5V-4

オペレーティングシステムの性能評価と監視の効率化

吉本安男　高橋雅宏

(富士通株式会社)

1　はじめに

ソフトフェアにとって性能は重要な要素の一つである．いかに優れた機能を備えても性能が悪ければ顧客に受け入れられないし，一たび顧客先で性能トラブルを起こせば製品の信頼を著しく損ない，それを挽回することは容易ではない．

しかし，ソフトウェアの性能と機能はしばしば互いにトレードオフとなり，性能の良し悪しは評価の難しさもあってソフト開発上二次的な問題として扱われることもあった．ここで述べる性能評価と監視の方式は，オペレーティングシステムの性能をソフト開発作業に定着させることを主眼にしたシステムテストの一方式である．

2　性能監視の必要性

製品のバージョンアップやレベルアップ時には，提供に先立って性能検証が十分に行われる．しかし，若干の機能追加や障害修正のためのプログラム修正（以後 PTFという）は，開発期間が少ないこともあって性能保証に対するチェックが机上レベルでの確認に留まることがある．一般に性能を測定するためにはハードウェアやオペレーティングシステムの知識が必要であり，数々の性能分析あるいは評価用のツールを操作しなければならない．こうした知識作業に加えて測定環境の設定が複雑になる場合もあり性能変化に対するチェックは難しい．具体的な性能変化の問題として仮想記憶容量の例を示す．

"OSIV/X8 FSPの仮想記憶の大きさは16MBである．PTFの適用毎に機能追加や障害修正のために LPA(注)が増加すればユーザプログラムが使用できるリージョンを圧迫し，プログラムがリージョン不足のために実行できなくなるという問題が発生する．"

このようにＰＴＦの修正内容を考慮しなければ知らない間に性能問題が潜在化し顕在化する恐れがある．したがって，性能を保証していくために，ＰＴＦによる性能変化を把握することが必要である．しかし，ＰＴＦに対する性能要件をすべて検証することは，一般の機能テスト以上に困難である．性能劣化を防止する基本は，ソフトウェア開発における全工程で性能を考慮した作業をすることであり，本性能監視もそのような認識の定着と日常作業への反映が目的である．

(注) LPAとは仮想空間内のオペレーティングシステムが占めるプログラム領域のことである．

3　性能監視の方針

我々が目指す性能監視は先に述べたように，監視によって性能を保証することではない．検証方法も測定値そのものを絶対的に評価することでもない．性能データが数量化(LPA使用量，走行ステップ数等)できることに着目し，前回のＰＴＦとの比較による変化の理由を開発担当者とともに考え（必要ならば改善を検討）次の開発作業にフィードバックすることが目的である．また，同時に性能測定作業の問題点や改善点もみつけていこうとしている．方針を次に示す．

- －　プログラム修正と性能の確認は一体であるとの認識の定着化．
- －　システム全体の測定をシステム統合時に一括して実施することにより作業の効率化を図り，同時に個々の修正のシステム全体への影響を確認する．
- －　ＰＴＦ毎に性能データを蓄積し，分析することにより今後の性能を予測できるようにする．

4　性能監視の方法

4.1　監視作業の標準化

ＰＴＦは約２ヶ月に一度のスケジュールで作成されるので，その度に監視するとなれば計算機時間と作業工数からみて作業の省力化，標準化を図らなければならない．さらに性能測定結果は，以前の性能値

と容易に比較できる必要がある．そのためには次の要件を満たさなければならない．

- 性能測定の結果は，コンポネント(注)別に前回（それ以前）の測定値と分類，比較できること．
- 性能測定ツールは，その他複数のツールと同時に使用でき，一度に複数の性能項目を測定できること．
- 性能測定環境は，どの開発担当部門でも簡単に構築できること

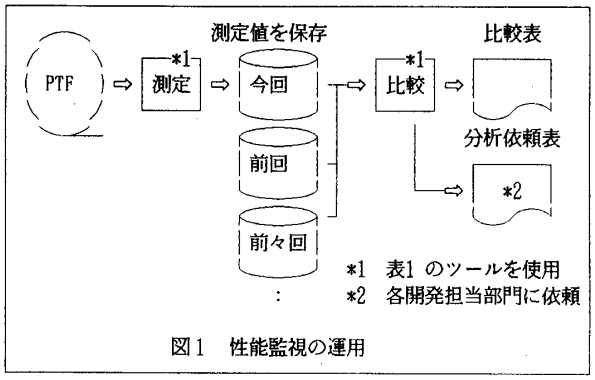

図1 性能監視の運用

4.2 性能監視項目と性能ツール

性能監視項目として4項目を設定し，上記の要件を満たす性能ツールを作成した（表1）．監視項目の中には，顧客先で要求される端末の応答時間やバッチ処理のスループットなど作業負荷の問題とモデル設定の難しさのため含んでいないが，システムの基本的なふるまいを監視することで十分だと判断している．性能ツールは，すべてコンポネント別に比較できるようになっている．

表1 ＰＴＦ性能監視項目

監視項目	内容	性能ツール *1
仮想記憶容量	仮想空間内のオペレーティグシステムが占める領域（プログラム域，データ域）をモジュール別に測定し，コンポネント別に分類，比較している．基本的なモデルが数種類用意されている	VSCHART
走行ステップ	特定モデルでの走行ステップ数をモジュール単位に測定し，コンポネント毎に分類，比較している．オペレーティングシステムの基本ルートが監視できる数種類のモデルを用意している．	DSCHART
I/O回数，スーパバイザ呼出し回数	同一ツールで得られる性能データだが，性能への影響度が大きいため項目を別にし，重点的に監視している．	
データセット容量	主要なシステムライブラリの大きさの内訳をメンバ単位に測定し，コンポネント別に分類，比較している．	FSCHART

*1 社内ツール

4.3 性能監視の運用方法

ＰＴＦ作成毎に以上に説明した監視項目を測定，比較している．通常のＰＴＦでは前回の測定値との比較を行い，バッジョンアップやレレベルアップ時には前バージョンレベルの測定値と比較を行っているまた，比較対象は出荷時のレベルまで遡ることもでき，任意の時点での比較が可能となっている．比較結果は，比較表としてコンポネント別に分類されたものが出力されるが，さらに監視基準値を設けて分析対象範囲の絞り込みを行い，監視作業の省力化を推進している．絞り込みの結果は分析依頼表として出力される（図1）．

(注)コンポネントとは，複数のモジュールの集合体である．一般に機能単位に一つのコンポネントを形成する．

5 性能監視の効果

OSIV/F4 MSP,OSIV/X8 FSP では2年前から，OSIV ESPⅢでは1年前から運用している．走行ステップ数と LPAサイズの変化の実例を示す（図2）．走行ステップ数は削減傾向にあり,CPUオーバヘッドは軽減されている． LPAサイズはやや増加傾向にあるが，著しい増加がないよう十分に監視効果が顕れている．

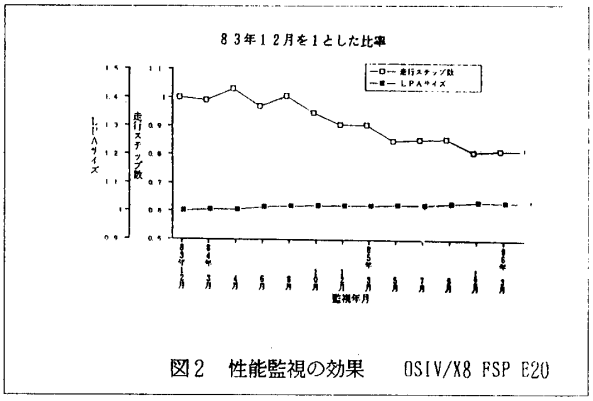

図2 性能監視の効果 OSIV/X8 FSP E20

6 最後に

先に性能劣化を防止する基本は，ソフト開発における全工程で性能を考慮した作業をすることであり，そのような認識の定着と日常作業への反映が目的であると述べた． CPUオーバヘッド削減， LPA増加の抑制にみられた効果は，開発担当者に性能の重要性がフィードバックされた結果である．次の開発作業にもこの効果は引継がれると確信している．なお，今後の課題はいかに早く的確に開発担当者にフィードバックするかである．そうすれば短期間にあるいはもっと大きい効果をあげることができるであろう．