

ネットワークの動的変化に対応する遠隔デバイス管理機構

尾崎 亮太[†] 日高 宗一郎^{†,††}
児玉 和也^{††} 丸山 勝巳^{†,††}

計算機に接続されたデバイスをネットワーク上の別の計算機，例えば PDA などのモバイル機器から利用可能とするシステムの構築技術に注目が集まっている．本研究では動的なネットワークの変化やデバイスの移動などハードウェア環境の変化に柔軟に対応できるアーキテクチャを提案する．アプリケーションプログラムへのインタフェースはデバイスファイルで実現し，計算機間の通信はユーザレベルプログラムで行なう．すべてをカーネルレベルで実現する方法に比べ柔軟であり，動的なネットワークの変化やデバイスの移動などに容易に対応できる．また本稿では本システムの設計と実装およびネットワークの変化に対応するための機能拡張について述べる．

Network Extended Device Management System Supporting Location Transparency

RYOTA OZAKI,[†] SOICHIRO HIDAKA,^{†,††} KAZUYA KODAMA^{††}
and KATSUMI MARUYAMA^{†,††}

Technologies which enable access to human-machine interaction support devices such as mice, speakers and displays from mobile computers over network are increasingly drawing attentions. We propose an infrastructure system architecture makes it easy to adapt to dynamic changes of hardware environment such as network location and a device movement between machines. Our system provides device file interface to application programs as to access remote devices, and communications with other machines are implemented at user-level. Our architecture is more flexible than all-in-kernel approach. In this paper we describe a design and an implementation of our system and methods to extend our system to cope with dynamic changes of hardware environment.

1. はじめに

人間と計算機のインタラクションを支援するデバイスを，モバイル機器などネットワーク上の別の計算機からも利用可能とするシステムの構築技術に注目が集まっている^{1)~3)}．我々は，このようなネットワーク透過なデバイスアクセスを提供する基盤ソフトウェアの実現を目的とし研究を進めている⁴⁾ (図 1)．本研究では動的なネットワークの変化やデバイスの移動などハードウェア環境の変化に柔軟に対応できるアーキテクチャを提案する．本アーキテクチャでは既存のインタフェースの維持と，物理的なネットワーク変化をアプリケーションプログラムに隠蔽しデバイスアクセス

を継続できるような機能拡張性を両立できる．

本システムでは，従来のファイルシステムインタフェースでアプリケーションプログラムが遠隔デバイスにアクセスすることを許し，遠隔計算機間の通信は柔軟なユーザレベルのプログラムで実現する．UNIX系 OS はデバイスへアクセスするためのインタフェースをデバイスファイル，一般的には/dev/以下のファイル群として提供している．本システムでは遠隔計算機のデバイスファイル群をローカル計算機のファイルシステムへバインディングする．これによりアプリケーションプログラムはローカル計算機に存在するように遠隔デバイスへアクセスすることができる．そのため既存のプログラムは新たなインタフェースに対応する必要はなく，独自インタフェースを持つミドルウェアに比べ適用できる範囲は広い．インタフェースはファイルシステムつまりカーネルレベルで実現する一方，計算機間の通信はユーザレベルプログラムで行なう．通信処理をカーネルレベルで実現する構成に

[†] 総合研究大学院大学 情報学専攻, Department of Informatics, The Graduate University for Advanced Studies.

^{††} 情報・システム研究機構 国立情報学研究所, National Institute of Informatics, Research Organization of Information and Systems.

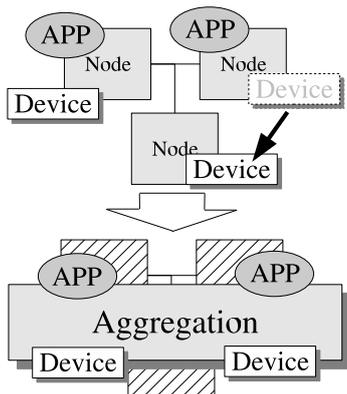


図 1 遠隔デバイス管理システム
Fig. 1 Overview of our goal

比柔軟であり、動的なネットワークの変化やデバイスの移動などに容易に対応できる。

また本稿ではハードウェア環境の変化をアプリケーションプログラムから隠蔽する手法について議論する。近年のネットワーク接続の無線化や着脱可能デバイスの増加により、動的なネットワークの変化や計算機間のデバイスの移動が起きることが多くなってきた。ユーザに利便性の高いサービスを提供するためにはハードウェア環境の変化を可能な限り隠蔽することが望まれる。このようなサービスを提供するためにはネットワークの動的変化を隠蔽するインタフェースを提供する必要がある。また着脱可能デバイスを取り外して別の計算機に再接続してもデバイスアクセスが継続できるようなシステムが望まれる。本稿ではこのようなシステムの設計方針について述べる。

以下、2章では遠隔デバイスアクセス機構の実現について述べる。遠隔デバイスアクセス技術や分散ファイルシステム技術の関連研究について述べ、その後システムの設計や実装について述べる。3章では実現したシステムを動的なネットワークの変化やデバイスの移動などへ対応させる方法について議論する。最後に4章でまとめと今後の課題を示す。

2. 遠隔デバイスアクセス機構

2.1 関連研究

遠隔デバイスアクセス技術としては Plan9⁵⁾ や GnomeVFS が挙げられる。

Plan9は資源をすべてファイルシステムとして抽象化している。また計算機間で統一なファイルシステム階層をユーザに提供している。それぞれの計算機に

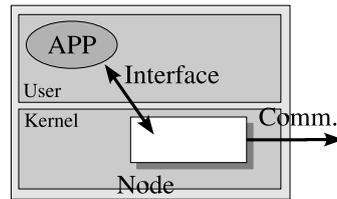


図 2 カーネル方式
Fig. 2 All-in-kernel approach

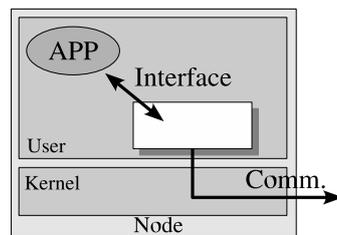


図 3 ユーザ方式
Fig. 3 User-level approach

存在するファイルシステムは一つの大きなファイルシステム階層の一部である。また別の計算機に存在するファイルシステムの一部をローカル計算機のファイルシステムの一部としてインポートすることもできる。一般的な分散ファイルシステムと異なり Plan9 ではデバイスファイルもインポートすることができるため、ユーザはデバイスファイルを通して遠隔計算機のデバイスにアクセスすることができる。しかし Plan9 は通信も含めた機能をすべてカーネル内で実現しているため柔軟性に欠ける。例えばネットワークの動的変化への対応など新たな機能を追加するためにはカーネルの変更やシステムの再起動などを必要とする。

GnomeVFS は OS が提供するファイルシステムとは異なる独自の仮想ファイルシステムレイヤを構築し物理デバイスの種類や位置を隠蔽している。システムはユーザレベルのプログラムやライブラリで構成されている。インタフェースは POSIX ファイルシステムインタフェースと酷似しているが、既存のアプリケーションがそのまま利用できるというわけでない。例えば POSIX の open 関数と同じように振舞う `gnome_vfs_open` 関数は独自のファイルディスクリプタを返却する。

このように既存のシステムは柔軟性の欠如や既存のアプリケーションプログラムへ改変を必要とするなどの問題が存在する。以降では Plan9 のようにカーネル内でシステムを実現する方式をカーネル方式(図2)、GnomeVFS のようにユーザレベルでシステムを実現

GNOME Resources, <http://www.gnome.org/>

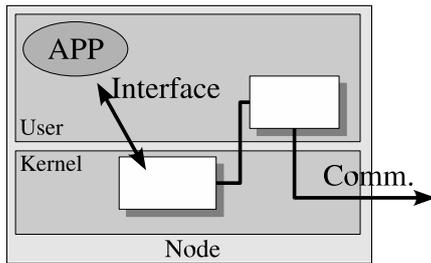


図 4 本方式 (ハイブリッド方式)
Fig. 4 Our approach (Hybrid approach)

する方式をユーザ方式 (図 3) と呼ぶ。

2.2 設 計

本研究ではユーザレベルのプログラムとカーネルの両方を用いて遠隔デバイスアクセスサービスを実現するハイブリッド方式を採用する (図 4)。本方式は図 4 のように 2 つのサブシステムに分かれて、一方がファイルシステムインタフェースをアプリケーションプログラムへ提供し、もう一方が遠隔計算機と通信を行なう。これにより従来と同様のデバイスファイルによるデバイスアクセスサービスを提供しつつ、遠隔計算機間の通信を容易に機能拡張できる。

このようにカーネル方式とユーザ方式の両方の利点を得る手法は他にも存在する。それはユーザ方式において動的リンクライブラリを活用する方式である。動的リンクライブラリのオーバーライド機能を用いると、アプリケーションプログラムを書き換えることなく、関数呼び出しを横取りし遠隔計算機へ要求を転送することができる。ActiveFiles³⁾ や ORFA⁶⁾ はこの機能を利用して、遠隔計算機にあるファイルへアクセスすることを可能としている。アプリケーションプログラムはローカルのファイルにアクセスするように遠隔計算機のファイルにアクセスできる。しかし動的リンクライブラリの機能を用いる方法には、静的リンクで生成されたプログラムでは使うことができない、他のプリロード機能を用いるプログラムと同居が難しいなどの問題がある⁷⁾。

2.3 デバイスファイルのインポート

通常のファイルシステムと同様にあらかじめ遠隔計算機のデバイスファイル群をローカルにマウントする必要がある。本システムではシステムに参加している全計算機のデバイスファイル群が自動的にローカルのファイルシステムへマウントされる (図 5)。遠隔計算機毎のデバイスファイル群はそれぞれ対応する IP アドレスや MAC アドレスの名前を持つディレクトリ以下に生成される。図 5 では 192.168.0.22 の IP アドレスを持つ Node B のデバイスファイル

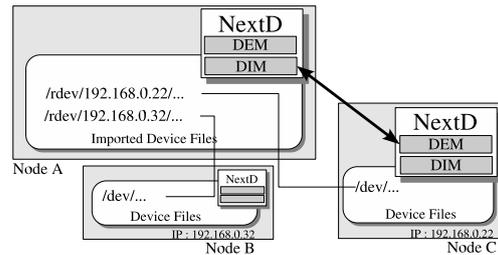


図 5 デバイスファイルのインポート
Fig. 5 Architecture of importing device files

群は /rdev/192.168.0.22/ ディレクトリ以下に生成される。

特定のネットワークポートを開き外部計算機からの要求を受け付ける常駐プロセス (以下 *NextD* : *Network EXTended Device manager*) を起動させることで遠隔デバイスアクセスサービスを利用することができる。NextD には DEM (Device file Export Module) と DIM (Device file Import Module) と呼ばれるデバイスファイル群の公開と取込みを行なうサブモジュールで構成される。DEM はローカル計算機のデバイスファイル群を遠隔計算機に公開する。DIM は遠隔計算機のデバイスファイル群をローカル計算機に再現する。

NextD は NFS などの分散ファイルシステムと同様のメカニズムで動作する。例えばアプリケーションプログラムが /rdev/<IP address>/ ディレクトリを走査する (opendir(2) や readdir(2) などと呼ぶ) と DIM が <IP address> に対応する遠隔計算機の DEM へ要求を送る。DEM はローカル計算機に存在するデバイスファイル群の情報を DIM へ返却する。DIM はその情報を元に /rdev/<IP address>/ ディレクトリへのエントリを生成する。分散ファイルシステムと同様 DIM と DEM が行なう処理はアプリケーションプログラムからは隠蔽されている。

/rdev/ のような特殊な新たなパス名を導入しても本システムで既存のアプリケーションプログラムを動作させることは可能である。シンボリックリンクを張ることでデバイスファイルを固定的に決められたアプリケーションプログラムにも対応できる。/rdev/ が通常と代わらないファイルシステムとして実現されているためこのようなことが可能となる。

2.4 ファイルアクセス

DIM が取り込んだデバイスファイル群へのアクセスは通常のファイルシステムと同様である。アプリケーションプログラムは POSIX の open, close, read, write 等の関数が利用できる。取り込んだデバイスファイ

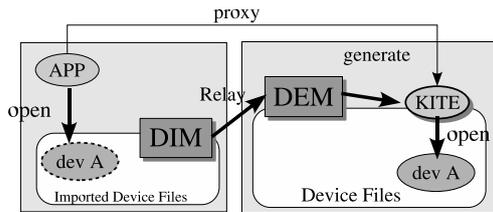


図 6 ファイル open

Fig. 6 Processing of open a remote device file

ルがアプリケーションプログラムによって open される毎にデバイスファイルの実体が存在する計算機側で DEM は新たなプロセス (以下 KITE) を生成する (図 6)。遠隔計算機のデバイスファイルは KITE によって通常のファイルシステムの作法通りに扱われる。つまり OS から見るとローカルに存在するプロセスの一つがデバイスファイルを open したようにしかみえない。以降のアプリケーションプログラムのファイル操作は遠隔計算機に転送され KITE がファイル操作を肩代りする。そして処理の結果がアプリケーションプログラムへ返却される。アプリケーションプログラムがデバイスファイルを close すると KITE は破棄される。

2.5 アクセス制御と資源排他

図 5 では /dev/ディレクトリを全て公開しているが ACL (Access Control List) でデバイスファイルごとに細かな制御が可能である。あるデバイスファイルを公開をしないように指定した場合、インポートする側にはそのファイルの存在は隠蔽される。

遠隔計算機のデバイスファイルを open するときには遠隔計算機のファイルシステムにおけるアクセス制御ポリシーが適用される。DEM や KITE に与えられたアクセス権限が遠隔計算機のアクセス権限としてそのまま適用される。例えば KITE に一般ユーザの権限しか与えられていない場合は遠隔計算機からは root 権限が必要なデバイスにはアクセスできない。

2.6 実装

本システムのプロトタイプ実装は Linux OS 上で行なった。DIM はファイルシステム部と DEM や KITE と通信する通信部で構成されている (図 7)。本研究ではファイルシステムの実装に FUSE を用いる。FUSE はユーザレベルでファイルシステムを実装するためのフレームワークであり、カーネルモジュールとライブラリ群で構成されている。カーネルモジュールは通常のファイルシステムとして VFS (Virtual File System) に登録され、ファイル操作システムコール

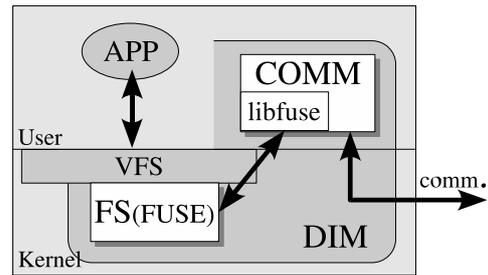


図 7 DIM の構成

Fig. 7 Structure of DIM

(open, read, write 等) が呼ばれたときに処理要求をユーザレベルのプログラムに渡す。転送されたシステムコールをユーザレベルプログラムで適切に処理することでファイルシステムとしての要求を満たす (後述するがユーザレベルプログラムで実装すべき関数はシステムコールそのままというわけではない。) 例えば read システムコールが発行されたときはアプリケーションプログラムに渡すデータをユーザレベルプログラムがどこから取得しなければならない。本システムの場合、それは遠隔計算機の DEM から取得することになる。本システムでは、ユーザレベルプログラムは通信部を通して DEM や KITE に RPC (Remote Procedure Call) と同じ要領でシステムコールを転送する。

DEM は DIM から転送されたシステムコールを自身が動作している計算機上で発行する。また KITE の生成・破棄を行なう。現在の実装では KITE は DEM を介して遠隔計算機の DIM と通信をする。性能を考えると DIM と直接通信するのが望ましく、将来的にはそのように改良を加える予定である。現在は UNIX パイプで DEM と通信を行なっている。

ここで FUSE を用いてファイルシステムを実装するために、ユーザレベルプログラムで実装しなければならない関数群について述べる。これらは VFS に登録する一連のファイル操作関数 (file_operations 関数) に基本的な部分は似ている。しかし file_operations 関数と比較しても異なる部分が多い。例えば VFS に登録する read, write などの関数の第一引数はカーネル内のデータ構造であるファイル構造体であるが、FUSE の場合はファイルのパス名である。また file_operations 関数に存在する readdir 関数は存在しない。代わりに getdir 関数が用意されている。この関数は open 関数+readdir 関数と同等である。

3. ハードウェア構成の動的変化への対応

ここでは本システムの機能拡張としてネットワークの動的変化とデバイスの移動に対応する方法について述べる。

3.1 ネットワークの動的変化

ネットワークの動的変化とは、移動体計算機の無線基地局間の移動やネットワークケーブルの付け替えなどの物理的なネットワーク変化により計算機に割り当てられた IP アドレスが変化することを指す。近年、無線環境が整備され計算機のネットワーク環境が動的に大きく変化するため、ソフトウェア的にこれに対応する技術に注目が集まっている⁸⁾。

3.1.1 関連研究

本節ではネットワークの動的変化を隠蔽する諸技術について述べる。Zandy らは従来の UNIX ソケットインタフェースを維持したままネットワークの動的変化やハードウェア的なネットワークの異常を隠蔽・修復する Rocks, Racks を提案している⁷⁾。また Kaneko らは同様のサービスをセッションレイヤにおいて実現している⁹⁾。これらの技術はソケットを用いて通信を行なうアプリケーションに対して信頼性のある通信路やモビリティを提供している。

CFS (Context-aware File System)¹⁰⁾ はコピキタスコンピューティング環境向け分散ファイルシステムである。ファイルシステムにメタデータとしてコンテキスト情報を持つことができ、コンテキストの違いを吸収する独自インタフェースを提供している。Francisco らはアプリケーションプログラムに物理的な通信路が透過的な通信サービスを提供する機構を提案している¹¹⁾。通信端点として抽象度の高いファイルシステムインタフェースを提供している。ORFA⁶⁾ や ActiveFiles³⁾ は動的リンクライブラリを用い、アプリケーションプログラムにネットワーク透過なファイルアクセス環境を提供している。

以下ではネットワークの動的変化を隠蔽するインタフェースについて設計方針を述べる。

3.1.2 ネットワークの動的変化を隠蔽するインタフェース

ネットワークの変化をアプリケーションプログラムから隠蔽するインタフェースには代表的なものとして次の 2 つの方法が存在する。(a) ネットワーク依存のインタフェースでネットワークの変化を隠蔽する方法^{7),9)} (b) ネットワーク非依存なインタフェースを提供する方法^{3),6),10),11)}、である (a) はいずれも UNIX ソケットレイヤのインタフェースを提供してい

る。接続開始時には通常通り相手先の IP アドレスを指定するが、ネットワークの変化が起きてもそれをアプリケーションプログラムに通知しない。そのため接続先がいつのまにか切り替わり、アプリケーションプログラムが想定している IP アドレスとは異なる相手と通信を行なっているという状態が起り得る。しかしこの状況は多くの場合アプリケーションプログラムにとって問題とならない (b) はいずれのシステムもインタフェースはファイルシステムである。ネットワーク処理を隠蔽しているため、アプリケーションプログラムはネットワークの変化を意識しなくてよい。しかしネットワークに関するエラーなどは隠蔽しきれないので特別な対応が必要となる。

本システムでは (a) と (b) のどちらのアプローチも取ることが可能である。2.3 節で述べたように本システムがアプリケーションプログラムに提供するインタフェースすなわちデバイスファイル群はパス名に IP アドレスや MAC アドレスを含むためネットワークに依存している。もし IP アドレスが変化したときにはデバイスファイル群のパス名が変化する。しかし (a) と同様にデバイスファイルを開いてファイルハンドラを取得したアプリケーションプログラムにとっては、パス名の変化は問題ではない。これは UNIX ファイルシステムの特徴でもある。またパス名に IP アドレスや MAC アドレスを使わないようにすることで (b) のアプローチを取ることできる。例えば IP アドレスの代わりに計算機を一意に定めるユニーク ID を指定する方法がある。これによりネットワークの変化が起きても同一のインタフェース (デバイスファイル名) により対象とするデバイスを指定できる。

3.2 デバイスの移動

本節では、デバイスの移動が起きてもサービスを継続できるように本システムを拡張するための諸要件について述べる。デバイスの移動とは計算機に接続された着脱可能デバイスが取り外され別の計算機に再び装着される状況を指す。このような状況はデバイスが無線ネットワークで接続されるようになるにつれ増加すると考えられる。デバイスに対応するためにはまずデバイスが移動したことを認識しなければならない。そのためにはデバイスの個体認識とデバイス着脱イベントの取得およびその伝搬が必要となる。

デバイスの個体識別にはデバイスに割り当てられた識別番号を使う方法と推論を用いる方法がある。現在はシリアル番号が付加され個体認識が容易なデバイス (例えば USB Flash メモリ) とそうでないデバイス (マウスなど) が存在する。後者の場合はシステムが

管理する計算機には同一の製品のデバイスは存在しないと仮定することで個体識別が可能となる。今後は無線化が進み、個々のデバイスに識別番号が割り当てられることが期待できる。

本システムではデバイスの着脱イベントの取得と伝搬をユーザレベルで行なう。近年の OS では着脱イベントの取得はユーザレベルのプログラムであっても行なえる。Linux OS ではデバイスの情報を管理する udev というシステムが存在し、D-BUS を介してアプリケーションプログラムにも着脱イベントを通知させることができる。

4. おわりに

モバイル機器や着脱可能デバイスが広まり基盤ソフトウェアにはハードウェア環境の動的な変化に柔軟に対応できる拡張性が求められるようになってきている。また基盤ソフトウェアには下位互換性を実現するという要求も同時に存在する。本研究ではユーザプログラムとカーネルの双方に手を加えることでこのような要求にも耐えうる構成を採用したシステムを構築した。

本システムではデバイスファイルという既存のインタフェースを維持しつつ、通信処理をユーザレベルで行なうことで柔軟な要求にも対応できる。遠隔計算機のデバイスファイル群をローカル計算機のファイルシステムにバインドすることでアプリケーションプログラムに従来と同等のインタフェースを提供する。アプリケーションプログラムによるデバイスファイルのアクセスは本システムを構成するユーザレベルのプログラムによって遠隔計算機へ転送される。通信処理をユーザレベルのプログラムで行なうことによって機能拡張を容易に行なうことができる。本稿では、計算機のネットワーク上の移動し IP アドレスに変化が生じる場合やデバイスの計算機間の移動する場合など、動的なハードウェアの構成変化に対してもサービスを継続できるようにシステムを拡張する方法について議論した。

現在ではデバイスが持つ固有のレジスタに対する処理やデバイスの移動時におけるレジスタ状態の保存や復元に対応していない。今後はこのようなデバイス固有の問題に対応しシステムの実現・評価を行なう。

参考文献

- 1) Brumitt, B., Meyers, B., Krumm, J., Kern, A. and Shafer, S.: EasyLiving: Technologies for Intelligent Environments, Proc. *2nd International Symposium on Handheld and Ubiquitous Computing*, pp.12–27 (2000).
- 2) Want, R., Pering, T., Danneels, G., Kumar, M., Sundar, M. and Light, J.: The Personal Server: Changing the Way We Think about Ubiquitous Computing, Proc. *the 4th international conference on Ubiquitous Computing*, pp.194–209, (2002).
- 3) Dasgupta, P., Itzkovitz, A. and Karamcheti, V.: Active Files: A Mechanism for Integrating Legacy Applications into Distributed Systems, Proc. *the 20th International Conference on Distributed Computing Systems (ICDCS 2000)*, pp.680–690 (2000).
- 4) 尾崎亮太, 丸山勝巳, 日高宗一郎, 児玉和也: ネットワーク上に動的に分散する多数のデバイスを制御する基盤ソフトウェアの検討, 並列 / 分散 / 協調処理に関する『青森』サマー・ワークショップ (SWoPP 青森 2004), pp.81–88 (2004).
- 5) Pike, R., Presotto, D., Dorward, S., Flan-drena, B., Thompson, K., Trickey, H. and Winterbottom, P.: Plan 9 from Bell Labs, Vol.8, No.3, pp.221–254 (1995).
- 6) Goglin, B. and Prylli, L.: Transparent Remote File Access through a Shared Library Client, the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '04), pp.1131–1137 (2004).
- 7) Zandy, V.C. and Miller, B.P.: Reliable Network Connections, Proc. *the 8th annual international conference on Mobile computing and networking*, pp.95–106 (2002).
- 8) 橋岡孝道: 変化する実行環境に動的適応するソフトウェアシステム構成法に関する研究, PhD thesis, 電気通信大学大学院情報システム学研究科 (2000).
- 9) Kaneko, K., Morikawa, H. and Aoyama, T.: Session Layer Mobility Support for 3C Everywhere Environments, Proc. *the 6th International Symposium on Wireless Personal Multimedia Communications (WPMP 2003)*, pp.347–351 (2003).
- 10) Hess, C.K. and Campbell, R.H.: A Context-Aware Data Management System for Ubiquitous Computing Applications, Proc. *23rd International Conference on Distributed Computing Systems*, pp.294–302 (2003).
- 11) Ballesteros, F.J., Castro, E.M., Muzquiz, G.G. and Algara, K.L.: A New Network Ab-

<http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev.html>
デスクトップアプリケーションプログラム向けのメッセージング機構である。 <http://dbus.freedesktop.org/>

straction for Mobile and Ubiquitous Computing Environments in the Plan B Operating System, Proc. *Sixth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'04)*, pp.112–121 (2004).
