社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－DPS－122 (56)
2005－CSEC－ 28 (56)
2005／3／23

コンポーネント連携によるサービスを オーバレイネットワーク上で実現するための サービス設計技法の提案

中村嘉隆* 山口弘純* 廣森聡仁* 東野輝夫*

本稿では，サービスオーバレイを設計するための一つの形式的手法を述べる．提案手法では，サービスノード群とそれらの間のリンクから構成されるサービスオーバレイネットワークと，拡張ペトリネットによるサービス要求記述にもとづいて，サービス提供のために各サービスノードが協調して動作するための動作記述を自動導出する．その際，各サービスノードの計算能力の制約下で帯域利用率の集中が最小となるように，サービスコンポーネントの各サービスノードへの配置を最適化している．これによって，オーバレイネットワークリソースのバランスのよい利用が期待でき，ユーザへのレスポンス時間を改善することができる．提案手法について，実ネットワークを想定したシミュレーション実験を行うことにより，その有用性を評価した．

Proposal of Service Design Technique to Achieve Service by Component Cooperation on Overlay Network

Yoshitaka Nakamura*, Hirozumi Yamaguchi*, Akihito Hiromori* and Teruo Higashino*

In this paper, we propose a design technique to provide services by collaboration of service components on overlay networks. This technique automatically derives a set of behavior descriptions of service nodes' on overlay networks where service nodes can be connected with each other from a given description of service. The descriptions of service nodes' behavior specify how those service nodes collaborate to provide the required service. The allocation of service components to the service nodes is also optimized to achieve well-balanced bandwidth utilization under the constraints of performance on service nodes. The experimental results using a practical example and real network simulator shows the effectiveness of the proposed methodology.

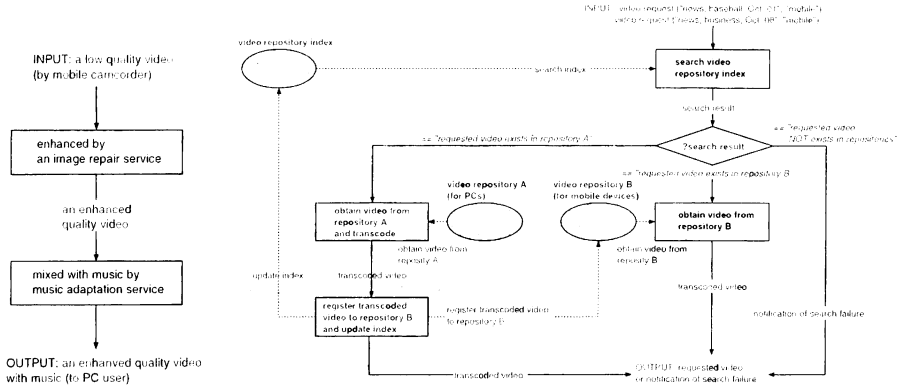
1 はじめに

近年の分散協調システムは，サービス提供者であるサーバ（以下，サービスノード）が仮想的なポイント間リンクにより結ばれたオーバレイネットワークアーキテクチャ上において，それらサービスノード上のサービスコンポーネント（例えばフィルタリングや動画像のトランスコーディング，プロキシなど）を連携させ，全体として高度なサービスを提供する形態で構成されることが多い．このような分散協調システムを以下ではサービスオーバレイと呼ぶ．これまでにサービスオーバレイの設計技法がいくつか提案されてきている．例えば，Xu ら [1] はクライアントの QoS 要求を考慮し，コンポーネントを線形的に接続したサービス（図 1(a)）をオーバレイネットワークにマッピングするアルゴリズムを提案している．また，Wang ら [2] や，Gu ら [3] はサービスオーバレイのクラスを線形結合から DAG へと拡張することで，より汎用性を高めている．

しかしこれらの既存手法は，携帯端末ユーザやデスクトップユーザなど有線や無線を利用してアクセスするユーザが混在し，それらが同時並行的にサービスを利用する分散協調システムを考慮した場合には十分でない．その理由の一つとして，サービスリソースと呼ばれる永続的なシステムデータの扱いを考慮していない点が挙げられる．たとえば図 1(b) は，PC ユーザ向けのビデオコンテンツを保持するレポジトリ A，モバイルユーザ向けの低ビットレート版レポジトリ B を用い，ユーザから与えられたキーワードに適合し，かつユーザ端末の再生可能ビットレートに応じた品質のビデオコンテンツを返すサービスを表している．これまでの線形的なサービスコンポーネントの結合（図 1(a)）と異なり，この例のようにレポジトリのような永続的なシステムデータを含む場合，サービスフロー自体が複雑化し，これに伴い，一貫性制御などサービスノード間のプロトコルが複雑化する．そのような複雑な制御フローを持つ一般的なサービスオーバレイのクラスにおいては，既存研究で採用されているグラフ上でのパスマッピングのような資源割り当て法を単純に適用できず，負荷分散のために最適化された新たな資源割り当て法が必要となる．

そこで本研究では，そのようなクラスのサービスオーバレイに対し，コストを最適化できる自動設計手法を提案している．提案手法では，サービスノード群とそれらの間の仮想リンクにより構成されるオーバレイネットワークと，拡張ペトリネットによるサービスの仕様記述に対し，そのサービスを提供するために各サービスノードが協調して動作するための，各サービスノードの動作記述を自動で導出する．さらに，サービスオーバレイのシステムコスト（帯域利用率のピーク値）を最小とするように，システムデータ（サービスリソース）のサービス

*大阪大学 大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University



(a) (b)

図 1: (a)Xu ら [1] のサービスオーバレイの例. (b) 本研究で対象とするサービスオーバレイの例（斜線の楕円で表されるサービスリソース及び条件分岐を含み，複雑なフローで実現される）.

ノードへの配置を最適化することができる．これにより，帯域利用率の集中を最低限に抑えることができ，ユーザへのレスポンス時間を改善することができる．

2 準備

2.1 サービスオーバレイアーキテクチャ

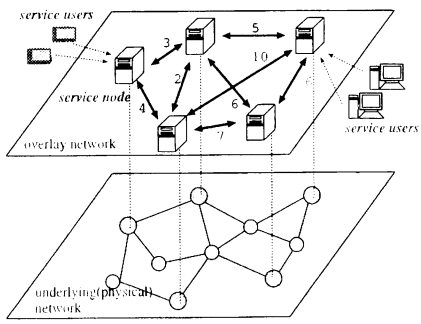


図 2: サービスオーバレイアーキテクチャ

サービスオーバレイのアーキテクチャはサービスコンポーネントが実行されるサービスノードと，オーバレイチャネルと呼ばれるそれらサービスノード間の仮想チャネル（ユニキャストチャネル）からなる．各サービスノードは計算能力についての独自の制限を持っており，サービス制限と呼ばれる非負整数として表される．また，各オーバレイチャネルの遅延や帯域，ネットワーク利用率に関するコストはオーバレイチャネルコストと呼ばれる非負整数で表される．

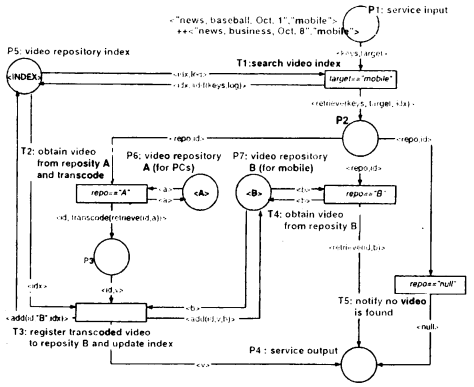
このアーキテクチャはノードがサービスノードを，辺がサービスノード間のオーバレイチャネルを表しているような無向グラフでモデル化できる．各サービスノードにはサービス制限が，各辺にはオーバレイチャネルコストが指定されている．

サービスユーザはサービスをサービスオーバレイネットワーク外から利用する．これらのユーザからはサービスオーバレイアーキテクチャは単一のサーバであるかのように見える．実際にはネットワークや計算リソースの制限のもとで協調して要求サービスを提供している．アーキテクチャ例を図 2 に示す．サービス制限は，サービスノードのマシン能力（CPU，メモリ等）から決定できる．また，チャネルコストは，遅延や帯域の逆数などで与えることができる．

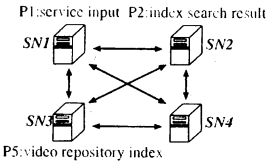
ユーザから要求を受け取ったサービスノードは現在利用できるサービスから要求されたサービスを探す．見つからなかった場合は，要求を受け取ったサービスノードが要求サービスについての動作記述を導出する．ユーザがサービスを容易に指定できるように，よく利用されるサービスの記述をあらかじめ準備しておくことも考えられる．また，リンク状態ルーティングのように，サービスノードの動作記述導出に用いられるオーバレイチャネルコストとサービス制限をサービスノード間で交換する．

2.2 サービスの記述とサービスノードの動作記述例

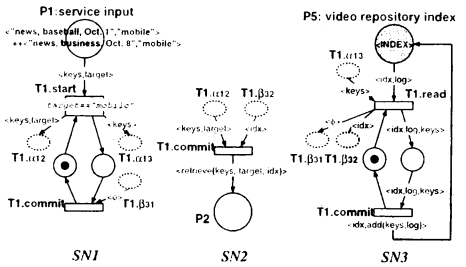
図 3(a) にサービス記述の例を示す．これは図 1(b) の例を拡張ペトリネットの一種である述語/トラン



(a) Pr/T ネットで記述されたサービス



(b) サービスオーバレイアーキテクチャ



(c) サービスノードの動作記述（トランジション $T1$ に対応する部分のみ抜粋）

図 3: 図 1(b) のサービスの Pr/T ネットによる記述とそれに対応する動作記述

ジションネット（Pr/T ネット）[4] で記述したものである．サービスを記述するために Pr/T ネットのクラスとして二種類のプレースを定義する．データベースのような恒久的なリソースをあらわすリソースプレース（図 3(a) の $P5$, $P6$, $P7$）は，内部でトークンの増減を行わない．それ以外のプレースをサービスフロープレースと呼ぶ．

図 3(b) に示すオーバレイネットワークとして構成されるサービスノード群は，一定のプロトコルに従って協調動作し，与えられたサービス要求を実現する．この際，各サービスノードが具体的にどのように動作すべきかを記述したものを動作記述と呼ぶ．ここで説明の簡単のため，例えば図 3(b) に示すように $P1$, $P2$, $P5$ はそれぞれサービスノード $SN1$, $SN2$, $SN3$ に配置されているとする．このような配置を仮定した上でのサービスノードの動作記述の一部（トランジション T_1 に対応する部分）を図 3(c) に示す．

動作記述では，サービスノード間の非同期高信頼性通信路（オーバレイチャネル）を表現するために，通信プレースと呼ばれるプレースを導入している．以降の図では，通信プレースは点線の丸印で表す．

サービスノード SNi 及び SNj の Pr/T ネットにおいて共通する名前 "$t.X_{i,j}$" を持つ 2 つの通信プレースはオーバレイチャネルを通って SNi から SNj に至るメッセージ通信を表すものとする．

トランジション $T1$ の動作は動作記述において以下のように記述される．まず，サービスノード $SN1$ は $P1$（ユーザからのリクエストのスプール）からトークン集合を取り出し，変数の組 "$\langle keys.target \rangle$" に割り当てる．ここで，この割り当てが $T1$ の条件を満たす場合，通信プレース $T1.\alpha_{12}$ 及び $T2.\alpha_{13}$ をそれぞれ経由して $SN2$ と $SN3$ にそれらの値が送られる．これは $SN3$ は $P5$ にトークンを生成するために "$keys$" の値が必要であることによる．これを受けて，$SN3$ は $P5$ からトークン集合を取り出す．$P5$ から取り出したトークンの値は $P2$ にトークンを生成するために必要であることから，"idx" の値は $T1.\beta_{32}$ を介して $SN2$ に送られる．$SN2$ は，$SN1$ 及び $SN3$ から受信した値を用いて，$P2$ にトークンを生成する．同様に，$SN3$ は $P5$ にトークンを生成する．ここで，$SN3$ は $SN1$ に対し，$SN3$ は $P5$ からトークン集合を取得したことを知らせるために，$T1.\beta_{31}$ を介してシンボル ϕ を $SN1$ に送信することに注意されたい．通知を受けたあと，$SN1$ では次のトークンが受理可能になる．

3 サービスノードの動作記述の自動導出と最適化

3.1 導出アルゴリズム

導出アルゴリズムの基本アイデアを示す．アルゴリズムは，Pr/T ネットによるサービス記述及びオーバレイネットワークのアーキテクチャ指定（サービスノード数，サービス制限やチャネルコスト）を入力として，各サービスノードの動作記述の組を自動的に導出する．説明の簡単のため，導出アルゴリズムではサービス記述のプレースからサービスノードへの割り当てが既に与えられているものとする．与えられたサービス記述の各トランジション t について，プレースの配置に基づき，t について最低 1 つの入力プレースを持つサービスノードを読み込みサービスノード，最低 1 つの出力プレースを持つサービスノードを書き出しサービスノードと呼ぶ．また，読み込みサービスノードの 1 つを主要サービスノードと呼ぶ．導出アルゴリズムは，任意のトランジションの振る舞いをサービスノード群によってシミュレートする手順を汎用的に記述したプロトコル（トランジション実行プロトコル）に基づき，各トランジション t の振る舞いを分解する．トランジション実行プロトコルに基づいてサービス記述のすべてのトランジションを分解することによって，最終的に各サービスノードの動作記述の組を得る．

t の実行可能性を調べる際に，t が実行可能，かつ SNi に割り当てられた t の各入力プレースに割り当て可能なトークン集合がある場合は，主要サービスノード（SNi と呼ぶ）はその一部を，出力プレースに新しいトークンを生成するために必要としている書き出しサービスノードに送る．これを α メッセージと呼ぶ．α メッセージはトークンを生成するために必要な値を送ると同時にトランジション t が選ばれて実行されることをサービスノードに通知する役割も果たす．読み込みサービスノード（SNj と呼ぶ）が α メッセージを主要サービスノードから受信したときは，SNj は SNj に割り当てられた t の入力プレースのうち，割り当て可能なトークン集合を選ぶ．その後 SNj は他の書き出しサービスノードにトークンを送る．この時に送られるメッセージを β メッセージと呼ぶ．β メッセージは SNj が入力プレースから選択されたことをサービスノードに通知するために用いられる．これによって読み込みサービスノードは新たな t の実行を受け入れる．

3.2 サービスオーバレイの最適化

導出アルゴリズムが用いるトランジション実行プロトコルは，プレースの配置が固定であることを想定している．ここで，導出アルゴリズムの適用前に，あらかじめプレース配置の最適化を以下のように行って，サービスオーバレイのコストを最小化する．

サービスコンポーネントを要求に応じてオンデマンドに結合するとしている既存研究とは異なり，本研究ではサービスは複数のユーザによってある程度継続的に利用されると仮定している．また，サービスが扱うシステムデータ（リソース）はマルチメディアファイルのような大容量ファイルであるとする．これらを考慮に入れた上で，提案手法ではオーバレイチャネルの帯域利用率のピーク値を最小化することで，システムの負荷分散を図る．まず，オーバレイチャネルコスト $cost(i,j)$ を伝送遅延，あるいは帯域の逆数とみなし，オーバレイチャネル (i,j) のコストを以下で定義する．

$$cost(i,j) * \sum_{t \in T} \sum_{v \in var(T)} size(v) * (\alpha^t_{i,j}[v] + \beta^t_{i,j}[v])$$

E はオーバレイチャネルの組，$\alpha^t_{i,j}[v]$ $(\beta^t_{i,j}[v])$ はトランジション t の実行において，SN_i から SN_j に α メッセージ（β メッセージ）を用いて変数 v が送信される場合のみ 1，それ以外は 0 である 0-1 整数（Boolean）変数である．今，上記のコストを $scost(i,j)$ と表記し，オーバレイチャネル (i,j) のサービスコストと呼ぶことにする．サービスオーバレイのコストは，全オーバレイチャネルの中でのサービスコストの最大値として，以下のように定義する．

$$\max_{(i,j) \in E} scost(i,j)$$

ここで一般に，ある 1 つのサービスは複数のユーザに同時並行的に利用されるものとし，各ユーザのサービスインスタンスは，他のユーザインスタンスとは独立して処理される（複数のインスタンスが並列に処理される）ものとする．これは，見かけ上逐次的であるサービスコンポーネントの組についても複数のユーザによって同時に実行される可能性があることを意味する．さらにはそれは複数のサービスコンポーネントによってオーバレイチャネルが同時に利用される可能性があることを意味する．従って，全サービスコンポーネントが複数ユーザににより同時に実行された場合に，各オーバレイチャネルの負荷がピーク値に達するとみなし，それをそのチャネルのサービスコストと定義している．また，オーバレイチャネル全体でサービスコストの最大値を最小化することでチャネル利用のピーク値を抑えることができ，全体のレスポンス時間改善につながる．

3.3 整数線形計画問題

以上のような仮定のもとで，サービスオーバレイのコストを最小化するようなプレースの最適配置問題を整数線形計画問題（ILP）に帰着する．

求解にあたり，前節で定義した0-1整数変数 $\alpha^t_{i,j}(v)$, $\beta^t_{i,j}(v)$ と，プレース p が SN_j に割り当てられた場合に限り1，そうでなければ0である ALC^p_j を用い，補助変数として $cost(i,j)$, $scost(i,j)$ と，SN_j がトラジション t の主要サービスノードである場合のみ1，それ以外は0であるような変数 $Spri^t_j$ を用いる．

[目的関数]

$$\min cost \tag{1}$$

[制約式] 定義から以下のような制約式が必要となる．

$$\forall(i,j) \in E;\ \ cost \geq scost(i,j) \tag{2}$$

$$\forall(i,j) \in E:$$
$$scost(i,j) = cost(i,j) * \sum_{t \in T} \sum_{v \in var(t)} size(v) * \{\alpha^t_{i,j}(v) + \alpha^t_{j,i}(v) + \beta^t_{i,j}(v) + \beta^t_{j,i}(v)\} \tag{3}$$

E はすべてのオーバレイチャネル集合，T はトランジション集合，$var(t)$ はトラジション t のアークで用いられる変数集合，$size(v)$ は変数 v のサイズ集合である．

以下の制約式で，α メッセージ（β メッセージ）を交換する SNi と SNj の組を指定する．例えば最初の制約式(4)では，(i) p と $p\prime$ がそれぞれ SNi と SNj に割り当てられ，(ii) SNi が主要サービスノードであり，(iii) $p\prime$ へのトークンを生成するのに変数 v が SNj で用いられる場合に，v の値を持った α メッセージが SNi から SNj に送られることを表す．$\bullet t$ ($t\bullet$) はトラジション t への入力（出力）プレースを表す．

$$\forall(i,j) \in E, \forall t \in T, \forall p \in \bullet t, \forall p\prime \in t\bullet,$$
$$\forall v \in var(L(p,t)) \cap var(L(t,p\prime));$$

$$\alpha^t_{i,j}(v) \geq ALC^p_i + ALC^{p\prime}_j + Spri^t_i - 2 \tag{4}$$
$$\beta^t_{i,j}(v) \geq ALC^p_i + ALC^{p\prime}_j - Spri^t_i - 1 \tag{5}$$

ここで，$L(p,t)$ ($L(t,p\prime)$) は p から t (または t から $p\prime$) へのアークのラベルである．

導出アルゴリズムに従うと，α メッセージと β メッセージの両方が，全てのサービスノードペアに同時に送られることはないため，以下の制約を得る．

$$\forall(i,j) \in E, \forall t \in T, \forall v \in var(t): \alpha^t_{i,j}(v)\beta^t_{i,j} \leq 1 \tag{6}$$

導出アルゴリズムでは，各プレースは必ずただ一つのサービスノードに割り当てられると仮定していることから以下の制約を得る．

$$\forall p \in P: \sum_j ALC^p_j = 1 \tag{7}$$

ここで P はプレース集合である．また，ただ一つのサービスノードが t の主要サービスノードになる．

$$\forall t \in T: \sum_j Spri^t_j = 1 \tag{8}$$

ある t の読み込みサービスノードが複数の出力トランジションを持つサービスフロープレース p を持つ場合，それは t の主要サービスノードとなる．それ以外の場合は，サービスフロープレースを持つ読み込みサービスノードの1つが主要サービスノードとなる．

$$\forall j \in S, \forall t \in T, \forall p \in \bullet t \cap Ps:$$
$$ALC^p_j = Spri^t_j \quad \text{iff } |p\bullet| > 1$$
$$ALC^p_j \geq Spri^t_j \quad \text{otherwise} \tag{9}$$

ここで，S はサービスノード集合であり，$Ps \subseteq P$ はサービスフロープレースの集合である．

また，与えられたサービス記述の実行負荷を分散するために，1つのサービスノードに割り当てられるプレース数を制限する．

$$\forall j \in S: \sum_{p \in P} ALC^p_j \leq limit(j) \tag{10}$$

$limit(j)$ は SNj のサービス制限である．

全てのサービスコンポーネントがどのサービスノードでも実行できるわけではない．すなわち，いくつかのサービスノードはあるサービスコンポーネントを実行する機能を持っていないかもしれない．例えば，ビデオレポジトリ p にはデータベース管理のような特殊機能が必要である．このようなプレース p に対し，次のような制約を与える．

$$\sum_{j \in run(p)} ALC^p_j = 1 \tag{11}$$

$run(p)$ はプレース p を割り当てることができるサービスノード集合である．

4 アプリケーション例を用いた性能評価

4.1 アプリケーション例

図4に映像の自動装飾及びトランスコードサービスの記述例を示す．このサービス例では，与えられたキーワードに基づき，映像にオープニングとエンディング映像が追加され，その後，MPEG4及びMPEG2フォーマットの動画に圧縮される．

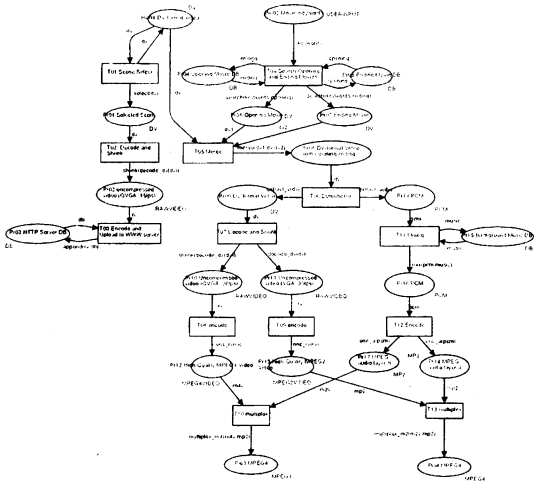


図 4: 映像の自動装飾及びトランスコードサービス

4.2 実験結果

我々は，カラーペトリネットの記述解析支援ツールである CPNtools[5] と協調して動作し，最適化された動作記述の導出を支援するツールを Perl で開発している．これを用いて，図 4 のサービス例に対し，5 つのサービスノードの動作記述集合を導出した．次に，高レベルペトリネットシミュレータ Maria[6] とネットワークシミュレータ [7] を連携させ，実環境シミュレーションにおける最適化動作記述のユーザあたりの平均レスポンス時間を計測した．比較のため，ランダムなプレース配置に基づき導出した動作記述と，試行錯誤を繰り返し人手で割り当てた手動配置に基づき導出した動作記述を用いた．図 5 はこれら 3 つの動作記述の平均レスポンス時間をグラフ化している．グラフから，提案手法による配置最適化の効果が現れていることがわかり，設計自動化と合わせた提案手法の優位性がわかる．

5 まとめ

本稿では，有線や無線を介してアクセスする複数のユーザが混在し，それらが同時並行的にサービスを利用するサービスオーバレイ分散協調システムの最適化自動設計技法について，その概略を述べた．今後の課題としては，オーバレイチャネル上での利用可能な帯域の変化に応じてよりアダプティブに QoS 制御を行う機構の導入及び評価などがあげられる．

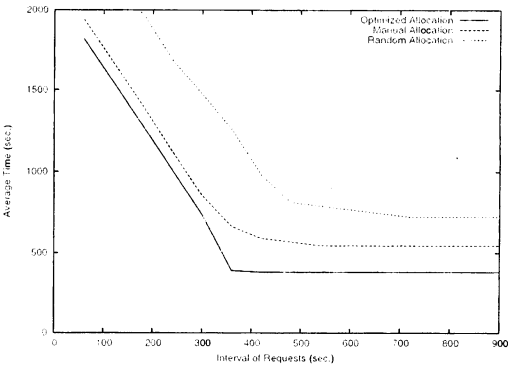


図 5: リクエスト到着間隔に対する平均レスポンス時間

参考文献

[1] D. Xu and K. Nahrstedt. Finding service paths in an overlay media service proxy network. In *Proceedings of the International Conference on Multimedia Computing and Networking 2002 (MMCN2002)*, 2002.

[2] M. Wang, B. Li, and Z. Li. sFlow : Towards resource-efficient and agile service federation in service overlay networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS2004)*, pp. 628–635, 2004.

[3] X. Gu, K. Nahrstedt, and B. Yu. Spidernet : An integrated peer-to-peer service composition framework. In *Proceedings of the 13th IEEE International Symposium on High-Performance Distributed (HPDC-13)*, pp. 110–119, 2004.

[4] H. J. Genrich and K. Lautenbach. System modelling with high-level petri nets. *Theoretical Computer Science 13*, pp. 109–136, 1981.

[5] A. V. Ratzer, L. Wells, H. M. Lassen, M. Laursen, J. F. Qvortrup, M. S. Stissing, M. Westergard, S. Christensen, and K. Jensen. Cpn tools for editing, simulating, and analyzing coloured petri nets. In *Proceedings of 24th International Conference on Application and Theory of Petri Nets*, 2003.

[6] Maria -the modular reachability analyzer-. `http://wwww.tcs.hut.fi/Software/maria/`.

[7] G. F. Riley. The georgia tech network simulator. In *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pp. 5–12, 2003.