

開発者必読!

“もしも”のときの デバッグ技法

.NETアプリケーション障害解析

株式会社NTTデータ
飯山 教史
IIYAMA, Takashi

第 6 回

.NETアプリケーションの メモリーク解析

Level

1 2 3 4 5

Technology Tools

- Visual Basic
- Visual C#
- Visual C++
- SQL Server
- Oracle
- Access
- ASP.NET
- Other:

WinDbg (Debugging Tools
for Windows)
SOS.DLL

Samples



はじめに

前回の記事では、.NETアプリケーションの稼働中に、クラス内フィールドの値やマネージヒープの状態を、WinDbgとSOS.DLLを利用して調べる方法を紹介した。稼働中のアプリケーションが利用しているクラスの状態を把握できるようになれば、アプリケーションのデバッグに有益な情報が手に入り、保守作業の効率化につながることは間違いない。

今回は調査対象を「メモリーク」に限定し、先月紹介した手法を取り入れながら解析を進めることにする。具体的には、Windowsが持つ機能を利用して、

- ・メモリークの発生を確認する方法
- ・メモリーク発生時に.NETアプリケーションにWinDbgをアタッチし、SOS.DLLのコマンドでマネージヒープの内容を確認する方法

を紹介する。



解析手順

説明を始める前に解析手順を紹介しておく。一般的にメモリークの調査は、

- ①メモリークの検出
- ②.NETアプリケーションでの発生を確認
- ③リークしているオブジェクトを確認
- ④リークを発生させているクラスの確認

の手順で進められる。この解析で利用するツールは表1のとおりである。以降で、この手順に従ってメモリークの解析を行なう。



メモリークの検出

.NETアプリケーションのメモリークを検出する一番簡単な方法は「タスクマネージャ」を利用することである。

タスクマネージャを起動して「パフォ

パフォーマンス」タブの「メモリ使用量」および「メモリ使用量の履歴」を調べると、アプリケーションが利用しているメモリ利用量がわかる。アプリケーション起動後から徐々に利用量が増えている場合は、メモリリークが発生している可能性がある。

このとき、タスクマネージャの「パフォーマンス」タブで表示される「コミットチャージ」の「合計」の値に注目する(図1)。この値は、プログラムとオペレーティングシステムに割り当てられたメモリの合計であり、アプリケーションが現在利用しているメモリ量(仮想メモリ含む)になる^[注1]。アプリケーションの実行中に「合計」の値が増加している場合には、メモリリークが発生している可能性が高い。

メモリリークは、VS.NETなどの開発ツールによる開発時には考慮されない傾向がある。そのため、メモリリークの発生は単体試験終了後の結合試験に入って初めて気がつくことが多い。しかし、発見するのが前工程であるほど修正コストが低くなるのはメモリリークも同様である。

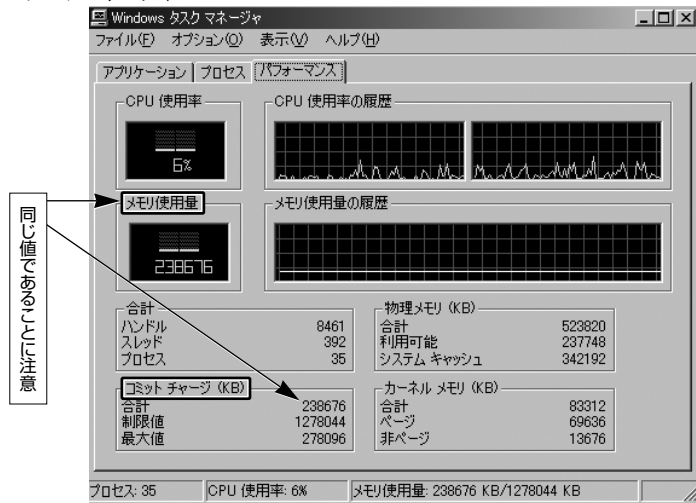
これは私個人の提案であるが、開発ツールで開発している段階からタスクマネージャは立ち上げておいたほうがよい。そして目視でもかまわないので、メソッド実行前後のメモリ量を把握し、リークが発生している場合はなるべく早い段階でデバッグを開始するようにすれば、後工程にメモリリークが持ち込まれることは少なくなる。

上記の作業で、.NETアプリケーションのメモリリークを検出できる。メモリリークが発生していることがわかっ

表1: 調査で利用するツール

使う順番	ツール	手順
1	タスクマネージャ	メモリリークの検出
2	システムモニタ	.NETアプリケーションでの発生を確認
3	WinDbg+SOS.DLL	リークしているオブジェクトを確認
4	WinDbg+SOS.DLL	リークを発生させているクラスの確認

図1: タスクマネージャ



たら、続いてシステムモニタを利用して、どのアプリケーションでリークしているかの確認に入る。

.NETアプリの状態を確認するシステムモニタ

どのアプリケーションでメモリリークが発生しているかを調べるには、システムモニタのカウンタログを利用する(図2)。システムモニタでは、システムの状態を確認するためにさまざまなカウンタを提供しているが、.NETアプリケーションのメモリリークを調査する場合は表2のカウンタを取得すればよい。

まずはじめに「Private Bytes」と「# Bytes in all Heaps」の値を確認する。「Private Bytes」はプロセス別に利用しているメモリ量が記録される。したが

って、メモリリークが発生しているとき、この値を見ればどのプロセスで発生しているかがすぐにわかる。

メモリリークを起こしているプロセスが特定できたら、そのプロセスの「# Bytes in all Heaps」の値を調べる。この値が高い場合、.NETアプリケーションのコードでメモリリークが発生していることを表わす^[注2]。

ここまでの作業で、メモリリークが発生した原因が.NETアプリケーションのコードにあることを確認できる。

注1) ディスク上に存在する仮想メモリpagefile.sysに書き込む可能性に備えて、仮想メモリマネージャが領域を確保するとそのサイズがコミットメモリとして計上される。このコミットメモリの特徴から、プロセスの実際のメモリ利用量を知りたいときには、コミットメモリの利用量を調べることになる。
注2) 「# Bytes in all Heaps」の値が低ければ、非.NETアプリケーションでメモリリークが発生していることを表わす。