How to sync SLA information from a Zendesk ticket to a Jira Issue?

A service level agreement, or SLA policy, is an agreed upon measure of the response and resolution times that your support team delivers to your customers.

For example, you can define an agreement where you respond to urgent tickets in 10 minutes and want to resolve them within a couple of hours.

In this post, we will be integrating SLA information from Zendesk with a plugin named Time to SLA which is available on Jira and makes tracking SLA's easy. In this use-case we will be Starting/Pausing/Ending SLA' based on statuses. It's also possible to do it with custom date fields and even comments. SLA's can even be differentiated by priority or narrowing down the scope using JQL functions.

Apart from this we will also be syncing over the below mentioned SLA metrics to a custom field in Jira :

- Latest Comment added at
- Reply Time
- First Resolution Time
- Full Resolution Time
- Agent Wait Time
- Requester Wait Time
- On Hold Time
- Solved At

THE SCRIPTS

Zendesk Outgoing Sync :

Zendesk Outgoing Sync

```
def response = httpClient.get("/api/v2/tickets/${ticket.id}/metrics")

def latest_comment_added_at = ("${response.ticket_metric.latest_comment_added_at}")
replica."latest_comment_added_at" = latest_comment_added_at

def reply_time_in_minutes = ("${response.ticket_metric.reply_time_in_minutes}")
replica."reply_time_in_minutes" = reply_time_in_minutes

def first_resolution_time_in_minutes = ("${response.ticket_metric.first_resolution_time_in_minutes}")
replica."first_resolution_time_in_minutes" = first_resolution_time_in_minutes

def full_resolution_time_in_minutes = ("${response.ticket_metric.full_resolution_time_in_minutes}")
replica."full_resolution_time_in_minutes" = full_resolution_time_in_minutes

def agent_wait_time_in_minutes = ("${response.ticket_metric.agent_wait_time_in_minutes}")
replica."agent_wait_time_in_minutes" = agent_wait_time_in_minutes

def requester_wait_time_in_minutes = ("${response.ticket_metric.requester_wait_time_in_minutes}")
replica."requester_wait_time_in_minutes" = requester_wait_time_in_minutes

def on_hold_time_in_minutes = ("${response.ticket_metric.on_hold_time_in_minutes}")
replica."on_hold_time_in_minutes" = on_hold_time_in_minutes

def solved_at = ("${response.ticket_metric.solved_at}")
replica."solved_at" = solved_at

replica.status       = issue.status
```

Note : We are using the endpoint provided by Zendesk to make a Rest API call using httpClient.

Jira Incoming Sync :

Jira Incoming Sync

```
if(replica.status.name == "solved")
{
def p = ""
p+= "Ticket ID : " + replica.key + "\n" + "Latest Comment Added At : "+replica.latest_comment_added_at.values
[0] + "\n" + "On Hold Time in Minutes : "+replica.on_hold_time_in_minutes.values[0].calendar + "\n" + "Agent
Wait time in Minutes : "+replica.agent_wait_time_in_minutes.values[0].calendar + "\n" + "Solved At : "+replica.
solved_at.values[0] + "\n" + "Requester wait time in minutes : " + replica.requester_wait_time_in_minutes.values
[0].calendar/60000 + "\n" + "First Resolution time in minutes : "+replica.first_resolution_time_in_minutes.
values[0].calendar/60000 + ""
issue.customFields."SLA Info".value = p
}


def statusMapping = ["new":"Open", "open":"Open","pending":"Pending","hold":"Work in Progress","solved":"Done"]
def remoteStatusName = replica.status.name
issue.setStatus(statusMapping[remoteStatusName] ?: remoteStatusName)
```

Note : Based on the status changes on Zendesk, we are starting/pausing/stopping SLA's configured on Jira (Time2SLA Pluign).

A full video highlighting the same is attached below!