D365 Fin Ops Interface Design Pattern

While D365 offers odata and data entities for the means of exchanging data, it isn't the greatest mechanism for initiating operations. The following interface design pattern utilizes a façade and a class factory so that a REST client need only know which class to call and what parameters to pass in as a string array. Additional concrete classes could be implements without needing to modify the framework or redeploy the service.

The following example is used to post an inventory movement journal.


```
using (HttpClient client = new HttpClient())
{
    client.BaseAddress = new Uri(ConfigurationManager.AppSettings["baseAddress"]);
    client.DefaultRequestHeaders.Add("Authorization", Program.getOauthHeader().Result.CreateAuthorizationHeader());
    HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Post,
        "api/services/IntegrationServiceGroup/IntegrationService/fireOffIntegration");//common service class
    String[] parms = { "00201" };          //only need to vary these two lines    //InventJournalId
    string className = "PostInvMovJournals"; //to call differnt concrete classes
    var parametersToPass = new
    {
        _className = className,
        _parms = parms
    };
    request.Content = new StringContent(JsonConvert.SerializeObject(parametersToPass),
                            Encoding.UTF8,
                            "application/json");
    var result = client.SendAsync(request).Result;
    Console.WriteLine("result.StatusCode = " + result.StatusCode);
    Console.WriteLine("result = " + result.Content.ReadAsStringAsync().Result);
}
```

```
file:///C:/chrysler/D365FinOpsConsoleTes
result.StatusCode = OK
result = "done!"
finished.
```

REST Client

D365FinOps

RestService
- fireOffIntegration
- defineParms

```
class IntegrationService
{
    public str fireOffIntegration(str _className, System.String[] _parms)
    {
        return IntegrationBuilder::buildConcreteClass(_className).run(_parms);
    }
}
```

<<Interface>>
ServiceClassInterface
- run

```
interface IntegrationInt
{
    public str run(System.String[] _parms){}
}
```

Builder
- className

```
class IntegrationBuilder
{
    public static IntegrationInt buildConcreteClass(str _className)
    {
        IntegrationInt concrete;
        SysDictClass sysDict;
        ;
        sysDict = new SysDictClass(classname2id(_className));
        concrete = sysDict.makeObject();
        return concrete;
    }
}
```

ConcreteServiceClass
- run

```
class PostInvMovJournals implements IntegrationInt
{
    public str run(System.String[] _parms)
    {
        journalCheckPost journalCheckPost;
        journalCheckPost = InventJournalCheckPost::newPostJournal(inventJournalTable::find(_parms.GetValue(0)));
        journalCheckPost.run();
        return "done!";
    }
}
```