

# 12. MESSAGERIE

# RAPPEL: LES STYLES D'INTEGRATION D'UNE APPLICATION

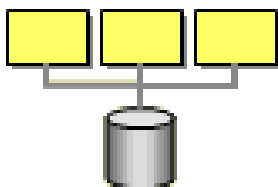
142

*File Transfer*



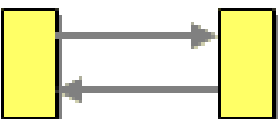
- Applications partagent leurs données, **mais** pas leurs fonctionnalités.

*Shared Database*



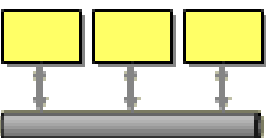
- Applications partagent leurs données, **mais** pas leurs fonctionnalités.

*RPI*

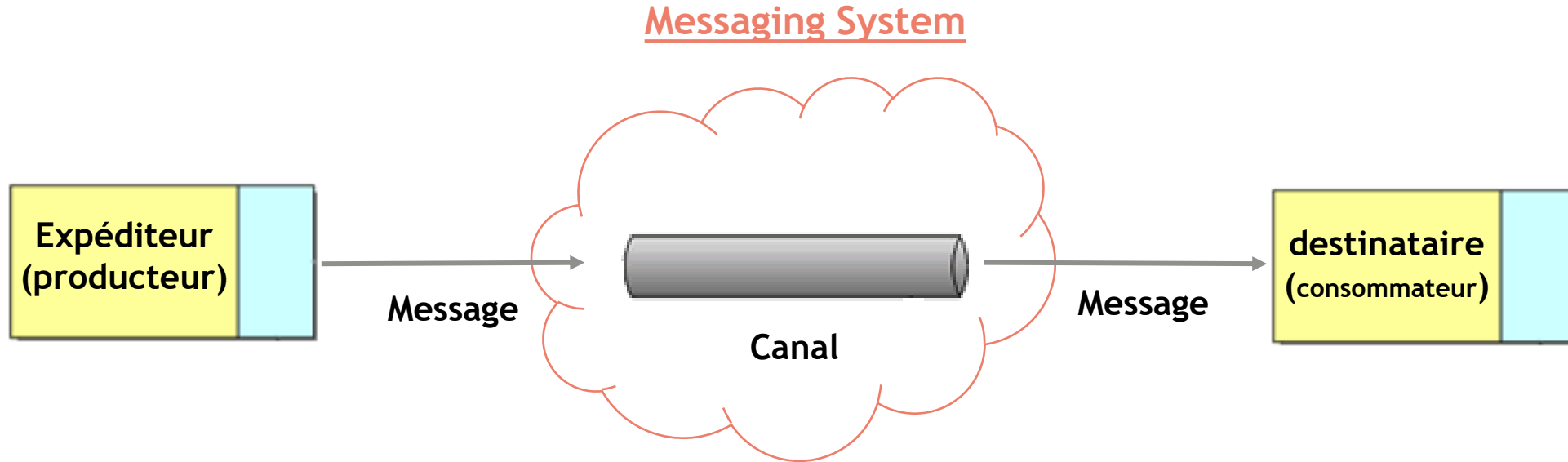


- Applications partagent leurs fonctionnalités, **mais** elles sont étroitement liées dans le processus.

*Messaging*

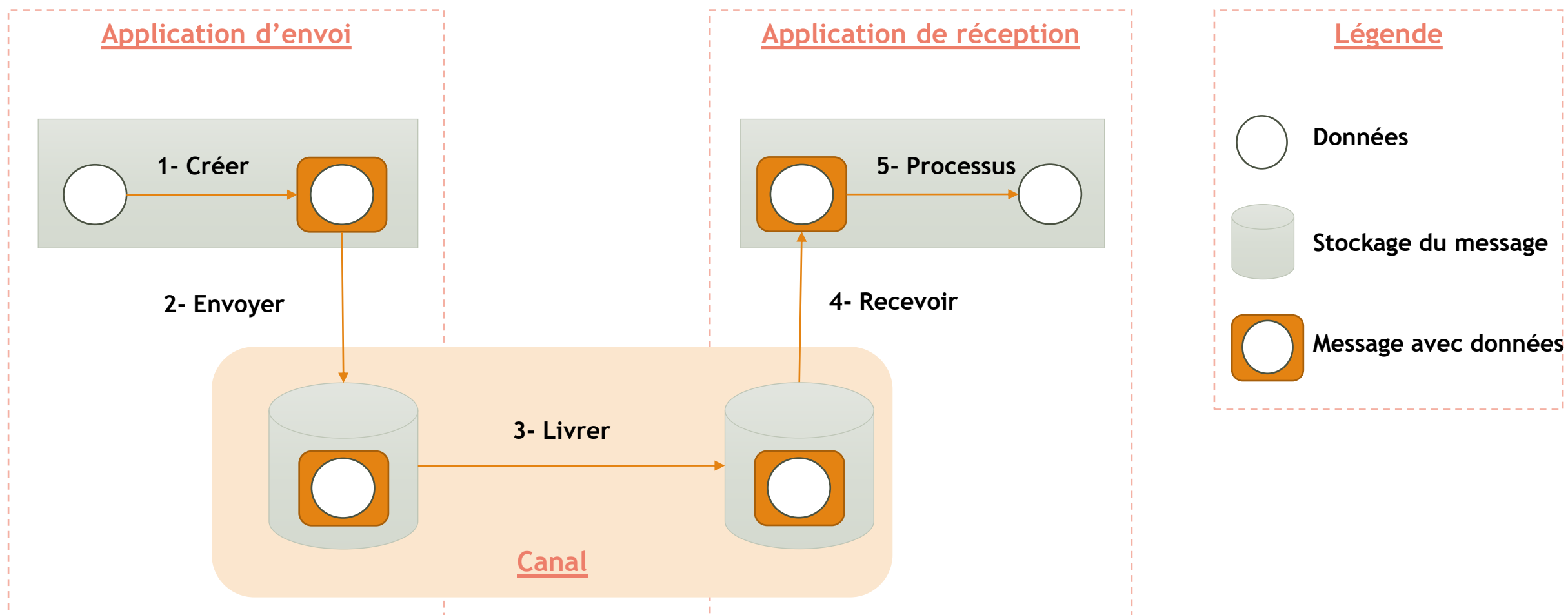


- Applications transfèrent des paquets de données fréquemment, immédiatement, de manière fiable et asynchrone, en utilisant des formats personnalisables.



*La messagerie est une technologie qui permet des communications à haut débit,  
asynchrones,  
communication fiable de programme-à-programme.*

- Un message est transmis en 5 étapes :



# 13 ENTREPRISE INTEGRATION PATTERNS (EIP)

# QU'EST-CE QUE L'IAE?

146

- Est un catalogue de modèles de conception pour le développement de systèmes permettant d'intégrer des logiciels nouveaux et existants dans un environnement professionnel.
- Se concentre sur les modèles de messaging pour l'intégration d'applications d'entreprise (IAE).
- Fournit 65 modèles de conception.

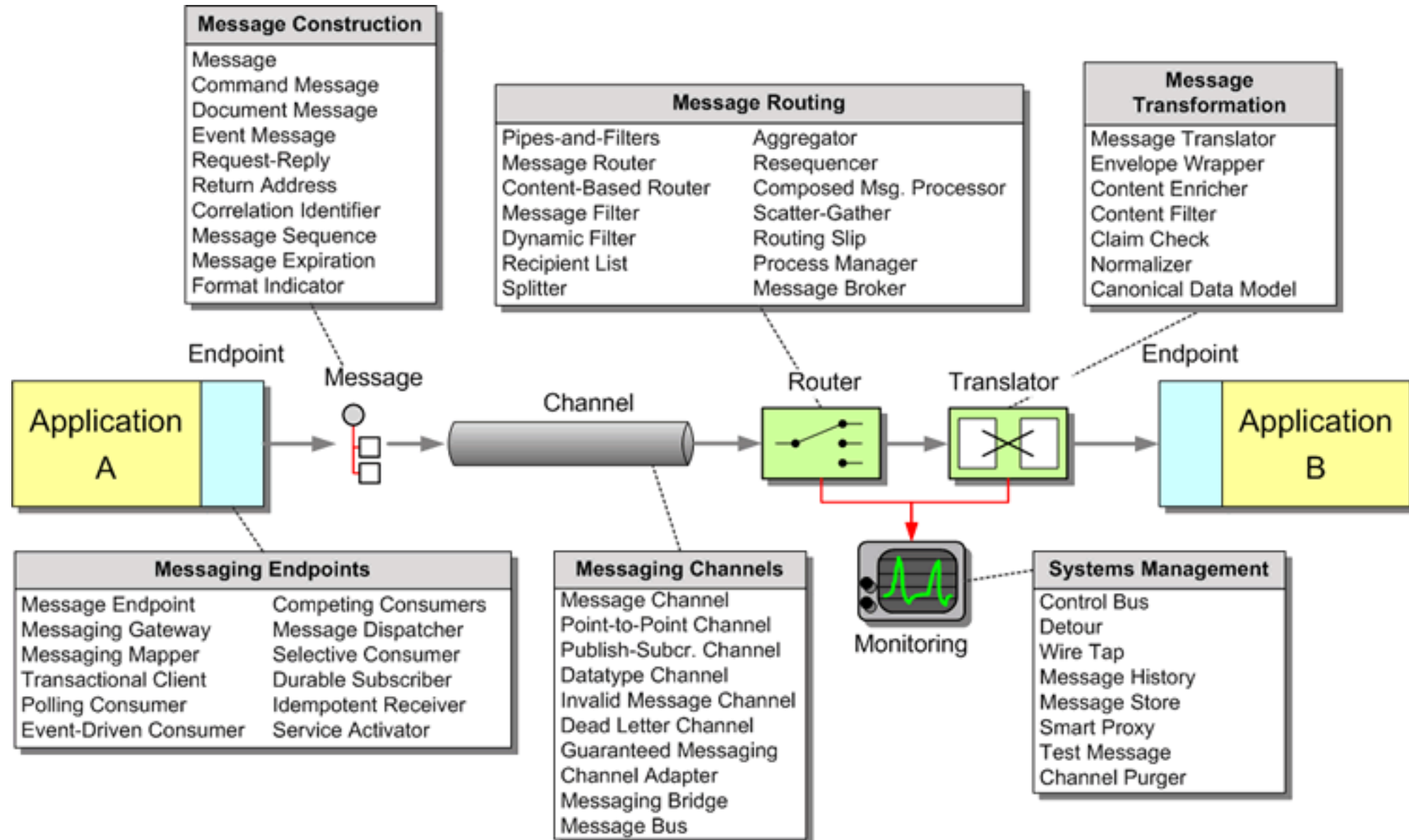


# QU'EST-CE QUE LE MODÈLE DE CONCEPTION ?

147

- Est une solution générale à un problème de conception, récurrent dans de nombreux projets.
- Décrit le problème et sa solution proposée

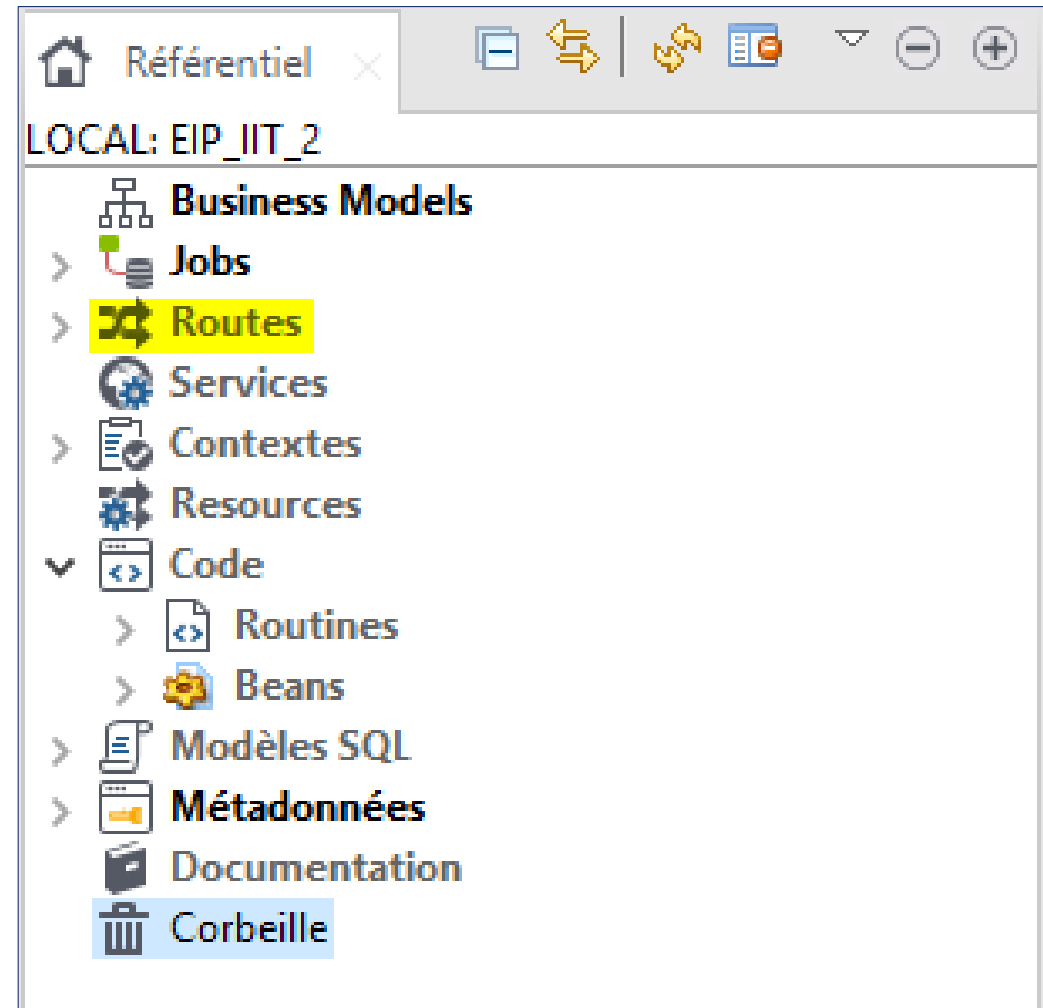






# 14 TALEND :: DÉVELOPPEMENT DES ROUTES

- Pour implementer les modèles d'intégration d'entreprise (EIP) avec Talend, on doit créer des Routes.



- **JOB**
  - Composants connectés pour configurer et exécuter les processus de gestion des flux de données.
- **Route**
  - Composants connectés pour exécuter des règles de routage.
  - Définir comment les messages sont déplacés d'un service ou d'un endpoint à un autre.
  - Basé sur le framework Apache Camel.

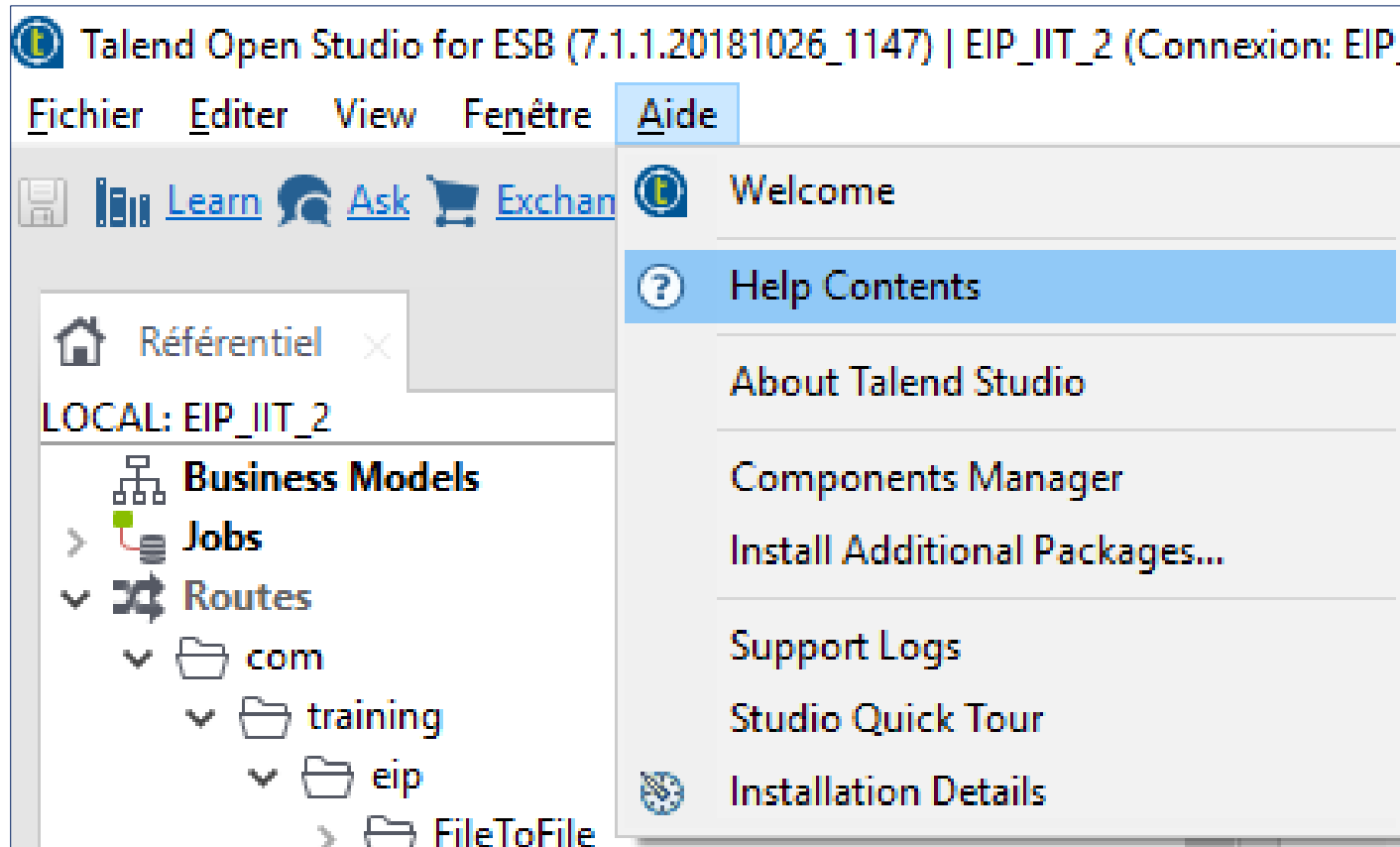
- Apache Camel est un framework d'intégration open-source basé sur des modèles d'intégration d'entreprise connus.
- Une compréhension approfondie d'Apache Camel n'est pas nécessaire pour le moment.
- Talend ESB dissimule une grande partie des complexités de Camel.



# NOTE :: TALEND DOCUMENTATION

153

- Vous pouvez accéder à la documentation Talend en sélectionnant le menu "**Help Contents**" dans l'onglet "**Help**".



# NOTE :: TALEND DOCUMENTATION

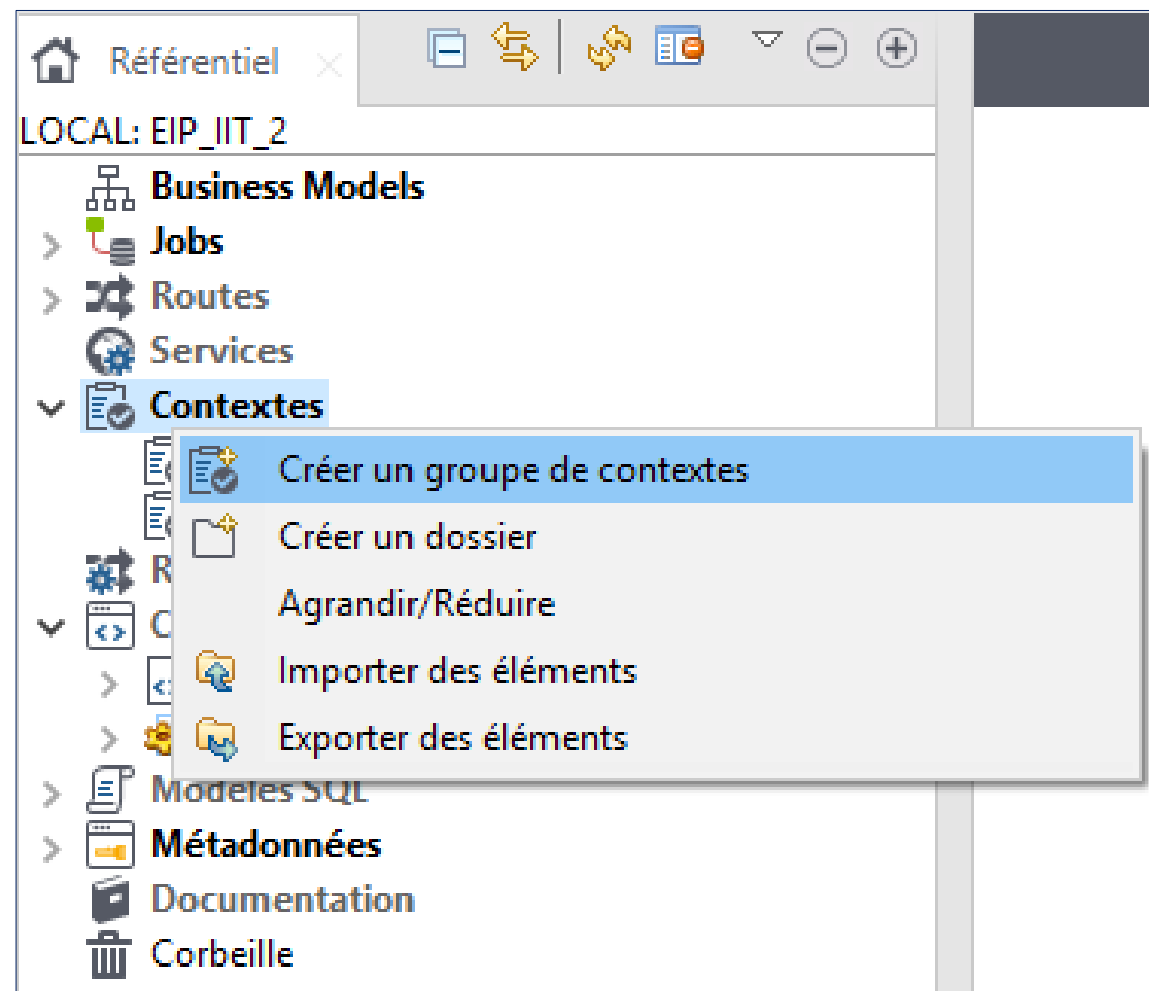
154

- Vous pouvez également sélectionner un composant spécifique et appuyer sur "**F1**" pour obtenir la documentation appropriée de ce composant.



- **Groupe de Contexte**
  - Groupe de Variables de Contexte liées.
- **Variable de contexte**
  - Permet l'injection de valeurs extérieures de la Route, sur base de l'environnement de contexte.
- **Environnement de contexte :**
  - Quelques exemples typiques : DEV, TEST, PRODUCTION ...

- Objectif :
  - Utiliser un groupe de contexte nommé “FileSystemVariables” pour sauvegarder le chemin d'accès du dossier source qui va être utilisé après dans les différentes Routes.





# TALEND :: ADD GLOBAL CONTEXT

157

- Étape 1: indiquer le Nom, l'Objectif et la Description du groupe de contexte.

Créer / Editer un groupe de contextes

**Etape 1 sur 2**

Toute information requise

Nom: FileSystemVariables

Objectif: Route component file system variables

Description: Directory paths of files

Créé par: user@talend.com

Verrouillé par: user@talend.com

Version: 0.1 [M] [m]

Statut: [v]

Chemin d'accès:

# TALEND :: ADD GLOBAL CONTEXT

158

- Étape 2: Ajouter la variable de contexte “dir\_path” et cliquer sur le bouton “Finish”.

Créer / Editer un groupe de contextes

**Etape 2 sur 2**  
Définir les contextes, les variables et les valeurs

	Name	Type	Comment	Default Value
1	dir_path	Directory ▼		C:/Users/Ibtissem/Desktop/cours IIT/studentFiles/routes/

+ ✖

Default context environment Default ▼

< Back Next > **Finish** Cancel

# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (CFILE)

159

- Objectif:
  - Développer une Route pour copier un fichier d'un répertoire vers un autre répertoire.

1- Nouveau fichier sauvegardé sous le dossier source



Répertoire source

2- Copier le fichier dans le dossier de destination



Répertoire de destination

# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (cFile)

160



on va utiliser le composant **cFile**.



Fournit un accès aux systèmes de fichiers, permettant aux fichiers d'être traités par d'autres composants afin d'être sauvegardés sur le disque.

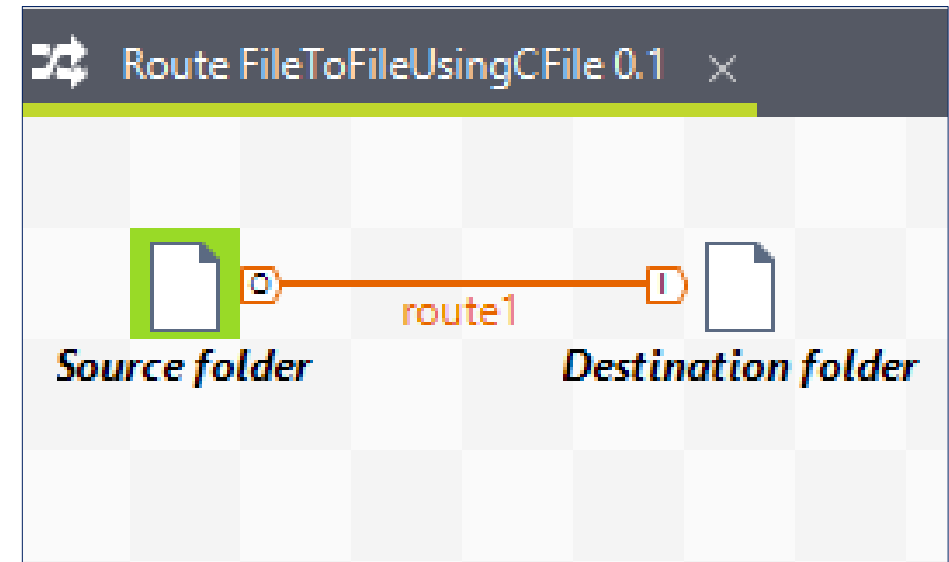
The screenshot shows the Talend Designer interface with the 'cFile\_1' component selected. The 'Paramètres simples' (Simple Parameters) tab is active, displaying the following configuration:

- Chemin d'accès :** ""
- Paramètres**
  - ☒ noop
  - ☐ flatten
  - ☒ autoCreate
- Taille de la mémoire tampon (ko)** : "128"
- Encodage** : CUSTOM (dropdown menu) and "" (text field)
- fileName** : ""

# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (cFile)

161

- Étape 1:
  - Créer une nouvelle "Route" appelée "FileToFileUsingCFile".
  - Ajouter 2 composants "cFile" appelés "Source folder" and "Destination folder".
  - Lier le composant "Source folder" avec le composant "Destination folder".



# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (cFile)

162

- Étape 2:
  - Configurer le composant "Source folder".

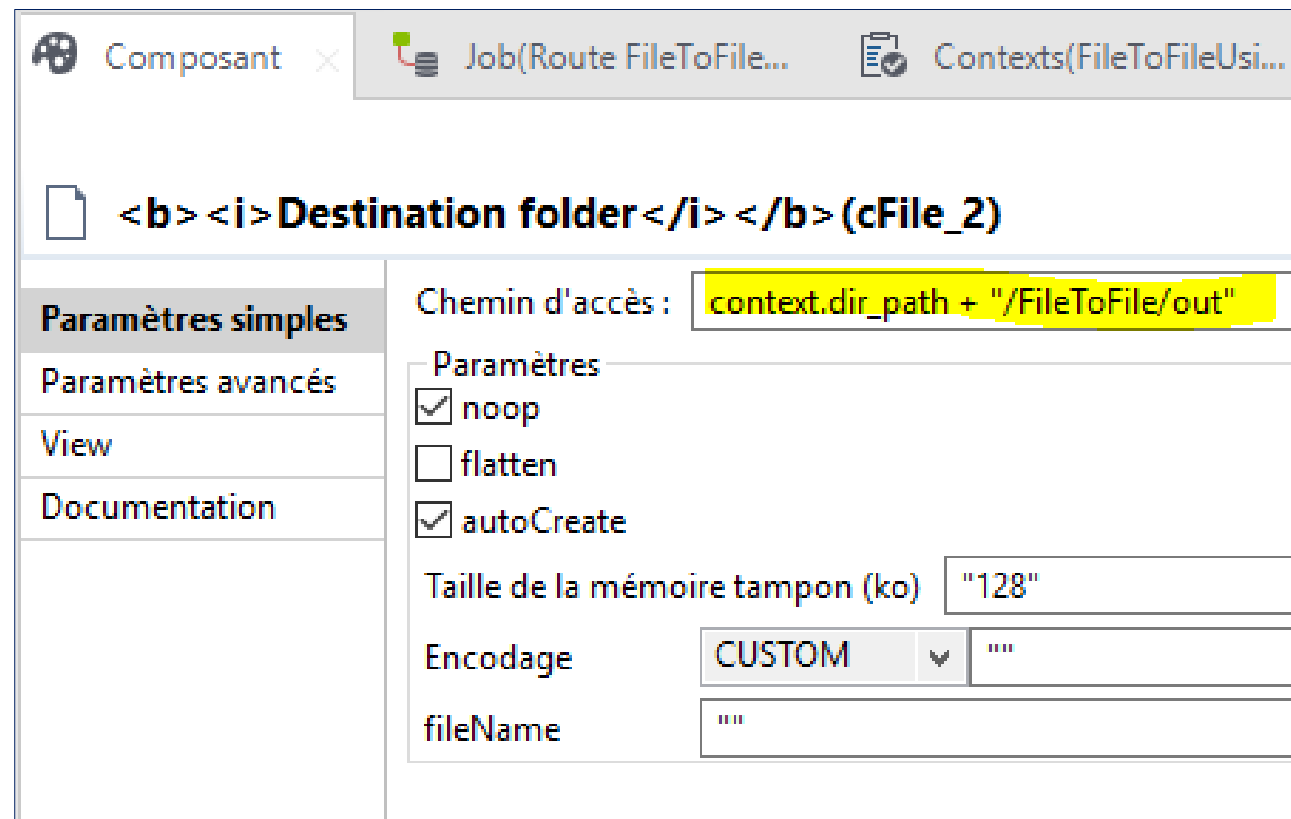
The screenshot shows the Talend Studio configuration window for the 'Source folder' component. The window has a title bar with 'Composant' and a close button. Below the title bar, there are two tabs: 'Job(Route FileToFile...' and 'Contexts(FileToFileUsi...'. The main area displays the component name '<b><i>Source folder</i></b> (cFile\_1)' with a document icon. On the left, there is a sidebar with 'Paramètres simples' selected, and other options like 'Paramètres avancés', 'View', and 'Documentation'. The main configuration area shows the 'Chemin d'accès' field with the value 'context.dir\_path + "/FileToFile/in"'. Below this, there are checkboxes for 'noop' (checked), 'flatten' (unchecked), and 'autoCreate' (checked). The 'Taille de la mémoire tampon (ko)' field is set to '128'. The 'Encodage' dropdown is set to 'CUSTOM' with a value of '""'. The 'fileName' field is set to '"market\_values.txt"'. The 'Paramètres' section is expanded, showing these settings.

<b><i>Source folder</i></b> (cFile_1)	
<b>Paramètres simples</b>	Chemin d'accès : <code>context.dir_path + "/FileToFile/in"</code>
Paramètres avancés	Paramètres
View	<input checked="" type="checkbox"/> noop
Documentation	<input type="checkbox"/> flatten
	<input checked="" type="checkbox"/> autoCreate
	Taille de la mémoire tampon (ko) : <code>"128"</code>
	Encodage : <code>CUSTOM</code> <code>""</code>
	fileName : <code>"market_values.txt"</code>

# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (cFile)

163

- Étape 3:
  - Configurer le composant "Destination folder".



Composant × Job(Route FileToFile... Contexts(FileToFileUsi...

**<b><i>Destination folder</i></b> (cFile\_2)**

**Paramètres simples**

Paramètres avancés

View

Documentation

Chemin d'accès : `context.dir_path + "/FileToFile/out"`

Paramètres

☒ noop

☐ flatten

☒ autoCreate

Taille de la mémoire tampon (ko) "128"

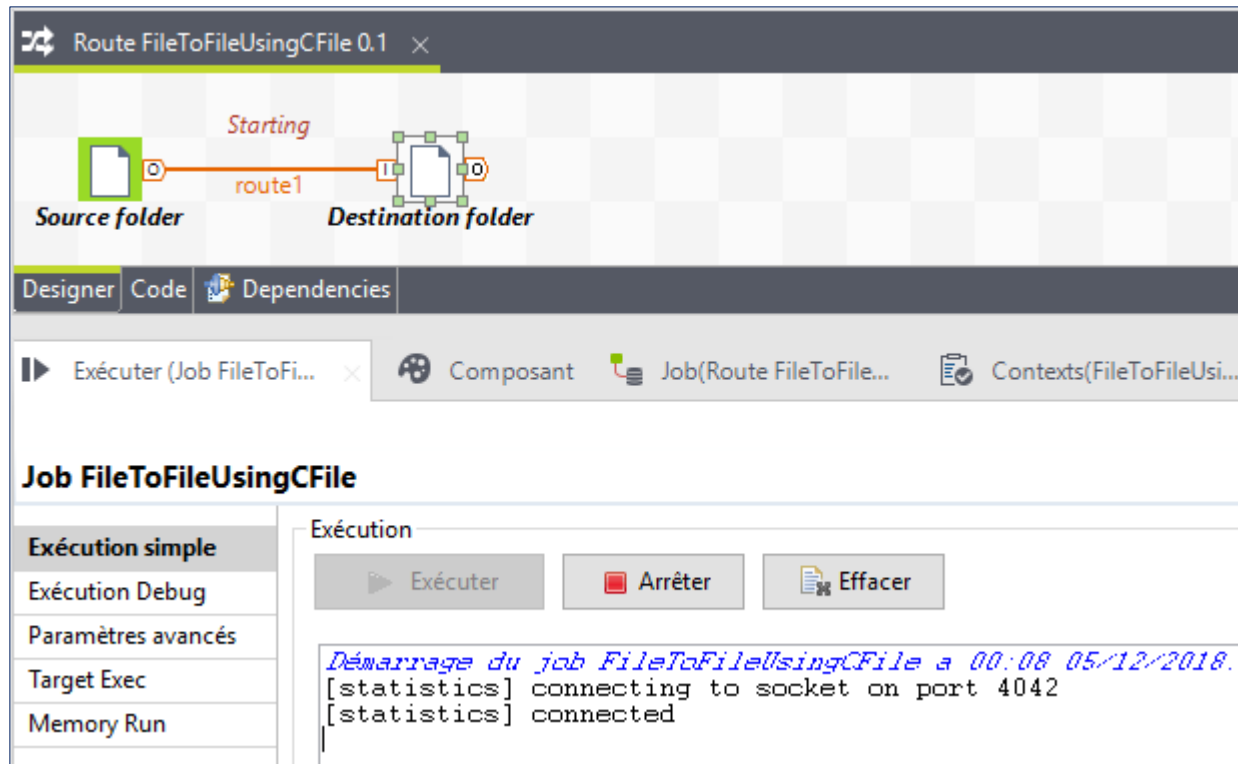
Encodage CUSTOM ▾ ""

fileName ""

# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (CFILE)

164

- Étape 4:
  - Exécuter la Route et tester the comportement attendu,





# TALEND EXEMPLE 1 :: ENVOIE D'UN FICHIER (CFILE)

165

- Par défaut, noop et autoCreate sont sélectionnés ::  
Ces paramètres signifient :
  - noop :: : Conserve le fichier dans le répertoire d'origine après sa lecture.
  - autoCreate :: Crée automatiquement les répertoires spécifiés dans le champ Path s'ils n'existent pas..
  - Encoding :: Spécifie comment les fichiers lus par ce composant sont encodés.
  - fileName :: Nom du fichier à lire. S'il est vide (""), tous les fichiers du chemin d'accès sont lus.





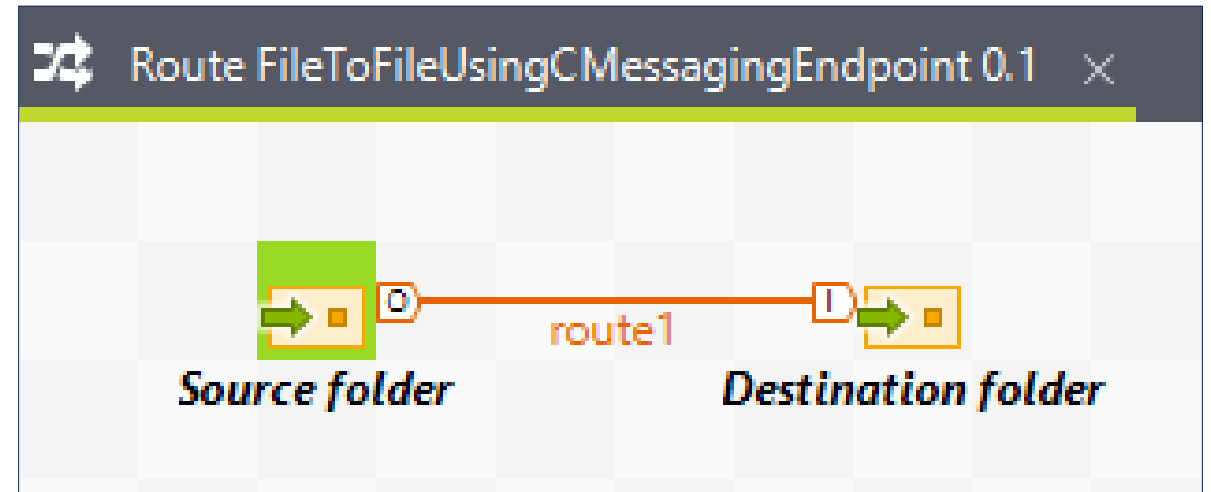
On va utiliser le composant **cMessagingEndpoint**.



Un composant générique qui peut prendre plusieurs formats Apache Camel différents tels que fichier, base de données ou protocole (FTP, HTTP).

Composant ×		
▶ Exécuter (Job test)    📄 Job(Route test 0.1)    📋 Contexts(test)    ⚙️ spring    ⓘ Help		
➡ <b>cMessagingEndpoint_1</b>		
Paramètres simples	URI	""
Paramètres avancés		
View		
Documentation		

- Étape 1:
  - Créer une nouvelle "Route" appelée "FileToFileUsingCMessagingEndpoint".
  - Ajouter 2 composants "cMessagingEndpoint" appelés "Source folder" et "Destination folder".
  - Lier le composant "Source folder" avec "Destination folder".



# TALEND EXEMPLE 2 :: ENVOIE D'UN FICHIER (cMESSAGINGENDPOINT)

168

- Étape 2:
  - Configurer le composant "Source folder".

The screenshot displays the Talend Studio interface for configuring a route. The top bar shows the route name: "Route FileToFileUsingCMessagingEndpoint 0.1". The main workspace shows a visual flow from a "Source folder" component to a "Destination folder" component, connected by a line labeled "route1". Below the workspace, the "Designer" tab is active, showing the configuration for the "Source folder" component. The configuration panel has tabs for "Composant", "Exécuter (Job FileT...", "Routes(FileToFileU...", and "Contexts(FileToFile...". The "Composant" tab is selected, showing the component name "<b><i>Source folder</i></b> (cMessagingEndpoint\_1)". Below this, the "Paramètres simples" section is expanded, showing the "URI" parameter set to "file://" + context.dir\_path + "/FileToFile/MessagingEndpoints/in". The "Paramètres avancés", "View", and "Documentation" sections are collapsed.

Paramètres simples	URI
Paramètres avancés	"file://" + context.dir_path + "/FileToFile/MessagingEndpoints/in"
View	
Documentation	

# TALEND EXEMPLE 2 :: ENVOIE D'UN FICHIER (cMESSAGINGENDPOINT)

169

- Étape 3:
  - Configurer le composant "Destination folder".

The screenshot shows the Talend Designer interface for a route named "Route FileToFileUsingCMessagingEndpoint 0.1". The route consists of two components: "Source folder" and "Destination folder", connected by a link labeled "route1". The "Destination folder" component is highlighted, and its configuration is shown in the bottom panel.

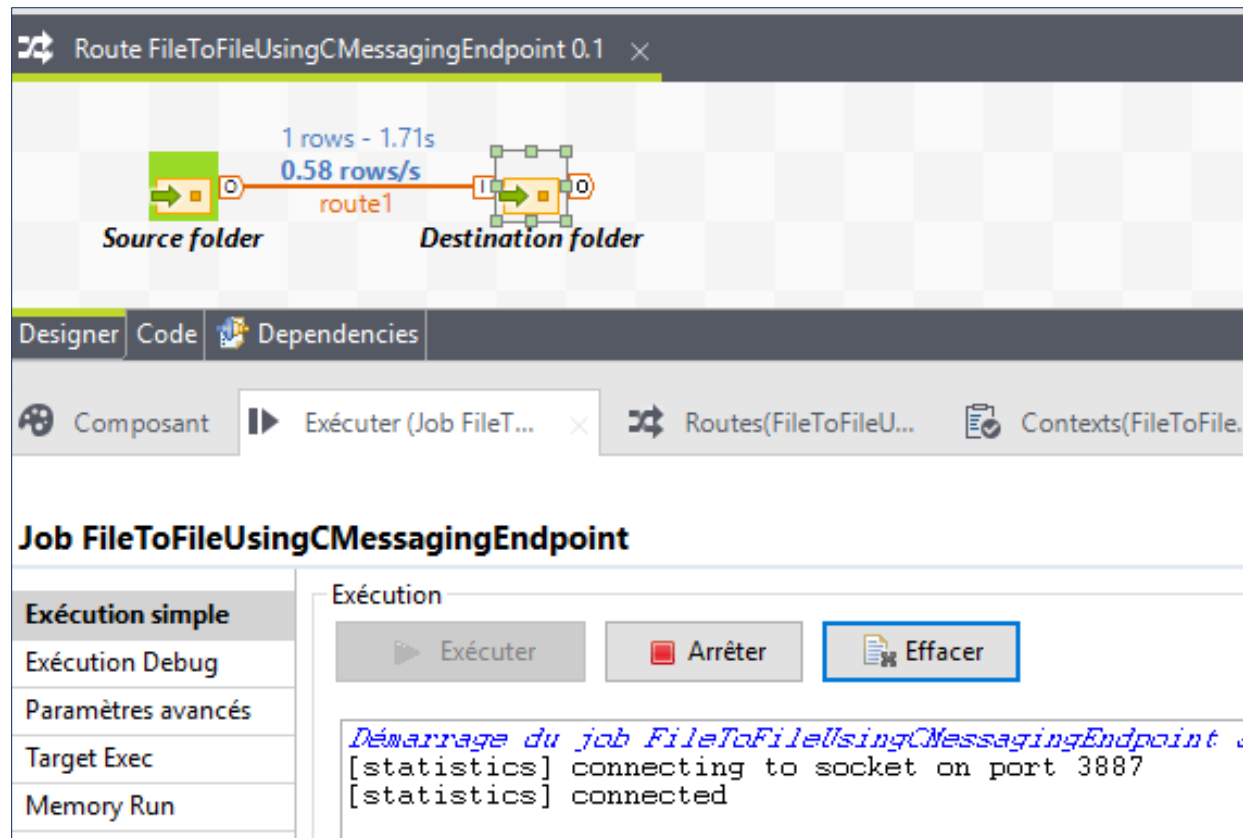
The bottom panel displays the configuration for the "Destination folder" component (cMessagingEndpoint\_2). The "Paramètres simples" (Simple parameters) tab is active, showing the "URI" parameter set to the following value:

```
"file://" + context.dir_path + "/FileToFile/MessagingEndpoints/out"
```

# TALEND EXEMPLE 2 :: ENVOIE D'UN FICHIER (CMessagingEndpoint)

170

- Étape 4:
  - Exécuter la Route et tester le comportement.



The screenshot displays the Talend Studio interface for a job named "Route FileToFileUsingCMessagingEndpoint 0.1". The job design is visible in the Designer tab, showing a "Source folder" component connected to a "Destination folder" component via a "route1" connector. The connector displays statistics: "1 rows - 1.71s" and "0.58 rows/s". Below the design, the "Job FileToFileUsingCMessagingEndpoint" section is active, showing the "Exécution simple" (Simple Execution) tab. This tab includes buttons for "Exécuter" (Execute), "Arrêter" (Stop), and "Effacer" (Clear). The "Exécution" (Execution) log shows the following messages:

```
Démarrage du job FileToFileUsingCMessagingEndpoint 0.1
[statistics] connecting to socket on port 3887
[statistics] connected
```



- Goal:

- Mettre à jour la Route déjà développée pour qu'elle accepte un fichier portant un nom spécifique.



On doit utiliser l'option "**fileName**" dans l'URI du composant "**cMessagingEndpoint**".

```
"file://" + context.dir_path + "/FileToFile/MessagingEndpoints/in?fileName=market_values.txt"
```



Consultez le lien suivant pour plus de détails ou d'autres types de configurations.

<http://camel.apache.org/file2.html>

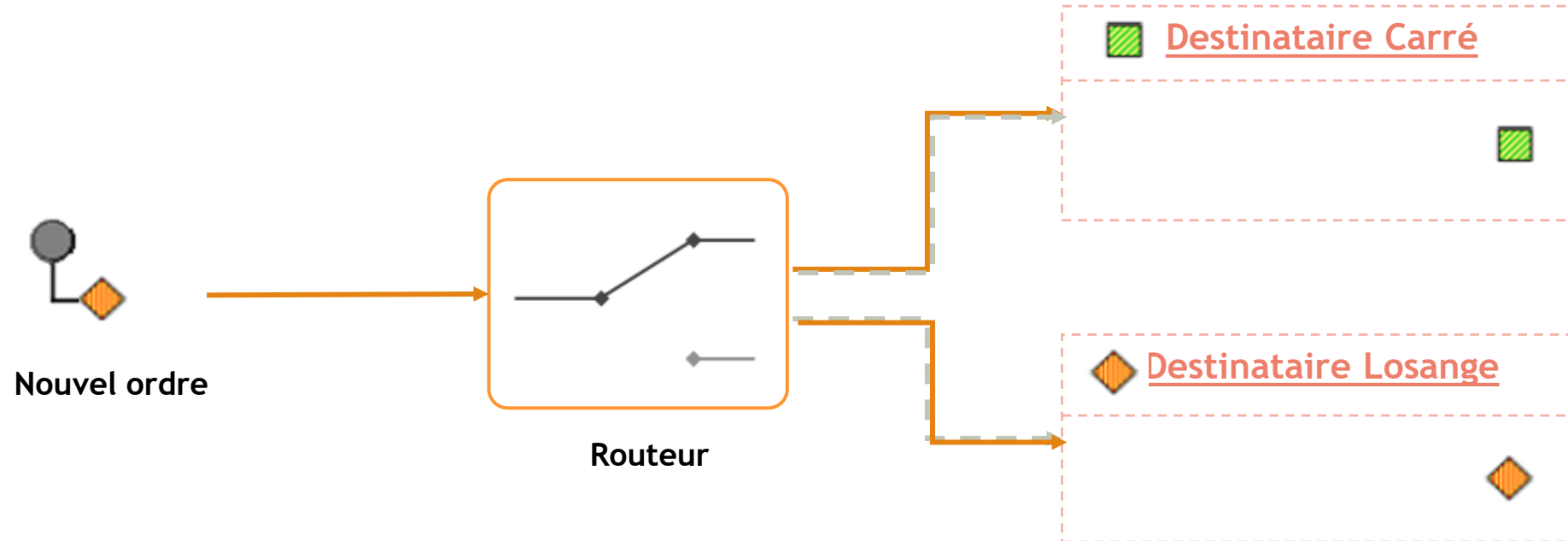
<http://camel.apache.org/components.html>

## Exécution:

- Mettre à jour la Route déjà développée pour prendre en charge un nom de fichier spécifique.
- Exécuter la route.







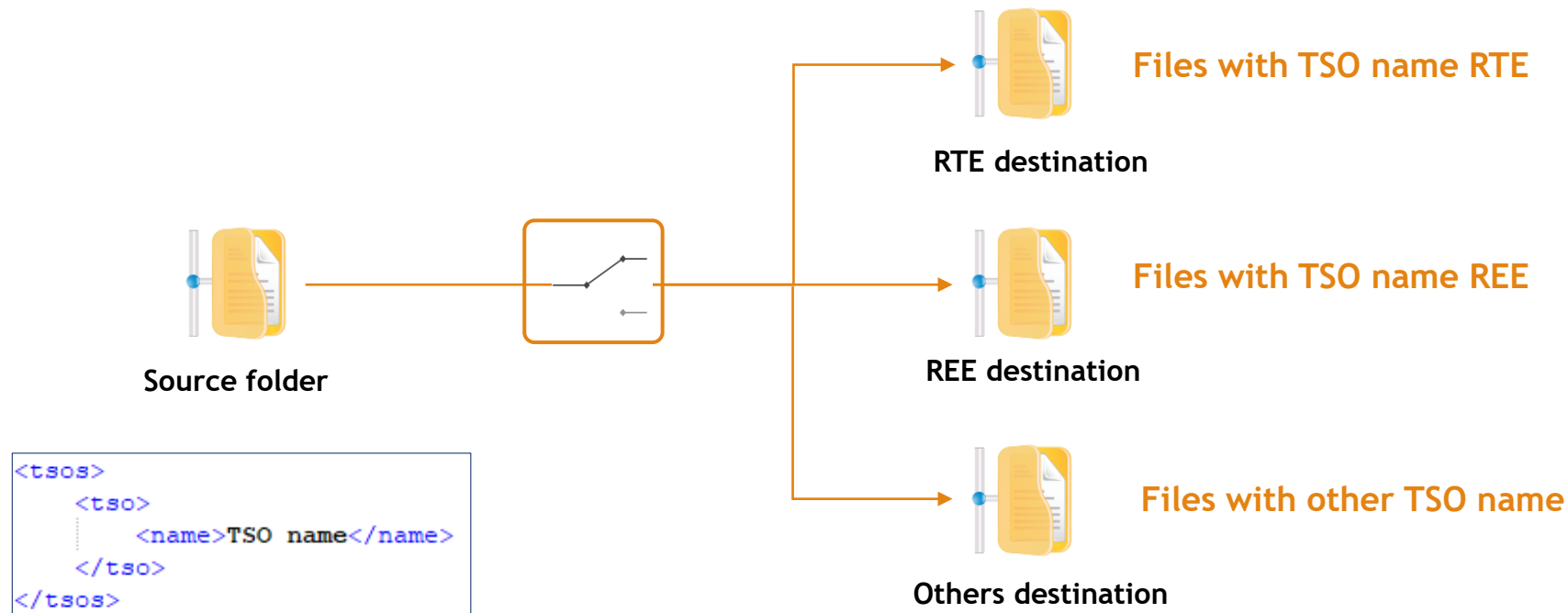
*Utiliser un routeur basé sur le contenu pour acheminer chaque message au bon destinataire en fonction du contenu du message.*

# TALEND EXEMPLE 3 :: CONTENT-BASED ROUTING

174

- Objectif:

- Développer une Route qui permet de copier un fichier XML à partir d'un dossier vers un autre dossier basé sur le nom du TSO (Transmission System Operator) mentionné dans le fichier source (XML).



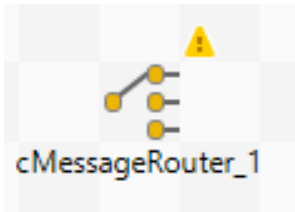
# TALEND EXEMPLE 3 :: CONTENT-BASED ROUTING

175



Nous allons utiliser les 2 composants suivants :

- cFile
- cMessageRouter

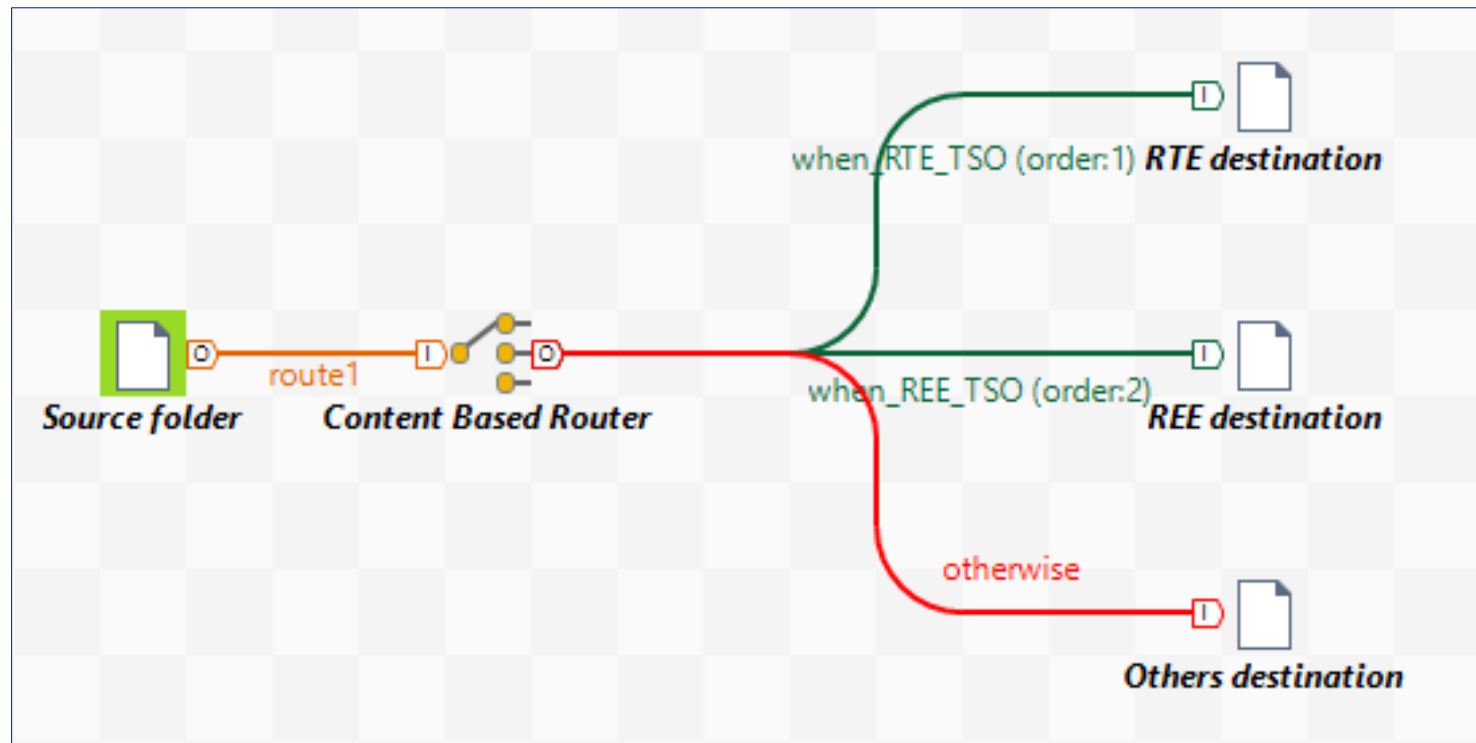


Crée différents canaux pour chaque type de message filtré en fonction des conditions spécifiées afin que les messages puissent être traités ultérieurement de manière plus précise dans chaque nouveau canal.

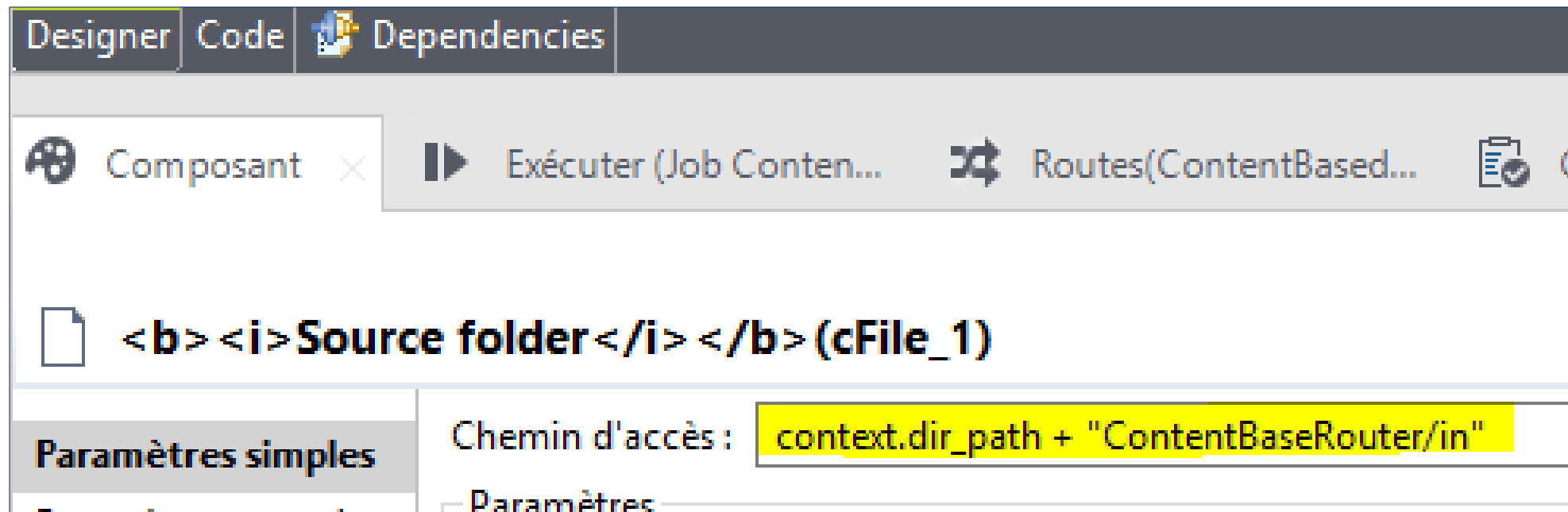
# TALEND EXEMPLE 3 :: CONTENT-BASED ROUTING

176

- Étape 1:
  - Créer une nouvelle "Route" appelée "ContentBasedRouter".
  - Ajouter les composants "cFile" et "cMessageRouter" comme décrit dans figure ci-dessous :



- Étape 2:
  - Configurer le composant "cFile" (source folder).



# TALEND EXEMPLE 3 :: CONTENT-BASED ROUTING


178

- Step 3:
  - Configurer les triggers “when”.

Composant x Exécuter (Job Conten... Routes(ContentBased...

⇒ **when\_RTE\_TSO**

Paramètres simples	Type	xpath	<input type="checkbox"/> Append endChoice()
Paramètres avancés	Condition	"/tsos/tso/name='RTE' "	

Composant x Exécuter (Job Conten... Routes(ContentBased... 


⇒ **when\_REE\_TSO**


Paramètres simples	Type	xpath	<input type="checkbox"/> Append endChoice()
Paramètres avancés	Condition	"/tsos/tso/name='REE' "	


# TALEND EXEMPLE 3 :: CONTENT-BASED ROUTING

179

- Étape 4:
  - Configurer les composants "cFile" (destination folders)

 <b>&lt;b&gt;&lt;i&gt;REE destination&lt;/i&gt;&lt;/b&gt; (cFile_3)</b>	
<b>Paramètres simples</b>	Chemin d'accès : <code>context.dir_path + "ContentBaseRouter/out/REE"</code>
Paramètres avancés	Paramètres
View	<input checked="" type="checkbox"/> noop
Documentation	<input type="checkbox"/> flatten
	<input checked="" type="checkbox"/> autoCreate
	Taille de la mémoire tampon (ko) <input type="text" value="128"/>
	Encodage <input type="text" value="CUSTOM"/> <input type="text" value=""/>
	fileName <input type="text" value=""/>

 <b>&lt;b&gt;&lt;i&gt;Others destination&lt;/i&gt;&lt;/b&gt; (cFile_5)</b>	
<b>Paramètres simples</b>	Chemin d'accès : <code>context.dir_path + "ContentBaseRouter/out/Others"</code>
Paramètres avancés	Paramètres
View	<input checked="" type="checkbox"/> noop
Documentation	<input type="checkbox"/> flatten
	<input checked="" type="checkbox"/> autoCreate
	Taille de la mémoire tampon (ko) <input type="text" value="128"/>
	Encodage <input type="text" value="CUSTOM"/> <input type="text" value=""/>
	fileName <input type="text" value=""/>

 <b>&lt;b&gt;&lt;i&gt;RTE destination&lt;/i&gt;&lt;/b&gt; (cFile_2)</b>	
<b>Paramètres simples</b>	Chemin d'accès : <code>context.dir_path + "ContentBaseRouter/out/RTE"</code>
Paramètres avancés	Paramètres
View	<input checked="" type="checkbox"/> noop
Documentation	<input type="checkbox"/> flatten
	<input checked="" type="checkbox"/> autoCreate
	Taille de la mémoire tampon (ko) <input type="text" value="128"/>
	Encodage <input type="text" value="CUSTOM"/> <input type="text" value=""/>
	fileName <input type="text" value=""/>

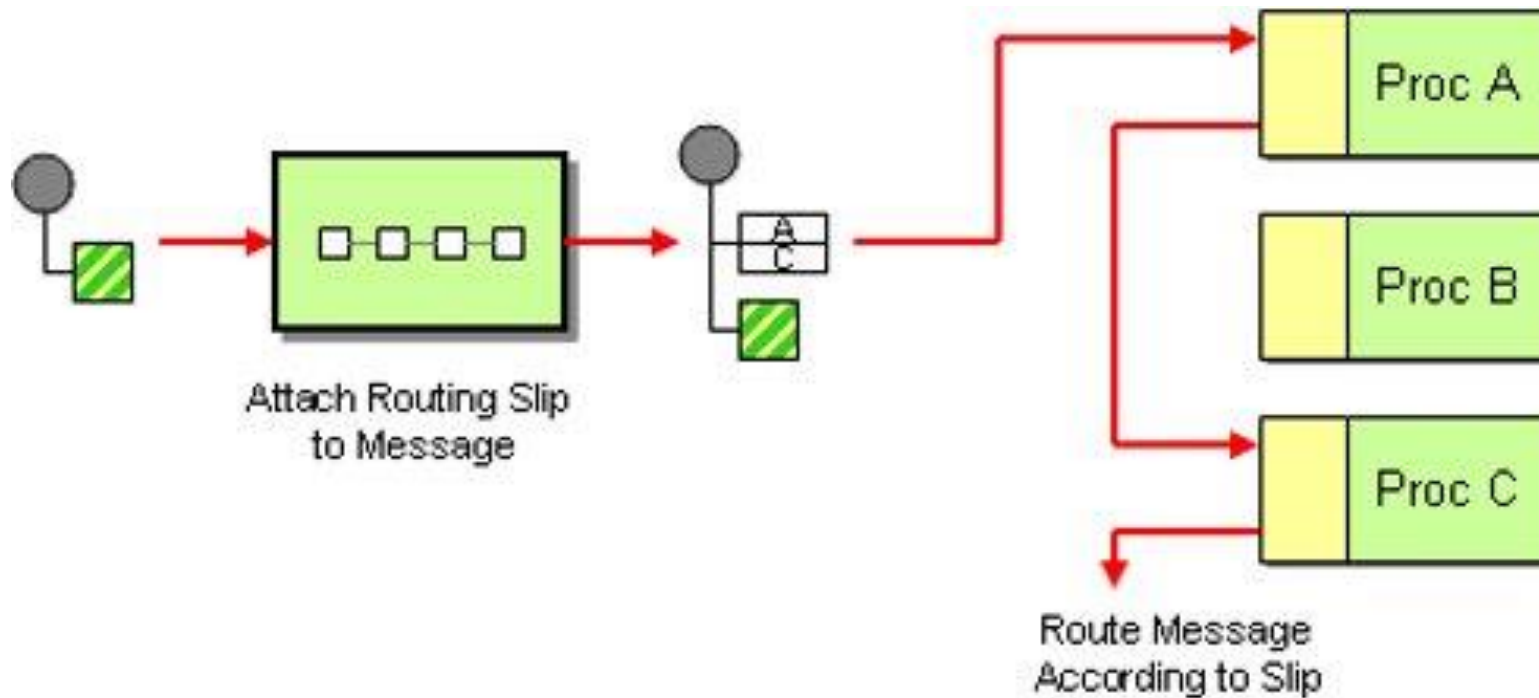
## Éxecution:

- Éxecuter la Route et tester le comportement en utilisant des différents fichiers sources pour les différents TSOs.





Comment acheminer un message de manière consécutive à travers une série d'étapes de traitement lorsque la séquence des étapes n'est pas connue au moment de la conception et peut varier pour chaque message



- Insérer au début du processus un composant spécial qui calcule la liste des étapes nécessaires pour chaque message.
- Il joint ensuite cette liste au message sous la forme d'un Routing Slip et lance le processus en acheminant le message vers la première étape de traitement.
- Une fois le traitement réussi, chaque étape de traitement consulte le Routing Slip et transmet le message à l'étape de traitement suivante spécifiée dans la table d'acheminement.

# TALEND EXEMPLE 4 :: ROUTING SLIP

183

- Étape 1:
  - Créer une nouvelle "Route" appelée "RoutingSlip".
  - Ajouter et configurer le composant "cFile" appelé "Original Message".



Designer | Code | Dependencies

Job(Route RoutingSlip 0.1) | Contexts(RoutingSlip) | Composant x | Exécuter (Job RoutingSlip)

**<b><i>Original Message</i></b> (cFile\_1)**

**Paramètres simples**

Paramètres avancés

View

Documentation

Chemin d'accès : `context.dir_path + "RoutingSlip/in"`

Paramètres

☒ noop

☐ flatten

☒ autoCreate

Taille de la mémoire tampon (ko)

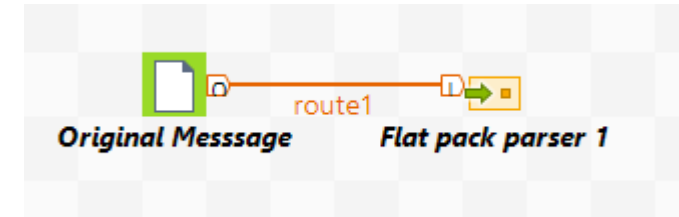
Encodage

fileName

# TALEND EXEMPLE 4 :: ROUTING SLIP


184

- Étape 2:
  - Configurer le composant "cMessagingEndpoint" appelé "Flat pack parser 1".



Designer | Code | Dependencies

Job(Route RoutingSlip 0.1) | Contexts(RoutingSlip) | Composant x | Exécuter (Job RoutingSlip)

 **<b><i>Flat pack parser 1</i></b> (cMessagingEndpoint\_1)**

<b>Paramètres simples</b>	URI	"flatpack:foo?delimiter=;&ignoreFirstRecord=true&splitRows=true"
Paramètres avancés		
View		
Documentation		

# TALEND EXEMPLE 4 :: ROUTING SLIP

185

- Étape 3:
  - Configurer le composant composant "cMessagingEndpoint" appelé "Flat pack parser 2".



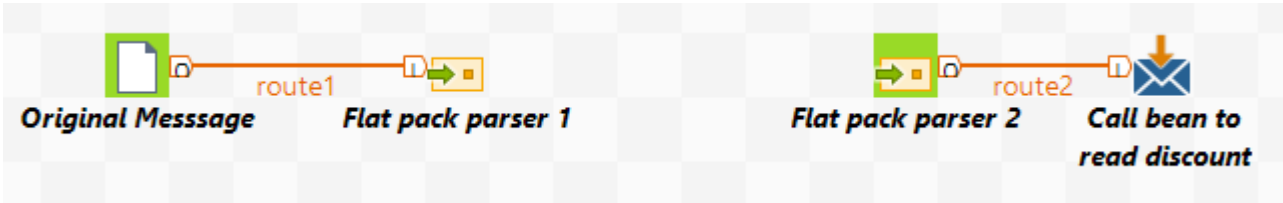
Designer | Code | Dependencies

Job(Route RoutingSlip 0.1) | Contexts(RoutingSlip) | Composant | Exécuter (Job RoutingSlip)

**<b><i>Flat pack parser 2</i></b> (cMessagingEndpoint\_2)**

<b>Paramètres simples</b>	URI	"flatpack:foo?delimiter=;&ignoreFirstRecord=true&splitRows=true"
Paramètres avancés		
View		
Documentation		

- Étape 4:
  - Configurer le composant cSetHeader.



Designer Code Dependencies

Job(Route RoutingSlip 0.1) Contexts(RoutingSlip) Composant Exécuter (Job RoutingSlip) spring

**<b><i> Call bean to<br> read discount</i></b>(cSetHeader\_1)**

Paramètres simples

Paramètres avancés

View

Documentation

En-têtes

Nom	Langue	Valeur
"mySlip"	Bean	beans.ReadDiscount.class

+ × ↑ ↓ 📄 📁

# TALEND EXEMPLE 4 :: ROUTING SLIP

187

- Étape 5:

- Configurer le composant cRoutingSlip.



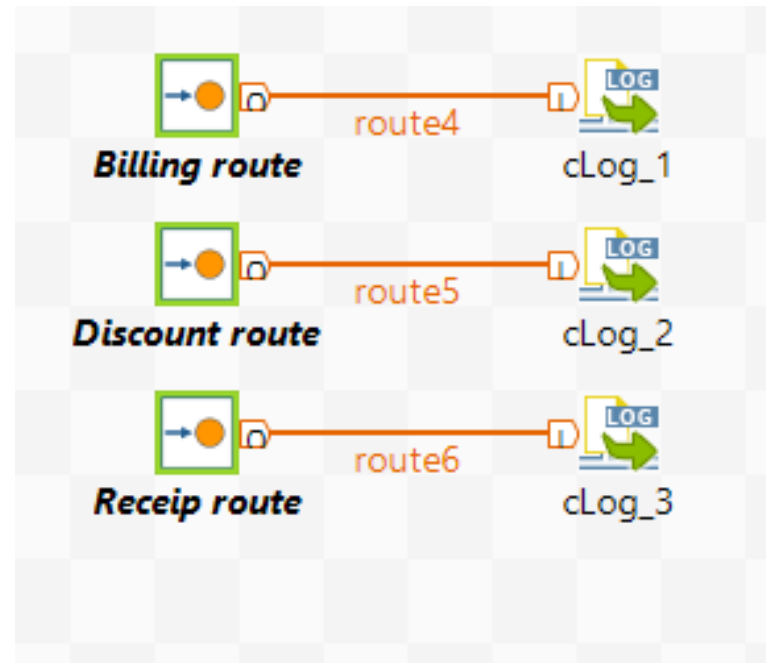
Designer | Code | Dependencies

Job(Route RoutingSlip 0.1) | Contexts(RoutingSlip) | Composant | Exécuter (Job RoutingSlip) | spring

### ◆◆◆ cRoutingSlip\_1

Paramètres simples	Nom de l'en-tête	"mySlip"
Paramètres avancés	Séparateur d'URIs	","
View		
Documentation		

- Étape 6:
  - Configurer les composants cDirect et cLog.





# TALEND EXEMPLE 4 :: ROUTING SLIP

189

- Configurer les 3 composants cDirect

The screenshot displays the Talend Designer interface with three cDirect components configured for a Routing Slip process. Each component has a 'Paramètres simples' (Simple Parameters) tab selected, showing the 'Nom' (Name) parameter.

**Billing route (cDirect\_1)**

Paramètres simples	
Nom	"billing"
Paramètres avancés	
View	
Documentation	

**Discount route (cDirect\_2)**

Paramètres simples	
Nom	"discount"
Paramètres avancés	
View	
Documentation	

**Receipt route (cDirect\_3)**


Paramètres simples	
Nom	"receipt"
Paramètres avancés	
View	
Documentation	

The interface also shows a breadcrumb trail: Job(Route RoutingSlip 0.1) > Contexts(RoutingSlip) > Composant >. The 'Designer' tab is active, and the 'Dependencies' tab is visible in the background.


# TALEND EXEMPLE 4 :: ROUTING SLIP

190


- Configurer les 3 composants cLog

 **cLog\_1**

<b>Paramètres simples</b>	Logging Category: "RoutingSlip.cLog_1"
Paramètres avancés	Niveau: WARN
View	Options
Documentation	<input type="radio"/> Aucune
	<input type="radio"/> Specify a group size for throughput logging
	<input type="radio"/> Group message stats by time interval (in millis)
	<input type="radio"/> Formater le log de sortie
	<input checked="" type="radio"/> Specify output log message
	Message: "direct:billing"

 **cLog\_2**

<b>Paramètres simples</b>	Logging Category: "RoutingSlip.cLog_2"
Paramètres avancés	Niveau: WARN
View	Options
Documentation	<input type="radio"/> Aucune
	<input type="radio"/> Specify a group size for throughput logging
	<input type="radio"/> Group message stats by time interval (in millis)
	<input type="radio"/> Formater le log de sortie
	<input checked="" type="radio"/> Specify output log message
	Message: "Discount Applied, sales manager approval needed!"

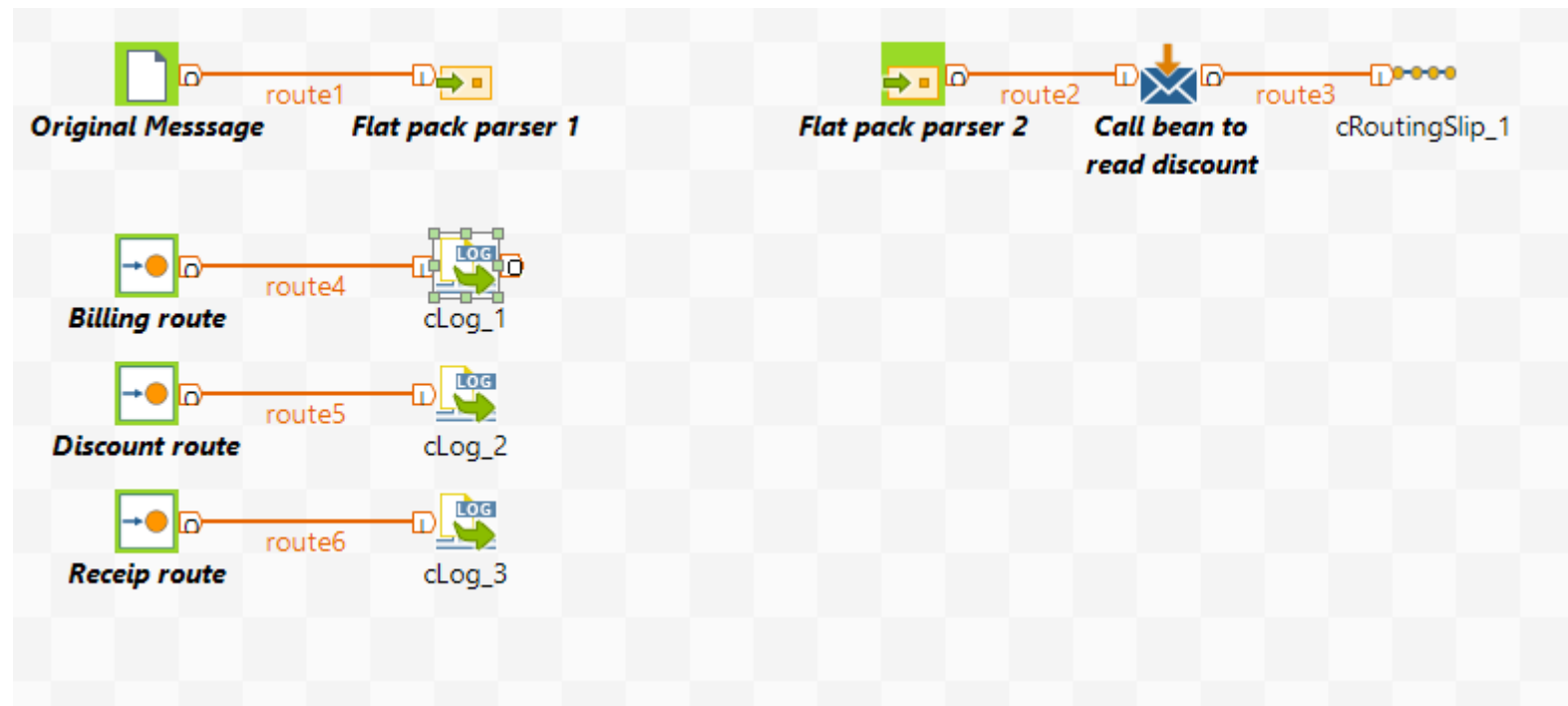
 **cLog\_3**

<b>Paramètres simples</b>	Logging Category: "RoutingSlip.cLog_3"
Paramètres avancés	Niveau: WARN
View	Options
Documentation	<input type="radio"/> Aucune
	<input type="radio"/> Specify a group size for throughput logging
	<input type="radio"/> Group message stats by time interval (in millis)
	<input type="radio"/> Formater le log de sortie
	<input checked="" type="radio"/> Specify output log message
	Message: "direct:receipt"

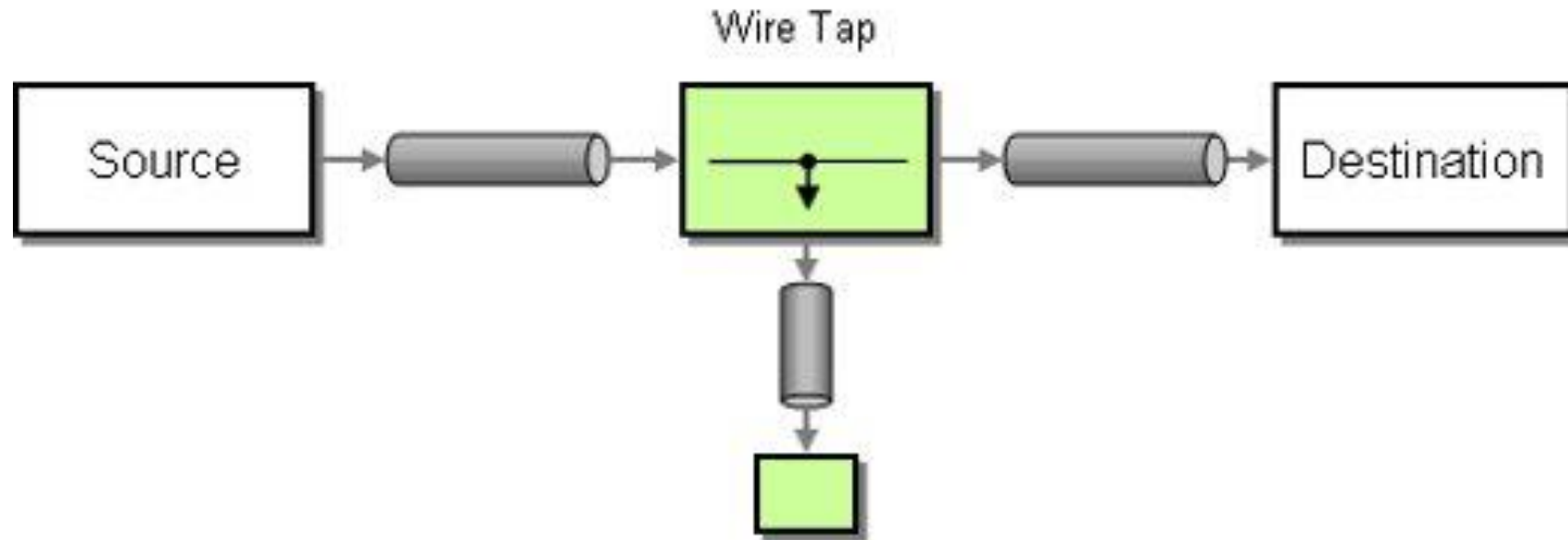
# TALEND EXEMPLE 4 :: ROUTING SLIP

191

- Étape 6:
  - Exécuter la Route

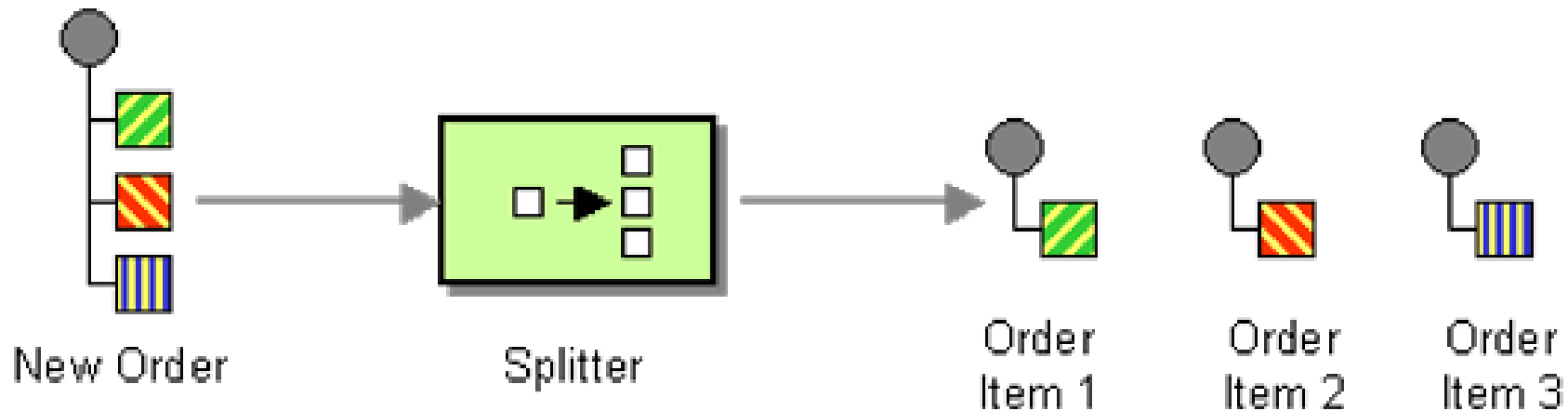


Comment inspecter les messages qui circulent sur un canal point à point ?



- Dans la plupart des cas d'intégration, il est nécessaire de surveiller les messages qui circulent dans le système.
- On y parvient généralement en interceptant le message et en le redirigeant vers un autre endroit, comme la console, le système de fichiers ou la base de données.
- Il est important que cette fonctionnalité ne modifie pas le message original et n'influence pas le chemin de traitement.

Comment pouvons-nous traiter un message s'il contient plusieurs éléments, chacun d'entre eux devant être traité d'une manière différente



- Décomposer le message en une série de messages individuels, chacun contenant des données relatives à un élément.
- Le "Splitter" publie un message pour chaque élément (ou sous-ensemble d'éléments) du message original.

- Étape 1:

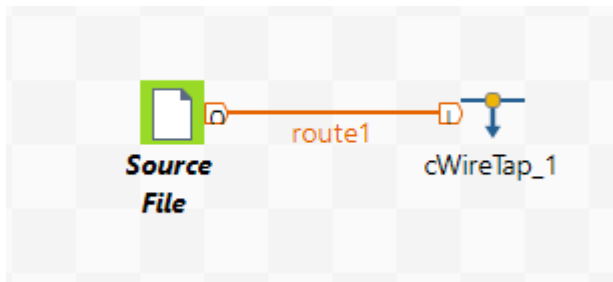
- Créer une nouvelle "Route" appelée "WireTap".
- Ajouter et configurer le composant "cFile" appelé "Source File".



<b>&lt;b&gt;&lt;i&gt;Source &lt;br&gt; File&lt;/i&gt;&lt;/b&gt; (cFile_1)</b>	
<b>Paramètres simples</b>	Chemin d'accès : <input in"="" type="text" value="context.dir_path + " wiretap=""/>
Paramètres avancés	Paramètres
View	<input checked="" type="checkbox"/> noop
Documentation	<input type="checkbox"/> flatten
	<input checked="" type="checkbox"/> autoCreate
	Taille de la mémoire tampon (ko) <input type="text" value="128"/>
	Encodage <input type="text" value="CUSTOM"/> <input type="text" value=""/>
	fileName <input type="text" value=""/>



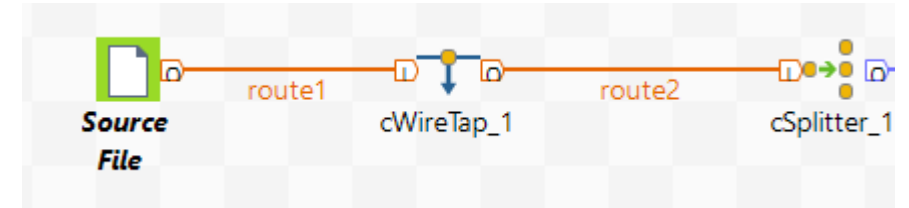
- Étape 2:
  - Configurer le composant "cWireTap".



**cWireTap\_1**

<b>Paramètres simples</b>	URI	"direct:Log"
Paramètres avancés	<input type="checkbox"/> Peupler le nouvel échange	
View	<input checked="" type="checkbox"/> Copier le message original	
Documentation		

- Étape 3:
  - Configurer le composant composant "cSplitter".



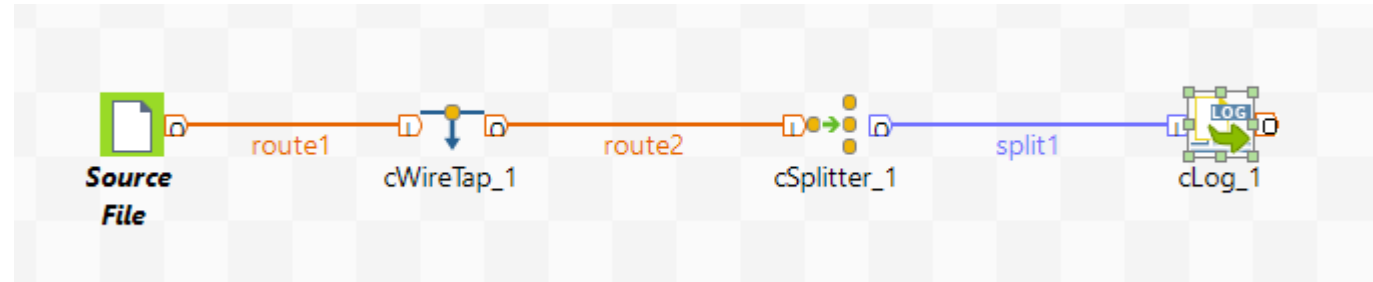
**cSplitter\_1**

<b>Paramètres simples</b>	Langue <span>Xpath</span>
Paramètres avancés	Expression de corrélation
View	Expression <span>"/people/person"</span>
Documentation	<input type="checkbox"/> Use Result Class Type
	<input type="checkbox"/> Add Namespaces
	<input type="checkbox"/> Use Strategy
	Paramètres
	<input type="checkbox"/> Traitement parallèle
	<input type="checkbox"/> Arrêt sur exception
	<input type="checkbox"/> Streaming
	<input type="checkbox"/> Share Unit of Work
	Délai avant suspension <input type="text"/>

# TALEND EXEMPLE 5 :: WIRETAP


199

- Étape 4:
  - Configurer le composant cLog.

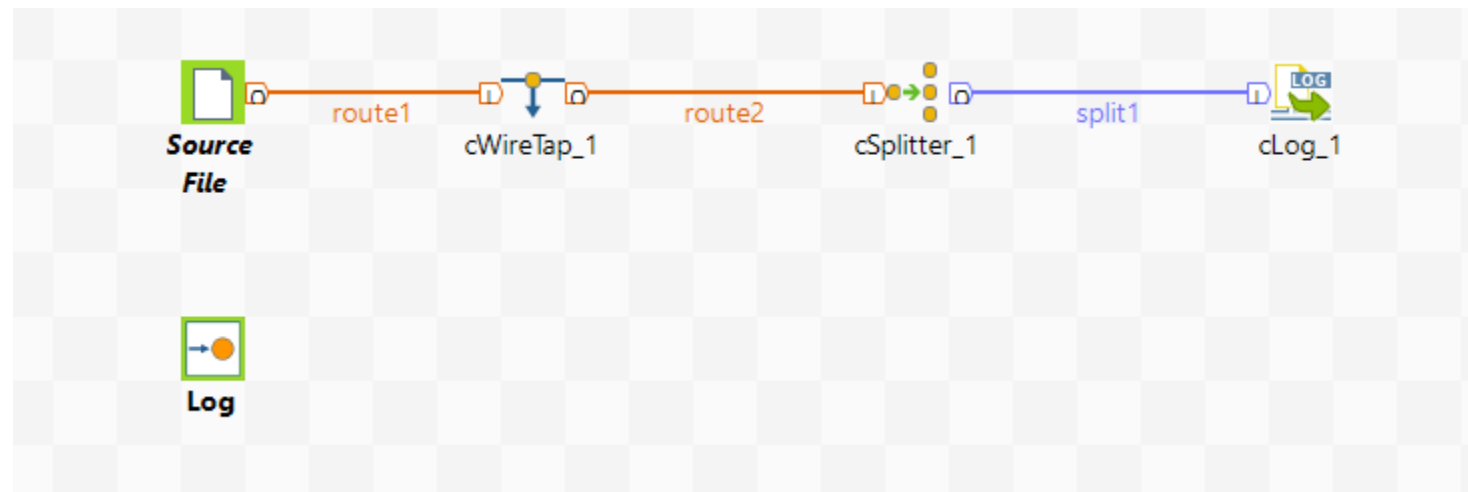


cLog_1	
<b>Paramètres simples</b>	Logging Category: "MyLogger - ouput person\n"
Paramètres avancés	Niveau: WARN
View	Options
Documentation	<input type="radio"/> Aucune
	<input type="radio"/> Specify a group size for throughput logging
	<input type="radio"/> Group message stats by time interval (in millis)
	<input type="radio"/> Formater le log de sortie
	<input checked="" type="radio"/> Specify output log message
	Message: "\${body}"

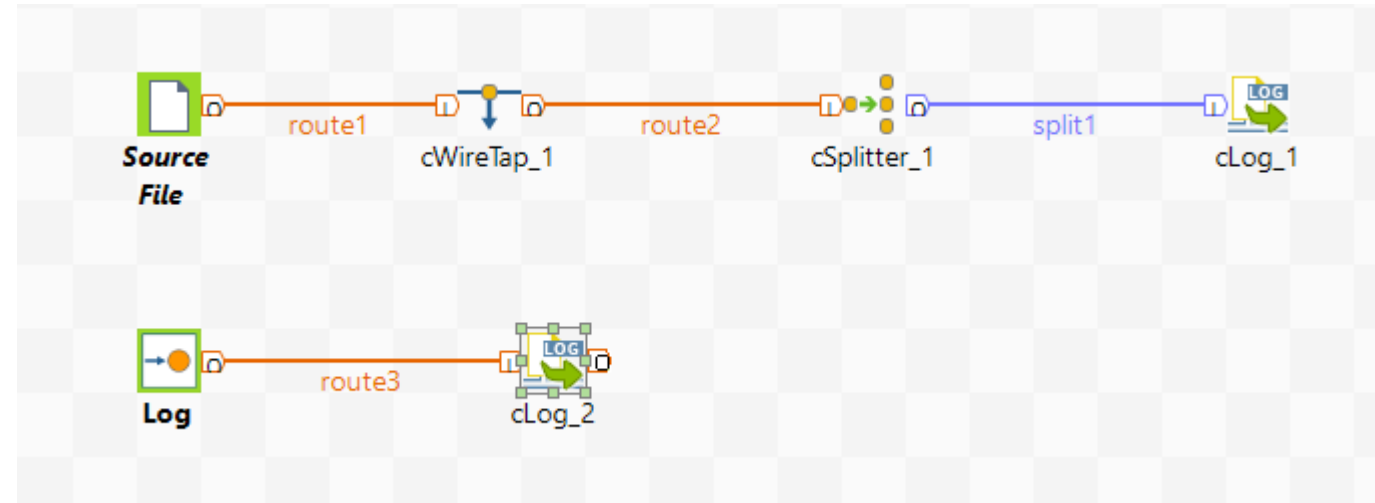
- Étape 5:
  - Configurer le composant cDirect appelé "Log".

 **<b>Log</b> (cDirect\_1)**

<b>Paramètres simples</b>	Nom	"Log"
Paramètres avancés		
View		
Documentation		



- Étape 6:
  - Configurer le composant cLog.



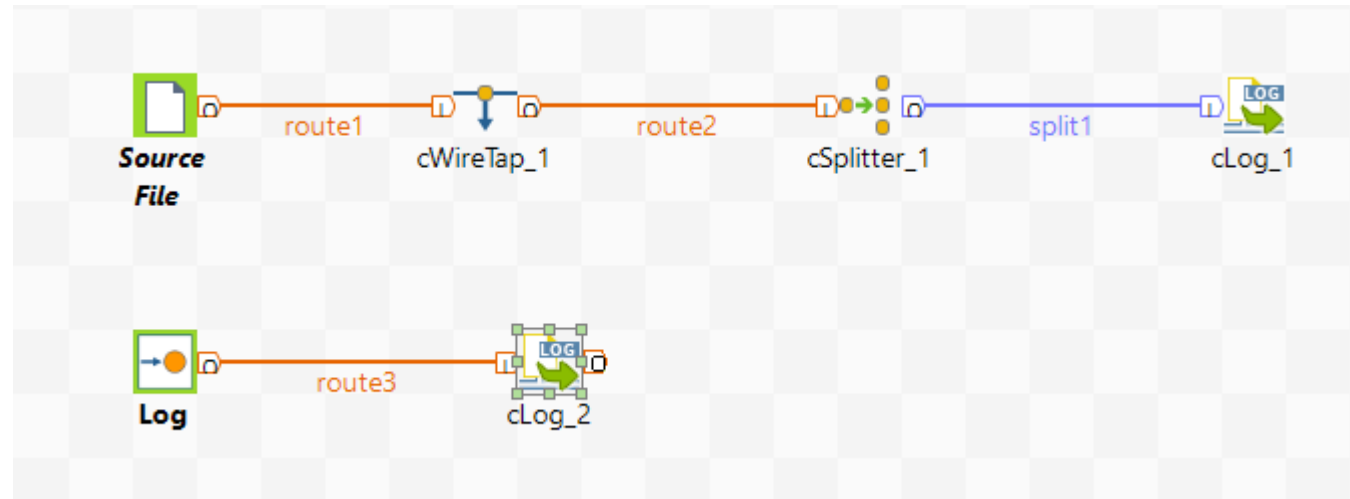
**cLog\_2**

<b>Paramètres simples</b>	Logging Category: "WireTap.cLog_2"
Paramètres avancés	Niveau: WARN ▼
View	Options
Documentation	<input type="radio"/> Aucune
	<input type="radio"/> Specify a group size for throughput logging
	<input type="radio"/> Group message stats by time interval (in millis)
	<input type="radio"/> Formater le log de sortie
	<input checked="" type="radio"/> Specify output log message
	Message: "My DEBUG message"

# TALEND EXEMPLE 5 :: WIRETAP

202

- Étape 6:
  - Exécuter la Route





tDie lance une erreur et met fin au job



tWarn lance un avertissement sans tuer le job



tLogCatcher récupère un ensemble de champs et de messages de Java Exception, tDie et/ou tWarn et les transmet au composant suivant.