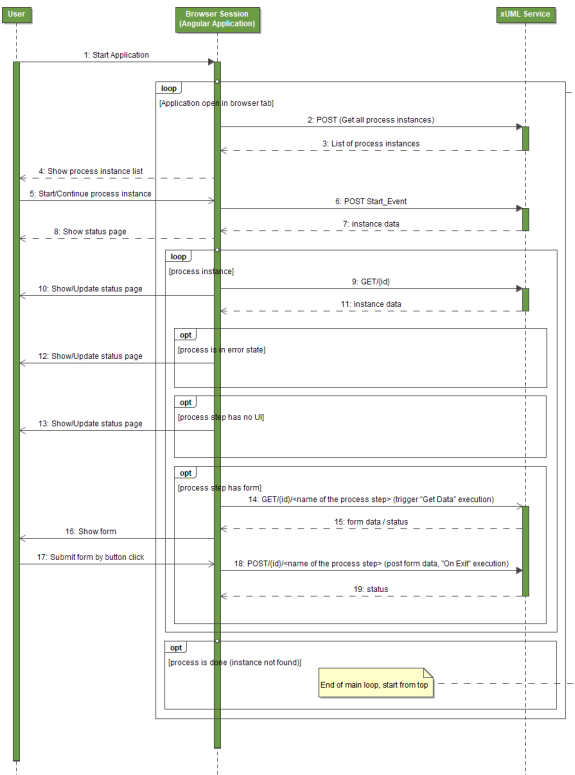# Form Communication

For xUML services that contain one or multiple forms, the Designer generates an Angular application and deploys it as a part of the service repository. Once the application has been started, that app runs in the web browser, and can communicate with the service via the service's REST interface.

The following sequence diagram shows the communication between the user, the browser session (Angular app), and the xUML service in a sequence diagram:

Sequence Number	Event
1	The user starts the Angular application in their web browser.
loop	
2	To build the instance list, the browser requests a list of all running process instances from the xUML service via a **POST/** request to the service API.
3	The service returns a list of all running process instances.
4	The browser displays the instance list.
5	The user starts a new process instance.
6	The browsers sends a **POST** request to the service API to trigger the start event of the process.
7	The service creates a new process instance and returns it to the browser.
8	The browser displays a status page.
loop	
9	In a loop that is running as long as the process instance is active, the browser requests the status of the instance with a **GET/{<instance id>}** in regular intervals.
10	The service returns the instance data, or HTTP 404. Depending on what the service returns different processing is done.
opt	The process instance is in error state.

		12	The browser updates the status page to display the current state to the user. Looping over the process instance, requesting the status and displaying it to the user will be done until either the error is resolved or the process instance is not available anymore.
	opt		The active process step is no UI task.
		13	The browser updates the status page to display the current state to the user.
	opt		The active process step is a UI task and should display a form.
		14	The browser triggers the **Get Data** execution of the UI task by a **GET/{<instance id>} /<name of the process step>** if there is one.
		15	The service returns the form data.
		16	The browser displays the form.
		17	The user enters data and submits the form.
		18	The browser sends the entered data to the service API by a **POST** request. This also triggers the **On Exit** execution of the process step.
		19	The service returns a status message.
End of the instance loop			
opt			If the **GET/{<instance id>}** does not return a valid process instance (e.g. because the process instance has reached its end), the instance loop finishes.
End of the main loop			