conditions for the sale of tickets	License with VDV
Wiener API	http://akirk.github.io/Wiener-Linien-API/	Public transport schedules	Open government data Wien (OGD)
OTP	Open Trip Planner http://www.opentripplanner.org/	Multimodal trip planner Passenger information and transportation network analysis	Open source
OTM	Open Trip Model www.opentripmodel.org	Exchange real-time logistics data	Creative Commons 4.0
TripGo API	https://developer.tripgo.com/ https://developer.tripgo.com/specs/#	Plan door-to-door trips using a large variety of public and private transport. It integrates real-time information and, for selected providers, allows users to book and pay for transport.	Apache License 2.0 Free testing below a threshold of API calls
Combitrip	https://www.combitrip.com/combitrip-api.php	APIs for maps, autocomplete a journey planning.	For small non-

			commercial use it is free for the first 500 daily requests.

A.4 – Overview of the User stories used as parameters for the MaaS functionalities applicable to the TOMP-API

To facilitate the definition of parameters and values that are required for full interoperability in MaaS, user stories have been defined from three different perspectives: the User, Transport Operator (TO) and MaaS Provider (MP).

By using these three perspectives, the chances are increased that all necessary functionalities for MaaS are taken into account. These functionalities can then be related to the necessary interface specifications between the TO and MP. This document does not set up any requirements for the human-machine-interface (HMI) between Users and MPs.

Definitions within the User Stories

Definition	Meaning
API	Application Programming Interface, using REST-APIs as architectural style
User	Customer wanting to make a journey
Maas Provider	Provider of travel advice, information, booking and invoicing
Transport Operator	Owner of (any) transport assets. This can be a bike-sharing or car-sharing platform, public transport operators, taxi companies, ferry operators etc.
Required for MaaS	Yes = mandatoryConditional = mandatory for some operators Optional = mandatory for no operators
Existing API description	Shows which API was used as the basis for implementing this user story. Which are mentioned in the A.3 – APIs available on the transportation ecosystem
User	Competence = is the user ableConditions = is the user compliantAuthentication = confirmation of identity/profile/token

Part 1: From a USER perspective

Definition	Meaning
Item	1.1
Who	As a USER
What	I want to depart from STARTLOCATION and arrive at DESTINATION
Why	To define from where to where I need mobility services for my trip
Required for MaaS	STARTLOCATION = yes, DESTINATION = conditional
Existing API description	GBFSMaaS-API
Comments	Some transport operators require the asset to be brought back to a specific station or zone. This requires knowledge about the desired destination or trip (single, return, multi-leg).

Item	1.2
Who	As a USER
What	I want to know the PRICING of my trip
Why	To determine how expensive my trip will be
Required for MaaS	PRICING = yes
Existing API description	GBFS
Comments	
Item	1.3
Who	As a USER
What	I want to receive a single INVOICE for my entire trip
Why	To simplify my cost overview
Required for MaaS	INVOICE = yes
Existing API description	
Comments	
Item	1.4
Who	As a USER
What	I want to give a RATING and see other ratings of a transport operator
Why	To leave my feedback or determine if I want to use a certain transport operator
Required for MaaS	RATING = optional
Existing API description	
Comments	
Item	1.5
Who	As a USER
What	I want to be able to REPORT an issue
Why	In case the asset I want to use has a problem/damage/issue
Required for MaaS	REPORT = yes
Existing API-description	
Comments	Maybe this doesn't have to be available in an API but needs to be covered by B2B arrangements. A User wants the MaaS Provider to solve any issues, as this is their travel interface. A booking should only be made if an asset has no known technical issues, a transport operator should facilitate this.
Item	1.6

Who	As a USER
What	I want to be able to select an asset-based on COMPETENCES of the vehicle
Why	To fit with the criteria for my trip
Required for MaaS	COMPETENCES = yes
Existing API description	GBFS+
Comments	E.g., selection of the number of seats, type of vehicle, range, fuel type etc.
Item	1.7
Who	As a USER
What	I want to receive SUPPORT during my trip
Why	In case I want to be guided along with my travel, get additional suggestions or need any kind of support.
Required for MaaS	SUPPORT = yes
Existing API description	
Comments	Added in v0.9

Proposals for 1.6	
No of passengers	Propulsion (e.g., hydrogen)
Vehicle class	Exclusive yes/no (in case of ridesharing)
Brand	Type of access/key
Type	Towing hook
Bicycle type (men, women, tandem)	Steering wheel on left or right
Colour	Airconditioning
State of charge (%)	Cabrio
Child's seat	Allowed to travel abroad
Winter tires	Pets allowed
Smoking allowed	Underground parking allowed
Easy accessibility to the location (lift, escalator)	

Part 2: From a MaaS Provider perspective

Definition	Meaning
Item	2.1
Who	As a MAAS PROVIDER
What	I want to know which travel means are available around STARTLOCATION which allow reaching DESTINATION
Why	To give travel advice to the USER
RequiredForMaaS	STARTLOCATION = yes, DESTINATION = conditional

Existing API description	GBFS, MaaS-API
Comments	The destination is not always relevant, but some assets need to be brought back to their specific station or zone or even if a one way trip is possible, to a specific zone or station at destination location
Item	2.2
Who	As a MAAS PROVIDER
What	I want to know if the trip starts at STARTLOCATION and ends at DESTINATIONOr will end at the STARTLOCATION
Why	To define my travel options to the USER
Required for MaaS	STARTLOCATION = yes, DESTINATION = conditional
Existing API description	GBFS, MaaS-API
Comments	Covered by user story 2.1The destination is not always relevant, but some shared bikes need to be brought back to their specific station or zone or even if a one way trip is possible, to a specific zone or station at destination location
Item	2.3
Who	As a MAAS PROVIDER
What	I want to know the ACCEPTABLE DISTANCE for the USER from LOCATION X to STARTLOCATION
Why	To define the travel options to the USER
Required for MaaS	ACCEPTABLE DISTANCE = optional, LOCATION X = optional
Existing API description	GBFS+
Comments	A user can have a preference for the maximum distance he/she wants to walk to reach a bicycle. Proposed standard value = 500 meters
Item	2.4
Who	As a MAAS PROVIDER
What	I want to know the CONDITIONS of a transport operator
Why	To define the travel options to the USER
Required for MaaS	CONDITIONS = yes (but can be periodical)
Existing API description	GBFS, MaaS-API
Comments	E.g., business conditions, user conditions for the rental of the asset etc. These can be updated every week or month (t.b.d.), and do not necessarily have to be requested with each query
Item	2.5
Who	As a MAAS PROVIDER
What	I want to be able to place a BOOKING with a TRANSPORT OPERATOR

Why	To book an asset beforehand
Required for MaaS	BOOKING=conditional
Existing API description	MaaS-API
Comments	This could also be done without a USER requesting a booking. In this case, the booking risk lies with the MAAS PROVIDER instead of the TRANSPORT OPERATOR. In this case, the TO's own clients might not have access to the assets if the MP books everything in advance.
Item	2.6
Who	As a MAAS PROVIDER
What	I want the USER to be able to OPEN/CLOSE/PAUSE the asset through my interface
Why	To make the use of the asset as easy as possible
Required for MaaS	OPEN = conditional, CLOSE = conditional, PAUSE = optional
Existing API description	GBFS+
Comments	Requires information on the locking systems of operators. Pausing is an optional function to allow different pricing models when the asset is temporarily parked by user
Item	2.7
Who	As a MAAS PROVIDER
What	I want to give my USER on-the-fly USAGE INFORMATION about the asset usage and the booking from the TRANSPORT OPERATOR
Why	To avoid having to keep and update all the information myself
Required for MaaS	USAGE INFORMATION = conditional
Existing API description	
Comments	A transport operator could like to send real-time usage instructions (e.g., *'please unlock the bike now using the QR-code'*) to the User through the MaaS-provider interface.
Item	2.8
Who	As a MAAS PROVIDER
What	I want to patch my USER through to the HELPDESK of the TRANSPORT OPERATOR in case of issues
Why	To deliver the best support possible
Required for MaaS	HELPDESK = yes
Existing API description	
Comments	A Transport Operator can give specific support about the asset in case of issues. A direct link between User and Transport Operator is required, the MaaS Provider can facilitate this link through their service. As a reference, insurance companies offer similar assistance, where a neutral helpdesk can take on the role of the insurance provider that manages the specific contract of the User.

Item	2.9
Who	As a MAAS PROVIDER
What	I want to be able to CANCEL/MODIFY a transaction or booking
Why	To inform the TRANSPORT OPERATOR about any changes
Required for MaaS	CANCEL = yes, MODIFY = yes
Existing API description	MaaS-API
Comments	MaaS providers need to be able to cancel or modify transactions or bookings on behalf of their users.

Item	2.10
Who	As a MAAS PROVIDER
What	I want to know if my USER can share a journey or booking with a USER from another MAAS PROVIDER
Why	To efficiently make use of available transportation through carpooling or ridesharing
Required for MaaS	No
Existing API description	
Comments	This allows higher occupancy of available assets through ridesharing and carpooling

Item	2.11
Who	As a MAAS PROVIDER
What	I want to receive information on public transport USERstops and line information
Why	To plan an efficient route for my USER and give the necessary SUPPORT along the journey
RequiredForMaaS	No
Existing API description	
Comments	For planning purposes, e.g., information on kerbs, ramps, lights, displays, line type and transport operator

Part 3: From a Transport Operator perspective

Definition	Meaning
Item	3.1
Who	As a TRANSPORT OPERATOR
What	I want to know from when to when (TIME T1 to TIME T2) the USER wants to use my assets
Why	To define if this fits my offer of assets
Required for MaaS	TIME T1(START TIME/DAY) = conditional, TIME T2(END TIME/DAY) = conditional
Existing API	GBFS, MaaS-API

description	
Comments	This is optional, only required in case of usage restrictions of the Transport Operator or to implement the option to book an asset beforehand (long-term).
Item	3.2
Who	As a TRANSPORT OPERATOR
What	I want to know the DESTINATION of the USER
Why	To determine if my assets are suitable or available
Required for MaaS	DESTINATION = conditional
Existing API description	GBFS, MaaS-API
Comments	The destination is not always relevant, but some shared bikes need to be brought back to their specific station or zone or even if a one way trip is possible, to a specific zone or station at destination location
Item	3.3
Who	As a TRANSPORT OPERATOR
What	I want to know if the USER has the right USER COMPETENCE
Why	To determine if the USER is allowed to use my assets
Required for MaaS	USER COMPETENCE = yes
Existing API description	Not available/necessary in GBFS, use other MaaS-API specs.
Comments	E.g., the user should have a driving license, correct contact details, a membership etc. This could be a liability issue that needs to be covered with insurance providers.
Item	3.4
Who	As a TRANSPORT OPERATOR
What	I want to know if the USER complies with my USER CONDITIONS before starting a trip
Why	To determine if the USER is allowed to use my assets
Required for MaaS	USER CONDITIONS = yes
Existing API description	
Comments	E.g., the user is not on a blacklist, registered member
Item	3.5
Who	As a TRANSPORT OPERATOR
What	I want to give a RATING and see other ratings of a USER
Why	To leave my feedback about and determine if USER can use my asset
Required for MaaS	RATING = optional
Existing API description	
Comments	A transport operator might want to rate a user or determine if a user is allowed to use an

	asset-based on their rating
Item	3.6
Who	As a TRANSPORT OPERATOR
What	I want to be able to receive USER AUTHENTICATION
Why	To confirm the identity of the USER using my asset
Required for MaaS	USER AUTHENTICATION = yes
Existing API description	MaaS-API
Comments	Authentication provides the transport operator with a confirmation of a user's identity, profile or token.
Item	3.7
Who	As a TRANSPORT OPERATOR
What	I want to be able to notify the MaaS provider to CONTACT the USER
Why	In case of problems, emergencies or other issues
Required for MaaS	CONTACT = yes
Existing API description	
Comments	A transport operator can give specific support about the asset in case of issues. A direct link between user and transport operator is required, the MaaS Provider can facilitate this link through their service (see also item 2.8).
Item	3.8
Who	As a TRANSPORT OPERATOR
What	I want to be able to CANCEL/MODIFY a transaction or booking
Why	To inform the MAAS PROVIDER about any changes
RequiredForMaaS	CANCEL = yes, MODIFY = yes
Existing API description	MaaS-API
Comments	Transport operators need to be able to cancel or modify transactions or bookings in case an asset is unavailable or delayed.

A.5 Authors, Architects, collaborators and stakeholders involved

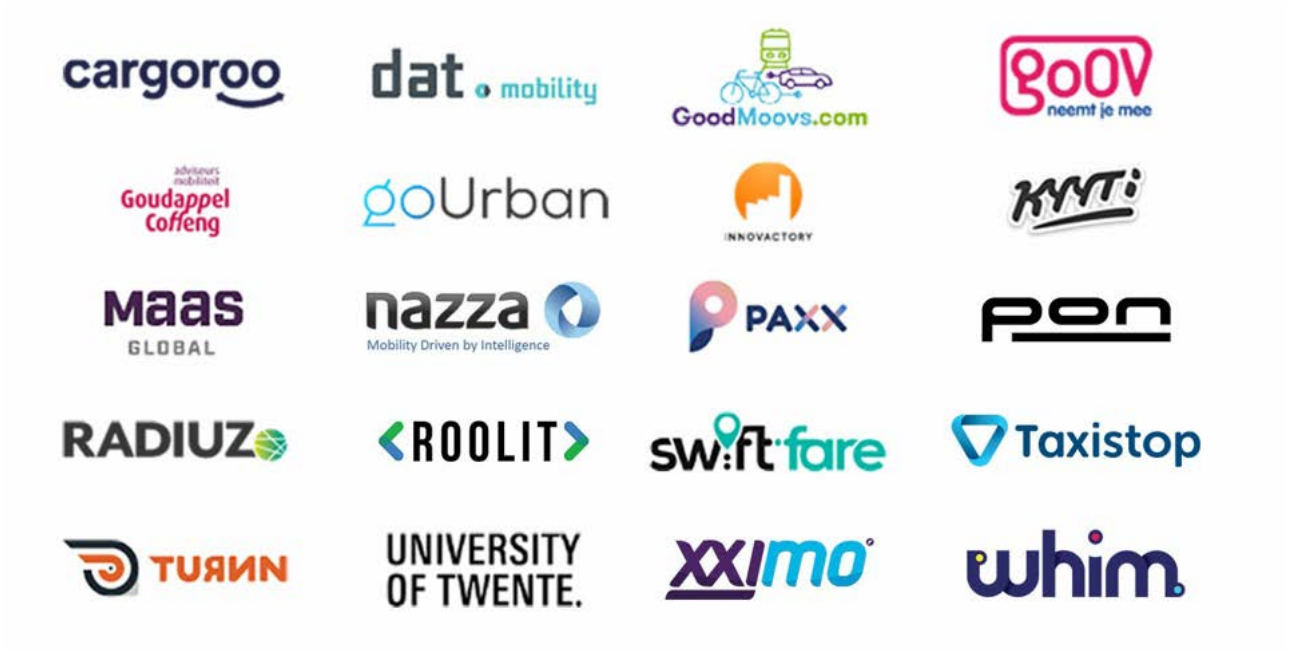
In alphabetical order :

Bon Bakermans – Ministry of Infrastructure and Water Management
Brylie Christopher Oxley – MaaS Global
Christiaan Rakowski – XXImo
Eddy Borremans – Nazza
Edwin van den Belt – DAT.Mobility
J. Roberto Reyes García – University of Twente
Jef Heyse – Radiuz
Jens Kjærby Frandsen – Donkey Republic
Kushagra Sharma – Radiuz
Marijn Roverts – PON
Martijn van der Linden - Mobiliteitsfabriek
Mitchel Smedts – Roolit
Perry Kruin – Emotion
Peter van Brakel – 9292
Pim van der Toolen – TURNN
Rens Wagenbuur – Translink
Robert Baart – Trevvel/Paxx
Ross Curzon-Butler – Cargoroo
Ruud Mollema – Ministry of IenW
Sonila Metushi – MaaS-Lab, a KNV initiatief
Stefan Bollars – Innovactory
Stephan Röösli – SBB/Digital Infrastructure for Mobility
Steven Pauwels – Roolit
Tjalle Groen – Taxistop

A.6 Adoption and Implementation of the TOMP-API

In this section, you will find an overview of some parties and transport operators involved with and/or implementing the TOMP-API.

Working on implementation and/or realization:



Implemented by:

Powered by: