Peter Simard

Abstract

This is an evaluation of the concept and potential design for the refresh mechanism

Oneocean technical review

Dynamic Refresh

Contents

[Request 2](#_Toc72556469)

[Concepts 2](#_Toc72556470)

[Birds Eye / High Level concepts 2](#_Toc72556471)

[System 2](#_Toc72556472)

[Sequence Scenarios 3](#_Toc72556473)

[Conclusion 5](#_Toc72556474)

# Request

Integrate a callback operation between the WEB Application and the Service that provides information for the web application to view.

# Concepts

The aspect of communication between two HTTP based systems will utilize TCP communication capabilities that provide the ability for two way DUPLEX communication. WEB Sockets communication was designed explicitly for this capability. While it is possible to use web sockets directly it may be more convenient to utilize a more generalized mechanism to perform this. One of the APIs provided by Microsoft is SignalR and has been suggested in the request documentation as a possibility. I will use SignalR as I am familiar with it and have used it multiple times and the project already has it integrated.

# Birds Eye / High Level concepts

## System



The general concept behind SignalR is somewhat a publish subscribe configuration (even though two way communication is active – but the server cannot receive responses from calls to the client). A client will subscribe to a SignalR proxy. Once subscribed the client will receive notifications from the client. This will occur on exposed operations on the client that the SignalR service will call. This requires that the client exposes the associated operation that the SignalR service can call.

# Sequence Scenarios



The client will make a simple call to the SignalR service to subscribe to the service. This is a simple operation that makes the request and a is successful response is returned. In general the connection can stay active but it is usually good to perform a call back to the service to keep the connection alive.



Operation that should be called on a regular basis to make sure that if connectivity is lost that it is re established.



On new data signaled will log the request within a queue and have processing thread handle request. This will keep the processing of updates to client from interfering with data processing activity and via versa.



The SignalR service will call back to clients (client instance of WEB App) and inform of new data. The Web App will query back to the Service application accessing controller to retrieve new data.

# Conclusion

This is the general idea that I have. Using SignalR as the basis for the publish subscribe mechanism. SignalR is an advanced API that utilizes the technology available. It will use web sockets if available – if not it will use other mechanisms to configure the communication. The web app client is responsible for making / subscribing to the service and it is a good idea to have a continual connectivity check – especially for browser specific interfacing as browsers can stop communicating on many different conditions.