200629009B

厚生労働科学研究費補助金
感覚器障害研究事業

視覚障害者、盲ろう者向け音声・点字コンピュータ・オペレーティングシステムの開発

平成16～18年度　総合研究報告書

主任研究者：石川　准（静岡県立大学）
分担研究者：河村　宏（国立身体障害者リハビリテーションセンター研究所）
寺島　彰（浦和大学）
湯瀬裕昭（静岡県立大学）

平成19年（2007）年　4月

目　　次

Ⅰ. 総合研究報告

第 1 章　目的 3

第 2 章　研究方法 3

第 3 章　倫理面への配慮 4

第 4 章　予備研究 4

第 5 章　平成 16 年度の研究 5

第 6 章　平成 17 年度の研究 8

第 7 章　平成 18 年度の研究成果 11

第 8 章　利用者をテスターとするシステム評価 13

第 9 章　Linux の今後 15

第 10 章　アクセシビリティ、ユニバーサルデザイン、支援技術 18

第 11 章　結論 24

第 12 章　補論 25

Ⅱ. 研究発表 31

Ⅰ．総合研究報告

第１章　目的

コンピュータは画面に情報を視覚的に提示し、人はポインティングデバイスで「選択」する。今日の人とコンピュータの相互作用は大部分がこの反復であり、それは GUI(Graphical User Interface)と呼ばれている。多くの視覚障害者や盲ろう者が利用している Windows 用スクリーンリーダは、じつはこの GUI を AUI(Auditory User Interface)や BUI(Braille User Interface)に変換するソフトウェアだといえる。だが、この GUI→AUI/BUI 変換は原理的に不完全なものにとどまる。そのため、GUI 搭載のコンピュータの操作は、視覚障害者、盲ろう者には大きな障壁、負担となっている。

一方、米国、ドイツ、ニュージーランド、韓国などの支援技術ベンダーにより点字表示機能と音声出力機能を有する専用多機能音声・点字携帯情報端末が開発され、広く利用されている。これらの製品の多くは独自のアプリケーションを搭載し、AUI と BUI の特性に合致したインターフェースを提供し、高い操作性を実現している。ただし、この種の専用機器の欠点として、一般社会で広く用いられているデータ形式、アプリケーション、ウェブサービスが利用できるとはかぎらないという問題がある。

本来ユーザインターフェースは、ユーザが自己の特性に応じて自由に選択できなければならない。我々は PC の汎用性と専用携帯情報端末の操作性を同時に実現すべく、Linux の CUI コンソール環境に注目し、それに音声合成、点字表示機能を組み込み、合わせて AUI/BUI 搭載の統合環境（エディタ、ブラウザ、メーラ等がシームレスに動作する）を開発する。それにより、視覚障害者、盲ろう者がコンピュータの専門的な知識がなくても就労や生活の場において自力で効率的に操作できるコンピュータ・オペレーティングシステムをオープンソースで実現する。

第２章　研究方法

本研究では、視覚障害者、盲ろう者向け音声・点字コンピュータ・オペレーティングシステムの設計、仕様策定、機能実装、試作への当事者参加と、当事者による評価を積極的かつ継続的に行った。

我々は既存の欧米の Linux 用スクリーンリーダの移植や改良といったアプローチの限界を確認したうえで、本格的な Linux 用スクリーンリーダをスクラッチから開発するという方法を選択した。Linux をターゲットとしたのはオープンソースであることと、コマンドの充実したソフトウェア資産があり、音声・点字コンピュータ・オペレーティングシステム開発のプラットホームとして最適と判断したからである。

また我々は、インストール作業という障壁を除去するために、CD 起動が可能な Linux ディストリビューションの提供という方法を選択し、その開発のための研究を行った。

さらには、AUI/BUI 対応アプリケーションとスクリーンリーダの連携を実現する方法として GRIF というブリッジを開発した。このブリッジでスクリーンリーダと通信するという方式で、エディタ、インターネットブラウザ、電子メールソフトウェアからなる統合環境を開発し OS に標準装備した。

第3章　倫理面への配慮

利用者をテスターとするシステム評価を実施するにあたり、研究の趣旨、目的、個人情報の扱いを説明し、同意を得ることを徹底した。

また学会等での研究成果報告も含め、テスターの個人情報守秘に万全を期した。

第4章　予備研究

4.1 BRLTTYの日本語移植

我々は本研究に先立ち平成15年度に予備研究を実施する機会を与えられた。

予備研究では、BRLTTYの存在に注目した。BRLTTYは点字ディスプレイで出力することを主眼に主としてヨーロッパにおいて開発されてきたスクリーンリーダである。すでに十年近い開発実績があり、欧米の視覚障害を有するLinuxユーザの間では広く知られたソフトウェアである。国内にも少数ながらユーザがいる。ただし、欧米言語の点字出力にフォーカスした設計であり、日本語への対応は考慮されていない。また、画面内容の音声による出力はあくまで補助的な機能である。

我々はBRLTTY Version 3.41を出発点に独自の拡張を加えて日本語スクリーンリーダを試作することをテーマとし、平成15年度の研究開発を行った。以下試作したスクリーンリーダをBRLTTY Plusと呼んで研究成果を報告する。

4.2 日本語対応の機能要件

スクリーンリーダを日本語対応にするには複数の要素を考慮する必要がある。日本語の文字コードへの対応はもちろんであるが、それ以外に、日本語自動点訳エンジンへの対応、日本語音声合成エンジンへの対応、日本語インプットメソッドへの対応等が必要となる。また、日本国内で多く使われている点字ディスプレイへの対応も必須である。

幸い、日本語の点字を扱うために必須の点訳エンジンは、石川の開発したExtra for UNIXが存在する。これはEUC-JPの日本語テキストを入力として与えると、点字形式の出力が得られるエンジンである。これを本研究の点訳エンジンとして使用することにした。

一方、日本語の音声合成システムはWindowsや組み込みシステム向けのものはいくつか存在するが、Linux向けのものは少数しか存在しない。

なお、多くの日本語インプットメソッドではユーザが変換したい文字や単語の読みを入力し、候補の中から適切なものを選択する。ところが音声で文字や単語を読み上げた場合、同音異義語の区別ができない。同音異義語の区別問題は仮名分かち書きで表現される日本語点字にも当てはまる。この問題に対応するために「詳細読み」という方法で変換候補として表示する文字がどのような漢字であるかを解説する。たとえば「日本語」という単語であれば「日曜日の『日』、本の『本』、語るの『語』」のように読み上げ、点字出力を行う。

4.3 移植において直面した問題と解決方法

オリジナルのBRLTTYは、Linuxの仮想コンソールデバイスである/dev/vcsaから画面内容を読み出し、点字出力を行うソフトウェアである。ところが、/dev/vcsaは日本語文字コードを正しく扱える設計になっていない。また、この方法では日本語インプットメソッドの詳細読みを実現することも困難である。幸い、BRLTTYには、/dev/vcsaを使用せずに、仮想端末ソフトウェアであるGNU Screen内のイメージを、共有メモリを使って読み出す試験的なパッチが存在した。我々はこのパッチを開発の出発点として利用することにした。

オリジナルのBRLTTYは、点字ディスプレイのボタン以外からのコマンドを受けとれない構造になっており、点字ディスプレイを接続していないPCでは起動さえできないという問題があった。音声合成を主として利用するユーザにとっては、キーボードからスクリーンリーダに対する指示が送れないことになる。そこで、BRLTTY Plusではコマンドをプロセスの外部から送れるようにIPCを用いたインターフェースを新たに設けることにし、併せて点字ディスプレイが接続されていなくてもBRLTTYが動作するように内部構造を変更した。また、Screenからはキー操作によってBRLTTY Plusに対してコマンドを送れるような拡張を行った。

さらに、日本語の点字を扱うために前述のEXTRA for Unix自動点訳エンジンを組み込み、日本語の音声合成を行うために、クリエートシステム開発株式会社のLinux音声合成ライブラリ用のドライバを開発した。また、ケージーエス株式会社のブレイルノート46X用のドライバを開発し、従来よりBRLTTYがサポートしていたALVAサテライト、パワーブレイル等の点字ディスプレイと合わせると、日本国内で使用されているほとんどの点字ディスプレイをサポートすることができた。

日本語かな漢字変換入力時の音声による詳細読みの提示機能を実現するために、詳細読みのライブラリも開発した。また、canfepというCannaのクライアントに詳細読みライブラリと、BRLTTY Plusと通信するためのIPCインターフェースを組み込むことで、同音異義語を詳細読みで情報提示しながら漢字入力を行えるようにした。

第5章　平成16年度の研究

5.1 概要

本研究の初年度に当たる平成16年度は、本格的な日本語Linuxスクリーンリーダの開発を行った。

平成15年度は、BRLTTYを元に開発を行ったが、BRLTTYにはいくつかの構造上の大きな問題点が存在することが明らかとなった。

欧米言語ではUS437やこれに類する点字体系を採用することで、画面に表示される英数字や記号類と点字ディスプレイに出力される点字に1対1の関係が維持できる。たとえば、1行80桁の標準的なVGAコンソール環境であれば、80桁の点字ディスプレイに対しては単純なテーブルを用いて1文字ごとに変換し、1行分の内容全部を出力できることが保障される。しかし、仮名分かち書きを行う日本語点字の場合1対1の関係は保障されない。たとえば、「南」という文字は、文脈によって「なん」(2桁)あるいは「みなみ」(3桁)という点字出力を行う場合があり、必要な桁数が変動する。BRLTTYはこ

のような桁数の変動を考慮に入れない設計となっている。

BRLTTY は音声合成機能を実現しているものの、音声合成に対する内部構造が貧弱であり本格的な用途には耐えない。各点字デバイス、音声合成デバイスに対する処理が直列的に実現されているために、音声合成中は点字ディスプレイからの操作がまったく行えない等の致命的ともいえる問題が明らかとなった。BRLTTY Plus では読み上げの中断機能という音声合成のスクリーンリーダにとって不可欠な機能を実現したが、BRLTTY の内部には読み上げの中断を行うための構造が存在しないため、IPC のインターフェースを音声合成ドライバが直接覗く極めて不自然な形で実現せざるを得なかった。

さらに、BRLTTY は一つのソフトウェアでスクリーンリーダとして完結しており、外部とのインターフェースが存在しない。我々は、外部のアプリケーションとスクリーンリーダが密に連絡を取りながら音声合成や点字表示を実現するモデルを描いているが、BRLTTY はこのようなモデルが実現できる基本設計となっていない。BRLTTY を元に引き続き開発を進めることは不可能ではないが、音声合成機能の充実や、本格的な日本語対応を行うには、基本構造を含む内部構造の大幅な手直しが必須であることが明らかであった。また、従来の内部構造に関する設計ドキュメントが存在しないことや、ソースコードにコメントが少ないソフトウェアであることも、今後の開発を行っていく礎とするのに大きな不安が残った。

また、BRLTTY のクライアントとして使用した Screen はデフォルトで Control+A をタイプした後に、コマンドキーを入力しなければならないという制約がある。MS-DOS や Windows のスクリーンリーダは、1 キーの入力でスクリーンリーダに対するコマンドを送ることができるので、これらと比較すると明らかに操作性の面で劣っている。また、Screen を使用する限りにおいては、削除文字を読み上げる機能の実現が極めて困難であることも明らかとなった。

以上のような観点から BRLTTY と Screen を元にして日本語対応の本格的なスクリーンリーダを開発するのは困難と判断し、BRLTTY の問題点を見据えた上で、BRLTTY Plus の経験を生かし、まったく新たな設計に基づいてスクラッチからの開発を行うこととした。設計に際しては、将来的な国際対応を視野に入れ、日本語固有の構造を避け、多言語環境が実現できる構造とした。また、Linux 固有の実装はできるだけ避け、他の UNIX システムでも動作できるような配慮を行った。実際に、Mac OS X でも使える音声合成エンジンの種類は異なるものの、ほぼフル機能が実現できている。

今回開発したスクリーンリーダは grd, grfep, grtty の三つのソフトウェアから構成される。grd はスクリーンリーダの中核を担うデーモンであり、grfep が日本語入力用のクライアント、grtty が仮想端末という役割分担となっている。BRLTTY Plus における BRLTTY デーモンが grd で、Screen が grtty に相当する。開発したそれぞれのコンポーネントの詳細は次のとおりである。

5.2 grd

GR for UNIX のシステムの中核を担うデーモンである。GRIF というインターフェースを通じて音声読み上げおよび点字表示サーバ機能を提供する。GRIF 対応クライアントからのリクエストに応じ、音声読み上げや点字表示、コマンドの実行等を行う。クライアントとの GRIF インターフェースは、UNIX

ドメインのソケットを使用し、各クライアントとの通信専用スレッドを作成することで対応した。さらには、後述の grtty 等の仮想コンソールソフトからは、共有メモリを通じて画面情報を受け取る。

また、チャンネルと呼ぶ複数の仮想画面を扱えるようにし、スクリーンリーダの読み上げ対象画面をいくらでも拡張でき、音声読み上げや点字表示は、複数の「チャンネル」から一つを選んで行える構造とした。ヘルプ機能はこのチャンネルを使用して実現したが、将来的にクリップボード用のチャンネルや、コマンドメニュー等への応用も可能である。

grd は点字ディスプレイからのコマンドの監視、音声合成、GRIF サーバの三つの役割をそれぞれ単独のスレッドを作成し実行する。このため特定のデバイスの処理により、別のデバイスが待たされることがない。複数の音声合成ドライバや、点字ディスプレイデバイスを同時に利用できる構造になっている。

grd では複数の音声合成ドライバをサポートした。日本語音声 TTS エンジンとして、クリエートシステム開発株式会社の Linux 音声合成ライブラリ用のドライバを開発した。また、英語 TTS エンジン用のドライバとして、Festival と AT&T Natural Voice 用のドライバを開発した。さらに、Mac OS X で grd を動作させるために、日本語と英語をサポートしているクリエートシステム開発株式会社の Document Talker 用のドライバも併せて開発した。

音声出力は生成された PCM データを esound (ESD) を経由して、デバイスに出力することで音声デバイスの占有を防ぐ構造としたが、esound の動作が著しく不安定であることが、音声ドライバ開発を非常に難しくした。esound の代わりとなる、ミキサー機能の選定が今後の課題といえる。

点字ディスプレイを使用するためには、各社の点字ディスプレイに対応した点字ディスプレイドライバが必要になる。grd では、石川の開発した視覚障害者用の Windows 上のエディタ、ブラウザ、メーラである Altair for Windows の点字ディスプレイドライバ部分をライブラリとして独立した構造に切り離し、UNIX に移植することで対応した。

grd では EXTRA for Unix 自動点訳エンジンを組み込んで点訳を行っているが、日本語のみに依存した構造を避けるため、他の点訳エンジンを組み込むことが可能な構造とした。

5.3 grtty

grtty は今回独自に開発した仮想コンソールソフトウェアである。grtty は、疑似ターミナルと、vt100 エミュレータ、GRIF クライアントから構成される。grtty 経由で起動したシェルの操作内容は、grd に伝えられ、音声読み上げや点字表示が行われる。また、現在画面に表示されている内容や、過去に表示された内容を、grd 側のレビュー機能により音声読み上げや点字表示することができる。

grtty は別端末で複数起動が可能なので、読み上げ対象とする仮想端末を切り替えながら利用することができる。コンソールで ALT+ファンクションキーによる複数の tty の切り替えを行えば、GNU Screen による複数仮想端末の起動と同等の効果を得ることができる。

ファンクションキーや、PgUp/PgDn 等の特殊キーを、スクリーンリーダのレビュー機能に割り当てることで、Screen を使用した場合と比べて操作性が格段に向上した。

しかし、UNIX端末の制約のため、現在はファンクションキーや特殊キーをレビュー用のキーに割り当てている。一般のスクリーンリーダではテンキーをレビューに用いるのが常道となっており、現在のキーアサインはユーザビリティとしてはまだ不十分と感じられるので、この部分の改善が今後の検討課題である。

5.4 grfep

ユーザがgrfepを使って入力したアルファベット文字列は、Cannaのサービスにより漢字列に変換される。grfepは候補漢字列をGRIF経由でgrdに送る。grdは与えられた漢字列の詳細読み情報の音声読み上げと点字表示を行う。あらかじめ、grfepを起動した上でgrttyをexecで起動すると、grttyの環境下での日本語入力が可能になる。

以上のような開発を行うことで、前年度に実現したLinuxスクリーンリーダを上回る機能と操作性を備えたスクリーンリーダが実現できた。ドキュメント等の整備が済み次第grd, grtty, grfepはフリーソフトウェアGR for UNIX Version 2として商用コンポーネントを除く部分を公開する予定である。

第6章　平成17年度の研究

6.1 概要

平成16年度に開発したスクリーンリーダは、Linuxディストリビューションに極力依存しない構造としている。このため、様々なディストリビューションで動作が可能であると考えられるが、実際にはディストリビューションの種類によっては、組み込まれているライブラリのバージョンの違いや、Linux カーネルのバージョンの違い等といった理由でまったく動作しない、あるいは音声合成が不安定になる等の問題が見られた。また、安定して動作するディストリビューションを使用した場合でも、動作させるPCによっては、サウンドカードとの相性により音声合成がうまく動作しない問題が明らかになった。

このようなLinuxディストリビューションとの組み合わせ問題や、使用するサウンドカードの問題を解消しない限り、我々が目指す実用的なオペレーティングシステムとはなり得ず、また、利用者によるシステムの十分な評価も行えない。当初は、平成17年度の課題として音声ユーザインターフェース・点字ユーザインターフェースを有する統合ソフトウェアの開発を計画していたが、研究計画を一部修正し、具体的なディストリビューションの開発とサウンドドライバの問題を優先することとした。（当初、ディストリビューションの開発は最終年度の計画であった。）この他には、Emacsという技術者向け高機能統合ソフトウェアを音声化するソフトウェアであるBEPを本システムの内部で動作できるようにするブリッジモジュール開発と、スクリーンリーダのユーザビリティの改良等を行った。各研究成果についての詳細は次のとおりである。

6.2 音声合成の安定化

スクリーンリーダの音声合成出力は、生成されたデジタル音声データを Esound (ESD) というミキサーシステムを経由して、サウンドデバイスに出力することで音声デバイスの占有を防ぐ構造としている。Esound は Linux で広く使われているミキサーシステムであるが、ディストリビューションによって組み込まれている Esound システムのバージョンがまちまちである。古いバージョンの Esound システムは非常に不安定であり、これを使用すると音声合成の出力が満足に行えない。さらに、最新バージョンの Esound システムを使用しても、使用するサウンドカードによって問題が発生することが判明した。ところが、Esound システムはこの数年間メンテナンスが行われておらず、今後もクオリティーが大幅に改善される目処は立っていない。

そこで、Esound のミキサーシステムの代わりに ALSA という別の音声ミキサーシステムを使用するように、スクリーンリーダの該当部分を差し替える大幅な変更を行った。また、音声合成の出力部分で Linux のバージョンによって異なる振る舞いをする箇所の修正等も併せて行った。このような改良の結果として、音声合成の安定性が向上し、サウンドドライバのハードウェアとの相性問題も最小化した。なお、ALSA ミキサーシステムは、ディストリビューションによってはデフォルトでは利用できないことが多いため、独自ディストリビューションの必要性が高まった。

6.3 BEP ブリッジの開発

Linux には Emacs というコンピュータ技術者向けのテキストエディタ、メーラ、ブラウザ、ファイラ等の統合環境がある。欧米では Emacs を音声合成対応した Emacspeak という研究があり、Emacspeak の日本語を含む多言語化を行った BEP という研究成果がある。BEP は、Emacs から bep-ss (BEP Speech Server) というモジュールを経由して音声合成システムを呼び出す構造となっている。我々は、音声合成システムの代わりに、スクリーンリーダに用意した GRIF 外部インターフェースを呼び出すブリッジとなる bep-ss を開発し、スクリーンリーダと同時に BEP を動作させることに成功した。BEP は単独では、Emacs が起動されるまでは音声合成が行えないシステムであるが、今回のブリッジの開発により、Emacs の外はスクリーンリーダ、Emacs の内部では BEP という分業が行える環境を実現した。

BEP ブリッジの開発により、GRIF 外部インターフェースを組み込んだアプリケーションが開発可能であることを実証できたと同時に、GRIF 外部インターフェースの課題 (話者のスピードやピッチの動的な変更、コネクションごとの動作モードの必要性等) も明らかになった。来年度の課題である一般ユーザ向けの統合環境の実現に向けて、この経験を生かしていきたいと考えている。

6.4 スクリーンリーダの改良

昨年度に開発したスクリーンリーダは、使用する点字ディスプレイや音声合成システムをスクリーンリーダのコンパイル時に設定する必要があった。スクリーンリーダを一般の利用者に使用してもらうには、点字ディスプレイの機種、接続パラメータ、音声合成システムの種類、話者、音声のスピードやピッチ等の設定を、後から変更できなければならない。

本年度は、スクリーンリーダに設定ファイルを設け、点字ディスプレイおよび音声合成の設定や、キーボードの特定のキーをどのようなスクリーンリーダのコマンドに割り当てるか等の設定ができるようにした。また、変更を動的にスクリーンリーダに反映させる機能を設けた。

他にも、ユーザからのコメントを基に、レビュー時の細かな動作の修正や、点訳の言語体系を動的に切り替える機能等、スクリーンリーダの使い勝手の向上を中心とする改良作業を行った。

6.5 CD起動が可能なディストリビューション開発

Linuxには数多くのディストリビューションが存在する。この多様性はLinuxの活力の源であるが、スクリーンリーダを開発する立場からすると、検証しなければならないシステムが他数存在することになる。前述のように、ディストリビューションによって組み込まれているEsoundシステムのバージョンが異なったり、Linux カーネルのバージョンやライブラリのバージョンが異なるため、スクリーンリーダの動作環境として、不特定のディストリビューションを対象とすると、このようなバージョンの違いを意識しなくてはならず、多様なシステムへの対応に貴重な研究開発の工数を多く費やしてしまう。特に、Linux のマルチメディア関連のデバイスやライブラリのサポートはシステムとして未熟な状態であるため、バージョンの違いが及ぼす影響度は非常に大きい。音声合成を前提とする本研究では、本年度より独自のディストリビューションに限定して研究開発を進めることで、開発リソースを最小限に抑えるアプローチを採った。

独自ディストリビューションは、KNOPPIXというディストリビューションを出発点とした。KNOPPIXは、CDからオペレーティングシステムを起動できることを特徴とするディストリビューションである。CD起動が可能であるということは、ハードディスクにオペレーティングをインストールしなくても利用できることを意味する。このようなシステムにスクリーンリーダを組み込めば、オペレーティングシステムをCDから起動した後に、ハードディスクへのインストールを行うプログラムを動作させることは、視覚障害者、盲ろう者であってもスクリーンリーダを用いることができるので十分可能と考えられた。本研究の最終目標は、視覚障害者、盲ろう者が技術的な知識が無くても自力でインストールすることが可能なオペレーティングシステムの開発であるが、KNOPPIX をディストリビューションの出発点とすることで、その目標に大きく近づくことができる。

ディストリビューションの作成にあたり、スクリーンリーダの組み込み、ALSAのサウンドシステムの組み込み、音声合成システムの組み込み、点訳エンジンの組み込み、BEPの組み込み、GUIからテキストベースのシステムへの変更、各種起動スクリプトの開発等を行った。また、KNOPPIXはGUIのLinuxが起動するシステムであったが、テキストベースのシステムとして動作するような変更も行った。このような開発の結果、CD起動が可能な、本研究の成果が組み込まれたディストリビューションが出来上がった。

第7章　平成18年度の研究成果

7.1 概要

平成18年度は、一般の利用者にとっても使いやすい統合環境を開発する目的で、音声・点字統合環境（Sirius on Linux）を開発しシステムに組み込むことに成功した。

7.2 Siriusの設計概要

Siriusは次世代型の音声・点字インターネットブラウザを実現する目的でスタートした研究である。ソフトウェアの構造としてXML情報をメモリ上に木構造に展開されたDOMという標準形式でデータを保持するのが大きな特徴である。

Siriusでは指定したURLの情報をインターネットからダウンロードした後、エンコーディングを自動判定し、DOM上にデータを読み込む。HTML情報をDOM形式で読み込むには、niggles というSGMLパーサを利用している。また、XHTMLやRSSなどのXML情報はJIS X4159の日本工業規格に準拠した独自のXMLパーサを通じてDOM形式に変換している。DOM部分は独自に開発したモジュールを使用しており、W3CのWD-DOM-Level-2-Core-20001113の勧告で定義されている機能を備えている。

さらに、Sirius の内部には REC-xpath-19991116 の勧告に基づく XPath のモジュールや、REC-xslt-19991616の勧告に基づくXSLTのモジュールがある。これらのモジュールは、ダウンロードしたコンテンツのレビューやトランスコーディングに利用している。

SiriusにはJavaScriptのエンジンが組み込まれている。KDEというオープンソースプロジェクトで開発されたKJSというJavaScriptのエンジンを組み込んでいる。コンテンツのダウンロード時や、onClickなどのイベントに応じてJavaScriptが動作する。

Siriusによるコンテンツのレビュー操作は、以上のような内部構造を利用して実現している。たとえば、コンテンツの閲覧は、メモリ上に展開されたDOMの枝を渡り歩きながらテキスト情報や、属性情報を利用者に対して情報提示する。フォームの入力は、DOM 形式の各エレメントの属性を動的に書き換える形で実現している。この構造により、フォームの入力中に、いったん別の場所にレビュー位置を移動しても、元のフォーム入力位置に戻ると、以前に入力中だった内容が残っているので入力を再開できる。RSSを閲覧する場合には、DOMに読み込まれたRSS情報を、XSLTを使って不要な情報をそぎ落とし、レビューに必要な情報のみに絞ることができる。

7.3 移植性

当初、Siriusは平成16-17年度にテクノエイド協会の福祉用具研究開発助成（研究課題名：「次世代音声・点字インターネットブラウザの開発」）を受け、Windows 環境を対象として開発を行ったが、開発初期より移植性を重視して設計を行った。OSの環境に極力依存しないように、画面やキーボードの入出力をViewというクラスにパッケージ化している。音声合成、点字ディスプレイ部分については新規に開発した ACC Driver というアクセシビリティ関連のドライバを通じて実装を行った。Siriusの多くの機能は CCC という C++のクラスライブラリを用いて実現しているが、このライブラリは

Windows, UNIX, Mac の 3 つのプラットホームをサポートしている。

Sirius の本音声・点字オペレーティングシステムへの主たる移植作業は、Windows の API で構成されていた##View 部分を、UNIX の端末用として新たに実装する作業であった。また、音声合成、点字ディスプレイの機能を実現するために ACC Driver を呼び出していた部分については、音声・点字オペレーティングシステムのスクリーンリーダインターフェースである GRIF を呼ぶ形式に変更した。

7.4 現段階で実装している機能、今後の目標

Sirius では、一般的な WEB ページの閲覧はもちろん検索操作などのフォーム入力が可能である。通常のテキストと、リンク、フォーム入力箇所、ボタンなどは音声合成の話者の違いで区別される。フォームの入力部品の区別については種別が音声合成で情報提示される。レビュー位置は、上下カーソルや TAB キーで移動できる他、Page Up, Page Down のキーで見出しへのジャンプ、Home, End のキーによるコンテンツの先頭や末尾への移動が可能である。F2、F3 のキーにより「戻る」「進む」操作が可能である。

フレームについては、通常のブラウザでは 1 画面中に表示される複数フレームを、一度に別々の DOM として読み込む機能を実装している。各フレームはスロットと呼ぶ単位にロードされ、利用者はスロットを切り替えながらそれぞれのフレームの内容を閲覧できる。

Sirius の JavaScript 機能は、基本的な演算や操作は一通りサポートしているが、まだ未実装の JavaScript 機能があることや、ブラウザにより実行結果が異なる部分があるため動作できるスクリプトが限られている。特にブラウザによって実装の異なる機能などは今後の大きな課題である。たとえば、実際のスクリプトでは実行環境がどのブラウザかを判定し動作を切り替えていることが多い。現在の Sirius では、Internet Explorer として判定されることが多いが、Windows の Internet Explorer でしか動作し得ない ActiveX のコンポーネントを動作させようとして Sirius の JavaScript エンジンがエラーになることが多い。このような対応については今後の大きな課題である。現在、Sirius では Ajax 機能は利用できないが、Ajax が動作するレベルまで JavaScript 機能を充実させることが次なる大きな目標である。

現在、Sirius は UNIX の端末ソフトウェアの内部で動作している。このため、インサートキーの押下のような特殊キーを機能に割り当てることができない。特殊キーを利用するには、UNIX の端末ソフトウェアではなく、X Window System の下で動作する端末ソフトウェアで Sirius を動作させる等の改良が必要である。

現状の Sirius には点字ディスプレイからの操作ができないという問題がある。Sirius は GRIF というスクリーンリーダへのインターフェースを通じて点字ディスプレイに情報提示している。ところが GRIF では、アプリケーション側が点字ディスプレイからのボタン押下などのイベント情報を補足できない。この部分については、GRIF のインターフェースの拡張が必要と考えられる。

その他 Sirius は、機能面やユーザビリティの点で荒削りの面は否めない。利用者からのフィードバックを生かし、さらなる操作性の向上をめざす予定である。

第8章 利用者をテスターとするシステム評価

8.1 概要

利用者によるシステム評価を実施した。

テスターは3名、いずれもUNIXの使用経験のある視覚障害者で、技術職ないしは技術系の仕事に従事している。

以下にテスターのコメントを要約する。問題点を指摘するものには※を前置する。

8.2 全般的所見

音声、点字ともに必要なレスポンスを達成している。

UNIX環境の日本語スクリーンリーダで初めて実用的な水準に到達したという点で画期的である。

また、端末の画面サイズの制約にとらわれず、スクロールアウトした画面領域までさかのぼって画面出力を確認でき、晴眼者用に設計された環境の制約を乗り越えた視覚障害者等の利用にフォーカスしたコンピュータ環境として有望である。

音声はWindows上のSAPIを用いた音声合成ライブラリと遜色のない品質である。音切れは実用的な水準である。

点字出力についてはEXTRA点訳ライブラリを用いて高度な正確さの日本語点字出力が得られている。

点字ディスプレイ上のボタンを用いたスクリーンレビューのサポートも確認できた。ステータスセルを設けて画面情報を通知する仕組みを準備しており、スクリーンレビューの効率を高めている。

※点字ディスプレイのボタンをもっと活用するべき。画面の上端/下端/左端/右端移動、カーソル位置に移動、任意のボタン+タッチカーソルでその位置の文字の詳細読み、任意のボタン+タッチカーソルクリップボードにコピー/アペンド、クリップボードの内容をペースト。

またCD-ROM起動のLinux環境でしばしば見られる反応の遅さ、とくにファイルシステム読み書きの遅さは感じることはなかった。現在のCD-ROMベースの環境で一般ユーザが日常的に利用できるシステムとなる。

ブートアップ時に端末1から3にそれぞれ自動でgrtty, Sirius, BEPが起動するように工夫されている。この配慮によってUNIX初心者であってもGR for UNIXの一通りの機能を試せる。

※ステータスセルで行内のどのあたりを点字ディスプレイに表示しているかを通知すべき。

※KNOPPIXなのでマルチユーザでの利用がサポートできない。全ての操作がroot権限という今の状況はよくない。

※CD-ROMのサイズ制限で利用できるコマンドが限られている(Ex: zshがない)。

※pdf/doc/xlsなどのデータを取り出すコマンドを収録してほしい。

※どの仮想端末に切り替えたか通知する機能が必要。

8.3 grtty

UNIXの端末環境が一通り利用できる水準に到達している。

UNIX の端末環境では terminfo や termcap を用いたエスケープシーケンスによる画面描画の制御が多用される。こうした画面制御では画面上で 1 文字書き換えられた場合に、当該行あるいは画面全体の書き換えを行うことが少なくない。このように画面の書き換えが行われると、音声環境では 1 行全体あるいは画面全体を全て読み上げることになり、ユーザビリティを大きく損なう。

この問題をカバーするには、適切な terminfo エントリの設定と個別アプリケーションの対応という二つの作業が必要である。grtty ではどちらも手がけており、基本的な端末での作業において、上に述べたような無意味な読み上げが発生することはなかった。

日本語表示は問題ない。Canna を利用した日本語変換も利用できる。また、詳細読みも分かり易い適切な表現が用いられている。

※GR for DOS のような読み上げ対象のアトリビュートの細かい設定機能や、画面スキャンによる表示内容の逐次検出がほしい。

※テンキーによるレビューがほしい。

8.4 Sirius

JavaScript インタープリタを内蔵したテキストブラウザとして必要な機能が整備されている。

JavaScript で逐次書き換えられるページを書き換えに従って随時読み上げできることを確認した。

JavaScript の別ウィンドウで URL を開く機能をサポートしている。

この機能を用いて開かれたページは別スロットとして分岐し、別系統のヒストリが形成されるようになっている。

そのため分岐する複数のヒストリをブラウザ内に保持することが可能となっている。

ヒストリは履歴として自由に移動でき、異なる系統のヒストリにも移動ができるようになっている。

この機能をサポートするテキストブラウザは初で、テキストブラウザによる本格的な広域ウェブサーフィンが期待できる。

grtty 同様日本語入力が可能で、エディットボックスに日本語文字列を入力しキーワード検索が実行できることを確認した。

レンダリングは XML のノード単位で改行が行われており、見通しを持ってページ内を移動することができる。また URL リクエストを送信してからレンダリングが完了するまでの待ち時間が極めて短く、待たされるストレスは殆ど皆無である。これは IE や Firefox など最新の高機能ブラウザと比較しても大きなアドバンテージと感じた。

※JAWS の IE 読み上げレベルの機能がほしい。

※リダイレクト関連の処理が不十分。

※各ページをサマリ表示してヒストリツリーを移動する機能がほしい。

※かならずしもノードごとに改行する必要はない。どのノードで改行するか、どのノードの場合改行しないか取捨選択する必要がある。

※ページ内の相対的位置(何%の位置にいるか)や現在のノードについての情報をステータスセルに

表示してほしい。

8.5 BEP

BEP(Bilingual Emacspeak Platform)の互換スピーチサーバが実装されており、BEP が利用できる。

これによりプログラムの作成やメールの読み書きなど Emacs の機能を一通り使うことができる。

完全な互換スピーチサーバの実現にはいたっていないが、概して従来の BEP と同じ操作感で利用することができた。

GR for UNIX に含まれるコンポーネント群の中でも特に即戦力としての利用が期待できる。

※BEP のインテキストコマンドを正しく処理できていない。

8.6 総合評価

初心者からパワーユーザまで多様な視覚障害者の利用に供することのできるコンピュータ環境としての基礎的な要素が整っていると感じた。

システムの設計に関るような根本的な問題点はおおむね解決されている。

既にプログラミングのような単純なテキスト編集とコマンド実行で完結する作業を行うには十分な環境が整備されている。

今後はユーザの具体的ニーズを反映しつつ個別的な問題点を一つ一つ解決して行くことが望まれる。

こうした個別的な問題としては

1. 点字ディスプレイによるステータス表示の一層の工夫
2. PC キーボードと点字ディスプレイのボタンをフルに活用したスクリーンレビュー機能の充実
3. 高機能音声ブラウザが有するコンテンツナビゲーション機能の Sirius への実装

を特に指摘しておく。

UNIX はテキストベースの環境であり、完成度を高められれば Windows よりも視覚障害者の利用特性にマッチした環境を構築できる。上述のようにこうした環境構築のための基礎は出来上がっていると判断した。

第 9 章　Linux の今後

9.1 Linux の普及状況と今後の可能性

この数年で Linux はサーバ用途と組み込み用途を中心に普及が進んだ。薄型テレビ、携帯電話、プリンタ、カーナビ、HDD レコーダ、飛行機客席のコンソール、ルータなどに Linux が組み込み用 OS として使われている。組み込み用途に関しては利用者が気がつかないで Linux を使っているケースが多い。サーバ用途についても、数年前より Windows サーバの強力なライバルというポジションが定着している。

国内での Linux に関するデスクトップ用途の普及はいまだ限定的である。しかし、近年、Linux で動作する Microsoft Office との互換性の高い OpenOffice や、Internet Explorer と比べて遜色のな

いインターネットブラウザであるFirefoxが登場したことで(いずれもフリーソフトウェア)、国内での一般利用者へのデスクトップ環境としての普及条件は揃いつつある。

デスクトップ用途の普及には、アプリケーションの充実だけでなく、プリンタ、サウンドなどの周辺装置類のドライバの充実も不可欠である。近年、ハードウェアベンダーはWindowsのドライバの開発に併せて、Linux用ドライバを開発する傾向が強まっており、Linuxにとっては追い風の状況である。

Windows Vistaの登場により、PCに必要とされるメモリ、ハードディスク容量などのハードウェアリソースは非常に大きくなった。一方Linuxも、それなりにハードウェアリソースが必要とされるように進化してはいるが、数年前のPCであっても十分に利用できるというメリットがある。サポートの打ち切られたWindowsの代わりにLinuxを使うという用途も広がり始めている。

しかし、このような追い風の要素は存在するとはいえ、まだ一般のユーザを引きつけるだけの導入メリットが少ないため、Linuxのデスクトップ環境がこの数年で爆発的に普及することは無いと予想できる。近年のIT環境は、大きくインターネットに舵を切った形で進化が続いている。これまではデスクトップのパソコン上で全ての作業を行うという利用形態が多かったが、徐々にブラウザ経由で作業を行う形態に移りつつある。ブラウザが主たる作業環境になれば、必然的にOSはどのようなものでも構わなくなる。このような局面において、Linuxは軽量で費用がかからないという面を生かし、デスクトップOSとは別の方向性であるインターネット端末用OSとして徐々に普及が進む可能性が高い。

9.2 Linuxのアクセシビリティ

近年Linuxのアクセシビリティを実現する取り組みが複数のベンダーにより試みられている。

たとえば米国SUNマイクロシステムズ社はORCAというスクリーンリーダを開発しており、また米国IBM社もLSRというスクリーンリーダを開発している。前者はLinux、後者はSolarisとLinux上で動作するGnome等のGUIデスクトップ用スクリーンリーダであり、いわば「もう一つのGUIスクリーンリーダ」を開発しようとするプロジェクトである。

より注目されるのはIcon Mobile ManagerというLinuxをOSとする音声携帯情報端末である。これはLevel Starというベンチャー企業により開発されているPDAで、まもなく製品化される見通しである。Level Starのアプローチは我々のそれと共通する点が少なくない。

9.3 Linuxの問題点

Linuxには多くのディストリビューションが存在する。ディストリビューションによってカーネル以外の実装状況は異なっている。開発方針や、開発目的もまったく異なるし、クオリティーも異なる。

Linuxのディストリビューションの多さは不要な混乱を利用者にもたらしている。たとえば、ディストリビューションによってインストール方法、ネットワークの設定、画面の解像度のコントロール方法などは異なる。商用のバイナリプログラムは、あるバージョンのライブラリを前提とする関係で、特定のディストリビューションでなければ動作しないことがある。このような混乱は、すべて利用者側のデメリットとなっているのが現状である。

また、数多くの利用者がおり、将来的にも開発が継続されると思われていたディストリビューションであっても急に開発が打ち切りになることがある。Linux のディストリビューションがビジネスとして成り立ちにくいのが主たる理由である。ディストリビューションの開発が終了すると、その後のセキュリティアップデートなどが行えなくなるため、利用者は別のディストリビューションに移行することになる。その際に、ディストリビューションによる使い方の違いは大きなハードルとなる。

現在の Linux のディストリビューションの乱立は進化論の実験場ともいえる状況になっている。進化論の立場からすると、ディストリビューションの多さはプラスであるが、運悪く衰退してしまうディストリビューションを使っていた利用者は大きなデメリットを被ることになる。利用者の混乱の他にも、貴重な開発リソースが分散してしまう問題もある。現在、一部で Linux のソフトウェア構成等を標準化しようという動きがあるが、このような動きに期待したい。

9.4 音声・点字オペレーティングシステムの基盤としての Linux

前述のディストリビューションによるソフトウェア構成の差異は、音声・点字オペレーティングシステムの研究段階に大きな影響があった。

音声合成出力を行うにはサウンドカードやサウンドドライバのサポートが不可欠である。特定のバージョンのディストリビューションでは音声合成が動作するが、アップデートを行うと音声合成がうまくいかないという問題に頻繁に遭遇した。また、日本語の扱いが、EUC コードであったり、Unicode であったりと多様であり、curses（テキスト画面の制御ライブラリ）の日本語サポート状況がディストリビューションにより扱いが異なるという問題もあった。

開発当初は RedHat Linux というディストリビューションを元に開発を行っていたが、RedHat Linux のデスクトップ版が今後リリースされないことになったため、開発の元となるディストリビューションを FedoraCore に変更した。

ディストリビューションの変更や、ディストリビューションのバージョンを変更する度に、上記のような問題に度々遭遇した。音声・点字オペレーティングシステムの開発後期は、オペレーティングシステムの出発点として使用するディストリビューションを CD-ROM起動が可能な KNOPPIX に固定することで、ディストリビューションの差異による混乱を防ぐこととした。

ディストリビューションの多様性に翻弄された一方で、突然変異のような CD-ROM 起動が可能な KNOPPIX というディストリビューションが現れたことで、インストールをせずとも音声・点字オペレーティングシステムが利用できるという大きなメリットを享受できることになった。

昨今の KNOPPIX はさらに進化を遂げ、USB メモリからの起動も可能になっている。音声・点字オペレーティングシステムについても、USB メモリから起動できるような機能拡張を行い、さらなる利便性の向上をめざしたい。

第10章　アクセシビリティ、ユニバーサルデザイン、支援技術

10.1　ユニバーサルデザインと支援技術

ユニバーサルデザイン(Universal Design)とは、身体的・能力的特性の違い、文化的・言語的差異などを問わずに利用することができる施設、設備、製品、サービス、情報などの設計(デザイン)のことであり、端的にいえば「できるだけ多くの人が利用できるようにデザインする」という物作りの基本原則のことである。

このようなユニバーサルデザインを進めるには、さまざまな特性をもった利用者、製造メーカー、サービス提供事業者、行政、研究者等が一堂に会して継続的に話し合い合意形成し、ユニバーサルデザインを具現する社会規範(固い法や柔らかい法)を策定して、全アクターの行動を等しく規制するとともに、ユニバーサルデザイン化に努力したアクターが利益を得られるような選択的報酬を社会全体として提供する仕組みを整える必要がある。

こうした社会的仕組みのないところでは、概して個々の事業者は、自己の費用対効果のバランスシートを無視してまでユニバーサルデザインのコストを引き受けようとはしない。市場原理の上で利益を追求する事業者、利益追求を怠ると市場から退場しなければならない事業者からすれば、顧客群の特性分布の周辺部を適当なところで切断し顧客の範囲を限定するのはきわめて合理的な行為だからである。

ユニバーサルデザインが最も上流の技術であるのに対して、支援技術(Assistive Technology)は最も下流の技術である。ユニバーサルデザインがユーザ特性の多様性に配慮して、なるべく多くのユーザにとって使える・使いやすい施設、設備、製品、サービス、情報を提供するものであるのに対して、支援技術はユーザに最も近いところに位置し、個々のユーザ、特定のユーザ群に対してアクセシビリティ(利用可能性)やユーザビリティ(使いやすさ)のためのソリューションを提供する(図1)。

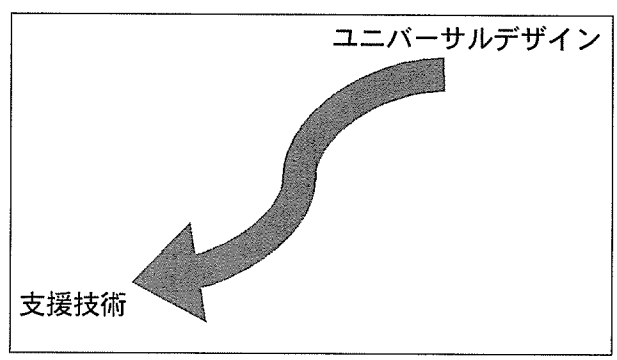

図1 上流の技術と下流の技術

支援技術を代表するものにスクリーンリーダがある。スクリーンリーダは、画面拡大ソフトウェアや点字ディスプレイ端末などとともに視覚障害者がコンピュータを操作する際の必須の道具である。また、手をうまく動かすことのできない上肢障害者には、オンスクリーンキーボードやスイッチが必要であり、運動障害がさらに重くなれば、視線入力や筋電入力という支援技術が必要となる。

ユニバーサルデザインと支援技術の関係は相補的である。一方が存在しないところ、不十分にしか存在しないところでは、アクセシビリティを実現するためのもう一方の負担は過度なものとなるのみならず、実現できるアクセシビリティの水準は低いものにとどまらざるをえない。言い換えれば、両者がそれぞれの役割を十分果たしてこそ高いアクセシビリティが実現するといえる。

10.2 アクセシビリティ

1998年の米国リハビリテーション法508条の改正とその施行規則にあたる「電子・情報技術アクセシビリティ基準(Electronic and Information Technology Accessibility Standards)」の策定は電子情報通信分野のアクセシビリティを進展させる出来事だった。

この法律によって、連邦政府が調達、使用する製品や、一般市民に提供する情報、サービスに対して、障害を持つ政府職員、一般市民が、障害を持たない人と同等にアクセスできるようにすることが義務付けられた。政府のウェブサイトや、連邦政府が新たに購入する情報機器やソフトウェアなどは、それが「過度の負担」とならない限り、電子・情報技術アクセシビリティ基準を満たさなければならなくなった。最大の顧客である米国政府向けの製品、サービスをアクセシブルにしなければならないということから、ICT業界のアクセシビリティへの関心は否応なく高まった。

とはいえ、米国リハビリテーション法という強い追い風があっても、コンピュータソフト、電子ファイル、ウェブサイトのユニバーサルデザイン化は思うように進んでいない。そのため、スクリーンリーダや音声ブラウザなどの支援技術は膨大なつじつま合わせを強いられている。したがって、開発コストもマーケット規模と比較して過度なものになっている。オーファンプロダクト(Orphan Product)などとも呼ばれ、支援技術のマーケットは極端に小さいため、支援技術開発ベンダーの経営基盤は脆弱である。「ハイリスク・ローリターンのビジネス」という構造的な困難があるといえる。

実際、日本には支援技術開発を助成する仕組みは一定程度あり、それらは米国などと比較しても比較的整っているといえるかもしれない。だが、支援技術機器を必要とする人々の平均購買力は概して低いうえに、支援技術機器はどうしても一般の機器に比べて高額となる。そこで利用者を支援する制度が必要となるが、我が国には日常生活用具や補装具を給付する事業はあるものの、ドイツやオランダなどの欧州諸国の制度と比較すると不十分である。擬似的にであれ、一定の開発力(開発資金)を有する開発ベンダーと、一定の購買力を有するユーザのいるマーケットを構築する必要がある。

ユニバーサルデザインから遠い領域ほど、支援技術はアクセシビリティを実現するためにつじつま合わせを駆使しなければならない。しかし、つじつま合わせは、不正確、不安定、短命、高コストという性質を帯びる。また、それはしばしば、本来別の目的で用意されている方法の流用、転用、サポート外の方法を用いざるをえない。さらには、認識系の技術のように、完全に正確な処理は原理的に不可能であるような技術に過度に依存せざるをえない。

ユニバーサルデザインという発想を持たない分野に出版がある。ここでは、支援技術はたとえ不正確であっても、OCRのような文字認識技術に頼らざるをえない。放送分野もユニバーサルデザインという発想からは遠い領域である。そのため、地上波デジタル放送のアクセシビリティ(データ放送、電

子番組表、番組参加、字幕放送など)の実現は、またもやつじつま合わせをせざるをえない。

電子情報通信分野のユニバーサルデザインはマシンアンダスタンダビリティと言い換えてもよい。マシンアンダスタンダビリティが実現するところでは、つじつま合わせは不要となる。マシンアンダスタンダビリティが実現すると支援技術の性能は飛躍的に向上する。

コンピュータは画面に情報を視覚的に提示し、人はポインティングデバイスで「選択」する。今日の人とコンピュータの相互作用は大部分がこの反復である。これが GUI (Graphical User Interface) と呼ばれる人と機械のインターフェースである。今日のスクリーンリーダは、端的にいえばこの GUI を AUI (Auditory User Interface) = 音声ユーザインターフェースや BUI (Braille User Interface) = 点字ユーザインターフェースに変換する作業を行うソフトウェアである。GUI が一望可能性という視覚情報の特性を利用し、画像とテキストを適切に配合して場面とフォーカスを提示するのに対し、AUI や BUI は聴覚情報や触覚情報の特性である揮発性、シーケンシャルな情報の提示に配慮するとともに、音声情報や触覚情報の言語処理可能性を生かして情報を構造化し言語的に提示しようとする (図 2)。

だが、低マシンアンダスタンダビリティという条件下では、この GUI-AUI 変換、GUI-BUI 変換は不十分なものにとどまらざるをえない。理想的には、ユーザインターフェースはユーザが自由に選択できるようにならなければならない (図 3)。それを可能にするのは高マシンアンダスタンダビリティであり、それによってもたらされるサービスやタスクとユーザインターフェースの分離である。(ただし、複雑なサービスやタスクほど切り離しのために開発しなければならない技術は高度なものとなり、開発コストも巨額になる。)

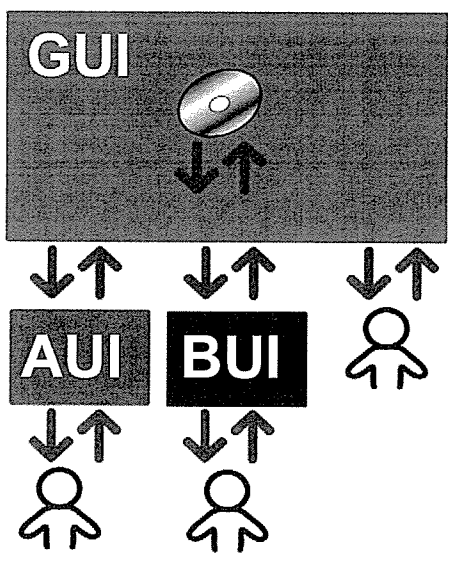

図 2 GUI はユニバーサルデザインではない

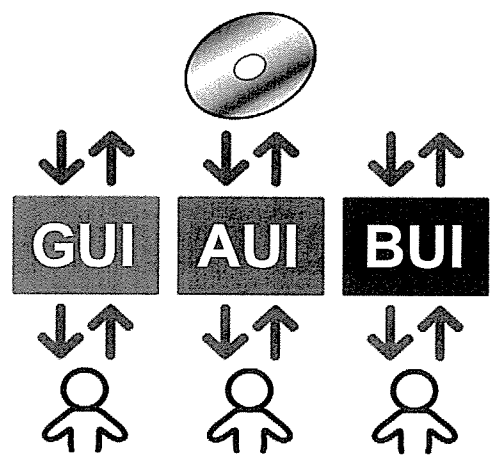

図 3 ユーザインターフェースの多文化主義

もっともサーバ・クライアントモデルの相互作用では、このサービスやタスクとユーザインターフェースは原理的に独立している。たとえば、ウェブサーバとウェブブラウザは実体としても分離して