PICKLES情報キオスクでのメッセージ形式の統一とその評価

東京工業大学 理学部
情報科学科
松岡保静
(学籍番号 9526761)

平成10年度学士論文
指導教官 大野 浩之 講師

平成11年2月9日

目 次

図 目 次

第1章　はじめに

コンピュータシステムにおけるメッセージには、アプリケーションと利用者の間で交換されるメッセージや、アプリケーション同士の通信に用いられるメッセージがある。インターネット上にはさまざまなアプリケーションが存在し、ネットワークを介したメッセージの伝達・交換が広く行われている。しかし、多くの場合これらのメッセージの形式は特定のアプリケーションに依存した形式であり、他のアプリケーションとのメッセージの伝達・交換は困難である。また、メッセージには世界各国の言語や文字の扱いに関する問題点もある。この問題点に対しては「地域化」と「国際化」の 2 つのアプローチがある。「地域化」とは、プログラムを、特定の地域の言語や文字を扱えるように適応することである。それに対し、同一のプログラムで世界各国の言語や文字を取り扱う試みを「国際化」という。インターネットは世界各国を結び、さまざまな言語による情報が世界規模で流通している。インターネット上でのメッセージ伝達・交換において、「地域化」のアプローチでは、メッセージが特定の地域での利用に限定されてしまう問題がある。

大野研究室では、インターネットを「いつでも、どこでも、だれでも」利用できる環境の構築を目指し、公衆情報端末である PICKLES 情報キオスク [1][2] の開発をすすめている。PICKLES 情報キオスクは、利用者の利便性を向上させるために、さまざまなサービスを提供している。しかし、これらのサービスは独立したメッセージ形式を用いている。例えば、アプリケーションが出力するログの形式はそれぞれ異なっている。さまざまなサービスを統合し、よりよい環境を構築するためには、メッセージの形式を標準化し、特定のサービスに依存しない汎用的な形式にする必要がある。また、PICKLES 情報キオスクが世界中に設置され、互いにメッセージの伝達・交換をする場合、利用者が言語や文字の制約を受けずにメッセージを利用できる必要がある。

本研究では、これらの問題を解決するために、メッセージ形式統一モデルを提案し、その上で用いる PICKLES 標準メッセージ形式を定義する。本モデルでは、アプリケーション同士の通信は、全て統一化された形式のメッセージでやりとりされる。一方、アプリケーションと利用者との入出力では、メッセージゲートウェイを用意し、メッセージを利用者側に適切な形式に変換して表現する。本モデルの導入による予想される効果として、異なるアプリケーション同士の柔軟な接続と、利用者に対する多様な情報表現手段の提供が考えられる。

本モデルの有効性を実証するために、PICKLES 情報キオスクでのアプリケーションを修正し、メッセージ形式を PICKLES 標準メッセージとして統一する。また、アプリケーションと利用者でのメッセージ交換におけるメッセージゲートウェイとして MFX システムを実装し、これを用いて実験を行う。

本論文の構成は次の通りである。まず第 2 章で、メッセージの表現形式における一般的

な問題点を述べ、メッセージ形式統一の必要性を述べる。第3章では、PICKLES情報キオスクでのメッセージ形式の現状を説明し、実際にさまざまなメッセージをPICKLES標準メッセージとして統一する。第4章では、メッセージ形式統一モデルを提案し、そのモデルのPICKLESへの導入について述べる。第5章では、メッセージ形式統一モデルを適用したさまざまなアプリケーションによる実験について述べる。第6章では本研究における考察について述べる。第7章で今後の展望について述べ、第8章でまとめを述べる。

第2章　コンピュータシステムにおけるメッセージの表現形式

本章では、コンピュータシステムにおけるメッセージ表現形式の問題点を指摘する。同様に、PICKLES 情報キオスクでのメッセージ形式の問題点を指摘し、メッセージ形式統一の必要性について述べる。

2.1　メッセージ表現形式の現状

コンピュータシステムにおけるメッセージの種類には、アプリケーションから利用者へ出力されるメッセージや、アプリケーション同士の通信に用いられるメッセージがある。本節では、標準化および国際化という視点から、現状のメッセージ形式の問題点について述べる。

2.1.1　メッセージの標準化

コンピュータネットワークの普及に伴い、ネットワークを介したメッセージの伝達・交換が行われるようになってきた。ネットワーク上には、さまざまなサービスが存在している。メッセージの形式が特定のサービスに依存している場合、そのメッセージは特定のサービスでの利用に限定され、ネットワークの利点を十分に活かせない。WWW が爆発的な普及を遂げたように、ネットワークの利点は、それに接続するあらゆるシステムやその上で提供されるサービスが、メッセージを伝達・交換できることである。あらゆるサービス間でメッセージの伝達・交換を実現するには、メッセージの標準化が重要になる。

例えば、商取引機構においては、EDI(Electronic Data Interchange) [1] と呼ばれる機構が盛んに展開され、組織の枠を越えた電子情報の交換がすすめられている。この電子情報の交換は、多くの場合メッセージの標準化を前提に行われている。とくに米国では、企業間で、製品の設計から製造、流通、保守に至るライフサイクル全般にわたる各種情報を標準化し、ネットワークを介して交換・共有する CALS(Commerce At Light Speed)[3] という構想がすすめられている。このような標準化は、企業間の商取引機構だけではなく、コンピュータシステムを利用するあらゆる分野において必要な技術であると考えられる。

また、サリュテーション・コンソーシアム [4] では、オフィス機器とコンピュータの間でデータのやりとりや制御を可能にすることを目指している。各機器には、サリュテーショ

[1] 企業間で電子情報交換を行うシステム

ン・マネージャと呼ばれる小さなプログラムを搭載し、オフィス機器やコンピュータ間でのデータのやりとりは、サリュテーション・プロトコルで標準化されたメッセージを用いて行われる。現在、多くの企業がサリュテーション・コンソーシアムに参加し、オフィス機器における共通のアーキテクチャの開発をすすめている。

このように、コンピュータネットワークの普及によって、あらゆる分野でのメッセージの標準化が求められている。

2.1.2 メッセージの国際化

インターネット上のアプリケーションが扱うメッセージ形式の問題点のひとつに、世界各国の言語や文字の扱いに関する問題点がある。例えば、我々は、電子メールやネット・ニュースで日本語を当り前のように使用しているが、それは「日本語化された環境」でこれらのサービスを利用しているためである。同様に、世界各国でも、自国語と英語を扱える環境を築く試みが行われている。これを「各国語化」あるいは「地域化」と呼ぶ。それに対し、単に自国語と英語の 2 つの言語だけでなく、複数の言語を取り扱うための試みを「国際化」という。国際化された環境を提供すれば、利用者は自由に好きな言語や文字を扱えるようになる。コンピュータシステムにおけるメッセージも国際化されれば、言語の違いにとらわれないメッセージの伝達・交換が可能になる。

インターネットの普及にともない、アプリケーションやメッセージの国際化が注目されてきている。国際化には、以下の要素を考慮する必要がある。

言語・文字集合

文字集合とは、言語情報を交換する際の要素となる個々の文字に符号を割り当てたものの集合である。文字集合の例として以下のものが挙げられる。

- ISO 646:1991 USA (ASCII と同様)
- JIS X 0208-1990 (日本の国内規格)
- GB 2312-1980 (中国の国内規格)
- KS C 5601-1992 (韓国の国内規格)
- CNS 11643-1986 (台湾の国内規格)

このように世界各国のさまざまな文字を表現するために、数多くの文字集合が定義されている。国際化された環境を作る場合は、これらの文字集合を利用者が自由に切替えられる必要がある。複数の文字集合の中から、利用者が自由に文字集合を選んで利用するモデルをロケール [5] という。このようなアプローチで国際化されているアプリケーションとして、エディタとして広く使われている Mule (MULtilingual Enhancement to GNU Emacs)[6] が挙げられる。この Mule 上で電子メールやネットワーク・ニュースを読み書きするアプリ

ケーションを起動することにより、これらのアプリケーションを国際化された環境で利用できる。また、メッセージ形式の国際化としてメッセージカタログ [5] という機構がある。これを用いると、メッセージをプログラムと分離し、ロケールに応じて切替えられる。

一方、上記のアプローチとは別に、世界中の全ての文字を含む単一の文字集合の定義を試みたものが、ISO/IEC 10646 [7] である。これにより、ロケールでは不可能であった、複数の言語を同時に扱うモデルが可能になる。今後は、この ISO/IEC 10646 を利用した国際化アプリケーションの開発が望まれている。

エンコーディング

エンコーディングとは、前節で述べた文字集合を表現するために使用される技法である。インターネットで日本語を表現するために一般的に用いられているのが、RFC-1468[8] で定義されている ISO-2022-JP[9] と呼ばれるエンコーディング方式である。現在は、複数の言語を扱う場合、これらの言語の文字集合に対して、エンコーディング方式を切替えることによって対応している。しかし、ISO/IEC 10646 を用いた場合は、複数の文字集合を切替える必要がないため、ISO-2022 のようなエンコーディングは不要である。現在は、ISO/IEC 10646 のエンコーディング方式として、ISO/IEC 10646 が定めている UCS-2、UCS-4、UTF-8、UTF-16 等がある。現在、国際化のアプローチとしては、UTF-8 を用いたアプリケーションの開発やメッセージの国際化が注目されている。

2.2 PICKLES 情報キオスクでのメッセージ形式

PICKLES プロジェクトは、1995 年から大野研究室ですすめられている研究プロジェクトである。PICKLES プロジェクトでは、公衆端末である PICKLES 情報キオスクを随所に設置することで、場所の制約をうけずにインターネットを利用できる環境を実現することを目指している。

本節では、PICKLES 情報キオスクのメッセージ形式の現状を分析し、メッセージ形式統一の必要性について述べる。

2.2.1 現状の分析

PICKLES 情報キオスクでは、利用者の利便性を向上させるために、さまざまなサービスを提供している。例えば、大学内で快適な情報サービスを提供するためのシステムである CITRUS[10] がある。これは、PICKLES 情報キオスクと ChipCard(カード型携帯端末) を組合せ、大学のように地理的に広い空間でも、大学内の誰もが平等にサービスを受けることができる環境を提供するシステムである。利用者はカード型携帯端末を持ち歩くことによって、大学内のあらゆる PICKLES 情報キオスクでのメッセージ交換が可能になる。一方、PICKLES 情報キオスクの運用を円滑にするために、管理者の面でもさまざまなサービスを提供している。例えば、ネットワークトラフィックを音声によって表現する stetho

システム [11][12] や、ネットワークワームによってネットワーク管理を行うための NMW システム [13][14] が提供されている。

このようなサービスにおけるメッセージの形式は、現在のところ、ある特定の利用形態に専用的な形式になっているものが多い。そのため、メッセージを他のアプリケーションに応用したり、他の利用形態によって利用することが困難である。

また、PICKLES 情報キオスクが世界中に設置され、それらが互いにメッセージの伝達・交換をする場合、利用者が言語や文字の制約を受けずにメッセージを利用できる必要がある。

2.2.2 メッセージ形式統一の必要性

/* この節は全て書き直します。 */ これまで、PICKLES プロジェクトでは、インターネットを「いつでも、どこでも、だれでも」利用できる環境を構築してきた。この環境を実現するために、PICKLES ではさまざまなサービスを提供していることは前節で述べた通りである。今後は、これらのサービスがお互いに連携し、よりよい環境を構築することが重要であると考えられる。例えば、NMW システムのメッセージを stetho システムが音で表現するという形態も考えられる。そのためには、各サービスが用いているメッセージの形式を統一的なものにし、それらのメッセージを各サービス間で共有できるようにすることが必要である。また、利用者の目的に応じて柔軟に対応できる汎用性の高いメッセージにすることも必要である。これにより、PICKLES 情報キオスクが提供する環境は、利用者にとっても管理者にとっても、今まで以上に使いやすく、また応用範囲も広がると考えられる。

2.3 関連研究

前節で述べたように、さまざまなサービスやアプリケーションが相互接続し、お互いにメッセージを伝達・交換するためには、メッセージ形式を統一する必要がある。そこで本節では、メッセージ形式の統一および標準化についての関連研究について述べる。

メッセージ形式の統一および標準化はさまざまな分野で研究されている。例えば、医薬情報学の分野では、ICH(International Conference on Harmonization of Technical Requirements for Registration of Pharmaceuticals for Human Use) というプロジェクトの M2 というワーキンググループにより、医薬品規制情報の標準化を行っている。[15] この M2 ワーキンググループでは、医薬品規制情報として副作用報告を取り上げ、副作用報告のメッセージ形式としては、SGML 言語を用い、副作用報告の SGML DTD(Document Type Definition) の開発を進めてきた。

また、臨床検査の分野では早くから標準化の取り組みがなされ、臨床検査データ交換については、「臨床検査データ交換規約」[16] が 1993 年に発表された。日本保険医療情報システム工業会 JAHIS 臨床検査システム委員会では、発足以来、医療情報の標準化動向を留意し、世界に幅広く通用する標準化に取り組んでいる。

さらに、いくつかの全国的あるいは国際的な研究プロジェクトや実験観測では、標準の

メッセージ形式が使われた。例えば以下のものが挙げられる。

- GWE(全球大気実験) の観測データ
- MONEX(モンスーン実験) の観測データ
- ALPEX(山岳実験) の観測データ
- WOCE(世界海洋循環実験) での国際海洋データ交換標準形式 (IODE)

メッセージの交換、特に通信による交換では、標準を守ることが義務となっていることがある。例えば、気象、海洋、地震等のデータを交換するために WMO が組織している全球通信システム (GTS) では特定の標準メッセージ形式が使われている。

第3章　PICKLES情報キオスクでのメッセージ形式の統一

本章では、PICKLES 情報キオスクでのメッセージ形式の統一に必要な条件およびメッセージ形式統一の手段について述べる。また、実際に、いくつかのメッセージを統一した例について述べる。

3.1　メッセージ形式統一に必要な条件

メッセージ形式を統一するためには、単に形式を統一するだけでなく、コンピュータが機械処理できるレベルまで形式化する必要がある。この場合、メッセージの大部分がコンピュータの機械処理に適したレベルまで形式化されていても、一部に形式化されていないものが入り込んでくると、もはやそのレベルまで形式化されているとはいえなくなる。したがって、メッセージの徹底した形式化が必要になる。

また、前章で述べたように、特定の利用者だけが利用できるメッセージ形式ではなく、サービスを利用する全ての利用者が利用できるメッセージ形式にする必要がある。つまり利用者のさまざまな目的に対応できるような柔軟で汎用的なメッセージ形式にする必要がある。さらに、メッセージがどのような情報を含んでいるのか、また、メッセージの発生場所や発生時刻等を示すための情報も表現する必要がある。このようなメッセージ自体についての情報を属性という。さまざまなメッセージを統合して扱う場合、メッセージの属性は重要な情報になる。

さらに、今後はアプリケーションの国際化が重要な課題となっているため、メッセージに用いる文字集合やエンコーディング方式に関しても国際化に対応するべきである。

3.2　PICKLES標準メッセージ形式の提案

これまで、メッセージ形式の問題点として、言語やエンコーディング、他のアプリケーションとの連携など、さまざまな視点から述べてきた。PICKLES 情報キオスクにおいて、これらの問題点を解決する手段のひとつに、既存のアプリケーションを全て修正することが挙げられる。本研究では、この既存のアプリケーションの修正に伴い、さまざまなアプリケーションのメッセージ形式を PICKLES 標準メッセージ形式として正規化し、統一することを提案する。そこで、メッセージ形式を正規化し、統一する手段として、汎用的なデータ記述言語である XML(Extensible Markup Language)[17] を用いることを考えた。XML

はさまざまな文字コードを使用できる。そのため、図??のようにどの文字コードを使っているのかを教える必要がある。

- メッセージ発生場所
- メッセージ発生時間

XMLは、タグによって要素ごとに属性を定義できる。図3.2は、メッセージに属性を付与した例である。

```
EUC-JPの場合
 <?xml version="1.0" encoding="EUC-JP"?>

UTF-8の場合
 <?xml version="1.0" encoding="UTF-8"?>
```

図 3.1: メッセージの属性

XMLはエンコーディング方式として、UTF-8やUTF-16を使えるため、メッセージの国際化にも対応できる。また、XMLは、従来のデータ記述言語とは異なり、固定化された情報表現にとどまらず、利用者側で柔軟に情報の見方を選択できるという特徴がある。これは、XMLが単に情報の形式や構造だけを表現するのではなく、情報の内容や意味も表現できるためである。現在、XMLを利用するためのライブラリも多く開発されている。特に、XML文書を解析するXMLパーサは数多く開発されている。したがって、XMLで表現されたメッセージは、解析が容易であり、アプリケーションにとって利用しやすい形式であると考えられる。

メッセージは、属性と内容の2つに分けられる。PICKLES標準メッセージ形式のメッセージは、メッセージの属性として以下の要素を含む。

- メッセージ発生場所
- メッセージ発生時間

XMLは、タグによって要素ごとに属性を定義できる。図3.2は、メッセージに属性を付与した例である。

3.3 メッセージ形式の統一化手法

本節では、実際にいくつかの例を挙げて、メッセージ形式をPICKLES標準メッセージ形式として統一する手法を述べる。

```
<?xml version ='1.0'?>
<message hostname="kodaiko" timestamp="918705402.404730">
        .
        .
        .
</message>
```

図 3.2: メッセージの属性

3.3.1 UNIX コマンドの出力の統一

PICKLES 情報キオスクの利用者は、主に UNIX コマンドを用いて作業をすすめる。また、管理者は管理作業における監視情報の収集手段として UNIX コマンドを用いる場合が多い。UNIX コマンドには、さまざまな種類があるが、代表的なコマンドとしては次のものが挙げられる。

利用目的	コマンド名
プロセス管理	ps, kill, time, top
ファイル管理	cp, mv, ls, chmod, find, file
テキストファイル操作	awk, sed, cat, diff, grep, wc
ネットワークサービス	rlogin, rcp, rsh, telnet, ftp, mail
環境設定	passwd, groups, tty, script, stty,
システム管理	df, du, iostat, vmstat, who
ネットワーク管理	netstat, traceroute, ping, ifconfig,

このようなコマンドを UTF-8 の文字コードを用いて国際化に対応するためには、各々のコマンドのソースプログラムを書き換える必要がある。そこで、PICKLES 情報キオスクのコマンドの国際化への対応に伴い、出力形式の統一と正規化を試みる。従来のコマンドの出力は、各々個々のフォーマットにしたがって表現され、その形式は統一されてなく、なかにはアプリケーションによる解析が困難な形式のものもある。例として、df の出力結果を挙げる。図 3.3 は、従来の df の出力結果で、それを XML によって形式化した結果が図 3.4 である。

このように、出力結果をタグによって構造化することで、この出力結果を利用するアプリケーションは、それぞれの要素がどういう関係にあるのかを知ることができる。また、データを形式化しているタグには、データの意味を示すタグ名がつけられており、属性も

```
Filesystem   512-blocks     Used    Avail Capacity  Mounted on
/dev/wd0a         32078    29350     1124    96%    /
/dev/wd0h       1584108  1045544   459358    69%    /usr
/dev/wd0e         31166      606    29000     2%    /etc3
/dev/wd0g       1901148   539190  1266900    30%    /local
/dev/wd0f        190078    31340   149234    17%    /var
```

図 3.3: df の出力結果

```
<?xml version="1.0" ?>
<!DOCTYPE df SYSTEM "/etc/dtd/df.dtd">
<df hostname="kodaiko.ohnolab.org" time="918708291.459108">
<filesystem>
<mounted>/</mounted>
<device>/dev/wd0a</device>
<block blocksize="512-blocks">
<total>32078</total>
<used>29350</used><avail>1124</avail>
<capacity>96</capacity>
</block>
</filesystem>
<filesystem>
<mounted>/usr</mounted>
<device>/dev/wd0h</device>
<block blocksize="512-blocks">
<total>1584108</total>
<used>1045544</used><avail>459358</avail>
<capacity>69</capacity>
</block>
</filesystem>
        .
        .
        .
```

図 3.4: df の出力結果 (PICKLES 標準メッセージ形式)

付与されているので、アプリケーションはデータがどういう意味を持っているのかを知ることができる。

UNIX コマンドのソースプログラムは公開されているので、各コマンドのソースプログラムに、XML 形式で出力するモジュールを加えることによって、出力結果を、従来の形式で出力することも XML 形式で出力することもできるようになる。

3.3.2 システムのログ形式の統一

システムに障害がおきたときには、管理者はログ情報から原因を探す場合がある。PICKLES 情報キオスクでのシステムのログ情報は、テキスト形式で表示され、ログ情報には、イベントが発生した日時、プログラム名や、イベントの内容が含まれている。これらの情報は、日時やプログラム名を検索キーとして検索することができるが、ログ情報を XML 形式で表現することによって、ログの重要度やログを出力したアプリケーションの種類等によっても検索でき、必要な情報をさらに効率的に取得できると考えられる。

アプリケーションが出力するログ情報は、主に syslog というライブラリ関数を利用して出力している。したがって、syslog ライブラリ関数のソースプログラムに、XML 形式でログを出力するモジュールを追加することによって、ログ情報を従来の形式と XML 形式の両方の形式で出力できる。(図 3.5) はシステムのログを XML 形式で出力した例である。これにより、システムのログ情報をアプリケーションが容易に解析できるようになり、必要なログ情報の検索も効率的に行えるようになる。

3.3.3 周辺機器のメッセージ形式の統一

現在、PICKLES 端末は、PICKLES プロジェクトをすすめている大野研究室のメンバーの日常の活動を行うプラットホームとしても利用されている。大野研究室では、PICKLES 端末から、電光掲示版やオーディオ機器、家電製品等、さまざまな周辺機器を制御する研究も行われている。そこで、これらの周辺機器とやりとりするメッセージの形式を統一することを考えた。実際に周辺機器のメッセージ形式の統一の例として、MIDI 対応のオーディオ機器を取り上げる。図 3.6 はデジタルミキサーの設定データを XML 形式に変換した例である。

このように、ミキサーの設定データの形式を統一することにより、ベンダーが異なっていても、同じアプリケーションによって、ミキサーとのメッセージ交換ができる。

```
<syslog>
<facility>kernel</facility>
<priority>notice</priority>
<time><year>1999</year><month>02</month><day>01</day>
<hour>10</hour><minutes>33</minutes><second>29</second></time>
<progname>login</progname>
<message>ROOT LOGIN (root) ON ttyv0</message>
</syslog>
<syslog>
<facility>kernel</facility>
<priority>info</priority>
<time><year>1999</year><month>02</month><day>01</day>
<hour>13</hour><minutes>18</minutes><second>34</second></time>
<progname>dhcpc</progname>
<pid>320</pid>
<message>Got DHCPACK (IP = 131.112.57.43)</message>
</syslog>
```

図 3.5: システムのログ (PICKLES 標準メッセージ形式)

```
<?xml version="1.0" hostname="tascam-mixer" time="918708836.977763"?>
<!DOCTYPE snapshot SYSTEM "/etc/dtd/snapshot.dtd">
<snapshot preset="101">
<master_volume>108</master_volume>
<channel ch="1">
<fader>105</fader><pan>64</pan>
</channel>
<channel ch="2">
<fader>105</fader><pan>64</pan>
</channel>
<channel ch="3">
<fader>103</fader><pan>64</pan>
</channel>
```

図 3.6: ミキサーの設定データ (PICKLES 標準メッセージ形式)

第4章　メッセージ形式統一モデルの提案

本章では、メッセージ形式統一モデルを提案し、そのモデルのPICKLESへの導入について述べる。

4.1　メッセージ形式統一モデル

コンピュータシステムにおけるメッセージの形式を統一する発想は、アプリケーション間の連携や、利用者側での表現体裁の自由な選択を実現したいという所にある。本節では、このような機構を実際に実現するためのモデルについて述べる。

アプリケーション間のメッセージ交換

アプリケーション同士のメッセージ交換では、全て統一された形式のメッセージでやりとりされる。したがって、アプリケーション間のインタフェースを統一する必要がある。アプリケーションの作成にあたって、統一された形式に従ったメッセージを出力するメッセージ作成モジュールとそのメッセージを解析するメッセージ解析モジュールを実装する必要がる。メッセージの解析には、次の2つの過程が考えられる。

- 形式的解析
- 意味的解析

メッセージの形式的解析とは、メッセージに含まれるデータの構造や、データ同士の関係を解析するものであり、メッセージの内容までは解析しない。この形式的解析をするモジュールは、全アプリケーションで共有できる。一方、メッセージの意味的解析とは、メッセージの内容まで解析し、データがどういう意味を持っているのかを解析するものである。この意味的解析をするモジュールは、メッセージ内容のスキーマごとに別々に用意する必要がある。

アプリケーションと利用者との間でのメッセージ交換

アプリケーションと利用者間でのメッセージ交換は、両者の間にメッセージゲートウェイを用意し、メッセージ変換をする。メッセージゲートウェイは、アプリケーションから出力される統一化された形式のメッセージを解析し、利用者の環境や目的に適切な形式に

変換して表現する。この機構を offramp 機構という。また、利用者が入力するメッセージを統一化された形式のメッセージに変換する。この機構を onramp 機構という。

メッセージゲートウェイは、利用者に出力するメッセージをプログラムの外部に用意する。例えば、世界各国のメッセージを用意すれば、利用者はメッセージの言語を自由に選択でき、出力メッセージを国際化できる。また、音声メッセージ等を用意すれば、利用者はテキストによるメッセージだけでなく、音声メッセージも併用でき、多様なメディアでの情報表現が実現できる。

4.2 PICKLES への導入

本節では、メッセージ形式統一モデルを PICKLES へ導入し、メッセージ形式を統一した PICKLES の上で動作するアプリケーションの実装とメッセージゲートウェイの実装について述べる。

4.2.1 アプリケーションの実装

PICKLES 上で動作するアプリケーションは、図 4.1 のように、PICKLES 標準メッセージを出力するメッセージ作成部と、解析するメッセージ解析部を持つ。

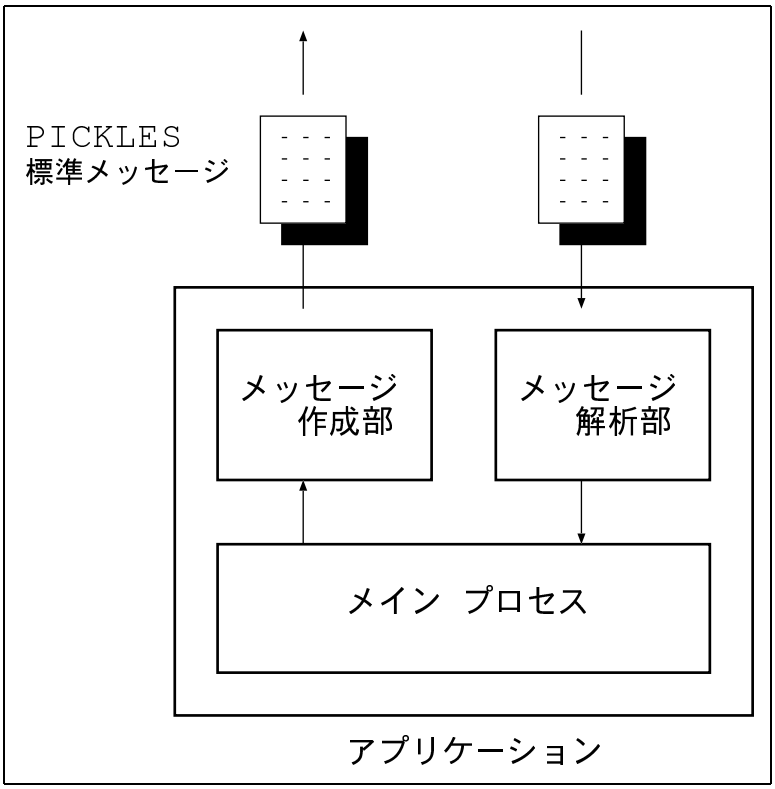

図 4.1: アプリケーションの構成

メッセージ作成部やメッセージ解析部の実装には、既に開発されているXMLライブラリを利用した。現在、XML形式のメッセージの作成や解析をするためのライブラリとして、さまざまなXMLツールが開発されている。特に、XMLのメッセージを解析するXMLパーサは数多く開発されている。例として、以下のものがある。

- IBM XML for Java
  IBMのalpha Worksで公開しているJavaによるXMLパーサである。UTF-8、UTF-16にも対応しており、XMLの仕様に忠実なパーサである。解析だけでなく、XML文書作成のライブラリも含む。

- Microsoft XML Parser for java
  Microsoftが公開しているJavaによるXMLパーサである。UTF-16には対応していない。

- Expat(XML Tok)
  James Clark氏が公開しているCによるXMLパーサである。UTF-8、UTF-16にも対応しており、perlやMozillaに組み込まれる予定。

上記以外のXMLパーサも数多く開発されているが、本システムの実装には、IBM XML for Javaを利用した。これを利用することにより、アプリケーションの作成者は、メッセージの作成や解析の部分を容易に実装できる。

4.2.2 メッセージゲートウェイMFXの実装

前節で述べたモデルにおけるメッセージゲートウェイとして、MFXシステムを実装した。MFXシステムは、あらゆるアプリケーションとの接続ができる仕組みになっている。実際には、UNIXシェルと接続し、さまざまなコマンドの出力メッセージを受け取ることを前提としている。このシステムは、アプリケーションが出力するPICKLES標準メッセージを解析し、その結果をテキストや音声等の表現手段によってマルチモーダルに表現するシステムである。図4.2はシステムの構成図である。

アプリケーションから出力されるPICKLES標準メッセージを解析し、利用者に表現する場合、そのメッセージを利用者の環境や目的に適した形式にする必要がある。そこで、メッセージ変換機能の実装に、メッセージカタログを利用する。(図4.3)

メッセージカタログは本来、サポートする各ロケールのメッセージをプログラムの外部に持ち、プログラムの再コンパイルなしで言語の選択を可能にするメカニズムである。しかし、ここでは単に出力するメッセージの言語を切替えるだけでなく、音声等も併用したマルチモーダルな表現機構を実装する。入力されてくるPICKLES標準メッセージは、XML言語によってメッセージの各要素が構造化されているため、メッセージに含まれているデータの意味を理解できる。これにより、メッセージの表現をマルチリンガルだけでなく、マルチモーダルにもできるようになる。

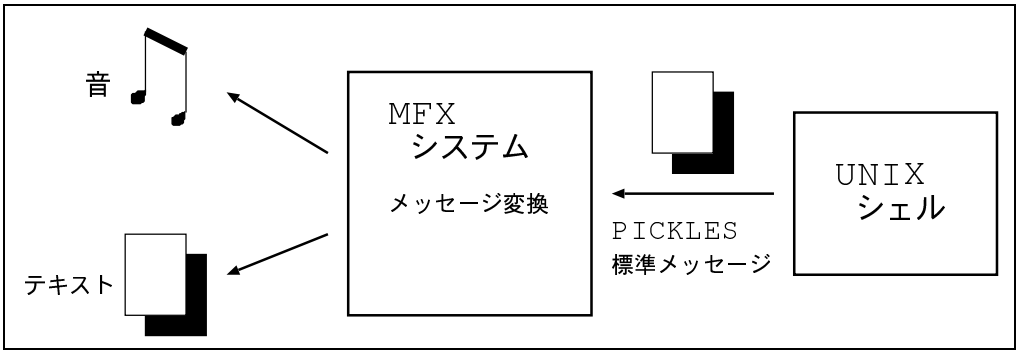

図 4.2: システム構成図

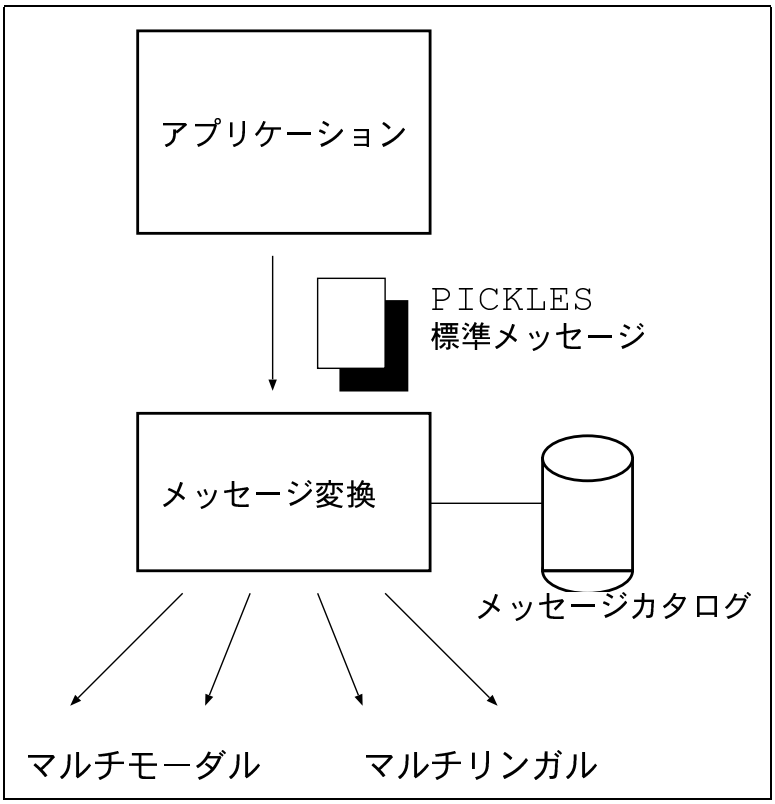

図 4.3: メッセージの変換

第5章　考察

本章では、メッセージ形式の統一についての考察およびその効果について述べる。

5.1　メッセージ形式統一の問題点

本節では、メッセージ形式統一における問題点について述べる。

メッセージ形式統一にかかるコスト

本研究で、実際にメッセージ形式を統一したものは、代表的なUNIXコマンドのうちの一部と、システムのログ、MIDI対応デジタルミキサーの設定データである。メッセージ形式を統一するには、既存のコマンドやアプリケーションのソースプログラムを修正する必要がある。また、実際にメッセージを利用者の希望に応じて変換するためには、変換後のメッセージをメッセージカタログ等を利用して、プログラムの外部に用意する必要がある。UNIXコマンドは数百種類あり、これらのソースプログラムを全て書き換えることは、膨大な時間と労力を必要とする。しかし、実際にUNIXの国際化として、既存のUNIXコマンドを全てUTF-8の文字コードで表現するように修正することを考えた場合、単に、言語の問題だけを考えるのではなく、前章までに述べたように、コマンドが出力するメッセージ形式を統一し、マルチリンガルにもマルチモーダルにも対応させることは有効であると考えられる。

形式化が困難なメッセージ

本研究では、UNIXコマンドの一部、システムのログ、MIDI機器の設定データをPICKLES標準メッセージとして形式化した。しかし、メッセージの中には形式化や構造化のスタイルを容易に見つけ出すことのできないものもある。例えば、UNIXコマンドでは、lessやmore等のユーザ端末の制御のためのコマンドが挙げられる。これらのコマンドは、端末の制御文字を出力する。このような形式化が困難なメッセージの扱いに関する問題も検討する必要がある。

DTD の集約

本研究では、メッセージの内容としては、アプリケーションごとに別々な DTD を定義した。この場合、メッセージの形式的解析は共通の解析モジュールで処理できるが、メッセージの意味的解析は別々のモジュールを作成する必要があった。しかし、全てのアプリケーションにおける DTD を統合的に定義することはできないが、ある程度の DTD の集約は可能であると考えられる。例えば、ping、traceroute が出力するデータには、IP アドレス、ホスト名等、双方に共通の要素が含まれる。そのため、これらは同一のスキーマとして定義できると考えられる。このように、アプリケーションごとに DTD を定義するのではなく、あるオブジェクトに対して DTD(スキーマ) を定義し、これらの DTD を使って、アプリケーションを形式化する方が望ましい。

メッセージの XML 形式化によるデータ量の増加

XML 形式化したデータを見てみると、従来の形式によるデータに比べて、XML 形式で出力したデータは明らかにデータ量が大きくなる。これは、XML 形式で出力したデータが、従来の形式で出力したデータよりも多くの情報を含んでいるためでもある。これらの情報をネットワーク経由で受信する場合、データ量が大きいことは問題である。しかし、データを圧縮して通信する場合、この問題はある程度解決される。この通信データの圧縮方法のひとつとして、ssh[18] による暗号化通信でのデータ圧縮方法がある。ssh では、データの圧縮方法として、GNU zip を用いている。この GNU zip を用いてメッセージを圧縮した場合と、圧縮前のメッセージのデータ量を比較してみる。以下の表は、コマンドの従来の出力形式と XML での出力形式を GNU zip で圧縮した場合について比較したものである。

出力形式	圧縮前	圧縮後
df	334 byte	220 byte
df(XML 形式)	1084 byte	378 byte

出力形式	圧縮前	圧縮後
netstat	2549 byte	509 byte
netstat(XML 形式)	4749 byte	659 byte

上の表を見てみてみると、GNU zip で圧縮する前は、データ量の差が大きいが、圧縮したあとは、ある程度データ量の差は緩和される。したがって、データを圧縮して通信することによって、データ量が大きいことによるネットワークの負荷の問題は、ある程度解決できる。

5.2　メッセージ形式統一モデルの有効性

本節では、実際に PICKLES 情報キオスクでのメッセージ形式を統一したことによる効果について述べる。

マルチモーダルの実現

前章で述べた実験により、単一のメッセージを、テキストで表示したり、音声や画像で表現できることができた。これは、メッセージを特定の表現手段に依存しない汎用性のある形式に統一したためである。つまり、従来のモデルでは、多くの場合、メッセージとその表現体裁は一体化していたが、メッセージを汎用性のある形式に統一することによって、メッセージとその表現体裁の分離を実現できた。

アプリケーションや周辺機器の連携

PICKLES 上のさまざまなアプリケーションのメッセージ形式を統一することによって、それらのアプリケーション間でメッセージを共有できた。また、PICKLES 環境で制御している周辺機器のメッセージを統一することによって、それらの機器を統合的に扱えることがわかった。今まで PICKLES では、各々のサービスが独立して存在していたが、これらのサービスにおけるメッセージの形式を統一することにより、個々のサービス同士が連携し、より快適な環境を提供できると考えられる。

第6章　今後の展望

PICKLES 情報キオスクのコンセプトは、UbiquitousComputing[19] である。将来のインターネットは単にパソコンやワークステーションを繋いだネットワークではなく、家電製品や音響機材等、さまざまな機器が接続するネットワークになることが考えられる。PICKLES プロジェクトの取り組みとして、将来的には、このようなさまざまな機器からもインターネットへアクセスできるような未来のインターネットへのアクセススタイルも視野に置いている。

本論文では、UNIX コマンド、システムのログ、オーディオ機器のメッセージ形式の統一について述べた。今後は、このような未来のインターネットのアクセススタイルに対応できるように、さまざまな機器やその上で動くアプリケーションのメッセージ形式統一を考えている。また、PICKLES が世界的に普及していくためには、提供しているさまざまなサービスやアプリケーションを国際化する必要がある。今後は、言語、文字コード、日付、単位等、さまざまな問題を考慮しながら、PICKLES でのメッセージ形式の統一をすすめていくことを考えている。

第7章　おわりに

本論文では、メッセージの標準化や国際化の必要性について述べ、PICKLES 情報キオスクでのメッセージ形式の統一を提案した。実際に、いくつかのアプリケーションのメッセージを PICKLES 標準メッセージ形式として統一した。実験・運用において、これらのメッセージを他のアプリケーションへ適用したり、多種のメディアによって利用することが容易に実現できた。

今後は、PICKLES 情報キオスクにおいて、さらにメッセージ形式の統一をすすめ、その上で動くさまざまなサービスやアプリケーションが相互に連携できる土台を築き、PICKLES 情報キオスクがよりよい環境を提供できることを期待する。また、メッセージの国際化についても検討し、PICKLES が世界的に普及することを期待する。

謝辞

本論文の執筆にあたり、多くの相談にのっていただいた多田謙太郎さん、酒井淳一さんに感謝致します。また、研究生活や大学生活を送るにあたって、いろいろとお世話になった大野研究室の皆様、どうもありがとうございました。最後に、研究活動にあたって、日頃から暖かく御指導して頂いた大野浩之講師に心より感謝致します。

参考文献

[1] 木本雅彦. 大学公衆情報端末 (pickles) を用いたネットワーク利用環境の構築に関する研究. 修士論文, 東京工業大学大学院情報理工学研究科, Feburuary 1997.

[2] 木本雅彦, 大野浩之. 公衆情報端末計画〜 pickles の概要〜. 第 52 回全国大会講演論文集 (6), March 1996.

[3] *Description of CALS Environment(Draft).* http://www.ornl.gov/cals/cals_isq.html.

[4] *Salutation Consortium.* http://www.salutation.org/.

[5] 清兼義弘, 末廣陽一. 国際化プログラミング (i18n ハンドブック). 共立出版, 1998.

[6] Mikiko Nishikimi, Kenichi Handa, and Satoru Tomura. Mule: Multilingual enhancement to gnu emacs. In *In Proceedings of INET'93*, 1993.

[7] *ISO/IEC 10646-1:1993 Universal Multiple - Octet Coded Character Set(UCS) – Part1*, http://www.iso.ch/cate/d18741.html, 1993.

[8] Jun Murai, Mark Crispin, and Enk M. van der Poel. Japanese character encoding for internet messages. Rfc 1468, 1993.

[9] International Organization for Standardization(ISO). Information processing – iso 7-bit and 8-bit coded character sets – code extension techniques. Iso 2022:1986(e), 1986.

[10] 木本雅彦, 大野浩之. 学内情報システム〜 citrus の概要〜 (大会優秀賞受賞). 情報処理学会第 52 回 (平成 8 年度前期) 全国大会講演論文集 (1),pp.277-278. 情報処理学会, March 1996.

[11] 成田哲也. ネットワーク環境における音声の有効利用. 卒業論文. 東京工業大学 理学部 情報科学科, January 1995.

[12] 成田哲也. ネットワーク環境における音声の有効利用. 修士論文, 東京工業大学大学院情報理工学研究科, February 1997.

[13] 清水亮博. 寄生プログラム (ネットワークワーム) を用いたコンピュータネットワークの管理方式. 卒業論文. 東京工業大学理学部情報科学科, Feburuary 1993.

[14] 小野木渡. Nmw systemによるネットワーク管理とその評価. 修士論文, 東京工業大学大学院情報理工学研究科, February 1997.

[15] 小出大介, 大江和彦, 岡田美保子, 開原成, 廣田光恵, 村山高久. 医薬品情報の電子的交換-ichにおける標準化への取り組み. 第16回医療情報学連合大会論文集, pp. 776–777, 1996.

[16] 医療情報システム開発センター. 臨床検査データ交換規約 (暫定版). 1993.

[17] Tim Bray, Jean Paoli, and C.M.Sperberg-McQueen. Extensible markup language(xml)1.0. http://www.w3c.org/TR/REC-xml, February 1998.

[18] *SSH Protocols and Secure Shell.* http://www.ssh.fi/sshprotocols2/index.html.

[19] *Ubiquitous Computing.* http://www.ubiq.com/hypertext/weiser/UbiHome.html.

付 録A　XMLの概要

A.1　XMLの歴史

1996年に至り、W3C(World Wide Web Consortium)では、Web機構に対する適合性を持ち、かつ人間側から見てもコンピュータ的に見ても扱いやすい言語の開発を開始した。XMLはWeb上の汎用的なデータ記述言語として最初から開発されたわけではなく、もともとの目的は、1986年に制定した文書記述言語であるSGML(Standard Generalized Markup Language)をWeb上で使用できるようにすることである。

SGMLは1980年代に発生していた「デジタル化された文書同士の互換性がない」という問題を解決した。文書交換の標準形式であるSGMLの特徴は次のとおりである。

- 通常使用するテキスト(ASCIIやShift-JIS等)で文書を表現する
- 基本的にはアプリケーション固有のレイアウト情報を入れない
- 文書を要素に分解し、タグを使って表現される要素の構造として文書を定義する
- 文書構造を定義する文書型定義(DTD)を自分で作成できる(タグを自分で定義できる)

SGMLは自由度が高く、インターネット上で電子化文書を交換・配布するのに向いている。しかし、SGMLをWeb上で使用するには次の問題があった。

- 処理パフォーマンスが悪い(DTDとの照合や省略タグの復元を行うため)
- 規格書の文法定義が複雑で難しい

XMLではSGMLをスリムにして不要な機能を削除し、Web上で使用するのに必要な機能を追加した。また、XMLの言語定義は、言語仕様の定義で一般的に仕様されているバッカス記法(BNF:Backus Naur from)に改められ、わかりやすくなった。

XML開発の概略は次のとおりである。

年月	事項
1996年7月	W3Cの作業部会(WG)活動開始
1996年11月	XML規格ドラフト(第1版)
1998年2月	XML1.0勧告(XML1.0 W3C Recommendation)公表

A.2　XMLが拓く新たなアプリケーション

インターネットの最大の利用法は、WWWで世界中から情報を得ることと、世界に向けて情報を発信することである。情報はHTMLによって表現され、配布される。しかし、インターネットやWWWはHTML文書を配布するだけのものではない。まだまだ多くの可能性が残されていると考えられる。XMLはHTMLの壁を乗り越え、今までになかった新しいアプリケーションをもたらすことが期待されている。それらは大きく次の3つに分けられる。

- 複数のデータベースをWWWによって連携させるアプリケーション
- WWWクライアント側で計算を行うアプリケーション
- WWWエージェントが情報を選択するアプリケーション

このようなアプリケーションが開発されることにより、WWWは、単なる文書の配送手段から、業務システムをネットワーク上で実現するためのインフラストラクチャへと変貌する。

参考文献

- XML入門 ： 村田 真 ： 日本経済新聞社
- XML完全解説：XML/SGMLサロン：技術評論社

付 録B　各種メッセージのDTD

B.1　netstatのDTD

```
<!ELEMENT netstat (interface|routing)*>
<!ATTLIST netstat hostname CDATA #IMPLIED>
<!ELEMENT interface (mtu?, link?, (netaddr|ipaddr)*, linkaddr?,
                                input?, output?, coll?, drop?, time?)>
<!ATTLIST interface name CDATA #REQUIRED>
<!ELEMENT input (pkts,errs,bytes?)>
<!ELEMENT output (pkts,errs,bytes?)>
<!ELEMENT mtu (#PCDATA)>
<!ELEMENT netaddr (#PCDATA)>
<!ATTLIST netaddr type CDATA #IMPLIED>
<!ELEMENT ipaddr (#PCDATA)>
<!ATTLIST ipaddr type CDATA #IMPLIED>
<!ELEMENT linkaddr (#PCDATA)>
<!ELEMENT pkts (#PCDATA)>
<!ELEMENT errs (#PCDATA)>
<!ELEMENT bytes (#PCDATA)>
<!ELEMENT coll (#PCDATA)>
<!ELEMENT drop (#PCDATA)>
<!ELEMENT time (#PCDATA)>

<!ELEMENT routing ((default|desthost|destnet), (gateway|link),
                                        flag?, netif?,expire?)>
<!ATTLIST routing type (inet|inet6|ipx|appletalk|ns) #IMPLIED>
<!ELEMENT default EMPTY>
<!ELEMENT desthost (#PCDATA)>
<!ELEMENT destnet (#PCDATA)>
<!ATTLIST destnet netmask CDATA #REQUIRED>
<!ELEMENT gateway (#PCDATA)>
<!ATTLIST gateway flags CDATA #IMPLIED>
<!ELEMENT link (#PCDATA)>
<!ATTLIST link flags CDATA #IMPLIED>
<!ELEMENT flag (#PCDATA)>
<!ELEMENT netif (#PCDATA)>
<!ELEMENT expire (#PCDATA)>
```

B.2 dfのDTD

```
<!ELEMENT df (filesystem)*>
<!ATTLIST df hostname CDATA #IMPLIED>

<!ELEMENT filesystem (device,block,inode?)>
<!ATTLIST filesystem mounted CDATA #IMPLIED>
<!ELEMENT device (#PCDATA)>
<!ELEMENT block (total?,used?,avail?,capacity?)>
<!ATTLIST block  blocksize CDATA #IMPLIED>
<!ELEMENT total (#PCDATA)>
<!ELEMENT used (#PCDATA)>
<!ELEMENT avail (#PCDATA)>
<!ELEMENT capacity (#PCDATA)>
<!ELEMENT inode (iused?,ifree?,icapacity?)>
<!ELEMENT iused (#PCDATA)>
<!ELEMENT ifree (#PCDATA)>
<!ELEMENT icapacity (#PCDATA)>
```

B.3 システムのログのDTD

```
<!ELEMENT syslog (hostname?,date?,time?,progname?,pid?,message)>
<!ATTLIST syslog facility
(auth|authpriv|cron|daemon|kern|lpr|mail|news|syslog|
user|uucp|local(0|1|2|3|4|5|6|7)) #REQUIRED>
<!ATTLIST syslog level (emerg|alert|crit|err|warning|notice|info|debug)>

<!ELEMENT hostname (#PCDATA)>
<!ELEMENT time (year?,month?,day?,hour?,minutes?,second?)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT month (#PCDATA)>
<!ELEMENT day (#PCDATA)>
<!ELEMENT hour (#PCDATA)>
<!ELEMENT minutes (#PCDATA)>
<!ELEMENT second (#PCDATA)>
<!ELEMENT progname (#PCDATA)>
<!ELEMENT pid (#PCDATA)>
<!ELEMENT message (#PCDATA)>
```

B.4 stethoのDTD

```
<!-- Root Element -->
<!ELEMENT packet (time,protocol,ether?,ip?,transport?)>
<!ATTLIST packet network CDATA #IMPLIED>

<!-- Required Element -->
<!ELEMENT time (#PCDATA)>
<!ELEMENT protocol (#PCDATA)>

<!-- Ethernet Header -->
<!ELEMENT ether (smacaddr?,dmacaddr?)>
<!ATTLIST ether type (data|arp|rarp) #IMPLIED>
<!ELEMENT smacaddr (#PCDATA)>
<!ELEMENT dmacaddr (#PCDATA)>

<!-- IP Header -->
<!ELEMENT ip (saddr?,daddr?)>
<!ATTLIST ip version (4|6) #IMPLIED>
<!ELEMENT saddr (#PCDATA)>
<!ELEMENT daddr (#PCDATA)>

<!-- Transport Header -->
<!ELEMENT transport (sport?,dport?)>
<!ATTLIST transport service (tcp|udp) #IMPLIED>
<!ELEMENT sport (#PCDATA)>
<!ELEMENT dport (#PCDATA)>
```

B.5 ミキサーの設定データのDTD

```
<!-- mixer DTD -->
<!ELEMENT snapshot (master_volume? ,(channel)*)>
<!ATTLIST snapshot preset CDATA #IMPLIED>

<!ELEMENT master_volume (#PCDATA)>

<!ELEMENT channel (fader?,pan?)>
<!ATTLIST channel  ch CDATA #IMPLIED>

<!ELEMENT fader (#PCDATA)>
<!ELEMENT pan (#PCDATA)>
```