Public Support - Support Request #19690

Creation/Use of System Services

2023-03-23 08:40 - hidden

Status:	Closed		
Priority:	Normal		
Category:			
Customer:	VW	**Product Issue Numbers:**	
Department:	K-GERD	**Affected Products:**	ADTF 3.14.3
Requester's Priority:	Normal	**Platform:**	Windows 10 64bit
Support Level:	2nd Level	**Topic:**	ADTF::SDK
Resolution:	Solved Issue	**FAQ Links:**	

Description

Support Anfrage:

Ich versuche gegenwärtig mich in ADTF3 einzuarbeiten. Die Erstellung und Verwendung von Filtern ist klar. Was mir aus Dokumentation und Beispielen nicht ganz verständlich wird ist wie ich einen System Service erstelle und verwende.

Meine Intention ist es in erster Näherung etwas wie einen "Hello-World" Service zu schreiben, der (noch) keine Nennenswerte Funktionalität hat. Hierfür habe ich einen TestService geschrieben, der von cADTFService erbt und die Funktionen ServiceInit und ServiceShutdown überschreibt. Beide Funktionen sollen jeweils nur eine entsprechende Log-Nachricht ausgeben. Der Konstruktor tut dasselbe. Dieses Minimalbeispiel kompiliert erfolgreich und es wird eine Plugindescription erstellt. Später möchte ich diesem TestService noch eine Property hinzufügen.

Nun meine Fragen:

- Ist diese Implementierung ausreichend? Bzw. muss ich noch weitere Funktionen implementieren, um einen funktionsfähigen Service zu bekommen?
- Wie füge ich diesen Service einer Konfiguration hinzu?
- Als ich den HelloWorld Filter (das Beispiel aus den Guides hier) ausprobierte viel mir auf, dass das Hinzufügen des Timer Runners auch zwei Services hinzufügt. Ist das der intendierte Weg Services zu einer Konfiguratiopn hinzuzufügen? Sprich, dass es Abhängigkeiten zu einem Service geben **muss**?

Ich füge dem Ticket noch meinen Minimal Service hinzu, um o.g. Beschreibung zu vervollständigen.

Lösung:

> Ist diese Implementierung ausreichend? Bzw. muss ich noch weitere Funktionen implementieren, um einen funktionsfähigen Service zu bekommen?

Das ist ausreichend. Du solltest allerdings beim PROVIDE_INTERFACE nicht "lügen" und Interfaces angeben die du gar nicht implementierst, und auch ausschließlich UCOM-Interfaces (ADTF_IID Macro) verwenden.

> Wie füge ich diesen Service einer Konfiguration hinzu?

Über den Configuration Editor per Rechtsklick an der Stelle wo die Services auch aufgelistet sind.

> Als ich den HelloWorld Filter (das Beispiel aus den Guides hier) ausprobierte viel mir auf, dass das Hinzufügen des Timer Runners auch zwei Services hinzufügt. Ist das der intendierte Weg Services zu einer Konfiguratiopn hinzuzufügen? Sprich, dass es Abhängigkeiten zu einem Service geben muss?

Services verhalten sich in den meisten Belangen wie ein "Singleton". Du solltest sie nur dann verwenden, wenn du unbedingt sicher stellen musst dass es nur **eine** Instanz davon in der kompletten Anwendung geben **darf**.
Es gibt im Normalfall nur 3 Anwendungsfälle wo das zutrifft:

- Der Service kapselt eine andere API die auf grundlegender Ebene nicht Thread-safe ist oder Multiplexing auf Anwenderseite

fordert.
- Der Service macht absolut grundlegendes Schedulings (das trifft nur auf IClock zu - nichts anderes sollte Scheduling versuchen!)
- Der Service wird benötigt um teure Ressourcen nur 1x zu halten.

In allen diesen Fällen zeichnet den Service aus, dass er ein Interface nach Außen hat, das von vielen Konsumenten genutzt wird. Wenn du kein Interface **anbietest**, hast du im Normalfall auch keinen Bedarf dafür das als Service zu tun.
Wenn es mehrere Instanzen geben **darf**, dann bist du mit einem Filter (auch wenn der keine Input/Output-Pins hat) und Interface-Bindings deutlich besser beraten.

Ausgerechnet die Signal Registry ist hier leider ein Beispiel für etwas das eigentlich niemals ein Service hätte sein sollen - seit der Einführung der Substreams ist es deutlich leichter geworden Signale **gruppiert** zu Displays zu leiten, während die Signal Registry zu einem riesigen Flaschenhals geworden ist die unter der Aufgabe Millionen von Signal-Updates pro Sekunde durch ein Nadelöhr zu pressen zusammen bricht. (Funktioniert in isolierten Minimal-Aufbauten - aber versagt spätestens nach der Integration oder unter Echtzeitanforderungen.)

History

#1 - 2023-03-23 12:37 - hidden

> Ist diese Implementierung ausreichend? Bzw. muss ich noch weitere Funktionen implementieren, um einen funktionsfähigen Service zu bekommen?

Das ist ausreichend. Du solltest allerdings beim PROVIDE_INTERFACE nicht "lügen" und Interfaces angeben die du gar nicht implementierst, und auch ausschließlich UCOM-Interfaces (ADTF_IID Macro) verwenden.

> Wie füge ich diesen Service einer Konfiguration hinzu?

Über den Configuration Editor per Rechtsklick an der Stelle wo die Services auch aufgelistet sind.

> Als ich den HelloWorld Filter (das Beispiel aus den Guides hier) ausprobierte viel mir auf, dass das Hinzufügen des Timer Runners auch zwei Services hinzufügt. Ist das der intendierte Weg Services zu einer Konfiguratiopn hinzuzufügen? Sprich, dass es Abhängigkeiten zu einem Service geben muss?

Services verhalten sich in den meisten Belangen wie ein "Singleton". Du solltest sie nur dann verwenden, wenn du unbedingt sicher stellen musst dass es nur **eine** Instanz davon in der kompletten Anwendung geben **darf**.
Es gibt im Normalfall nur 3 Anwendungsfälle wo das zutrifft:

- Der Service kapselt eine andere API die auf grundlegender Ebene nicht Thread-safe ist oder Multiplexing auf Anwenderseite fordert.
- Der Service macht absolut grundlegendes Schedulings (das trifft nur auf IClock zu - nichts anderes sollte Scheduling versuchen!)
- Der Service wird benötigt um teure Ressourcen nur 1x zu halten.

In allen diesen Fällen zeichnet den Service aus, dass er ein Interface nach Außen hat, das von vielen Konsumenten genutzt wird. Wenn du kein Interface **anbietest**, hast du im Normalfall auch keinen Bedarf dafür das als Service zu tun.
Wenn es mehrere Instanzen geben **darf**, dann bist du mit einem Filter (auch wenn der keine Input/Output-Pins hat) und Interface-Bindings deutlich besser beraten.

Ausgerechnet die Signal Registry ist hier leider ein Beispiel für etwas das eigentlich niemals ein Service hätte sein sollen - seit der Einführung der Substreams ist es deutlich leichter geworden Signale **gruppiert** zu Displays zu leiten, während die Signal Registry zu einem riesigen Flaschenhals geworden ist die unter der Aufgabe Millionen von Signal-Updates pro Sekunde durch ein Nadelöhr zu pressen zusammen bricht. (Funktioniert in isolierten Minimal-Aufbauten - aber versagt spätestens nach der Integration oder unter Echtzeitanforderungen.)

#2 - 2023-03-23 13:55 - hidden

- Project changed from Public Support to 20

- Status changed from New to Customer Feedback Required

#5 - 2023-03-28 07:14 - hidden

Guten Morgen Bernhard,

konnte dir auf deine Fragen ausreichend Informationen übermittelt werden?
Sind noch Fragen offen?

Bitte gebe uns kurzes Feedback ob weitere Fragen sind oder ob das Ticket geschlossen werden kann.

Mit besten Grüßen

Sascha

#6 - 2023-03-28 08:00 - hidden

Guten Morgen!

Ja, die Antwort war sehr informativ. Den Rechtsklick zum Hinzufügen von Services konnte ich kurz vor der Antwort auch finden. Fragen sind gegenwärtig keine mehr offen und das Ticket kann meines Erachtens geschlossen werden.

Ich habe mich tatsächlich sehr über die ausführliche Antwort gefreut, meinen Dank an die Person, die das Ticket ursprünglich bearbeitet hat.

Grüße
Bernhard

#7 - 2023-03-28 08:31 - hidden

- Subject changed from Erstellung/Verwendung von System Services to Creation/Use of System Services

- Description updated

- Status changed from Customer Feedback Required to To Be Closed

- Resolution set to Solved Issue

- Topic set to ADTF::SDK

#8 - 2023-04-03 07:18 - hidden

- Project changed from 20 to Public Support

- Status changed from To Be Closed to Closed

- Private changed from Yes to No

Files

TestService.cpp	683 Bytes	2023-03-23	hidden
TestService.h	607 Bytes	2023-03-23	hidden