

PIXI Interface Problems and Solutions

Tips and tricks using SAP PI / XI servers
29.03.2009

Introduction

SAP XI/PI technology is new. It's been on the market for 2-3 years and mainstream clients have just started using it. We collected some experience we would like to share, so you can spare time and concentrate on your important tasks.

PIXI

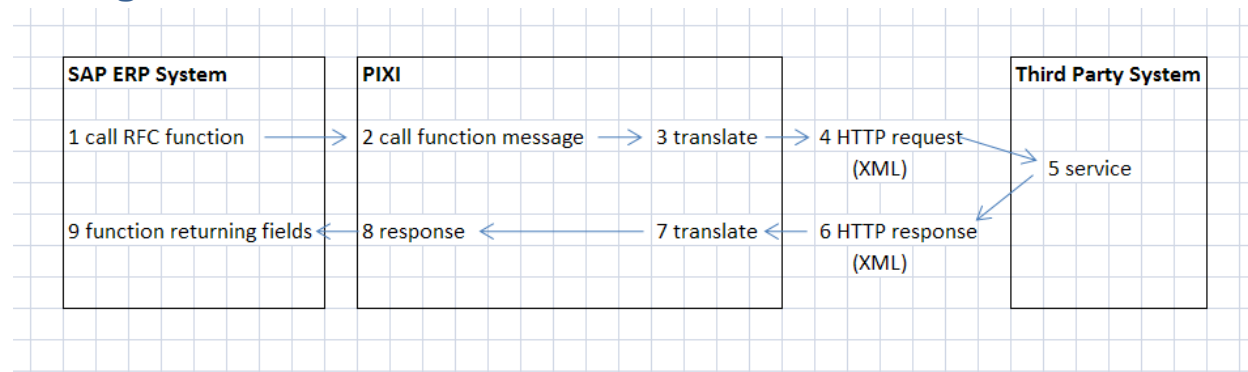
If you wonder what PIXI is, then I tell you: It is an SAP product that was called eXchange Infrastructure(XI) and then unaccountably renamed to Process Integration (PI). Some were calling it XI, some PI leading to frequent misunderstandings. So Common Sense intervened and since then they are unofficially called PIXI servers.

Anyway behind the good idea there are some fine adjusting you might need some help with. We created a nice online interface that looks like this:



It is not a SOAP interface, but very similar actually without the SOAP structures and envelopes.

Message Flow



The Checklist I've used

on basis ERP system:	1	create transfer structures (SE11)
	2	create basis system RFC functions (SE37)
	3	import RFC functions to PIX1
in enterprise service builder:	4	message mapping RFC->request.xml
	5	message mapping response.xml->RFC
	6	operation map
	7	service interface (you dont need message or data types, if you have example XML files)
in integration builder:	8	source business component or communication component (if you dont have that already)
	9	target business component or communication component (if you dont have that already)
	10	receiver determination
	11	interface determination
	12	sender agreement
	13	receiver agreement

Message and data types

I have good news: you don't need them. If you have example XML files, you just use them to create your message mappings. You can push the import button as seen on the following picture and just choose your message. The other side is, of course, the RFC function interface. Don't forget that for each function you need two message mappings: 1 request and 2 response.

The screenshot displays the SAP Enterprise Services Builder interface. The main window is titled 'Edit Message Mapping' for the message 'MM_VC_GET_ATTENDANCE'. The 'Definition' tab is active, showing the structure of the RFC message 'Y_VC_RFC_GET_ATTENDANCE' and the imported message 'attendance_request_v01.xml'. The structure comparison table is as follows:

Structure	Occurrences	Type	Structure	Value
Y_VC_RFC_GET_ATTENDANCE	1..1	rfc:YHG_VC_GET_ATTENDANCE	Root	
P_PARAM	1..1	xsd:string	URL	
VC_SERVERCODE	0..1	xsd:string	XMLVersion	
VC_URL	0..1	xsd:string	VendorName	
TAB_MODERATORS	0..1	rfc:YHG_VC_USE_ATTENDANCE	VendorVersion	
Item	0..unbounded	rfc:YHG_VC_USE_ATTENDANCE	Target	GetAttendance
FNAME	0..1	xsd:string		868155
LNAME	0..1	xsd:string		
TEMPID	0..1	xsd:string		

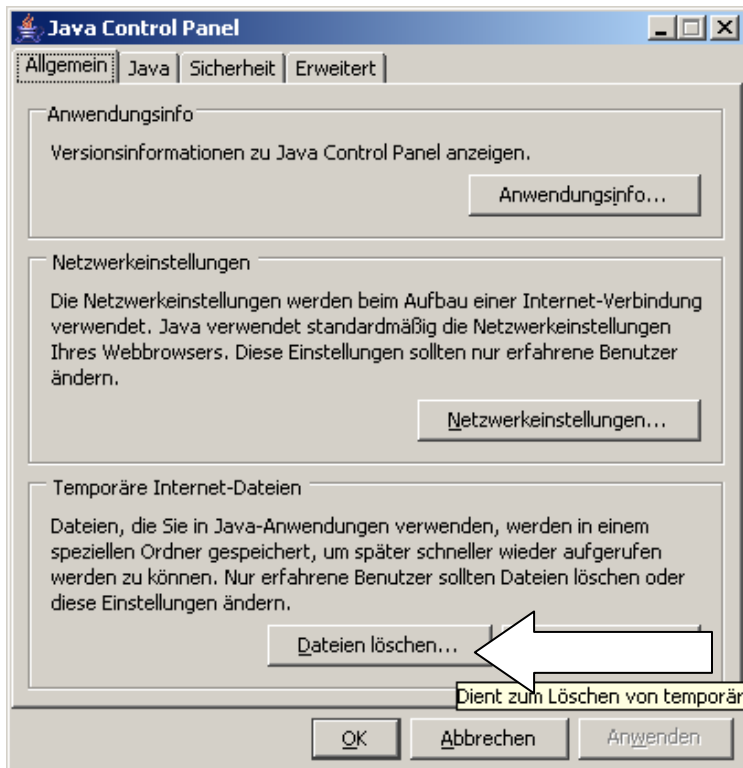
The 'Used Function Libraries' section at the bottom shows a flow diagram: Constant -> removeCo... -> Target. The function library 'FL_UserDefined' is selected, and the functions 'FL_UserDefined.AssignURLValues' and 'FL_UserDefined.ConcatenateValues' are visible in the bottom right.

Computer memory requirements

My computer had only about 800 mb of memory. As I needed access to one erp and two PIXI servers I really used up all my memory and then the whole thing got stuck. For one click I needed about 15 seconds. This made me crazy, but my dear colleague helped quickly by installing 2 GBs of RAM into my computer. I would say this is the minimum, if you want to work in flow.

Java problems

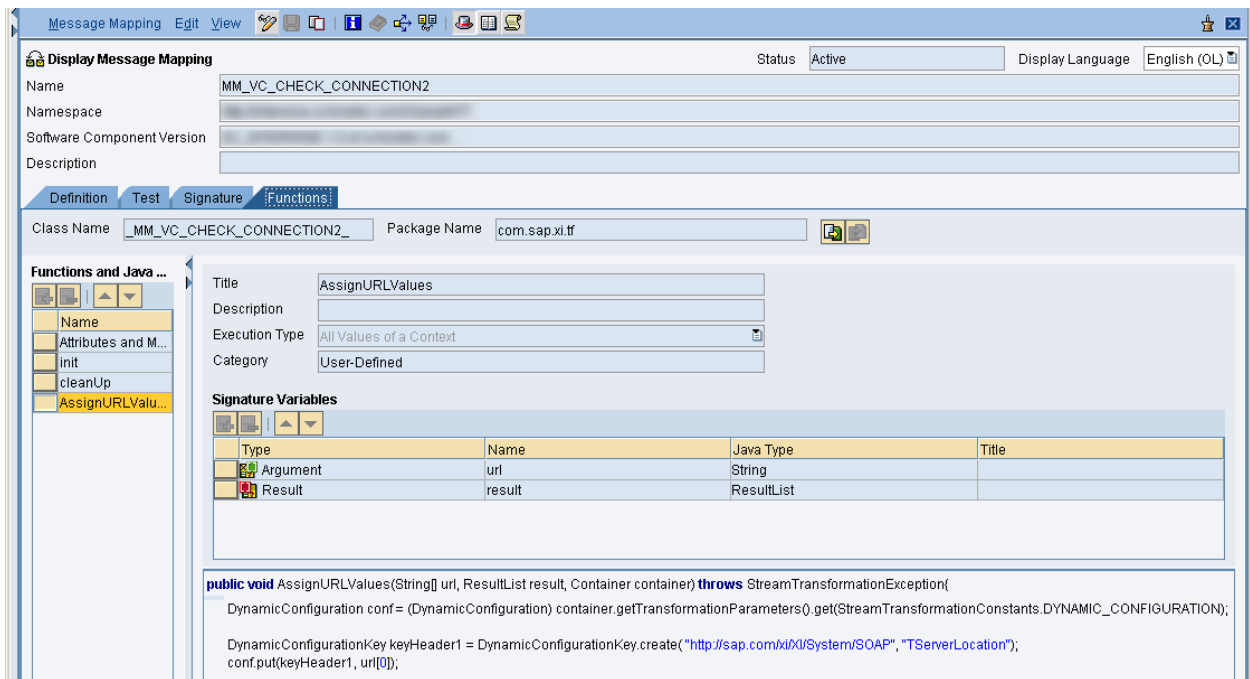
God created the Sun and Sun made Java. ;) It doesn't mean however, that it is perfect. For example we had an error like "Enterprise Services Directory cannot be started" ("Enterprise Services Repository kann nicht gestartet werden"). I've not yet found a perfect solution, but if you go "Control Panel" in windows then Java Control Panel and you remove all temporary internet files, it will start well.



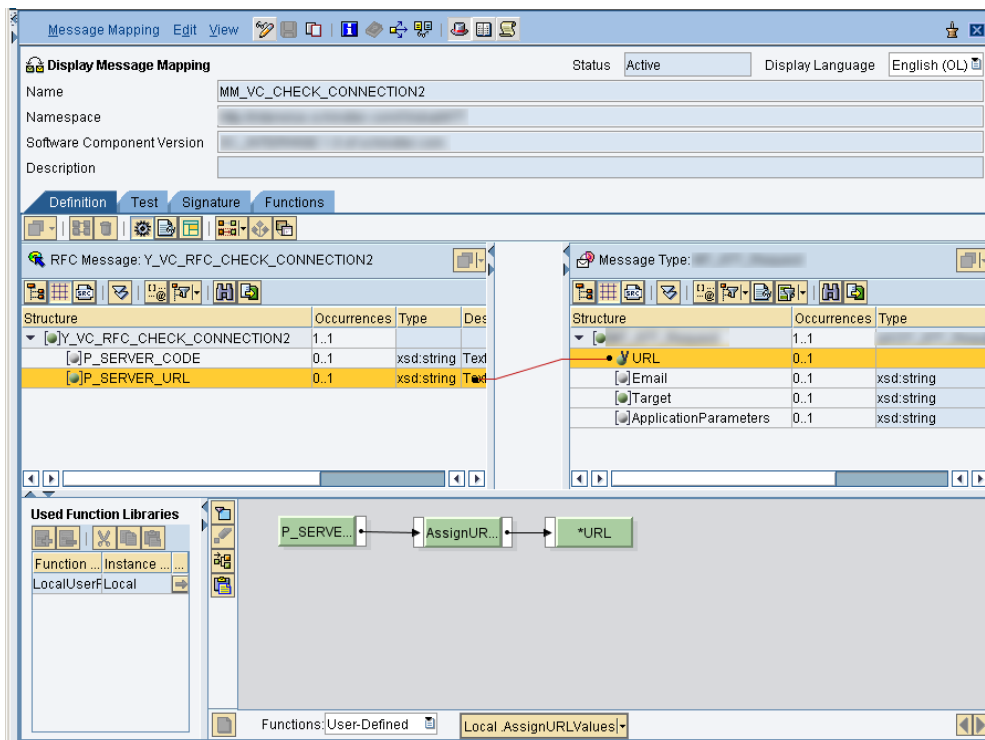
Dynamic URL routing

So you need to communicate with more than one servers? This will be tricky.

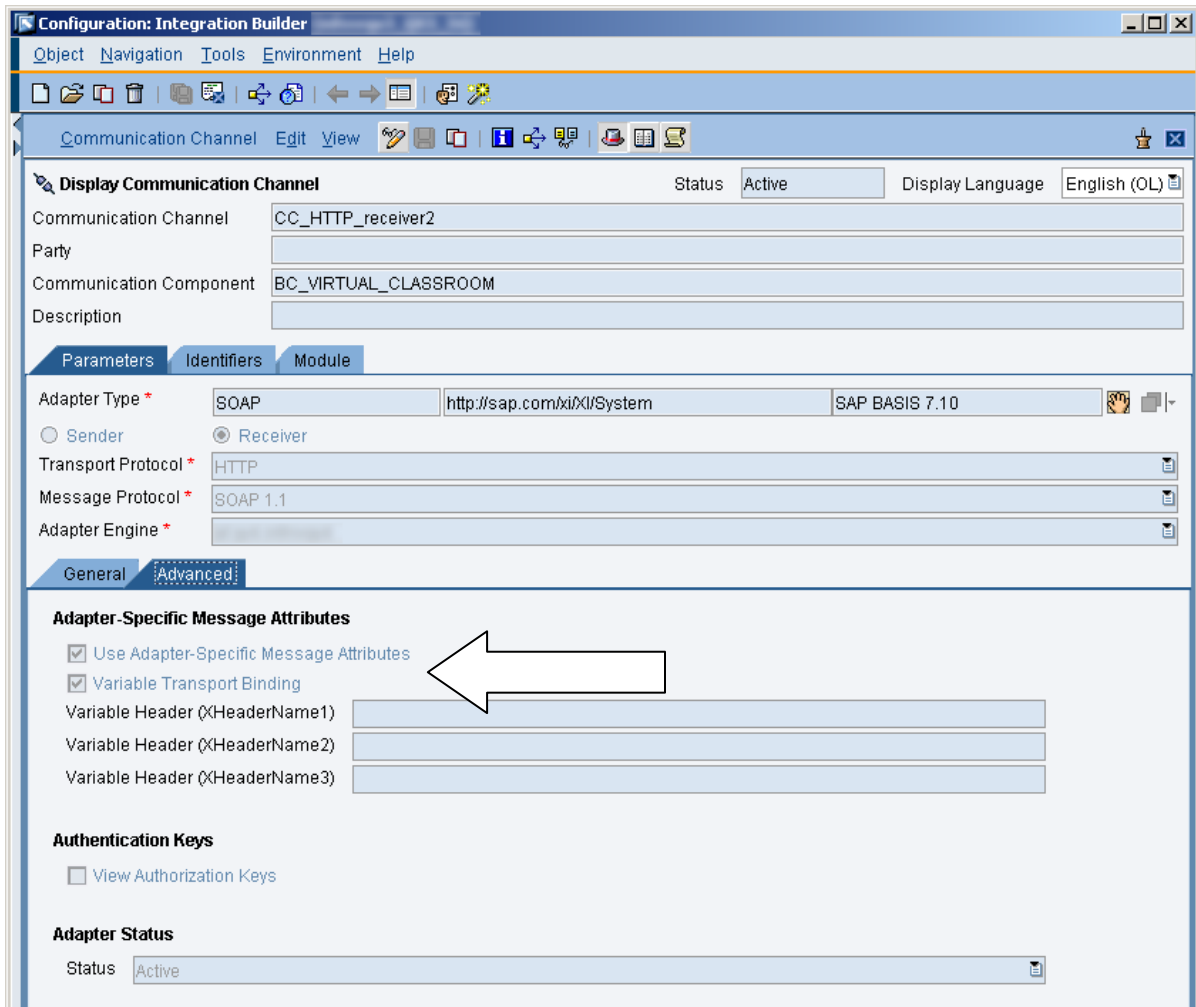
Steps needed:



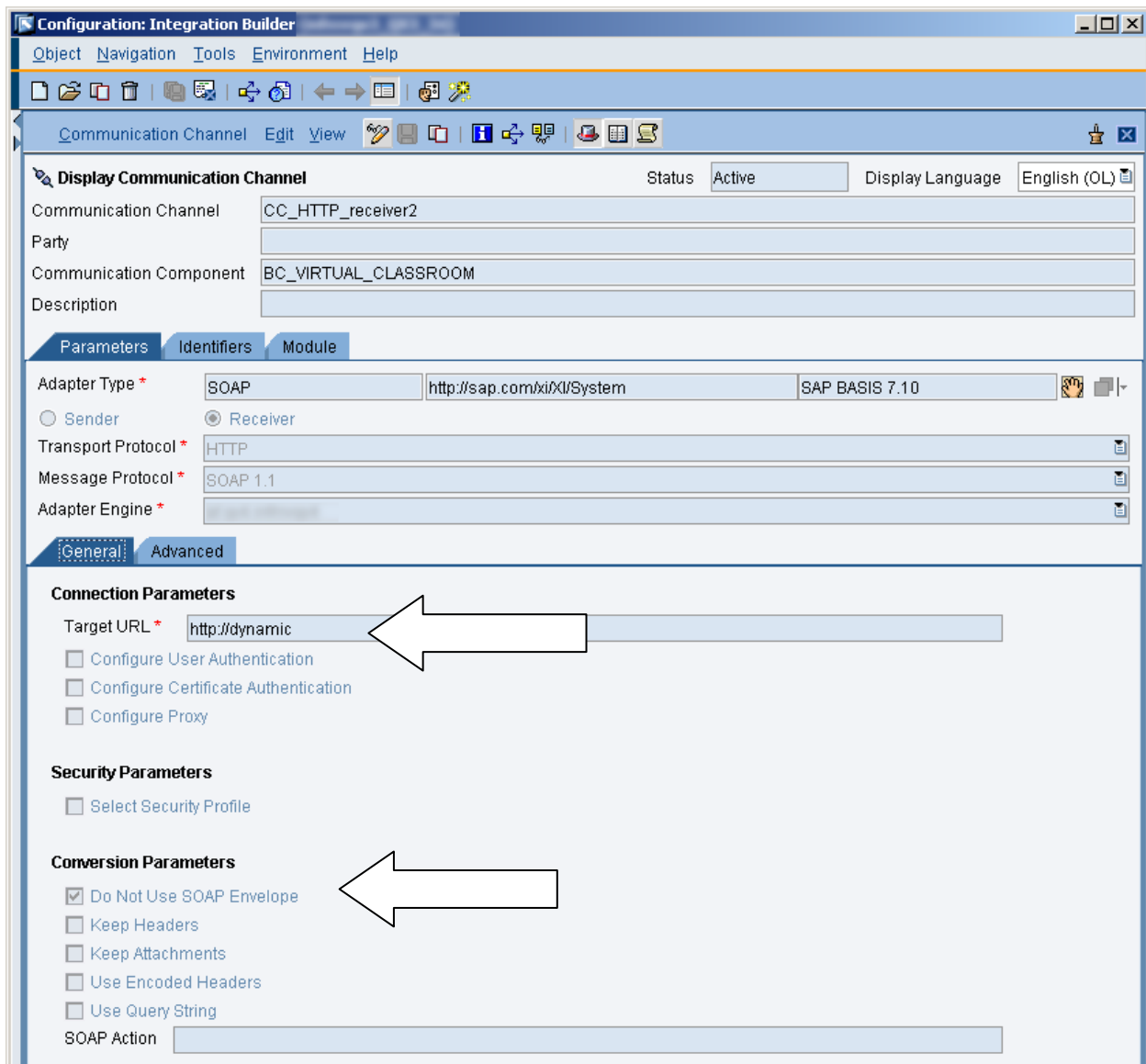
1. Create a function to put the URL into the header



2. On the input side, you deliver the URL, what you then transfer to a variable using your function from step 1



3. In Integration builder you set up a communication channel like this



4. You also set these parameters by the communication channel

Take special care for the URL you filled in as default, it has to have “HTTP://something” in it. (see screenshot)

Then, as a grotesquely beautiful additional barrier: runtime cache of PIXI. I’m not talking about the SLD cache, what you can easily refresh from your “Enterprise Services Builder”. No, you have to start transaction SXI_CACHE in the SAP GUI of PIXI and then go and refresh the caches. We had a message mapping problem – short dump – and it took us one hour to find this problem and clean the cache system.

XML-Header: fix HTTP content type

This problem even caused a bodily reaction, some funny feelings somewhere in the back of my stomach. The XML content was painstakingly assembled and was checked again, again and again... It failed, failed and failed. I installed a beautiful XML and SOAP tester tool, [soapUI](#) (cheers to its developers) and tested

the same XML with it. It worked. The XML-content was the same – except from some whitespace characters – and still it gave different results.

So I installed a home made HTTP request saver and saved both requests for later analysis. This way I’ve found the solution:

Both SAP and this third party are sure, how an XML header should look like. Both are sure, what the content type is. So they never even mention it. The only problem is, that they think differently:

HTTP content type is

- SAP: "application/xml"
- Third party: "text/xml"

This caused that uncertainty in the back of my stomach!

The solution is to go to your “communication channel” and extend it with a module according to the screen here:

The screenshot shows the SAP 'Display Communication Channel' interface. The communication channel is 'CC_HTTP_receiver', status is 'Active', and display language is 'English (OL)'. The communication component is 'BC_VIRTUAL_CLASSROOM'. The 'Module' tab is selected, showing a 'Processing Sequence' table and a 'Module Configuration' table. Two white arrows point to the 'Module Name' and 'Parameter Value' fields respectively.

Number	Module Name	Type	Module Key
1	localejbs/AF_Modules/MessageTransformBean	Local Enterprise Bean	Transform
2	sap.com/com.sap.aif.soapadapter/XISOAPAdapterBean	Local Enterprise Bean	3

Module Key	Parameter Name	Parameter Value
Transform	Transform.ContentType	text/xml; charset=utf-8

Basis Problems

Our PIXI server was new – it was installed a few months before: as a result, about 10-20 hours were wasted because of basis problems like these:

- PIXI adapter not working
- PIXI cannot start Enterprise Services Builder
- PIXI buffers not initialized
- PIXI updated and communication channel not restarted (RFC connection short dumps)

This means that you need some time, until such a server is getting stable and has a good uptime percentage.

Conclusion

In the long run it makes sense to use PIXI interfaces, but you have to have enough experience and be prepared for all kinds of tricks – see above – to get the job done.

Appendix

Software components used

SAP PI(XI) 7.1

<http://service.sap.com/xi>

http://en.wikipedia.org/wiki/SAP_Exchange_Infrastructure

SAP ERP

<http://www.sap.com/solutions/business-suite/erp/index.epx>

SOAPUI

<http://www.soapui.org/>

About the developers

Andreas Katzer is an experienced PI/XI consultant working in Switzerland.

[Zsolt Balai](#) – author of this document – is an SAP/Internet technical consultant/developer working in HR/FI/CO/MM/SD modules.

Keywords

PI, XI, SAP, interface, online, HTTP, XML, RFC, function, message mapping, problems, solutions