# PROCE55® Mobile: Example Mobile App – SOAP based Integration

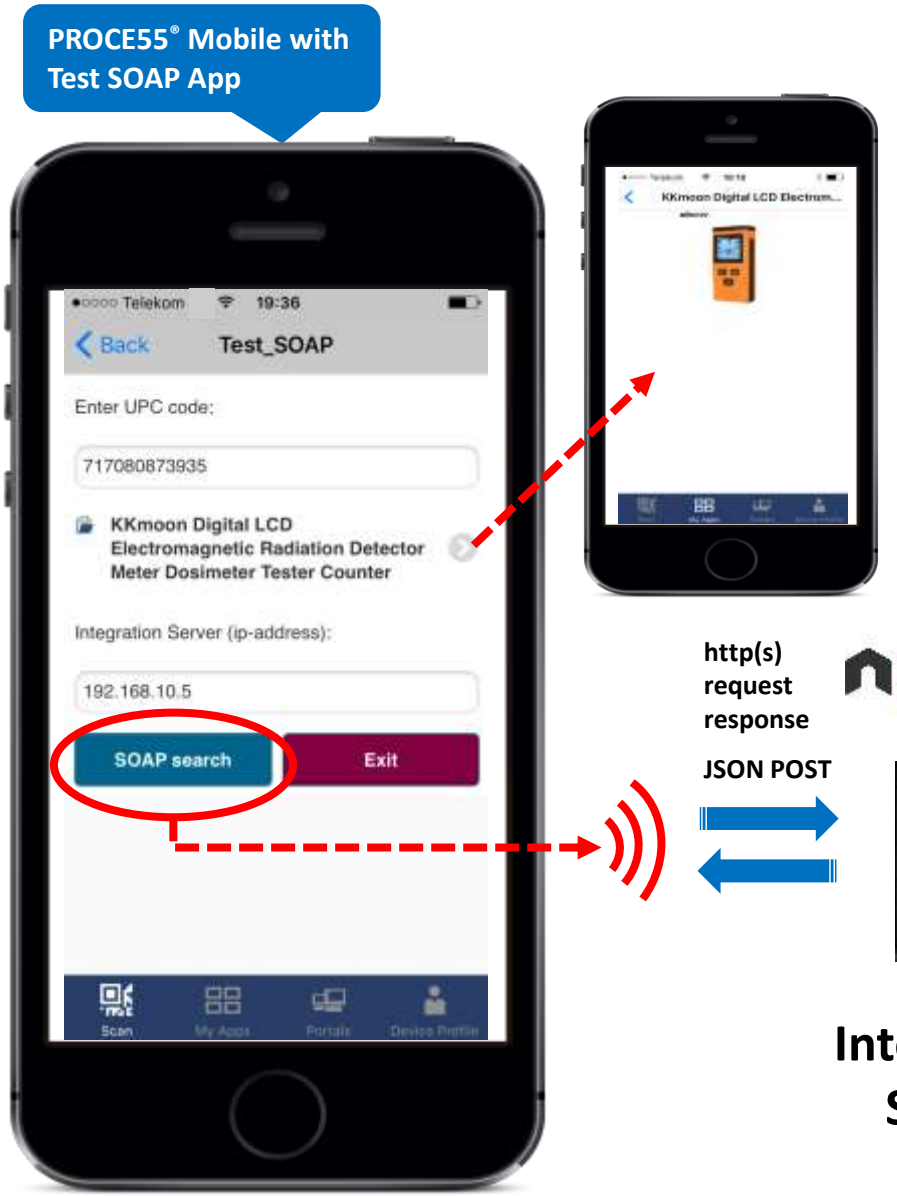
*This example shows how to establish a connection to a public SOAP service from your PROCE55 Mobile app using a small integration server.*
*At the end of this document we provide the server source code for your testing.*
*You can install the node.js environment and check the functionality immediately.*

**Note:**

*Before you can use the service shown in the above example, you have to register at*
*www.searchupc.com/developers*
*and ask for your own access token.*

☺ *Please, do not forget to open related ports on your firewall (port 80 by default)*

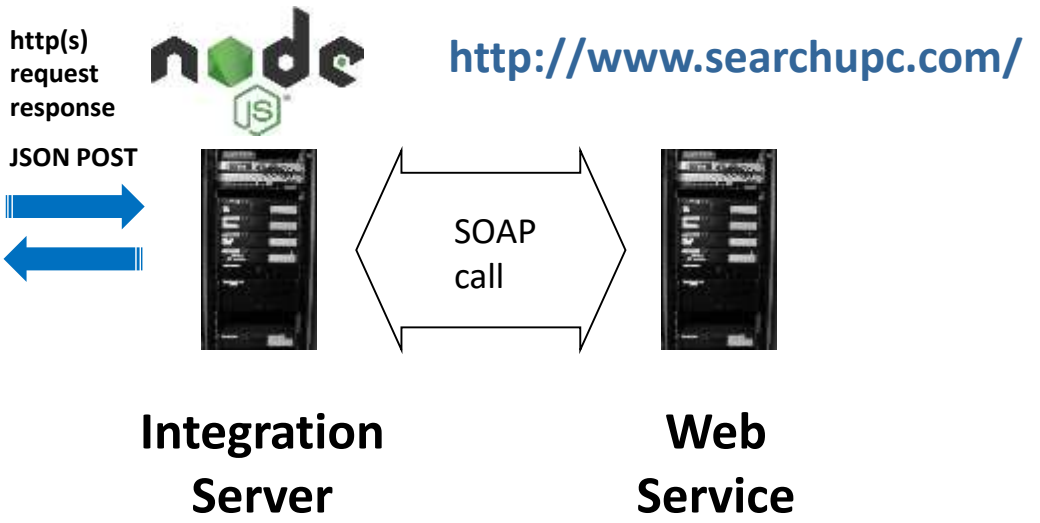
# PROCE55® Mobile: Example Mobile App – SOAP based Integration





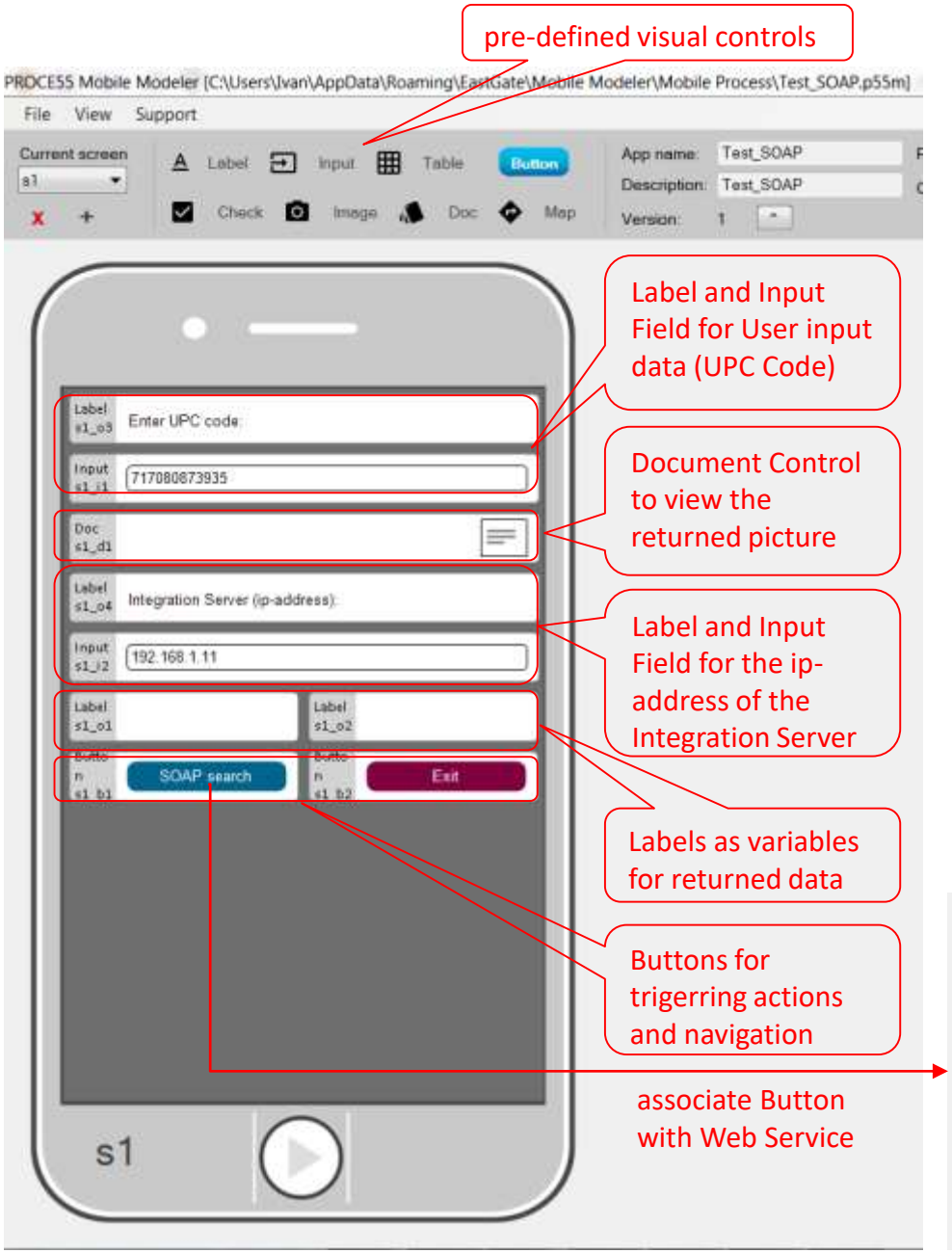

**Test SOAP App**

This example mobile app connects the public Web Service using SOAP call via Integration Server to transform the http POST request/response with JSON data to the SOAP call. This picture shows how it works.



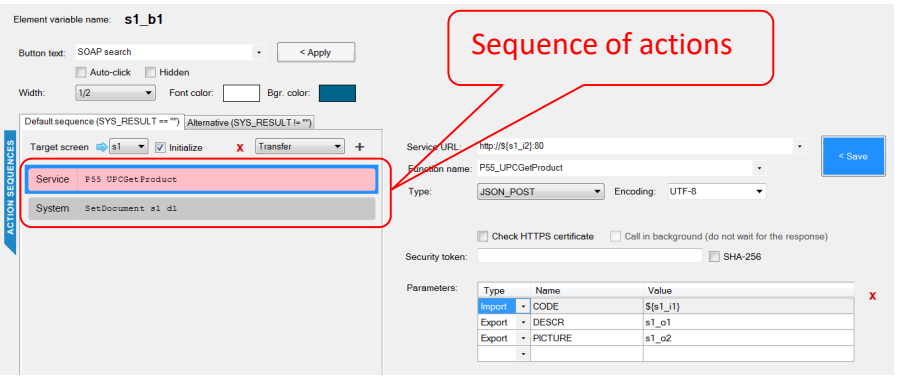

http://www.searchupc.com/service/UPCSearch.asmx?wsdl

# PROCE55® Mobile: How to create Test SOAP App – Step 1



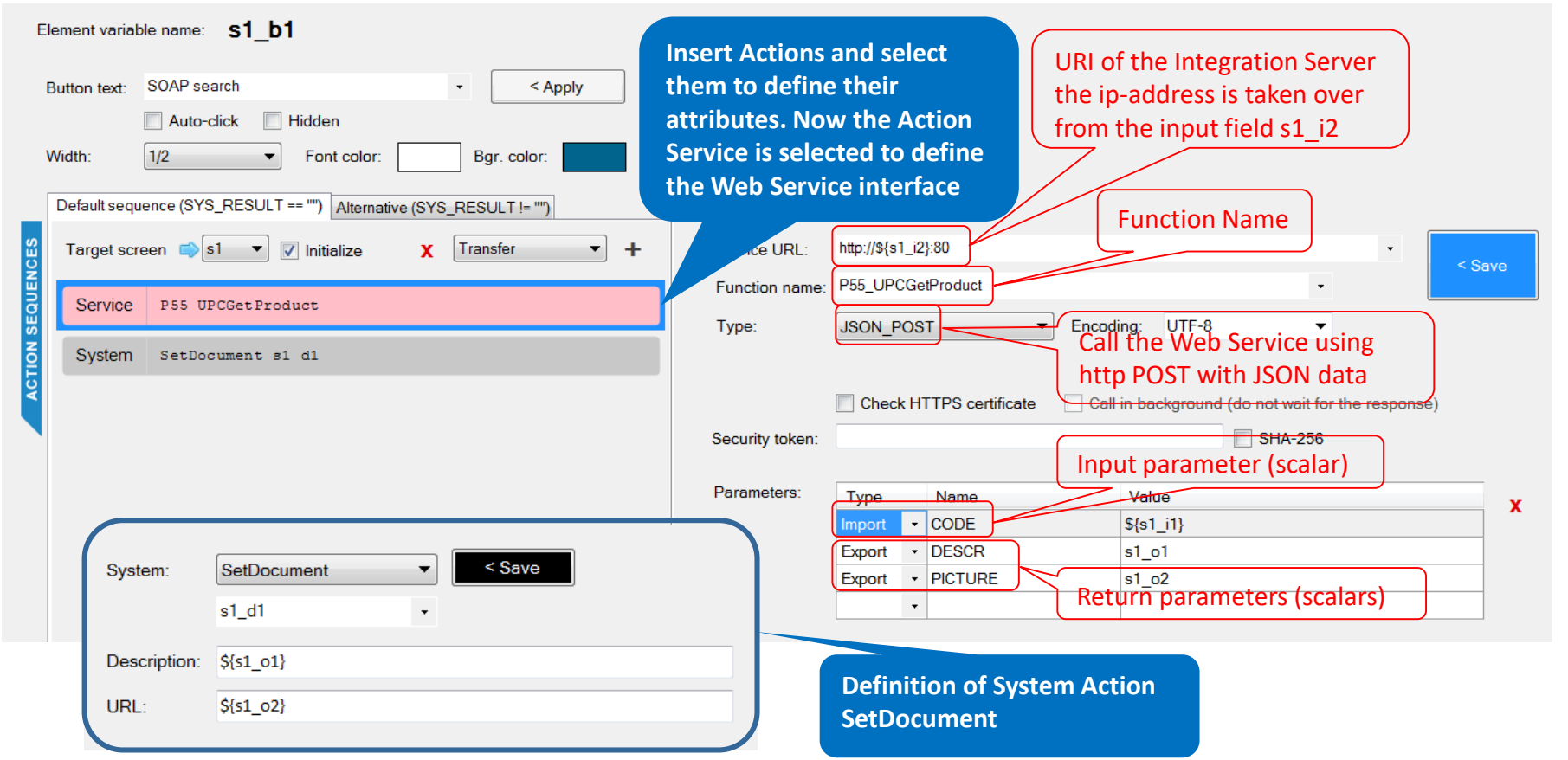

## Design User Interface

The visual User Interface can be modelled with the PROCE55® Modeler using the pre-defined **visual controls** which can be placed on the screens of the mobile App. **Buttons** are used to navigate among the screens. Each button can be associated with a **sequence of pre-defined actions**, the flow of actions is controlled by **conditions**.



# PROCE55® Mobile: Example Mobile App – Step 2

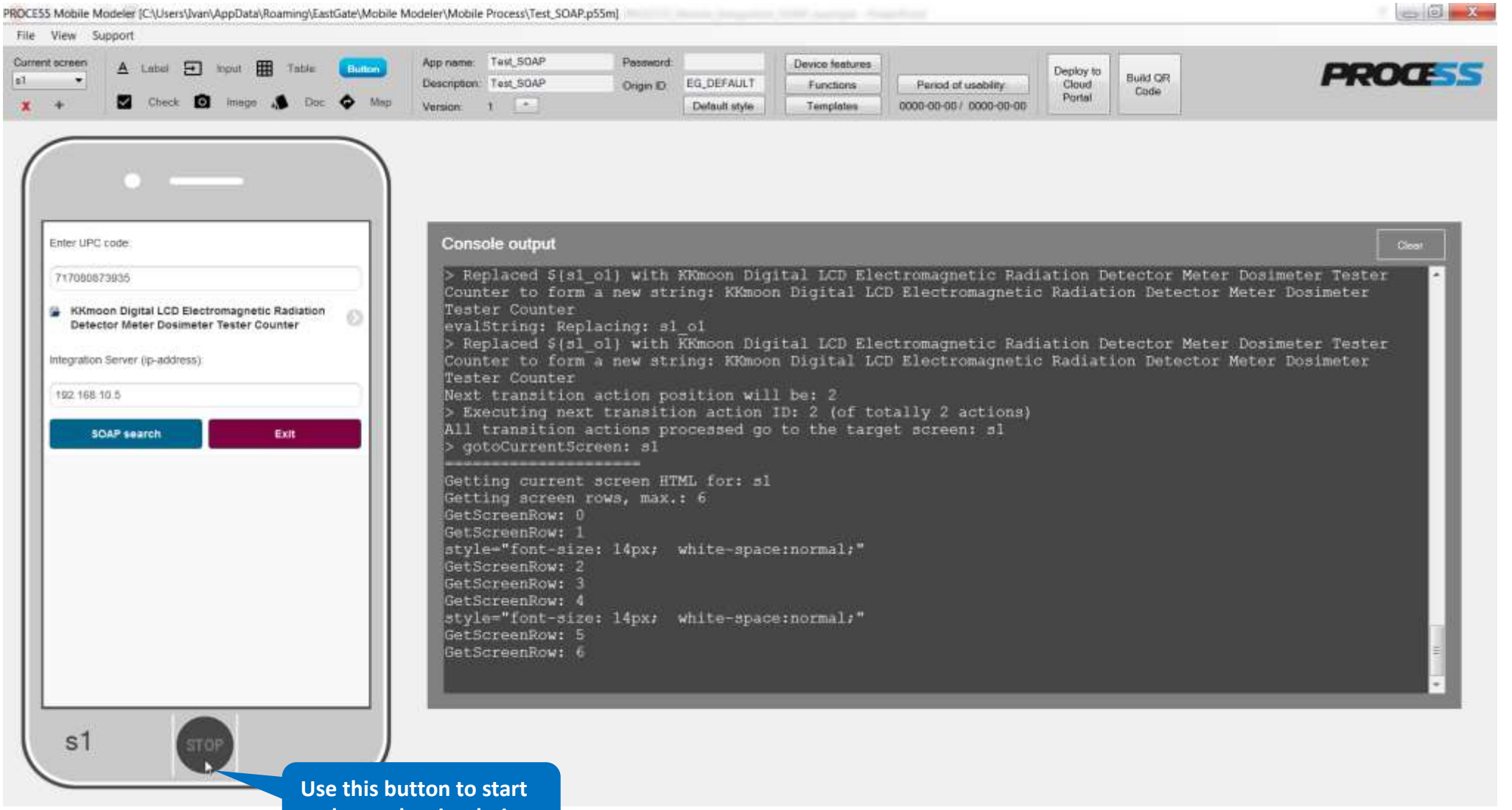
**Define the Application Logic (Client side)** the application logic of the app can be defined as a sequence of predefined actions which encapsulate typical functionality including the conditions (building blocks). If more flexibility is needed then the JavaScript functions can be built-in into application – you have no restrictions regarding application logic then.




According to the definition of the **Web Service** interface the function **P55_UPCGetProduct** will be called at run-time. The function has one **input parameter CODE** which value is transferred from the input field **s1_i1**. The value of the **return parameter DESCR** is transferred to the invisible label **s1_o1** and the value of the **return parameter PICTURE** (which is an URL of the picture) is transferred to an invisible label **s1_o2**. The invisible labels are used as internal variables for setting the Document control **s1_d1** to show the picture and its description. For setting the Document control the **system action SetDocument** is used.

# PROCE55® Mobile: How to create Test SOAP App – optional Step

**Testing and Simulation** you can use the **PROCE55® Modeler** to test the basic functions of the mobile App before deployment.



# PROCE55® Mobile: How to create Test SOAP App – Step 3

**Deployment of the mobile App with PROCE55® Modeler is very simple** compared with the procedures used in the App Stores. You can deploy the mobile App directly from the PROCE55® Modeler using QR Code or via Portals for complex deployment scenarios. **No compilation** is needed, you can deploy to **iOS, Android or Windows** directly without any transformation.

# PROCE55® Mobile: Example Mobile App – Server source code

Please, put the content below into 4 separate text files named as shown here:

## server.js

```
// PROCE55 node.js service
//
var http = require('http');

// Custom functions:
var CUSTOM = require('./_function_defs.js');
//var UPCfunctions = require('./UPCfunctions.js');
var server_port_number = 80;

var ip = require("ip");
console.log ( "PROCE55 Application Server listening on: " +
ip.address() + ":" + server_port_number );

http.createServer(function(request, response) {

  var headers = request.headers;
  var method = request.method;
  var url = request.url;
  var body = [];
  //var userAgent = headers['user-agent']; // headers always low-case!
  //console.log("Method: " + method + ", URL: " + url + ", agent: " +
userAgent);

  request.on('error', function(err) {
    console.error(err);
  }).on('data', function(chunk) {
    body.push(chunk);
  }).on('end', function() {
    body = Buffer.concat(body).toString();

  response.on('error', function(err) {
    console.error(err);
  });

  // process JSON request
  var client_resp = processJSONRequest(body.trim());
  console.log("Returning response.");
  //
  response.statusCode = 200;
  response.setHeader('Content-Type', 'application/json');
  response.setHeader('Access-Control-Allow-Origin', '*');
  response.write(JSON.stringify(client_resp));
  response.end();

  });
}).listen(server_port_number);

// -----------------------------------------------------

function processJSONRequest(request) {
  var response = "";
  console.log("====================\r\nProcessing JSON request: " +
request);
  // ...
  try {
    var msg = JSON.parse(request);
    // Send the P55Message object to the function specified:
// if PREFIX then...
// msg.FunctionName UPCfunctions.GetUPCData
//    response = eval(msg.FunctionName + "(msg)");
// else ...
    response = eval("CUSTOM." + msg.FunctionName + "(msg)");
    console.log("====================");
  }
  catch (err) {
    console.log("Error: " + err.toString());
    response = "";
  }
  return response;
}
```

## PROCE55_functions.js

```
// PROCE55_function.js
//

// =====================================================
// Exported functions (accessible outside PROCE55_functions.js)
module.exports = {

  P55Message: function () {
    this.FunctionName = "";
    this.Status = "";
    this.StatusText = "";
    this.TextEncoding = "";
    this.SecurityToken = "";
    //
    this.Parameters = [ ];
    this.TableValues = [ ];
  },

  P55Param: function (ParamName, ParamValue) {
    this.ParamName = ParamName;
    this.ParamValue = ParamValue;
  },

  P55TableValue: function (TableName, RowNumber, FieldName, Value) {
    this.TableName = TableName;
    this.RowNumber = RowNumber;
    this.FieldName = FieldName;
    this.Value = Value;
  },

  GetParamValue: function ( msg, Name ) {
    var Value;
    var ParLen = msg.Parameters.length;
    var i;
    for (i = 0; i < ParLen; i++) {
      if ( msg.Parameters[i].ParamName === Name )
      { Value = msg.Parameters[i].ParamValue; break; }
    }
    return Value;
  },

  // --------------------------------

  myfunction1: function () {
    // function logic

  },

  myfunction2: function () {
    // function logic
  }

};
// =====================================================


// internal functions (not accessible outside PROCE55_functions.js)
//

var internalFunction1 = function () {

}

var internalFunction2 = function () {

}
```

## _function_defs.js

## _start_service.bat

```
start node server.js
```