PRD

### Background

Currently, in journeys, event context tags are not resolved in the email subject line, although they work correctly in the email body. This creates an inconsistency in user experience. To address this, event context tags should be supported uniformly across all communication channels and in every applicable input field.

### Scope

* Do a detailed analysis to find out if the tags are supported in all the input fields across all the channels in journeys
* Ensure that event context tags are resolved across all communication channels, including SMS, Email, WhatsApp, and others.
* They should also be supported in all input fields within journeys, such as email subject, images, GIFs, and videos.

#### Design

Here are the mocks in case of beefree and ck editor

<https://www.figma.com/design/k3uJmCDewYsimzKcMeUW5e/Event-context-in-Journeys?node-id=1500-129970&t=oZaoAGngLgXCW4Fo-0>

For new HTML editor :- <https://www.figma.com/design/Cy8YxNkKCXqAjUFpJoSyPI/WYSIWYG-email-HTML-editor?node-id=72-11649&t=bKlpk8sXdseQfcQN-11>

Tech Doc

Tech Doc

#### **Author(s):** @Swasthik J P

#### **Reviewers:** @Nishant Kumar (Unlicensed)@Prashu Chaudhary@Prasenjit Saha

#### **Links:**

#### **Loading app...**

* + [Author(s): @Swasthik J P](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Author(s)%3A--%40Swasthik-J-P)
  + [Reviewers: @Nishant Kumar (Unlicensed) @Prashu Chaudhary @Prasenjit Saha](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Reviewers%3A-%40Nishant-Kumar-(Unlicensed)-%40Prashu-Chaudhary-%40Prasenjit-Saha)
  + [Links:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Links%3A-%5BinlineExtension%5D)
* [Requirement:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Requirement%3A)
* [Current or Existing Solution:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Current-or-Existing-Solution%3A)
* [Suggested or Proposed Solution:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Suggested-or-Proposed-Solution%3A)
* [Module-wise changes:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Module-wise-changes%3A)
* [Adiona:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Adiona%3A)
  + [Introducing tagName and isTaggable in fact and dynamic fact:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Introducing-tagName-and-isTaggable-in-fact-and-dynamic-fact%3A)
  + [Message content in Engagement blocks should store contextBasedOperand map:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Message-content-in-Engagement-blocks-should-store-contextBasedOperand-map%3A)
  + [Executing an Engagement block with context based tags:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Executing-an-Engagement-block-with-context-based-tags%3A)
* [Veyron:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Veyron%3A)
  + [Validation of context based tags:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Validation-of-context-based-tags%3A)
  + [Sending the tags to veneno:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Sending-the-tags-to-veneno%3A)
* [Test Plan:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Test-Plan%3A)
* [Monitoring and Alerting Plan](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Monitoring-and-Alerting-Plan)
* [Async-Automation Testcase:](https://capillarytech.atlassian.net/wiki/spaces/CAM/pages/4272848930/Support+of+Event+Context+Based+Tags+and+Resolution+in+Journeys#Async-Automation-Testcase%3A)

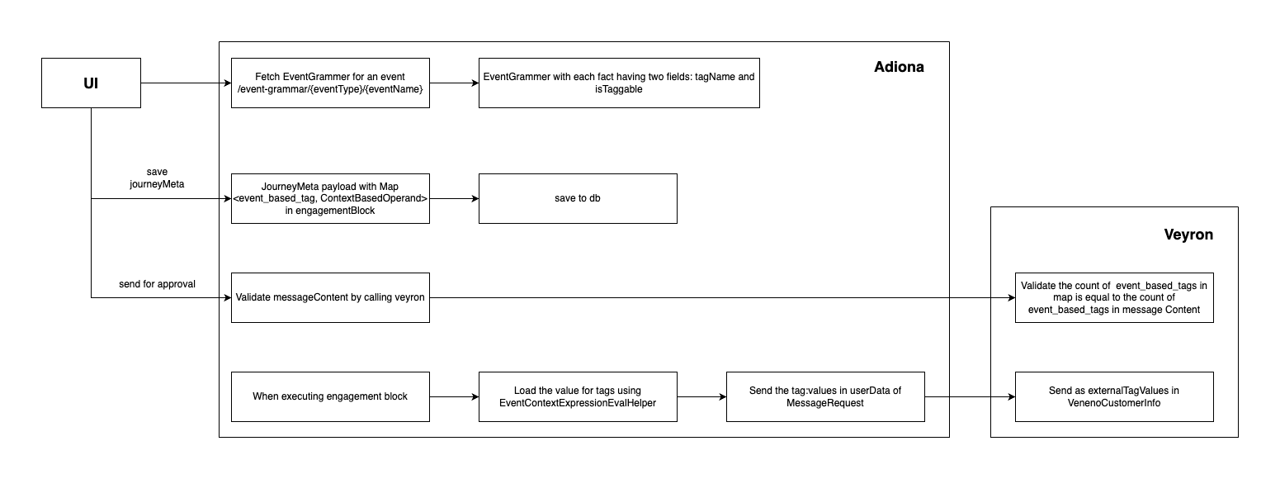
### **Requirement:**

* When an user enters a journey using external event or consumes an event during wait for event or decision block, these events are stored as journey event context
* Allow the use of external event entities like transactionAmount, billId in the message content of engagement blocks
* Resolve the values for these event context based tags and pass to veneno as external tags with both tag and values

### **Current or Existing Solution:**

### **Suggested or Proposed Solution:**

* Facts are used to access the entities of an external event like transaction number, amount in entry block or decision block conditions
* The same facts can be used in message content by mapping a tagName to each fact
* The tag value can be evaluated similar to how conditions are resolved, except here values other than boolean can also be expected
* Add two new fields tagName and isTaggable to Fact in order to map a tagName to each fact
* Using the get grammer api, UI will get all the facts for an event along with its tagName
* The tagName of these facts will be available in the message template creation
* When UI sends the journeyMeta payload for saving to db, it will also send a map of <Tag, ContextBasedOperand> in the messageContent of engagementBlock
* In the send for approval flow, a validation will be done on whether the event contexts specified in the engagement blocks are for the blocks that are present before the engagement block or not. For any event contexts that are part of a block that occurs after engagement block should be thrown as error
* During the execution of engagement block, the contextBasedOperand will be used to fetch the appropriate eventLog from the db using unique Id
* The external event will be loaded using eventLog and the fact for these tag will be resolved using expression evaluator
* The map of tag and value will be passed to veyron using userData in messageRequest
* From veyron, these tag:value will be sent as externalTagValues in VenenoCustomerInfo



Context Based Tags in Journeys

### **Module-wise changes:**

### **Adiona:**

#### **Introducing tagName and isTaggable in fact and dynamic fact:**

public class GFact implements Serializable {

private String id;

private String name;

private String description;

private String exprPrefix;

private List<GParameter> params;

private GReturnType returnType;

private boolean isTaggable;

private String tagName;

}

public class GDynamicFact implements Serializable {

private String method;

private String field;

private String name;

private String description;

private String exprPrefix;

private GReturnType returnType;

private List<String> possibleValues;

private Boolean isTaggable;

private String tagName;

}

public @interface Fact {

String id();

String name();

String description();

boolean isTaggable() default false;

String tagName() default "";

}

public @interface DynamicFact {

String namePrefix();

String exprPrefix();

DynamicFactMetaResolverName metaResolverName();

boolean isTaggable() default false;

String tagNamePrefix() default "";

}

* Introduce tagName and isTaggable in @fact annotation
* tag name for fact will be stored in tagName field of @fact annotation
* Introduce tagNamePrefix and isTaggable in @dynamicfact annotation
* tagName for dynamicFact will be in format of tagNamePrefix(fieldName)
* Suppose the fact method supports DST parameter and if the same fact is used with different parameter value multiple times in same message content, then add the parameter value within brackets ( ) like customField(fieldValue1) to differentiate these tags

isTaggable value will be False by default. It will be enabled for certain facts based on requirement

* To ensure that the tagName is unique for all the facts across all the profiles, a check should be performed during the pod start and store the result. The facts from all the profiles will be fetched to check if any fact has the same tagName or not. When any request comes for fetching grammer info, exception should be thrown if the tagNames are not unique
* A unit test can also be added to check the uniqueness of tagName for all the fact in event-profiles

Dynamic Fact field value is dynamic, it is org and eventType dependent, hence it will be not able to initialise for all the values of field at the start of pod

#### **Message content in Engagement blocks should store contextBasedOperand map:**

public abstract class MessageContent<T extends MessageContentRO> implements BO<T> {

protected List<Offer> offers;

protected Map<String, ContextBasedOperand> contextBasedTags;

}

public class ContextBasedOperand extends ExpressionNode implements Operand, BO<ContextBasedOperandRO> {

private String uniqueId;

private FactOperand factOperand;

private DynamicFactOperand dynamicFactOperand;

}

* contextBasedTags is a map of tagName and contextBasedOperand
* ContextBasedOperand contains uniqueId(blockId\_\_STANDARD\_\_transactionAdd) which maps to the eventContextIdentifierToEventIdMap in journey collection
* ContextBasedOperand can have either a factOperand or dynamicFactOperand

#### **Executing an Engagement block with context based tags:**

* During the execution of engagement block, the contextBasedOperand will be used to fetch the appropriate eventLog from the db for a fact tag
* The external event will be loaded using eventLog and the fact for this tag will be resolved using expression evaluator
* This needs to be done for all the context based tags sequentially for each journey user. If multiple tags in message content requires the same eventLog, then instead of fetching it again and again, we will first fetch all the required eventLogs based on contextBasedTag’s uniqueId and userId. Then use the eventLogs to create external event and resolved each tag
* The map of tag and value will be passed to veyron using userData in messageRequest

### **Veyron:**

#### **Validation of context based tags:**

* During send for approval call, adiona sends the message content to veyron for tag validation
* For each context based tag in message body, the tag should also exist in externalTags set

MessageContent class in veyron used by MessageRequest, MessageMeta

public abstract class MessageContent implements BO<MessageContentRO> {

private Channel channel;

private String creativesId;

private Set<String> externalTags;

* }
* This externalTags field helps in validation of message content, to check whether for all the external tags in message body, relevant tags are present in externalTags field or not. This is meta level validation
* Also a user level validation can be applied by comparing the size of externalTags in meta with the size of userData map in recieverDetails of messageRequest
* This MessageContent class is used in messageRequest, messageBulkRequest, messageMeta, and central comm-systems messageMetaConfig

#### **Sending the tags to veneno:**

* The tag:value will be sent as externalTagValues in VenenoCustomerInfo
* In veneno, these will be resolved using preFetched tagLoader

### **Test Plan:**

1. Use automation to monitor the context based tag resolution
2. Test Cases sheet -
3. Loading app...

### **Monitoring and Alerting Plan**

1. Push appropriate NewRelic stats

### **Async-Automation Testcase:**

* Create a journeyMeta with 3 blocks - entryblock, smsblock and endblock
* Entryblock expects transactionAdd event where transaction value should be between 200 to 499
* Emailblock contains 1 event-context tag - {{entryblock.transaction\_number}}, {{entryblock.transaction\_value}},and 1 normal tag - {{first\_name}}
* When the transaction is performed for a new user, the user should enter the journey and email communication should be sent and then exit the journey
* Automation testcase will assert whether the user completed the journey and whether the final message sent to the user has resolved values for the 3 tags

Documentation

Doc link: Not available