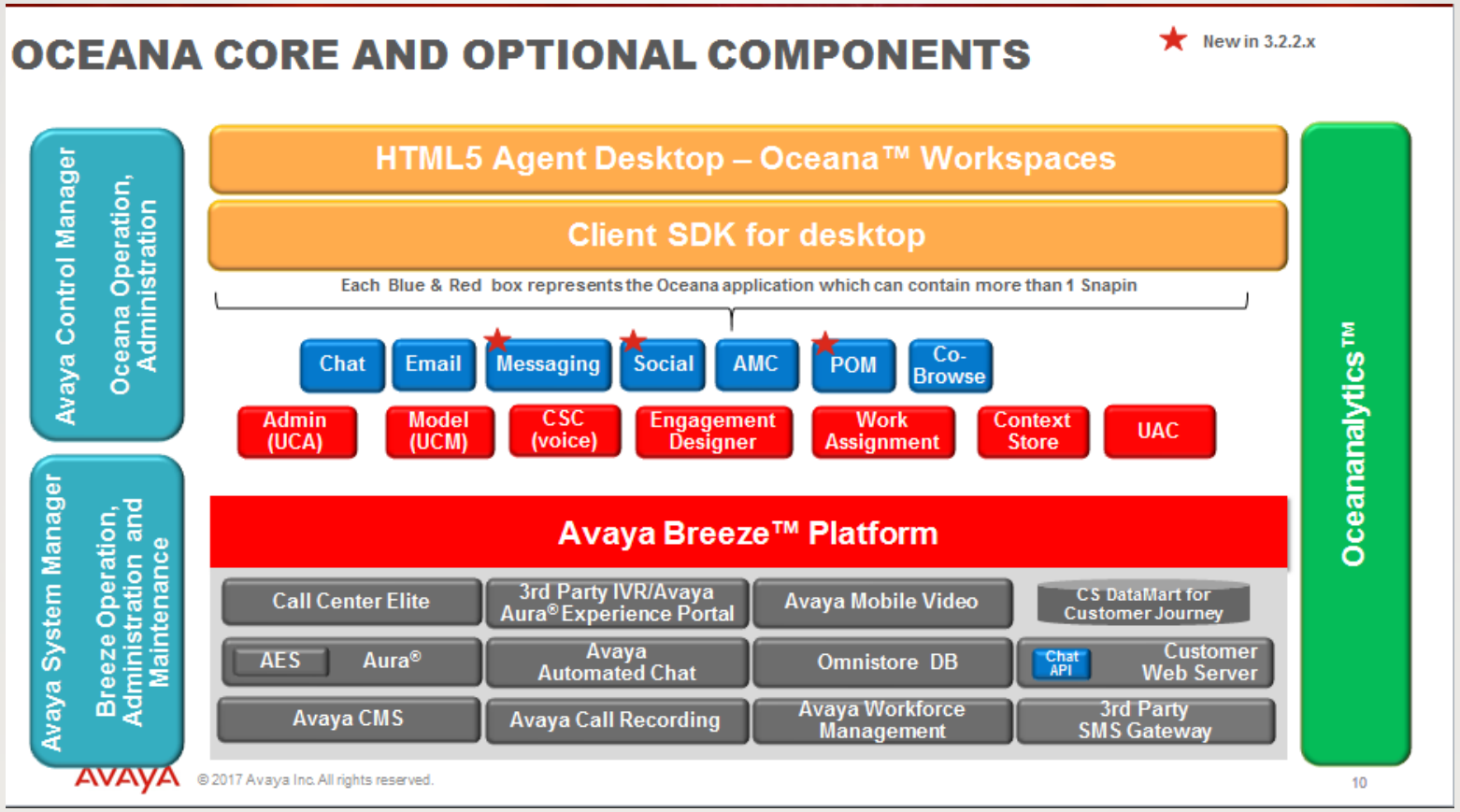
Send Information to Context Store through Oceana

Oceana

- Avaya Oceana is the first delivery of Avaya's next generation multi-touch contact center solution that helps organizations deliver a seamless customer experience across any channel of the customer's choosing. With Oceana, contact centers can seamlessly handle customer interactions in context across all devices and channels – mobile, web, chat, SMS, social media, voice, email and video – supported by strategic business rules and criteria, modern omnichannel agent desktop workspaces (Avaya Oceana™ Workspaces), and, with Avaya Oceanalytics™, cradle to grave reporting of the end to end customer journey across all omnichannel touch points.

Oceana


OCEANA CORE AND OPTIONAL COMPONENTS
New in 3.2.2.x
Avaya Control Manager
Oceana Operation, Administration
Avaya System Manager
Breeze Operation, Administration and Maintenance
HTML5 Agent Desktop – Oceana™ Workspaces
Client SDK for desktop
Each Blue & Red box represents the Oceana application which can contain more than 1 Snapin
Chat
Email
Messaging
Social
AMC
POM
Co-Browse
Admin (UCA)
Model (UCM)
CSC (voice)
Engagement Designer
Work Assignment
Context Store
UAC
Avaya Breeze™ Platform
Call Center Elite
3rd Party IVR/Avaya Aura® Experience Portal
Avaya Mobile Video
CS DataMart for Customer Journey
AES
Aura®
Avaya Automated Chat
Omnistore DB
Chat API
Customer Web Server
Avaya CMS
Avaya Call Recording
Avaya Workforce Management
3rd Party SMS Gateway
Oceananalytics™
AVAYA
© 2017 Avaya Inc. All rights reserved.
10

Oceana

- Questions:

- https://confluence.forge.avaya.com/display/JOULE/Oceana+3.x+Knowledge+Base+Library

Oceana Context Store

- Avaya Oceana enables organizations to map the customer journey across self-service and assisted service channels by storing data during the journey in the context store in-memory data grid. Data may consist of customer information either historical or provided in real-time, enterprise data, as well as situational / environmental data (e.g. device using, location, etc.). This customer journey data can be used by the workflows, resource matching engine, as well as agents and staff to make smarter real-time decisions on behalf of the customer and business.
- This information is temporary.

Context Store Request Contents

Context Store Request.json

Oceana Service Maps

- When making a transfer to an agent it may occur that the calls need to be routed to an agent with specific skills. That's the reason why service maps exist.
- Service Maps contain attribute information on what agents can take the call.
- The Service Map attributes are compared against the agent attributes in order to get an agent with the specific skills.

How to Use It

- There are to ways to write information to Context Store. One is hitting that service directly and the other is through Oceana. Keep in mind that the Service Maps are used by Oceana when routing the call, so the first method won't work with Service Maps.
- In both cases, we have two ways of accomplishing this. By consuming the REST API, these services have published (this is equivalent to a WS call like when we access a customer back), or by using an OD plugin (DO NOT USE THIS METHOD).

Using the REST API through Oceana

- All the URLs use HTTPS, with SSL.
- There is no Authorization.
- Requests and Responses are JSON.
- What do you need?
  - The URL of the cluster where the Context Store is.
  - The information to build the service maps, which attributes are you going to pass.
  - The custom information you want to pass to the agent.

The POST Request

- The URL will be http://<IP or DNS>/services/OceanaCoreDataService/oceana/data/context/schema
- With Header: [Content-Type: application/json]
- You can optionally send [lease=number of seconds] as a parameter, it Will be the duration of the registry in the ContextStore DB.
- And body:

Context Store Request.json

The Response

- If no contextId was specified in the request, the response will be a contextId.
- The Body:

Context Store Response.json

Example Code

Post to Context.java

GetCustomerID.java

The GET Customer Id

- Before calling the above method, we need a customerId to pass on the Request body.
- This is obtained by a different API call, to another service, the getCustomerId.
- It's a REST GET call to http://<IP or DNS>/ services/CustomerManagement/rest/customers/customerId?phone=<caller ani>
- The IP/DNS is the same as before, the Cluster IP or DNS.
- We use the caller ANI as parameter.

The GET Customer Id

- The response is a Json that contains the customerId.
- We will always get a customerId, even with new caller. All customerIds are valid to be used in the ContextStore Request.
- The Body is:

Get Customer Id Response.json

Questions?