WEBMETHODS[1] FLOW LANGUAGE DESIGN AND PRACTICE

This paper explores and suggests on coding of webMethods flow service. The sole purpose of article is to make first-timer aware how to code webMethods flow service.

Rankesh Kumar

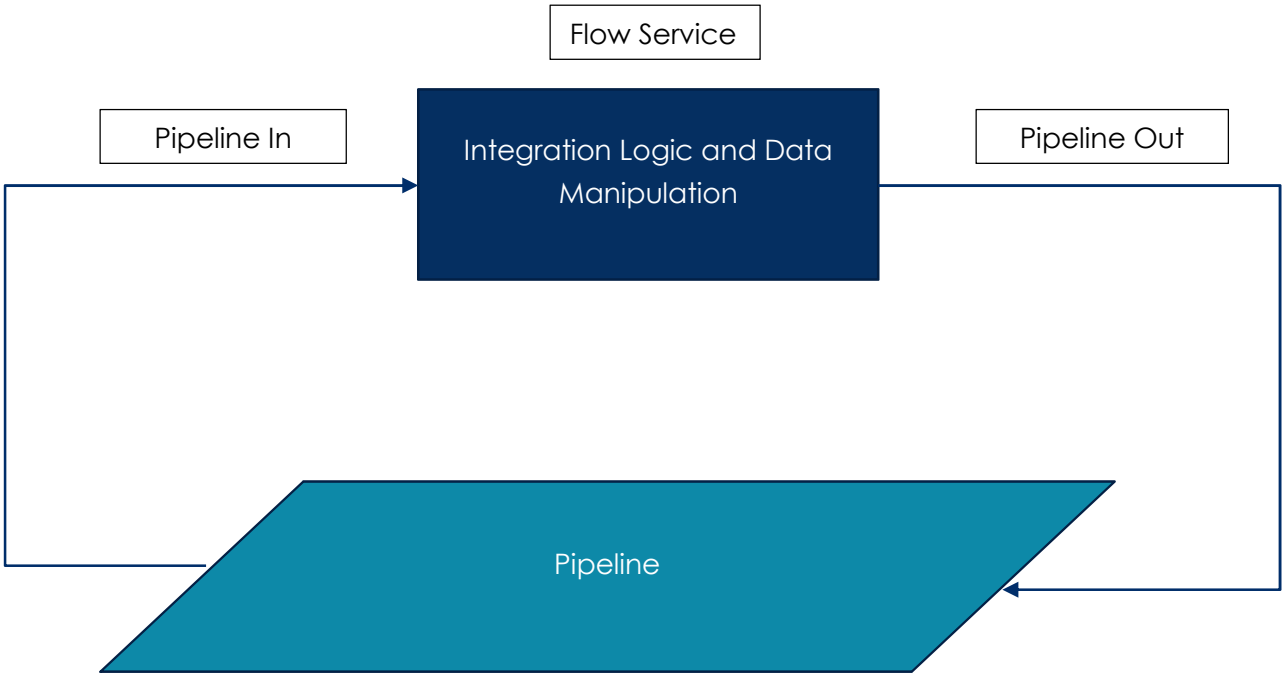
http://www.linkedin.com/in/rankesh

[rankeshk@gmail.com]

Document Name	Version	Release Date
webMethods Flow Service Design and Practice	1.0	04-Jun-2014

Definition: Flow language provides a set of simple but powerful constructs that you use to specify a sequence of actions (steps) that the Integration Server will execute at run time.[2]

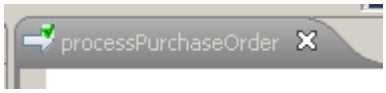
A flow service can be presented as:



Flow service operates on variables/objects stored in pipeline as name-value pair. Pipeline is an IData[3] object which is implementation of Java Hashtable[4]

Suggestions on Flow Service:

1. Name it like, java method.
    a. Flow Service have a meaningful name that succinctly describe its purpose. E.g. processPurchaseOrder

    b. Self-documenting and reducing the need for additional comments.

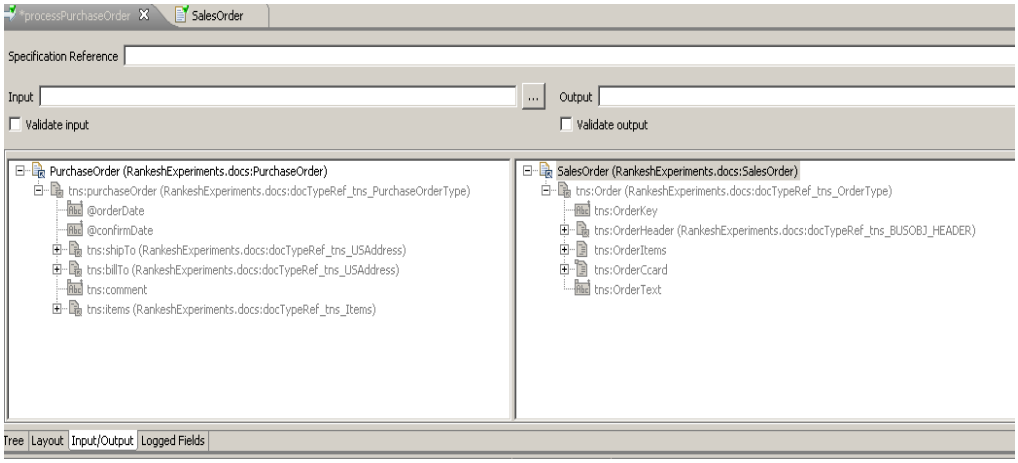
    

    c. Compose service name using mixed case letters, beginning with a lower case letter and starting each subsequent word with an upper case letter.
    d. Begin method names with a strong action verb. E.g. process, map, etc.
    e. Add adjectives if necessary to clarify the noun i.e. convert**To**Date
    f. Do NOT use special characters [exception is underscore], it may cause problem during execution.

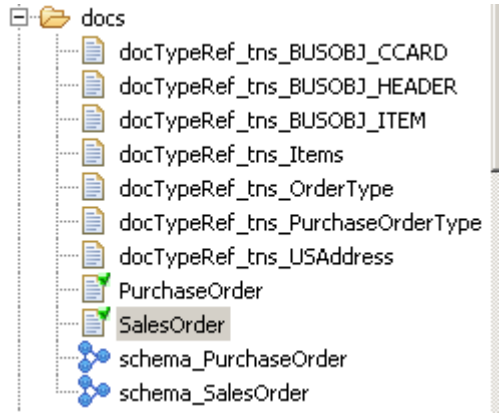
2. Java/C++ service Vs Flow Service
    a. Flow Service is visual, so easy to understand, debug and modify. SAG recommends coding in flow language. Programming language, Java/C++ is preferred where you need to implement a complex logic, multiple operations on large document, or for better throughput, and or loops of 3rd degree of higher.
    b. While writing Java service, generate input output pipeline initialization code with help of designer tool[5].

3. Code for OS-Neutrality
    a. Do not hard code file separator, Instead use *File.separator*

4. Signature of Service
    a. Always define input and output of a service. It eases the implementation.



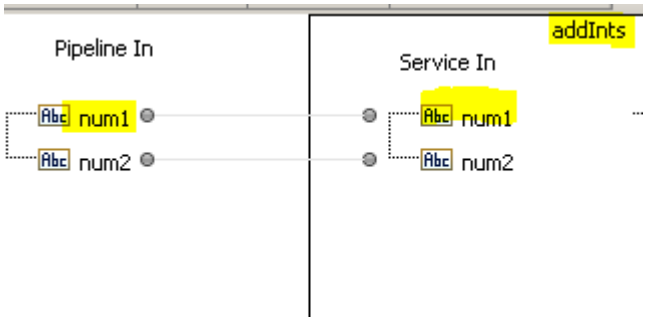
b. Ask connecting system(s) owners to provide the XSD, XML Schema or sample XML to define the input/output document(s).



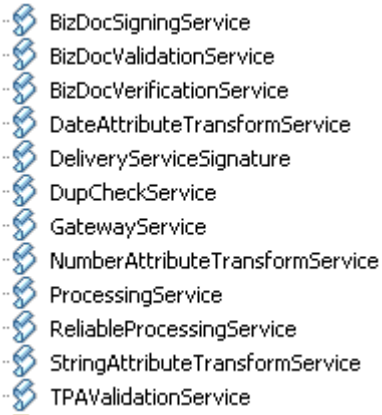
c. Prefer to define one Input and one Output Doc reference. It eases future signature change(s) and mapping(s).

d. If possible, name input and output same for helper services.

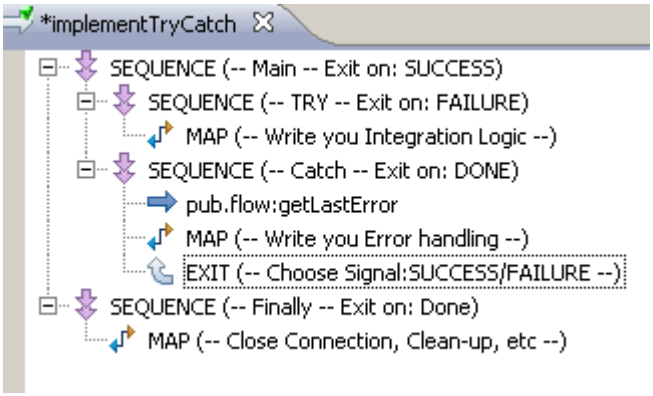
e. Name the variables of helper service intelligently. Take advantage of implicit mapping.



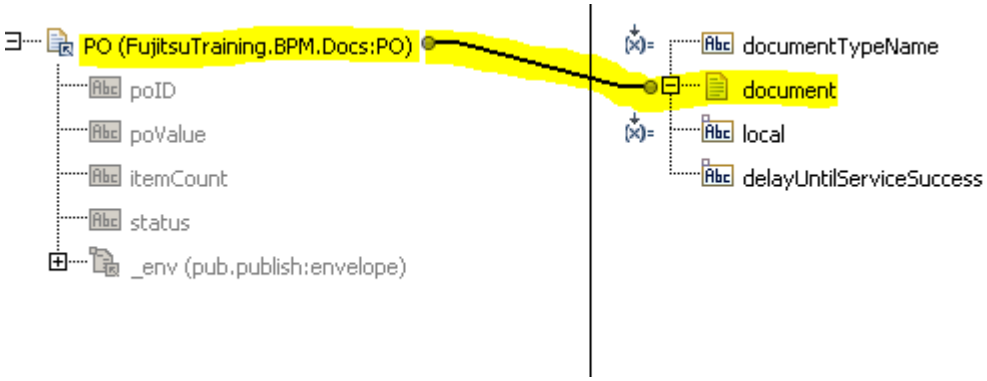
f. TN services force you to use specifications. It helps to build a good ecosystem based on rigid input/output. If you have multiple services with the same input and output requirements, you can point each service to a single specification rather than manually specify individual input and output fields in each service.[6]



5. TOP level service should have a Main-Try-Catch [Additional Sequence is shown for finally block] logic using three sequences as follow:

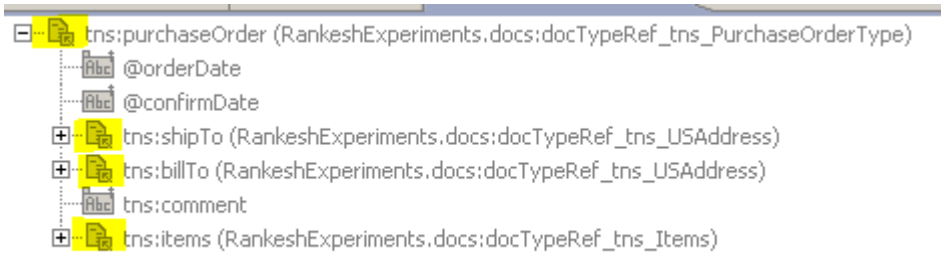


6. Mapping Suggestions
    a. Map docs, not independent variables [inside the doc], whenever possible.



    b. Use Map step to process multiples small unit of work, like string trim, concatenation, date format, etc.

7. Document Structure
    a. Split complex document into sub-documents to maximize reusability and flexibility.

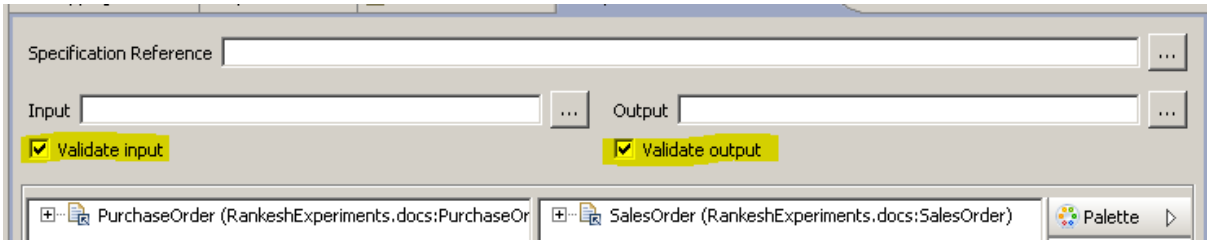


8. Do not put disabled steps in flow, it consume precious CPU time during execution.

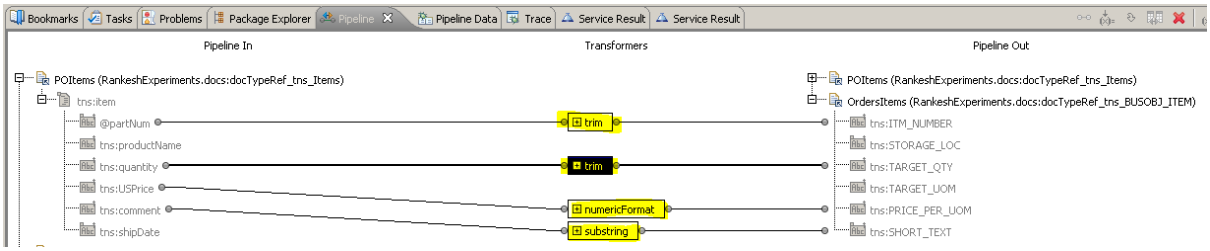
9. Good practice is to define, data constraints and its type.

Property	Value
General	
Name	num1
Data type	String
XML namespace	
Comments	
String display type	Text Field
Pick list choices	
Document reference	
Substitution group	
Constraints	
Required	True
Allow null	True
Allow unspecified fields	True
Content type	
Java wrapper type	UNKNOWN

10. When developing flows for eStandards, IDoc, XSDs Documents it is good to validate input and output.



11. Transformers are services invoked in MAP step in a special way. Multiple transformers can be executed in one MAP step. This is recommended to use when you have a large set of variables under a document and you need to map them independently to output after some processing, as shown below.



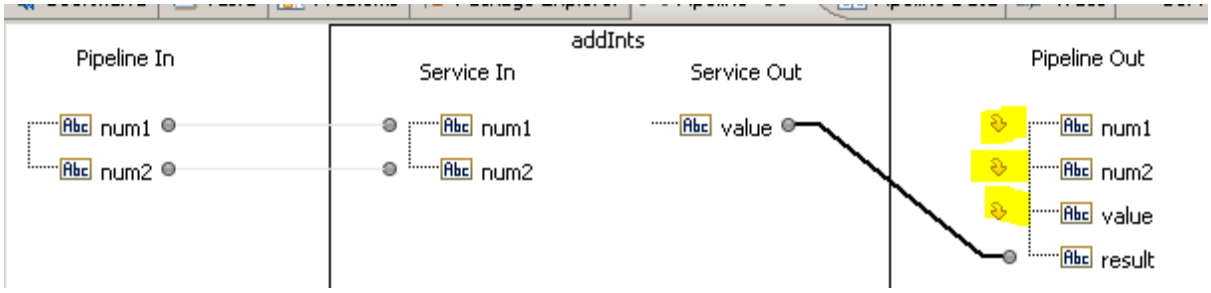
    a. You can initialize a variable in MAP step [preferably] or in any step in Pipeline Out.

12. When using I/O services, make sure you close them. E.g. The readerToString service does not automatically close the reader object. To close the reader, use the pub.io:closeservice[7].

13. Use link properties for conditional mappings.

Property	Value
General	
Evaluate copy condition	True
Copy condition	
Indices	/value --> /result

14. Drop variables which are not required in subsequent steps.



15. Restrict the data to invoked services, by using scope. It passes limited variables to invoked service.

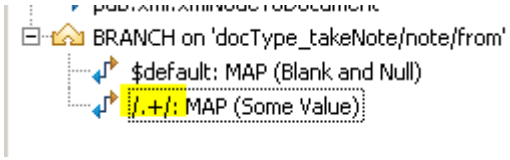
Property	Value
General	
Comments	
Scope	PurchaseOrder/tns:purchaseOrder/tns:items
Timeout	
Label	
Service	RankeshExperiments.services:mapPOItemsToOrdersItems
Validate input	False
Validate output	False

16. Always use a timeout for I/O and network operations. E.g. pub.art.transaction:setTransactionTimeout

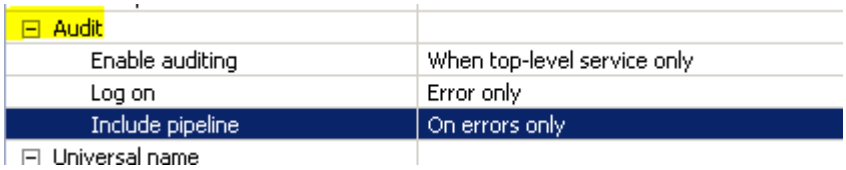
Property	Value
General	
Comments	
Scope	
Timeout	300
Label	
Service	pub.io:readAsString
Validate input	False
Validate output	False

a. Label is very interesting feature. It is used by EXIT and BRANCH step types to accomplish different logic.

17. Regular expression not just shortens your code but also performs better. You can use it in branch, link mapping, and a few String services.

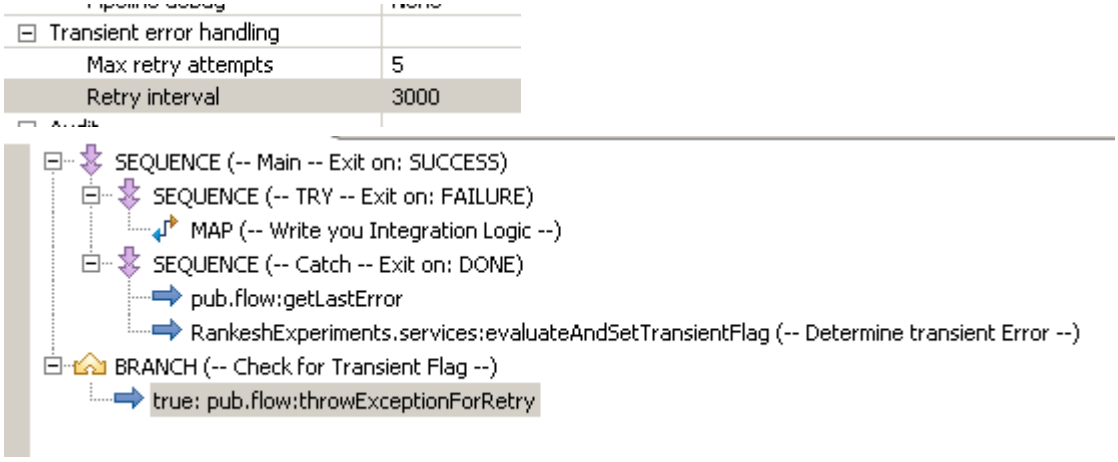


18. Configure TOP level service for auditing. Recommended setting is:



19. Use property template of flow service in designer. Ref: http://techcommunity.softwareag.com/web/guest/techniques-blog/-/blogs/hidden-gems-property-templates-in-webmethods-designer

20. Configure *Transient error handling* and use pub.flow:throwExceptionForRetry service for network related operations like, JDBC operations, file read-write, etc on TOP level service or on Trigger(s).



21. Control run-time of service through property.
    a. It is recommended to put Stateless = True.

Run time	
Stateless	True
Cache results	False
Cache expire	15
Reset cache	
Prefetch	False
Prefetch activation	1
Execution locale	[$null] No Locale Policy
HTTP URL alias	
Pipeline debug	None

b. Caching / Prefetch are available for performance tuning.
c. HTTP URL alias is an excellent option to hide the namespace, put security and ability to change service without changing the end point.
d. Pipeline debug is helpful during development. webMethods flow service is inherently TDD, [Test Driven Development].

22. Developer's best friend is http://techcommunity.softwareag.com/ecosystem/documentation/webmethods/wmsuites/wmsuite9-6/Integration_Server/9-6_Integration_Server_Built-In_Services_Reference.pdf. It contains built-in-services of Integration server, which are used to build your flow service.

23. There are 7 step types[9] [BRANCH, EXIT, INVOKE, LOOP, MAP, REPEAT, SEQUENCE], with help of them you control the flow and write logic in flow service.

24. For DML [SQL] queries, use TRANSACTION TYPE = LOCAL TRANSACTION in JDBC connection and following services[10]:
    a. pub.art.transaction:commitTransaction
    b. pub.art.transaction:rollbackTransaction
    c. pub.art.transaction:setTransactionTimeout
    d. pub.art.transaction:startTransaction

25. Please make effort to put comment in steps and service level comments. It helps in maintaining the good health of service.

Comments

Enter a value for Comments.

Version: 1.0
Date: 28-Apr-2014
Last Modified on: <N/A>
Modified By:Rankesh Kumar
Input(s): num1[Mandatory], num2 [Mandatory], operation [Pre-defined List]
Output(s):result[optional], error [optional]

Version	Date	Modified By	Description
1.0	28-Apr-2014	Rankesh Kumar	A sample calcultor to compute add, subtract, multiply and divide of two numbers.

A SPECIAL NOTE

Thanks to following persons who have taken their time out to review this document and suggested to improve it:

Sachin Mantri

Rakesh Kumar

Vicky Anand

Shradha Gundannavar

REFERENCES

1: Product of Software AG,
http://www.softwareag.com/corporate/products/az/webmethods/default.asp
2: Page 34,
http://techcommunity.softwareag.com/ecosystem/documentation/webmethods/wmsuites/wmsuite9-6/Designer/9-6_Service_Development_Help.pdf
3:
http://techcommunity.softwareag.com/ecosystem/documentation/webmethods/wmsuites/wmsuite9-6/Integration_Server/9-6_Integration_Server_Java_API_Reference/com/wm/data/IData.html
4: http://docs.oracle.com/javase/7/docs/api/java/util/Hashtable.html
5: 9-5-SP1_Service_Development_Help.pdf, Page 308
6: 9-5-SP1_Integration_Server_Built-In_Services_Reference.pdf, page 353
7: 9-5-SP1_Service_Development_Help.pdf, Page 510
8: 9-5-SP1_Service_Development_Help.pdf, Page 961
9: 9-5-SP1_Service_Development_Help.pdf, Page 813
10:
http://techcommunity.softwareag.com/ecosystem/documentation/webmethods/adapters_estandards/Adapters/JDBC/JDBC_6-5/6-5_JDBC_Adapter_Install_and_Users_Guide.pdf, page 234