ソフトウェア工学 135−1
（２００１．１１．２１）

ポリシー管理ツール開発における
数学的手法に基づく要求仕様の定義と実現

登内 敏夫
NEC **ネットワーキング研究所**

ポリシーに基づく分散システム管理支援ツールとして，ツリー構造のデータを操作する Domain Browser を開発した．通常のプロトタイプ開発手法で獲得した要求仕様に加え，ツリー視覚化アルゴリズムに対する要求仕様を数学的に定義した．採用した Hyperbolic Tree 視覚化アルゴリズムがこれらの要求仕様を満たすことを証明した．本手法は，色などのプロトタイプ手法で獲得・実現する嗜好的な要求仕様とは異なる，本質的要求仕様を獲得・実現するのに有効であった．

A mathematical approach to definition and fulfillment of requirements for development of a policy management tool

Toshio TONOUCHI
Networking Research Laboratories, NEC Corporation

We developed a policy management tool called a Domain Browser for management of distributed computing systems. We defined mathematically requirements of the Domain Browser and proved that the Hyperbolic Tree visualization algorism we selected satisfies the requirements. This proof-based method is useful for acquiring essential requirements other than sensitive requirements, such as colors, acquired in the prototype method.

1. はじめに

我々はポリシーによる分散システム管理のためのツールとして Domain Browser[1]を開発した．ポリシー言語 Ponder[1]では，ドメインと呼ぶ集合で分散システムの構成要素を管理する．ポリシーの適用対象を，個別の管理対象ではなく，ドメインで指定する．ドメインが含む要素を変更することで，ポリシーの適用対象を変更することができる．ドメインはドメインも含むことができ，ツリー構造をなす[1]．Domain Browser を使い，ユーザはツリー構造を眺め，サブツリーを削除，移動することができる．このようにドメインの構成を変更し，ポリシーの適用対象を変更することで，分散システムを管理する．

[1] 正確には非循環グラフであるが本稿では簡単のためツリー構造とする．実際には非循環グラフからスパニングツリーを取りだし，表示する．ツリーの枝以外のリンクは描画するが，ツリーのレイアウトには影響しない．

Domain Browser を構築する際, ツリーデータを視覚化する必要がある. これまで多くのツリー視覚化アルゴリズムが提案されている[3][5][6][7][8][9][10][11][12] . Domain Browser の開発では, 要求仕様からいくつかの満たすべき性質を選び出し, 数学的に厳密に定義し, これらの性質を満たすことを証明できる可視化アルゴリズムを選択した.

一般に視覚化アルゴリズムを含む GUI デザインはプロトタイプ手法で開発を進めることが多い. 開発者はプロトタイプシステムを構築し, 利用者がプロトタイプをみて GUI デザインの変更を要求する.

プロトタイプ手法では色などの具体的な項目が変更要求として挙がる. しかし, その変更要求の根拠は明示されないし, 利用者もその根拠を意識していないことが多い. そのため, プロトタイプ手法でのレビュー時では, 具体的な変更要求がどのような性質に基づき, なぜ行なわれたが不明確になる. レビュー時に発生した変更要求の仕様が不明確なまま作業が進捗するので, システムをバージョンアップする際など要求仕様を再利用することが難しくなる.

本手法とプロトタイプ手法を組み合わせることで, 具体的かつ嗜好的な要求仕様をプロトタイプ手法により実現し, 将来のバージョンアップで継承する必要があるような定性的要求仕様を本手法で実現することができる.

2. Domain Browser とツリー視覚化アルゴリズムに対する要求される性質

Domain Browser は分散システム管理ツールとして, 管理者に好ましいユーザインタフェースを提供する必要がある. そこで, 我々は以下の 5 つの要求を定めた.

【要求 1】(展望性)

Domain Browser はツリー全体を管理者に示すべきである. 分散システムでは数百個から数万個の管理対象が存在する. 管理者が対象の関係を一目で把握できるべきである.

【要求 2】(焦点性)

Domain Browser は管理者の関心がある部分については強調して表示すべきである. 管理者が関心のあるノードを指定することを「焦点を当てる」と呼ぶ. 強調表示には, 色を変えるなど, 様々な方法がある. ここでは, ノードに充分に広い領域を割り当てることにより強調表示を行なう方法を採用する(図 1).

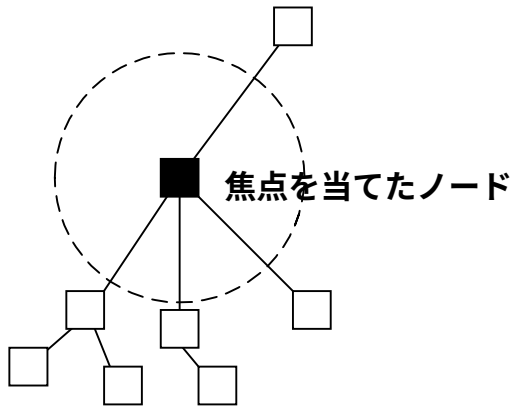

図 1 強調表示

【要求 3】(均質性)

焦点を当てたノードはツリー上, どのような位置にあっても同様に強調されて表示すべきである. つまり, ノードがツリーの中間ノードであろうと末端であろうと, 焦点を当てたら同等に充分な領域を割り当てるべきである.

【要求 4】(連続性)

管理者がマウス等のポインティングデバイスでツリーを移動させるとツリーは連続的に移動すべきである.

【要求 5】(追随性)

管理者がマウス等のポインティングデバイスでツリーを移動させると, ポインタが示しているツリー上の点はポインタと同じ位置を辿るべきである.

以上 5 つの要求を以下の 5 つの性質として, 数学的に定義する.

R を実数の集合とする. P は 2 次元平面とする. ツリーT = (V, E) はノードの集合 V⊂P と枝 e (**p**, **c**)の集合 E の集合とする. ただし, ノード **p**∈P が親で, **c**∈P が子である. T はツリーなので, 子ノード **c** の親ノードは高々1 つしか存在しない. つまり, e_1 = (**p**$_1$, **c**), e_2 = (**p**$_2$, **c**) ∈ E ⇒ **p**$_1$ = **p**$_2$.

2 点 **p**$_1$, **p**$_2$ ∈P 間のユークリッド距離を | **p**$_1$, **p**$_2$ | と表す.

ノード **p**$_1$, **p**$_2$ をベクトルとみなし, 両者の内積を **p**$_1$・**p**$_2$ と表す.

移動関数 f : P×P → V → P は, ポインティングデバイスを用いて点 **p**$_1$ を **p**$_2$ に動かしたとき, ノード **v** の動いた先の座標 f(**p**$_1$, **p**$_2$)(**v**) を表す関数である.

【性質 1】(展望性)

2 次元平面 P が展望性を満たすとは,

定数 d∈R が存在し, 2 次元平面 P 上の任意の点 **p**$_1$, **p**$_2$∈P が, | **p**$_1$, **p**$_2$ | < d を満たすことをいう.

ツリー T = (V, E) が展望性を満たすとは,

任意のノード v∈V が展望性を満たす平面 P 上に存在することをいう.

本性質はツリーが大きさの定まった平面内に表示可能であることを示す(図 2).

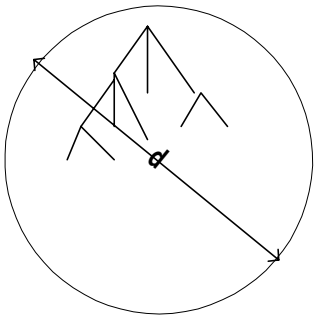

図 2 展望性

次に, 焦点性を定義するために,「サブノード」と「支配的扇形」(図 3)を定義する.

【定義 1】(サブノード)

ツリーT=(V, E)において, ノード v'∈V がノード v∈V のサブノードであるとは, 下記 3 つの条件のいずれかが成り立つことである.

- **v** = **v'**.
- (**v**, **v'**) ∈ E.
- **v** のサブノード **v**$_m$ が存在し, (**v**$_m$, **v'**) ∈ E.

なお, ツリーは非巡回なので, ノード **v'**∈V がノード **v**∈V のサブノードでかつ **v** が **v'**のサブノードであることはない.

【定義 2】(支配的扇形)

ツリーT=(V, E)において, ノード **v**∈V の支配的扇形 S_v ⊂P は, **v** を扇の要とする半径無限長の扇型であり, 以下の性質を満たすものをいう.

- **v'**∈V が **v** のサブノード ⇒ $S_{v'}$ ⊂ S_v.
- **v'**∈V が **v** のサブノードではない ⇒ **v'** ∉ S_v.
- **v'**∈V が **v** のサブノードではなく, かつ, **v** が **v'**のサブノードではない ⇒ $S_{v'}$ ∩ S_v = φ.

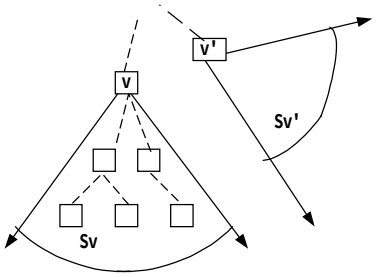

図 3 支配的扇形

【性質 2】（焦点性）

距離 d∈R, 角度 θ ∈R が与えられたとき, ツリーT = (V, E) がノード **v**∈V において焦点性を満たすとは, 次の 2 つがともに成り立つことである.

- **v** のサブノード **v'**において,
  v'≠**v** ⇒ |**v'**, **v**| ≧ d.
- **v** の支配的扇形 S_v の中心角 t が
  t ≧ θ.

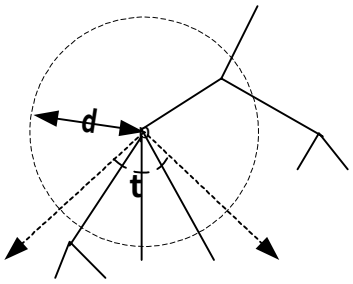

図 4 焦点性

【性質 3】（均質性）

ツリーT = (V, E) が均質であるとは, ∃**p**∈P, ∃d, θ ∈R が与えられたとき, f(**p**$_1$, **p**$_2$)(**v**) = **p** ⇒ ツリー f(**p**$_1$, **p**$_2$)(T)は点 **p** において d, θ で焦点性を有する.
ただし, f(**p**$_1$, **p**$_2$)(T) は移動関数 f(**p**$_1$, **p**$_2$)により移動したツリーT を表す.

【性質 4】（連続性）

移動関数 f : P×P → V → P が連続であるとは, 以下を満たすことである.

$$\forall p_1, p_2 \in P,\ \forall v \in V.\ \lim_{P_1 \to P_2} f(p_1, p_2)(v) = v.$$

【性質 5】（追随性）

移動関数 f : P×P → V → P が追随性を有するとは, 以下を満たすことである.

$$\forall \mathbf{v_1}, \mathbf{v_2} \in V.\ \lim_{P_1 \to P_2} f(\mathbf{v_1}, \mathbf{v_2})(\mathbf{v_1}) = \mathbf{v_2}$$

図 5 Domain Browser

図 6 Domain Browser (左上に移動したとき)

3. Hyperbolic Tree 視覚化アルゴリズムとその性質

Hyperbolic Tree 視覚化アルゴリズム[3]とは非ユークリッド空間の一つである双曲平面のモデル Poincare 円板 D = {(x, y) | $x^2 + y^2 < 1$}上にツリーを配置するツリー視覚化アルゴリズムである．Poincare 円板 D ではその中心から遠いほど双曲平面上での距離は増大する．2 点 $\mathbf{p_1}, \mathbf{p_2}$ 間の双曲平面距離 $|\mathbf{p_1}, \mathbf{p_2}|_h$ は次のように定義される[2].

$$|\mathbf{p_1}, \mathbf{p_2}|_h = \int_0^1 \left| \frac{d\gamma(t)}{dt} \right| \frac{2}{1-|\gamma(t)|^2} dt$$

ここで，γ は点 $\mathbf{p_1}, \mathbf{p_2}$ を通る双曲平面線分を表し，$\gamma(0) = \mathbf{p_1}$，$\gamma(1) = \mathbf{p_2}$ である．γ(t)が D の中心から離れ，1 に近づくほど双曲平面距離が増大することがわかる．また，Poincare 円板上でのユークリッド角度は双曲平面角度と同じである[2].

Hyperbolic Tree 視覚化アルゴリズムでは，各ノードの支配的扇型の中心角が等しく(本事例では角度π），接続されたノード間の双曲平面距離が一定になるようにノードを配置する．Poincare 円板では与えられた直線と交わらず，与えられた点を通る直線が無数に引けることが知られている[2]．そのため，中心角πの支配的扇型同士が重なることはない．円板 D の円周に近づくほど双曲平面距離が増大するため，ノード間の双曲平面距離が一定のノードを無限個配置することができる．

Hyperbolic Tree 視覚化アルゴリズムでの移動関数 f は鏡像写像 T_C を使って定義する．

【定義 3】鏡像写像 T_C

鏡像写像 T_C は以下のように定義される．ただし，C は円もしくは円板Dの中心を通る直線である．

- 円 C={$\mathbf{p}$ = (x,y) | | $\mathbf{p}$ - $\mathbf{c}$ |² = r²}による鏡像写像 T_C とは，円板 D 内の点 $\mathbf{p}$ を $\mathbf{p'}$に移す写像である．ただし，$\mathbf{p'}$ は $|\mathbf{p'}- \mathbf{c}||\mathbf{p} - \mathbf{c}| = r^2$ かつ ($\mathbf{p'}- \mathbf{c}$) と ($\mathbf{p} - \mathbf{c}$)は並行となる．
- 直線 C による鏡像写像 T_C は円板 D 内の点 $\mathbf{p}$ を直線 C で線対称の位置に移す写像である．

円 C が Poincare 円板 D と直行する場合，鏡像写像 T_C は D を D に写し，双曲平面距離と双曲平面角度において，等距離・等角変換であることが知られている[2]．また直線 C が円板 D の中心を通る場合も同様である．

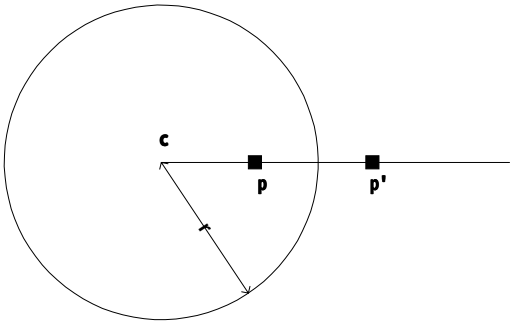

図 7 円 C による鏡像写像 T_C

【定義 4】Hyperbolic Tree 視覚化アルゴリズムでの移動関数 f : P×P → V → P

Hyperbolic Tree 視覚化アルゴリズムでの移動関数 f : P×P → V → P は $\mathbf{p_1}$ を $\mathbf{p_2}$ に移す鏡像写像 T_{C1} と，$\mathbf{p_2}$ を $\mathbf{p_2}$ に写し，かつ，円 C2 の中心が $\mathbf{p_1}$ と $\mathbf{p_2}$ を結ぶ直線上に中心が存在する円 C_2 の鏡像写像 T_{C2} とを使って，$f(\mathbf{p_1}, \mathbf{p_2}) = T_{C2} \cdot T_{C1}$ と定義できる．ただし，C_1, C_2 は Poincare 円板 D と直行する円，もしくは円板 D の中心を通る直線である．「・」は関数の合成を表す．

本移動関数は，鏡像写像 T_C が等距離・等角変換であるため，ツリーの形を保存する．

Hyperbolic Tree 視覚化アルゴリズムを適用して開発した Domain Browser を図 5，図 6に示す．

以下, Hyperbolic Tree 視覚化アルゴリズムが性質 1-5 を満たすことを証明する.

【命題 1】展望性

Hyperbolic Tree 視覚化アルゴリズムで表示されたツリーは展望性を満たす.

【証明 1】

円板Dの半径がユークリッド距離で 1 なので, Poincare 円板 D は展望性を満たすことは, 明らか. Hyperbolic Tree 視覚化アルゴリズムでは, すべてのノードを D 上に配置するので, 展望性を満たすことは明らか.

Hyperbolic Tree 視覚化アルゴリズムでは, ツリーのノードが Poincare 円板 D の中心にあるとき, そのノードに焦点が当たっているとする.

【命題 2】焦点性

Hyperbolic Tree 視覚化アルゴリズムで表示されたツリーは, そのノードのひとつが円板Dの中心にあるとき, そのノードにおいてユークリッド距離 $\frac{\exp(d)-1}{\exp(d)+1}$, 角度πで焦点性を有する.

【証明 2】

Hyperbolic Tree 視覚化アルゴリズムで表示されたツリーの任意のノードが支配的扇形の中心角がπであることはそのアルゴリズムより明らか. また, 任意の連結されたノード間の双曲平面距離を d とすると, その一端が円板 D の中心にあるとき, そのユークリッド距離は $\frac{\exp(d)-1}{\exp(d)+1}$ である.

以上より, 焦点性を有する.

【命題 3】均質性

Hyperbolic Tree 視覚化アルゴリズムで表示されたツリーは均質である.

【証明 3】

円板Dの中心を p とする. また, 距離 $r=\frac{\exp(d)-1}{\exp(d)+1}$, 角度 t =π とする. 証明 2 より, $\forall \mathbf{p_1}, \mathbf{p_2} \in P$ において, $f(\mathbf{p_1}, \mathbf{p_2})(\mathbf{v}) = \mathbf{p}$ ならば, 点 **p** において, 距離 r, 角度 t で焦点性を有する. ゆえにツリーは均質である.

【命題 4】連続性

Hyperbolic Tree 視覚化アルゴリズムにおける移動関数 f : P×P → V → P は連続である.

【証明 4】

定義より $f(\mathbf{p_1}, \mathbf{p_2}) = Tc_2 \cdot Tc_1$ である.

<u>C_1が円のとき</u>

c_1 を C_1 の中心とする.

$$\lim_{p_1 \to p_2} \mathbf{c_1} = \frac{1-|\mathbf{p_2}|^2}{2\mathbf{e} \bullet \mathbf{p_2}}\mathbf{e} + \mathbf{p_2}$$

となる. ただし, $\mathbf{p_1} = t\,\mathbf{e} + \mathbf{p_2}$ とする.

また, $\mathbf{c_2}$ を C_2 の中心とする. このとき,

$$\mathbf{c_2} = \frac{1-|\mathbf{p_2}|^2}{2\mathbf{e} \bullet \mathbf{p_2}}\mathbf{e} + \mathbf{p_2}$$

となり, $\mathbf{c_2} = \lim_{p1 \to p2} \mathbf{c_1}$ となる.

次に半径について r_1 を C_1 の半径とする.

$$\lim_{p1 \to p2} r_1^2 = \frac{(1-|\mathbf{p_2}|^2)^2}{(2\mathbf{e} \bullet \mathbf{p_2})^2}$$

また, r_2 を C_2 の半径とする.

$$r_2^2 = \frac{(1-|\mathbf{p_2}|^2)^2}{(2\mathbf{e} \bullet \mathbf{p_2})^2}$$

ゆえに $\mathbf{r_2} = \lim_{p1 \to p2} \mathbf{r_1}$ である.

以上より $\lim_{p1 \to p2} C_1 = C_2$ となり, 鏡像写像の性質より

$$\forall \mathbf{p_1}, \mathbf{p_2} \in P,\ \forall \mathbf{v} \in V.\ \lim_{P_1 \to P_2} f(\mathbf{p_1}, \mathbf{p_2})(\mathbf{v}) = \mathbf{v}$$

である.

<u>C_1が直線のとき</u>

Poincare 円板を回転させても一般性を失

わないので，$\mathbf{p_2}$ = (d, 0)とし，直線 C_1 の方向を($\mathbf{p_1}$ + $\mathbf{p_2}$)とする．

$\mathbf{p_1}$ = d(cos t, sin t)とすると，直線 C_1 の方向は($\mathbf{p_1}$ + $\mathbf{p_2}$) = d(1 + cos t, sin t)となる．

$t \to 0$ つまり $\mathbf{p_1} \to \mathbf{p_2}$ ならば直線 C1 の方向は(2d, 0) つまり(1, 0)となる．

一方 C_2 の中心 $\mathbf{c_2}$ は次のように表される．

$$\mathbf{c_2} = \mathbf{p_2} + \frac{(1-d^2)\begin{pmatrix}\frac{\cos t - 1}{t} \\ \frac{\sin t}{t}\end{pmatrix}}{d\,\frac{\cos t - 1}{t}}$$

$\mathbf{c_2}$ は $t \to 0$ で無限遠となり，その方向は(0, 1)となる．つまり，C_2 は直線とみなせその方向は(1, 0)となる．また，C_2 の半径と $|\mathbf{c_2}|$ の比は $t \to 0$ で 1 となり，C_2 は D の中心を通る．以上より，$t \to 0$ で $C_1 = C_2$．鏡像写像の性質より

$$\forall \mathbf{p_1}, \mathbf{p_2} \in P,\ \forall \mathbf{v} \in V.\ \lim_{P_1 \to P_2} f(\mathbf{p_1}, \mathbf{p_2})(\mathbf{v}) = \mathbf{v}$$

である．

【命題 5】追随性

Hyperbolic Tree 視覚化アルゴリズムの移動関数 $f : P \times P \to V \to P$ が追随性を有する．

【証明 5】

$f(\mathbf{p}_s, \mathbf{p}_d)(\mathbf{p}_s) = T_{C2} \cdot T_{C1}$ である．ただし，T_{C1} は $\mathbf{p}_s$ を $\mathbf{p}_d$ に移し，T_{C2} は $\mathbf{p}_d$ を $\mathbf{p}_s$ に移す．ゆえに $\forall \mathbf{p}_s, \mathbf{p}_d \in P.\ f(\mathbf{p}_s, \mathbf{p}_d)(\mathbf{p}_s) = \mathbf{p}_d$ である．

4. 議論

他のツリー(グラフ)視覚化アルゴリズムが前記性質 1 から 5 を満たしているかを表 1にまとめた．「○」は性質を満たすことが容易に証明できると思われるもの，「×」は性質を満たさない反例を挙げることができるもの，「?」は証明が容易でないもの，「---」はアルゴリズムの性質上，無関係な性質である．例えば Cone Tree が展望性を見たさないことは[9]で証明されている．また，DocSpace[12]のようにバネモデルによる視覚化アルゴリズムでは各性質の証明が難しい．たとえば，連続性について考える．バネモデルでいくつかの局所解が存在し，マウスでノードを移動すると局所解から他の局所解に「ジャンプ」してしまう可能性がないことを証明することは困難である．このように物理モデルを用いた視覚化アルゴリズムでは，一般に与えられた性質を証明することが困難である．一方，Fisheye View や Fractal View のように数学的に定義されたアルゴリズムでは証明が容易なことが多い．

「Win Explorer」は Windows での Explorer のように，フォルダ(i.e. ツリーのノード)を展開，折り畳みする機能とスクロール機能によりツリーを可視化するアルゴリズムを述べている．スクロール機能では，同時にすべてのノードを見ることができないので，展望性は当然満たさない．

表 1 視覚化アルゴリズムの比較

	性質 1	性質 2	性質 3	性質 4	性質 5
Cone Tree[5]	×	×	×	○	○
InformationCube[6]	×	○	○	○	○
Generalized Fisheye View[7]	○	○	×	○	○
H3[8]	○	○	○	○	○
Fractal View[9]	×	○	○	---	---
納豆 View[10]	---	○	○	○	○
TreeMap[11]	×	○	○	○	○
DocSpace[12]	?	?	?	?	○
Win Explorer (スクロール+折り畳み)	×	○	○	○	○

○：満たすことを証明できる． ×：満たさないことを証明できない． ？：証明が困難． ---：サポートせず，もしくは不明

5. まとめ

要求仕様を数学的に厳密に与えることで, それを満たす可視化アルゴリズムを選択する実例を述べた. 本件のように要求仕様を証明するには, Hyperbolic Tree 視覚化アルゴリズムが, 物理モデルではなく, 数学モデルに基づいていることが大きかった. 物理モデルで解析的に解ける場合ではなく, 発見的に解法を得ている場合であると, 要求仕様を満たすことを示すことが困難である. Hyperbolic Tree 視覚化アルゴリズムは Poincare 円板という数学モデルに基づいているために証明が容易であった. この点が数学的モデルに基づく視覚化アルゴリズムが物理モデルに基づくアルゴリズムや, モデルに基づかない発見的な視覚化アルゴリズムより有利な点である.

参考文献

[1] Toshio TONOUCHI, *Domain Browser Homepage*, http://www.doc.ic.ac.uk/~tton/DomainBrowser/, 2001

[2] N. Damianou, N. Dulay, E. Lupu, and M. Sloman *Ponder: A Language for Specifying Security and Management Policies for Distributed Systems*, 2001

[3] Marvin Jay Greenberg, *Euclidean and Non-Euclidean Geometries — Development and History*, W. H. Freeman and Company, 1973, ISBN 0-7167-0454-4

[4] John Lamping, Ramana Rao, and Peter Pirolli, *A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies*, CHI 95

[5] George Robertson, Jock D. Mackinlay, Stuart K. Card, *Cone Trees: Animated 3D Visualizations of Hierarchical Information*, CHI'91, pp. 189-194, April 1991

[6] 暦本, "Information Cube: 半透明表示を用いた 3 次元情報視覚化技法", WISS 93, pp 1--8, 1993

[7] W. Furnas, *Generalized fisheye views*, CHI '86, pp. 16--23. ACM Press, 1986.

[8] Tamara Munzner, *H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space*, Proc. of the 1997 IEEE Symposium on Information Visualization, pp. 2-10, October 1997

[9] Hideki Koike, Hirotaka Yoshihara, *Fractal Approaches for Visualizing Huge Hierarchies*, Proc. of 1993 IEEE Symposium on Visual Languages (VL'93), pp. 55-60, 1993

[10] 塩澤 秀和, 西山 晴彦, 松下 温, "「納豆ビュー」の対話的な情報視覚化における位置づけ", 情報処理学会論文誌, Vol. 38, No. 11, pp. 2331-2342, November 1997

[11] Brian Johnson, Ben Shneiderman, *Treemaps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures*, Proc. of the 2nd International IEEE Visualization Conference, pp. 284-291, October 1991

[12] 館村, "DocSpace: 文献空間のインタラクティブ視覚化", インタラクティブシステムとソフトウェア IV: 日本ソフトウェア科学会 WISS'96, pp. 11-20 , December 1996

[13] Julian Hatchwell, *Distributed Policy Management*, MSc thesis of Imperial College of Science, Technology and Medicine, September 2000