性能面からみたクライアント／サーバシステムの構築

3Ｌ－6

－快適なシステム稼働のために～「７６の鉄則」－

横田　康敏
(株)第一勧業銀行

１．はじめに

クライアント／サーバシステム（以下ＣＳＳ）はすでに一般的なシステム形態である。ＣＳＳは数年前に比べ適用範囲も広がり、基幹システムさえＣＳＳ化している企業も決して珍しくなくなってきている。しかし、一方で未だに「性能が出ない」、「遅くて使い物にならない」という声をよく耳にする。そこでＬＳ研（(株)富士通主催 Leading-edge System Users Group）で開発現場で役に立つ実践的なノウハウの活用を研究し、「快適なシステム稼動のために～７６の鉄則」を作成した。本稿ではその概要を報告する。

２．ＣＳＳ構築時の問題点

業務システムを実際に開発している現場では次のような問題点があげられる。

(1) 情報・ノウハウがいろいろな場所に分散しており、設計者に情報が不足している。
(2) ベンダからの無償サポートは期待できない。
(製品価格が安いためサポートは有償で、しかも製品価格に比して非常に高価である。)
(3) 製品のライフサイクルが短いので、開発担当者の経験の蓄積が生かせない。
(4) ＤＢや言語に“はやりすたり”がある。
(5) ソースコードが公開されていない部分が多く、ブラックボックス化されている。
(6) トラブルの予測が極めて困難。
(7) 組み合わせが柔軟なだけに価値基準が明確でない。
(8) バグ等のトラブル情報が全てのユーザーに必ず届くわけではない。
(9) 導入、開発にあたりメーカの話を鵜呑みにすると痛い目にあう。

また、４ＧＬ等を使用すると基本的な約束事を知らなくてもシステムを構築出来てしまう。正しい結果は返ってくるが、全く性能がでていないという現象に陥りやすい。つまり、つきつめたチューニング方法などよりも、もっと基本的な部分で注意しなければならない事柄が大切である（できてしまったものをいくらチューニングしても限度がある）といえる。さらに、回避できるトラブルの防止策がまだ知られていないため同じようなトラブルに遭遇してしまうケースがＣＳＳの場合非常に多い。これを未然に防止するということが重要である。そこで手軽に参照できて、ハードやソフトが変わっても使える「快適なシステム稼働のための鉄則集」を作成した。

３．鉄則集

鉄則集をつくるに当たって「ＣＳＳを成功に導くための定石・ノウハウの共有」、「他人の失敗から学び、

---

76 Rules For Developing An Efficient Client Server System

Yasutoshi Yokota
The Dai-ichi Kangyo Bank,Limited
1-5 Uchisaiwaicho 1-chome Chiyoda-ku,Tokyo

同じ失敗を繰り返さない」、「今、ＣＳＳの現場で起きている問題が、担当ごと・開発フェーズごとに一覧できる」といった点をポイントとした。鉄則とする条件は過去にメンバーが経験している、失敗事例を知っている、特定のハード／ソフト／ミドルウエアに依存しないなどである。これを、鉄則の説明・事例・対策という項目でフォーマット化した。目指したものは巷に出回っている体系だった内容の書籍とはひと味違った、現場の本音が反映された形の「鉄則集」である。赤裸々な失敗事例と現場の本音を集めたのが目玉である。ハード・ソフト・ミドルウエアを特定せずに永続性のあるものについて特定のカテゴリを決めずに洗い出しを行ったため、さまざまな観点・立場からの鉄則があげられ、ＣＳＳにおけるポイントを浮き彫りにすることができた。鉄則の総数は７６個である。以下が鉄則である。

「快適なシステム稼動のために～７６の鉄則」

NO	鉄　　則	NO	鉄　　則
1	業務改善ができないＣＳＳは成功しない	39	オンライン系でのORDER BY句はデンジャラス
2	業務面の見直し・改善を伴ってこそ	40	ＪＯＩＮによるＲＤＢへの負担も考慮せよ
3	ＣＳＳ導入は、経営面での目的を明確に	41	ロールバックセグメント領域だってパンクする
4	コストダウンをだけを目的としたら失敗する	42	クライアントとサーバの比重は柔軟に
5	隠れたコストを事前に把握せよ	43	ＳＱＬ文生成ツールは要注意
6	ユーザーに近づかなければＣＳＳではない	44	プロとタイプは回数限定
7	ＣＳＳ費用は導入時よりも維持管理にかかる	45	機能検証はモデルである
8	時流に流されてのＣＳＳ導入はダメ	46	原因追求はほどほどに
9	“作る”部分と“作らない”部分を見極める	47	ＯＳ／ＲＤＢにだってバグがある
10	分散システムでは、データの同期化がポイント	48	ディスクへのアクセス回数は極力減らす
11	印刷物は極力減らす	49	パッケージはカスタマイズしない方がよい
12	導入は前回の経験だけでは不充分	50	空値の列は発生させないように
13	情報収集は自分でやらなきゃ誰がやる	51	３２ビットと１６ビットを混ぜない
14	クラウアントの導入はまとめて！	52	ＣＰＵの最大利用率は１００％ではない
15	サーバーは過去実績のあるものを	53	開発ツールは万能ではない
16	マルチベンダは力量にあわせて	54	ＣＳＳの外部発注は、丸投げ禁物
17	CSSを構成する機器は、追加可能な機器にする	55	退屈させるな
18	開発ツールとシステムには相性がある	56	４ＧＬ、慣れない高機能に浮気しない
19	ツール選択は作る人の実力にあわせて	57	インデックスは本当に使われていますか？
20	開発ツールはＶＢのみならず	58	レスポンスが悪いときには自動生成SQLを疑え
21	プロジェクトメンバは必要最小限に！	59	ＳＱＬ文の条件式で思わぬ性能低下
22	早くからプログラマを参画させろ	60	４ＧＬのコンパイルはデバック後
23	金は最初にかけろ！	61	代替案、回避策の検討を常に行う
24	基幹業務、問答無用で標準化	62	ユーザにテストさせなきゃ動かない
25	節目節目に定量的な検証を行なうべし	63	テストデータは運用時にあわせて
26	ユーザの役割を明確にする	64	チューニング自らやって一人前
27	エンドユーザと意志疎通せよ	65	事前検証は“予測”ではなく“実証”のために
28	自分の言葉の意味は１００％相手に伝わらない	66	自動生成されたＳＱＬ文は必ずチェック
29	一貫したシステム・デザインを重視せよ	67	レスポンスを要求するならメモリを要チェック
30	マウス操作にこだわる必要はない	68	何でもかんでもﾊｰﾄﾞｳｴｱが悪いわけじゃない
31	データ設計は「トリガ」の作成まで	69	できる限りシステム運用はユーザに任せる
32	図を中心にしたデータベース設計を行なう	70	ＥＵ教育を軽視するな！
33	テーブルの正規化を正しく行う	71	新事業の付加価値はエンドユーザに明確に！
34	状況によっては非正規化する	72	自動化できるものは自動化を
35	インデックスを貼ればええってもんとちゃう	73	ﾊﾞｰｼﾞｮﾝｱｯﾌﾟ後は動かなくなるかもしれない
36	インデックスは一表につき８個まで	74	ウイルスチェックは全クライアントに徹底する
37	インデックスについての正しい理解を	75	ディスク障害時の対応はシナリオ通りに
38	Ｒｅａｄ、Ｗｒｉｔｅから集合操作へ	76	真に必要な機能でまず稼動

４．終わりに

ＣＳＳ構築の鉄則は上記だけで終わりというわけではない。我々が用いた分類方法を利用して技術進歩にあわせて鉄則をさらに追加していくことが必要である。

ＣＳＳ開発は、オープンシステムという性質上、「新技術の発展」や「ソフト・ハードの組み合わせが自由」など無限の可能性を秘めている。「このやり方しかない」というような決定的なものはない。業務にあわせて、自分で情報を蓄積し自分なりの鉄則を導き出し、蓄積していくということが必要である。ＣＳＳ開発を成功させるためには、情報の共有・継承（鉄則集の作成：定石、ノウハウの共有）を行い、同じ失敗を繰り返さないことが重要なポイントである。