An Open Source Collaboration Framework with Inter-Desktop Movable Windows and Remote Multicursor Desktops

Robert Matzinger
University of Applied Science Burgenland GmbH
Eisenstadt, Austria
Email: robert.matzinger@fh-burgenland.at

***Abstract*—Many current tools for conferencing and synchronous collaborative work suffer from being limited to certain types of documents or to specific, often proprietary collaboration software. In this paper we introduce a system for assisting synchronous collaborative work with arbitrary software by enabling to move any window on a typical computer desktop to other desktops in the system, may it be another computer in the company or a virtual desktop being run on a shared desktop-server. We equip these desktop-servers with multiuser multicursor capabilities, such that groups of users can join to work concurrently on a shared desktop, both locally and from a distance. As any kind of window can be moved from desktop to desktop, the choice of software that can be used for collaborative tasks is unlimited. Our system is built from reliable open source components only and is freely available for everyone. In this paper we present the current version of our system that works for arbitrary Unix-based computers and demonstrate the feasibility of our approach. Thus we show how collaboration suites, terminal systems, desktop sharing and multicursor desktops can be generalized into a consistent system, where the window on the screen is the users essential movable item that is used for remote work and collaboration.**

Keywords*-*Real-time collaboration, technologies for collaborative work, open source, interface sharing.

I. Introduction

In [1], we augment seldom-used multiuser multicursor desktop technology [2] [3] [4] [5] by a remoteness-mechanism for allowing many users to connect to virtual remote multiuser desktops for accomplishing collaborative work. By combining well-tested open source components we achieve a system that supports small work groups in working together synchronously from afar. This appeares to be beneficial for e.g., an interactive meeting in a project design phase, but it can be useful in many other work and teaching situations as well. This approach models the natural "come together" on a shared conference table, where workers join for collaborative work on (paper) documents layed out on the table. However the data exchange with the shared desktop was limited to using shared data with some software that must be installed on the shared desktop server, thus having data sharing restricted to files or documents that must be somehow transferred to or accessed from the shared desktop.

With the current work we combine our remote shared desktop solution with a solution that allows to move arbitrary windows among different computer desktops without interrupting the running application. The target desktop can be just another desktop computer in the company, but can also be a (virtual) shared multiuser desktop where collaborative work can be carried out.

Such an approach substantially broadens the access to non-local information at a particular computer screen. We can make the freely movable window of a running application be the essential item of collaboration. And we can overcome the borders of the current desktop and create something like a "shared desktop space" on which project-wide or company-wide collaboration can take place, both locally and from afar.

In this paper we describe the architecture and the current version of our system. We start in Section II with comparing our work with existing collaboration software. In Section III we sketch various usage scenarios of our system. Section IV lists the components and describes the concepts of our systems architecture. In Section V we describe the functionality of our system from the users point of view with respect to the screenshots in Figure 4. Section VI and Section VII report on the hardware and network requirements and the availability of our software, followed by a conclusion and a proposal for future work in Section VIII.

II. State of the Art and Benefits

Most current systems for remote and (synchronous) collaborative work fall into the following categories or can be considered a combination of: ◦) e-Collaboration platforms like Alfresco [6], e-groupware [7], and Zimbra [8] basically allow for coordinated editing and sharing of documents and information. ◦) Software for collaborative editing of single documents like Gobby [9], Ace [10] [11], and Abiword [12] allow for synchronous collaborative editing, but only for limited types of documents. ◦) Terminal Systems like Citrix [13] and X2go [14] allow for accessing remote services (i.e., running software) that are run on a dedicated central server. ◦) Desktop sharing systems like the variants of VNC [15] [16] [17] or those based on RDP [18] allow to share the entire desktop with other remote users or to perform work on a distant computer. ◦) Web onferencing software like gotomeeting [19] and Openmeetings [20] usually combines some of these features, but still needs special (often proprietary) software to be installed, is bound to web interfaces or does not allow for the natural concurrent work of many users on the same desktop, like it is possible with the multicursor concept.

With our system we try to generalize these approaches to a consistent system of a shared desktop space among group members, that is tightly integrated into a standard user system, such that moving windows and concurrent and collaborative work of many users on multicursor group-work desktops (that can be accessed locally or from afar) gets a trivial part of everydays work routine.

For building blocks and communication between them we rely on open source components and on open protocols like SSH, X11, RDP, such that our system can be extended or integrated with other systems as needed. Our system is built for arbitary X11-based software and desktops (i.e., Unix-based systems). But with Cygwin [21] and X11 for MacOSX [22], it can be accessed from Windows or MacOSX powered computers too.

III. Usage Scenarios

Inter-desktop window movement has numerous applications by itself e.g., when visiting a colleague in another room and pulling over a window that is currently shown on the own computer screen for discussion, or when a teacher pulls a well-prepared presentation of some running software to a lecture hall's presentation computer, or when a system administrator delegates difficult problems to a specialist by pushing the application window of the running supervision software to the screen of the responsible specialist. Pulling windows may also be helpful for simply changing the workplace e.g., from company to home, or for letting an application do background work and reattach to it as soon as this is of interest again, independent from the location of the user.

Multiuser multicursor desktops could be installed in meeting rooms or at presentation computers. Meeting attendees could pull the relevant application windows to the shared desktop to work on it with the group collaboratively, like on a shared whiteboard. The same principle can be applied for student groups in a computer lab, which practice team work or get assistance from an instructor.

All these work situations can also be carried out from a distance by using remote shared multiuser desktops. Users could connect to the shared desktop from a remote computer, such that the view of the shared desktop is propagated to all participating users in real-time, thus being simultaneously visible to the participants.

All in all we give the application window on a computer desktop a new role, namely being the essential item that can be carried around, sent and received, like it was the "document" or the sheet of paper in pre-computer times.

Clearly, our system can freely be combined with any other collaboration solution, as this would most probably be just another software showing up in some window on the users' screens. In practice remote multicursor desktops will have to be accompanied by voice-chat systems to accomplish about the same group work as working locally.

IV. System Architecture

In this section we describe the main building blocks of our system and sketch how they are combined to a system that achieves the different functionalities we proposed.

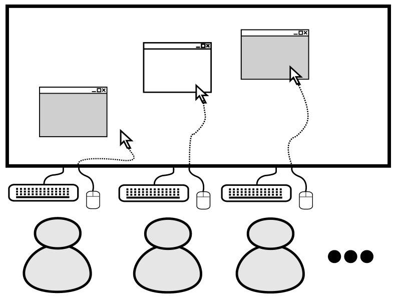

Figure 1: Multiuser multicursor desktops.

A. System Components

Our system is built from well-known open source components: SSH (Secure Shell) [23] [24] provides secure authentication mechanisms and secure encrypted data traffic and is nowadays the most widely used workhorse of secure communication.

VNC (Virtual Network Computing) [17] is used for graphical desktop sharing to communicate the contents of a remote computer screen to another computer. It is based on the platform-independent RFB protocol [25]. VNC viewers can connect to VNC servers running on different operating systems.

Our prototype system is implemented with Debian GNU/Linux [26] but can be used with any desktop system based on an implementation of the X windowing system [27] [28] like X.org [29], which is the central building block of most current Linux desktops.

Additionally we use xpra [30] as X11 proxy server. The xpra server can serve as a virtual X server for the application, while the xpra client can connect to it to fetch the interface of the application and make it visible on a computer screen (see Section IV-B). xpra can apply various degrees of compression to the traffic of user-interface information, making it possible to balance responsiveness and quality of the visible view according to the available network bandwidth.

Finally, the Multicursor Window Manager [2] [3] [4] [5] is an X11 window manager based on IceWM [31] that allows small groups of users to use a shared desktop concurrently by assigning a mouse cursor plus keyboard and a distinct input focus on the shared desktop to each participating user. By a simple click, a users can assign a window to his/her input focus to work on the window, while other users may utilize other windows on the desktop at the same time. This allows for simultaneous work of different users on different windows on the same desktop. Still, for the single window, no adaption of the application's user interface is required. Collaboration on a single window is achieved naturally by quickly switching input focus as necessary (see Figure 1).

A modified version of x2x [32] serves for sending mouse and keyboard events to the multi-cursor desktop from remote.

 ISBN: 978-1-61208-143-4

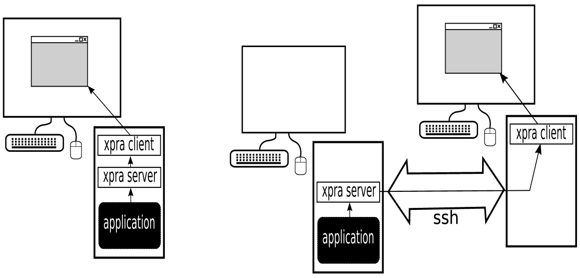

Figure 2: Pulling and pushing windows with xpra.

B. Pushing and Pulling Windows

For enabling inter-desktop window movement, we can make use of an ancient design principle of X11, namely the separation of applications from their user interfaces (i.e., the windows on the desktop), where the X11 protocol [27] [28] is the protocol for application-interface communication.

We add an xpra client-server pair to the line of X11-communication between application and interface (window) (see Figure 2, left), which is invisible for the user and has virtually no impact on the percepted performance. In case we want to push (or pull) the application window to another desktop, the local xpra client is terminated and another xpra client is started at the target machine, which then connects to the running xpra server to fetch the user interface and pass it on to its local X server, which takes care of displaying it on the local screen (see Figure 2, right). Meanwhile the xpra server serves as an X server for the application and as it is not interrupted by the switch of the xpra clients, the application can carry on undisturbed. Note that the window indeed is moved from one desktop to the other as it disappears from the original one (see Figure 4). Still, the target desktop itself can possibly be a (remote) shared desktop beeing equiped for group work, see Section IV-E.

C. Connection Tunneling

We tunnel all connections for running a remote desktop-session (the xpra connection, the mouse and keyboard events communicated via x2x, and the VNC connection that propagates the view of a remote desktop, see also Section IV-E) through a single SSH connection. As tunneling can be performed in both directions, it does not matter which computer was the originator of the SSH connection. Because of the reliable encryption of SSH, our system is capable to communicate safely, even behind firewalls and NATs. Note that as an alternative a VPN-solution like e.g., OpenVPN [33] could take the role of SSH accordingly.

D. Remote Multicursor Desktops

In the center of a remote multicursor desktop system we run a (virtual) X11 desktop with a multicursor window manager and with an attached VNC server on a server computer. Users can get a real-time view of the multicursor desktop by using an appropriate VNC client that connects to the VNC server (see the dashed lines in Figure 3). Thus, any action taken by one participant of the shared desktop is made visible to all participants simultaneously and synchronously. To allow participation on the work on the shared desktop, mouse and keyboard events are sent from the users to the remote multicursor desktop via x2x connections, where each mouse/keyboard is assigned to one of the mouse cursors and input focuses of the multicursor system (dotted lines in Figure 3). With a mouse click, users can assign their input focus to a window on the shared desktop to work on this window, while other work can be performed by other users on other windows independently, but visible to all participants.

E. Putting it together: Pushing Windows to Remote Multicursor Desktops

To being able to send local windows to a shared multicursor desktop for groupwork, we combine connection tunneling, desktop sharing and the pushing of windows via xpra as shown in Figure 3: For accessing the remote shared multicursor desktop, a user first establishes a single SSH connection to the shared desktop-server. Then a VNC-client is started locally to bring a real-time view of the remote desktop to the user, as described in Section IV-D. Subsequently, a locally run x2x-client is used to attach to one of the cursors of the multicursor desktop, enabling the user to do work on the remote shared desktop. Finally, for moving a window from the users local desktop to the remote desktop, a window pushing step with xpra as described in Section IV-B is performed.

By this combination we achieve the required functionality of a simultaneously visible remote shared multicursor desktop, to which each participant can push application windows.

Note that remote shared multicursor desktops are just the most complicated scenario that can be handled with our system. Simpler cases like window-movement to other ordinary desktops can be handled accordingly. In safe environments like computer labs, it is even possible to omit the SSH connection and to allow direct connections between the participating computers.

F. Security and Access Control

As mentioned in Section IV-C, all relevant connections (the VNC-connection, the x2x-connection, and the xpra-connection) are tunneled through a single SSH connection to guarantee encrypted safe communication. Via the well-tested SSH login-mechanism (password-protected or public-key-protected), security and access control can safely be handled by the SSH daemon. All the other daemons (xpra server, VNC server, X server) are restricted to local connections, which can only be accessed from the local SSH subsystem, thus there is no connection possible, before a proper SSH connection between the participating computers is established. This approach inhibits improper connections and enables various access control schemes that can be configured with the SSH configuration.

V. Screenshots

In Figure 4 (left) a locally running application (the spreadsheet) with a shared remote multicursor desktop in

 ISBN: 978-1-61208-143-4

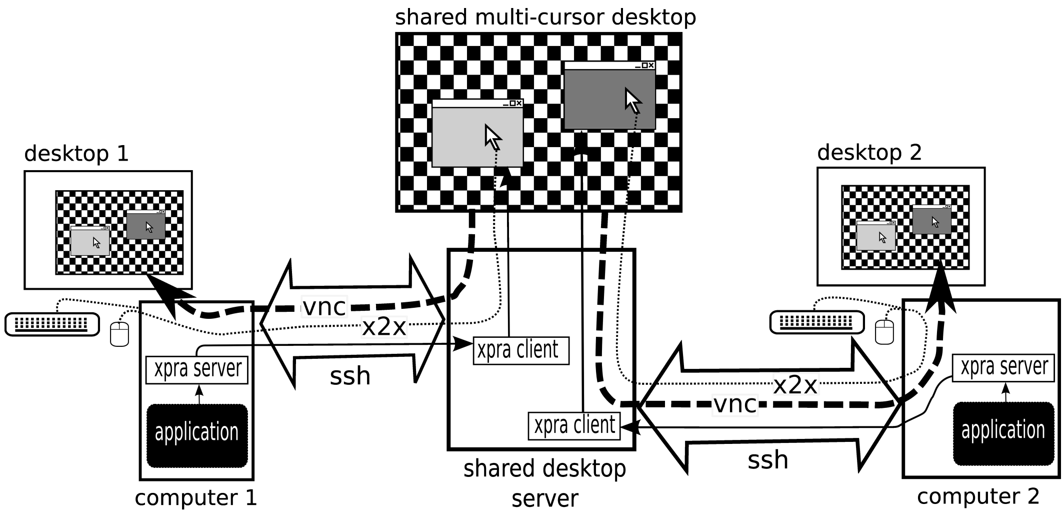

Figure 3: System architecture of a remote shared multiuser multicursor desktop with two participating users.

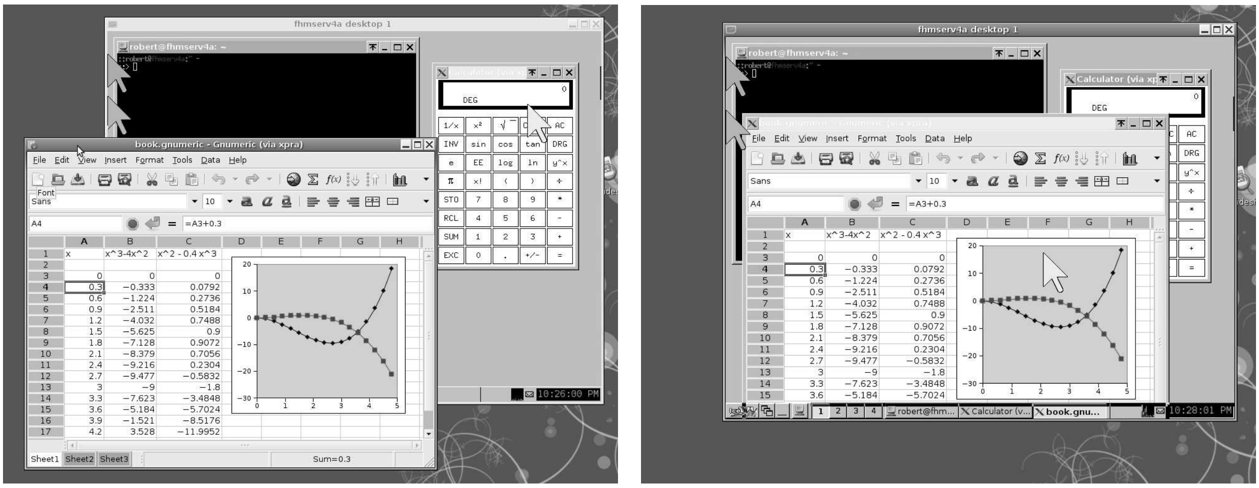

Figure 4: Screenshots of a remote multicursor desktop in use.

the background can be seen. On the remote desktop, 3 mouse pointers are visible, indicating 3 users currently attending the shared desktop. The different input focuses are indicated by different colors of the title-bars of the windows on the multicursor desktop. In Figure 4 (right) the spreadsheet application was pushed to the remote multicursor desktop to be worked upon by the participants collaboratively. At any time, the spreadsheet window can be pulled back to the local desktop without closing the application e.g., for completion steps. Note that in practice the shared desktop is configured to be substantially larger to provide enough screen space for collaborative work. It is even possible to add a dedicated monitor for groupwork on shared desktops to a workstation.

VI. Hardware and Network Requirements

As the xpra server and xpra client consume very little system ressources, the requirements for the client computer are essentially based on the software, that is planned to be used. For hosting a single shared desktop, a standard PC in the 1GHz/512MB/30GB-class is sufficient. A more powerful computer can consequently host more than one shared desktop.

As the xpra connections and the VNC connections can adapt to different network bandwidths the system can adapt itself to quite different network conditions by decreasing the image quality of the transmitted user interface, if necessary. Our system was tested to work with reasonable internet connections including ADSL, XDSL, modem and (with slightly degraded performance) even with mobile UMTS connections, thus we conclude that a bandwidth of 5-10 MBit/s per user of a multiuser desktop is sufficient for smooth operation. As mentioned in Section IV-C, firewalls and NATs pose no difficulty to the connectivity of our system.

VII. Availability

The current version of our system runs on a couple of Debian Linux-based demo computers, including a shared desktop server for remote multiuser multicursor desktops. Our system consists of the software and the system configuration necessary to combine the components as described in Section IV. It is currently packed into packages for easy installation on Debian Linux-based systems. We are also working on a proper GUI and on the close integration of our system into the user interface of desktop envirenments (e.g., GNOME [35]) as a supplement to the currently existing command line interface, and we are eager to make our system available to some production environment.

VIII. Conclusion

By combining multicursor desktops with VNC-based desktop sharing, we achieved a system for synchronous work of small work groups [1]. In this paper we enhance this system with an xpra-based feature for moving arbitrary desktop windows between different desktops, including the (virtual or real) shared desktops. This makes our solution applicable in many situations of everyday computer work, like e.g., team meetings, creative design phases, e-learning and system administration, both locally and from afar.

Generally, we declare the “window on the screen” to be the essential movable data unit, which can be taken along to other computers or workgroup sessions, enabling to collaboratively work with arbitrary software in a seamless manner. Thus our system meets the basic needs of coming together for collaborative work and of carrying the relevant information from one place to another.

By this design we could overcome limitations of being restricted to specific software, data formats or platforms and solve many problems at once, that are normally addressed by very different systems like e-collaboration platforms, collaborative editors, desktop sharing software and web conferencing tools.

Still our system can be freely combined with any other (collaboration) software suitable for a specific task, as this would be just another application with some GUI window on the desktop.

As we uniformly tunnel all data traffic through SSH connections, we can rely on the safe encryption and access control of SSH and can implement various access schemes with an appropriate SSH configuration. As SSH is so widely used and well developed, communication encryption and access control can be considered adequately safe even for critical operation. Additionally, the connection tunneling makes our system capable of working with a wide range of network configurations including NATs and firewalls.

Our system is designed to work with most Unix-based operating systems and is available as a running prototype to prove feasibility and usefulness of our approach. As we built our system from open source components only, it is freely available for everyone. In practice it has to be accompanied by a voice-chat system to support a wide range of collaborative work tasks even from a distance. A tight integration into the users desktop and a comfortable user interface is under active development.

Our system can be seen as a kind of virtualization of GUIs, which can be transferred freely among desktops, where we use X11 as an universal and flexible protocol for communication between an application and its GUI. Having achieved this, we could address the virtualization of the underlying application independently of its GUI. This is the starting point of our future plans to explore possibilities of application virtualization so that not only the application interface can be moved between computers, but also the running application itself, giving work groups a heterogeneous network of computers to flexibly assist location-independent collaborative work in many ways.

Acknowledgement

This study was partially funded by FFG/Österreichische Forschungsförderungsgesellschaft mbH in grant No. 821111 ”eCollaboration 2.0: Collaboration Tools und Social Media für Teamarbeit in KMUs”.

References

[1] R. Matzinger, “Remote on-screen support and cooperation with open source components in e-learning,” in Pannonia Research Report: ausgezeichnete angewandte Forschungsergebnisse des Awards der Fachhochschulstudiengänge Burgenland. Eisenstadt, Austria: E. Weber, 2008, ISBN 978-3-85253-407-7.

[2] G. Wallace, O. J. Anshus, P. Bi, H. Chen, Y. Chen, D. Clark, P. Cook, A. Finkelstein, T. Funkhouser, A. Gupta, M. Hibbs, K. Li, Z. Liu, R. Samanta, R. Sukthankar, and O. Troyanskaya, “Tools and applications for large-scale display walls,” IEEE Comput. Graph. Appl., vol. 25, pp. 24–33, July 2005.

[3] G. Wallace, P. Bi, K. Li, and O. Anshus, “A multicursor X window manager supporting control room collaboration,” Dept. of Computer Science, Princeton University, Tech. Rep. TR-707-04, 2004, retrieved on 2011-04-15. [Online]. Available: http://www.cs.princeton.edu/omnimedia/papers/multicursor.pdf

[4] K. Li, H. Chen, Y. Chen, D. W. Clark, P. Cook, S. Damianakis, G. Essl, A. Finkelstein, T. Funkhouser, T. Housel, A. Klein, Z. Liu, E. Praun, R. Samanta, B. Shedd, J. P. Singh, G. Tzanetakis, and J. Zheng, “Building and using a scalable display wall system,” IEEE Comput. Graph. Appl., vol. 20, pp. 29–37, July 2000.

[5] G. Wallace and K. Li, “Virtually shared displays and user input devices,” in 2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference. Berkeley, CA, USA: USENIX Association, 2007, pp. 31:1–31:6.

[6] Open source enterprise content management system by alfresco. Retrieved on 2011-04-15. [Online]. Available: http://www.alfresco.com

[7] EGroupware - online groupware, CRM. Retrieved on 2011-04-15. [Online]. Available: http://www.egroupware.org

[8] Zimbra offers open source email server software and shared calendar for linux and the mac. Retrieved on 2011-04-15. [Online]. Available: http://www.zimbra.com

[9] Gobby, a collaborative text editor. Retrieved on 2011-04-15. [Online]. Available: http://gobby.0x539.de

[10] Ace - a collaborative editor. Retrieved on 2011-04-15. [Online]. Available: http://www.sourceforge.net/projects/ace

[11] D. Chen, "A survey of real-time collaborative editing systems," in Proceedings of the Eighth International Workshop on Collaborative Editing Systems. ACM CSCW, 2006.

[12] AbiWord - word processing for everyone. Retrieved on 2011-04-15. [Online]. Available: http://www.abisource.com

[13] Citrix systems - virtualization, networking and cloud. Retrieved on 2011-04-15. [Online]. Available: http://www.citrix.com

[14] x2go - server based computing. Retrieved on 2011-04-15. [Online]. Available: http://www.x2go.org

[15] RealVNC - VNC remote control software. Retrieved on 2011-04-15. [Online]. Available: http://www.realvnc.com

[16] TightVNC: VNC-compatible free remote control / remote desktop software. Retrieved on 2011-04-15. [Online]. Available: http://www.tightvnc.com

[17] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper, "Virtual network computing," IEEE Internet Computing, vol. 2, pp. 33–38, January 1998.

[18] Microsoft Corp., Understanding the Remote Desktop Protocol (RDP), March 2007, revision 2.3, Retrieved on 2011-04-15. [Online]. Available: http://support.microsoft.com/?scid=kb%3Ben-us%3B186607

[19] Web conferencing - gotomeeting. Retrieved on 2011-04-15. [Online]. Available: http://www.gotomeeting.com

[20] openmeetings - open-source web-conferencing. Retrieved on 2011-04-15. [Online]. Available: http://code.google.com/p/openmeetings

[21] Cygwin/X - X Windows on Windows! Retrieved on 2011-04-15. [Online]. Available: http://x.cygwin.com/

[22] X11 for Mac OS X. Retrieved on 2011-04-15. [Online]. Available: http://developer.apple.com/opensource/tools/x11.html

[23] D. J. Barrett, R. E. Silverman, and R. G. Byrnes, SSH: The Secure Shell (The Definitive Guide). O'Reilly, 2005, ISBN 0-596-00895-3.

[24] M. Stahnke and J. Traenkenschuh, Pro Openssh (Pro). Berkely, CA, USA: Apress, 2005.

[25] T. Richardson, The RFB Protocol, version 3.8, (last updated 18 june 2007) ed., RealVNC Ltd, (formerly of Olivetti Research Ltd/ATnT Labs Cambridge), 2007.

[26] Debian - the universal operating system. Retrieved on 2011-04-15. [Online]. Available: http://www.debian.org

[27] R. W. Scheifler and J. Gettys, "The X window system," ACM Trans. Graph., vol. 5, pp. 79–109, April 1986.

[28] ——, X Window system: core and extension protocols. Acton, MA, USA: Digital Equipment Corp., 1997.

[29] X.org foundation home. Retrieved on 2011-04-15. [Online]. Available: http://www.x.org

[30] A window manager library and "screen for X11". Retrieved on 2011-04-15. [Online]. Available: http://code.google.com/p/partiwm/wiki/xpra

[31] The IceWM window manager. Retrieved on 2011-04-15. [Online]. Available: http://www.icewm.org

[32] dottedmag/x2x - GitHub. Retrieved on 2011-04-15. [Online]. Available: https://github.com/dottedmag/x2x

[33] OpenVPN - Open Source VPN. Retrieved on 2011-04-15. [Online]. Available: http://www.openvpn.net

[34] N. Kock and J. Nosek, "Expanding the boundaries of e-collaboration," IEEE Transactions on Professional Communication, vol. 48, 2005.

[35] The GNOME project. Retrieved on 2011-04-15. [Online]. Available: http://www.gnome.org