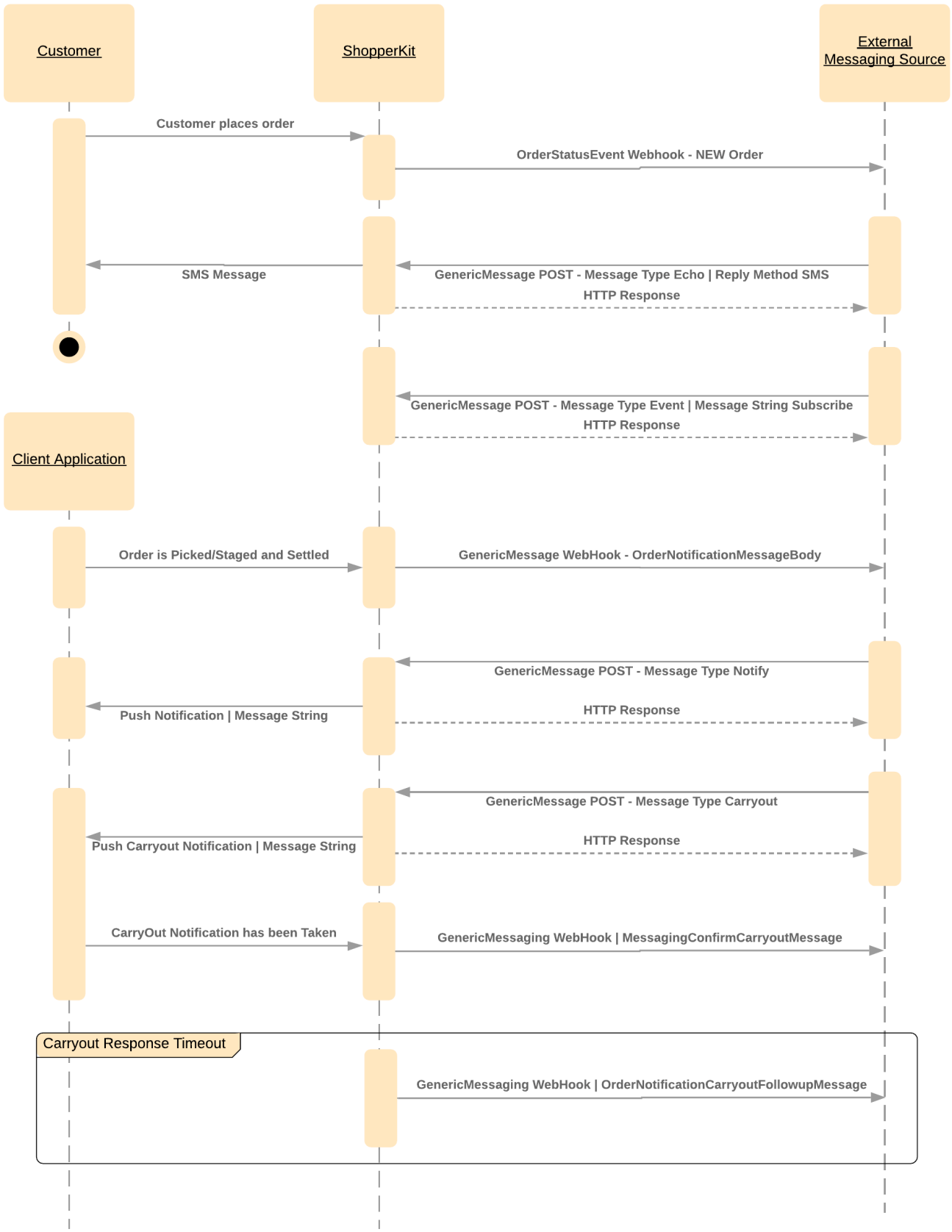
Generic Messaging

Summary:

The Generic Messaging Service provides a means for external providers to utilize ShopperKit's Client application notification functionality. These notifications mirror the existing SMS notification functionality that is used to alert client users that customers are On their Way or they have Arrived, though through this integration the application has been made a little more generic to allow for additional use cases. Like with the SMS functionality, there are 2 distinct notification behaviors. There is the simple notification that can be dismissed by the user which does not trigger any sort of response and there is the Carryout Arrival notification which forces a client user to take ownership of the notification before the notification can be dismissed from any of the tablets. Unlike the SMS functionality which uses canned and templated notification text, the generic messaging assumes that the calling service will be responsible to defining the notification message to be displayed to the user. The generic messaging service also provides a means for the external system to take ownership of any any system generated messages that may be created throughout the fulfillment process. By utilizing the Subscribe or Unsubscribe events, the external messaging system can instruct ShopperKit to broadcast any system messages for a specific order over the Generic Messaging webhook instead of via SMS, which is the default communication mode.

Workflow Overview:



Workflow Example:

Workflow Example

```
1 - New order is placed - OrderStatusEvent webhook
{
 "EventTypeName": "OrderStatusEvent",
 "EventId": "45135149",
 "ClientId": "12",
 "StoreId": "20001",
 "StoreSourcekey" "179",
 "data": {
  "OrderSourcekey": "10001",
  "OrderStatus": "New",
  "PreviousOrderStatus": "",
  "OrderType": "Pickup",
  "PromiseTime": "2019-04-18 11:00:00.0000000 -04:00",
  "CustomerFirstName": "Mary",
  "CustomerLastName": "Poppins",
  "Loyalty": "78756687742".
  "MobilePhone": "4048675309"
 }
}

2 - External messaging system POSTs to Generic Messaging to send SMS on
it's behalf

{
 "MessageId": "C2C60690-2A04-4573-AA0E-85F548769519",
 "MessageType": "echo",
 "ReplyMethod": "sms",
 "ReplyPhone": "4048675309",
 "ClientId": "12",
 "StoreSourceKey": "179",
 "OrderSourceKey": "10001",
 "MessageString": "To follow this order click the link to subscribe
www.subscribe.com"
}

3 - External Messaging system informs ShopperKit that system messages
will route through it vs. SMS.

{
 "MessageId": "B22BF10B-F1F8-4AE8-9F36-95BADFDC28FE",
 "MessageType": "subscribe",
 "ReplyMethod": "http",
 "ReplyPhone": "",
 "ClientId": "12",
 "StoreSourceKey": "179",
 "OrderSourceKey": "10001",
 "MessageString": "Subscribe"
}
```

4 - Order is settled at POS and is now ready for carryout OrderStatusEvent webhook.

```
{
 "EventTypeName": "OrderStatusEvent",
 "EventId": "45135149",
 "ClientId": "12",
 "StoreId": "20001",
 "StoreSourcekey" "179",
 "data": {
  "OrderSourcekey": "10001",
  "OrderStatus": "POS Done",
  "PreviousOrderStatus": "POS Request Recevied",
  "OrderType": "Pickup",
  "PromiseTime": "2019-04-18 11:00:00.0000000 -04:00",
  "CustomerFirstName": "Mary",
  "CustomerLastName": "Poppins",
  "Loyalty": "78756687742".
  "MobilePhone": "4048675309"
 }
}
```

5 - External Messaging system sends message based on geolocation data to Generic Messaging endpoint to have MessageStgring displayed as a notification.

```
{
 "MessageId": "6F9D7475-0321-499A-9F9D-7D15C633DA23",
 "MessageType": "notify",
 "ReplyMethod": "http",
 "ReplyPhone": "",
 "ClientId": "12",
 "StoreSourceKey": "179",
 "OrderSourceKey": "10001",
 "MessageString": "Mary Poppins will arrive in 10 minutes to pickup order 10001"
}
```

6 - External Messaging system sends message based on geolocation data to Generic Messaging endpoint to trigger the carryout notification

```
{
 "MessageId": "AF612B99-8AA5-4CB8-B225-2E50F042B66E",
 "MessageType": "carryout",
 "ReplyMethod": "http",
 "ReplyPhone": "",
 "ClientId": "12",
 "StoreSourceKey": "179",
 "OrderSourceKey": "10001",
```

```
 "MessageString": "Mary Poppins has arrived to pickup order 10001"
}
```

7 - User Takes the carryout notification in ShopperKit

```
{
 "MessageId": "AF612B99-8AA5-4CB8-B225-2E50F042B66E,
 "ErrorMessage": "success",
 "ResponseString": "We're happy that you have arrived. We'll be right
out with your order!",
 "StoreSourcekey": "179",
 "OrderSourcekey": "10001",
}
```

Generic Messaging Connection info:

UAT: https://messaginguat.shopperkit.com/api/v1/generic/receive

Staging: **https://messagingstage.shopperkit.com/api/v1/generic/receive**

Production: **https://messaging.shopperkit.com/api/v1/generic/receive**

Web Method: POST

Required Header Information:

Authorization: fulfiller <API Token> *Your API token will be assigned to you by ShopperKit and will only be valid for a single customer.*
Fulfiller-storeId: <Store ID>
Content-Type: application/json;charset=UTF-8

Generic Messaging Schema:

GenericMessagingPayload

```
{
"$schema": "http://json-schema.org/draft-07/schema#",
"type": "object",
"title": "GenericMessagingPayload",
"properties": {
 "MessageId": {
  "type": "string",
  "description": "Unique identifier for incoming message as assigned by
sender. Value will be sent back in all responses"
 },
 "MessageType": {
  "type": "string",
  "description": "Identifies the type of message and the expected
behavior. carryout will trigger actionable and non-dismisable
notification in app, notify will display a dismissable notification and
event will not directly notify app but will generate a ShopperKit event.
Subscribe will inform ShopperKit that all system generated messages
should be sent via GenericMessaging WebHook and Unsubscribe will set
ShopperKit back into default SMS communication mode.",
  "values": [
   "carryout",
   "notify",
   "event",
   "subscribe",
   "unsubscribe"
  ]
 },
 "ReplyMethod": {
  "type": "string",
```

```
  "description": "Tells ShopperKit how to respond to the incoming
message",
  "values": [
   "http",
   "sms"
  ]
 },
 "ReplyPhone": {
  "anyOf": [
   {"type": "string"},
   {"type": "null"}
  ],
  "description": "Optional value used in conjunction with ReplyMethod
sms. If Null, the mobile phone on file will be used."
 },
 "ClientId": {
  "type": "integer",
  "description": "Unique ShopperKit identifier for Client"
 },
 "StoreSourceKey": {
  "type": "string",
  "description": "Retailers unique identifier for store",
  "examples": [
   "179",
   "106",
   "SKR1"]
 },
 "OrderSourceKey": {
  "type": "string",
  "description": "Unique identifier for the order as assigned by
commerece and generally known by customer and reatiler"
 },
 "MessageString": {
  "type": "string",
  "description": "Contains the body text of the message that will be
displayed on the notifications",
  "examples": [
   "John Doe is 10 minutes away for order 55555",
   "John Doe has arrived to pickup order 55555"
  ]
```

```
  }
}
}
```

Generic Messaging Responses:

As most Messaging responses are based on a user taking action (or inaction) in the client application, and are asynchronous in nature, any system responses that are a direct result of an inbound message will be sent via the Generic Messaging Response webhook. There is a single Generic Messaging Response webhook defined per ShopperKit client that will service all responses for that client. When a Generic Messaging Response if a direct result of an inbound message, the MessageID attribute in the Generic Message Response will reference the Message ID that triggered the response. When the message is generated by ShopperKit, the MessageID will be null.

This webhook URL is defined in Application Attributes section of the ShopperKit Admin tool under the GenericMessagingResponseWebhook Attribute.

Generic Messaging Response Schema

```
{
"$schema": "http://json-schema.org/draft-07/schema#",
"type": "object",
"title": "GenericMessagingResponse",
"properties": {
 "MessageId": {
  "type": "string",
  "description": "References the inbound Message ID that initiated the
response"
 },
 "ErrorMessage": {
  "type": "string",
  "description": "Error information in the event that Inbound message
was found to be invalid",
  "examples:": [
   "success",
   "The referenced ordersourcekey could not be found"
  ]
 },
 "ResponseString": {
  "type": "string",
  "description": "Contains the user defined response text to the various
different pre-defined message types.",
  "examples": [
   "We're happy that you have arrived! We'll be right out with your
order.",
   "We're sorry but all associates are busy, please call 555-810-2222
and let us know that you have arrived."
   ""
  ]
 },
 "StoreSourceKey": {
  "type": "string",
  "description": "Retailers unique identifier for store",
  "examples": [
   "179",
   "106",
   "SKR1"]
 },
 "OrderSourceKey": {
  "type": "string",
  "description": "Unique identifier for the order as assigned by
commerece and generally known by customer and reatiler"
 }
}
}
```

Order Status Events:

In an effort to better equip third party messaging solutions of any interesting and relevant order activity, ShopperKit has expanded the existing store level WebHook to include Order Status Events. Order Status Events are triggered any time an orders status changes, and the WebHook payload includes the relevant attributes to allow an external system to understand the customer, order type and state data for the order. The Order Status Event WebHook is defined at a store level via the Store Attributes. There are several other types of WebHooks that ill go out over this WebHook URL, so it's up the external solution to filter for only those events that are of interest based on the use cases being supported.

OderStatusEvent WebHook Schema:

WebHookSchema - OrderStatusEvent


```
{
"$schema": "http://json-schema.org/draft-07/schema#",
"type": "object",
"title": "OrderStatusUpdateEvent",
"properties": {
 "EventTypeName": {
  "type": "string",
  "description": "Used to distinguish the web hook event",
  "examples": [
   "OrderStatusEvent"
  ]
 },
 "EventId": {
  "type": "string",
  "description": "Unique identifier of event"
 },
 "ClientId": {
  "type": "integer",
  "description": "Unique ShopperKit identifier for Client"
 },
 "StoreId": {
  "type": "string",
  "description": "Retailers unique identifier for store",
  "examples": [
   "179",
   "106",
   "SKR1"]
 },
 "data":{
  "type": "object",
  "description": "Contains data attributes for Order Status Event",
  "properties":{
   "OrderSourceKey": {
    "type": "string",
    "description": "Unique identifier for the order as assigned by
commerece and generally known by customer and reatiler"
   },
   "OrderStatus":{
    "type": "string",
```

```
    "description": "Name value of the current order status",
    "values": [
     "New",
     "In Progress",
     "Picked",
     "Staged",
     "POS In Progress",
     "POS Request Received",
     "POS Done",
     "Loaded",
     "Cancelled",
     "Delivery In Prorgess",
     "Done"
    ]
   },
   "PreviousOrderStatus":{
    "type": "string",
    "description": "Name value of the order status that order was in
prior to changing to current status",
    "values": [
     "New",
     "In Progress",
     "Picked",
     "Staged",
     "POS In Progress",
     "POS Request Received",
     "POS Done",
     "Loaded",
     "Cancelled",
     "Delivery In Progress",
     "Done"
    ]
   },
   "OrderType": {
    "type": "string",
    "description": "Named order type value. Indicates if order is a
pickup or delivery order.",
    "values":[
    "pickup",
    "delivery"]
   },
   "PromiseTime": {
    "type": "string",
    "description": "The timezoneoffset due date for pickup orders or the
beginning of the delivery window for delivery orders",
    "examples": [
     "2019-04-18 11:00:00.0000000 -04:00"
     ]
   },
   "CustomerFirstName": {
```

```
    "type": "string",
    "description": "The first name value of the customer associated to
the order"
   },
   "CustomerLastName": {
    "type": "string",
    "description": "The last name value of the customer associated to
the order"
   },
   "Loyalty": {
    "type": "string",
    "description": "Defined Loyalty number for customer assiociatted
with the order"
   },
   "MobilePhone": {
    "type": "string",
    "description": "Mobile phone number for customer associated with the
order."
   }
  }
```

```
  }
}
}
```