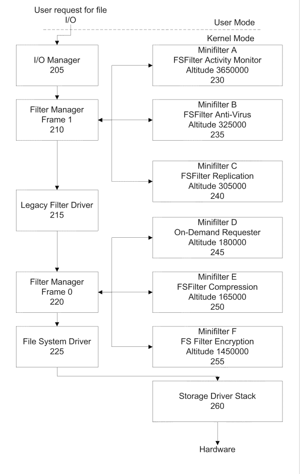
US8380808

Assignee:	O'BRIEN ROYAL
Title:	Dynamic medium content streaming system
Filing Date:	2008-11-24
Publication Date:	2013-02-19
Inventor:	O'Brien, Royal
Earliest Priority:	2008-11-24 US 12/276998
Fee Status:	2016-05-27 Payment of Maintenance Fee, 4th Yr, Small Entity.
Legal Status (PAIR):	Patented Case


User request for file I/O
User Mode
Kernel Mode
I/O Manager 205
Filter Manager Frame 1 210
Legacy Filter Driver 215
Filter Manager Frame 0 220
File System Driver 225
Minifilter A FSFilter Activity Monitor Altitude 3650000 230
Minifilter B FSFilter Anti-Virus Altitude 325000 235
Minifilter C FSFilter Replication Altitude 305000 240
Minifilter D On-Demand Requester Altitude 180000 245
Minifilter E FSFilter Compression Altitude 165000 250
Minifilter F FS Filter Encryption Altitude 1450000 255
Storage Driver Stack 260
Hardware

Claims:

1	A streaming on demand system comprising an on-demand requester object installed on a computing device, said on-demand requester object being configured to receive I/O requests on behalf of an application for which data is available in data packs for streaming delivery, said on-demand requester object being responsive to application I/O requests, said	The limitations of Claim 1 are satisfied by an application program running on a computer system using Windows. Windows satisfies application I/O requests utilizing an I/O stack, which includes at least a filter manager and one or more software components (drivers) that work in combination to produce the necessary response to the application’s I/O request. The term “Windows” means at least the current and more recent versions of Windows 10 running on a computer.

	application I/O requests corresponding to sequential and non-sequential data packs, said data packs being portions of a complete data stream, said I/O requests determining an order in which data packs are sought for the application, said on-demand requester object comprising a minifilter associated with a filter manager in an I/O stack, said filter manager being configured to call the minifilter, said minifilter having a pre-operation callback routine for the I/O request registered with the filter manager, and said minifilter being configured to receive each I/O request from the application, said minifilter being further configured to reference a table that includes names and paths with at least one offset address and length where each data pack required to fulfill each I/O request is located, said minifilter being further configured to determine if the data pack has been streamed to the system, and said application being a program configured to use the data packs that fulfill each I/O request when the data packs are available on the computing device.	The term "OneDrive" is used to describe a data provider element, which is part of the on-demand requester object defined by the claim, that consists of components that are both resident on the local computer system (e.g., the "OneDrive for Windows" service, which is part of Windows) and the supporting infrastructure, including software and hardware, provided and/or managed by Microsoft to at least manage, read, write, replicate, and enable transfer of data from a remote storage location to the local Windows system, as needed by Windows to satisfy the I/O requests of the application that generated the I/O request. This use is consistent with Microsoft's own use of the name **OneDrive**. Of these required elements, the following are present when running an application on Windows: • The application program that issues I/O requests within the accused environment, though the application itself is normally not part of the on-demand requester object. Any application program can trigger the behavior described within the patent because executing an application inherently requires generating I/O request, simply to execute the application. Given that the function of a modern computer system is to run application programs, this element is always present. The application's generation of I/O requests is not part of the claim but is a necessary pre-requisite to their receipt and processing of the on-demand requester object. • A filter manager. All versions of Microsoft Windows include a component known as the "Filter Manager". Microsoft documents this: "The filter manager is installed with Windows, but becomes active only when a minifilter driver is loaded. The filter manager attaches to the file system stack for a target volume. A minifilter driver attaches to the file system stack indirectly, by registering with the filter manager for the I/O operations the minifilter driver chooses to filter." (https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts). • An I/O stack. Windows has included an I/O stack in all versions of Windows since at least Windows XP. Microsoft documents this in numerous places and uses a variety of terms to describe how this I/O stack is implemented, such as describing the layers of the I/O stack

		(https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/overview-of-the-windows-i-o-model). Thus, this claim element is inherent in Windows and is always present. A nonlimiting example of how Microsoft describes the action of the accused product is presented in Windows. Windows includes **OneDrive**. Microsoft documents that this is included in Windows: “If you have Windows 10, OneDrive is already installed on your PC. If you are using another version of Windows, install OneDrive to get started.” (https://www.microsoft.com/en-us/microsoft-365/onedrive/download) **OneDrive**, which is part of Windows, implements a streaming on-demand data storage service, in which data is stored in data packs within a Microsoft data center and streaming downloaded, on demand, to the local computer system, said demand being initiated by the I/O requests of the application and if the data has not previously been downloaded and stored on the local computer system running Windows. The on-demand requester object of the claim is realized when **OneDrive** is combined, directly or indirectly, with the **cldflt.sys** minifilter driver, which is distributed as part of Windows. This on-demand requester object satisfies the application I/O request, either by using data that has been stored locally, or data that it retrieves over the network from Microsoft’s data center. Since its introduction to Windows, Microsoft has formalized the use of **cldflt.sys** by implementing and documenting a library that can be used by specialized applications to allow both Microsoft and third parties to develop “Cloud Sync Engines”. Microsoft documents these on their website (https://docs.microsoft.com/en-us/windows/win32/cfapi/cloud-files-api-portal). Microsoft explains in detail how Windows incorporates support for streaming on demand file download. The **cldflt.sys** minifilter is sufficient to activate the function of the Windows Filter Manager.

		The version of **OneDrive** that is included in Windows is now an implementation of a "Cloud Sync Engine" within Windows (https://docs.microsoft.com/en-us/windows/win32/cfapi/cloud-files-api-portal). The combination of **OneDrive** and **cldflt.sys** correspond to the "on-demand requester object" defined by this claim. To satisfy this definition, the on-demand requester object must include a minifilter and must have a mechanism for retrieving the necessary data on-demand, which need not be part of the minifilter. The storage data provider service, which is an inherent part of the on-demand requester object, downloads data as needed and may store a copy of that data on the local computer. **OneDrive** works with the **cldflt.sys** minifilter, the Windows I/O stack, filter manager, and other necessary components to realize an on-demand requester object as defined by the claim. All versions of Microsoft Windows 10 since at least the release of Version 1709 include the required elements: • An on-demand requester object installed on the computing device configured to receive I/O requests on behalf of that application and either satisfy the I/O request if the data is located on the local computer system, or fetched from a remote location. Said on-demand requester object consisting of: o **OneDrive** which is a cloud storage solution provided by Microsoft as part of current Windows versions. **OneDrive** permits seamless integration of streaming on-demand cloud storage with Windows (https://www.windowscentral.com/how-get-started-onedrive-windows-10). o A dynamic link library, **cldapi.dll** that: (1) Is included with Windows and provides a programmatic mechanism for **OneDrive** to communicate with the **cldflt.sys** minifilter, described below, that,

		when combined, are consistent with the "on-demand requester object" of the patent. (2) Provides at least one mechanism for transferring data from a remote storage service to the local file system service. For example, **CfGetTransferKey** is a documented mechanism that is made available from **cldapi.dll** that Microsoft describes as "… an alternative to **CfHydratePlaceholder** to proactively initiate data transfer into a placeholder." https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfgettransferkey (3) Provides at least one mechanism for determining information about the status of regions of a specific locally maintained file. The application programming interface (API) **CfGetPlaceholderRangeInfo**, which is made available from **cldapi.dll**, specifies a file handle, which is sufficient to determine the identity of a file, including a path and name, the status of the requested range (the **CF_PLACEHOLDER_RANGE_INFO_CLASS** parameter), the **StartingOffset** and **Length** parameters, as well as a location in which the requested information is returned (**InfoBuffer, InfoBufferLength,** and **ReturnedLength**). https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfgetplaceholderrangeinfo. The **cldapi.dll** and **cldflt.sys** minifilter work in tandem to maintain this information and ensure it is updated properly within the minifilter. This is accomplished by using a communications protocol. From this, it appears likely that the minifilter maintains at least one table of state information. o A minifilter, **cldflt.sys** that: (1) Is associated with the Microsoft Windows filter manager (https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts). According to Microsoft the filter manager is included in all versions

		of Windows, as it is universal (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltregisterfilter); (2) The filter manager is configured to call the minifilter. The **cldflt.sys** minifilter does this by invoking **FltRegisterFilter** and specifying entry points that filter manager is to invoke to handle specific input and output (I/O) operations, at least including opening, reading, and writing the contents of files, such files residing on the local computer system and/or on the cloud storage system; (3) The **cldflt.sys** minifilter driver uses existing functionality of the Windows operating system core operating system to maintain tables of information, including **RtlInsertElementGenericTableAvl** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-rtlinsertelementgenerictableavl)**, FsRtlGetNextMbaseMcbEntry** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntifs/nf-ntifs-_fsrtl_advanced_fcb_header-fsrtlgetnextmcbentry), **FltParseFileNameInformation (**https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltparsefilename), **FltSetStreamContext** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltsetstreamcontext), and **FltSetFileContext (**https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltsetfilecontext). These routines are consistent with both the use of tables and tracking regions of files that have been allocated. This, combined with the detailed range information provided by **cldflt.dll,** at a minimum corresponds to the table that includes this information and which data packs have been streamed to the local system. (4) The **cldflt.sys** minifilter driver ensures that the data required to satisfy an application I/O request is present on the local system before that I/O operation is completed.

		(5) The **cldflt.sys** minifilter driver is a necessary component of the on-demand requester object. When **cldflt.sys** is bypassed, we do not observe the on-demand behavior of normal Windows applications when using **OneDrive**. This is apparent if one uses Windows Subsystem for Linux (WSL), which bypasses the **cldflt.sys** driver functionality. Thus, Linux applications within the WSL environment do not see the correct data content of the file. Thus, a computer system with Windows installed includes all of the necessary elements to meet the requirements of this claim: at least one application that generates I/O requests, an I/O stack, an I/O Manager, a filter manager, and an on-demand requester object in which the on-demand requester object includes a minifilter that registers and handles specific I/O operations. This satisfies the limitations of this claim.
3	A streaming on demand system according to claim 1, said minifilter attaching to the I/O stack by registering with a filter manager in the I/O stack for I/O requests for the application for which data is available in data packs for streaming delivery.	The limitations of Claim 1 are incorporated by reference. In addition, the limitations of Claim 3 are satisfied by the **cldflt.sys** minifilter component of the "on-demand requester object" by registering with the Windows Filter manager in the I/O stack. This is satisfied by the fact that the **cldflt.sys** minifilter calls the Windows kernel routine **FltRegisterFilter** and specifies a list of callbacks for I/O requests for any application accessing the specific storage device. The **cldflt.sys** minifilter is constructed in such a way that if the data is not provided by the "on-demand requester object" the I/O requests for the application are passed along the I/O stack. Thus, **cldflt.sys** does intercept operations for I/O requests for application for which the data is available in data packs for streaming delivery. This satisfies the limitations of this claim.

4	A streaming on demand system according to claim 1, said table including a size indicator for each data pack.	The limitations of Claim 1 are incorporated by reference. In addition, the limitations of Claim 4 are satisfied by the **cldflt.sys** minifilter because it maintains detailed size information for each data pack, which may include one or more portions of a single file. The **cldflt.sys**has a function called **HsmiRecallAllocateHydrationContext**, which is part of the code within **cldflt.sys**, which is used to construct the on-demand requester object. In addition, that this size information is maintained both within the operating memory of **cldflt.sys** as well as the "placeholder" data that it uses to maintain information. Further, this information is maintained within a table, which is demonstrated by the use of the Adelson-Velsky and Landis tree ("AVL Tree") package, which is a part of the Windows operating system kernel; for example, the function **RtlInsertElementGenericTableAVL**, which is documented by Microsoft (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-rtlinsertelementgenerictableavl). These functions are used when a cloud file is opened (**HsmiFltpECPCreate**) and when responding to specific requests by Windows OneDrive, which is a data provider that is part of the on-demand requester object (**CldiPortProcessServiceCommands**). Microsoft also documents that the cloud filter API allows the data provider component of the on-demand requester object to retrieve size information (**CF_FS_METADATA**) using the API **CfHydratePlaceholder**, which includes a handle to identify the specific file (**FileHandle**), the starting offset of the file region (**StartingOffset**), and the length of that region (**Length**). In order to implement this API, the on-demand requester object must maintain a table of regions within the file that are present. An additional example of how the **cldflt.sys** minifilter component of the on-demand requester object in Windows meets this claim limitation is that it uses the "map control block" (MCB) API that is used by Windows file systems for tracking the allocation of media blocks to a file; this is a second implementation of a table and media control blocks are an example of "data packs" within the meaning of the specification. Microsoft documents the MCB API. For example, the routine **FsRtlLookupBaseMcbEntry** is documented on the Microsoft website

		(https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntifs/nf-ntifs-_fsrtl_advanced_fcb_header-fsrtllookupbasemcbentry). This satisfies the limitations of this claim.
5	A streaming on demand system according to claim 1, said minifilter being configured to pass on a read call if a required data pack is available locally and if all data in the required data pack has been received.	The limitations of Claim 1 are incorporated by reference. In addition, the limitations of claim 5 are satisfied by the **cldflt.sys** minifilter, which is part of the on-demand requester object. The **cldflt.sys** minifilter passes read operations to the next component of the I/O stack for a given file if the data is available within the local file. The **cldflt.sys** minifilter achieves this in at least two ways: • The minifilter modifies the file meta-data so that it can be satisfied locally by either not attaching, or by removing the in-memory minifilter stream context (**HsmpCldNullifyStreamContext**) • When a file is completely downloaded and stored locally, the minifilter can remove the reparse point (**HsmpRpRemove**) that is stored on the underlying media where the file data is persistently stored. Microsoft documents how reparse points work in Windows (https://docs.microsoft.com/en-us/windows/win32/fileio/reparse-points) and support for them is a standard feature of the Windows operating system. This satisfies the limitations of this claim.
6	A streaming on demand system according to claim 1, said minifilter being configured to withhold a read call if a required data pack is not available locally.	The limitations of Claim 1 are incorporated by reference. In addition, the limitations of claim 6 are satisfied by the **cldflt.sys** minifilter component of the "on-demand requester object". It does so by blocking ("withhold") a read call if data required to satisfy the application read I/O is not present in the locally downloaded copy of the file.

		The **cldflt.sys** minifilter meets this claim limitation because **cldflt.sys** blocks the read call while the **OneDrive** component of the on-demand requester object downloads the necessary data. The function **HsmiRecallInitiateHydrationEx** is invoked by **HsmFltPreREAD**. The function **HsmFltPreREAD** is the read call handler that **cldflt.sys** registers with the Filter Manager and is invoked to handle read calls from the application that have been sent to the I/O stack, and have been routed to the Filter Manager, which then invokes the **cldflt.sys** minifilter. Once **OneDrive** has fetched the data, the read call is permitted to continue, and the original application read is satisfied. This satisfies the limitations of this claim.
7	A streaming on demand system according to claim 1, said minifilter being configured to withhold a read call if all data in the required data pack has not been received.	The limitations of Claim 1 are incorporated by reference. In addition, to the limitations of Claim 7, the minifilter must block ("withhold") a read call if the data pack containing the data required to satisfy the read is not present in the locally downloaded copy of the file. The **cldflt.sys** minifilter satisfies this claim limitation because **cldflt.sys** withholds the read call while the **OneDrive** component of the on-demand requester object downloads the data from its alternative (remote) location. The function **HsmpRecallInitiateHydrationEx** in **cldflt.sys** demonstrates this behavior. The minifilter inspects the table that it maintains of data which has been written locally and if it does not find the required data, then notifies **OneDrive** that an application requires the data from the given file. **OneDrive** then retrieves the specified data and blocks the original read request until at least the entire data region required by the application has been downloaded to the local copy of the file. Once the data is present, the **cldflt.sys** minifilter permits the read request to continue being processed, which ultimately provides the application with the data that it read. This can also be observed by an ordinary user of Windows by accessing a file that is not stored locally when using Microsoft **OneDrive** in Windows. The application blocks until the contents of at

		least some portion of the file have been retrieved and the delay involved is generally noticeable. The second access to that file once it has been stored locally is much faster because the read request can be satisfied without withholding the read call. This satisfies the limitations of this claim.
8	A streaming on demand system according to claim 1, said at least one address including and address for at least one local source for a first data pack and an address for at least one remote source for a second data pack.	The limitations of Claim 1 are incorporated by reference. We suggest the phrase "including and address" should be read as meaning "including an address". The limitations of Claim 8 are satisfied by **OneDrive** and **cldflt.sys** because it maintains an address for at least one local source for a first data pack and an address , the on-demand requester object must utilize an address for at least one locally stored data pack and one remotely stored data pack. The on-demand requester object including **OneDrive** and **cldflt.sys** satisfies the limitations of this claim: • The **OneDrive** service combined with **cldflt.sys** permits the data of a file to be downloaded to local storage, which requires having address information for at least one remote source data pack; • The **cldflt.sys** file uses the "placeholder file" to identify the location of the locally downloaded data, the unique combination of device, file identifier, path name, and file name constitute the address of the local source. • The **OneDrive** service combined with **cldflt.sys** permits the locally stored copy of the remote data to be removed, leaving a "placeholder file" behind. This requires that the address of the remote storage location be maintained by these services. Microsoft alludes to this in their "Cloud Sync Engine" documentation where they describe placeholder files

		and say "The file has been hydrated implicitly and could be dehydrated by the system if space is needed." • The "Cloud Sync Engine" documentation provides additional detail about how this is supported by **cldflt.sys** via **cldapi.dll**. Note that by itself, these components do not provide this functionality and thus require **OneDrive** to implement this behavior. See the **CfHydratePlaceholder** function (https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfhydrateplaceholder). This function ensures that specific data is stored locally. In this situation, one unit of data (the "first data pack") is local and the second unit of data (the "second data pack") is not local, which corresponds to the ordinary meaning of the term "remote". This satisfies the limitations of this claim.
9	A streaming on demand system according to claim 1, said at least one address including and address for at least one local source for a first data pack, said local source being a source from the group consisting of a hard disk, an optical storage medium, a nonvolatile memory, a magnetic storage medium, and an address for at least one remote source for a second data pack.	The analysis provided for Claim 1 is incorporated by reference. In addition, the limitations of Claim 9 are satisfied by the on-demand requester object that includes **Onedrive** and Windows, which utilizes at least one address including an address for at least one local source for a first data pack. Further, they store the local data packs on at least one of a hard disk, an optical storage medium, a nonvolatile memory, and a storage medium. In addition, the on-demand requester object maintains at least one remote source for a second data pack. The on-demand requester object including **OneDrive** and **cldflt.sys** satisfies the limitations of this claim: • The **OneDrive** service combined with **cldflt.sys** permits the data of a file to be downloaded to local storage, which requires having address information for at least one remote source data pack;

		- The **cldflt.sys** file uses the “placeholder file” to identify the location of the locally downloaded data, the unique combination of device, file identifier, path name, and file name constitute the address of the local source. - The **OneDrive** service combined with **cldflt.sys** permits the locally stored copy of the remote data to be removed, leaving a “placeholder file” behind. This requires that the address of the remote storage location be maintained by these services. Microsoft alludes to this in their “Cloud Sync Engine” documentation where they describe placeholder files and say “The file has been hydrated implicitly and could be dehydrated by the system if space is needed.” The “Cloud Sync Engine” documentation provides additional detail about how this is supported by **cldflt.sys** via **cldapi.dll**. Note that by itself, these components do not provide this functionality and thus require **OneDrive** to implement this behavior. See the **CfHydratePlaceholder** function (https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfhydrateplaceholder). This function ensures that specific data is stored locally. In this situation, one unit of data (the “first data pack”) is local and the second unit of data (the “second data pack”) is not local, which corresponds to the ordinary meaning of the term “remote”. Modern computers with Windows installed typically utilize either solid state (“solid state drive” or “SSD”) or rotating magnetic media devices (“hard disk”), or variations of these devices (“hybrid” drives). The NTFS file system, which is required by Microsoft to support **OneDrive**, supports any writeable form of media, including re-writeable optical media (“read/write bluray”, “read/write DVD”, and “read/write CD”), magnetic media (including hard disks), and **Intel® Optane® DC Persistent Memory** (https://www.intel.com/content/www/us/en/architecture-and-technology/optane-dc-persistent-memory.html), which is an exemplary form of nonvolatile memory.

		Thus, Windows, when combined with the on-demand requester object that includes **OneDrive** and **cldflt.sys** implements a streaming on demand system, with at least one local address, which is stored on at least one of a hard disk, an optical storage device, a non-volatile memory, or a magnetic storage medium, and the data maintained by **OneDrive** and **cldflt.sys** includes the address of the remote location where the original file data is stored, which can be any suitable network location, which includes both a server and a peer computerThis satisfies the limitations of this claim.
10	A streaming on demand system according to claim 1, said at least one address including and address for at least one local source for a first data pack, and an address for at least one remote source for a second data pack, said remote source being a source from the group consisting of a server and a peer computing device.	The limitations of Claim 1 are incorporated by reference. In addition the limitations of Claim 10 are satisfied by the on-demand requester object that includes **Onedrive** and Windows, which utilizes at least one address including an address for at least one local source for a first data pack and an address for at least one remote source for a second data pack, said remote source being a source from the group consisting of a server and a peer computing device. The on-demand requester object including **OneDrive** and **cldflt.sys** satisfies the limitations of this claim because it: • Enables the downloading of the file data from remote to local storage, which requires having address information for both at least one local source data pack and at least one remote source data pack; • Uses the “placeholder file” to identify the location of the locally downloaded data, the unique combination of device, file identifier, path name, and file name constitute an address of the local source. • Enables the locally stored copy of the remote data to be removed, leaving a “placeholder file” behind. This requires that the address of the remote storage location be maintained by these services. Microsoft alludes to this in their “Cloud Sync Engine” documentation where

		they describe placeholder files and say "The file has been hydrated implicitly and could be dehydrated by the system if space is needed." Further, it associates the remote address with the local address. - Microsoft documents the use of "cloud storage" as an inherent part of **OneDrive**. See https://www.microsoft.com/en-us/microsoft-365/blog/2017/05/11/introducing-onedrive-files-on-demand-and-additional-features-making-it-easier-to-access-and-share-files/. The ordinary meaning of "cloud" in this context is "another computer, accessible via the network". Such a computer might be a server, such as is used by Microsoft to implement their **OneDrive** service The "Cloud Sync Engine" documentation provides additional detail about how this is supported by **cldflt.sys** via **cldapi.dll**. Note that by itself, these components do not provide this functionality and thus require **OneDrive** to implement this behavior within the on-demand requester object. See the **CfHydratePlaceholder** function (https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfhydrateplaceholder). This function ensures that specific data is stored locally. In this situation, one unit of data (the "first data pack") is local and the second unit of data (the "second data pack") is not local, which corresponds to the ordinary meaning of the term "remote".This satisfies the limitations of this claim.
11	A streaming on demand system according to claim 1, said minifilter being configured to request to retrieve a required data pack from a first source corresponding to a first address in the table if the data pack is available from the first source, and to identify an alternative address in	The limitations of Claim 1 are incorporated by reference.. The limitations of Claim 11 are satisfied because Microsoft's on-demand streaming system utilizes at least two addresses that can be used to retrieve the data for at least one data pack. The on-demand requester object including **OneDrive** and **cldflt.sys** satisfies the limitations of this claim:

	the table if the data pack is not available from the first source.	• The **OneDrive** data provider implements the ability to retrieve data from multiple remote servers, as Microsoft documents because it is an inherent part of the guarantees they provide for data stored remotely by the **OneDrive** component. This is described by Microsoft on their website (https://docs.microsoft.com/en-us/sharepoint/safeguarding-your-data). • The minifilter retrieves and updates the data within the placeholder file, which includes all information needed by **OneDrive** to access those multiple servers. The **cldapi.dll** exposes this mechanism to the **OneDrive** data provider via APIs such as **CfConvertToPlaceholder** (https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfconverttoplaceholder). This placeholder information is provided back to **OneDrive**, including all information necessary to identify the multiple remote servers, when a file is subsequently needed to satisfy I/O operations from an application. Thus, the on-demand requester object that includes **OneDrive** and **cldflt.sys** does utilize multiple addresses for downloading the data from one of the remote locations where it is stored. This satisfies the limitations of this claim.
12	A streaming on demand system according to claim 1, said minifilter being configured, if the I/O request is a write operation, to mark an area as delivered and then write an address and size indicator corresponding to the write operation to the table.	The limitations of Claim 1 are incorporated by reference. The limitations of Claim 12 are satisfied by Windows because the minifilter component of the on-demand requester object intercepts write I/O operations for files that are used by the streaming on-demand system and maintains information regarding the location, specified by the address and size, in a table. The on-demand requester object including **OneDrive** and **cldflt.sys** satisfies the limitations of this claim:

		• The **cldflt.sys** minifilter registers a function with the filter manager for intercepting write I/O operations (**HsmFltPreWrite**). This registered I/O handler in the minifilter will then be invoked by the filter manager for writes to files. The minifilter will pass along write operations for files that are not part of the on-demand requester system. • The **cldflt.sys** minifilter's **HsmFltPreWrite** function includes logic for handling write operations to a file. For example, it invokes the function **HsmRecallInitiateHydrationEx**, which in turn invokes a function **HsmiMarkModificationBitmapNoLock**, which in turn invokes a function **HsmiMarkFileRangeNoLock**. The function **HsmiMarkFileRangeNoLock** uses both the offset and size of the write operation, which are passed to it as function arguments and then used to mark the modified region of the file. A bitmap in which each block represents some region of the file represent one way to implement marking an area as delivered. • This range information is provided to the **OneDrive** Cloud Sync Engine in order to allow it to read the modified data from the local file and store it on the remote server. See **CfConnectSyncRoot** function, for example, which is exported by **cldapi.dll** and establishes the bidirectional communications between **OneDrive** and **cldflt.sys**, which enables the retrieval of the status of ranges (address and length pairs) within the file. Thus, the streaming on-demand system, which includes a requester object that includes **OneDrive** and **cldflt.sys** implements a system for tracking regions within a remotely stored file that tracks address and size information in at least one table. This satisfies the limitations of this claim.

13	A streaming on demand system according to claim 1, said minifilter being configured, if the I/O request is a create operation, to attach extended data to a file stream, said extended data corresponding to the create operation.	The limitations of Claim 1 are incorporated by reference. The limitations of Claim 13 are satisfied by **OneDrive** running on Windows because the **cldflt.sys** minifilter attaches extended data to a file stream, said extended data corresponding to the create operation. The on-demand requester object including **OneDrive** and **cldflt.sys** satisfies the limitations of this claim because: • The **cldflt.sys** minifilter driver attaches extended data to the file stream. One example of this is that **cldflt.sys** minifilter uses the Filter manager “stream context” functionality. One instance of this is seen when the **cldflt.sys** minifilter function **HsmpiFltPostECPCreate**, which is invoked by the **cldflt.sys** minifilter as part of its processing of the create operation in turn invokes the function **HsmpSetupContext** and that function in turn invokes the Filter manager function **FltSetStreamContext** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltsetstreamcontext). • The **OneDrive** component of the on-demand requester object works with **cldflt.sys** to determine if this is the case by creating a second type of per-stream context within the Windows NTFS file system, known as a **reparse point**. Microsoft documents this: “The cloud files API implements the placeholder system using reparse points.” (https://docs.microsoft.com/en-us/windows/win32/cfapi/build-a-cloud-file-sync-engine). • The presence of the reparse point on a file that is managed by the streaming on-demand remote service implementing that includes **OneDrive** is what causes **cldflt.sys** to attach the “stream context” to the given file stream as part of processing the create operation. • The use of reparse points is one reason why Microsoft restricts support for **OneDrive** to storage devices that are managed by the NTFS file system. Microsoft documents this restriction: “Cldflt.sys currently only supports NTFS volumes because it depends on some

		features unique to NTFS.” (https://docs.microsoft.com/en-us/windows/win32/cfapi/build-a-cloud-file-sync-engine) - The **cldflt.sys** minifilter stores this file stream context specific information in at least one reparse point on a specific file. This can be seen by its use of the **FltTagFile** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-flttagfileex), **FltTagFileEx** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-flttagfileex), and **FltUntagFile** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-flttagfileex) APIs, which are used to set and remove reparse points on the placeholder files. Thus, the streaming on-demand system that includes **OneDrive** and **cldflt.sys** attaches extended data to a file stream within the **cldflt.sys** minifilter while processing the create I/O operation using multiple techniques. This satisfies the limitations of this claim.
14	A streaming on demand method comprising installing an on-demand requester object on a computing device having an I/O manager, said on-demand requester object being configured to receive I/O requests from the I/O manager on behalf of an application for which data is available in data packs for streaming delivery, said on-demand requester object being responsive to application I/O requests, said application I/O requests corresponding	The limitations of Claim 14 are satisfied by an application program running on a computer system using Windows because the distinctive claim elements are present after installation of one or more software components. Windows satisfies application I/O requests utilizing an I/O stack, which includes at least a filter manager and one or more software components (drivers) that work in combination to realize the I/O stack and produce the necessary response to the application’s I/O request. The term “Windows” means at least the current version and more recent versions of Windows 10 running on a computer.

	to sequential and non-sequential data packs, said data packs being portions of a complete data stream, said I/O requests determining an order in which data packs are sought for the application, said on-demand requester object comprising a minifilter associated with a filter manager in an I/O stack, said filter manager being configured to call the minifilter, said minifilter having a preoperation callback routine for the I/O request registered with the filter manager, and said minifilter receiving each I/O request from the I/O manager, said minifilter referencing a table that includes names and paths with at least one offset address and length where each data pack required to fulfill each I/O request is located, said minifilter determining if the data pack required to fulfill each I/O request been streamed to the system, and said application using the data packs that fulfill each I/O request when the data packs are available on the computing device.	The term “OneDrive” is used to describe a data provider element, which is part of the on-demand requester object defined within the claim, that consists of components which are both resident on the local computer system (e.g., the “OneDrive for Windows” service, which is part of Windows) and the supporting infrastructure, including software and hardware, provided and/or managed by Microsoft to at least manage, read, write, and replicate and enable transfer of data from a remote storage location to the local Windows system, as needed by Windows to satisfy the I/O requests of the application that generated the I/O request.Of these required elements, the following are present in every computer running an application in Windows after installation or an update of Windows: • The application program that issues I/O requests within the accused environment, though the application itself is normally not part of the on-demand requester object. Any application program can trigger the behavior of the on-demand requester object described within the patent because executing an application inherently requires generating I/O requests, simply to execute the application. Given that the function of a modern computer system is to run application programs, this element is always present. The application’s generation of I/O requests is not part of the claim but is a necessary pre-requisite to their receipt and processing of the on-demand requester object. • A filter manager. All versions of Microsoft Windows include a component known as the “Filter Manager”. Microsoft documents this: “The filter manager is installed with Windows, but becomes active only when a minifilter driver is loaded. The filter manager attaches to the file system stack for a target volume. A minifilter driver attaches to the file system stack indirectly, by registering with the filter manager for the I/O operations the minifilter driver chooses to filter.” (https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts). • An I/O stack. Windows has included an I/O stack in all versions of Windows since at least Windows XP. Microsoft documents this in numerous places and uses a variety of terms to describe how this I/O stack is implemented, such as describing the layers of the I/O stack

		(https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/overview-of-the-windows-i-o-model). Thus, this claim element is inherent in Windows and is always present. Nonlimiting examples of how Microsoft installs software that realizes a system meeting the limitations of this claim, including the presence of an on-demand requester object are: • A clean installation of Windows onto a new computer; • An upgrade installation of Windows onto a computer running Windows 7; • An upgrade installation of Windows onto a computer running Windows 8; • An upgrade installation of Windows onto a computer running Windows 8.1; • An upgrade installation of Windows onto a computer running an older version of Windows, whether or not the prior version of Windows supported **OneDrive** and/or included **cldflt.sys**; • Installation of Windows into a virtual machine, for example (but not limited to) Windows and Windows Server with Hyper-V (https://www.onmsft.com/how-to/how-to-install-a-virtual-machine-on-windows-10-using-hyper-v-now-even-easier-with-quick-create). Each of these typically results in a Windows installation that includes support for **OneDrive** and uses **cldflt.sys** to provide access to Microsoft's streaming on-demand cloud storage service. The combination of these elements meets the claim definition of an on-demand requester object, working with a filter manager, and satisfying I/O requests of the application for the OneDrive maintained data, which utilizes one or more tables to determine if the data has been streamed to the local system and if not, retrieving said data by utilizing data packs when they are available on the local computing device. A nonlimiting example of how Microsoft achieves this is their own description of how the accused product is realized in Windows. Windows includes **OneDrive**. Microsoft documents that this is included in Windows: "If you have Windows 10, OneDrive is already installed on your PC. If you're

		using another version of Windows, install OneDrive to get started." (https://www.microsoft.com/en-us/microsoft-365/onedrive/download) **OneDrive**, which is part of Windows, implements a streaming on-demand data storage service, in which data is stored in data packs within a Microsoft data center and streaming downloaded, on demand, to the local computer system, said demand being initiated by the I/O requests of the application and if the data has not previous been downloaded and stored on the local computer system running Windows. This use is consistent with Microsoft's own use of the name **OneDrive**. The on-demand requester object of the claim is realized when **OneDrive** is combined with the **cldflt.sys** minifilter driver, which is distributed as part of Windows. This on-demand requester object satisfies the application I/O request, either by using data that has been stored locally, or data that it retrieves over the network from Microsoft's data center. Since its introduction to Windows, Microsoft has formalized the use of **cldflt.sys** by implementing and documenting a library that can be used by specialized applications to allow both Microsoft and third parties to develop "Cloud Sync Engines". Microsoft documents these on their website (https://docs.microsoft.com/en-us/windows/win32/cfapi/cloud-files-api-portal). Microsoft explains in detail how Windows incorporates support for streaming on demand file download. The **cldflt.sys** minifilter is sufficient to activate the function of the Windows Filter Manager. The version of **OneDrive** that is included in Windows is now an implementation of a "Cloud Sync Engine" within Windows (https://docs.microsoft.com/en-us/windows/win32/cfapi/cloud-files-api-portal). **OneDrive** consists of both local and remote components, which includes executables, dynamic link libraries, and control panel "applets" that implement the streaming on-demand cloud storage system.

The combination of **OneDrive** and **cldflt.sys** correspond to the "on-demand requester object" defined by this claim. To satisfy this definition, the on-demand requester object must include a minifilter and must have a mechanism for retrieving the necessary data on-demand, which need not be part of the minifilter. The storage data provider service, which is an inherent part of the on-demand requester object, downloads data as needed and may store a copy of that data on the local computer. **OneDrive** works with the **cldflt.sys** minifilter, the Windows I/O stack, filter manager, and other necessary components to realize an on-demand requester object as defined by the claim.

All versions of Microsoft Windows since at least the release of Version 1709 include the required elements:

- An on-demand requester object installed on the computing device configured to receive I/O requests on behalf of that application and either satisfy the I/O request if the data is located on the local computer system, or fetched from a remote location. Said on-demand requester object consisting of:
  - o **OneDrive** which is a cloud storage solution provided by Microsoft as part of current Windows versions. **OneDrive** permits seamless integration of streaming on-demand cloud storage with Windows (https://www.windowscentral.com/how-get-started-onedrive-windows-10).
  - o A dynamic link library, **cldapi.dll** that:

    (4) Is included with Windows and provides a programmatic mechanism for **OneDrive** to communicate with the **cldflt.sys** minifilter, described below, that, when combined, are consistent with the "on-demand requester object" of the patent.

    (5) Provides at least one mechanism for transferring data from a remote storage service to the local file system service. For example, **CfGetTransferKey** is a documented mechanism that is made available from **cldapi.dll** that Microsoft

		describes as "… an alternative to **CfHydratePlaceholder** to proactively initiate data transfer into a placeholder." https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfgettransferkey (6) Provides at least one mechanism for determining information about the status of regions of a specific locally maintained file. The application programming interface (API) **CfGetPlaceholderRangeInfo**, which is made available from **cldapi.dll**, specifies a file handle, which is sufficient to determine the identity of a file, including a path and name, the status of the requested range (the **CF_PLACEHOLDER_RANGE_INFO_CLASS** parameter), the **StartingOffset** and **Length** parameters, as well as a location in which the requested information is returned (**InfoBuffer, InfoBufferLength,** and **ReturnedLength**). https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfgetplaceholderrangeinfo. The **cldapi.dll** and **cldflt.sys** minifilter work in tandem to maintain this information and ensure it is updated properly within the minifilter. This is accomplished by using a communications protocol . From this, it appears likely that the minifilter maintains at least one table of state information. o A minifilter, **cldflt.sys** that: (6) Is associated with the Microsoft Windows filter manager (https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts). According to Microsoft the filter manager is included in all versions of Windows, as it is universal (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltregisterfilter); (7) The filter manager is configured to call the minifilter. The **cldflt.sys** minifilter does this by invoking **FltRegisterFilter** and specifying entry points that filter manager is to invoke to handle specific input and output (I/O) operations, at

		least including opening, reading, and writing the contents of files, such files residing on the local computer system and/or on the cloud storage system; (8) The **cldflt.sys** minifilter driver uses existing functionality of the Windows operating system core operating system to maintain tables of information, including **RtlInsertElementGenericTableAvl** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntddk/nf-ntddk-rtlinsertelementgenerictableavl)**, FsRtlGetNextMbaseMcbEntry** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/ntifs/nf-ntifs-_fsrtl_advanced_fcb_header-fsrtlgetnextmcbentry), **FltParseFileNameInformation (**https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltparsefilename), **FltSetStreamContext** (https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltsetstreamcontext), and **FltSetFileContext (**https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/fltkernel/nf-fltkernel-fltsetfilecontext). These routines are consistent with both the use of tables and tracking regions of files that have been allocated. This, combined with the detailed range information provided by **cldflt.dll,** at a minimum corresponds to the table that includes this information and which data packs have been streamed to the local system. (9) The **cldflt.sys** minifilter driver ensures that the data required to satisfy an application I/O request is present on the local system before that I/O operation is completed. (10)The **cldflt.sys** minifilter driver is a necessary component of the on-demand requester object. When **cldflt.sys** is bypassed, we do not observe the on-demand behavior of normal Windows applications when using **OneDrive**. This is apparent if one uses Windows Subsystem for Linux (WSL), which bypasses

		the **cldflt.sys** driver functionality. Thus, Linux applications within the WSL environment do not see the correct data content of the file. Thus, a computer system with Windows installed includes all of the necessary elements to meet the requirements of this claim: at least one application that forms I/O requests, an I/O stack, an I/O Manager, a filter manager, and an on-demand requester object in which the on-demand requester object includes a minifilter that registers and handles specific I/O operations. This satisfies the limitations of this claim.
15	A streaming on demand method according to claim 14 further comprising attaching said minifilter to the I/O stack by registering with a filter manager in the I/O stack for I/O requests for the application for which data is available in data packs for streaming delivery.	The limitations of Claim 14 are incorporated by reference. In addition, the limitations of Claim 15 are satisfied by the **cldflt.sys** minifilter component of the "on-demand requester object" registering with the Windows Filter manager in the I/O stack. This is satisfied by the fact that the **cldflt.sys** minifilter calls the Windows kernel routine **FltRegisterFilter** and specifies a list of callbacks for I/O requests for any application accessing the specific storage device. The **cldflt.sys** minifilter is constructed in such a way that if the data is not provided by the "on-demand requester object" the I/O requests for the application are passed along the I/O stack. Thus, **cldflt.sys** does intercept operations for I/O requests for application for which the data is available in data packs for streaming delivery. This satisfies the limitations of this claim.
16	A streaming on demand method according to claim 14 further comprising, said minifilter passing on a read call if a required data pack is available locally and if all data in the required data pack has been received.	The limitations of Claim 14 are incorporated by reference. In addition, the limitations of Claim 16 are satisfied by **cldflt.sys** minifilter blocks the read call while **OneDrive for Windows**, the data provider component of the "on demand requester object", retrieves the data from its alternative location (**HsmiRecallInitiateHydrationEx** which is invoked by **HsmFltPreREAD**, which is the read call

		handler that **cldflt.sys** registers with the Filter Manager and is invoked to handle read calls from the application) and once that data has been retrieved satisfies the application request. Further, the **cldflt.sys** minifilter blocks the read call while the data provider component of the "streaming on-demand requester object" retrieves the data from its alternative location and once that data has been retrieved satisfies the application request. One example of how **cldflt.sys** minifilter implements this is in **HsmpRecallInitiateHydrationEx**, where the minifilter inspects the table that it maintains of data which has been written locally and if it does not find the required data, then requests that the data provider retrieve the specified data and blocks the original read request until the data has been provided by the data provider and can then be returned to the application. Thus, once the conditions are satisfied and the data needed by the application is locally resident ("required data pack") the minifilter completes the read request and the application is provided with the relevant data. This can also be observed by an ordinary user of Windows by accessing a file that is not stored locally when using **OneDrive** in Windows. The application blocks until the contents of at least some portion of the file have been retrieved and the delay involved is generally noticeable. The second access to that file, once it has been stored locally, is much faster. This satisfies the limitations of this claim.
17	A streaming on demand method according to claim 14 further comprising, said minifilter withholding a read call if a required data pack is not available locally.	The limitations of Claim 14 are incorporated by reference. In addition, the limitations of Claim 17 are satisfied by the **cldflt.sys** minifilter component of the on-demand requester object. It does so by blocking ("withhold") a read call if data required to satisfy the application read I/O is not present in the locally downloaded copy of the file.

		The **cldflt.sys** minifilter meet this claim limitation because **cldflt.sys** blocks the read call while the **OneDrive** component of the on-demand requester object downloads the necessary data. This program logic is present in the code routine named **HsmFltPreREAD**, which is part of **cldflt.sys**. This satisfies the limitations of this claim.
18	A streaming on demand method according to claim 14 further comprising, said minifilter withholding a read call if all data in the required data pack has not been received.	The limitations of Claim 14 are incorporated by reference. In addition, the limitations of Claim 18 are satisfied by the **cldflt.sys** minifilter component of the "on-demand requester object". Specifically, the **cldflt.sys** minifilter blocks the read call while **OneDrive** component of the "on demand requester object" retrieves the data from its alternative location. Until the required data has been retrieved by the data provider, the application is normally blocked from proceeding. The **cldflt.sys** minifilter implements this functionality in the code routine named **HsmFltPreREAD**. Further, an ordinary user of Windows can demonstrate that it blocks simply by accessing a file that is stored in **OneDrive**. The application blocks until the contents of the file requested by the application becomes available. Once the conditions are satisfied and the data needed by the application is locally resident ("required data pack") the application is provided with the relevant data. This satisfies the limitations of this claim.
19	A streaming on demand method according to claim 14, said at least one address including and address for at least one local source for a first data pack, said local source being a source from the group consisting of a	The limitations of Claim 14 are incorporated by reference. The limitations of Claim 19 are satisfied by the on-demand requester object. It does this by utilizing an address for at least one locally stored data pack and one remotely stored data pack and must utilize a local storage source that consists of at least one of the specified storage media.

	hard disk, an optical storage medium, a nonvolatile memory, a magnetic storage medium, and an address for at least one remote source for a second data pack, said remote source being a source from the group consisting of a server and a peer computing device.	Specifically, the on-demand requester object including **OneDrive** and **cldflt.sys** by: • Enabling the data of a file to be downloaded to local storage to satisfy I/O requests issued by an application. The process of downloading the data from a remote location requires having address information for at least one remote source data pack; • The **cldflt.sys** file uses the “placeholder file” to identify the location of the locally downloaded data, the unique combination of device, file identifier, path name, and file name constitute the address of the local source. • The **OneDrive** service combined with **cldflt.sys** permits the locally stored copy of the remote data to be removed, leaving a “placeholder file” behind. This requires that the address of the remote storage location be maintained by these services. Microsoft alludes to this in their “Cloud Sync Engine” documentation where they describe placeholder files and say “The file has been hydrated implicitly and could be dehydrated by the system if space is needed.” • Microsoft documents the use of “cloud storage” as an inherent part of **OneDrive**. See https://www.microsoft.com/en-us/microsoft-365/blog/2017/05/11/introducing-onedrive-files-on-demand-and-additional-features-making-it-easier-to-access-and-share-files/. The ordinary meaning of “cloud” in this context is “another computer, accessible via the network”. Such a computer might be a server, such as is used to implement Microsoft’s **OneDrive** service, or another individual user computer system, such as is described by Microsoft as “fetching”; the file comes from one computer running Windows, to a second computer running Windows. See https://www.pcmag.com/how-to/how-to-manage-sync-and-share-files-in-microsoft-onedrive for a description of this feature. This is determined on at least a file-by-file basis: local storage for files that have already been downloaded to the computer system, OneDrive storage, which is maintained on Microsoft owned and managed servers,

		and other peer devices, which includes computers running Windows with OneDrive and being properly configured to enable fetch. Modern computers running Windows typically utilize either solid state (“solid state drive” or “SSD”) or rotating magnetic media devices (“hard disk”), or variations of these devices (“hybrid” drives). The NTFS file system, which is required by Microsoft to support **OneDrive**, supports any writeable form of media, including re-writeable optical media (“read/write bluray”, “read/write DVD”, and “read/write CD”), magnetic media (including hard disks), and **Intel® Optane® DC Persistent Memory** (https://www.intel.com/content/www/us/en/architecture-and-technology/optane-dc-persistent-memory.html), which is an exemplary form of nonvolatile memory. The “Cloud Sync Engine” documentation provides additional detail about how this is supported by **cldflt.sys** via **cldapi.dll**. Note that by itself, these components do not provide this functionality and thus require **OneDrive** to implement this behavior. See the **CfHydratePlaceholder** function (https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfhydrateplaceholder). This function ensures that specific data is stored locally. In this situation, one unit of data (the “first data pack”) is local and the second unit of data (the “second data pack”) is not local, which corresponds to the ordinary meaning of the term “remote”. This satisfies the limitations of this claim.
20	A streaming on demand method according to claim 14, said minifilter requesting to retrieve a required data pack from a first source corresponding to a first address in the table if the data pack is available from the first source,	The limitations of Claim 14 are incorporated by reference. The limitations of claim 20 are satisfied by the system including **OneDrive** and **cldflt.sys**: • The **OneDrive** data provider implements the ability to retrieve data from multiple remote servers, as Microsoft documents because it is an inherent part of the guarantees they

	and identifying an alternative address in the table if the data pack is not available from the first source.	provide for data stored remotely by the **OneDrive** component. This is described by Microsoft on their website (https://docs.microsoft.com/en-us/sharepoint/safeguarding-your-data). • The minifilter retrieves and updates the data within the placeholder file, which includes all information needed by **OneDrive** to access those multiple servers. The **cldapi.dll** exposes this mechanism to the **OneDrive** data provider via APIs such as **CfConvertToPlaceholder** (https://docs.microsoft.com/en-us/windows/win32/api/cfapi/nf-cfapi-cfconverttoplaceholder). This placeholder information is provided back to **OneDrive**, including all information necessary to identify the multiple remote servers, when a file is subsequently needed to satisfy I/O operations from an application. Thus, the on-demand requester object that includes **OneDrive** and **cldflt.sys** does utilize multiple addresses for downloading the data from one of the remote locations where it is stored. This satisfies the limitations of this claim.