

# Drain Model を使った SLA に基づく CPU のキャパシティ予測

松本征大 河野知行

株式会社 アイ・アイ・エム 技術部

## 概要

企業で稼働するコンピュータ・システムは、管理部門と使用者の間で結ばれる SLA (サービスレベル契約) に基づいてパフォーマンス管理およびキャパシティ計画の立案がなされる。処理すべき業務量を CPU 使用時間で計量することにより、Drain Model を使って CPU 能力と CPU 処理に要する時間との関係を見積もることができる。これにより SLA に基づいた CPU のキャパシティ計画立案が可能となる。

## CPU capacity planning based on SLA management using The Drain Model

Yukio Matsumoto

Tomoyuki Kawano

Engineering Department, IIM Corporation

## Abstract

Computer systems working in companies are managed for their performance and capacity planning based on SLA (Service Level Agreement: the contract of service quality between the departments of computer management and users). If you measure the overall volume of the workload as the total CPU time to process it, it will be possible to estimate the relation between CPU capacity and CPU process time for the target workload by using The Drain Model. In this way they can make the capacity plan to keep SLA.

## 1 背景

「必要な分だけ能力を買う」。企業におけるパフォーマンス管理／キャパシティ・プランニングの意味するところはここにある。しかし「必要なだけ」を判断するのは容易ではない。ざっと分けてもハードウェア資源には CPU、メモリ、I/O サブシステムがあり、増強／変更による効果の現れ方は資源毎に異なる。またそこで稼働するソフトウェア資源もチューニングやスリム化によってパフォーマンスを改善することができる。従ってシステム管理者は「ハードウェアの増強による可能性」と「ソフトウェアのチューニングによる可能性」とを見極める必要がある。

加えてビジネス全体がスピードアップした今では、これらを見極めて結論を出すまでの「時間」を短縮することが要求されている。新しいサービスを実現するため、あるいは合併による業務の倍増に対応するため、「どの部分に」「どれだけの能力を」増強（投資）するのがもっとも効果的かをスピーディーに判断しなければならない。

決定に至るプロセスの短縮には普段のパフォーマンス管理が不可欠なのは言うまでもない。これに加えて稼働管理データから「新たに必要なキャパシティ」を見積もるシンプルで理にかなったキャパシティ・プランニングの方法論があれば、決定に至る時間は大幅に短縮できる。

本稿では業務量W、処理能力Cと処理時間Tの関係を表す簡単なモデルを用いたSLAに基づくキャパシティ・プランニング方法を紹介する。このモデルは「**Drain Model**」と呼ばれ、考え方はきわめてシンプルである。しかしその意味するところは全ての資源と業務量の関係に当てはまる。ここでは「資源の増強／変更による効果の現れ方」が特に**Drain Model**に当てはまるCPU資源に焦点を当て、このキャパシティ・プランニング方法論を解説する。

## 2 Drain Model とは？

「**Drain**」とは英語で「排水」の意である。100リットルの水が入った水槽をイメージしよう。水槽を空にするには栓を抜くわけだが、仮にその排水管が「毎秒2リットル」排水することのできる太さだったら、すべて排水するのに必要な時間は $100 / 2 = 50$ 秒である。しかしこの管の太さが半分の「毎秒1リットル」しか排水できない太さだったら、水を抜き終わるまでに要する時間は先の2倍の100秒かかる。

どちらにしても総排水量は100リットルで固定であるから、

$$\begin{aligned} & \text{排水すべき水量 (定数)} \\ &= (\text{排水管の太さ}) \times (\text{排水にかかる時間}) \end{aligned}$$

が成り立つ。

同じことが業務処理にも言える。

$$\text{排水すべき水量} = \text{処理すべき業務量} : W \text{ (定数)}$$

$$\begin{aligned} \text{排水管の太さ} &= \text{業務処理能力} : C \\ & \quad (\text{与えた資源のキャパシティ}) \end{aligned}$$

排水にかかる時間 = 業務処理に要する時間 : T  
と置き換えれば、これらの間に次の関係が成り立つ。

$$\text{処理能力} C \times \text{処理に要する時間} T = W \text{ (定数)}$$

## 3 Drain Model を実際の業務へ

### ～ CPU への適用 ～

**Drain Model** を適用するにはまずキャパシティ・プランの策定対象とする資源を選ぶ。それからその資源の処理能力 : C とその資源からみた処理すべき業務の業務量 : W を元に、その資源で処理するのに要する時間 : T を求める。

我々は夜間バッチ業務の処理に必要なCPU能力を求めた。解析した環境は論理分割を使用しており、業務用2区画と開発用1区画の計3区画を運用している。解析対象の区画1に割り当てたCPU能力には余裕があるが、システム統合によって第3の業務用区画が稼働することになり、配分比を見直すことになった。この区画が夜間バッチを処理するのに必要十分なCPU配分を次のようにして見積もる。

図1は**Drain Model**の概念をCPUに絞って描いたもの。与えたCPUの処理能力が大きければCPU処理時間は短く、逆に小さければ長くなることが解る。排水管の太さ = CPU能力 : C と処理に要する時間 : T CPU の積は、水槽の水に相当する「夜間バッチのCPU業務量 : W (一定)」に常に一致する。WすなわちCPUにとっての夜間バッチの業務量とは「夜間バッチの処理に必要な総CPU使用時間」である。よって

$$C \times T_{\text{CPU}} = \text{夜間バッチのCPU業務量} W \text{ (一定)}$$

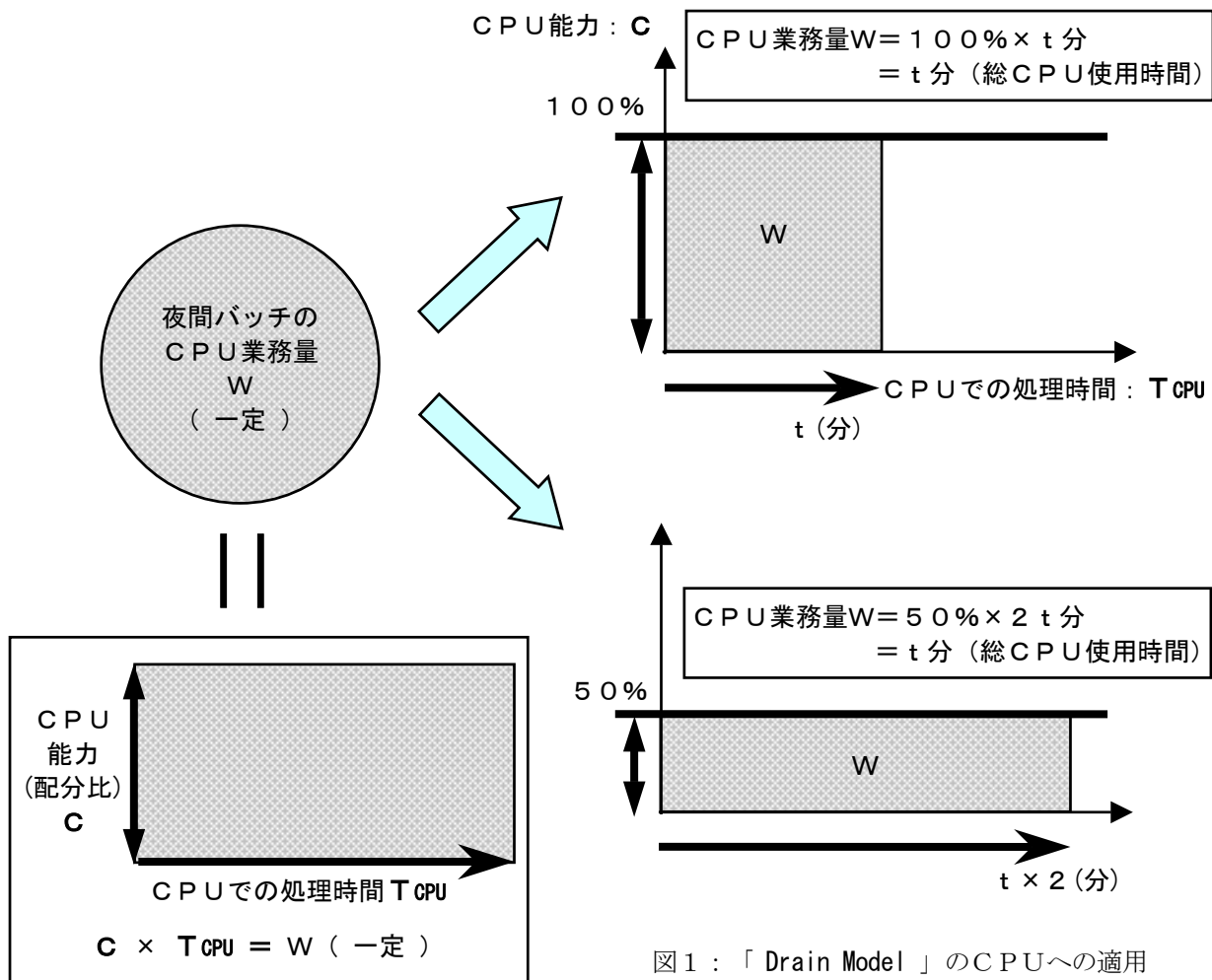
C : 区画に与えられたCPU能力  
(単位 : MIPS値、配分比 (%))

T<sub>CPU</sub> : バッチ業務を消化するのに必要なCPU処理時間

(単位 : インターバル数、  
又は時間の単位 (秒, 分))

が成り立ち、T<sub>CPU</sub>は次式で得られる。  
夜間バッチのCPU業務量W (定数)

$$T_{\text{CPU}} = \frac{W}{C}$$



#### 4 夜間バッチの「CPU業務量」Wとは？

図2は解析環境のある晩のCPU使用率時系列グラフである。区画毎に色分けしており、本稿では区画1を解析事例として紹介する。

既に触れたが、WすなわちCPUにとっての夜間バッチの業務量とは「夜間バッチの処理に必要な総CPU使用時間」である。つまり「夜間バッチの開始から終了までに使用する総CPU使用時間」である。これは図2における時刻A～Bまでのグラフの面積に相当する。Aはバッチ開始時刻、Bは夜間バッチ業務全体の最後を締めくくるジョブが終了した時刻である。

各インターバルのCPU使用率にインターバル長（ここでは5分）をかけると、そのインターバル内で使用した正味のCPU使用時間が得られる。これを時刻A～Bまで累積すれば、目的のWすな

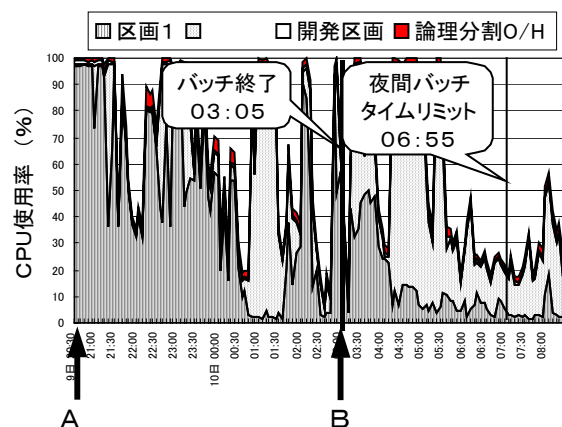


図2: CPU使用率時系列グラフ

わち「夜間バッチの開始から終了までに使用する総CPU使用時間」が得られる。本例の場合は  $W=187.3$  分

従ってCPU処理に要する時間は次式で得られる。

$$T_{CPU} = \frac{187.3 \times 100}{C(\%)} (\text{min})$$

これは図3の双曲線を描く。

## 5 SLAに基づくCPUのキャパシティ予測へ

SLAとは Service Level Agreement：サービスレベル契約のことである。これはコンピュータ・システムの管理部門と使用者部門の間で結ばれる、提供するサービスの品質に関する契約である。企業のシステム運用ではSLAが設けられ、システム管理者はこのサービスレベルを維持しつつ、予算をにらみながらパフォーマンス管理やキャパシティ・プランニングを行なう。

SLAでは、例えばオンラインの使用可能時間やレスポンス時間などを取り決める。夜間バッチではSLAの一つとして、夜間バッチ業務全体が終了すべきタイムリミットを予め決める。「〇時〇分までに夜間バッチを終了する」というタイムリミットを設けることで、オンライン業務開始までにオンラインサブシステムなどの起動時間を確保し、オンライン処理の円滑な開始を保証するのである。夜間バッチの終了時刻がタイムリミットを超える現象を「夜間バッチの突き抜け」などと呼ぶ。

Drain Model はSLAに基づいたCPUのキャパシティ・プランニングに有用である。夜間バッチは昼間のオンライン業務が終了した後開始する。従って夜間バッチの処理に許される時間  $T_{Limit}$  は「夜間バッチ開始時刻からタイムリミットまで」である。そこで図3のグラフを元に、

$$T_{CPU} < T_{Limit}$$

となるCPU能力： $C$ を与えれば良い。これがDrain Model を用いたSLAに基づくCPUのキャパシティ・プランニングである。

解析環境では図2より  $T_{Limit}=625$  分、従ってSLAを満たすCPU配分比は

$$T_{CPU} = \frac{187.3 \times 100}{C(\%)} < T_{Limit} = 625 (\text{min})$$

$$\therefore C > 30.0(\%)$$

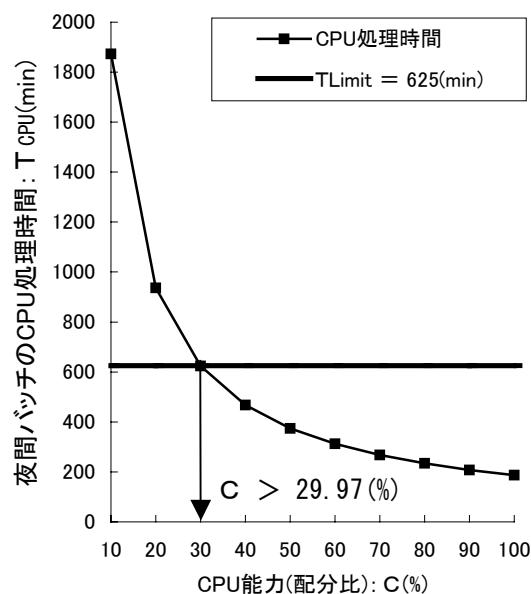


図3：Drain Model を用いたSLAに基づくCPUのキャパシティ・プランニング

## 6 T CPUの意味

$T_{CPU}$ とは、区画に与えたCPU能力を常に目一杯使い続けて夜間バッチを処理した場合の所要時間である。しかし論理分割の場合、設定によっては割り当てを超えてCPUを使うことができるし、逆に割り当てたCPU能力を使い切らないこともある。前者の状態が続けば夜間バッチの実行時間  $T$  はDrain Model で見積もった  $T_{CPU}$  よりも短くなり、後者の場合は逆に  $T_{CPU}$  より長くなる。

## 7 考察1：業務量が増えるとき

図4は業務量が異なる場合のDrain Modelのプロット例である。業務量が一定の場合双曲線は1本だが(図3)、業務量の変化を見込んでCPUのキャパシティ・プランを策定する場合には  $W$  の値が異なる複数の双曲線を描く必要がある(図4)。

図4ではCPU業務量  $W$  が異なる4つの双曲線を示した。このうち■でプロットされるものは図3と同じ曲線である。これに対しCPU業務量が1.5倍、2.0倍、2.5倍に増えたものがそれぞれ◆、▲、×でプロットされたものである。業務量が増

えても1日が24時間であることは変わらないから、通常夜間バッチに許される処理時間が増えることはない。従って図3と同じタイムリミット（ $T_{Limit}=625$ 分）を遵守させるなら、最低CPU能力 $C_1 \sim C_4$ も同じ倍率で表1のように変化する。

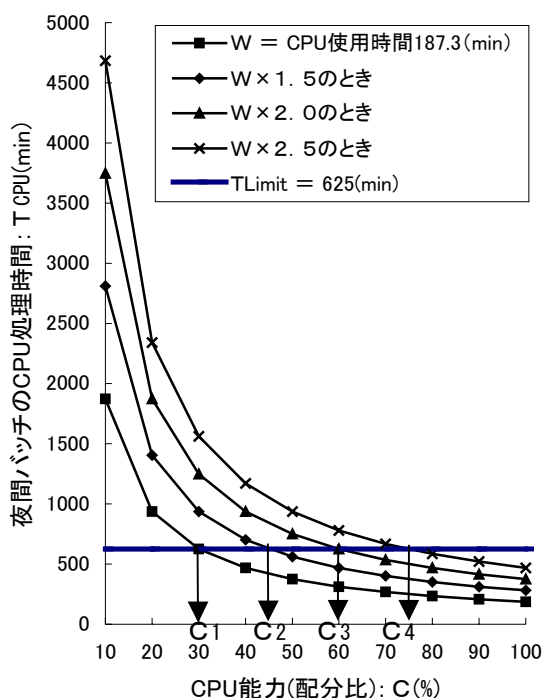


図4：業務量が増える場合の  
CPUのキャパシティ・プランニング

| CPU業務量<br>(CPU使用時間)   | SLA(タイムリミット)<br>を守るための最低CPU能力 |
|-----------------------|-------------------------------|
| $W=187.3(\text{min})$ | $C > C_1 = 30.0\%$            |
| $W \times 1.5$        | $C > C_2 = 45.0\%$            |
| $W \times 2.0$        | $C > C_3 = 59.9\%$            |
| $W \times 2.5$        | $C > C_4 = 74.9\%$            |

表1：業務量が増える場合の  
最低CPU能力の推移

## 8 考察2：Totalの業務処理時間 $T_{Total}$

解析事例では、実際の夜間バッチ所要時間から $T_{CPU}$ を引いた後に残る時間が存在する。これは何の時間だろうか。

コンピュータ・システムでの業務処理過程は大

まかに言ってプロセッサ使用と入出力装置へのアクセスの繰り返しである。複数の業務を並行処理する際は、CPU処理とI/O処理とを並行処理することによりスループットの向上を図る。しかしプログラム多重度をどれだけ増やしたとしても、並行処理するいずれの業務もCPUでの実行可能状態ではなく、それら全てがI/O処理を行なっている時間 $T_{I/O}$ が存在し得る。またシステムによってはスケジュールの設定により敢えて業務処理を実行しないデッドタイム $T_{Dead}$ も存在する。

このような理由から Drain Model で $T_{CPU}$ から見積もった配分比を割り当てても与えた能力を使い切らず、その結果夜間バッチの所要時間が $T_{CPU}$ より長くなる。従って正確なキャパシティ・プランニングを行なうためには

夜間バッチの処理に要する時間 $T_{Total}$

$$= \text{CPU 処理時間 } T_{CPU} + \text{I/O 処理時間 } T_{I/O} \\ (\text{+デッドタイム } T_{Dead})$$

と考える必要がある。

## 9 I/O処理時間 $T_{I/O}$ の導出

Drain Model の考え方はI/O処理にも当てはまる。 $T_{I/O}$ は次式で与えられる。

$$T_{I/O} = \frac{\text{夜間バッチのI/O業務量 } W_{I/O} \text{ (定数)}}{\text{I/Oサブシステムの処理能力: } C_{I/O}}$$

しかし実際のシステムでこれを解析するには根本的な問題がある。データの入手が困難なのだ。

既に見たように $T_{CPU}$ はCPU使用率の累積値から Drain Model で見積もることができる。メーカー提供のパフォーマンス計測ツール（RMF、PDL、SAR等）は例外なくCPU使用率を提供するのでこれは問題ない。しかしI/O業務量 $W_{I/O}$ やI/Oサブシステムの処理能力 $C_{I/O}$ に相当するデータはどのパフォーマンス計測ツールも提供していない。従って Drain Model でSLAに基づく正確なキャパシティ・プランニングを行なうには、「 $T_{I/O}$ の見積り」の問題が残る。

| CPU<br>配分比<br>(%) | バッチの<br>所要時間<br>(min) | CPU<br>使用時間<br>W CPU(min) | 総CPU<br>待ち時間<br>(min) |
|-------------------|-----------------------|---------------------------|-----------------------|
| 25                | 709.8                 | 184.7                     | 1665.0                |
| 35                | 399.8                 | 187.3                     | 322.5                 |
| 差                 | 310.0                 | -2.6                      | 1342.5                |

表2：異なるCPU配分比での

夜間バッチ実行結果

そこで今回の事例では  $T_{I/O}$  を間接的に求めるアプローチをとった。

表2は区画1で同一内容の夜間バッチを異なるCPU配分比の下に実行した結果である。配分比を減少させると予想通り所要時間は伸びたが、注目したいのはCPU業務量  $W_{CPU}$  すなわちCPU使用時間がほとんど同じである点だ。これは両日の夜間バッチ業務が質・量共に同じであることを示す。従って両日のI/O業務量  $W_{I/O}$  も同じである。2つの実行日の間でI/Oサブシステムの構成に変更は無く、処理能力  $C_{I/O}$  も同じだから、両日の  $T_{I/O} = \text{一定}$  と見なすことができる。故に、より正確な Drain Model の式は次のようになる ( $W_{CPU}$  の値はこのまま 187.3 を使用する)。

$$T_{Total} = \frac{187.3 \times 100}{C} + T_{I/O}(\text{一定}) \quad (\text{min})$$

最後に夜間バッチの所要時間に占めるCPU処理時間を求めて所要時間から差し引けば、定数  $T_{I/O}$  が得られる。

我々は表2で総CPU待ち時間とバッチの所要時間が共に増加していることに注目した。両者の増加は共通の原因による。CPU配分比を減らしたためにプロセッサへのアクセス待ちが増加したことが原因である。論理区画間のCPU競合が激化したことによって新たなCPU待ちが発生し、同時にCPU処理時間  $T_{CPU}$  が長くなって全体の所要時間が伸びたのである。

このことから両日の総CPU待ち時間の比は、両日のバッチ所要時間のうち「区画間競合（すなわちCPU使用制限）に起因して伸びたCPU処理時間」の比に等しいと考え、図6に示す計算より図のようにバッチ所要時間の内訳を求めた。

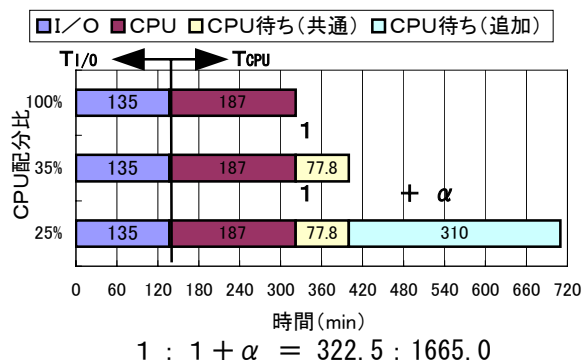


図6：夜間バッチ所要時間の内訳

よって  $T_{I/O}(\text{一定}) = 135(\text{min})$  より

$$T_{Total} = \frac{187.3 \times 100}{C} + 135(\text{min})$$

グラフは図7のようになる。SLAを守るには

$$T_{Total} < T_{Limit} = 625 \therefore C > 38.4\%$$

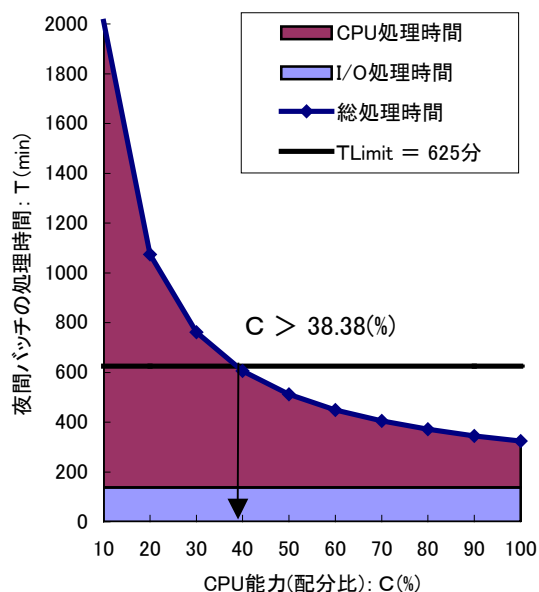


図7：区画1に必要なCPU配分比

## 10 終わりに

解析事例のようにSLAでは「時間」で定義するものが多い。時間とキャパシティとを直接結びつける点で Drain Model は有用なキャパシティ・プランニング方法論である。しかし入手できるデータの制限により適用できるケースが限られる。

以上