Public Support - Support Request #11236

Best practice to implement a generic filter which monitors specific streaming values

2020-05-13 11:00 - hidden

Status:	Closed		
Priority:	Normal		
Category:			
Customer:	AUDI	**Product Issue Numbers:**	
Department:	AST	**Affected Products:**	ADTF 3.7.0
Requester's Priority:	Normal	**Platform:**	Windows 10 64bit
Support Level:	3rd Level	**Topic:**	ADTF::FilterSDK
Resolution:	No Customer Feedback	**FAQ Links:**	

Description

Supportanfrage

aktuell implementiere ich einen Filter, der unten aufgeführten Funktionalitäten besitzt und möchte Sie über die Unterstützung bei der Umsetzungs-Vorgehensweise bitten.

Unten habe ich drei Methoden aufgelistet, die ich aktuell als geeignet für die Filterumsetzung finde. Welche von dieser Methoden würde aus Ihrer Sicht am besten passen?
Oder gibt es vielleicht eine bessere Lösung für dieses Problem?

Filterfunktionalitäten:

1. Bestimmte Ausgangssignale von festgelegten Filtern in dem Filtegraph beobachten (standardmäßig alle Signale von allen Filtern in dem Filtergraph)

2. Bei der Änderung der festgelegten Ausgangssignalen die Werte von diesen Signalen in einen File speichern.

3. Keine Verdrahtung zwischen diesem Filter und allen anderen Filtern in dem Filtergraph

4. Die Signale müssen zeitlich korrekt gespeichert werden, d.h. wenn eine Signaländerung bei allen Filtern in einem folgenden Filtergraph stattfindet: Filter1->Filter2->Filter3, müssen die Signale auch in einer gleiche Reihenfolge geschrieben werden Filter1(Signal1), Filter2(Signal2), Filter3(Signal3).

Methoden, die ich bisher als passend identifiziert habe:

1. Eine "virtuelle" SampleStream nutzen (die nicht im Filtergraph auftaucht) -> anlegen einer SampleStream im Filter/ServiceCode und Registrierung bei den ADTF-Diensten, um die Daten abgreifen zu können.

2. Substream-Konzept nutzen, in dem man die Ausgänge von den anderen Filtern als Substreams definiert und in den zu entwickelnden Filter einspeist.

3. Signal-Listener-Service (über Signal Registry) verwenden, wie es in dem Beispiel Demo_Signal_Listener gezeigt ist. -> Diese Lösung funktioniert wahrscheinlich am einfachsten, ich bin mir da aber nicht sicher wegen der Warnhinweis in der Dokumentation: "Warning: Please be aware that you must not use the Signal Registry to transmit data. You cannot rely on an accurately timed notification about incoming new values."

Vielen Dank im Voraus!

Lösung

1. Bestimmte Ausgangssignale von festgelegten Filtern in dem Filtegraph beobachten (standardmäßig alle Signale von allen Filtern in dem Filtergraph)
2. Bei der Änderung der festgelegten Ausgangssignalen die Werte von diesen Signalen in einen File speichern.

Wenn ich dich richtig verstehe, möchtest du eine Art Watchdog / Monitoring Filter schreiben richtig ?

3. Keine Verdrahtung zwischen diesem Filter und allen anderen Filtern in dem Filtergraph

Geht es dir dabei rein um den Aufwand Vebrindungen ziehen zu müssen ?
Wenn ja, dann haben wir hierzu bereits Möglichkeiten zur Automatisierung und künftig ggf. noch mehr Richtung Autoconnect.

4. Die Signale müssen zeitlich korrekt gespeichert werden, d.h. wenn eine Signaländerung bei allen Filtern in einem folgenden Filtergraph stattfindet: Filter1->Filter2->Filter3, müssen die Signale auch in einer gleiche Reihenfolge geschrieben werden Filter1(Signal1), Filter2(Signal2), Filter3(Signal3).

Das garantiert dir ADTF bei sequentieller Abarbeitung und Zeitstempelung.

1. Eine "virtuelle" SampleStream nutzen (die nicht im Filtergraph auftaucht) -> anlegen einer SampleStream im Filter/ServiceCode und Registrierung bei den ADTF-Diensten, um die Daten abgreifen zu können.

2. Substream-Konzept nutzen, in dem man die Ausgänge von den anderen Filtern als Substreams definiert und in den zu entwickelnden Filter einspeist.

3. Signal-Listener-Service (über Signal Registry) verwenden, wie es in dem Beispiel Demo_Signal_Listener gezeigt ist.

Du könntest Daten ohne "Verbinung" bewerten so wie es z.B. der Sample Stream Trace View macht, allerdings möchtest du ja Werte valdieren, deshalb bleibt dir nur die Möglichkeit deinen Filter zu verbinden (Sample) Ebene oder dich an die Signal Registry zu hängen (Signalebene, keine Verbindung).

Ich meine auch, dass du nicht alle Daten abgreifen kannst (dynamische Arrays z.B. gehen nicht in der Signal Registry).

Die Substreams sehe ich hier nicht, das ist nur eine Übertragungsmöglichkeit, ändert aber nichts daran, Verbindungen ziehen zu müssen.
Du kannst auch nicht einfach mehrere (sub)streams an einen Pin hängen, ADTF erlaub immer nur eine eingehende Verbindung, d.h. egal wie und wo du multiplexed, die Eingangsverbindungen müssen existieren.

-> Diese Lösung funktioniert wahrscheinlich am einfachsten, ich bin mir da aber nicht sicher wegen der Warnhinweis in der Dokumentation: "Warning: Please be aware that you must not use the Signal Registry to transmit data. You cannot rely on an accurately timed notification about incoming new values."

Damit ist gemeint, dass zur Übertragung von Nutzdaten ausschließlich das Data Streaming verwendet werden soll.
Die Signal Registry soll nicht dafür missbraucht werden, Daten auszutauschen.
Sie kann aber zusätzlich für Event Checks und zur Visualsierung hergenommen werden, aber nur für Daten, die ohnehin im Streaming übertragen werden.

History

#1 - 2020-05-14 09:24 - hidden

- Project changed from Public Support to 11

- Topic set to ADTF::FilterSDK

- Customer set to AUDI

- Department set to AST

- Platform Windows 10 64bit added

Hallo Victor,

bitte gib zur Vollständigkeit auch noch die verwendete ADTF Version an.
Habt Ihr euch schon ADTF 3.7 gezogen?

#2 - 2020-05-14 09:28 - hidden

Matthias Fick-Gredler wrote:

> Hallo Victor,
>
> bitte gib zur Vollständigkeit auch noch die verwendete ADTF Version an.
> Habt Ihr euch schon ADTF 3.7 gezogen?

Hallo Matthias,
ja, wir nutzen aktuell die Version 3.7.0.

#3 - 2020-05-15 14:37 - hidden

- Status changed from New to Customer Feedback Required

- Affected Products ADTF 3.7.0 added

Hallo Victor,

> 1. Bestimmte Ausgangssignale von festgelegten Filtern in dem Filtegraph beobachten (standardmäßig alle Signale von allen Filtern in dem Filtergraph)
> 2. Bei der Änderung der festgelegten Ausgangssignalen die Werte von diesen Signalen in einen File speichern.

Wenn ich dich richtig verstehe, möchtest du eine Art Watchdog / Monitoring Filter schreiben richtig ?

> 3. Keine Verdrahtung zwischen diesem Filter und allen anderen Filtern in dem Filtergraph

Geht es dir dabei rein um den Aufwand Vebrindungen ziehen zu müssen ?
Wenn ja, dann haben wir hierzu bereits Möglichkeiten zur Automatisierung und künftig ggf. noch mehr Richtung Autoconnect.

> 4. Die Signale müssen zeitlich korrekt gespeichert werden, d.h. wenn eine Signaländerung bei allen Filtern in einem folgenden Filtergraph stattfindet: Filter1->Filter2->Filter3, müssen die Signale auch in einer gleiche Reihenfolge geschrieben werden Filter1(Signal1), Filter2(Signal2), Filter3(Signal3).

Das garantiert dir ADTF bei sequentieller Abarbeitung und Zeitstempelung.

> 1. Eine "virtuelle" SampleStream nutzen (die nicht im Filtergraph auftaucht) -> anlegen einer SampleStream im Filter/ServiceCode und Registrierung bei den ADTF-Diensten, um die Daten abgreifen zu können.
>
> 2. Substream-Konzept nutzen, in dem man die Ausgänge von den anderen Filtern als Substreams definiert und in den zu entwickelnden Filter einspeist.
>
> 3. Signal-Listener-Service (über Signal Registry) verwenden, wie es in dem Beispiel Demo_Signal_Listener gezeigt ist.

Du könntest Daten ohne "Verbinung" bewerten so wie es z.B. der Sample Stream Trace View macht, allerdings möchtest du ja Werte valdieren, deshalb bleibt dir nur die Möglichkeit deinen Filter zu verbinden (Sample) Ebene oder dich an die Signal Registry zu hängen (Signalebene, keine Verbindung).
@Martin: Ist das perfomant genug ?

Ich meine auch, dass du nicht alle Daten abgreifen kannst (dynamische Arrays z.B. gehen nicht in der Signal Registry).

Die Substreams sehe ich hier nicht, das ist nur eine Übertragungsmöglichkeit, ändert aber nichts daran, Verbindungen ziehen zu müssen.
Du kannst auch nicht einfach mehrere (sub)streams an einen Pin hängen, ADTF erlaub immer nur eine eingehende Verbindung, d.h. egal wie und wo du multiplexed, die Eingangsverbindungen müssen existieren.

> -> Diese Lösung funktioniert wahrscheinlich am einfachsten, ich bin mir da aber nicht sicher wegen der Warnhinweis in der Dokumentation:
> "Warning: Please be aware that you must not use the Signal Registry to transmit data. You cannot rely on an accurately timed notification about incoming new values."

Damit ist gemeint, dass zur Übertragung von Nutzdaten ausschließlich das Data Streaming verwendet werden soll.
Die Signal Registry soll nicht dafür missbraucht werden, Daten auszutauschen.
Sie kann aber zusätzlich für Event Checks und zur Visualsierung hergenommen werden, aber nur für Daten, die ohnehin im Streaming übertragen werden.

#4 - 2020-05-18 09:30 - hidden

Hallo Florian,

vielen Dank für deine ausführlichen und hilfreichen Antworten.
Auf Basis von diesen werde ich dann in die Richtung von Variante 1 gehen und den "virtuellen"/direkt verbundenen SampleStream nutzen.

Unten habe ich die Antworten auf deine Fragen geschrieben.

> Wenn ich dich richtig verstehe, möchtest du eine Art Watchdog / Monitoring Filter schreiben richtig ?

Genau, dass muss ein Monitoring Filter sein, der bei den Änderung von bestimmten Signalen aus den festgelegten Pins diese Signale in den File

schreibt.

> Geht es dir dabei rein um den Aufwand Verbindungen ziehen zu müssen ?

Nicht ganz. Es geht mir mehr um die Erstellung eines generischen Filters, der im Idealfall einfach zu einem Filtergraph hinzugefügt werden kann (der ziemlich kompliziert werden kann und mehreren geschachtelten Subgraphs enthalten) ohne zusätzliche Outports/ Verbindungen zu erstellen. Es ist dann auch die Frage von Übersichtlichkeit des Filtergraphs, wenn da viele Verbindungen existieren.

Viele Grüße
Victor

#5 - 2020-05-19 08:32 - hidden

- Project changed from 11 to Public Support

- Subject changed from Unterstützung bei der Filtererstellung to Best practice to implement a generic filter which monitors specific streaming values

- Description updated

- Status changed from Customer Feedback Required to To Be Closed

- Private changed from Yes to No

- Resolution set to Solved Issue

#6 - 2020-05-19 08:41 - hidden

Hi Viktor,

> Genau, dass muss ein Monitoring Filter sein, der bei den Änderung von bestimmten Signalen aus den festgelegten Pins diese Signale in den File schreibt.

Wie legst Du denn die Signale und Pins fest? Dafür würde sich doch die "normale" Verbindung im CE zu Konfiguration eigenen.

Wenn Du wirklich auf alle Outputs gehen willst, muss das auf jeden Fall ein Service werden, so wie von Flo mit dem Sample Stream Trace Service angeführt. Dann musst Du aber selber eine Konfigurationsmöglichkeit schaffen.

Ganz wichtig: dabei muss man höllisch aufpassen, dass dieser Service nicht zum Synchronisationspunkt für alle Trigger/Daten im Filtergraphen wird. Wenn es wirklich nur ums Monitoring geht ist dafür die Signal Registry das Mittel der Wahl. Die puffert die Daten und verarbeitet sie asynchron.

Bitte führe nochmal genauer aus, welche Inputs wie verarbeitet (interpretiert) werden sollen und was dann geschrieben werden soll?

Danke und beste Grüße,

Martin

#7 - 2020-05-19 08:51 - hidden

- Status changed from To Be Closed to Customer Feedback Required

#9 - 2020-05-20 08:00 - hidden

Hi Martin,

> Wie legst Du denn die Signale und Pins fest? Dafür würde sich doch die "normale" Verbindung im CE zu Konfiguration eigenen.

Idee ist so, dass man die Information über den vorhandenen Strukturen/Signale aus der .description-Datei lädt und in ein Qt-GUI-Fenster (ähnlich wie in dem Signal Tree View) auflistet. Dann kann man aus diesem GUI-Fenster die Signale in ein anderes GUI-Fenster rüber ziehen und so die Trigger-Signale definieren. In diesem GUI liegt man auch die Trigger Bedingungen fest, z.B. die Daten auf Ausgangspin schreiben sobald der Wert vom Triggersignal bestimmte Schwelle überschreitet.
Dann erstellt man code-basiert die Eingangspins für den TriggerFilter (über die IPin- und ISamplStream-Schnittstellen), die man mit allen vorhandenen Sample-Streams verbindet (dabei ist uns noch unklar, ob das geht und ob dann alle Verbindungen doch noch in der Konfigurationseditor gezeichnet werden oder nicht).

Kurze Frage zum Signal Registry:

Die puffert die Daten und verarbeitet sie asynchron.
Wenn die Daten asynchron verarbeite werden, werden wir immer noch feststellen können, in welcher Reihenfolge die Filtern die Daten geschickt haben?

Viele Grüße
Victor

#10 - 2020-05-20 11:33 - hidden

- Status changed from Customer Feedback Required to In Progress

#12 - 2020-06-08 15:37 - hidden

Hallo Victor,

> Idee ist so, dass man die Information über den vorhandenen Strukturen/Signale aus der .description-Datei lädt und in ein Qt-GUI-Fenster (ähnlich wie in dem Signal Tree View) auflistet. Dann kann man aus diesem GUI-Fenster die Signale in ein anderes GUI-Fenster rüber ziehen und so die Trigger-Signale definieren.

Woher kommt denn die .description Datei? In ADTF3 liefern die Filter die Media Description eigentlich fast ausschließlich über die Stream Types an den Output Pins.

> Dann erstellt man code-basiert die Eingangspins für den TriggerFilter (über die IPin- und ISamplStream-Schnittstellen), die man mit allen vorhandenen Sample-Streams verbindet (dabei ist uns noch unklar, ob das geht und ob dann alle Verbindungen doch noch in der Konfigurationseditor gezeichnet werden oder nicht).

Nichts anderes machen die Signal Provider im Zusammenspiel mit der Signal Registry. Die Verarbeitung muss ja auch asynchron erfolgen, auch das macht wie besprochen die Signal Registry.

> Kurze Frage zum Signal Registry:
> Die puffert die Daten und verarbeitet sie asynchron.
> Wenn die Daten asynchron verarbeite werden, werden wir immer noch feststellen können, in welcher Reihenfolge die Filtern die Daten geschickt haben?

Die Signal Registry verarbeitet die Daten in der gleichen Reihenfolge wie sie ankommen, also ja, die happend-before Beziehungen bleiben intakt.

Grüße,

Martin

#13 - 2020-06-08 16:03 - hidden

- Status changed from In Progress to Customer Feedback Required

#14 - 2020-06-22 09:10 - hidden

- Status changed from Customer Feedback Required to To Be Closed

- Resolution changed from Solved Issue to No Customer Feedback

#16 - 2020-07-07 11:18 - hidden

- Support Level changed from 2nd Level to 3rd Level

#18 - 2020-07-07 12:50 - hidden

- Status changed from To Be Closed to Closed