メインフレームの特徴を生かした
IT 全般統制（ITGC）構築のポイント

株式会社アイビスインターナショナル

■ 執筆者Ｐｒｏｆｉｌｅ ■

有賀　光浩

1985 年　富士通株式会社入社
2003 年　富士通株式会社退職
2004 年　株式会社アイビスインターナショナル
　　設立　代表取締役

■ 論文要旨 ■

多くのメインフレームは長年，性善説に立ったシステム開発，運用管理を行っている．具体的には，本番環境と開発環境が同一システムに混在し，ID 管理を行っていないケースも非常に多い．2006 年末から，メインフレームで内部統制を構築するための問題点を整理し，富士通をはじめ各部門の方々とも意見交換や情報共有を行ってきた．

内部統制の当事者のほとんどはメインフレームを知らない現実の中で，「メインフレームの特徴を生かした IT 全般統制（ITGC）」を実現するには，システムの稼動状況の見える化が有効である．これによりリスクを網羅的に検出するとともに，権限を持ったシステム管理者の透明性を高めることが狙いである．

事例を通して見える化の手法を紹介するとともに，ITGC 構築のポイントを考察する．統制はすべてを自動化（IT 化）する必要はなく，本来の目的を理解し，人間系も使って対応していくことが重要である．

■ 論文目次 ■

■ 図表一覧 ■

１．はじめに

１．１　当社の概要

株式会社アイビスインターナショナル（所在地：東京都）は，富士通メインフレーム（GS21，PRIMEFORCE）の性能コンサルティングを事業としている（ホームページ：http://www.ibisinc.co.jp/）．

代表の有賀光浩(Ariga Mitsuhiro)は富士通株式会社で 18 年間 SE として活躍，1992 年からの 11 年間は共通技術部門でメインフレームの性能に関する技術支援，顧客システムの性能トラブル対応を担当した．対応システム数は国内外合わせて 1,000 以上に及ぶ．

2004 年に株式会社アイビスインターナショナルを設立し，お客様がメインフレームをより快適に，適切なコストで使って頂くためのコンサルティングを行っている．

１．２　背景

（１）メインフレームの国内市場

(社)電子情報技術産業協会(JEITA)の調査によると，2006 年度のメインフレームの出荷台数は 872 台（前年度比 92%），金額は 1,801 億円（前年度比 93%）であった．金額ベースでは UNIX サーバが 2,800 億円（前年度比 87%），IA サーバが 3,088 億円（前年度比 103%）である．富士通の発表では，現在でも 3,000 以上のお客様で使われている．

（２）金融商品取引法における内部統制

金融商品取引法が 2006 年 6 月 7 日に成立し，内部統制報告制度が 2008 年 4 月 1 日以後開始する事業年度から適用されることになった（以降，日本版 SOX と表す）．図１－右上に示す通り，内部統制には４つの目的があるが，日本版 SOX では，財務報告の信頼性確保を唯一の目的としている．約 3,900 の上場企業とその関連会社が対象となるが，富士通のメインフレームが対象となるお客様は数 100 になると予想している．

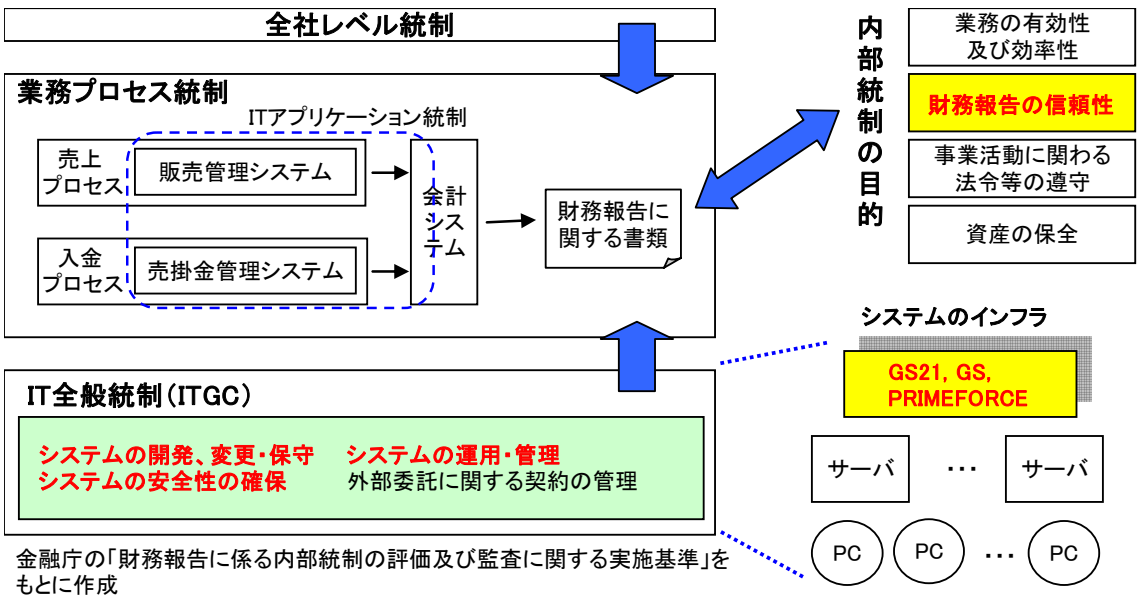

図１　内部統制の概要図

（３）問題意識

2006 年末から内部統制の調査を進めていくと，IT 全般統制（以下 ITGC と略す）おいて以下の問題が見えてきた．

①　システム開発・運用管理の基本は性善説である

私の知る限り，多くの富士通メインフレームは日本流性善説が前提である．アクセス管理を提供する RACF という製品はほとんど使われていない．また，RACF の機能や支援体制も IBM に比べると明らかに劣っていることがわかった．

②　RACF は内部統制のためのツールでない

あるリスクを統制するために RACF を採用することはあるが，RACF の導入は ITGC を実現することの必要条件でも十分条件でもない．RACF でできることには限界があり，議論は内部統制と関係のない細かなセキュリティ管理の話しに飛びやすい．

③　当事者はメインフレームを知らない

内部統制に関わっている人（お客様，監査法人，コンサル会社，メーカ）のほとんどはメインフレームを知らない．逆にメインフレームをわかっている人は，内部統制を理解するまで手がまわらない．

ITGC 対応は早急の課題を抱えており，メインフレームの技術者が智恵を出し合う必要性を感じ，2006 年 12 月から図２に示すような活動を行った．

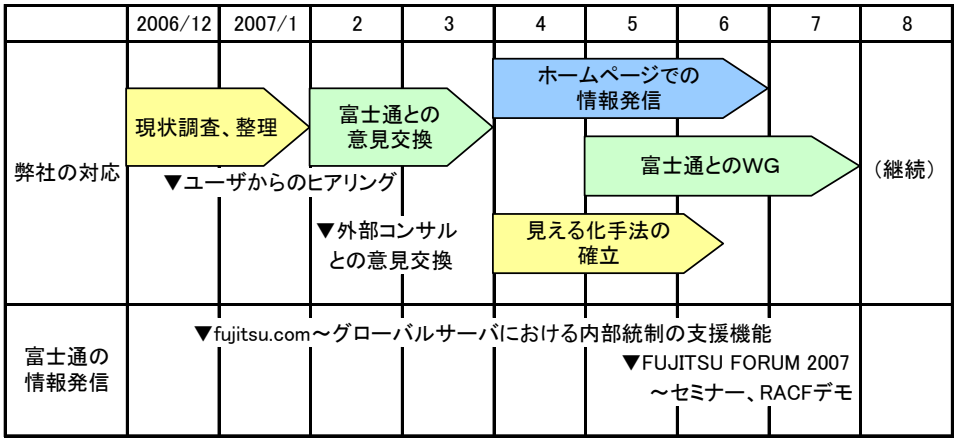

図2　活動実績

富士通の４つの事業部，２つの SE 会社の幹部社員と以下の点で意見交換を行った．

・GS での内部統制（当面は日本版 SOX）の動き，現状の取組みについて
・RACF のサポート体制について　～　デモ，提案ツール，人材育成（研修）
・情報共有，クロスファンクショナルな対応について
・日本版 SOX 対応には RACF の導入が必要か
⇒　稼動ログを使って GS の現状を可視化し，ミニマム・スタンダードのセキュリティシステムを短期間に構築する方法が必要ではないか

この結果を踏まえ，５月からは実務的な関係者と WG を立上げ，情報共有や意見交換を行っている．弊社独自の調査結果等はホームページで随時公開している．

１．３　論点

オープン系システムで開発された業務パッケージをメインフレームに移植したところ，品質，性能，保守性の悪いものができあがった例がある．原因は，メインフレームの常識やルールを守っていないためである．同様に，オープン系システムで実現させた ITGC の仕組みをそのままメインフレームで実現しようとすると，運用に耐えられないものになる可能性がある．

本論文ではメインフレームの特徴を生かした ITGC の構築を目指し，稼動状況の見える化と構築のポイントを紹介する．本論文は内部統制攻略書ではないので，内部統制の基本的な説明は極力省略した．内容の確認には参考資料を参照して頂きたい．OS や製品に関する情報は，2007 年 7 月末にユーザ公開されているものを採用した．

２．ITGC から見たメインフレームの現状

２．１　セキュリティレス・統制レスのシステム

金融庁の実施基準では，IT に係る全般統制の具体例として図１－左下に示した４つが例示されている．その中でメインフレームにおける代表的な問題点を表１に示す．多くのメインフレームは，セキュリティレス・統制レスで運用されているが，ほとんどのシステムでは 10 年以上大きな問題もなく（問題に気づかず）運用し続けている．

表1　ITGC におけるメインフレームの問題点

ITGC の観点	代 表 的 な 問 題 点	コ　メ　ン　ト
システムの開発，変更・保守	①アプリケーションの変更管理が弱い，ドキュメントがメンテナンスされていない	「変更管理」や「ライブラリ管理」の習慣がなく，ツールも整備されていない
	②本番環境と開発環境が分離されていない	開発機や開発用 VM（仮想計算機）を持っているお客様は少ない
システムの運用・管理	③システムの運用状況を網羅的に把握していない，管理規約がない	一日に数千のジョブが実行され，運用状況を網羅して把握することはむずかしい
	④誰でも端末からジョブを実行できる，その実行状況を把握していない	ジョブの実行（SUBMIT）が自由にできるシステムが多い
システムの安全性の確保	⑤OS 標準の ID 管理が弱い，すべて特権ユーザである	ID（TSS/AIF 用）の棚卸をしない，パスワードが無いことも珍しくない
	⑥ログが取得されておらず事故の原因が究明できない，事故を把握していない	SMF やシスログを捨てているケースも多い
	⑦RACF を導入していても職務分掌が不十分である	セキュリティ管理が目的であり，内部統制が目的ではない

注）外部委託に関する契約の管理は，メインフレームに依存しない問題のため割愛

２．２　把握しきれないシステムの運用状況

内部統制では，業務プロセスを文書化しリスクを洗い出し RCM(Risk Control Matrix)を作成するが，この方法でリスクを網羅的に抽出できるのか疑問を持っている．多くのメインフレームでは，一つのシステムに本番環境と開発環境が同居し（②），誰でも好きなジョブ名でジョブを実行でき（④），稼動ログの分析を行っていない（⑥）．

ESPⅢ（2000 年頃まで使われていた OS）移行ユーザは，独自の CL(Control Language)を使って業務を実行するため，運用状況を把握することは非常に困難である．更に，データベースを直接更新できる優れたユーティリティが標準提供されており，ユーザは業務とし

てこの機能を使っている．この手のツールはどのユーザでも大変重宝している．

オンライン業務は，複数のプログラムがマルチタスク構造のジョブ（システム ACP という）として動作しており，稼動状況を把握するにはシステムごとに工夫が必要である．

性能の悪いプログラムを分析していると，仕様書上使っていないデータベースをアクセスしているケースがある．これは PG 工程のバグであり，ユーザの受け入れテストで検出することは無理である．

業務プログラムが格納されているロードモジュールライブラリのメンバはいつ更新されたのかわからない．

即ち，システムは今，A)知らないプログラムが実行されていて，B)データの変更も手軽にでき，C)プログラムが仕様と違った動きをしていても，D)ロードモジュールをちょっと変更しても全く認知できない可能性さえある．

システム管理者であっても，システムの運用状況を網羅的に把握していることは稀である．現在取得している稼動ログを使ってシステムの運用状況を見える化し，リスクの洗い出しをすることはシステムの運用・管理に必要不可欠であると考える．

２．３　RACF に対する誤解

RACF は役に立つ製品ではあるが「内部統制のためのツールでない」と言い続けている．理由は以下の通りであり，有識者とも共通した認識である．RACF の機能を理解せずに導入を推奨したり，過大な期待をかけるケースが見受けられるので注意が必要である．

- 富士通の RACF のサポート範囲は図３に示す通りシステムの一部である．
  - ・基本的には，バッチジョブ，TSS/AIF を対象にした機能である．
  - ・業務プロセスでの職務分掌を RACF の機能だけで実現することはかなりむずかしい．
  - ・RACF のログ＝監査証跡ではなく，監査証跡の一つである．
- 日本版 SOX 対応とセキュリティ管理はまったく違う．
  - ・前者の目的は財務報告の信頼性のためであり，例えば個人情報が漏洩しても，財務情報が見られても直接的には問題ない．
  - ・RACF 導入済ユーザでも，内部統制視点でのアクセス管理が不十分である例がある．
- OS の脆弱性をカバーするものではない．導入には制限事項も多い．

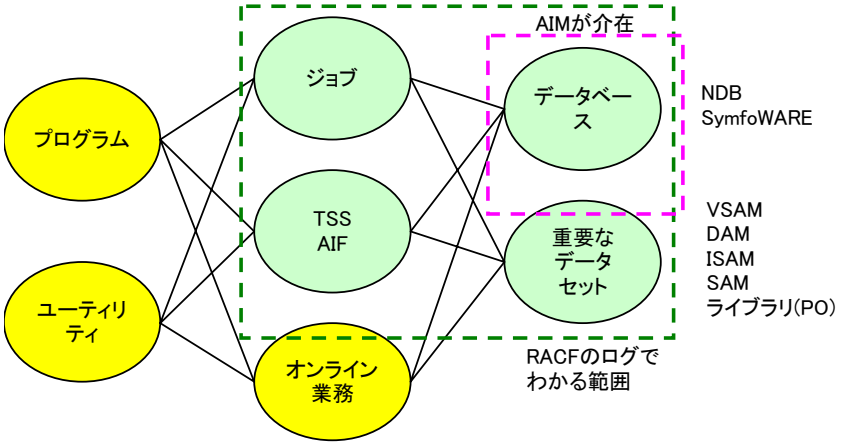

図3　RACF のサポート範囲

３．システムの稼動状況の見える化（事例）

システム標準の稼動ログ（SMF 等）を使い，システムの稼動状況の見える化と検出された問題やリスクの事例を紹介する．見える化することにより，システムの管理規約や整備された内部統制と実態とのギャップを正確かつ網羅的に確認することができる．

３．１　ジョブの稼動状況

図４は，一日に数千も実行されるジョブの稼動状況に，データベースをアクセスしているものを一目でわかるように色分けしたグラフである．

ITGC の視点で見ると，DB 未使用と参照のみのジョブのリスクは小さい．認識していないジョブがデータベースを更新していれば問題である．

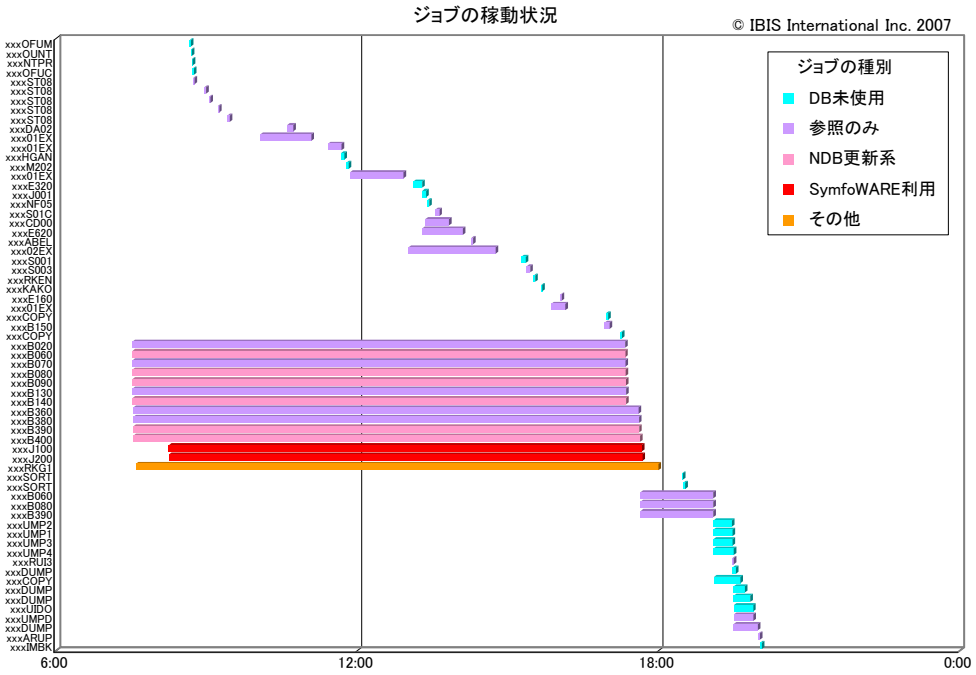

図4　ジョブの稼動状況　（縦軸:ジョブ名）

【検出された問題やリスク】

・データベースを更新しないはずのジョブが更新系になっている（プログラム管理）

⇒　プログラムとデータベースを特定し，仕様書とプログラムのどちらが正しいのか確認する

・管理者，内容がわからないジョブが動いている（ジョブ管理，オペレーション管理）

⇒　ジョブの調査，ジョブ実行のルールを確認する

・AIM/VSAM を利用している（セキュリティ管理）

⇒　内部統制に影響するデータか確認する

・複数の業務の本番系と開発系が混在している（構成管理，変更管理等）

⇒　内部統制の整備状況，運用状況を確認する

３．２　TSS/AIF の稼動状況

図５は TSS/AIF の稼動状況で，フォアグラウンド（FG）でプログラムを実行していると思われるものを一目でわかるように色分けしたグラフである．想定している処理は，業務アプリケーション（特に ESPⅢ移行ユーザ），簡易データベース保守機能（EXCIDM，DBSP 等），その他（コンパイル等）である．データベースをアクセスしている処理については，図６でプログラム名と処理開始時刻を把握することができる．

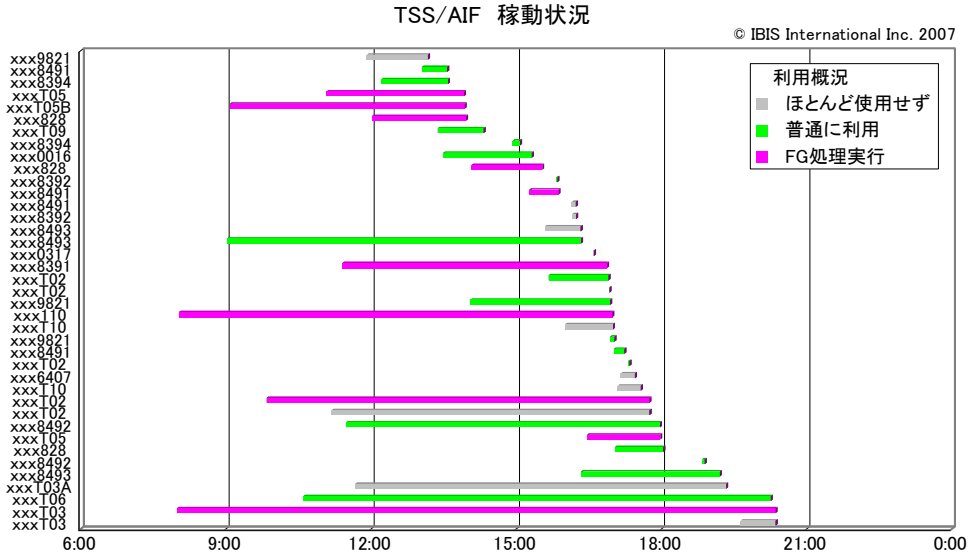

図5　TSS/AIF 稼動状況　（縦軸:ユーザ ID）

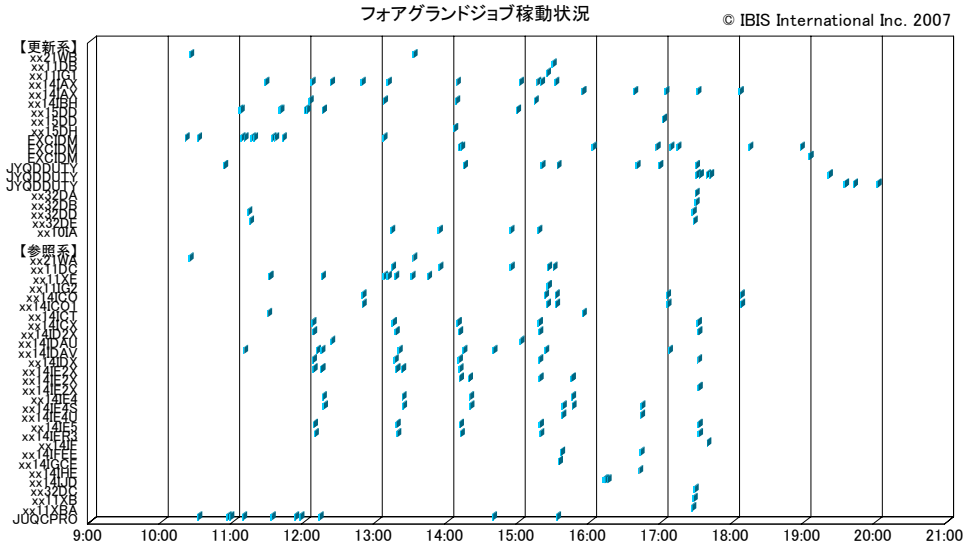

図6　フォアグラウンドジョブ稼動状況　（縦軸:プログラム名）

【検出された問題やリスク】

・利用者や用途がかわからない ID が存在（ID 管理）⇒　詳細を調査する
・フォアグラウンドでプログラムを実行（オペレーション管理）⇒　ルールを確認する
・データベース保守機能を使用（セキュリティ管理）⇒　実態を調査する

３．３　オンライン業務の稼動状況

表２はオンライン業務の稼動状況の一例である．SMQN(Secondary Message Queue Node)がプログラム名に相当する．オンラインジョブは通常マルチタスク構造になっており，一つのジョブ配下で複数のプログラムが動作するため，基本的な稼動ログでは，プログラムごとの稼動状況を把握することができない．内部統制の対象範囲であれば，オンライン環境を調査し，システムごとに最適な方法を考えていく必要がある．これは RACF を導入しても同様である．

OS や AIM（オンライン環境，DB 環境等を支えるミドルウェア）の知識なしにオンライン業務に統制をかけることはむずかしく，統制ありきで対応すると運用できないシステムを作り上げる可能性がある．

表2　オンライン業務の稼動状況(トランザクション件数)

No.	JOB	MQN	SMQN	6月29日	6月30日	7月1日	7月3日	7月4日	7月5日	総計	参照/更新	データベース
2	xxxSB010	xxxRB010	xxxRB010	6,928	8,873	515	7,079	6,373	6,274	36,042	更新	NDB
3	xxxSB020	xxxRB020	xxxRB020	2,675	4,383	344	2,688	2,561	2,922	15,573	参照	NDB
4	xxxSB030	xxxRB030	xxxRB030	2,809	3,696	165	2,828	2,931	2,645	15,074	参照	NDB
5	xxxSB040	xxxRB040	xxxRB040	399	1,088	0	217	159	151	2,014	参照	NDB
6	xxxSB050	xxxRB050	xxxRB050	998	1,215	0	1,105	575	804	4,697	参照	NDB
7	xxxSB060	xxxRB060	xxxRB060	786	1,112	7	1,214	964	1,698	5,781	更新	NDB
8	xxxSB070	xxxRB070	xxxRB070	2,358	5,633	99	3,104	3,607	3,654	18,455	参照	NDB
9	xxxSB080	xxxRB080	xxxRB080	65	200	0	155	101	248	769	更新	NDB
10	xxxSB090	xxxRB090	xxxRB090	1,421	1,898	170	907	1,108	1,588	7,092	参照	NDB
152	xxx1DC10	xxxRMQ10	xxxRSM10	664	485	30	892	520	638	3,229	更新	SymfoWARE
153	xxx1DC11	xxxRMQ11	xxxRSM11	3,064	2,394	115	3,624	2,268	2,987	14,452	更新	SymfoWARE
154	xxx1DC12	xxxRMQ12	xxxRSM12	167	133	11	236	137	162	846	更新	SymfoWARE
155	xxxM41	xxxRMQ41	xxxRSM41	5,497	4,736	382	8,098	4,003	5,687	28,403	更新	SymfoWARE

【検出された問題やリスク】

・トランザクションが発生していない SMQN がある　⇒　現在使われているか確認する
・SMQN 名では業務プログラムを把握できない　⇒　プログラムの構造設計を確認する
・一日だけ動作した SMQN がある　⇒　内容，目的，起動時間等を確認する
・稼動状況が見えない　⇒　詳細データの採取，分析を検討する

３．４　プログラム，ユーティリティの使用状況

３．１のジョブはプログラム単位に掘下げることが可能である．

実行されたプログラムに着目することで，発見的統制をかけることができる．RACF のログではプログラム名を認識できないが，稼動ログを使うと見える化することができる．

【検出された問題やリスク】

・コンパイル（COBOL）の実行状況　⇒　本番・開発の分離ができているか確認する
・データベースユーティリティの実行状況　⇒　セキュリティの確保を確認する
・データセットユーティリティの実行状況　⇒　内容と目的を確認する
・AIM 環境定義（DDMS）の実行状況　⇒　内容と目的を確認する
・異常終了の状況　⇒　本番環境でプログラムが異常終了する根本原因は調査する

３．５　データベース，データセットのアクセス状況

データベースについてはどのジョブ（プログラム）がアクセスしているか把握することができる．即ち，想定外のジョブがデータベースをアクセスしていることを感知することができる．表３は NDB のアクセス状況だが，SymfoWARE についても同様の把握が可能である．AIM/VSAM は AIM 配下外でもアクセスできることがリスクであり，見える化の方法は個別に検討する必要がある．

メインフレームの強みは，AIM と DBMS が様々な統制をかけていることである．統制の強さは，NDB ＞ SymfoWARE ＞ AIM/VSAM ＞ VSAM,ISAM となる．

表3　NDB のアクセス状況

スキーマ名→	xxxHK11		xxxHK21	xxxHK22			xxxHK23
レコードタイプ名→	xxxK501	xxxK502	xxxK001	xxxK005	xxxK006	xxxK007	xxxK003
↓ジョブ名							
xxxDN7			1,026,775				0
xxxDO78			400,430				0
xxxK0070			138,442				0
xxxK02				126	0	0	
xxxK20			329,681				2,361
xxxK21			27				23
xxxK24			85,809				28,309
xxxF09			145				0
xxxSB010	0	0	2,931	0	0	0	2,789

より詳細なデータを採取することで，表４のようにデータセットにどのジョブがアクセスしているか認識することができるが，OS により機能差があるので注意が必要である．

表4　データセットのアクセス状況

		ジョブごとのI/O回数			
データセット名	種別	xxxxIBX	xxxxICX	xxxxILX	xxxxWIT
xxx.xxNDS.KANR.DT	VSAM	366	293	2698	
xxx.xxNDS.KANR.IX	VSAM		4	16	
xxx.xxNDS.SPOOL.DT	VSAM	337		175	
xxx.xxNDS.SPOOL.IX	VSAM	1		69	
xxx.xxF00081.UJFO	NDB	93			287
xxx.xxF00082.UJFO	NDB	93			260
xxx.xxF00083.UJFO	NDB	372			273
xxx.xx1.LOAD	LOAD				100

【検出された問題やリスク】

・知らないジョブがデータベースをアクセスしている　⇒　仕様，環境定義を確認する
・データベースのアクセス回数が異常に多い　⇒　原因を調査する
・VSAM を使用している　⇒　データセットのアクセス状況の分析を検討する

システムの稼動状況を見える化する事例を紹介したが，これらは一度やって終わりというものではない．情報システム部門が問題意識を持ち，改善を進めていく PDCA サイクルを自主的に回していけることがポイントとなる．

４．ITGC 構築のポイント

セキュリティレス・統制レスでよかったシステム（２．１）にも，ITGC を構築しなければならない現実が迫ってきてた．統制はすべてを自動化（IT 化）する必要はなく，リスクが軽減できるなら手動でも問題ないが，現実的に運用が回らなければ意味がない．

４．１　システムの開発・変更，保守

（１）変更管理

メインフレームでは，手作りの（またはカスタマイズされた）業務プログラムが動いている．そのため，変更管理プロセスを整備・運用する必要がある．オープン系では変更管理システム（ツール）が充実しているが，メインフレームには対応していないので，基本的に人間系重視（手動）で行わなければならないのが現状である．監査では「本番プログラム登録時にシステムが自動生成した証跡」を要求されることが予想され，システムの状況に応じた工夫が必ず必要となる．

（２）本番環境と開発環境の分離

変更管理プロセスを整備した上で，本番環境と開発環境を分離する統制を考える．業務プログラムをわかっている人の職務分掌が目的である．１システムで運用しているお客様は，手動による統制の手間とコストを考え，表５の A～D の形態を選ぶことになる．

メインフレームの担当は少人数で仕事も兼務しているため，現実的な統制方法は監査人と十分に詰めていく必要がある．

表５　本番環境と開発環境の分離パターン

	実機数	仮想システム	RACF	コ メ ン ト
A	1	なし	なし	・開発・運用者が少人数のとき，人間系で統制する ・本番業務終了後，開発業務を行う
B	1	あり	なし	・本番と開発が分離するので一般的にわかりやすい形態 統制なしに両系をアクセスできたら意味がない
C	1	なし	あり	・RACF を使って職務分掌の設計が必要
D	1	あり	あり	・複数の部門で大勢の人がプログラムの開発，変更を行っているとき

仮想システム ･･･ AVM/EX を使った仮想化

４．２　システムの運用・管理

（１）運用状況の把握

運用管理ツールを導入しなくても，３章で紹介したようにシステム標準の稼動ログを使って見える化し，運用状況を把握することができる．このとき実施した簡単にできる工夫を以下に紹介する．

・SUBMIT ジョブの命名規約の見直し（例．ジョブ名＝ユーザ ID＋α とする）
・重要な業務については，異なる処理を同じジョブ名で実行しない．
・データベースユーティリティ，データセットユーティリティ，AIM の環境定義などの実行結果リストは保管する．

（２）ジョブの実行管理

ジョブを実行するプロセスを整備・運用することだが，本番環境と開発環境が同居して

いるシステム（表５の A）では複数の統制が必要となる．稼動ログを使って発見的統制をかける工夫も必要である．

３．２で示した TSS/AIF のフォアグラウンド処理（特に，簡易データベース保守機能）の運用は最も慎重に検討すべきである．

業務やメインフレームの知識を持っている特定の人しかジョブは実行できない．

４．３　システムの安全性の確保

（１）ID 管理

TSS/AIF の ID を登録・削除するプロセスを整備・運用する．業務プログラムの ID 管理は IT アプリケーション統制の範囲とするのが一般的であるためここでは除外する．

最も重要なことは，OS の標準機能では，TSS/AIF の ID はすべてが特権ユーザで，簡易的な ID/パスワード管理しか提供されていない事実を正しく理解することである．

セキュリティ管理の一環で ID 管理を考えると，RACF の機能を使うしか考えられない．ITGC として ID 管理をどこまで厳密に行うかを，システムの状況に応じて考える必要がある．

業務を熟知し意思がなければ，プログラムもデータも簡単には変更できない．

（２）監査証跡

コンピュータ上のログだけでなく，ドキュメントも大切な監査証跡である．

コンピュータ上のログで重要なのは，AIM や DBMS と連携しているシステム標準の稼動ログであり，RACF のログがこの機能を包含する訳ではない．ログは採取しただけでは何の意味もなく，意味のあるログを取得するためには以下のアプローチが必要である．

①目的を明確にし，取得するログデータを決定
②抜けなく継続してログデータを取得する仕組み
③ログデータを改ざんされない仕組み
④ログデータを蓄積する仕組み
⑤蓄積したログデータから必要な情報を早く正確に取り出す仕組み

①～③前半まではメインフレームに閉じた話しであり，③後半～⑤はオープン系ではツールが充実しているがメインフレーム対応はまだまだである．

いずれも ISV との連携を強化し，早期対応が必要である．

（３）アクセス管理

システムでアクセス管理を行うには，RACF が必須であると考える．OS 標準では簡易的な ID/パスワード管理しか提供されていないことが簡潔な理由である．

ITGC 上守るべきものは，財務報告に関係するプログラムとデータ（データベース）である．データには AIM や DBMS がある程度統制をかけており，弱点はプログラム（ロードモジュール）である．

ライブラリやソースプログラムの情報を知っているのはごく一部の人だけである．

４．４　提言

多くのお客様ではメインフレームはごく一部の人しかわからず，触ることもできないのが現状である．ITGC は情報システム部門が中心となって整備し，自らがその下で運用する立場となる．内部統制に何か問題があったとき，最初に槍玉にあがるのは情報システム部

門である.

メインフレームでプログラムやデータの改ざんを行えるのは，残念だが内部事情を知っている人しか考えにくい.

重要なことは，システムをできるだけ透明にし，権限を持ったシステム管理者自らが率先して証跡を残しておくことである．そうすれば問題が起きても自信をもって原因調査に取り組み，説明責任を果たしていくことができる.

５．評価と課題

メインフレームの特徴を生かしたITGCの基本は，システムの見える化を徹底しリスクを感知することだと考えている.

３章で紹介したシステムの稼動状況の見える化は数社で適用しているが，ITGC の一部を切り出して評価できるものでなく，効果が見えてくるのはこれからである．お客様をはじめ関係者の意見を取り入れブラシュアップしていきたい.

また，メインフレーム上でのITGC構築に関する情報を全国で唯一ホームページを使って発信している．最近では，IT ベンダーだけでなく，お客様や監査法人からのアクセスも急増している.

今後の課題は次の５点である.

（１）メインフレームの強みを生かしたITGC構築手法の確立
　お客様，監査法人の意見を取り入れブラシュアップしていくこと

（２）メーカと連携したITGC構築支援の推進
　お客様の日本版SOX対応で，メインフレームがボトルネックにならないこと

（３）内部統制対応によるシステムの性能・信頼性低下の未然防止
　RACF導入やログ採取で，性能トラブルを誘発しないこと

（４）ツールの整備，強化
　できることできないことの整理，ISVとの連携強化

（５）RACFサポート力の強化
　ノウハウの集約，サポートの透明化，ユーザ向け研修の立上げ

今後も論文やホームページを通して事例やノウハウを発信していきたい.

以上

参考文献

［１］金融庁：“財務報告に係る内部統制の評価及び監査に関する実施基準”