論文 / 著書情報
Article / Book Information

題目(和文)	異種統合型情報サービスシステムにおける自律分散アシュアランス技術の研究
Title(English)	
著者(和文)	椎橋章夫
Author(English)	AKIO SHIIBASHI
出典(和文)	学位:博士（工学）, 学位授与機関:東京工業大学, 報告番号:甲第6697号, 授与年月日:2006年12月31日, 学位の種別:課程博士, 審査員:
Citation(English)	Degree:Doctor of Engineering, Conferring organization: Tokyo Institute of Technology, Report number:甲第6697号, Conferred date:2006/12/31, Degree Type:Course doctor, Examiner:
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

東京工業大学　大学院

情報理工学研究科計算工学専攻

博士論文

異種統合型情報サービスシステムにおける
自律分散アシュアランス技術の研究

指導教官　森　欣司

提出者　　椎橋　章夫

目　次

序　論

１　研究の背景及び目的

社会環境は日々変化しており，グローバル化，高度情報社会，企業間の連携と統合，景気の低迷・回復などに見られるように大きな環境の変化を受けており，社会の構造・組織などはこの変化する環境への即応などを迫られ，これに対応して変化している．これらの影響はシステムにも及んでおり，これまでのシステムには信頼性・安全性だけが求められていたものが，今や社会や利用者からの要求の変化に即座に対応し，その責任を果たしつづけることが求められてきている．また，ネットワークの発達に伴い多くのシステムが有機的に接続されるようになっている．このようなシステムでは異種のニーズを持つだけでなく頻繁にニーズが変化しているなど，システムの外的変化及び内的変化の両方に対して対応が求められており，システムの安定稼動が今まで以上に求められている．システムの社会における責任はより一層重くなっていると言える．

一方，このような変化への対応は，システムの大規模化・複雑化をもたらし，システム全体を正確に把握することを困難にしている．これはシステムの修正変更に大きな危険が伴うことを意味する．このような環境下でシステムを安全に変化させる技術が求められている．これは従来の信頼性・安全性などとは異なる概念を必要とし，この概念をアシュアランスと呼んでいる[16, 17]．これを実現するためのアシュアランス技術は，異種のニーズを持ったシステムの共存を許容する「異種性」と，状況変化に柔軟に対応でき常にニーズを満たす性質である「適応性」を併せ持ったシステム技術の総称である．

これらの変化への対応は，いろいろな企業活動の分野でも見られる．例えば，製造業，通信会社などの企業合併にともなう統合システム化，国際会計基準の導入によって連結経営を行うためのシステム統合，結合などがある．これらのシステムを統合，結合，更新をする際，システムの安定稼動を阻害させずに，新しく機能を追加することが必須である．鉄道分野における列車の運行システムを例にとると，従来は運行管理システム，ＣＴＣ（列車集中制御装置），連動装置，旅客案内装置，情報伝達システム等，メインフレームを中心とする階層構造のシステムを構成していた．各システムはニーズの要求レベルも異なり，各々独立したシステムを構成していた．しかし，最近の大規模輸送管理システムはこれらのシステムを小型の汎用サーバを中心に構成

し，ネットワークで接続して統合化する自律分散システムとして構成している．この結果，今までは各システム間で途切れていた情報が一貫して流れ，運行状況の変化に即応して制御がなされ，更に関連のシステム（装置）へも一貫して情報が流れるようになった．このため，運行業務の効率向上，乗客へのサービス向上等，大きなシステム効果を上げている．

このことは鉄道事業全体についても同様のことがいえる．従来は安全で正確な輸送を行えば良かったものが，現在では安全性，快適性，利便性などの多様な質の高いユーザ・サービスを求められている．一方で鉄道輸送を支える重要なシステムの 1 つである自動出改札システム（ＡＦＣ: Automatic Fare Collection system）は，異種システムとの共存，異種ニーズとの連携，円滑なシステム更新などのシステムニーズが存在する．具体的には高密度輸送におけるピーク時の乗降客の「流動性の確保」に対応するための重要な課題の一つとして「自動改札機の処理速度向上」がある．一方で，金券である乗車券を処理するためシステムの信頼性の確保が不可欠である．このため，高速処理と高信頼性の両方を備えた乗車券システムが必要不可欠である．しかし，従来の磁気式自動出改札システムでは駅内は簡易な集中システム構成で各端末は原則オフライン処理しており，保守コストも高く，拡張性も乏しいものである．これらの課題を解決し，更に乗客の利便性向上やメンテナンスコスト低減のために新しい無線通信方式のＩＣカード乗車券システムの開発が必要となった．

この新しい無線通信式ＩＣカード乗車券システムは無線通信により改札機との処理を行うため，改札時に処理を高速化するとデータ処理の信頼性が低下すると言う問題がある．このため，「ＩＣカード」「自動改札機」「センターサーバ」という処理時間の異なるシステムを統合した，「異種統合型自律分散システムアーキテクチャ」を提案する．このシステムアーキテクチャを活用し，高速処理性と高信頼性を実現するアシュアランス技術を併せて提案する．

本論文においては，異種（無線通信と有線通信，処理時間の異なるシステムなど）を統合した情報処理システムにおける高速性と信頼性を実現するためのニーズと課題を明確にし，これを解決するためのシステム構築技術と高速処理性と高信頼性という要件を満たすための２つの自律分散処理技術，さらにはアシュアランス性を評価する技術について研究する．

２　論文の構成

本論文の構成を図 0.1 に示す．本論文において，第１章「異種統合型情報サービスシステムのニーズと課題」では，鉄道乗車券システムやインターネットを使った電子

商取引などのような高負荷トランザクションを処理する情報サービスシステムと設備機器を含む制御システムとの異種統合型情報サービスシステムのアプリケーションニーズとシステムニーズについて述べた．このような異種統合型システムにおいてはシステムの障害，拡張，保守などのシステム状況変動時にも，高速性，高信頼性とリアルタイム性を両立させ，サービスの継続を保証するアシュアランス性が求められることを示す．しかし，従来の集中・分散管理システムでは，上記のようなアシュアランス性には対応できないことを示し,異種統合の視点に立ったシステムアーキテクチャ，アシュアランス性保証のための技術とその評価方法の必要性を明らかにする．

第２章「分散技術動向」では，はじめに，本研究の前提条件となるアーキテクチャについて調査し，従来の分散システムアーキテクチャと技術の動向を明確にする．また，従来システムでは異種ニーズ／異種モード共存下でシステムのアシュアランス性を満たすことが困難であることを述べる．

異種統合システムの構築技術の１つはデータ駆動型アーキテクチャを持つ「自律分散システム技術」であり，2 つめは異種ニーズ，異種モードを共存させ，またその変化に適応していくための「アシュアランス技術」とその動向である．まず自律分散システムでは，データ駆動型アーキテクチャであるデータフィールドアーキテクチャについて明らかにし，さらにオンライン中にシステム構築を行う時に有効なオンライン保守技術，オンライン拡張技術について明確にする．また異種ニーズ，異種モードの共存する環境下でのシステム安定稼動を保証する技術としてのアシュアランス技術について述べる．システム共存に必要な基礎技術であるデータフィールドとシステムの共存方法を，前者については分離型，統合型，共存型に，後者については分離型と統合型にそれぞれ分類いついて述べる，その上で，これらを組み合わせた異種システム共存技術について明らかにする．

第３章「異種統合型情報サービスシステムアーキテクチャとアシュアランス技術」では，異種統合型情報サービスシステムのニーズを解決するためのアーキテクチャ，アシュアランス性を実現するための自律分散整合化技術と自律連携処理技術を提案した．異種統合型情報サービスシステムアーキテクチャでは，異種データの特性に応じてデータフィールドを構成し，それらが目的に応じてゲートウェイを介してデータを交換できるようにしてある．ここで各サブシステムに自律性を持たせ，それらがデータフィールドを介した連携により，システムの部分的障害，拡張，保守においても，稼動の継続性が保証できることを明らかにする．

「自律分散整合化技術」では，リアルタイム性制限の下で高トランザクション処理において生じるデータの欠損を回復するため，データフィールドごとにデータの滞留時間を変化させた時間差異種データフィールドを構成し，それらの連携によるデータ間の論理的整合化を図る技術を提案する．この技術により，トランザクションの発生特

性に応じて整合化達成時間を最小にする，異種データフィールド数と各データフィールド間の時間差を導出した．
「自律連携処理技術」では，各サブシステムが持つ情報の局所的条件下で，サブシステムごとの処理の分散化／サブシステム間の連携と処理時間との関係を明らかにする．この技術により，高速処理を達成するための最適なシステム分割／連携度合をトランザクションの発生特性に応じ求められることを示す．

さらに，このようなアーキテクチャと技術のもとで，システム変動下における稼動の保証度合を評価するためのアシュアランス性評価として，機能信頼性評価技術を提案する．

第４章「鉄道乗車券システムへの適用」においては，上記の「異種統合型情報サービスシステム」を，無線ICカード／有線ネットワーク／ゲート装置など複数の手段を含む制御／トランザクション処理の統合された鉄道乗車券システムに対し，乗客の流動性を妨げることなく，処理の高速性，高信頼性を実現するため，提案したシステムアーキテクチャと2つのアシュアランス技術を適用し，それらの有効性が実証できることを示す．

まず，現行の鉄道乗車券システムの概要と問題点を述べ，次に，ICカードを使った無線と有線の異種システムが統合され，端末，駅サーバ，センターサーバを自律的に制御するという新しい鉄道乗車券システム（異種統合型自律分散乗車券システム）についてその機能と動作を概説する．さらに，このシステムにおける，無線通信方式のIC カードゆえの欠点としての「データ抜け」という重大な問題が発生してもシステムを止めること無く，データの信頼性を確保できるように「自律分散整合化技術」を適用しその有効性を評価する．また，高密度輸送におけるピーク時の乗降客に対応するための重要な課題として「自動改札機の処理速度向上」の問題を明らかにし，その解決技術としての「自律連携処理技術」を適用し，その有効性を評価する．これらの技術を導入した場合と導入しない場合とを比較，評価し，アシュアランス度を高めるためのシステム構築技術の有効性を検証する．

また，システム設計への適用として設計パラメータの最適値を求める評価方法を提案し，これにより，ユーザーにとって最適な状態での設計値（改札機リーダ／ライタの通信エリアの大きさ）について自動改札機において検証する．

第５章｢結論｣は本研究全体を総括し，異種統合型情報サービスシステムでの処理の高速性と高信頼性を実現するためのシステム構築技術と２つの自律分散技術，アシュアランス性評価技術を示す．これらの技術は，今後ますます要求の強まる異種統合型情報サービスシステム，高信頼性システムへ適用が想定され，本技術は今後の更なる発展が期待されるものであることを述べる．

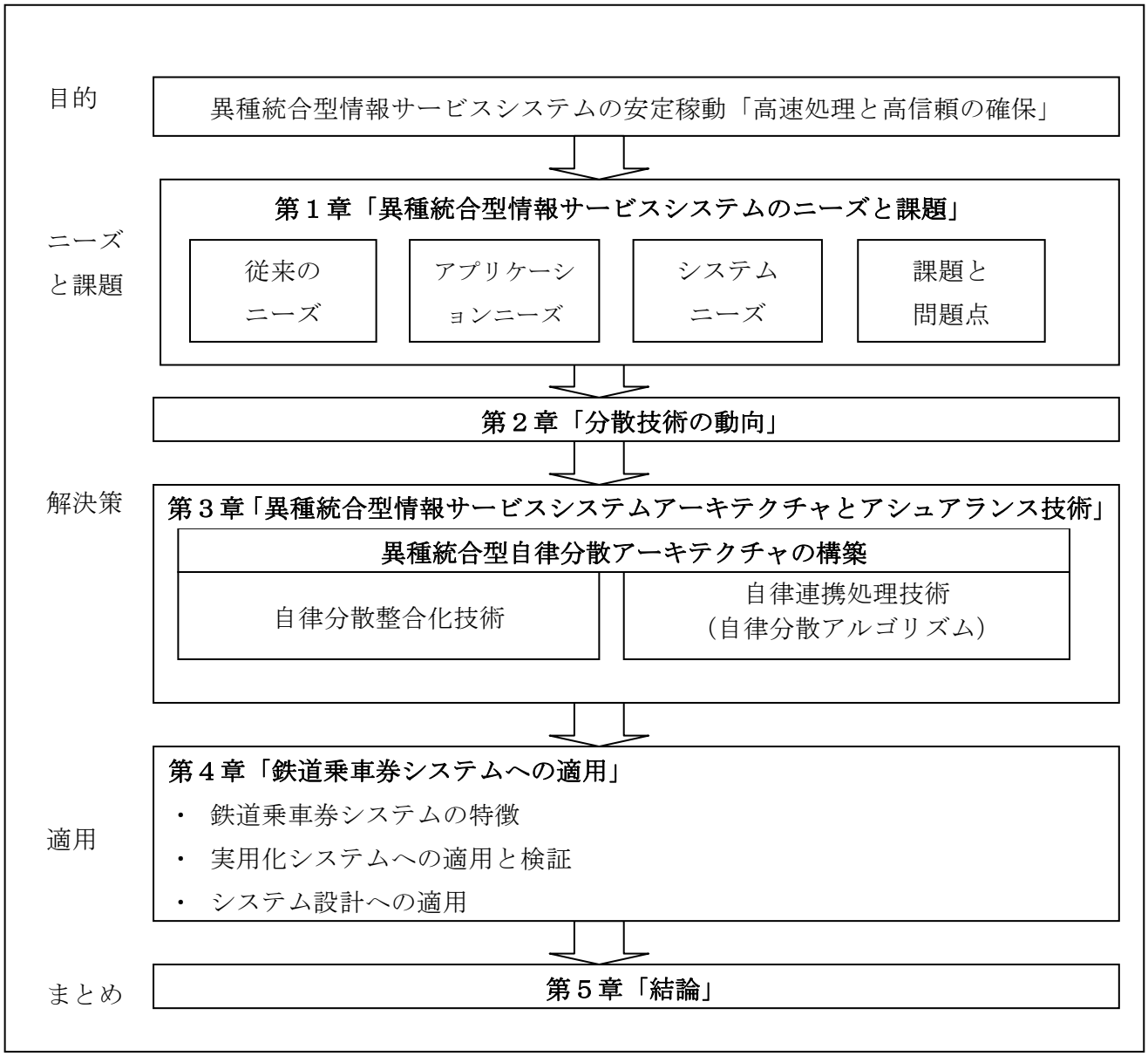

目的
異種統合型情報サービスシステムの安定稼動「高速処理と高信頼の確保」
ニーズ
と課題
第１章「異種統合型情報サービスシステムのニーズと課題」
従来の
ニーズ
アプリケーシ
ョンニーズ
システム
ニーズ
課題と
問題点
第２章「分散技術の動向」
解決策
第３章「異種統合型情報サービスシステムアーキテクチャとアシュアランス技術」
異種統合型自律分散アーキテクチャの構築
自律分散整合化技術
自律連携処理技術
（自律分散アルゴリズム）
適用
第４章「鉄道乗車券システムへの適用」
・ 鉄道乗車券システムの特徴
・ 実用化システムへの適用と検証
・ システム設計への適用
まとめ
第５章「結論」

図 0.1　本論文の構成

第1章

異種統合型情報サービスシステムの

ニーズと課題

1.1　背景

前節の序論でも述べたが，社会環境は日々変化しており，グローバル化，高度情報社会，企業間の連携と統合，景気の低迷・回復などに見られるように大きな環境の変化を受けており，社会の構造・組織などはこの変化する環境への即応などを迫られ，これに対応して変化している．これらの影響はシステムにも及んでおり，これまでのシステムには信頼性・安全性だけが求められていたものが，今や社会や利用者からの要求の変化に即座に対応し，その責任を果たしつづけることが求められてきている．また，ネットワークの発達に伴い多くのシステムが有機的に接続されるようになっている．このようなシステムでは異種のニーズを持つだけでなく頻繁にニーズが変化しているなど，システムの外的変化及び内的変化の両方に対して対応が求められており，システムの安定稼動が今まで以上に求められている．システムの社会における責任はより一層重くなっていると言える．

また，昨今の情報技術の進展，とりわけ半導体の高集積化，高性能ＣＰＵ，ネットワーク，モバイル通信技術等々により，システムをメインフレーム中心の集中型のシステムから，近年は小型汎用機による大規模分散型システムへと変ってきた．このような背景の下で大規模オンラインシステムを取り巻く環境は大きく変化している．特に，最近のリアルタイム制御システムにおいては情報サービス機能が付加されるようになってきている．例えば従来はリアルタイム制御を主体にシステム化してきた鉄道の輸送管理システムシステムにおいて，制御に関連して集めた運行情報を乗客への情報サービスに直接使用する業務を１つのシステム内に共存させようという考え方である．従来は種類の異なるアプリケーションは別のシステムに分けて，実現させることが多かったが，近年は小型汎用機やネットワークの進展と共に，別なシステムを作るよりも，１つのシステムの中で共存させた方が多様なニーズに対して柔軟に応えることができる上，経済的にも有利であるとの判断の下，このような異なる種類のアプリケーションを共存させるシステムが多くなっている．また，ＩＣカード技術も記憶容量の増大，セキュリティの強化，カードの薄型化などが実現し，特に無線通信を用いたＩＣカードはその利便性から鉄道乗車券や電子マネーなどへの利用が拡大している．情報サービスシステムにＩＣカ

ードを利用した例では，多数のユーザのカードと端末との高負荷トランザクション処理においてはリアルタイム処理による高速性と，あわせて高い信頼性が求められている．

このように，高負荷トランザクションを処理する情報サービスシステムと設備機器を含む制御システムを統合したシステムを「異種統合型情報サービスシステム」と呼ぶこととする．

1.2　アプリケーションニーズ

前述のような環境の中で，情報サービスシステムのアプリケーションニーズ（システムによるサービスを提供する側と受ける側の両方のユーザニーズを「アプリケーションニーズ」と呼ぶこととする）について考察する．

鉄道では，以前は安全で正確な運転がされていれば利用者のニーズを満足していたものが，最近では，より便利で快適であることが求められており，例えば，夜間の運転時間の拡大，自社内だけでなく他社の鉄道線路への直通運転，駅の中をスムーズに移動したい，会社が変わるたびに切符を買わなくとも乗り換えしたいとの要望など，さまざまなニーズ・要望が出ている．

金融・流通業界では，昔は窓口での入金･引き出ししか出来なかったものが，昭和 50 年代に実用化された ATM（Automatic Teller Machine）の普及により取引銀行だけでなく他行のカードによっての口座取引が出来るばかりでなく，最近ではインターネットによる取引Ｅ-Comerce（電子商取引）によっていつでもどこでも，これらができるようになってきている．さらに，セキュリティの高いＩＣカードの導入により，電子マネー機能などの決済機能と連動した付加サービスも出来るようにもなってきた．これらは，以前はサービス時間が限定されていたが，社会生活の変化の中で，夜間においても，土日・祝日にもサービスを受けたいという要望から営業時間帯が次第に拡大してきおり，何時でもどこでもどこからでも利用できる環境が求められている．したがって，このような「サービスの継続」というニーズに対応するためにシステムも無停止化せざるを得ない状況となっている．さらに，システムが単に継続して稼動するだけで無く，高負荷トランザクションの時も高速かつ信頼性の高い処理を滞りなく行い，待ち時間を最小化するなど，システムにおける処理の「流動性の確保」も求められている．

このように，情報サービスシステムにおけるアプリケーションニーズは高負荷トランザクション「サービスの継続」と「流動性の確保」が重要となっている．

1.3　システムニーズ

前項のアプリケーションニーズによるシステムニーズについて考察する．

アプリケーションニーズとしての「流動性の確保」をシステムとして実現するには高負荷トランザクションを端末レベルで高速処理するために「設備制御システムによるリアルタイム性」が必要である．また，一方で「情報システムにおけるデータの信頼性確保」も必要である．

もう１つのアプリケーションニーズである「サービスの継続」についてはシステムが故障，拡張，改修などの変動環境下でもシステムは「稼動の継続」が必要であり，また，このような変動に対してシステムが適応することが求められてきている．これらの対応はシステムを新しいシステムに更新することによって対応できるが，新たなシステム構築はコスト増をまねき，既存の稼動しているシステムを段階的に変更していくということも必要となってきている．しかも，これらの対応はシステムを停止して行うのではなくシステムを稼動させサービスを停止しない中で行うことも要求されている．つまり，「システムの適応性」が必要となっている．

このように，異種統合型情報サービスシステムにおけるアプリケーションニーズとシステムニーズをまとめると図 1.1 に示すとおりとなる．

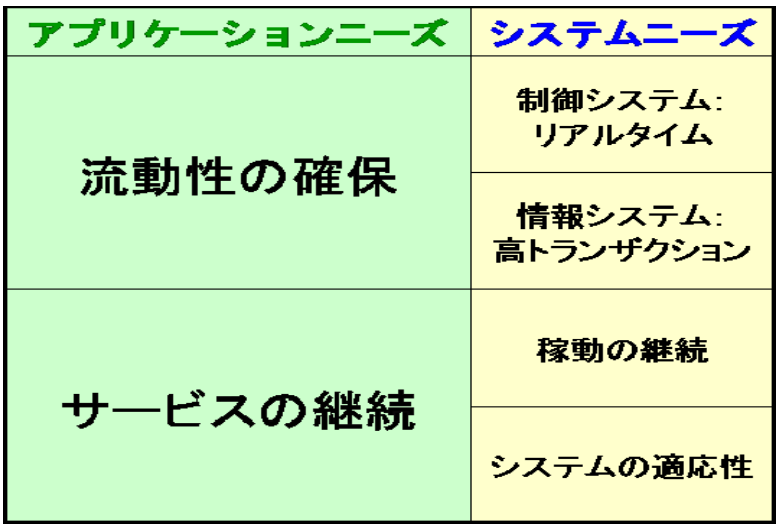


図 1.1　アプリケーションニーズとシステムニーズ

1.4　システム要件

本項では異種統合型情報サービスシステムにおけるシステム要件を従来システムとの違いの視点から明確にする．

まず，日本での代表的な業種のシステムを例にとって前節に述べたニーズを満足するためにシステムに求められる要件について，以下に整理する[21]．

各種企業システム

・企業合併などの環境変化に応じて対応できること（適応性）

・さまざまな職種の企業間での情報交換に対応できること（異種性）

鉄道システム

・運転制御情報と旅客案内情報の混在が出来ること（異種性）

・新線開通，停車駅パターンの変更に対応できること（適応性）

・会社間相互の乗車にシステムが対応できること（異種性，適応性）

製造システム

・異種システムが互いに連携して動くことを保障する（異種性）.

・ユーザニーズに即応できる製造システム（適応性）.

システムの統合・結合･更新を行うに当たって，このように異種性，適応性，が求められる.

1.4.1　　従来システムとの違い

異種統合型情報サービスシステムについて，従来のメインフレーム中心の集中型システムの例と比較して述べる．従来型の代表的な集中型のシステムに銀行システムやクレジット会社のシステムを図 1.2 に示す．システムの中心となる計算機システムは２重化複数台系システムで複数台が常時デュアル運転を行い，一部が待機予備系となっているのが通常である．銀行カードやクレジットカードによりＡＴＭ（Automatic Teller Machine）などの端末からユーザが入出金などの処理を行うが，オンラインで数秒かかり，データ量は数十バイト程度，トランザクションは一日当たり百数十万程度であり，いわゆる階層型集中情報処理システムである．情報処理の視点からシステムが構築されており，ここでは「情報処理システムビュー・アプローチ」と呼ぶ．しかし，このシステムでは高負荷トランザクションを高速処理しかつ，高信頼性が必要な「情報サービスシステム」の対応には限界がある.

これに対して最近の大規模分散システム（例えば東京圏輸送管理システムＡＴＯＳやＩＣカード乗車券システム）は駅の端末や駅サーバ（連動装置など）も含めて全体をネットワークで接続し，有機的に結合する大規模自律分散システムである[20]．これらシステムの特徴は制御系システムと情報系のシステムが統合されたシステムであり，オンラインリアルタイムで処理される制御システムの視点からシステムが構築されており，ここでは制御システムビュー・アプローチと呼ぶ.

図 1.3 に示すように，異種統合型情報サービスシステムは高負荷トランザクションを

処理する情報サービスシステムにおいて「処理の高速性／情報の信頼性」などの「異種のニーズ」,「制御システム／情報システム」などの「異種のシステム」,「有線通信／無線通信」などの「異種の手段」を統合したシステムである.

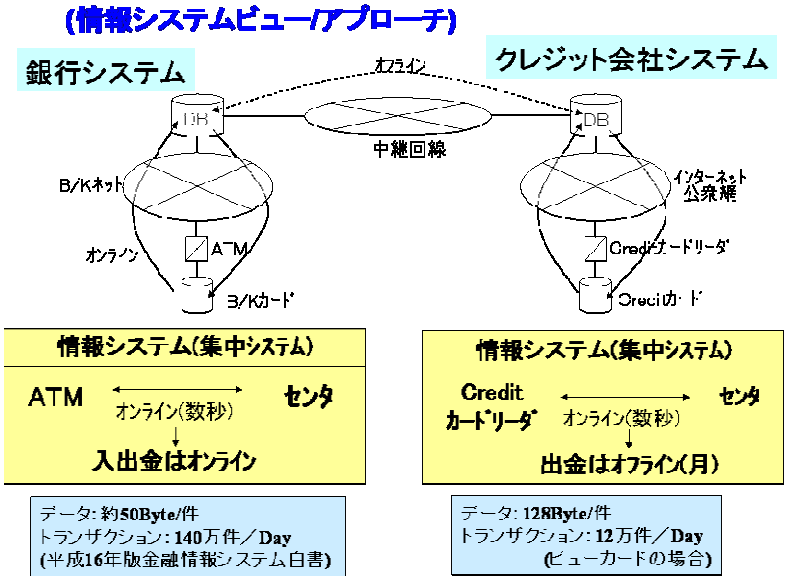


図 1.2　従来情報システム（集中システム）の例

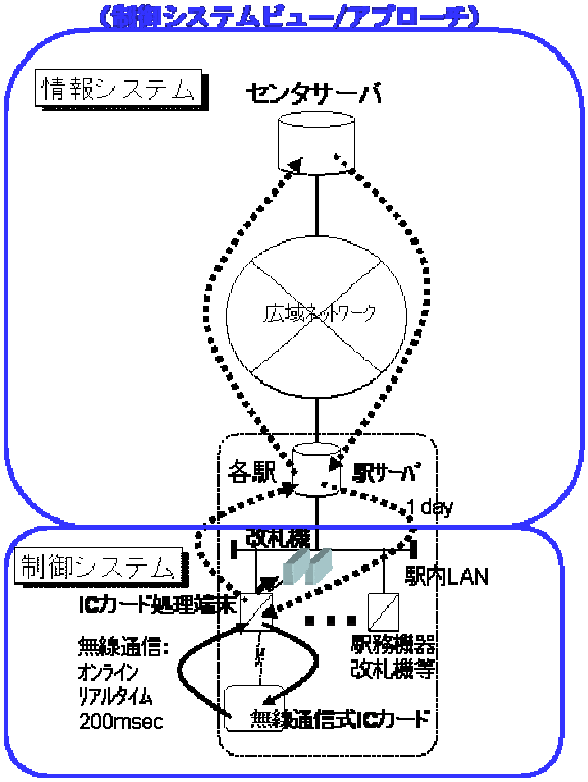


図 1.3　異種統合型情報サービスシステムの例

1.5　異種統合型情報サービスシステムにおける課題

異種統合型情報サービスシステムにおけるニーズは 1.2 項，1.3 項にある通り，システムの安定稼動と高速処理性，高信頼性の確保である．この３つの要素を実現するための課題は以下のとおりである．

(1) システムの安定稼動

有線通信と無線通信などの異種システムが統合されたシステムや制御システムと情報システムが統合されたシステムなどの異種統合型情報サービスシステムの安定稼動を実現するには，システムアーキテクチャと処理技術の２つの問題がある．システムアーキテクチャとしては，このようなことが可能なシステムアーキテクチャにしておかなければならない．また端末レベルでの高速処理を行う技術，システムの信頼性を確保する技術が必要である．この観点では自律分散システムのシステムアーキテクチャが極めて有効である[4,5,6,7,8]．

異種統合の視点に立ったシステムアーキテクチャ構築技術を本論文では研究対象としている．

(2) 高速処理性

従来の集中処理システムでは情報処理システムの視点から論じることが多く，昨今のような高負荷トランザクション大規模システムにおける失敗事例はその多くがトランザクション(処理量)急増時の対応である．このため，高負荷トランザクションに対応するため，端末レベルの処理をオンラインリアルタイムの制御システムとし高速処理する方法がある．最近は駅の自動改札の例のように，さらに個々の端末レベルでの処理を高速化する必要が生じており，課題となっている．高速処理技術として並列処理や分散処理の技術があるが，高負荷トランザクションの条件下で高速処理を可能とする技術の研究はまだ進んでいない．このため，システムのアシュアランス性の観点から，処理を前処理と後処理に分割して高速処理行うなどの自律分散処理の技術とその評価方法を本論文では研究対象としている．

(3) 高信頼性

分散システムでデータ更新処理の信頼性確保の技術として，トランザクションデータが複数のデータベースサーバを更新する場合には全てのデータベースサーバが更新されたか，または全くされなかったかのいずれかであることを保証する技術として２相コミットメント制御[29]がある．しかし，無線通信や異種統合型情報サービスシステムにお

いてはサービスの継続，処理の流動性確保の観点から信頼性を確保する前提で，ある一定の条件下ではデータの不一致な状態を許容する必要がある．このためシステムのアシュアランス性の観点から，本論文ではデータ整合化の技術とその評価方法を研究対象としている．

1.6　研究の方向性

第１章では異種統合型情報サービスシステムのアプリケーションニーズ，システムニーズ，システムの要件と課題などについて述べてきた．この中で，高負荷トランザクションを処理する情報サービスシステムと設備機器を含む制御システムとの異種統合型情報サービスシステムのアプリケーションニーズとシステムニーズについてその関係を明確にした．このような異種統合型システムにおいてはシステムの障害，拡張，保守などのシステム状況変動時にも，高速性，高信頼性とリアルタイム性を両立させ，サービスの継続を保証するアシュアランス性が求められることを示した．しかし，従来の集中・分散管理システムでは，前記のようなアシュアランス性には対応できないことも述べた．

本論文では異種統合の視点に立ったシステムアーキテクチャ構築技術，アシュアランス性保証のための技術とその評価方法を研究し，更に，実システムへ適用し，その有効性を実証することを研究課題とした．

第2章

分散技術の動向

2.1　データ駆動型アーキテクチャを持つ自律分散システム

本節では，本論文で扱う自律分散システムについて述べる．自律分散システムを実現する上での基本となるのがデータ駆動型アーキテクチャである．自律分散システムには，オンライン拡張性，オンライン保守性とフォールトトレランス性の三つの特徴がある[1]．また，異種ニーズ・異種モードの共存環境下でのアシュアランス技術についてシステム構築の観点から調査し，以下にまとめた．

2.1.1　　自律分散システムのコンセプト

(1)　自律分散システムの特徴

システムの大規模化と共に，各システムがネットワークを通して広域に接続されるようになってきている．しかも社会，経済の頻繁な状況変化へ対応し，各システムもそれに適応し，変化に対応して行くことが求められるようになってきている．

このような背景の下，自律分散システムの特徴としてオンラインプロパティがある．具体的には以下の 2 つの特徴をいう．

① オンライン拡張性；

システムの周囲の状況に合わせてシステムを建設，変更，修正を行わなければならないことがある．更にこの変更，追加があっても，他のサブシステムの稼動を止めることは許されないようになってきている．このようなシステムのオンライン稼動中に部分的な変更や段階的な構築ができることをオンライン拡張性と呼ぶ．最近では経済環境が頻繁に変化し，ハードウェアやソフトウェアの技術も急速に変わるようになると，このオンライン拡張性が不可欠となる．

② オンライン保守性；

システムの一部を拡張したり，障害部分を修復したりした時には，必ず確認（テスト）しなければならない．システムを止めず，もちろんこの時にもシステムへの影響を考え，全面的な稼動停止は許されない．従

来はシステムが機能停止する夜間に保守をしていた．しかし夜間の業務までシステム化されてきており，またビジネスのグローバル化，サービス性向上によりシステム稼動時間が長くなり，機能を停止できる時間帯がほとんどなくなり，稼動中に保守をせざるを得ない状況なってきている．このように稼動中に保守ができることをオンライン保守性と呼ぶ．

③ フォールトトレランス性；

システムのハードウェア，ソフトウェアを含めた品質，信頼性を向上したとしても，システムが広域，大規模になるに従い，どこかで故障の生じてしまうことは避けられない．この時，例え部分的な障害に対しても，その障害波及を阻止でき，システム全体の稼動停止が避けられることをフォールトトレランス性と呼ぶ．

(2) コンセプト

自律分散システムコンセプトは次の観点に立ってシステムを定義している[4]．

① 観点；

自律分散システムでは，

（ａ）サブシステムを統合したものがシステムである．

サブシステムを，その目的，機能を持つものとして定義する．それらのサブシステムが統合されたものがシステムで，トータルシステムが前もって決まっている訳ではない．

（ｂ）システムには機能不稼動なものを含む．

システム内には，障害，建設途中，保守中などのサブシステムが全くないということは現実的にはない．システム内には，機能不稼動なものが全くないとの前提に立ったものが従来のシステムである．そのため，機能不稼動なものがあれば，それをシステムの異常状態とし，システムを停止するか，特別な運用に切り換えていた．しかし自律分散システムでは「異常が正常(常にどこかが異常になっていることが正常な姿である)」という考え方をとる．

② 性質；自律分散システムでは，上記２つの観点に立ち，次の２つの性質を定義できる．

（ａ）自律可制御性（autonomous controllability）；いかなるサブシステムが機能不稼動になっても残りのサブシステムは自らを制御できる．

（ｂ）自律可協調性（autonomous coordinability）；いかなるサブシステムが機能不稼動になっても残りのサブシステムは互いに協調できる．

③ 条件；２つの性質，自律可制御性，自律可協調性を満たすためには，システムは次の条件を持てば十分である．

（ａ）構造：均質

サブシステムは均質な構造である．構造的には均質であるが，もちろんそれぞれは異なったアプリケーションの機能を持って良い．構造的に均質とはシステムの入出力関係が同一構造ということで，いかなるサブシステムも接続できる．

（ｂ）機能：平等

サブシステムの機能は平等である．それぞれが自らの判断で制御，協調できることである．他のサブシステムに指令を出して動かしたり，他のサブシステムから指令を受けて動くのではない．

（ｃ）情報：局所

サブシステムは局所的な情報を用いて制御，協調できる．大局的情報によって，より評価を向上させることはできたとしても，自律的な制御と協調は，局所的情報だけでできなければならない．障害や拡張によって，常に変動するシステムにあっては，システム全体の構造や機能が変化し，いつ，どこまでが全体かを決めることはできない．そのため従来のように，大局的情報を前提とすることはできない．

(3) 適用アプリケーションシステム

当初は制御システムのネットワークにこの技術が適用され，その後鉄鋼，ＦＡ，大規模鉄道システムの制御に適用され，さらにインターネットサービスプロバイダシステム等，ネットワークアプリケーションへもその適用範囲が拡大されてきている．

2.1.2 データフィールドアーキテクチャ

(1) 構成

自律分散システムを実現する上での基本となるのがデータ駆動型アーキテクチャとしてのデータフィールドアーキテクチャである．図 2.1 に示すように分散するサブシステムを均質な構造を持つアトムと呼び，これらをネットワークで接続するコミュニケーション手段としてデータフィールドを持っている．１つのア

トムの構造はデータフィールドとアトム内のアプリケーションソフトウェアとの間で，データ駆動のエンジンとなるＡＣＰ（Autonomous Control Processor）から構成される．さらに１つのサブシステム内にある複数のアプリケーション間も同様の仕組みでコントロールするためにアトム内データフィールドを持っている[5]．

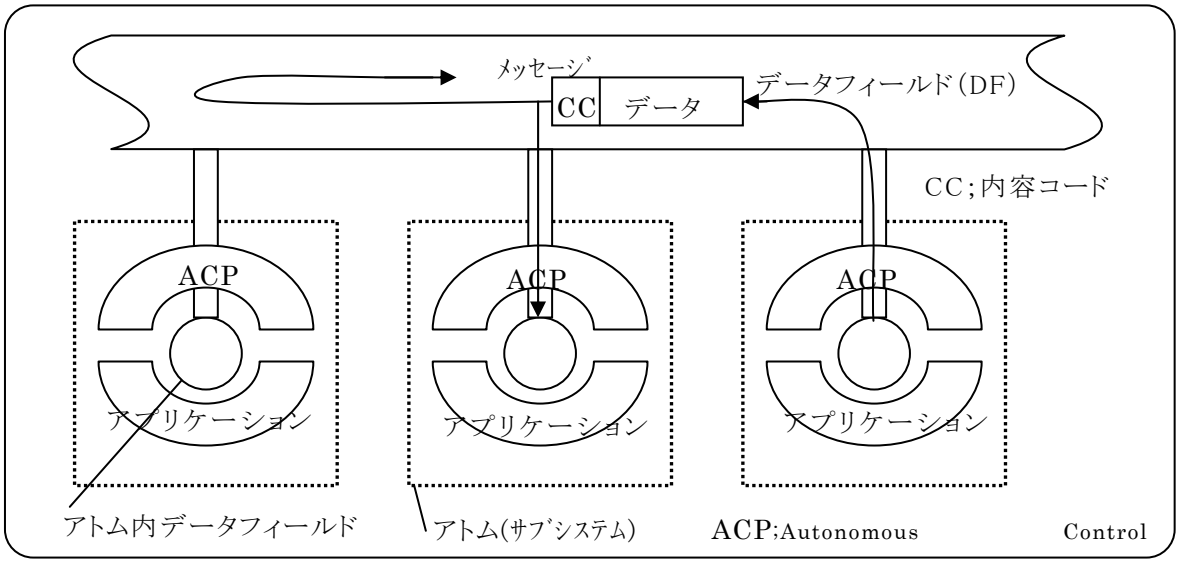


図 2.1　データフィールドアーキテクチャ

(2)　特長

自律分散システムにおいては，データは流れているもの(フロー)と見なしている．サブシステム間をながれるメッセージは内容を示す内容コード（ＣＣ：Content Code）をデータに付けて流す．これを受けて各ＡＣＰは予め登録している必要な内容コードのメッセージだけを取り出し，その内容コードに対応したアプリケーションソフトウェアを駆動する方式を採用している．このようにしてメッセージに送信アドレスを特定せず，内容を示すコードを付けてデータを流し，受信側が主体的に必要なメッセージを選択して受信してアプリケーションソフトウェアを駆動することにより，自律性を実現しているのがデータフィールドアーキテクチャの特長である．

(3)　内容コード通信方式

従来の通信は送信側が受信側のアドレスを指定していた．これに対しデータフィールドアーキテクチャでは受信アドレスではなく，データの内容に対応した内容コードを用いる．送信側はデータに内容コードをつけてデータフィールドにブロードキャストする．各サブシステムはデータフィールドに自分の必要とする内

容コードを持つデータが流れてくればそれだけを取り込む．従来は送信側が受信側を指定するもので，「選択送信」である．これに対して内容コード通信は，送信側は受信側を指定せず，受信側が自らの判断でデータをその内容で選択して受信する「選択受信」である．この内容コード通信により，いかなるサブシステムも通信に対して他のサブシステムから送信や受信を指示されることはなく，サブシステムは，その送信，受信を自ら制御でき，かつ，協調が取れるという自律性を達成できる．

データフィールドは物理的には，ネットワークであるが，メモリでも良い．1つのアトム（サブシステム）の中に複数のアプリケーションソフトウェアを同居させるのが現実解であり，このため，アトム内データフィールドを構成して，1つの計算機の中でもデータフィールドアーキテクチャを実現している．

(4) データ駆動型処理方式

従来の計算機ではソフトウェアが複数あると，それらの順番を決め，順次駆動していた．これに対してデータフィールドアーキテクチャではサブシステム内のアプリケーションソフトウェアは，必要な内容のデータがそろえば処理を開始するというデータ駆動方式である．駆動に必要なデータは複数あっても良い．複数の内容のデータに対してはデータの AND 駆動となる．OR の駆動条件でも良いが，これは入力データごとにアプリケーションソフトウェアを分けたことに対応している．処理結果はデータとしてデータフィールドに送出され，そのデータが他のアプリケーションソフトウェアに順次利用されて行く．

2.1.3 オンライン保守技術[6,59]

(1) システムニーズ

自律分散システムはオンラインプロパティ（即ち稼動中システムの段階的拡張性，システムを止めずにテストや保守を行うオンライン保守性，いかなる部分の障害に対してもシステム全体を止めないフォールトトレランス性）が特長であり，このためデータフィールドアーキテクチャにおいて，システムを止めずに，システム稼動中に，テストや保守を行う仕組みを実現する必要がある．

(2) オンラインテストの定義

テストモードのアプリケーションソフトウェアがテストデータを用いてテストするのが従来のオフラインテストである．オンラインデータをテストモードのアプリケーションソフトウェアが用いてテストしたり，テストデータを用いてオ

ンラインアプリケーションソフトウェアがテストするのをオンラインテストと呼ぶ．図 2.2 にオフラインテストとオンラインテストの定義を示す．

データ / Apl	テスト	オンライン
テスト	①オフラインテスト	②オンラインテスト
オンライン	③オンラインテスト	オンライン

オフラインテスト;①テストAplのテストデータによるオフラインテスト
オンラインテスト;②テストAplのオンラインデータを用いたオンラインテスト
;③オンライン稼動中のテストデータを用いたオンラインテスト

図 2.2 オンラインテストとオフラインテストの定義

(3) データフィールド上のオンラインテストのための仕組み

オンラインテスト実現のためにはシステム内にオンラインモードとテストモードのアプリケーションソフトウェア，及びオンラインデータとテストデータを混在できなければならない．このためデータフィールドに流れるメッセージには内容コードの他にテスト用データであることを識別するテストフラグ TF（Test Flag）を付加する．このメッセージがデータフィールドに送出されると，各サブシステムはテストフラグを基にオンラインかテストかの処理を行う．アプリケーションソフトウェアはオンラインモードと同じ環境下でテストするが，その結果として生成されたデータはテストフラグをつけてデータフィールドに送出し，それ以外の出力装置への出力を抑止する．この出力抑止により実稼動中のプロセスを妨害することはない． BIT（Built In Tester）は ACP の中の機能で，テストフラグのついたデータを受けてテスト処理するための機能である．EXT(External Tester）はテスト結果の監視や，テストデータ生成のためのアプリケーションソフトウェアである．EXT はシステム内にいくつあっても良い．EXT はデータフィールドを監視し，オンラインデータやテストデータとそれらが処理されて生成されたテストデータを対応付けることにより，テストの結果を把握できる．

オンライン保守技術により，オンライン稼動中のサブシステムを妨害することなしにテストができ，かつ現場でのテストを可能とし，テストデータ生成の負担も大幅に改善できる．

以下にオンラインデータを用いたオンラインテストとオンライン稼動中のオ

ンラインテストの２つの例について示す．

(4)　オンラインデータを用いたオンラインテスト

オンラインデータを用いたオンラインテストは図 2.3 のように実行される．テストアプリケーション（Test　APL）は，オンラインアプリケーション（Online APL）が DF からデータを受信するのと同様に，同じ内容コードを持つオンラインデータを DF から受信する．ACP はこのオンラインデータにテストフラグをオンしてアプリケーションに渡す．

ACP に組み込まれている組み込みテスタ（BIT）はテスト APL から生成される出力データに対して，テストフラグ・オンを付加する．テストフラグ・オンが付加されたデータは対応する内容コードを付加して DF に送信される．仮にテスト APL が同じサブシステムに接続されているデバイスに，あるコマンドを実行しようとすると，BIT はそのコマンドを抑制する．このことにより，テスト APL はオンラインアプリケーションの処理の実行への干渉やオンラインデータへの損傷を与えることなく，オンラインアプリケーションとテストアプリケーションの並行テストが可能となる．

また，オンラインテストによって生成されたテストデータの収集や，テスト状況の把握は外部テスタ（EXT）と呼ばれるアプリケーションソフトウェアによって実現される．EXT は DF に接続されたアプリケーションソフトウェアであり，システムには複数の EXT が存在しうる．

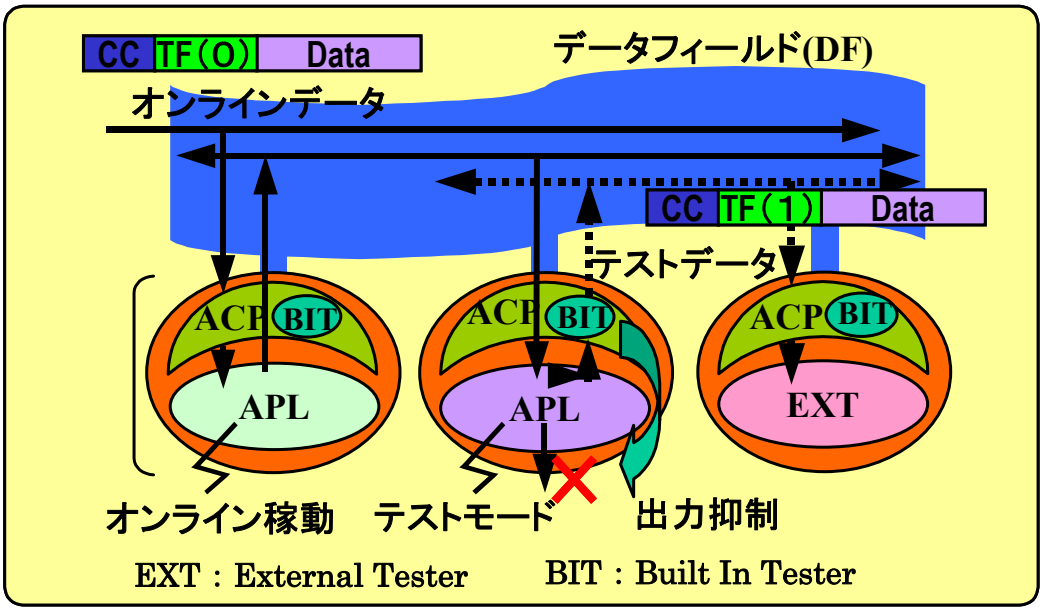


図 2.3　オンラインデータを用いたオンラインテスト

(5)　オンライン稼働中のオンラインテスト

オンライン稼働中のオンラインテストは図 2.4 のように実行される．テストフラグ・オンのデータが DF に送信されると，通常のオンラインデータと全く同様にオンライン APL によって受信され，処理される．しかし，受信側サブシステムに組み込まれている BIT は，テストフラグにより，このデータがテスト目的であることを知り，出力データにテストフラグ・オンをセットし，DF に送出する．オンライン APL からデバイスに対するコマンドもやはり BIT によって抑制される．このようにテストアプリケーションとオンラインアプリケーションは，あたかも実環境のもとでオンラインデータを処理しているかのごとく稼動することができる[24]．

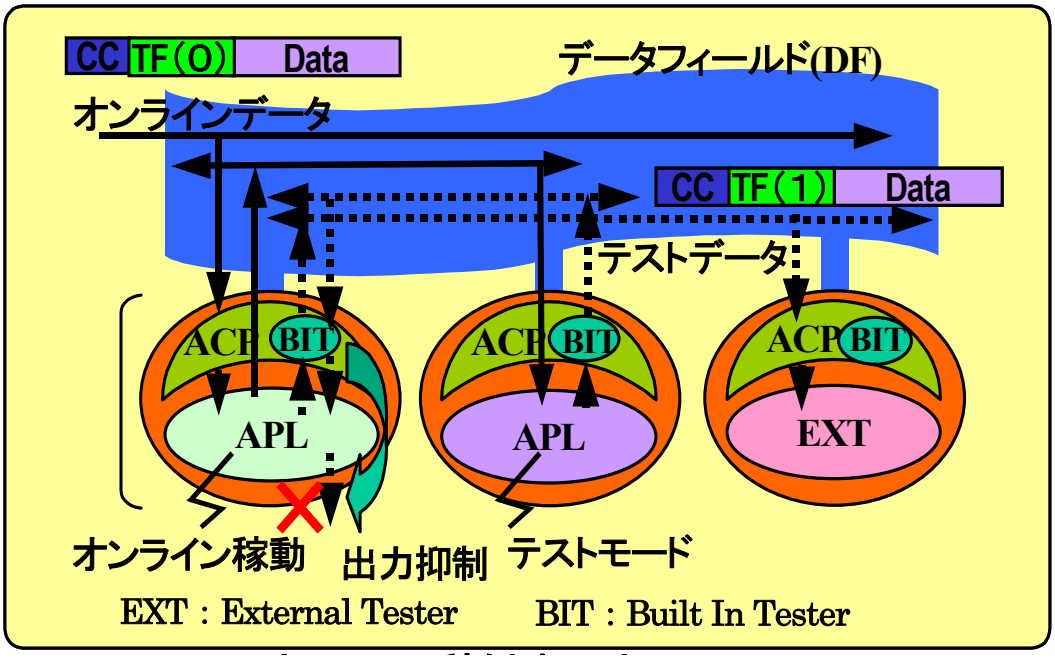


図 2.4　オンライン稼動中のオンラインテスト

2.1.4　　オンライン拡張技術[7]

システムの拡張には，幾つかのレベルがある．個々のソフトウェアモジュールや，複数のソフトウェアモジュールが格納された計算機というコンポーネントレベルから，システムとシステムとの統合というシステムレベルまである．自律構成化の考え方から，いかなるレベルも同じように対応できることも含め，それぞれのレベルでのオンライン拡張技術について述べる．

(1) コンポーネントレベル拡張技術

データフィールドアーキテクチャでは，アプリケーションソフトウェアが，データフィールドとのみインターフェースを持ち，アプリケーションソフトウェア間の直接的な関係はない．そのため，アプリケーションソフトウェアは，計算機内に新たにインストールされると，ＡＣＰ(Autonomous Control Processor)に必要な入力内容コードと，処理した結果生成されるデータの出力内容コードを登録するだけで，処理を開始できるようになる．

ＡＣＰを介したデータフィールドからの受信によって得られた入力データを格納するアトム内データフィールドは，アプリケーションソフトウェアごとに分割して管理される．データフィールドへの出力データも，同様にアプリケーションソフトウェアごとに独立に管理される．このため，アプリケーションソフトウェアの追加においても他のアプリケーションソフトウェア間でのデータ利用に関して何ら影響を与えるものでない．アプリケーションソフトウェアは内部データを持たず，処理の途中で生じたデータを保持し，次の駆動時にこのデータを処理に用いることはない．このため，アプリケーションが拡張したアプリケーションソフトウェア間の関連が変化したとしても，影響を受けずそれらソフトウェアは処理を続行できる．

複数のアプリケーションソフトウェアが格納されている計算機の拡張とは，論理的には，データフィールドにサブシステムを追加することである．そのため，いかなる他のサブシステムの変更も要せず，処理を停止する必要もない．

(2) システムレベル拡張技術

複数のシステムが互いに接続され，新たなシステムとして拡大するとき，二つの統合方式がある．第一は，各システムが構成するデータフィールドを，統合によって一つにする方式，第二は，それぞれのデータフィールドがゲートウェイを介して接続され，元々のデータフィールドを保存する方式である．データフィールドでは，そこに接続されたサブシステムがすべてのデータを共有できる．データフィールドが一つになるということは，異なったシステムのデータすべてがデータフィールドに流れ込むことである．詳細は２．３節で述べる．

2.1.5 自律分散システムの応用動向

自律分散システムは，当初は制御システムのネットワークにこの技術が適用され，その後鉄鋼プラント制御，ファクトリーオートメーション（ＦＡ），鉄道システムに適用され，さらにインターネットサービスプロバイダシステム等，ネットワークアプリケーションへとその適用が拡大されてきている．

ここでは鉄道の列車制御システムとファクトリーオートメーションシステムでの応用事例を述べる.

(1) 車上自律分散型の新しい列車制御システム（デジタルＡＴＣ）[19]

高密度運転線区の列車制御システムでは高輸送力，高安全性，高信頼性が求められる．従来は固定した区間ごとに列車の存在を検知し，そこに列車を進入させるかどうかの判断により，各列車の速度を地上の集中制御装置で求め，それを各列車の自動列車制御装置（ＡＴＣ；Automatic Train Control）に指示していた．このため下記のような問題を持っていた.

① 列車本数を増やすことができない
区間ごとにブレーキがかかったり，緩んだりすることで，無駄な時間が生じる上，列車間隔をある程度までしか縮めることが出来ない.

② 下位の速度区間に入ると急激なブレーキがかかる.
区間ごとに許容速度が決められているため，下位の速度区間に入ると急に急激なブレーキがかかり，低速になればなるほどその傾向は顕著に表れ，乗り心地が極めて悪くなる.

③ 地上装置が重厚長大である.
地上装置に速度情報，ブレーキ情報などすべての情報を持たせているため，設備が複雑となる.

この問題を解決するため車上自律型のデジタルＡＴＣ（Ｄ－ＡＴＣ）が開発された．この新しい列車制御装置は自律分散システムのコンセプトにより作られている．即ち，各列車が自らの位置検知を行い，地上からは停止する位置のみを列車に伝送し，車上ではそれに基づく列車速度の決定と制御を自律的に行う自律分散型ＡＴＣシステムである.

地上システムは分散配置としてネットワークで結ぶ構成とし，車上システムは地上からの停止点情報をもとに自律的に列車のブレーキ制御を行う自律分散システムとしている．地上システムは，ネットワーク範囲内に在線する列車の位置を把握し進路開通情報などを加味して，各列車にデジタル電文として送信する．従来のアナログ伝送方式と異なりデジタル符号伝送のＣＲＣチェックによって安全性は飛躍的に増大している．このように各論理部に自律性を持たせた構成とすることにより，システムの異常が全体に波及することを防止するとともに，段階的な構築も可能となっている.

一方，車上システムは，地上から停止する位置情報をもらうだけで，車上におい

て自身の速度と，勾配・曲線などの地理的条件を加味してブレーキ制御をする．したがって，新しい車両の機能が向上すれば，地上のシステムを変更しなくても，その車両の性能に合わせたブレーキ制御をすることができるようになり，自律性を持った車上システムとなっている．その結果，将来列車本数を増加させることがきわめて容易となり，混雑緩和となり，乗客へのサービス向上につながっている．

(2) 東京圏輸送管理システム（ATOS）

首都圏の列車輸送管理システムは制御と情報という異質なミッションであるオンラインプロパティと異種プロパティを満足するように設計されている．列車の運転を止めることなしに段階的にシステムを構築するという要請とすでに稼働しているシステムに影響を与えないようにするという要請との両方を満足している．このシステムは ATOS(Autonomous Decentralized Transport Operation Control System)と呼ばれ，10 年以上の開発期間を経て，1994 年に中央線で使用開始された[3,60]．構築に当たっては，駅システムを順次構築していって最後に線区全体のシステムが出来上がるという段階的構築手法が取り入れられている[61]．

① 駅システムの構築
② 駅ネットワークを通じて同一線区の他の駅システムとの接続
③ 上記２ステップの繰り返し
④ 駅システムが数割出来た段階で，運行モニターシステムの構築
⑤ ③ステップ
⑥ 駅ネットワークが出来てきた段階で，線区ネットワークとの接続
⑦ 線区ネットワークと輸送管理システムとの接続

駅システムは GW（ゲートウェイで結ばれた）制御と情報の２つのサブシステムよりなり，異種のデータを同時に扱えるように構成されている．

このように ATOS システムは大規模システムであり段階的に構築していき，駅数を次第に増やし，線区を構成し，さらに線区を結んでゆくというボトムアップ手法により構築されている．

自律分散技術を使ったことによって，システムは完成した駅から逐次使用開始していくというオンラインプロパティが実証された．使用開始後も駅システムの改修，機能追加，線区指令装置の増設が頻繁に行われているが，システムの稼働停止に至ることは無く，列車乱れが生じた後の平復時間（ダイヤが平常に戻るまでの時間）も大幅に短縮されるなどの効果が上がっている．1992 年 6 月にまず中央線（東京

ー甲府間）から構築が開始され，1994 年 3 月に第 1 号となる相模湖駅装置が完成し，稼動を開始した．以降，次々と駅装置と線区システムを構築し稼動を開始しながら，（2006 年 3 月末）現在は首都圏の主要線区（中央快速・緩行線，山手線，京浜東北線，横須賀線，総武本線，東海道線，常磐線（上野～羽鳥間），東北線，高崎線，埼京線，川越線，南武線など）に導入されている．

(3) タイヤ生産管理システム

近年，四輪駆動や RV 車などの SUV（Sports Utility Vehicles）など，市場での急速なエンドユーザニーズの変化や多様化により，これまで以上に迅速，且つきめ細かなタイヤの生産が要求されている．タイヤの耐久性や性能など，顧客ニーズにマッチした高品質な製品をタイムリーに提供する必要がある．

また，タイヤはさまざまな用途に使用されている．スクータのような小さいものからジャンボジェットのタイヤ，あるいは，鉱山などで使用される 1 本が数トンに及ぶ建設車両用タイヤまである．このような多種多様なタイヤの構造や製法はかなり複雑である．原材料は，天然ゴム，合成ゴム，カーボン，配合剤，補強材のナイロン，ポリエステル，スチールワイヤー，アラミド繊維など数百種類以上にのぼる．これらの原材料を加工して，つぎの工程で使用できるような中間部材を作り，成型工程，加硫工程，検査工程など 15 工程以上を経て最終的な製品であるタイヤが完成する[69]．

ブリヂストンでは，1980 年代から工場内のネットワーク化を進めてきた．当初は，制御機器メーカ独自のハードや，プロトコルを使用したネットワークであったが，80 年代後半からは，Ethernet によるネットワーク化を進め，物理的な部分は標準化をすすめていった．しかし，情報アクセス技術はまだバラバラであった．その後 90 年代になって，クライアントーサーバ型による情報活用を進めていった．

そのような経緯から，ブリヂストンは情報を加工して蓄えることなく，生のまま流す考え方に着目した．生データを，品質，性能，稼動等の情報に分類し，たとえば，長さ，重量，厚さ，製品タイヤの運動特性など 5,000 項目を超える情報に分類・整理した．現時点で使用されていない項目も多数含んでいる．この生データを「広義の改善」活動を行うための情報基本単位とする．基本単位である生データを利用者が自由自在に収集して，利用する側で必要に応じて加工して取り扱えるようにすることで，製造現場部門のみならず，製品開発・設計部門にわたる顧客ニーズに立脚した，改善活動の多様な情報要求対応できるようにした情報利用のコンセプトを作り上げ，情報の Just In Time を実現することを目指した．

このような仕組みを FOA（Flow Oriented Approach）と定義した．FOA では，工場内のネットワークを活用して，情報すなわち生データが発生のつど送出する発信

型の情報システム構造を基本にしている．そこで FOA を実現するアーキテクチャとして柔軟性，拡張性を備えていた自律分散アーキテクチャを採用することとなった[1,11]．90 年代始めには自律分散はまだメーカ独自プロトコルであったが，メーカにオープン化を促し，かつ，他社制御機器メーカにもプロトコル実装を働きかけ，ユーザ主導での標準化（ADS-net：Autonomous Decentralized System network）を進めた[70]．自律分散システムのデータフィールドアーキテクチャは，すべての情報を蓄えずに流しておき，必要なユーザが選択受信するという考え方で，まさに FOA が狙っているコンセプトと合致していた．

(4) 新聞生産工程管理システム

デジタル技術の急速な進歩により，情報伝送媒体は大きな変革期を迎え変貌してきている．インターネット上で個人組織を問わず，さまざまな情報を発信し，また，情報を収集しているなど，既存のメディアを使わない情報交換が容易に出来る状況となっている．このインターネットを媒体として使用すれば，現在のように新聞をいちいち紙に印刷し配達するという非効率的な作業がいらなくなってしまう．

21 世紀になり，このように新聞を取り巻く状況が大きく変わってきている．当然ながら，読者が新聞に求めているニーズは変化し，多様化してきている．新聞の持っている「モノと情報が一体化」しているという特徴は，他のマスメディアにはないもので，メディアとしてその価値を保ち続けるには，状況やニーズの変化を先取りできるシステムの構築が求められた．

このニーズに基づいて開発されたシステムが新聞生産工程管理システムである．新聞の生産工程は，刷版工程，印刷工程，宛名工程，積込工程の４つからなっている[2]．

刷版工程 ： 大量生産の入り口であり，新聞専用の印刷機であるオフセット輪転印刷機用の版である刷版を作成する．

印刷工程 ： 高速で低コストが求められる印刷であり，朝刊で一時間に８万部の印刷をしている．

宛名工程 ： 店単位に細分化した，宛名を新聞の束ごとに書き込む．

積込工程 ： 分単位で着発するトラックに新聞の束を仕分けして積み込む．

このように，作業内容もスピードも違う種類の作業をシステム化することは，相当の難作業になるといえる．しかも，刻々変化するニュースバリューへの対応，顧客数の変更への対応などをも同時に行う必要がある．

これらの処理能力の違い，変化への即応を可能ならしめたのが自律分散アーキテクチャである．本社システムと工場システムとを一つのデータフィールドで結び，工場内の各工程のサブシステムを別のデータフィールドで結んで，制御データとサイズの違いを吸収するようにし，また，各レイアが自律してデータフィールドを介してデータを共有することにより，フォールトトレランス性をも実現できている．

朝日新聞の世田谷工場で初めて採用された後，より多くの工場で汎用的に使用できるように修正を加えた後，1997 年に川崎工場に導入，他の工場にも次々と導入されていった．また，1998 年春には大阪と西部の両本社にもシステムが入って完成した．

2.2　アシュアランス技術

アシュアランス技術はシステムの安定稼動を保証する技術で，異種性と適応性の２つの要素を考慮したシステムに適用される技術の総称である．異種性と適応性を持ちシステムの安定稼動を保証するシステムをアシュアランスシステムと定義されている[14, 15, 16, 17]．

2.2.1　　異種性

システムは多くのニーズを満たすものとして構成されるが，このニーズの要求度合いはシステムにより異なっており，これをニーズレベルと呼ぶ．

システムの稼動の保証に求められている評価基準は様々であり，例えば以下のような異種の評価基準が挙げられる．

① 信頼性
② アベイラビリティ
③ フェールセーフ
④ 安全性（セキュリティ）
⑤ リアルタイム性

システムを設計する際にはこれらの評価基準ごとにニーズレベルを満たす必要があるが，そのレベルはまちまちである．例えば制御システムにおいては一般に信頼性とリアルタイム性が求められているのに対し，情報システムではアベイラビリティや安全性が求められる．もちろん，これらのレベルをすべてのシステムのニー

ズよりも高く設定すれば，すべてのシステムのニーズを満たすことができるが，それは現実的ではない．そこでシステムごとに要求されるレベルを満たすことになるわけであるが，システムのネットワーク化の背景により，ニーズレベルが異なるシステムが接続されていることから，システムが単独でニーズレベルを満たすことができたとしても，他のシステムと連携した時にはその限りではない．

異種のニーズレベルを持ったシステムの稼動を保証するためには，異種のニーズレベルの共存を許容するシステム技術と異種の評価基準を統合した新たな評価基準が必要となる．

2.2.2　適応性

ユーザニーズが異種性を持つのみでなく，それが時間的に変化することが，近年より顕著になってきている．

このような異種性を持つユーザニーズの変化に対応して，システムとしても適応して行かねばならない．従来のように，システムを量的に順次拡大するだけでは解決しなくなってきている．ユーザニーズの変化に合わせ，システムを質的にも適用できるような対応が必要になってきている．こういった変化するニーズに対応するためには，いかなる状況の変化が起こっても，常に柔軟に対応しうるシステムの構築が必要となる．このような状況変化への対応能力を適応性と呼ぶ．

2.2.3　アシュアランス性

時間とともに変化する異種のニーズに適応でき，システムの稼動を保証できうる性質をアシュアランス性と呼ぶ．

システムを適応させるには，システムの中に異種のモード（建設，障害，テスト，拡張など）が当然共存しうる．特に，この異種モードを共存させたとしてもシステムの稼動を保証する性質をオンライン・プロパティ（on-line property）と呼び，これを実現したのが自律分散システムである[4,5,6,7,8]．このオンラインプロパティは，３つの性質，フォールトトレランス性，オンライン拡張性（段階的拡張性），オンライン保守性(稼動中テスト/修復性)からなる．アシュアランス性実現のために，この自律分散システムのさらなる研究が進められている．

これまで，このアシュアランス，自律分散システムの技術は，一般産業，情報サービス，鉄道，宇宙などの分野を主な対象としてきた．以上のような社会インフラに対する要求が強くなるにつけ，社会インフラでもアシュアランス技術が検討されるようになってきた．

2.2.4　アシュアランス技術の動向

アシュアランス技術は，米国国防省やエネルギー省，また，日本の研究者も加わり，1996 年設立した IEEE 主催になる HASE(High Assurance Systems Engineering Symposium)[10]がきっかけとなり，研究が加速された．日本でも，電子情報通信学会の FTS 研究会のもとに第 2 種のアシュアランスシステム研究会が 2000 年秋に発足[3]，以来，活発な研究活動が進み，北米以外では初めて HASE が 2002 年 11 月に日本で開催された．米国などでは，このアシュアランス技術は，国防省ではインターネットなどの分野で，エネルギー省では，電力(原子力発電を含む)，交通（BART など)，都市公共設備など分野で検討されてきた．日本では，アシュアランスの言葉は用いられてはいなかったものの，この技術では世界をリードしており，既に実用化も進んでいる．以下，鉄道分野での実用例を紹介する．

国鉄の民営化以降，JR 各社は，鉄道輸送の安全性のみならず，乗客へのサービス向上のためのシステムを研究，開発してきた．この乗客へのサービス向上を目的とすることにより，ニーズの異種性，そして適応性，つまりアシュアランス性を満足させなければならなくなり，これを最初に実現したのが，世界最大規模の鉄道輸送管理システム　ATOS(Autonomous Decentralized Transport Operation Control System)である．このシステムは，従来のような列車追跡，進路制御といった制御システムのみでなく，乗客への情報サービス，柔軟なダイヤ計画サービスなどの情報システムを統合している．さらに，このシステムは，東京圏 19 線区，288 駅，路線長 1045.5km，1 日あたりの乗客数約 1400 万人（2005 年度版 ATOS 見学パンフレットより)を対象とする世界最大規模の社会インフラである．規模が大きいため，このシステムは，1990 年の計画以来　1992 年中央線での建設開始，1996 年実用化から，順次他線区でも開発，実用化を進め，現在でも続行中である．このように開発期間が長いことからも，ユーザニーズを完全に予測できず，システムの適応性は不可欠である．また，情報と制御システムが連携して，それぞれの異種の要求を満たし乗客指向の運行を実現し，アシュアランス性を実現した自律分散システムの技術が用いられている[20,21]．

また，乗客へのサービス向上のため，デジタル自動列車制御装置（D-ATC）も開発し，順次，新幹線や東京圏の列車に導入を進めている．このシステムは，列車自体を自律化させるもので，各列車が位置を検知し，列車相互の交信により，それぞれで走行速度やブレーキ制御により安全な運行を保証するものである．これにより，従来のような地上側での集中的な制御でなく，状況に応じた制御ができるため，列車間の間隔を狭めることができ，列車の混雑度を緩和できる．このシステムはもちろん，地上側のシステムとの連携も必要である．また，このような装置は，全車両

に同時に設置することは経済的にも難しい．そこで，順次，この装置を導入し，線路上には，新旧の装置を搭載した列車が混在する．当然，安全性は，実環境下でテストされなければならないが，営業時間の増大により，テストも，営業時間内にやらなければならないようになってきている．このような，稼動／テストモード，新／旧装置搭載列車を共存させるといった異種ニーズと，運行状況の変動に合わせた適応性からなるアシュアランス技術が研究，実用化されている[18]．

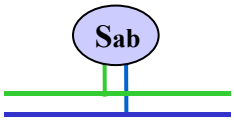
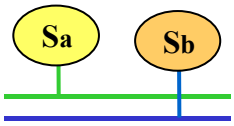
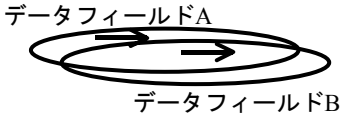
この他，アシュアランス技術は，宇宙の分野で民生品を用いた高信頼，高機能，高性能な高アシュアランスの宇宙用搭載コンピュータの実現を目指してアーキテクチャが提案されている[22,23]．また，医療分野でもペースメーカのような埋め込み型医療装置は，機械的機能，電気的機能，ソフトウェア機能は分解されてデザインされ，各サブシステムはそれぞれに対する要求を満たすものとして設計され，そのうえで再び統合されている．このように，異種の目的を持ったサブシステム群を統合し共存させることはアシュアランス技術の一つであり，今後の研究に期待される．以上のように，アシュアランス技術は情報サービスシステムなどの社会インフラ向けに，研究，実用化が進んでいる．

2.3　異種システム共存技術

自律分散システムは，システムを構成している要素であるデータフィールドとサブシステムとからできている．また，自律分散以外のシステムでもシステム同士が互いに結ばれている場合，個々の結合が１つのデータフィールドであると見なすことができる．

共存のための技術について，基本的な構成方法をデータフィールドに対して３つ，サブシステムに対して２つ提示し（表 2.1），その構成方法の組合せによって異種システム共存の６のタイプを説明する．

表 2.1　システム共存の基本的構成方法

	データフィールド	サブシステム
統合型		
分離型		
共存型		

2.3.1　　データフィールド

データフィールドの性質は，そのデータフィールドを構成している伝送速度，光ファイバー，無線，メモリなどの物理的な要素とデータフォーマット，データ形式などの論理的な要素によって次の３つに分類できる（表 2.2）.

表 2.2　データフィールドの三つのタイプ

論理的＼物理的	同じ	異なる
同じ	統合データフィールド	分離データフィールド
異なる	共存データフィールド	

(1)　統合型データフィールド

物理的にも論理的にも同一のデータフィールドを統合型と呼ぶが，以下に述べる分離型，共存型のデータフィールドを２つのシステムが共存できるようにしたものも含むものである．前者を一体型統合データフィールド，後者を接続型統合データフィールドと呼んで区別する．

システム双方が同じように相手のデータを利用し，かつ，時間的にもほぼ同時に

連携を図りたいという要求がある．また，状況に応じてアプリケーションプログラムを別システムの計算機に移行して実行することで，信頼性や応答性を改善したいという要求がある．このとき，データフィールドを統合して一つにまとめる．このデータフィールドの統合は，全く物理的にも１つのネットワークとすることによって実現できるが，２つのネットワークを接続するのみの対応でもよい．もちろん，別に１つネットワークを新たに構成することでも対応できるがコスト面で得策でない．

一体型統合データフィールドに流れるデータは，これまで別々に流れていたすべてのデータが合流するため，ネットワーク上のデータ伝送量は増大する．各サブシステムは，内容コードを基に選択受信するため，どのサブシステムからのデータも受信できる．

接続型統合データフィールドでは，全サブシステムを複数のデータフィールドと接続してデータ伝送量の増大を回避することができる．この方法によると，内容コードを基に選択受信するため，どのネットワークから受信してもよい．また，計算機は，それぞれのネットワークに同一データを重複して送信しても問題ない．このときは，受信側の計算機で，同一内容のデータを複数受信でき，ネットワークのフォールトトレランス性を向上させることができる．このように，ネットワークの負荷分散による性能向上とフォールトトレランス性向上を容易に実現できる．

(2) 分離型データフィールド

分離型データフィールドは物理的に異なる媒体を持ったデータフィールドとして定義される．流れるデータの論理的な構成は同一でもよく，また異なっていてもよい．このタイプの２つの分離型データフィールドを共存させるにはゲートウェイを必要とする．システムのデータフィールドを保存したままで，それぞれをゲートウェイで接続する．ゲートウェイは，接続するデータフィールド間にあって互いに共有したいデータのみを選択し通過させる．従来のシステムにおいては，システムをゲートウェイでつなぎ，その間のデータのやり取りを制御するために特別な装置を必要とする．

一方，自律分散システムにおいてはデータフィールドアーキテクチャによって以下のように行う．各計算機の ACP が，そのサブシステムで必要とする内容のデータのみを取り込むのと同じ機能である．データフィールドを構成するシステムにとっては，そこに接続されるいかなるサブシステムも自律したサブシステムと見なせる．ゲートウェイでも，それがデータフィールドに接続されれば，そのデータフィールドを構成するシステムの自律したサブシステムになり得る．つまり，ゲートウ

ェイとシステムを合わせて他方のシステムにとっての自律したサブシステムと見なせる．ゲートウェイを自律させるために ACP を持たせる．システム内には，幾つかのサブシステムが存在し，それらには，複数のアプリケーションソフトウェアがインストールされている．ゲートウェイの ACP は，これらアプリケーションソフトウェアが必要とするデータの内容コードを登録する．これにより，一方のデータフィールドを流れるデータの中から，他方のシステムが必要とする内容コードのデータのみがゲートウェイを介して取り込まれ，他方のデータフィールドに送出される．このゲートウェイから取り込まれたデータは，システム内で発生したデータと全く同様に，そのデータフィールド上を流れ，それを必要とするサブシステムで選択収集される．このデータが，他のシステムから送出されたかどうかにかかわりなく，アプリケーションソフトウェアは，それを用いて処理を実行する．また，この処理結果は，データフィールドに送出され，もし，他方のシステムがそれを必要とし，ゲートウェイにその内容コードが登録されていれば，そこを通過して他方のデータフィールドに送出される．

(3) 共存型データフィールド

物理的には同じデータフィールドの中に，論理的に別の体系のデータを流して，システム間で区別して使用する場合，これを共存型データフィールドという．データフィールドは別に存在するとみなすが，コンテンツコードが異なるデータが流れていると見れば同一のデータフィールドともみなせ，その意味では統合型データフィールドと同じ働きをする．システムのデータフィールドは保存したままであるが，これを実際に共存システムとして働かせるには，２つ以上のデータフィールドに１つのサブシステムから接続できるように構成して，サブシステムからそれぞれのデータフィールドに流れるデータを取り込めるようにするか，ゲートウェイで２つのデータフィールドを接続してもよい．

元々あるネットワークに，別サブシステムを接続し，それぞれのシステムのネットワークを用いるため，サブシステムはネットワークの本数だけ通信ポートを用意する必要がある．サブシステムの ACP は，アプリケーションソフトウェアがインストールされたときに，必要とする入力データの内容コードを登録する．共存型にするために，サブシステムをそれぞれのデータフィールドに接続する方法はいろいろ考えられるが，たとえば実際にケーブルでつないだり，無線で構成されたデータフィールドであればそれらを受信発信できるようにしたりするなどである．

この接続法では，ネットワークが複数あり，データフィールドに流れるデータは，それぞれのデータフィールドを流れるため，ネットワーク上のデータは共存させる

前と変化はしない．

2.3.2　　サブシステム

データフィールドの場合と同様，サブシステムを共存する方法にも２つの種類がある．サブシステムが２つのシステムと結ばれているか，あるいは２つの別システムのデータフィールドと結ばれているとき，いずれか１つのシステムあるいはデータフィールドとの間でしかデータのやり取りができない分離システムと，サブシステムが２つのシステムとデータをやり取りすることができる統合システムとの２つがある．以下にこの２つのシステムについて説明する．

(1)　統合型サブシステム

このサブシステムは，複数の種類のアプリケーションを持っており，これがサブシステム内に統合した形で存在する．それぞれのアプリケーションは，対応するシステムあるいはデータフィールドとの間でデータ交換を行い，その結果をサブシステム内の別のアプリケーションで処理を行うことによってシステムの共存を可能としている．

(2)　分離型サブシステム

このサブシステムには，１種類のアプリケーションプログラムしか組み込まれておらず，このアプリケーションプログラムは１つのシステムあるいはデータフィールドとしかデータのやり取りができない．したがって，システム共存を行うためにはデータフィールド側で対応を取る必要がある．

2.3.3　　異種システム共存技術

サブシステムとデータフィールドから成る２つのシステム A，B があるとき，この２つのシステムを共存させるには，サブシステムとデータフィールドの型の組合せにより，表 2.3 に示すような６つのタイプのシステム共存タイプがある．以下にそれぞれのタイプの特徴を説明する．

表 2.3 サブシステムとデータフィールドの分類

サブシステム ＼ データフィールド	統合型	分離型	共存型
統合型	Type-1	Type-2	Type-3
分離型	Type-4	Type-5	Type-6

Type-1; 統合型サブシステム－統合型データフィールド（図 2.5 (a)）

サブシステムとデータフィールドが共に統合型であり，サブシステムからブロードキャストされるデータは統合型データフィールドに流れる．この場合データフィールドが一体型統合データフィールドである場合は，このシステムは１つのシステムとなる．一方，接続型データフィールドである場合は２つのシステムが共存できたことになる．当然のことながらデータフィールドは統合されているため，それぞれのシステムのデータは同一データフィールド内に同時に流される．このためデータフィールドの伝送容量の検討が必要となる．

Type-2; 統合型サブシステム－分離型データフィールド（図 2.5 (b)）

サブシステムは統合型で，２つの分離型データフィールドの双方に接続されている．データはサブシステムのアプリケーションプログラムが作成し，対応するデータフィールドにブロードキャストする．データの受信はデータフィールドに流れているデータを選択受信して，対応したアプリケーションプログラムがそのデータにより処理を行う．これは，統合システムによりシステムを結合することとなる．

Type-3; 統合型サブシステム－共存型データフィールド（図 2.5 (c)）

データフィールドは共存型であるため，統合型サブシステムのアプリケーションプログラムはブロードキャストするが，データフィールドに合わせた形でデータフォーマット作成する．接続しているデータフィールドからはデータを受信して処理を行う．データフィールドを介してシステムの共存が可能となる．

Type-4; 分離型サブシステム－統合型データフィールド（図 2.5 (d)）

サブシステムは分離型であるので，それぞれに別のアプリケーションプログラムを持っているが，データフィールドにブロードキャストされるデータは同一の仕様のデータとして扱われる．統合型のデータフィールドに流れているデータはそれぞ

れのサブシステムにおいて選択受信される．データフィールドの統合によりシステムは共存している．

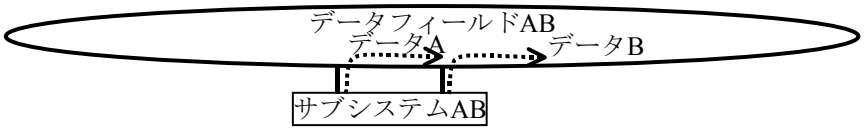
Type-5;　分離型サブシステム－分離型データフィールド（図 2.5 (e)）

サブシステムとデータフィールドが共に分離型であるということは，それぞれのシステム自体は互いに独立しているということである．これは，２つの独立したシステムである．ゲートウェイで両方のデータフィールドを結ぶことによりシステム共存ができる．

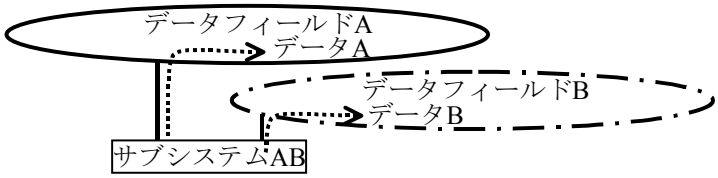
Type-6;　分離型サブシステム－共存型データフィールド（図 2.5 (f)）

サブシステムは分離型で，それぞれのアプリケーションプログラムはブロードキャストされるデータフォーマットをデータフィールドに合わせた形で作成できる．接続しているデータフィールドからはデータを受信して処理を行う．このままではシステムの共存はできていないが，センターシステム同士をゲートウェイで結ぶか，データフィールド間をゲートウェイで結ぶことによりシステム共存が可能となる．

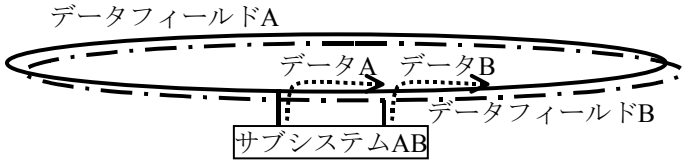
以上説明したことを整理すると，**Type-1** と **Type-4** はシステムは統合型データフィールドによってすでに共存しているといえ，**Type-2** と **Type-3** は統合型サブシステムによってシステム共存が実現でき，**Type-5** と **Type-6** はゲートウェイでデータフィールドを接続することによるシステム共存の方法である．



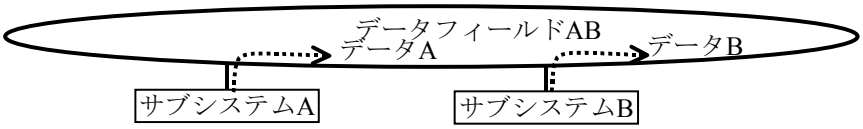
(a) Type-1　統合型サブシステムー統合型データフィールド



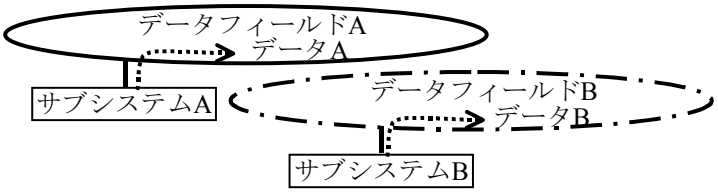
(b) Type-2　統合型サブシステムー分離型データフィールド



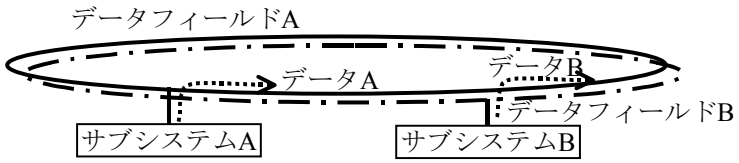
(c) Type-3　統合型サブシステムー共存型データフィールド



(d) Type-4　分離型サブシステムー統合型データフィールド



(e) Type-5　分離型サブシステムー分離型データフィールド



(f) Type-6　分離型サブシステムー共存型データフィールド

図 2.5　サブシステムとデータフィールドの分類

また，この共存型データフィールドによる共存技術を使って，異種システムの共存技術として次の２方式がある[24].

さらに，共存型データフィールドによる共存技術を使って，異種システムの共存技術としてこの２種類を統合した方式を新たに提案する.

(1) データ交換型システム共存技術

２つのデータフィールドをゲートウェイ(GW)により結び，２つのシステムの間でのデータを交換することによって，ニーズレベルの違う２つのシステムの共存，あるいはニーズの異なる他システムとの共存が可能となる．すなわち，別々のデータフィールド DF（Data Field）を有した，ニーズレベルの異なるまたはニーズそのものが異なる２つのシステムを共存させるのがこの方式である．このように，ニーズレベルもしくはニーズそのものの異なるシステム間に，相互のデータ交換処理を行う GW によってニーズの差を吸収させ，システムの共存を可能とした(図 2.6).

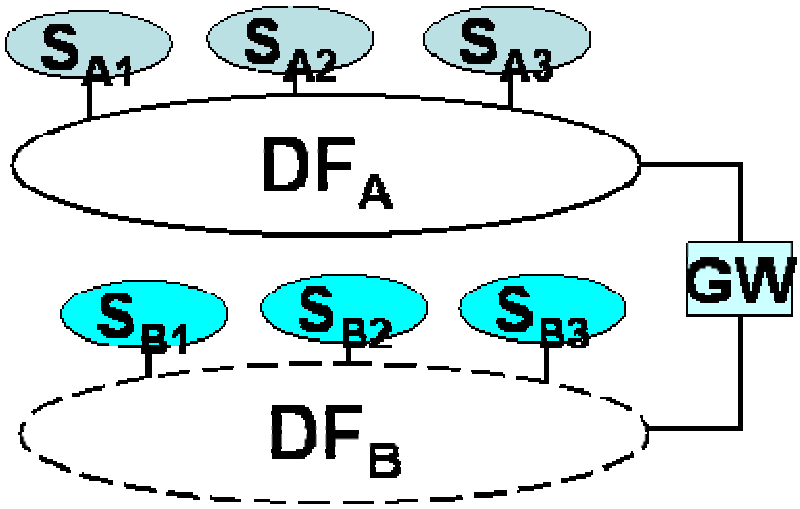


図 2.6　データ交換型システム共存技術

(2) データフィールド型システム共存技術

統合型のサブシステムが両方のデータフィールドにアクセスできるように構成する．このサブシステムは２つのシステムのデータを読むことによって，そのコンテンツコードに従ったリアルタイムな自律制御が可能となる．データにどちらのデータフィールドのデータかを区別できるフラグが付いているため，サブシステムはどちらかの処理をデータによって制御できる（図 2.7).

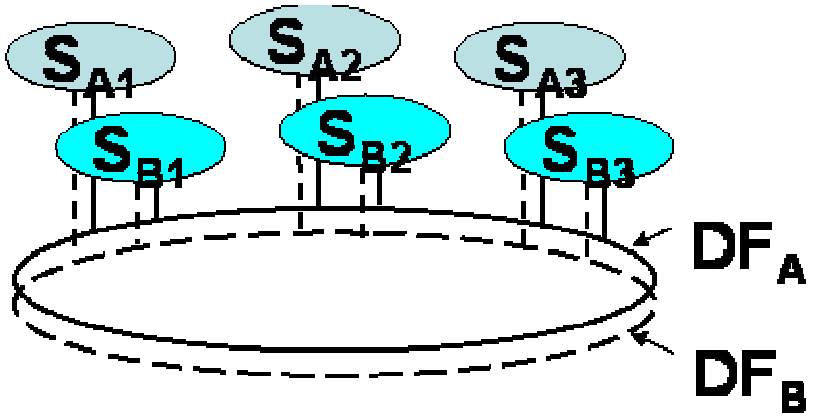


図 2.7　データフィールド型システム共存技術

(3)　データ交換・データフィールド統合型システム共存技術

データ交換型とデータフィールド型の２つの方式を統合したシステム共存技術である．データ交換型の持つ，「ニーズレベルの異なるまたはニーズそのものが異なる２つのシステムを共存させる技術」とデータフィールド型の持つ「２つのシステムのデータを読むことによって，そのコンテンツコードに従ったリアルタイムな自律制御が可能となる技術」の両方式の利点を活かしたシステム共存技術を提案する．（図 2.8）これを「データ交換・データフィールド統合型システム共存技術」と呼ぶ．

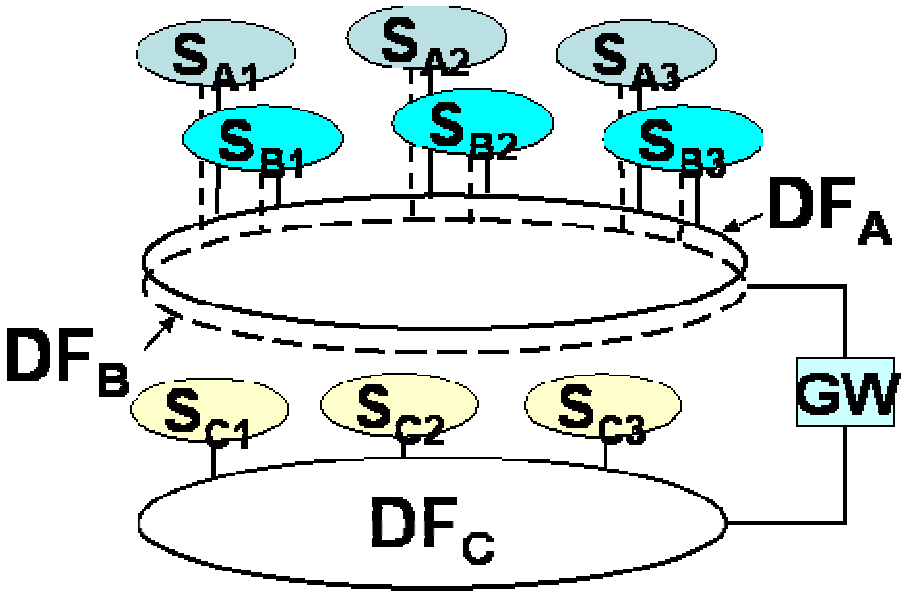


図 2.8　データ交換・データフィールド統合型システム共存技術

第3章

異種統合型情報サービスシステムアーキテクチャとアシュアランス技術

本章では，異種統合型情報サービスシステムのモデルを提案し，アシュアランス性を定義する．さらに，異種統合型情報サービスシステムにおける「システムアーキテクチャ」構築技術を提案し，このシステムアーキテクチャにより，高速処理のための「自律連携処理（自律分散アルゴリズム）」と高信頼性のための「自律分散整合化技術」とを提案し，高速性と高信頼性を両立させる自律分散アシュアランス技術について述べる．

また，このようなアーキテクチャと技術のもとで，システム変動下における稼動の保証度合を評価するためのアシュアランス性評価技術として，「機能信頼性評価技術」を提案し，その技術について解説する．

3.1　異種統合型情報サービスシステムアーキテクチャ構築技術

本節では，前第１章で述べた異種統合型情報サービスシステムのニーズを解決するための「システムアーキテクチャ」構築技術について解説する．

「異種統合型情報サービスシステムアーキテクチャ」は，異種データの特性に応じてデータフィールドを構成し，それらが目的に応じてゲートウェイを介してデータを交換できるようにしてある．ここでは各サブシステムに自律性を持たせ，それらがデータフィールドを介した連携により，システムの部分的障害，拡張，保守においても，稼動の継続性が保証できるようにした．

詳細は３．１．３項で述べるが，本論文で提案する「異種統合型情報サービスシステムアーキテクチャ」は，データフィールド毎に処理の時間を変化させた「時間差異種データフィールド」技術により構成したシステムである．

3.1.1　異種統合型情報サービスシステムの論理モデル

前第２章では，異種ニーズ，異種モードの共存する環境下でシステムの安定稼動を図るための技術として自律分散システムとアシュアランス技術について述べてきた．

以下に，複数の異種システムを共存させる場合についての論理モデルを考察する．図 3.1 は複数の GW を用いて分離型システム共存を実現している．自律制御プロセッサーACP を内蔵した自律分散サブシステムである GW によって，異種システムを統合することにより，接続されたシステム自体が 1 サブシステムとして取り扱えるので，統合に伴うパラメータの再設定や，システムの再稼動などの複雑な作業は不要であり，容易にシステム統合が可能となる．任意のシステムが自律サブシステムを内蔵した GW を介して容易に統合，離脱が可能である[25]．

図 3.1 のリカーシブモデルにより，複数の自律分散システムを統合したシステムは，それ自身が自律分散システムとなる．個別のミッションを遂行するために構築されたシステム S_1 と S_2 を統合接続するとき，これらを自律分散サブシステムであるゲートウェイ GW_{12} を介して互いに接続する．GW_{12} はシステム S_1 の 1 サブシステムでもあり，同時にシステム S_2 の 1 サブシステムにもなる．GW_{12} はシステム S_1 のデータフィールド DF_1 を流れるデータの内容コード(CC)をもとに選択し，システム S_2 の DF_2 に透過したり，あるいは反対に S_2 の DF_2 から S_1 の DF_1 にデータを透過したりする．このことは GW_{12} 自身に自律制御プロセッサー（ACP）ミドルウェアが組み込まれ，各システムの 1 サブシステムとして機能することにより，可能となる．S_1 のアプリケーション・オブジェクト（AO）に必要とされる内容コードは前もって AO 自身により GW_{12} に登録される．この時接続されているアプリケーション・システム S_2 全体，及び GW_{12} は S_1 にとっては 1 つの自律サブシステムとみなせる．同様にシステム S_1 全体，及び GW_{12} は S_2 にとって，1 つの自律サブシステムとみなせる．

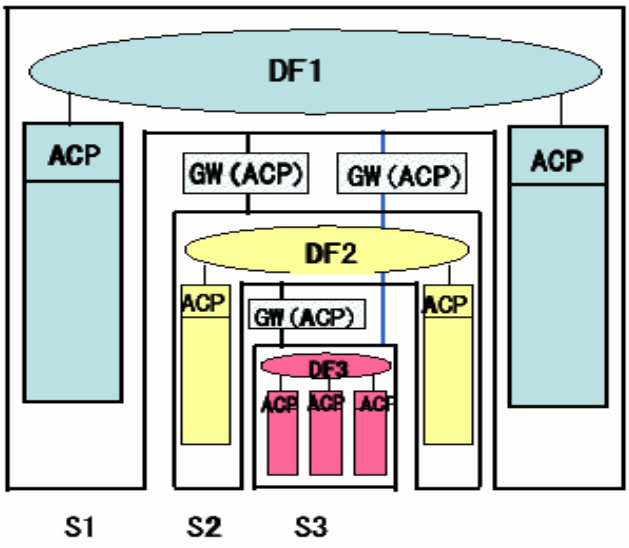


図3.1　リカーシブモデル

以上において GW_{12} は S_1 の AO に必要となる，事前に登録された内容コードをもつデータを S_2 より選択的に受信し，S_1 に透過する．さらにシステム S_2 が GW_{23} を介してアプリケーション・システム S_3 に接続されているとき，S_2 と S_3 を接続している GW_{23} 及びアプリケーション・システム S_3 は，S_2 の 1 つの自律サブシステムとなる．GW_{23} は S_2 の AO に必要な内容コードをもつデータを S_3 より選択し，S_2 にパスする．従って GW_{23} には S_1 及び S_2 の両方の AO に必要な内容コードを登録しておく．

このように ADS リカーシブモデルにおいては，自律分散システムは ACP を内蔵した自律分散サブシステムである GW によって，異種システムを統合することにより，接続されたシステム自体が 1 サブシステムとして取扱えるので，統合に伴うパラメータの再設定や，システムの再稼働などの複雑な作業は不必要であり，容易にシステム統合が可能となる．任意のシステムが自律サブシステムを内蔵した GW を介して容易に統合，離脱が可能である．

3.1.2　異種統合型情報サービスシステムアーキテクチャ

(1)　アプリケーションモデル

図3.2に示すような無線通信式ＩＣカードを使った異種統合型情報サービスシステムのアプリケーションモデルを考察する．このシステムはＩＣカードと端末，駅サーバ，センターサーバがネットワークで統合されている自律分散システムである．

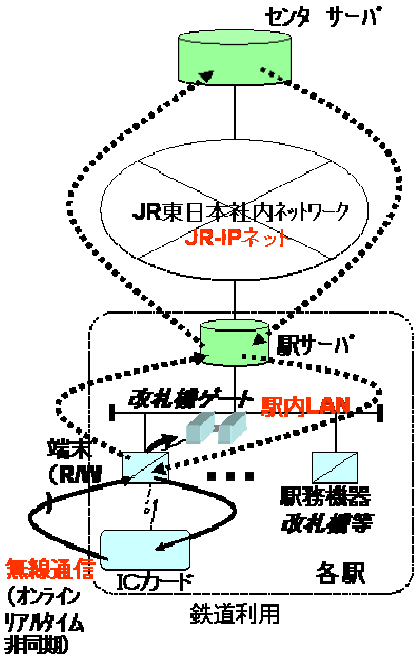


図3.2　アプリケーションモデル（鉄道ICカード乗車券システム）

ＩＣカードと端末間は無線通信によりオンラインリアルタイムで高速な処理が行われる「制御システム」である．IC カードは識別コード（CC: Content Code）が付いたデータをDF（Data Field）にブロードキャストし，端末（改札機）はセレクトしてデータの収集・処理を行っている．

各端末と駅サーバ間は駅内LAN，駅サーバとセンターサーバはWANで結ばれており，改札機からの高負荷トランザクション（2400万トランザクション／日；2006年7月）を処理する「情報システム」である．各端末・サーバ間は，データフィールドを介して自律分散処理をしている．駅レベルでは，駅サーバからDFにデータをブロードキャストしており，各端末（自動改札機，自動券売機など）はセレクトしてデータを受け不正カードチェックや運賃計算などの処理をしている．また，各端末は自律して稼動しており，一部の端末が故障しても他の端末には，その影響を及ぼさないシステムとなっており，データの信頼性を確保している．

このアプリケーションシステムモデルの特徴は高速処理をする「制御システム」とデータの信頼性を確保して高負荷トランザクションを処理する「情報システム」とが統合されたシステムである[56]．

(2)　システムモデル

前項の「異種統合型情報サービスシステム」のシステムモデルを図3.3に示す．

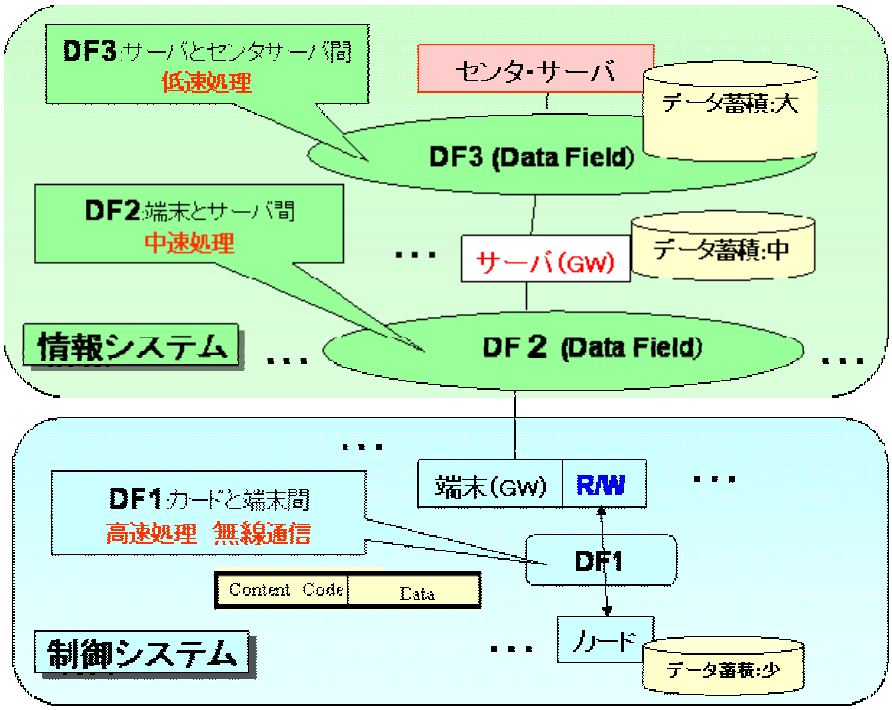


図3.3　異種統合型情報サービスシステムのモデル

「異種統合型情報サービスシステム」の課題は制御システムと情報システム統合による異種のニーズが混在しており，更にシステムの動的な変動（故障，拡張，改修など）環境下でもその稼動を保証するシステムであることが求められている．異種のニーズは制御システムのリアルタイム性と情報システムの信頼性であり，異種の手段は無線システムと有線システムなどである．このような異種のシステム，手段，ニーズなどを統合してもシステムのフォルトトレランス性，拡張性，保守性確保し，システムの自律性，安定性を確保する必要がある．

このため，本論文では，データフィールド（DF）毎に処理の時間を変化させたDFを提案し,これを「時間差異種データフィールド」と呼び，この「時間差異種DF」によりシステムを構築した．このシステムアーキテクチャ構築技術により，異種ニーズ／異種モード下でシステムのアシュアランス性を満たすことが可能になっている．

図3.3では処理速度が「高速処理」「中速処理」「低速処理」と3種類の処理時間の異なるDF群から構成されている．この異種統合型システム自身も自律分散システムであり，前節で述べた「異種統合型情報サービスシステム」である．このように異種統合型情報サービスシステムはシステムニーズに合わせた「時間差異種DF」により構成されているという特徴がある．このシステムアーキテクチャにより，高速性と信頼性という異種ニーズの両立を実現している．

3.1.3 時間差異種データフィールド

前項で述べた異種統合型情報サービスシステムのアーキテクチャには異なる処理速度のDFである「時間差異種DF」技術が提案され，これがこのシステムの特徴となっている．以下に従来のDFと時間差異種DFとの違いを述べる．

(1) 従来のデータフィールド

２．３．３項で異種システムの共存技術としてゲートウェイでデータフィールドを接続することによるシステム共存の方法について述べてきた.図3.4に示すようにサブシステムで処理されたデータは処理直後にDFにブロードキャストされる．さらにそのデータはゲートウェイ（GW）を介して直ちに他のDFに送信される．このように従来のDFではGWは時間差無しにデータの送信を実施している．

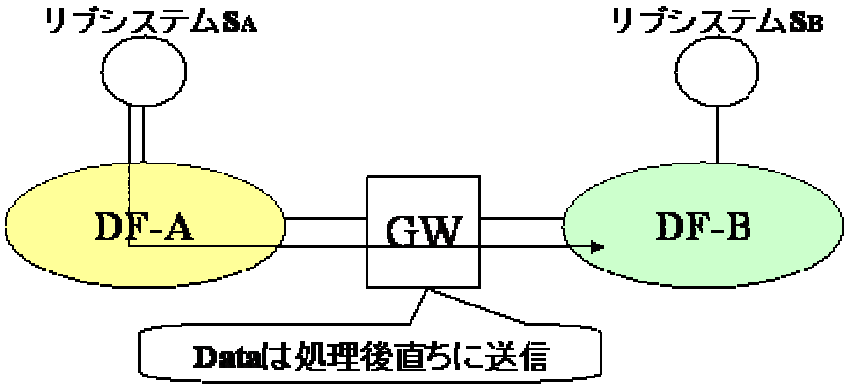


図3.4　従来のデータフィールドとゲートウェイ

(2)　時間差異種データフィールド

従来のデータフィールドでは前述したように，処理直後にデータの送信が行われていた．しかし，システムにおいては，その利用環境やニーズにより時々刻々と状況が変化している．例えば，ネットワークの輻輳の度合い，障害発生の有無，サブシステム間の処理速度（バッファの大きさ）の違い，等による状況の変化が考えられる．これらの状況の変化に合わせてシステムを柔軟に対応させ，システムの安定稼動を図る技術として時間差異種DFを提案した．この時間差異種DFは，図3.5に示すように，各ゲートウェイ（GW）が状況を判断した上で時間差を持ってデータの送信を行うものである．

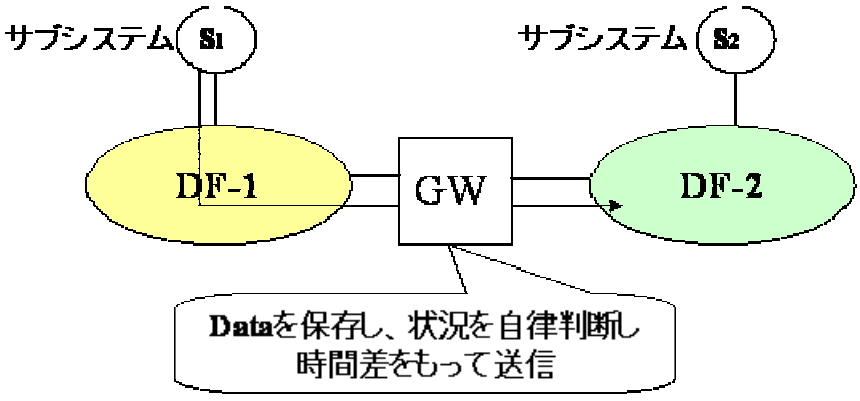


図3.5　時間差異種データフィールド

3.2　　アシュアランス性の定義

第２章で述べたように，アシュアランス性とは異種性と適応性の２つの要素を考慮したシステムに適用される技術の総称であり，この２つの性質を併せ持つシステムをアシュアランスシステムという．

異種統合型情報サービスシステムにこの２つの性質を対応させたとき，図 3.6 に示すように，異種性とは，システムのニーズレベルの異なる制御／情報システムなどの異種システムの共存と高速性，高信頼性という異種ニーズの共存が該当する．

また，適応性としては，時間的に変化する環境の中で段階的にシステムを，統合・結合・更新していくという段階的システム構築が該当する．このような性質をうまく利用してシステムの稼動を保証することが，異種統合型情報サービスシステムにおけるアシュアランス技術である．

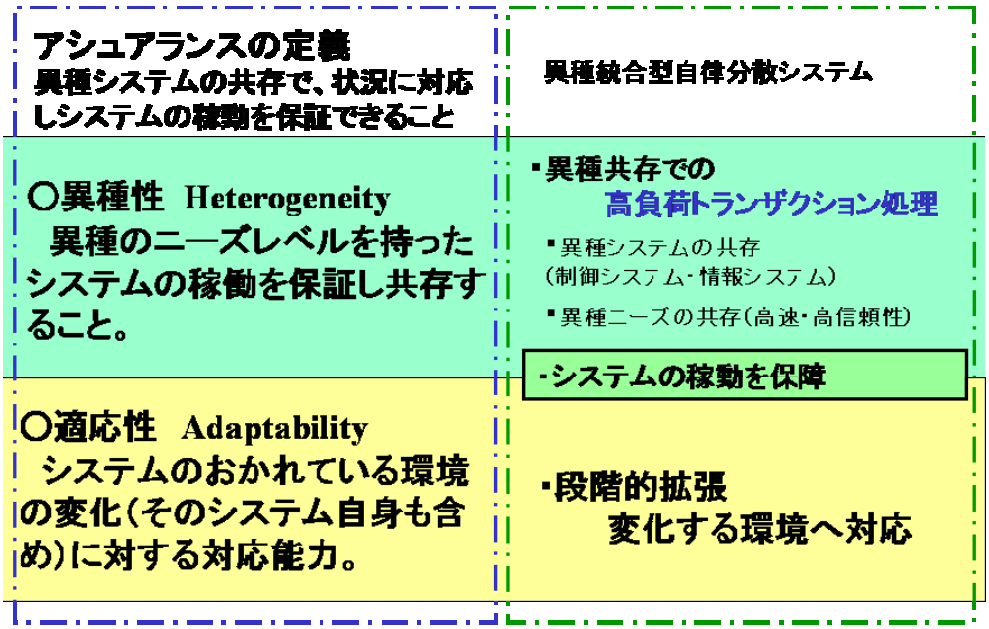


図 3.6　アシュアランス技術と異種統合型情報サービスシステム

3.3　自律分散高速処理技術

異種統合型情報サービスシステムにおいては「制御システム」リアルタイム処理の条件下で「処理の高速性」と「サービスの継続性」が求められているが，サブシステムレベルでは情報（データや処理法）が不完全だったり，情報が局所的だったりするという課題がある．この課題を解決し，「高負荷トランザクション処理の高速化」という目的を達成するため，異種統合型情報サービスシステムのアーキテクチャを使った高速処理技術として「自律連携処理技術（自律分散アルゴリズム)」を以下に提案する．

また，従来の高速処理技術である「並列／分散処理技術」,「最短経路アルゴリズム」についても以下に述べる．従来技術との比較から，今回，提案した「自律連携処理技術（自律分散アルゴリズム)」について，基本理論，システムのモデル化，情報と処理の局所性など，自律連携処理技術の理論を解説する．

3.3.1　従来の高速処理技術

(1)　並列／分散処理技術

処理を高速化するための技術として分散処理技術がある．これは複雑な処理，大規模な処理を複数に分割して処理し，その結果を結合して実現する．大きく分けて分散実行と分散開発の二つの技術がある．

分散実行は，複数のコンピュータで部分処理し，処理結果を通信路で統合して全体処理を行う技術である．利点としては，高信頼，故障部分以外稼動可能であることなどがある．また，分散開発は各々に分散したサブシステムを１つずつ，又は1部並行して，段階的に構築する開発体系で，主にソフトウェアの開発における仕組みである．利点としては，複雑・大規模なシステムの開発が容易であること，仕様が全て明確でなくても，部分毎に積上げ，徐々に開発可能であること，稼働中でも拡張・変更が容易であることなどが上げられる．

他にも並列処理技術がある．これは，コンピュータで複数の処理を同時に進行させる事により，効率の向上をはかる技術のことである．基本的には，一つの処理目標を複数の演算装置に分割して物理的な並列化を行う場合を指す．ただしマルチスレッドなどで仮想的な資源の多重化を行い，演算装置の待ち時間を他のタスクに利用する場合も並列処理と呼ぶ場合がある．並列処理は，大きくメモリ共有型とメモリ非共有型に大別される．代表的なものにグリッド・コンピューティングがある．これは，インターネットなどの広域のネットワーク上にあるコンピュータ資源（記憶資源も含む）を結びつけひとつの複合したコンピュータシステムとして処理を行

う仕組みである．これらの技術は何れもハードウェア資源を活用した技術である．

ソフトウェアによる分散処理の例として，遺伝的アルゴリズム（Genetic Algorithm：GA）における代表的な並列/分散モデルがある．この 遺伝的アルゴリズムにおける並列/分散化は大きく 2 つに分類される．1 つはアルゴリズムとして探索性能の向上を目指したモデルの分散計算モデルと，並列化による高速化を図るモデルの並列実行モデルとに分類される[32]．

また，他にもエージェントを使ったアルゴリズムにより，高速処理する手法が提案されている．たとえば，複数の自律ロボットが分散協調して作業に当たる場合，各ロボットの移動・行動計画立案，センシング情報の統合などのためには，複数の制約問題を解く必要がある．しかもこれらの制約問題は，センサーやロボットの限られた計算資源で，分散した状態のまま，実時間で解かなければいけない．この問題は分散制約充足問題に一般化することができる．分散制約充足問題とは，制約充足問題の変数，制約が複数のエージェントに分散されたものである．分散制約充足問題を解き，高速処理を実現するための手法である[33]．

以上のように，従来の高処理技術は次のような特徴がある．

① ハードウェアの分散により高速処理を実現する技術
② ソフトウェア（アルゴリズム）はハードウェアの状態に合わせて，あらかじめ決められた方法で高速処理可能としたものである．
③ アルゴリズムは統計や確率的手法を用いており，シミュレーションなどの分野での活用が有効である．

このため，高負荷トランザクションを高速処理可能とする分野での研究はまだ進んでいない．

(2)　最短経路問題のアルゴリズム

前項で高速処理についての従来の技術を述べた．異種統合型情報サービスシステムを高速処理するためには処理技術と併せてトランザクション処理の内容，そのもののアルゴリズムを工夫することにより高速化が可能になるケースがある．例えば，ネットワークの効率の良さを求めるために，ノード間の最短経路長を求める必要がある場合などの「最短経路を求めるアルゴリズム」がそうである．この分野の技術についても以下に調査した．

最短経路長を求めるアルゴリズムには，全ての 2 点間の最短路・最短距離を求める方法であるウォーシャル・フロイド法（Warshall-Floyd 法）[64]，特定の 2 点間の最短路・最短距離を求めるダイクストラ法（Dijkstra 法）[65]などがある．

① ウォーシャル・フロイド法（Warshall-Floyd 法）

全ての 2 点間の最短路・最短距離を求める方法にウォーシャル・フロイド法（Warshall-Floyd 法）がある．まず，点 i から j に向かう枝がある場合はその距離 d_{ij} を，ない場合は∞とした直接距離行列を作成する．次に，i から j に別の 1 点を経由した場合（枝 2 本）の最短路・最短距離を求め，距離行列を更新する．その際，次に経由する点を p_{ij} として記憶しておく．更新された距離行列を用いてこの操作を再度行えば，枝 4 本まで使った最短路・最短距離が求まる．この操作を $2^h \geqq n-1$ となるまで h 回繰り返せば，最終的な最短経路・最短距離が求まる．

手順をアルゴリズムとして以下に示す．

1. $p_{ij}=j(i \in N,\ j \in N)$，$h=1$ とおく．
2. 全ての $i,j,k \in N(k \neq i,j)$ に対し，

   $d_{ij} > d_{kj}$ であれば，$d_{ij}=d_{ik}+d_{kj}$，$p_{ij}=p_{ik}$ とする．
3. $2^h \geqq -1$ であれば終了．
4. $h=h+1$ として 2 にもどる．

② ダイクストラ法（Dijkstra 法）

ダイクストラ法（Dijkstra 法）は特定の点からの最短路・最短距離を求める方法である．n 個の点（集合 N）からなるネットワーク上で，枝(i, j)の距離が d_{ij} で与えられている．始点 s から点 j までの最短距離を v_j その路の直前の点を p_j とする．始点 s から v_j の小さい順に点 j へ枝を延ばしていけば，このネットワークの最短経路を発見できる．

以上のような方法で 2 点間の最短経路と最短距離は求められる．しかし，本論文では，鉄道ＩＣ乗車券システムをアプリケーションモデルとしており，単に 2 点間の最短経路と最短距離を求めることではなく，定期券などの特殊な乗車券と組み合わせた時でも，最安運賃計算を端末のレベルでいかに高速処理するか，ということである．このような例はインターネットでのサービスを受けるケースで割引権限（一定の条件で割引や無料のサービスを受けられる権利）を有する場合，その権限を利用した場合と利用しない場合でサービス料金がどちらの場合が少ないかを選択する，という問題を端末のレベルで高速処理する場合などが同様のケースである．

3.3.2　　自律分散高速処理技術

本項の「自律分散高速処理技術」は各ノード（サブシステム）が持つ情報／処理が局所的であるという条件下で，処理アルゴリズムを工夫して高速性を達成する方法として「自律連携処理技術（自律分散アルゴリズム)」を提案する．以下に自律連携処理技術の基本理論とそのモデル化，そして，システムモデルによる自律分散アルゴリズムとそのフローチャート，情報／処理の局所性，最適処理時間などについて述べる．

(1)　自律連携処理技術の基本理論

図3.7に示すように，端末でデータD_1をシステムScでD_2に変換する場合の処理時間をt_cとする場合に，この処理時間を高負荷トランザクション時の端末の視点で高速化する方法について考察する．

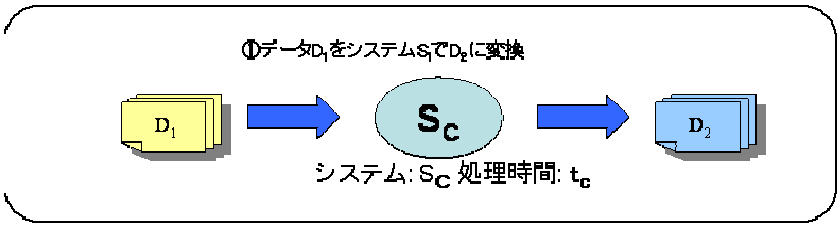


図3.7　通常の処理

高速処理のための方法として，図3.8に示すように処理を前処理と後処理に分割し，各々の処理結果を自律連携処理し図3.7の通常の１回処理と同様の結果を得る技術を提案する．これを「自律連携処理技術（自律分散アルゴリズム)」と呼び，処理のアルゴリズムを「自律分散アルゴリズム」と呼ぶ．トータルの処理時間はt_cより少なくなるとは限らないが，個々の端末レベルでの処理間の短縮は可能である．

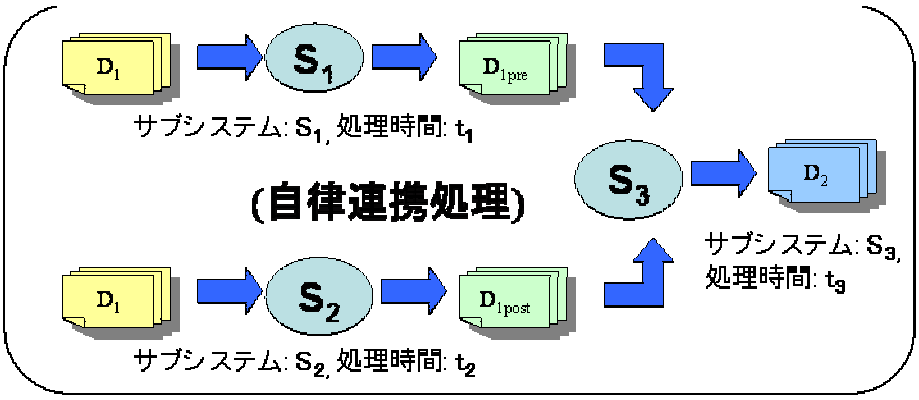


図 3.8　自律連携処理技術（自律分散アルゴリズム）

(2)　モデル化

前項の基本理論を利用し，ノードでの高速処理を可能にする「自律連携処理技術（自律分散アルゴリズム)」のシステムの基本モデルを図 3.9 に示す.

システム全体をＳとし，Ｓはノードの集合 Ｕ とノード間の最短経路の集合Ｌから構成される．ここで，ノード間相互を結んだものを「リンク」，ノード間の最短ルートを「パス」と呼び，その値を「パス値」（|Ｌ|＝１）と表す．また，パス値が特別のルール（|Ｌ|＝α（U_{PX}，U_{Py}））であるノードの集合を集合Ｐと定義し，その集合を「ドメイン」と呼ぶ．詳細は以下のとおりである．

① ノードの集合をＵとし，Ｕ={U_1，U_2，U_3，…，U_n}と表す．集合Ｕは ｎ 個のノードから構成されている.
② 各ノードは 1 個以上の他のノードと繋がっており，ノード間の最短経路の集合をＬとする.
③ ノードU_iとU_jの最短経路はL_{ij}と表す.
　最短経路の値（絶対値）を$|L_{ij}|=l_{ij}$（パス値)とする.
④ ノードの集合Ｕとリンクの集合Ｌを合わせたものを システムＳ（Ｕ，Ｌ）とする.
⑤ 各ノードは自律分散処理を行う.
・ 自ノードは全てのノードまでの「パス値：l_{ij}（ローカル情報)」を持つ.
・ 全ノード間相互のパス値は持っていない.
⑥ 集合Ｕの中で「パス値が特別のルール，|Ｌ|＝α（U_{PX}，U_{Py}）（エリア内の各ノードの関係はα（U_{PX}，U_{Py})で表される.」であるノードの集合をＰとし，Ｐ={U_{P1}, U_{P2}，…，U_{Pk}} で表し，「ドメイン」と呼ぶ.
⑦ ドメイン集合Ｐはｋ個のノードから構成される.
　但し，Ｐ⊂Ｕ，　ｎ>ｋ である.

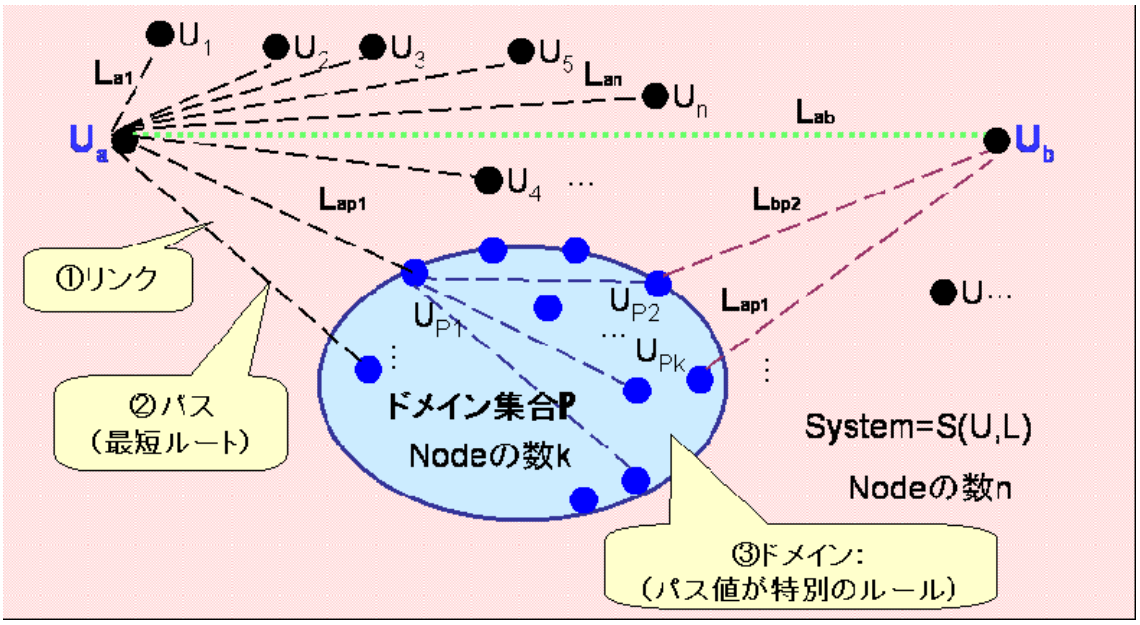


図 3.9　自律連携処理技術のシステムモデル（基本）

また，図 3.10 に一般モデルを示す．一般モデルではパス値が異なるドメイン集合 P1，P2，P3 ... Pi が複数存在する．

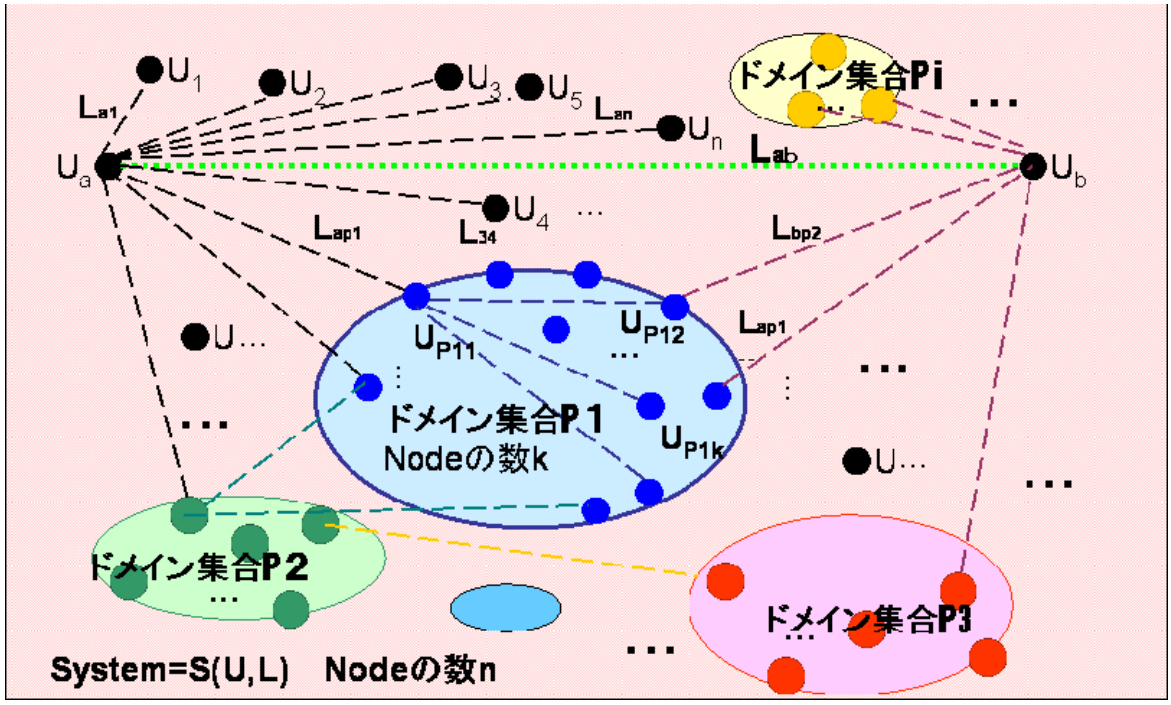


図 3.10　自律連携処理技術のシステムモデル（一般）

(3)　自律分散アルゴリズム

図 3.9 のシステムモデルのノードでの高速処理を可能にする「自律連携処理技術（自律分散アルゴリズム)」について述べる.

ノードの処理時間とノード数，トランザクションとの関係を表 3.1 に示す．例えば，ノード数が多いと，ノード間の組合せが多くなり，処理は複雑化し，結果として，処理時間は長くなる．また，トランザクションが多い場合は，処理が停滞し，処理時間は長くなる.

表 3.1　処理時間とノード数・トランザクションの関係

		処理時間
ノード	多い	長い
	少ない	短い
トランザクション	多い	長い
	少ない	短い

このような特徴があるシステムにおいて，トランザクションの発生特徴に応じた，処理方式（分割・連携と統合一括）の最適解を求めるのが，自律連携処理技術（自律分散アルゴリズム）である.

①基本モデルのアルゴリズム

図 3.9 に示すように「ユーザ」がノードＡからノードＢまでのサービスを受け，その処理をする場合（例えば，乗車による運賃，ネットワークサービスの付加料金など）にドメイン集合Ｐ（パス値=特別のルール）を経由する場合としない場合のケースを考察する．ただし，ユーザは始点ノードでの処理データを終点ノードに伝達可能（例えば，ＩＣカードやインターネットを利用するなどの方法）と仮定する.

図 3.9 の「ノード A:U_a からノード B:U_b までの最小値を求めるアルゴリズム」を以下に示す.

前提

システムＳ（Ｕ，Ｌ）．Ｕ：ノードの集合（ｎ個）．Ｌ：ノード間の最短経路の集合．

ドメイン集合Ｐ（ただし集合Ｐ内ノード間のパス値：｜Ｌ｜＝０），P∈U.

但し，min(X):X の最小値．Min(X,Y):X と Y の最小値

Step1：ノード U_a からドメイン集合 P 内の各ノードまでの最短経路のパス値の最小値を求める．　　$\min(|L_{a\text{-}P}|)$

$|L_{a\text{-}P}|$はノード U_a から集合Ｐ内のノード U_{Pi} (i=1~k)までの最短経路のパス値．

Step2：ノード U_b からドメイン集合 P 内の各ノードまでの最短経路のパス値の最小値を求める．　　$\min(|L_{b\text{-}P}|)$

$L_{b\text{-}P}$ はノード U_b から集合Ｐ内のノード U_{Pj}(j=1~k)までの最短経路のパス値．

Step3：　Step1 と Step2 の値を連携し

$\min\{\min(|L_{a\text{-}P}|) + \min(|L_{b\text{-}P}|), |L_{a\text{-}b}|\}$　を求める

②一般モデルのアルゴリズム

図 3.10 の一般モデルにあるように，特別なパス値のルールが異なるドメイン集合 P1，P2,P3...Pi が複数存在する場合に，「ユーザがノード U_a からノード U_b まで自由に移動する場合の最小パス値を求めるアルゴリズム」を以下に示す．

前提

システムＳ（Ｕ，Ｌ）．Ｕ：ノードの集合（ｎ個）．Ｌ：ノード間の最短経路の集合．|L|：最短経路のパス値，Ｐ：ドメイン集合［(P1，P2，P3，...(q 個)］，但しドメイン集合 Pi のノードの数を k_i(i=1~q)，Pi 内のノード U_{Pij} と U_{Pik} 間のパス値 $|L_{pij-pik}|$ は特別なルールで決定する．P∈U，$n > k_i$，$k_i \geqq k$，$k_i \geqq j$.

また，ドメイン集合 Pi 内の $\min\{|L_{pij-pik}|\}$，…及び各ドメイン集合間の最短経路のパス値 min(P1,P2)，min(P1,P3)，min(P2,P3)，…は既知とする．

この場合，ノード U_a からドメイン集合 Pi を経由して，ノード U_b まで到達するルートの組合せは $\sum_{r=0}^{q} {}_qp_r$ 個ある．

但し，min(X):X の最小値，min(X,Y):X と Y の最小値である．

Step1：ノード U_a からドメイン集合 Pi 内のノードまでの最短経路のパス値の最小値を，ルートの組合せ（$\sum_{r=0}^{q} {}_q p_r$ -1 個）について求める．

min（$|L_{a-P1}|$），min（$|L_{a-P2}|$），…

但し，$|L_{a-P1}|$ はノード U_a から集合 P1 内のノード U_{P1i}（i=1~k_1）までの最短経路のパス値である．同様に $|L_{a-P2}|$ はノード U_a から集合 P2 内のノード U_{P2i}（i=1~k_2）までの最短経路のパス値である．

Step2：ノード U_b からドメイン集合 Pi 内のノードまでの最短経路のパス値の最小値を，ルートの組合せ（$\sum_{r=0}^{q} {}_q p_r$ -1 個）について求める．

min（$|L_{b-P1}|$），min（$|L_{b-P2}|$），…

但し，$|L_{b-P1}|$ はノード U_b から集合 P1 内のノード U_{P1j}（j=1~k_1）までの最短経路のパス値．以下同様．

Step3：Step1 と Step2 の値を連携し，全てのルートの組合せ（$\sum_{r=0}^{q} {}_q p_r$ 個）から最短ルートパス値の最小値を求める

$$\min\{\min(|L_{a-Pi}|) + \min\{|L_{pij-pik}|\} + \min(|L_{b-Pi}|), \cdots$$

$$\min(|L_{a-P1}|) + \min\{|L_{p1j-p1k}|\} + \min(P1,P2) + \min\{|L_{p2j-p2k}|) + \min(|L_{b-P2}|), \cdots, |L_{a-b}|\}$$

③基本モデルのフローチャート

また，図 3.9 の基本モデルを使って，自律連携処理について図 3.11 に「一括統合処理」の場合，図 3.12 に「分割・連携処理」の場合のフローチャートを示す．

個々の処理における時間短縮が高負荷トランザクションの情報サービスシステムでは重要である．分割・連携処理の場合の前処理時間：t_{pre} と後処理時間 t_{post} は統合一括処理の場合の処理時間 t_c と比較すると，「$t_c > t_{pre} + t_{post}$」とは限らない．しかし，$t_c > t_{pre}$，$t_c > t_{post}$ である．このため，各ノードの高速処理が実現している．

（但し，図 3.11 の L は全ノード間相互の最短経路のデータ，図 3.12 の $L_{a\text{-}P}$ はノード U_a から全てのノードへの最短経路のデータ，$L_{b\text{-}P}$ はノード U_b から全てのノードへの最短経路のデータである．）

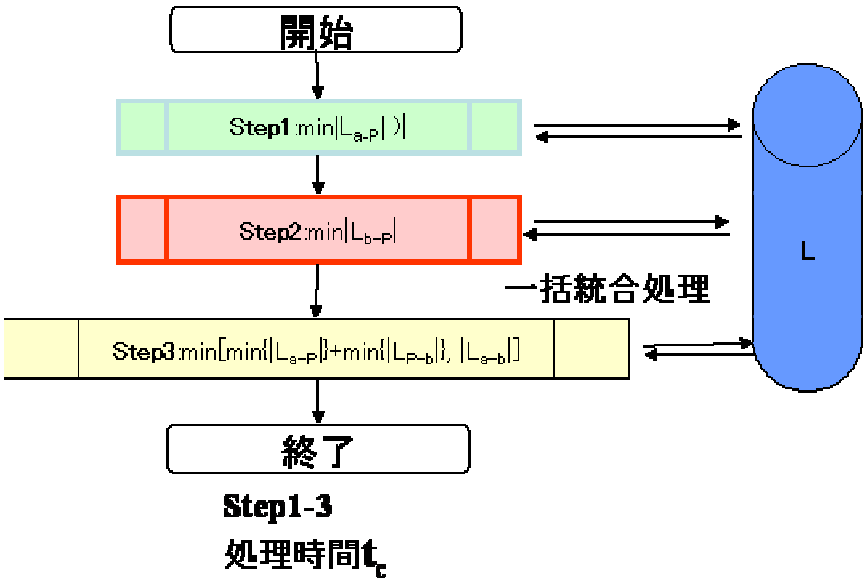


図 3.11　一括統合処理の場合のフローチャート

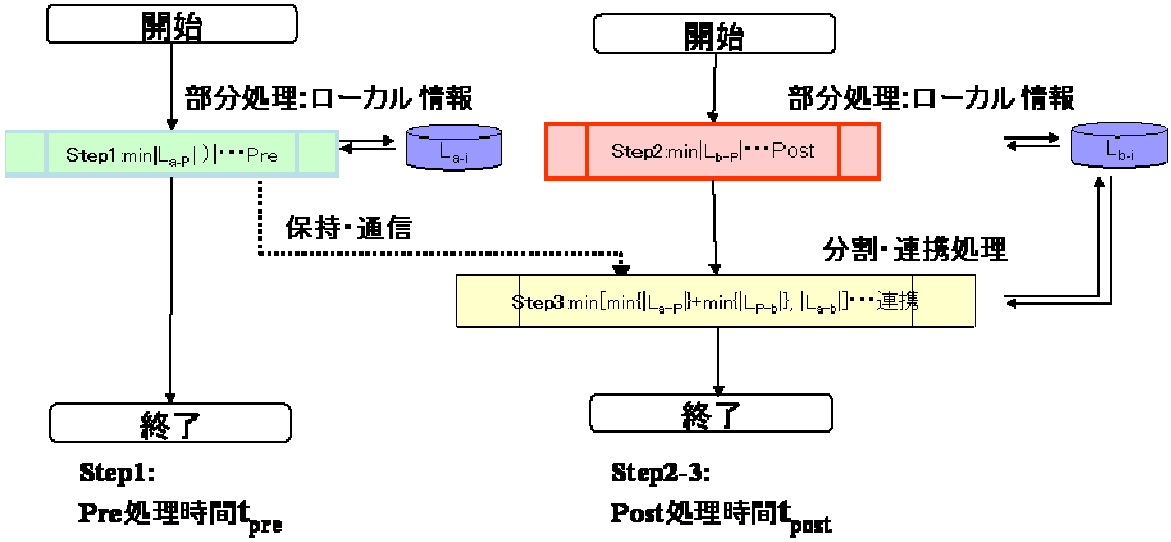


図 3.12　分割・連携処理の場合のフローチャート

(4)　ノード情報の局所性

本項では，ノードの特徴である，情報の局所性と処理の局所性について述べる．この場合，重要な条件は，各ノードは自律分散システムであり，ローカル情報（自ノードから各ノードまでの最短経路のパス値）のみを保有していることと，ノード

での処理はリアルタイム処理で高速処理することである．何故ならば，もし，各ノードがシステムＳ全体の全てのデータを保有していれば，最短経路の値は求めることが可能であるが，この場合はノードでの処理量が多くなり，ノードでの高速リアルタイム処理は困難である．前項でも述べたが，高負荷トランザクションの情報サービスシステムにおいてはノードでの高速処理は必須である．このため，全体の処理を各ノードでのローカル情報による分割処理し，後で連携することとした．

各ノードには，固定情報としての「ローカル（Local）情報」と，処理の都度与えられる，変動情報としての「インプット（Input）情報」がある．

具体的には，図3.13に示すように，ノードは「自ノードから全てのノード間」のパス値を持っており，この情報は各ノードの持つ「ローカル（Local）情報」であり，各ノード固定の情報である．

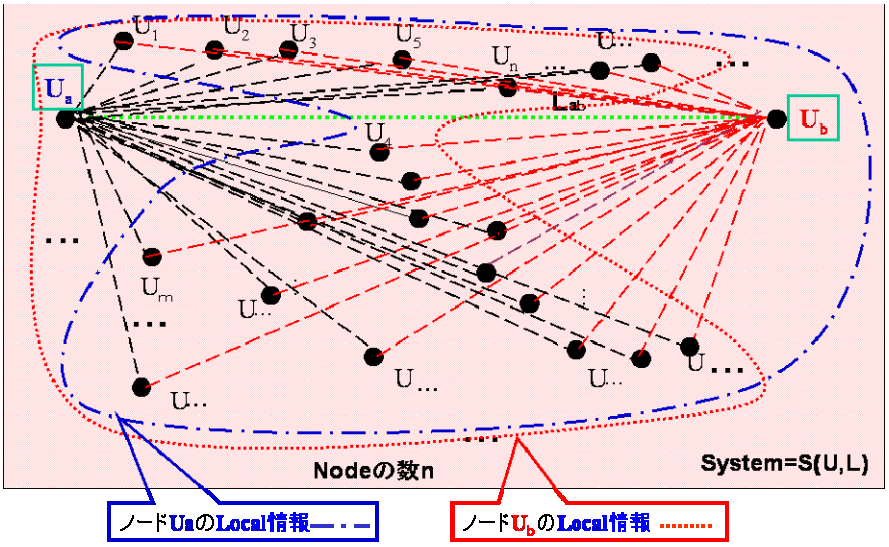


図 3.13　ノードの Local 情報

また，図 3.14 に示すように，ドメイン情報はノード U_a または U_b での処理時に取得するデータで，「インプット（Input）情報」と呼ぶ．このインプット情報はノードでの処理の都度与えられる，変動情報である．

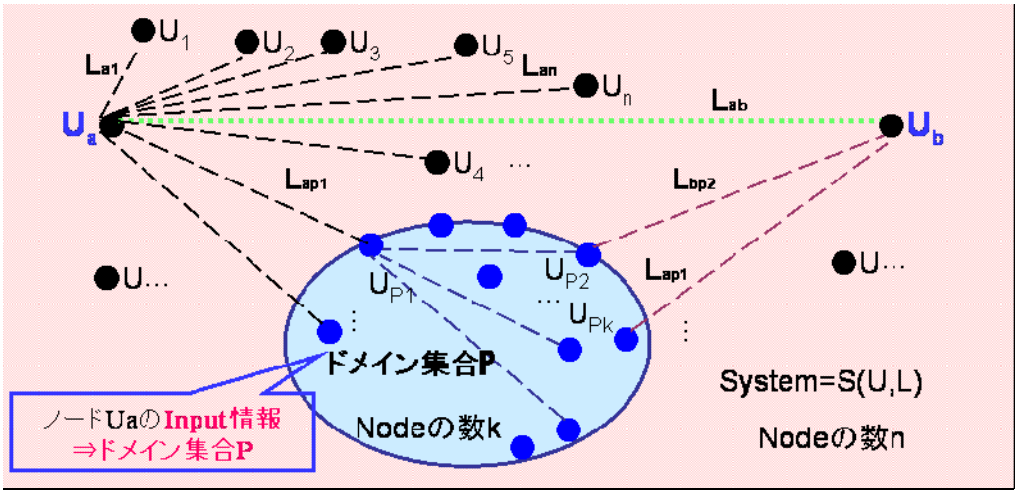


図 3.14　ノードの Input 情報

前述の Local 情報と Input 情報により，各ノードで高速リアルタイム処理をし，この処理を自律的に分割・連携する技術が「自律連携処理技術」である．

(5)　ノードの処理の局所性

図 3.15 は「U_a から U_b までの最短パス値を処理する場合」の例である．

ユーザは自由に移動しており，U_a 利用後はどのノードへ向かうかは予測不可能である．このため，各ノードは前項で述べた Local 情報と Input 情報により，自律して可能な部分についての分割処理を行う．これを「ノードの処理の局所性」と言う．このようにして，ユーザが U_b まで移動した場合のノードの処理については，「ノードの処理の局所性」による「U_a と U_b の分割・連携処理」の場合と「U_b 一括統合処理」の場合が考えられる．この場合，U_a から U_b までの最短経路のパス値はドメイン集合 P を経由した場合と経由しない場合のパス値を比較し少ない方を求めるものである．

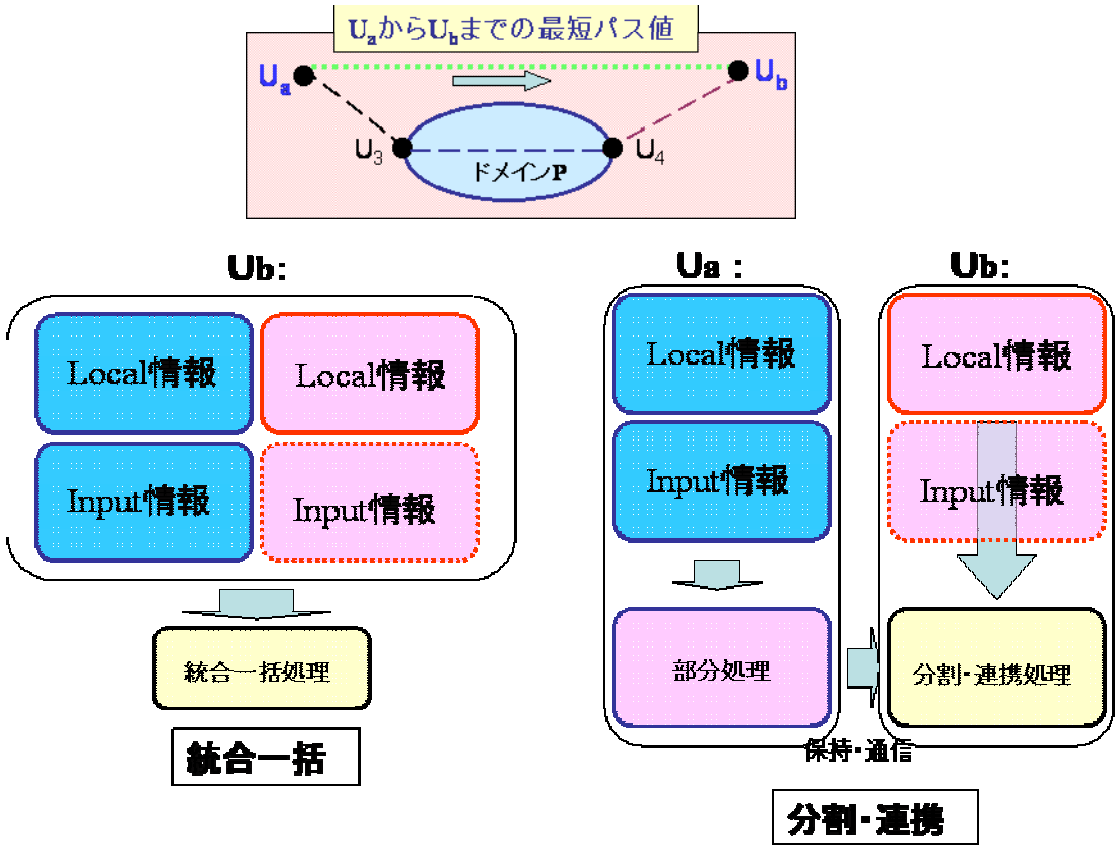


図 3.15　U_a から U_b までの最短パス値を処理する場合の例

(6)　処理時間の最適値

図 3.15 の例から処理時間の最適値を求める論理式を考察する．

前項でも述べたように，ノードの処理については，「U_a と U_b の分割・連携処理」の場合と「U_b 一括統合処理」の場合がある．各ケースについて処理時間を求め，その最小値が最適処理時間と処理方式である．この場合，ノードでの処理時間はトランザクション数，ノード数，ノードにおける Local 情報の量及び Input 情報の量などの関数であるとする．

一般的に処理時間にはデータのチェックや計算などの処理時間と通信などの付帯処理時間がある．ノードでの基本的な処理時間を以下のように定義する．

基本処理時間 $\mathbf{T = T_1 + T_2 + T_0}$

$\mathbf{T_1}$：データチェック処理時間

$\mathbf{T_2}$：データ計算処理時間

$\mathbf{T_0}$：通信などの付帯処理時間

① 「U_a と U_b による分割・連携処理」の場合

$T_{AB} = T_A + T_B$

$= T_{1A} + T_{2A} + T_{0A} + T_{1B} + T_{2B} + T_{0B}$

T_{AB}：処理時間の合計

T_A：U_a での処理時間

T_B：U_b での処理時間

$T_A = f(p, j, I_{sa}, I_{ia})$

$T_B = f(p, j, I_{sb}, I_{ib})$

I_{sa}（I_{sb}）：ノード U_a（U_b）の Local 情報（自ノードから全ノード間の情報）

I_{ia}（I_{ib}）：ノード U_a（U_b）の Input 情報（ドメインの情報）

p:トランザクション数

j:ノード数

② 「U_b による一括統合処理」の場合

$T_B' = T_{1B}' + T_{2B}' + T_{0B}'$

T_B'：全て U_b で処理した時間

$T_B' = f(p, j, I_{sb}', I_{ib})$

I_{sb}'：ノード U_b の Local 情報（全ノード間相互の情報）

I_{ib}：ノード U_b の Input 情報（ドメインの情報）

③ 処理条件

ⅰ）処理時間

また，ノード（高速リアルタイム処理）の処理時間はリアルタイム処理時間以下となる必要がある．

$T_A \leqq T_R$

$T_B \leqq T_R$

$T_B' \leqq T_R$

T_R：ノードのリアルタイム処理時間

ⅱ）Local 情報

Local 情報は①のケースと②のケースで異なり，以下の関係がある．

$I_{sb}' \subset (I_{sa} \cup I_{sb})$

$|I_{sb}'| > |I_{sa}|$

$|I_{sb}'| > |I_{sb}|$

④ 処理時間の最適値

以上から，処理時間の最適値は以下の式で表される．この式からも解るように，$T_A + T_B$は①分割・連携処理であり，T_B’は一括統合処理である．この比較をトランザクション数，ノード数などから行い，まず，最適な処理方式が決まり，更にその場合の最適な処理時間が決定する．

$$T^*(p, j) = \min\{\underline{T_A + T_B}, \underline{T_B'}\}$$

⑤ シミュレーションによる比較

「U_aと U_bの分割・連携処理」の場合と「U_b一括統合処理」の場合の 2 つの場合について，具体的にノードを自動改札機の事例で，処理時間とトランザクションおよび，ノード数との関係をシミュレーション（詳細は 4 章「4.3.2　高速処理のための自律連携処理技術」を参照）する．図 3.16 に処理時間とトランザクションの関係を，図 3.17 に処理時間とノード数の関係を示す．

この結果からも解るように，トランザクション数・ノード数が少ない場合は統合一括処理の方が処理時間は短い．しかし，トランザクションやノードが増加すると分割・連携処理のほうが処理時間は短くなっている．このように，「自律連携処理技術」は高速処理を達成するための最適なシステム分割・連携度合をトランザクションの発生特性に応じ求められることがわかる．

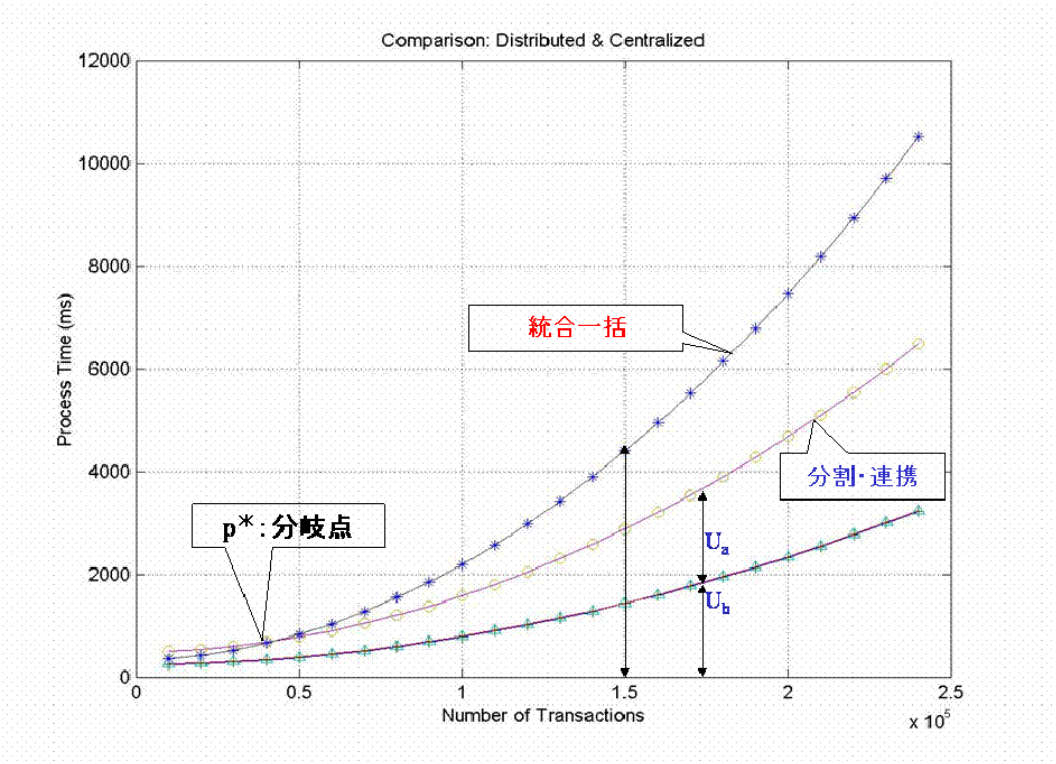


図 3.16　処理時間とトランザクションの関係

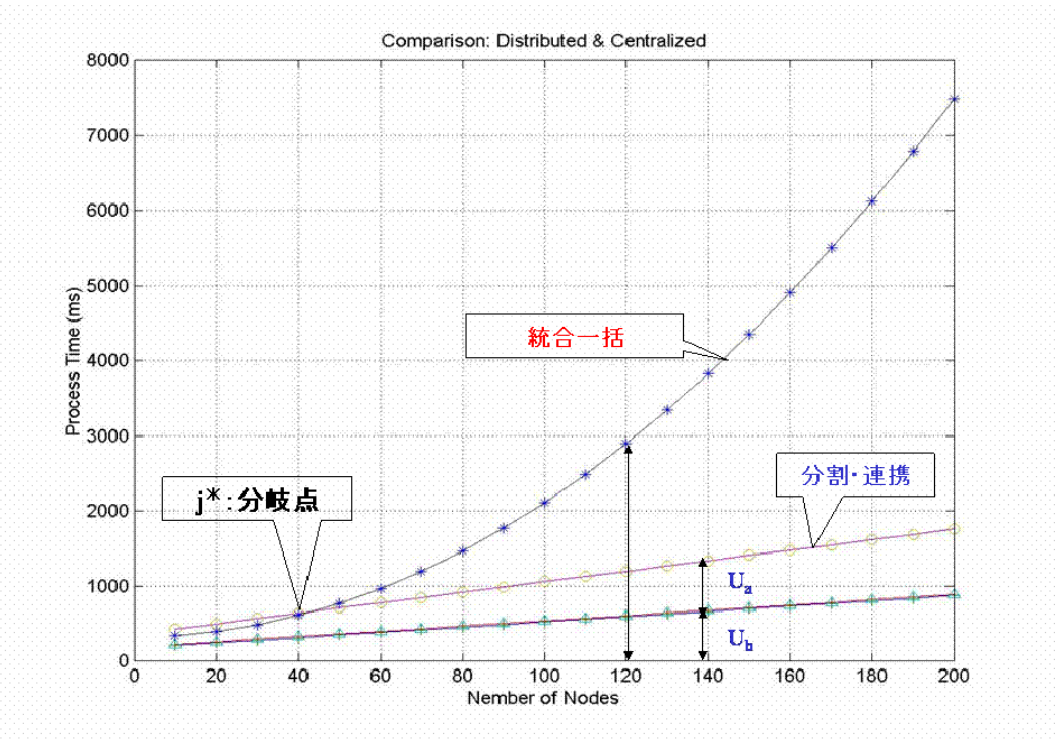


図 3.17 処理時間とノード数の関係

3.4　自律分散整合化技術

高負荷トランザクションを処理する情報サービスシステムと設備機器を含む制御システムとの異種統合型情報サービスシステムでは高速処理と高信頼性の確保に加えて，サービスの継続を保証するアシュアランス性も求められている．本節では異種統合型情報サービスシステムのシステムアーキテクチャにより，システムの高信頼性を確保する技術として「自律分散整化技術」を提案する．この技術は，リアルタイム処理制限下での高トランザクション処理において生じるデータの欠損を回復するため，データフィールドごとにデータの滞留時間を変化させた「時間差異種データフィールド」を構成し，それらの連携によるデータ間の論理的整合化を図る技術である．

本論文で提案した「自律分散整合化技術」について，従来技術との比較から考察するため，以下に従来の整合化技術として，分散システムで複数のデータベースサーバを更新する場合の「二相コミットメント制御」とデータベースのトランザクションデータ障害時のデータリカバリーする場合の「ライト・アヘッド・ログ（ＷＡＬ： write ahead logging）」について述べる．その後，「自律分散整合化技術」について基本理論，システムのモデル化，整合化理論などについて述べる．

3.4.1　従来のデータ整合化技術

(1)　二相コミットメント制御

分散処理ステムでは，複数のコンピュータ内のプロセス間での協調動作により処理が行われる．各コンピュータのプログラム実行状態をプロセス p とする．プロセスは変数の値の集合で与えられる状態を持ち，メッセージの受信，命令実行などの事象の生起により状態が遷移していく．ここでプロセスpiの初期状態をSi^0とする．事象ei^1により状態Si^1に遷移する．このように状態Si^jは，事象 $e\,i^{j+1}$により状態Si^{j+1}に遷移する．前述の処理が複数のコンピュータで行われることから，異なったプロセスでの事象間ではどちらが先に起きたかをという，事象間の因果関係(causality)や各プロセスがどのような状態にあるか等が重要である[30].

このため，分散システムで複数のデータベースサーバを更新する場合は全てのデータベースサーバが更新されたか，または全くされなかったかのいずれかであることを保証する必要がある．このための制御がコミットメント制御である(図3.18).

但し，Tはトランザクションで，「分散データベースシステム内のオブジェクトに対する1つ以上の基本演算の系列の実行状態」を言う．Wはプロセス (ex. write process)を表す.

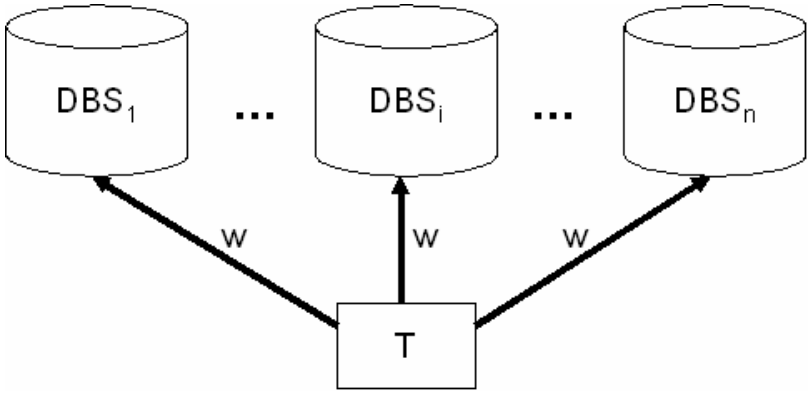


図3.18　コミットメント制御

分散システム上でアトミシティを実現する代表的な技術は二相コミット技術である．トランザクションＴに対して，1つの指揮プロセスCと，n個のデータベースサーバ DBS_1，…，DBS_nをプロセスとして考える．ここで，指揮プロセスCはＴが実行されているクライアントである．各 DBS_iは，障害しても記憶が消滅しない安全記憶としてのログ L_iを持つとする．

二相コミットメントの基本手順は以下のとおりである．

① 指揮プロセスＣは，トランザクションＴの処理をデータベースサーバ DBS_1，…，DBS_nに送信し，各 DBS_iは処理を実行する．
② ＣはＴからコミット処理を受けたとき，全データベースサーバ DBS_1，…，DBS_nに VOTE-REQ を送信する．
③ 各 DBS_iは VOTE-REQ を受信したとき，Ｔをコミットできるならば，ＹＥＳメッセージをＣに送信する．このとき，DBS_iでのＴのログ L_iを記録する．コミットできない場合は，ＮＯをＣに送信してアボートする．
④ Ｃは全データベースサーバからＹＥＳを受信したら，ＣＯＭＭＩＴメッセージを全データベースサーバに送信し，Ｔをコミットする．ある DBS_iからＮＯを受信したら，ＹＥＳを受信した全てのデータベースサーバにＡＢＯＲＴメッセージを送信し，Ｔをアボートする．
⑤ 各 DBS_iは，ＣからＣＯＭＭＩＴを受信したら，Ｔをコミットする．このときログ L_i内のＴの更新データを用いて，データベース DBS_iの更新を行う．また，ＡＢＯＲＴを受信したらアボートする．このときは，更新を行わずに，ログ L_i内のＴの更新データを消去する．

図 3.19 は，全データベースサーバがトランザクションＴをコミットする場合を示す．各 DBS_iで，VOTE-REQ を受信して，ＹＥＳを送信してから，ＣＯＭＭＩ

ＴまたはＡＢＯＲＴするまでの状態が不確定状態である．DBS_iが不確定状態にあるときは，コミットの意思を他に通知しているため，意思に反するアボートは行えない．ＡＢＯＲＴの通知を待つことだけが出来る．ＣがVOTE-REQを送信してから，データベースサーバにＹＥＳまたはＮＯがＣに届くまでを第一相（問合せ送信とその返信），次に，ＣがＣＯＭＭＩＴまたはＡＢＯＲＴをデータベースサーバに送信する相を，第二相（結果の送信とデータベースの更新）とし，このように2つの相から構成されているため二相コミットメントプロトコルと言われる．また，ログを保持していることにより，コミットの処理が途中で失敗しても，どこまで進んでいるかがデータベースで判断でき，トランザクション実行前の状態に戻すことが出来る[35] [36].

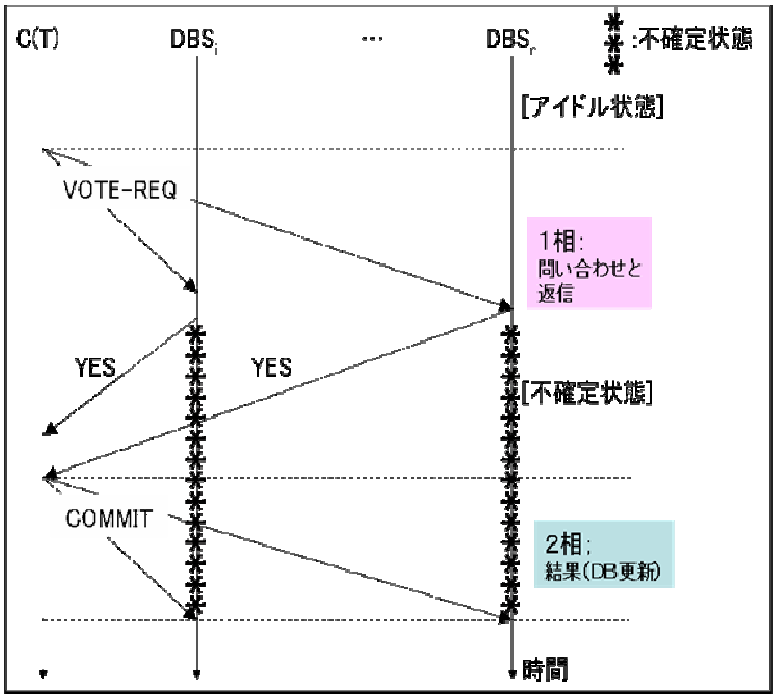


図 3.19　二相コミットメント

このように，二相コミットメント技術はトランザクションデータを複数のデータベースサーバで完全に更新することを保証するという点では優れているが，全てのデータベースにアクセスするため，異種統合型情報サービスシステムのシステムニーズである，「リアルタイム性」や「稼動の継続」については不十分である．

(2)　ライト・アヘッド・ログ（ＷＡＬ： write ahead logging）

WALとは，トランザクションロギング(Transaction logging)を行うための一般的な手法で，データベース更新の際にまずトランザクション・ログを確実に同期書き

込みし，それが完了してからデータファイルやインデックスを書き込む（非同期）．この方式はトランザクション・ログを先に書き込むことからWAL（Write Ahead Logging：ログ先行書き込み）と呼ばれ，多くのデータベースシステムで広く採用されている．［66］

WALの基本的な考え方は，テーブルやインデックスがあるデータファイルに書く前に変更分が先にロギングされていなければいけないという点にある．すなわち，ログレコードが永続的な記憶装置に書き込まれてから更新などの処理を行う．このような手順に従って処理を行えば，たとえクラッシュがおきてもログを使ってデータベースをリカバリできる．このため，トランザクションのコミットの度にデータページをディスクに書き込む必要がなくなる．［67］

障害が発生し，リカバリする場合は，データページに対してまだ行われていない変更分はログレコードを使って再実行（「REDO」ロールフォワードリカバリ Roll Forward Recovery ）する．また，コミットされていないトランザクションはデータページから削除（「UNDO」ロールバックワードリカバリ：Roll Backward Recovery ）される．

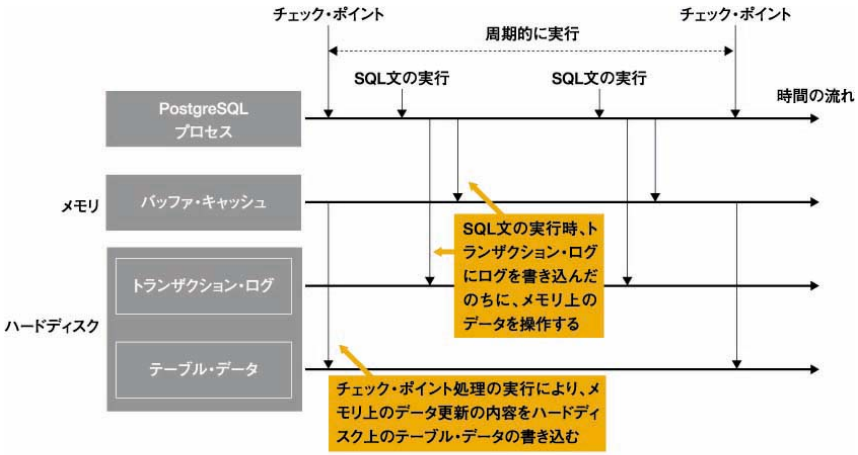


図 3.20　ＷＡＬ処理手順「例：PostgreSQL」

図 3.20 により，ＷＡＬの手順を「PostgreSQL」を例に説明する［68］．

トランザクション・ログへの書き込みは 1 つの SQL 文が実行されるたびに，ハードディスク上のトランザクション・ログ・ファイルに対して行われる．ただし，個々の書き込みは，非常に小さなサイズのデータをシーケンシャルに追加するだけの処理である．そのため，（ハードディスク上の）テーブル・データ・ファイルに対する書き込みに比べて 処理の時間は格段に短い．

データをメモリ上にキャッシュして，データ操作は原則そのキャッシュ上で行うよ

うにする．そして，一定周期(デフォルト 300 秒)で発生するチェック・ポイント処理のときにのみ，キャッシュ上のデータをハードディスクのテーブル・データ・ファイルに書き込む．

ＷＡＬの実用例として，ハードディスク障害時のデータベース復旧処理の流れを図 3.21 に示す．

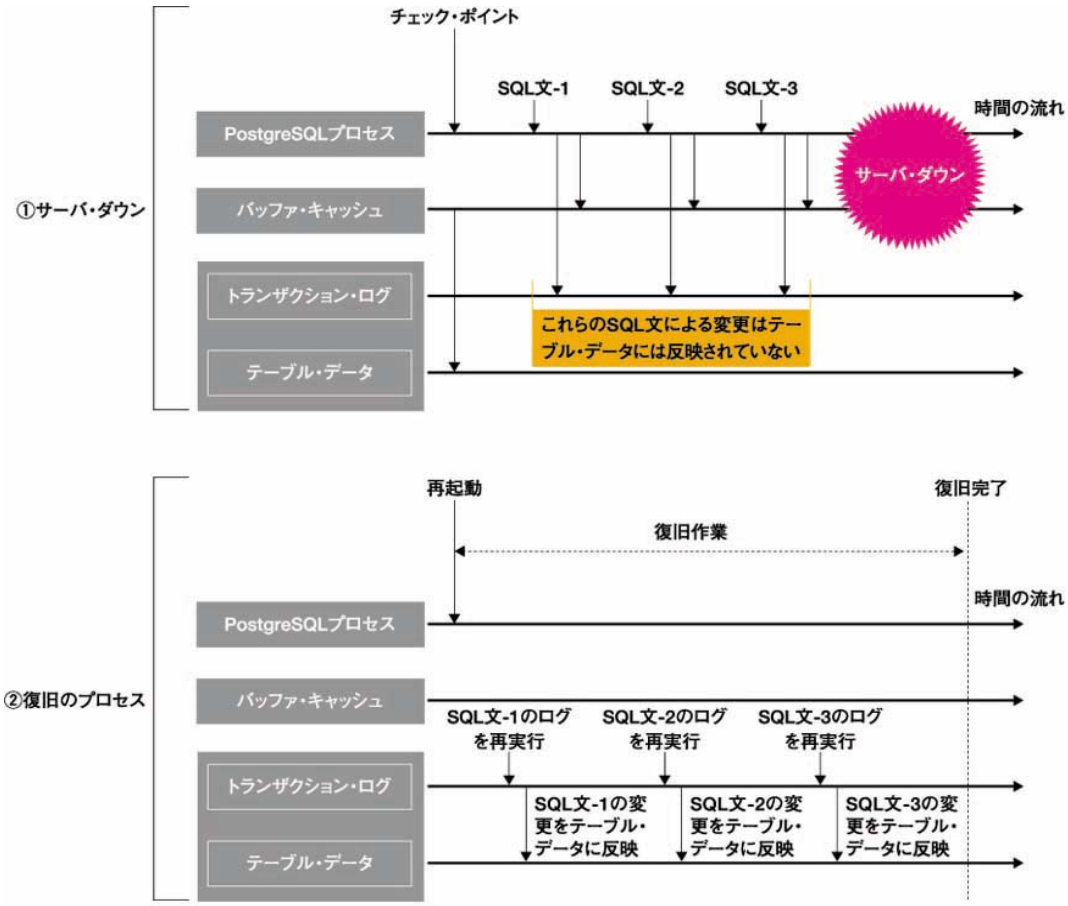


図 3.21　WAL による障害時のデータベース復旧処理

この図の上部にある①サーバ・ダウンの部分は，いくつかの SQL 文が発行されたのちに，サーバがダウンした状況を表している．ダウンの直前においては，メモリ上のデータ操作は ハードディスク上のテーブル・データ・ファイルに反映されていない．この状態でサーバを再起動させると WAL による復旧作業が始まり，トランザクション・ログの内容が順次実行されていく（図 3.21 の下部②）．こうしてサーバ・ダウン以前のメモリの状態がハードディスク上のテーブル・データ・ファイルに反映されていき，データベースの不具合が徐々に解消される．そして最終的にサーバ・ダウン直前の状態にデータベースが復旧されるのである．

ハードディスク自体に障害が発生すると，データベースの復旧はおろか，データのサルベージすら不可能になる場合がある．しかも，データ更新が頻繁に行われるシステムにおいては，チェック・ポイント処理時にハードディスクに対する大量のアクセスが集中し，結果として，DBMS(Data Base Management System)のパフォーマンスが低下する可能性もある．そこで最近ではハードディスク障害によるデータの消失を防ぐアーカイブ・ログ(Archive Log)の機能が加えられたものがある．この機能は，トランザクション・ログと同等のログ（つまり，アーカイブ・ログ）を，トランザクション・ログを記録するのとはまた別のハードディスク やバックアップ装置に直接保管する方法である．

WAL はトランザクションのデータリカバリーについては一定の効果がある．しかし，システムの処理手順の管理が集中（Resource Manager）である事やデータ復旧（チェック・ポイント）時のシステム停止などについて，適用するシステムの条件によっては問題となる場合がある．

3.4.2　自律分散整合化技術の理論とモデル化

３．４．１（１）項で述べたが，分散データベースシステムにおいて，トランザクションが複数のデータベースサーバを更新する場合には，更新の原子性，すなわち，全てのデータベースサーバが更新されたか，または全く更新されなかったかのいずれであるのかを保障する必要がある．このための制御方式として，コミットメント制御がある．代表的なものはコミットメント制御のためのプロトコルである二相コミットメント（two-phase commitment）プロトコルがある[29]．

しかし，この方法では障害が発生した場合に動作中のプロセスが一時停止する場合がある．また，全プロセスが終了するまでの時間が掛かり，メッセージの数も多くなるという問題点がある．この問題を解決するため，三相コミットメントプロトコル[31]などが示されているが根本的な解決にはなっていない．

また，３．４．１（２）項で述べた WAL は特に端末レベルでのトランザクションのデータリカバリー問題の解決には一定の効果がある．しかし，この方法では，チェック・ポイント時や障害時はシステムを停止してデータを復旧する必要がある．異種統合型情報サービスシステムにおいては高速性，高信頼性の異種ニーズに加えて，サービスの継続が必須条件である．このため，部分障害や保守，拡張など，システムの環境が変化しても，システム全体が停止しない仕組みが必要である．

本項では，異種統合型情報サービスシステムのアーキテクチャの特徴である「時間差異種データフィールド 」を使い，時間差によりデータ間の論理的整合化を図る「自律分散整合化技術」を提案する．この技術について，モデル化とその理論を

以下に述べる．

(1)　基本理論

異種統合型情報サービスシステムにおいては「制御システム」のリアルタイム処理条件下で「サービスの継続」や「流動性の確保」を図る必要があるため，どうしても端末での処理において「データの抜け，データの誤り」などが発生する．しかし，このような障害があっても，サービスの継続性の観点から一定の条件下ではサービスを継続し，その後，早期にデータの整合化を実施した方が有効な場合がある．「高速処理」自体が基幹サービスである高負荷トランザクションの情報サービスシステムの場合はさらにその点の考慮が必要である．このため，サービスの継続性を確保しつつ高速処理性と高信頼性を確保するアシュアランス技術が「自律分散整合化技術」である．

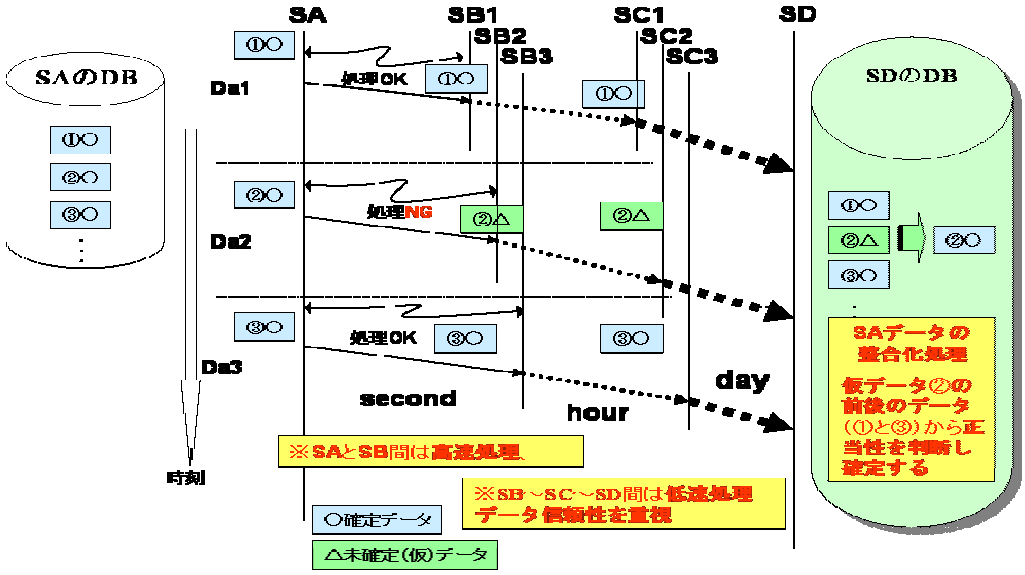


図 3.22　自律分散整合化技術

この技術は図 3.22 に示すように，サブシステムＳＡがＳＢ１，ＳＢ２，ＳＢ３などと高速処理をするときに，処理に障害が発生しても，時間差でデータを滞留する事で，その前後のデータの状況などを連携し，障害のあった当該データを事後に論理的に整合化処理するものである．このことにより，サービスを継続しつつ，信頼性の確保も可能となる．

(2)　システムのモデル化（多段階整合化モデル）

図 3.23 に示すように，Ｎ個のレイヤーからなる「時間差異種データフィールドによる多段階整合化モデル」を考える．

異種統合型情報サービスシステムにおいては，全てのデータがデータフィールド（ＤＦ：Data Field）を介して各ノードへ伝達される．各ノードはデータを一定時間滞留させ，集約した後に整合化処理する．この一連の処理する階層をレイヤーと呼ぶ．高速処理時にデータにエラーが発生してもサービスは継続し，各レイヤーで整合化処理が行われる．データは順次時間差でノードに集約され，論理的な整合化条件が整った場合は整合化処理される．されなかったデータは時間差で次のレイヤーに移行する．このように，端末のノードはリアルタイム処理の高速性を確保し，データの欠損が発生しても，データフィールド毎にデータの滞留時間を変化させた時間差異種データフィールドを多段階に構成し，それらの連携による論理的整合化図るモデルである．

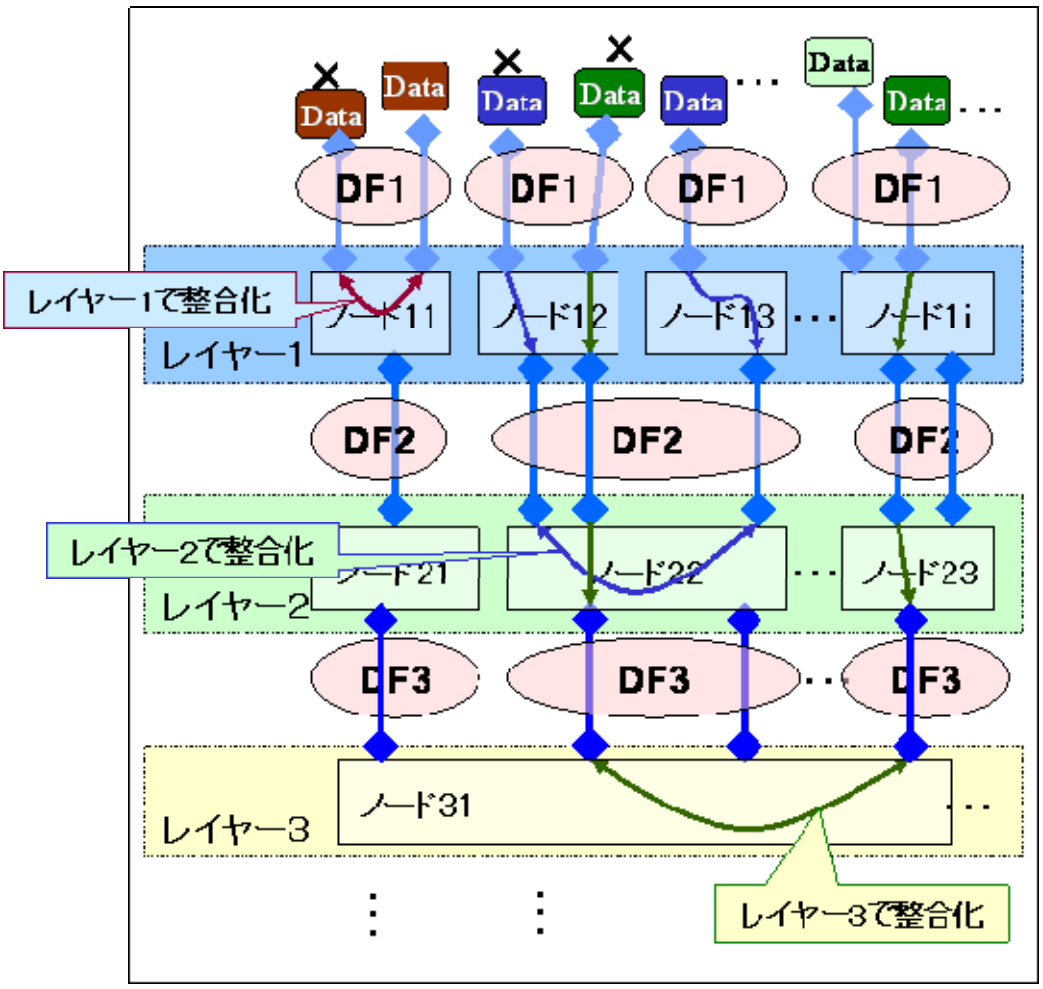


図 3.23　時間差異種データフィールドによる多段階整合化モデル

3.4.3　　時間差異種データフィールドによる多段階整合化モデル

前項で述べたように，自律分散整合化技術の多段階整合化モデルではレイヤー毎のデータ収集の状況（データの滞留時間）が整合化処理に大きく関係する．このため本項では，この関係を明確にし，各レイヤーでの整合化に必要な時間について考察する．

以下に「トランザクションの開始から整合化終了までに要する累計整合化時間」と「データの平均整合化時間」などを考察し，多段階整合化モデルにおける最適レイヤーの分割，最適整合化時間などについて，シミュレーションにより検証する．

(1)　多段階整合化モデル

図 3.24 に単一レイヤーの場合の整合化モデルを示す．

この場合は，レイヤーにデータが到着したときに整合化処理を開始する条件とし，完了まではデータは蓄積される．単一レイヤーのため整合化処理のデータが大となる．また，高トランザクションのため，レイヤーに格納するデータも多くなり，処理の遅延が発生する．このため，各データは，整合化処理時間 t_h（通信時間を含む）の他に，処理の待ち時間 t_wが発生する．

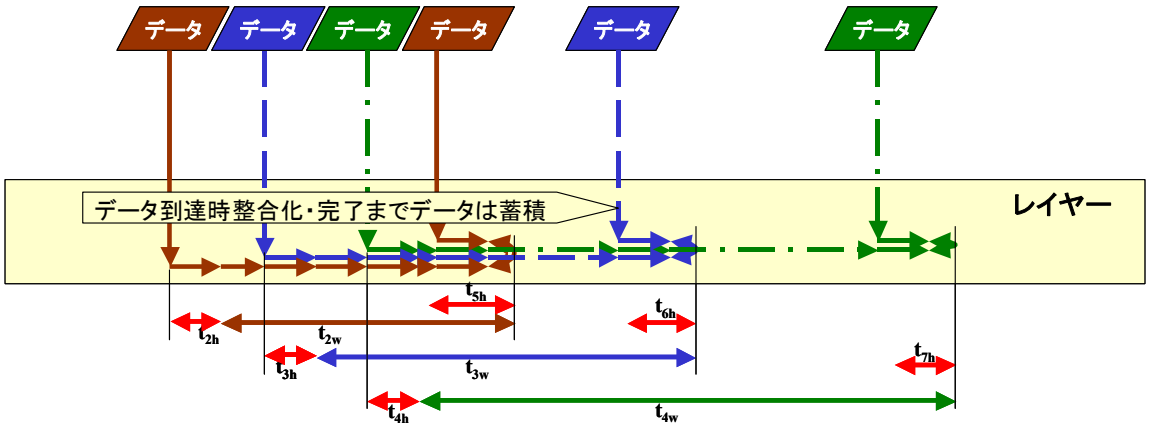


図 3.24　整合化モデル（単一レイヤーの場合）

図 3.25 に複数レイヤーの場合の整合化モデルを示す．

この場合は，データは各レイヤーで一定時間滞留した後で整合化処理が行われる．整合化できなったデータは，次のレイヤーに送信される．

例えば，データ 1 はレイヤー1 のノード $S1_{1j}$ で平均滞留時間 t_1 後に同一ノードに来た次のトランザクションデータ 1 と整合化処理（平均処理時間 t_{1h}）が行われている．同様にデータ 2 はレイヤー2 のノード $S2_1$ で整合化が行われ，データ 3 は

レイヤー3 のノード $S3_1$ で整合化処理が行われている．このように，トランザクションが複数レイヤーに分散し，結果的に，整合化に要する時間も短縮可能である．

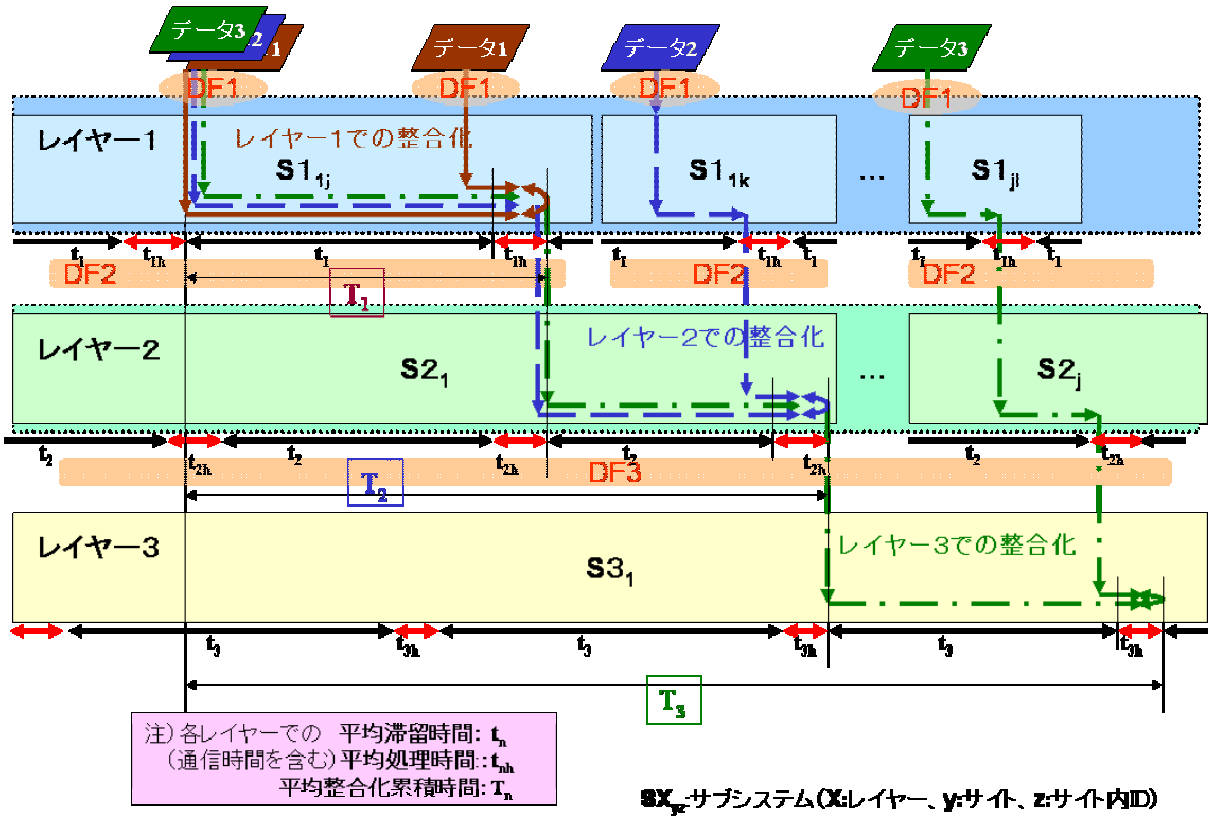
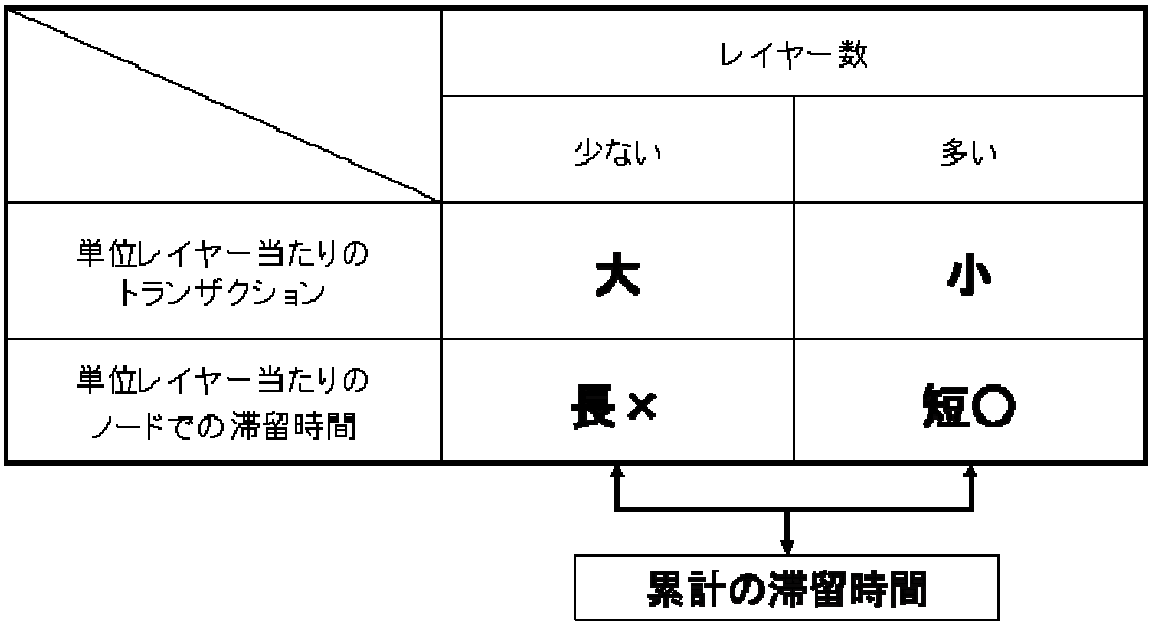


図 3.25　整合化モデル（複数レイヤーの場合）

多段階整合化モデルにおけるレイヤー数とトランザクション，処理時間（滞留時間）の関係を表 3.2 に示す．

レイヤー数が少ない場合は単位レイヤー当たりのトランザクションが多くなり，ノードでの処理が停滞し，単位レイヤー当たりの滞留時間は長くなる．レイヤー数が多い場合はその逆になる．整合化を早期に行うためには，累計の滞留時間の最適値の決定が重要である．このため，データの発生パターンに応じた，整合化時間の期待値（平均整合化時間）を最小にする「レイヤー数」とその時の「滞留時間」を求める理論式について以下に解説する．

表 3.2　レイヤー数とトランザクション，処理時間(滞留時間）の関係



(2)　整合化時間

各レイヤーでの整合化に要する時間をｔ，開始から整合化終了までに要する累計時間をＴと表し，これを「整合化累積時間」と呼ぶ．

・レイヤー１では

$$T_1 = t_1 + t_{1h} \qquad (3.1)$$

但し，　t_1　：レイヤー１でのデータ滞留時間

　　t_{1h}　：レイヤー１での処理時間（整合化処理，通信時間を含む）とする．

・レイヤー２では，同様にレイヤー１での時間T_1を加えて，以下のとおりである．

$$T_2 = T_1 + t_2 + t_{2h} = t_1 + t_{1h} + t_2 + t_{2h} \qquad (3.2)$$

但し，　t_2　：レイヤー２でのデータ滞留時間

　　t_{2h}　：レイヤー２での処理時間（整合化処理，通信時間を含む）とする．

・レイヤーＮでは，レイヤー１からレイヤーＮまでの全体の整合化が終了するまでの累計時間は，各レイヤーでの時間を加えて，

$$T_n = T_{n-1} + t_n + t_{nh} = \sum_{j=1}^{n} \left(t_j + t_{jh} \right) \qquad (3.3)$$

但し，t_n　：レイヤーＮでのデータ滞留時間

　　t_{nh}　：レイヤーＮでの処理時間（整合化処理，通信時間を含む）とする．

(3)　整合化時間の期待値

前項で整合化に要するレイヤー毎の時間と累計時間については明らかにした．しかし，データの単位で見たときの整合化時間はどのレイヤーで行われてかによって時間が異なっている．例えば，レイヤー１で整合化が行われたデータとレイヤーＮで行われたデータでは整合化までの時間が異なる．このため．システムの整合化を評価する場合には「データの平均整合化時間」を指標とする．ここではこれを「整合化時間の期待値」と呼ぶ.「整合化時間の期待値」について以下に考察する．

整合化時間の期待値T_{exp}は「不整合が発生してから整合化されるまでにかかるデータ 1 件当たりの平均時間」と定義する．従って，（レイヤーj での整合化累積時間）＊（レイヤーjでの処理量q_j：件数）}÷（システム全体の処理量Q：件数）であり，以下の式で表される．

$$T_{exp} = \sum_{j=1}^{n} [(t_j + t_{jh}) \times q_j] \div Q \qquad (3.4)$$

但し，（レイヤーjでの処理件数q_j）÷（全体の処理件数Q）＝p_j　とする．
この時，p_jを「整合化分担率」と呼ぶ．一方，t_{jh}は単位ノード当たりの整合化処理量のq_j / ノード数i_jに比例すると仮定すると，　$t_{jh} = \alpha q_j / i_j$（α：単位処理データ数あたりの処理時間)となる．従って，式 3.4 は以下のようになる．

$$T_{exp}(n, t_n) = \sum_{j=1}^{n} \left\{ p_j \left[\sum_{k=1}^{j} (t_k + t_{kh}) \right] \right\} = \sum_{j=1}^{n} \left\{ p_j \left[\sum_{k=1}^{j} (t_k + \alpha p_k Q / i_k) \right] \right\} \qquad (3.5)$$

$$\text{但し，} \quad \left(\sum_{j=1}^{n} p_j = 1 \right)$$

また，p_jはデータを蓄積する時間τ（トランザクションの発生特性）に依存すると仮定すると，以下のとおりである．

$$p_j = \int_{T_{j-1}}^{T_j} P(\tau) d\tau = \int_{\sum_{k=1}^{j-1}(t_k + t_{kh})}^{\sum_{k=1}^{j}(t_k + t_{kh})} P(\tau) d\tau \qquad (3.6)$$

但し，P(τ)：トランザクション発生の確率分布関数，T_j：平均整合化累積時間
式 3.5 に式 3.6 を代入して，最終的に整合化時間の期待値は以下の式となる．式からも判るように各レイヤーの滞留時間 t_j とレイヤー数 j の関数である．この事からT_{exp}を最小にするような最適なレイヤー数と滞留時間が求められる．

$$T_{exp}(n, t_n) = \sum_{j=1}^{n} \left(\int_{\sum_{k=1}^{j-1}(t_k + t_{kh})}^{\sum_{k=1}^{j}(t_k + t_{kh})} P(\tau)\, d\tau \right) \left[\sum_{s=1}^{j} \left\{ t_s + \frac{\alpha Q}{i_s} \left(\int_{\sum_{m=1}^{s-1}(t_m + t_{mh})}^{\sum_{m=1}^{s}(t_m + t_{mh})} P(\tau)\, d\tau \right) \right\} \right] \quad (3.7)$$

(4)　シミュレーション

前項の整合化分担率 p は,時間とともに変化し，その合計は「１」である．どのレイヤーでの整合化分担率が高いかは，実際の場面ではユーザの挙動（トランザクションの発生特性）によって決定される．従って,整合化分担率 p は時間の関数で表す事が出来ると仮定し，合計が「１」となる関数として，今回はポアソン分布によるものとした．

（参考：実際に鉄道による乗客の移動時間は，約 25 分が平均値であり，極近距離（数分）は少なく，数十分程度が最も多く，遠距離（１時間以上）は再び少なくなる．）

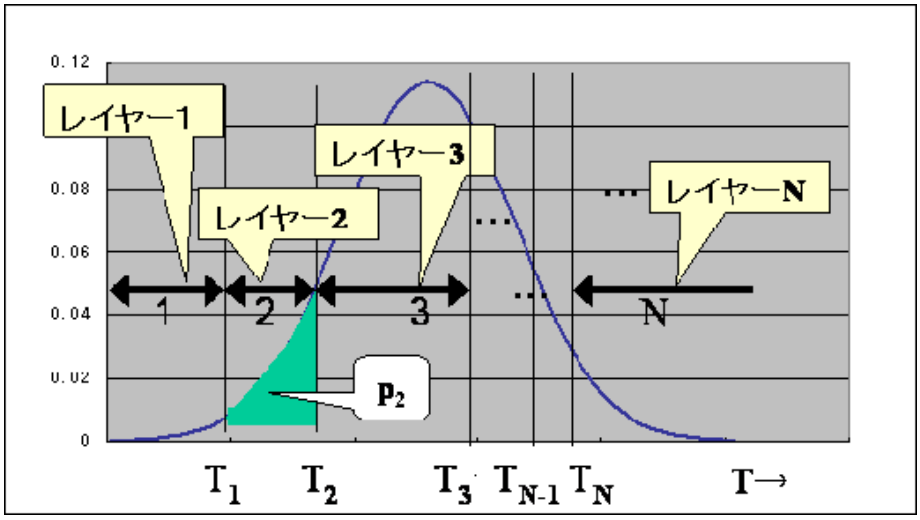


図 3.26　整合化分担率の分布（例：ポアソン分布）

図 3.26 に示すようにポアソン分布によるものとすると，

$$P(\tau) = \frac{e^{-\lambda} \cdot \lambda^{\tau}}{\tau!} \quad (3.8)$$

各レイヤーの整合化分担率pは$\tau = T_{j-1}$と$\tau = T_j$とで区切られた面積に等しいため，

$$p_j = \int_{T_{j-1}}^{T_j} P(\tau)d\tau = \int_{T_{j-1}}^{T_j} \frac{e^{-\lambda} \cdot \lambda^{\tau}}{\tau!} d\tau \qquad (3.9)$$

となる．

従って，この場合の整合化時間の期待値は，式3.5より，

$$T_{exp}(n, t_n) = \sum_{j=1}^{n} \left\{ \left[\int_{T_{j-1}}^{T_j} \frac{e^{-\lambda} \cdot \lambda^{\tau}}{\tau!} d\tau \right] \left[\sum_{k=1}^{j} t_k + \alpha Q \left(\int_{T_{k-1}}^{T_k} \frac{e^{-\lambda} \cdot \lambda^{\tau}}{\tau!} d\tau \right) / i_k \right] \right\} \qquad (3.10)$$

である．

シミュレーションはシステムをn分割し，n-1個のT_nを固定して残り1つのT_nを動かした時のT_{exp}の変化を表したものである．

図3.27に2レイヤーの場合，図3.28に3レイヤーの場合をそれぞれシミュレーションした時の「整合化時間の期待値T_{exp}」と各レイヤーの「整合化累積時間T」との関係を示す．このT_{exp}を最小にするようなレイヤーの分割数が最適なシステムである．

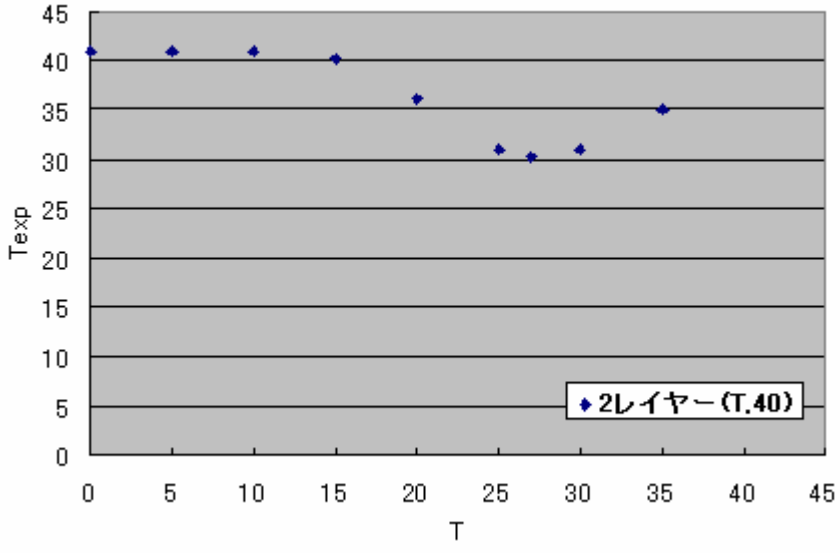


図3.27　T_1を変化させた時のT_{exp}の変化(2レイヤー)

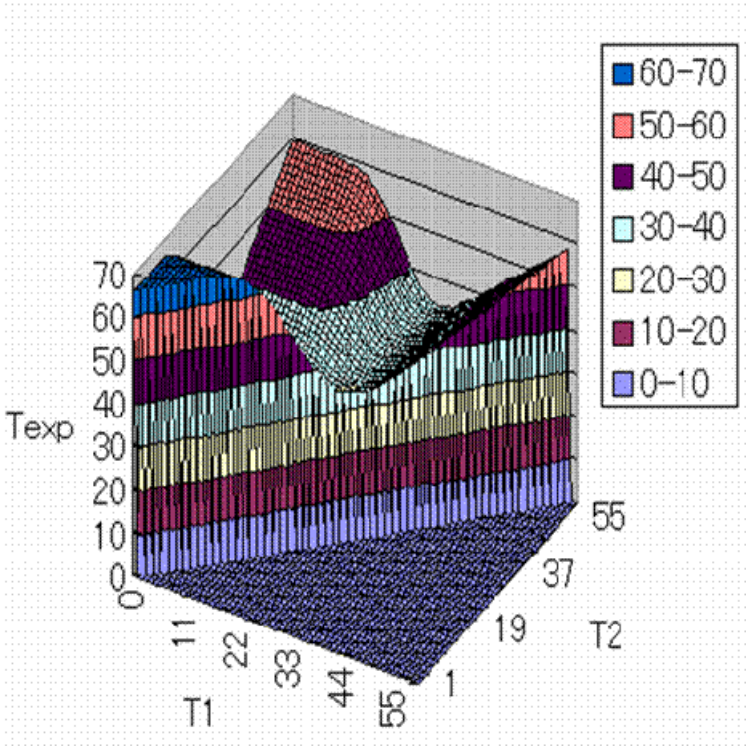


図 3.28　T_1 と T_2 を変化させた時の T_{exp} の変化(3 レイヤー)

図 3.29 は図 3.27 と図 3.28 の各レイヤーの最適値のグラフを抜粋し，プロットしたものである．シミュレーションの前提値を λ=24.28, i_1=3511, i_2=424, i_3=1 としたとき，T_{exp} は最小値 28.44 を得る．この結果，最適レイヤー数は 3，最適滞留時間は T_1=24, T_2=30, T_3=40 が得られる．

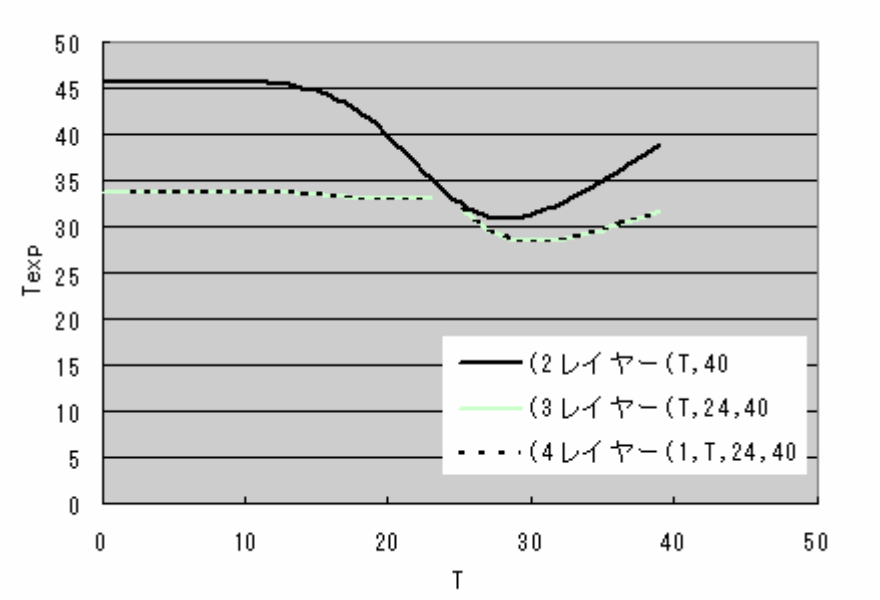


図 3.29　最適レイヤー分割のシミュレーション

(5)　トランザクションの発生特性と最適レイヤー分割

前項でも述べたが,整合化分担率はトランザクションの発生特性によって決定される時間の関数として,今回はポアソン分布を利用してシミュレーションを行った.参考までに,ポアソン分布のλを変化させた場合の最適レイヤー分割について,シミュレーションした結果を以下に述べる.

図 3.30 に整合化分担率のポアソン分布をλ＝12,　24.87,　50 の場合を示す.前項のシミュレーションの場合のλの約１／２倍と２倍とした.

図 3.31 にλを変えた場合の,最適レイヤー分割のシミュレーション結果を示す.λが小さいと平均整合化累積時間,整合化時間の期待値は小さくなる.　λが大きいと同様に大きくなる.レイヤー数はλが何れの場合でも最適レイヤー分割は「３レイヤー」である.

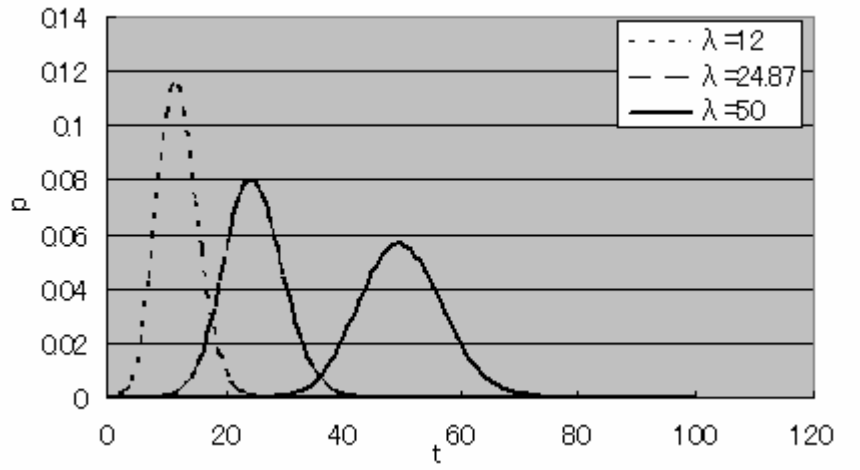


図 3.30　整合化分担率の分布（ポアソン分布λ＝12, 24.87, 50）

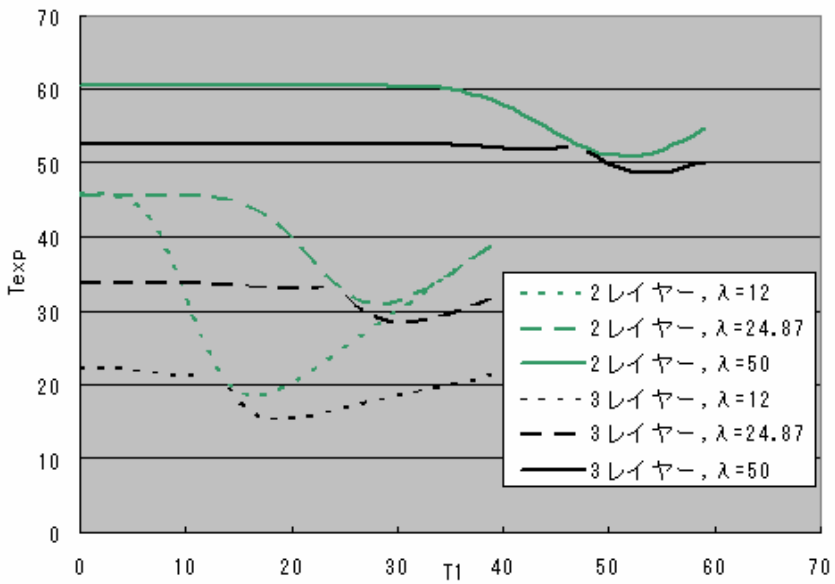


図 3.31　最適レイヤー分割（ポアソン分布λ＝12, 24.87, 50）

3.5　システムのアシュアランス性評価技術

3.5.1　評価技術の目的

今回の評価技術の目的は異種モード共存環境下におけるシステム稼動の保証度合の評価は,「機能量×稼働時間」の最大化を実現するための評価基準を明らかにすることである.このための評価手法として機能信頼度評価技術を提案する.

3.5.2　評価技術のコンセプト

(1) ニーズ

システムの分散化により，計算機や制御分野でシステムの構成，制御技術の革新がなされつつあり，評価技法の面でも，これらに対応した新しい技法が要求されている．評価指標において代表的なものに「信頼性」があるが，この「信頼性」のJIS(JIS-Z8115)による定義を以下に示す.

① システム:所定の任務を達成するために，選定され，配列され，互いに連係して動作する一連のアイテム(ハードウェア，ソフトウェア人間要素)の組み合わせ
② アイテム:信頼性の対象となるシステム(系)，サブシステム，機器，装置，構成品，部品，素子，要素などの総称又はいずれか
③ 信頼性:アイテムが与えられた条件で規定の期間中，要求された機能を果たすことができる性質
④ 信頼度:アイテムが与えられた条件で規定の期間中，要求された機能を果たす確率

従来，高信頼化技法には，多重冗長システムや優美劣化システム（gracefully degrading system）がある．これらのシステムは，部分的故障の発生に対してもなお，一定の機能を維持させることを狙ったものである．そこで，一定の機能が完全に働いている時間をもとにしてＭＴＢＦ（mean time between failures），ＭＴＴＲ（mean time to repair）などの信頼性尺度がある[25,43].

また，優美劣化システムのように，故障に応じて性能が低下するシステムでは，性能と信頼性を結びつける尺度が提案されている[28]．たとえば,Beaudry は計算機の単位時間あたりの計算処理能力を，計算容量（computational capacity）と定

義し，システムダウンまでに可能な平均積算計算容量を MCBF（Mean Computational Before Failures）とよび信頼性尺度とした[38]．また,Meyer は，ユーザがシステムを使用して得られる効用と，信頼性を結びつけ，期待効用を Performability と呼び定義している[39]．

さらに，故障による社会や経済への影響と信頼性を結びつける尺度も考えられ，これは Risk Analysis とよばれている[63]．また，多重系の予防保守のための有効な点検方策決定アルゴリズムも検討されている[40]．

以上の従来の議論では，ともに信頼性評価にあたりつぎの前提に立っている．

① トータルシステムとして一定機能が維持されていれば正常，それ以外を異常とする．
② トータルシステムとしての一つの性能，効用の基準がある．
③ トータルシステムに一つの仕事が与えられる．

このように，従来システムはシステム全体としての機能，性能が達成されるか否かを評価の基準としてきた．しかし，システムが大規模になると常にどこかのコンポーネントが故障していたり，保守や拡張がなされたりするため，部分的にも機能させ続けねばならない．そのためには，各サブシステムに自律性を与え，これらが，自律的に状況に合わせ連携をとりながら機能し，その結果として，システムの機能，性能をみる見方が必要となる．

サブシステムが自律的に振舞う分散システムを評価するため，つぎのような前提に立っている．

① 各サブシステムは，それぞれの目的，機能をもつ．これらの統合結果として，システムを考える．
② 各サブシステムの連携により，システムの機能が達成される．サブシステムの故障や保守，拡張により，サブシステム間の結合関係が変化すれば，達成されるシステムの機能も変わる．
③ 各サブシステムで処理すべき仕事は発生しうる．これらの統合結果としてシステムの仕事を考える．

このように，分散システムでは故障のしにくさ（信頼度）だけでなく，部分的故障発生後のサブシステム間結合の再構成による連携のとりやすさ（耐故障度）を評価する必要がある[37]．

(2) 課題

前述の分散システムに対する前提や信頼性評価に対する要求は，現在のシステム設計には当然考慮されてきた．しかし，従来の信頼性評価技法は，トータルシステムを第一義的に考える集中，または階層システムを対象としている．また，故障や保守，拡張によるシステム構造の変化も考慮していない．

最近の情報通信技術の発達は，分散システムからの観点に立った信頼性評価技法の発達を必要としている．このような観点から機能の残存度合を評価するため，機能信頼度［41, 42, 43］が提案された．

異種モード共存環境下におけるシステム稼動の保証度合の評価において，「機能量×稼動時間」最大化を実現するための評価基準について，考察する．異種のモードが共存し，しかもそれらが時間と共に変化する環境下で，システムの稼動を保証しうる性質であるアシュアランス性に着目し，その観点から，システム全体を評価するための評価指標として，機能信頼度[26, 44, 45]の概念を使用した，機能信頼性評価技術を提案する．

(3) システムモデル

ここでシステムモデルを図 3.32 で定義する．Ｎ個のサブシステム（S_1, S_2, …, S_N）からなるシステムＳを考える．

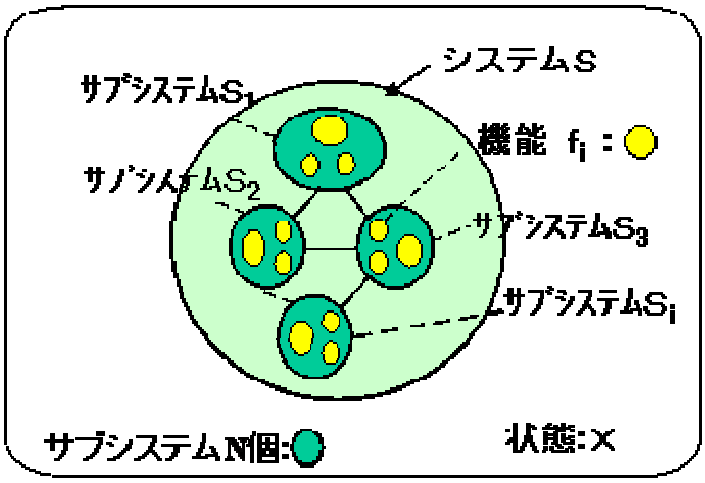


図 3.32　システムモデル

システムＳはＮ個のサブシステム（サブシステム群）から構成され，各サブシステムS_iは機能 f_iを持つ．多段階整合化モデルの場合，整合化の単位はサブシステム単位，で考える．いずれも機能の単位は機能量として表現する事とする．

3.5.3　機能信頼性評価技術

(1)　システムのアシュアランス性

アシュアランス性はこれまでも述べたように,「時間と共に変化する異種のモードに適応でき，システムの稼動を保証しうる性質」[16,17]と定義されている.

これに基づいてシステムにおける異種性,適応性とアシュアランス性を具体化し,以下のように定義した.

① 異種性；異種モード（制御／情報システム等）の共存下でのシステム稼動の保証度合の評価
② 適応性；この異種モードの変化（障害によるシステム停止／復旧，高負荷変動）する環境下でのシステム稼動の保証度合の評価
③ システム運用のアシュアランス性；異種モード共存環境下におけるシステム稼動の保証度合の評価において，システム全体機能の最大稼動を保証する性質．具体的には所定期間内の「機能量×稼動時間」の最大化を保証する事である.

(2)　評価技術のコンセプト

前項で述べた通り，システムのアシュアランス性は「機能量×稼動時間」の最大化を保証する事であるため,その評価指標のアシュアランス度を下記のように定義した.

即ち,システム稼動の保証度合の評価するためのアシュアランス性評価として機能信頼度[62]により，アシュアランス度Ａｆを定義した.

システム稼動の保証度合のアシュアランス度 Af
　＝ΣΣ（機能×信頼度×稼動時間）／(全体機能×所定期間)
　＝ΣΣ（機能達成度×信頼度×稼動時間）／所定期間

ここで機能達成度＝機能／全体機能である．このように，機能達成度とは，サブシステムS_1の機能が働いたというだけでない．他サブシステムとの連携により，一連の処理が終了し，その結果，どの程度有効な出力を生じるかを評価するものである.

さらに,「機能達成度×信頼度」を「機能信頼度」と定義し，システム稼動の保証度合のアシュアランス度 Af を次のように定義する.

システム稼動の保証度合のアシュアランス度 Af

＝Σ Σ（機能信頼度×稼動時間）／所定期間　　(3.11)

(3)　機能信頼度の定義[9]

前項の評価技術のコンセプトにより，機能信頼度を定義する．

①機能の定義

図 3.32 に示すように，システムＳはＮ個のサブシステム（S_1, S_2, …, S_N）からなり，各サブシステムS_iは機能 f_iを持つ．但し，S:システム，Ｓｉ:サブシステム，Ｘ:システムの状態，X_0:無障害時の状態である．

状態Ｘの下で，システムＳが達成する機能 F(Ｓ,Ｘ)は各サブシステムが持つ機能 $f_i(S_i, X)$の総和であり次式で定義する．

$$\mathbf{F(S,X)}=\sum_{\mathbf{i}\in\mathbf{S}}\mathbf{f_i(Si,X)} \qquad (3.12)$$

②機能達成度の定義

システムＳが状態Ｘの時，まったく異常がない状態X_0の時のシステムＳに比べ，サブシステムが発揮する機能の割合を，状態Ｘの下での「サブシステムＳｉの機能達成度：ϕ_i（Ｓｉ，Ｘ)」と呼ぶ．また，状態Ｘの下での「システムＳの機能達成度：Ψ（Ｓ，Ｘ)」と呼ぶ．機能達成度ϕ_i（Ｓｉ，Ｘ)，Ψ（Ｓ，Ｘ）は次式により定義する．

$$\phi_{\mathbf{i}}\left(\mathbf{Si,X}\right)=\frac{\mathbf{f_i(Si,X)}}{\mathbf{F(S,X_0)}} \qquad (3.13)$$

$$\Psi\left(\mathbf{S,X}\right)=\sum_{\mathbf{i}\in\mathbf{S}}\phi_{\mathbf{i}}\left(\mathbf{Si,X}\right) \qquad (3.14)$$

このように，機能達成度とは，サブシステムS_iの機能が働いたというだけでなく，他サブシステムとの連携により，一連の処理が終了し，その結果，どの程度有効な出力を生じるかを評価するものである．

③時刻ｔでの機能信頼度R_t

時刻ｔでのシステムＳの機能達成度の期待値を，時刻ｔでの機能信頼度R_tと呼ぶ．R_tは機能達成度と期待値Ｐ(Ｘ|t)=Prob[Ｘ|t]の積で表し，次式により定

義する．なお，期待値Ｐ（Ｘ｜t）は時刻ｔ，状態Ｘのもとでシステム Ｓが達成する機能がＦ（Ｓ，Ｘ）となる確率である．

$$\mathbf{R_t(t)} = \sum_{\mathbf{X}} \Psi\left(\mathbf{S},\mathbf{X}\right)\cdot \mathbf{P}\left(\mathbf{X}\,|\,\mathbf{t}\right) \qquad (3.15)$$

(4)　信頼度の定義

従来の信頼性は，高信頼化手段として，多重冗長化やバックアップなどの技法を用いたとしても，正常なとき，またはバックアップなどの切替が成功したとき信頼度＝１，異常なとき，またはバックアップなどの切替が失敗したとき信頼度＝０の２値のみを考えた特別な場合である．多段階整合化モデルの場合，滞留時間による時間差で整合化処理を行っているため，「時間差異種データフィールドによる多段階整合化モデル」における信頼度（Ｒ$_e$）は平均故障間隔(ＭＴＢＦ)，平均修復時間(ＭＴＴＲ)の他に平均滞留時間（ＭＴＦＳ：mean time for stay ）も考慮し，次式 3.16 で表す（図 3.33）．

$$\mathbf{R_e} = \frac{\mathbf{MTBF}}{\mathbf{MTBF} + \mathbf{MTTR} + r_e\cdot \mathbf{MTFS}} \qquad (3.16)$$

但し，r_e は障害滞留係数と呼ぶ．r_e はシステムのエラー発生確率と滞留時間で決まる．r_e ＝障害発生後の滞留時間／平均滞留時間（ＭＴＦＳ）で表される．

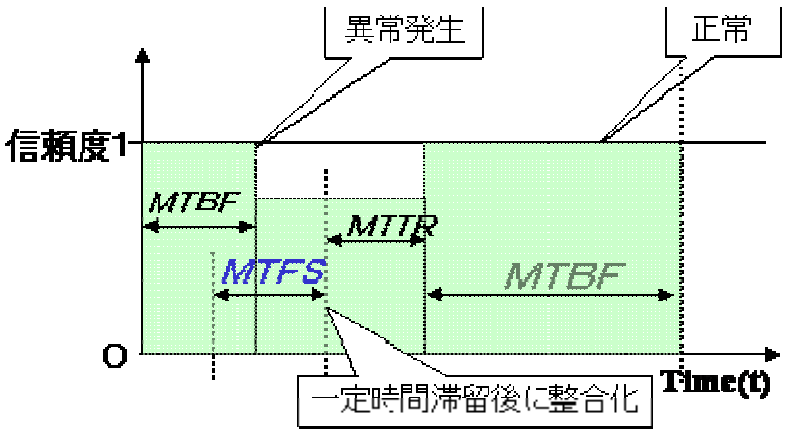


図 3.33　平均滞留時間と信頼度の関係

(5)　平均機能信頼度（アシュアランス度）の定義

システム稼動の保証度合を評価するアシュアランス度は図 3.34 に示すように機能信頼度の所定期間（ミッション時間）全体にわたる「平均機能信頼度」と定義し，次式 3.17 で表す．

$$Af = (1/T_M) \cdot \int_0^{TM} Rt(t)dt \qquad (3.17)$$

ＴＭ　：ミッション時間

Ｒｔ（ｔ）：時刻ｔにおける機能信頼度

すなわち，「機能信頼度×稼動時間」の面積最大化がアシュアランス度を高めることである．

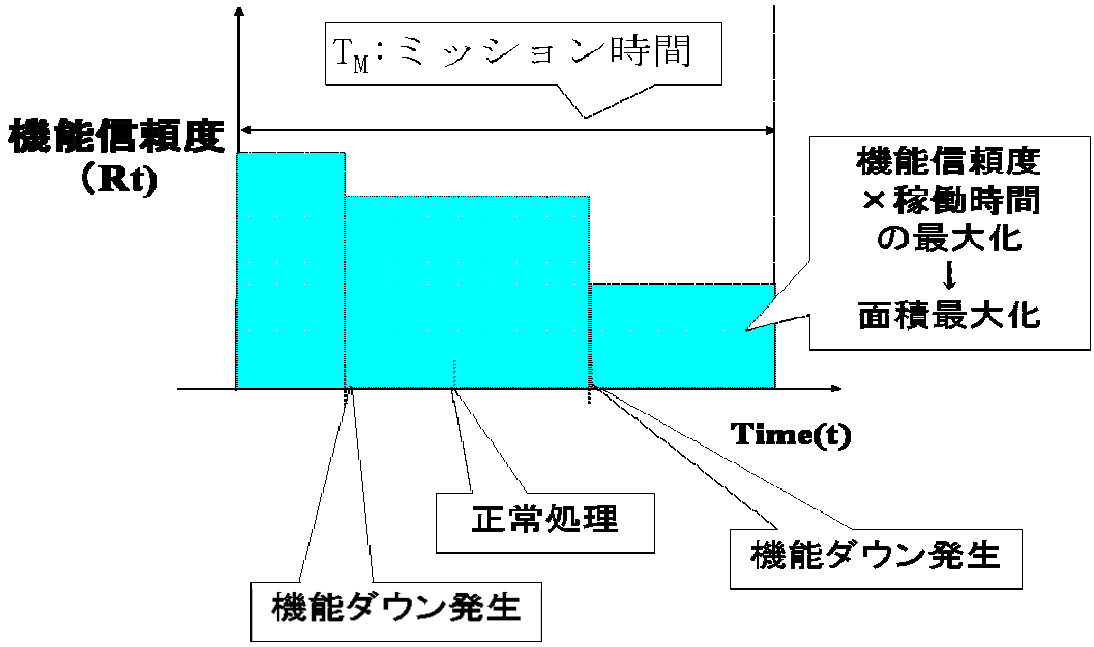


図 3.34　平均機能信頼度（アシュアランス度）

第4章

鉄道乗車券システムへの適用

日本の社会は，少子高齢化，グローバル化などの大きな社会環境の変化を受けている．このことは鉄道事業にも影響しており，従来は安全で正確な輸送を行えば良かったものが，現在では安全性，快適性，利便性などの多様な質の高いユーザー・サービスを求められている．一方で鉄道輸送を支える重要なシステムの１つである自動出改札システム（AFC: Automatic Fare Collection system）は，異種システムとの共存，異種ニーズとの連携，円滑なシステム更新などが難しいシステムであり，これらの改善というシステムニーズが存在する．具体的には高密度輸送におけるピーク時の乗降客に対応するための重要な課題の一つとして「自動改札機の処理速度向上」がある．一方，金券である乗車券を処理するため信頼性が不可欠である．このため，高速処理と高信頼性の両方を備えた乗車券システムが必要不可欠である．しかし，従来の磁気式自動出改札システムでは駅内は簡易な集中システム構成で各端末は原則オフライン処理しており，保守コストも高く，拡張性も乏しいものである．これらの課題を解決し，更に乗客の利便性向上やメンテナンスコスト低減のために新しい無線通信方式の IC カード乗車券システムの開発が必要となった（図4.1）．

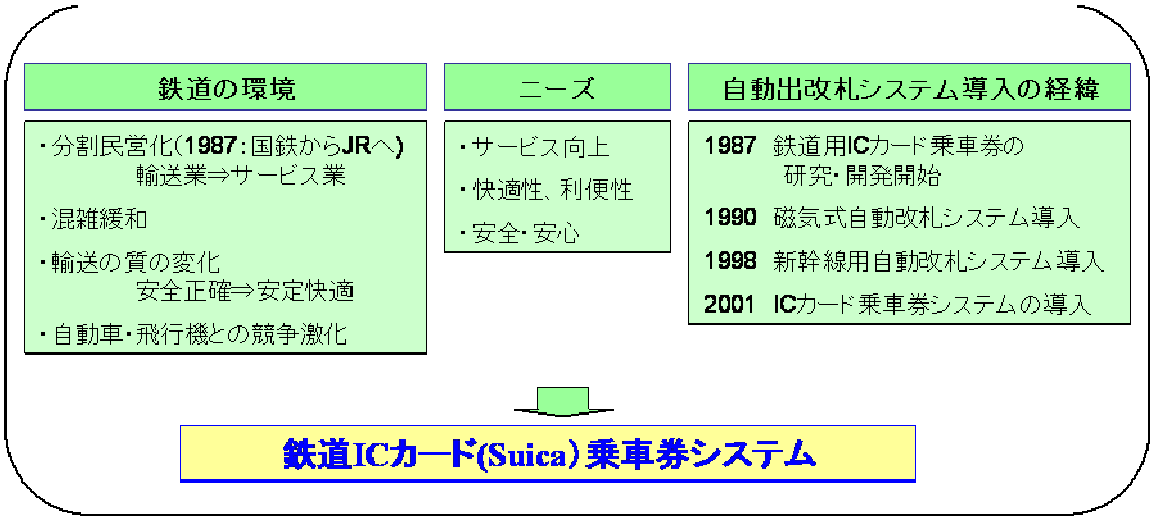


図 4.1　鉄道ＩＣカード乗車券システムを取り巻く環境

この新しいシステムにおいては拡張性，異種性，適応性を十分に考慮したシステムアーキテクチャでなければならない．高密度輸送における旅客の流動性確保，サ

ービスの継続は必須である．この新しい無線通信式 IC カード乗車券システムを磁気式 AFC システムの老朽取替えにあわせて導入することとしたが，このシステムでは無線通信により改札機との処理を行うため，改札時に処理を高速化するとデータ処理の信頼性が低下すると言う問題がある．

本章においては，無線通信方式 IC 乗車券システムにおける高速処理性と高信頼性という要件を満たすため，第 3 章で述べた，自律分散アシュアランス技術を適用する．その上で，それらの有効性を実証する．具体的には異種統合型情報システムのシステムアーキテクチャ技術と 2 つのアシュアランス技術を適用する．高速処理性を実現するための IC カードと自動改札機による「自律連携処理（運賃計算の自律分散アルゴリズム）」を適用し，さらにその有効性についてモデル化し，比較検討を行い証明する．また，高信頼化のために各サブシステムでの「自律分散整合化技術」を導入する．また，システムのアシュアランス性評価という観点から，第 3 章で述べた「機能信頼性評価技術」を適用し，システム稼動の保障度合いを評価するアシュアランス性評価を行い，その有効性を示した．

また，システム設計への適用として設計パラメータの最適値を求める評価方法を提案し，これにより，ユーザーにとって最適な状態での設計値（改札機リーダ／ライタの通信エリアの大きさ）について自動改札機において検証した．

4.1　鉄道システム

4.1.1　輸送業からサービス業へ

通勤通学，旅行に行く際に利用する鉄道は，各線が平常通りに運転されていることが当たり前となっている．もし，少しでも遅れれば多くのご利用のお客様に大きな迷惑をかけてしまうこととなる．ご利用者にとっては，レール，車両と電気があれば，そして駅員と乗務員がいれば電車は動くと思われている．

戦後まもなくの頃は，鉄道は人間系だけで動いていた．すなわち，電話での連絡，直接言葉による情報伝達で，列車運転の安全性と正確性を担保していたことになる．しかし，現在は大量高速輸送の時代である．ＪＲ東日本を例にとってみると，１日当たり 12300 本の列車を運転し，1600 万人を輸送している．このためには，23000 機の信号機を正しくコントロールし，12000 台のポイントを正確に転換しなければならない．

したがって，戦後の鉄道の歴史は輸送量の伸びに従い，この輸送を安全に行うことを目的として，一つずつシステム化してきたものと言える．その結果，世界で一

番正確で安全な鉄道となったのである.

しかし，1975 年以降輸送量はほぼ横ばいを保ち，国鉄が民営化してＪＲとなってからまた少し伸びてきてはいるが，自動車や飛行機との競争の激化の中では，単に安全に輸送するだけでなく，より便利で快適に目的地まで行けることが求められている.これは，ＪＲとなってからの会社の経営方針にも反映され,新型車両の導入，新在直通運転の実施，編成両数の増による混雑緩和などを行ってきた．つまり，これまでの単に人を運ぶ輸送業からお客様本位のサービス業へと脱皮したわけである.

4.1.2　　鉄道の安全を支えるシステム

1960 年代後半頃のコンピュータの発展は鉄道システムにも大きな変革をもたらした．各分野の業務をコンピュータによりシステム化し，業務の効率化と安全性の向上を図ったのである．世界をリードしたのが日本の鉄道システムである.

その代表的なシステムは次の３つである[58].

・座席予約システム	MARS	1964.2～
・ヤード自動化システム	YACS	1968.9～
・新幹線運行管理システム	COMTRAC	1972.3～

MARS は，それまで人手で販売していた座席指定券を新幹線の開業にあわせて，コンピュータにより発売できるようにしたもので，指定席数を飛躍的に増大できた.

YACS は，貨物の連結・開放の作業を自動的に管理するシステムであり，世界に例を見ないものであった.

COMTRAC は，後述する新幹線の輸送をコンピュータにより一元的に管理するものである.

これらのシステムは中央集中システムとして開発され，当時，世界の先端をリードするシステムであった.

1987 年の国鉄分割民営化でＪＲになってからは経営の基本方針として，お客様・地域・社会に貢献する「生活創造企業」，最新の技術を開発・活用する「未来志向企業」，社員・家族の幸福を実現する「人間尊重企業」をかかげ，東京圏，新幹線のシステムを見直して，高速化・快適化を図り，サービスレベルの向上に資するシステムの開発に着手した.

技術革新時代にふさわしいサービスを提供するため，列車が乱れた時でもきめ細

かな旅客サービスの提供を行え，さらに早く平常ダイヤに復帰できる機能を有する，東京圏輸送管理システム(ATOS)を 1996 年に中央線に導入し，以後逐次線区の拡大を行ってきている．さらに，このシステムは，安全性の確保，指令業務の軽減と業務の効率化，駅業務の省力化，保守作業管理など社員の働きやすい環境を作ることも可能となっている．

一方，新幹線においても，新幹線運行管理システム(COMTRAC)が東北・上越新幹線開業時から設備されてきたが，北陸新幹線への対応をはじめ新しい施策に対応ができないものであった．そこで老朽取り替えに際して，新幹線の業務全体をネットワークで結び，情報の共有化・一元化を行うようにし，また運行管理は危険分散とレスポンスの向上のため自律分散方式とした．このシステムはニュー新幹線総合システム(COSMOS)と呼び，1995 年に使用開始した．

4.1.3　鉄道のサービスを支える自動改札システム

わが国に自動改札機（自動改札機には乗車券を改札する機能の機械と集札する機能の機械，そして改札・集札両用の機械があるがここでは総称して自動改札機と呼ぶ）が本格的に導入されたのは，1971 年頃である．導入に先立って，乗車券の形態やラッシュ時の問題に対応するための研究が 1962 年に近鉄技術研究所で開始された．そして，1965 年（昭和 40 年）に試作機による実験が行われた．さらに翌年の 1966 年には試作機を近鉄「あべの橋」駅に仮設して，定期券の実地試験が行われた．その後，国鉄や大手の鉄道会社が本格的に研究を進め，1967 年大阪万博に合わせて開業した阪急「北千里」駅にわが国初の実用機が導入された．

試作段階の自動改札機は，現在の乗車券のように裏に磁気材料を塗布したものではなく，光学パンチ式や磁気バーコード式が用いられた．すなわち，定期券にデータを穿孔し，それを光学的に読み込んで判定するもので，情報量と耐久性に問題があった．最終的には情報量，耐久性，耐改ざん性，コストなどの面で優れている磁気塗膜方式が採用され，現在も改良を加えて，IC カード乗車券が登場した今も多く利用されている．

一方,乗車券の開発に並行して,自動改札機の具備すべき機能の検討も行われた．その主なものは以下のとおりである．

① 有人改札口と同程度に通過旅客を処理可能なこと
② 乗車券の投入と取り出しが容易に行えること
③ 乗車券が旅客の歩行に合わせて高速に搬送できること
④ 乗車券の情報が迅速に判定できること

⑤ 乗車券が無効な場合は安全に通過阻止が出来ること

当時から，旅客の流動性を安全に実現するため，高速処理と信頼性の確保が鉄道サービスにとって重要であることは今も同じである．

(1) 高速処理への取り組み

自動改札機の機能で重要なのは処理速度である，少なくとも改札機内を歩行する人の速度１ｍ／秒以上にしなければならない[52]．

しかし，当時のマイクロエレクトロニクス技術ではそこまでの高速化は出来なかった．そのため，サンプリング・チェック方式を採用し，見かけの処理速度を確保する方法とした．その後の技術の進歩により，現在では，磁気券の改札機内の搬送速度は，2.5m／秒になっており，人が早足で改札機を通過しても磁気券は前方の取り出し口で待っている．

(2) 信頼性への取り組み

乗車券は金券であるため，その磁気情報を正確にリード／ライトすることは重要な機能である．しかし，導入期の磁気券は記録密度や磁気保持力が低く問題であった．記録密度が低いと新しいサービスを付加する場合に問題となり，磁気保持力が低いとハンドバックの口金やその他の磁気製品の影響を受け，1987 年頃には磁気券の消磁によるトラブルが利用者の３％程度にまで達し，大きな問題となった．なぜなら，現在でも自動改札機の障害（異常券，券詰まり，機器障害など）でドアが閉じる割合は 0.3%程度であり，これを基に自動改札機の通路数などの設計を行っているからである．直ぐに高保持力券に改良した．機器には当たりが少ないことから早期に実現し現在に至っている．

1990 年代に入ると，関東圏で自動化札システムが積極的に導入され，その後 10 年，自動改札機は社会生活にすっかり定着してきた．

そして，2001 年からはこれまでの磁気券とは全く異なるＩＣカード利用した乗車券システムが導入された．このシステムの実用化までには前述の自動改札システム特有の問題点を解決するため，その開発に十数年を費やしている．現在（2006 年 10 月末）では，発行枚数が 1800 万枚を超え，2001 年 11 月の導入後 5 年を経過したが大きなトラブルもなく順調に稼動している．世界的に見ても注目されているシステムである．

4.2　異種統合型自律分散ＩＣカード乗車券システム

4.2.1　従来の磁気式自動改札システム

鉄道乗車券システムの特徴は，多くの旅客利用が特定または不特定の時間に集中するなど負荷変動が大きいため，「自動改札機の高速処理（旅客流動の確保）」が必須である．また，各機器は始発から終電まで継続して使用されるため，「稼働時間が長い（サービスの継続）」ことである．

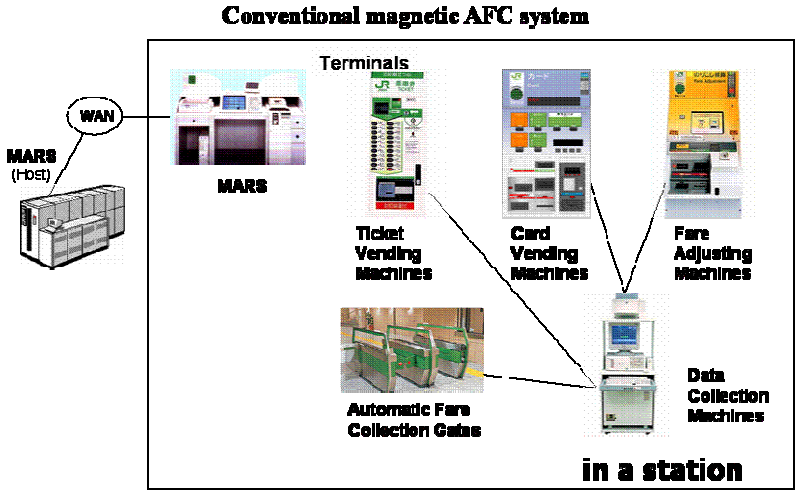


図 4.2　従来の磁気式自動改札システム

図 4.2 に従来の磁気式自動改札システムのシステム構成図を示す．各機器の基本はスタンドアロンで機能する．各機器は駅内の簡易な集中通信ネットワークと接続されている．このため，拡張性が乏しく，保守コストも高い，という問題点がある．

4.2.2　異種統合型自律分散ＩＣカード乗車券システム（Ｓｕｉｃａ）

(1)　基本機能

鉄道乗車券システムの基本機能は「運賃の収受」と「不正カードのチェック」である．

図 4.3 に異種統合型自律分散ＩＣ乗車券システムの機器構成図を示す．システムは「IC カード」「端末（自動改札機など）」「駅サーバ」「センターサーバ」から構成される．機器の故障時も最低限のシステムの稼動が補償されないと駅が大混乱しシステムとしては利用不可である．このため，「自律分散システムアーキテクチャ」としている．

ＩＣカードと端末（リーダ／ライタ）間は高速処理を要求されているため，運賃計算処理は 200msec，非同期，オンラインリアルタイムの制御システムである．これにより，旅客の流動性の確保が図られている．

自動改札機と駅サーバ，センターサーバ間は不正 ID や運賃データが 1 時間から 1 日の周期で送受信される，ネットワーク情報システムである．運賃データは IC カード，端末，センターサーバなどに一定量が蓄積され，情報の確実性，整合性を図っている．

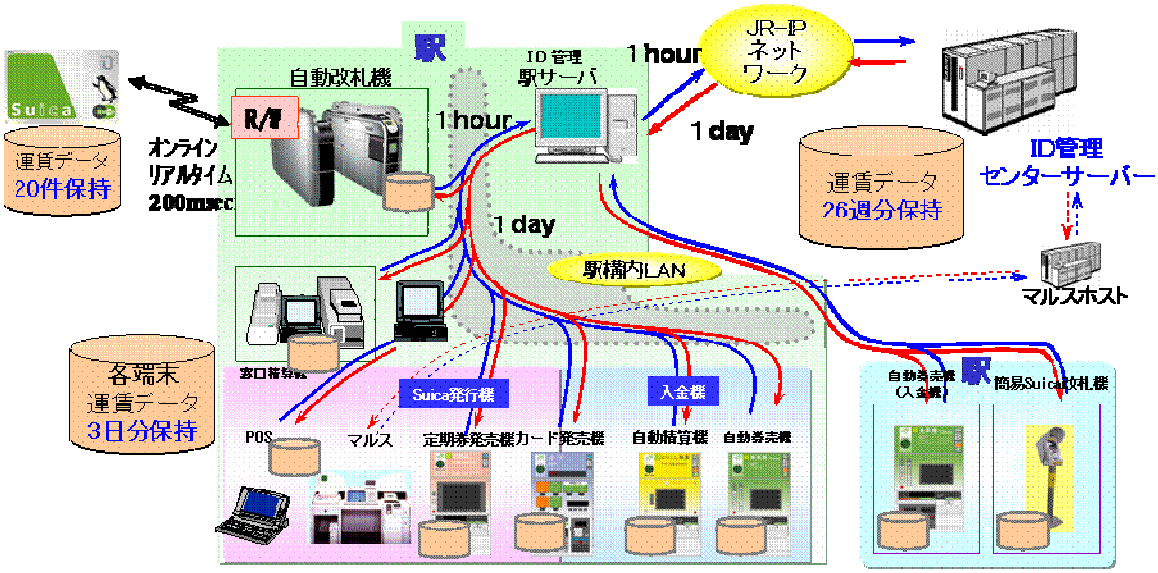


図 4.3　異種統合型自律分散ＩＣ乗車券システムの機器構成図

4.2.3　　システム構成

図 4.5 にシステムの全体構成を示す．このシステムは時間単位の異なる 3 つのデータフィールド(DF)を持つという特異な構造をしている．DF1 において無線通信は 1 秒以内に行われるが，DF2 ではデータは 1 時間毎に流れている．DF3 では 1 日毎と 1 時間毎の 2 種類の時間単位を持っている．これらの時間単位はデータの必要性に応じてそれぞれ設定されており，高速性と高信頼性に寄与している．

図4.4にＩＣカードと端末間の詳細を示す．端末である各出改札機器はＩＣカードとの間で無線通信を行う．ＩＣカードは内容コード（ＣＣ：コンテンツコード）が付いたデータをＤＦ（Data Field）にブロードキャストし，端末（改札機）はセレクトしてデータの収集・処理を行っている（図4.4）．

各端末と駅サーバ間は駅内ＬＡＮで結ばれており，データフィールドを介して自律分散処理をしている．駅レベルでは，駅サーバからＤＦにデータをブロードキャストしており，各端末（自動改札機，自動券売機など）はセレクトしてデータを受け不

正カードチェックや運賃計算などの処理をしている．また，各端末は自律して稼動しており，一部の端末が故障しても他の端末には，その影響を及ぼさないシステムとなっている．駅サーバが故障した場合はセンターからのデータが来ないため一部の機能が利用できないが，各端末は一定量のデータを保存可能なため，駅での出札や改札などの業務処理は継続可能である．また，各駅サーバはDF を介してセンターサーバと結ばれており，乗客がIC カードを持って駅サーバが正常な他の駅へ移動した場合は駅間DFよりデータを取得しており，すべての機能が利用可能である（図4.5）[10,11,12,13]．

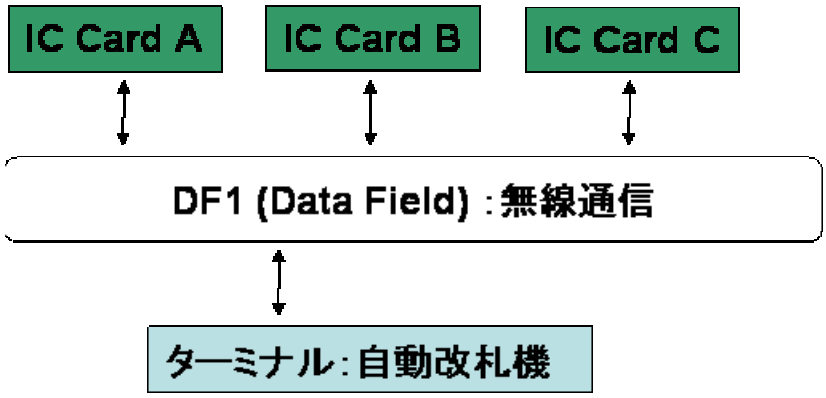


図4.4　IC カードとターミナル間のデータフィールド

前項でも述べたが,IC カード乗車券システムは端末での高速処理が必須要件であるため，IC カードと端末間のＤＦ１ は1 ／ 10 秒単位でデータの授受がなされている．また，端末は前述のとおり一定量のデータ保存が可能であるため駅サーバ間とのＤＦ2ではデータが1 時間単位で授受されている．さらに駅サーバについても一定量のデータ保存が可能であるためセンターサーバ間とのＤＦ3 は1 日に1 回データの授受がなされている．このように「秒」「時間」「日」の3 種類の異なるDF 群を「時間差異種ＤＦ」と呼ぶ[56]．このように自律分散型IC 乗車券システムはシステムニーズに合わせた「時間差異種ＤＦ 」により構成されているという特徴がある．これによりシステムの高速性と信頼性の確保を可能としている（図4.5）．

また，各Suica カードには固有のID 番号（Identification number）が付けられており，Suica を利用すると時間差異種DF を介した各機器に履歴が伝達されバックアップ的に処理データを一定量蓄積（データのダム化）し，障害時に対応する技術が導入されている．これにより，一部の機器の故障時も最低限のシステムの稼動が保証され，不正IC カードの監視や，詳細については後述するが改札機とＩＣカードとの処理時に「データ抜け」が発生した場合でも，データの整合性を確保する「自律分散整合化技術」が導入されている．

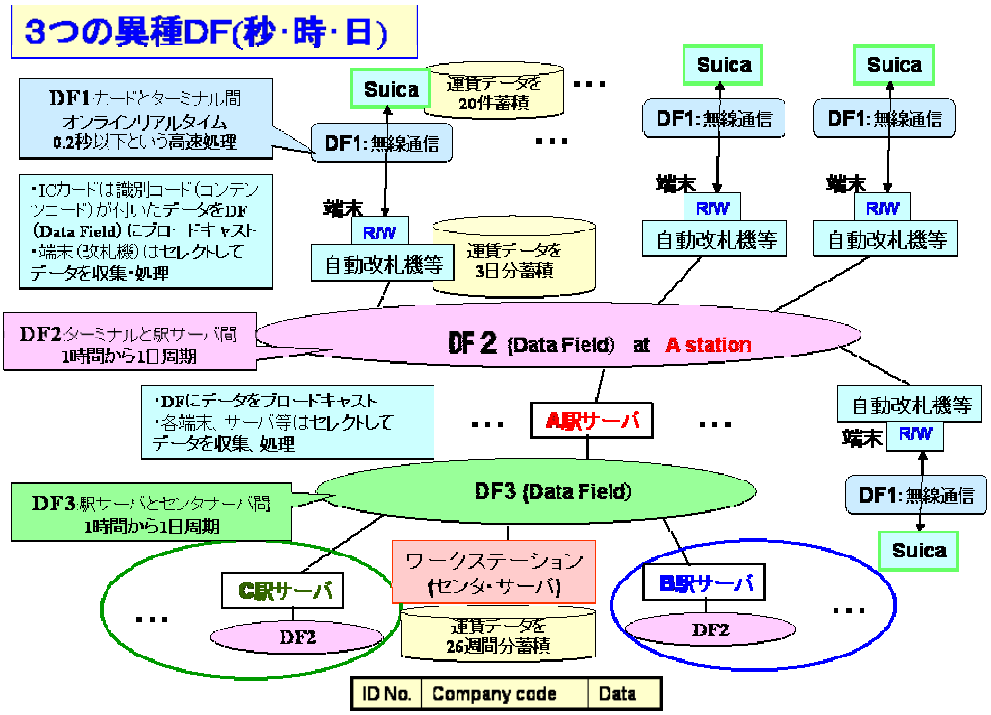


図4.5　異種統合型自律分散型IC カード乗車券システムの構成

IC カードの処理データはカード内に一定量（20 件），端末には一定期間（3 日間）蓄積可能となっている．各出改札機器端末で蓄積しているデータは一定時間毎にDF2 にブロードキャストされ駅サーバは必要なデータを内容コード（ＣＣ）により「選択受信」し，そのデータを蓄積する．さらに，駅サーバはDF3 へデータを一定時間毎に送信する．センターサーバは同様にＣＣにより必要なデータだけ「選択受信」し，一定期間（26 週間）分が蓄積される．これにより，センターサーバや駅サーバが故障した場合であっても，駅の出改札機器は一定期間（3 日間）稼動可能となり，旅客サービスは継続される．復旧後は直ちに各段階で蓄積されたデータは再送信され，データの欠落を防ぐことが出来る．

収集データから判明した不正カードに関する情報はリアルタイムで端末へ配信され，当該カードの利用を停止する．異種ＤＦを用いてのシステム監視により，セキュリティーも大幅に向上している[57]．

4.2.4　　ＩＣカード乗車券システムの課題と解決技術

以上に述べたように磁気式自動化札システムから，設備更新に合わせてＩＣカード乗車券システムを開発・導入したがこの過程でいくつかの課題があった．ここではその主なものとして，高速処理性と高信頼性について述べる．

4.3　高速処理技術

改札機においてカードの不正判定と運賃計算処理を行うが，この場合に特に重要なのは高速処理である．日本の鉄道の特徴は朝夕の猛烈なラッシュである．こうした状況において，旅客が通常，駅構内の平面を自由歩行する速度は１．４m/秒である[46]．また，改札機内の歩行速度は１．０m/秒であり[47]，この速度で通過する人の動作に自動改札機の処理スピードが対応する必要がある．[48,49,50,51]

高速処理は鉄道乗車券ステムにとって必須の要件となる．本節では，まず，ICカードと改札機の高速処理の課題を説明する．この課題を解決するために第３章で提案した，高速処理技術をSuica乗車券システムに導入したので，この技術の有効性の評価を行う．このために，システムのモデル化を行い，シミュレーションにより評価を行う．

4.3.1　システムの高速処理性の課題

磁気乗車券とＩＣカードを自動改札機で処理した場合の比較を図4.6に示す．磁気式と非接触ICカード式の改札機によるカード処理の違いは以下のようになる．磁気式では，改札機に乗車券が入ってくると，データの「読み出し」を行い，それが正規の乗車券であるかを「判定」し，必要な事項を「書き込み」それを「確認」するという４段階をおよそ0.7秒（実測値）で処理している．これに対して，非接触ICカードでは，まず，カードがその電波の中に存在しているかどうかの「存在確認」をし，それが処理するに値するかどうかの「認証」をする．その後はデータの「読み出し」「判定」「書き込み」「再確認」を行う．このように，磁気式では乗車券を投入してから放出するまでの間，すなわち，改札機内の 1.15mの距離を秒速１mで歩行する 1.15 秒以内に処理をすればよい．しかし，ＩＣカードの場合はＩＣカードとリーダ／ライタとの通信・処理エリアが，隣接改札機への影響や技術的問題から，あまり大きく取れないため現在は約0.2mの半球状になっている．このため，処理時間は歩行速度から 0.2sec 以下が必要となる．このため，自動改札機の処理が最難関技術として十数年研究を続けてきた[53,55]．

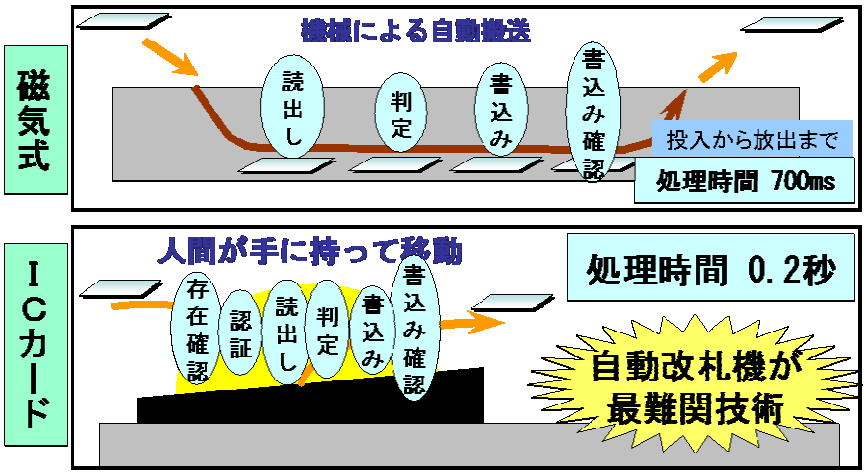


図 4.6　自動改札機による処理の比較

4.3.2　　高速処理のための自律連携処理技術

前述のようにハードウェアによる処理技術は一定の限界がある．一方，鉄道の運賃は駅 2 点間によって決定するため，この処理を図 4.7 に示すように，降車駅で行うと乗車経路によっては複雑な運賃計算となり処理時間が長くなり，スムーズな旅客流動を阻害する恐れがある．このため，第 3 章の 3．3 節「自律分散高速処理技術」で提案した，自律分散システムアーキテクチャを使った新しい「自律連携処理技術（自律分散アルゴリズム)」を鉄道 IC 乗車券システムの運賃計算処理に導入し高速処理を実現したのでその内容を以下に述べる．

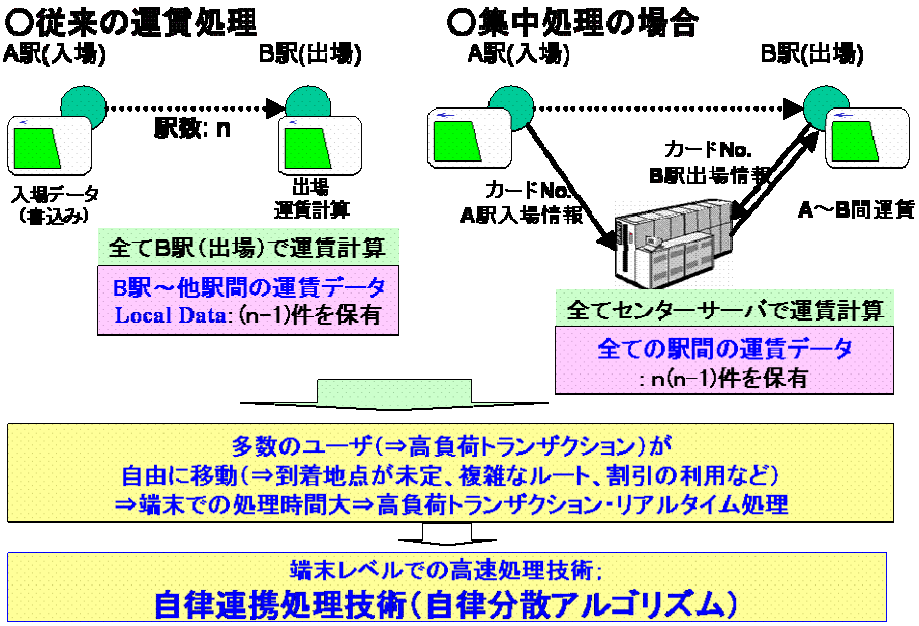


図 4.7　運賃計算処理の高速化の考え方

(1)　運賃計算の自律連携処理（自律分散アルゴリズム）

図 4.8 は，改札機での処理を高速化するため，３．３節の「自律連携処理技術」を運賃計算に適用した場合の概要を示す．

X 駅と Y 駅間有効の定期券を所持する利用者は，定期区間外の A 駅で切符を購入して入場し，同じく定期券区間外の B 駅で出場する時は不足運賃を精算する必要がある．この処理は IC カードを改札機の R/W にかざすだけで，必要な運賃を自動精算する．この複雑な計算が降車駅の改札出口で行われる場合，その処理時間は非常に長くなる．したがって，処理時間を短縮するために，乗車駅における前処理（乗車時に予め定期区間内の乗継駅（仮精算駅）および乗車駅-乗継駅間の運賃を IC カード内に記録する）と降車駅での後処理（乗車ルートの運賃を計算，比較して，最安な運賃を確定する）と自律連携処理する．

具体的に，図 4.8 では，A 駅改札機入場時に A 駅から最短の定期区間内駅（J 駅）を選択し，A-J 間運賃（FAJ）と共にカード内に書き込みを行う．そして，B 駅改札機出場時には B 駅から最短の定期区間内駅（K 駅）を選択する．次に「A-B 間運賃」および「A-J 間運賃＋B-K 間運賃」を比較し安価なルートを選択すると共にカード内処理を行う．

このように運賃計算を各駅の改札機のローカル情報とインプット情報（定期券情報）でローカル処理し，この処理を前処理と後処理に最適分割して計算し，最後に連携処理するものである．この技術により，端末レベル（改札機）での処理時間を大幅に高速化することができる[54]．

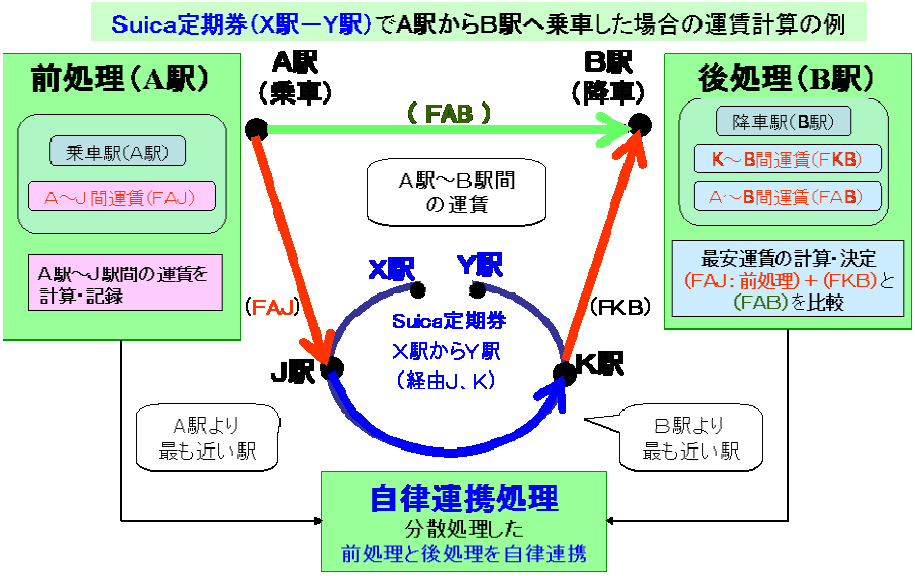


図 4.8　運賃計算の自律連携処理（自律分散アルゴリズム）

(2)　処理時間の定義

４．２．２項でも述べたが，IC乗車券システムの基本機能は，不正カードのチェックと運賃計算処理である．図4.8に示すように，定期区間外のＡ駅から乗車してＢ駅で降車した場合には定期区間を考慮した運賃がＢ駅で精算される．また，Ａ，Ｂ両駅では不正カードのチェックが行われる．

システムの基本処理に必要な処理時間を「カード・端末による分散処理」の場合と「センターサーバによる集中処理」の場合について求める．処理時間は「不正カードのチェック・処理」と「運賃計算」 および付帯する通信時間などに要する時間の合計と定義する．

処理時間 $\mathbf{T = T_1 + T_2 + T_0}$ (4.1)

$\mathbf{T_1}$：不正カードのチェック・処理時間

$\mathbf{T_2}$：運賃計算処理時間

$\mathbf{T_0}$：通信などの付帯処理の時間

①「カード・端末による分散処理」の場合

$$\mathbf{T_{AB} = T_A + T_B}$$
$$\mathbf{= T_{1A} + T_{2A} + T_{0A} + T_{1B} + T_{2B} + T_{0B}} \quad (4.2)$$

$\mathbf{T_{AB}}$：Ａ駅とＢ駅の処理時間の合計

$\mathbf{T_A}$：Ａ駅での前処理の時間

$\mathbf{T_B}$：Ｂ駅での後処理の時間

②「センターサーバによる集中処理」の場合

$\mathbf{T_C = T_{1C} + T_{2C} + T_{0C}}$ (4.3)

$\mathbf{T_C}$：センターサーバでの処理時間

4.3.3　　処理フローとモデル化

４．３．２（２）節で定義した処理時間を図 4.9 の基本乗車パターンに合わせて分散処理の場合と集中処理の場合について，処理フローと，これをモデル化したものを以下に示す．

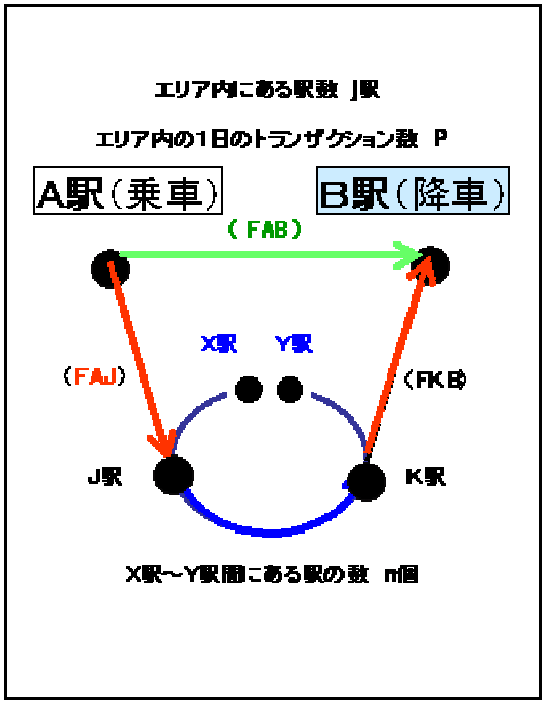


図 4.9　基本乗車パターン

（１）　分散処理

① Ａ駅の処理フロー（図 4.10）
② Ｂ駅の処理フロー（図 4.11）
③ Ａ駅の処理モデル（図 4.13）
④ Ｂ駅の処理モデル（図 4.14）

（２）　集中処理

① 駅・センターサーバ間の処理フロー（図 4.12）
② 駅・センターサーバ間の処理モデル（図 4.15）

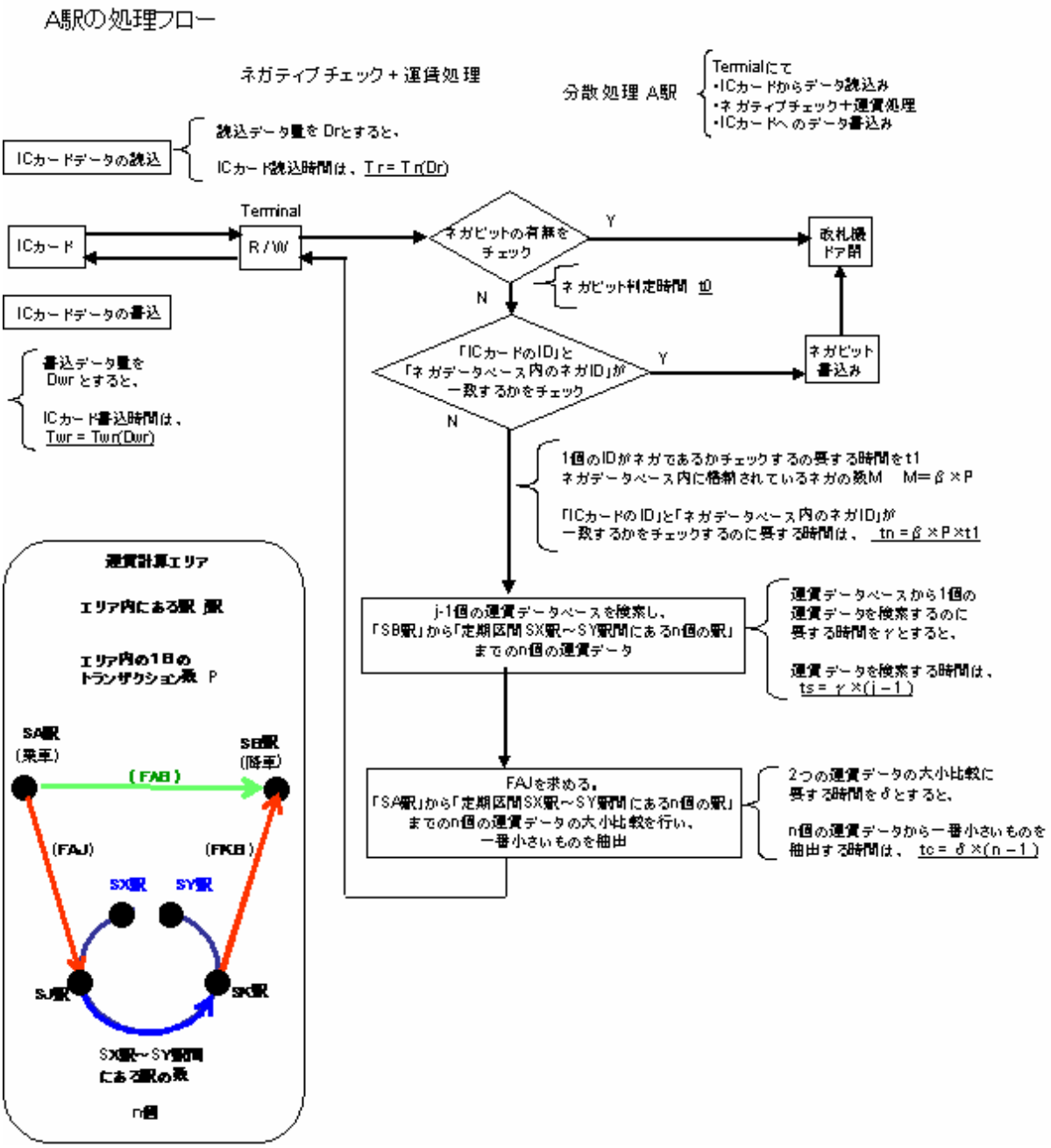


図 4.10　分散処理：A駅の処理フロー

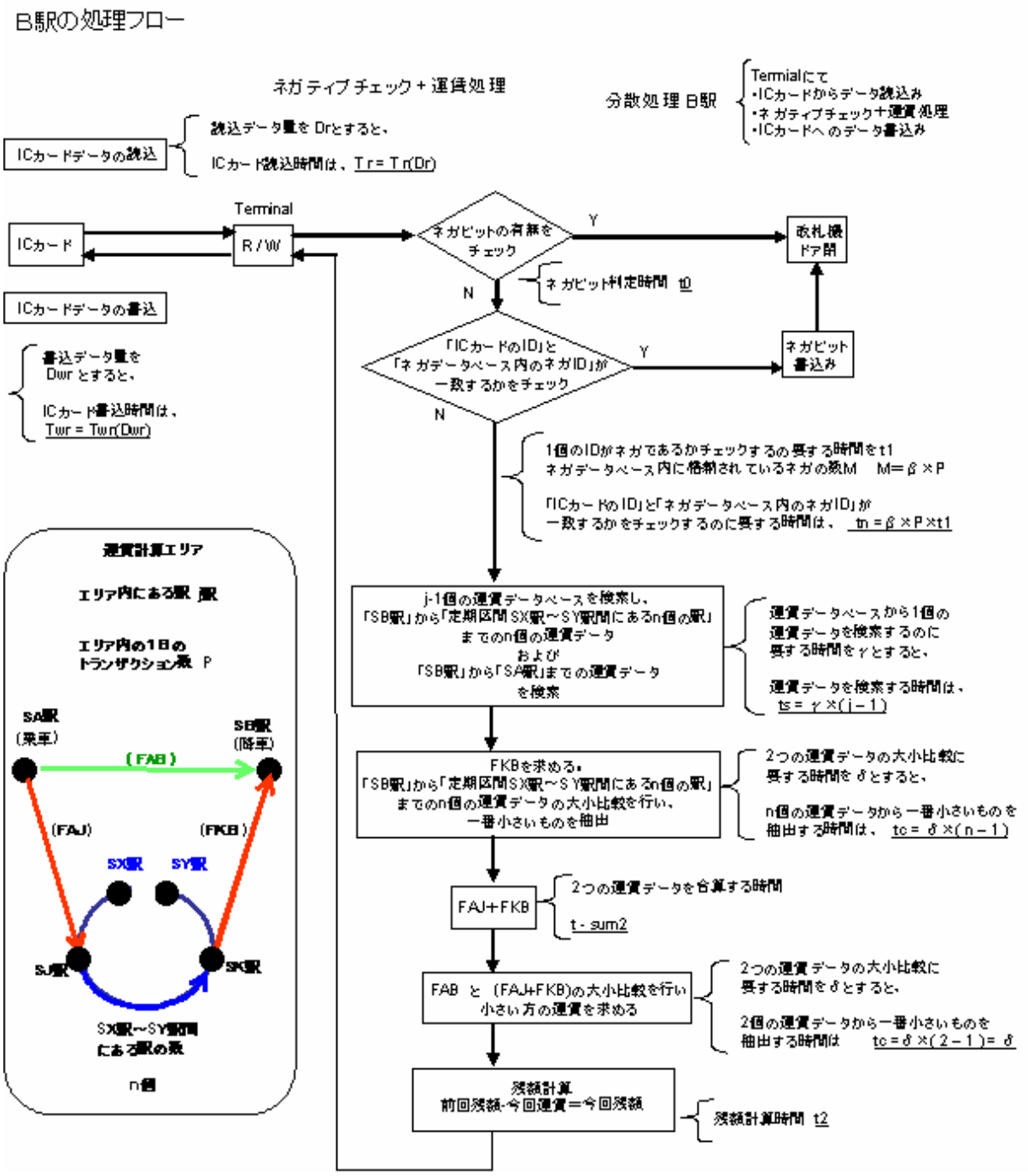


図 4.11　分散処理：B駅の処理フロー

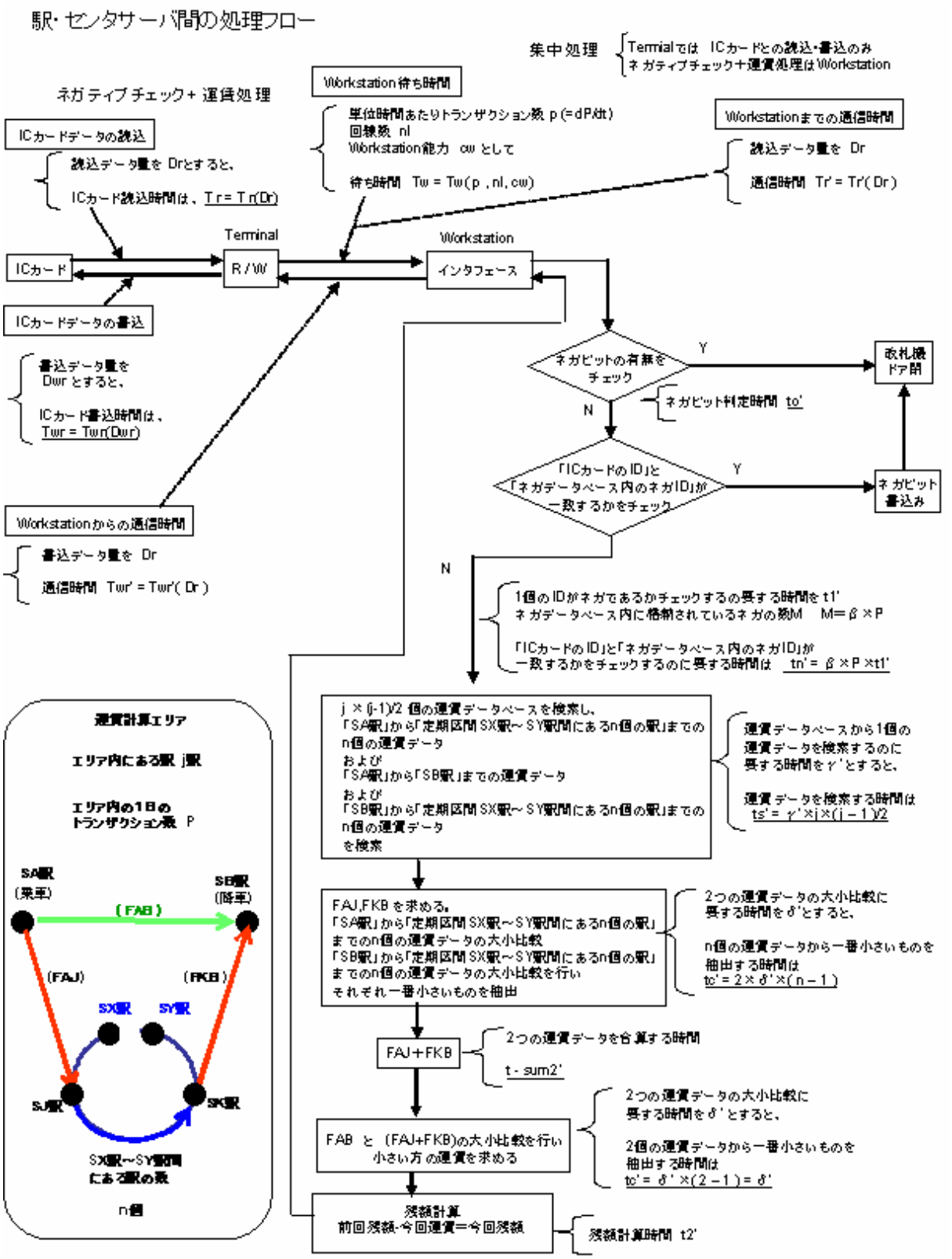


図 4.12 集中処理：駅・センタサーバ間の処理フロー

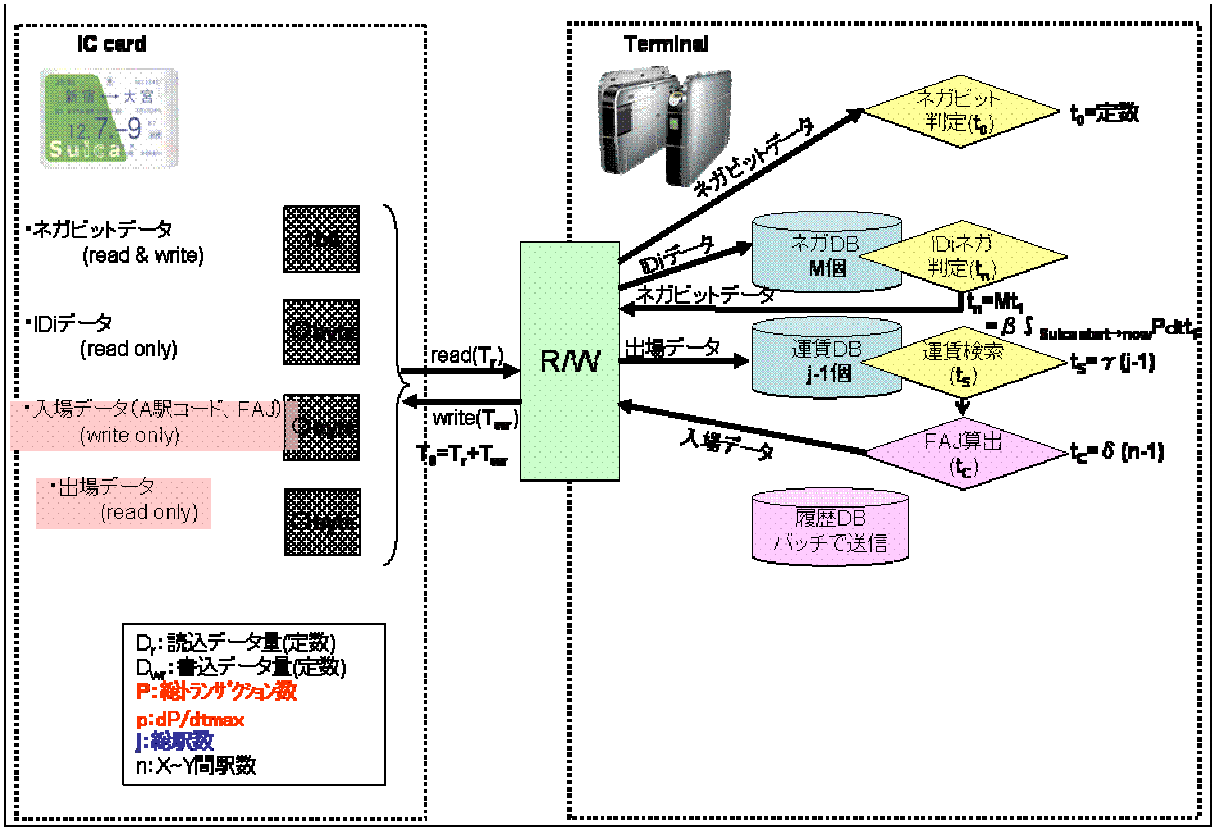


図 4.13　分散処理：A駅の処理モデル

①A駅での前処理の場合を図 4.13 に示す．この場合の処理時間は以下の式で表される．

$$
\begin{aligned}
T_A &= T_{1A} + T_{2A} + T_{0A} = (t_0 + t_n) + (t_s + t_c) + T_{0A} \\
&= \left\{ t_0 + \beta \int_{\text{Suica start} \to \text{now}} pdt \cdot t_1 \right\} + \{\gamma(j-1) + \delta(n-1)\} + \{T_r(D_r) + T_{wr}(D_{wr})\} \\
&= t_0 + \beta t_1 \int_{\text{Suica start} \to \text{now}} pdt + \gamma j + \delta n + T_r(D_r) + T_{wr}(D_{wr}) - \gamma - \delta
\end{aligned}
\tag{4.4}
$$

t_0：ネガビットチェック時間　　β：不正カード発生率

t_n：カード ID チェック時間　　γ：1 個データ検索時間

t_s：運賃データ検索時間　　δ：2 つデータ比較時間

t_c：最小運賃データ抽出時間　　p：トランザクション数

t_1：一個 ID チェック時間　　j：総駅数

T_r(Dr)：カード読込時間　　n：X～Y 間駅数

T_r(Dr)：カード書込時間

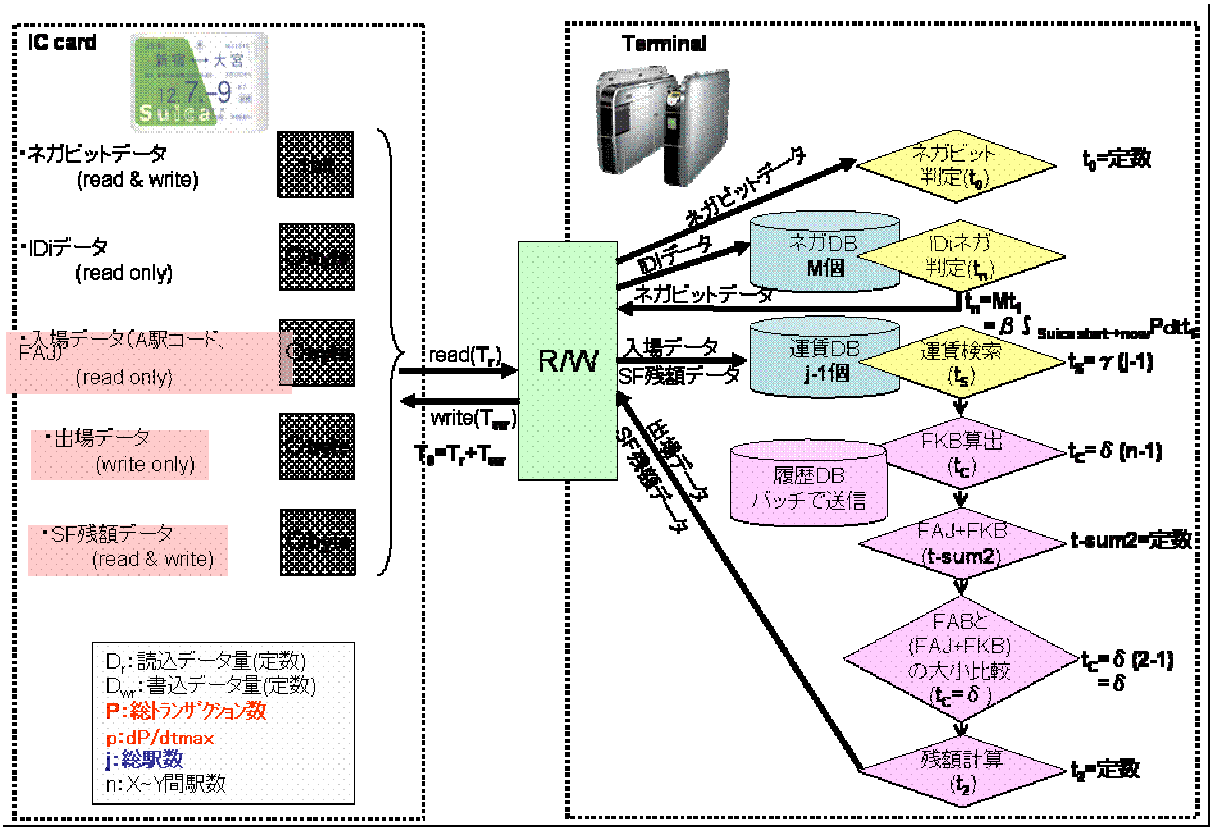


図 4.14　分散処理：B駅の処理モデル

②同様にB駅の場合の処理時間は以下の式で表される.

$$\begin{aligned} T_B &= T_{1B} + T_{2B} + T_{0B} = (t_0 + t_n) + (t_s + t_c + t_{sum2} + t_c + t_2) + T_{0B} \\ &= \left\{ t_0 + \beta \int_{t=0}^{t=now} pdt \cdot t_1 \right\} + \{\gamma(j-1) + \delta(n-1) + t_{sum2} + \delta + t_2\} + \{T_r(D_r) + T_{wr}(D_{wr})\} \\ &= t_0 + \beta t_1 \int_{t=0}^{t=now} pdt + \gamma j + \delta n + t_{sum2} + t_2 - \gamma + T_r(D_r) + T_{wr}(D_{wr}) \end{aligned} \tag{4.5}$$

t_{sum2}: 2つ運賃データ合算時間, t_2: 残額計算時間

以上の結果から，分散処理の場合の処理時間 $T_{AB} = T_A + T_B$ である．式からもわかるように，処理時間は **P**：総トランザクション数と **j**：総駅数の関数になっている．Pやjが増加するとTも増加する．

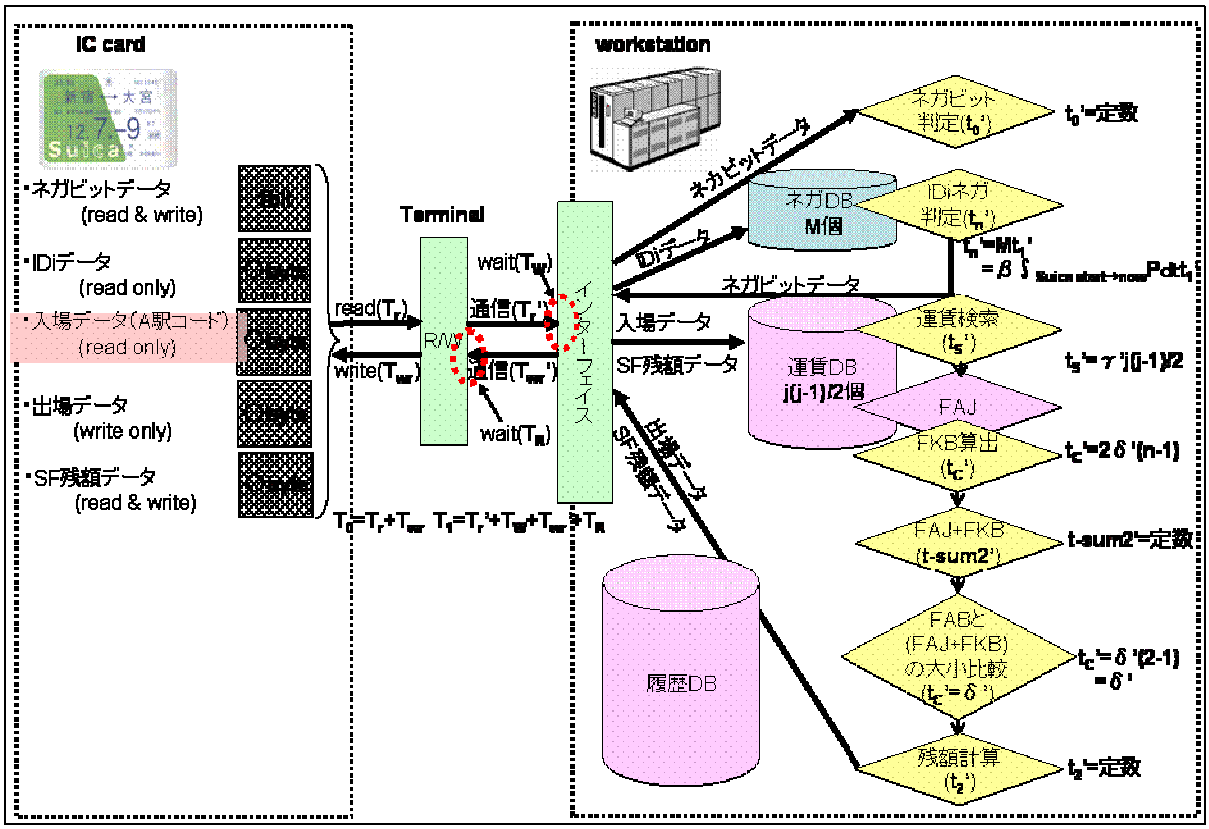


図 4.15　集中処理：駅・センタサーバ間の処理モデル

センターシステムで集中処理する場合はＡ駅入場時は不正カードチェックのみとし，Ｂ駅出場時にカードチェックと運賃計算処理を行う，この場合の処理時間は以下の式で表される．

$$
\begin{aligned}
T_C &= T_{1C} + T_{2C} + T_{0C} \\
&= (t_0' + t_n') + (t_S' + t_C' + t_{sum\,2}' + t_C' + t_2') + (T_0 + T_1) \\
&= T_r(D_r) + T_{wr}(D_{wr}) + T_r'(D_r) + TW(p, nl, cw) \\
&\quad + T_{wr}'(D_{wr}) + TR(p, nl', cr) + t_0' \\
&\quad + \beta \int_{t=0}^{t=now} pdt \cdot t_1' + \frac{\gamma' j(j-1)}{2} + 2\delta'(n-1) \\
&\quad + t_{sum\,2}' + \delta' + t_2' \\
&= \beta t_1' \int_{t=0}^{t=now} pdt + \frac{\gamma' j^2}{2} - \frac{\gamma' j}{2} + 2\delta' n \\
&\quad + TW(p, nl, cw) + TR(p, nl', cr) \\
&\quad + T_r(D_r) + T_{wr}(D_{wr}) + T_r'(D_r) + T_{wr}'(D_{wr}) \\
&\quad + t_0' - \delta' + t_2'
\end{aligned}
\qquad (4.6)
$$

$T_r'(D_r)$：読込データの通信時間，$T_{wr}'(D_{wr})$：書込データの通信時間

ここで，待ち時間 TW，TR は以下のとおりである．

$$TW = \frac{\rho(nl\rho)^{nl}}{nl!(1-\rho)^2\sum_{i=0}^{nl-1}\left\{\frac{(nl\rho)^i}{i!}+\frac{(nl\rho)^{nl}}{nl!(1-\rho)^2}\right\}p}$$

$$TR = \frac{\rho'(nl'\rho')^{nl'}}{nl'!(1-\rho')^2\sum_{i=0}^{nl'-1}\left\{\frac{(nl'\rho')^i}{i!}+\frac{(nl'\rho')^{nl'}}{nl'!(1-\rho')^2}\right\}p} \qquad (4.7)$$

$$\rho = \frac{p}{nl \cdot cw}$$

$$\rho' = \frac{p}{nl' \cdot cr}$$

nl: workstation の回線数(定数)

cw: workstation の処理能力

ρ：workstation の 1 つの回線の平均処理率(定数):

nl': R/W の回線数(定数)

cr: R/W の処理能力 1 つの回線の平均処理率(定数)

ρ': R/W の 1 つの回線の平均処理率(定数)

4.3.4　シミュレーションと評価

(1)　シミュレーション

４．３．３節の処理フローとモデル化から処理時間について導き出した式より処理時間が「駅数」及び「トランザクション数」によってどのように変化するかシミュレーションを実施した．

ここで実際の鉄道線区を想定して，駅数が比較的少ない場合（十数駅）から中規模（数十駅）大規模（数百駅）を想定して，処理時間と駅数の関係をグラフ化したものが図 4.16 である．同様に，処理時間とトランザクション数の関係をグラフ化したものが図 4.17 である．シミュレーションにあたっては，表 4.1 のパラメータを使用した．

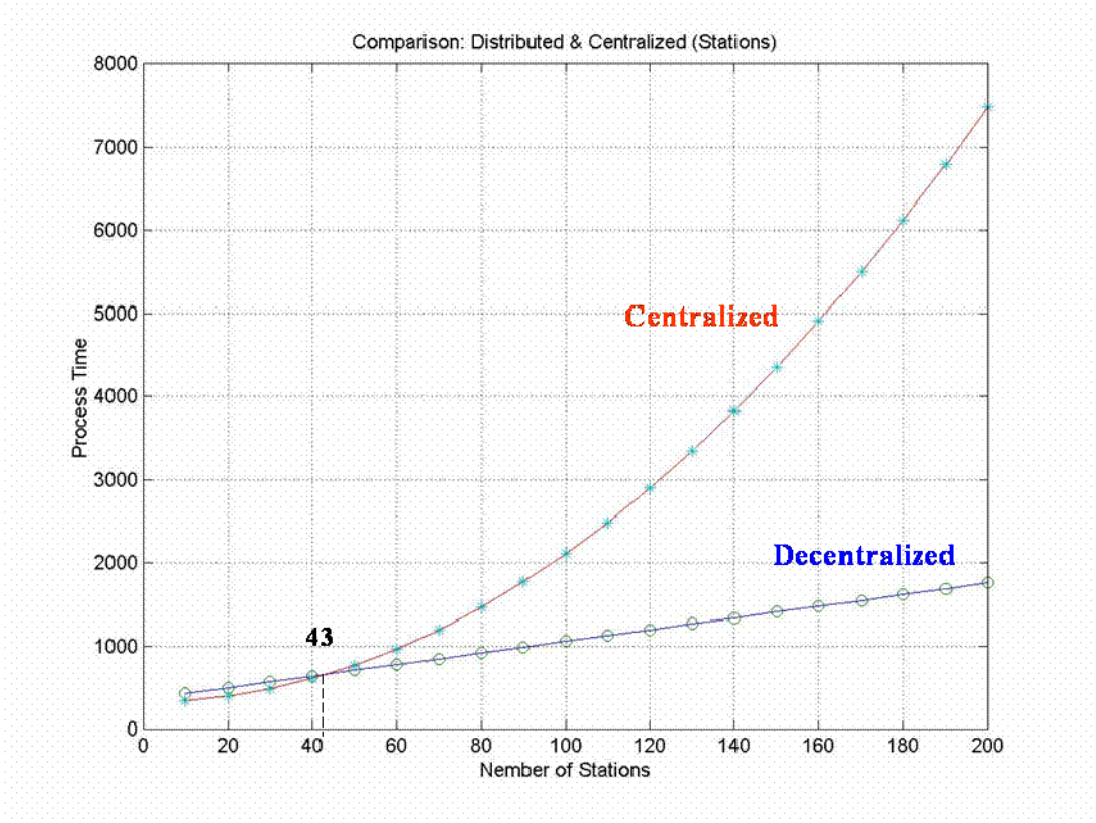


図 4.16　処理時間と駅数（集中・分散）

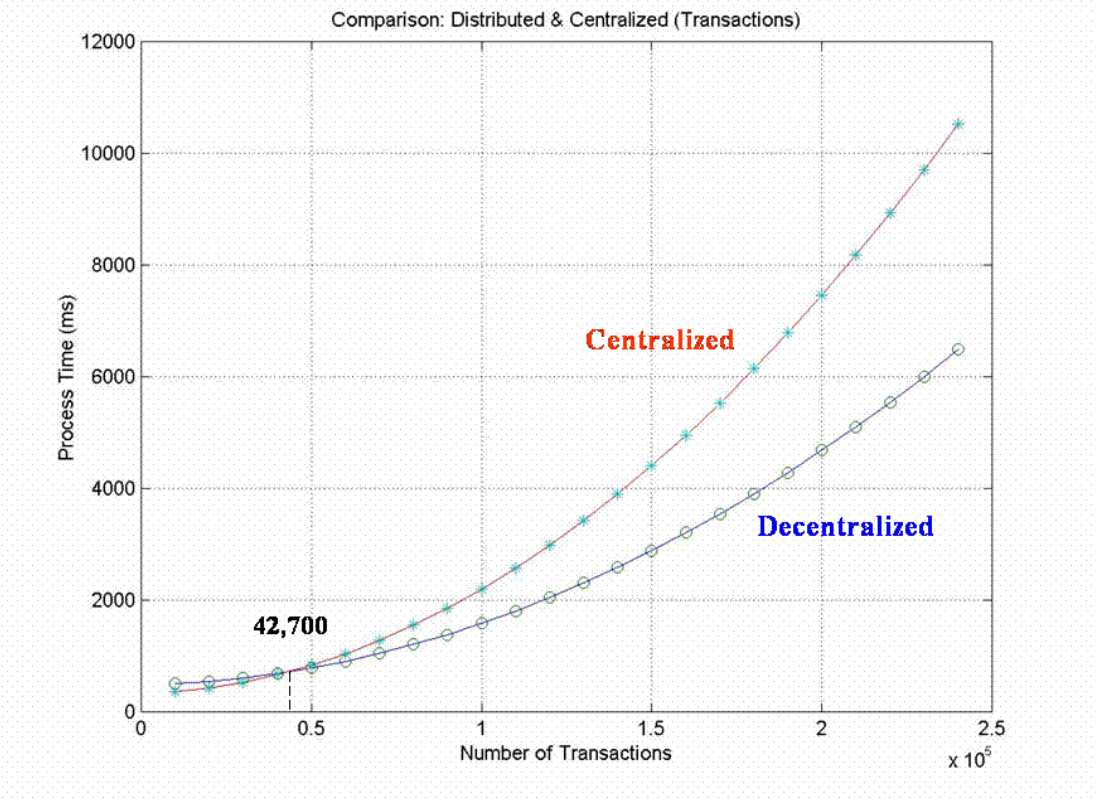


図 4.17　処理時間とトランザクション数（集中・分散）

表 4.1　シミュレーションにおけるパラメータ

	分散	集中
Tr(Dr)	40	
Twr(Dwr)	80	
Tr'(Dr)	-	10
Twr'(Dwr)	-	10
t_0	5	2
t_1	10	4
β	0.01*	
t_s	10	4
t_c	5	2
$t_{sum2}+t_2$	6	1
cw	-	40*
cr	-	20*
nl=nl'	-	10*

(*を除き単位 ms)

(2)　評価

前項のシミュレーション結果からわかるように駅数が小規模な場合（43 駅以下）は集中処理が有効であるが中規模（43 駅以上）は分散処理が有効である.
また，処理時間とトランザクション数の関係では，トランザクション数が少ない場合（数万件）は集中方式が有効であるが，多い場合（42,700 トランザクション/day 以上）は分散方式が有効である.

これらの結果からもわかるように「自律連携処理技術（自律分散アルゴリズム)」は駅数が多く，トランザクション数も多い，都市型の鉄道などには有効な技術であることが証明された．同様に，端末数が多く，高負荷トランザクションの他のシステム，例えば，インターネットの E-Comerce などで順次サービスを受ける場合に割引券を利用する場合としない場合の処理を高速に行うケースなどにも有効な技術である.

4.4　　システムの信頼性

本節では，まず，ＩＣカードと端末間の通信処理に起因する問題点を明らかにし，この対応として，システムの信頼性を確保する技術として，「自律分散整合化技術」のIC カード乗車券システムへの適用を解説する.その後，第３章で述べた機能信頼性評価技術により,自律分散整合化技術の有効性を評価する.比較対象としては，整合化技術を導入した自律分散システムと，整合化技術を用いないシステムとして従来の集中管理型乗車券システムを扱う．まず，各システムのモデル化を行い機能信頼性の式を導出する．最後にシミュレーションを行い，結果の考察を行う.

4.4.1　　ＩＣカードと端末間の通信処理

(1)　IC カードと端末間の通信処理

具体的に改札機によるカード処理がどのように行なわれているか述べる．まず，カードがその電波の中に存在しているかどうかの存在確認をし，それが処理するに値するかどうかの相互認証をする．その後は判定に必要なデータをカードから読み取り，不正カードチェック，運賃計算,SF 残金が十分か等の判定を行い，結果のデータをカードに書き込み，最後に再確認を行なう（図4.18）．改札側は判定後のデータを仮データとしてデータベースに登録を行い，最後の再確認でカードからAck信号(更新完了応答）が返ってくると，その仮データを確定データとして確定する.

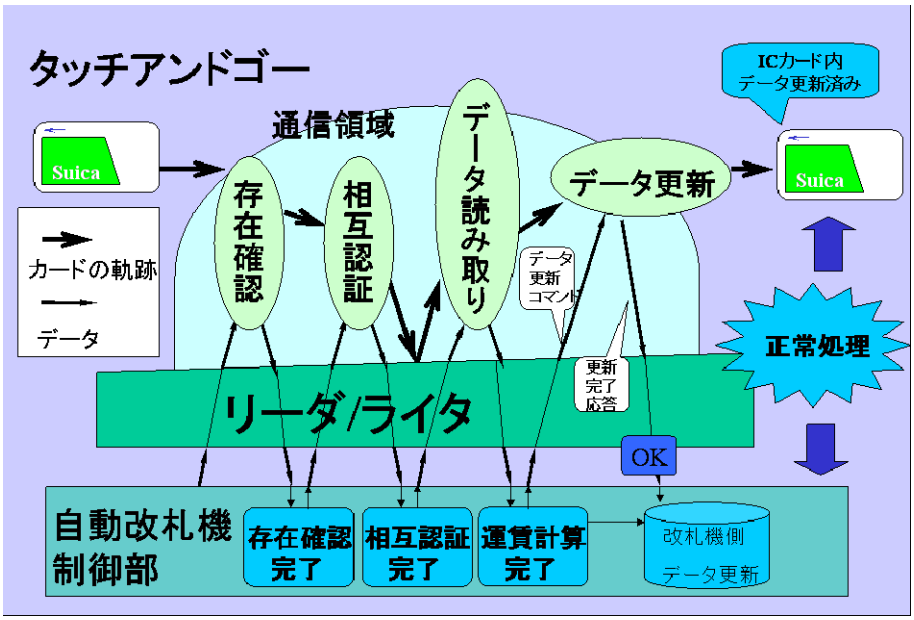


図 4.18　IC カードとターミナル間の処理:正常時

(2)　無線通信に起因する処理未了エラー

前述したように，Suica システムではカード側のデータ処理が完了したことをR/W側が受け取って，処理を完了する仕組みである．しかし，リーダにカードを挿入して使用する接点形IC カードとは異なり，非接触式であるが故に，カードのかざし方等旅客の使用方法に依存してしまうという欠点を持つ．一連の処理が完全に終了する前にエリアの外にカードを出されてしまい処理未了になる場合がどうしても発生する．以下にそのような処理未了エラーのパターンについて説明していく．

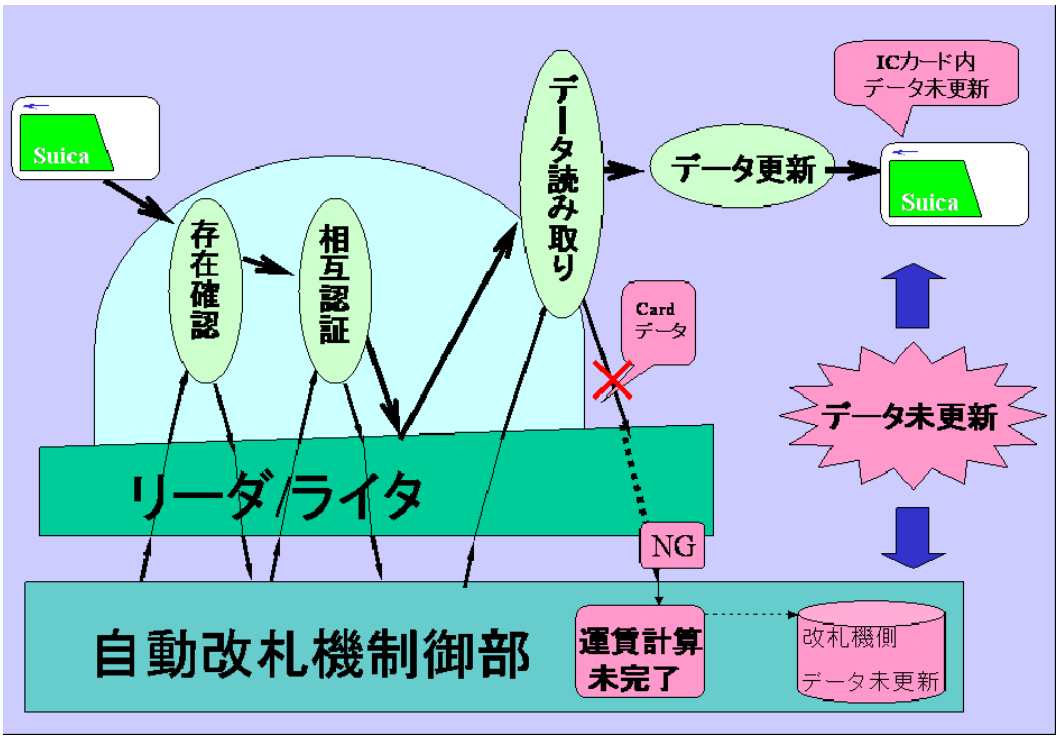


図4.19　IC カードとターミナル間：エラーe_0

まず図4.19に示すエラーe_0 は，データ読み取りを完了する以前に，カードがエリアの外に出てしまった場合である．R/W 側にデータが届いていないので当然運賃計算なども実行できず，カード側もR/W も更新がなされていない状態になる．

続いて図4.20に示すエラーe_1は，データ読み取り後，運賃計算結果をカードに書き込む以前にカードがエリアの外に出てしまった場合である．R/W 側には運賃計算結果のデータはあるがカードからAck 信号が返ってこないためデータを仮更新状態のまま確定できない状態になる．

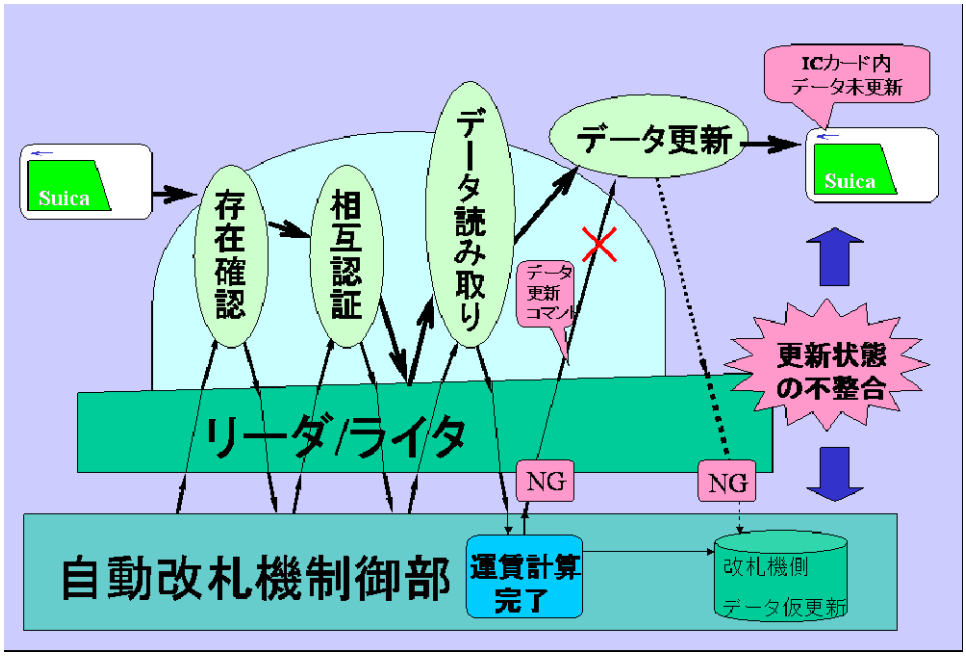


図4.20　IC カードとターミナル間：エラーe_1

図4.21のエラーe_2 は，カードへの書き込みは成功したが最後のAck 信号が返らなかった場合である．このエラーもR/W側には運賃計算結果のデータはあるがカードからAck 信号が返ってこないためデータを仮更新状態のまま確定できない状態になる．カード側のデータは更新されているが，R/W 側が更新されていないという状態の不整合が生じる．

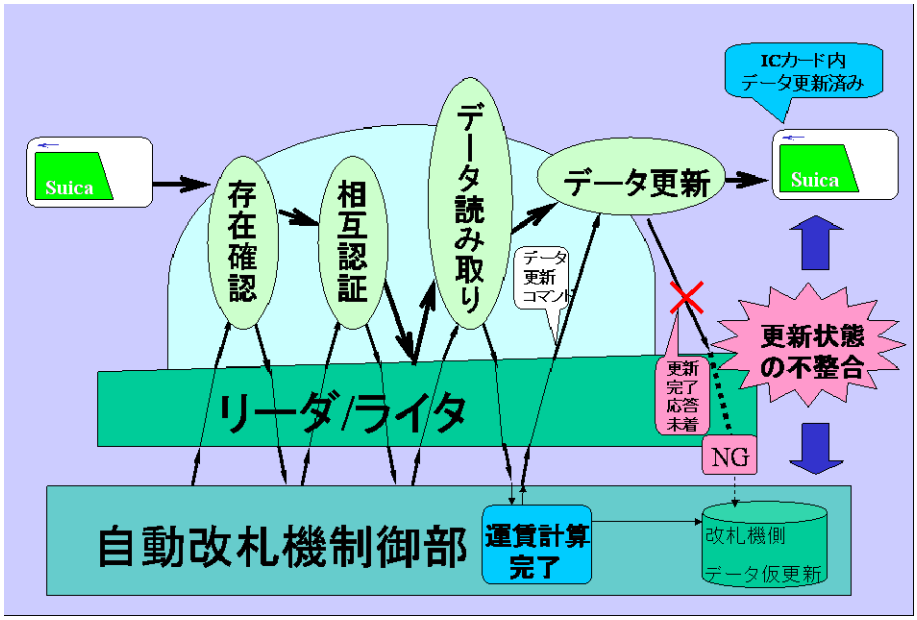


図4.21　IC カードとターミナル間：エラーe_2

(3)　エラーに対しての従来技術での対応

これらのエラー発生に対して，システム側がどのように対処するかについて述べていく．まず従来技術での対応の仕方について述べる．各エラーと処理のパターンを表4.2に示す．従来技術ではR/W 側(センター側）をメインのデータベースとし，カード側のデータはバックアップとしてシステムを構築していた．そして，カード側とR/W 側のデータが不整合な状態でのシステム稼動を許していない．

本来，トランザクション処理では，複数の処理がひとつの処理単位としてまとめられ，トランザクションと呼ばれる単位で監視され「すべて成功」か「すべて失敗」のいずれかであることが保証されなければならない．従来技術でのシステムでは,Ack 信号が返ってこない場合は計算結果を全て破棄し，改札扉を閉める事によって，トランザクションをアボートする．つまり，どちらか一方が失敗したらもう片方も失敗させ，どちらも成功したときに，初めて全体を成功としようとする．しかし,$e2$ のエラーの場合，既にカードには書き込まれてしまっており，トランザクションをアボートしようにも，肝心のカードは通信エリア外にあるためその内容を棄却させる事ができない.このような状態になった時は窓口に行ってカードの内容を書き換えて復旧するしかない.

例えば,A 駅で乗車しようとしてe_2 のエラーが生じた場合，カードにはA 駅で乗車したというデータが書き込まれ,R/W 側にはデータが残らない．もう一度タッチしてもR/W 側からは，そのカードはA 駅から乗車したはずなのに,A 駅から乗車しようとしていると見えてしまう.

表4.2　従来：発生するエラーと処理のパターン

パターン	カード内データ Card	改札機内データ R/W	発生する障害	処理
①正常処理	○	○	なし	旅客通過
②エラー e_0 運賃計算前	×	×	流動阻害	改札扉閉め
③エラー e_1 C write miss Ack error	×	△ ⇒ ×	流動阻害	改札扉閉め
④エラー e_2 C write Ack error	○	△ ⇒ ×	流動阻害	改札扉閉め ⇒窓口で復旧

○ :確定データ　× :データ無し　△ :未確定データ

4.4.2　　自律分散整合化技術

正常処理(図4.22)に対し，第４．４．１項で述べた処理未了エラーe_2 が発生した場合は端末側にあるデータを「仮一件明細データ」としてDF に送りサーバ側はそれを「選択受信」し蓄積しておく(図4.23)．次の処理で送られてきたデータと前のデータとの整合性を確認し，仮データから確定データへ補正する(図4.24)．この方法は各端末による部分処理を統合処理することによって，結果としてデータの信頼性を確保する技術である[57]．

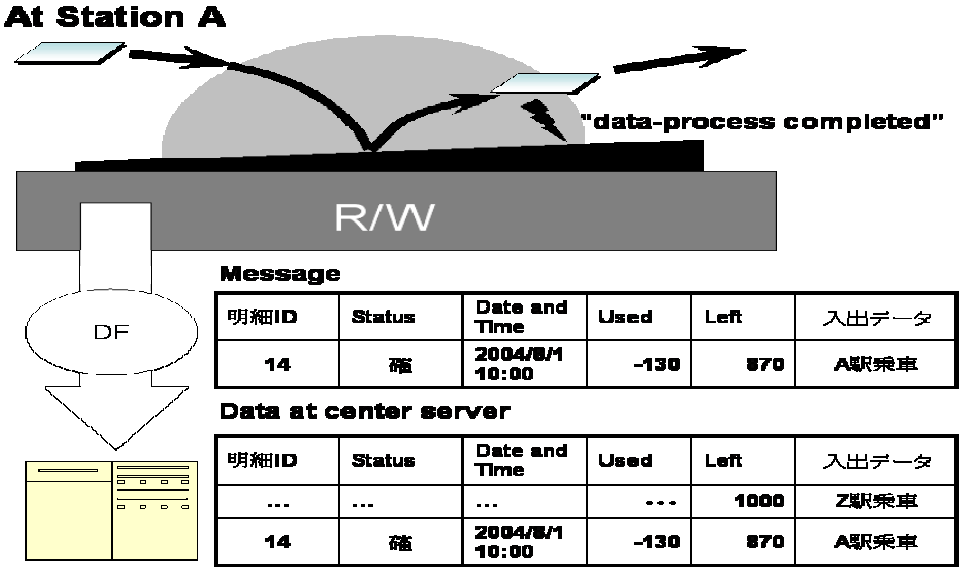


図4.22　自律分散整合化技術(1)：正常時

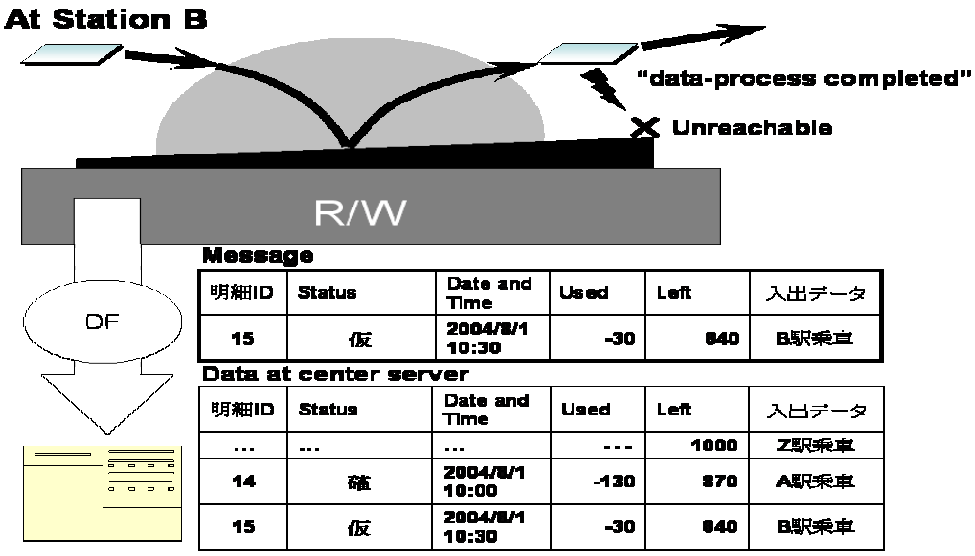
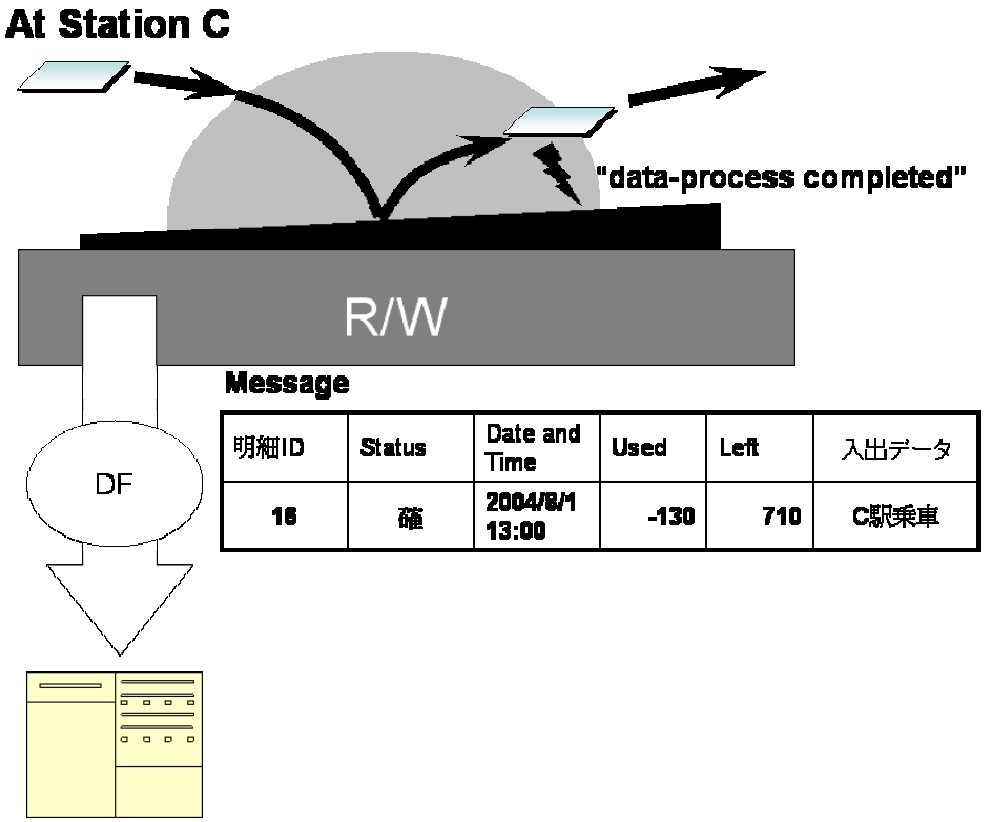


図4.23　自律分散整合化技術(2)：データ抜け時



明細ID	Status	Date and Time	Used	Left	入出データ
16	確	2004/8/1 13:00	-130	710	C駅乗車

Data at center server

明細ID	Status	Date and Time	Used	Left	入出データ
…	…	…	…	1000	Z駅乗車
14	確	2004/8/1 10:00	-130	970	A駅乗車
15	仮	2004/8/1 10:30	-30	940	B駅乗車
16	確	2004/8/1 13:00	-130	710	C駅乗車

仮データは前後の確定データから正当性を確認

センターサーバは仮データを確定データへ補正する。

Data at center server

明細ID	Status	Date and Time	Used	Left	入出データ
…	…	…	…	1000	Z駅乗車
14	確	2004/8/1 10:00	-130	970	A駅乗車
15	確	2004/8/1 10:30	-30	940	B駅乗車
16	確	2004/8/1 13:00	-130	710	C駅乗車

図4.24　自律分散整合化技術(3)：データの補正

(1)　整合化技術導入後のエラーに対しての対応

この自律分散整合化技術を導入した場合の処理のパターンについて説明する．Suicaシステムではカードのデータをメインとし,R/W 側をバックアップとして捉えている．Ack 信号が返って来ない場合，端末側にあるデータを「仮データ」として登録しておき，次回以降の処理が正常の場合には，正常なデータと前回の正常なデータとからデータの整合性を確認し,仮データから確定データへ補正する(表4.3)．つまり,Suica システムではシステム稼動の継続を重視しており，Ack 信号が返って来ない場合でも改札を通過させ，その後の使用時にその正当性を確認する．

この技術によって，カード側に正常にデータが書き込まれている場合はAck 信号が届かなくても旅客に意識させる事なく，流動の阻害を防ぐ事ができる．

表4.3　整合化技術導入後：発生するエラーと処理のパターン

パターン	カード内データ Card	改札機内データ R/W	発生する障害	処理
①正常処理	○	○	なし	旅客通過
②エラーe_0 運賃計算前	×	×	流動阻害	改札扉閉め
③エラーe_1 C write miss Ack error	×	△ ⇒ ×	流動阻害	旅客通過 ⇒次回改札扉閉め ⇒窓口で復旧
④エラーe_2 C write Ack error	○	△ ⇒ ○	一件明細抜け	旅客通過 ⇒整合化技術

○ :確定データ　× :データ無し　△ :未確定データ

整合化技術導入前と導入後の各サブシステム間のデータの送受信について図4.25から図4.28に示す．通常，センターサーバはカードのデータをバックアップ的に保存しておく．データ抜けが発生した場合は，前後のデータから正当性を確認してから補正を行なう(図4.25)．

しかし，カードに保存できる件数を超えるデータ抜けが連続で発生した場合，整合化できる許容範囲をオーバーしたと考え，そのカードの使用を禁止する(図4.26)．これはカード側に正常に処理が行なわれたデータが押し出されて1つも残っておらず．非常に信頼性にかける状態にあるためである．

また，カード紛失時やなんらかの理由で不正カードが発覚した場合など緊急的にセンターから各駅にそのカードの利用を禁止するネガデータが送られる(図4.27)．整合化技術を用いない従来の場合はデータ抜けが発生する度にカードの使用を停止させその度に復旧作業を行なっていた(図4.28)．

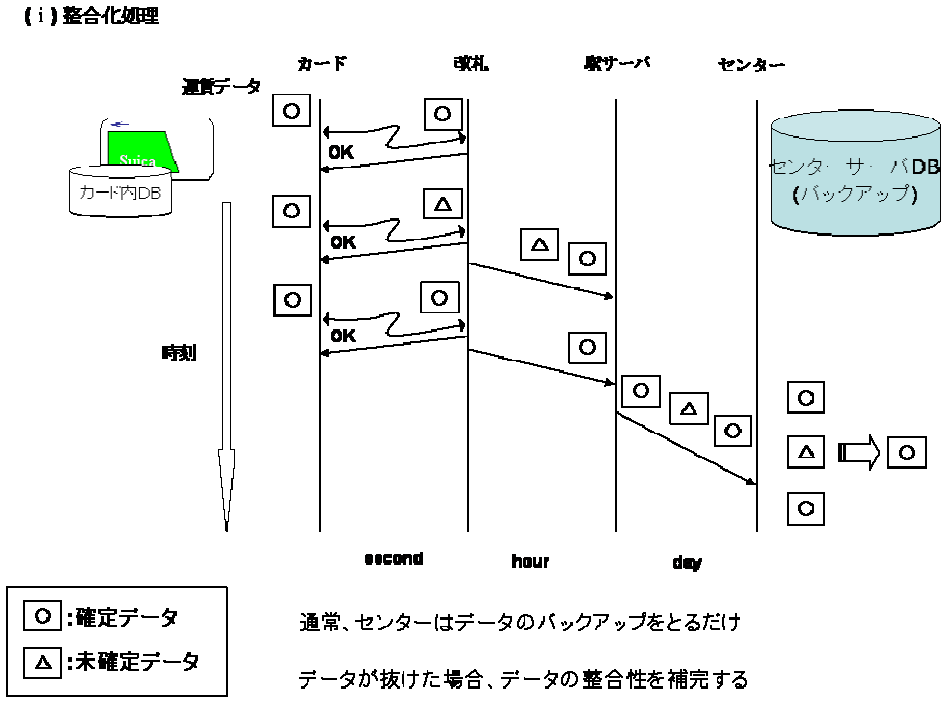


図4.25　各サブシステム間の通信(整合化技術)

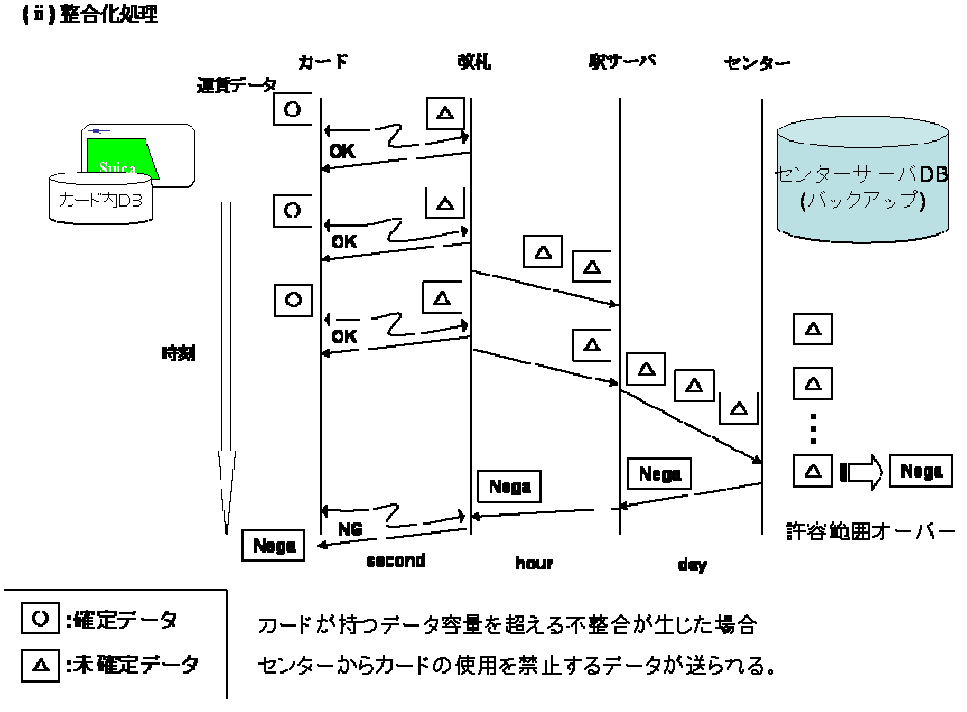


図4.26　各サブシステム間の通信(整合化技術)

(iii) 整合化処理

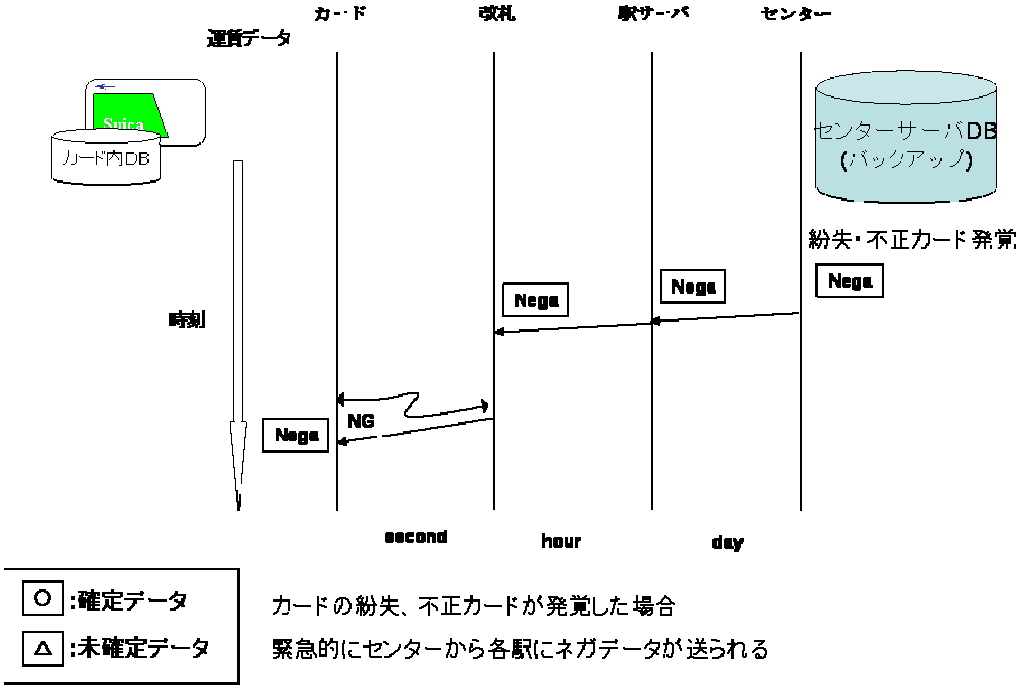


図4.27　各サブシステム間の通信(整合化技術)

集中管理型

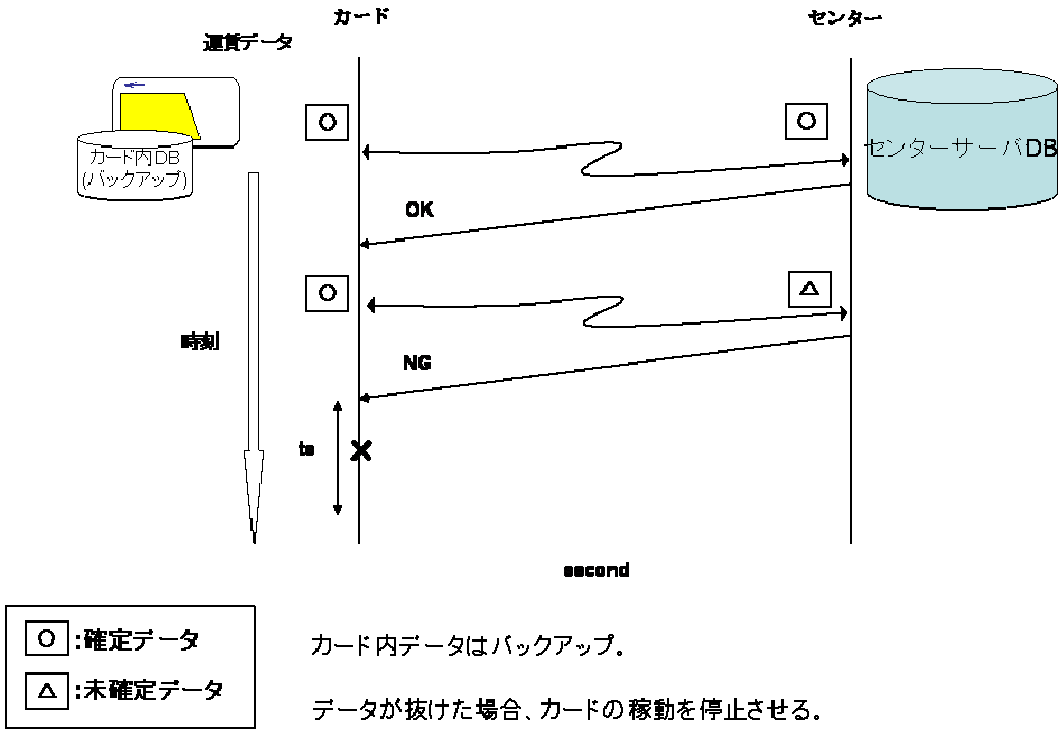


図4.28　各サブシステム間の通信(従来)

4.4.3　　システムのモデル化

「自律分散整合化技術」をＩＣカード乗車券システムに導入した場合のシステムを「自律分散型システム」と呼ぶ．本節では３．４．４項で述べた「機能信頼性評価技術」を異種統合型IC カード乗車券システムへ適用し，「自律分散整合化技術」のアシュアランス性を評価する．自律分散型システムの評価の比較対象としては，整合化技術を用いないシステムとして従来技術の「集中管理型システム」を扱う．まず，各システムのモデル化を行い機能信頼性の式を導出する．その後，次項４．４．４でシミュレーションを行い，その結果の考察を行う．

(1)　システムのモデル化

システムの機能信頼性を評価するためのモデル化の前提として，機能量と機能達成度について，以下のように定義する．

ＩＣカード 乗車券システムの基本機能は，不正カードのチェックと運賃計算処理である．このチェックと運賃計算処理はカード内データを基本に処理が行われるので，各サブシステムの機能量として，「カード内データ」と「サーバ内データ」が一致した状態の時，そのデータを「一致データ」と定義する．また，ＩＣ 乗車券システムに求められる機能達成度（式3.14）を「データの一致度」（図4.29）と定義する．ここで「データの一致度」とは，「サーバ」と「カード」内のデータが一致している度合い（例えば「一致データ」の件数の割合など）を表すものである．

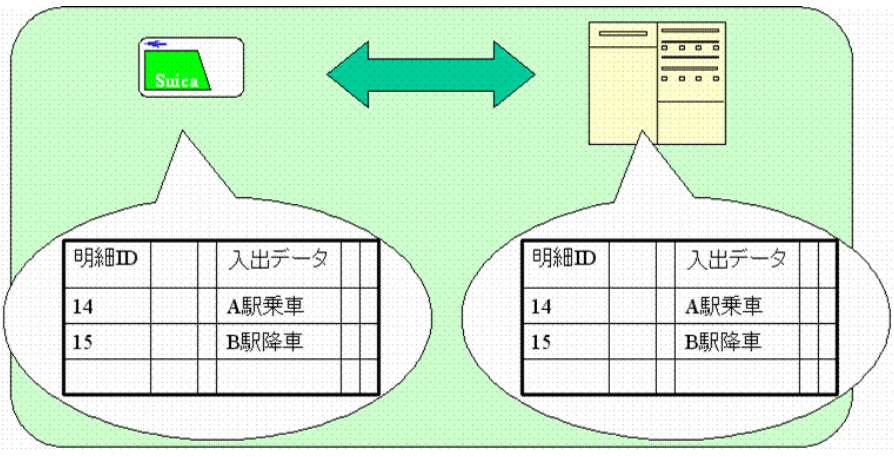


図4.29　データの一致度

(2)　自律分散型システム

自律分散整合化技術を用いた自律分散システムの機能信頼性のモデルを図4.30に示す．各サブシステムは機能量として「一致データ」を持っている．

サブシステム中の粒１つが「一致データ」１つを表している．これまでにも述べてきたように,Suica システムでのセンターサーバはバックアップとしてデータを蓄積し，他のサービスや異常時に利用している．

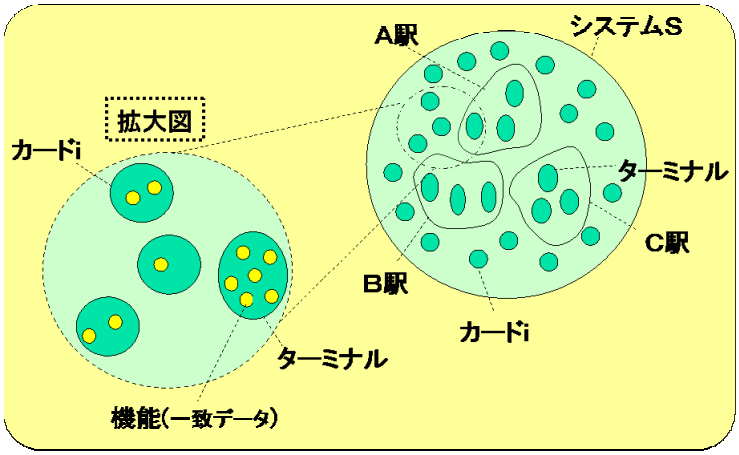


図4.30　自律分散型システムの機能信頼性モデル

(3)　集中管理型システム

図4.31に集中管理型システムの機能信頼性モデルを示す．運賃計算を行うのは各ターミナルであり，複数のターミナルが連携する事によって集中管理型システムにおけるセンターサーバの役割を果たしている．集中管理型システムではターミナルはR/Wの機能しか持たず運賃計算・データの保管はセンターサーバが全て行なう．

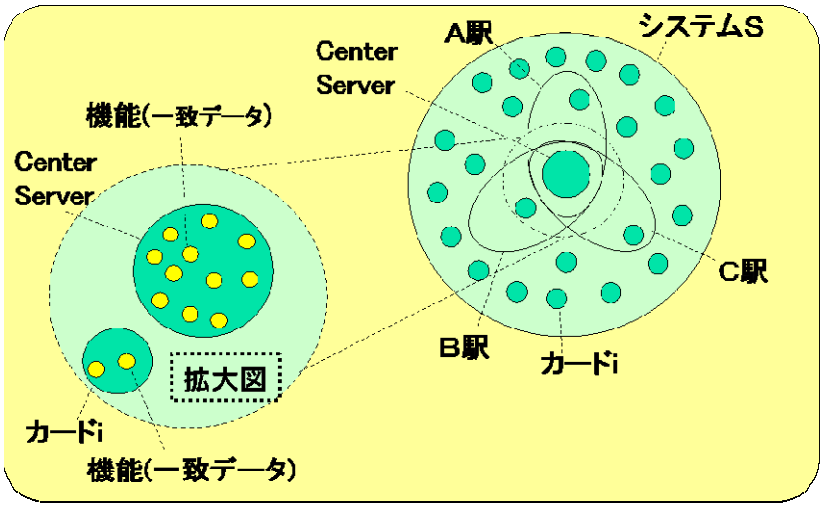


図4.31　集中管理型システムの機能信頼性モデル

4.4.4　機能量の定義

(1)　機能量の定義（基本）

機能量は「3.5.3　機能信頼性評価技術」の定義より，以下のようになる．

状態Ｘの下で，システムＳが達成する機能Ｆ(Ｓ，Ｘ)は各サブシステムが持つ機能$f_i(S_i, X)$の総和であり次式で定義する．

$$\mathbf{F(S,X)} = \sum_{\mathbf{i \in S}} \mathbf{f_i(Si,X)} \tag{3.12}$$

(2)　機能量の定義（ＩＣカード乗車券システム）

前項の2つのシステムモデルにおいて機能信頼性の定義を適用した場合，各数式がどうなるかを示す．まず，「一致データ」である機能量は以下のようになる．自律分散型乗車券システムではカードと各ターミナルが持つデータを比較し，集中管理型システムでは各サブシステムである，カードとセンターサーバのデータを比較する．状態Xにおけるカードiとターミナルjが持つ機能量と機能達成度をそれぞれ$\mathbf{f(S_{Card(i)},X)}$，$\mathbf{f(S_{Terminal(j)},X)}$，$\mathbf{f(S_{Server},X)}$として定義する．

①自律分散型システムの機能量

・データベース一致データ

$$\mathbf{f(S_{Card(i)},X)} = \alpha(\mathbf{i}) \sum_{\mathbf{j \in Terminal}} \mathbf{D_{C(i)T(j)}} \tag{4.8}$$

$$\mathbf{f(S_{Terminal(j)},X)} = \alpha(\mathbf{i}) \sum_{\mathbf{j \in Card}} \mathbf{D_{C(i)T(j)}} \tag{4.9}$$

$\alpha(\mathbf{i})$:サブシステムｉのステータスデータが論理的に正しいかどうかの判定係数

$\mathbf{D_{C(i)T(j)}}$:カードとターミナル間で一致しているデータ数

②集中管理型システムの機能量

・データベース一致データ

$$\mathbf{f(S_{Card(i)},X)} = \alpha(\mathbf{i}) \sum_{\mathbf{j \in Terminal}} \mathbf{D_{C(i)T(j)}} \tag{4.10}$$

$$\mathbf{f(S_{Server},X)} = \alpha(\mathbf{i}) \sum_{\mathbf{j \in Card}} \mathbf{D_{C(i)S}} \tag{4.11}$$

$\alpha(\mathbf{i})$:サブシステムｉのステータスデータが論理的に正しいかどうかの判定係数

$\mathbf{D_{C(i)S}}$:カードとサーバ間で一致しているデータ数

4.4.5　機能達成度の定義

(1)　機能達成度の定義（基本）

機能達成度は「3.5.3　機能信頼性評価技術」の定義より，以下のようになる．

状態Ｘのもとでの「サブシステム Si の機能達成度：ϕ_i(Si，Ｘ)」と呼ぶ．また，状態Ｘの下での「システムＳの機能達成度：Ψ（Ｓ，Ｘ)」と呼ぶ．機能達成度ϕ_i（Si，Ｘ)，Ψ（Ｓ，Ｘ）は次式により定義する．

$$\phi_{\mathbf{i}}(\mathbf{Si},\mathbf{X})=\frac{\mathbf{f_i}(\mathbf{Si},\mathbf{X})}{\mathbf{F}(\mathbf{S},\mathbf{X}_0)} \tag{3.13}$$

$$\Psi(\mathbf{S},\mathbf{X})=\sum_{\mathbf{i}\in\mathbf{S}}\phi_{\mathbf{i}}(\mathbf{Si},\mathbf{X}) \tag{3.14}$$

(2)　機能達成度の定義（ＩＣカード乗車券システム）

ＩＣカード乗車券システムの機能達成度である「データ一致度」の定義式を示す．自律分散型システムにおけるカードi の機能達成度$\psi(\mathbf{S_{Card(i)}},\mathbf{X})$は式3.13と式4.8により式4.12となる．またターミナルj の機能達成度$\psi(\mathbf{S_{Terminal(j)}},\mathbf{X})$は式3.13と式4.9より式4.13と表せる．同様に集中管理型システムにおけるカードi とセンターサーバの機能達成度$\psi(\mathbf{S_{Card(i)}},\mathbf{X})$，$\psi(\mathbf{S_{server}},\mathbf{X})$は式4.14,式4.15となる．

①分散型システムの機能達成度

・データ一致度

$$\psi(\mathbf{S_{Card(i)}},\mathbf{X})=\frac{\alpha(\mathbf{i})\sum_{\mathbf{j}\in\mathbf{Terminal}}\mathbf{D_{C(i)T(j)}}}{\mathbf{D_{ALL}}} \tag{4.12}$$

$$\psi(\mathbf{S_{Terminal(j)}},\mathbf{X})=\frac{\alpha(\mathbf{i})\sum_{\mathbf{j}\in\mathbf{Card}}\mathbf{D_{C(i)T(j)}}}{\mathbf{D_{ALL}}} \tag{4.13}$$

α(i):サブシステムi のステータスデータが論理的に正しいかどうかの判定係数

D_{ALL}:システム内データ数

②集中管理型システムの機能達成度

・データ一致度

$$\psi(\mathbf{S}_{\mathbf{Card(i)}},\mathbf{X})=\frac{\alpha(\mathbf{i})\mathbf{D}_{\mathbf{C(i)S}}}{\mathbf{D}_{\mathbf{ALL}}} \tag{4.14}$$

$$\psi(\mathbf{S}_{\mathbf{server}},\mathbf{X})=\frac{\sum_{\mathbf{j\in Card}}\mathbf{D}_{\mathbf{C(i)S}}}{\mathbf{D}_{\mathbf{ALL}}} \tag{4.15}$$

α(i):サブシステムi のステータスデータが論理的に正しいかどうかの判定係数

D_{ALL}:システム内データ数

4.4.6　　システム全体の機能信頼性

(1)　機能信頼性の定義（基本）

機能信頼性の評価に必要な「機能信頼度」と「平均機能信頼度（アシュアランス度）は「3.5.3　機能信頼性評価技術」の定義より，以下のようになる．

時刻ｔでのシステムＳの機能達成度の期待値を，時刻ｔでの機能信頼度$Ｒ_{t}$と呼ぶ．$Ｒ_{t}$は機能達成度と期待値Ｐ(Ｘ|t)=Prob[Ｘ|t]の積で表し，次式により定義する．なお,期待値Ｐ(Ｘ|t)は時刻ｔ,状態Ｘのもとでシステムｓが達成する機能がＦ(Ｓ,Ｘ)となる確率である．

$$\mathbf{R_t(t)}=\sum_{\mathbf{X}}\Psi(\mathbf{S},\mathbf{X})\cdot\mathbf{P}(\mathbf{X}\mid\mathbf{t}) \tag{3.15}$$

システム稼動の保証度合を評価するアシュアランス度は図 3.34 に示すように機能信頼度の所定期間（ミッション時間）全体にわたる「平均機能信頼度」と定義し，次式で表す．

$$\mathbf{Af}=(1/\mathbf{T_M})\cdot\int_{0}^{\mathbf{TM}}\mathbf{Rt(t)dt} \tag{3.17}$$

ＴＭ　：ミッション時間

Ｒｔ（ｔ）：時刻ｔにおける機能信頼度

(2)　機能信頼性の定義（ＩＣカード乗車券システム）

ＩＣカード乗車券システムの機能信頼性がどうなるか考察する．これまでに述べたように，エラーe_0, e_1が生じた場合は集中管理型システムでも整合化技術を用いた自律分散型システムでも救う事はできず，改札扉を閉めてトランザクションをア

ボートするしかない．そこで今回は，正常に処理が行なわれるか，エラー$e2$ が発生するかのどちらかしか生じないと仮定する．また，各サブシステムの物理的なダウン等「データ抜け」以外の障害は発生しないものとする．さらにモデルをシンプルなものにするため，平均的なユーザを想定しその平均的なカードに基づき各サブシステムの機能量,機能達成度，システム全体の機能信頼性を求めていく．

表4.4　利用例：集中管理型システム

前日ラスト	1回目(乗)	2回目(降)	3回目(乗)	4回目(降)	5回目(乗)	6回目(降)
OK	OK	OK	e_2	OK	e_2	OK

まず，集中管理型システムにおけるシステム全体の機能信頼度を求める．集中管理型システムでは整合化技術を使用しない（データの不一致を許容しない）ため，エラーe_2 が発生した場合，旅客は自動改札機を通過することができず，係員窓口でデータを修正しなければならない．データ不一致が発生した時刻g_n，修正するまでに要する時間をt_s とすると，g_n よりt_S の時間だけカードとサーバのデータが一致していないため，カードの機能量$f(S_{Card(i)},X)$ は0となる.表4.4の例では,3 回目(時刻g_3)と5 回目(時刻g_5) でデータ不一致が起きたため，それぞれ時刻g_3 +t_s, g_5 +t_s にデータ修正が完了するまで$f(S_{Card(i)},X)$= 0 となっている(図4.32)．その影響によってサーバの機能量$f(S_{Server},X)$もその間(n-1) ・M に低下する．

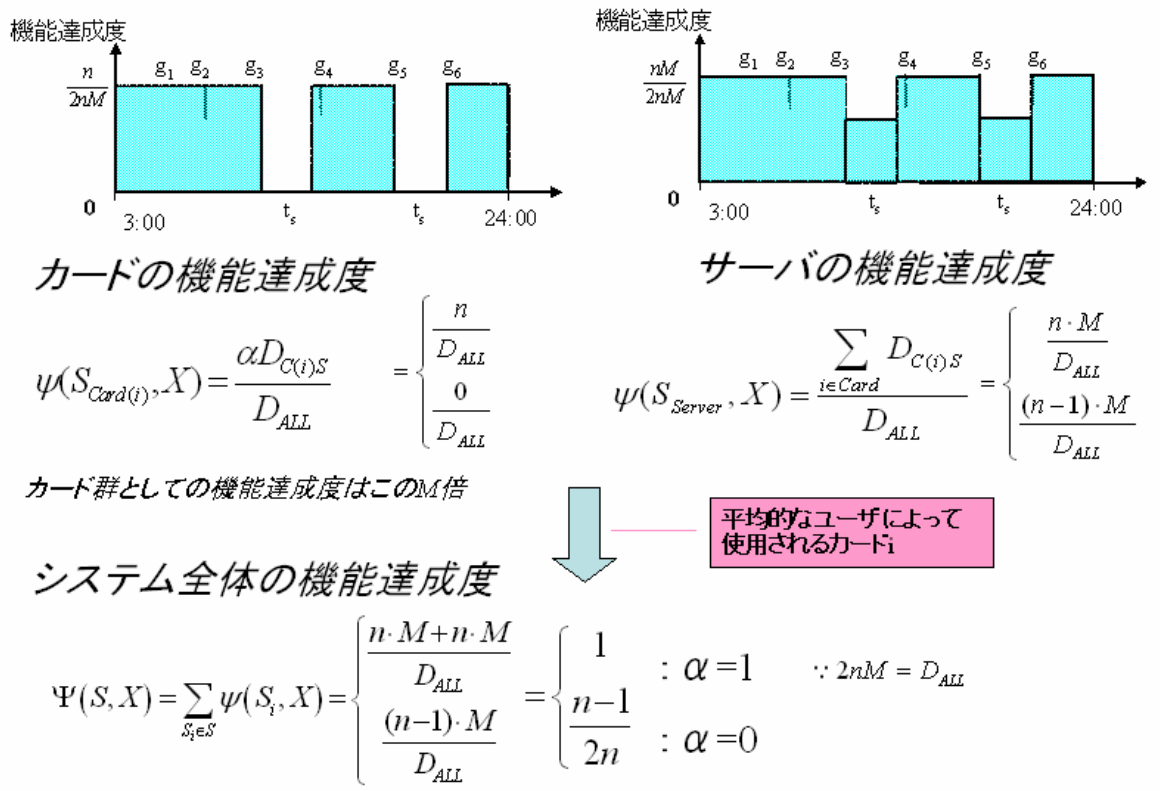


図4.32　集中管理型システムの機能達成度

式4.14，4.15と先ほど求めた機能量により，カードの機能達成度とサーバの達成度は図4.32 のようになる．これによりシステム全体の機能達成度Ψ(*S*, *X*) は1 か(*n*-1)/2*n* となる．機能信頼度はこの機能達成度Ψ(*S*, *X*) と期待値*P*(*X* / *t*) の積であり(図4.33),この時間積分をミッションタイムで割ったものが平均機能信頼度A_f式3.17である．

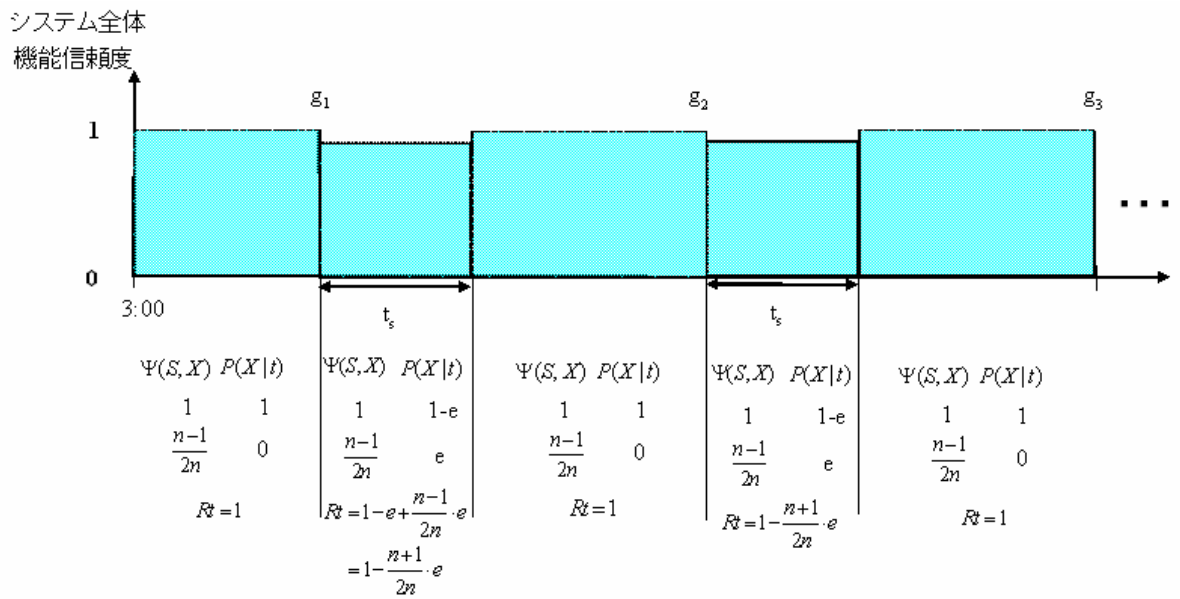


図4.33　集中管理型システムの平均機能信頼度

最終的に集中管理型システムの平均機能信頼度A_fは式3.17より次式となる.

$$\mathbf{A_f} = \frac{\left(1\times\mathbf{T_M}-\mathbf{t_S}\times\mathbf{q}\right)+\left\{\left(1-\frac{\mathbf{n}+1}{2\mathbf{n}}\times\mathbf{e}\right)\times\mathbf{t_S}\times\mathbf{q}\right\}}{\mathbf{T_M}} = \frac{\mathbf{T_M}-\frac{\mathbf{n}+1}{2\mathbf{n}}\times\mathbf{e}\times\mathbf{t_S}\times\mathbf{q}}{\mathbf{T_M}} \qquad (4.16)$$

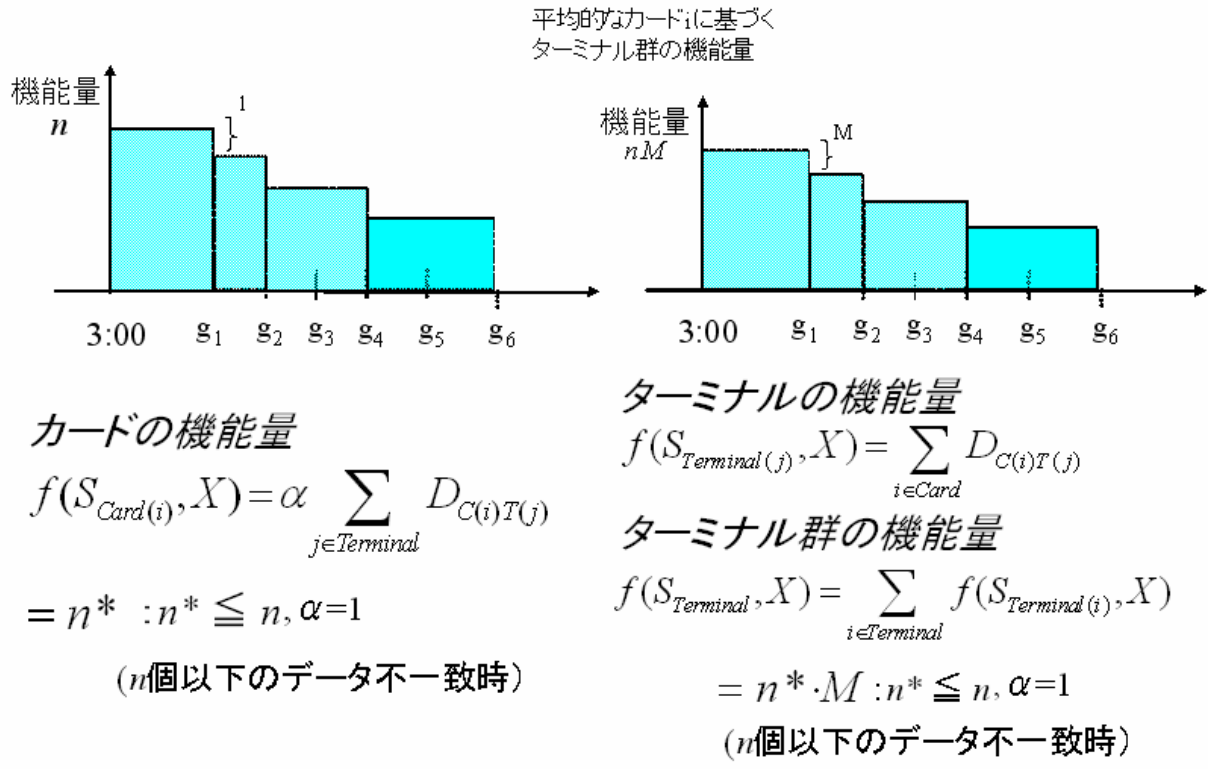
T_M: ミッションタイム，t_s: ダウンタイム，q: タッチ回数

e: エラー発生確率，n: データ件数

続いて，自律分散型システムにおけるシステム全体の機能信頼度を表4.5の例から求める．自律分散型のシステムは整合化技術を用いるため，データ抜けをカードに保存できる件数(*n*) まで許容できる．データが抜けた状態で稼動を継続することができるが，機能量としてはその度に段階的に低下していく(図4.34)．その時の機能達成度は図4.35のようになり，各時刻における機能達成度の期待値は図4.36となる.

表4.5　利用例：自律分散型システム

前日ラスト	1回目(乗)	2回目(降)	3回目(乗)	4回目(降)	5回目(乗)	6回目(降)
OK	e_2	e_2	OK	e_2	OK	OK



4.34　自律分散型システム：カードとターミナルの機能量

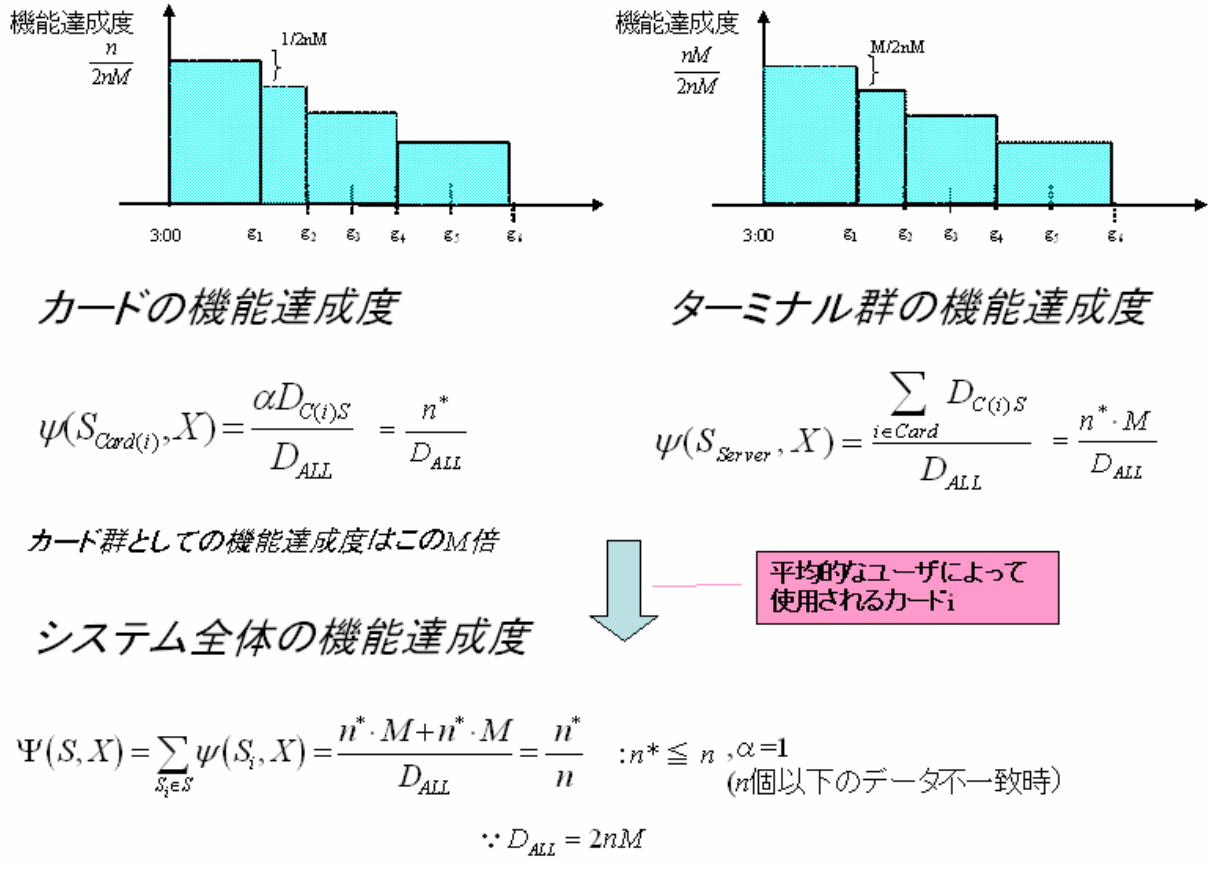


図4.35　自律分散型システム機能達成度

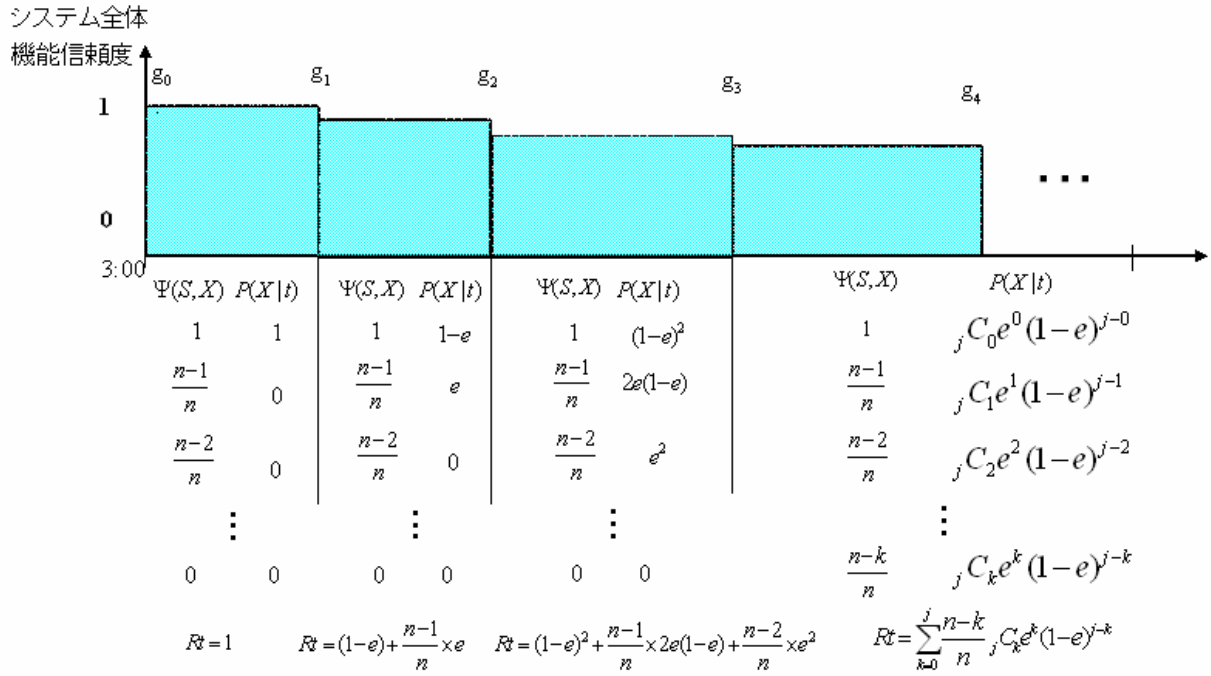


図4.36　自律分散型システムの平均機能信頼度

これらの事から式3.17より平均機能信頼度A_fは式4.16と同様に，

$$A_f = \frac{\sum_{J=0}^{q}(g_{j+1}-g_j)\sum_{k=0}^{j}\frac{n-k}{n}{}_jC_k e^k (1-e)^{j-k}}{T_M} \qquad (4.17)$$

g_i：タッチ時刻，n：カード内データ件数

となる．

4.4.7　　シミュレーションと評価

(1)　シミュレーション

シミュレーションを行うに当たって，平均的なユーザを考える．図 4.37 は時間帯によって改札利用人数がどのように分布しているかを表す，ある駅での実測値である．このデータから支配的な駅利用を行うユーザを想定した（図 4.38）．

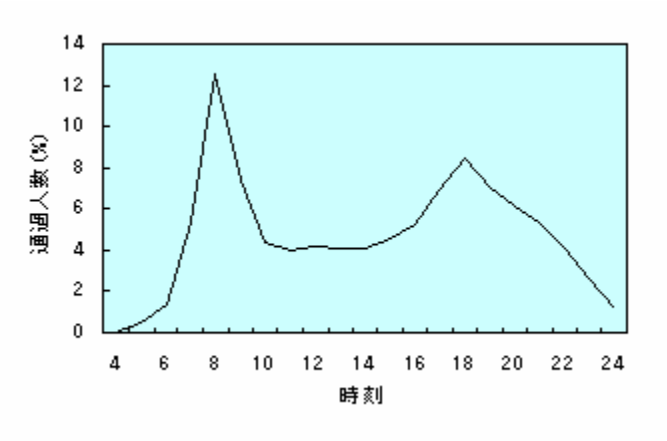


図4.37　各時間帯における駅改札利用人数の分布

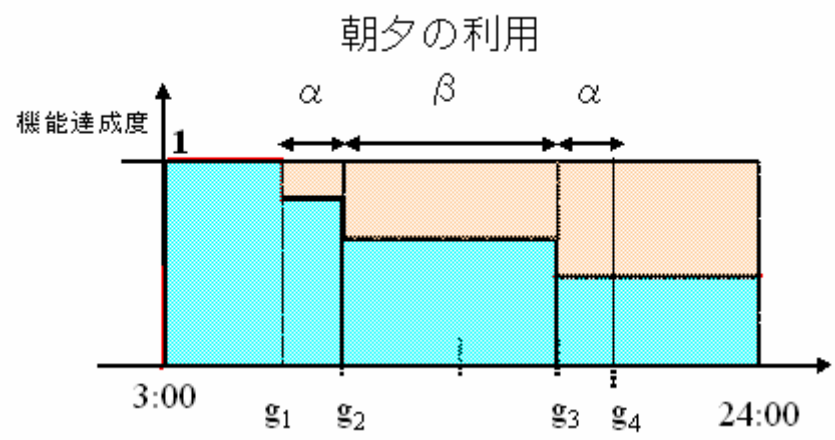


図4.38　シミュレーションに用いるパラメータ：ユーザの利用方法

シミュレーションに用いた核パラメータを表 4.6 にまとめた．今回は，システム構成とエラーが生じた後の処理の仕方によってどのような差が生じるかを明らかにしたいので，エラー発生確率は自律分散型と集中管理型で共に同じ値を用いた．この確率も Suica システムのフィールドにおける実測値を用いた．

整合化処理はバックグラウンドでセンターサーバによって実行される．これまでは，夜中に 1 度行われるだけであったが，トランザクション数の増加に伴い 2006 年 1 月からは日中に 2 回，夜中に 1 回行われるようになった．この整合化処理をいつ行うかは重要な決定要素である．トランザクションのピーク時にデータ抜けが多数生じる可能性が大きいので，その後に整合化処理を行うことが理想的だと考えられる.今回は図 4.39 に示す時間に整合化処理を行うものとしてシミュレーションを行った.

表 4.6　シミュレーションに利用するパラメータ

パラメータ	値
ミッションタイム（T_M）	21［hour］
ダウンタイム（t_S）	10［min.］
タッチ回数（q）	4
エラー発生確率（e）	0.4［%］
乗車時間	30［min.］
1回目タッチ時刻（g_1）	08:00
1回目タッチ時刻（g_2）	08:30
1回目タッチ時刻（g_3）	18:00
1回目タッチ時刻（g_4）	18:30
設計パラメータ	値
整合化処理の回数	1～3
カード内データ保持件数（n）	1～40

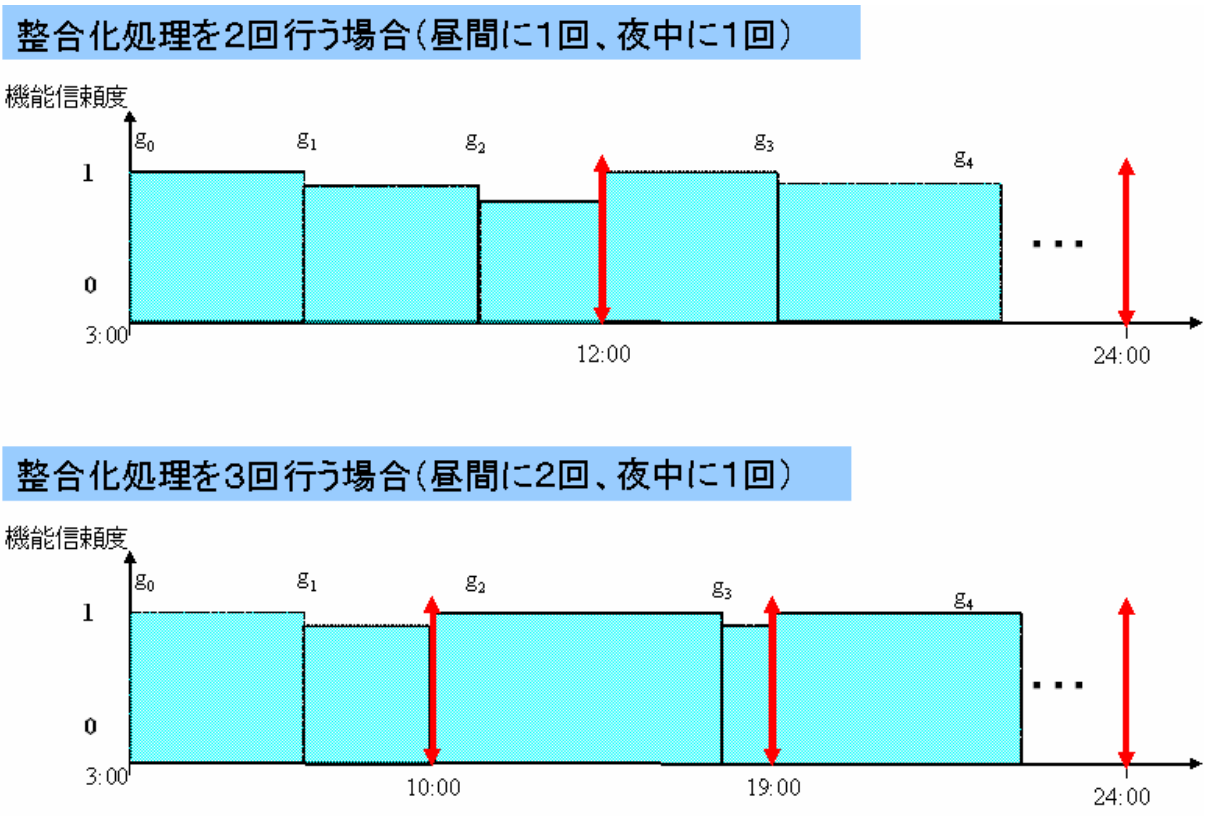


図4.39　整合化処理の回数

(2)　評価

前節までの評価条件に従ってシミュレーションを行った結果を図 4.40 に示す.

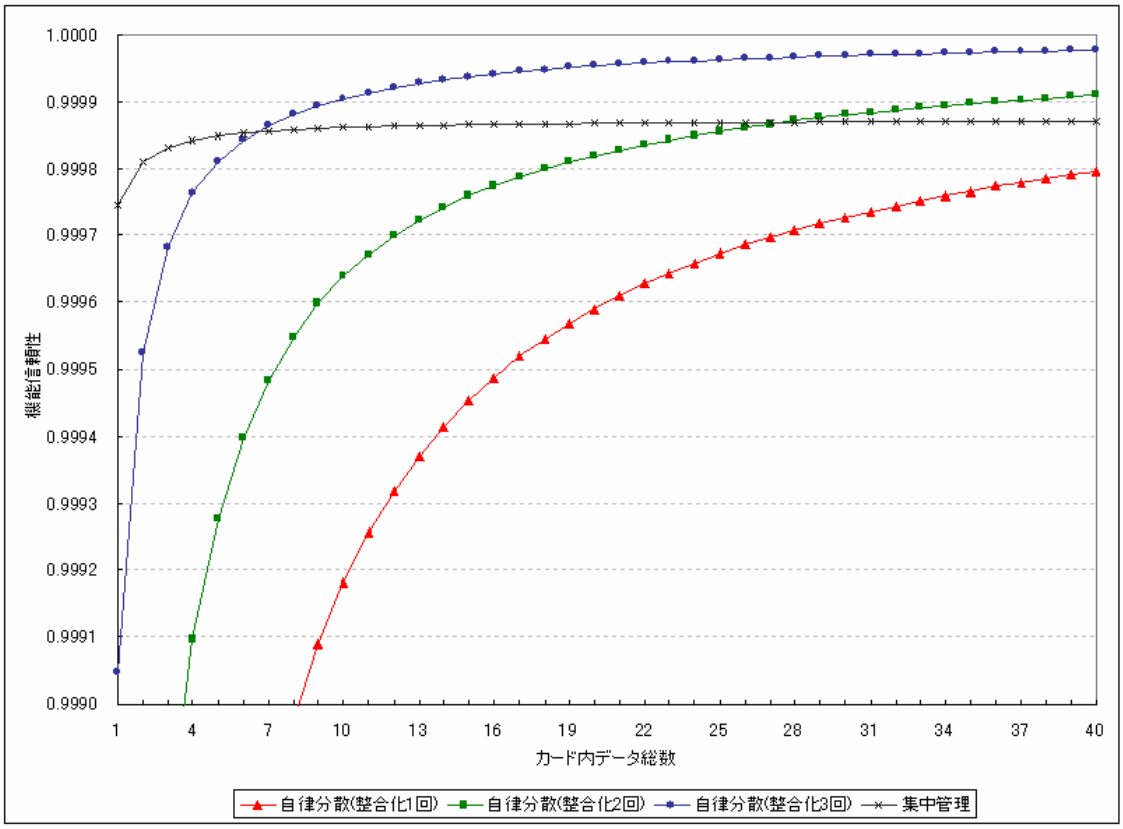


図4.40　シミュレーション結果(データ保持件数)

横軸がカードに保持されるデータ件数，縦軸がシステム全体の機能信頼性である. IC 乗車券システムにおける機能信頼性は「データ一致度」を用いているので，これはデータの信頼性を確保し，システムが安定的に稼動できることの性質を表している．自律分散整合化技術はカードに保存されているデータ件数を許容範囲として，それ以下のデータ不整合を補完することが出来る．カードが多くのデータを持つほど,「データ抜け」をたくさん許容できるというわけである．シミュレーション結果からも，十分なデータ数を持たせれば自律分散型システムが高信頼となることがわかった.ただし,データ保存件数が 20 件を超えた辺りから,その効果は小さくなる. これはエラー発生確率の小ささに対して，必要以上のデータを持たせても効果がないという事である．コストの問題もあり，適切なデータ容量を決める事が重要であり，評価の結果，現在の設計地である 20 件，整合化処理の回数 3 回が適切であることがわかった.

続いて，エラー発生確率を変化させた場合のシミュレーション結果を示す．図 4.41～4.43 は整合化処理を行う自律分散型で，図 4.44 は整合化処理を行わない集中管理型のシミュレーション結果である.

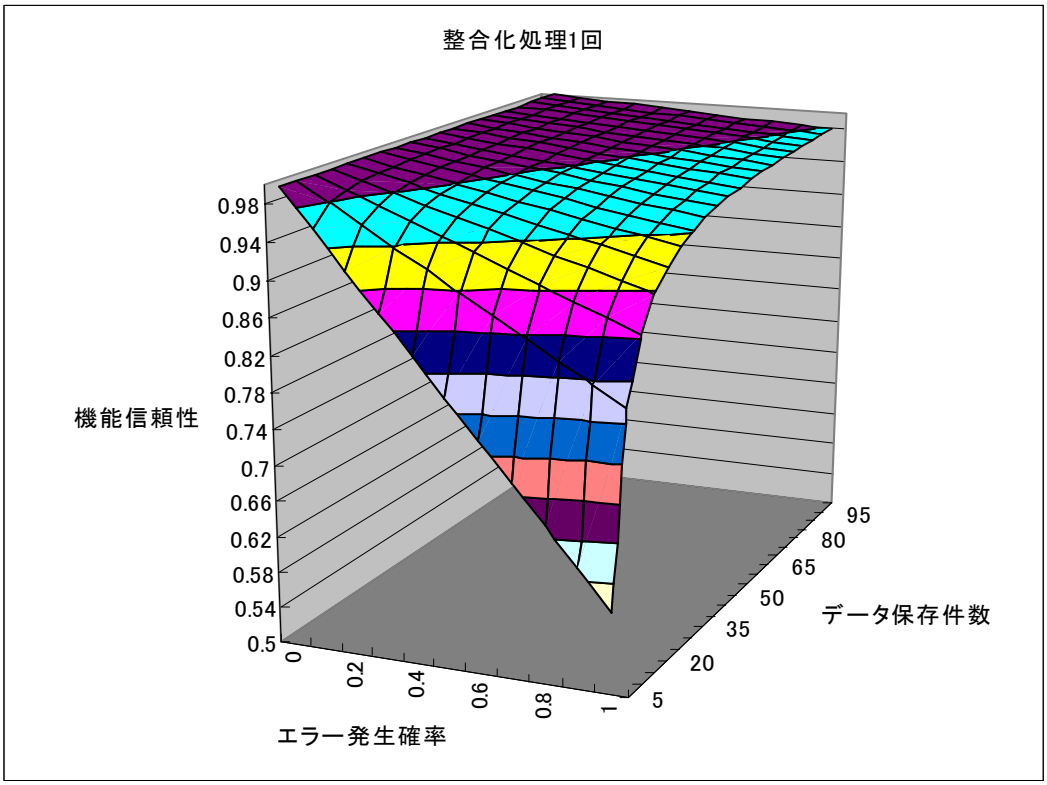


図 4.41　シミュレーション結果(整合化技術 1 回)

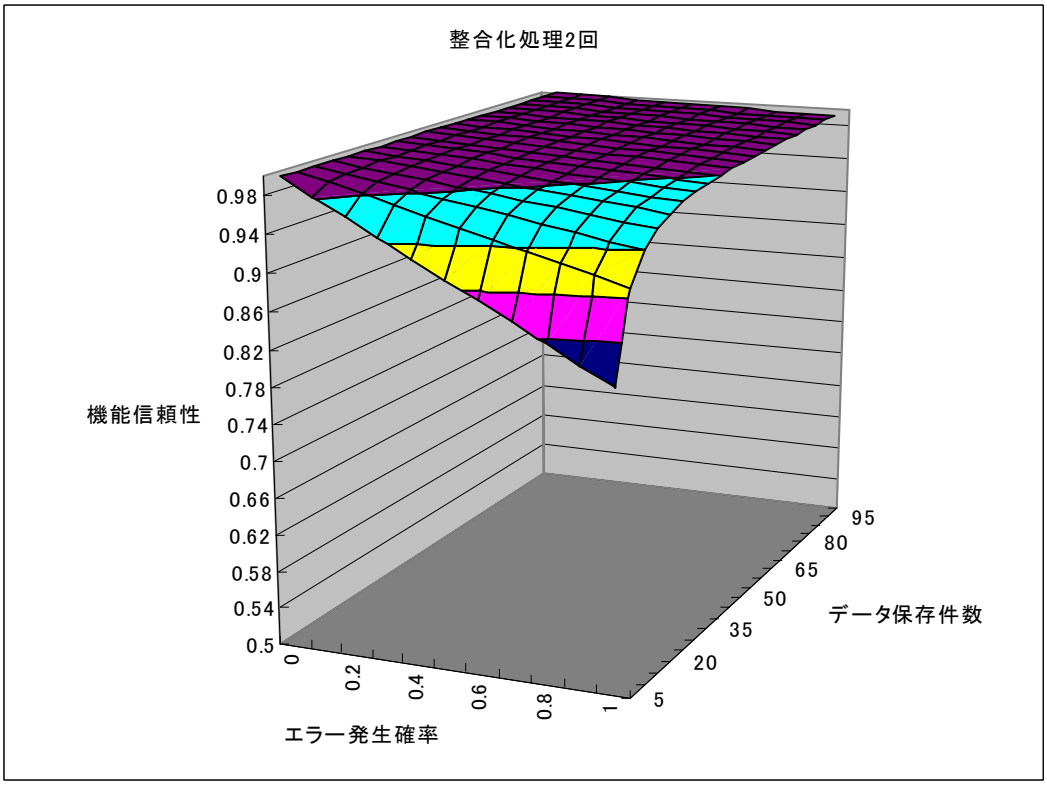


図 4.42　シミュレーション結果(整合化技術 2 回)

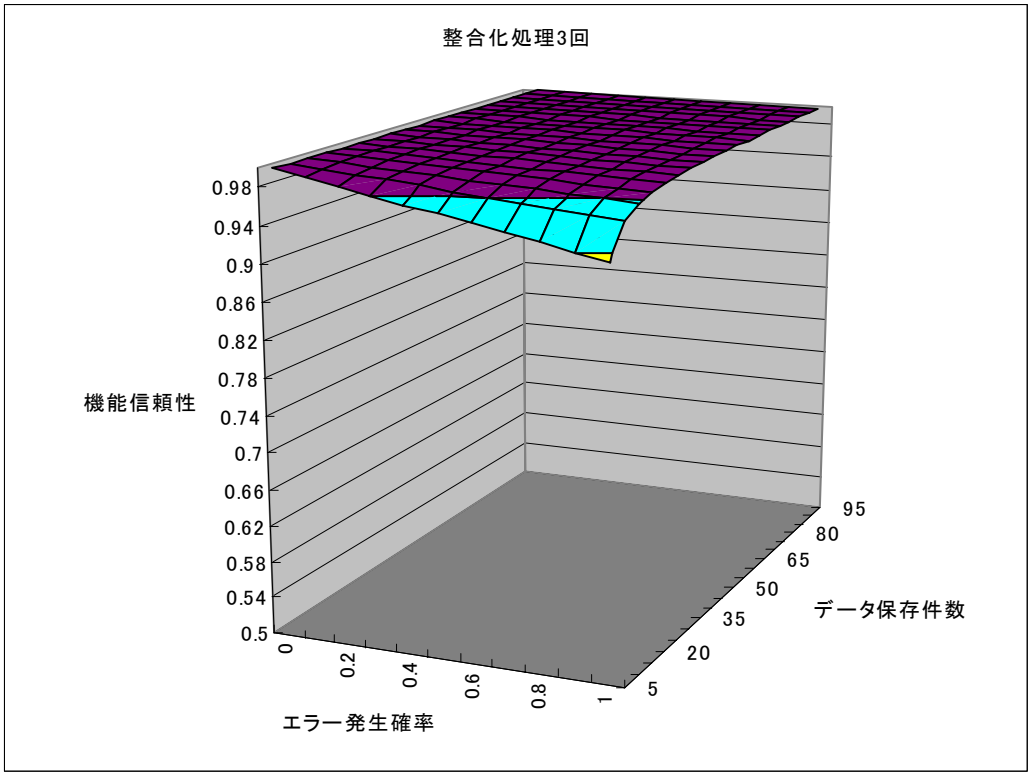


図 4.43　シミュレーション結果(整合化技術 3 回)

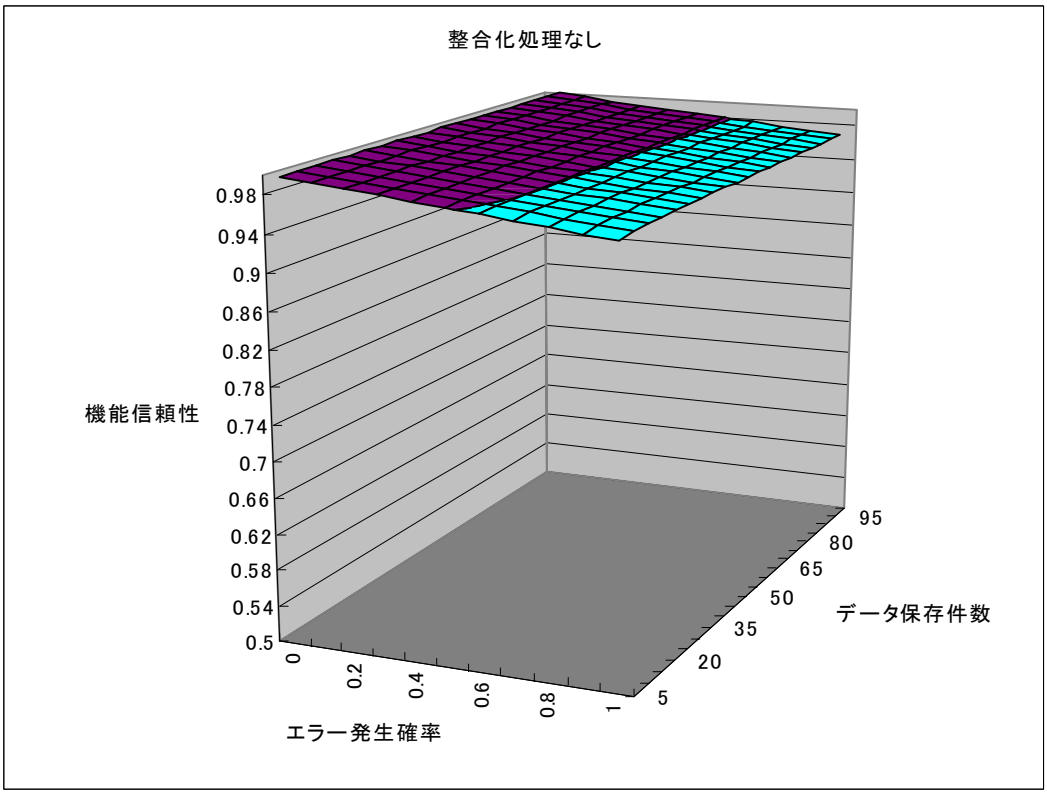


図 4.44　シミュレーション結果(整合化技術なし)

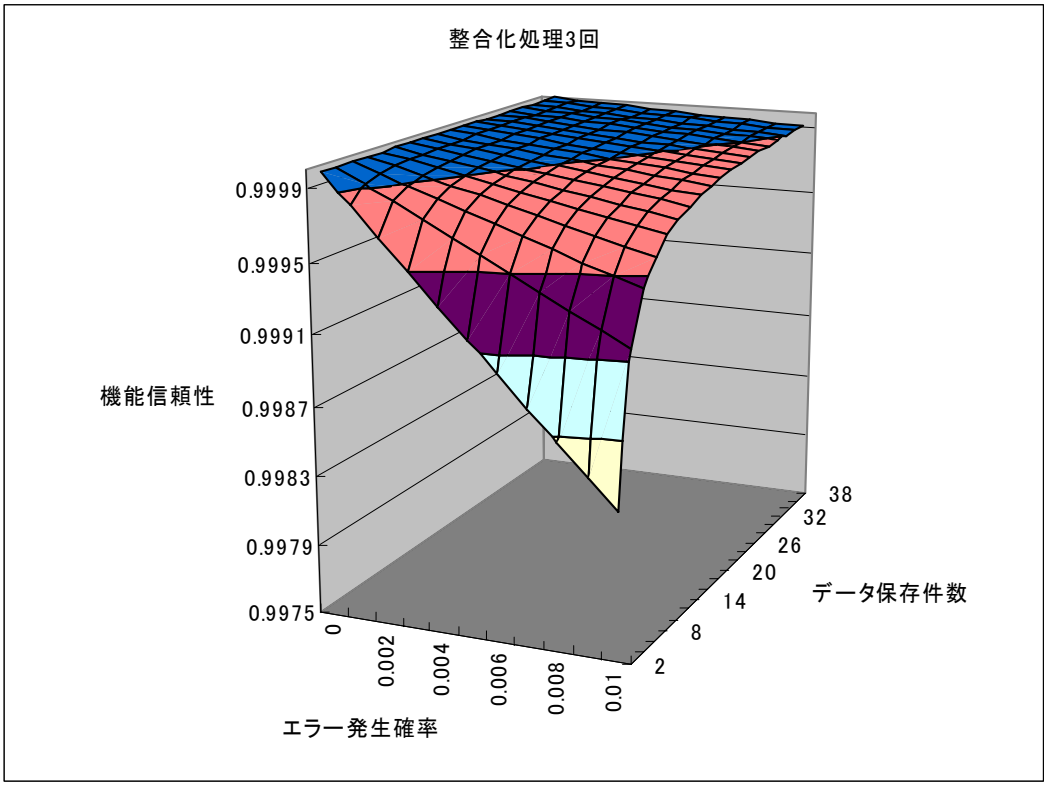


図 4.45　シミュレーション結果(整合化技術 3 回)

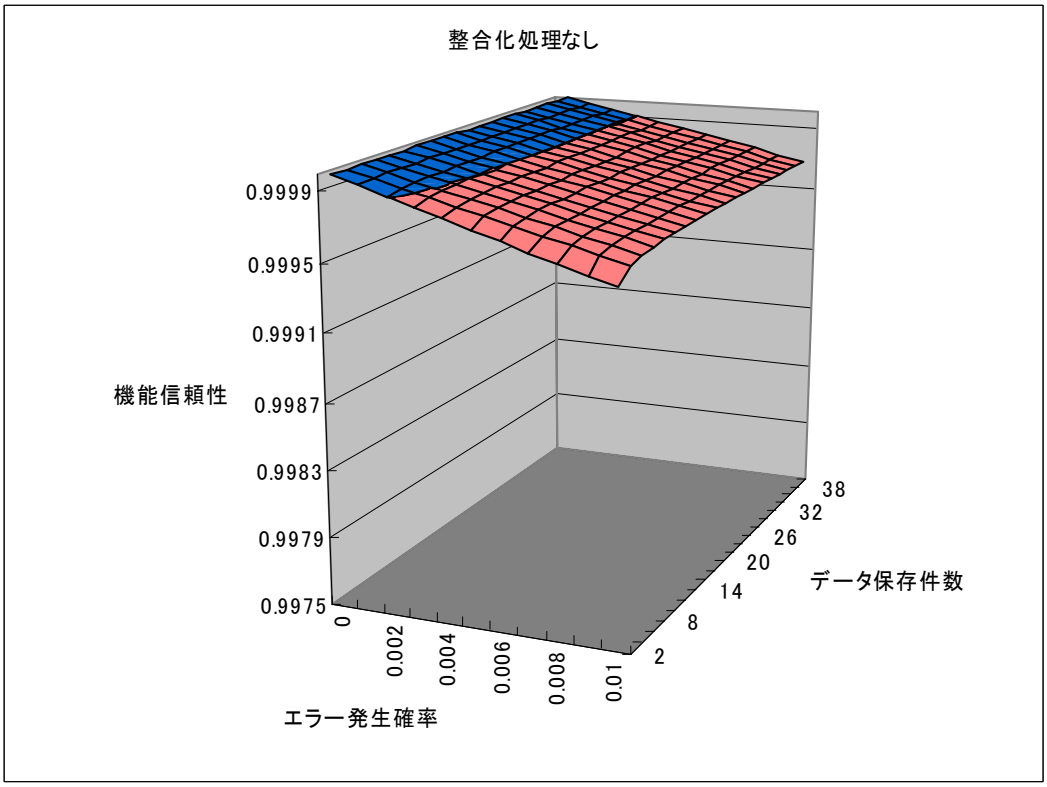


図 4.46　シミュレーション結果(整合化技術なし)

図 4.45 と図 4.46 はエラー発生確率の範囲を 0-1%として実際の数値に近いものを用いてシミュレーションを行った場合である．

一見，整合化技術利用と利用なしでは，利用しない場合の方が機能信頼性が高いように見える．確かに，エラー発生確率が高いにも関わらず，データ容量が少ない場合は低い機能信頼性となってしまうが，そのような場合でもカードに十分なデータを持たせる事によって集中管理型システムと比較してもかなり高い水準を維持することができる(図 4.43，4.44)．

このように，異種統合型ＩＣカード乗車券システムに「機能信頼性評価技術」を適用し，システムのモデル化，機能信頼性の適用，その評価を行う事によって「異種統合型情報サービスシステムアーキテクチャ」と「自律分散整合化技術」の有効性が実証できることを示した．

4.5　　最適システム設計値の算出とその評価手法

これまでに，異種統合型情報サービスシステムの高速処理性と高信頼性を実現し，システムを安定稼動するために，「時間差異種データフィールド」による，異種統合型情報サービスシステムの「システムアーキテクチャ」構築技術を提案した．このアーキテクチャにより，高速処理を実現する技術として「自律分散連携処理技術」，高信頼性を実現する技術として「自律分散整合化技術」を提案し，シミュレーションによりその有効性を評価してきた．

本節では高速性と高信頼性の技術を実際のシステム設計に適用する方法について示す．特に，ＩＣカードとリーダ／ライタ間の処理においては「通信エリアの大きさ」が重要な設計パラメータになる．まず，システムのモデル化を行い，その評価方法を提案し，シミュレーションにより設計パラメータの最適値を求める．

4.5.1　　システムのモデル化

これまでも述べてきたように，非接触ＩＣカードシステムにおいては，システムの高速性と高信頼性を確保し，アプリケーションニーズである，「流動性」と「システムの継続性」を確保する必要がある．

特に重要な部分はＩＣカードとリーダ／ライタとの関係である．ＩＣカードはリーダ／ライタの「通信エリア」内で処理がなされ，信頼性を阻害する要因として「データ抜け」がある．エラー率がその一因であることなどの詳細については４．４節で述べた．また，高速処理においては改札機の処理能力を無限に高速化できないことから，一定の能力値において人が利用する場合，マンマシンインターフェイスの関係からどうしてもエラーが発生する．このように，高速性と高信頼性はエラー率によって変化する．このエラー率は通信エリアの大きさに関係し，通信エリアの大きさは改札機の流動性に関係する．ＩＣカードとリーダ／ライタ間の関係をモデル化したものを図 4.47 に示す．

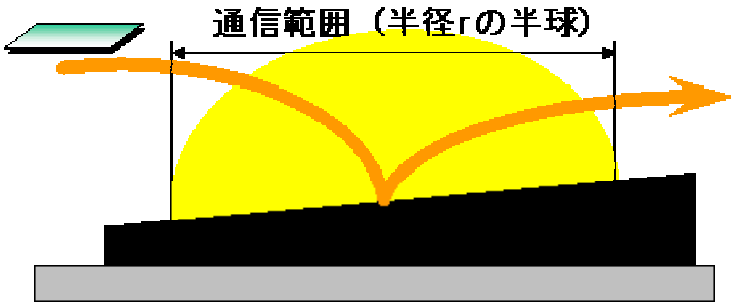


図 4.47　ＩＣカードとリーダ／ライタ間の関係

ＩＣカードを人が手に持って移動する場合，通信エリア内を正常にカードが通過するには，通過時間（エリア内滞在時間）以下で改札機はデータ処理をする必要がある．この関係は次式のようになる．

$$t_p \leq t \qquad (4.18)$$

tp：改札機の処理時間

t：カードの通信エリア滞在時間

改札機の処理能力はハードの性能値によって決まるが，本モデルにおいては改札機の処理時間（処理能力）は「エリア内通過時間（カードがエリア内を通過する時間）」と等しいもの，すなわち，t_p＝t とする．

(1)　エリア内通過時間（実測値）

図 4.47 に示す，自動改札機の通信エリア内をカードが通過する時間（エリア内滞在時間）の実測値を図 4.48 に示す．エリア内通過時間 t で通過した人数の実測値である．この場合，通信エリアの大きさは r ＝0.1m である．

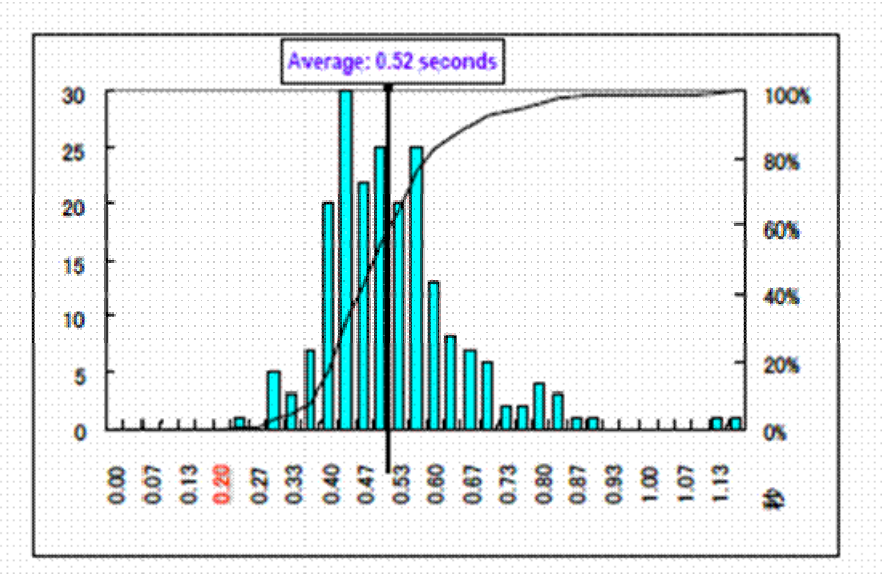


図 4.48　改札機 R/W の通信エリア内のカードの通過時間（実測値）

(2)　改札機 R/W の通信エリア内のカードの通過時間（近似式）

実測値は正規分布に近似可能であり，図 4.49 に示す．

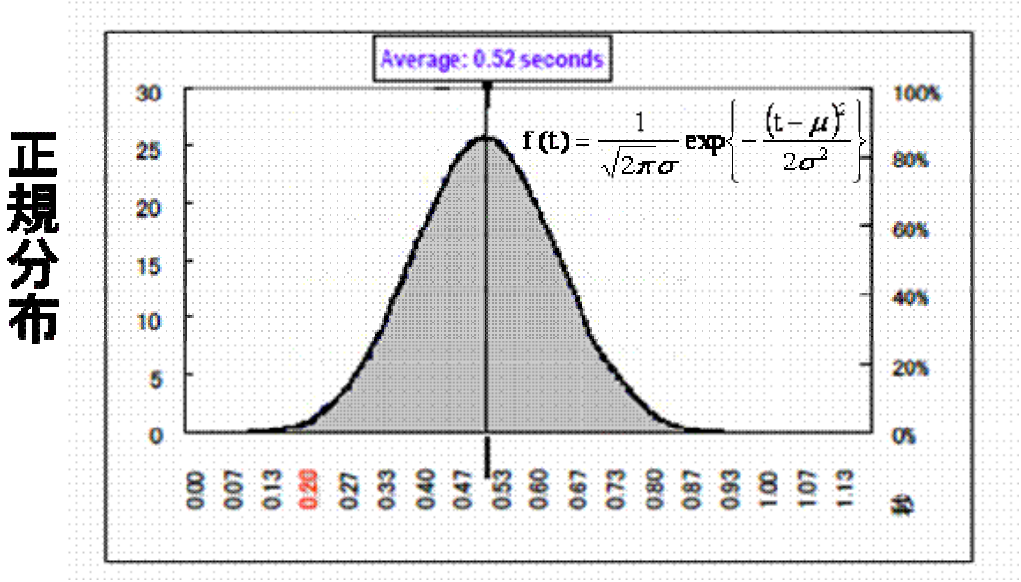


図 4.49　改札機 R/W の通信エリア内のカードの通過時間（近似グラフ）

従って，エリア内通過時間 t の確率は以下の式で表される．

$$\mathbf{f(t)} = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{(t-\mu)^2}{2\sigma^2}\right\} \qquad (4.19)$$

μ :tの平均値　　σ :tの標準偏差

4.5.2　　設計パラメータの評価方法

本項では設計パラメータの評価方法について「サービスの継続性（信頼性）」と「流動性（処理性）」の観点から述べる．４．２節の高速性の評価で述べたエラー率 e はプロセスタイム（処理時間）t の関数であり，t は通信エリア r の関数である．また，４．３節で述べた機能信頼性による評価では機能信頼度 Rt はエラー率 e の関数である．この関係は以下のとおりである．

e = f(t)　　(4.20)

t = f(r)　　(4.21)

Rt = f(e)　　(4.22)

この場合，詳細は後述するが，r により t が変化し，t が小さい（高速：通過人数は大）と e は大きくなる．e が大きいと，Rt は小さく（信頼性が低下）なる．このように，エラー率は通信エリアの関数である．

このため，「サービスの継続性（信頼性）」と「流動性（処理性）」を両立させる最適な通信エリアの設計値を求めるためには，通過人数を最大にし，エラー率を最小にする

通信エリアの大きさを求める必要がある．

(1)　サービス継続性指数

図 4.49 と式 4.19 からエラー率 e とエリア内通過時間 t との関係は以下のとおりとなる．エリア内通過時間が t_p 以下である割合は図 4.50 の面積 A で表せる．

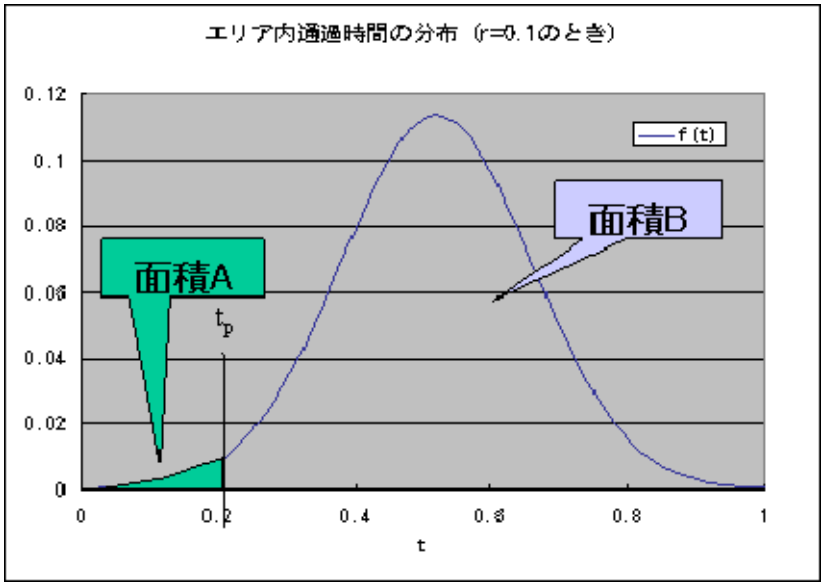


図 4.50　エリア内通過時間の分布と確率

正規分布の面積は式 4.19 を積分し，以下の式 4.23 で表せる．

$$\mathbf{F(t_p)=\int_0^{t_p} f(t)dt} \qquad (4.23)$$

改札機の処理能力が tp であると，面積 A は通過できないため，エラーとなる．従って“tp”で改札機を通過できないエラーの発生する確率を e とすると e は右グラフの面積比で表される．

$$\mathbf{e=A/(A+B)} \qquad (4.24)$$

ここで，エラーの発生しない確率を E とし，「サービス継続指数」と定義する．
E は以下の式で表される．

$$\mathbf{E=1-e=\frac{B}{A+B}=\int_{t_p}^{\infty} f(t)dt} \qquad (4.25)$$

また，これまでは通信エリア r ＝0.1m の実測値から近似式を求めてきたが，この実測データを使って，通信エリアを変化させた場合を図 4.51 に示す．この場合，人の挙動は通信エリアが目には見えないために変わらないとした．

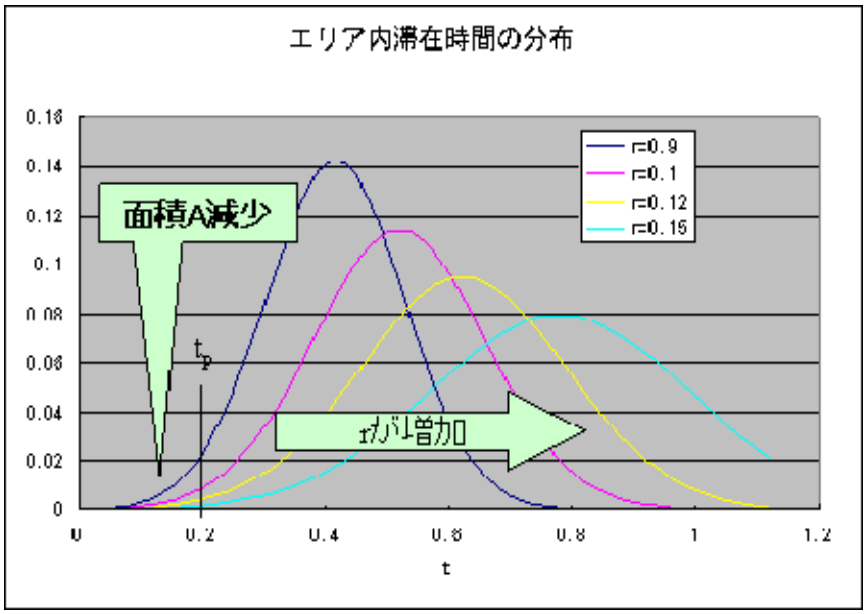


図 4.51　通信エリアの大きさによるエリア内通過時間の分布

図 4.51 からわかるように，r（エリアの半径）が増加（大きくなる）と 面積 A は減，面積 B は増する．このため，e が減少し，E は増加する．

(2)　流動性指数

通信エリアの大きさと改札機の流動性について以下に述べる．これまでにも述べてきたが，高速処理では通過人数が増加し，流動性は良くなる．しかし，信頼性が低下し，この信頼性すなわちエラー率と通信エリアとの関係は前項でのべたとおりである．

図 4.52 に改札機通過時のカードと人の関係を示す．

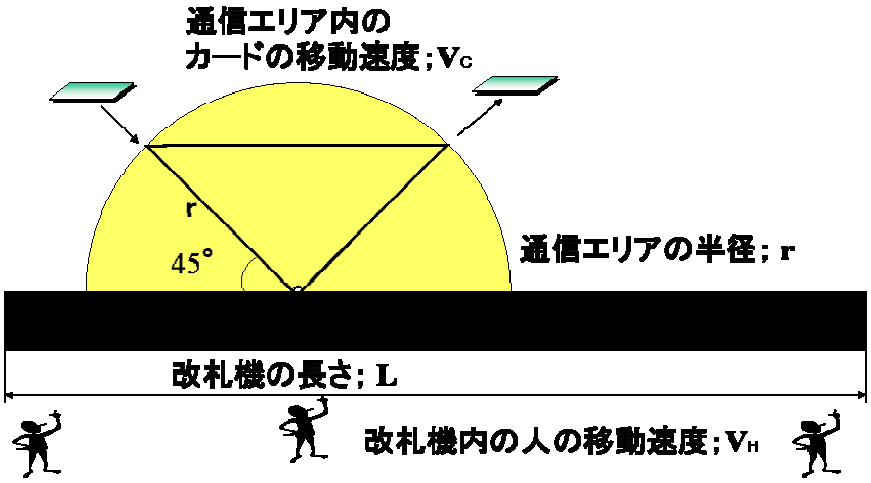


図 4.52　改札機〕通過時のカードと人の状態

① カードの流動性

図 4.52 に示すように，カードは通信エリア内に 45° で突入し，45° で出て行くものする．このとき，エリア内のカードの移動速度を V_C とすると通過時間 tc は以下の式で表される．

$$\mathbf{t} = \frac{2\mathbf{r}}{\mathbf{V_C}} \tag{4.26}$$

t_c が小さいほど，通過数は増加するため，カードの流動性を表す指数 Tc を以下のように定義する．

$$\mathbf{T} = \frac{1}{\overline{\mathbf{t}}} = \frac{\overline{\mathbf{V_C}}}{2\mathbf{r}} \tag{4.27}$$

② 人の流動性

図 4.52 に示すように，改札機内を人が歩行して移動する場合はその通過時間は以下の式で表される．

$$\mathbf{t_G} = \frac{\mathbf{L}}{\mathbf{V_G}} \tag{4.28}$$

改札機内の人の移動速度；$\mathbf{V_G}$

改札機の長さ； $\mathbf{L}$

t_G が小さいほど通過人数は増加するため，人の流動性を表す指数 T_G を以下のように定義する．ただし，R/W での処理が終了すると改札機内には複数の人が同時に滞在可能であるため，処理が改札機内に同時に滞在できる人数; $\mathbf{n}$ とすると以下の式となる．

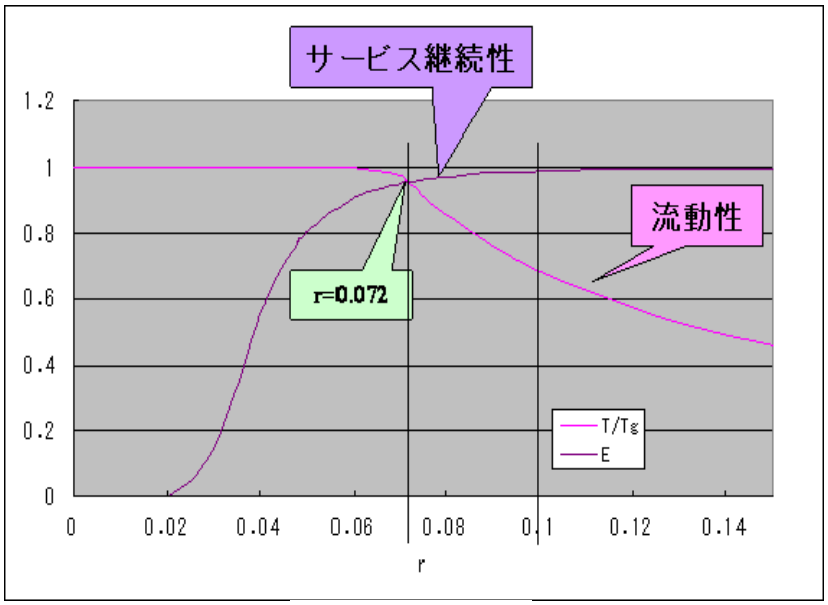
$$\mathbf{T_G} = \frac{1}{\overline{\mathbf{t_G}}} \times \mathbf{n} = \frac{\overline{\mathbf{V_G}}}{\mathbf{L}} \times \mathbf{n} \tag{4.29}$$

改札機の流動性を評価するため，以上の①カードの流動性と②人の流動性を合わせて，以下のように「流動性指数 F」を定義する．

$$\mathbf{F} = \frac{\mathbf{T}}{\mathbf{T_G}} = \frac{\dfrac{\overline{\mathbf{V_C}}}{2\mathbf{r}}}{\dfrac{\overline{\mathbf{V_G}}}{\mathbf{L}} \times \mathbf{n}} = \frac{\mathbf{L}\overline{\mathbf{V_C}}}{2\mathbf{nr}\overline{\mathbf{V_G}}} \tag{4.30}$$

4.5.3　　シミュレーションによる最適値の決定

前項で定義した，「サービス継続性指数 E」と「流動性指数 F」によるシミュレーション結果を図 4.53 に示す．



但し，
L=1.0
n=2
V_c=0.38
V_G=1.4
μ=0.52
σ=0.14

図 4.53　シミュレーションの結果

図 4.53 からもわかるように，サービス継続指数は通信エリアの増加に伴って増加するが，ある時点から飽和していく．流動性指数は通信エリアの拡大に伴って，急激に減少する．

シミュレーションの結果から，通信エリアの設計最適値は 0.072m である．現在，実際に導入している R/W の通信エリアは約 0.09m であり，本シミュレーションでは，やや信頼性が優位にあり，流動性が低下しているが，ほぼ現実と一致している．このため，本シミュレーションの有効性は実証された．

第5章

結　論

5.1　　本論文の研究対象と成果

5.1.1　　研究のアプローチ

近年，電子マネーや鉄道乗車券システムなどに見られる非接触ＩＣカードを使ったシステムは，高負荷トランザクションを処理する情報サービスシステムと設備機器を含む制御システムとの異種統合型情報サービスシステムである．このような異種統合型システムにおいてはシステムの障害，拡張，保守などのシステム状況変動時にも，高速性，高信頼性とリアルタイム性を両立させ，サービスの継続を保証するアシュアランス性が求められている．

このため，本論文では，これまでのような，情報システムの視点ではなく，制御システムの視点から情報サービスシステムを考察し，高速性と高信頼性の両立とシステムの安定稼動を実現するため，時間軸の異なる「時間差異種データフィールド」を提案し，この技術による「異種統合型情報サービスシステムアーキテクチャ」の構築とアシュアランス性保証のための自律分散技術とその評価方法が必要であることを示し，このシステムアーキテクチャ構築技術とそれを基にして，高速性と高信頼性を実現するための 2 つの自律分散アシュアランス技術とアシュアランス性の評価技術について提案した．さらに，実システムに適用し，その有効性を実証した．

本論文の２章では異種統合型情報サービスシステムに必要な分散技術の動向について，３章ではシステムアーキテクチャ，アシュアランス性を実現する技術について，４章では鉄道乗車券システムへ適用したときの，システム構築技術と２つのアシュアランス技術，アシュアランス性評価技術について，それぞれ考察，検証を行った．以下では，各アプローチにおける成果の概要についてまとめる．

5.1.2　　異種統合型情報サービスシステムアーキテクチャ構築技術の確立

異種統合型情報サービスシステムにおいては異種ニーズとして高速性と高信頼性があり，これらを実現させるためには制御システムと情報システムという異種の

システムの共存技術が必要である．このために，従来システムでは異種ニーズ／異種モード共存下でシステムのアシュアランス性を満たすことが困難であることを述べた．これらの課題を解決するため，異種統合の視点に立ったシステムアーキテクチャが必要であることを示し，異種統合型情報サービスシステムアーキテクチャを提案した．このシステムアーキテクチャでは，異種データの特性に応じてデータフィールドを構成し，それらが目的に応じてゲートウェイを介してデータを交換できるようにしてある．ここで各サブシステムに自律性を持たせ，それらがデータフィールドを介した連携により，システムの部分的障害，拡張，保守においても，稼動の継続性が保証可能にした．

このように，時間差により処理を行う「時間差異種データフィールド」技術を提案し，異種統合型情報サービスシステムのシステムアーキテクチャ構築技術を確立した．

5.1.3　高速性と高信頼性のためのアシュアランス技術の確立

前項のシステムアーキテクチャを基に高速性と高信頼性を確保するアシュアランス技術について以下の 2 つの技術を提案した．さらにそのようなアーキテクチャと技術のもとで，アシュアランス性を評価する技術を提案し，その有効性を明らかにした．

(1)　自律分散高速処理技術の確立

異種統合型情報サービスシステムにおいては「制御システム」リアルタイム処理の条件下で「処理の高速性」と「サービスの継続性」が求められているが，サブシステムレベルでは情報（データや処理法）が不完全だったり，情報が局所的だったりするという課題がある．この課題を解決し，「高負荷トランザクション処理の高速化」という目的を達成するため，異種統合型情報サービスシステムのアーキテクチャを使った高速処理技術として「自律連携処理技術（自律分散アルゴリズム)」を提案した．

「自律連携処理技術」では，各サブシステムが持つ情報の局所的条件下で，サブシステムごとの処理の分散化／サブシステム間の連携と処理時間との関係を導出した．この技術により，高速処理を達成するための最適なシステム分割／連携度合をトランザクションの発生特性に応じ求められることを示し，その技術を確立した．

(2)　自律分散高信頼性技術の確立

高負荷トランザクションの情報サービスシステムにおいては，高速処理と高信頼性の確保に加えて，サービスの継続を保証するアシュアランス性も求められている．サービス継続性の観点から，障害が発生してもシステムを停止することなく稼動を継続させることが必要であり，このため，サービスの継続と高信頼性を確保する技術として「自律分散整合化技術」を提案した．

この技術は，リアルタイム処理制限下での高トランザクション処理において生じるデータの欠損を回復するため，データフィールドごとにデータの滞留時間を変化させた「時間差異種データフィールド」を構成し，それらの連携によるデータ間の論理的整合化を図る技術である．この技術により，トランザクションの発生特性に応じて整合化達成時間を最小にする，異種データフィールド数と各データフィールド間の時間差を導出可能とし，その技術を確立した．

(3)　アシュアランス性評価技術の確立

異種統合型情報サービスシステムの異種モード共存環境下におけるシステム稼動の保証度合の評価において，「機能量×稼動時間」最大化を実現するための評価基準について，明らかにした．異種のモードが共存し，しかもそれらが時間と共に変化する環境下で，システムの稼動を保証しうる性質であるアシュアランス性に着目し，その観点から，システム全体を評価するための評価指標として，「機能信頼性評価技術」を提案した．

システムのアシュアランス性は「機能量×稼動時間」の最大化を保証する事であるため，その評価指標のアシュアランス度を以下のように定義した．

システム稼動の保証度合のアシュアランス度Ａｆ

＝Σ Σ（機能×信頼度×稼動時間）／(全体機能×所定期間)

＝Σ Σ（機能達成度×信頼度×稼動時間）／所定期間

ここで機能達成度＝機能／全体機能である．

さらに，「機能達成度×信頼度」を「機能信頼度」と定義し，システム稼動の保証度合のアシュアランス度Ａｆを機能信頼度の所定期間（ミッション時間）全体にわたる「平均機能信頼度」と定義した．

＝Σ Σ（機能信頼度×稼動時間）／所定期間

5.1.4　鉄道乗車券システムへの適用によるアシュアランス技術の確立

鉄道乗車券システムは朝夕のラッシュ時に高負荷トランザクションとなり，このときに高速な処理が求められる．また，運賃というお金にかかわる情報を処理するため，信頼性の確保も必須事項である．最近ではさらに安定稼動が求められており，流動性の確保はもとより，サービスの継続性についても重要になっている．

このような，鉄道乗車券システムは，無線 IC カード／有線ネットワーク／ゲート装置など複数の手段を含む制御／トランザクション処理の統合された「異種統合型情報サービスシステム」である．乗客の流動性を妨げることなく，処理の高速性，高信頼性を実現するため，この鉄道乗車券システムに対し提案したシステムアーキテクチャと 2 つのアシュアランス技術を適用し，それらの有効性を実証した．

また，実システムの設計において機能信頼性評価技術が有効に活用されることも明確にした．さらに，システムの規模拡大とトランザクションの急激な増大に対しても，アシュアランス性が満たされていることを実証し，鉄道乗車券システムへの適用によるアシュアラン技術を確立した．

(1)　自律連携技術の確立

鉄道乗車券システムの基本機能は，不正カードのチェックと運賃計算処理である．この処理は通常は鉄道の降車する駅の端末で行われるため，降車駅の端末の処理負荷は大きい．このため，端末レベルでの高速処理を可能とする「自律連携技術」を提案し，その高速処理技術の有効性を評価した．自律連携処理を使った分散システムと自律連携処理を使わないシステムとして，集中システムの 2 つのシステムモデルを提案した．システムの有効性とその条件を明らかにし，技術の評価するため，この 2 つのモデルについてシミュレーションを行った．

その結果，この技術により，一定の条件下で，最適なシステムの選定と，トランザクション数（42700 件／日）や駅数（43 駅以上）などが，トランザクションの発生特性に応じて求められることを示し，この技術の有効性を証明した．

(2)　自律分散整合化技術の確立

鉄道乗車券システムにおける，無線通信方式のIC カードと自動改札機リーダ／ライタとの処理は，ユーザーの使用方法に依存するため，リアルタイム処理制限下で高トランザクション処理による， データの欠損，いわゆる「データ抜け」という重大な問題がどうしても発生してしまう．このため，「データ抜け」が発生してもシステムを止める事無く， データの信頼性を確保するため「自律分散整合化技術」を

鉄道乗車券システムに導入した．従来技術の集中システムではデータが損失した状態での稼動を許す事ができず，システムを停止して，障害に対応していた．しかし，「自律分散整合化技術」では一定の条件下でシステムの稼動を許容するため，従来の評価法では両者の正当な比較・評価ができないといった問題があった．

この問題に対応するため，機能信頼性評価法を用いて，整合化処理を行う自律分散型システムと，従来の整合化処理を行わない集中管理型システムを比較した．まず，システムアーキテクチャ，障害の起きるパターン，その障害がシステムにどのような影響を与えるかという事を調査し，それに従いシステムのモデル化を行った．また，IC 乗車券システムの基本機能は，不正カードのチェックと運賃計算処理であるので機能信頼性における，最小単位である機能量を「データの一致」と定義した．さらに，旅客の利用方法とエラー確率は実測値を用い，カードに保存できるデータ件数，整合化技術の回数を設計パラメータとしてシミュレーションを行った．

また，エラーの発生確率を変化させた場合のシミュレーションも行い，そのことからエラー発生確率に対応して，カードに適切なデータ容量を持たせることが重要であることを明らかにした．シミュレーションによる評価の結果，現在の実測値であるエラー発生確率においては現状の設計値であるカードの保存件数20 件，整合化処理3 回が適切である事を証明した．

このように，システムのモデル化，機能信頼性評価技術の適用，評価を行うことによって「自律分散整合化技術」の有効性を実証した．

5.1.5　最適な設計パラメータ評価手法の確立

改札機のリーダ／ライタの通信エリア（半球状の半径）は高速処理性と高信頼性にかかわる重要な設計パラメータであることを明らかにし，その最適な設計値を求める方法を提案した．

「サービスの継続性」という観点から，特に信頼性に関する「エリア内通過時間」の実測値から近似式を求めて，エラー率と通信エリアとの関係を明らかにし，サービス継続指数」として定義した．一方で「流動性」の観点から，カードと人の改札機の通過度合いとして，「流動性指数」を定義し，通過量と通信エリアの関係を明らかにした．

この 2 つの関係は流動性が良くなると信頼性が低下し，サービス継続性が低下する．この 2 つの関係を実ステムでシミュレーションしその結果，通信エリアの最適値が 0.07m であることがわかった．これは実際のシステムとほぼ一致しており，この手法の有効性が実証された．

5.2　今後の課題

本論文では，異種統合型情報サービスシステムの高速性と高信頼性を実現するためのニーズと課題を明確化し，これを実現する技術として「異種統合型情報サービスシステムアーキテクチャ」構築技術と２つの自律分散技術，アシュアランス評価技術を提案した．その結果を鉄道乗車券システムへ適用しアシュアランス性の面から評価比較してその有効性を確認し，システムの方式を決定することが出来た．さらに，最適なシステム設計を行うための手法について提案し，最適値を求め，その有効性を確認した．

今後，本論文で提案したアシュアランス技術とアシュアランス性評価技術についての課題としては以下のように考える．

鉄道乗車券システムについて，モデルを用いてアシュアランス性の評価を理論的に求めてきたが，実際のシステムはもっと複雑で多岐にわたっており，「データの一致度」以外にも機能は多くある．今後は，現在の「鉄道乗車券システム（ＩＣカード Suica 乗車券システム)」での実績をもとに各アシュアランス技術，評価手法の検証をさらに深度化したいと考える．

また，高速処理では一定の条件（高トランザクションや駅数が多い）では分散処理がよいとの結論を得た．これは高トランザクションや駅数の多い中規模以上の鉄道会社へは本論文で提案した技術の導入が適していることとなる．また，高信頼性の面でも整合化技術を導入した分散システムが良いとの結論を得ている．鉄道のシステムはトータルとして機能する必要があり，分散システムの導入がトータルとして優位性があることを示唆している．今後，多くの鉄道・バス各社が IC カード乗車券システムの導入を検討している．導入にあたっては，短期間に多くの投資が必要となるため，中小規模の鉄道･バス会社などにおいては年間の投資額の制約から，一括してすべての機能を持ったシステムの導入は出来ないところもある．そこで，本論文で提案した，異種統合型情報サービスシステムの特徴である，異種システム共存可能技術により，サービスを継続しながら，長期間にわたってシステムを構築する事が可能となる．具体的には，まず，IC カード乗車券ステムの基本機能を導入し，その後，順次その時々に合わせたサービス機能を付加していくことも可能であるので，年間の投資額を抑えることが出来る．さらに言えば，このような機能の拡張がサービスを停止することなく継続して稼動した状態で新サービスと旧サービスの切り替えや機能増強が可能で在ると考えられる．サービスという視点だけでなく，投資計画というマネージメントビューという視点から見たときには，また違

った構築技術が必要になる場合も考えられる．今後は，このような視点からの検討も必要になると思われる．

現在，「ＩＣカード Suica 乗車券システム」は導入当初の完結した自社インフラから各種の施策により，他社システムとの結合が進み，オープンなインフラへと変化している．他社とのシステム結合は今後益々多くなるものと思われる．また，最近の銀行や証券取引所などのシステム障害に見られるように高負荷トランザクションの情報サービスシステムにおける，高速処理性と高信頼性を確保することは，今後システム間の結合が進んでくると，企業経営の面からも非常に重要な位置を占めてくるといえる．

本論文の「異種統合型情報サービスシステムアーキテクチャ」構築技術や高速性と高信頼性を両立させる 2 つのアシュアランス技術は他の企業システム（E-Commerce 等）への本技術の適用が有効であると考える．他企業との間で，システムニーズは同じであっても，システムの持っている目的がそれぞれの企業の経営方針の違いから，ニーズの質と量は異なっていると考えられる．そのような異種システムを共存させながらシステムを統合･結合する技術，つまりサービスを提供しながらシステム拡張が行えるようになれば，利用者にとってもメリットがあるばかりでなく，企業経営にとっても，より柔軟な企業戦略とシステムの品質が確保できるので大きなメリットを持つことが出来る．

以上のように，今後，交通業界においては，本研究で提案したアシュアランス技術と評価手法を用い，より具体的，定量的な検討を行い実用に供していきたいと考えている．さらに，鉄道乗車券システム以外においても，今後ますます重要性の高まる，高トランザクションの「異種統合型情報サービスシステム」に対するアシュアランス性を保証する技術として，高速処理性，高信頼性の必要なシステムに本研究の成果を導入したいと考えている．

謝　辞

本研究を行うに当たり，終始適切な御指導，御鞭撻をいただいた森欣司教授に心より感謝いたします．社会人博士課程への進学を助言してくださったばかりでなく，論文の投稿，国際学会での発表などを懇切丁寧にご指導いただき，社会人としてプロジェクト業務の遂行に専念していた小生にとっては大変貴重な経験をさせていただきました．産学連携という視点から，大学での学術的研究と企業での実用的研究・開発とがそれぞれ持つ長所と短所とを体感できました．

本論文をまとめるに当たり，社会人の陥りがちな点を技術的，論理的な観点から，多くのご指摘と有益なご助言をいただいた，藤原英二教授，酒井善則教授，荒木純道教授，横田治夫教授に心よりお礼を申し上げます．

また，本研究に適切な指導と助言をいただいた助手の呂暁東さんをはじめ，森研究室のメンバーの皆様には多くのお手伝いをしていただきました．特に大橋克弘君（現佐藤研究室）とは多くの活発な議論をし，その結果としてのデータ解析などに尽力いただき大変感謝しております．また，濱幸太郎君とはディスカッションにおいて有益なヒントを得ることができ感謝しています．さらに，研究室での研究活動において，学内の事務手続きを行っていただいた森研究室の秘書の皆様に感謝します．特に大原なおみさんには，研究の進捗のご配慮や発表会への準備なども快く行っていただき感謝いたします．

今回，ＩＣカード乗車券システムの開発・実用化の業務を担当させていただいたうえ，博士号取得への機会を与えていただき，会社の業務と研究との両立に積極的にご支援いただきました，東日本旅客鉄道㈱ 大塚陸毅会長，石田義雄副会長，小縣方樹常務取締役，小倉雅彦常務取締役，浅井克巳取締役はじめ，Suica 部の皆様には心より感謝いたします．特に，本研究を進めるにあたり，論文作成，データの整理，海外発表の準備などを直接手伝ってくれた，山名基晴君，矢島武幸君，中川公人君には改めてお礼を申し上げます．

本研究を行うに当たり，研究対象である異種統合型自律分散ＩＣカード乗車券システムを開発し，研究に必要なデータを提供していただいた㈱ジェイアール東日本情報システム，ジェイアール東日本メカトロニクス㈱の関係者の皆様にもお礼を申し上げます．

最後に，家庭にあって終始私を支え，会社と大学とが両立するよう協力してくれた妻八重子と声援してくれた娘達に感謝します．

参考資料

［自律分散システム］

[1] 森欣司,宮本捷二,井原廣一:「自律分散概念の提案」, 電学誌C, Vol.104-C, No.12, pp15-22, 1984-12

[2] 藤沢真二:「世界にはばたく技術－自律分散システム－ニュースの印刷から流通を統合する新聞生産管理システム」, 電気学会誌, Vol.121, No. 3, pp115-118, 2001-3

[3] K.Mori, “Autonomous Decentralized Systems Technologies and Their Application to Train Transport Operation System”, HIS1999, pp1-13, Nov.1999

[4] 森欣司:「自律分散システム［Ⅰ］」, 電子情報通信学会誌, Vol.184,No.6, pp403-408,2001-6

[5] 森欣司:「自律分散システム［Ⅱ］」, 電子情報通信学会誌, Vol.184,No.7, pp484-490,2001-7

[6] 森欣司:「自律分散システム［Ⅲ］」, 電子情報通信学会誌, Vol.184,No.8, pp611-617,2001-8

[7] 森欣司:「自律分散システム［Ⅳ］」, 電子情報通信学会誌, Vol.184,No.9, pp663-669,2001-9

[8] 森欣司:「自律分散システム［Ⅴ］」, 電子情報通信学会誌, Vol.184,No.10, pp734-740,2001-10

[9] 織茂, 森, 井原:機能信頼度に基づくシステム分割の評価, 計測自動制御学会論文集, Vol,28,No.２,1992

[10] Kinji.Mori: ” Autonomous Decentralized System Concept, Data Field Architecture and Future Trends ”ISADS1993, Kawasaki, Japan, pp23-34, Apr.1993

[11] Kinji.Mori: et. al ” Autonomous Decentralized System Software Structure and It ’ s Application ”, IEEE Fall Joint Computer Conference, pp.1056-1063, Nov.1986

[12] Kinji.Mori: ” Autonomous Decentralized System [I. － V.] ”, IEICE, No.6-10,Vol.84,2001

[13] Kinji.Mori: ” Technology that flaps in the world -Autonomous Decentralized System(1) (2) ” ,The Institute of Electrical Engineers of JapanNo.2-3,Vol.121,2001

［アシュアランスシステム］

[14] I-Ling Yen, R.Paul and K.Mori, “Toward Integrated Methods for High-Assurance Systems”,IEEE Computer, Vol.31, No.4, pp32-34, 1998

[15] I-Ling Yen, R.Paul, “Key Application for High Assurance Systems”, IEEE Computer, Vol.31, No.4, pp35 -36, 1998

[16] 森欣司:「アシュアランスシステムのニーズと技術動向」, 電子情報通信学会, アシュアランス研究会, 1998-50, pp1-8, 1998-6

[17] 森:「情報と制御の統合した社会インフラに対するアシュアランス技術の動向」電子情報通信学会,2003

[18] M.Matsumoto, A.Hosokawa, S.Kitamura, D.Watanabe, A.Kawabata, “Development of theAutonomous Decentralized Train Control System”, IEICE Trans.Commun., Vol.84-D, No.10, pp1333-1340, Oct.2001

[19] 松本雅行, 森欣司:「自律分散型列車制御システムにおけるアシュアランス技術と評価法」,電子情報通信学会, Vol.J86-DI, No.1, pp14-22, Jan.2003

[20] 解良, 外 “大規模輸送管理システムにおけるアシュアランス技術”, Technical Report of IEICE. FTS99-29(1999-06)

[21] K.Kera, et. al, “Assurance System Technologies Based on Autonomous Decentralized System for Large Scale Transport Operation Control System” IEICE, Trans. COMMUN. Vol. E83-B, No.5, May 2000

[22] 矢代裕之, 高橋吉郎, 藤原暉雄:「宇宙機搭載分散型コンピュータシステムのアシュアランス性の検証」,第 44 回宇宙科学技術連合会講演会, pp83-89, 2000-10

[23] Leon Alkalai, Ann T.Tai, “Long-Life Deep-Space Applications”, IEEE Computer, Vol.31, No.4, pp37-38, 1998

[24] 松本雅行 2003.3 博士論文「システムの段階的構築におけるテストのアシュアランス性評価技術と列車制御システムへの適用の研究」

[25] 梶功夫: 異種システム共存のための自律分散アシュアランスシステムの研究,2001 年度博士課程卒業論文

[26] 解良和夫: アシュアランス性評価技術に基づく段階的システム構築技術の研究“ Step-by-step system constructiontechnology based on assurance evaluation technology ”,2003 年度博士課程卒業論文

［信頼性工学他］

[27] 薦田憲久, 矢島敬士:「企業情報システム」, コロナ社, 1999-10

[28] 当麻, 南谷: フォールトトレラントシステム, 電子通信学会誌, 63-10, 1031/1041,1980

[29] Skeen,D., ”Nonblocking Commit Protocols ”,Proc. Of the ACMSIGMOD(1981)

[30] Lamport,L., “Time Clocks, and the Ordering of Events in a Distributed System”, Comm. ACM, Vol.21,No.7, pp558-564(1978)

[31] Skeen, D. and Stonebraker, M., “A Formal Model of Crash Recovery in Distributed System” IEEE Trans. Software Eng., Vol.SE-9,No.3(1983)

[32] 谷村勇輔, 「クラスタおよび広域計算環境における並列分散遺伝的アルゴリズム」. 同志社大学大学院 博士論文, 2004.

[33] 飯塚泰樹 鈴木浩之 2005.12「Tabu Search を用いたマルチエージェント型分散制約充足手法の提案」社団法人電子情報通信学会　信学技報

[34] 2001 年 12 月 18 日 NRI 野村総合研究所「2006 年までの IT 主要分野の市場規模とトレンドを展望」

[35] A.S. Tanenbaum,: “Distributed Operating Systems”, Prentice Hall, Inc., 1995

[36] A.S. Tanenbaum,: “Computer Network”, Prentice Hall, Inc., pp502-506, 1996

［機能信頼性］

[37] 森欣司著:「自律分散システム入門（システムコンセプトから応用技術まで)」森北出版㈱ 2006.09

[38] M.D.Beaudry:“Performance-Related Reliability Measures for Computing Systems”, IEEE Trans.on Computers, C-27-6, 540/547, 1978

[39] J.F.Meyer: “ On Evaluating the Performability of Degradable Computing Systems”, IEEE Trans. on Computers, C-29-8, 720/731, 1980

[40] T.Inagaki et al.: “ Optimization of Staggered Inspection Schedules for Protective Systems ”, IEEE Trans. Reliability, R-29-2, 170/173, 1980

[41] Ihara.H and K.Mori: “ Fault-Tolerance Through Autonomous Decentralized System ”, IFIP Working Conf. On Reliable Computing and Fault-Tolerance in the 1980s, London, 1 10, Sept., 1979

[42] K.Mori, et al.:“ Autonomous Decentralized Loop Network ”, Compcon-Spring 82, 192/195, 1982

[43] H.Ihara and K. Mori:“Highly Reliable Loop Computer Network System Based on Autonomous Decentralization Concept ”, FTCS-12, 187/194, 1983

[44] 森・宮本・井原: “ 分散システムにおける機能と信頼性の評価法”, 計測自動制御学会論文集, pp. 314-321, 1984

[45] 織茂，森，井原：機能信頼度に基づくシステム分割の評価，計測自動制御学会論文集,Vol,28,No.2,1992

［自動出改札システム］

[46] Y. Naka “Study on complicated passenger flow in a railway station” Railway Technical research report, No.1079, 1978.

[47] A.Imai, “Examination of parameter of size of automatic fare collection gate”, OMRON TECHNICS, Vol.12, No.1, pp, 25-40, 1972

[48] S. Miki et al., “Development of contact-free IC card for railway ticket system,” Proc. IFAC CCCT ’89, Sep. 1989.

[49] S. Miki et al., “Contact-free IC card for new railway ticket system,” ASCE 2nd Conf. Appl. Adv. Tech. in Transportation Engineering, Aug. 1991.

[50] A.Shiibashi et al., “JR East Contact-less IC Card Automatic Fare Collection System ‘Suica’, IEICE (Institute of Electronics Information Communication Engineers), Vol. E86D－No.10, 2003

[51] S. Miki,“Total marketing strategy with non-contact IC card,” World Congress on Railway Research(WCRR,94), vol.1,pp.53-58, 1994

[52] Y. Naka “Study on complicated passenger flow in a railway station” Railway Technical research report, No.1079, 1978.

[53] A. Imai, “Examination of parameter of size of automatic fare collection gate”, OMRON TECHNICS, Vol.12, No.1, pp, 25-40, 1972

[54] S. Miki et al.,”Contactless smart card AFC trials in East Japan Railway Co.,”WCRR99, 1999.

[55] 椎橋章夫著：「自動改札のひみつ」成山堂書店（2003）4.1.3

[56] 椎橋章夫: IC カード乗車券システムにおける自律分散高速処理・高信頼性技術とそのアプリケーション，電子情報通信学会第 15 回アシュアランスシステム研究会，pp. 1-9, 2005

[57] 椎橋章夫: IC カード乗車券システムにおける自律分散高速処理技術とそのアプリケーション，計測自動制御学会産業論文集 vol4,No.7 41/49 (2005)

［鉄道システム］

[58] S.Yamanouchi, “Railways and Information Systems”, ISADS1999, Tokyo, Japan, pp2-9, Mar.1999

[59] F.Kitahara, H.Katano, T.Ono, Y.Kakumoto, K.Kikuchi, M.Shinomoto, “Distributed Management for Software Maintenance in a Wide-Area Railway System”, ISADS1997, Berlin, Germany, pp311-318, April.1997

[60] F.Kitahara, “Realization of the train operation control system for next generation(IROS & ATOS)”, UIC Signaling Seminar, Tokyo, Japan, pp.87-94, 1998

[61] F.Kitahara, H.Katano, T.Ono, Y.Kakumoto, K.Kikuchi, M.Shinomoto, “Distributed Management for Software Maintenance in a Wide-Area Railway System”, ISADS1997, Berlin, Germany, pp311-318, April.1997

［その他］

[62] 塩見: 『信頼性工学入門』, 丸善, 1967

[63] E.J.Henley and H. Kumamoto: “Reliability Engineering and Risk Assessment”, Prentice-Hall, Inc., 1981

[64] 佐藤　史隆, 廣安 知之, 三木 光範. 最短経路問題におけるアルゴリズム【ウォーシャル・フロイド法】の調査 ISDL Report　No. 20040716001.

[65] 佐藤　史隆, 廣安 知之, 三木 光範. 最短経路問題におけるアルゴリズム【ダイクストラ法】の調査 ISDL Report　No. 20040716002.

[66] 「データベース論Ⅰ　第 12 回データベース管理システム(2)－トランザクションと信頼性制御－」　Tomonori Gotoh Lab.

[67] 「WEB +DB PRESS Vol.20　実践 PostgreSQL 散策」日本 PostgreSQL ユーザ会理事長　石井達夫

[68] 月刊 Linux World　2005 年 5 月号 特別企画「PostgreSQL vs. MySQL」鈴木啓修

[69] 大村賢ほか:『顧客ニーズ指向のグローバル生産管理システム』, 電気学会誌, vol.121, No.2, 2001

[70] http://www.odva.org :(the organization that supports network technologies built on the Common Industrial Protocol (CIP™))

研究実績

１．　論文

[1] 椎橋「IC カード出改札システム“Suica”の開発と導入」;日本信頼性学会, Vol.25, No.8, pp.718-727, 2003.11

[2] 椎橋「リサイクル券売機の開発・導入」; 日本鉄道技術協会, Vol.44, No.10, pp.27941-27944 2001.10

[3] 白川・椎橋　「JR East Contact-less IC Card Automatic Fare Collection System “Suica” 」; IEICE(The Institute of Electronics Information Communication Engineers) , Vol.E86D No.10, pp.2070-2076, 2003.10

[4] 椎橋「Introduction and Future Development of Suica Non-contact IC Card Ticketing System」; Japan Railway & Transport Review, No.32,pp.20-27 2002.9

[5] 椎橋「有線／無線統合型 IC カードシステムにおける自律分散高速処理技術とそのアプリケーション」;計測自動制御学会 SICE 産業論文第４巻第７号[自律分散システム特集号], pp.41-49, 2005.08

[6] 椎橋「Autonomous Decentralized High-speed Processing Technology and the Application in an Integrated IC Card Fixed-line and Wireless System」 電子情報通信学会 (IEICE)「 自律分散小特集号」 Vol.E88-D, No.12, pp.2699-2707, Dec.2005

[7] 椎橋他 「AUTONOMOUS DECENTRALIZED DATA CONSISTENCY FOR HIGH-ASSURANCE EMBEDDED SYSTEM」International Scientific Journal of “Computing” Vol.4, Issue 2, pp.105-112, Jan.2006

[8] 椎橋「有線・無線統合型自律分散 IC カード乗車券システムの信頼性評価技術の研究」電子情報通信学会 (IEICE)「ディペンダブルコンピューティング」特集号, Vol.J89-D, No.8, pp.1623-1630, 2006.8

[9] 椎橋他「異種統合型自律分散 IC カード乗車券システムにおける高速処理・高信頼性技術の研究」情報処理学会論文誌「産学連携論文」特集号　2007.2 予定

[10]　椎橋他「High-speed Processing in Wired-and-Wireless Integrated Autonomous Decentralized System and Its Application to IC Card Ticket System」Innovations in Systems and Software Engineering, “A NASA Journal” 2007.3 予定 (NASA; National Aeronautics and Space Administration)

２．　査読付き国際学会発表

[1]「Autonomous decentralized high-speed processing technology and the application in an integrated IC card system with fixed-line and wireless」 ISADS, pp.215-223, 5 April, 2005, Chengdu, China

[2]「High-speed Processing and High Reliability in a Wired-and-Wireless Integrated Autonomous Decentralized IC Card Ticket System」 6th Asia-Pacific Symposium on Information and Telecommunication Technologies, pp.248-253, 10 November, 2005, Yangon, Myanmar

[3]「High-speed Processing in Wired-and-Wireless Integrated Autonomous Decentralized IC Card Ticket System」 ADVANCES IN COMPUTER SCIENCE AND TECHNOLOGY, pp.145-150, 23 January, 2006, Puerto Vallarta, Mexico

[4]「An Application of Autonomous Decentralized Architecture to an IC card Ticket System」 International Conference, Applied Computing, pp.107-114, 26 February, 2006, San Sebastian, Spain

[5]「High-speed Processing in Wired-and-Wireless Integrated Autonomous Decentralized System and Its Application to IC Card Ticket System」 3rd IEEE Workshop on Engineering of Autonomic and Autonomous Systems, pp.19-24, 29 March, 2006, Potsdam, Germany

[6]「Achievement of High-speed Processing by Autonomous Decentralized Processing and Decentralized Algorithm in a Wired-and-Wireless Integrated IC Card Ticket System」 3rd Workshop on Software Technologies for Future Embedded & Ubiquitous Systems, pp.163-174, 28 April, 2006, Gyeongju, Korea

[7]「HIGH PERFORMANCE IN A WIRED-AND-WIRELESS INTEGRATED IC CARD TICKET SYSTEM」 5th International Workshop on Wireless Information Systems, pp.56-65 23, May, 2006, Paphos, Cyprus

[8]「The evaluation of high reliability in an autonomous decentralized IC card ticket system」 7th International Symposium On Computer Networks, pp.209-213, 17 June, 2006, Istanbul, Turkey

[9]「High-speed Processing by Autonomous Decentralized Architecture and Decentralized Algorithm in a Wired-and-Wireless Integrated IC Card Ticket System」 3rd IEEE Conference on Enterprise Computing, E-Commerce and E-Services, p6, 26-29 June, 2006, San Francisco, US

[10]「Autonomous Decentralized Processing and Decentralized Algorithm for High Speed in a Wired-and-Wireless Integrated IC Card Ticket System」 IEEE

Symposium on Computers and Communications, pp.857-862, 26-29 June, 2006, Pula-Cagliari, Italy

[11] 「High Reliability in Autonomous Decentralized IC Card Ticket System」 International Workshop on Assurance in Distributed Systems and Networks, p2, 4 July, 2006, Lisbon, Portugal

[12] 「AN APPLICATION OF AUTONOMOUS DECENTRALIZED ARCHITECTURE TO AN IC CARD TICKET SYSTEM」 International Symposium on Speed-up and Service Technology for Railway and Maglev Systems, pp.359-366, 15 July, 2006, Chengdu, China

３． 招待講演

[1] Invited Speech ;「Development and Introduction of a Contact-less IC Card System “Suica”」;Fraunhofer Institute for Open Communication Systems, 8.July.2004. Berlin Germany

[2] Invited Speech ; 「Development and Introduction of a Contact-less IC Card System “Suica”」 ; University of Vienna Institute of Technology, 9.July.2004 Vienna Austria

[3] 「IC カード Suica の鉄道事業から生活サービス事業への展開戦略－IC カードとネットワークインフラとの連携技術」電子情報通信学会総合大会 2006.3.26

[4] 「IC カード(Suica)の開発と展開」日本信頼性学会 第 19 回秋季信頼性シンポジウム, 2006.10.20

４． 国内シンポジウム，研究会等発表

[1] 「IC カード (Suica) 出改札システム技術」; 日本画像学会, 技術研究会, 2001.11

[2] 「IC カード (Suica) の導入と IT 技術」; 日本機械学会, 関東支部 例会 2002.11

[3] 「Suica の IT 技術」; 日本フルードパワーシステム学会, 特別研修会 2004.6.11

[4] 「IC カード乗車券システムにおける自律分散高速処理・高信頼性技術とそのアプリケーション」 ; 電子情報通信学会 (IEICE) 第 15 回アシュアランスシステム研究会 2005.07.15

５． 受賞

[1] 「IC 乗車券改札機システムの開発と実用化」; 日本機械学会 技術賞受賞 2003.4 , 椎橋他４名

[2] 日本産業技術大賞・内閣総理大臣賞；日刊工業新聞社　2002.4.17,　企業
[3] 国土交通大臣賞　；国土交通省　2002.10.1,　企業
[4] 日本鉄道賞「情報化への貢献」部門；国土交通省　2002.10.14,　企業
[5] WITSA IT賞2004民間部門; World Information Technology and Services Alliance, 2004.5.20 ,Athens Greece　企業
[6] JR 東日本社長賞「IC カード出改札システムの開発と導入」; 東日本旅客鉄道(株)2002.2.6　椎橋
[7] ＩＣＴ事業奨励特別賞；自律分散型高アシュアランス技術；財団法人電気通信協会　2006.5.20　椎橋

６．　著書

[1] 「自動改札のひみつ」; 成山堂書店,2003.12　椎橋
[2] 「次世代交通カード革命」(共著)；NTT 出版, 1998.8　椎橋他 9 名

７．　特許

[1] 「記録媒体処理方法及び記録媒体処理装置」;特願平 7-292813　1995.11.10　椎橋他 8 名 1998/03/06 登録
[2] 「プリペイドカードの残り表示方法」;特願平 7-292812　1995.11.10　椎橋他 8 名 1997/09/12 登録
[3] 「割引定期券自動発券システム」; 特願平 9-40873　1997.2.25　椎橋他 3 名
[4] 「割引定期券自動発券システム」; 特願平 9-40874　1997.2.25　椎橋他 3 名
[5] 「入口規制システム」; 特願 2000-269921　2000.9.6　椎橋他 3 名
[6] 「移動機」; 特願 2000-269920　2000.9.6　椎橋他 3 名 2006/06/23 登録

８．　その他

[1] 非常勤講師：東京工業大学大学院イノベーションマネジメント研究科　技術経営専攻　技術経営戦略第一　2003 年度から現在に至る
[2] NHK「プロジェクトＸ」に Suica システムの開発者として出演　；2005.11.1