



# Implementing hybrid clients for Polarion

by Nick Entin

Where today meets tomorrow

**SIEMENS**

## Agenda

- Scope
- Why hybrid client
- Polarion-Simulink Connector as an example
- Implementation blueprints
- Implementation tips & tricks

Note: The innovative interaction between remote-API-based application and Polarion UI in Embedded browser is reflected in a patent application, and you're more than welcome to enjoy it by creating new integrations with Polarion!

## Scope

- There are continuous demand to create client applications, which communicate with Polarion Server
- Those clients fulfill different purposes
  - Automation of operations (exporting data in specific format, imports and creation of objects, etc.)
  - Synchronization of the systems (including run-time synchronization, and model mapping)
  - Alternative UIs for Polarion data
  - **Hybrid solutions – data is synchronized, but likely only partially, and the rest might be seen in Polarion directly**

## Why hybrid clients

- Hybrid clients offer usually best-of-the-greed experience for the user:
  - Synchronizing of the data allows using native clients for processing of the data
  - Data synchronization could be limited to the extend, required for the client application, and not full-blown data replication
  - Possibility to see all the data in one place and in context of other objects in Polarion
- Such clients may operate on level of APIs to read/write raw data, while integration with an embedded browser can help in interoperability with rendered content of Polarion.

This presentation will concentrate on use-case with Polarion-Simulink integration, which applies the hybrid client approach.

# Orchestration Simulink Controls Development with Polarion

Embedded Polarion

Simplified User Experience

Traceability & Coverage

Improved Collaboration

**Requirements - FuelSysWithReqLinks**

Index	ID	Summary
LiveDoc_Lambda FuelRateController Specifications_Fault Tolerant Fuel System Architecture Requirements Specification*		
Import1	https://lambda-latest...	References to Fault Tolerant Fuel System Architecture Requirements Specification?hide_navigator=false
1	FRC-141	Main Sub Systems
1.1	FRC-167	Airflow
1.1.1	FRC-143	Mass airflow estimation (Airflow calculation)
1.2	FRC-168	Pumping Constant
1.2.1	FRC-145	Determination of pumping efficiency (Pumping Constant)
1.3	FRC-171	Airflow Integrator
1.3.1	FRC-147	Adjustment of estimated airflow (Airflow calculation/Integrator)

**Coverage Analysis Report**

Select a document

selectedDoc: Specifications / Fault Tolerant Fuel System Architecture Requirements Specifi -

Apply Save as Default

Coverage Analysis

Not Covered: 2

Covered: 10

Legend: Covered (Green), Not Covered (Red)

**Fault Tolerant Fuel System Architecture Requirements Specification**

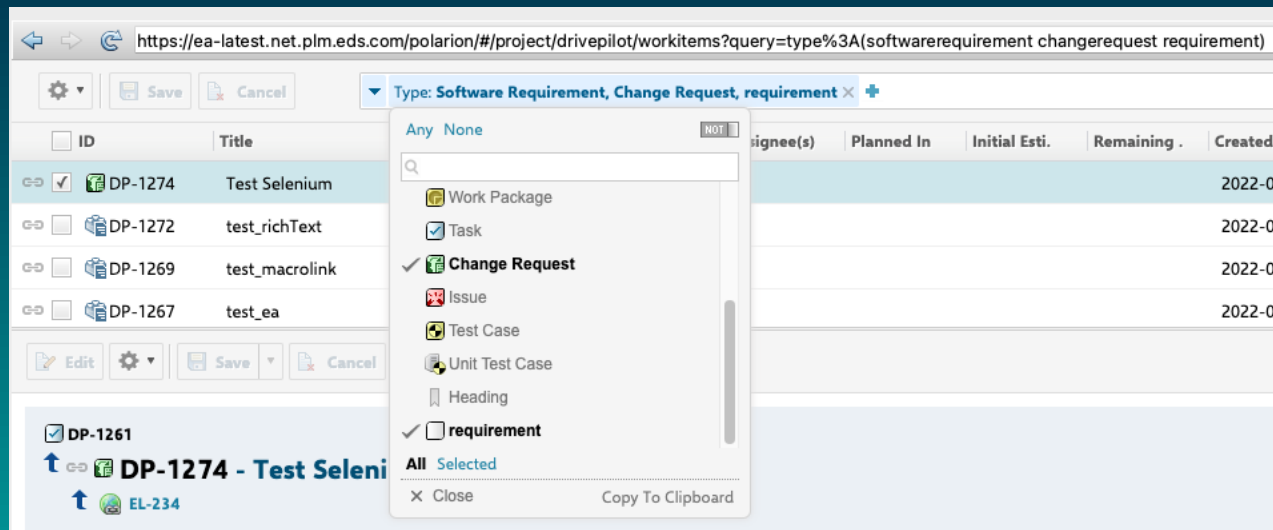
# Implementation blueprints

## Embedded browser integration

- Allows client application to set the URL (e.g. URL within Polarion)
- Allows client to read current URL in Polarion (even if this URL was updated by Polarion as redirection)

## Polarion specifics

- Each view in Polarion has an unique URL
  - E.g. users can send URLs to each-other to recover views on other machine
- The URL incorporates all necessary parameters, characterizing the view



Using query builder – selections are immediately reflected in the browser's URL

## Implementation blueprints (Contd.)

URL doesn't only reflect query or filter for the content to be displayed, but also shape and form of it

ID	Title	Status	Prio.	Assignee(s)	Planned In	Initial Esti.	Remaining .	Created
DP-345	iPad User Interface application can be dowr	✓ Appr	50.0					2017-05-21 2:
DP-393	iPad User Interface application auth code	Draft	55.0	Philip GUI	Iteration 3		1/2d	2017-05-23 1:
DP-3'	Implement: iPad User Interface application :	In Prc	50.0	Robert Projec		13 1/2d	13 1/2d	2017-05-23 1:
DP-334	The User Console will resemble a typical da:	✓ Appr	50.0					2017-05-21 2:

Switching to the Tree view and parameterizing it with link role(s), depth of the tree, etc. is also reflected in the URL

# Implementation blueprints (Contd.)

## Even versioning is reflected in URL

The screenshot shows a web browser displaying a document titled "System Requirement Specification" within a project named "Drive Pilot". The URL in the address bar is `https://ea-latest.net.plm.eds.com/polarion/#/project/drivepilot/wiki/Requirements/System Requirement Specification ?revision=9913`. The document content includes a table of contents with sections: "1 Introduction", "1.1 Purpose", and "1.2 Scope". The "1.1 Purpose" section contains the text: "The System Requirement Specification (SysRS) describes what the system's sponsor expects the system to do, the system's expected environment, the system's usage profile, its performance parameters, and its expected quality and effectiveness. The SysRS is a structured collection of information that embodies the requirements of the system. The SysRS completely describes all inputs, outputs, and required relationships between inputs and outputs." On the right side, a "Work Item Properties" panel shows details for work item "DP-1252 - System Requirement S", including a severity of "Should Have" and a status of "Open". A "Links" section indicates it is the parent of "DP-1257 - Introduction" and "DP-1253 - Requirements".

Specific version or baseline of the document is reflected in the URL



## Implementation blueprints (Contd.)

### Reading the URLs

- For majority of the integration cases, client needs to differentiate access to Work Items and LiveDocs
- You might use following regular expression to parse the URL and get 3 most important tokens:
  - Server URL (if you're using any remote APIs – WebServices or REST, you'd need this anyway)
  - Project ID
  - LiveDoc location (if presented, with its space(s))  
`(\S+)/polarion/(\#|/redirect)?project/(\S[^\?]+)((/collection/.[^\?]*wiki/)|(/wiki/))?(.[^\?]*)?.*`
  - If you tokenize result of the RegExp, you should see: Server URL as token 1, Project ID as token 3, and LiveDoc location as token 5
- If you need to parse individual Item URL, you can use following RegExp  
`(\S+)/polarion/(\#|/redirect)?project/(\S+)/workitem\?id=(\S[^\&]+).*`
  - In this case you'll get: Server URL as token 1, Project ID as token 3, and Work Item ID as token 4

# Implementation blueprints (Contd.)

## Parsing Parameters

- Reading of the parameters is easier, just extract part of the URL after '?' delimiter, and split by the '&'.
- You might need to decode HTML codes for encrypted characters to the ASCII ones. There are different approaches, in MATLAB language it might look... mathematically :-)

```
function str = DecodeURL(s)
    str = "";
    k = 1;
    while k<=length(s)
        if s(k) == '%' && k+2 <= length(s)
            str = sprintf('%s%c', str, char(hex2dec(s((k+1):(k+2)))));
            k = k + 3;
        else
            str = sprintf('%s%c', str, s(k));
            k = k + 1;
        end
    end
end
```

\* Example from internet

# Implementation blueprints (Contd.)

## Composing the URLs

- When you need to open a browser for specific item, document or query, you can compose the URL simply by concatenating those pieces:
  - Project root: [serverURL '/polarion/#/project/' projectID]
  - Individual Work Item: [serverURL '/polarion/#/project/' projectID '/workitem?id=' wiID]
  - LiveDoc: [serverURL '/polarion/#/project/' projectID '/wiki/' liveDocLocation]
  - Work Items in table: [serverURL '/polarion/#/project/' projectID '/workitems']

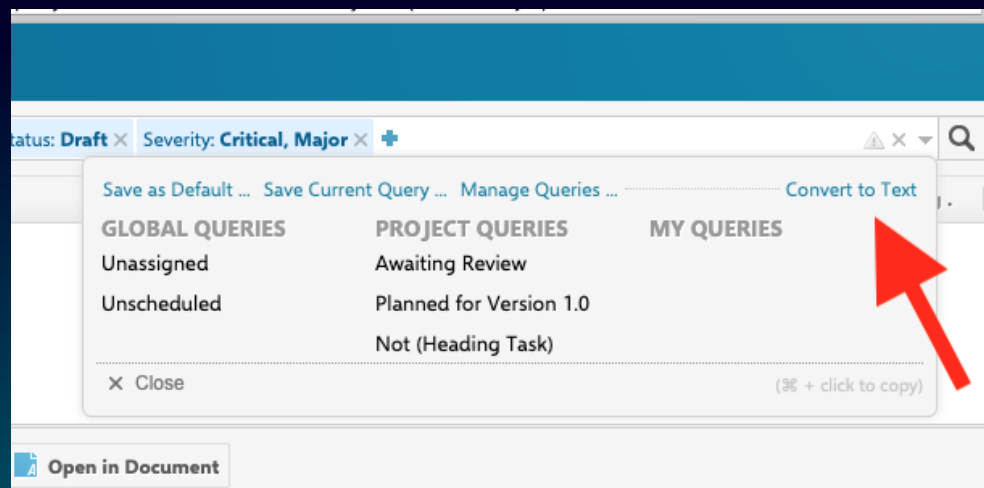
## Parameterizing the URLs

- You can pass parameters in same form, as you read the URLs or simply copying those from the URL of the browser, where you've configured desired output, e.g.:
  - '?query=something' for filtering Work Item table according to the 'something'.
  - '?revision=123' allows to open particular revision of the LiveDoc

# Implementation Tips & Tricks

## Queries

- You can use Query Builder in Polarion, and then “Convert to Text” to create queries for the client



## Special Parameters:

- You can pass parameter: `'hide_navigator=true'` to tell Polarion to completely hide left-side navigation
- You can pass parameter: `'selection=workitemID'` to navigate within LiveDoc
- You can use parameter: `'track_url=false'` if you want to tell Polarion not to remember this URL as last visited

# Implementation Tips & Tricks (Contd.)

## Redirects

- Very likely you've noticed that majority of Polarion URLs have a '#' symbol inside. This is a pointer to the server to load whatever it is there on the right of this symbol however it decides – e.g. to load from the cache, etc. In most of the cases, it improves UX impression from Polarion, because it doesn't reload all the panels and views, but can decide which one is affected. For example, if you had this document already opened, and now add **'?selection=DP-473'**, the document will be simply scrolled, and target element highlighted:

The screenshot shows a web browser window with the URL: <https://ea-latest.net.plm.eds.com/polarion/#/project/drivepilot/wiki/Requirements/System Test Case Specification?selection=473>. The interface is for the 'Drive Pilot' project, showing a 'Requirements' section for 'System Test Case Specification'. The main content area displays a table of test steps:

Step	Step Description	Expected Result
vehicle with "auto-parking"		
Start DrivePilot System	Push the start button	Welcome screen is displayed
Start AutoPilot	Click on "AutoPilot"	The system says the car has an "auto-parking" feature and DrivePilot cannot be started

Below this table, a specific test case is highlighted: **DP-473 - DrivePilot will disengage when User shouts "Stop"**. This test case has its own table of test steps:

Step	Step Description	Expected Result
Install and Start DrivePilot		DrivePilot is started and vehicle is controlled by DrivePilot
User shouts "Stop"		Auto pilot mode is stopped with audible and visual notifications

The interface also shows a sidebar with navigation options like 'Development', 'Maintenance', 'Planning', 'Reports', and 'Requirements', and a user profile for 'SimulinkTester My Polarion'. The Siemens logo is visible in the bottom right corner.

## Implementation Tips & Tricks (Contd.)

### Redirects

- However, it might not work, if a second before you've created a new item in the document via API, and server didn't yet re-indexed this save. Using such URL may fail, then.  
To avoid this case, you can replace '#' symbol by 'redirect' to tell Polarion completely reload specified content:

[https://ea-latest.net.plm.eds.com/polarion/redirect/project/drivepilot/wiki/Requirements/System Test Case Specification?selection=DP-473](https://ea-latest.net.plm.eds.com/polarion/redirect/project/drivepilot/wiki/Requirements/System%20Test%20Case%20Specification?selection=DP-473)

instead of

[https://ea-latest.net.plm.eds.com/polarion/#/project/drivepilot/wiki/Requirements/System Test Case Specification?selection=DP-473](https://ea-latest.net.plm.eds.com/polarion/#/project/drivepilot/wiki/Requirements/System%20Test%20Case%20Specification?selection=DP-473)

## Additional Information

# Thank You

Website	<a href="https://polarion.plm.automation.siemens.com/">https://polarion.plm.automation.siemens.com/</a>
Extensions	<a href="http://extensions.polarion.com">http://extensions.polarion.com</a>
Tutorials	<a href="https://polarion.plm.automation.siemens.com/tutorials">https://polarion.plm.automation.siemens.com/tutorials</a>
Documentation	<a href="https://polarion.plm.automation.siemens.com/guides-and-manuals">https://polarion.plm.automation.siemens.com/guides-and-manuals</a>
Templates	<a href="http://extensions.polarion.com/searches?category=1-templates">http://extensions.polarion.com/searches?category=1-templates</a>
Blog	<a href="https://community.plm.automation.siemens.com/t5/Polarion-Blog/bg-p/Polarion_blog">https://community.plm.automation.siemens.com/t5/Polarion-Blog/bg-p/Polarion_blog</a>
YouTube Channels	<a href="https://www.youtube.com/user/PolarionSoftware">https://www.youtube.com/user/PolarionSoftware</a> <a href="https://www.youtube.com/playlist?list=PLYGbB3CefExR30eGImwx6M-ECMXYHIAJm">https://www.youtube.com/playlist?list=PLYGbB3CefExR30eGImwx6M-ECMXYHIAJm</a>

