ヒューマンインタフェース　96－7
モバイルコンピューティングと　19－7
ワ イ ヤ レ ス 通 信
（２００１．１１．１６）

Network Accessible Device における協調型インターフェース

木俵　豊[†1]　篠宮　俊輔[†1, †2]　土池　政司[†1, †2]　中川晋一[†1]

IPv6 による次世代インターネットでは，グローバルな IP アドレスが大量に利用可能となるため，様々なデバイスがネットワーク対応になると予想されており，そのネットワークデバイスを管理する技術が求められている．様々なネットワークデバイスを協調的に動作させるためには，デバイス特性の記述やそれを用いた協調的なインターフェース機能の実現が必要となる．それを実現する手法として，XML を用いたデバイス特性の構造化記述を設計した．その基本概念と記述仕様について述べる．

Cooperative Interface for Network Accessible Device

Yutaka Kidawara[†1],Shunsuke Shinomiya[†1, †2],Seiji Tsuchiike[†2],
Shin-ichi Nagawaga[†1]

With the next generation IPv6 Internet that provides a large number of IP addresses, various devices are expected to gain network connectivity; thus, it is becoming important to develop network device management technologies. The cooperative interface functionality, is being developed to harmonize the devices on the net by describing their specifications. In order to realize the interface, we designed structured description method using XML, and its basic concept and specifications are detailed in this paper.

1. はじめに

次世代の IPv6 ネットワークにおいて，様々なデバイスが接続されることを想定して，我々はネットワークデバイス NAD(Network Accessible Device)と，その NAD をネットワークを用いて管理するための技術として，NADIA(Network Accessible Device on the Internet Architecture)の研究を進めている．

既に多くのネットワークデバイスが多くの研究者によって研究されているが，既存のネットワークデバイスの多くが専用のユーザインターフェースをもつ専用のアプリケーションで利用されている．インターネットが構築される以前はクライアントコンピュータで動作するプログラムから，そのデバイスの制御を行ってきたが，インターネットの普及に伴い Web の機構を用いて Web をユーザインターフェースとして用いる動きも活発になっている．

Xavier では，Web のインタフェースを用いた制御が行われ，世界中から操作できるようになっている[1)2)]．このような，デバイス側からユーザインターフェース情報を送出してそれに従ってインタフェースが構築できれば，必要十分なインターフェースがデバイス側から提供することが出来るため，ネットワークデバイスのインターフェースとしては，適切であるように見える．しかし，単一機能のデバイスであれば，この仕組みはシンプルで有益であるが，我々が提案するネットワークデバイスのように，複合的に組み合わせることを考えると，単純なユーザインタフェースだけでなく，ネットワーク上からデバイスの特徴と機能を認識して制御する仕組みが必要となる．従って，デバイスの入出力とその機能をデバイスに記述するか，もしくは，代替のコンピュータで管理して，その情報を提供する仕組みが必要となる．また，データを交換するためには，そのデータがどのような性質を持ち，どのような意味的な内容であるのかを理解する必要がある．

このような目的に対して，近年では，メタデータが広く利用されるようになってきた．メタデータは，データへの特性記述が容易であり，理解しやすく構造化モデルを構築することも可能である．インターネットにおいては，HTML が標準的なメータデータとして用いられてきたが，HTML は，レイアウトを重視したメタデータであり，それ以外の情報を記述するには適切ではない．その問題を解決するものとして XML が提案

†1 独立行政法人 通信総合研究所 次世代インターネット Gr.
{kidawara,shinomiya,tuchiike,snagawaw}@crl.go.jp
†2 株式会社ユグドラジル・テクノロジー
{shinomiya,seiji}@yggr-drasill.com

され，その拡張性や表現力の高さから，インターネットにおける標準的なメタデータとして利用され始めている．

メタデータによる記述は，明確なスキーマがあるわけではないが，全く無いとまではいえないようなデータを管理することを可能とする．このようなデータモデルとして，半構造データモデル(semistructured data model)が提案されている．このデータモデルに関しては，Stanford 大学の OEM[3]や Pennsylvania 大学のエッジラベル付きグラフ [4]などがある．XML は，その半構造データモデルの表現を行う場合にも有効であり，半構造データモデルの実装を行うメタデータとして利用され，情報の分類や検索に利用する情報を記述するためのものとして利用されている．[5]

その一方で，その構造化記述の特性を生かして，コンテンツのメディア変換の記述モデルに利用される場合[6]や，XML をオブジェクトリレーショナルデータベースにマッピングして，XML で表記された情報を効率的に管理するといったデータ交換の技術としても提案されている [7].

このように XML の柔軟かつ強力な記述能力は，ネットワーク上の複数のデバイス間での情報交換において，そのデバイスや入出力データに関する特性情報の記述に有益である．さらに，それぞれの特性や機能に関して，ユーザに提示する場合にはユーザインタフェースが必要となるが，XML で記述された情報は，XSL(extensible Stylesheet Language)を用いることで，ユーザに適切なユーザインターフェースを実現することが出来る．

我々は，NAD のインターフェースとして，デバイスがそれぞれの情報を認識して情報交換を行うための協調型インターフェース機能の研究・開発を行っている．ここで述べる”協調”とは，デバイス間でお互いの特性を理解した上で，データの整合性を取ることと定義する．本稿では，その機能の基本的な概念と，XML を用いた記述について述べる．

2. ネットワークデバイスの特徴

コンピュータをはじめとするネットワークに接続されるデバイスは，複数のデバイス間で情報交換をおこなって，与えられた機能を実現する．情報を伝達するためには，お互いの機能を理解するか，もしくは，あらかじめ決められた手順を守って通信することが必要となる．しかし，それらの手順を厳格に決定すると，機能の拡張において制約を与えることがある．また，ネットワークに接続されるデバイスは多様なものとなる可能性があると共に，通常のコンピュータのように複雑な機能を持たず，単一の機能だけを有するものが多い．従って，ネットワークデバイスには，柔軟でシンプルな通信手順が求められる．

現在のネットワークデバイスの多くは，ネットワークデバイスをサーバ，操作するデバイスをクライアントとしたサーバ・クライアントモデルである場合が多い．このようなサーバ・クライアントモデルの構成を持つネットワークデバイスは，データ送受信のためのインターフェースとユーザインターフェースが一体化されたプログラムで利用される場合が多い．本章では，このようなユーザインターフェースとデータのインターフェースが一体となったサーバクライアント型のインターフェースについて，その特徴を考察する．

一般に，ネットワークに接続された機器との間の通信には，大別して二種類の方法がある．一つは処理される側の機器(サーバ)からそのデバイスを操作するための情報を送信し，操作する側のデバイス(クライアント)でその情報に基づいて，操作インターフェースを構築して，通信するものである．他方は，操作する側のデバイスに処理される側の機器を操作する機能を持つインターフェースををあらかじめ保有させることによって通信するものである．我々は，これを操作提供型インターフェースと操作要求型インターフェースと定義して，ネットワークデバイスへ適応させた場合の特徴について述べる．

2.1 ネットワークデバイスにおける操作提供型インターフェース

本稿における操作提供型インターフェースとは，操作されるネットワーク接続デバイスから，そのデバイスを制御するために必要十分な操作情報と図１に示すように,利用者が操作するためのユーザインターフェース情報を提供するものとする．

この最も一般的な例としては，Web サーバとブラウザがある．Web サーバを持つネットワークデバイスは，インターフェース情報を HTML で記述することで，そのデバイスを制御するユーザインターフェースを提供することが出来る．また，HTML だけでなく CGI を用いることで，Web サーバから外部のプログラムを起動することを可能としている．この操作提供型のインターフェースにおいては，利用者が操作したいデバイスに必要なデータが存在するのかどうかを理解することが容易であり，提供されたインターフェー

スに基づいてデータを入力すればよい．しかし，どのようなデータを入力すれば良いのかを理解することは必要であり，何らかの手段で入力すべきデータの意味的な情報を提供する必要がある．

HTML は，ユーザインターフェースのレイアウトを重視した設計となっている．そのため，GUI 等を通じてデータの入力を促す場合には，利用者がそこに何を入力するかを理解できるような説明や仕様をあらかじめ提供する必要がある．

HTML と同様に Java アプレットは，プログラムを要求元に送信し，操作側で操作のためのインターフェースの構築を実現するものである．Java アプレットは HTML では処理できないような高度な処理をクライアント側で実現することが出来る．しかし，前述の問題は HMTL 同様に持っている．また，Java アプレットを動作させる処理側のデバイスにそれを動作させるための VM が必要である．さらに，セキュリティ構造のため，処理側のデバイスからのリソースを読み込むことが出来ない構造となっているため，アプレットによる処理は制約を受けることがある．

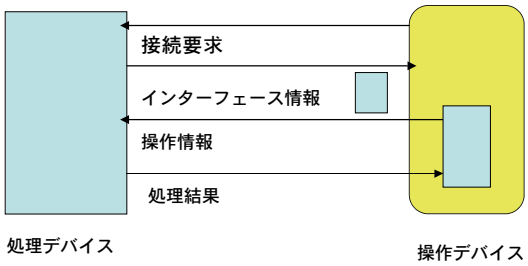

図 1　操作提供型インターフェースの概念図

2.2 ネットワークデバイスにおける操作要求型インターフェース

本稿において，操作要求型インターフェースとは，遠隔地のネットワークデバイスをあらかじめ用意されたインターフェースを用いて，ネットワークデバイスに処理を要求する機能をもつものとする．提供型インターフェースと異なり，処理すべきサーバとの通信の有無にかかわらず，操作可能な機能が存在している．

操作要求型インターフェースでは，操作側デバイスからの操作要求が処理側デバイスに十分なものであるかが問題となる．操作するインターフェース側では，処理できる要求が遠隔地のデバイスを操作できるものであるかどうかは，必要な情報を送信するまでは分からない．そのため，汎用的なインターフェース機能を構築することが困難であり，デバイス固有のインターフェースとなる可能性が強い．例えば，図 2 で示すように操作デバイスは，処理デバイス A を操作できるが，処理デバイス B とは，入出力情報のマッチングが困難で操作することは出来ないということが発生する．しかし，処理要求を受け取るデバイスにとっては，それ自身が必要とする情報の提供を待つだけでよく，処理できる十分な情報を受け取った時にその処理結果を送信し，それ以外の時はエラーを送信するといった単純な処理を行うだけでよい．従って，高機能なリソースを必要としないため，モータや各種センサといったプリミティブなデバイスにとっては適した仕組みである．

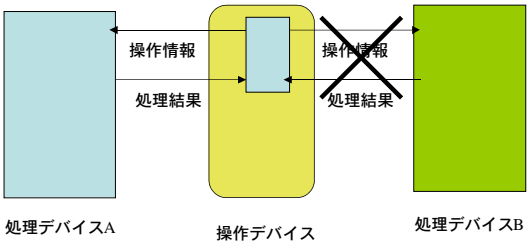

図 2　操作提供型インターフェースの概念図

2.3 複合ネットワークデバイスおけるインターフェース

ネットワークデバイスは，単純なサーバ・クライアントモデルだけでなく複数のネットワークデバイスが組み合わされた複合的なデバイスとなることがある．このような複合的なデバイスにおいては，デバイス同士がお互いの機能を理解した上で必要となる情報を交換する必要がある．また，ユーザとの接点となるデバイス以外では，明示的に情報を表示する必要は無く，ユーザインターフェースの機能と情報交換のためのインタフェース機能を分離する必要がある．

我々が提案する NAD(Network Accessible Device)は，複数のネットワークデバイスを組み合わせたネットワーク上の機能体を実現することを目指している．従って，そのような複合ネットワークデバイスにおけるインターフェースも，従来のサーバ・クライアントモデルに基づいたインターフェースではなく，end-to-end モデルに基づくインターフェース構造が必要である．次章で，NAD に必要なインターフェースの機能について述べる．

3.　NAD に要求されるインターフェース機能

NAD においては，複数のデバイスが組み合わされるため，単純なサーバ・クライアントモデルでの通信で

はなくなる．データの送受信は各デバイスから相互に行われ，一部のデバイスによって利用者への情報提供が行われる．
つまり，NAD のインターフェース機能としては以下のような機能が必要となる．

- **デバイスの機能の提示**
- **入出力データの整合化**
- **ユーザインターフェース情報の集約化**

これらの要求される機能について次節に述べる．

3.1 デバイス機能の提示

複数のデバイスを組み合わせるにあたっては，それぞれのデバイスがどのような機能を持っているのかを理解する必要がある．機能する複合デバイスを構築した後には，各デバイスの機能を単独で利用することは少なくなるが，ネットワークデバイスを検索し，利用可能なデバイスを組み合わせる場合には，制作者がそのデバイスの機能を理解する必要がある．従って，利用者が各デバイスにアクセスした際に，そのデバイスがどのような機能を保有し，それを利用するために必要な入力データと処理機能によって出力される出力データの内容が理解できる仕組みが必要となる．

その機能を実現するためには，NAD が持つ Function Element の内容を提示する機能が必要となる．

3.2 入出力データの適合化

デバイスの機能を理解した後には，そのデバイスにデータを送信し，その結果を受信する必要がある．しかし，その通信においては，プロトコルだけでなく必要となるデータ項目数やデータのタイプなどを適合化させなければならない．また，データの送受信を行うためにネットワークを通じてデバイスを接続する必要があるが，その接続には，セキュリティについても十分考慮する必要がある．また，接続出来るデバイスを制限したり，接続する場合に課金を行う必要もある．

従って，入出力においては，デバイスを組み合わせて複合デバイスを制作する制作者に入出力データの種類を明示する仕組みと，実際の使用時においてデバイス間でデータの種別を判断し，送受信を行う仕組みが必要となる．

3.3 ユーザインターフェース情報の集約

デバイスを複合的に組み合わせた場合には，各要素となるデバイスに直接データを受け渡さなければならない場合がある．つまり，各デバイスにおいて，他のデバイスから得られる情報以外には利用者に入力を促す必要があるため，それらの入力値に関しては，ユーザインターフェースを表示する必要がある．それを実現するためには，ユーザと接するインターフェースを構築するデバイスに，必要となるユーザインターフェースを集約して表示することが必要となる．

4. NAD におけるインターフェース機構の設計

前節で述べたインターフェース機能を実現するにあたって，それぞれの特性を記述して理解させるためのメタデータによる内容記述を行う．本節では，内容記述のための NAD に必要な記述情報について述べた上で，メタデータ記述のためのスキーマ構成について述べる．

4.1 記述情報の属性

デバイスに関する機能や入出力データに関しては，複数のデータを柔軟に記述することが要求されるが，その記述においてはデータの特性を明示的に記述しながら，その対象物の構造についても表現できるものが望ましい．その要求に基づくものとしてメタデータの利用が考えられる．

NAD は，一つの Function Element と一つ以上の Input/Output/InputOutput Element が必要となる．その特徴を記述するために，以下のような属性を定義する．

- データ種別
  - DataName:データの名前
  - DataType:データのタイプ
  - DataSize:データのサイズ
  - DataNumber:データの個数
- 機能
  - FunctionName:機能名
  - NADData:必要とするデータ
- 接続属性
  - ConnectionType:データの入力もしくは出力
  - AccessibleDevice:アクセス可能デバイス名
  - AccessibleUser:アクセス可能ユーザ
- 識別情報

- DeviceName:デバイス名
- DeviceType:デバイスカテゴリ
- Note:デバイスに関する注釈

4.2 メタデータによる記述

前節で述べたように，デバイスの記述においては，上記の属性値が必要となり，それを明示的に記述する必要がある．これらの記述に関しては，メタ記述言語が有効である．また，メタデータを用いるためにはデータのスキーマモデルを定義する必要がある．我々は，NAD のスキーマモデルを図 3 のように定義した．

まずこのスキーマモデルの第一層には，NAD の識別情報と FunctionElement 及び，入出力のための InputOutputElement が定義されている．また，オプションとして，複合 NAD の要素 NAD を記述するためのフィールドも用意した．また，InputOutput Element と Function Element にもそれぞれ，子供の属性を持っており，それらは NAD のスキーマモデルの第二層，三層，四層を構成する．

図の中で，それぞれのデータ名の後に付与された"+"は，上位の定義の中で複数の定義が可能であることを示しており，?はオプションとして，0 回以上登場することを示している．

このスキーマモデルは，NAD の基本構造を表しており，このスキーマモデルのインスタンスとして各 NAD を表現することを考える．

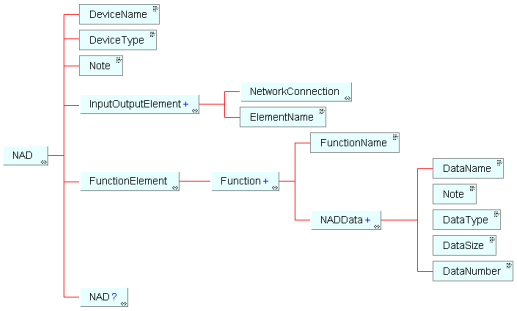

図 3　NAD のスキーマモデル

5.　XML による NAD の記述

前章で述べたスキーマモデルを実現するためには，いくつかの手法が考えられる．我々は，インターネットにおける標準技術を用いることで，既存の仕組みに適合しやすいシステム構築が可能となると考え，このスキーマモデルを基にして，XML で実装した．本章では，それらの実装例について述べる．

5.1 DTD の記述

前節で述べたスキーマモデルに基づいて XML で表現するためには，その構造を DTD(Document Type Definition)として定義する必要がある．DTD には，タグの親子関係で構造を記述することが可能であり，ドキュメント内で登場する回数なども規定することが出来る．この NAD の DTD は，NAD の特性を記述する最もプリミティブなものである．

NAD の"一つの Function Element"と，"一つ以上の Input Element,Output Element, InputOutput Element"を実現できるように root タグのすぐ下のレベルに FunctionElementElement タグ,InputOutputElement タグを定義した．また，複合 NAD を構成する要素 NAD の記述を行うために，入れ子構造として NAD を記述出来るものとした．この NAD は，複合デバイスの記述を行う際に，必要に応じて記述できるようにオプション指定としている．設計した DTD を図 3 に示す．

各種デバイスと通信する際には，この DTD の記述に基づいて，必要とするデータの種別を検索し，その中に記述されている属性値を取得することで，デバイス間でのインターフェースを協調的に整合化することが可能となる．

```
<?xml version ="1.0" encoding='UTF-8' ?>
<!--DTD for NAD Ver. 0.01-->

<!--ROOT TAG NAD-->
<!ELEMENT NAD (DeviceName , DeviceType , Note ,
InputOutputElement+ , FunctionElement , NAD? )>
<!ELEMENT DeviceName (#PCDATA )>
<!ELEMENT DeviceType (#PCDATA )>
<!ELEMENT Note (#PCDATA )>

<!--InputOutputElement TAG-->
<!ELEMENT InputOutputElement (NetworkConnection ,
ElementName )*>
<!ELEMENT NetworkConnection (ConnectionType , AccessibleDevice ,
AccesibleUser )>
<!ELEMENT ConnectionType (#PCDATA )>
<!ELEMENT AccessibleDevice (#PCDATA )>
<!ATTLIST AccessibleDevice e-dtype NMTOKEN #FIXED 'uri' >
<!ELEMENT AccesibleUser (#PCDATA )>
<!ELEMENT ElementName (#PCDATA )>

<!--FunctionElement TAG-->
<!ELEMENT FunctionElement (Function+ )>
<!--Function TAG-->
<!ELEMENT Function (FunctionName , NADData+ )>
<!ELEMENT FunctionName (#PCDATA )>
<!--NADData TAG-->
<!ELEMENT NADData (DataName , Note , DataType , DataSize ,
DataNumber )*>
<!ELEMENT DataType (#PCDATA )>
<!ELEMENT DataSize (#PCDATA )>
<!ELEMENT DataNumber (#PCDATA )>
<!ELEMENT DataName (#PCDATA )>
```

図 4　NAD の DTD

5.2 XML での記述

前節の DTD を用いて，NAD の特性について記述することが可能となる．本節では，我々が実装した Camera

Car NAD について，その特性を前節の DTD を用いて記述した．図 5 にその内容を示す．

```
<?xml version ="1.0"?>
<!DOCTYPE NAD SYSTEM "file:///D:/Home/kotaro/Xml/NAD.dtd">
<NAD>
        <!-- (DeviceName , DeviceType , Note , InputOutputElement+ ,
FunctionElement , NAD? )-->
<DeviceName>Camera Car NAD</DeviceName>
<DeviceType>INPUT_OUTPUT</DeviceType>
<Note>IP ControlCar Attached Camera</Note>
<InputOutputElement>

        <!-- INPUT CONNECTION TO DRIVE
                (NetworkConnection , ElementName )*-->
<NetworkConnection><!--  (ConnectionType  ,  AccessibleDevice  ,
AccesibleUser )-->
<ConnectionType>INPUT</ConnectionType>
<AccessibleDevice>http://controller1.tky.apii.net</AccessibleDevice>
<AccesibleUser>USER1</AccesibleUser>
</NetworkConnection>
<ElementName>CarControlCommand</ElementName>
</InputOutputElement>
<InputOutputElement>

        <!-- OUTPUT CONNECTION TO DISPLAY
    (NetworkConnection , ElementName )*-->
<NetworkConnection><!--  (ConnectionType  ,  AccessibleDevice  ,
AccesibleUser )-->
<ConnectionType>OUTPUT</ConnectionType>
<AccessibleDevice>http://controller1.tky.apii.net</AccessibleDevice>
<AccesibleUser>ANY</AccesibleUser>
</NetworkConnection>
<ElementName>DISPLAY CAMERA IMAGE</ElementName>
</InputOutputElement>

<FunctionElement><!-- (Function+ )-->
<Function><!-- (FunctionName , NADData+ )-->
<FunctionName>Network Control Car</FunctionName>

<NADData><!-- (DataName , DataType , DataSize , DataNumber )*-->
<DataName>RunDirection</DataName>
<DataType>Float</DataType>
<DataSize>4Byte</DataSize>
<DataNumber>1</DataNumber>
</NADData>

<NADData><!-- (DataName , DataType , DataSize , DataNumber )*-->
<DataName>Speed Control</DataName>
<DataType>Float</DataType>
<DataSize>4Byte</DataSize>
<DataNumber>1</DataNumber>
</NADData>

<NADData><!-- (DataName , DataType , DataSize , DataNumber )*-->
<DataName>Steering Angle</DataName>
<DataType>Float</DataType>
<DataSize>4Byte</DataSize>
<DataNumber>1</DataNumber>
</NADData>

</Function>
<Function><!-- (FunctionName , NADData+ )-->
<FunctionName>Show Front</FunctionName>

<NADData><!-- (DataName , DataType , DataSize , DataNumber )*-->
<DataName>Set Resolution of Image</DataName>
<Note>Resolution Parameter width*hight</Note>
<DataType>String</DataType>
<DataSize>16byte</DataSize>
<DataNumber>1</DataNumber>
</NADData>

<NADData><!-- (DataName , DataType , DataSize , DataNumber )*-->
<DataName>Set Frame Rate of Image</DataName>
<Note>Resolution Parameter 1-30(frames/sec)</Note>
<DataType>String</DataType>
<DataSize>16byte</DataSize>
<DataNumber>1</DataNumber>
</NADData>

<NADData><!-- (DataName , DataType , DataSize , DataNumber )*-->
<DataName>Switch</DataName>
<Note>Switch Parameter 1:on,0:off</Note>
<DataType>Integer</DataType>
<DataSize>2Byte</DataSize>
<DataNumber>1</DataNumber>
</NADData>

</Function>
</FunctionElement>
<NAD>EMPTY</NAD>

</NAD>
```

図 5 Camera Car NAD の内容記述

このような XML 記述を行うことで，デバイスの特性を容易に検索することが出来る．例えば，この NAD に関して，デバイス名やその特性を知りたい場合には，DeviceName タグや DeviceType タグを検索すればよい．また，入出力情報を取得したい場合には，InputOutput タグの下位にある NetworkConnection タグを抽出して，必要な情報を手に入れることが出来る．

つまり，XML のパーサ機能を持つことで，任意の特性を抽出することが可能となる．この抽出された結果をネットワークで接続されたデバイス間で送受信することで，両者の間でデバイスの特性を通知したり，入出力情報として必要な情報を入手することが出来る．

5.3 XSL によるユーザインターフェースの構築

前節では，XML による NAD の記述例について述べた．XML そのものは，NAD の構造について記述されているものであり，デバイスの特徴を理解することが出来る．デバイスを解析するプログラムにとっては，XML のパーサからの出力を機械的に処理することが出来るが，その結果をユーザに提示したり，ユーザが操作するための情報を送信するためには，少々わかりにくいものとなっている．

ユーザが NAD の特性を理解しようとする場合や，操作を行う場合，操作結果を評価する場合には，適切なインターフェースが必要となる．XML は，XSL を用いることで，XML データから，ユーザインターフェースを構築することが可能となる．つまり，デバイスの記述をメタスキーマとして XML で記述を行えば，その情報を基にして集約したユーザインターフェースを XSL を用いることで容易に構築することが出来る．

例として，図 5 で示した Camera Car NAD の XML 記述の中から，入出力情報を必要とするユーザに対して HTML ブラウザ用に入出力情報を出力する XSL を図 6 に示すと共に，その出力結果を図 7 に出力する．この例は，利用者がこの Camera Car NAD に対して制御するデバイスを組み合わせようとした場合に使用することを想定している

```
<?xml version="1.0" encoding="Shift_JIS"?>
<xsl:stylesheet       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:output method="html" indent="yes"/>
<xsl:strip-space elements="*"/>
<xsl:template match="/">
<HTML>
<HEAD>
<TITLE>NAD 入出力データ</TITLE>
</HEAD>
<BODY>
<p><font size="+4">NADIA 構成：入出力情報</font></p>
<TABLE BORDER="1">


<TR>
    <td >入出力名</td>
    <td >入出力属性</td>
     <td >接続可能デバイス</td>
    <td >接続可能ユーザ</td>
</TR>
<xsl:for-each select="/NAD/InputOutputElement">
  <tr>
<td><xsl:value-of select="ElementName"/></td>
<td><xsl:value-of select="NetworkConnection/ConnectionType"/></td>
      <td><xsl:value-of      select="NetworkConnection/AccessibleDevice
"/></td>
      <td><xsl:value-of          select="NetworkConnection/AccesibleUser
"/></td>
  </tr>

</xsl:for-each>
</TABLE>
</BODY>
</HTML>
</xsl:template>
</xsl:stylesheet>
```

図 6　入出力情報表示用 XSL

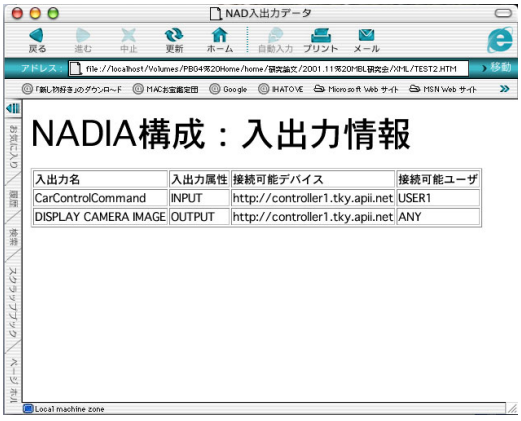

図 7 HTML への出力

6.　NADIA における協調型インターフェース

6.1 複合 NAD の制作

NAD は，複数のデバイスを組み合わせて，より高機能なものへと組み合わされる．例えば，例として挙げた，Camera Car NAD は，ユーザが直接操作することも可能である一方で，別の制御デバイスがプログラムの処理結果に基づいて操作することも出来る．つまり，見方を変えれば Camera Car NAD もさらに大きな概念の複合 NAD の要素となる．

例えば，Camera Car NAD を一つの要素とする Driving Camera Car NAD の構築を考える．Camera Car NAD を対話的に操縦する場合には，カメラの映像を写し出すネットワーク接続可能なディスプレイと，車のステアリング角とモータの出力を制御するデバイスが必要となる．そこで，ステアリング型のデバイスと，ネットワークに接続して表示するデバイスとして以下の性質を持つデバイスを想定する．

- Network Display NAD
  - InputNAD としての性質を持つ
  - 入力値は映像ストリーム
  - 接続先の制約はなし
- Steering Pedal NAD
  - InputOutput NAD としての性質を持つ
  - 出力値はステアリング角とアクセルペダルの開度
  - 入力値は，ステアリング角とアクセルペダルのスケーリング値

これら２つのデバイスを前述の Camera Car NAD と組み合わせることで，ユーザが操縦できる Driving Camera Car NAD を構成できる．その概念図を図 8 に示す．

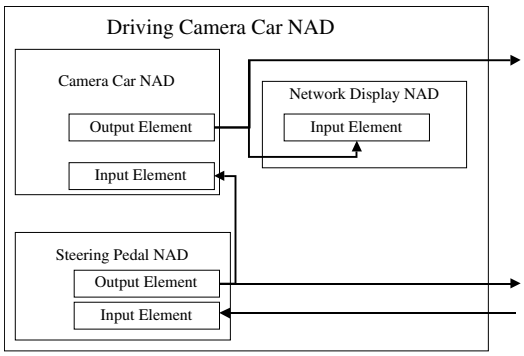

図 8Driving Camera Car NAD の概念図

このような構成は，それぞれの NAD に関する記述に基づいて行われる．つまり，NAD に記述された情報を用いて，ネットワークデバイスの設計者が NAD の入力と出力を結びつけて，必要に応じて新しい調整機能を付与することで，新しい機能体を制作（オーサリング）することが可能となる．

6.2 複数 NAD の協調制御

NAD は，ネットワーク上に存在するデバイスであり，それらを複数組み合わせて動作させることが可能となる．それらのデバイスは，意味的な特徴は同じであっても，個々の特性は微妙に異なることがある．

例えば，前述の Driving Camera Car NAD においても，Steering と Pedal を操作して，ディスプレイに映る画像を見ながら運転するものであるという意味では同じであっても，それぞれの走行性能は異なる場合がある．その場合，それぞれの NAD が個々に用いられる時には，その特徴の差は，Driving Camera Car NAD の個性となり問題にはならないが複数の NAD をそれぞれを協調して動作させたい場合には，NAD 間でその特性を調整する必要がある．

例えば，Driving Camera Car NAD を用いてレースを行う場合には，レギュレーションにあわせて，各デバイスの特性を均一にすることを要求される場合もある．つまり，複数のデバイスを群制御する場合において，デバイス間で協調的に動作させる仕組みが必要となる．

7. おわりに

本稿では，NADIA における協調型インターフェースを実現するための，特徴記述について述べた．NADIA においては，その構造と特性を表現するために XML を用いて NAD の特徴を記述する．NAD は複合的に組み合わせることで，新しい複合 NAD を構築することが可能であり，XML はその特徴を表現するために最適なメタデータであると考える．現在は，NAD の性質を記述するにとどまっているが，今後は，複合 NAD の組み合わせを表現するための記述仕様とそれを管理するミドルウェアの開発を行っていく予定である．そして，それらを用いて，XML ベースの記述で複合 NAD のオーサリングの実現を目指す．

また，その一方で，NAD の特性記述を中心に設計した記述仕様を，そのデバイスによって利用可能なコンテンツに関する記述も可能となるように拡張する予定である．

参 考 文 献

1) R.Simmons,J.Fernandes,R.Goodwin,S.Koening,J.O 'sullivan Xavier:An Autonomous Mobile Robot on the Web,Robotics and Automation Magazine, 1999.cs.cmu.edu/Web/People/Xavier/.
2) L.R.Lab: Where in the world is Xavier,the robot?.,http://www.cs.cmu.edu/afs/cs.
3) R.Goldman,S.Chawathe,A.Crespo,K.McHugh:A Standard Textual Interchange Format for the Object Exchange Model(OEM),TechnicalReport.October,1996
4) P.Buneman,S.Davidson and D.Suciu: Programming Constructs for Unstructured Data.Proceedings of International workshop on DBPL electronic workshop in Computing'96,pp506-516,1996
5) 高橋，川村，遠山:芸術情報のデジタルアーカイビングにおける XML の利用，情報処理学会データベース研究会報告 DBS122-1,pp.9-15,2000
6) 服部，沢中，灘本，田中:Web の受動的視聴のための同期化可能領域の発見と番組化用マークアップ言語 S-XML,情報処理学会データベース研究会報告 DBS122-2,pp.9-16,2000
7) 西岡,鬼塚:XML のオブジェクトリレーショナルマッピングに関する一手法，情報処理学会データベース研究会報告 DBS122-3,pp.17-24,2000