|  |
| --- |
|  |
| Windows Server Advanced Power Management |
| Benutzerhandbuch |
|  |
|  |
| Version 1.6.0 |

|  |
| --- |
| [decatec.de](https://decatec.de/) |

Inhalt

[Einführung 2](#_Toc494449846)

[Voraussetzungen 3](#_Toc494449847)

[Open Source Lizenz 4](#_Toc494449848)

[Einschränkungen 5](#_Toc494449849)

[Installation 6](#_Toc494449850)

[Deinstallation 9](#_Toc494449851)

[Update 10](#_Toc494449852)

[Bedienung 11](#_Toc494449853)

[Hauptfenster 12](#_Toc494449854)

[Einstellungen 16](#_Toc494449855)

[Allgemein 16](#_Toc494449856)

[Überwachung (System) 21](#_Toc494449857)

[Überwachung (Erweitert) 25](#_Toc494449858)

[Nach Überprüfung 29](#_Toc494449859)

[Aufwachen 31](#_Toc494449860)

[Uptime 35](#_Toc494449861)

[Plugins 38](#_Toc494449862)

[Remote-Shutdown 40](#_Toc494449863)

[Beispielszenarien 42](#_Toc494449864)

[Fehlerbehebung 44](#_Toc494449865)

[FAQ 47](#_Toc494449866)

[Windows Server Advanced Power Management für Entwickler 48](#_Toc494449867)

[Plugin-Entwicklung 48](#_Toc494449868)

[Normale und erweiterte Plugins 48](#_Toc494449869)

[Schnelleinstieg 48](#_Toc494449870)

[Entwicklung einfacher Plugins 49](#_Toc494449871)

[Entwicklung erweiterter Plugins 51](#_Toc494449872)

[Debuggen von Plugins 54](#_Toc494449873)

[Bereitstellen von Plugins für andere Benutzer 55](#_Toc494449874)

[Remote-Shutdown 56](#_Toc494449875)

# Einführung

Windows Server Advanced Power Management (im Folgenden auch WSAPM genannt) ist ein Programm für Windows Betriebssysteme, mit dem Richtlinien definiert werden können, die verhindern, dass Windows in den Standby-Modus wechselt, wenn der Rechner gerade in Verwendung ist. Dadurch kann Strom gespart werden, weil die Hardware nur dann läuft, wenn sie auch tatsächlich benötigt wird. Dabei ist WSAPM v.a. für den Einsatz auf (Home-)Servern konzipiert, indem es dafür ausgelegt ist, auch ohne Benutzer-Anmeldung zu laufen. WSAPM kann aber auch auf Desktop-PCs eingerichtet werden, um auch hier ein unbeabsichtigtes Wechseln in den Standby-Modus zu verhindern.

In den Windows-Energieoptionen kann definiert werden, nach welcher Zeit der Rechner in den Standby-Modus wechseln soll, wenn dieser nicht in Verwendung ist. Jedoch erkennt Windows prinzipiell nur direkte Nutzereingaben am Rechner als Aktionen an, welche das Wechseln in den Standby-Modus verhindern. Wenn beispielsweise ein Programm längere Zeit unbeaufsichtigt laufen muss, schaltet Windows trotzdem nach dem vorher definierten Zeitplan in den Standby-Modus.

In WSAPM können nun verschiedene Richtlinien festgelegt werden, die definieren, wann der Rechner noch in Verwendung ist. Nach diesen Richtlinien wird dann das Wechseln in den Standby-Modus verhindert.

Dabei unterstützt WSAPM folgende Szenarien, die eine Verwendung des Rechners signalisieren:

* CPU-Auslastung
* Speicher-Auslastung
* Netzwerk-Auslastung (Download, Upload und kombiniert, für eine oder mehrere Netzwerkkarten)
* Zugriff auf Netzwerk-Freigaben (Dateien und/oder Verzeichnisse)
* Festplatten-Auslastung (logische Datenträger)
* Verfügbare netzwerkfähige Geräte (PCs, Smartphones, TV-Geräte, Streaming-Clients, Spielkonsolen, etc.)
* Laufende Programme

Nach der Überprüfung der Richtlinien können benutzerdefinierte Aktionen durchgeführt werden.

Darüber hinaus kann in WSAPM ein Zeitplan festgelegt werden, nach dem der Rechner automatisch aus dem Standby-Modus erwacht.

Ebenso können Uptimes definiert werden. Während dieser Zeitspanne wird verhindert, dass der Computer in den Standy-Modus wechselt (unabhängig von anderen definierten Richtlinien).

Das Programm kann ebenfalls über eine Plugin-Schnittstelle auf einfache Weise erweitert werden. Durch Plugins können dabei eigene Richtlinien definieren werden, nach den der Rechner nicht in den Standby-Modus wechseln darf.

In Verbindung mit entsprechenden Anwendungen oder Apps (z.B. [MagicPacket](https://decatec.de/software/magicpacket/)) kann mittels WSAPM ein Rechner aus der Ferne heruntergefahren, neu gestartet, oder in den Standby- bzw. Ruhezustand versetzt werden.

# Voraussetzungen

Voraussetzung für WSAPM ist prinzipiell nur ein Rechner, welcher den Windows Standby-Modus unterstützt. Dabei werden folgende Windows Versionen unterstützt:

* Windows XP
* Windows Vista
* Windows 7
* Windows 8/Windows 8.1
* Windows 10
* Windows Server 2003 (R2)
* Windows Server 2008 (R2)
* Windows Server 2012 (R2)
* Windows Home Server 2011

Ebenso wird das [.NET Framework 4.0](http://de.wikipedia.org/wiki/.NET) benötigt. Dieses wird automatisch bei der Installation mit installiert, wenn es auf dem System noch nicht vorhanden ist.

# Open Source Lizenz

Die Software Windows Server Advanced Power Management ist unter eine Open Source Lizenz verfügbar. Sämtlicher Sourcecode und weitere Informationen hierzu sind auf [GitHub](https://github.com/DecaTec/Wsapm) zu finden.

Dank der Verfügbarkeit als Open Source Software kann jeder aktiv an diesem Projekt mitwirken. Jegliche Mithilfe (Coding, Übersetzungen, etc.) unterstützen des Projekt uns bringen es weiter voran.

# Einschränkungen

Derzeit gelten folgende Einschränkungen für WSAPM:

* Zum Konfigurieren von WSAPM werden Administrator-Rechte benötigt. Ohne Administrator-Rechte ist WSAPM zwar auch lauffähig, allerdings kann die aktuelle Konfiguration nicht verändert werden.
* Die von WSAPM geschriebenen Log-Einträge werden immer in der (original) Sprache des Betriebssystems erstellt, auch wenn die Sprache des Nutzers nachträglich geändert wurde.

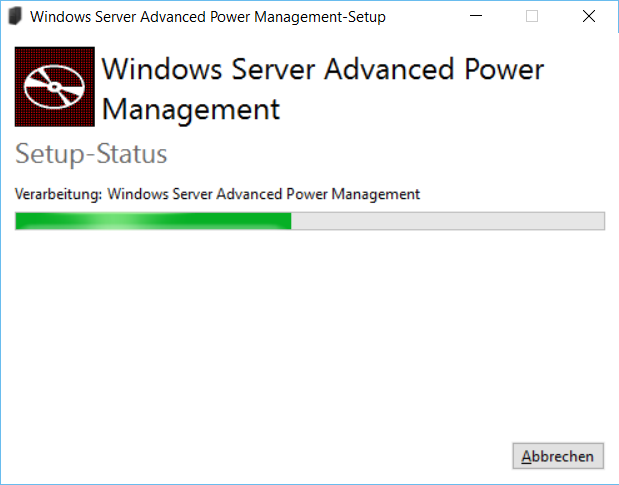
# Installation

Die Installation wird über die Setup-Datei (WsapmSetup.exe) gestartet. Zur Installation werden Administrator-Rechte benötigt. Falls das .NET Framework 4.0 noch nicht auf dem System vorhanden ist, wird dieses automatisch mit installiert. In diesem Fall wird dann eine Internetverbindung benötigt.



Mit einem Klick auf *Optionen* kann der Pfad festgelegt werden, in dem WSAPM installiert wird.  
Standardmäßig wird das Programm im Verzeichnis *C:\Programme (x86)\Windows Server Advanced Power Management* installiert.

Mit einem Klick auf *Installieren* wird WSAPM auf dem System installiert und eingerichtet.



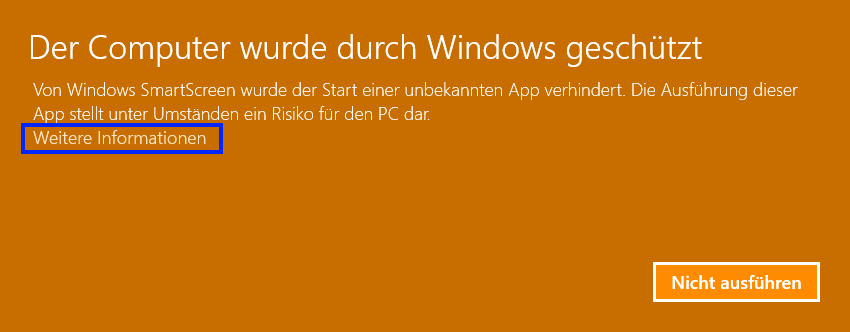
**Firewall-Ausnahmen**

Während der Installation wird eine Ausnahme für die Windows Firewall eingerichtet, so dass WSAPM Remote-Shutdown-Befehle über UDP empfangen kann.

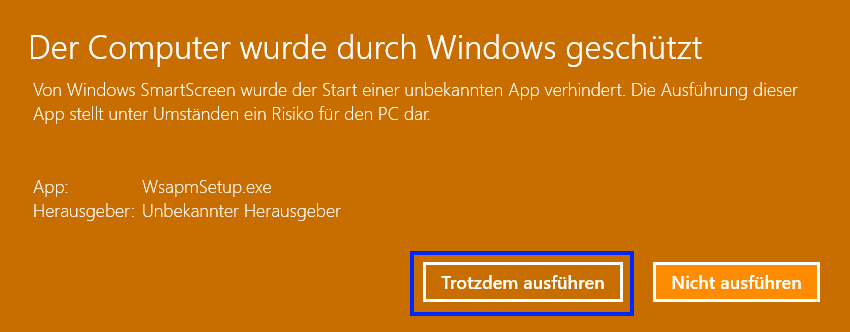
Wenn eine Firewall eines Drittanbieters installiert ist, muss hier manuell eine Ausnahme für die Datei *WsapmService.exe* (zu finden im Installationsverzeichnis) für UDP hinzugefügt werden.

**SmartScreen**

Ab Windows 8 gibt es ein Windows-Feature namens SmartScreen, welches die Ausführung von schädlicher Software verhindern soll. Leider wird SmartScreen auch bei unbekannter Software aktiv und unterbricht in diesem Fall erst einmal die Installation. Fall SmartScreen auf dem Rechner aktiviert wurde kann bei der Installation von WSAPM ein solcher Hinweis erscheinen.



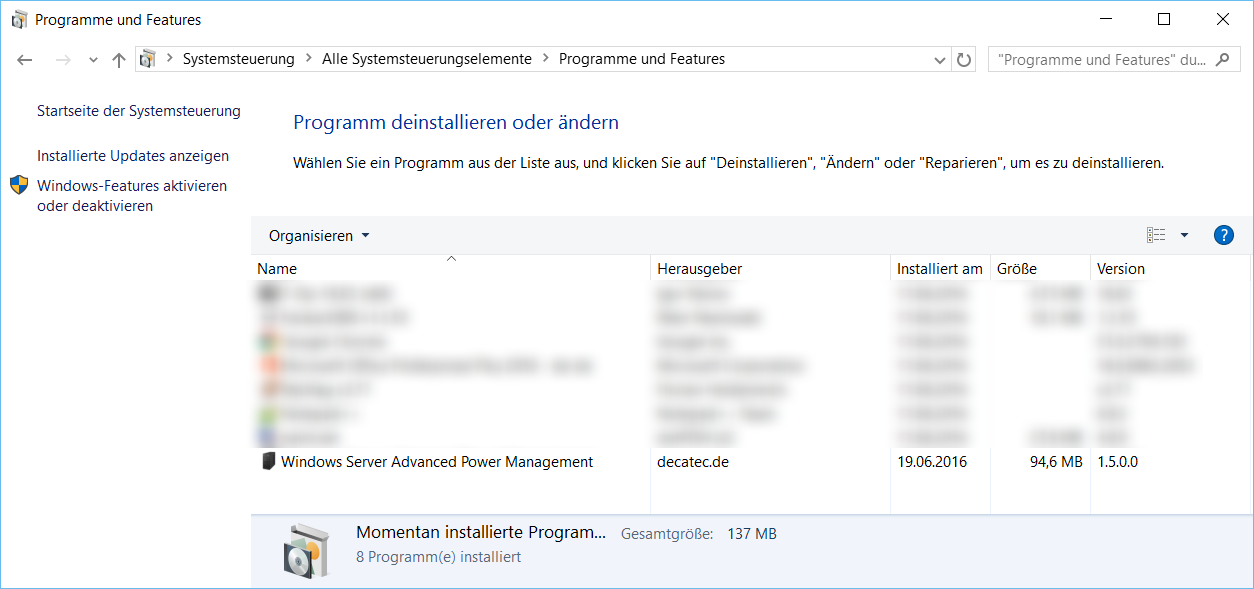
In diesem Fall bedeutet dies lediglich, dass SmartScreen die aktuelle Version von WSAPM noch nicht kennt. Der Installer für WSAPM enthält jedoch keine Schad-Software, Adware, o.ä. Nach einem Klick auf *Weitere Informationen* kann die Installation über die Schaltfläche *Trotzdem ausführen* fortgeführt werden.



# Deinstallation

Bei der Deinstallation wird das Programm restlos vom System entfernt, d.h. es werden auch alle Einstellungen gelöscht und die Ausnahme in der Windows Firewall wieder entfernt.

Die Deinstallation wird über die Windows-Funktion zum Deinstallieren von Software ausgeführt (*Programme und Funktionen* in der Systemsteuerung von Windows). Mit einem Doppelklick auf den Eintrag *Windows Server Advanced Power Management* wird das Programm deinstalliert.



# Update

Wenn eine neuere Version von Windows Server Advanced Power Management erscheint, kann diese einfach über eine bestehende Version installiert werden.

Die vorhandenen Einstellungen werden dabei nicht gelöscht und bleiben erhalten.

# Bedienung

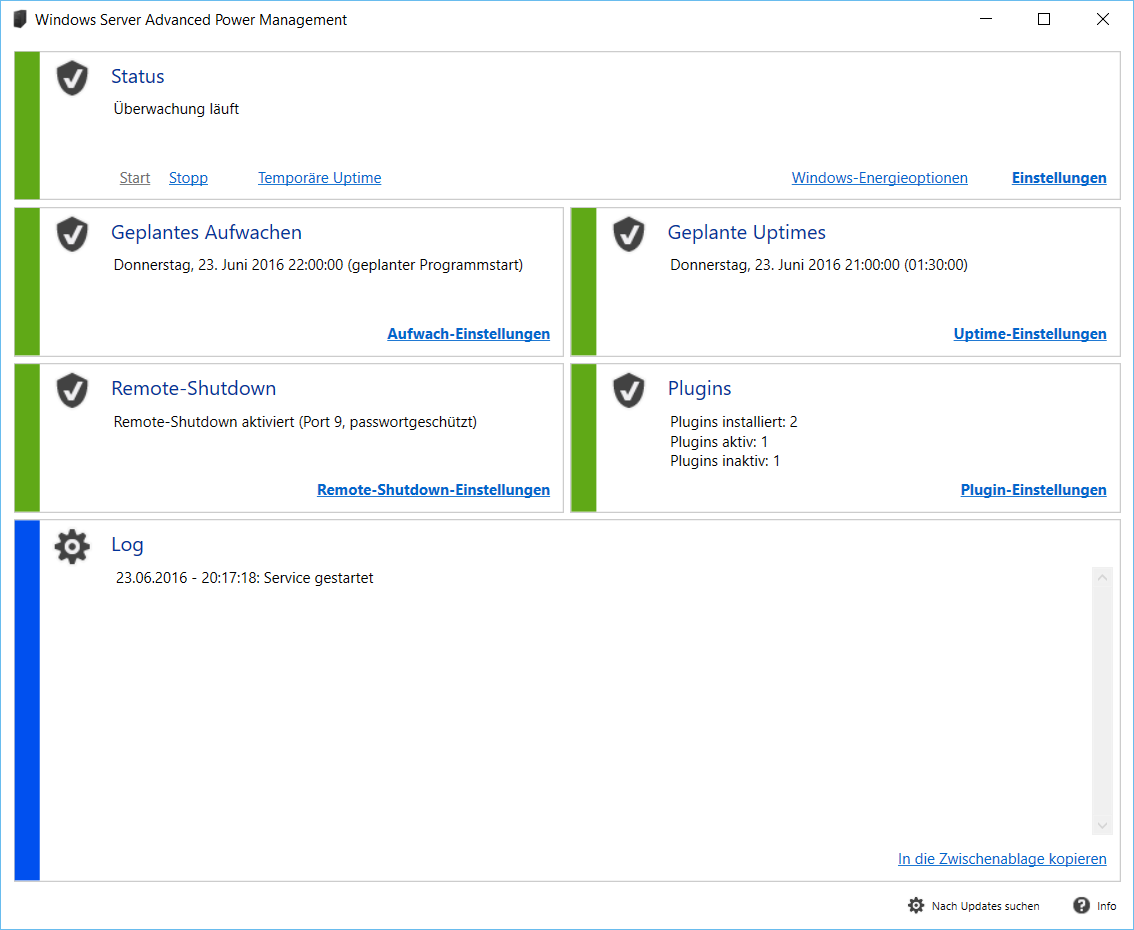
Windows Server Advanced Power Management ist in zwei Komponenten aufgeteilt:

* Das Programm Windows Server Advanced Power Management: Über dieses Programm kann das Verhalten von WSAPM geändert und angepasst werden. Dieses Programm kann über die Verknüpfung auf dem Desktop gestartet werden, welche im Rahmen der Installation angelegt wird. Ebenso ist es möglich, das Programm über das Windows Startmenü aufzurufen (*Alle Programme* 🡪 *Windows Server Advanced Power Management* 🡪 *Windows Server Advanced Power Management*).  
  Wichtig: Dieses Programm muss nicht laufen, damit die angegebenen Richtlinien überprüft und ggf. der Standby-Modus unterdrückt wird. Es dient lediglich dazu, die Einstellungen von WSAPM festzulegen und weitere Informationen (z.B. Log-Einträge) bereit zu stellen.
* Der Windows Server Advanced Power Management Dienst: Dieser Windows-Dienst übernimmt die eigentliche Überwachung und das Ausführen von Aktionen. Dieser wird beim Systemstart automatisch gestartet und bedarf keiner Interaktion mit dem User.

Die folgende Beschreibung bezieht sich auf das ausschließlich Programm Windows Server Advanced Power Management, der Windows-Dienst wird bei der Installation des Programms eingerichtet und sollte im weiteren Verlauf auch nicht verändert werden.

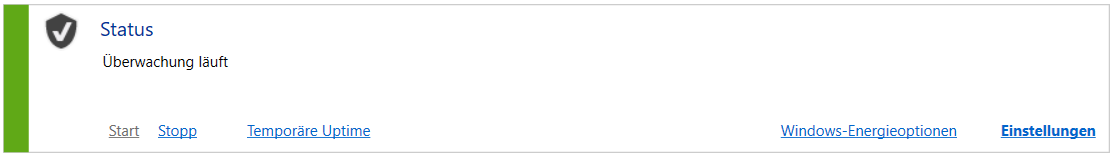
## Hauptfenster

Nach dem Programmstart von Windows Server Advanced Power Management wird das Hauptfenster angezeigt:



**Status-Panel *Status***

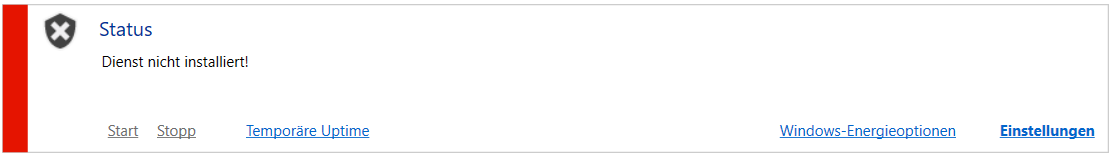
Das Status-Panel *Status* zeigt auf einen Blick den Status der Überwachung an. Dieser Status wird über den zu Grunde liegenden Dienst definiert. Wenn der Dienst läuft und einwandfrei funktioniert, befindet sich WSAPM im Status *Überwachung läuft*:



Wenn der Dienst nicht oder fehlerhaft läuft, befindet sich WSAPM im Status *Überwachung deaktiviert*:

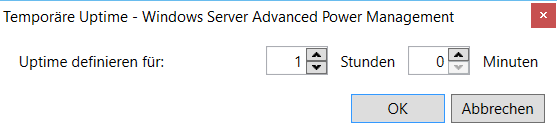


Wenn der Dienst aus irgendeinem Grund nicht korrekt installiert wurde, ist dies ebenfalls über dieses Status-Panel ersichtlich:



Mit den Links *Start* und *Stopp* lässt sich die Überwachung manuell ein- bzw. ausschalten. Dazu wird der im Hintergrund laufende Windows-Dienst gestartet bzw. gestoppt.

Mit dem Link *Temporäre Uptime* kann eine Zeitspanne festgelegt werden, in der der Computer nicht in den Standby wechseln darf (unabhängig von anderen Richtlinien):



Diese temporäre Uptime „überlebt“ dabei auch einen Neustart des Rechners und kann nur beendet werden, indem die Option *Temporäre Uptime beenden* gewählt wird. Der Status einer evtl. definierten Uptime wird ebenso im Hauptfenster angezeigt. Neben einer temporären Uptime kann ebenfalls eine geplante Uptime in den Einstellungen definiert werden (siehe [*Einstellungen - Uptime*](#_Uptime)).

Mit dem Link *Einstellungen* kann der Einstellungs-Dialog zum Konfigurieren von WSAPM geöffnet werden (siehe [*Einstellungen*](#_Einstellungen_1)).

Mit dem Link *Windows-Energieoptionen* können die Energieoptionen von Windows schnell gestartet werden. Dies ist oftmals hilfreich, da WSAPM eng mit den Windows-Energieoptionen zusammenarbeitet.

**Status-Panel *Geplantes Aufwachen***

Das Status-Panel *Geplantes Aufwachen* gibt Auskunft über die nächsten Termine, an den der Rechner von WASPM automatisch aus dem Standby geweckt werden soll. Wenn beim geplanten Aufwachen Programme gestartet werden sollen, ist dies durch den Hinweis *geplanter Programmstart* ersichtlich.

Mit dem Link *Aufwach-Einstellungen* kann der Einstellungs-Dialog zum Festlegen von Zeitplänen zum automatischen Aufwachen aufgerufen werden (siehe [*Einstellungen - Aufwachen*](#_Aufwachen_1)).

**Status-Panel *Remote-Shutdown***

Dieses Status-Panel zeigt an, ob Remote-Shutdown für den Computer aktiviert oder deaktiviert ist.

Der Link *Remote-Shutdown-Einstellungen* öffnet den Einstellungsdialog zum Festlegen der Remote-Shutdown-Optionen (siehe [*Einstellungen - Remote-Shutdown*](#_Remote-Shutdown_1)).

**Status-Panel *Geplante Uptimes***

Dieses Status-Panel zeigt die nächsten geplanten Uptimes und die jeweilige Dauer an.

Der Link *Uptime-Einstellungen* öffnet den Einstellungs-Dialog zum konfigurieren geplanter Uptimes (siehe [*Einstellungen - Uptime*](#_Uptime)).

**Status-Panel *Plugins***

Dieses Status-Panel zeigt den Status der installierten Plugins an.

Mit dem Link *Plugin-Einstellungen* wird der Einstellungs-Dialog geöffnet, so dass Plugins schnell hinzugefügt/entfernt oder aktiviert/deaktiviert werden können (siehe [*Einstellungen - Plugins*](#_Plugins)).

**Status-Panel *Log***

Im Status-Panel *Log* können sämtliche Log-Einträge gefunden werden, sofern das Logging über die Einstellungen aktiviert ist (siehe [*Einstellungen - Allgemein*](#_Allgemein_1)). Die Log-Einträge geben Auskunft darüber, wie WSAPM gerade arbeitet, z.B. wann der Standby-Modus unterdrückt wurde oder wann der Rechner in den Standby-Modus gewechselt ist.

Teile des Logs können einfach durch Auswählen und Kopieren in die Zwischenablage kopiert werden. Der Link *In die Zwischenablage kopieren* überträgt alle Log-Einträge in die Zwischenablage.

**Nach Updates suchen**

Über die Schaltfläche *Nach Updates suchen* kann nach Updates für WSAPM gesucht werden. Fall ein Update verfügbar ist, kann dieses sofort heruntergeladen und installiert werden.

**Info**

Über die Schaltfläche *Info* können erweiterte Informationen zu WSAPM (z.B. Versions-Informationen, Änderungshistorie, etc.) aufgerufen werden.

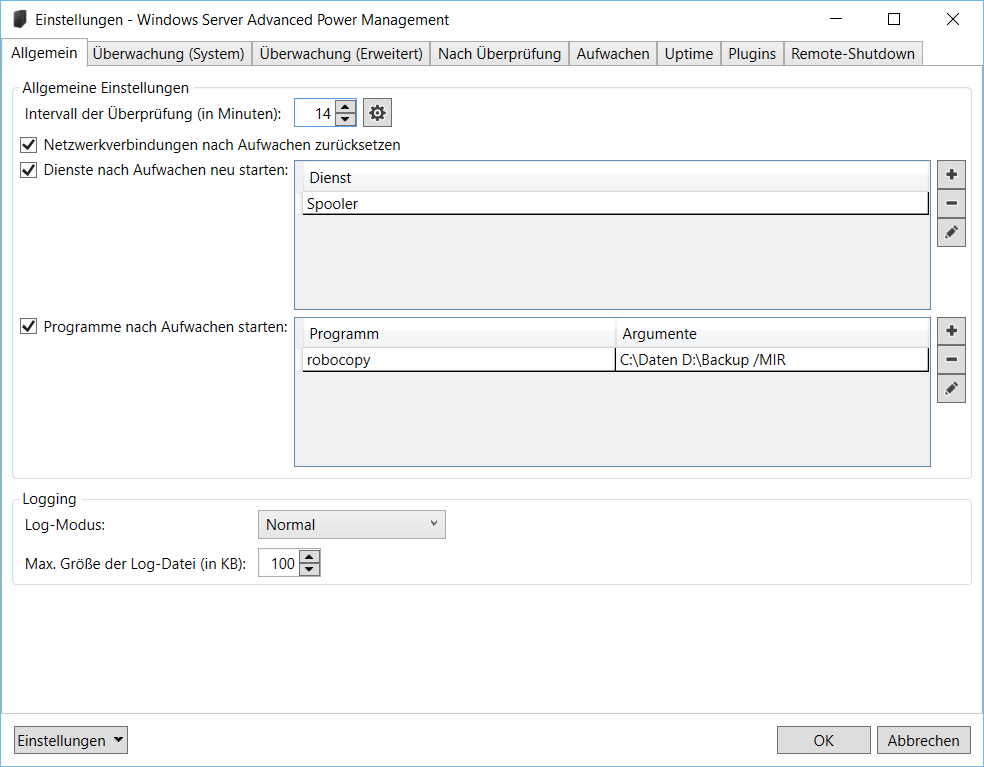
## Einstellungen

Mit Hilfe der Einstellungen kann Windows Server Advanced Power Management für die jeweiligen Anforderungen konfiguriert werden.

Die Einstellungen können über den Link *Einstellungen* im Hauptfenster geöffnet werden.

### Allgemein

Im Reiter *Allgemein* können allgemeine Einstellungen festgelegt werden.



**Intervall der Überprüfung (in Minuten)**

Gibt das Intervall der Überprüfung in Minuten an, d.h. es wird periodisch nach Ablauf der eingestellten Zeit eine Überprüfung sämtlicher definierter Richtlinien ausgeführt. Falls dabei eine oder mehrere Bedingungen zutreffen, die eine Verwendung des Rechners signalisieren, wird der Standby-Modus bis zur nächsten Überprüfung unterdrückt.

Wichtig: Da WSAPM mit den Windows-Energieeinstellungen zusammenarbeitet, muss diese Option immer an die jeweils im System eingestellte Standby-Zeit angepasst werden. Wenn beispielsweise in den Windows-Energieeinstellungen konfiguriert ist, dass der Standby-Modus nach 15 Minuten Inaktivität aktiviert wird, muss WSAPM so eingestellt werden, dass das Intervall der Überwachung kürzer ist als 15 Minuten. Nur dann können die in WSAPM definierten Richtlinien vor dem Wechsel in den Standby-Modus überprüft werden. Wenn nach der Konfiguration von WSAPM die Standby-Zeit in den Windows-Energieeinstellungen nachträglich geändert wird, muss diese Option auf jeden Fall auch in WSAPM angepasst werden, um einen einwandfreien Betrieb zu ermöglichen.

Mit der Schaltfläche neben dem Eingabefeld kann die optimale Zeit für das Überprüfungs-Intervall ermittelt werden. Dieses ist immer der in den Windows-Energieoptionen eingestellt Wert abzüglich einer Minute.

**Netzwerkverbindungen nach Aufwachen zurücksetzen**

Wenn diese Option aktiviert ist, werden die aktiven Netzwerkverbindungen nach dem Aufwachen zurückgesetzt (d.h. neu initialisiert). Diese Option ist dann besonders sinnvoll, wenn es nach dem Aufwachen zu Problemen mit den Netzwerkverbindungen kommt.

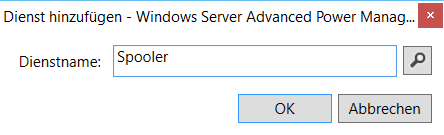
**Dienste nach Aufwachen neu starten**

Die hier aufgeführten Windows-Dienste werden nach dem Aufwachen automatisch neu gestartet. Dies geschieht bei jedem Aufwachen des Rechners, ganz gleich ob es sich dabei um ein in WSAPM geplantes Aufwachen oder um ein Aufwachen durch andere Programme oder Benutzereingaben handelt.   
Diese Funktion ist beispielsweise nützlich bei Diensten, die nach dem Aufwachen aus dem Standby nicht mehr richtig funktionieren.  
Die Dienste werden ca. 30 Sekunden nach dem Aufwachen des Computers neu gestartet.  
Wenn mehrere Dienste angegeben wurden, ist die Reihenfolge jedoch nicht festgelegt, mit der die Dienste neu gestartet werden.

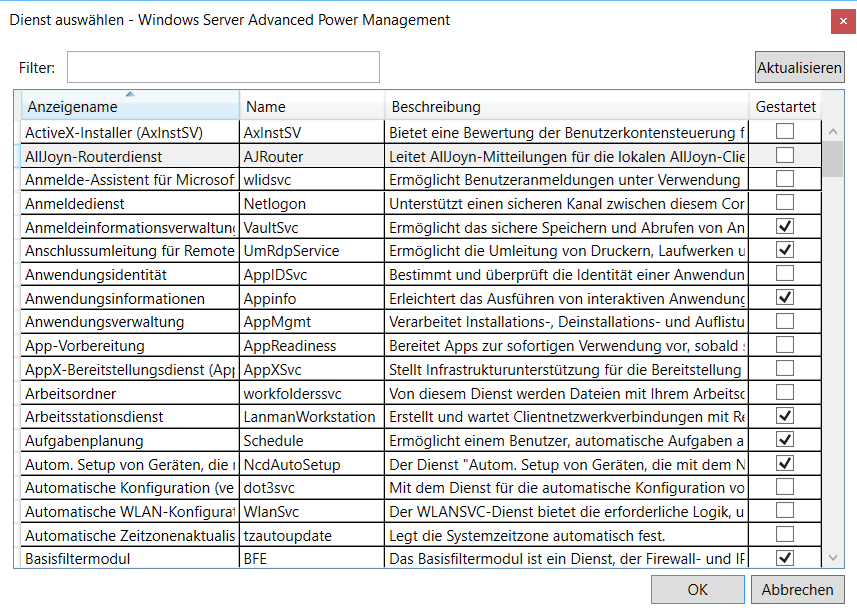
Mit der Plus-Schaltfläche kann ein Dienst hinzugefügt werden.  
Mit der Minus-Schaltfläche kann ein Dienst aus der Liste entfernt werden.  
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann der ausgewählte Dienst bearbeitet werden.

**Eingabemaske *Dienst hinzufügen***

Wenn ein Dienst hinzugefügt oder bearbeitet wird, erscheint eine Eingabemaske, in die der Name des Dienstes eingegeben werden kann. Dabei kann es sich sowohl um den Dienstnamen (z.B. „Spooler“), als auch um den Anzeigenamen (in diesem Beispiel „Druckwarteschlange“) handeln.



Mit der Schaltfläche neben dem Eingabefeld kann ein weiterer Dialog geöffnet werden, mit dem aus der Liste der aktuell installierten Windows-Dienste ein Dienst ausgewählt werden kann.



**Programme nach Aufwachen starten**

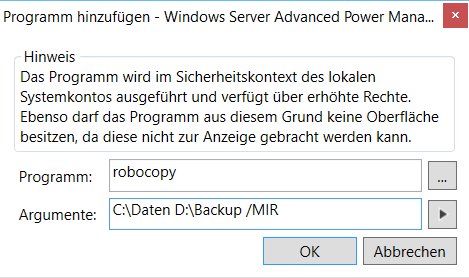
Die hier aufgeführten Programme werden nach dem Aufwachen automatisch mit den angegebenen Argumenten ausgeführt. Dies geschieht bei jedem Aufwachen des Rechners, ganz gleich ob es sich dabei um ein in WSAPM geplantes Aufwachen oder um ein Aufwachen durch andere Programme oder Benutzereingaben handelt.   
Wenn ein Programm nur dann gestartet werden soll, wenn der Computer durch ein von WSAPM geplantes Aufwachen startet, sollte dazu die Option *Programme nach Aufwachen starten* unter *Aufwachen* genutzt werden (siehe [*Einstellungen - Aufwachen*](#_Aufwachen_1)).  
Die Programme werden ca. 30 Sekunden nach dem Aufwachen des Computers gestartet.  
Wenn mehrere Programme angegeben wurden, ist die Reihenfolge jedoch nicht festgelegt, mit der die Programme gestartet werden.

Wichtig: Die hier aufgeführten Programme werden im Sicherheitskontext des lokalen Systemkontos ausgeführt und besitzen daher erhöhte Rechte.  
Ebenso sollten diese keine Benutzeroberfläche anzeigen, da es nicht möglich ist, in diesem Kontext eine Oberfläche zur Anzeige zu bringen. D.h. Programme mit Benutzeroberfläche werden zwar gestartet, werden aber dem aktuell angemeldeten Benutzer nicht angezeigt.

Mit der Plus-Schaltfläche kann ein zu startendes Programm hinzugefügt werden.   
Mit der Minus-Schaltfläche kann ein Programm aus der Liste entfernt werden.   
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann das ausgewählte Programm und die dazugehörigen Argumente bearbeitet werden.

**Eingabemaske Programm hinzufügen**

Wenn ein Programm hinzugefügt oder bearbeitet wird, erscheint eine Eingabemaske, in die der Pfad zum Programm (die Angabe einer Batch-Datei ist ebenfalls möglich) und Argumente für den Programmstart eingegeben werden können. Das Programm kann mit den angegebenen Argumenten durch die Abspielen-Schalfläche testweise gestartet werden.



**Log-Modus**

Hier kann der Log-Modus festgelegt werden. Es stehen folgende Optionen zu Auswahl:

* *Kein Logging*: Es erden keine Log-Einträge geschrieben.
* *Nur Fehler*: Es werden nur Fehler von WSAPM geloggt.
* *Normal*: Normales Logging, alle relevanten Ereignisse werden geloggt.
* *Detailliert*: Detaillierter Log, sämtliche Aktionen von WSAPM werden geloggt. Nur empfehlenswert für Fehlersituationen, in denen WSAPM nicht wie erwartet funktioniert.

Wenn das Logging aktiv ist, wird automatisch eine Log-Datei geschrieben, deren Inhalt auch im Status-Panel *Log* angezeigt wird und Aktionen von WSAPM aufzeigt. Wird die Einstellung deaktiviert (Modus *Kein Logging*), werden keine Log-Einträge mehr geschrieben, eine bestehende Log-Datei wird allerdings auch nicht gelöscht.

**Max. Größe der Log-Datei (in KB)**

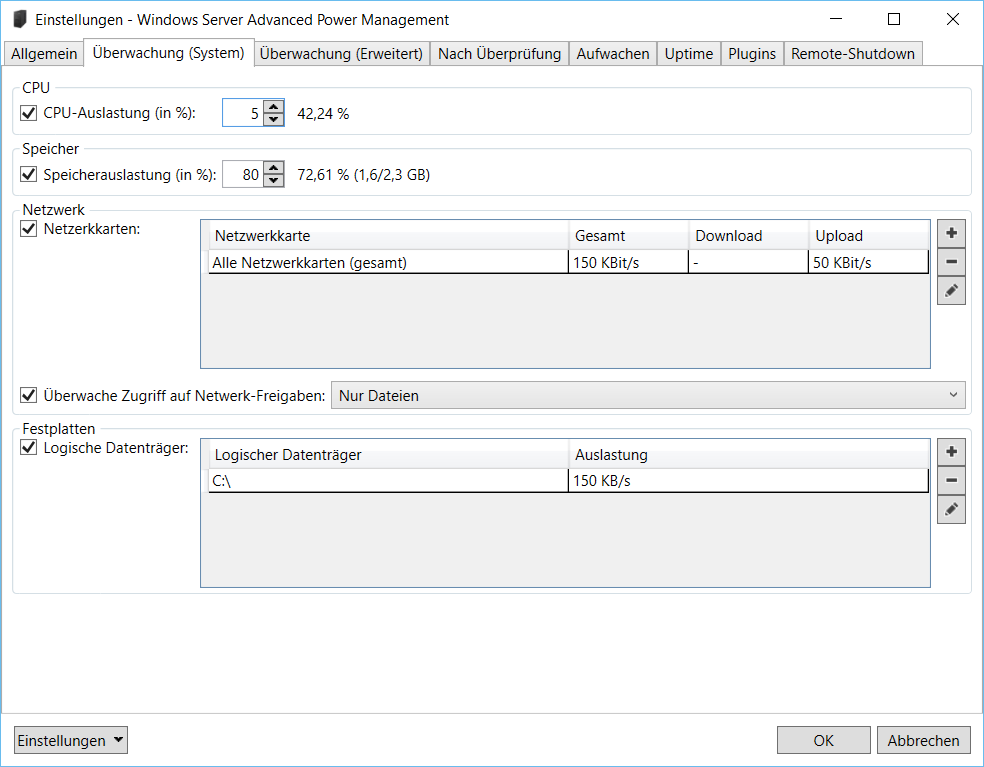
Über diese Option wird die max. Größe der Log-Datei in KB angegeben. Wenn dieser Wert auf 0 gesetzt wird, kann die Log-Datei beliebig groß werden. Wenn ein Wert ungleich 0 angegeben wird, wird die Log-Datei beim Erreichen dieser Größe automatisch gelöscht und dafür eine neue angelegt.

**Einstellungen-Schaltfläche**

Diese Schaltfläche bietet die Optionen, die Einstellungen von WSAPM auf die Standard-Einstellungen zurückzusetzen oder die aktuellen Einstellungen zu exportieren oder zu importieren.

### Überwachung (System)

Im Reiter *Überwachung (System)* können Richtlinien zum Unterdrücken des Standby-Modus festgelegt werden, die bestimmte System-Parameter überwachen. Alle hier festgelegten Überwachungs-Richtlinien werden periodisch nach Ablauf des entsprechenden Intervalls (siehe [*Einstellungen - Allgemein*](#_Allgemein_1)) überprüft. Wenn mindestens eine Überprüfung ergibt, dass der Rechner in Verwendung ist, wird der Standby-Modus bis zu nächsten Überprüfung unterdrückt.



**CPU-Auslastung (in %)**

Dieser Wert gibt an, ab welcher Prozessorauslastung (in Prozent) eine Verwendung des Rechners festgestellt wird.

Neben dem Eingabefeld wird dabei der aktuelle Wert der CPU-Auslastung angezeigt.

**Speicherauslastung (in %)**

Gibt den Prozentsatz der Speicherauslastung an, ab dem der Rechner nicht mehr in den Standby-Modus wechseln darf.

Die aktuelle Auslastung in Prozent und Speicher in Verwendung/Gesamt (in GB) wird ebenso angezeigt.

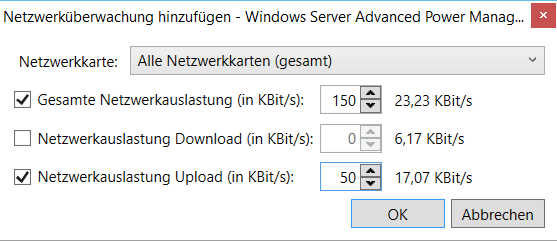
**Netzwerkkarten**

In diesem Bereich kann die Überwachung von Netzwerkkarten aktiviert werden. Die Festlegung der Parameter erfolgt dabei für jede Netzwerkkarte einzeln, oder für alle im System vorhandenen Netzwerkkarten. Dies ist besonders für Systeme nützlich, die über mehrere Netzwerkkarten verfügen.

Durch die Plus-Schaltfläche kann eine Überwachung zur Liste hinzugefügt werden.  
Mit der Minus-Schaltfläche kann ein Eintrag aus der Liste entfernt werden.  
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann die ausgewählte Netzwerk-Überwachung bearbeitet werden.

**Eingabemaske *Netzwerküberwachung hinzufügen***

Beim Hinzufügen einer Überwachung sollte zunächst einmal die Netzwerkkarte gewählt werden. Wird hier *Alle Netzwerkkarten (gesamt)* angegeben, werden die Werte aller einzelnen Netzwerkkarten aufaddiert.



Die folgenden Parameter können hier angegeben werden:

* Auslastung Download (in KBit/s)
* Auslastung Upload (in KBit/s)
* Gesamte Netzwerkauslastung (Upload + Download) (in KBit/s) - wenn beispielsweise ein Download mit 100 KBit/s läuft und gleichzeitig ein Upload mit 50 KBit/s stattfindet, ergibt das eine gesamte Netzwerkauslastung von 150 KBit/s

Wenn der hier angegebene Wert bei der Überprüfung der Richtlinien überschritten wird, wechselt der Computer nicht in den Standby-Modus.

Neben den Eingabefeldern wird der aktuelle Wert für Upload/Download/gesamte Auslastung angezeigt.

**Überwache Zugriff auf Netzwerk-Freigaben**

Wenn diese Option aktiviert ist, wird der Computer nicht in den Standby-Modus versetzt, wenn ein Zugriff auf Netzwerk-Freigaben erfolgt. Dies ist besonders nützlich, wenn nur sporadische Netzwerk-Zugriffe auf einen Server stattfinden.

Dabei können folgende Zugriffsarten eingestellt werden:

* *Nur Dateien*: Es wird nur der Zugriff auf Dateien erfasst, z.B. wenn ein Client ein Word-Dokument auf der Netzwerkfreigabe des Servers geöffnet hat.
* *Nur Verzeichnisse*: Es wird nur der Zugriff auf Verzeichnisse erfasst, z.B. wenn ein Client einen Ordner auf einer Netzfreigabe geöffnet hat.
* *Dateien und Verzeichnisse*: Es werden sowohl Zugriff auf Dateien, als auch auf Verzeichnisse erfasst.

Wichtig: Der Zugriff auf freigegebene Verzeichnisse findet im Netzwerk u.U. sehr häufig und auch im Hintergrund statt. Ebenfalls wird beim Schließen eines solchen Verzeichnisses am Client der Zugriff oftmals nicht sofort wieder aufgehoben. Daher kann eine Überwachung auf freigegebene Verzeichnisse dazu führen, dass der Server nur noch sehr selten in den Standby wechseln kann. Die Überwachung auf freigegebene Verzeichnisse sollte daher mit Vorsicht eingesetzt werden.

**Logische Datenträger**

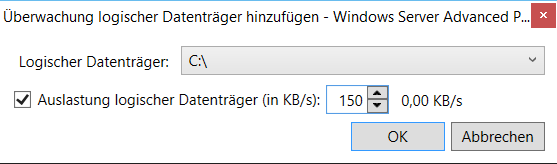
Analog zu den Einstellungen der Netzwerkkarten kann in diesem Bereich die Auslastung einzelner logischer Datenträger überwacht werden. Dabei ist es wieder möglich, die Werte für jeden Datenträger einzeln oder für alle Datenträger zusammen zu definieren.

Durch die Plus-Schaltfläche kann eine Überwachung zur Liste hinzugefügt werden.  
Mit der Minus-Schaltfläche kann ein Eintrag aus der Liste entfernt werden.  
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann der ausgewählte logische Datenträger bearbeitet werden.

**Eingabemaske *Überwachung logischer Datenträger hinzufügen***

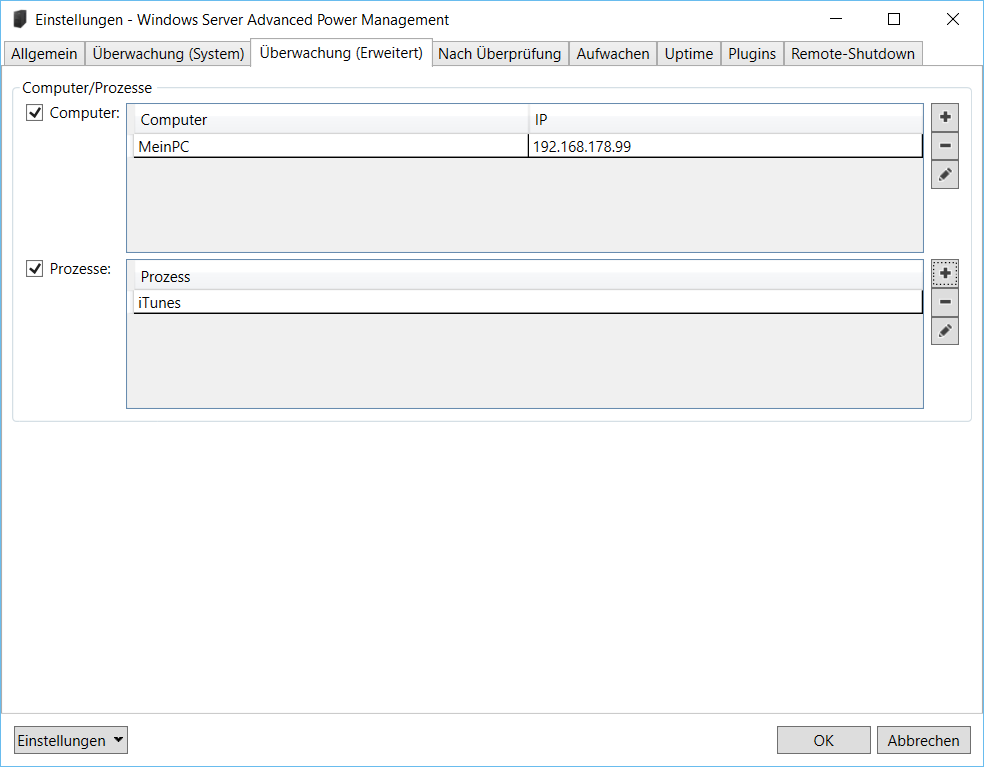
Dieser Dialog bietet die Möglichkeit, einen logischen Datenträger und die Auslastung in KB/s anzugeben, ab welcher eine Benutzung des Rechners signalisiert werden soll. Wird hier *Alle Laufwerke (gesamt)* gewählt, werden die Werte der einzelnen logischen Datenträger bei der Überprüfung aufaddiert.

Neben dem eingestellten Wert wird ebenfalls die aktuelle Auslastung des gerade gewählten logischen Datenträgers angezeigt.



### Überwachung (Erweitert)

Im Reiter *Überwachung (Erweitert)* können weitere Richtlinien zum Unterdrücken des Standby-Modus festgelegt werden.



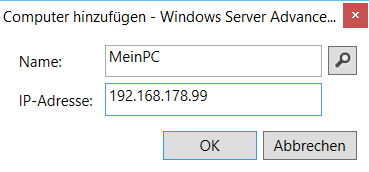
**Computer**

Hier können Computer oder andere netzwerkfähige Geräte angegeben werden. Wenn bei der Überprüfung mindestens ein Gerät über das Netzwerk erreichbar ist, wird der Standby-Modus unterdrückt. Diese Liste ist nicht auf Computer im Heimnetzwerk beschränkt, es können sämtliche netzwerkfähigen Geräte (wie z.B. Smartphones, netzwerkfähige TV-Geräte, Spielekonsolen, etc.) angegeben werden.

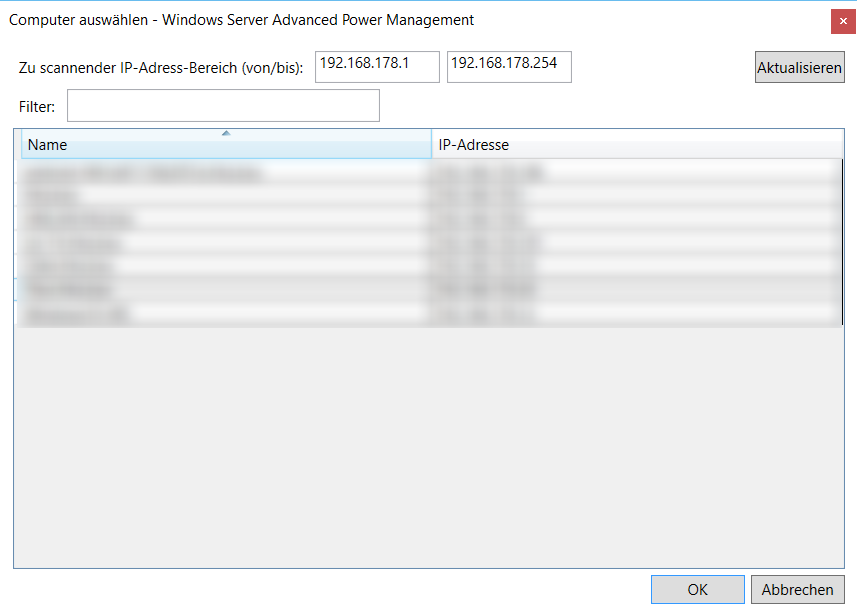
Durch die Plus-Schaltfläche kann ein Computer zur Liste hinzugefügt werden.  
Mit der Minus-Schaltfläche kann ein Netzwerk-Gerät aus der Liste entfernt werden.  
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann das ausgewählte Netzwerk-Gerät bearbeitet werden.

**Eingabemaske *Computer hinzufügen***

Wenn ein Computer hinzugefügt oder bearbeitet wird, erscheint eine Eingabemaske, in die der Name des Gerätes, über den es im Netzwerk angesprochen werden kann (z.B. „Mein PC“) und/oder eine IP-Adresse eingegeben werden kann.



Mit der Auswahl-Schaltfläche können aktive Netzwerk-Geräte im lokalen Netzwerk gesucht werden. Dabei wird standardmäßig immer ein bestimmter IP-Adress-Bereich durchsucht, der von der IP-Adresse des aktuellen Rechners abhängig ist. Wenn der Rechner beispielsweise die IP-Adresse 192.168.178.**20** hat, wird der Bereich von 192.168.178.**1** bis 192.168.178.**254** durchsucht. Der zu durchsuchende Adress-Bereich kann im Auswahl-Dialog geändert werden.  
Achtung: Das Durchsuchen von großen Netzwerk-Bereichen kann lange Zeit in Anspruch nehmen.



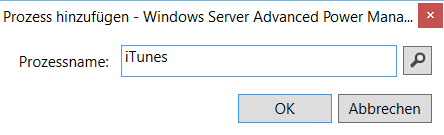
**Prozesse**

Die hier angegebenen Programme/Prozesse signalisieren ebenfalls eine Verwendung des Rechners.

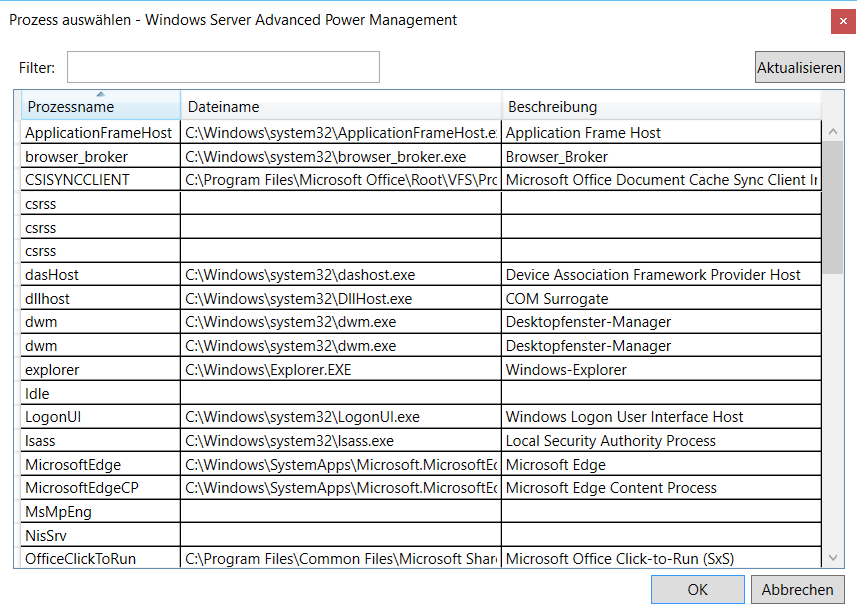
Durch die Plus-Schaltfläche kann ein Prozess zur Liste hinzugefügt werden.  
Durch die Minus-Schaltfläche kann ein Programm aus der Liste entfernt werden.  
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann der ausgewählte Prozess bearbeitet werden.

**Eingabemaske Prozess hinzufügen**

Wenn ein Prozess hinzugefügt oder bearbeitet wird, erscheint eine Eingabemaske, in die der Name des Prozesses angegeben werden kann. Der Name des Prozesses ist z.B. über den Windows Task-Manager zu ermitteln. Die Eingabe muss hierbei ohne Datei-Erweiterung (z.B. „.exe“) erfolgen. Wenn beispielsweise iTunes in die Liste der Programme hinzugefügt werden soll, ist hier „iTunes“ (und nicht „iTunes.exe“) anzugeben.

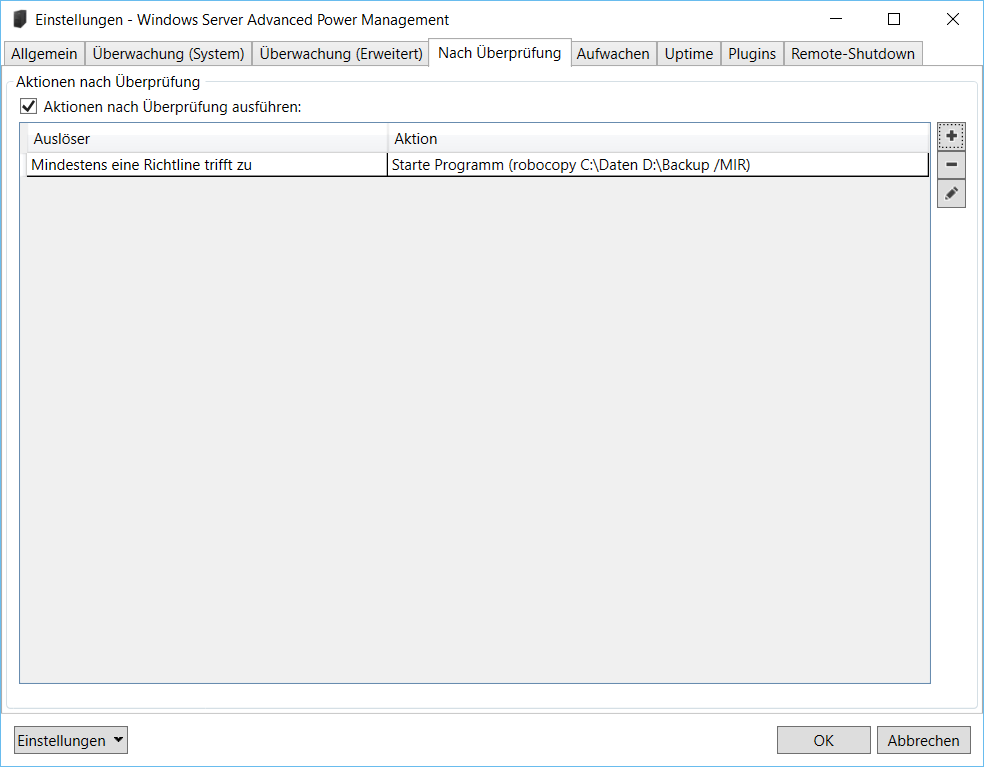


Mit Hilfe der Schaltfläche neben dem Eingabefeld kann ein weiterer Dialog geöffnet werden, aus dem aus der Liste der aktuell laufenden Prozesse ein Prozess ausgewählt werden kann.



### Nach Überprüfung

Im Reiter *Nach Überprüfung* können benutzerdefinierte Aktionen angegeben werden, die nach der Überprüfung der Richtlinien des Programms ausgeführt werden sollen.



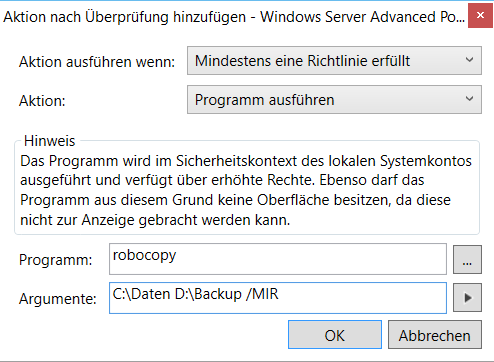
**Aktionen nach Überprüfung ausführen**

Diese Option gibt an, ob die Aktionen nach der Überprüfung der Programm-Richtlinien ausgeführt werden sollen. Wenn deaktiviert, werden die aufgeführten Aktionen niemals ausgeführt.

Durch die Plus-Schaltfläche kann eine Aktion zur Liste hinzugefügt werden.  
Mit der Minus-Schaltfläche wird eine gewählte Aktion aus der Liste entfernt.  
Mittels Doppelklick oder der Bearbeiten-Schaltfläche kann eine ausgewählte Aktion bearbeitet werden.

**Eingabemaske Aktion nach Überprüfung hinzufügen**

Wenn eine Aktion hinzugefügt oder bearbeitet werden soll, wird eine Eingabemaske aufgerufen, mit der die auszuführende Aktion spezifiziert werden kann.



**Aktion ausführen wenn**

Hier stehen zwei Optionen zur Auswahl: *Mindestens eine Richtlinie erfüllt* und *Keine Richtlinie erfüllt*. Die erste Einstellung gibt an, dass eine Aktion nur dann ausgeführt wird, wenn mindestens eine Richtlinie, die das Wechseln in den Standby-Modus verhindern würde, erfüllt ist. Die zweite Option sorgt dafür, dass die Aktion immer dann ausgeführt wird, wenn keine Richtlinie erfüllt ist.

**Aktion**

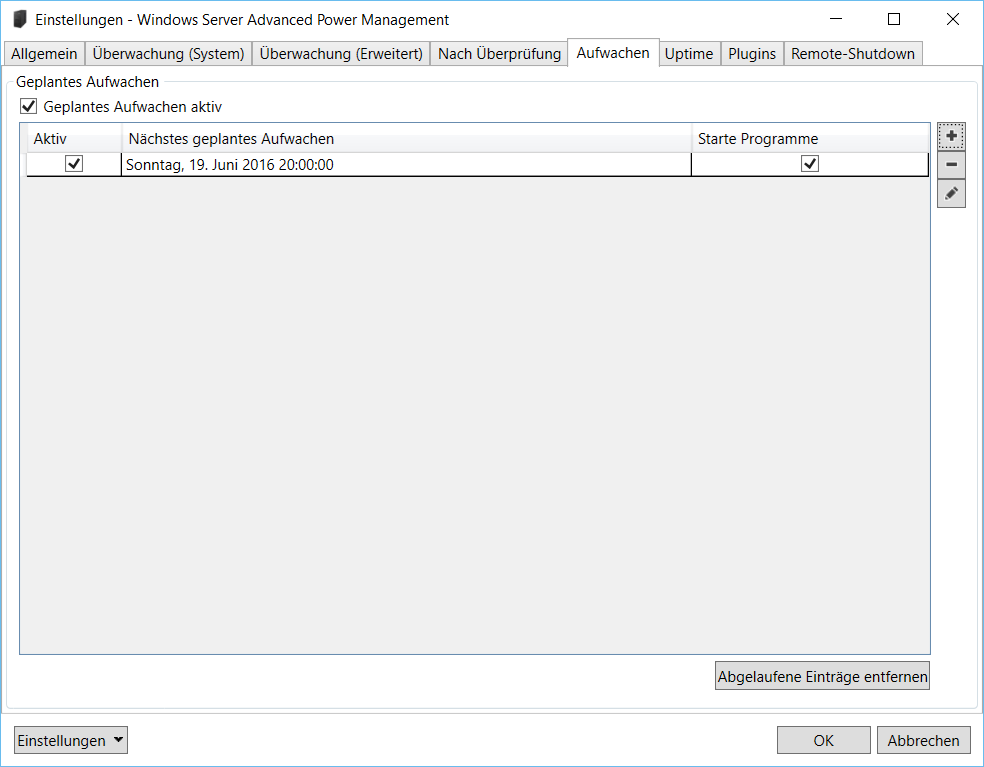
Hier wird die Aktion angegeben, die ausgeführt werden soll:

* *Standby*: Der Rechner wird in den Standby-Modus versetzt.
* *Ruhezustand*: Der Rechner wechselt in den Ruhezustand (Hibernate).
* *Herunterfahren*: Der Computer wird heruntergefahren.
* *Programm ausführen*: Es wird das Programm mit den angegebenen Parametern ausgeführt.

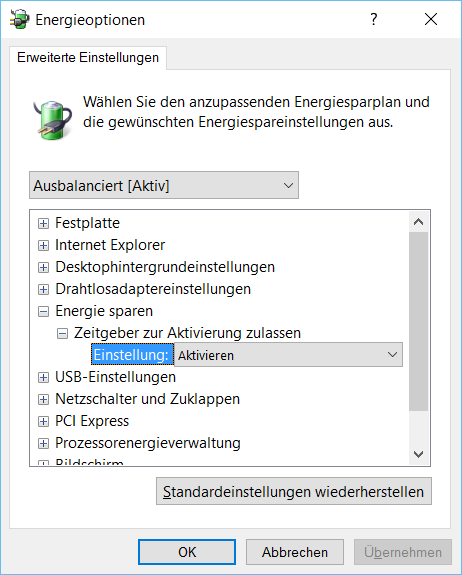
**Hinweis:** Es kann für einen Auslöser (*Mindestens eine Richtlinie erfüllt*/*Keine Richtlinie erfüllt*) nur eine Aktion definiert werden, die den Energie-Zustand des Rechners festlegt (*Standby*/*Ruhezustand*/*Herunterfahren*), da sich diese Aktionen gegenseitig ausschließen.

### Aufwachen

Im Reiter *Aufwachen* können die Einstellungen zum zeitgesteuerten Aufwachen aus dem Standby-Modus festgelegt werden.



Wichtig: Damit der Rechner automatisch aus dem Standby-Modus erwachen kann, muss in den Windows-Energieoptionen die Einstellung *Zeitgeber zur Aktivierung zulassen* aktiviert sein. Diese Option findet man unter *Systemsteuerung* 🡪 *Energieoptionen* 🡪 *Energiesparplaneinstellungen ändern* (beim gerade aktiven Energiesparplan) 🡪 *Erweiterte Energieeinstellungen ändern* 🡪 *Energie sparen* 🡪 *Zeitgeber zur Aktivierung zulassen*.



**Geplantes Aufwachen aktiv**

Wenn diese Option aktiviert ist, wird der Rechner nach den weiter unten Regeln zum Aufwachen automatisch aus dem Standby geweckt.

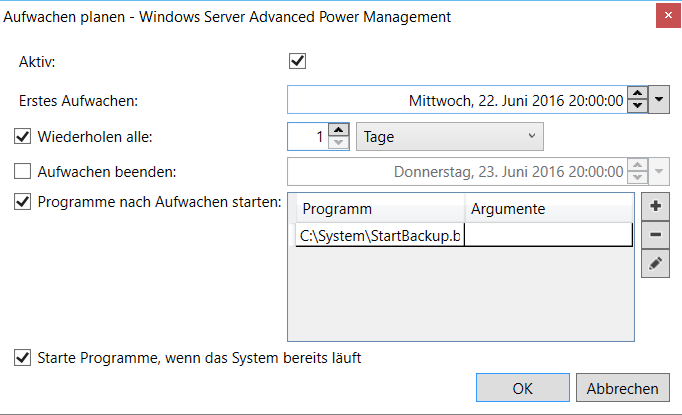
Durch die Plus-Schaltfläche kann ein Plan zum automatischen Aufwachen zur Liste hinzugefügt werden.  
Mit der Minus-Schaltfläche kann ein Plan aus der Liste entfernt werden.   
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann der ausgewählte Plan bearbeitet werden.

**Abgelaufene Einträge entfernen**

Mit dieser Schaltfläche werden sämtliche Einträge in der Liste entfernt, die bereits abgelaufen sind, d.h. die niemals dazu führen, dass der Rechner aus dem Standby aufgeweckt wird (z.B., weil der letzte Zeitpunkt zum Aufwachen bereits in der Vergangenheit liegt).

**Eingabemaske *Aufwachen planen***

Wenn ein Plan zum Aufwachen hinzugefügt oder bearbeitet wird, erscheint eine Eingabemaske, in die sämtliche Optionen zum automatischen Aufwachen eingegeben werden können.



**Aktiv**

Diese Option gibt an, ob der gewählte Plan zum Aufwachen aktiv sein soll. Wenn ein Plan zum Aufwachen deaktiviert wird, wird der Rechner nicht mehr entsprechend des Plans von WSAPM geweckt.

**Erstes Aufwachen**

Die Startzeit gibt den genauen Zeitpunkt an, an dem der Rechner aus dem Standby erwachen soll.

**Wiederholen alle**

Wenn diese Option aktiviert ist, kann ein Intervall definiert werden, welches sich auf die weiter oben eingestellte Startzeit bezieht. Der Rechner wird dann immer nach Ablauf dieses Intervalls aus dem Standby aufgeweckt.

Wenn die Startzeit z.B. mit dem 01. Januar 2014 um 13:00 angegeben ist und die Wiederholen-Option auf 1 Tag steht, wird der Rechner jeden Tag um 13:00 aufgeweckt, beginnend mit dem 01. Januar 2014.

Wenn diese Option deaktiviert wird, wird das System nur einmal zum Zeitpunkt, welcher unter *Erstes Aufwachen* angegeben wird, geweckt.

**Aufwachen beenden**

Hier wird der Zeitpunkt angegeben, an dem ein wiederholtes automatisches Aufwachen des Rechners beendet werden soll. Nach dem hier angegebenen Zeitpunkt wird der Rechner nicht mehr automatisch von WSAPM aus dem Standby-Modus aufgeweckt.

**Programme nach Aufwachen starten**

Die hier aufgeführten Programme werden nach dem geplanten Aufwachen automatisch mit den angegebenen Argumenten ausgeführt. Dies geschieht nicht nach jedem Aufwachen des Rechners, sondern nur dann, wenn der Computer vom einem von WSAPM geplanten Aufwachen geweckt wurde. Wenn ein Programm nach jedem Aufwachen gestartet werden soll (und nicht nur, wenn der Rechner durch ein geplantes Aufwachen von WSAPM gestartet wurde), sollte die Option *Programme nach Aufwachen starten* unter *Allgemein* genutzt werden (siehe [*Einstellungen – Allgemein*](#_Allgemein_1)).  
Die Programme werden ca. 30 Sekunden nach dem Aufwachen des Computers gestartet.  
Wenn mehrere Programme angegeben wurden, ist die Reihenfolge jedoch nicht festgelegt, mit der die Programme gestartet werden.

Wichtig: Die hier aufgeführten Programme werden im Sicherheitskontext des lokalen Systemkontos ausgeführt und besitzen daher erhöhte Rechte.  
Ebenso sollten diese keine Benutzeroberfläche anzeigen, da es nicht möglich ist, in diesem Kontext eine Oberfläche zur Anzeige zu bringen. D.h. Programme mit Benutzeroberfläche werden zwar gestartet, werden aber dem aktuell angemeldeten Benutzer nicht angezeigt.

Mit der Plus-Schaltfläche kann ein zu startendes Programm hinzugefügt werden. Es erscheint eine Eingabemaske, in die der Pfad zum Programm (die Angabe einer Batch-Datei ist ebenfalls möglich) und Argumente für den Programmstart eingegeben werden können. Das Programm kann mit den angegebenen Argumenten durch die Abspielen-Schalfläche testweise gestartet werden.

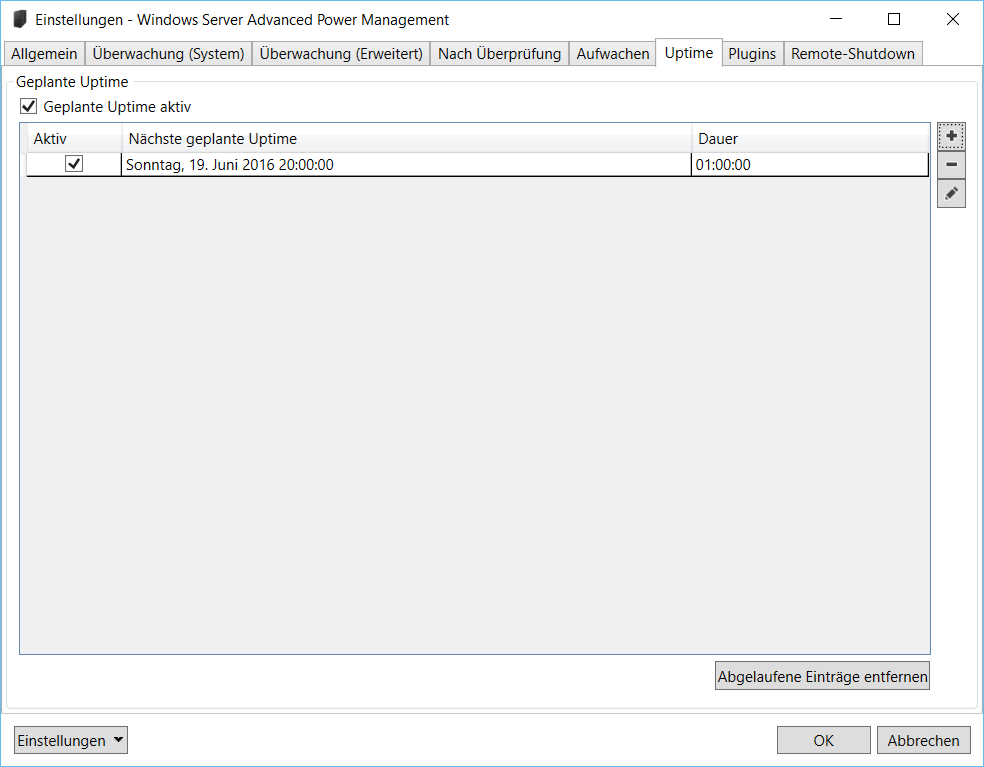
**Starte Programme, wenn das System bereits läuft**

Wenn diese Option aktiviert wurde, werden die weiter oben aufgeführten Programme auch dann gestartet, wenn der Rechner zum Zeitpunkt des geplanten Aufwachens gerade läuft. Wenn die Programme nur dann gestartet werden sollen, wenn der Rechner zum geplanten Zeitpunkt auch tatsächlich aus dem Standby erwacht, muss diese Option deaktiviert werden.

### Uptime

In diesem Reiter können geplante Uptimes definiert werden, d.h. Zeiten, zu den der Rechner nicht in den Standby-Modus wechseln soll (unabhängig von allen anderen Richtlinien).

**Wichtig:** Bei Eintreten einer Uptime wird der Rechner nicht automatisch gestartet. Somit greift die Uptime nur dann, wenn der Rechner zu dieser Zeit gerade läuft. Wenn der Rechner zu diesem Zeitpunkt auch automatisch gestartet werden soll, muss dies im Reiter Aufwachen definiert werden (siehe [*Einstellungen - Aufwachen*](#_Aufwachen_1)).



**Geplante Uptime aktiv**

Wenn diese Option deaktiviert ist, werden sämtliche geplanten Uptimes deaktiviert.

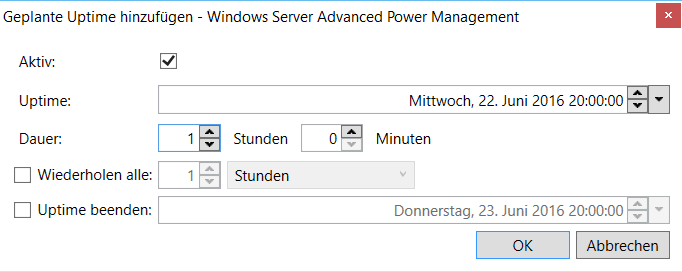
Durch die Plus-Schaltfläche kann eine geplante Uptime zur Liste hinzugefügt werden.  
Mit der Minus-Schaltfläche kann ein Eintrag der Liste entfernt werden.   
Mit einem Doppelklick oder der Bearbeiten-Schaltfläche kann die ausgewählte Uptime bearbeitet werden.

**Abgelaufene Einträge entfernen**

Wenn eine geplante Uptime in Zukunft auf Grund der Konfiguration nicht mehr ausgeführt werden wird, können solche Einträge mit einem Klick aus der Liste entfernt werden.

**Eingabemaske *Uptime hinzufügen***

Wenn eine geplante Uptime hinzugefügt oder bearbeitet wird, können durch dieses Fenster sämtliche Optionen eingestellt werden.



**Aktiv**

Gibt an, ob die jeweilige Uptime aktiv ist. Wenn diese Option deaktiviert wurde, wird diese Uptime ignoriert werden.

**Uptime**

Legt das Datum und die Zeit fest, wann die Uptime beginnen soll.

**Dauer**

Definiert die Dauer der jeweiligen Uptime. Der Computer wird nach dem Beginn der Uptime für diese Dauer nicht in den Standby-Modus wechseln.

**Wiederholen alle**

Wenn die Uptime in bestimmten Intervallen wiederholt werden soll (z.B. alle 5 Stunden oder alle 2 Tage), kann dies hier angegeben werden.

Wenn diese Option deaktiviert ist, wird die Uptime nur einmal zum definierten Zeitpunkt für die angegebene Dauer ausgeführt.

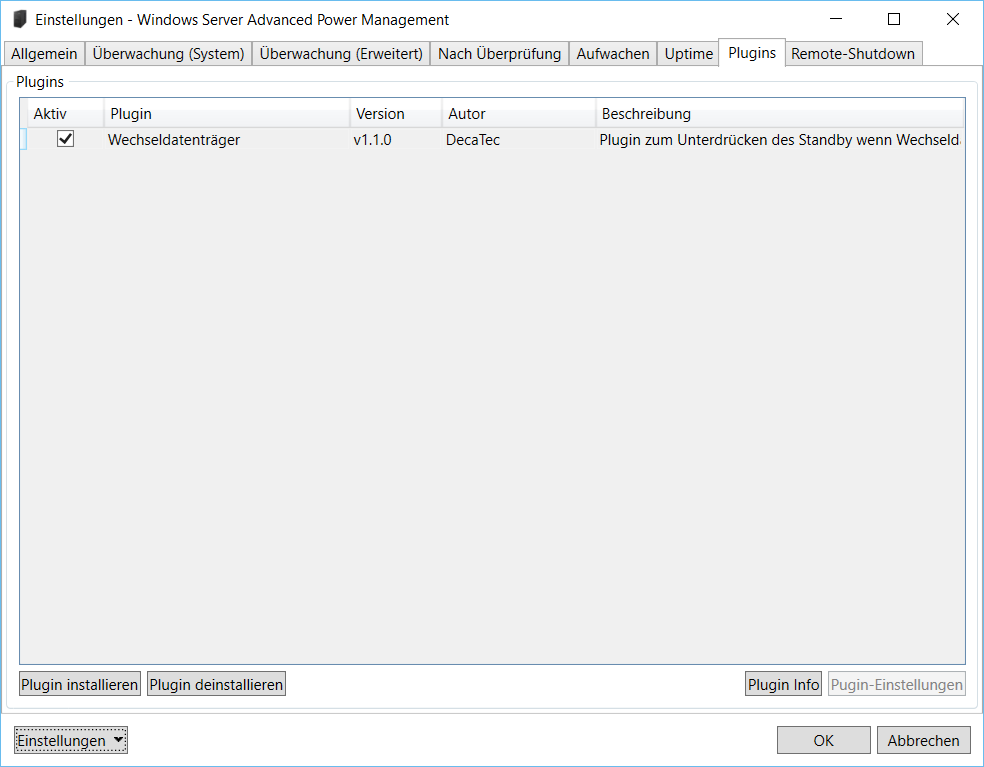
**Uptime beenden**

Wenn eine Uptime für die kommenden Tage um jeweils 22:00 geplant werden soll, allerdings nach 5 Tagen nicht mehr benötigt wird, kann hier ein Zeitpunkt definiert werden, ab dem die Uptime nicht mehr aktiv sein soll.

### Plugins

WSAPM bietet eine Schnittstelle zur Entwicklung eigener Plugins. Mit Hilfe von Plugins kann WSAPM auf einfache Art und Weise mit weiteren Überprüfungs-Richtlinien erweitert werden.

Im Reiter *Plugins* können Plugins für WSAPM verwaltet werden.



Eine genaue Beschreibung der Plugin-Schnittstelle für Entwickler ist im Abschnitt [*Plugin-Entwicklung*](#_Plugin-Entwicklung_1) zu finden.

**Liste der Plugins**

Die Liste zeigt die aktuell installieren Plugins an. Damit ein installiertes Plugin aktiv ist, d.h. die Überprüfungs-Richtlinien des Plugins aktiv in die Überprüfungen von WSAPM mit einbezogen werden, muss die jeweilige Checkbox vor dem Plugin aktiviert sein.

**Plugin installieren**

Mit der Schaltfläche *Plugin installieren* kann ein neues Plugin installiert werden. Im darauffolgenden Auswahldialog kann das zu installierende Plugin angegeben werden. Plugins für WSAPM werden als Zip-Dateien ausgeliefert, daher können auch nur Zip-Dateien ausgewählt werden. Nach der Auswahl wird das Plugin installiert. Dazu muss WSAPM neu gestartet werden.  
Man sollte nach der Plugin-Installation nicht vergessen, nochmal in die Einstellungen zu gehen und das gerade installierte Plugin zu aktivieren, damit dieses in die Überprüfungs-Routine von WSAPM mit einbezogen wird.

**Plugin deinstallieren**

Mit der Schaltfläche *Plugin deinstallieren* kann ein ausgewähltes Plugin entfernt werden. Auch zum Deinstallieren von Plugins muss WSAPM neu gestartet werden.

Wenn ein Plugin (im Fehlerfall) nicht über den Button deinstalliert werden kann, ist es auch möglich ein Plugin manuell zu deinstallieren: Dazu muss zunächst einmal der Windows Server Advanced Power Management Service gestoppt werden. Dies geschieht am einfachsten über den Link *Stopp* im Hauptfenster der Anwendung (siehe [*Bedienung - Hauptfenster*](#_Hauptfenster)). Die Anwendung muss dann ebenfalls beendet werden. Anschließend wechselt man in das Verzeichnis, in dem die WSAPM Plugins installiert sind (für gewöhnlich *C:\Program Files (x86)\Windows Server Advanced Power Management\Plugins*). Hier kann der Order des zu deinstallierenden Plugins manuell gelöscht werden. Ebenso sollten die Einstellungen des Plugins gelöscht werden, falls das Plugin eigene Einstellungen hat. Dazu einfach in den Ordner wechseln, in dem WSAPM die Plugin-Einstellungen speichert (*C:\ProgramData\Windows Server Advanced Power Management\Plugins*) und hier den Ordner des zu löschenden Plugins entfernen. Anschließend kann der Windows Server Advanced Power Management Dienst wieder gestartet werden.

**Plugin Info**

Mit dieser Schaltfläche werden weitere Informationen zum ausgewählten Plugin angezeigt.

**Plugin-Einstellungen**

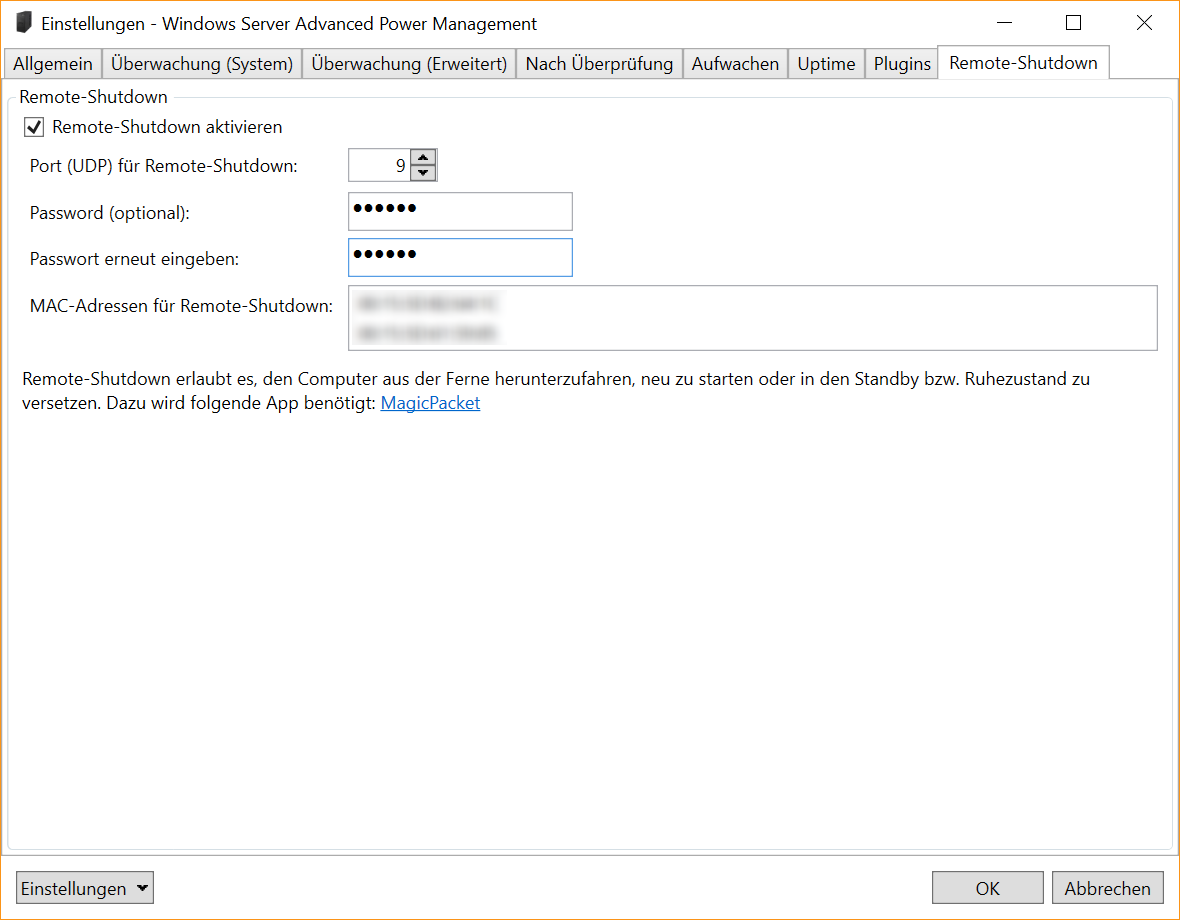
Manche Plugins für WSAPM bieten eigene Einstellungen an. Falls das in der Liste gewählte Plugin solche Einstellungen anbietet, können diese mit der Schaltfläche *Plugin-Einstellungen* bearbeitet werden.  
Wenn das gewählte Plugin keine Einstellungen bietet, ist diese Schaltfläche nicht aktiv.

### Remote-Shutdown

Im Reiter *Remote-Shutdown* können Optionen bzgl. Remote-Shutdown konfiguriert werden. Mit Remote-Shutdown kann der Computer aus der Ferne heruntergefahren, neu gestartet, oder in den Standby- oder Ruhezustand versetzt werden.

Um dieses Feature zu aktivieren, muss die entsprechende Option eingeschaltet und eine entsprechende App oder ein Programm verwendet werden, welches die Remote-Shutdown-Befehle senden kann – z.B. [MagicPaket](https://decatec.de/software/magicpacket/) (verfügbar für Windows und Windows Phone). Eine detaillierte Beschreibung, wie Remote-Shutdown mittels Windows Server Advanced Power Management und MagicPacket realisiert werden kann, ist in [diesem Artikel](https://decatec.de/remote-shutdown-waspm-magicpacket/) zu finden.

*Hinweis für Entwickler:* Das Protokoll für den Remote-Shutdown mittels WSAPM ist offengelegt (siehe [*Windows Server Advanced Power Management für Entwickler – Remote-Shutdown*](#_Remote-Shutdown)). Somit kann jeder Entwickler eigene Anwendungen/Apps entwickeln, mit der ein Remote-Shutdown möglich ist.



**Remote-Shutdown aktivieren**

Wenn diese Option eingeschaltet wird, lauscht WSAPM im Netzwerk nach Remote-Shutdown-Befehlen (UDP). Wenn der Computer nicht aus der Ferne heruntergefahren werden soll, ist diese Option zu deaktivieren.

**Port (UDP) für Remote-Shutdown**

Gibt den zu verwendenden UDP-Port für Remote-Shutdown an (Standard: Port 9). Derselbe Port muss in der entsprechenden App eingestellt sein.

**Passwort**

Optional kann ein Passwort für Remote-Shutdown festgelegt werden, welches dann auch in der entsprechenden App oder Anwendung angegeben werde muss, die die Befehle senden soll.

Wichtig:Das Passwort für Remote-Shutdown stellt dabei keinen echten Sicherheits-Aspekt dar, da das Passwort als Hash-Wert (d.h. unverschlüsselt) über das Netzwerk übertragen wird. Daher sollten hier keine Passwörter zum Einsatz kommen, die auch an anderer Stelle verwendet werden und deren Verlust ein Sicherheits-Risiko darstellen würde.

**MAC-Adressen für Remote-Shutdown**

Diese Liste zeigt die MAC-Adressen aller aktiver Netzwerkkarten im PC. Der Computer wird auf alle Remote-Shutdown-Befehle reagieren, die an diese MAC-Adressen gerichtet sind.  
Über das Kontextmenu der Liste kann eine einzelne Adresse in die Zwischenablage kopiert werden.

# Beispielszenarien

Dieser Abschnitt zeigt ein paar Beispiele, wie Windows Server Advanced Power Management konfiguriert werden kann, um bestimmte Szenarien abzudecken.

**Server für die geplante Ausführung eines Programms aus dem Standby-Modus wecken**

Manche Programme, wie z.B. Virenscanner können so konfiguriert werden, dass diese regelmäßig geplanten Aktivitäten ausführen (z.B. Virenscan jeden Tag um 02:00). Viele dieser Programme sind jedoch nicht in der Lage, den Rechner zu wecken, falls sich dieser zum geplanten Zeitpunkt im Standby befindet. Die führt meist dazu, dass geplante Aktivitäten gar nicht, oder nur dann ausgeführt werden, wenn der Rechner zufälligerweise genau zu diesem Zeitpunkt läuft. Hier kann WSAPM aushelfen: Zunächst muss das Programm, welches nach einem Zeitplan läuft, in die Liste der Programme eingetragen werden (siehe [*Einstellungen - Überwachung (Erweitert)*](#_Überwachung_(Erweitert))). In diesem Fall wäre das beispielsweise „avscan“ (Virenscanner von [Avira](http://www.avira.com/)). Als nächstes wird dann das automatische Aufwachen in WSAPM aktiviert (siehe [*Einstellungen - Aufwachen*](#_Aufwachen_1)). Dies sollte immer ein paar Minuten vor der eigentlichen geplanten Aktivität passieren, damit der Rechner genug Zeit hat, komplett aus dem Standby zu erwachen. In diesem Beispiel wäre tägliches Aufwachen um 01:55 optimal. Der Rechner erwacht nun jeden Tag um 01:55. Um 02:00 startet der Virenscanner mit seinem Scan-Vorgang. Dies wird wiederum von WSAPM registriert und der Standby daraufhin unterdrückt. Wenn der Scan-Vorgang beendet ist und der Prozess des Virenscanners nicht mehr aktiv ist, geht der Rechner wieder nach den definierten Richtlinien in den Standby.

**Streaming-Server während Streaming nicht in den Standby-Modus versetzen**

Wenn auf dem Rechner eine Streaming-Software wie z.B. [Twonky](http://twonky.com/) oder [AssetUPnP](http://www.dbpoweramp.com/asset-upnp-dlna.htm) installiert ist, können Medien auf Streaming-Clients (z.B. PlayStation) gestreamt werden. Es gibt in WSAPM keine direkte Option, um bei aktivem Streaming den Standby-Modus zu unterdrücken. Auch die Option *Überwache Zugriff auf Netzwerk-Freigaben* hilft hier nicht weiter, da Streaming im technischen Sinne keinen Zugriff auf Netzwerk-Freigaben darstellt. Dennoch ist Streaming im Grunde genommen nur ein Abspielen von Medieninhalten über das Netzwerk. Somit kann ein aktives Streaming durch einen Netzwerk-Upload am Streaming-Server festgestellt werden. Um hierfür den richtigen Wert in KBit/s zu ermitteln, sollte das Streaming gerade aktiv sein. Dann kann in den Einstellungen eine Überwachung der entsprechenden Netzwerkkarte hinzugefügt werden. Den wichtigen Wert für die Upload-Auslastung kann man über die Anzeige der aktuellen Auslastung ablesen (siehe [*Einstellungen - Überwachung (System)*](#_Überwachung_(System))).  
Man sollte dabei nur beachten, dass das Streaming von beispielsweise HD-Filmen viel mehr Bandbreite erfordert als das Streamen einfacher MP3-Dateien. Daher sollte man den Wert für die Upload-Netzwerkauslastung am besten mit dem Streamen von kleinen Dateien ermitteln.

**Manuelles Unterdrücken des Standby-Modus bei Bedarf**

Wenn ein manuelles Unterdrücken des Standby-Modus bei Bedarf erforderlich ist, kann dies sehr einfach realisiert werden. Einfach ein Programm, mit dem im Moment nicht gearbeitet wird (z.B. Notepad oder WSAPM selbst) in die Liste der Programme (siehe [*Einstellungen - Überwachung (Erweitert)*](#_Überwachung_(Erweitert))) in den Optionen einfügen. Wenn nun eine manuelle Unterdrückung des Standby-Modus erforderlich ist, muss einfach das hier angegebene Programm gestartet werden. Wird das Programm anschließend beendet, wechselt der Rechner wieder nach den vorgegebenen Richtlinien in den Standby-Modus.

**Den Zustand des Rechners (Standby/Ruhezustand/etc.) selbst verwalten**

Windows Server Advanced Power Management wurde ursprünglich so konzipiert, dass es mit den Windows-Energieoptionen zusammenarbeitet - und den Zustand des Rechners dabei nicht selbst verwaltet. Jedoch gab es die genau diese Anforderung: WSAPM soll den Zustand des Rechners selbst verwalten können.  
Genau dies ist mit den Optionen *Nach Überprüfung* möglich (siehe [*Einstellungen - Nach Überprüfung*](#_Nach_Überprüfung)). Wenn beispielsweise der Ziel-Zustand des Rechners der Ruhemodus sein soll, kann hier angegeben werden, dass der Rechner in den Ruhemodus wechseln soll, wenn keine der zu überprüfenden Richtlinien erfüllt ist. In diesem Fall können auch die Windows-Energieoptionen dahingehend geändert werden, dass der Rechner nicht mehr automatisch in den Energiesparmodus versetzt wird.

Dennoch wird empfohlen, den Zustand des Rechners nicht auf diese Art und Weise zu forcieren. Die bessere Lösung stellt nach wie vor die Zusammenarbeit mit den Windows-Energieoptionen und der Unterdrückung des Standby-Modus in bestimmten Fällen dar.

# Fehlerbehebung

Dieser Abschnitt liefert einige Lösungsansätze, wie bei bestimmten Problemen mit Windows Server Advanced Power Management vorgegangen werden kann.

**WSAPM scheint nicht richtig zu funktionieren**

Falls WSAPM nicht wie erwartet funktioniert, sollte man zunächst einmal die den Log-Modus auf *Detailliert* stellen (siehe [*Einstellungen - Allgemein*](#_Allgemein_1)), damit alle Aktivitäten von WSAPM geloggt werden. Dadurch wird evtl. ersichtlich, wo das eigentliche Problem liegt.

Unter Betriebssystemen ab Windows 7 gibt es auch eine Möglichkeit, die Funktionsweise von WSAPM in Bezug auf Unterdrückung des Standby-Modus und zeitgesteuertem Aufwecken zu kontrollieren: das in Windows integrierte Tool *powercfg*. Dazu muss die Kommandozeile mit Administrator-Rechten gestartet werden und folgendes eingegeben werden:

* *powercfg -requests*: Über diesen Befehl werden alle Programme aufgeführt, welche momentan ein Wechseln in den Standby-Modus verhindern. Wenn eine WSAPM-Richtlinie zur Unterdrückung des Standby-Modus erfüllt ist, sollte man hier einen Eintrag von WSAPM finden.
* *powercfg -waketimers*: Dieser Befehl zeigt alle Timer an, die Windows aus dem Standby-Modus wecken sollen und gerade aktiv sind. Wenn automatisches Aufwachen aus dem Standby-Modus in WSAPM konfiguriert ist, sollte hier ein Eintrag von WSAPM zu finden sein.

Unter Windows Vista oder älteren Windows-Versionen bietet das Tool *powercfg* keinen von beiden Parametern, daher kann die Funktionsweise von WSAPM hier nicht auf diese Art und Weise kontrolliert werden.

**WSAPM gibt Fehlermeldungen aus („Die Kategorie ist nicht vorhanden“) wenn die Auslastung von CPU/Speicher/Netzwerk/logische Laufwerken gelesen wird bzw. kann die passende Netzwerkkarte/das passende logische Laufwerk nicht ermitteln**

WASPM nutzt häufig WMI ([Windows Management Instrumentation](http://de.wikipedia.org/wiki/Windows_Management_Instrumentation)) zur Ermittlung von Werten, die für die Überprüfung wichtig sind (z.B. Auslastung von CPU- oder Speicher-Auslastung, etc.). Falls in diesem Zusammenhang Fehler auftreten (z.B. Fehlermeldung: „Kategorie ist nicht vorhanden“) oder die im PC vorhandenen Netzwerkkarten/logischen Laufwerke nicht gefunden werden können, ist die Wahrscheinlichkeit gegeben, dass die entsprechenden Windows-Leistungsindikatoren defekt sind. Dies kann die unterschiedlichsten Gründe haben, z.B. wenn ein anderes Programm die Leistungsindikatoren in unzulässiger Weise verändert hat.

Diese Leistungsindikatoren können allerdings von Windows automatisch repariert werden. Dies geschieht über die Kommandozeile (ausführen als Administrator). Es bestehen folgende Möglichkeiten:

1. Durch folgenden Befehl werden die Leistungsindikatoren aus einem (von Windows erstelltem) Backup wiederhergestellt: *cd c:\windows\system32  
   lodctr /R:PerfStringBackup.ini*Danach sollte der Rechner neu gestartet werden.
2. Falls das Problem dadurch nicht gelöst werden kann, können die Leistungsindikatoren durch folgenden Befehl komplett wiederhergestellt werden:  
   *cd c:\windows\system32  
   lodctr /R*Danach sollte der Rechner wiederum neu gestartet werden.
3. Falls das Problem nun immer noch besteht, sollte nach diesem [Microsoft Knowledge Base-Artikel](http://support.microsoft.com/kb/300956) vorgegangen werden

Anschließend sollte WASPM wieder ordnungsgemäß funktionieren.

**Die Überwachung kann nicht aktiviert werden**

Wenn WSAPM nicht richtig läuft und beispielsweise die Überwachung nicht richtig funktioniert, kann es sein, dass der zu Grunde liegende Windows-Dienst nicht läuft. Dies kann über das MMC-Snap-In *services.msc* kontrolliert werden. Dies kann über *Systemsteuerung* 🡪 *Verwaltung* 🡪 *Dienste* gestartet werden. Alternativ Windows-Taste + R („Ausführen“) und hier *services.msc* eingeben. Hier sollte nun ein Eintrag *Windows Server Advanced Power Management* zu finden sein. Fall dies nicht der Fall sein sollte, wurde WSAPM nicht richtig installiert und die Installation sollte wiederholt werden.

Wichtig für das einwandfreie Funktionieren des Dienstes sind folgende Einstellungen:

* *Status*: *Gestartet* – wenn der Dienst nicht läuft, findet auch keinerlei Überwachung statt.
* *Starttyp*: *Automatisch* – dies sorgt dafür, dass der Dienst bei Systemstart automatisch gestartet wird.
* *Anmelden als*: *Lokales System* – der Dienst muss unter diesem System-Konto laufen um die benötigten Rechte zur Überwachung und zum Steuern des Standby-Modus zu haben.

Prinzipiell sollte es keinen Grund geben, etwas an den Einstellungen des Windows-Dienstes zu verändern. Sämtliche Einstellungen werden hier einmalig bei der Installation gesetzt und müssen nachträglich nicht verändert werden.

**Das zeitgesteuerte Aufwachen aus dem Standby-Modus funktioniert nicht.**

Hier sollte kontrolliert werden, ob die Option *Zeitgeber zur Aktivierung zulassen* in den Windows-Energieoptionen auf *Aktiviert* steht (siehe [*Einstellungen - Aufwachen*](#_Aufwachen_1)). Wenn diese Funktion deaktiviert ist, kann Windows generell nicht zeitgesteuert aus dem Standby-Modus erwachen.

**Das Starten von Programmen nach dem Aufwachen aus dem Standby funktioniert anscheinend nicht richtig. WSAPM gibt im Log an, dass das Programm gestartet wurde, es wird aber auf dem Desktop nirgends angezeigt**

Die Programme, die nach dem Aufwachen aus dem Standby ausgeführt werden, laufen im Sicherheitskontext des lokalen Systemkontos. In diesem Kontext ist es nicht möglich, eine Benutzeroberfläche anzuzeigen. D.h. Programme mit Benutzeroberfläche werden zwar korrekt gestartet, werden dem aktuell angemeldeten Benutzer nicht angezeigt. Daher sollte man nur Programme von WSAPM starten lassen, die keine Benutzeroberfläche besitzen und somit im Hintergrund ablaufen.

**Die Log-Einträge erscheinen in einer anderen Sprache als die übrigen Programmteile**

Dies hat seine Ursache darin, dass der WSAPM Dienst unter dem Konto *Lokales System* läuft. Dies ist ein System-Konto, welches bei der Installation von Windows automatisch angelegt wird. Daher erhält dieses Konto die Regions- und Sprachoptionen, welche bei der Installation von Windows angegeben wurden. Genau diese Sprachoptionen verwendet dann auch der WSAPM Dienst, der für das Schreiben der Log-Einträge verantwortlich ist.  
Es ist kein Problem, die Sprache des Betriebssystems nachträglich für einen Benutzer umzustellen (man muss lediglich ein sog. MUI-Paket installieren), das Konto *Lokales System* ist davon allerdings nicht betroffen. In genau diesem Fall wird die Oberfläche von WSAPM in der eingestellten Sprache des Benutzers angezeigt, die Log-Einträge werden allerdings weiterhin in der Sprache erstellt, die bei der ursprünglichen Windows-Installation angegeben wurde.

**Ein Plugin lässt sich nicht über die Schaltfläche *Plugin deinstallieren* entfernen, es erscheint nur eine Fehlermeldung**

Wenn ein Plugin nicht ordnungsgemäß deinstalliert werden kann, liegt das vermutlich an einem Fehler im Plugin selbst. In diesem Fall kann man das Plugin auch manuell deinstallieren: Zunächst ist dafür der Windows Server Advanced Power Management Dienst zu stoppen (siehe [*Bedienung - Hauptfenster*](#_Hauptfenster)) und die Anwendung zu beenden. Anschließend wechselt man in das Verzeichnis, in dem WSAPM-Plugins installiert werden (für gewöhnlich *C:\Program Files (x86)\Windows Server Advanced Power Management\Plugins*). Hier kann der Order des zu deinstallierenden Plugins manuell gelöscht werden. Ebenso sollten die Einstellungen des Plugins gelöscht werden, falls das Plugin eigene Einstellungen hat. Dazu einfach in den Ordner wechseln, in dem WSAPM die Plugin-Einstellungen speichert (*C:\ProgramData\Windows Server Advanced Power Management\Plugins*) und hier den Ordner des zu löschenden Plugins entfernen. Abschließend kann der WSAPM Dienst wieder gestartet werden.

**Die Funktion Remote-Shutdown funktioniert nicht**

Für Remote-Shutdown muss diese Option zunächst einmal in den Einstellungen von WSAPM aktiviert und ein UDP-Port gewählt werden. Ebenso ist ein Programm oder eine App notwendig, die genau die Befehle sendet, die WSAPM für Remote-Shutdown erwartet (z.B. [MagicPacket](https://decatec.de/software/magicpacket/)).

Falls Remote-Shutdown dann immer noch nicht funktioniert, sollte kontrolliert werden, ob evtl. eine Firewall den Netzwerk-Verkehr zwischen der App und Windows Server Advanced Power Management blockiert. Während der Installation von WSAPM wird dafür extra eine Ausnahme in der Windows Firewall eingerichtet. Falls eine Firewall eines Drittanbieters zum Einsatz kommt, muss eine entsprechende Freigabe hier manuell erfolgen.

Eine detaillierte Beschreibung, wie Remote-Shutdown mittels Windows Server Advanced Power Management und MagicPacket realisiert werden kann, ist in [diesem Artikel](https://decatec.de/remote-shutdown-waspm-magicpacket/) zu finden

# FAQ

Dieser Abschnitt liefert Antworten auf einige häufig gestellten Fragen über Windows Server Advanced Power Management.

**Was ist ein Windows-Dienst und warum benötigt WSAPM einen solchen Dienst?**

Ein Windows-Dienst (oder auch Systemdienst genannt) ist ein Programm, welches ohne Benutzeroberfläche läuft und häufig wichtige Low-Level-Funktionalitäten auf Windows-Rechnern bereitstellt. Ein großer Unterschied zu Desktop-Applikationen ist, dass ein Dienst auch ohne Benutzeranmeldung läuft. Aus diesem Grund ist die Hauptfunktionalität von WSAPM auch in einem Windows-Dienst implementiert. Weitere Informationen über Windows-Dienste gibt es bei [Wikipedia](http://de.wikipedia.org/wiki/Windows-Systemdienst).

**Wo werden die Programmeinstellungen gespeichert?**

Die Einstellungen von WSAPM, wie auch die Log-Datei, sind *unter C:\ProgramData\Windows Server Advanced Power Management* zu finden. Das Verzeichnis wird standardmäßig von Windows ausgeblendet und kann in den Windows-Ordneroptionen mit der Einstellung *Ausgeblendete Dateien, Ordner und Laufwerke anzeigen* sichtbar gemacht werden.

**Unter *Systemsteuerung* 🡪 *Programme und Funktionen* wird angezeigt, dass WSAPM über 90 MB Speicherplatz benötigt. Warum benötigt das Programm so viel Speicher auf der Festplatte?**

WSAPM benötigt eigentlich nur knapp 5 MB Speicherplatz auf der Festplatte. Die Angabe unter *Programme und Funktionen* bezieht allerdings auch das benötigte .NET Framework mit ein, welches erheblich mehr Speicherplatz benötigt. Da das .NET Framework allerdings in vielen Fällen bereits auf dem Rechner installiert sein wird, werden durch die Installation von WSAPM praktisch nur die oben erwähnten 5 MB beansprucht.

# Windows Server Advanced Power Management für Entwickler

Der folgende Teil des Handbuchs richtet sich an Entwickler, die Windows Server Advanced Power Management mittels Plugins erweitern möchten oder die Remote-Shutdown-Funktionalität des Programms in eigenen Anwendungen oder Apps nutzen möchten.

Diejenigen, die Windows Server Advanced Power Management lediglich nutzen möchten und nicht selbst programmieren wollen, können diesen Abschnitt getrost überspringen.

## Plugin-Entwicklung

Windows Server Advanced Power Management bietet eine Schnittstelle zur Entwicklung eigener Plugins. Mit Hilfe von Plugins kann WSAPM auf einfache Art und Weise mit weiteren Überprüfungs-Richtlinien erweitert werden.

Da WSAPM in C# und dem .NET Framework implementiert wurde, werden auch Plugins mit C# oder eine beliebigen .NET Framework kompatiblen Sprache programmiert. Alle Beispiele werden an Hand von Visual Studio 2013 beschrieben, sollten aber auch auf andere Entwicklungsumgebungen übertragbar sein.

Im folgenden Abschnitt wird diese Plugin-Schnittstelle von WSAPM und die Vorgehensweise zur Entwicklung eigener Plugins erklärt.

### Normale und erweiterte Plugins

WSAPM kann mit zwei Arten von Plugins erweitert werden:

* Einfach Plugins: Dies sind Plugins, welche im Grunde genommen nur eine Überprüfungs-Routine bieten, die bei der Überprüfung der Richtlinien von WSAPM mit aufgerufen wird.
* Erweiterte Plugins: Diese unterstützen zusätzlich zur Funktionalität von einfachen Plugins auch eigene Einstellungen und die dazu gehörende Oberfläche.

Bei der Entwicklung eigener Plugins sollte man grundsätzlich den einfachen Plugin-Typ wählen. Dies macht die Entwicklung um einiges einfacher und auch schneller. Einzig und allein wenn das Plugin später eigene (von WSAPM unabhängige) Einstellungen und eine Oberfläche haben soll, mit der diese Einstellungen verändert werden können, sollte das Plugin als erweitertes Plugin implementiert werden.

### Schnelleinstieg

Damit der Einstieg in die Entwicklung eigener Plugins für WSAPM möglichst einfach fällt, gibt es vorgefertigte Visual Studio Solutions zum Download, die als Basis für eigene Entwicklungen dienen können. Es wird empfohlen, mit diesen vorgefertigten Solutions zu beginnen, da diese auch weiteren Erläuterungen enthalten.

* Vorgefertigte Visual Studio Solution, welche ein einfaches Plugin implementiert: [Download](https://decatec.de/downloads/wsapm/plugintemplates/WsapmPluginTemplate.zip)
* Vorgefertigte Visual Studio Solution, welche ein erweitertes Plugin implementiert: [Download](https://decatec.de/downloads/wsapm/plugintemplates/WsapmPluginAdvancedTemplate.zip)

### Entwicklung einfacher Plugins

Es folgt eine Schritt für Schritt Anleitung zur Entwicklung einfacher Plugins für WSAPM.  
Wenn die vorgefertigte Visual Studio Solution als Basis genommen wird (siehe [*Plugin-Entwicklung - Schnelleinstieg*](#_Schnelleinstieg)), sind darin bereits viele Schritte erledigt.

1. Neues Visual Studio Projekt erstellen: Da ein WSAPM Plugin aus mindestens einer DLL besteht, wird ein neues Visual Studio Projekt als Klassenbibliothek (Class Library) angelegt.
2. Hinzufügen der Referenz auf WSAPM: Dem Projekt ist anschließend eine Referenz auf *Wsapm.Extensions.dll* hinzuzufügen. Diese Datei ist im Installationsverzeichnis von WSAPM zu finden (für gewöhnlich *C:\Program Files (x86)\Windows Server Advanced Power Management*).
3. Ableitung von der Basisklasse *WsapmPluginBase*: Die Klasse, in der die Logik des Plugins implementiert werden soll, muss von der Basisklasse *WsapmPluginBase* abgeleitet werden.
4. Implementierten der abstrakten Basismethoden: Da die Klasse *WsapmPluginBase* eine abstrakte Basisklasse ist, müssen folgende Methoden in der abgeleiteten Klasse überschrieben werden:
   * protected override bool Initialize()  
     Dies ist die Initialisierungs-Methode für das Plugin. Diese wird mindestens einmal aufgerufen, wenn das Plugin von WSAPM geladen wurde. Sämtliche Funktionen, die nur ein einziges Mal ausgeführt werden müssen, sollten in dieser Methode implementiert werden.  
     Der Rückgabewert gibt Auskunft darüber, ob die Initialisierung des Plugins erfolgreich abgeschlossen wurde. Wird hier *false* zurückgeliefert, wurde das Plugin nicht erfolgreich initialisiert (z.B. im Fehlerfall) und wird von WSAPM nicht in die Überprüfungs-Routinen mit einbezogen. Wenn keine Initialisierung für das Plugin nötig ist, muss hier einfach *true* zurückgeliefert werden.
   * protected override bool Prepare()  
     Dies ist die Methode, die zum Vorbereiten der eigentlichen Überprüfungs-Routine ausgeführt wird. Diese Methode wird vor jedem Aufruf der Überprüfungs-Routine aufgerufen.  
     Der Rückgabewert gibt wiederum Auskunft darüber, ob die Vorbereitung erfolgreich abgeschlossen wurde. Wird hier *false* zurückgeliefert (z.B. im Fehlerfall), wird WSAPM die Überprüfungs-Routine des Plugins im Folgenden nicht ausführen. Wenn das Plugin keinerlei Vorbereitungen zur Ausführung der Überprüfungs-Routine benötigt, muss hier einfach *true* zurückgeliefert werden.
   * protected override PluginCheckSuspendResult CheckPluginPolicy()  
     Dies ist die zentrale Überprüfungs-Methode, die die Richtlinie(n) des Plugins überprüft. Das Ergebnis der Überprüfung wird als *PluginCheckSuspendResult* zurückgeliefert. Der Konstruktor dieser Klasse erwartet zwei Parameter: Der erste Parameter gibt Auskunft darüber, ob der Standby-Modus von WSAPM unterdrückt werden soll (*true*: Standby-Modus soll unterdrückt werden; *false*: Standby-Modus soll nicht unterdrückt werden). Der zweite Parameter ist ein String, welcher den Grund für eine mögliche Unterdrückung des Standby-Modus angibt. Dieser Parameter wird von WSAPM nur ausgewertet, wenn der Standby-Modus tatsächlich unterdrückt werden soll.
   * protected override bool TearDown()  
     Diese Methode wird unmittelbar nach Durchlauf der Überprüfungs-Routine aufgerufen und kann für notwendige Aufräumarbeiten nach der Überprüfung verwendet werden.   
     Der Rückgabewert gibt wieder an, ob das Aufräumen erfolgreich beendet wurde. Wenn das Plugin keine solcher Aufräumarbeiten ausführen muss, ist hier wieder einfach nur *true* zurückzugeben.
5. Hinzufügen von Attributen zur Plugin-Klasse: Zum Schluss müssen noch zwei Attibute zur Plugin-Klasse hinzugefügt werden, damit WSAPM die Klasse als Plugin erkennt.
   * [Export(typeof(WsapmPluginBase))]  
     Dieses Attribut muss immer in genau dieser Form über der Klassen-Definition stehen. Damit der Typ des Attributs gefunden werden kann, muss dem Projekt eine Referenz auf .NET Assembly *System.ComponentModel.Composition* hinzugefügt werden.
   * [WsapmPlugin("Internal name of plugin", "v1.0.0", "{YOUR-GUID-HERE}")]  
     Dieses Attribut liefert drei wichtige Informationen zum Plugin. Der erste Parameter ist der Name des Plugins. Dies ist nur ein interner (technischer) Name, der z.B. den Namen des Ordners festlegt, in dem das Plugin installiert wird. Der zweite Parameter stellt die Versionsnummer des Plugins dar. Der letzte Parameter gibt eine GUID an. Diese GUID identifiziert das Plugin eindeutig.  
     Wichtig:Diese GUID darf sich für ein Plugin niemals ändern. Auch wenn eine neue Version des Plugins erscheint, muss dieses die gleiche GUID aufweisen (aber eine andere Versions-Nummer), damit WSAPM die Verknüpfung zu früheren Versionen des Plugins erkennen kann.  
     Eine neue GUID kann in Visual Studio einfach über das *Create GUID* Tool erzeugt werden (*Tools* 🡪 *Create GUID*).
6. Das Plugin-Manifest: Jedes Plugin benötigt darüber hinaus auch eine Manifest-Datei. Dies ist eine XML-Datei, in der weitere Angaben zum Plugin gemacht werden können. Ein solche Datei hat einen genau definierten Aufbau:

<?xml version="1.0" encoding="utf-8" ?>  
<WsapmPlugin>  
 <DescriptionSet lang="en">  
 <PluginName>My plugin</PluginName>  
 <Description>My plugin’s description </Description>  
 <AuthorName>decatec.de</AuthorName>  
 </DescriptionSet>

<DescriptionSet lang="de">  
 <PluginName>Mein Plugin</PluginName>  
 <Description>Beschreibung meines Plugins </Description>  
 <AuthorName>decatec.de</AuthorName>  
 </DescriptionSet>   
</WsapmPlugin>

Dabei stellt ein sog. *DescriptionSet* die Beschreibung eines Plugins für eine bestimmte Sprache dar. Die Sprache wird dabei mit dem Attribut *lang* festgelegt. Dies ist ein zweistelliger Sprachcode nach [ISO 639-1](http://de.wikipedia.org/wiki/ISO_639) (z.B. „en“ für Englisch, oder „de“ für Deutsch).  
Der Name des Plugins, die Beschreibung und der Name des Autors werden in der Tabe Plugins in den Einstellungen angezeigt (siehe [*Einstellungen - Plugins*](#_Plugins)).

1. Weitere Informationen zum Plugin für den Benutzer: Es kann eine Datei *ReadMe.txt* mit weiteren Informationen zum Plugin hinzugefügt werden (z.B. eine genaue Beschreibung, was das Plugin tut). Diese Informationen werden angezeigt, wenn man auf die Schaltfläche *Plugin Info* in der Tabe *Plugins* in den Einstellungen klickt (siehe [*Einstellungen - Plugins*](#_Plugins)). Es können auch weitere Dateien für andere Sprachen hinzugefügt werden, hier sollte dann wieder ein zweistelliger Sprachcode an den Dateinamen angehängt werden (z.B. *ReadMe\_de.txt* für eine deutsche Beschreibung).  
   Diese Dateien sind optional. Wenn diese nicht Teil des Plugins sind, werden nur allgemeine Informationen zum Plugin angezeigt, wenn die Schaltfläche *Plugin Info* geklickt wird.
2. Damit ist die wesentliche Entwicklung eines einfachen Plugins für WSAPM abgeschlossen. Damit das Plugin weitergegeben und installiert werden kann, muss es noch in eine Zip-Datei gepackt werden. Dazu werden einfach alle zum Plugin gehörenden Dateien aus dem Project-Output-Verzeichnis gezippt.  
   Wichtig: Im Project-Output-Verzeichnis werden auch folgende Dateien zu finden sein, die per Referenz zum Projekt hinzugefügt wurden. Diese sollten nicht mit dem Plugin ausgeliefert werden und daher nicht der Plugin Zip-Datei hinzugefügt werden:
   * *Wsapm.Extensions.dll*
   * *de\Wsapm.Extensions.resources.dll*

### Entwicklung erweiterter Plugins

Es folgt eine Schritt für Schritt Anleitung zur Entwicklung erweiterter Plugins für WSAPM. Wie bereits erwähnt, sollte ein erweitertes Plugin nur dann entwickelt werden, wenn Plugin-spezifische Einstellung und die dafür passende Oberfläche benötigt wird. Ansonsten sollte man ein einfaches Plugin implementieren, da die Entwicklung hier schneller und einfacher von der Hand gehen sollte.  
Wenn die vorgefertigte Visual Studio Solution als Basis genommen wird (siehe [*Plugin-Entwicklung - Schnelleinstieg*](#_Schnelleinstieg)), sind darin bereits viele Schritte erledigt.

1. Neues Visual Studio Projekt erstellen: Da ein WSAPM Plugin aus mindestens einer DLL besteht, wird ein neues Visual Studio Projekt als Klassenbibliothek (Class Library) angelegt.
2. Hinzufügen der Referenz auf WSAPM: Dem Projekt ist anschließend eine Referenz auf *Wsapm.Extensions.dll* hinzuzufügen. Diese Datei ist im Installationsverzeichnis von WSAPM zu finden (für gewöhnlich *C:\Program Files (x86)\Windows Server Advanced Power Management*).
3. Implementieren der Settings-Klasse: Die Einstellungen des Plugin werden über eine eigene Settings-Klasse implementiert. Diese muss lediglich serialisierbar sein, also mit dem Attribut *Serializable* dekoriert und als *public* deklariert werden.
4. Implementieren der Oberfläche des Plugins: Die Oberfläche des Plugins kann sowohl mit WPF ([Windows Presentation Foundation](http://de.wikipedia.org/wiki/Windows_Presentation_Foundation)), als auch mit [Windows Forms](http://de.wikipedia.org/wiki/Windows_Forms) implementiert werden. Dazu wird einfach an ein neues User Control zum Projekt hinzugefügt.  
   Die Oberfläche sollte nur die Controls beinhalten, die die Einstellungen des Plugins repräsentieren, d.h. keine Buttons wie *OK* oder *Abbrechen* um die Kontrolle wieder an WSAPM zu übergeben.  
   Die Oberflächen-Klasse muss das Interface *IWsapmPluginSettingsControl* implementieren, zu dem zwei Methoden gehören:
   * public void SetSettings(object settings)  
     Mit dieser Methode werden die aktuellen Einstellungen in das Control geladen. Hier sollten die einzelnen Oberflächen-Elemente (TextBoxen, CheckBoxen, etc.) mit den Werten aus den Settings gefüllt werden. Vorher ist noch ein Cast auf den Typen der Settings-Klasse notwendig.
   * public object GetSettingsBeforeSave()  
     Diese Methode wird aufgerufen, wenn die Plugin-Einstellungen gespeichert werden sollen (von WSAPM, nicht vom Plugin selbst). Hier sollte eine Instanz der Settings-Klasse zurückgegeben werden, die die aktuell vorliegenden Einstellungen repräsentiert. Diese Einstellungen werden anschließend von WSAPM gespeichert.
5. Implementieren der eigentlichen Plugin-Klasse: Nun wird noch eine weitere Klasse benötigt, die das eigentliche Plugin repräsentiert. Alle folgenden Tätigkeiten beziehen sich auf diese Plugin-Klasse.
6. Ableitung von der Basisklasse *WsapmPluginAdvancedBase*: Die Klasse, in der die Logik des Plugins implementiert werden soll, muss von der Basisklasse *WsapmPluginAdvancedBase* abgeleitet werden.
7. Hinzufügen eines *public* Konstruktors: Die Plugin-Klasse benötigt einen Konstruktor, der den Konstruktor der Basisklasse aufruft und diesem den Typen der Settings-Klasse übergibt.
8. Implementierten der abstrakten Basismethoden: Da die Klasse *WsapmPluginAdvancedBase* eine abstrakte Basisklasse ist, müssen folgende Methoden in der abgeleiteten Klasse überschrieben werden:
   * protected override bool Initialize()  
     Dies ist die Initialisierungs-Methode für das Plugin. Diese wird mindestens einmal aufgerufen, wenn das Plugin von WSAPM geladen wurde. Sämtliche Funktionen, die nur ein einziges Mal ausgeführt werden müssen, sollten in dieser Methode implementiert werden.  
     Der Rückgabewert gibt Auskunft darüber, ob die Initialisierung des Plugins erfolgreich abgeschlossen wurde. Wird hier *false* zurückgeliefert, wurde das Plugin nicht erfolgreich initialisiert (z.B. im Fehlerfall) und wird von WSAPM nicht in die Überprüfungs-Routinen mit einbezogen. Wenn keine Initialisierung für das Plugin nötig ist, muss hier einfach *true* zurückgeliefert werden.
   * protected override bool Prepare()  
     Dies ist die Methode, die zum Vorbereiten der eigentlichen Überprüfungs-Routine ausgeführt wird. Diese Methode wird vor jedem Aufruf der Überprüfungs-Routine aufgerufen.  
     Der Rückgabewert gibt wiederum Auskunft darüber, ob die Vorbereitung erfolgreich abgeschlossen wurde. Wird hier *false* zurückgeliefert (z.B. im Fehlerfall), wird WSAPM die Überprüfungs-Routine des Plugins im Folgenden nicht ausführen. Wenn das Plugin keinerlei Vorbereitungen zur Ausführung der Überprüfungs-Routine benötigt, muss hier einfach *true* zurückgeliefert werden.
   * protected override PluginCheckSuspendResult CheckPluginPolicy()  
     Dies ist die zentrale Überprüfungs-Methode, die die Richtlinie(n) des Plugins überprüft. Das Ergbnis der Überprüfung wird als *PluginCheckSuspendResult* zurückgeliefert. Der Konstruktor dieser Klasse erwartet zwei Parameter: Der erste Parameter gibt Auskunft darüber, ob der Standby-Modus unterdrückt werden soll (*true*: Standby-Modus soll unterdrückt werden; *false*: Standby-Modus soll nicht unterdrückt werden). Der zweite Parameter ist ein String, welcher den Grund für eine mögliche Unterdrückung des Standby-Modus angibt. Dieser Parameter wird von WSAPM nur ausgewertet, wenn der Standby-Modus tatsächlich unterdrückt werden soll.
   * protected override bool TearDown()  
     Diese Methode wird unmittelbar nach Durchlauf der Überprüfungs-Routine aufgerufen und kann für notwenige Aufräumarbeiten nach der Überprüfung verwendet werden.   
     Der Rückgabewert gibt wieder an, ob das Aufräumen erfolgreich beendet wurde. Wenn das Plugin keine solcher Aufräumarbeiten ausführen muss, ist hier wieder einfach nur *true* zurückzugeben.
   * protected override object LoadDefaultSettings()  
     Diese Methode sollte die Standard-Einstellungen des Plugins liefern. Dazu muss eine Instanz der Settings-Klasse (siehe Punkt 3 *Implementieren der Settings-Klasse*) erzeugt und mit Standard-Daten befüllt werden.  
     Diese Standrad-Einstellungen werden immer dann verwendet, wenn noch keine Settings für das Plugin vorhanden sind, z.B., wenn das Plugin das erste Mal gestartet wird.
   * public override object SettingsControl  
     Hier ist im Getter eine Instanz der Oberflächen-Klasse zurückzuliefern. Wenn diese Instanz einmal erzeugt ist, sollte diese in einer privaten Variablen gespeichert werden und immer nur diese eine Instanz zurückgeliefert werden.
9. Hinzufügen von Attributen zur Plugin-Klasse: Zum Schluss müssen noch zwei Attibute zur Plugin-Klasse hinzugefügt werden, damit WSAPM die Klasse als Plugin erkennt.
   * [Export(typeof(WsapmPluginBase))]  
     Dieses Attribut muss immer in genau dieser Form über der Klassen-Definition stehen. Damit der Typ des Attributs gefunden werden kann, muss dem Projekt eine Referenz auf .NET Assembly *System.ComponentModel.Composition* hinzugefügt werden.  
     Der Typ ist in diesem Fall immer *WsapmPluginBase*, auch wenn es sich hierbei um ein erweitertes Plugin handelt, welches sich von der Basisklasse *WsapmPluginAdvancedBase* ableitet.
   * [WsapmPlugin("Internal name of plugin", "v1.0.0", "{YOUR-GUID-HERE}")]  
     Dieses Attribut liefert drei wichtige Informationen zum Plugin. Der erste Parameter ist der Name des Plugins. Dies ist nur ein interner (technischer) Name, der z.B. den Namen des Ordners festlegt, in dem das Plugin installiert wird. Der zweite Parameter stellt die Versionsnummer des Plugins dar. Der letzte Parameter gibt eine GUID an. Diese GUID identifiziert das Plugin eindeutig.  
     Wichtig:Diese GUID darf sich für ein Plugin niemals ändern. Auch wenn eine neue Version des Plugins erscheint, muss dieses die gleiche GUID aufweisen (aber eine andere Versions-Nummer), damit WSAPM die Verknüpfung zu früheren Versionen des Plugins erkennen kann.  
     Eine neue GUID kann in Visual Studio einfach über das *Create GUID* Tool erzeugt werden (*Tools* 🡪 *Create GUID*).
10. Zugriff auf Einstellungen innerhalb des Plugins: Zugriff auf die aktuellen Einstellungen erfolgen über den Aufruf von *CurrentSettings* der Basisklasse. Hier muss noch ein Cast auf den Typen der eigenen Settings-Klasse vorgenommen werden.
11. Zugriff auf die Oberfläche innerhalb des Plugins: Im Plugin darf nur ein einer Stelle auf Oberflächen-Klassen stattfinden. Diese Stelle ist der Getter von *SettingsControl*, welche von der Basisklasse überschrieben werden muss. Ansonsten darf an keiner anderen Stelle auf Oberflächen-Klassen zugegriffen werden!
12. Speichern und Laden von Einstellungen: Das Speichern und Laden der Plugin-Einstellungen übernimmt WSAPM und liegt nicht in der Kontrolle des Plugins. Daher ist im Plugin keinerlei Logik zum Laden/Speichern von Einstellungen zu implementieren.   
    Die Einstellungen sind im folgendem Verzeichnis zu finden: *C:\ProgramData\Windows Server Advanced Power Management\Plugins\<Name des Plugins> (<GUID des Plugins>)\PluginSettings.xml*
13. Das Plugin-Manifest: Jedes Plugin benötigt darüber hinaus auch eine Manifest-Datei. Dies ist eine XML-Datei, in der weitere Angaben zum Plugin gemacht werden können. Ein solche Datei hat einen genau definierten Aufbau:

<?xml version="1.0" encoding="utf-8" ?>  
<WsapmPlugin>  
 <DescriptionSet lang="en">  
 <PluginName>My plugin</PluginName>  
 <Description>My plugin’s description </Description>  
 <AuthorName>decatec.de</AuthorName>  
 </DescriptionSet>

<DescriptionSet lang="de">  
 <PluginName>Mein Plugin</PluginName>  
 <Description>Beschreibung meines Plugins </Description>  
 <AuthorName>decatec.de</AuthorName>  
 </DescriptionSet>   
</WsapmPlugin>

Dabei stellt ein sog. *DescriptionSet* die Beschreibung eines Plugins für eine bestimmte Sprache dar. Die Sprache wird dabei mit dem Attribut *lang* festgelegt. Dies ist ein zweistelliger Sprachcode nach [ISO 639-1](http://de.wikipedia.org/wiki/ISO_639) (z.B. „en“ für Englisch, oder „de“ für Deutsch).  
Der Name des Plugins, die Beschreibung und der Name des Autors werden in der Tabe Plugins in den Einstellungen angezeigt (siehe [*Einstellungen - Plugins*](#_Plugins)).

1. Damit ist die wesentliche Entwicklung eines erweiterten Plugins für WSAPM abgeschlossen. Damit das Plugin weitergegeben und installiert werden kann, muss es noch in eine Zip-Datei gepackt werden. Dazu werden einfach alle zum Plugin gehörenden Dateien aus dem Project-Output-Verzeichnis gezippt.  
   Wichtig: Im Project-Output-Verzeichnis werden auch folgende Dateien zu finden sein, die per Referenz zum Projekt hinzugefügt wurden. Diese sollten nicht mit dem Plugin ausgeliefert werden und daher nicht der Plugin Zip-Datei hinzugefügt werden:
   * *Wsapm.Extensions.dll*
   * *de\Wsapm.Extensions.resources.dll*

### Debuggen von Plugins

Da Plugins keine ausführbaren Dateien sind und von WSAPM ausgeführt und kontrolliert werden, kann man diese nicht einfach per F5 (Visual Studio) debuggen. Um trotzdem ein Debuggen zu ermöglichen, kann man an entsprechenden Stellen im Plugin-Code folgende Code-Zeilen hinzufügen:

if (!System.Diagnostics.Debugger.IsAttached)

System.Diagnostics.Debugger.Launch();

System.Diagnostics.Debugger.Break();

Darüber hinaus muss das Plugin im Debug-Modus kompiliert und die PDB-Dateien in das Plugin-Verzeichnis (üblicherweise *C:\Program Files (x86)\Windows Server Advanced Power Management\Plugins\<Name des Plugins>*) kopiert werden. Nun kann der Debugger im Visual Studio attached werden (*Debug* 🡪 *Attach to Process…*). In der darauf erscheinenden Liste muss dann einer der folgenden Prozesse ausgewählt werden:

* *Wsapm.exe*: wenn Teile der Oberfläche des Plugins gedebuggt werden sollen
* *Wsapm.Service.exe*: wenn die eigentliche Plugin-Logik gedebuggt werden soll

Wenn das Plugin geladen/ausgeführt oder die Oberfläche des Plugins geladen wird, sollte nun ein Fenster erscheinen, welches das Visual Studio zum Debuggen startet und anschließend sollte der Debugger an der markieren Stelle stehen.

### Bereitstellen von Plugins für andere Benutzer

Wenn Sie ein eigenes Plugin für WSAPM entwickelt haben und denken, dass dieses Plugin auch für weitere Nutzer von WSAPM nützlich sein könnte, schreiben Sie mit bitte eine [E-Mail](mailto:jr@decatec.de). Dann kann ich das Plugin mit Ihrem Einverständnis auf der Website von WSAPM ([decatec.de](https://decatec.de/)) veröffentlichen.

## Remote-Shutdown

Windows Server Advanced Power Management bietet Nutzern die Möglichkeit, den Computer aus der Ferne herunter zu fahren, neu zu starten oder in den Standby- oder Ruhezustand zu versetzen.  
Client-seitig ist dazu eine Anwendung oder App notwendig, die die entsprechenden Befehle senden kann (z.B. [MagicPacket](https://decatec.de/software/magicpacket/)). Allerdings ist das Protokoll für Remote-Shutdown offengelegt, so dass jeder Entwickler diese Funktionen in eigenen Programme oder Apps implementieren kann. Der folgende Abschnitt beschreibt dieses Protokoll für Remote-Shutdown.

Wenn die entsprechende Option in Windows Server Advanced Power Management aktiviert wurde, lauscht das Programm auf dem spezifizierten Port auf [UDP-Pakete](https://de.wikipedia.org/wiki/User_Datagram_Protocol). Die Pakete, die für Remote-Shutdown genutzt werden, ähneln den UDP-Paketen, die bei [Wake On LAN](https://de.wikipedia.org/wiki/Wake_On_LAN) verwendet werden.

**Wake On LAN**

Wake On LAN Befehle sind einfache UDP-Pakete (sog. Magic Packets), die folgendermaßen aufgebaut sind:

* Sechsmal der hexadezimale Wert FF
* Eine 16-fache Wiederholung der MAC-Adresse des zu weckenden PCs

Diese Befehle werden dann an die [Broadcast-Adresse](https://de.wikipedia.org/wiki/Broadcast) des Netzwerks gesendet (255.255.255.255).

**Befehle für Remote-Shutdown**

Hierbei handelt es sich im Grunde genommen um ein Magic Packet, das jedoch nicht mit sechsmal dem Wert FF beginnt, sondern mit den folgenden Werten:

* Standby: AA
* Ruhezustand: BB
* Neustart: CC
* Herunterfahren: DD

Ebenso kann optional ein Passwort übermittelt werden, welches am Ende des UDP-Pakets angehängt wird. Wenn in den Optionen von WSAPM ein Passwort für Remote-Shutdown gewählt wurde, wird ein Remote-Shutdown-Paket nur verarbeitet, wenn das übertragene Passwort mit dem im Programm hinterlegten übereinstimmt.

**Wichtig:** Ein Passwort für Remote-Shutdown wird unverschlüsselt über das Netzwerk übertragen, daher stellt dies keinen Sicherheits-Aspekt dar.

*Beispiel:* Der Zielrechner hat die MAC-Adresse 23:D0:43:F1:2C:0A und soll aus der Ferne herunter gefahren werden. Dazu muss folgendes UDP-Paket aufgebaut werden und an die Broadcast-Adresse (255.255.255.255) versendet werden:

* Sechsmal der hexadezimale Wert DD (für Herunterfahren)
* Eine 16-fache Wiederholung der MAC-Adresse (in diesem Fall also die 23:D0:43:F1:2C:0A)

Eine Methode für die Implementierung in C# könnte folgendermaßen aussehen:

private static byte[] GetShutdownPacket(byte[] macAddress, string password)

{

byte[] passwordBytes = System.Text.Encoding.UTF8.GetBytes(password);

byte[] packet = new byte[102 + passwordBytes.Length];

// 6x AA -> standby

// 6x BB -> hibernate

// 6x CC -> restart

// 6x DD -> shut down

byte headerByte = 0xDD;

// 6x DD (for shut down)

for (int i = 0; i < 6; i++)

{

packet[i] = headerByte;

}

// 16x the MAC address of the target machine

for (int i = 1; i < 17; i++)

{

macAddress.CopyTo(packet, i \* 6);

}

// The password is attached at the end of the packet

for (int i = 0; i < passwordBytes.Length; i++)

{

packet[102 + i] = passwordBytes[i];

}

return packet;

}

Ich würde mich freuen, wenn die Remote-Shutdown-Funktionalität für Windows Server Advanced Power Management auch in anderen Programmen oder Apps implementiert werden würde. Gerne biete ich hierfür auch Unterstützung an (an einfachsten per [E-Mail](mailto:jr@decatec.de)).