

NON-INVASIVE REMOTE ACCESS TO AN APPLICATION PROGRAM

BACKGROUND

[0001] Ubiquitous remote access to application programs and data has become commonplace as a result of the growth and availability of broadband and wireless network access. In addition, users are accessing application programs and data using an ever-growing variety of client devices (e.g., mobile devices, table computing devices, laptop/notebook/desktop computers, etc.). Data may be communicated to the mobile device from a remote server over a 3G and 4G mobile data networks or wireless networks such as WiFi and WiMax. Most mobile devices have access to the Internet and are able to interact with various types of application programs.

SUMMARY

[0002] Disclosed herein are systems and methods for non-invasively providing remote access to an application program executing on, e.g., a server. In accordance with some implementations, a method for providing remote access to a server-based application program is disclosed. The method includes providing a server remote access program at a server computing device; receiving a communication at the server computer computing device from a client computing device to access the application program; providing a screen scraping application at the server computing device to scrape screen data associated with the application program; generating presentation data from the screen data, the presentation data being generated in accordance with a selection of a display region of the application program; updating a state model containing a corresponding state of the application program; and communicating the presentation data and the state model to the client computing device using the server remote access program.

[0003] In accordance with some implementations, another method of providing remote access to at least one server-based application program is disclosed. The method may include providing a server remote access program at a server computing device; enumerating at least one Uniform Resource Locator (URL) that is associated with the at least one application program; receiving a communication from a client at the at least one URL; providing a screen scraping application at the server computing device to scrape screen data associated with the at least one application program associated with the at least one URL; generating presentation data from the screen data, the presentation data being generated in accordance with a selection of a display region of the at least one application program; and communicating the presentation data to the client computing device using the server remote access program.

[0004] In accordance with some implementations, there is provided a method for providing non-invasive remote access to an application program. The method may include providing a server remote access program at a server computing device; providing a screen scraping application at the server computing device that monitors a display interface generated by the application program for changes to the display interface; scraping screen data associated with the display interface; generating presentation data from the screen data, the presentation data being generated in accordance with a selection of a display region of the application program; receiving a request at the server computing device to access the application program; updating a state model containing a corresponding state of the application program, wherein the corresponding state includes at least one logical element containing information about the display interface of the application program; and communicating the presentation data and the state model using the server remote access program in response to the request.

[0005] In accordance with yet other implementations, there is provided a method of providing remote access to a server-based application program that includes providing a remote access program at a server computing device; determining application programs that are executing on the server computing device; enumerating a Uniform Resource Locator (URL) that is associated with each application program determined by the server computing device to be executing; providing access to the URL of each application program determined by the

server computing device to be executing; receiving a selection from a client computing device at the URL of a selected application program; and providing remote access to the application program at the URL.

[0006] Other systems, methods, features and/or advantages will be or may become apparent to one with skill in the art upon examination of the following drawings and detailed description. It is intended that all such additional systems, methods, features and/or advantages be included within this description and be protected by the accompanying claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] The components in the drawings are not necessarily to scale relative to each other. Like reference numerals designate corresponding parts throughout the several views.

[0008] Fig. 1 is a simplified block diagram illustrating a system for providing remote access to an application program via a computer network;

[0009] Figs. 2A-2C are simplified block diagrams illustrating operation of a screen scraping application and server and client remote access program(s);

[0010] Fig. 3 illustrates a state model of the system of Fig. 1;

[0011] Fig. 4 illustrates a flow diagram of example operations performed within the system of Figs. 1-3;

[0012] Fig. 5 illustrates a flow diagram of example operations performed within the system of Figs. 1-2; and

[0013] Fig. 6 illustrates an exemplary computing environment.

DETAILED DESCRIPTION

[0014] Unless defined otherwise, all technical and scientific terms used herein have the same meaning as commonly understood by one of ordinary skill in the art. Methods and materials similar or equivalent to those described herein can be used in the practice or testing of the present disclosure. While implementations will be described for remotely accessing and viewing application programs, it will become evident to those skilled in the art that the implementations are not limited thereto.

[0015] Referring to FIGS. 1 and 2A-2C, there is illustrated a system 100 for providing remote access to an application program via a computer network. The system 100 may include one or more server computing devices 102A, 102B and one or more client computing devices 112A, 112B, 112C ... 112N. The client computing devices may include, but are not limited to, a wireless handheld device such as, for example, an IPHONE 112A or a BLACKBERRY 112B connected via a communication network 110 such as, for example, the Internet, to the server computing device 102A and/or the server computing device 102B. Similarly, the client computing devices may also include a desktop/notebook personal computer 112C or a tablet device 112N that are connected by the communication network 110 to the server computing device 102A and/or the server computing device 102B. It is noted that the connections to the communication network 110 may be any type of connection, for example, Wi-Fi (IEEE 802.11x), WiMax (IEEE 802.16), Ethernet, 3G, 4G, etc.

[0016] The server computing device 102A may be connected to a first Local Area Network (LAN) 109A and the server computing device 102B is connected to a second Local Area Network (LAN) 109B that are connected to the communications network 110. It is noted that any number of server computers may be connected to the communications network 110 in accordance with the present disclosure. FIG. 6 illustrates an exemplary computing device that

may be used as the server computing device 102A, the server computing device 102B, or the client computing devices 112A, 112B, 112C ... 112N.

[0017] As shown in FIG. 1, a client remote access program is executed on a processor 118A, 118B, 118C...118N of the client computing devices 112A, 112B, 112C ... 112N. The client remote access program is provide as executable commands stored in memory 120A, 120B, 120C ... 120N of the client computing devices 112A, 112B, 112C ... 112N. The client remote access program communicates with a user interaction program such as, for example, a web browser or native application. The user interaction program may display the screen data (from the server computing device 102A or 102B), or receive user input data for interacting with the first and the second screen scraping application using, for example, a graphical display with touch-screen 114A, 114N, a graphical display 114B, or a keyboard 116B, 116C of the client computing devices.

[0018] The server remote access programs 107A and 107B and the client remote access program may be implemented using standard programming languages and communication is enabled using standard communication technologies such as, for example, Hyper Text Transfer Protocol (HTTP), virtual private networks (VPN), and secure socket layers (SSL) which are well known to those skilled in the art. Further, an instance of a server remote application may be provided on a different server than the server executing the screen scraping application. *In such an implementation, screen data may be communicated over a direct or networked connection between the server executing the screen scraping application and the other server executing the server remote application.* In some implementations, the server remote access programs 107A and 107B and the client remote access program enable the implementation of aspects of the present disclosure as a retrofit to existing technologies on the server side as well as on the client side.

[0019] As shown in FIG. 2A, in accordance with some implementations, the server computing device 102A may execute a screen scraping application 105A that monitors a display interface generated by an application program 103A for events that may cause the pixels in the display to change. The screen scraping application 105A is stored in memory 106A and executed on the processor 104A of the server computing device 102A. As events occur, the screen scraping application 105A reads the changed regions of the display from a frame buffer (not shown), and may compress the screen image data, and send the compressed image data to the respective server remote access program 107A. The server remote access program 107A is stored in the memory 106A and is executed on the processor 104A of the server computing device 102A. In accordance with the present disclosure, the interaction of the screen scraping application 105A with the application program 103A provides for a non-invasive implementation to enable remote access to the application program 103A, as described below.

[0020] To provide remote access to the application program 103A, communication is established between, e.g., the client computing device and the server computing device 102A at an enumerated URL associated with the application program 103A. In particular, the communication is enabled by the server remote access program 107A and the client remote access program, as shown in FIG 2A. The communication may be over any type of network, as noted above. It is noted that any of client computing devices 112A, 112B, 112C ... 112N may be the client device illustrated in Fig. 2A.

[0021] During a remote communications session with the client computing device, the server remote access program 107A receives screen data from the screen scraping application 105A. Upon receipt, the server remote access program 107A generates presentation data 109A of the screen image data and transmits the same to the client remote access program.

[0022] Optionally or additionally, the presentation data 109A may be generated according to hardware capabilities of the client computing device 112A, 112B, 112C or 112N, for example, accounting for processing capacity, memory size, type of graphical display, and type of user interface. Thus, the server computing device 102A may scale or resize the screen data that is communicated to the client computing device such that the screen data is displayed on the client computing devices 112A, 112B, 112C ... 112N in accordance with the characteristics of the device. As such, each type of device that is participating in the collaborative session presents the user interface having a device-appropriate resolution based on information contained in the state model of FIG. 3, described below. For example, presentation data generated and transmitted for a laptop computer are different from presentation data generated and transmitted for a handheld device such as, for example, an IPHONE.

[0023] In some implementations, the presentation data 109A is generated based on a selection of a window or display region of the application program 103A. For example, a selection of a region or window to be displayed on the client computing device may be received at the server computing device 102A. The server computing device 102A may then crop the screen data such that only the selected region or window is generated in the presentation data 109A and communicated by the server remote access program 107A to the client remote access program.

[0024] In some implementations, the presentation data 109A may also be provided to the server computing device 102A. This enables a user at the server computing device 102A to see what is being displayed at the client computing devices 112A, 112B, 112C or 112N.

[0025] With reference now to Fig. 2B, in some implementations, the client computing device 112A, 112B, 112C or 112N may interact with both application programs 103A

and 103B running on the server computing devices 102A and 102B by accessing the application programs 103A and 103B at a respective enumerated URL associated with each of the application programs 103A and 103B. As such, the client computing device 112A, 112B, 112C or 112N may participate in a collaborative session with the server computing device 102A and 102B, and may present a view containing the display of the application programs 103A and 103B.

[0026] Optionally or additionally, the display of the application programs 103A and 103B may be presented through a process called recombination, which may be based on information contained in the state model of FIG. 3, described below. With recombination, the application programs 103A and 103B are seamlessly integrated into the same user interface of the client computing device 112A, 112B, 112C or 112N such that views of both application programs 103A and 103B may be presented simultaneously. Optionally or additionally, functionalities and controls associated with the application programs 103A and 103B may be combined within the same user interface.

[0027] As shown in Fig. 2B, in accordance with some implementations, the server computers 102A and 102B execute respective screen scraping applications 105A and 105B that monitor a display interface generated by an application program 103A and 103B for events that may cause the pixels in the display to change. As events occur, the screen scraping applications 105A and 105B read the changed regions of the display from a frame buffer (not shown), and may compress the screen image data, and send the compressed image data to the respective server remote access program 107A and 107B. As noted above, the interaction of the screen scraping applications 105A and 105B with the application program 103A and the application program 103B, respectively, provides for a non-invasive implementation to enable remote access to the application program 103A and the application program 103B. Other aspects of

the environment of Fig. 2B operate in a substantially similar fashion, as described with regard to Fig. 2A.

[0028] With reference now to Fig. 2C, in some implementations, two or more of the client computing devices 112A, 112B, 112C ... 112N may collaboratively interact with the application program 103A running on the server computing device 102A by accessing the same enumerated URL. As such, each of the client computing devices 112A, 112B, 112C ... 112N (and the server computing device 102A) participating in a collaborative session may present a synchronized view of the display of the application program 103A that is made available at an enumerated URL. Alternatively, the client computing devices may interact with the application program 103B running on the server computing device 102B.

[0029] During a remote communications session with the client computing device, the server remote access program 107A receives screen data from the screen scraping application 105A. Upon receipt, the server remote access program 107A generates first presentation data 109A of the first screen image data and transmits the same to the client remote access programs 112A and 112N. Other aspects of the environment of Fig. 2C operate in a substantially similar fashion, as described with regard to Fig. 2A.

[0030] The environment of FIG. 2C further provides for collaborative browsing (co-browsing) of the application program 103A. For example, user of client computing device 112A and a user of the client computing device 112N may wish to participate a session that includes the application program 103A at the server computing device 102A. The client computing device 112A and the client computing device 112N may remotely access the application 103A using a client remote access program that communicates with a user interaction program such as, for example, a web browser. Co-browsing of the application program 103A enables users at both the client computing device 112A and the client computing device 112N to interact and

view the same documents, images, etc. on the user's respective web browser. Thus, if the user client computing device 112A requests information from the application program 103A for display, the other browser in the session running on client computing device 112N will receive the same information.

[0031] In such an implementation, the remote access program 107A may act as a proxy server to intercept requests from the various web browsers in a co-browsing session. A web browser, once configured to use the remote access program 107A as a proxy server, sends a complete URL request to the remote access program 107A. The remote access program 107A then retrieves screen data from the screen scraping application 105A associated with the application program 103A, and sends the result of the request back to the web browsers.

[0032] The web browsers may execute a client remote access program (e.g., Flash, Silverlight, Java) to enable the co-browsing session to receive information. Additionally or alternatively, HTML5 may be used to enable the web browser to natively receive information. As such, the web browser may be the client remote access program.

[0033] In some implementations, users co-browsing within the system 100 may be interacting with plural application programs 103A and 103B (e.g., a hybrid of the environments of FIG 2B and 2C). Accordingly, the present disclosure may provide for recombination of the application programs 103A and 103B, where the applications are seamlessly integrated into the same user interface, which is presented on each of the client computing devices 112A, 112B, 112C...112N participating in the co-browsing session. Recombination may be provided based on information contained in the state model of FIG. 3, described below.

[0034] In some implementations, the operation of the remote access program (server and client) is performed in cooperation with a state model 300, as illustrated in Fig. 3. When executed, the client remote access program updates the state model 300 in accordance

with user input data received from a user interface program. The remote access program may generate control data in accordance with the updated state model, and provide the same to an application program running on the server computing device 102A or 102B (via the screen scraping application). The application program may be any application running on the server computing device 102A or 102B with which an end user interacts using one of the client computing devices 112A, 112B, 112C ... 112N.

[0035] Upon receipt of application data from the screen scraping application, the server remote access program updates the state model 300 in accordance with the screen or application data, generates presentation data in accordance with the updated state model, and provides the same to the user interface program on the client computing device. The state model 300 comprises an association of logical elements of the application program with corresponding states of the application program, with the logical elements being in a hierarchical order. For example, the logical elements may be a screen, a menu, a submenu, a button, etc. that make up the application program user interface. This enables the client device, for example, to natively display the logical elements. As such, a menu of the application program 103A that is presented on a mobile phone will look like a native menu of the mobile phone. Similarly, the menu of the application program 103A that is presented on desktop computer will look like a native menu of the desktop computer operating system. In accordance with aspects of the present disclosure, the screen image may be communicated as a single logical element in a separate channel to the client. The screen image may be referenced in the state model to enable the client remote access program to retrieve the screen image from the separate channel and display it on the client. As such, the screen will have a substantially similar appearance on each of the client computing devices 112A, 112B, 112C ... 112N, although it may be scaled according to the characteristics of each device.

[0036] The state model 300 is determined such that each of the logical elements is associated with a corresponding state of the application program. The state model 300 may be determined such that the logical elements are associated with user interactions. For example, the logical elements of the application program are determined such that the logical elements comprise transition elements with each transition element relating a change of the state model 300 to one of control data and application representation data associated therewith.

[0037] In some implementations, the state information may include a reference or references to one or more rectangular regions that comprise the screen image generated by the application program. A control may be provided by the client remote access program to enable a user to select among the one or more rectangular regions for display at the client device. In some implementations, the control may provide a selection among one or more rectangular regions of screen image data created by plural application programs (e.g., 103A and 103B) for selection by a user.

[0038] In some implementations, as noted above with regard to FIGS. 2A-2C, the state information may include information about a state of a display of the client computing device. For example, display resolution may be maintained in the state model 300, which may be used as part of the refactoring of the presentation data in accordance with the client computing device's capabilities. The state model may also include information that may be used for recombination purposes.

[0039] Fig. 4 illustrates an operation flow diagram 400 of processes performed to provide remote access to a server-based application program utilizing the state model 300. At 402, a remote access program is provided at a server computing device. For example, server remote access program 107A may be provided on the server computing device 102A to enable

remote access by client computing devices running a complementary client remote access program.

[0040] At 404, a communication is received from a client. A user may contact the server computing device 102A using the user interaction program on a client computing device. The client may also provide information to the server remote access program 107A about the client computing device's characteristics and capabilities.

[0041] At 406, the screen associated with the application is scraped. The display generated by the application program 103A scraped by the screen scraping application 105A. At 407, the screen may be scaled or cropped, if desired, by the application program (service). At 408, the presentation data is generated for communication to the client. In accordance with the client computing device's characteristics and capabilities, the service (i.e., screen scraping application) may scale or crop display information that is provided to the client computing device such that it is appropriately sized for the client computing device.

[0042] At 410, the state model is updated. As described above, the state model 300 is determined such that each of the logical elements is associated with a corresponding state of the application program. The screen image, which is communicated in a separate channel, may be as a single logical element that is referenced in the state model. At 412, the presentation data and the state model are communicated to the client. The communication may be over the communications network 110. At 414, the screen data is retrieved from the separate channel and presented on the client device for viewing.

[0043] In accordance with some implementations, a user may interact with the application program 103A through the user interaction program and the cooperation of the client and server remote access programs. As such, the user interaction(s) may be used to

update the state model at 416, which may be returned to the server computing device 102A to update the application program 103A.

[0044] Thus in view of the above, there is provided a system and method for non-invasively providing remote access to an application program running on a server.

[0045] Fig. 5 illustrates an operation flow diagram 500 of processes performed to provide remote access to a server-based application program. In some implementations, the server remote access programs 107A and 107B generate one or more Uniform Resource Locators (URL) by which the client computing device is able to connect to one or more application programs running on the server computing device 102A or 102B. Each application program may have a unique URL associated therewith. The URLs are communicated to a client computing device to enable a user to select and interact with one or more of the application programs on the server computing device. For example, in the user interaction program, icons, links, menu items, etc. may be presented that represent each of the enumerated URLs. A user may select the URL by a selection of the appropriate icon, link, menu item, etc.

[0046] In the operational flow 500, the URL mechanism described above is employed to provide remote access to the server-based application program. At 502, a remote access program is provided at a server computing device to enumerate one or more URLs associated with one or more applications. For example, server remote access program 107A may be provided on the server computing device 102A. A URL may be accessed within the user interaction program on a client computing device to access, e.g., the application program 103A running on the server computing device 102A. The server remote access program 107A may determine which application programs are running on the server computing device 102A to enumerate the URL associated with each of the applications and to provide access to the determined application programs at the URL.

[0047] At 504, a communication is received from a client. A user may contact the server computing device 102A by connecting to one or more of the enumerated URLs that are provided in the user interaction program on a client computing device. The client may also provide information to the server remote access program 107A about the client computing device's characteristics and capabilities.

[0048] At 506, the screen associated with the application is scraped. The display generated by the application program 103A scraped by the screen scraping application 105A. At 508, the presentation data is generated for communication to the client. In accordance with the client computing device's characteristics and capabilities, the server computing device 102A may scale, resize or crop display information that is provided to the client computing device such that it is appropriately sized for the client computing device.

[0049] At 510, the presentation data is communicated to the client. The communication may be over the communications network 110. At 512, the screen data is presented on the client device for viewing. In accordance with some implementations, a user may interact with the application program 103A through the user interaction program and the cooperation of the client and server remote access programs. As such, the user interaction(s) may be used to access one or more URLs presented in the user interaction program. At 514, a connection may be established via a selected URL such that the user may interact with a different application or currently with multiple applications. Further, more than one client computing device may remotely interact with an application program in accordance with Fig. 3 to provide a collaborative environment where plural client computing devices concurrently interact with the application program at an enumerated URL.

[0050] Thus in view of the above, there is provided yet another system and method for non-invasively providing remote access to an application program running on a server.

[0051] Fig. 6 shows an exemplary computing environment in which example embodiments and aspects may be implemented. The computing system environment is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality.

[0052] Numerous other general purpose or special purpose computing system environments or configurations may be used. Examples of well known computing systems, environments, and/or configurations that may be suitable for use include, but are not limited to, personal computers, server computers, handheld or laptop devices, multiprocessor systems, microprocessor-based systems, network personal computers (PCs), minicomputers, mainframe computers, embedded systems, distributed computing environments that include any of the above systems or devices, and the like.

[0053] Computer-executable instructions, such as program modules, being executed by a computer may be used. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Distributed computing environments may be used where tasks are performed by remote processing devices that are linked through a communications network or other data transmission medium. In a distributed computing environment, program modules and other data may be located in both local and remote computer storage media including memory storage devices.

[0054] With reference to Fig. 6, an exemplary system for implementing aspects described herein includes a computing device, such as computing device 600. In its most basic configuration, computing device 600 typically includes at least one processing unit 602 and memory 604. Depending on the exact configuration and type of computing device, memory 604 may be volatile (such as random access memory (RAM)), non-volatile (such as read-only

memory (ROM), flash memory, etc.), or some combination of the two. This most basic configuration is illustrated in Fig. 6 by dashed line 606.

[0055] Computing device 600 may have additional features/functionality. For example, computing device 600 may include additional storage (removable and/or non-removable) including, but not limited to, magnetic or optical disks or tape. Such additional storage is illustrated in Fig. 6 by removable storage 608 and non-removable storage 610.

[0056] Computing device 600 typically includes a variety of tangible computer readable media. Tangible computer readable media can be any available media that can be accessed by device 600 and includes both volatile and non-volatile media, removable and non-removable media.

[0057] Tangible computer storage media include volatile and non-volatile, and removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Memory 604, removable storage 608, and non-removable storage 610 are all examples of computer storage media. Tangible computer storage media include, but are not limited to, RAM, ROM, electrically erasable program read-only memory (EEPROM), flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computing device 600. Any such computer storage media may be part of computing device 600.

[0058] Computing device 600 may contain communications connection(s) 612 that allow the device to communicate with other devices. Computing device 600 may also have input device(s) 614 such as a keyboard, mouse, pen, voice input device, touch input device, etc.

Output device(s) 616 such as a display, speakers, printer, etc. may also be included. All these devices are well known in the art and need not be discussed at length here.

[0059] It should be understood that the various techniques described herein may be implemented in connection with hardware or software or, where appropriate, with a combination of both. Thus, the methods and apparatus of the presently disclosed subject matter, or certain aspects or portions thereof, may take the form of program code (i.e., instructions) embodied in tangible media, such as floppy diskettes, CD-ROMs, hard drives, or any other machine-readable storage medium wherein, when the program code is loaded into and executed by a machine, such as a computer, the machine becomes an apparatus for practicing the presently disclosed subject matter. In the case of program code execution on programmable computers, the computing device generally includes a processor, a storage medium readable by the processor (including volatile and non-volatile memory and/or storage elements), at least one input device, and at least one output device. One or more programs may implement or utilize the processes described in connection with the presently disclosed subject matter, e.g., through the use of an application programming interface (API), reusable controls, or the like. Such programs may be implemented in a high level procedural or object-oriented programming language to communicate with a computer system. However, the program(s) can be implemented in assembly or machine language, if desired. In any case, the language may be a compiled or interpreted language and it may be combined with hardware implementations.

[0060] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as

example forms of implementing the claims.

WHAT IS CLAIMED:

1. A method of providing remote access to at least one application program, comprising:
 - providing a server remote access program at a server computing device;
 - enumerating at least one Uniform Resource Locator (URL) that is associated with the at least one application program;
 - receiving a communication from a client at the at least one URL;
 - providing a screen scraping application at the server computing device to scrape screen data associated with the at least one application program associated with the at least one URL;*
 - generating presentation data from the screen data, the presentation data being generated in accordance with a selection of a display region of the at least one application program; and
 - communicating the presentation data to the client computing device using the server remote access program.
2. The method of claim 1, further comprising determining that the at least one application program is executing on the server computing device; and enumerating the URL in response to the determining.
3. The method of any of claims 1-2, further comprising enumerating a unique URL for each application program determined to be executing.
4. The method of claim 3, further providing access to the unique URL using one of a link, a menu time, or an icon.
5. The method of any of claims 1-4, further comprising:

receiving plural requests at plural URLs from the client computing device;
scraping screen data associated with plural applications associated with the plural URLs;
generating presentation data from the screen data; and
communicating the presentation data to the client computing device.

6. The method of any of claims 1-5, further comprising:
receiving requests from plural client computing devices at the at least one URL;
scraping screen data associated with the at least one application associated with the at least one URL;
generating presentation data from the screen data; and
communicating the presentation data to the plural client computing devices using the server remote access program.

7. The method of claim 6, further comprising providing collaboration among the plural client computing device, wherein the presentation data is synchronized among the plural client computing devices.

8. A method for providing remote access to an application program, comprising:
providing a server remote access program at a server computing device;
receiving a communication at the server computing device from a client computing device to access the application program;
providing a screen scraping application at the server computing device to scrape screen data associated with the application program;
generating presentation data from the screen data, the presentation data being generated in accordance with a selection of a display region of the application program;
updating a state model to contain a corresponding state of the application program;
and

communicating the presentation data and the state model to the client computing device using the server remote access program.

9. The method of claim 8, further comprising updating the state model in accordance with interactions at the client computing device.

10. The method of any of claims 8-9, further comprising receiving information about display characteristics of the client computing device.

11. The method of claim 10, further comprising scaling or cropping the screen data in accordance with the characteristics.

12. The method of any of claims 8-11, further comprising:

receiving a communication at the server computing device from the client computing device to access a second application program;

scraping second screen data associated with the second application program;

generating second presentation data from the second screen data;

updating the state model to contain a corresponding state of the second application program; and

communicating the second presentation data and the state model to the client computing device using the server remote access program.

13. The method of claim 12, further comprising combining the presentation data and the second presentation data into a single user interface at the client computing device.

14. The method of any of claims 8-13, further comprising
receiving a communication at the server computing device from plural client computing devices to access an application; and
communicating the presentation data and the state model to the plural client computing devices.

15. The method of claim 14, further comprising enabling collaboration among the plural client computing devices through the state model.

16. A method for providing non-invasive remote access to an application program, comprising:
providing a server remote access program at a server computing device;
providing a screen scraping application at the server computing device that monitors a display interface generated by the application program for changes to the display interface;
scraping screen data associated with the display interface;
generating presentation data from the screen data, the presentation data being generated in accordance with a selection of a display region of the application program;
receiving a request at the server computing device to access the application program;
updating a state model containing a corresponding state of the application program, wherein the corresponding state includes at least one logical element containing information about the display interface of the application program; and
communicating the presentation data and the state model using the server remote access program in response to the request.

17. The method of claim 16, further comprising:

receiving the request from client remote access program executing on a client computing device that is logically connected to the server remote access program.

18. The method of any of claims 16-17, further comprising:

receiving a second request from a second client remote access program executing on a second client computing device that is logically connected to the server remote access program;

synchronizing, at the server computing device, the presentation and the state model between the client computing device and the second client computing device; and

communicating synchronized presentation data and a synchronized state model to the client computing device and the second client computing device such that both the client computing device and the second client computing device display the same information.

19. The method of any of claims 16-18, further comprising:

obtaining characteristics of a display of the client computing device from the state model; and

modifying the presentation data in accordance with the characteristics.

20. The method of any of claims 16-19, further comprising:

providing remote access to a second application program;

scraping second screen data associated with the second application program to generate second presentation data from the second screen data;

updating the state model to contain a corresponding state of the second application program; and

combining the presentation data and the second presentation data into a single user interface using the state model.

21. A method of providing remote access to a server-based application program, comprising:

providing a remote access program at a server computing device;

determining application programs that are executing on the server computing device;

enumerating a Uniform Resource Locator (URL) that is associated with each application program determined by the server computing device to be executing;

providing access to the URL of each application program determined by the server computing device to be executing;

receiving a selection from a client computing device at the URL of a selected application program; and

providing remote access to the application program at the URL.

22. The method of claim 21, further comprising:

scraping screen data associated with the selected application program;

generating presentation data from the screen data; and

communicating the presentation data to the client computing device.

23. The method of any of claims 21-22, further comprising:

receiving requests at plural URLs that each are associated with plural application programs;

scraping screen data associated with the plural applications;

generating presentation data from the screen data associated with plural programs; and

communicating the presentation data to the client computing device.

24. The method of any of claims 21-23, further comprising providing collaboration among plural client computing devices.

25. The method of any of claims 21-24, further comprising:

receiving a selection from the plural client computing device at the URL of the selected application program; and

providing remote access to the selected application program at the URL.

26. The method of claim 25, further comprising providing a synchronized view of presentation data of the selected application program to the plural clients.