研究業績目録

A. 学術論文誌論文

1. T. Moritsu and M. Kato: “Disparity Mapping Technique and Fast Rendering Technique for Image Morphing,” *IEICE Transactions on Information and Systems*, Vol. E83-D, No. 2, pp. 275–282 (2000).

2. T. Nakae, T. Moritsu, K. Baba, and N. Komoda: “A Design and Evaluation of Needs-Based Collateralized Debt Obligation,” *WSEAS Transactions on Information Science and Applications*, Vol. 3, Issue 1, pp. 105–112 (2006).

3. 島村敦司, 森津俊之, 染谷治志 : “非ツリー状に接続された証券管理機関における証券移転の経路長と証券保有木の最小化,” 電気学会論文誌 C (電子・情報・システム部門誌), Vol. 126, No. 4, pp. 506–512 (2006).

4. 中江達哉, 森津俊之, 薦田憲久: “投資条件に合わせた債務担保証券設計方式,” 電気学会論文誌 C (電子・情報・システム部門誌), Vol. 126, No. 8, pp. 1026–1032 (2006).

5. T. Nakae, K. Baba, T. Moritsu, and N. Komoda: “Fast Calculation Method for Fine-tuned Design of Needs-based Collateralized Debt Obligation (CDO),” *IEEJ Transactions on Electrical and Electronic Engineering*, Vol. 2, Issue 3, pp. 379-386 (2007).

6. 森津俊之，薦田憲久: “整数線形計画を用いた電子債権譲渡における期日変更マッチング最適化方式,” 電気学会論文誌 C (電子・情報・システム部門誌), Vol. 128, No. 4, pp. 569–575 (2008).

B. 国際会議

1. T. Moritsu, K. Hisaki, A. Takahashi, and M. Kato: "Noh mask - Application of Image-based Rendering," in *Electronic Art and Animation Catalog of ACM SIGGRAPH'98*, p. 154 (1998).

2. T. Moritsu and M. Kato: "Disparity Mapping Technique and Fast Rendering Technique for View Morphing," in *Proc. of Sixth Pacific Conference on Computer Graphics and Applications(Pacific Graphics'98)*, pp. 216–217 (1998).

3. T. Moritsu, A. Shimamura, H. Mizuno, H. Someya, and K. Takeuchi: "Protocol of the Book Entry System based on the Arrival Principle," in *Proc. of International Conference on Electrical Engineering (ICEE 2004)*, OE 6-1, Vol. 1, pp. 170–175 (2004).

4. H. Someya, T. Murata, O. Yoshie, and T. Moritsu: "Scenario Setup Support Functions in Scenario-based Simulation," in *Proc. of International Conference on Electrical Engineering (ICEE 2004)*, OE 6-1, Vol. 1, pp. 109–113 (2004).

5. T. Nakae, T. Moritsu, K. Baba, and N. Komoda: "A Method for Designing CDO Conformed to Investment Parameters," in *Proc. of 4th WSEAS International Conference on E-ACTIVITIES (E-ACTIVITIES '05)*, pp. 163–169 (2005).

6. T. Moritsu, M. Hiltunen, R. Schlichting, J. Toyouchi, and Y.Namba: "Using Web Service Transformations to Implement Cooperative Fault Tolerance," in *Proc. of The 3rd International Service Availability Symposium (ISAS 2006)*, pp. 76–91 (2006).

7. T. Nakae, K. Baba, T. Moritsu, and N. Komoda: "Speed Up Optimization for CDO Design Method Conformed to Investor Needs," in *Proc. of*

The Third IASTED International Conference on Financial Engineering and Applications (FEA 2006), pp. 36–41 (2006).

8. T. Moritsu and N. Komoda: "An Optimization Method for Redemption and Due Date Matching in Assignment of Electronic Receivables by Using Integer Linear Programming," in *Proc. of International Conference on e-Business (ICE-B 2008)*, pp. 349–356 (2008).

C. 学会講演

1. 加藤誠, 久木和也, 森津俊之: "多視点・多光源位置撮影画像を用いたイメージベースレンダリング技術," 情報処理学会全国大会講演論文集, 第58回平成11年前期, pp. 4-235–4-236 (1999).

2. 中江達哉, 森津俊之, 馬場健治, 島村敦司, 難波康晴: "投資条件に合わせた二次証券ポートフォリオ構築方式," 電気学会情報システム研究会, IS-05-27, pp. 53–58 (2005).

3. 中江達哉, 馬場健治, 森津俊之, 薦田憲久: "投資条件合致型債務担保証券設計方式の高速化," 電気学会情報システム研究会, IS-05-48, pp. 29–34 (2005).

4. 森津俊之, 豊内順一, 難波康晴, リチャード シュリクティング, マティ ヒルトネン : "変換ウェブサービスの半自動生成技術を用いた類似サービス間フェィルオーバ方式," 電気学会情報システム研究会, IS-07-3, pp. 11–16 (2007).

5. 森津俊之, 薦田憲久: "電子債権譲渡における期日変更マッチング最適化方式," 電気学会情報システム研究会, IS-07-30, pp. 25–29 (2007).

D. その他

1. 中川浩子, 坂尾秀樹, 森津俊之，西美千子: “高品質三次元コンピュータグラフィクス技術の応用 -ディジタルアーカイブへの適用例-,” 日立評論, Vol. 80, No. 7, pp. 25–30 (1998).

内容梗概

本論文は，筆者が 1995 年から現在まで (株) 日立製作所システム開発研究所，ならびに 2007 年から現在まで大阪大学大学院情報科学研究科マルチメディア工学専攻在学中に行った，電子商取引システムの柔軟な提供に関する研究成果をまとめたものである．

1990 年代半ばから本格的に普及が始まった電子商取引が成熟期を迎えつつある．市場が成熟する中で，顧客ニーズの細分化に伴うサービス・商品の種類，提供形態，決済手段の多様化が進んでおり，これを実現する柔軟性の高いシステムの実現が求められている．

この背景を踏まえ本論文では，電子商取引システムの 3 つの柔軟化に着目する．第 1 の柔軟化は，電子商取引のユーザインタフェースに関わるものである．電子商取引システムでは，ユーザに商品情報を正確に伝達する必要があり，現実に近い画像で，かつ任意視点での商品紹介が行えることが好ましい．そこで，実写画像による視点変更を実現するモーフィング技術に着目し，その合成画像のぼやけを防ぐことと，リアルタイムでのレンダリングを可能にすることを実現する．画像のぼやけの課題に対しては，その原因が入力画像間の不正確な対応付けであることに着目し，合成画像の品質向上に寄与する画像対応付け方式を提案する．リアルタイムのレンダリングの実現という課題に対しては，画像合成時に画素単位で画像処理を行うことが原因であることに着目し，汎用の 3 次元アクセラレータを利用したモーフィングのレンダリング方式を提案する．

次に第 2 の柔軟化は，電子商取引システムのバックアップに関するものである．近年様々なサービスがネットワーク上に構築されており，これらの中には類似するサービスが数多く含まれる．このような中で，特に情報投資が限られる中小の企業を対象に，他社のサービスを活用したバックアップ形態が今後実

現されていくものと想定し，それがどの程度のオーバヘッドで実現できるかを明らかにする．具体的には，類似サービスの差異を吸収する変換ウェブサービスを生成する仕組みを実現することで，協業する他社のサービスをバックアップ先として活用することを可能にし，そのオーバヘッドを明らかにする．

続いて第 3 の柔軟化は，電子債権の電子商取引決済への柔軟な活用に関するものである．電子債権を用いた債権譲渡による支払では，支払を行う企業は手元資金を用意しなくてよい，支払に関して振出人の企業の信用力を利用することができるといったメリットがある反面，電子債権の償還期日と支払の支払期日に違いがある場合には債権譲渡を行いにくいといったデメリットがある．そこで期日マッチング方式を実現することで，電子債権の債権譲渡の利用機会を高め，その活用範囲を広げる．

本論文は全 5 章で構成する．第 1 章の序論では，電子商取引システムの概要を述べ，その柔軟性を高める上での課題について述べる．さらにそれらの課題に対する従来研究を概観し本研究の位置づけを明らかにした上で，本研究の方針を述べる．次に第 2 章から第 4 章では，前述した 3 つの具体的な柔軟化に関して提案を行う．最後に，第 5 章では，結論として本研究で得られた成果を要約し，今後に残された課題について述べる．

目 次

第1章

序論

1.1 研究の背景

インターネットや WWW (World Wide Web) 技術の発展により，1990 年代半ばより本格的に普及が始まった電子商取引 [1] が成熟期を迎えつつある [2]. その市場規模は，2006 年時点において 235 兆円であり，1998 年時点の 9.3 兆円と比較すると約 25 倍にまで拡大している [3]．市場が成熟する中で，顧客ニーズの細分化に伴うサービス・商品の種類，提供形態，決済手段の多様化が進んでおり，これを実現する柔軟性の高いシステムの実現が求められている [4].

図 1.1 は，一般的な電子商取引システムの全体像を示すものである．電子商取引システムを大別すると，顧客にサービス (商品も含む) を販売・提供するサービス販売・提供機能と代金の決済を行う決済機能からなる．サービス販売・提供機能は，顧客との窓口を実現するフロントエンドシステムと後方事務を担うバックエンドシステムから構成される．フロントエンドシステムは，目的のサービスまで顧客をナビゲートする顧客ナビゲート機能，提供するサービスや商品がどのようなものかを顧客に提示し販売するサービス内容提示・販売機能，またオンライン上で提供可能な情報コンテンツの場合はその場でサービスを提供するサービス提供機能により構成される．またバックエンドシステムは，マーケティング，販売管理，在庫管理などの事務管理の機能により構成される．決済機能は，法人向けの決済手段と，個人向けの決済手段に大別できる．

法人向け決済手段は，法人顧客や仕入れ先への決済手段であり，振込決済や今後普及が期待される電子債権決済などがある．一方で個人向け決済手段には，電子マネー決済，クレジット決済などがある．

本論文では，電子商取引システムの 3 つの柔軟化に着目する．第 1 の柔軟化は，電子商取引のユーザインタフェースに関わるもので，実写画像を用いた視点自由度の高いユーザインタフェースの実現である．

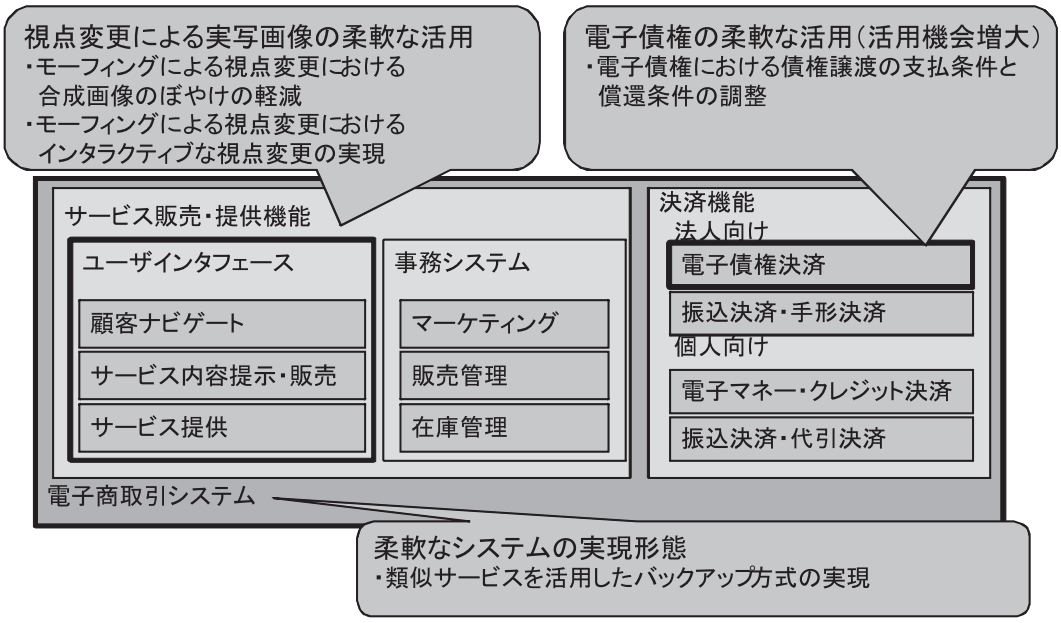


図 1.1: 一般的な電子商取引システムの全体像

近年のコンピュータの処理向上に伴い，高精細な CG (Computer Graphics) 画像を目にする機会が多くなった．これらの多くは三次元モデリングに基づく幾何学ベースの CG 技術により実現されている．しかし，幾何学ベースの CG 技術では，モデリングや質感の表現が困難であったり手作業によるモデリング作業量の問題から，人の肌，髪の毛，風景など，表現が難しい対象物が存在する．そこで対象物を選ばず，かつリアリティの高い画像を合成する技術として実写ベース CG 技術が注目されている [5]．実写ベース CG 技術は，Image-based Rendering (IBR)[6] と，Image-based Modeling に大別できる．IBR は，ある一連の画像群から新たな画像を直接合成する手法であり，モーフィング [7]，光線空間法 (Light Field Rendering)[8] がある．Image-based Modeling は，画像

群から三次元モデルを作成するものでありステレオ法 [9][10] がこれにあたる．この中でモーフィング技術は，ある物体から別の物体へ自然に変形する画像を見せる技術であり，特殊な撮影機材を必要としないことから，映画などの多くの特殊効果で活用されている．

モーフィング技術による視点変更 [7][11] や，光源位置変更技術 [12] を電子商取引システムに応用することで，商品のプレゼンテーションなど，実写画像を用いたインタフェースを実現することができる [13][14]．

電子商取引システムでは，ユーザに商品情報を正確に伝達する必要があり，現実に近い画像で，かつ任意視点でのインタラクティブな商品紹介が行えることが好ましい．しかし，従来のモーフィング技術は，合成画像のぼやけを防ぐことが難しいことと，リアルタイムでのレンダリングが難しいという課題がある．従来，モーフィング技術は，異なる画像間を連続的に変化させるアニメーション効果を生む技術 (例えばある 2 人の人間の顔を連続的に変化させる) として活用されるケースが多いことから，合成画像のぼやけを防ぐというニーズやリアルタイムに画像を合成するというニーズは低い．しかし，電子商取引分野にモーフィング技術を応用する上では，合成画像のぼやけ軽減とリアルタイムのレンダリングを行うことが不可欠と考え，この 2 つの課題を解決することを第 1 の研究の目的とする．

第 2 の柔軟化は，電子商取引システムのバックアップに関するものである．システムの高信頼化に関する要素研究は数多く存在する．例えば通信メッセージの高信頼化技術 [15] や，システムの多重化技術 (グループ管理，障害検知，リカバリ管理) [16][17] があげられる．一方，近年様々なサービスがネットワーク上に構築されており，これらの中には類似するサービスが数多く含まれる．また，近年アウトソーシング [18] や SaaS (Software as a Service) [19] の考え方も浸透しつつあり，全てのソフトウェアを自ら所有するのではなく，他の提供するサービスを積極的に利用してシステムを実現する方向で進みつつある．本研究では，特に IT (Information Technology) 投資か限られる中小の企業を対象に，他社のサービスを活用したバックアップ形態が今後実現されていくものと想定し，それがどの程度のオーバヘッドで実現できるかを明らかにする．

具体的には，類似サービスの差異を吸収する変換ウェブサービスを生成する仕組みを実現することで，協業する他社のサービスをバックアップ先として活用するといったフェィルオーバを可能にし，そのオーバヘッドを明らかにする．

第 3 の柔軟化は，電子債権 (法律上の名称は電子記録債権) の電子商取引決済への柔軟な活用に関するものである．近年，米国での EBPP / EIPP (Electronic Bill/Invoice Presentation and Payment)[20] や韓国での電子手形法 [21] など各国で債権の電子化が進展している．日本でも 2002 年の CP (Commercial Paper: 現在の短期社債)，2003 年社債の電子化，2009 年に開始される株券の電子化など，有価証券の電子化が大企業を中心に着実の進展している．その一方で，日本の企業数の約 99 %，従業員数で約 70 %を占める中小企業金融の活性化が急務となっている [22]．中小企業金融の活性化に向けては，これら企業が資金決済もしくは資金調達の目的で発行する債権の流通促進が求められており，その解決策として，指名債権や手形の電子化が，債権の二次証券化 [23][24][25][26][27] と共に熱望されている [28]．これを受けて電子記録債権法が 2008 年 12 月に施行された [29]．電子債権は，これまで紙ベースでやりとりされた手形を置き換えるものであり，管理機関への登録により債権の発行，流通，消滅を行うものである [30][31].

電子債権による決済のメリットとして，債権譲渡が可能という点があげられる．企業が保有する電子債権を支払に充当する債権譲渡は，譲渡人は振出人の信用力を活用できるメリットがある [32]．しかし，これまでの指名債権，手形債権では，保有する債権を債権譲渡に活用できる機会は限定的であることが指摘されている [33]．その理由の一つとして，支払日と電子債権の償還日が必ずしも一致しないことがあげられる．本研究では期日マッチング方式を実現することで，電子債権の債権譲渡の利用機会を高め，その活用範囲を広げることを第 3 の研究の目的とする．

以上述べたように，本論文では，電子商取引システムの柔軟化に関する課題として，実写画像を用いたぼやけが少なくリアルタイムのレンダリングが可能なユーザインタフェースの実現と，類似サービスを活用したバックアップ方式の実現ならびにそのオーバヘッドの明確化と，電子債権の期日マッチング方式

を実現の 3 点を取り上げ，情報技術を用いてこれらを解決する．

1.2 関連研究

1.2.1 モーフィング技術に関する関連研究

入力画像間の対応付けを自動的に求める研究は Computer Vision の分野を中心として様々に行なわれている．例えばパターンマッチングによる方法 [34][35] がある．これは入力画像に写る同一の特徴点を抽出して，それらを自動で対応付ける技術である．しかし，対象物に影響されずに対応付けを安定的に求めるのは難しく，現状において実用化されているモーフィング技術では人手で対応を付けるものがほとんどである．人手により入力画像間の対応を求める技術として，入力画像のエッジ部分 (色あるいは濃淡の変化の大きい部分) を線分や点で指定し，線分上以外の画素の対応は指定した線分から推定する研究がある [36][37][38]．これらの手法は，オペレータは対応する線分や点を適宜指定するだけで中間画像が生成できるというシンプルな特徴を持つ反面，顔の輪郭，円周などの曲線をなすエッジの対応を指定する場合，複数の線分を用いて指定する必要がある．この為，線分上に乗っていないエッジが発生する可能性があり，これが合成画像がぼやける原因になりうる．モーフィング技術を電子商取引システムに応用し，ぼやけの少ない合成画像を生成するためには，入力画像のエッジ全域にわたり，正確に対応付ける技術の実現が必要である．

一方，モーフィングのレンダリング処理の高速化に関する研究として，Mesh Warping という技術が知られている [39]．この技術では，オペレータからの入力に従い，入力画像を Mesh に分割する．さらに各 Mesh の頂点の他の入力画像における移動先の入力を受ける．これらの情報をもとに Mesh 単位で画像を変形することで高速なレンダリングが可能となる．また別の技術として，人の顔などの変形物の形状がある程度パターン化できるものについては，あらかじめ用意した 3 次元形状モデルに画像をマッピングした上で視点変更を行う手法が知られている [40]．この手法では， 3 次元形状モデルにより変形を行うため視点変更の自由度が高い．これらの手法はいずれも任意の対象物に対して高

速にレンダリングすることは難しいという課題がある．Mesh Warping では，人手により Mesh を指定するため複雑な形状の対象物を正確に対応づけて変形することは難しい．また 3 次元形状モデルへ画像をマッピングする方法では，予め形状が用意してある対象物のみが対象となる．このため電子商取引システムで，任意の対象物に対してインタラクティブに視点変更を可能にするには，汎用的な対象物に対してモーフィングのレンダリング処理を高速化できる技術の実現が必要である．

1.2.2 類似サービスを活用したバックアップ方式に関する関連研究

類似サービスを活用したバックアップを可能にする変換ウェブサービスの実現方法として，サービスを動的に構成する方法 (Dynamic Composing) とサービスを静的に構成する方法 (Static Composing) の 2 種類の方法がある [41]. サービスを動的に構成する研究は，DAML-S[42], OWL[43] といったオントロジ記述を元にコンポーネントを実行時に結合し目的とするサービスを構成する [44][45][46][47]．動的にサービスの構成を行う技術では，互換性のあるサービスの検索，実行パスの作成を実行時に行う．これにより動的なサービスの入れ替えが可能となり柔軟性が高い．一方，静的にサービスの構成を行う技術の研究もおこなわれている [48][49]．これらの技術は，変換フローの定義に基づいて，サービスをコンパイル時に構成する．このため動的にサービスを構成する方法と比較して，実行時の柔軟性は低いものの，パフォーマンスの面で優れる．これらのサービス構成技術のバックアップへの応用については，例えば通貨の変換など，部品レベルで代替サービスへ切り替えるといった検討はなされている [44]．しかし証券取引システム，銀行システムなど，システム全体として，どのように類似サービスへバックアップを行い，またその実行パフォーマンスがどの程度になるかを分析する研究は見当たらない．

1.2.3 電子債権に関する関連研究

電子債権の法制度は，電子債権を活用した様々なサービスの提供が可能となるように基本的な枠組みだけが規定されている．このため利便性向上を目指した電子債権の様々な活用方法に関する検討がなされている．例えば，複数管理機関による電子債権の分散管理方法 [50] や，到達主義に基づく電子債権管理機関の実現 [51] や，Peer to Peer 方式による電子債権の管理方式 [52] などがある．しかし，これらの研究は，電子債権の実現方式に関するものである．電子債権の応用サービスに関する研究としては，企業グループ内の資金管理である CMS (Cash Management System) や，複数企業団による融資であるシンジケートローンなど，既存サービスへ電子債権を応用する検討がなされている [53][54]．しかし電子債権の電子的管理という特徴を生かした新たなサービスの研究は見当たらない．特に，今後電子債権の利用促進に向けては，振出人の信用力を補完し，電子債権の流通を活性化する債権譲渡の機能強化が望まれるが，これに関する研究はこれまで行われていない．

1.3 研究方針

図 1.2 に電子商取引システムの柔軟化に対し，本論文で提案する解決策の位置づけを示す．

1.3.1 モーフィングによる視点変更における合成画像のぼやけ軽減とインタラクティブな視点変更の実現

画像の対応付けに関して，合成画像のぼやけは入力画像間のおけるエッジ部分の対応が正しく取れていないことによる影響が大きいことに着目し，エッジ部分全域にわたりその対応付けを効率的に行う方法を実現する．具体的には入力画像をエッジに沿って複数領域に分割し，入力画像間で領域の境界線同士の対応付けを行う方式を実現する．

一方，モーフィングにおける高速レンダリング方式に関しては，既存の 3D (3 次元: Three Dimentional) ハードウェアを用いてモーフィングが行なえるようにモーフィングデータをデータ変換し高速化を実現する方法を提案する．さらに前記データ変換により実現できるレンダリング方式を実装し，そのパフォーマンスについて評価を行う．

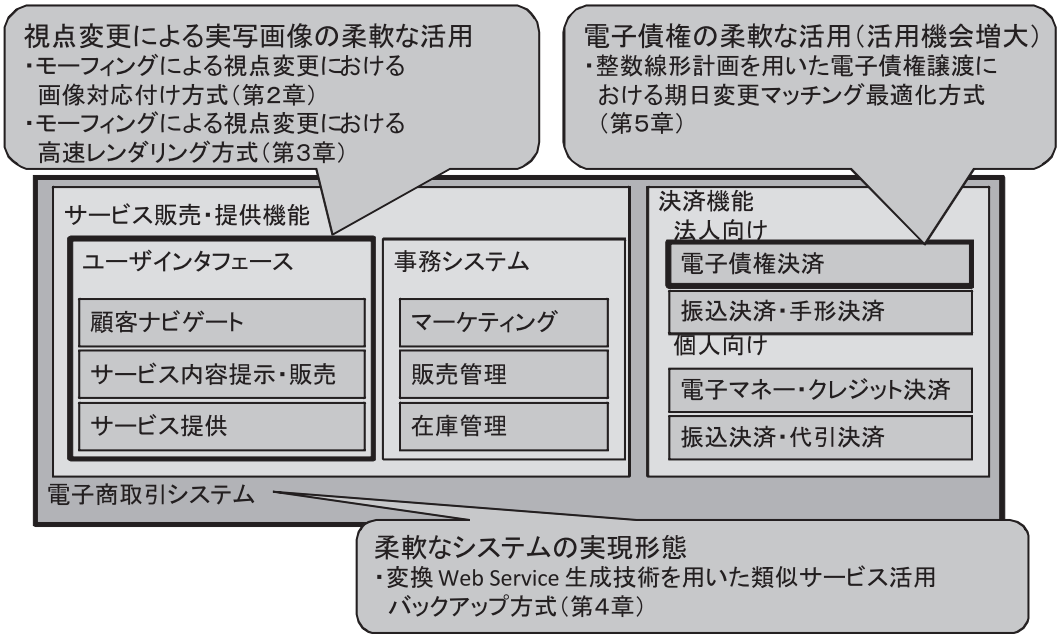


図 1.2: 電子商取引システムに対する柔軟化の解決策の位置づけ

1.3.2 類似サービスを活用したバックアップ方式におけるオーバヘッド検証

企業の枠組みを越え類似サービスを活用したバックアップ方式の実現を目指し, そのオーバヘッドを明らかにする．具体的には，類似サービスの差異を吸収する変換ウェブサービスを静的に半自動で生成する技術の開発を行ない，そのオーバヘッドを明らかにする．変換ウェブサービスは，オリジナルサービス (プライマリとバックアップ) のインタフェース記述情報，変換ロジック群，変換フロー定義から変換ウェブサービスのソースコードから生成する．具体的には，まずオリジナルサービスのインタフェース記述情報を，それらサービスに

付随する公開情報である WSDL (Web Service Definition Language) [55] から抽出する．さらに開発者が GUI ツールを用いて，一連の変換ロジック群の中から適切なロジックを選択し，プライマリとバックアップのインタフェース間を結合する変換フローを構成することで，変換ウェブサービスのソースコードを生成する．また，一連の変換が既存の変換ロジックだけで構成できない場合も想定されることから，開発者が独自に変換ロジックをプラグインとして追加することを可能とする．さらに，証券取引を具体例としたサンプルアプリケーションにより，上記バックアップ方式の性能評価を行う．

1.3.3 電子債権の債権譲渡における支払日と償還日の調整

電子債権の債権譲渡の利用機会を向上することを目的とし，支払の支払日と電子債権の償還日とをマッチングさせる方式を実現する．具体的には，参加者が事前に登録する資金の増減目標と期日の変更を許容する変更条件を元に，これらの制約を満たす電子債権と支払の組み合わせを決定する．その上で全参加者の資金の増減目標を極力達成するように電子債権と支払の組み合わせを最適化するマッチング方式を提案する．特にこの電子債権と支払とのマッチングでは，単にどの電子債権とどの支払を組合せるかだけでなく，新しい期日 (償還日/支払日) をいつに設定してマッチングさせるかによっても参加者の資金の目標達成度に影響を与える．そこで電子債権と支払の組み合わせ問題を，それらの組合せにより選択範囲が決定される新期日の選択も含めて定式化し，その結果が整数線形計画問題としてとらえることができることを示す．

また国内の企業の手形や売掛債権の発行状況を模した状況下で電子債権取引シミュレーションを行い，まず期日マッチング機能によりどの程度債権譲渡が促進できるかを検証する．さらに最適化したマッチング結果を，他のいくつかマッチング方式と比較して最適化の効果を検証する．

1.4 本論文の構成

本論文では，第 2 章以降を以下のように構成する．

第 2 章では，実写ベース CG 向け画像対応付けと高速レンダリング方式について，文献 [56][57][58] で公開した結果に基づき議論する．画像対応付けに関しては，合成画像のぼやけの軽減を目的として，入力画像間のエッジ部分に沿った対応付け方式を提案する．さらに提案した対応付けによる合成画像の品質の評価を行う．一方，高速レンダリングに関して，データ変換により実写ベースのレンダリングを 3D アクセラレータを活用して行う方式を提案する．さらに提案方式に基づいて合成した画像の品質と合成時間について評価を行う．

第 3 章では，類似サービスを活用したバックアップのオーバヘッドの検証について，文献 [59][60] で公開した結果に基づき議論する．企業の枠組みを越え類似サービスへのフェイルオーバを可能にすべく，類似サービスの差異を吸収する変換ウェブサービスの生成技術の開発を行う．さらに具体的に証券取引を題材とし差異を吸収する変換ウェブサービスを生成し，生成した変換ウェブサービスのパフォーマンスの測定を行う．

第 4 章では，整数線形計画を用いた電子債権譲渡における期日変更マッチング最適化方式について，文献 [61][62][63] で公開した結果に基づき議論する．債権譲渡のメリットに着目し，電子債権を活用した債権譲渡による支払いを促進するための電子債権の償還日と支払の支払期日のマッチング方式を提案し，整数線形計画問題として定式化できることを示す．さらに日本国内の債権の発行状況を元にシミュレーションを行い期日マッチングそのものの効果，最適化の効果，さらには最適化のための計算時間について評価を行う．

最後に，第 5 章では，結論として本研究で得られた成果を要約し，今後に残された課題について述べる．

第2章

モーフィングによる視点変更における画像対応付け方式と高速レンダリング方式

2.1 緒言

本章では，電子商取引システムのユーザインタフェースの柔軟化に関わる，モーフィングによる視点変更における画像対応付け方式と高速レンダリング方式について述べる．

実店舗では，顧客であるユーザは商品の見定めるのに，商品を手に取り任意の視点からこれらを観察することができる．電子商取引システムにおいても，これと同等の視覚効果の提供ができれば，従来の静止画像を用いる場合と比較して，ユーザに対し，より正確な商品イメージを伝えることができる．また美術品など鑑賞目的においても，任意視点の実写画像を提供することで，より現実感の高い仮想空間が提供できる．このようなユーザインタフェースを実現するには，任意視点での高品位な画像をインタラクティブに生成する必要がある．

モーフィング技術を用いてこれを実現する上で 2 つの課題がある．第一の課題は合成画像のぼやけの問題である．モーフィングで，ぼやけの少ない画像を合成するためには，入力となる画像間の対応付けをいかに正確に行うかが重要となる．そこで，入力画像のエッジに沿って正確に対応付ける方式を提案

し，合成画像のぼやけを軽減する．具体的には入力画像をエッジに沿って複数領域に分割し，入力画像間で領域の境界線同士の対応付けを行う．さらに境界線の対応情報から境界以外 (エッジ以外) の対応を生成する．まず領域分割に関しては，エッジに沿った画像分割に関する研究は古くから行なわれている分野であり [66][67]，また市販のレタッチソフトの機能でも領域分割の機能を含むツールも市販されていることから，これらのツールを使い領域分割を行う．次に各領域の境界線上のいくつかの特徴点の対応の入力をオペレータより受け，これらの対応情報を元に境界線全域に対する対応を求める．これにより入力画像のエッジは必ず他の入力画像のエッジに結び付くために合成画像のぼやけは生じにくくなる．上記対応付け方式を実装し，合成画像の品質に関して評価を行う．

第二の課題は画像合成時間の問題である．本章では，モーフィングデータを変換し 3D アクセラレータでレンダリングを可能にすることで，対象物を選ばずにモーフィングのレンダリング処理を高速化する方式を提案する．モーフィングによる画像合成処理は，入力画像の各画素に対して，合成画像の画素位置への移動 (Warping) と，他の入力画像の画素とのミキシング (Cross-Dissolve) を行う．これらの画素ごとの画像処理は計算量が多く，通常の汎用の CPU で実行し，リアルタイムで画像を合成することは難しい．そこで既存の 3D アクセラレータを用いたモーフィングを可能にするデータ変換方式を提案する．具体的には，Warping 処理を 3D アクセラレータのテキスチャ付きのポリゴンの変形に置き換え，さらに Cross-Dissolve 処理を 3D アクセラレータの半透明な物体の表現手法である α-blending に置き換えレンダリングを行う．

以降，2.2 節で画像の対応付け方式について述べ，2.3 節で，高速レンダリング方式について述べる．さらに 2.4 節で提案方式に対する評価を行い，2.5 節にて，これらの技術を応用した電子商取引システムについて述べる．2.6 節で検討内容をまとめ，今後の課題を示す．

2.2 モーフィングによる視点変更における画像対応付け方式

2.2.1 基本方針

原画像間の対応付け方法を開発するにあたり，(1) 各原画像のエッジどうしが対応しないと合成画像がぼやける，(2) 各原画像のエッジ以外の画素については対応がずれてもぼやけが少ない，という 2 つの仮定の元に対応付けを行う．ぼやけの画像品質への影響は，既存の画質評価ツール [64] やディスプレイの評価においてそのエッジ部分に着目した研究事例 [65] があり，その影響が大きいと考えられる．また Feature-based Image Metamorphosis[36] でも画像の特徴であるエッジに着目した対応付けを行っている．上記仮定に基づき，以下の設計方針を立てる．

1. 各入力画像をエッジに沿って正確に対応付ける．
2. 前記対応を極力少ない作業量で実現する．
3. エッジ以外の画素は自動で対応付ける．

図 2.1: 入力画像

2.2.2　画像対応付け方式

以下に示す作業手順に基づく画像対応付け方式を実現する．図 2.1 に示す 4 枚の入力画像から合成画像を生成する場合を具体例に詳細を述べる．

1. 画像の複数領域への分割

   オペレータは，それぞれの入力画像をエッジに沿って複数の領域に分割する．エッジに沿った画像分割に関する研究は古くから行なわれている [66][67]．また市販のレタッチソフトの機能としても領域分割の機能を含むツールも市販されており，これらを用いて領域を切出す．図 2.1 の画像の場合は，図 2.2 に示すように，髪の毛，左目，右目， 鼻，口，顔，首，服の 8 つの領域に分割したものである．

図 2.2: 領域の分割例

2. 境界線と交差する線分の指定

各入力画像の対応する領域の境界線間の対応付けを指定する．これはオペレータが，それぞれの入力画像に対して，対応する領域の境界線位置を，境界線と交わる線分の交点として指定することで実現する．図 2.3 は，画像の対応付けを行う一連の過程を示す図である．図 2.3 (a) では，それぞれの入力画像に対して対応する線分として，眉毛上部に黒い 2 本の線分を指定している．線分上の白い点が，線分と境界線との交点であり，これらが互いに対応している．

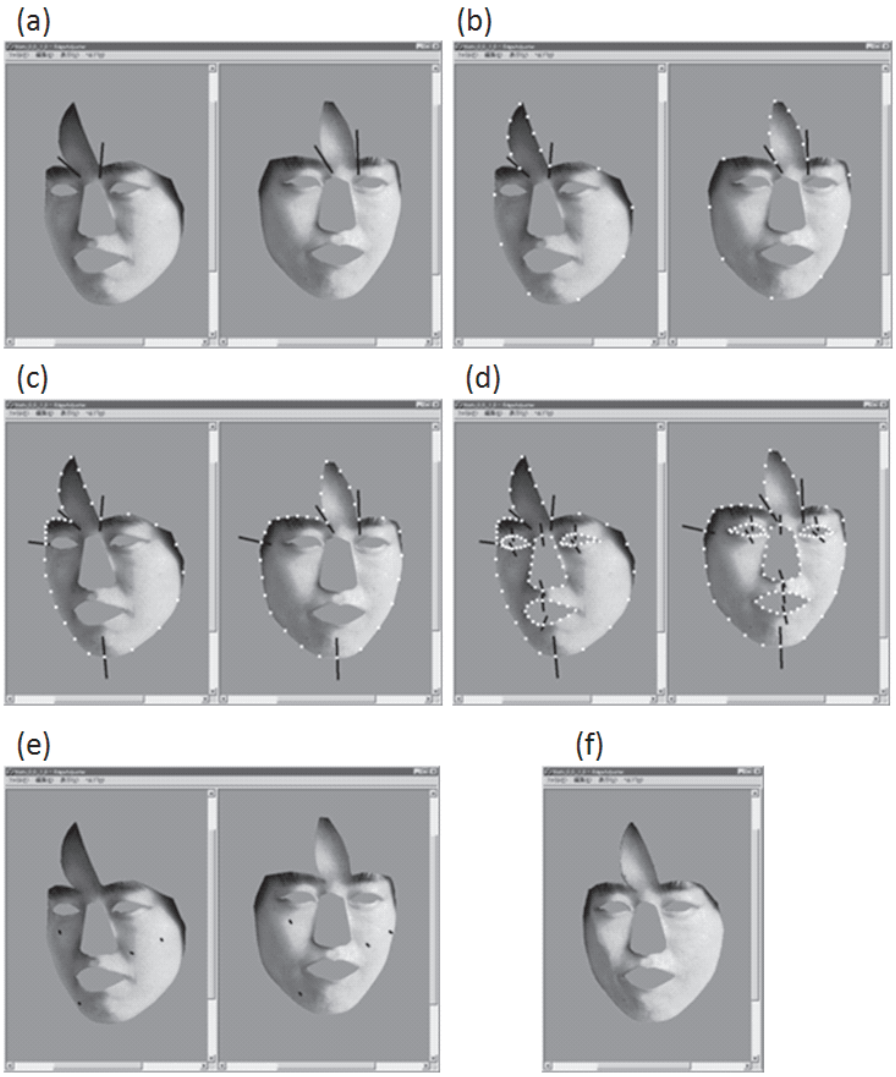


図 2.3: 画像対応付けの一連の過程

3. 境界線上の他の画素の対応付け

前ステップで指定した交点を元に境界線上の他の画素の対応を求める方式を，図 2.4 に示す 2 枚の顔画像の輪郭の対応付けを行う場合を例に説明する．図 2.4 で，それぞれの入力画像の輪郭上に 3 つの交点 (交点 A，B，C ならびに，それに対応する交点 A ’，B ’，C ’) がオペレータにより指定されているものとする．次に，境界線上で隣接する 2 つの交点を選択し (図 2.4 の例では，交点 A，交点 B が選択されたものとする)，交点 A と交点 B を両端とし境界線をたどる線分 (線分 AB) を求める．他の画像においても同様に，前記 2 つの交点と対応する交点 (交点 A’，交点 B’) に対して，交点 A’ と交点 B’ を両端する線分 (線分 A’B’) を求める．線分 AB と線分 A’B’ を同じ割合 α ($0 \leq \alpha \leq 1$) で内分する内分点 x，内分点 x ’ を求め，内分点 x と内分点 x’ がそれぞれ対応するものとみなす．内分する割合 α を連続的に変化させ線分 AB と線分 A’B’ 上の全ての画素に対して対応を求める．前記の処理を，入力画像上の隣接する交点で決定される全ての線分 (図 2.4 の例では，線分 BC ならびに線分 CA) に対して繰り返す．本処理ステップの対応付けはツールにより自動で行う．

上記処理による境界線上の対応付け状況を，オペレータは図 2.3 (b) の境界線上の白点で表わされるサンプルとなる対応点で確認できる．白点は，入力画像の境界線上の対応点を一定の内分割合ごとに例示したものである．オペレータは各画像において対応する白点の位置を確認することで境界線が正しく対応付けられているか確認できる．図 2.3 (b) の例では，額の部分は正確に対応付けられている．一方，顎の部分は対応付けが正しくない．この場合，オペレータは，線分を追加することでより正確な対応付を行うことができる．図 2.3 (c) は線分を追加して対応付けを行ったものである．これにより対応付けが適切でなかった顎の部分も，より正確に対応付けられたことがわかる．顔の他の境界 (目，鼻，口との境界) も同様に対応付けを行った結果を図 2.3 (d) に示す．

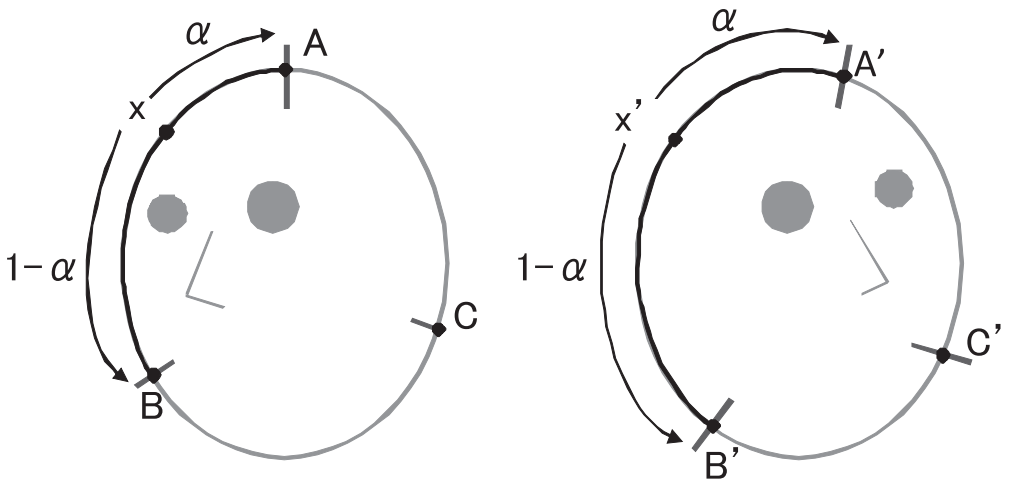


図 2.4: 境界線上の画素の対応付け

4. 境界線以外のエッジの対応付け

   画像の中には領域分割を行うのが困難なエッジが含まれる場合がある (顔の例ではほくろや皺など)．このようなエッジについてはオペレータが点，線分を用いて対応を指定する．なお，この部分についてもエッジをトレースし，エッジに沿った線分で指定できることが望ましいが今回は省略する．図 2.3 (e) では，入力画像のほくろの位置をそれぞれ点を使って指示することで，それらが対応することを指定する．

5. エッジ以外の対応付け

   エッジ以外の対応を持たない画素の対応付けは，対応付けを行なったエッジから推定する．具体的には，対応を持たない画素の対応点を，周囲の対応を有する画素の対応点から決定する処理を再帰的に繰り返すことで，入力画像の全画素に対する対応点を求める．対応を持たない画素 $pixel$ の対応点の算出は，その画素を中心としたある一定の半径内に存在する，対応点を有する各画素 $pixel_m$ の対応先への移動ベクトルに，画素 $pixel$ と画素 $pixel_m$ との距離の逆数を乗じた加重平均として算出する．これらの画素の対応付けアルゴリズムを図 2.5 に示す．図中のゴシック体で示

した部分は予約語もしくは関数呼び出しを示す．また イタリック体で示した部分は変数である．`PositionOf(`*pixel*`)` は *pixel* の座標を求める関数である．`MappedPixelOf(`*pixel*`)` は，他の入力画像 *image'* における *pixel* に対応する画素を求める関数である．R は，まだ対応点を決定していない画素の対応点を類推する周囲の画素の範囲 (半径) を示す定数である．

最初に第 1 行目である画像 *image* 上で対応点を持たない画素 (*pixel*) を順次選択し，対応を持たない画素が無くなるまで 第 2 行目から 13 行目まで処理を繰り返す.

第 2 行目から第 4 行目まででテンポラリ変数 *totalDistortion*, *totalInvDistance*, *distance* を宣言している．

第 5 行目では，*image* 上で対応点を持つ画素 ($pixel_m$) を順次選択する．第 6 行目では，$pixel_m$ と *pixel* の距離 (*distance*) を算出し，第 7 行目では，距離 *distance* が R よりも小さい場合 ($pixel_m$ が *pixel* の近傍の場合) には 第 8 行目から 10 行目までの処理を実行し，画素 ($pixel_m$) を画素 (*pixel*) の対応点算出に使用する．第 8 行目では，画素 ($pixel_m$) の *distotion* (画素 $pixel_m$ の位置から画素 $pixel_m$ の対応点の位置へ向かうベクトル) を *totalDistortion* に加える．第 9 行目では，画素 $pixel_m$ と画素 *pixel* の距離の逆数 (画素 $pixel_m$ の影響力をコントロールする) を *totalInvDistance* に加える．第 11 行目では，画素 *pixel* の周辺に対応付けを持つ画素 $pixel_m$ が存在したかを判定し，存在しない場合は，画素 *pixel* の対応付けを後回しにする (本アルゴリズムを繰り返すことで対応付けを持つ画素の領域が次第に拡大し，最後に全体の対応付けができる) ．第 12 行目では，画素 *pixel* の対応点を周囲の対応点を持つ画素 $pixel_m$ の対応点の加重平均により求める．以上の処理による対応付けにより合成した画像を図 2.3 (f) に示す.

```
1.  while(pixel in image which doesn't have a disparity map exists) {
2.      var totalDistortion = (0,0);
3.      var totalInvDistance = 0;
4.      var distance;
5.      for(each pixel_m in image which has a disparity map) {
6.          distance = || PositionOf(pixel_m) - PositionOf(pixel) ||;
7.          if (distance ≥ R) continue;
8.          totalDistortion +=
              PositionOf(MappedPixelOf(pixel_m)) - PositionOf(pixel_m);
9.          totalInvDistance += 1 / distance;
10.     }
11.     if (totalInvDistance == 0) continue;
12.     PositionOf(MappedPixelOf(pixel))) =
          PositionOf(pixel) + totalDistortion / totalInvDistance;
13. }
```

図 2.5: エッジ以外の点の対応点を求めるアルゴリズム

2.3 モーフィングによる視点変更における高速レンダリング方式

2.3.1 基本方針

モーフィングによる画像合成は，画像変形を行う Warping の処理と Warping 後の画像のミキシングを行う Cross-Dissolve の処理の 2 ステップから構成される．Warping では 対応情報とユーザの入力した合成比率 (幾何ベース CG ではカメラパラメータ) を元に各画像の各画素の移動先を求める．さらに隣接する画素間に裂け目が生じる場合はこの補修を行う．合成比率は，それぞれの入力画像に対して合成画像がどの程度類似するかを規定するパラメータである．次に変形した各画像を合成比率に応じてミキシングして合成画像を生成する．これらの処理では画素ごとの移動・ミキシングを伴うために計算量が多く，特に画像が大きくなると通常の汎用の CPU でリアルタイムに画像を合成することは難しい．そこで既存の 3D アクセラレータに注目する．既存の 3D アクセ

ラレータは，画素単位ではないが，多角形を単位としてテキスチャを変形する機能を有する．また半透明な物体を表現する手法として前面にある画像と背面にある画像を混合して表示する α-blending の機能を有する．これらの機能を活用し， Warping による画像変形をテキスチャのポリゴンを用いた変形として処理し，画像のミキシングを α-blending を用いて処理するようにデータ変換を行うことで，モーフィングのレンダリング処理を既存の 3D アクセラレータを用いて実行可能にする．

2.3.2 モーフィングデータ

今回用いたモーフィング手法は構造化モーフィング [68] と呼ばれる手法に基づいている．構造化モーフィングは画像を複数のレイヤに分割して，レイヤごとにモーフィングを行うことで，距離の違う隣接する物体をまとめてモーフィングすることにより生じる画像の歪みを無くし，またオクルージョン (視点変更に伴いある物体が他の物体の背面に隠れること) が生じる場合の表現を行いやすくする．例えば今回の画像の例では，オクルージョンが発生する領域を互いに独立させるために，次の三つのレイヤに分割しモーフィングを行っている．なお本手法は通常のモーフィング (構造化モーフィングにおけるシングルレイヤの場合に相当) も同様に適用可能である．

1. 髪の毛
2. 顔
3. 首，体

2.3.3 3D アクセラレータによる Warping の実現

モーフィング処理を 3D アクセラレータで行う場合，各画素の移動を 3D アクセラレータでどのように行うかが重要な問題である．一般に 3D アクセラレータでは画像の変形に関して，三角形，四角形を変形するスプライトと呼ばれる機能を有するが，画素ごとの変形・移動の機能は持たない．そこで画素を

幾つかまとめ三角形 (三角パッチ) を生成し，視点変更に伴い三角パッチをどう変形するかを画素の対応データから算出する．また三角パッチは画像のエッジ部分が細かくなるようにする．これは前節で述べたようにエッジの対応がずれることによる合成画像のぼやけを少なくするためである．具体的には以下に述べる処理手順に従って三角パッチへの分割ならびに変形を行う．

1. 画像の三角パッチへの分割

   レイヤごとに各画像を同一サイズの三角パッチに分割する．分割の作業はオペレータによる三角パッチのサイズ指定を受けた上で自動で行う．図 2.6 (a) は，画像のある 1 つのレイヤである．図 2.6 (b) は，前レイヤを同一サイズの三角パッチに分割をしたものである．

2. 三角パッチの細分化

   三角パッチ内に画像のエッジ部分が含まれていたり，画像の変形量が大きく変異する部分については，三角パッチの細分化を行う．具体的には三角パッチを 4 分割し，細かな三角パッチを作る．三角パッチの細分化を行う際には隣接する三角パッチも適宜分割し，三角パッチの辺上に隣接する三角パッチの頂点がこないようにする．これは三角パッチの辺上に他の三角パッチの頂点があると変形に伴って隙間が生じる可能性があるというクラック問題 [69][70] に対処するためである．

   図 2.7 を用いてクラックが発生する原因と防止方法ついて述べる．まず図 2.7 (a) に示すように中央の三角パッチを細分化する場合について考える．この状態で三角パッチの変形を行うと図 2.7 (b) に示すようにクラックが発生する可能性がある．これは隣接する三角パッチで同一場所の頂点同士は同じ位置に移動するが，隣接する三角パッチの辺と頂点は必ずしも同じ位置に移動しないことによる．例えば，図 2.7 (a) の点 A の，変形後の位置が図 2.7 (b) の点 A’ の場合，点 A を頂点としていた三角パッチの頂点は点 A’ に移動するが，点 A を辺上としていた三角パッチの辺の位置は必ずしも A’ には移動しない (図 2.7 (b) の点 A” に移動)．

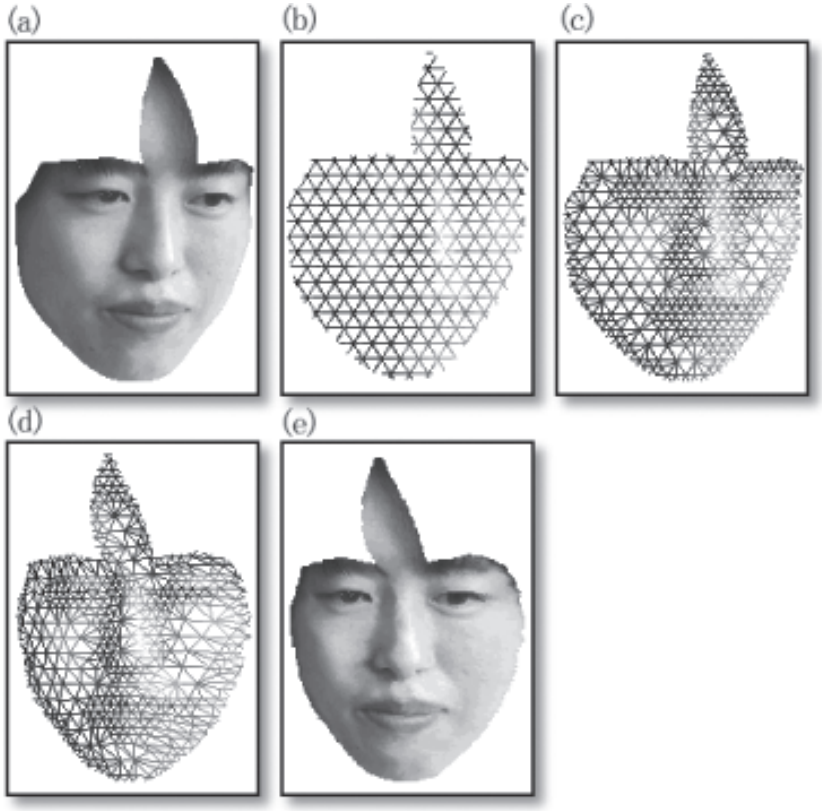


図 2.6: 三角パッチへの分割手順

これは辺上の各点の位置は，辺の両端をなす頂点の位置を元に近似されることによる．そこで隣接する三角パッチを図 2.7 (c) のように分割する．この分割により隣接する三角パッチの各頂点は互いに同一の位置となり，変形を行っても図 2.7 (d) のようにクラックを生じない．

以上の考えを踏まえた三角パッチ細分化のプロセスを，図 2.8 に示す 8 つの三角パッチを細分化する例を用いて説明する．図 2.8 の三角パッチ 2, 6 にエッジもしくは画像の変形量が大きい部分が存在し，これらを細分化する場合を考える．まず 図 2.8 (b) に示すように，三角パッチ 2, 6 を 4 分割に分割する．この場合，細分化された三角パッチの頂点が他の三角パッチの辺上に存在するため，変形に伴いクラックが生じる可能性がある．

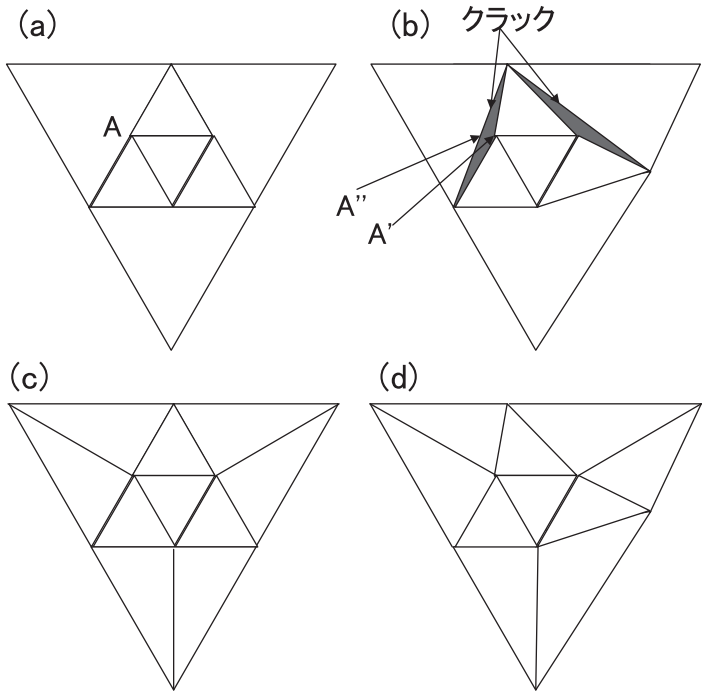


図 2.7: クラック発生の原因と防止方法

そこで隣接する三角パッチを細分化して，隣接する三角パッチの頂点が同一画素になるようにする．具体的には，まず 2 つ以上の隣接する三角パッチが 細分化される三角パッチに関しては同様に 4 分割を行う (図 2.8 (c) における 三角パッチ 1 を参照)．なおこの細分化の処理は，再帰的に行い， 2 つ以上の隣接する三角パッチ全てが細分化されるまで行う．次に隣接するパッチのいずれか 1 つが細分化される三角バッチを 2 分割する (図 2.8 (d) における三角パッチ 3,5,7 を参照)．以上により隣接する三角パッチの頂点が同一画素上に存在することになりクラック問題を解消できる．三角パッチの細分化のプロセスは，細分化を行う変形量の閾値をオペレータが設定することで，編集ツールが自動で分割を行う．以上の処理により，図 2.6 (b) の三角パッチを細分化した画像を図 2.6 (c) に示す.

3. 三角パッチの変形

視点変更に伴う各三角パッチの変形は，三角パッチの各頂点の画素が対

応する画素を頂点とする三角形へと変形するようにする．画像の全ての画素を三角パッチで覆い尽くすためには，画像の周辺部で頂点に画素を持たない三角パッチが生じる．このような対応点を持たない頂点については変形後の位置を推測する必要がある．今回の実装では，単純に頂点に最も近い画素の移動量と同じだけ頂点が移動するものと仮定した．図 2.6 (c) を変形した例を 図 2.6 (d) に示す．さらに 図 2.6 (d) にテキスチャを貼り付けた画像を 図 2.6 (e) に示す.

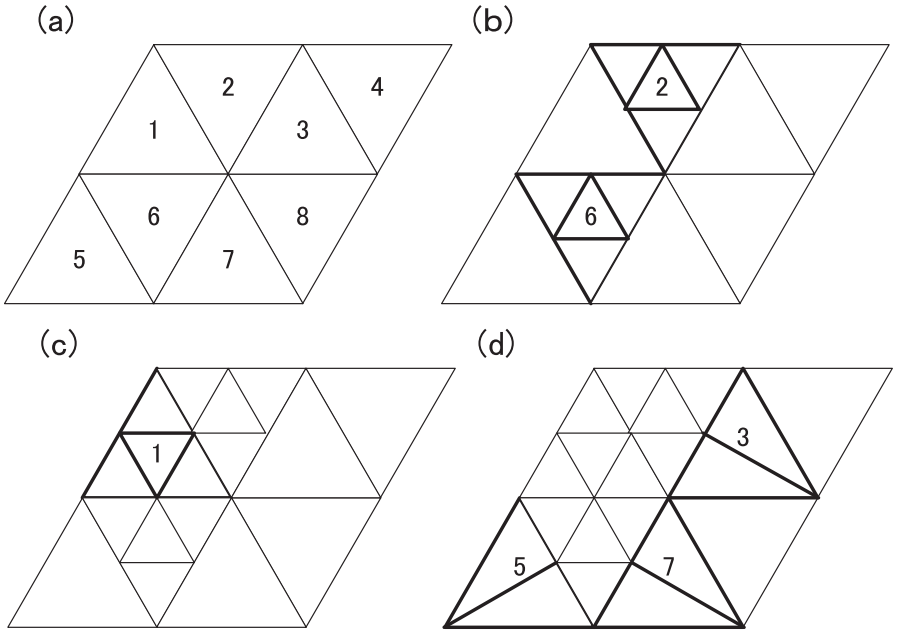


図 2.8: パッチ分割の例

2.3.4 3D アクセラレータを用いた画像のミキシング

モーフィングのレンダリングプロセスでは，入力画像を前節で述べた方法で変形した上で，変形後のそれぞれの画像を合成比率に応じてミキシング (Cross-Dissolve) することで合成画像を生成する．3D アクセラレータでは，半透明の物体を表現するために α-blending の機能を有しており，この機能を用いて画像のミキシングを実現する．具体的には，図 2.9 に示すように正射影カメラの前に各レイヤごとに定義したメインプレートと複数のサブプレートを，各レイ

ヤの前後関係に従い配置することでミキシングを実現する．以下，その詳細を示す．

1. 正射影カメラ

   メインプレートとサブプレートを射影する．正射影カメラを用いることで遠近感の表現は排除され，手前にあるプレートも奥にあるプレートも同一サイズにレンダリングされる．

2. メインプレート

   メインプレートは，レイヤごとに一つ存在するプレートであり，同一レイヤを構成するサブプレートの最後部に配置する．図 2.9 で示した太線で示されたプレートがメインプレートである．メインプレートには，変形を行った入力画像のいずれかを貼り付ける．メインプレートには画像を不透明で貼り付ける．なおプレート自体は透明であり，画像が貼り付けられない部分は，後方のプレートをそのままカメラに映し出す．このように配置することで 3D アクセラレータの z-buffering の機能により，前方のレイヤに画像が存在する場合は後方のレイヤの画像は描写されず，レイヤの前後関係を保つことができる．

3. サブプレート

   サブプレートには，レイヤごとにメインプレートに貼り付けた画像以外の変形後の入力画像を貼り付ける．各レイヤにおいて，サブプレートは，メインプレートの前方に (正射影カメラ寄りに) 配置する．なお前方に別レイヤがある場合には，前方のレイヤのメインプレートよりは後方に配置する．サブプレートには，変形画像を半透明で貼り付ける．透明度は入力画像の混合割合で決定する．図 2.9 の細枠で囲まれたプレートがサブプレートである．以上により同一レイヤの画像のミキシングが α-blending の機能を用いて実現できる．

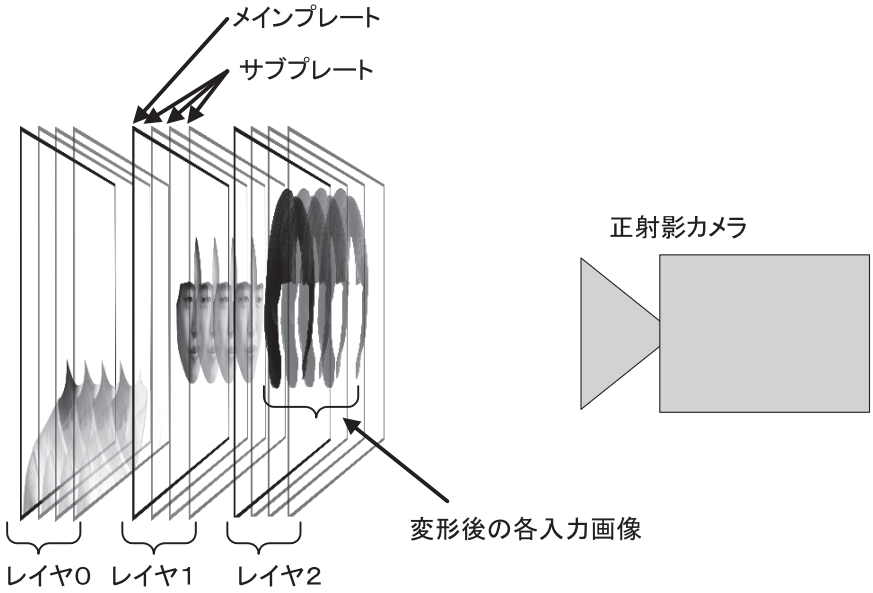


図 2.9: 3D アクセラレータを用いてミキシングを実現するための仮想的な三次元空間

2.3.5 3D アクセラレータへの要求機能

本レンダリング方式で必要となる 3D アクセラレータ機能を以下にまとめる．

- テキスチャマッピング： ポリゴンへの画像の貼り付け
- 正射影カメラ： 正射影でのレンダリング
- α-blending： 半透明の表現
- z-buffering： ポリゴンを重ね合わせる際の前後関係の制御

現在主流の 3D API はこれらの機能を有している．またその多くの機能は 3D アクセラレータの基本的な機能として実装されており，ハードウェア的に処理が可能である．

2.4 評価

本節では，まず 2.2 節で述べた画像対応付け方式の評価について述べ，次に 2.3 節で述べた高速レンダリング方式の評価について述べる．

図 2.10 の合成画像は，2.2 節での対応付け方式により，図 2.1 に示す 4 つの入力画像から，非リアルタイムで (ピクセル単位の変形で) 合成した画像である．なお合成比率は全て同じで 25 % ずつである．図 2.11 は正面から撮影した実際の画像である．また図 2.12 は従来の線分による対応付けを行い [7][11] 合成した画像である．なお図 2.12 は，既存ツールの制約により，4 枚の入力画像ではなく左上と右下の 2 枚の画像から合成を行ったものである．

図 2.10: 提案手法のマッチングにより合成した画像 (非リアルタイム生成)

本手法で対応付けを行った図 2.10 の画像は，4 つの画像を合成しているにも関わらずぼやけが少ないことがわかる．この画像のぼやけに関して，図 2.13 (a) (b) のパワースペクトルにより検証する．図 2.13 (a) は，本画像で対応付けにより生成した合成画像 (図 2.10) の目の部分をフーリエ変換したパワースペクトル画像である．一方，図 2.13 (b) は，従来の対応付け方式により生成した合成画像 (図 2.12) の目の部分を同様に処理したパワースペクトル画像である．画像の周辺部が高周波成分を表す．これらを比較すると図 2.13 (a) のパ

ワースペクトルの方が，図 2.13 (b) のパワースペクトルよりも高周波成分が多く含まれることがわかる．画像がぼやけると画像の高周波成分が失われることから，本手法で対応付けを行っている方がぼやけが少ないことがわかる．

図 2.11: 正面から撮影した実際の画像

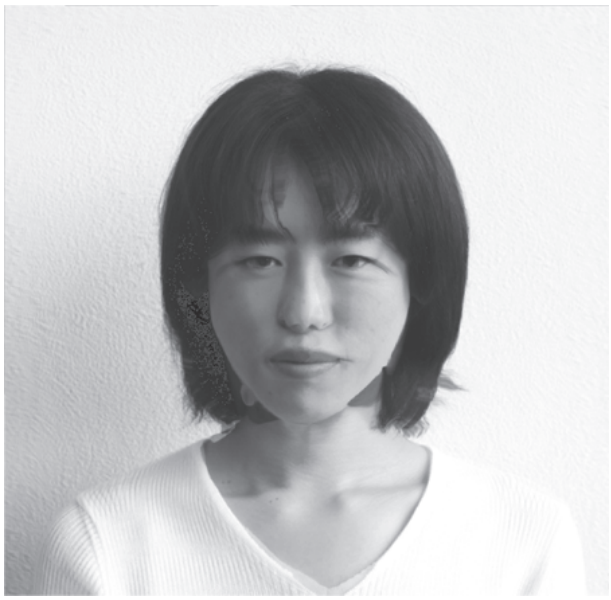


図 2.12: 従来のマッチングにより合成した画像 (非リアルタイム生成)

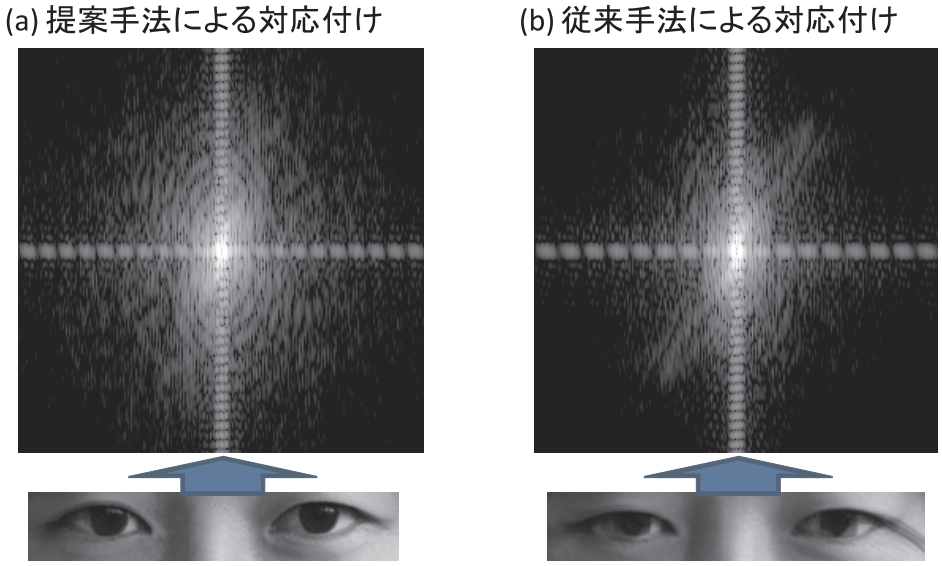


図 2.13: 目の部分のパワースペクトラム比較

図 2.14 (b) は，図 2.14 (a) に示す 4 つの入力画像から合成した別の例である．図 2.14 (c) が同じ位置から見た実際の画像である．本例においては，合成画像と実際の画像の間に違いが見られる．具体的には，鹿の人形の足がぼやけていることと顔が丸みを帯びていることである．これらの差異はオクルージョンに起因するものである．現在のシステムでは，ある入力画像の領域が別の画像において他の領域の背面に隠されてしまう場合，正確な対応付けが行なえない．したがって視点の変化により，オクルージョンが多く発生する入力画像については，比較的視点変更の変化量の少ない入力画像を多く用意する必要がある．この問題にする解決方針は，2.6 節の将来の課題で述べる．

次に 2.3 節で述べた高速レンダリングに関する性能評価について述べる．測定条件は表 2.1 に示すように，MPU (Micro Processing Unit): PentiumII[1] 266 MHz，メモリ: 96　MByte，OS (Operating System): WindowsNT[2] 4.0，3D API(Application Program Interface): OpenGL，3D アクセラレータ: PERMEDIA-II[3]，画像サイズ: 410 × 400 である．

[1]Pentium は，米 Intel 社の登録商標
[2]WindowNT は，米 Microsoft 社の登録商標.
[3]PERMEDIA は 米 3D Labs 社の登録商標.

表 2.1: 高速レンダリングに関する測定環境

MPU	PentiumII 266 MHz
メモリ	96MByte
OS	WindowsNT4.0
3D API	OpenGL
3D アクセラレータ	PERMEDIA-II
画像サイズ	410 × 400

図 2.15 は，本研究で提案した 3D アクセラレータを用いてレンダリングした合成画像である．また表 2.2 にレンダリングに要した時間を示す．これらの画像各 1 フレームの画像を合成するのに要した時間は 82 ms であり，秒 12 コマのインタラクティブなアニメーションが行える．PERMEDIA-II を除く上記ハードウェアと同条件で 3D Accelerator を使用しないで通常の画素ごとの処理で合成画像を生成した場合の画像合成時間は 920 ms であり，約 11 倍の高速化が実現できたことがわかる．

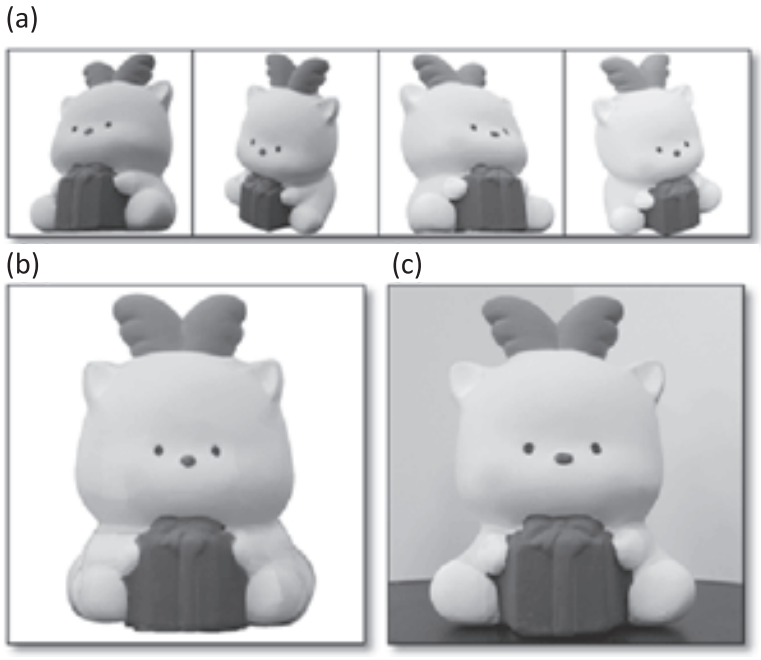


図 2.14: 別の事例での合成結果

図 2.15: リアルタイムで合成された合成画像

表 2.2: レンダリング時間

	画素単位での レンダリング（従来技術）	3D アクセラレータによる レンダリング
1 フレームあたりの レンダリング時間	920 msec	82 msec

2.5　モーフィング技術の電子商取引システムへの応用

本章では，これまで述べたモーフィング技術をどのように電子商取引システムに応用するかについて述べる．具体的には，ウェブを用いた仮想的な住宅プレゼンテーションシステムへの応用例について述べる．

図 2.16 は住宅プレゼンテーションシステムの画面イメージである．画像上でのマウス操作により，インタラクティブな実写ウォークスルーが可能である．その実装の概要を，図 2.17 に示す．図中の網掛けを行った部分が，視点変更に必要となるモーフィングデータとその処理プログラム (プラグイン) である．

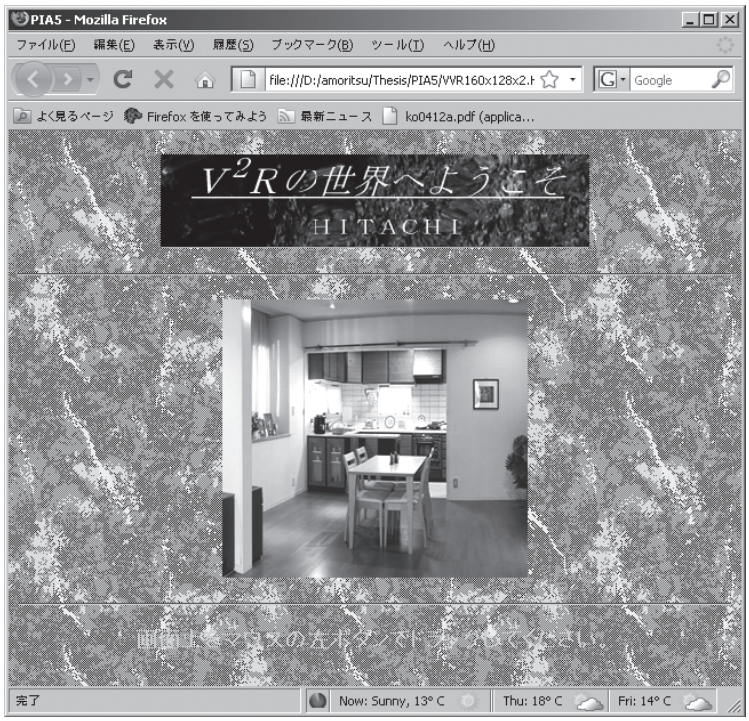


図 2.16: ウェブを用いた住宅プレゼンテーションシステムの画面イメージ

まず HTTP (Hyper Text Transfer Protocol) サーバ上にウォークスルー用の HTML (Hyper Text Modeling Language) ファイルと，入力画像，三角パッチ情報，三角パッチの変形情報からなるモーフィングデータを配置する．また HTML ファイルには，モーフィングデータへのリンクを記述しておく．

一方，クライアントのウェブブラウザにはモーフィング用のプラグインを組み込んでおく．クライアントのウェブブラウザが前記 HTML ファイルをダウンロードすると，リンクを辿りモーフィングデータのダウンロードが行われる．モーフィングデータに，独自に設定した MIME (Multipurpose Internet Mail Extension) タイプ (図 2.17 の例では application/morphing) を設定しておくことで，前記 MIME タイプにあらかじめ関連付けておいたモーフィング用のプラグインが呼び出される．モーフィング用プラグインは，モーフィングデータをウェブブラウザから受け取り，画面上のマウス操作を受けて，前記データに基づく任意視点の画像をインタラクティブに合成することで実写ウォークスルーを実現する．

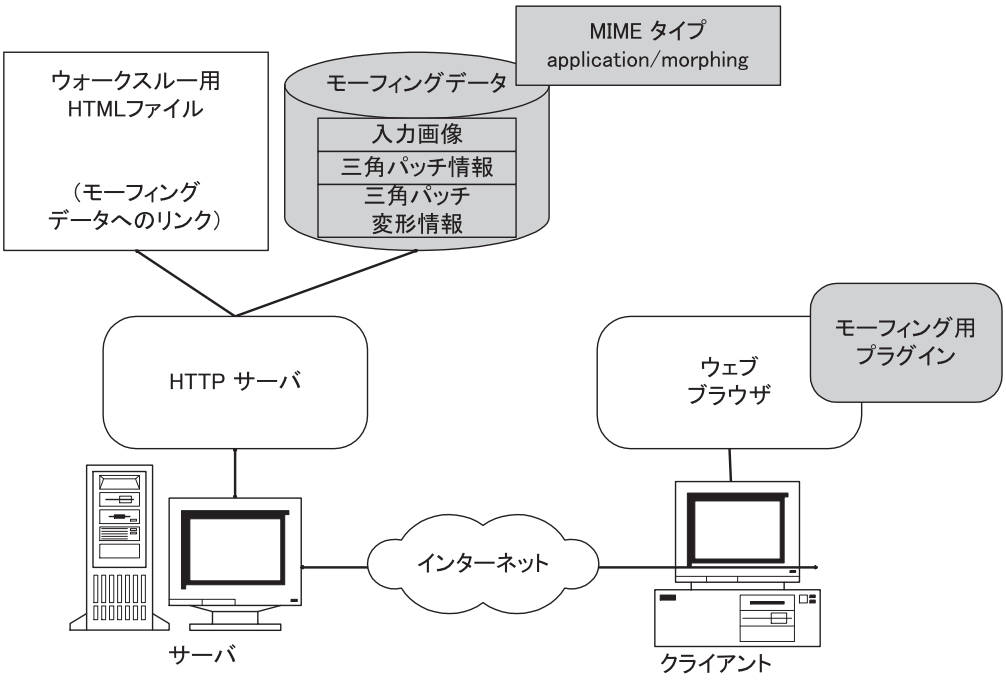


図 2.17: 住宅プレゼンテーションシステムの実装概要

2.6 結言

本章ではモーフィング技術に関して，合成画像のぼやけを軽減する画像対応付け方式とインタラクティブな視点変更を実現する高速レンダリング方式を提案した．画像のエッジ部分に沿った対応付けを可能にする画像対応付け方式を実現することで，合成画像のぼやけを軽減した．またモーフィングのレンダリングプロセスであるワーピング処理を三角パッチの変形に置き換え，さらに画像のミキシングを α-blending に置き換えるデータ変換を実現することで，3D アクセラレータを用いたモーフィングを可能にし，約 11 倍の高速化を実現した．また本技術を Web ブラウザのプラグインとして実装し，電子商取引システムに応用した例を示した．これにより，クライアント側に独自のプラグインのインストールが必要であるが，電子商取引システムの任意の Web ページで，モーフィングによるインタラクティブな視点変更を用いたコンテンツの提供が可能である．

本技術を活用した美術品鑑賞システムを実用化し，1998 年徳川美術館の電子美術館特別展示にて能面「小面 (こおもて)」を題材とした鑑賞システムの展示を行った．また前記コンテンツは SIGGRAPH ’98 のアニメーションフェスティバルに入選した [56].

現在のシステムでは，画像の対応付け方式に関して，以下の 4 つの課題が残る．

1. オクルージョンにより対応付けができない点に関する取扱い

   現状のシステムでは，入力画像のある領域が別の画像における他の領域の背面に隠れてしまう場合，正確な対応付けが行なえない．本課題の解決策としては，陰に隠れていない部分の対応情報から隠れた部分の対応を補間する，あるいはオペレータが経験的に判断した対応位置を指示できるようにする方法が考えられる．

2. 領域分割が困難な対象物の取扱い

   本手法では，領域分割が行なえれば領域の境界について正確な対応付けが行なえるが，領域分割が困難な場合は従来のモーフィングと同様の対応付け (直線線分による対応付け) しか行なえない．本課題の解決策としては，エッジをトレースした線分に対して対応付けを行えるようにする方法や，パターンマッチングの技術と組み合せる方法が考えられる．

3. 幾何ベース CG と融合した電子商取引システムの実現

   既存の幾何ベースの CG 技術とモーフィング技術を含む実写ベース CG 技術を融合した電子商取引システムのインタフェースの実現も今後の課題である．幾何ベースの CG 技術，実写ベースの CG 技術ではそれぞれ得意とする表現領域が異なっている．これらの技術をどう組みわせて電子商取引システムのインタフェースを構築していくかを検討する必要がある．

4. 電子商取引システムへの応用することの効果の検証

本手法を電子商取引システムに応用することで，具体的に顧客に対してどのような訴求効果があるかに関する検証は現状不十分である．例えば，本技術を商品プレゼンテーションに適用することで売り上げ向上にどの程度寄与するのかといった評価や，本技術を用いた合成画像によるユーザインタフェースは，実際の実写画像を用いたユーザインタフェースを用いた場合と比較して，どの程度顧客に違和感を与えるのかといった評価が考えられる．

第3章

類似サービス活用バックアップのための変換ウェブサービス生成方式とそのオーバヘッド検証

3.1 緒言

本章では，電子商取引におけるバックアップの柔軟性に関し，類似サービスを活用する変換ウェブサービスの生成方式を示し，そのオーバヘッドを明らかにする．

変換ウェブサービスを生成する上で，変換ウェブサービスを動的に構成するか，あるいは静的に構成するかという検討軸と，変換ウェブサービスを自動で構成するか，あるいは半自動で構成するか，もしくはスクラッチで開発するかという大きく 2 つ検討軸がある．まず最初の検討軸に関して，バックアップという利用形態では，事前にプライマリのプロバイダとバックアップのプロバイダ間で何らかの合意が存在すると想定できる点に着目する．このような合意のもとでは，バックアップ先となるシステムは静的に定まるものと考えられる．したがって，動的なサービス構成といった柔軟性の高い仕組みよりも，むしろ実行パフォーマンスに優れる方式が好ましいため，静的にサービスを構成する方法で変換ウェブサービスを生成する．

また第 2 の検討軸に関して，変換ウェブサービスが変換対象とするプライマ

リサービスとバックアップサービスのモデル化が可能かという点に着目する．変換ウェブサービス生成の省力化に向けては，自動でサービスを構成することが望ましい．しかし自動で構成するためには，変換対象のプライマリサービスとバックアップサービスが提供するサービス内容をオントロジ [42][43] などの記述を用いてモデル化し，その差異を機械的に識別する必要がある．このようなモデル化を，例えば証券取引システムなど，システム全体に対して行うことは難しい．またその一方，変換ウェブサービスをスクラッチで開発することは開発コストの面から現実的でない．そこで人手によるフロー定義に基づく半自動の方法で変換ウェブサービスを生成する．

上記 2 つの方針に基づき，変換ウェブサービスを生成する開発環境を構築する．具体的には，顧客システムとバックアップシステム間のやり取りを変換するクライアント変換ウェブサービスと，プライマリサービスとバックアップサービスの間で，バックアップに備えた情報のやり取りと障害後のリストア時の情報のやり取りを仲介するサーバ変換ウェブサービスを構築するための開発環境を実現する．さらにこの開発環境を用いて，証券取引を具体例として，類似する証券会社に対するバックアップを可能にする変換ウェブサービスの生成を行い，その性能測定を行う．

以降，3.2 節で，類似サービスを活用したバックアップ方式のねらい，実現課題ならびに解決策を示す．3.3 節で，実装を行ったサンプルアプリケーションの概要を示し，具体的にどのようなサービス仕様あるいはプロトコルの差異を吸収することが可能であるかを例示する．3.4 節で，前記差異を吸収する変換ウェブサービスをどのように生成するかについて述べる．3.5 節で，生成を行った変換ウェブサービスの性能測定を行い，3.6 節で，検討内容をまとめる．

3.2　類似サービス活用バックアップ方式

本研究のねらいは，類似サービス間のフェイルオーバを実現することで，図 3.1 に示すような様々なバックアップ形態を実現することである．例えば，最もシンプルな形態として，図 3.1 のケース 1 で示すように，他のプロバイダが提供する仕様の異なる類似サービスをバックアップ先として活用するといっ

た形態が考えられる．また，図 3.1 のケース 2 は，類似するサービスを提供するサービスプロバイダ間でバックアップセンタを共有し，それぞれのサービスプロバイダが各々のサービスとバックアップサービスの差異を吸収する仕組みを構築するといった利用方法である．また，図 3.1 のケース 3 のように，システムのリプレイスが生じた際に，仕様に違いが生じてしまった旧システムをバックアップとして利用するケースも想定できる．さらに，図 3.1 のケース 4 は，企業統合が行われた場合に，それぞれ独自で開発した差異があるシステムを互いにバックアップとして活用するといった形態である．

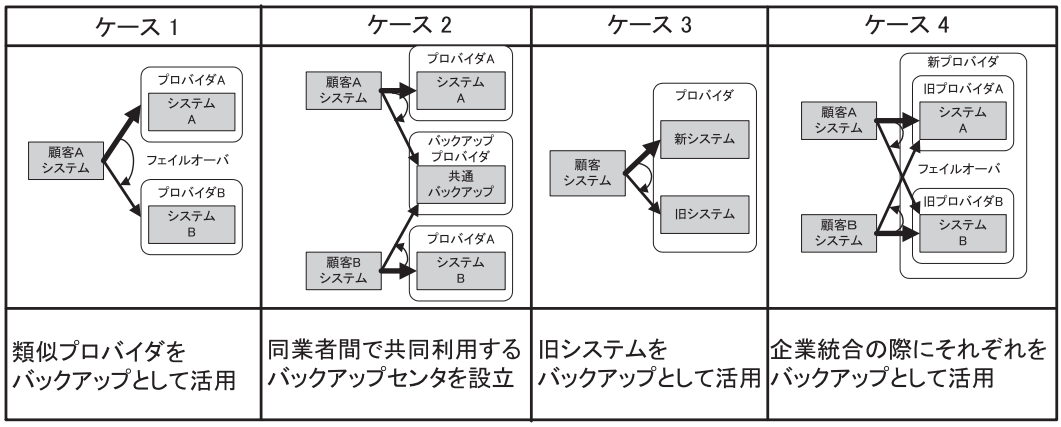


図 3.1: 類似サービス間でのフェィルオーバの活用形態

図 3.2 (a) は，図 3.1 のケース 1 の例であり，類似するサービスを提供するプロバイダ A とプロバイダ B が存在し，プロバイダ A が プロバイダ B のサービスを障害時のバックアップ先として活用する場合の例である．通常それらのサービス仕様ならびにプロトコルが完全に一致することは希であるため，そのままではプロバイダ A はプロバイダ B のシステムをバックアップとして活用することはできない．具体的には図 3.2 (a) において プロバイダ A の顧客は，障害時にプロバイダ B に接続しサービスの提供を受けることができないし，またプロバイダ A もプロバイダ B との間で障害に備えた情報の共有や障害後のリカバリといった情報共有が行えない．

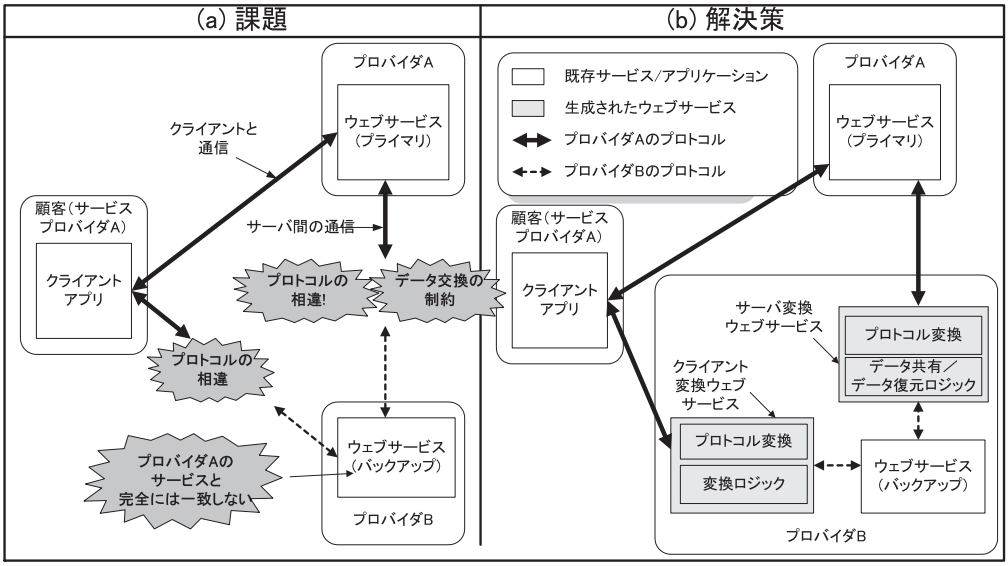


図 3.2: 解決すべき課題と解決方針

そこでサービス仕様ならびにプロトコルの違いを吸収することを目的として，図 3.2 (b) に示すように，クライアント変換ウェブサービスとサーバ変換ウェブサービスの 2 つのウェブサービスを生成する．クライアント変換ウェブサービスは，クライアントからプライマリ (プロバイダ A) の仕様で送信されたメッセージをバックアップ側 (プロバイダ B) に適合するように変換しバックアップサービスに転送するものである．またバックアップ側からの返答に対しても同様の逆変換を行う．サーバ変換ウェブサービスは，プライマリ側からバックアップ側へ送られる障害に備えたデータ共有 [71] (例えば，ホテル予約であれば，予約可能な残り部屋数を共有しておくといった利用など) や，逆にバックアップ側からプライマリ側へ障害回復後に送信されるデータの復元情報に関し，それら情報交換のサービス仕様ならびにプロトコルの違いを吸収する役割を担う．

3.3 サンプルアプリケーションと想定する差異

類似サービスを活用したバックアップのサンプルアプリケーションとして，類似するサービスが存在し，また高い信頼性と応答性能が求められるアプリ

ケーションが望ましい．そこで証券取引に着目する．証券取引では，複数の証券会社が様々なサービスの提供を行っている．また顧客の注文の確実な執行する必要があり高信頼性が求められる．さらに価格優先，時間優先の原則があり，同価格では先に注文を投げた注文が優先されることから，高い応答性能が求められる．以上を踏まえ，サンプルアプリケーションとして，証券取引アプリケーションを取り上げることとする．

実装を行った証券取引アプリケーションの概要を図 3.3 を用いて説明する．サンプルアプリケーションは証券会社 A，証券会社 B，証券会社 A の顧客，ならびに証券取引所から構成する．証券会社 A，証券会社 B は類似するサービスを提供しており，証券会社 A が証券会社 B にバックアップを依頼するものとする．証券会社 A ならびに証券会社 B は，商品として株式と投資信託を扱うものとする．類似サービスを活用したバックアップでは障害時に処理をそのまま継続できるケースとそうでないケースが想定でき，本サンプルアプリケーションではその例として株式と投資信託を想定する．株式については，証券会社 A，B は顧客からの株式の注文を受け付けると，証券取引所に注文を転送するものとする．このため証券会社 A の障害時には証券会社 B が代理で取引所に注文を発注し，処理を継続するものとする．一方で投資信託は証券会社 A，証券会社 B が独自の商品であり，それぞれの証券会社内で注文を処理するものとする．このため証券会社 A の障害時に証券会社 B は，注文の予約のみを受付し，回復時点で証券会社 B のシステムは予約内容を証券会社 A のシステムに伝達する．

以下，想定するシナリオについて説明する．以下文中に付随する番号は図中の番号と対応する．まず障害が起きていない通常時に証券会社 A は，顧客からの株式の注文を受け付けると (1)，証券取引所に注文を転送する (2)．次に証券取引所は，注文のマッチングを行い約定するとその結果を約定記録に保存し，またその結果を証券会社に返す (3)．一方，証券会社 A が投資信託の注文を受け付けると (4)，証券会社 A 内で注文を処理し残高を記録する (5)．

一方，証券会社 A のシステムに障害が発生した場合は，クライアント側のスタブ内で障害が検知され，バックアップである証券会社 B 側のサービスへ

処理が切り替わる (a). スタブは，クライアントアプリケーションが証券会社システムとの通信を行うための部品モジュールであり，証券会社システムの WSDL 定義を元に生成する．なお今回はクライアント側に障害検知ならびに切り替えの仕組みを有するスタブを持たせたが，この実現方法は，例えばサーバサイドでの検出と切り替えなどアプリケーションごとに違う実現方法も選択しうる．証券会社 B が証券会社 A の顧客から株式の注文を受けると (b)，証券取引所に代行して注文を転送する (c)．さらに取引所にて注文のマッチングならび約定が行われると，証券会社 B は取引結果を受信し記録する (d).

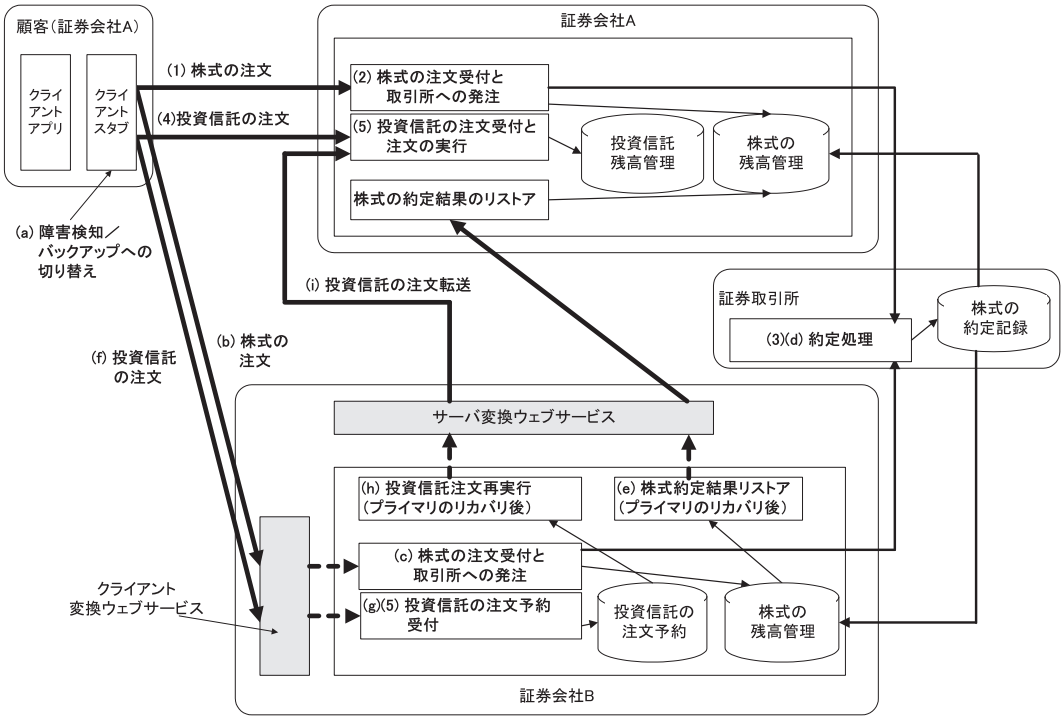


図 3.3: サンプルアプリケーション

なお，ここで取引の前提となる顧客の認証情報や残高情報はサーバ変換ウェブサービスを介して，証券会社 A，証券会社 B 間で共有しておく．ただし，情報の共有頻度によって注文の発注枠を適宜制限する [71] などの対応が必要となるが，この問題に関しては本研究では検討範囲から除外する．その後に証券会社 A が回復すると，証券会社 B は株式の注文結果を証券会社 A にリストアする (e)．一方で投資信託は，証券会社 A 内で管理される商品であり証券会社

B では受付処理が行えないものと想定する．このため証券会社 B は注文を受信すると (f)，注文予約としてのみ記録する (g)．そして証券会社 A の障害が回復した時点で，証券会社 B は証券会社 A に対して，予約を受け付けた投資信託の注文を処理するように要求を行う (h)(i)．

以上まとめると，証券会社 B は障害時において，株式の注文に関しては，証券会社 A を代行し注文処理を継続する．一方投資信託に関しては，注文の予約のみを受け付けるといったようにサービスレベルを落とした上で処理を継続する．

次に上記シナリオにおいて想定した証券会社 A (プライマリ) と証券会社 B (バックアップ) のプロトコルの差異について図 3.4 を用いて述べる．具体的には，下記 5 種類の差異を想定する．

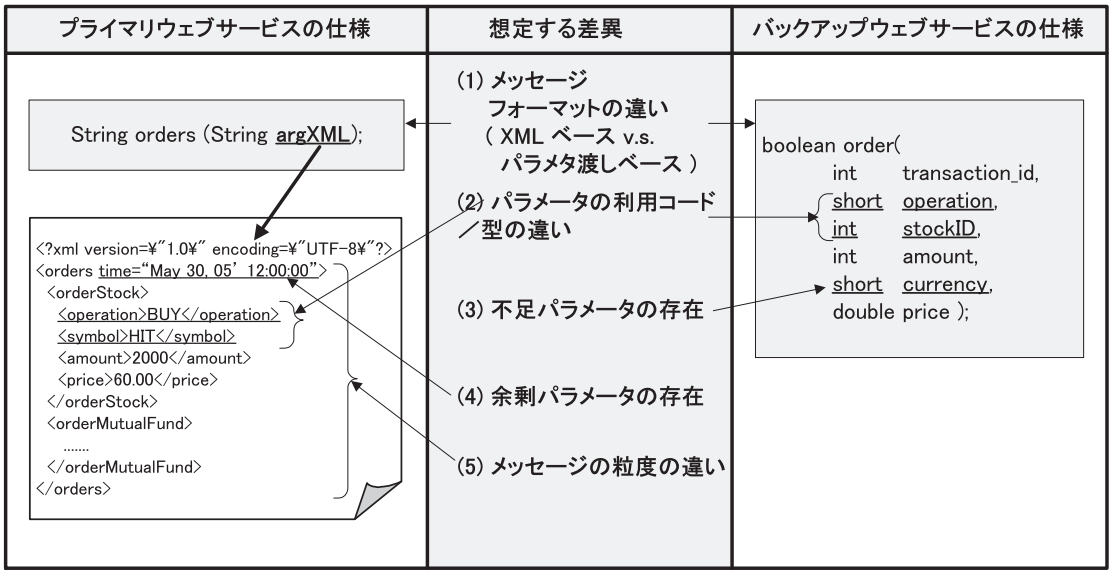


図 3.4: 想定する差異の具体例

- メッセージのフォーマットの差異

  メッセージの送受信のフォーマットが異なることを想定する．具体的にプライマリは，XML ベースでのメッセージ交換を行うのに対して，バックアップは全てのパラメータを手続き呼び出しの引数として送信するものと想定する．

- 異なるコード体系／異なる型

  パラメータの使用するコード体系やその型が異なることを想定する．具体的にプライマリのプロトコルは，売り/買いを表す文字列や銘柄名などテキストベースのコード体系を持つのに対して．バックアップは売り買いのコード，証券コードなど数字ベースのコード体系を持つことを想定する．

- 不足パラメータの存在

  バックアップ側で必要となるが，プライマリ側に存在しないパラメータがあることを想定する．具体的にプライマリ側で必要としない通貨の単位指定が，バックアップ側で必要となると想定する．

- 余剰パラメータの存在

  プライマリ側で存在したが，バックアップ側に存在しないパラメータがあることを想定する．具体的には，プライマリ側で必要である発注時刻の情報が，バックアップ側では必要とされないことを想定する．

- メッセージの粒度の差異

  一回のメッセージで伝達する内容の粒度が異なることを想定する．具体的には，プライマリは単一のメッセージで複数の注文を同時に受け付けるが，バックアップは各メッセージでは単一の注文のみを受け付けることを想定する．

3.4 変換ウェブサービスの生成

3.4.1 基本方針

変換ウェブサービスを生成する方式として図 3.5 に示す 3 つの方針が考えられる．1 つは，変換ウェブサービスのオントロジ記述を元に変換サービスを自動生成する方法である．この方法は，変換サービスの自動生成が行える反面，

生成対象のモデル化や標準化が必要である．このため部品単位でのサービスの変換は比較的容易であるがサービス全体での変換は難しい．

2 つ目は，開発者による変換ロジックの組合せにより変換サービスを半自動で生成する方法である．この方法は，人手による開発が必須となるが，変換を行いたいサービスの内容に合わせてオペレータが変換フローを定義することで，生成対象の制約が少ないというメリットがある．

3 つ目は，変換サービスを手作業にてスクラッチで開発する方法である．この方法は，半自動の場合と同様に生成対象範囲は広いが，人手による開発コストが大きく，仕様変更時の変更コストも高い．

今回は特にシステム全体としてのバックアップ (ある証券会社のシステムを別の証券会社がバックアップ) を行うことを主眼に置くことから，変換サービスを半自動で生成することが適切であると考える．以下，半自動で変換サービスを生成する上での具体的な実現方針を述べる．

	自動生成	半自動生成	スクラッチ開発
概要	オリジナルサービスの オントロジ記述を元に 変換サービスを自動生成	開発者による変換ロジックの 組み合わせにより 変換サービスを生成	変換サービスを 手作業で実装
特徴	生成対象のモデル化／ 標準化が必須	人手による開発が必須	人手による開発が必須
部品単位での活用	◎	○	△
サービス全体での活用	×（サービス全体の モデル化/標準化が困難）	○	△

半自動生成を基本方針とする

図 3.5: 変換ウェブサービスの生成方式の比較

- オリジナルのウェブサービスのインタフェース記述の活用

  オリジナルサービス (プライマリとバックアップ) の WSDL 定義を用いて，変換元ならびに変換先のインタフェースの仕様を取得する．なおサービスのメタ情報の記述としては，インタフェース記述の他に，セマンティックウェブの分野で検討されている DAML-S や OWL といったオントロジ記述 (やりとりするデータ内容の意味まで含めて表現可能) や，

BPEL[72] などの手続き呼び出しの順序関係や依存関係が表現可能なビジネスプロセス記述が存在する．しかし，現時点においてその標準化が定まり，その関連ツールも充実している WSDL をメタ情報として利用する．

- ウェブサービスとして実現

  変換ウェブサービス自体をウェブサービスとして実現する．これにより変換ウェブサービスは，オリジナルのウェブサービスと同様にアクセスが可能となる．

- 一連の変換ロジックを組み合わせたフローを指定することでコードを生成

  開発者が一連の変換ロジックを組み合わせたフローを指定することで，変換ウェブサービスのソースコードを生成可能とする．これにより開発者が自ら記述するソースコードの実装量を削減する．

- 開発者が自ら変換ロジックを追加可能なプラグインアーキテクチャを採用

  全ての開発者のニーズを満たす汎用的な変換ロジックをあらかじめ全て提供することは困難である．そこでいくつかの標準的な変換ロジックに加えて，開発者が独自の変換ロジックを追加可能なプラグインアーキテクチャを提供する．

3.4.2 変換ウェブサービスの生成アーキテクチャと生成フロー

図 3.6 に変換ウェブサービスを生成するアーキテクチャとその生成フローを示す．変換ウェブサービスを生成するためのアーキテクチャは，独自開発部分と既存ツールから構成する．独自に開発を行ったのは，変換フローを定義する GUI ツール，ソースコードジェネレータ，ならびにプラグインである．プラグインは変換ロジックを実装したものである．開発者は，標準でプラグインとして提供されるロジックに加えて，独自の変換ロジックをプラグインアーキテクチャを利用して実装することができる．なお本実装では，Java[1] JDK (Java

[1]Java は，米 Sun Microsystems 社の登録商標．

Development Kit) 1.5.02 を基盤技術として用いており，ソースコード生成ツールの開発言語，プラグインの開発言語，生成するソースコードの使用言語は，全て Java を用いる．既存ツールには，生成したソースコードをコンパイルするコンパイラ (Java) と，生成した変換ウェブサービスをウェブサービスとして登録するデプロイツール (Axis) を用いる．

以下，開発者が本アーキテクチャを用いて変換ウェブサービスのソースコードを作成する生成フローを説明する．なお下記項番は図 3.6 中の吹き出しの項番と一致する．

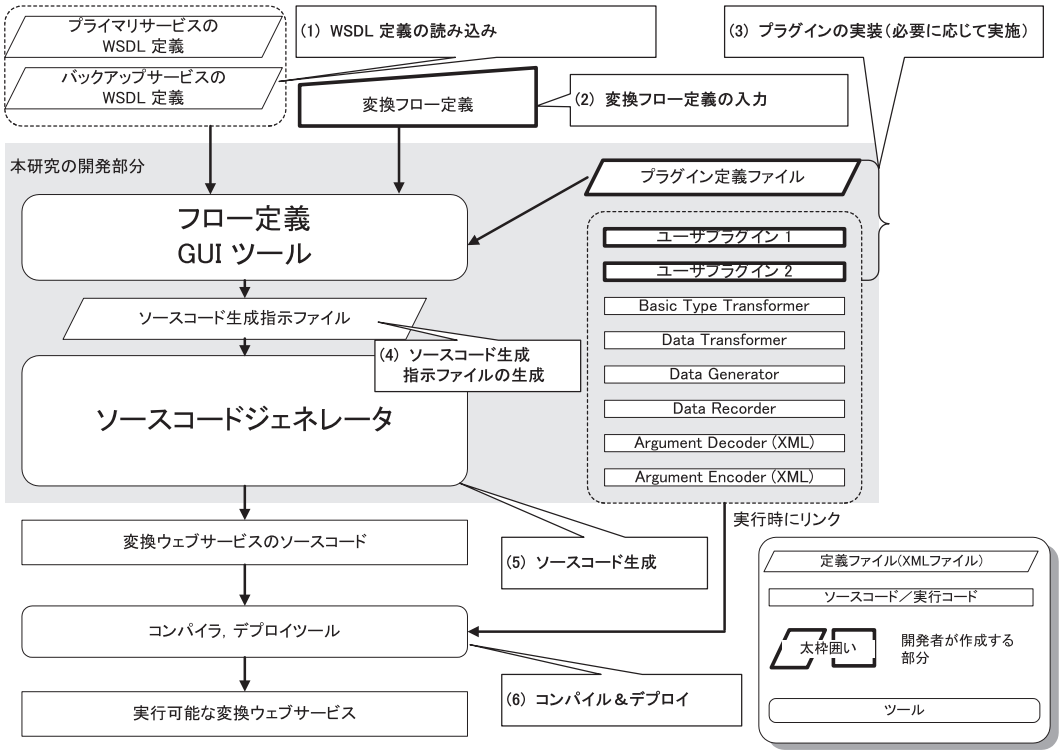


図 3.6: 変換ウェブサービス生成のためのアーキテクチャならびに生成フロー

1. WSDL 定義の読み込み

   開発者は，GUI ツールに，プライマリならびにバックアップそれぞれのウェブサービスの WSDL 定義ファイルを読み込ませる．なお WSDL 定義ファイルは，オリジナルのウェブサービスから標準のツールを用いて自動的に生成されるファイルである．

2. GUI ツールを用いた変換フロー定義

   開発者は GUI ツールを使って，変換に必要となる変換ロジックを選択し，それらを一連のシーケンスとして結合した変換フローを作成する．変換フローの定義の詳細については 3.4.3 項にて述べる．

3. プラグインの実装 (必要に応じて)

   開発者は，変換フローの作成において，既存のプラインで用意された変換ロジックで不足がある場合には，必要に応じて独自のロジックをプラグインとして追加可能である．具体的には変換ロジックを実装したクラスとそのクラスの内容を GUI ツールあるいはジェネレータが把握するためのプラグイン定義ファイルを作成する．プラグインの作成方法については，3.4.4 項にて述べる．

   以下に具体的なプラグインの例として，サンプルプログラムを実現する上で作成したプラグインを示す.

   - Basic Type Transformer

     基本タイプ間の変換を行う．Long, Integer, Short 間の変換，Float, Double 間の変換，文字列データと数値データ間の変換機能を提供する．

   - Data Transformer

     変換テーブルを読み込みデータのコード変換を行う．例えば商品コードが違う場合には，商品コードの対応テーブルを読み込みその間の変換機能を提供する．

   - Data Generator

     Backup 側の呼び出し時に必要となるが，Primary 側での呼び出しには存在しないような不足パラメータの生成を行う．例えば時刻情報などの生成機能を提供する．

   - Data Recorder

前項とは逆に，Primary 側では必要とされていたが，Backup 側では不要のパラメータが，データのリストア時などで必要となる場合には，そのデータを記録する機能を提供する．

- Argument Decoder

  ある構造を持ったデータを解析しパラメータを抽出する．本実装ではその具体例として，XML のデコーダを実装し，XPATH[73] に準拠したパスを指定することで，パラメータの抽出を可能とする．

- Argument Encoder

  ある構造を持ったデータを生成する．本実装では，一連の入力パラメータとそれを配置する XPATH の入力を受けて，XML 文書を合成する機能を提供する．

4. ソースコード生成指示ファイルの生成

   GUI ツールは，入力を受けた WSDL 定義ファイル，変換フロー定義，プラグイン定義ファイルから，ソースコード生成指示ファイルを中間データとして生成する．なお GUI ツールが直接ソースコードを生成しないのは，ソースコード生成支持ファイルにソースコードを生成するために必要な情報に加えて GUI の作業を継続するのに必要な情報もあわせて保存するためである．

5. ソースコード生成

   ソースコードジェネレータは，ソースコード生成指示ファイルを参照し，変換ウェブサービスのソースコードを生成する．実際に生成されるソースコードの例を 3.4.5 項で示す．

6. コンパイル，デプロイ

   生成された変換ウェブサービスのソースコードをコンパイルしデプロイする．また実装したプラグインも適宜コンパイルを行い実行時にリンク可能なように配置する．

上記一連の流れで，実行可能な変換ウェブサービスが構築できる．

3.4.3 変換フローの定義

GUI ツール

GUI ツールの画面イメージを図 3.7 に示す．開発者は GUI ツールを用いて，(1) フローを定義するメソッドの選択，(2) コンポーネントの結合，(3) 各コンポーネントのパラメータの指定を行うことで一連の変換フローを作成可能である．以下具体的な処理ステップについて述べる．

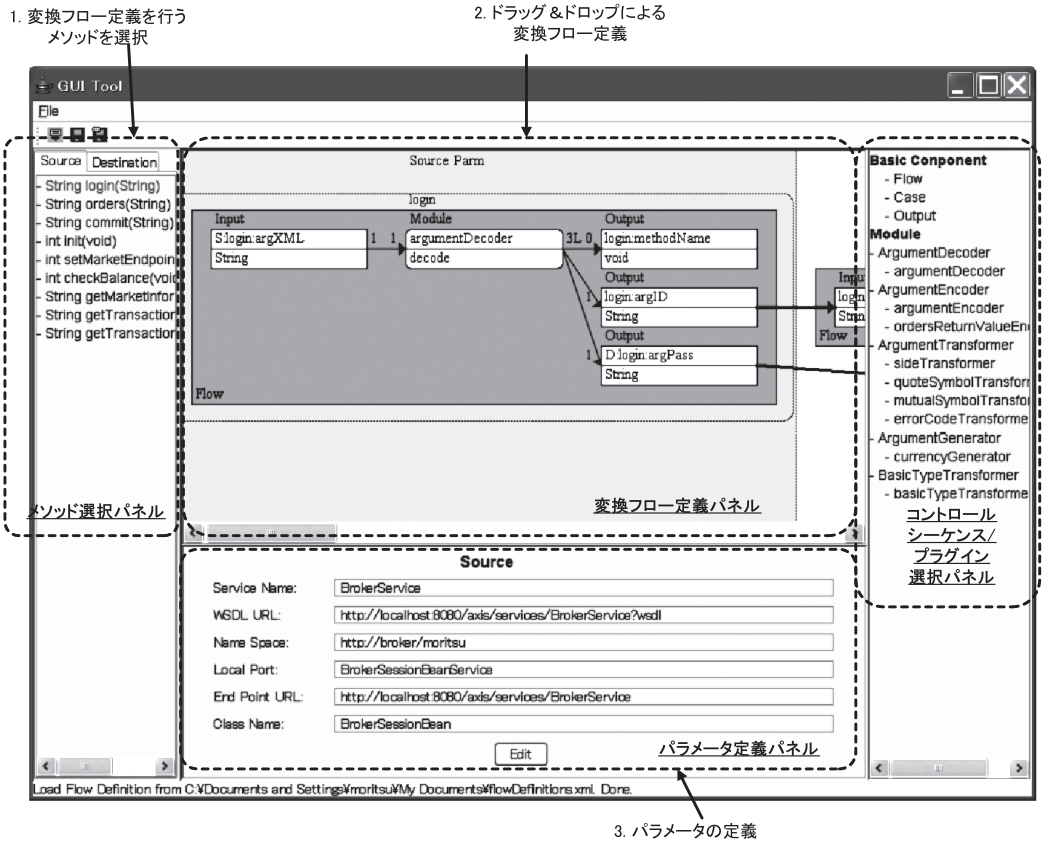


図 3.7: GUI ツールの画面イメージ

1. 変換フローを定義するメソッドの選択

   プライマリならびにバックアップのウェブサービスのメソッドは，その WSDL の定義を元に GUI ツールに読み込まれ画面上 (メソッド選択パネル) に表示される．開発者は変換フローを定義したいプライマリウェブサービスのメソッドを選択する．

2. コンポーネントの結合

GUI ツールはフローの構成要素となるコンポーネントの読み込みを行い画面上に表示する．開発者はドラッグ & ドロップを中心としたインタフェースによりコンポーネントを変換フロー定義パネルにドロップし，同じくマウスにてコンポーネントの結合することで変換フローを定義する．具体的に以下がコンポーネントとして利用可能である．

- コントロールシーケンス

  フローを形成するために必要な条件分岐などの基本的な制御シーケンス群．

- 変換ロジック (プラグインモジュール)

  プラグインとして実装された変換ロジック．GUI ツールは，プラグイン定義ファイルを読み込み，読み込んだプラグインを画面に表示する．開発者は表示されたプラグインの一覧の中から組み込みたい変換ロジックを選択することでフローを構成することができる．

- バックアップメソッド

  フローに結合するバックアップサービスのメソッド．GUI ツールは，バックアップウェブサービスの WSDL 定義を読み込み，バックアップのメソッドを画面に表示する．開発者はフローに結合したいバックアップのメソッドを一覧から選択し変換フロー定義パネルにドロップすることで，フローから呼び出すメソッドを指定できる．

3. パラメータの定義

コンポーネントによっては，フローに組み込む際にいくつかのパラメータを指定する必要がある．開発者は，このようなパラメータをパラメータ定義パネルにより入力する．

変換フローの例

図 3.4 で示した差異を吸収する変換フロー (正確にはその一部で引数の変換までを示したもの) を，GUI ツールを用いて定義する例を図 3.8 に示す．実際には，変換フローは GUI ツールの画面上で定義を行うが，図 3.8 は説明のためにその定義内容を図示したものである．図中の番号は以下のステップの番号と一致する．

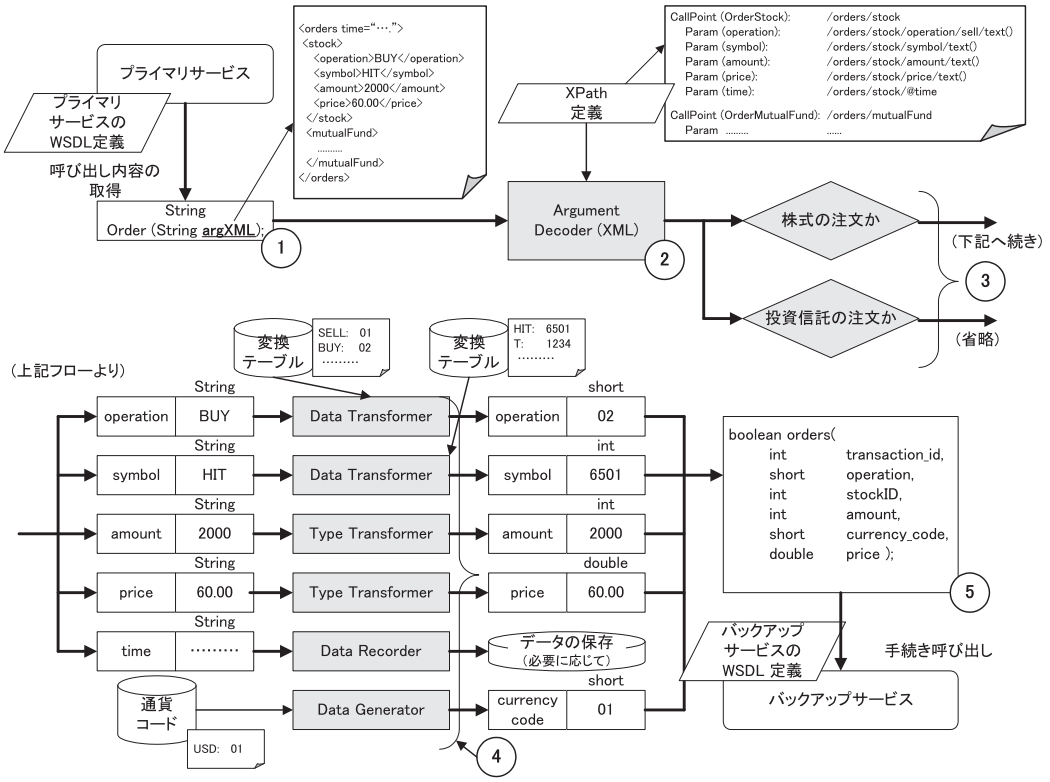


図 3.8: 変換フローの定義例

1. プライマリ側の WSDL 定義を参照し，手続き名称，引数の数と型，返値の型の情報を抽出する．以下，引数ごとに変換のフローを定義する．本例では引数として，XML 文書を格納した String データが存在するので，この String データに関する変換フローを定義する．

2. 引数の XML 文書を解析する必要があるので， XML の解析を行うプラグインである Argument Decoder を配置する．本プラグインは，手続き

(CallPoint) とそのパラメータを指定するパスとを XPath 形式で指定することで，XML 文書内に格納されたデータを抽出する．本例では具体的に，株式の注文，投資信託の注文が XML 文書内に組み込まれているので，それぞれの注文を特定するパスとそれらの注文に用いられるパラメータのパスを XPath 表記で与えることでそれぞれの注文データを抽出することができる．

3. 前処理ステップで抽出された手続きの種類によりフローの分岐を行う．本例では，株式の注文と投資信託の注文があるのでこれを分岐させる．この分岐機構により，商品ごとにバックアップのサービスレベルが異なる場合に処理を切り替える．例えば本例では，株式に関してはバックアップ側はプライマリ側に替わり取引所へ代理発注を行うが，投資信託に関してはキューイングのみを行う．このため株式に関してはバックアップの発注処理へ接続する変換フローをこの後定義し，投資信託に関してはバックアップ側のキューイング処理へ接続する変換フローを引き続き定義する．以下は，投資信託のフローについては省略し，株式の注文に関するフローを記載する．

4. 抽出されたパラメータごとに変換を行うプラグインを配置し，バックアップ側の呼び出しに適したデータフォーマットに変換する．本例では変換の具体例として，単純な型変換を行う場合 (Type Transformer) や，マッピングテーブルを用いた変換 (Data Transformer) や余剰パラメータの保管 (Data Recorder), 不足パラメータの生成を行う．

5. バックアップ側の WSDL 定義を参照し，変換を行ったパラメータをそれぞれの引数に結びつけることにより，バックアップ側の呼出を作成する．

なお本フローに引き続き，バックアップ側の返値をプライマリ側の返値へマッピングするフローが別途必要となるが，そのフローについては省略する．

3.4.4 プラグインの開発

プラグインを開発するために開発者は，変換ロジックを実装するクラスとプラグイン定義ファイルを記述する必要がある．プラグイン定義ファイルは GUI ツールにより参照され，ソースコードジェネレータにより生成されるソースコードからプラグインで実装されたロジックを呼び出すための情報として用いられる．クラスの実装はいくつかのルールに従って行う必要がある．以下，それらの実装ルールとそれに関連したプラグイン定義の記述を図 3.9 に示した記述例を用い説明する．

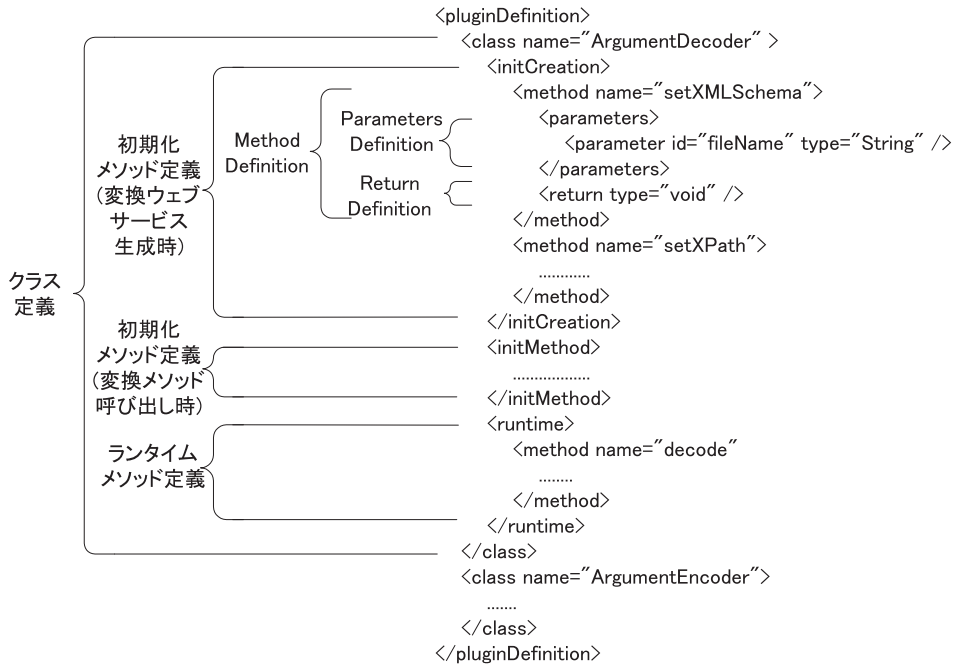


図 3.9: プラグイン定義ファイルの例

- ランタイムメソッドの定義

  開発者は，変換ロジックを実装したクラス内に，一つ以上のランタイムメソッドの定義を行う必要がある．ランタイムメソッドは，変換ロジックを定義したメソッドで変換フローの中から呼び出される．ランタイムメソッドの第 1 パラメータは，変換フローでの前のコンポーネントから値を受け取る変数であり，一方，ランタイムメソッドの返値は変換フロー

の次のコンポーネントへ値を渡す変数となる．また必要に応じてランタイムメソッドにはいくつかのオプションのパラメータ (2 番目以降の引数) を追加できる. これらオプションのパラメータにより例えば XML のデコードに用いるプラグインであれば，参照する XML Schema ファイル (文法ファイル) を指定するといった利用が可能である．

一方，プラグイン定義ファイルには，どのメソッドがランタイムメソッドであるかを定義する．また，変換フローの入力 (第 1 引数)，出力 (返値) となるパラメータの型情報を記述する．これにより変換フローで渡される値の型チェックが可能となる．さらにオプショナルなパラメータが存在する場合にはその型情報を記述する．GUI ツールはこの情報を元に，変換フロー定義時にオプションパラメータの入力を受け付ける．

- 初期化メソッドの定義

  開発者は，変換ロジックを実装したクラス内に，2 種類の初期化メソッドを定義することができる．一つは変換ウェブサービス生成時に起動される初期化メソッドで，もう一つは変換メソッドが呼び出された時に起動される初期化メソッドである．プラグイン定義ファイルには，そのメソッドが変換ウェブサービス生成時に呼び出される初期化メソッドかあるいは，変換メソッド起動時に呼び出される初期化メソッドかを記述する．さらにこれらのメソッドが引数を持つ場合には，そのパラメータ名と型情報を指定することで，上記ランタイムメソッドの場合と同様に変換フロー定義時にパラメータの入力を促す.

3.4.5 生成する変換ウェブサービスのソースコード

ソースコードジェネレータが生成するソースコードの例を図 3.10 に示す. ジェネレータは，プライマリのウェブサービスの WSDL を元に，プライマリと同一のインタフェースを有するクラスを生成する．具体的には生成するクラスは，プライマリウェブサービスと同一名称かつ同一のパラメータを有するメソッド群 (図 3.10 中の (3) 変換メソッド) を有する．これによりプライマ

リウェブサービスのクライアントは，プライマリウェブサービスと同様に変換ウェブサービスにアクセスすることが可能となる．生成するソースコードは，上記変換メソッド群と，初期化ルーチンから構成される．初期化ルーチンとして，以下の機能を有するコードを生成する．

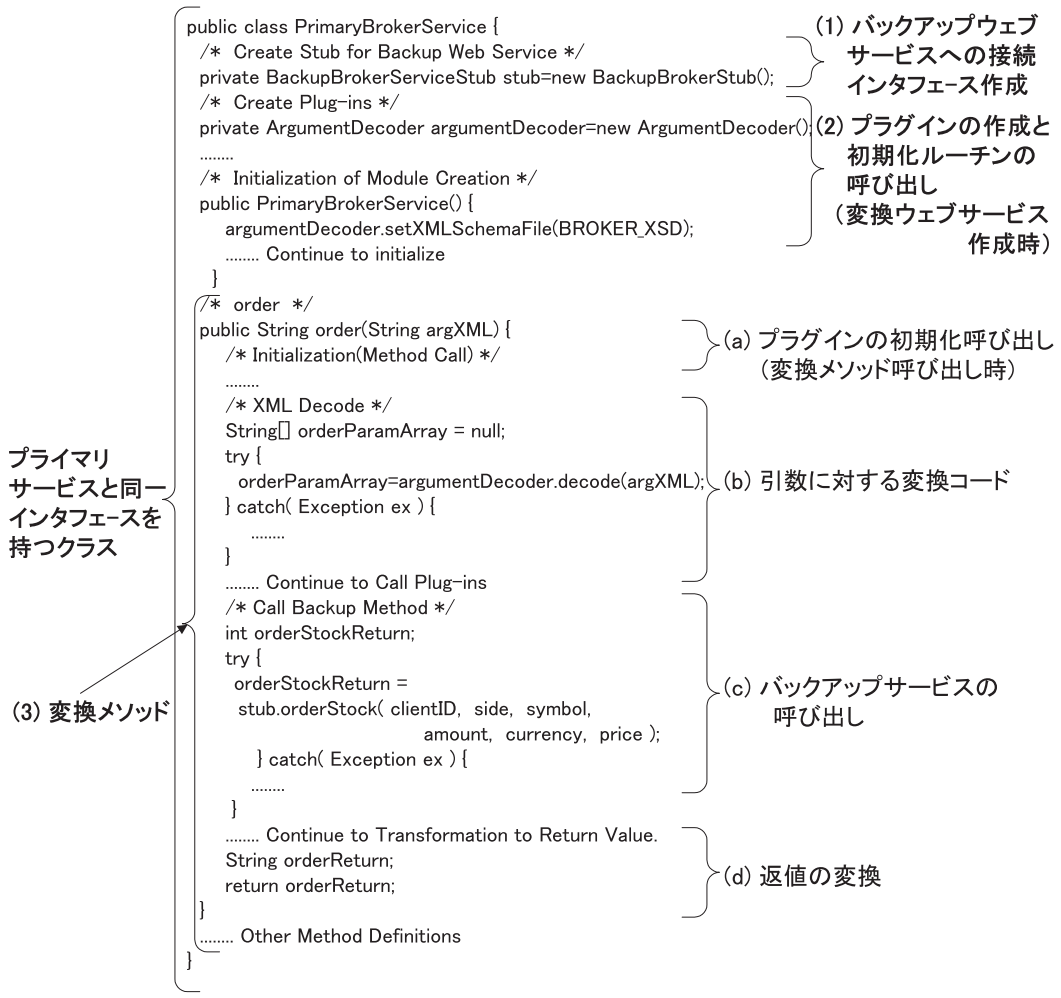


図 3.10: 生成されるソースコードの例

- バックアップウェブサービスへの接続インタフェース

  変換ウェブサービスからバックアップウェブサービスを呼び出すためのインタフェースを作成する機能．本コードの例では単純に別クラスとして実現されたバックアップウェブサービスのスタブの生成を行う．

- プラグインの作成と初期化

変換フローの中で利用されたプラグインのインスタンス生成を行う．またそれらプラグインの初期化 (変換ウェブサービス作成時) メソッドを呼び出す.

次に変換メソッドの構造についてその概要を示す.

- プラグインの初期化

  ジェネレータは，変換メソッドの開始処理として，プラグインの初期化メソッド (変換メソッド起動時) を呼び出す.

- 引数に対する変換

  プライマリウェブサービスの仕様で受け取った変換メソッドの引数を，バックアップウェブサービスの仕様に変換にする一連の処理．変換フロー定義を元に一連の変換ロジックを順次呼び出すコードを生成し変換を行う.

- バックアップウェブサービスの呼び出し

  バックアップウェブサービスの仕様に変換したパラメータを元にバックアップサービスのメソッドを呼び出す.

- 返値に対する変換

  バックアップウェブサービスから受け取った返値を，プライマリウェブサービスの返値の仕様に変換する一連の処理．引数の変換と同様に変換フロー定義を元に作成する.

3.5 類似サービス活用バックアップ方式のオーバヘッド評価

障害時にバックアップの証券会社に処理を切り替えに要する時間とバックアップ時のスループットを示し，その内訳を分析する．まず評価を行ったアプリケーションの実装について述べ，続いて評価結果を述べる.

3.5.1 評価アプリケーションの実装

評価を行った証券取引アプリケーションの実装の概要は図 3.11 に示すように，顧客システム，プライマリとなる証券会社 A システム，バックアップとなる証券会社 B システム，証券取引所システムを 100 Mbps のイーサネットで接続して評価を行った．図 3.11 中で，濃い網掛けを行った部分がこれまで述べた変換ウェブサービスの生成技術で生成した部分である．また薄い網掛けを行った部分が評価のため実装を行った証券取引の業務アプリケーションである．評価を行うハードウェアは，いずれのシステムも Intel Xeon 2.40 GHz CPU の 2 プロセッサ構成で，1GB のメモリを搭載した Dell[2] 2650 を用いた．また OS には Red Hat[3] Linux 7.3 を用い，アプリケーションの開発環境/実行環境には Java JDK 1.5.02 を用いた．

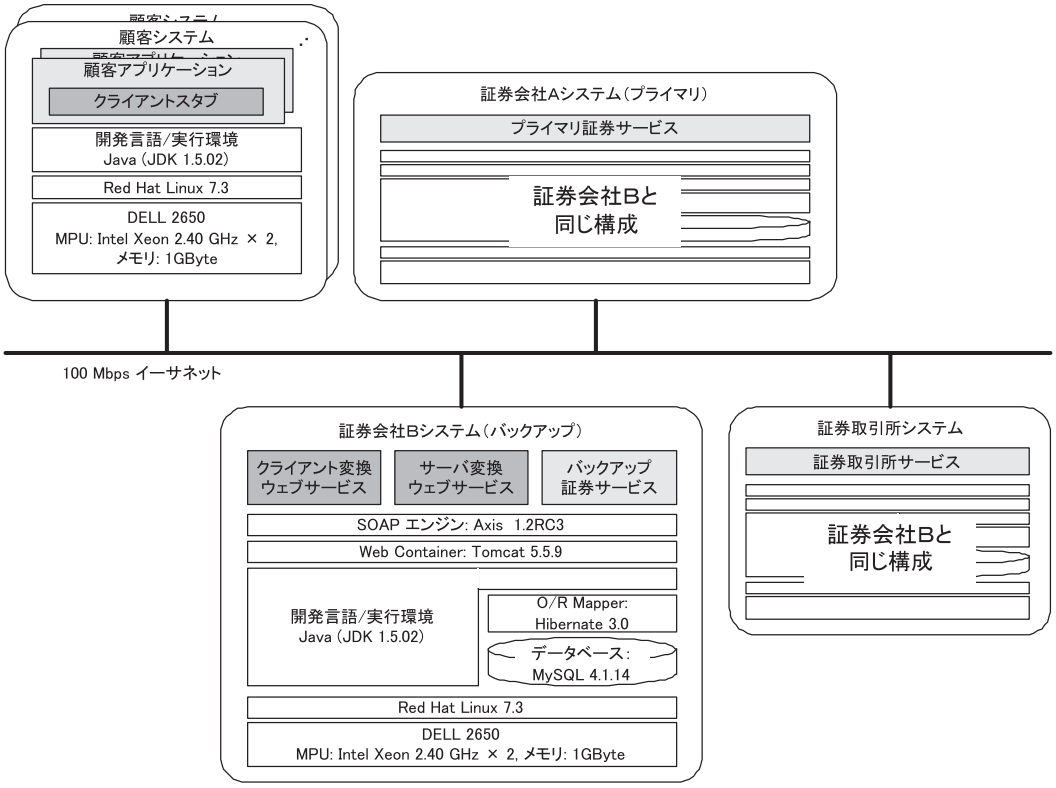


図 3.11: 評価を行った証券取引アプリケーションの実装

[2]Dell は，米 Dell 社の登録商標．
[3]Red Hat は，米 Red Hat 社の登録商標．

顧客システムにはこの環境上に，自動で株式や投資信託の発注を行う顧客アプリケーションの実装を行った．さらに顧客アプリケーションには，変換ウェブサービスと共に生成したスタブを，3.3 節で述べた障害検知と障害時の目的で組み込んだ．なお評価は，ハードウェア数の制約から，顧客システムに 2 台のハードウェアを用い，それ以上の顧客数を想定する場合は，これらの実行パフォーマンスがボトルネックになっていないことを確認した上で，同一マシン上の複数のプロセスとして実行した．

一方，証券会社 A，B システム，証券取引所システムには，ウェブサービスを実装するための環境として，Servlet の Web Container である Apache の Tomcat 5.5.9 と，SOAP[74] エンジンである Apache の Axis (1.2RC3) を用いた．また取引結果を保存するデータベースとして，MySQL 4.1.14 を用い，Java 上のオブジェクトを MySQL に格納する O/R Mapper (Objects to Relational Databases Mapper) として Hiberante 3.0 を用いた．これらの実行環境上に，業務ロジックである，プライマリ証券サービス，バックアップ証券サービス，証券取引所サービスを実装した．さらにバックアップ先となる証券会社 B システムには，これまで述べた生成技術で生成を行ったクライアント変換ウェブサービス，サーバ変換ウェブサービスを，バックアップ証券サービスと同一の Web Container 上で実行した．

3.5.2 評価結果

表 3.1 は，障害時にバックアップの証券会社システムに切り替えるオーバヘッドを示したものである．切り替え時間は 3.95 sec であった (測定は 100 回のトライアルの平均)．またその内訳は，障害検知で 12.0 %，変換ウェブサービスへの接続で 38.4 %，変換サービスで初期化に 11.0 %，バックアップサービスへの接続で 32.9 %，バックアップサービスの初期化で 5.6 % のオーバヘッドがそれぞれかかっている．この中で特にオーバヘッドが高いのは，変換ウェブサービスへの接続とバックアップウェブサービスへの接続時間である．その中身は，前記サービスの接続，具体的に処理を実行するウェブサービスのインスタンス (プロセス) の生成である．この詳細については Axis 内で処理される

ため今回の実験では測定できなかったが，クライアントと変換ウェブサービスの接続はローカルエリア内での接続であり，また変換ウェブサービスとバックアップウェブサービスとの接続はプロセス間通信であり，その通信オーバヘッドは限定的であることから，処理を実行するウェブサービスのインスタンス生成のオーバヘッドが高いと予想される．なお表 3.1 の測定結果では，各時間のばらつきが小さかったため，標準偏差の記載は省略した．

表 3.1: バックアップシステムへの切り替え時間とその内訳

内訳/合計	時間 (全切り替え時間に占める割合)
障害検知	475msec (12.0%)
変換ウェブサービスへの接続	1,520msec (38.5%)
変換ウェブサービスの初期化	435msec (11.0%)
バックアップサービスへの接続	1,300msec (32.9%)
バックアップサービスの初期化	220msec (5.6%)
合計	3950msec (100.0%)

図 3.12 は，通常時にクライアントがプライマリの証券会社を用いて株式の発注 (証券取引所への注文転送時間も含む) を行った場合の処理時間と，障害時に同じくクライアントが同株式の発注をサーバ変換ウェブサービス経由でバックアップの証券会社を用いて行った場合の処理時間を示したものである．

測定時間は，100 回トライアルを行った結果の 1 トランザクションあたりの処理時間 (クライアント側でのラウンドトリップタイム) の平均値である．実験では複数クライアント (1, 2, 4, 8) から並列かつ連続的にそれぞれ 1 秒間隔で注文要求を行った．

測定結果を見ると 1 クライアントにおける場合では，変換処理を介することで，通常時のプライマリの証券会社が処理を行う場合と比較して, 約 801 ms の応答時間の低下が見られる．またクライアント数が 2, 4, 6, 8 と増えるにしたがって，遅延は，917 ms, 3223 ms, 6393 ms, 7303 ms と拡大する．

次に，オーバヘッドの詳細について分析を行った結果を，図 3.13 に示す．図

3.13 は，同時アクセス数が 1 クライアントの条件で，障害時に，サーバ変換ウェブサービス経由でバックアップの証券会社が注文処理を行った場合の処理時間の内訳を示すものである．

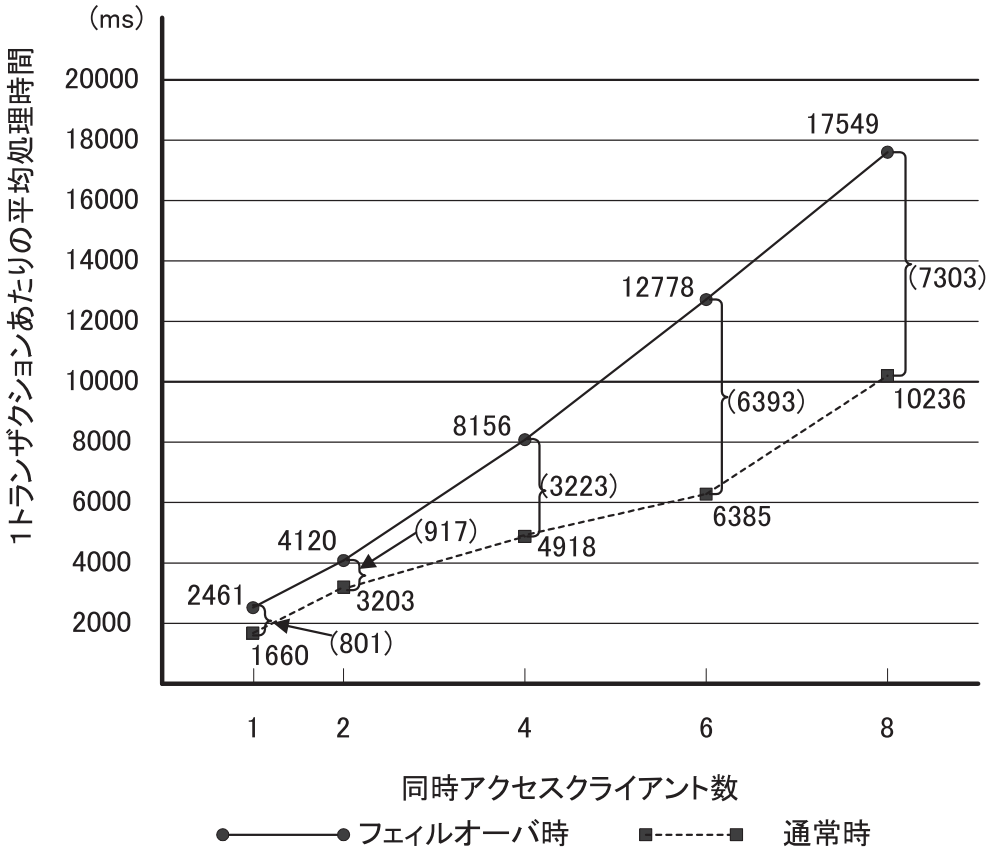
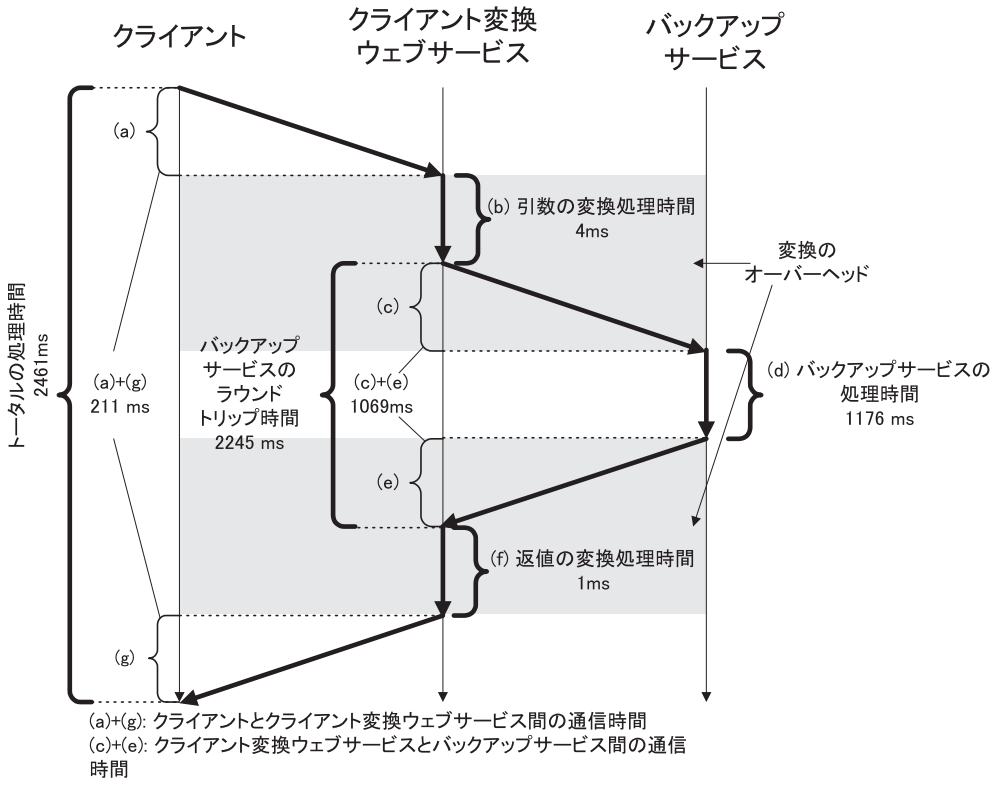


図 3.12: トランザクションごとの処理時間

図中の括弧で示された時間が実際に測定を行った平均時間であり，クライアントとクライアント変換ウェブサービス間の通信時間ならびに，クライアント変換ウェブサービスとバックアップの証券会社との通信時間は差分により算出した．また，これらの通信時間は，SOAP エンジンの呼出し時点を基準とするものであり，実際のネットワークの通信時間に加えて SOAP エンジン内で，メッセージのエンコードならびにデコード時間も含む．これらの処理時間の中で，変換に要するオーバヘッドは，図中の網掛けを行った部分であり，(b) 引

数の変換に要する時時間と，(c)(e) サーバ変換ウェブサービスとバックアップサービス間の通信時間と，(f) 返値の変換に要する時間との合計 1074 ms である[4] .



	変換ロジックの処理時間 (b)+(f)/(b)+(c)+(e)+(f)	通信時間 (c)+(e)/(b)+(c)+(e)+(f)
全オーバヘッドに占める割合	0.47 %	99.53 %

図 3.13: 処理時間の内訳

測定の結果，オーバヘッドの内訳は，引数ならびに返値の変換に要する時間 (b)+(f) が 5 ms で，サーバ変換ウェブサービスとバックアップサービス間の通信時間 (c)+(e) が 1069ms であり，通信時間がオーバヘッドの約 99.53 % を

[4]本オーバヘッドの時間が 図 3.12 の 1 クライアント時の通常時とフェイルオーバ時の差 801 ms と異なるのは，プライマリサービスとバックアップサービス内での処理時間に違いがある (プライマリの証券会社とバックアップの証券会社の扱うプロトコルが違う) ことによるものと考えられる.

通信時間が占めている．この割合は変換ロジックの複雑さにより変動するものではあるが，その差が非常に大きいことから，現状では通信時間がボトルネックになっているものと考えられる．

そこで，さらに通信時間のオーバヘッドの内容を調べるために，上記と同じ環境にてウェブサービスの通信時間の測定を行った．具体的には処理に何も含まない空のウェブサービスを呼出すことで SOAP の手続き呼出のオーバヘッドを測定した結果を表 3.2 に示す．測定は，クライアントとして Java アプリケーションから呼び出す場合と，本例のようにウェブサービスからさらに別のウェブサービスを呼び出す場合を測定する．また呼出時のパラメータについても，パラメータなしの場合と，本実験の場合と同条件であるパラメータ数 (7 個) の場合を測定した．

結果を見ると，Java アプリケーションから呼び出す場合よりもウェブサービスから呼び出す場合の方がオーバヘッドが大きいことがわかり，また引数によるパフォーマンスの低下も見られることがわかる．また性能評価と同一条件の，ウェブサービスからの呼び出しで 7 パラメータの場合は 1100 ms のオーバヘッドが生じており，性能評価時に得られた通信のオーバヘッド (c)+(e) である 1069 ms とほぼ一致する．以上のように，SOAP の通信のオーバヘッドが比較的大きいことから，全ての変換ロジックを内包する一つのウェブサービスを静的に生成することは，変換のための通信オーバヘッドを最低限に限定する意味で効果があると考えられる．

表 3.2: SOAP 通信のオーバヘッド

	引数なし	本評価と同一条件（引数 7 つ）
Java アプリからの呼び出し	150 ms	570 ms
ウェブサービスから呼び出し	415 ms	1100 ms

3.6 結言

サービスの違いを吸収する変換ウェブサービスを生成する仕組みを構築し，類似サービスによるバックアップのオーバヘッドを検証した．具体的には変換ウェブサービスのソースコードを，オリジナルサービス (プライマリ，バックアップ) の WSDL の定義，変換ロジック群，変換フロー定義から生成した．これを実現とするシステムとして，開発者による変換フロー定義を支援する GUI ツール，変換ウェブサービスのソースコードを生成するジェネレータ，開発者が独自の変換ロジックを追加可能なプラグインアーキテクチャの開発を行った．

さらに証券取引を具体例としたサンプルアプリケーションを開発し，バックアップシステムへの切替時間とバックアップシステムを用いて代理注文を行った場合のスループットについて測定を行った．特にスループットのオーバヘッドは，その 99.53 % は SOAP の通信時間によるものであることを確認した．この事実から通信のオーバヘッドの高いウェブサービスにおいては，変換ロジックを事前にコンパイルして 1 変換サービス内に集約する本手法は，通信オーバヘッドを最低限に押さえる上で効果があると考えられる．

本研究には 3 つの課題が残る．1 つ目の課題は処理性能の向上である．現状での処理性能は，1 ユーザの場合でも 2.5 秒のオーバヘッドを要しており，実用レベルに達していない．そのオーバヘッドの多くは SOAP による通信のオーバヘッドであるが，今後 SOAP エンジンの処理性能の向上に加えて更なる高速化技術が求められる．解決の一つの方向性はトランザクションの処理工程を細分化し並列に処理を行うパイプライン処理である．近年，処理時間を要す SOAP のエンコード/デコードの部分をアプリケーションサーバから切り離し専用ハードウェアで処理を行うという製品が実現されている [75]．今後，証券メッセージが XML 化された場合には SOAP 上に乗るメッセージも XML 化されることから，SOAP に関する XML 処理，証券メッセージに関する XML 処理，アプリケーションサーバの処理など，それぞれの処理を細かい処理に分割して，これらの一連の処理をパイプラインとして高速に実行することで応答性能を向上できると考えられる．なお一方，全体のスループット向上に関しては，トランザクション単位で処理を振り分けて実行する分散処理も有効である

と考えられる．

2 つ目の課題は，変換ウェブサービスの生成の省力化があげられる．現状は，標準化の進展具合からオリジナルのサービスのメタ情報として，インタフェース記述 (WSDL) のみを利用した．このため一連の変換フローの作成の多くは，ユーザの入力に頼っている．メタ情報としてセマンティックウェブ [76] で行われているような具体的な通信内容の意味まで表現可能なオントロジ記述 [42][43] や，BPEL などのビジネスプロセス記述活用することで，変換フローの生成の人手作業の省力化が促進可能であると考えられる．

3 つ目の課題は，バックアップ以外の目的での類似サービスの活用に関する検討である．類似サービスの活用機会としてはバックアップ以外にも，負荷分散や，サービス統合 (異なる地域や市場に対して類似サービスを統合してサービスを提供する) といった様々な利用機会が考えられる．これらそれぞれのケースに応じて，どのような類似サービスの連携が好ましいかについて今後さらなる検討が必要である．

第4章

整数線形計画を用いた電子債権譲渡における期日変更マッチング最適化方式

4.1 緒言

本章では，電子商取引決済の柔軟化に関し，電子債権の活用を促進する，整数線形計画を用いた電子債権譲渡における期日変更マッチング最適化方式について述べる．

企業が保有する電子債権を支払に充当する債権譲渡は，譲渡人は振出人の信用力を活用できるメリットがある [32]．図 4.1 は，手形による支払の概略を示した図である．図 4.1 の企業 A が商品の購入やサービスの対価として代金を手形で行う方法として，手形の新規発行と手形譲渡の二種類の方法がある．新規発行は，支払に対して企業 A が新たな手形を発行して支払に充当するものである．一方，手形の譲渡は，企業 A が保有する手形を支払に充当する方法である．この手形譲渡による支払は，企業 A の信用力が低く，企業 A が新規に発行する手形を企業 B に受け取ってもらえない場合においても，手形の振出人 (図 4.1 の企業 C) の信用力が高ければ，支払に充当できるメリットがある．このような第三者による信用補完は，国内で約 6 割を占める [77] 企業系列の下請企業のファイナンスでは，親会社の信用力を活用できるため，特にそ

の有用性が高い.

しかしこれまでの指名債権，手形債権では，保有する債権を債権譲渡に活用できる機会は限定的であることが指摘されている [33]．その理由の一つとして，支払条件 (金額と期日) と債権の償還 (払い戻し) 条件が必ずしも一致しないことがあげられる.

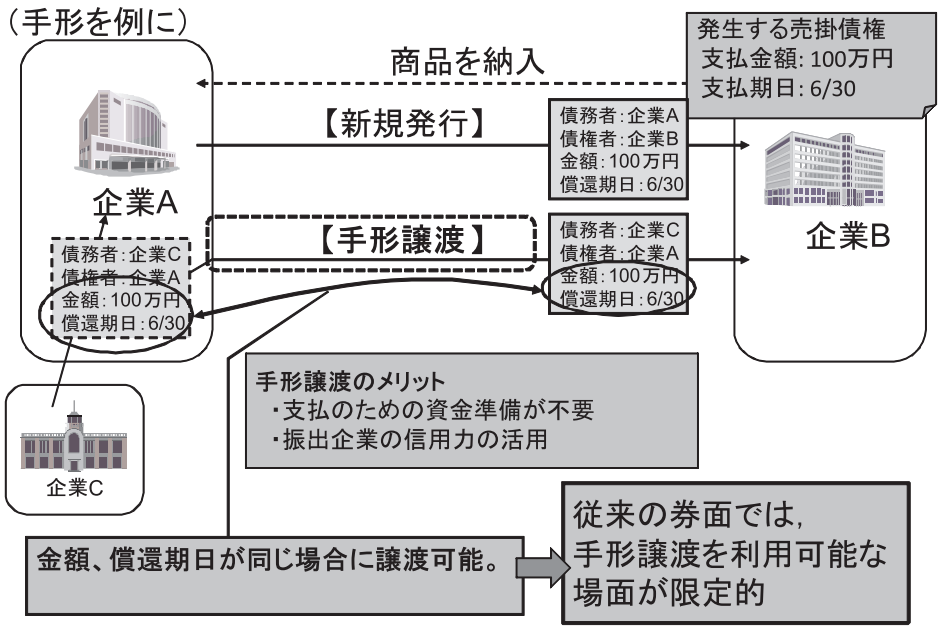


図 4.1: 手形譲渡の例

電子債権では条件の書換が電子的に簡易に行えることから金額の差異に関しては，電子債権の分割により比較的容易に対応でき，既に実用化しているシステムも存在する [78]．一方，期日が一致しない場合は，条件変更に先立ち参加者 (振出人，譲渡人，譲受人) 間での利害調整が必要である．この期日変更は，企業が余剰している手元流動資金 (以下単に資金と記載) を有効活用したり，不足する資金を調達する手段を提供するという側面を持つ．このため，金利をどの程度にするか，延長短縮をどのくらいの期間に設定するかといった事項について合意する必要がある.

これら期日変更に伴う合意形成を，電子債権の譲渡による支払の都度，参加者間でワークフローを形成し条件の提示と承認を繰り返すことは，条件の食い

違いによる出戻りを考慮すると時間がかかり困難である．この合意を効率的に行うには，参加者が事前に資金の増減目標と期日の変更を許容する変更条件を登録した上で，これらの制約を満たす電子債権と支払の組み合わせを決定する仕組みが必要である．そこで，全参加者の資金の増減目標を極力達成するように電子債権と支払の組み合わせを最適化するマッチング方式を提案し，これが整数線形計画問題 [79][80] として定式化できることを示す．また国内の企業の手形や売掛債権の発行状況を模した状況下で電子債権取引シミュレーションを行い，まず期日マッチング機能によりどの程度債権譲渡が促進されるかを検証する．さらに提案した最適化方式によるマッチング結果を，他のマッチング方式と比較して最適化の効果を検証する．

以下，4.2 節で電子債権の期日マッチング方式の概要を示す．次に 4.3 節で電子債権と支払の組合せ問題の定式化と，定式化の具体例を示す．さらに 4.4 節でその評価を行い，4.5 節で検討をまとめ，今後の課題を示す．

4.2　電子債権の期日マッチング方式

図 4.2 に，電子債権の期日マッチングの一連の処理の流れを示す．まず (1) 参加者がそれぞれ資金の増減目標と期日変更条件を登録する．その上で，(2) 支払の登録を受け付け，(3) これらの情報を元に電子債権と支払の組合せを決定し，(4) マッチング結果に対する承認を得る．期日変更条件として，償還/支払期日の変更可能範囲，金利条件を想定する．以下，参加者が指定する資金の増減目標，償還/支払期日の変更可能範囲ならびに金利条件について述べる．

- 資金の増減目標

  各参加者は，どのタイミングでどの程度，流動資金を増減させたいかを指定する．例えば，今後の流動資金の推移見込み (今後の支払い受け取り予定等から作成) と今後の各タイミングでの流動資金の保有目標を設定した上で，保有目標と推移見込みの差により増減目標を決定する．この増減目標で，例えば，図 4.3 で示すように振出人 i が流動資金の増額を要求した場合には，マッチングシステムは，振出人 i が振り出した電

子債権の償還日を延長してマッチング可能な支払を見つける．このマッチングにより，振出人 i は支払日が遅延されることで，その間，資金を手元に置くことが可能となり流動資金が増える．

- 償還/支払期日の変更可能範囲

  参加者は，支払期日や償還期日の延長ならびに短縮範囲を指定する．

- 金利条件

  参加者は，償還期日の延長もしくは支払期間を短縮した場合に受け取る下限金利，逆に償還期日を短縮もしくは支払期日を延長した場合に支払う上限金利を指定する．なお金利は債務者の信用力に応じて設定する．

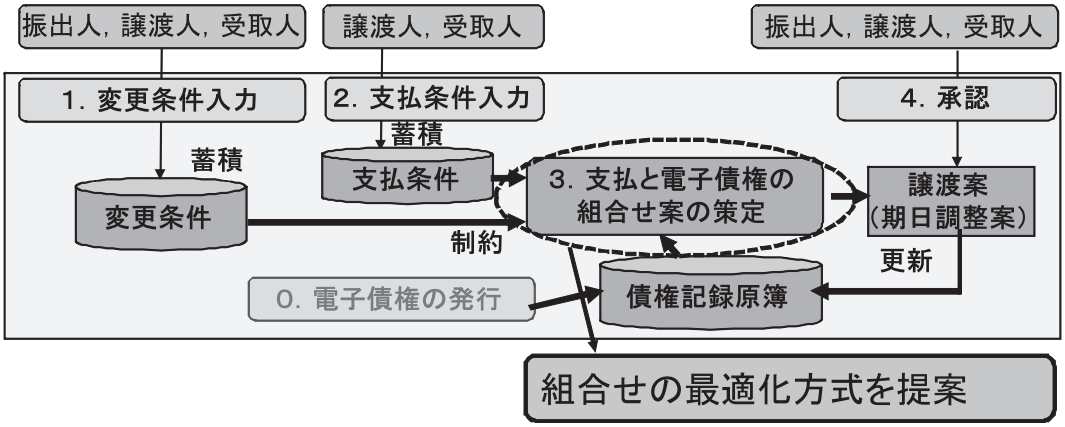


図 4.2: 期日変更マッチングのプロセス

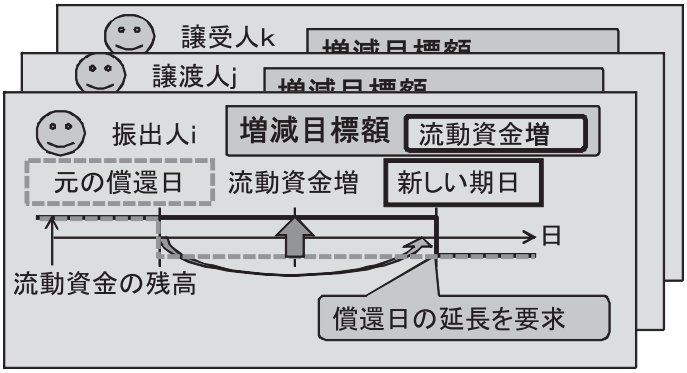


図 4.3: 流動資金の増減目標により償還日の制御を行う例

4.3 電子債権と支払の組合せ問題の定式化

4.3.1 定式化

電子債権と支払の組合せ問題を以下の記法を用いて定式化する．最初に本組み合わせ問題の定数ならびに変数について定義する．

定数

R_{ijm} : 振出人 i が発行し，譲渡人 (現在の所持人) j が所持する m 番目の電子債権

$D(R_{ijm})$: 電子債権 R_{ijm} の償還日

$DE(R_{ijm})$: 振出人 i が設定する電子債権 R_{ijm} の償還日の延長上限

$DS(R_{ijm})$: 振出人 i が設定する電子債権 R_{ijm} の償還日の短縮上限

$V(R_{ijm})$: 電子債権 R_{ijm} の償還額

P_{jkn} : 譲渡人 j による受取人 k に対する n 番目の支払

$D(P_{jkn})$: 支払 P_{jkn} の支払日

$DE(P_{jkn})$: 受取人 k が設定する支払 P_{jkn} の支払日の延長上限

$DS(P_{jkn})$: 受取人 k が設定する支払 P_{jkn} の支払日の短縮上限

$V(P_{jkn})$: 支払 P_{jkn} の支払額

$IP(i, R_{ijm}, f)/\ IR(i, R_{ijm}, f)$:

電子債権 R_{ijm} の償還日を f 日へ延長/短縮する場合において振出人 i が支払う/受け取る利息の上限/下限

$IP(j, R_{ijm}, f)/\ IR(j, R_{ijm}, f)$:

電子債権 R_{ijm} の償還日を f 日へ短縮/延長する場合において譲渡人 j が支払う/受け取る利息の上限/下限

$IP(j, P_{jkn}, f)/\ IR(j, P_{jkn}, f)$:

支払 P_{jkn} の支払日を f 日へ短縮/延長する場合において譲渡人 j が支払う/受け取る利息の上限/下限

$IP(k, P_{jkn}, f)/\ IR(k, P_{jkn}, f)$:

支払 P_{jkn} の支払日を f 日へ延長/短縮する場合において受取人 k が支払う/受け取る利息の上限

C_{xe} : 参加者 x の e 日の流動資金の増減目標額

変数

$v(R_{ijm}, P_{jkn}, f)$: 新しい期日 f で，電子債権 R_{ijm} を支払 P_{jkn} に割り当てた譲渡金額

目的関数

本問題の目的は，電子債権ならびに支払の期日変更による流動資金の増減を全参加者の増減目標に極力近づけることである．本目標を達成するには，下記制約の範囲内で，より長い期日の差異を持つ電子債権と支払の組合せを，より多くの譲渡金額でマッチングさせることが望ましい．従って目的関数は，償還期日と支払期日との差異に譲渡金額を乗じた値の合計値で表され，式 4.1 で表現できる．

$$Max(\Sigma_{ijkmnf}|D(R_{ijm}) - D(P_{jkn})| \cdot v(R_{ijm}, P_{jkn}, f)) \tag{4.1}$$

制約式

制約式は，以下の三種類からなる．

1. 譲渡金額の取りうる範囲を規定する制約 (式 4.2〜式 4.4)
2. 期日変更を伴う譲渡により生じる流動資金の増減を増減目標内に抑える制約 (式 4.5〜式 4.6)
3. 組み合わせ可能な電子債権と支払を規定する制約 (式 4.7〜式 4.16)

$$v(R_{ijm}, P_{jkn}, f) \geq 0 \tag{4.2}$$

$$V(R_{ijm}) \geq \Sigma_{knf} v(R_{ijm}, P_{jkn}, f) \tag{4.3}$$

$$V(P_{jkn}) \geq \Sigma_{imf} v(R_{ijm}, P_{jkn}, f) \tag{4.4}$$

$$\begin{aligned} C_{xe} \;\; &\geq \;\; \Sigma_{jkmnf} v(R_{xjm}, P_{jkn}, f) + \Sigma_{ijmnf} v(R_{ijm}, P_{jxn}, f) \\ &where \;\; C_{xe} \geq 0 \;\&\&\; D(R_{xjm}) \leq e < f \;\&\&\; f < e \leq D(P_{jxn}) \end{aligned} \tag{4.5}$$

$$\begin{aligned} C_{xe} \;\; &\leq \;\; -\Sigma_{jkmnf} v(R_{xjm}, P_{jkn}, f) - \Sigma_{ijmnf} v(R_{ijm}, P_{jxn}, f) \\ &where \;\; C_{xe} < 0 \;\&\&\; f < e \leq D(R_{xjm}) \;\&\&\; D(P_{jxn}) \leq e < f \end{aligned} \tag{4.6}$$

$$v(R_{ijm}, P_{jkn}, f) \;\; = \;\; 0 \tag{4.7}$$

$$where \;\; (f \leq D(R_{ijm}) \;||\; D(P_{jkm}) \leq f) \;\&\&\; D(R_{ijm}) < D(P_{jkm}) \tag{4.8}$$

$$where \;\; (f \leq D(P_{jkm}) \;||\; D(R_{ijm}) \leq f) \;\&\&\; D(P_{jkm}) < D(R_{xjm}) \tag{4.9}$$

$$\begin{aligned} &where \;\; f < DS(R_{ijm}) \;||\; f > DE(R_{ijm}) \;|| \\ &\qquad f < DS(P_{jkn}) \;||\; f > DE(P_{jkn}) \end{aligned} \tag{4.10}$$

$$\begin{aligned} &where \;\; (IR(j, R_{ijm}, f) < IP(i, R_{ijm}, f) \;|| \\ &\qquad IR(j, P_{jkn}, f) < IP(k, P_{jkn}, f)) \;\&\&\; D(R_{ijm}) < D(P_{jkm}) \end{aligned} \tag{4.11}$$

$$\begin{aligned} &where \;\; (IR(i, R_{ijm}, f) < IP(j, R_{ijm}, f) \;|| \\ &\qquad IR(k, P_{jkn}, f) < IP(j, P_{jkn}, f)) \;\&\&\; D(P_{jkm}) < D(R_{xjm}) \end{aligned} \tag{4.12}$$

$$where \;\; C_{ie} > 0 \;\&\&\; f < e \leq D(R_{ijm}) \tag{4.13}$$

$$where \;\; C_{ie} < 0 \;\&\&\; D(R_{ijm}) \leq e < f \tag{4.14}$$

$$where \;\; C_{ke} > 0 \;\&\&\; D(P_{jkn}) \leq e < f \tag{4.15}$$

$$where \;\; C_{ke} < 0 \;\&\&\; f < e \leq D(R_{jkn}) \tag{4.16}$$

制約式 4.2 は，譲渡金額が正であるという条件である．

制約式 4.3 は，電子債権を分割して譲渡した譲渡額の合計は，元の電子債権の金額以下という条件である (電子債権を分割しても譲渡に充当しない分もあるのでイコールとは限らない)．

制約式 4.4 は，電子債権の譲渡によりなされる支払の合計は，支払の総額以下であるという条件である (電子債権の譲渡のみで支払額を充当しきれない場合は新規発行と組み合わされるのでイコールとは限らない)．

制約式 4.5 は，参加者 x の e 日の流動資金増減目標が増加の場合，e 日を跨いで延長となる参加者 x が振り出した電子債権の合計金額と，e 日を跨いで短縮となる参加者 x が受取人となる支払の合計金額の和が，参加者 x の e 日の増加目標よりも小さいことを示す (自らが振り出した電子債権の償還期日の延長あるいは自らが受け取る支払の支払期日の短縮により参加者の流動資金は増加する)．

制約式 4.6 は，制約式 4.5 とは逆の場合であり参加者 x の e 日の流動資金増減目標が減少の場合，e 日を跨いで短縮となる参加者 x が振り出した電子債権の合計金額と，e 日を跨いで延長となる参加者 x が受取人となる支払の合計金額のマイナスを乗じた和が，参加者 x の e 日の減少目標よりも大きいことを示す (自らが振り出した電子債権の償還期日の短縮あるいは自らが受け取る支払の支払期日の延長により参加者の流動資金は減少する)．

制約式 4.7〜4.15 は，電子債権と支払の組み合わせ可能な範囲を定義する．具体的には，逆に組み合わせ可能でない範囲を，譲渡額が 0 として式 4.7 で定義しその適用範囲を式 4.8 〜式 4.16 により規定する．

範囲 4.8 ならびに範囲 4.9 は，変更後の新期日が電子債権の償還日と支払の支払期日の間以外では，電子債権と支払とのマッチングができないことを規定する．範囲 4.10 は，変更後の新期日が電子債権の償還日ならびに支払の支払期日の変更可能範囲以外では，電子債権と支払のマッチングができないことを規定する．

範囲 4.11 は，電子債権の償還期日が支払の支払期日より前の場合には，振出人が電子債権の償還期日の延長に伴い支払う金利が，譲渡人が前記延長に

伴い要求する金利よりも低く，かつ譲受人が償還期日の短縮に伴い支払う金利が，譲渡人が前記短縮に伴い要求する金利よりも低くなければ電子債権と支払のマッチングができないことを規定する．

範囲 4.12 は電子債権の償還期日が支払の支払期日より後の場合であり，範囲 4.11 の場合と債務者，譲渡人，受取人の金利の支払と受払の立場が逆となる．

範囲 4.13 は，振出人の e 日の流動資金増減目標が増額の場合に，振出人の発行した電子債権で e 日を跨いで償還期日短縮するような償還期日の変更 (流動資金を減額させる変更) を伴うマッチングができないことを規定する．これはいずれか一方の組合せのみが約定すると目標を超過する可能性があるため，本定式化では，増減目標の超過を許さないことを前提に本制約を加えている．

範囲 4.14 は，振出人の流動式増減目標が減額の場合で範囲 4.13 とは逆の場合である．

範囲 4.15 範囲 4.16 は，受取人 k における支払期日の変更の制限で，振出人が償還期日に対して制限する範囲 4.13 範囲 4.14 と同様である．以上目的関数，制約式の定義内容は，まず目的関数 4.1 は，$v(R_{ijm}, P_{jkn}, f)$ を変数とする一次関数の最大化であり，またその値は整数値を取る．また制約式もいずれも $v(R_{ijm}, P_{jkn}, f)$ を変数とする一次不等式である．このように本課題は整数線形計画問題に分類できる．

4.3.2 定式化の例

具体的にどのような制約式ならびに目的関数を生成するかについて，図 4.4 の簡単な例を用いて示す．図 4.4 の上段左の表は存在する電子債権を示し，また図 4.4 の上段右の表は登録されている支払を示すものとする．ここで例えば，参加者 C は，電子債権 R_{AC}, R_{BC} を所持し，支払 P_{CF} の支払義務を負うものとする．さらに電子債権 R_{AC}, R_{BC} の償還期日 (0 日) は，支払 P_{CF} の支払期日 (2 日) と異なるため，期日変更を行わないとマッチングができない．図 4.4 の中段，下段は，振出人 A, B ならびに譲受人 F が設定した日々の流動資金の増減目標である．ここでは，いずれも増加 (正) を指定している．これは振出人 A, B は，償還期日の延長を希望しており，譲受人 F は，償還期

日の短縮を希望していることを意味する．なお本例においては単純化のため，期日の変更可能範囲は設定しておらず，また金利の制限もなく，流動資金の増減目標の範囲内であればマッチングが可能であるものとする．また同一参加者の組合せ間で発生する電子債権ならびに支払は複数存在しないものとし，それらを区別する識別子 (m,n に相当) も省略する．

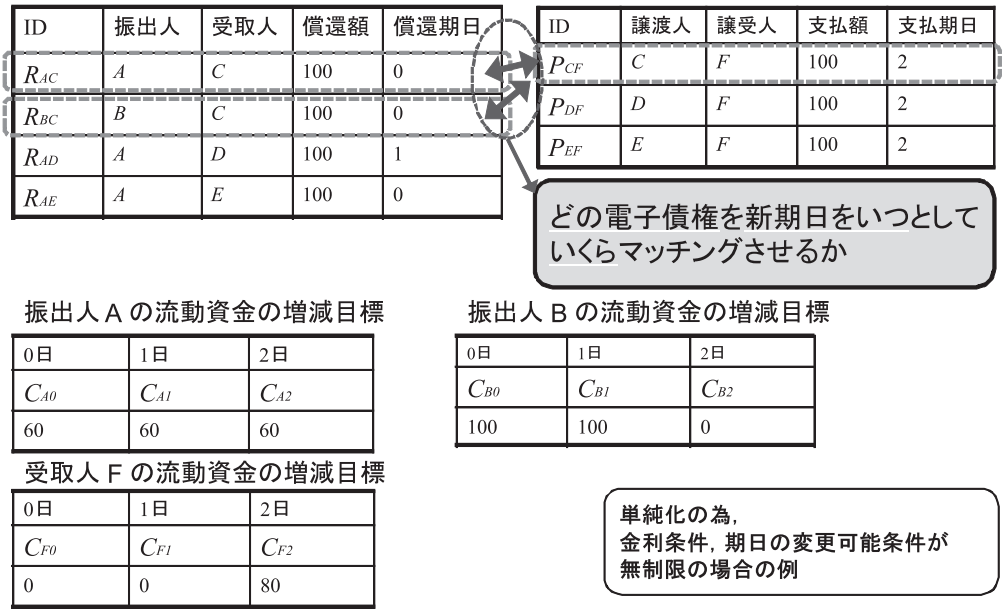


図 4.4: 単純なマッチングの例

本例に対する目的関数は式 4.17 で表わされる．例えば，電子債権 R_{AC} と支払 P_{CF} の場合 (式 4.17 の第 1 列目) を例に取れば，電子債権 R_{AC} の償還日と支払 P_{CF} の支払期日の差 (2 日) に，それぞれの新期日での譲渡額 $v(R_{AC}, P_{CF}, f)\{0 \leq f \leq 2\}$ を乗じた値を加算している．

$$
\begin{aligned}
&MAX(\\
&\quad 2 \cdot (v(R_{AC}, P_{CF}, 0) + v(R_{AC}, P_{CF}, 1) + v(R_{AC}, P_{CF}, 2)) + \\
&\quad 2 \cdot (v(R_{BC}, P_{CF}, 0) + v(R_{BC}, P_{CF}, 1) + v(R_{BC}, P_{CF}, 2)) + \\
&\quad 1 \cdot (v(R_{AD}, P_{DF}, 1) + v(R_{AD}, P_{DF}, 2)) + 2 \cdot (v(R_{AE}, P_{EF}, 0) + \\
&\qquad v(R_{AE}, P_{EF}, 1) + v(R_{AE}, P_{EF}, 2))) \qquad (4.17)
\end{aligned}
$$

一方，図 4.5 は制約式を，ネットワーク図を用いて表現したものである．

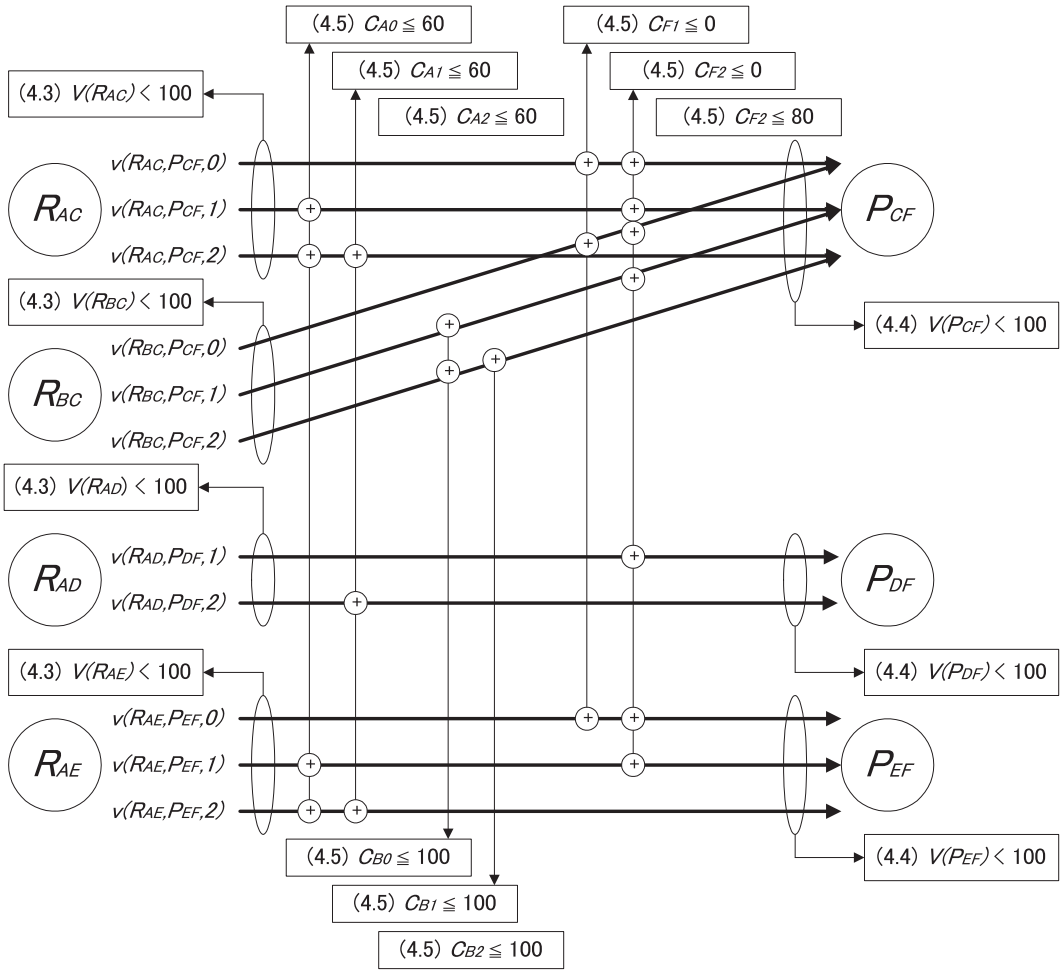


図 4.5: 制約条件のネットワーク図

それぞれの電子債権 R_{ij} から譲渡対象となる支払 P_{jk} の組み合わせに対して選択可能な新期日 f 分の譲渡の線が引かれる．例えば 電子債権 R_{AC} を支払 P_{CF} に充当する場合では，新期日として 0-2 日の選択子があり 3 本の線が引かれる．これに対する制約は，電子債権の合計金額，支払の合計金額，参加者の増減目標の三種類の制約が生じる．図中の括弧付の数字はそれぞれ対応する制約式の番号を示すものとする．電子債権の合計金額の制約は，個々の電子債権 R_{ij} から発生する譲渡の合計が電子債権の額以下となる制約で制約式 4.3 に相当する制約である．支払の合計金額の制約は，同様に個々の支払に充当される譲渡の合計が支払の額以下となる制約で制約式 4.4 に相当する．参加者の増減目標の制約は，参加者の流動資金の増減に影響を与える譲渡の合計が参加

者の設定する増減目標以下になるように，それぞれの日に対して制御するものであり制約式 4.5 に相当する．

以上の目的関数ならびに制約式により最適化を行うと表 4.1 に示す最適解が求められる．表 4.1 は，目的関数の各項の値とそれにより求まる評価値を示すものである．重み (Weight) はそれぞれの譲渡における支払期日と償還期日の差であり，値 (Value) は譲渡金額である．これらから評価値 (Obj. Value) を算出する．

表 4.1: 最適解の算出結果

	$v(R_{AC},P_{CF},f)$			$v(R_{BC},P_{CF},f)$			$v(R_{AD},P_{DF},f)$			$v(R_{AE},P_{EF},f)$			Total
	f=0	f=1	f=2	f=0	f=1	f=2	f=0	f=1	f=2	f=0	f=1	f=2	
Weight	2			2			1			2			
Value	0	0	0	0	20	80	-	0	100	0	60	0	
Obj.Value	0	0	0	0	40	200		0	100	0	120	0	460

4.4 評価

電子債権の期日マッチングの最適化に関して，期日マッチングの効果，最適化の効果，最適化に要する計算時間について，国内企業の財務情報を基づくシミュレーションにより評価する．最初にシミュレーションの前提条件について述べ，次に測定結果について述べる．

4.4.1 前提条件

電子債権の発行に関する統計情報は現時点では存在しない．このため国内の企業の財務情報を元に，電子債権の発行条件を想定してシミュレーションを行う．前提とする財務情報と電子債権の発行条件を表 4.2 に示す．財務情報は，国税庁の平成 15 年度の企業法人統計 [81] ならびに日本銀行の平成 15 年の決済動向 [82] から，国内の企業の平均の売上高，支払債務 (買掛金，支払手形)，

受取債権 (売掛金, 受取手形)，手形交換の平均額，売掛金の平均額を参照する．これらの財務情報から電子債権の発行条件を想定する．具体的には，電子債権の平均償還期間，平均発行頻度，平均金額を想定する．それぞれの算出式を以下に示す.

- 電子債権の平均償還期間 = 受取債権 / 売上高 × 365
- 電子債権の平均発行件数 = (受取手形 / 手形交換の平均額 + 売掛金 / 売掛金の平均額) / 365
- 債権の平均金額 = 受取債権 / (受取手形 / 手形交換の平均額 + 売掛金 / 売掛金の平均額)

表 4.2: 日本企業の平均的な財務情報と電子債権の発行想定条件

売上高 (k¥)	513,131
受取債権 (受取手形＋売掛金) (k¥)	82,887
受̂取手形 (k¥)	16,600
受̂取債権 (k¥)	66,287
手形交換の平均単価 (k¥)	4,000
売掛債権の平均決済単価 (k¥)	2,000
電子債権の平均償還期日 (想定値) (日)	59
電子債権の平均発行件数 (想定値) (件/日)	0.102
電子債権の平均金額 (想定値) (k¥)	2222

シミュレーションでは会社数を 260 社 (実際の企業数の 1/1000) として， 2 年間の取引を繰り返し測定を行う．なお実際の場面では，全ての債権が電子債権に置き換わるとも限らず，またさらに全ての電子債権が譲渡の対象になるとも限らないため，企業の保有する電子債権の保有債権額に対する譲渡可能額の割合をいくつか変更してシミュレーションを行う．シミュレーションの実行環境は MPU: Xeon[1] 2.8 GHz，Memory: 3 GByte, WindowsXP[2]，Java JDK

[1]Xeon は，米 Intel 社の登録商標.
[2]WindowsXP は 米 Microsoft 社の登録商標.

1.6.0_01，Lpsolve[3] 5.5.0.10 を用いて行う．なお，他のパラメータ (電子債権の償還期間のばらつき，電子債権の発行頻度のばらつき，電子債権の金額のばらつき，シミュレーションを行う会社数) についても，いくつか条件を変えて測定を行なったが，その影響は表 4.2 で示したパラメータと比較して小さいため説明を省略する．

4.4.2 測定結果

図 4.6 のグラフは，全支払に対して債権譲渡にて支払を行った割合を示す．

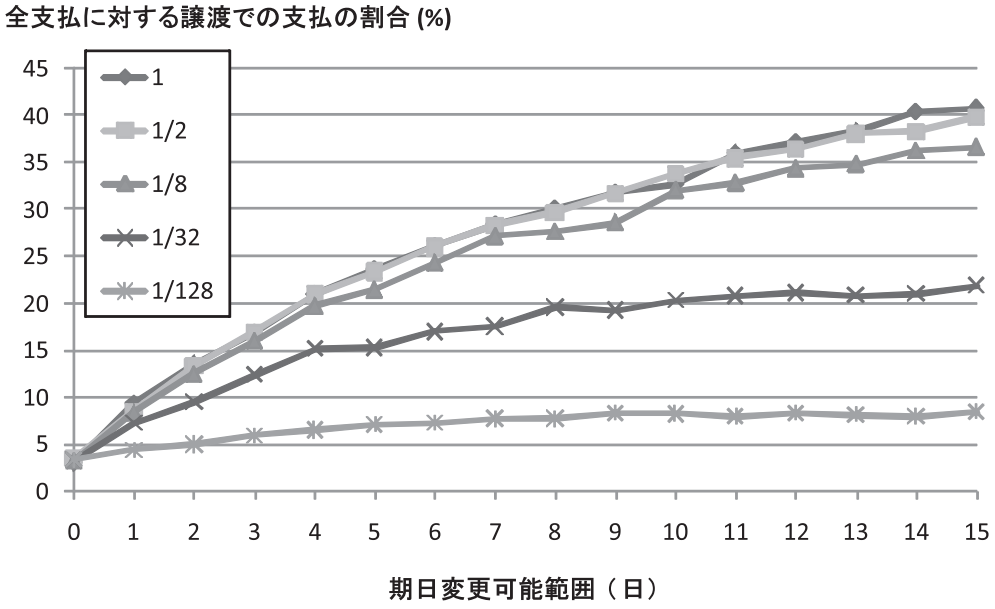


図 4.6: 譲渡による支払の割合

グラフの横軸は，支払期日ならびに償還期日を前後何日間まで変更可能にするかを示す．またグラフの複数の測定結果は，企業が所有する電子債権の内，金額ベースでどの程度を譲渡対象とするか条件を変えて測定したものである．なお本評価は，電子債権と支払とのマッチングを最適化した場合の結果である．

本結果において，変更可能日が 0 の場合が，電子債権の償還期日の変更できない場合である．この場合，電子債権の譲渡で充当できる支払の割合は，全

[3]Lpsolve は，Samuel E. Buttrey 氏の開発した線形計画パッケージ．

体の約 3.3 % である．これに対して，例えば，償還期日/支払期日をそれぞれ前後 3 日変更可能と設定した場合，譲渡での支払の割合は，保有する全電子債権を譲渡対象とした場合で 16.7 % (約 5 倍) に上昇する．また譲渡対象とする電子債権の割合を保有する全電子債権の 1/32 程度に減らした場合でも，12.3 % (約 3.7 倍) の上昇が見られる．以上のことから各企業が償還期日ならびに支払期日に関して数日程度の調整枠を提供することで，譲渡による支払の割合が大きく改善できることがわかる．

次に最適化の効果について示す．図 4.7 は，いくつかの電子債権と支払のマッチング方法ごとに参加者が設定する流動資金の増減目標をどの程度達成できるかを示すものである．具体的なマッチング方法として，以下の三通りの方法について測定を行った．

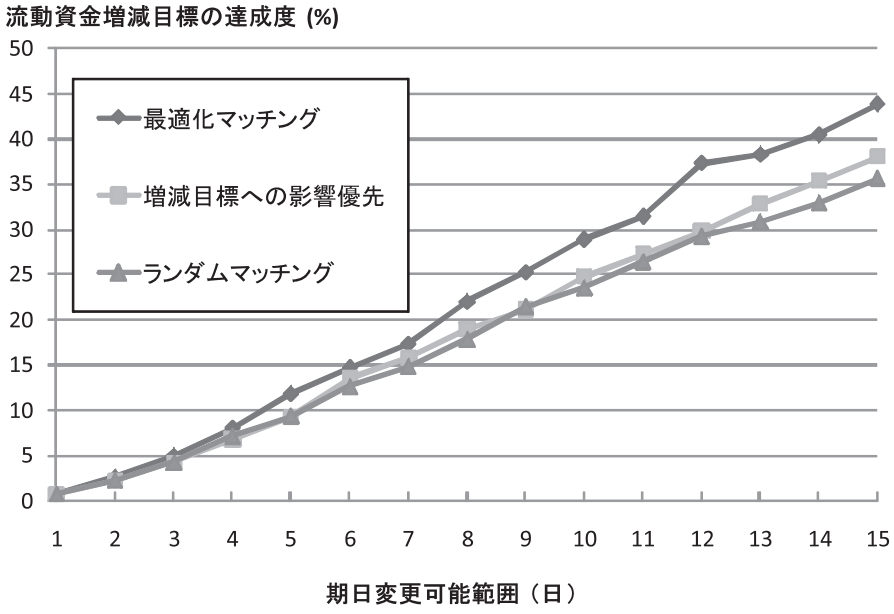


図 4.7: 流動資金の増減目標の達成度

- ランダムな順序でのマッチング

  電子債権と支払の組み合わせ可能な候補の中から，ランダムな順序でマッチングを行う場合．

- 目標達成度への影響が高い順でのマッチング

電子債権と支払の組み合わせ可能な候補の中から，増減目標の達成に影響が高い順にマッチングを行う場合．

- 最適パターン検索に基づくマッチング

  全体として増減目標により近づく最適パターンを検索しマッチングを行う場合．

図 4.7 のグラフの横軸は，支払期日ならびに償還期日を前後何日間変更可能にするかを示している．グラフの縦軸は，企業が予め設定した増減目標をどの程度達成できたかを示すものである．なお，企業の増減目標は企業の保有する電子債権の額の範囲でランダムに設定した．図 4.8 は，最適化の改善率を，対増減目標への影響優先でマッチング，ならびに対ランダムでマッチングと比較したものである．

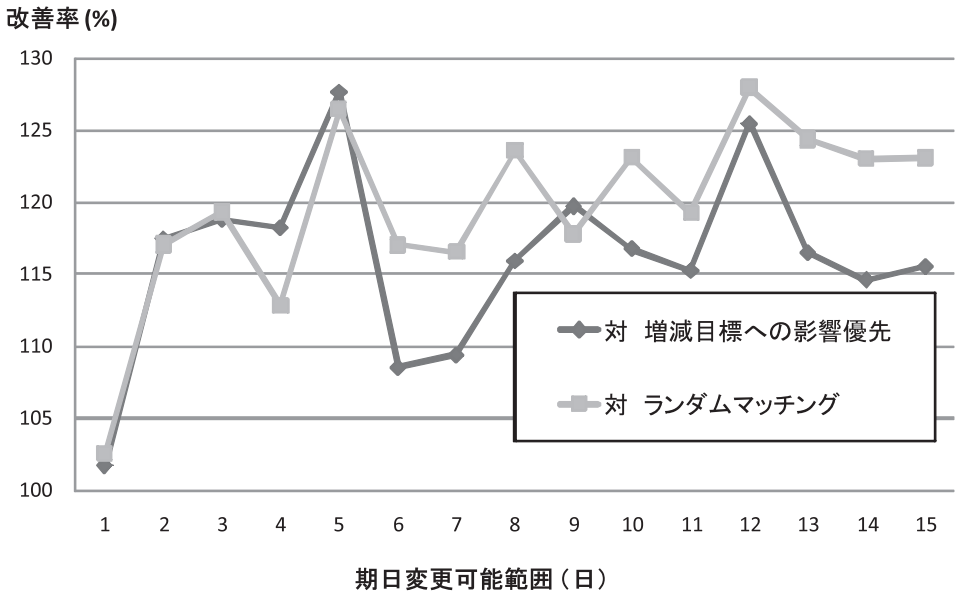


図 4.8: 最適化による改善割合

その結果，変更可能日が少ない場合は多少ばらつきが多いものの，対増減目標への影響優先に対しては平均で約 18.1 %，対ランダムに対しては約 19.6 % の改善が見られる．なお最適化の改善率にばらつきが生じるのは，表 4.2 に示すように，電子債権の平均発行金額が約 222 万円と大きく，この個々の電子

債権の振出や譲渡が変更可能範囲内で発生するか否かが，改善率に影響するものと考えられる．

次に電子債権と支払の組合せの最適化の計算時間について 図 4.9 を用いて検証する．図 4.9 は，横軸が会社数であり，縦軸を一日あたりの最適化の計算に要した時間を示すグラフである．その結果，260 社を対象としたシミュレーションでは最適化に要する計算時間は，1 日あたり平均 113 秒であった．実際の国内の企業数は，シミュレーションの 1000 倍の 約 26 万社であり，計算時間が同じペースで増加すると最適化に要する時間は約 1500 時間となる．仮に一日に電子債権システムが 8 時間稼働し，残りの 16 時間で最適を行うと想定すると，処理可能な会社数は 6000 社程度となる．実用化に向けては，同時に最適化を行う会社の範囲を分割し，処理可能な会社数に抑え込む必要がある．この最適化範囲の分割については，今後の課題であり 4.5 節で述べる．

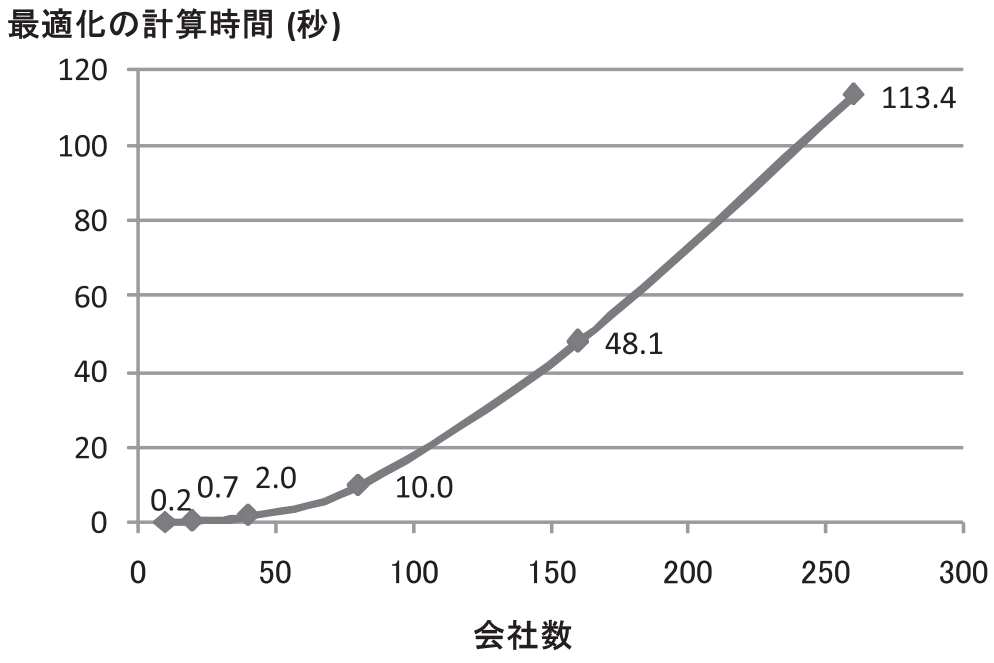


図 4.9: 最適化に要する計算時間

最後に期日のマッチングによる電子債権がどの程度分割されるかについて表 4.3 に示す．分割数は，オリジナルの電子債権が償還時点において平均でいくつの電子債権に分割されたかを示す値である．分割数の平均は，ランダムな順

序でのマッチングの場合で 2.08 件，目的達成度への影響が高い順でのマッチングの場合で 1.38 件，最適パターン検索に基づくマッチングの場合で 2.06 件であった．目的達成度への影響度が高い順でのマッチングの分割数が少ないのは，本アルゴリズムでは，金額が大きくかつ期日の差が大きい電子債権と支払の組を優先的にマッチングさせており，金額が大きいという条件が電子債権の細分化を抑制しているものと考えられる．

表 4.3: 電子債権の分割数

	平均	標準偏差	最大
ランダムマッチング	2.09	1.05	10
増減目標への影響優先	1.38	0.57	7
最適化マッチング	2.06	1.93	9

4.5 結言

本章では，債権譲渡を促進する電子債権の償還期日と支払期日のマッチングの最適化方式を提案した．具体的には各企業が指定する流動資金の増減目標，期日の変更範囲，金利条件をもとで，期日の変更により生じる各企業の流動資金の増減を増減目標に極力近づける最適化方式を示した．さらに最適化方式を定式化し，それが整数線形計画問題に分類できることを示した．また日本の平均的な企業の財務状況を元にシミュレーションを行い，期日変更による譲渡の促進効果ならびに最適化による増減目標の達成度の改善効果を示した．

今後の課題として 4 つの課題が残る．

第 1 の課題は，最適化により電子債権が細切れに分割されてしまう問題がある．本研究の方式では，参加者の資金の増減目標が向上するのであれば，電子債権を任意に分割し，いくつかの支払にいくつかの新期日で割り当ててしまう．この電子債権の細分化は管理コストの増大につながることから，実際運用においては，電子債権の分割をどのように制御するか検討する必要がある．

第 2 の課題は，最適化を行う会社の範囲の分割の問題である．本研究では，

全社の条件を一つの目的関数で表し最適化を行っているが，これは対象とする企業規模を増やす上で非効率的である．実際には，企業間の取引関係には局所性があると考えられ，適切な範囲で企業群を分割できれば，変数，制約式とも少ない複数の最適化問題に分割できる．これにより計算コストの低減ならびに並列処理化による計算時間の短縮が期待できる．

第 3 の課題は，本アルゴリズムにより示される全体最適なマッチング結果は，必ずしも個々の企業にとっての最適なマッチング結果とは限らないことへの対応である．このため実際の運用では，マッチング結果を企業が拒否した場合の代替案をどのように提示するかも重要な課題の一つである．

第 4 の課題は，期日以外のマッチングに関するものである．現在，企業マネーが発行するポイントやマイルが，相互の交換可能になりつつある [83]．その結果，これらが通貨 (企業通貨) の代替として機能するようになってきた．これら企業通貨にはそれぞれ利用に関する制限 (利用店舗，有効期限) があり，今後企業通貨の利用促進に向けては，利用者の許容する制限を加味した上で交換・譲渡するといった利用シーンも考えられる．これを実現する上で，本研究で提案した期日マッチング機能を拡張し，他の制約のマッチングに活用していくという研究の方向性が考えられる．

第5章

結論

5.1 本研究のまとめ

本研究では，電子商取引システムの，ユーザインタフェース，決済機能，バックアップ機能を構築する上で，その柔軟性を高める実現方式について述べた．

第 1 章では，顧客ニーズの細分化に対応する柔軟な電子商取引システムの実現に向け，実写画像を活用した視点自由度の高いユーザインタフェースの実現，類似サービスを活用したバックアップ方式の実現，期日マッチングによる電子債権の利用機会の促進が課題になることを述べ，その研究方針を示した．

第 2 章で，実写画像による視点変更を可能にするモーフィング技術に関して，合成画像のぼやけを軽減する画像対応付け方式とインタラクティブな視点変更を実現する高速レンダリング方式について述べた．画像のエッジ部分に沿った対応付けを可能にする画像対応付け方式を実現することで，合成画像のぼやけを軽減した． またモーフィングのレンダリングプロセスであるワーピング処理を三角パッチの変形に置き換え，さらに画像のミキシングを α ブレンディングに置き換えるデータ変換を実現することで，3D アクセラレータを用いたモーフィングを可能にし，約 11 倍の高速化を実現した．

第 3 章では，サービスの違いを吸収する変換ウェブサービスを生成する仕組みについて述べ，類似サービスによるバックアップのオーバヘッドを検証した．具体的には変換ウェブサービスのソースコードを，変換ロジックを結合する変換フローの定義により生成する方式を示した．さらに証券取引を具体例と

したサンプルアプリケーションによりオーバヘッドを測定し，特にスループットのオーバヘッドの 99.53 % は SOAP の通信時間によるものであることを確認し，この事実から通信のオーバヘッドの高いウェブサービスにおいては，変換ロジックを事前にコンパイルして 1 変換サービス内に集約する手法が，オーバヘッドを最低限に押さえる上で効果がある見通しを得た．

第 4 章では，電子債権譲渡を促進する電子債権の償還期日と支払期日のマッチングの最適化方式を提案した．具体的には各企業が指定する流動資金の増減目標，期日の変更範囲，金利条件をもとで，期日の変更により生じる各企業の流動資金の増減を増減目標に極力近づける最適化方式を示した．さらにその定式化し，それが整数線形計画問題に分類できることを示した．また日本の平均的な企業の財務状況を元にシミュレーションを行った．その結果，期日変更による譲渡の促進効果は，償還期日/支払期日をそれぞれ前後 3 日変更可能と設定した場合，期日変更機能がない場合と比較して，譲渡による支払の割合が保有する全電子債権を譲渡対象とした場合で約 5 倍上昇することがわかった．またさらにマッチングの最適化により，各参加者の増減目標の達成度が，ランダムにマッチングした場合と比較して約 19.6 % 改善することがわかった．

5.2 今後の研究課題

より柔軟性の高い電子商取引システムを実現に向けて，2 つの課題が考えられる．

1 つは，グローバルにサービスを迅速に展開可能な電子商取引システムの実現である．企業間の国際競争が激化する中で，シェアの獲得に向けて，世界各国へほぼ同時にサービスを提供できることが望ましい．しかし現実的には，各国の法制度，商習慣，言語，顧客ニーズなどが違うことから，電子商取引システムを各国の実情にあったようにカスタマイズする必要がある [84]．このような各国の違いを吸収するモジュールをパッケージ化し，コアとなるサービスに結合することができれば，サービスインまでのコストや時間を削減できると考えられ，この実現方式に関して，今後さらなる検討が必要である．

もう一つは，サービスプロバイダに対する柔軟性の強化である．現在，IT

の利用形態が，システムの所有からシステムの利用に移行しつつある [19]．このような中で，サービスプロバイダは自らシステムを所有するのではなく，システムプロバイダから提供を受けたシステムを活用し顧客にサービスを提供するといった，アウトソーシングの形態が主流になりつつある [18]．このような形態では，サービスプロバイダが顧客のニーズの多様化に答えるのに加えて，システムプロバイダがサービスプロバイダのニーズの多様化に柔軟に答える必要がある．具体的には例えば，これまで議論してきた，実写画像を活用したユーザインタフェースの仕組みや，電子債権などの決済の仕組みや，類似サービスを活用したバックアップの仕組みを，システムプロバイダが，どのようにコンポーネント化してサービスプロバイダに提供すれば，サービスプロバイダがこれらのサービスを自由に組み合わせて独自のサービスが提供できるようになるか検討を行う必要がある．

謝辞

本研究の全過程を通じて，終始懇切丁寧なる御指導と御鞭撻，格別の御配慮を賜わりました大阪大学大学院情報科学研究科マルチメディア工学専攻 薦田憲久教授に深く感謝申し上げます.

本研究をまとめるにあたり，貴重なお時間を割いて頂き，丁寧なる御教示を賜わりました大阪大学大学院情報科学研究科マルチメディア工学専攻 岸野文郎教授，ならびに原隆浩准教授に深く感謝申し上げます.

大学院博士後期課程において，情報工学全般に関して親切なるご指導とご助言を賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 西尾章治郎教授，藤原融教授，ならびに秋吉政徳准教授に深く感謝申し上げます.

本研究の推進に賛同くださり，大阪大学大学院博士後期課程に在学することへの御配慮と援助を賜わりました (株) 日立製作所システム開発研究所 所長 前田章博士に心より感謝申し上げます.

また，大阪大学大学院博士後期課程に在学することへの御配慮と援助を賜わり研究会等の場において数々の貴重な御意見を賜りました（株）日立製作所システム開発研究所 主管研究長 舩橋誠壽博士に深く感謝申し上げます.

本研究の方向性について御指導頂いた（株）日立製作所システム開発研究所前第 1 部 部長 佐々木敏郎博士 (現 ディフェンスシステム事業部主管技師)，第 1 部 部長 浜田成泰氏，uVALUE イノベーション研究部 部長 難波康晴博士，第 1 部 主任研究員 阿部正弘氏に深く感謝致します.

第 2 章のモーフィングによる視点変更における画像対応付け方式と高速レンダリング方式に関して，研究の方向付けならびにご指導を頂いた，(株) 日立製作所基礎研究所 所長 武田晴夫博士，(株) 日立コンサルティング テクニカルディレクタ 加藤誠博士，(株) 日立製作所システム開発研究所 主管研究員 山崎眞見博士に心より感謝致します．また本研究の共同者として多大なるご協

力を頂いた，Linux テクノロジー研究部 研究員 久木和也氏に心から感謝申し上げます．また本研究のコンテンツ制作ならびにモデルとして画像の使用承諾を快く頂いた（株）日立製作所都市開発システムグループ セキュリティ本部 主任技師 西美千子氏に心より感謝致します．

第 3 章の類似サービス活用バックアップのための変換ウェブサービス生成方式に関して，研究のご指導と米国での 1 年間の滞在を全面的にサポートして頂いた AT&T Labs, Inc. Shannon Lab. Richard D. Schlichting 博士, Matti A. Hiltunen 博士に心より感謝致します．また研究のご助言と，AT&T との共同研究の日立側の窓口としてサポートを頂きました（株）日立製作所システム開発研究所 uVALUE イノベーション研究部 主任研究員 豊内順一氏に深く御礼申し上げます．

第 4 章の整数線形計画を用いた電子債権譲渡における期日変更マッチング方式に関しまして，事業的な観点からの内容精査にご協力くださいました（株）日立製作所金融システム事業部 主任技師 竹内国人氏に深く感謝致します．また方式のブラッシュアップのために貴重なお時間を頂き議論を頂いた，(株）日立製作所システム開発研究所 主任研究員 馬場健治氏，研究員 島村敦司氏, 中江達哉博士に心から感謝致します．

最後に，学位取得にあたり，様々な不便がありながらも暖かく励ましてくれた妻 裕子，長女 結奈，長男 結貴に心から感謝致します．

参考文献

[1] A. Winston, D. Stahl, and S. Choi, "*The Economics of Electronic Commerce,*" New Riders Pub (1997).

[2] U.S. Department of Commerce Economics and Statics Administration, "Digital Economy 2003," https://www.esa.doc.gov/2003.cfm (2003).

[3] 経済産業省, "電子商取引市場規模調査," http://www.meti.go.jp/policy/-it_policy/statistics/index.html (2008).

[4] H. Someya, T. Murata, O. Yoshie, and T. Moritsu, "Scenario Setup Support Functions in Scenario-based Simulation," in *Proc. of International Conference on Electrical Engineering (ICEE 2004)*, OE 6-1, Vol. 1, pp. 109–113 (2004).

[5] 新谷幹夫, 杉山和弘, "実写ベースのコンピュータグラフィクス技術," 情報処理, Vol. 41, No. 6, pp. 1–6 (2000).

[6] M. Cohen, M. Levoy, J. Malik, L. McMillan, and E. Chen, "Image-Based Rendering : Redally New or Déjà Vu?," in *Proc. of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*, pp. 468-470 (1997).

[7] S.M. Seitz and C.R. Dyey, "View Morphing," in *Proc. of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'96)*, pp. 21–30 (1996).

[8] M. Levoy and P. Hanrahan, "Light Field Rendering," in *Proc. of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*

(SIGGRAPH'96), pp. 31–42 (1996).

[9] T. Kanade and M. Okutomi, "A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, Issue 9, pp. 920–932 (1994).

[10] M. Okutomi and T. Kande, "A Multiple Baseline Stereo," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 15, Issue 4, pp. 353–363 (1993).

[11] S.E. Chen and L. Williams, "View Interpolation for Image Synthesis," in *Proc. of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'93)*, pp. 279–288 (1993).

[12] K. Hisaki and M. Kato, "Changing Lighting Positions in Photographs without 3D Geometric Modeling," in *Proc. of 4th International Conference on Virtual Systems and Mulimedia (VSMM98)*, Vol. 2, pp. 534–539 (1998).

[13] 中川浩子, 坂尾秀樹, 森津俊之，西美千子 "高品質三次元コンピュータグラフィクス技術の応用 -ディジタルアーカイブへの適用例-," 日立評論, Vol. 80, No. 7, pp. 25–30 (1998).

[14] 加藤誠，久木和也，森津俊之，"多視点・多光源位置撮影画像を用いたイメージベースレンダリング技術," 情報処理学会全国大会講演論文集，Vol. 第 58 回平成 11 年前期, pp. 4-235–4-236 (1999).

[15] OASIS, "WS-Reliability," http://docs.oasis-open.org/wsrm/ws-reliability/v1.1/ (2004).

[16] D. Liang, C. Fang, and C.Lin, "Fault Tolerant Web Service," in *Proc. of the 10th Asia-Pacific Software Engineering Conference (APSEC'03)*, pp. 310–319 (2003).

[17] K. Birman, R. Renesse, and W. Vogels, “Adding High Availability and Autonomic Behavior to Web Services,” in *Proc. of the 26th Annual International Conference on Software Engineering (ICSE 2004)*, pp. 17–26 (2004).

[18] 武直行, 齋藤達也, “IT アウトソーシングの動向と日立グループの取り組み,” 日立評論, Vol. 87, No. 3, pp. 9-14 (2002).

[19] 目次康男, 今井俊之, “SaaS の衝撃,” 日経コンピュータ, 2006.4.17, pp. 41–53 (2006).

[20] A. M. Fairchild, “Possible Distinermediation: What Role for Banks in Electronic Invoicing (EIPP),” in *Proc. of 16th Bled eCommerce Conference eTransformation*, pp. 107–118 (2003).

[21] 徐熙錫, “韓国における電子手形法の制定とその法理 –韓国電子売掛債権制度との比較–,” 金融研究研修センター 17 年度ディスカッションペーパー (Dec. 2005), pp. 1–45, http://www.fsa.go.jp/frtc/seika/discussion/2005/-20051220.pdf (2005).

[22] 蝋山昌一, “金融システムと行政の将来ビジョン,” 財経詳報社 (2003).

[23] T. Nakae, T. Moritsu, K. Baba, and N. Komoda: “A Design and Evaluation of Needs-Based Collateralized Debt Obligation,” *WSEAS Transactions on Information Science and Applications*, Vol. 3, Issue 1, pp. 105–112 (2006).

[24] 中江達哉, 森津俊之, 薦田憲久: “投資条件に合わせた債務担保証券設計方式,” 電気学会論文誌 C (電子・情報・システム部門誌), Vol. 126, No. 8, pp. 1026–1032 (2006).

[25] T. Nakae, K. Baba, T. Moritsu, and N. Komoda: “Fast Calculation Method for Fine-tuned Design of Needs-based Collateralized Debt Obli-

gation (CDO)," *IEEJ Transactions on Electrical and Electronic Engineering*, Vol. 2, Issue 3, pp. 379–386 (2007).

[26] T. Nakae, T. Moritsu, K. Baba, and N. Komoda: "A Method for Designing CDO Conformed to Investment Parameters," in *Proc. of 4th WSEAS International Conference on E-ACTIVITIES (E-ACTIVITIES '05)*, pp. 163–169 (2005).

[27] T. Nakae, K. Baba, T. Moritsu, and N. Komoda: "Speed Up Optimization for CDO Design Method Conformed to Investor Needs," in *Proc. of The Third IASTED International Conference on Financial Engineering and Applications (FEA 2006)*, pp. 36–41 (2006).

[28] 電子債権制度に関する研究会, "電子債権制度に関する研究会 第二次報告 -中小企業の資金調達円滑化に向けて-," 経済産業省, http://www.meti.go.jp/press/20070518005/denshisaiken-houkoku.pdf (2007).

[29] Financial Services Agency of the Japanese Government, "Summary of Discussion on Electronic Receivable Legislation from a Financial System Perspective," FSA News Letter (Oct. 2005), http://www.fsa.go.jp/en/-newsletter (2005).

[30] 池田真朗, "電子登録債権 -中間試案の検討と若干の試論-," 金融法務事情, Vol. 1781, pp. 8–19 (2006).

[31] 森田宏樹, "有価証券のペーパレス化の基礎理論," 日本銀行金融研究所 金融研究 2006.11, http://www.imes.boj.or.jp/japanese/kinyu/2006/kk25-h-1.pdf (2006).

[32] 大垣尚司, "債権流通によるファイナンスと電子登録債権," 金融法務事情, Vol. 1781, pp. 20–28 (2006).

[33] 関俊彦, "金融手形小切手法," 商事法務 (2003).

[34] P. Fua, "Reconstructing Complex Surfaces from Multiple Stereo Views," in *Proc. of IEEE Fifth International Conference on Computer Vision (ICCV'95)*, pp. 1078-1085 (1995).

[35] D. Scharstein and R. Szeliski, "Stereo Matching with NonLinear Diffusion," *International Journal of Computer Vision,* Vol. 28, No. 2, pp. 155-174 (1998).

[36] T. Beier and S. Neely, "Feature-Based Image Metamorphosis," in *Proc. of the 19th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'92)*, pp. 35–42 (1992).

[37] 寺沢幹雄, 小高金次, 佐藤創, 和田重久, 外山武徳, "Web 2.5 D 用電子カタログシステム実現方式," 芸術科学会論文誌, Vol. 1, No. 1, pp. 54–63 (2002).

[38] 四倉達夫, "顔合成G 2006 年度版 概要," 東京大学 ISTC セミナー 2006.10.26, http://www.astem.or.jp/istc/seminar06/06-10-seminar-fsm.pdf (2006).

[39] G. Wolberg, "Image Morphing: a Survey," *The Visual Computer*, Vol. 14, No. 8-9, pp. 360–372 (1998).

[40] S. Morishima, "Modeling of Facial Expression and Emotion for Human Communication System," *Displays*, Vol. 17, Issue 1, pp. 15–25 (1996).

[41] J. Rao and X. Su, "A Survey of Automated Web Service Composition Methods," *Semantic Web Services and Web Process Composition*, Vol. 3387, pp. 43-54 (2005).

[42] A. Ankolekar, "DAML-S: Semantic Markup For Web Services," in *Proc. of the 1st International Semantic Web Conference ISWC2002*, pp.348–363 (2002).

[43] W3C, “OWL Web Ontology Language Overview,” http://www.w3.org/-TR/owl-features/ (2004).

[44] S. Chandrasekaran, S. Madden, and M. Ionescu, “Ninja Paths: An Architecture for Composing Services over Wide Area Networks,” *UC Berkeley, CS262 Class Project Writeup* (2002).

[45] D. Richards, S. Splunter, F. Brazier, and M. Sabou, “Composing Web Services using an Agent Factory,” *Multiagent Systems, Artificial Societies, and Simulated Organizations*, Vol. 13, pp. 229–251 (2004).

[46] K. Fujii and T. Suda, “Dynamic Service Composition using Semantic Information,” in *Proc. of the 2nd International Conference on Service Oriented Computing (ICSOC2004)*, pp. 39–48 (2004).

[47] E. Sirin, J. Hendler, and J. Parsia, “Semi-Automatic Composition of Web Services using Semantic Descriptions,” in *Proc. of Web Services: Modeling, Architecture and Infrastructure Workshop in ICEIS2003*, pp. 17–24 (2003).

[48] F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, and M. Shan, “eFlow: a Platform for Developing and Managing Composite e-Service,” in *Proc. of Academia/Industry Working Conference on Research Challenges*, pp. 341-348 (2000).

[49] S. Chandrasekaran, J. Miller, G. Silver, B. Arpinar, and A. Sheth, “Performance Analysis and Simulation of Web Services,” *The International Journal of Electronic Commerce and Business Media*, Vol. 13, Issue 2, pp. 120-132 (2003).

[50] 島村敦司, 森津俊之, 染谷治志, “非ツリー状に接続された証券管理機関における証券移転の経路長と証券保有木の最小化,” 電気学会論文誌 C (電子・情報・システム部門誌), Vol. 126, No. 4, pp. 506–512 (2006).

[51] T. Moritsu, A. Shimamura, H. Mizuno, H. Someya, and K. Takeuchi, "Protocol of the Book Entry System based on the Arrival Principle," in *Proc. of International Conference on Electrical Engineering (ICEE 2004)*, OE 6-1, Vol. 1, pp. 170–175 (2004).

[52] K. Saito, E. Morino, and J. Murai, "Reduction Over Time to Facilitate Peer-to-Peer Barter Relationships," in *IEICE Transactions on Information and Systems*, Vol. E89-D, No. 1, pp. 181-188 (2006).

[53] 片山謙, "日本が取り組む金融イノベーション：電子記録債権," 野村総合研究所 Financial Information Technology Focus (2008.3), pp. 6–7, http://www.nri.co.jp/opinion/kinyu_itf/2008/pdf/itf20080303.pdf, (2008).

[54] 電子記録債権制度の活用に関する研究会, "電子記録債権制度活用に関する研究会報告書 ～利用者が安心して使えるために～," 経済産業省, http://www.meti.go.jp/press/20080325002/03_houkoku.pdf (2008).

[55] E. Christensen, F. Curbera, G. Meredish, and S. Weerawarana, "Web Services Description Language (WSDL) 1.1," W3C, http://www.w3.org/-TR/wsdl (2001).

[56] T. Moritsu, K. Hisaki, A. Takahashi, and M. Kato, "Noh mask - Application of Image-based Rendering," in *Electronic Art and Animation Catalog of ACM SIGGRAPH'98*, p. 154 (1998).

[57] T. Moritsu and M. Kato, "Disparity Mapping Technique and Fast Rendering Technique for View Morphing," in *Proc. of Sixth Pacific Conference on Computer Graphics and Applications (Pacific Graphics'98)*, pp. 216–217 (1998).

[58] T. Moritsu and M. Kato, "Disparity Mapping Technique and Fast Rendering Technique for Image Morphing," *IEICE Transactions on Information and Systems*, Vol. E83-D, No. 2, pp. 275–282 (2000).

[59] T. Moritsu, M. Hiltunen, R. Schlichting, J. Toyouchi, and Y.Namba, “Using Web Service Transformations to Implement Cooperative Fault Tolerance,” in *Proc. of The 3rd International Service Availability Symposium (ISAS 2006)*, pp. 76–91 (2006).

[60] 森津俊之，豊内順一，難波康晴，リチャードシュリクティング，マティヒルトネン, “変換ウェブサービスの半自動生成技術を用いた類似サービス間フェィルオーバ方式，” 電気学会情報システム研究会，IS-07-1〜8，pp. 11–16 (2007).

[61] 森津俊之，薦田憲久, “電子債権譲渡における期日変更マッチング最適化方式，” 電気学会情報システム研究会，IS-07-25〜31，pp. 25–29 (2007).

[62] 森津俊之，薦田憲久, “整数線形計画を用いた電子債権譲渡における期日変更マッチング最適化方式,” 電気学会論文誌 C (電子・情報・システム部門誌), Vol. 128, No. 4, pp. 569–575 (2008).

[63] T. Moritsu and N. Komoda, “An Optimization Method for Redemption and Due Date Matching in Assignment of Electronic Receivables by Using Integer Linear Programming,” in *Proc. of International Conference on e-Business (ICE-B 2008)*, pp. 349–356 (2008).

[64] ニコン, “分析的画質評価ツール VQ-1000,” http://www.nikon-sys.co.jp/products/pdf/VQ-1000.pdf (1997).

[65] 染谷潤, “液晶ディスプレイの動きぼやけ評価方法と標準化,” 映像情報メディア学会誌, Vol. 60, No. 4, pp 510-515 (2006).

[66] E.N. Mortensen and W.A. Barrett, “Intelligent Scissors for Image Composition,” in *Proc. of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH’95)*, pp. 191-198 (1995).

[67] M. Kass, A. Witkin, and D. Terzopouls, “Snakes: Active Contour Models,” *International Journal of Computer Vision,* Vol. 1, No. 4, pp. 321-331

(1998).

[68] M. Kato and H. Noyama, “Interactive Visual Simulation in a Quasi-Three-Dimensional World based on Structuralization of Images,” in *Proc. of International Conference on Artificial Reality and Tele-Existence/Conference on Virtual Reality Software and Technology (ICAT/VRST'95),* pp. 101–110 (1995).

[69] 山本裕之, 内山晋二, 田村秀行,”3 次元形状モデリングのためのドロネー網生成法,” 電子情報通信学会論文誌 D, Vol. J78-D2, No.5, pp. 745–753 (1995).

[70] H. Tanaka and F. Kishino, “Adaptive Mesh Generation for Surface Reconstruction: Parallel Hierarchical Triangulation without Discontinuities,” in *Proc. of Computer Vision and Pattern Recognition (CVPR'93)*, pp. 88-94 (1993).

[71] J. Sussman and K. Marzullo, “The Bancomat Problem: an Example of Resource Allocation in a Partitionable Asynchronous System,” *Theoretical Computer Science*, Vol. 291, No. 1, pp. 103–131 (2003).

[72] IBM, “Business Process Execution Language for Web Services version 1.1,” http://www.ibm.com/developerworks/library/specification/-ws-bpel/ (2002).

[73] W3C, “XML Path Language (XPath),” http://www.w3.org/TR/xpath (1999).

[74] W3C, “Simple Object Access Protocol (SOAP) 1.1,” http://-www.w3.org/TR/soap/ (2000).

[75] 出雲教郎, “Web 2.0 に向けてのマルチレイヤーアクセラレーション,” 日本ラドウェア株式会社, http://www.radware.co.jp/techinfo/whitepaper/-index.php (2008).

[76] S. McIlraith, T. Son, and H. Zeng, "Semantic Web Services," *Intelligent Systems*, Vol. 16, No. 2, pp. 46-53 (2001).

[77] 中小企業庁, "中小企業白書," http://www.chusho.meti.go.jp/pamflet/-hakusyo/index.html (2007).

[78] 高橋康文, "逐条解説 短期社債法," 金融財務事情研究会 (2003).

[79] K. Aardal, G. Nemhauser, and R. Weismantel, "*Optimization: Handbooks in Operations Research and Management Science*," Elsevier (2005).

[80] A. Schrijver, "*Theory of Linear and Integer Programming*," Wiley-Interscience (1986).

[81] 財務省財務総合政策研究所, "法人企業統計年報," http://www.mof.go.jp/-kankou/hyou07.htm (2003).

[82] 日本銀行 信用機構室, "決済動向," http://www.boj.or.jp/theme/-research/index.htm (2003).

[83] 安藤寛道, "企業通貨におけるポイント・マイレージの現状と将来性," 日本大学大学院総合社会情報研究科紀要, No. 8, pp. 113-124, http://atlantic2.gssc.nihon-u.ac.jp/kiyou/pdf08/8-113-124-yasuoka.pdf (2007).

[84] 経済産業省, "グローバル経済戦略," http://www.meti.go.jp/press/-20060412001/20060412001.html (2006).