大規模分散処理システムのソフトウェア試験とその実践

坂井　俊之（日本電信電話株式会社）
梅田　昌義（日本電信電話株式会社）
中村　英児（日本電信電話株式会社）
本庄　利守（日本電信電話株式会社）

概要　現在 BigData を利用するための処理基盤として，様々な大規模分散処理システムが開発されている．筆者らも大規模分散処理システムである CBoC タイプ 2(Common IT Base over Cloud Computing)の開発において，システム検証の業務に従事してきた．本論文では，大規模分散処理システムにおける検証観点として，「資源効率性」，「障害許容性」，「回復性」が重要であることを主張する．これにより，一般的なソフトウェアを想定した網羅的検証に比べ，検査項目を 43%に絞り込みつつ高い品質の実現を図れた．また，検証実施後の分析からは，さらに検査項目を半減させつつ，300/100/30 の台数規模と検査項目の組み合わせ最適化を図ることで品質確保が可能であるとの見通しも得た．大規模分散処理システムにおける有効な試験についての考察を報告する．

1. はじめに

近年，BigData の利用が進んでいる．BigData とは，web ページや画像・動画，センサ・データ等の，大容量かつ発生速度が速い，多種多様なデータを指すキーワードである[1,2]．このような BigData を利用する機運が高まってきたのは，Google の検索基盤[3]や，Amazon の各種サービス[4]等， BigData を利用したサービスが登場してきたためであると考えられる．

しかし，BigData を利用する際，従来のリレーショナルデータベースでは容量や速度の面で，要求に応えられないという課題があった[5]．そのため，各企業や組織では BigData を利用するための NoSQL データベース[6]と呼ばれる大規模分散処理システムを独自に開発してきた．

NTT ソフトウェアイノベーションセンタ（旧ＰＦ研）においても，BigData を扱うための処理基盤である大規模分散処理システム（CBoC タイプ 2(Common IT Base over Cloud Computing)[7]）のソフトウェア開発を進めてきた．ソフトウェア開発においては，ソフトウェアの動作を確認する試験が必須であり，Ｖ字モデルの開発モデルでは，開発フェーズに応じて複数の試験が存在する．最終のシステム検証のフェーズでは，実運用する環境やユースケースに沿った試験によって，機能要件や非機能要件に対して要求された品質を満たすことを確認する．ただし，要求される品質はシステムの特性によって大きく異なるため，検証観点もシステムの特性によって異なる．そのため，試験計画時には，まず重視する検証観点を決定する必要がある．本論文では，大規模分散処理システムのシステム検証において重視すべき検証観点を明確化する．

本論文の構成を以下に述べる．第 2 章では，大規模分散処理システムが持つ特性，及び，CBoC タイプ 2 について述べる．第 3 章では大規模分散処理システムの検証で重視すべき検証観点について述べる．第 4 章では，上述の検証観点に基づく試験項目の実施結果から，抽出できた問題が重視すべき検証観点に対応した問題であることを示す．加えて，問題発生の要因を考察し，大規模分散処理システムの検証で有効な試験を紹介する．最後に第 5 章では, 全体のまとめと今後の課題について述べる．

2. 大規模分散処理システム

大規模分散処理システムの検証について述べる前に，まず大規模分散処理システムの特性，及び，今回検証対象となる CBoC タイプ 2 の紹介を行う．

2.1 大規模分散処理システムの特性

第 1 章で述べたように，現在様々な大規模分散処理システムが開発されているが，これらのシステムはコモディティサーバを大量に並べることで，大規模なデータの蓄積，処理を可能にするものである．それらの特徴としては，リソース（サーバ数）を動的に変化させることで，要求されるデータ量や処理量にスケーラブルに適応することができる「スケールアウト性」を備えている．また，大量のコモディティサーバを扱うことに伴い頻発するサ

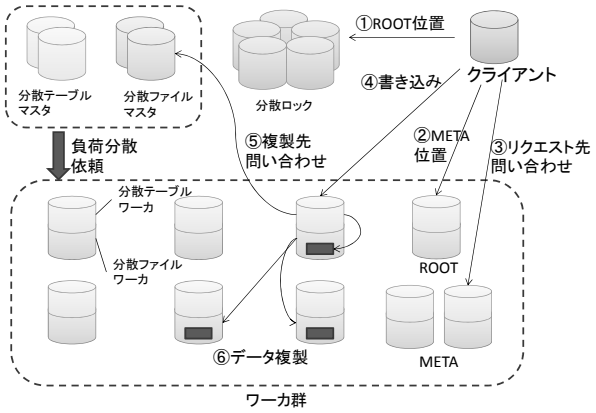


図 1. 書込みの処理モデル

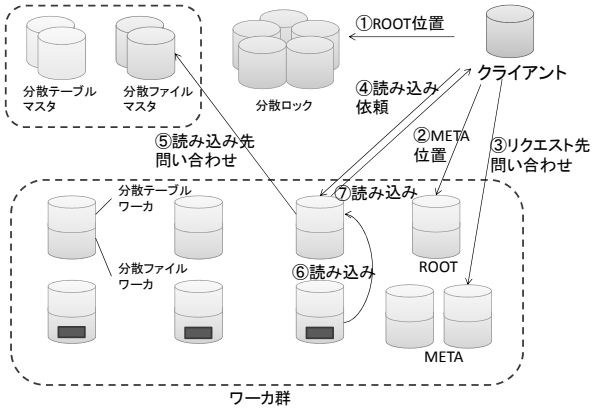


図 2. 読込みの処理モデル

ーバやネットワークの故障に対する「耐障害性」を備えており，故障を前提としたソフトウェアの作りとなっている．

2.2 CBoC タイプ 2

CBoC タイプ 2 は大規模分散処理システムの 1 つである NoSQL データベースである．分散データベースでは整合性・分断耐性・可用性の 3 つを完全には満たせないと言われている[8,9,10]．これに対して，CBoC タイプ 2 は Google の大規模分散処理システム（「Bigtable[11]」，「GFS[12]」，「Chubby[13]」）と同様の特性をもち，整合性，分断耐性が強いシステムである．ユーザは自身で整合性を取る必要がなく，ネットワークが分断された際でも，分断されたサーバ群のうち，小規模な群を自動で切り離して正常に動作することができる．可用性に関しては，単一機器の障害によるデータ消失はなく，リカバリ機能によって障害が発生した部分を自動で切り離して動作するが，リカバリ中は障害が発生した部分に関して応答をすることができない．

2.2.1 CBoC タイプ 2 の構成

CBoC タイプ 2 は, 以下の 3 システムから構成される．

- ●分散テーブル：クライアント（ユーザ）にカラム指向のデータベース機能を提供する．
- ●分散ファイル：ファイルシステム機能を提供し，分散テーブルに書き込まれたデータの実体を格納する．
- ●分散ロック：分散テーブル，分散ファイルに対して，死活監視やイベント通知を実施する．

2.2.2 CBoC タイプ 2 の処理モデル

ここでは CBoC タイプ 2 の主要処理である，データベースへの書込み・読込み処理とリカバリ処理の処理モデルについて説明する．

データベースへの書込み・読込み処理では，サーバ台数に応じて処理量にスケーラブルに適応するため，処理の分散と, 2.2 節で述べた整合性による処理待ちの発生に関して，「スケールアウト性」を意識したデザインとなっている．リカバリ処理においては, 2.2 節で述べた分断耐性や可用性に関して「耐障害性」を意識したデザインとなっている．以下では，各処理モデルの詳細について説明していく（図 1〜3 参照）．

- ●書込み・読込みの処理モデル：書込み・読込み処理では，クライアントからのリクエストに応じて，データベースへの書込み・読込み処理を実施する．その際，処理の分散やデータの複製が実施される．処理の分散では，クライアントからの書込み・読込みリクエストを複数のワーカに分散させ，並行に処理することによって，システム全体の処理性能が向上する．以下に，分散テーブル, 分散ファイルの分散方式を示す.
  - ▪ 分散テーブルにおける処理分散方式: 分散テーブルでは，マスタが処理を担当するワーカを決定する．マスタによって決定された情報は ROOT・META に保持され, クライアントからのリクエストに応じて処理を担当するワーカの位置を通知する. その結果, 処理が分散される（図 1，2 の①〜④）．
  - ▪ 分散ファイルにおける処理分散方式: 分散ファイルでは, リクエストを処理するワーカの決定をマスタが行う．分散ファイルへのリクエストは，リクエストを処理するワーカの位置をマスタに問い合わせることで処理が分散される（図 1，2 の⑤〜⑥）．

  また，データの複製処理では，書込み時に複数個の複製を作成し，かつ，複製間での整合性を保つことでデータの消失を防ぐ．読込み時は，最も近いワーカの複製データを読み出す.
- ●リカバリの処理モデル：リカバリ処理では，ネットワークの分断やサーバ障害が発生した場合に，分断され

た小サーバ群や障害発生サーバを自動的に切り離し，別のサーバに切り離したサーバの役割を担わせる．ここで，マスタはフェールオーバにより役割を交代し，ワーカは，別のワーカに処理の割り当てを変更する，又は，複製データをコピーすることによって役割を変更する．そして，分断や障害の検知は分散ロックの死活監視によって実施される．これにより，システム全体を停止させずにクライアントからのリクエストに応答することが可能である．

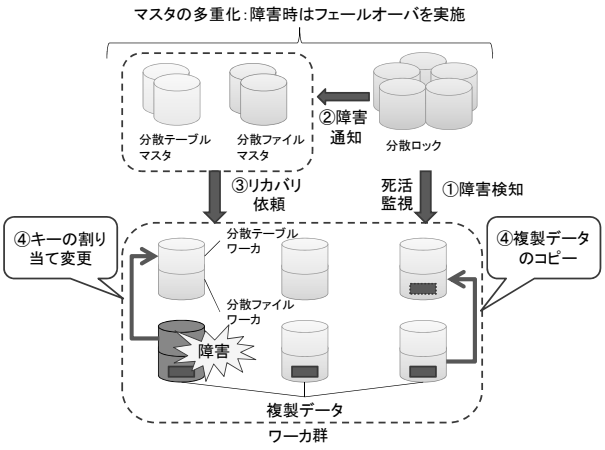


図 3. リカバリの処理モデル

3. 大規模分散処理システムの検証

本章では大規模分散処理システムの検証における特徴について，一般的なソフトウェアテストのプロセスを用いて説明する．特に「分析と設計」プロセスに関して，大規模分散処理システムで重視すべき検証観点の詳細を述べる．

3.1 システム検証の位置づけ

ソフトウェア工学の分野において，ウォーターフォール型のソフトウェア開発は，図 4 のように V 字モデルで形式化できる．また，V 字モデルで抽出される不具合例を以下の表 1 に示す．システム検証は V 字モデルの検証の最後の工程であり，全プロダクトを連携させて検証を行い，システムの要求仕様の満足と実運用での安定性を確認する．

3.2 システム検証のテスト設計

大規模分散処理システムにおけるシステム検証のプロセスは，一般のソフトウェアテストのプロセスと同様であり，JSTQB（ISTQB）[14]で定義されているプロセスと同じである． ただし，大規模分散処理システムでは数百台規模で検証を実施することから，JSTQB のプロセス毎

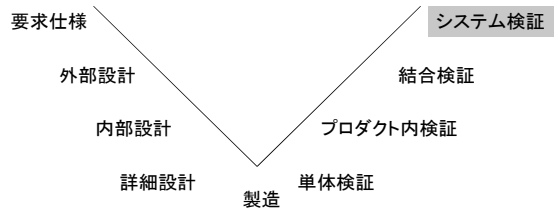


図 4. V 字モデル

表 1. V 字モデルにおけるシステム検証の位置づけ

検証工程	検証単位	品質達成レベル	抽出される不具合例
単体検証	サブシステム（各プロダクトを構成）	C0/C1 100%	単体ロジック誤り，コーディング誤り
プロダクト内検証	サブシステム間連携	機能・性能の確保	プロダクト内ロジック誤り，単体性能劣化，詳細設計誤り
結合検証	各プロダクト間連携	機能・性能の確保	プロダクト間ロジック誤り，IF 誤り，基本/機能設計誤り
システム検証	プロダクト全体	実運用でのシステム安定性確認	ユースケースロジック誤り，性能劣化，リソース不足，仕様誤り

表 2. 大規模分散処理システムでの各プロセスの特徴

プロセス			特徴
①	計画とコントロール		特になし
②	分析と設計	分析	テスト条件の，マシン台数に応じた設定
		設計	テスト設計は，特徴のある検証観点に対して重点的に実施
③	実装と実行	実装	ツールによる環境設定の自動化
		実行	ツールによる自動化
④	終了基準の評価とレポート	評価	特になし
		レポート	特になし
⑤	終了作業		特になし

の特徴を示した表 2 のように，検証の各プロセスにおける具体的な作業内容に関しては特徴が生じる．

これらのプロセスのうち「分析と設計」すなわちテストを設計する工程は，検証自体の有効性を左右する重要な工程である．このテスト設計に関してはソフトウェア工学では様々な手法が提案されているが，検証対象とするシステムに応じてどの手法が適切で効果的かはオープンクエスチョンであり，各システム開発に応じて試行錯誤しているのが現状である．

3.3 大規模分散処理システムの検証観点

3.2 節で述べた検証プロセスにおいて, 大規模分散処理システムの品質を確保する上でポイントとなる「分析と設計」のプロセスを中心に説明する.

「分析と設計」のプロセスでは，テスト対象の検証観点を抽出することがポイントであることから，検証観点は JIS X 0129-1(ISO9126-1)[15]で示された品質特性の観点の中からシステムの特徴と対応する観点を抽出する必要がある．したがって大規模分散処理システムの場合，第 2 章で述べたように，大規模分散処理システムの特徴である「スケールアウト性」と「耐障害性」を検証する内容が含まれるべきである．以下に各特徴に対する検証観点を述べる.

「スケールアウト性」の検証観点としては，「資源効率性」が主要な観点である．その理由は，要求されるデータ量や処理量にスケーラブルに適応するため，システムを構成する多数のサーバ上で負荷が均等に分散されるか，システムの構成上で処理が集中する箇所がボトルネックになっていないかなどを検証し，リソース全体を効率的に利用して所望の性能が得られるか否かを検証する必要があるからである． 特に, データの蓄積量の増加に伴って，サーバの台数を増やした際に，それに応じた処理性能の向上が得られるか否かの検証も行う必要がある.

「耐障害性」の検証観点としては，「障害許容性」，「回復性」が主要観点である．その理由は，「障害許容性」ではシステムが動作するサーバやネットワークが故障した際にもシステム全体が正常に機能を提供し続けることを検証する必要があるからである．つまり，一部のサーバやネットワークが故障により機能しなくなった際にも，システム全体として，正常に機能を提供し続けることを確認する検証を行うことがポイントとなる．ただし，システムの基幹となる役割を担うサーバーが故障を受けた場合には，スタンバイ中のサーバが機能を始めるまでの間，短期間ではあるが，サービスが停止することを考慮する必要がある．一方，「回復性」の検証としては，上記のような場合において，サービスが停止する期間，サーバの役割が切り替わる時間，さらにはこれらの機能が正しく動作するかなどの検証を行う必要があるからである.

以上から，大規模分散処理システムのシステム検証においては，「障害許容性」，「回復性」，「資源効率性」の観点が重要となる．これらの観点に従い検証項目を作成した結果，一般的なシステム検証の分類[16]（表 3）と比べ, 大規模分散処理システムでは表 4 の分類となった. この理由が上記観点にあることを第 4 章で示す.

表 3. 一般的なシステム検証の項目抽出

	機能性	信頼性	使用性	効率性	保守性	移植性
正常系・準正常系	○	○	○	○		
障害復旧	○	○	○	○		
性能・高負荷	○	○	○	○		
長期安定性	○	○	○	○		

表 4. 大規模分散処理システムの検証項目抽出

	機能性	信頼性	使用性	効率性	保守性	移植性
正常系・準正常系	○					
障害復旧		○	○		○	
性能・高負荷				○		
長期安定性		○	○		○	

4.検証実施結果と考察

本章では，第 3 章で述べた検証観点に従って検証項目を作成し，実際に検証を行った結果を分析する．また，発生した問題の要因を分析することで，有効な試験項目を考察し，「障害許容性」，「回復性」，「資源効率性」が重要な観点であることを示す.

4.1 検証項目の精査

実際に実施した検証項目を表 5 に示す．検証は網羅的に抽出した時点では約 600 項目あったが，第 3 章で述べた「障害許容性」，「回復性」，「資源効率性」の検証観点において精査をした結果，約 260 項目となった.

一般的に項目の精査は，「同値分割」と「境界値分析」により実施するが，通常は項目抽出時にすでにこの精査は行われており，更なる精査を行う必要がある.

今回，当初は「ユースケース機能試験」，「正常系負荷」，「スケールアウト」の内容を網羅的に抽出していたが, 正常系機能試験で確認を兼ねることができるため, 大幅に項目を削減した．一方，「上位アプリのプロセス中断」，「フェールオーバ」，「CBoC プロセスダウン」については，「障害許容性」，「回復性」の観点で項目を追加した．最後に，「組み込み」については「資源効率性」の観点で項目を追加した.

上記のようにユースケースの観点と大規模分散システムのアーキテクチャの特徴に基づく「障害許容性」，「回復性」，「資源効率性」の観点で精査を行い，大規模分散処理システムに特化した項目が完成した.

表 5. 精査をした大規模分散処理システムの検証項目

大項目	中項目	
ユースケース機能試験	システム起動・停止	今回削除
	上位アプリからの書き込み	今回削除
	上位アプリからの読み込み	今回削除
	コマンド確認	今回削除
高負荷，異常系複合試験	正常系負荷	今回削除
	異常系複合	
	機能試験用負荷	今回削除
性能		
長期安定試験	スケールアウト	今回削除
	長期安定性試験	
運用コマンド		

約600項目

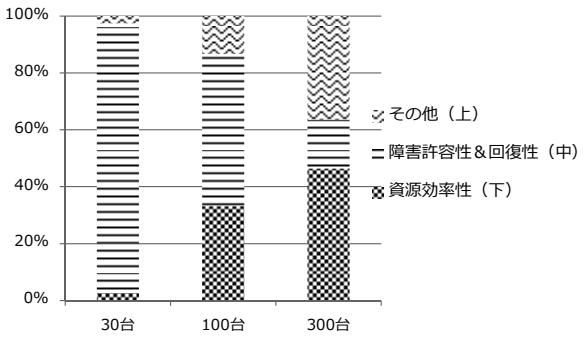
大項目	中項目	
ユースケース機能試験	正常系機能	今回追加
	上位アプリのプロセス中断	
CBoCの系構成変更	フェールオーバ	今回追加
	サーバ組み込み	今回追加
CBoCの単一障害	CBoCプロセスダウン	今回追加
	NW障害	今回追加
複合・多重障害	CBoCフェールオーバ（1次、2次）	今回追加
	CBoCプロセスダウンの組み合わせ	
性能		
長期安定試験		
運用コマンド		

約260項目

4.2 効率的な検証の実施方法

試験実施の方法としては，試験環境として，300/100/30 台の 3 環境（15～20 サーバ/1 ラック）で試験を実施し，それぞれ 300TB/7.4TB/21TB 規模のデータを登録した状態で検証を実施した．長期安定性試験や，障害系機能試験における負荷としては，1 サーバあたり 0.41MB/sec（ランダムライト），6.9MB/sec（シーケンシャルリード），2.1KB/sec（ランダムリード）で実施した．検証項目に関しては，30 台環境では，ほとんどが障害系観点の検証であり，小規模環境において，短時間で数多くの検証を実施した．300 台環境では，性能（資源効率性）を中心に正常機能（その他観点），準正常・異常機能（障害許容性・回復性）を実施した．100 台環境は，当初の実商用想定環境であり，正常機能の確認とともに運用時に発生すると想定される準正常・異常系機能の確認を効率的に実施した（表 6）．

表 6. 300/100/30 台の環境の項目割合



4.3 発生した問題の分析

4.1 節で完成した検証項目によって抽出できた問題を表 7 に示す．本節では，問題発生要因の分析結果から，更に効率的かつ効果的な検証項目について考察する．

表 7. 抽出した具体的問題

概要	問題
正常に動作しない	マシン不具合
上位APが停止	通信途絶時のリカバリ不具合による管理情報の欠如
登録のスループットが低下	分割の閾値が大きすぎた
サーバの切り替えが突然発生	サーバ側での旧接続に関する後処理の抜け
圧縮の処理のタイムアウトが発生	ログレベルの設定誤り
登録のスループットが低下	通信途絶時でのデッドロック
登録したデータが見つからない	デッドロック
復旧処理時にデータが破損	他の管理情報の混在による影響
サーバ追加ができない	復旧処理の改修漏れ
登録のスループット低下	ロック取得による処理遅延
書き込みのスループットが不安定となる	圧縮処理のスケジューリングが同時に実施
サーバ追加ができない	古いセッション管理情報を使ってしまっている
サーバの切り替わりが突然発生	リソース不足による切り替わり発生
収集したデータが見つからない	通信ディスクリプタのマップからの削除もれ
障害復旧の処理が完了しない	要求処理の滞留
上位APが停止	リカバリが完了する前にクローラのリトライアウト
障害復旧できずにプロセス停止	差分データが大きい場合の仕様
上位APが停止	MapReduceでのタイマとの差異
障害復旧の処理が完了しない	ロック期間の短縮不足
上位APが停止	システムコール不具合
障害復旧の処理が完了しない	ハーフオープンされたFile Descriptionの処理の考慮漏れ
書き込みのスループットが不安定となる	圧縮処理のスケジューリングが同時に実施

表 8. 発生した問題の主な要因

検証観点	要因分類	要因の概要
① 資源効率性	1. 処理集中	・分散条件による処理集中 ・永続化処理の時間的集中
	2. 排他処理	・大量データ処理との競合 ・通信処理でのデッドロック
	3.リソース設定	・リソース設定誤り
② 障害許容性	1. 大規模障害	・大規模障害による処理遅延
	2. 障害復旧タイミング	・接続情報のキャッシュ削除漏れ ・複製データへのアクセスタイミングの考慮漏れ
	3. 障害復旧パターン	・通信障害パターン考慮漏れ ・旧接続のリンク切り漏れ ・想定外の戻り値に起因するリトライ動作誤り
	4. OS	・システムコールの動作不良
③ 回復性	1. キャッシュ	・キャッシュの削除漏れ

4.3.1 問題発生要因の分析

発生した問題に関して要因の分析を実施した（表 8）．以下では各要因について詳細を述べる．

●要因分類①-1 処理集中（資源効率性）

処理集中では，分散条件による処理集中として，META へのアクセス集中が見られた．META はデータ量・サー

バ台数の増大で応答が遅くなるため，今回設定した分散条件（データ量）では分散する前に応答が遅くなり，アクセスが集中した状態が継続してしまった．

一方，永続化処理の時間的集中においては，分散テーブルワーカが増大したメモリ内の更新データをディスクに永続化する（コンパクション）処理が大量に発生し，ビジー状態になった．

●要因分類①-2 排他処理（資源効率性）

大量データ処理との競合では，分散ファイルのバックグラウンド処理である削除処理と更新処理の競合が発生した．この場合データ量が多いほど削除期間も長くなり，競合の発生期間も長くなっていた．

また，通信処理でのデッドロックは，分散テーブルへのリクエストが強制切断された際に発生するが，問題の発生はタイミング依存であった．

●要因分類①-3 リソース設定（資源効率性）

リソース設定誤りでは，長期運転中に分散ロックでメモリプールサイズ不足が発生した．

●要因分類②-1 大規模障害（障害許容性）

大規模障害による処理遅延では，多数のワーカダウンにより，分散ファイルマスタの管理情報からダウンワーカ情報の削除が遅れ，マスタからダウンワーカに要求が送られたため，ダウンワーカは要求を処理できず，マスタで処理中の要求が増加し，ビジー状態となった．

●要因分類②-2 障害復旧タイミング（障害許容性）

接続情報のキャッシュ削除漏れでは，クライアントでキャッシュしている以前の接続情報（ソケットの fd と IP アドレス）が削除されないタイミングがあり，同じ fd に対し 2 つの IP アドレスが存在してしまった．

一方，複製データへのアクセスタイミングの考慮漏れでは，分散ファイルのクライアントがワーカの位置情報のキャッシュを参照した際，ダウンした最近傍のワーカにアクセスし，次のワーカに読みに行くまでのリトライで処理遅延が発生した．これは，キャッシュがすぐに削除されない読み方だったために問題が発生した．

●要因分類②-3 障害復旧パターン（障害許容性）

通信障害パターン考慮漏れでは udp が通信可能であるのに対し，tcp のみ通信不可であったため，分散ロックは udp の死活監視でサーバが正常に動作していると判断したが，分散テーブルや分散ファイルは tcp の通信障害により正常に動作しなかった．

また，旧接続の切り漏れでは，分散ロックのクライアントとサーバ間の死活監視で接続が一時的に切れたため，再接続を開始したが，クライアントから遅れて届いた FIN パケットによって再度接続が切断されてしまった．

最後の想定外の戻り値に起因するリトライ動作誤りでは，コアスイッチの一部のポートが閉じられたため，分散ロック間でシステムコールを用いてセッションクローズを要求したが，スイッチ上でパケットがドロップし，システムコールからエラーが返却されなかったため，リトライが実施されなかった．

●要因分類③-1 キャッシュ（回復性）

キャッシュの削除漏れでは，分散ロックのフェールオーバ後に実施される復元処理でキャッシュのフラグが無効になっているケースがあり，フェールオーバ後のキャッシュ関連処理で問題が発生した．

4.3.2 大規模分散処理システムにおいて有効な試験

大規模分散処理システムは，大規模環境で安定動作すること，一部の機器における故障発生時も動作し続けることが重要である．4.3.1 節で述べた問題の大部分はこれらの点に反する問題であり，「資源効率性，障害許容性，回復性」の項目を重視したからこそ発見できた問題である．仮に上述の項目を重視せず，これらの問題を発見できなかった場合，システムの再立ち上げ，スループットが不安定，データ消失による再投入，上位アプリにおける処理の再実施等の事象が発生すると考えられ，非常に使い勝手の悪いものになる．

本項では，4.3.1 節で述べた要因から考察して導き出した，「資源効率性，障害許容性，回復性」の各検証観点において有効な試験（確認観点含む）を表 9 に示す．

「資源効率性」の観点では，「処理集中」において，データの分散やコンパクションという大規模分散処理システムに特有な処理の発生前後において性能低下が発生しており，試験を実施する際は上記処理の発生状況や発生前後の性能について注意する必要があることが分かった．また，「排他制御」に関しては，バックグラウンド処理や運用コマンドにおいて，大規模分散処理システムの特徴である大規模データを取り扱う際の性能への影響を確認する必要があることが分かった．他にも，大規模分散処理システムはネットワーク(NW)を介して各サーバが協調して動作するため，NW 系の高負荷時に通常とは異なる動作（システムコールの失敗や，タイムアウト等）が発生することで，動作に支障がないかも確認する必要がある．

一方，「障害許容性」や「回復性」の観点では，大規模分散処理システムがある程度の多重障害時でも動作するように設計されていることから，多重障害の試験を重視して実施することにより，様々な障害パターン，タイミングでの試験が実施され，問題が顕在化するケースが

見られた．このことから，障害系の加速試験も有効であると思われる．また，大規模障害によって処理遅延が発生し，1 台の故障では発見できないような問題も抽出できた．他には，長期安定性試験により，特殊ケースの障害に関する問題の抽出ができると考えられる．

表 9. 大規模分散処理システムで有効な試験や観点

検証観点	試験項目分類	有効な試験（確認観点含む）
資源効率性	・性能，高負荷試験 ・長期安定性試験	・大規模サーバ環境でデータが分散する過程と分散完了後の性能差が設計通りか． ・バックグラウンド処理（GC，コンパクション等）が設計通りに動き，性能への影響が想定通りか． ・リード/ライト中かつ大規模データで運用コマンドが要件時間内に完了するか． ・バックグラウンド処理数の時間推移が想定通りか． ・NW の高負荷が断続的に発生した場合，上位アプリへの影響は設計通りか．
障害許容性 回復性	・障害復旧 ・長期安定性試験	・多重障害（障害中/復旧後の障害） ・大規模障害時（ラックダウン等）の障害復旧とサービス持続性 ・障害系加速試験（クライアントからの連続接続・切断時の安定動作等） ・特殊障害発生時（特定プロトコル，一部ポートの遮断等）に設計通りの動作か．

4.4 観点の試験項目へのフィードバック

4.3 節の分析結果で得られた有効な試験や観点を 4.1 節の試験項目にフィードバックし，更に項目の精査を行った結果を表 10，11 に示す．表 10 の試験項目へのフィードバック結果としては，「資源効率化」のフィードバックとして「上位アプリ性能」，「性能」の項目の強化を実施し，「障害許容性」，「回復性」のフィードバックとして「CBoC プロセスダウンの組み合わせ」，「運用コマンド」の強化を実施した．また，表 11 の検証項目数の割合では障害・復旧試験が大半を占めており，一般的な検証の割合の，正常系 60%，異常系 15%，その他が 25% とは異なる．これは障害発生プロセスや障害種別，同時障害発生数などの組合せをもとに検証項目を抽出しているためである．検証観点との対応については，「資源効率性」は性能・高負荷試験や長期安定性試験，「障害許容性」「回復性」は障害・復旧試験，長期安定性試験に対応している．

表 10. 問題分析によるフィードバックを行った項目

大項目	中項目	
ユースケース機能試験	正常系機能	今回追加 次版削除
	上位アプリのプロセス中断	
CBoC の系構成変更	フェールオーバ	今回追加
	サーバ組み込み	今回追加
CBoC の単一障害	CBoCプロセスダウン	今回追加
	NW障害	今回追加
複合・多重障害	CBoCフェールオーバ（1次、2次）	今回追加 次版削除
	CBoCプロセスダウンの組み合わせ	
性能		
長期安定試験		
運用コマンド		

約260項目

大項目	中項目	
ユースケース機能試験	上位アプリ性能	次回追加
CBoC の系構成変更	フェールオーバ	
	サーバ組み込み	
CBoC の単一障害	CBoCプロセスダウン	
	NW障害	
複合・多重障害	CBoCプロセスダウンの組み合わせ	強化（フェールオーバと組み合わせ）
	ラックダウン・復旧	次回追加
性能	データを変化させた場合の測定	強化
長期安定試験		
運用コマンド		強化（異常系と組み合わせ）

約130項目

表 11. 検証実施項目数の割合

試験項目分類	主な試験内容	項目数割合
正常/準正常系	ユースケースでの読み書き，読み書き中の保守コマンド投入	23%
障害・復旧	障害プロセス・障害種別・試験時の負荷の組合せ（二重/三重故障含む），大規模故障，障害復旧中の障害	43%
性能・高負荷	読み書き，NW・DISK 高負荷，障害復旧性能	25%
長期安定性	長期運転時の障害・復旧（保守コマンド投入含む）	8%

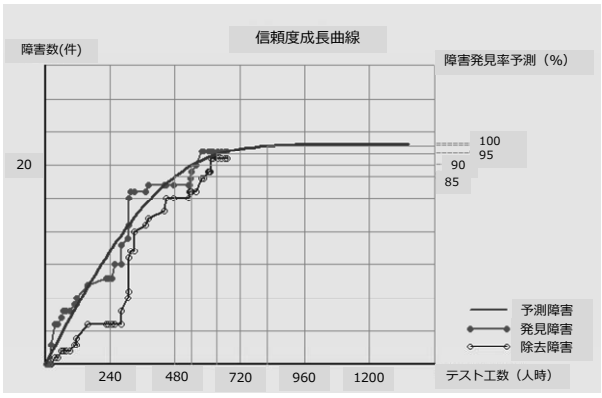


図 5. 表 5 の完成項目での信頼度成長曲線

4.5 検証の結果と考察

表 5 で示した 260 項目のシステム検証の実施により得られた信頼度成長曲線を図 5 に示す．障害も収束していることから，観点の絞込みにより網羅的な項目抽出と同等な確認が可能であることが判明した．また，表 10 で示

した 130 項目のシステム検証の実施により予測される信頼度成長曲線を図 6 に示す．この結果により，注力すべき項目に絞って更に効率的に確認が可能となることが判明した．

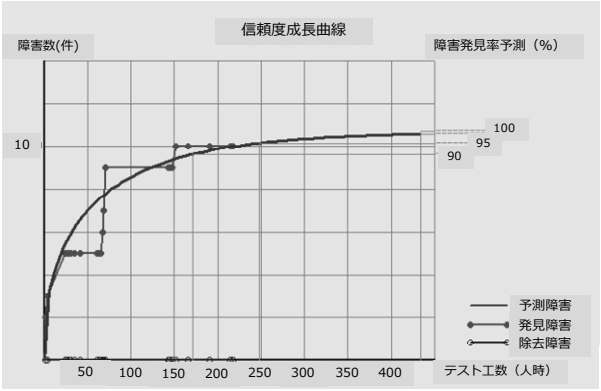


図 6. 表 10 の完成項目での信頼度成長曲線

環境別の問題発生件数では, 表 12 を見ると検証 1 項目あたりに発生する平均問題数が 0.069 件であり，30 台・100 台では障害許容性と回復性の観点に対応する問題が平均より多く抽出できることになる．また，資源効率性についても，100 台・300 台環境で平均程度の問題数を抽出できる．これは，狙い通りに効率良く検証が実施できることを表す． 一方，その他観点では，ログの設定の問題の摘出であり，システム検証における重要性は低い．

表 12. 検証環境別の問題発生割合

	資源効率性	障害許容性&回復性	その他	平均
30台	0.000	0.083	1.000	0.105
100台	0.071	0.132	0.000	0.094
300台	0.062	0.042	0.000	0.036
平均	0.065	0.102	0.014	0.069

5. まとめと今後の課題

本論文では，大規模分散処理システムにおけるシステム検証の観点として，「資源効率性」，「障害許容性」，「回復性」が重要であることを述べ，実際にその観点に対応する問題が多く発生することを確認した．更に，問題の要因を分析することで有効な試験や観点を示した．

今後は更なる多重障害や，タイミング系の問題を抽出する加速試験のため，検証の自動化が重要である．

謝辞 本論文の作成にあたり, 貴重なコメントを頂きました，旧 NTT 情報流通プラットフォーム研究所や NTT ソフトウェアイノベーションセンタ等の CBoC タイプ 2 プロジェクトにご協力頂いた皆様に深謝します．

参考文献

1) 武田浩一,井出剛: ビッグデータ処理の展望,IBM PROVISION No.72 ,pp40-45 (2012).
2) Gartner: Gartner Says Solving ‘BigData’ Challenge Involves More Than Just Managing Volumes of Data, http://www.gartner.com/it/page.jsp?id=1731916(2011).
3) 西田　圭介: Google を支える技術～巨大システムの内側の世界,技術評論社(2008).
4) G.DeCandia *et al*: Dynamo: Amazon’s Highly Available Key-value Store, Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles,October 14-17,Stevenson,Washington, USA(2007)
5) Robin Hecht and Stefan Jablonski: NoSQL evaluation: A use case oriented survey,csc,2011 International Conference on Cloud and Service Computing ,pp336-341(2011)
6) Junichi Niino: 「NoSQL」は「Not Only SQL」である，と定着するか？, http://www.publickey1.jp/blog/09/nosqlnot_only_sql.html(2009).
7) 鷲坂 光一ほか: 大量データ分析のための大規模分散処理基盤の開発,NTT 技術ジャーナル,Vol.23,No.10,pp22-25(2011).
8) Eric A. Brewer: Towards Robust Distributed Systems, PODC Keynote, http://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf (2000).
9) Nancy Lynch and Seth Gilbert: Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services, ACM SIGACT News,Volume 33 Issue 2,pp51-59 (2002).
10) Jing Han *et al* : Survey on NoSQL Database,Pervasive Computing and Applications (ICPCA),2011 6th International Conference,pp 363-366(2011).
11) Fay Chang *et al* : Bigtable: a distributed storage system for structured data,Proceedings of the 7th symposium on Operating systems design and implementation,November 06-08,2006,Seattle, Washington(2006).
12) Sanjay Ghemawat *et al*: The Google file system, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22(2003).
13) Mike Burrows: The Chubby lock service for loosely-coupled distributed systems,7th USENIX Symposium on Operating Systems Design and Implementation (2006).
14) JSTQB: テスト技術者資格制度 Foundation Level シラバス ,http://jstqb.jp/dl/JSTQB-Syllabus.Foundation_Version2011.J01.pdf(2011).
15) JISC: JIS X0129-1,http://www.jisc.go.jp/app/pager?id=7755(2003)
16) 加藤 大受: テスト設計のとっかかり, JaSST'07 Kyushu, http://jasst.jp/archives/jasst07k/pdf/T1.pdf

坂井　俊之（正会員）
E-mail: sakai.toshiyuki@lab.ntt.co.jp
NTT ソフトウェアイノベーションセンタ 第二推進プロジェクト 研究員. 2007 年東北大学大学院工学研究科修士課程修了．同年，日本電信電話（株）入社．現在，大規模分散処理システムの研究開発に従事.

梅田　昌義（正会員）
E-mail: umeda.masayoshi@lab.ntt.co.jp
NTT ソフトウェアイノベーションセンタ 第二推進プロジェクト 主任研究員. 1991 年 電気通信大学 電気通信学部卒．同年，日本電信電話（株）入社．現在，大規模分散処理システムの研究開発に従事.

中村　英児（非会員）
E-mail: nakamura.eiji@lab.ntt.co.jp
NTT ソフトウェアイノベーションセンタ 第二推進プロジェクト 主幹研究員. 1987 年北海道大学大学院原子工学専攻修士課程修了. 同年, 日本電信電話（株）入社．現在，大規模分散処理システムの研究開発に従事.

本庄　利守（正会員）
Email: honjo.toshimori@lab.ntt.co.jp
NTT ソフトウェアイノベーションセンタ 分散処理基盤技術プロジェクト 主任研究員. 1998 年東京工業大学大学院 情報理工学研究科修了．同年，日本電信電話(株)入社．並列分散処理，量子情報の研究開発に従事．博士(工学).

投稿受付：2012 年 06 月 04 日
採録決定：2012 年 11 月 30 日
編集担当：西　直樹（日本電気）

© 2013 Information Processing Society of Japan