

***Specyfikacja***

# **S-I API Design Guidelines**

**Release 1.1.0**

**Marzec 2025**

### Notice

Dokument ten jest uzupełnieniem dokumentacji TMF a dokładnie dokumentu TMF630 dotyczącej założeń projektowych oraz generalnych zasad dotyczących tworzenia usług zgodnych z REST API TMF. Omawiane są również specyficzne dla wszystkich API konstrukcje.

## 1. General

Reguły jakimi należy się kierować budując komunikację z S-I w ramach API.

|     |  |
|-----|--|
| 100 | Komunikacja między operatorem a S-I jest oparta o operacje na zasobach a nie interakcje w rozumieniu legacy MWDK i MWDP. MWDP ma jednak wpływ na kształt API ograniczając liczbę implementowanych metod HTTP.  |
| 102 | S-I udostępnia jedną postać zasobu opisaną w JSON. Reprezentacja w XML nie będzie dostępna.  |
| 103 | Zasoby są materialnymi bytami zarządzanymi przez S-I. Operator wykonuje na nich operacje <b>insert</b> (POST), <b>update</b> (PATCH), <b>select</b> (GET) tak jakby korzystał z bazy danych. Operacje są atomowe.  |
| 104 | W sytuacji gdy nie istnieje możliwość wykonania operacji <b>insert</b> (POST) na zasobie w krótkim czasie, na tyle by można było wysłać pełną odpowiedź synchroniczną, S-I wysyła komunikat 202 partial creation.  |
| 105 | Wymagalność pól zasobu w kontekście operacji na zasobie wynika z wymagalności zawartej w modelu i wymagalności określonej na poziomie operacji.  |
| 106 | Jeśli dokumentacja nie stanowi inaczej atrybuty zasobów akceptują następujące długości: identyfikatory, id – 50 znaków, pozostałe pola 2048 znaków.  |
| 107 | Pola rozpoczynające się @ są polami technicznymi które umożliwiają zbudowanie generycznego rozwiązania.  |
| 108 | Każde przesłanie obrazu zasobu między stronami czy to w wyniku operacji czy notyfikacji wymaga wykonania operacji json MERGE po stronie odbierającej zgodnie z dokumentem zasad json/merge ( <a href="https://tools.ietf.org/html/rfc7386">https://tools.ietf.org/html/rfc7386</a> ) oraz json/patch ( <a href="http://tools.ietf.org/html/rfc5789">http://tools.ietf.org/html/rfc5789</a> ) |
| 109 | Operator przygotowuje jeden endpoint obsługujący wszystkie notyfikacje wysłane przez S-I dla wszystkich API.   |
| 110 | S-I nie implementuje HUB'a dla notyfikacji. Operator ma obowiązek odbierać wszystkie wysyłane mu notyfikacje zgodnie z danym API.  |
| 111 | Notyfikacje dotyczące zmiany atrybutu informującego o stanie zasobu zawierają zawsze kompletny obraz zasobu jakiego dotyczą.   |

## S-I API Design Guidelines

|     |   |
|-----|---|
| 112 | Operator i S-I mogą w ramach referencji przysyłać sobie kompletne reprezentacje zasobów. Gdy dana operacja wymaga spełnienia jakiś warunków na zasobie tylko atrybuty konieczne do przeprowadzenia walidacji są brane pod uwagę. Dotyczy to również list i relacji. Możliwe i wskazane jest reużywanie obrazów zasobu z różnych API oraz wykorzystanie mechanizmu dziedziczenia atrybutów i zachowania (@type, @baseType).                              |
| 113 | Usługa REST API zawsze zwraca <b>Etag</b> który jest wyznaczony jako funkcja skrótu aktualnego stanu zasobu. W żądaniu PATCH operator musi zamieścić w nagłówku <b>If-Match</b> zawartość pozyskanego wcześniej <b>Etag</b> . Gdy wartość <b>Etag</b> nie zgadza się z zawartością aktualną zasobu operacja aktualizacji nie zostanie wykonana a klient otrzyma informację <b>412 Precondition Failed</b> wraz z reprezentacją stanu aktualnego zasobu. |
| 114 | Wszystkie daty występujące w komunikacji są datami zgodnymi z ISO8601 i zawierają datę i czas lokalny wraz z przesunięciem czasowym względem czasu UTC lub datę w formacie czasu Greenwich. Tam gdzie występują daty bez czasu również musi wystąpić zapis przesunięcia czasu względem czasu UTC lub data w formacie czasu Greenwich.   |
| 115 | Gdy nie jest możliwe wykonanie synchronicznej operacji PATCH na zasobie S-I udostępnia w odpowiednich API asynchroniczne zadanie. Zadanie posiada swój własny stan i notyfikacje. Po zakończeniu zadania np. anulowania zamówienia operator jest powiadamiany odpowiednią notyfikacją dotyczącą zadania.  |
| 116 | Operacje tworzące zasoby zwracają kod następujące kody http: <ul style="list-style-type: none"><li>• 201 (create) i pełną reprezentację zasobu z modelu danych,</li><li>• 202 (partial create) częściową reprezentację zasobu, te pola które S-I mogła zwrócić w synchronicznej zwrotce. Pozostałe pola zasobu są zwracane np. w czasie notyfikowania o zmianie statusu.</li></ul>  |
| 117 | Operacje aktualizujące zasób PATCH zwracają kod HTTP 204 wraz z reprezentacją zasobu. (odstępstwo od znaczenia kodu http – No content).   |
| 118 | Serwery S-I akceptują żądania tylko z określoną jako UTF-8 strona kodową w Content-Type tj. charset=UTF-8. Żądania bez tak określonej strony kodowej nie będą procesowane i zakończą się błędem 415.  |
| 119 | Zwracane listy mają określoną w konfiguracji systemów długość. System zawsze zwraca w nagłówku X-Total-Count ilość rekordów które spełniają wybrane w metodzie GET kryteria.<br>Parametry<br>offset - pozycja początkowa listy  |

## S-I API Design Guidelines

|            |   |
|------------|---|
|            | limit – ilość zwracanych pozycji  |
| <b>120</b> | Konstrukcja wszystkich URI jakich używa system kliencki w komunikacji z systemem S-I powinna być zgodna z dokumentem RFC 3986. URI nie powinien zawierać znaków niedozwolonych, znaki niedozwolone nie powinny być również kodowanie z wykorzystaniem percentage-encoding. Dotyczy to zwłaszcza identyfikatorów o konstrukcji podobnej do konstrukcji numeru faktur. W sytuacji konieczności pobrania zasobu o niedozwolonej konstrukcji należy zastosować mechanizm zapytań QUERY i pole identyfikatora zawrzeć w konstrukcji zapytania. Na przykład: GET ....resource?externalId=10/06/2023/123534. |
| <b>121</b> | Przyjęcie notyfikacji jest potwierdzanie przez system je odbierający kodem http równym 200 lub 201  |

## 2. Resources

Wszystkie zasoby którymi operuję S-I i operator posiadają następujące atrybuty:

|   |  |
|---|--|
| <b>id</b><br><i>optional lub required</i>   | Pole obowiązkowe gdy nie jest wypełnione pole href dla wszystkich zasobów zarządzanych przez SI. Pole to wypełnia SI unikalnym na poziomie zasobu identyfikatorem. |
| <b>href</b><br><i>optional lub required</i> | Pole obowiązkowe gdy nie jest wypełnione pole id dla wszystkich zasobów zarządzanych przez SI. Pole to wypełnia SI unikalnym na poziomie zasobu identyfikatorem.   |
| <b>@type</b><br><i>required</i>             | Pole techniczne, wypełnione wartością wskazującą na typ zasobu znany z API.  |
| <b>@baseType</b><br><i>optional</i>         | Pole techniczne opcjonalne gdy wskazuje na typ bazowy.   |

### 3. Reference resource

Referencje do zasobów API opisane są w postaci w postaci typów z Ref na końcu w nazwie danego typu. Zasoby referencji posiadają dodatkowo następujące atrybuty:

|   |   |
|---|---|
| <b>@referredType</b><br><i>Required</i> | Wymagane pole określające typ docelowy który dana referencja reprezentuje. Id oraz obiektu referencyjnego jest ZAWSZE takie same jak id i obiektu do którego odnosi się referencja. |
|---|---|

W poszczególnych API zasoby referencyjne związane z danym API mogą posiadać dodatkowe pola np.: **role**.

## 4. Event resource

Obiekt zdarzenia jest przekazywany jako notyfikacja wysyłana na adres operatora. Zawiera on następujące pola:

|                                     |  |
|-------------------------------------|--|
| <b>eventId</b><br><i>required</i>   | Unikalny identyfikator emitowanego zdarzenia, jeśli to samo zdarzenie trafia do kilku operatorów ma ten sam id.  |
| <b>eventTime</b><br><i>required</i> | Czas wygenerowania zdarzenia   |
| <b>eventType</b><br><i>required</i> | Typ zdarzenia który pozwala operatorowi zrozumieć na jakim API operuje zdarzenie.  |
| <b>event</b><br><i>required</i>     | Zasób którego dotyczy zdarzenie. Jest to z reguły jego pełna reprezentacja chyba że notyfikacja dotyczy zmiany konkretnych atrybutów wówczas znajdują się tutaj tylko te pola które uległy zmianie. Komplet czy całość nie ma znaczenia bowiem operator zawsze jest zobowiązany do wykonania merge na reprezentacji obiektu po swojej stronie. |

Pola są wypełniane przez S-I w komplecie. Jeśli S-I chce przekazać operatorowi konieczność uzupełnienia obiektu przekazywana jest wówczas notyfikacja o konieczności uzupełnienia informacji **\*InformationRequiredNotification**.

Zawiera ona oprócz powyższych kilka dodatkowych, wymaganych pól opisanych poniżej:

|  |  |
|--|--|
| <b>fieldPath</b><br><i>required</i>    | <p>Pole określające atrybut i związaną z nim akcją akcja=atrybut lub atrybut[liczba], gdy atrybut jest liczbą.</p> <p>Kolejne pola z dotyczącymi ich akcjami są rozdzielone średnikami.</p> <p>Rodzaje akcji wynikają z przebiegu procesu i są opisane w poszczególnych API.</p> |
| <b>resourcePath</b><br><i>required</i> | Pole określa zasób którego dotyczy wymieniona lista akcji, jest to identyfikator zasobu.   |



## 5. relatedEntity attribute/field

Pole to zawiera zawsze listę powiązanych z danym zasobem, obiektem innych obiektów (**refOrValue**) mogących pochodzić z innych z API. Obiekty te są referencjami do innych **obiektów** lub **wartościami** (value).

Obraz zasobu w postaci **Wartości** zawiera wszystkie pola zasobu którego jest wartością z wyjątkiem identyfikatorów tj. **id**. S-I posługuje się przekazywaniem przez **Wartość** tylko w przypadku zasobów którymi nie zarządzają a występują one w danym, opisywanym API.

**UWAGA:** przekazywana między operatorem i S-I lista zawsze powinna zawierać wszystkie obiekty które rzeczywiście są w relacji z danym obiektem i takie relacji istnieją.

## 6. sub-resource versus reference

Bardzo często w API pojawia się możliwość przekazywania informacji w postaci referencji lub wartości (**refOrValue**). Gdy dany atrybut wyszczególniony jest jako **sub-resource** oznacza to że atrybut może być przekazany jako referencja lub wartość.