



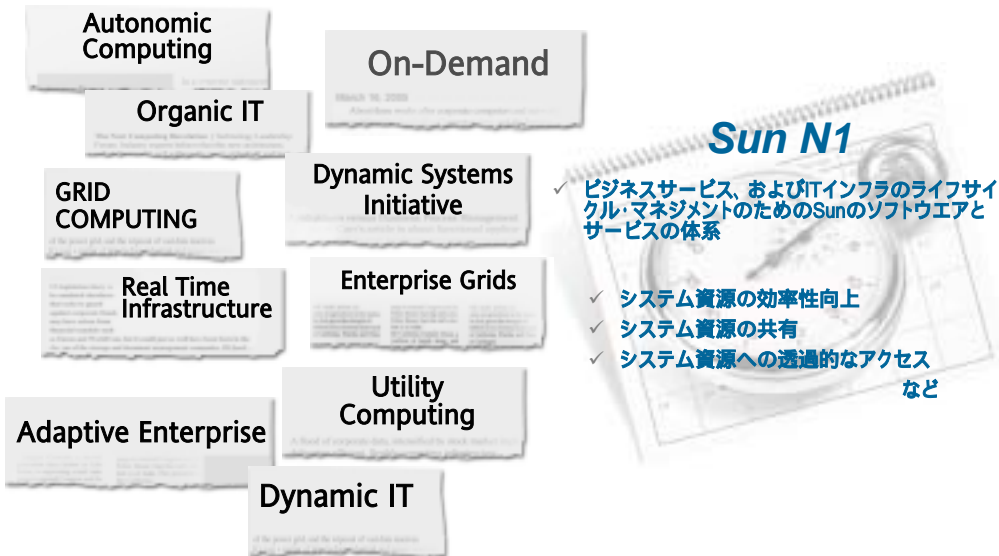
N1 Grid Engine 6 スケジューリングとリソース管理

Sun Microsystems K.K.
Client Solutions
Nobuchika Kobayashi

Outline

- ✓ N1 Vision / N1 Products
- ✓ N1 Grid Engine の概要
- ✓ Grid Computing Model
- ✓ N1 Grid Engine の機能概要
- ✓ N1 Grid Engine のしくみ概要
- ✓ スケジューリングとリソース管理
 - ✓ スケジューリング概要
 - ✓ Job ソートのプライオリティ
 - ✓ Job requirement と Complex (Job と Queueのマッチング)
 - ✓ リソース配分ポリシー
 - ✓ その他のスケジューリングとリソース管理 : Limit / Load-Suspend Threshold / Array Job
- ✓ 参考資料

N1 Vision



N1 Products



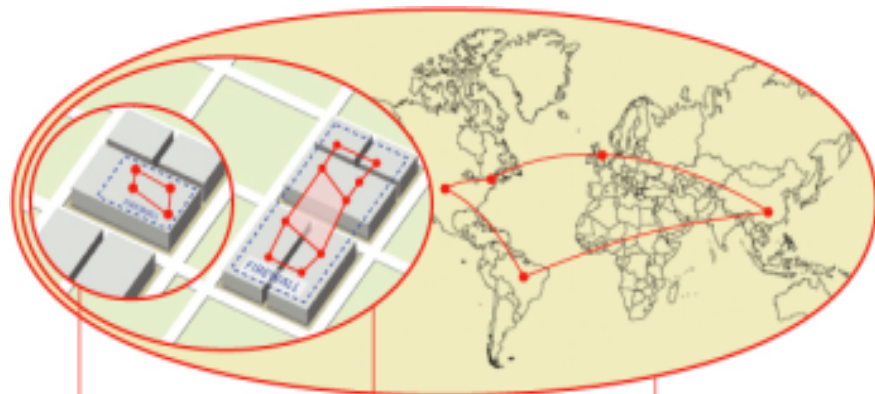
N1 Grid Engine 機能概要

Compute Grid from Sun

N1 GRID Engine 6 はCompute Gridを実現するソフトウェア

- | | |
|---|------------------------|
| ノードの負荷に応じてジョブ配分を行い、高いシステム・スループットと高いリソース使用率を実現 | アカウントティング機能 |
| 実行前ジョブのスケジューリング順序を制御可能 | ジョブコントロール機能 |
| リソース定義機能 | アクセス制限機能 |
| ポリシーによるリソース配分が可能 | メール通知機能 |
| ジョブを実行可能なノードの自動制御 | ノード間通信にSSLを使用 |
| アレイジョブを実行可能 | チェックポイント・ジョブを実行可能 |
| クラスター・キューによる管理性向上 | 並列計算ジョブ(MPI, PVM)を実行可能 |
| ジョブの状態監視 | |
| 実行ノードの負荷情報および状態の定期的な取得 | |
| マスターホストの冗長化構成 | |
| ジョブ障害時の他ノードへのジョブ・マイグレーション | |
| DRMAA (Distributed Resource Management Application API) | |

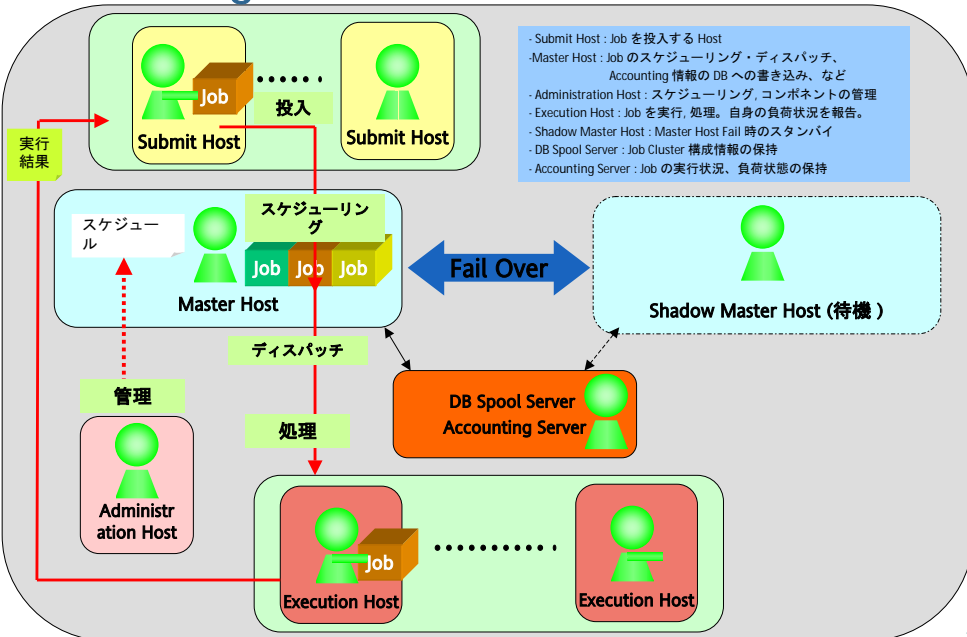
Grid Computing Model



- | | | |
|--|--|---|
| Cluster Grid
- 単一のプロジェクト、組織
- 単一サイト
- N1 Grid Engine 6.0 | Enterprise(Campus) Grid
- 複数のプロジェクト、単一の組織
- 単一サイト
- N1 Grid Engine 6.0 | Global Grid
- 複数のプロジェクト、複数の組織
- 複数のサイト |
|--|--|---|

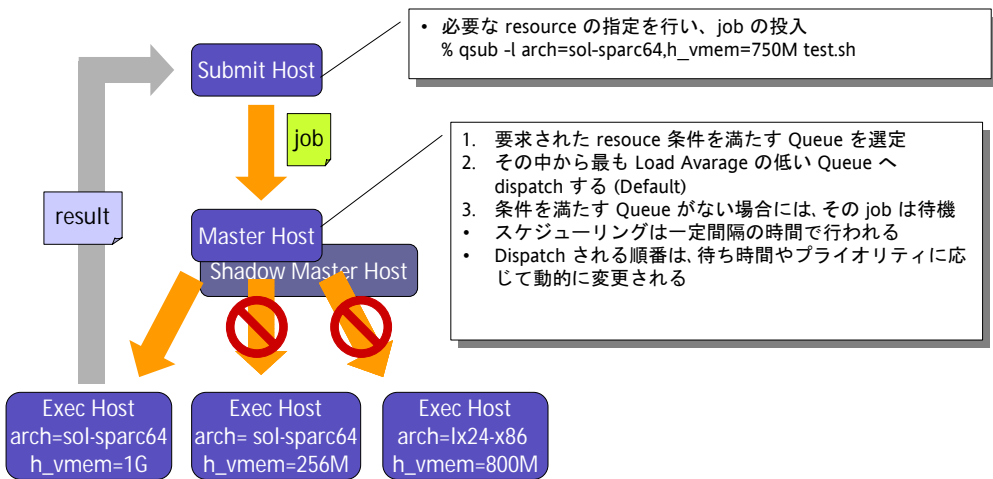


N1 Grid Engine のしくみ概要



スケジューリングとリソース管理

スケジューリング概要



Job の投入 (Job Requirement)

The screenshot shows the N1 GE interface for job submission. Annotations explain various fields: '投入するジョブを記述した shell script の場所' (Shell script location), '投入するジョブを記述した shell script の引数' (Shell script arguments), 'ジョブを開始する日時 YYYYMMDDhhmm.[ss] の形式で指定' (Job start time in YYYYMMDDhhmm.[ss] format), 'stdout/stderr を出力するディレクトリ' (Output directory for stdout/stderr), '指定しなければユーザのホームディレクトリに出力される' (Default output to user's home directory), 'ジョブの実行に必要なリソース要求を指定' (Specify resource requirements), and '指定可能なリソース値一覧' (List of available resource values). A 'Hard Resource' section notes that jobs are not executed if requirements are not met, while a 'Soft Resource' section notes they are executed even if requirements are not met.

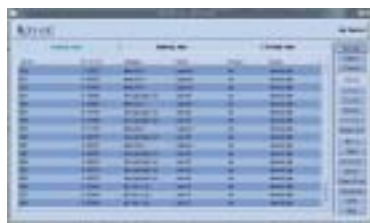
- 投入するジョブを記述した shell script の場所
- 投入するジョブを記述した shell script の引数
- ジョブを開始する日時 YYYYMMDDhhmm.[ss] の形式で指定
- stdout/stderr を出力するディレクトリ
• 指定しなければユーザのホームディレクトリに出力される
- ジョブの実行に必要なリソース要求を指定
- 指定可能なリソース値一覧
- Hard Resource: 必ず要求を満たさなければならないリソース値 (条件を満たす Queue が存在しない場合ジョブは実行されない)
- Soft Resource: 満たすことが望ましいリソース値 (条件を満たす Queue が存在しない場合にもジョブは実行される)

Job ソートのプライオリティ

N1 Grid Engine 6.0におけるジョブの優先度

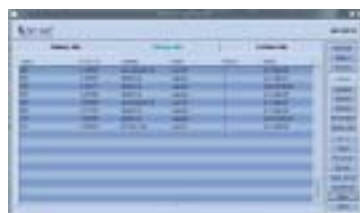
$$= A \times \text{Priority} + B \times \text{Urgency Policy} + C \times \text{Ticket Policy}$$

- Priority: ジョブ投入の際にユーザーが指定可能 () A, B, Cは管理者によって指定可能
- Urgency Policy: 要求するリソース量やDeadline JobによってN1GEが調整
- Ticket Policy: (後に記述するポリシーを参照) 管理者によって割り当てるリソース配分量



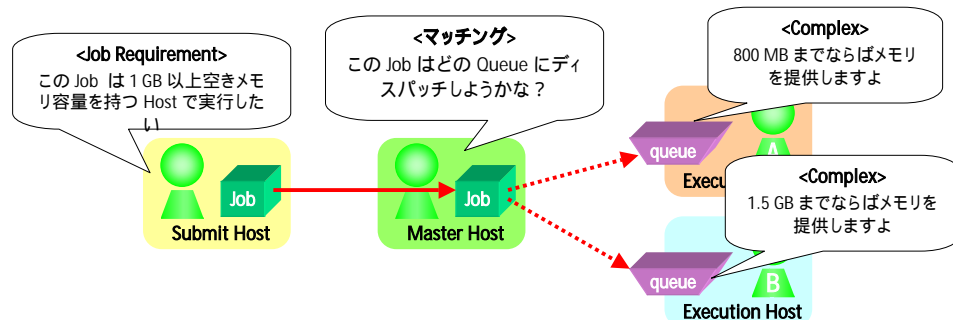
Jobsの状態監視、コントロールの例 (QMON: 保留中のジョブ)

Jobsの状態監視、コントロールの例 (QMON: 実行中のジョブ)



Job Requirement と Complex Job と Queue のマッチング

- ✓ Complex の設定を基に Job Requirement と Queue のマッチングが行われ、Job が最適な Queue で実行される
- ✓ Complex は Host/Queue における制限を設定できる



Complex の適用例 (Queue/Host)



リソース配分ポリシーの種類

✓ Functional Policy

- ✓ 常にリソースの利用割合を各 Project / Department (ユーザグループ) のシェアの割合に保たれるようにスケジューリングを行う
- ✓ 他の Project/Department がリソースを使用していない場合にはリソースを占有することが可能

✓ Share Tree Policy

- ✓ Functional Policy が常に実行中のリソースの割合を一定に保つのにに対し、過去のリソース利用履歴を参照し、過去に遡って利用割合を一定に保つようスケジューリング

✓ Deadline Policy

- ✓ ある Project/Department がある期日までに完了させなければならないジョブがある場合に、一時的に利用割合を高めることで期日に間に合うようスケジューリングを行う(ただし、Job の実行時間を考慮したものではない)

✓ Override Policy

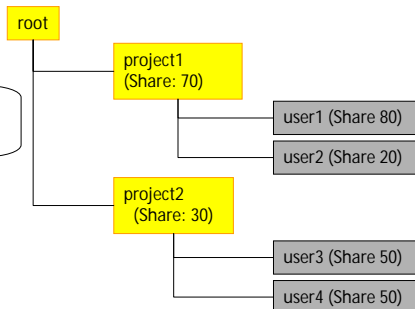
- ✓ 管理者が一時的にリソースの利用割り当てを変更

ポリシーにおける Share の概念

リソースの割り当ては Ticket という概念で決定する

Share Tree Policy
(Ticket: 10,000)

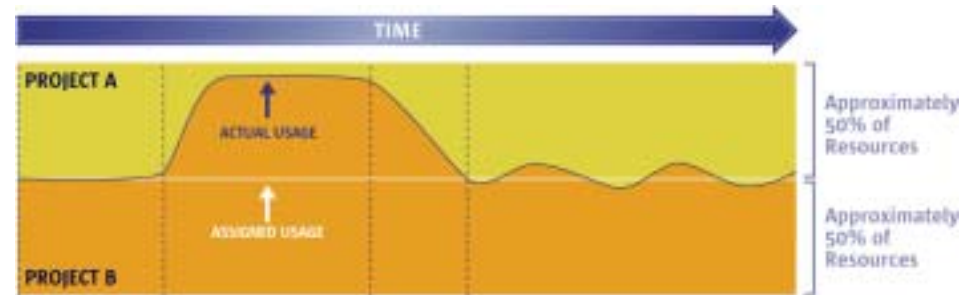
Ticket 数が必要なのは他のポリシーとの適用割合を決定するため



user	ticket	total Share
user1	$10,000 * 70 / (70+30) * 80 / (80+20) = 5,600$	$5,600 / 10,000 = 56\%$
user2	$10,000 * 70 / (70+30) * 20 / (80+20) = 1,400$	$1,400 / 10,000 = 14\%$
user3	$10,000 * 30 / (70+30) * 50 / (50+50) = 1,500$	$1,500 / 10,000 = 15\%$
user4	$10,000 * 30 / (70+30) * 50 / (50+50) = 1,500$	$1,500 / 10,000 = 15\%$

Functional Policy

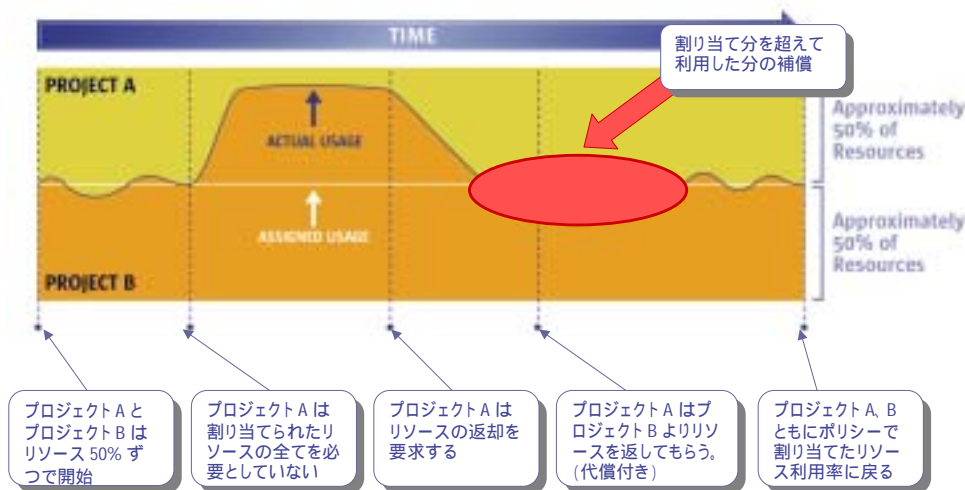
リソース利用割合を一定に保つ



- プロジェクト A と プロジェクト B は リソース 50% ずつで開始
- プロジェクト A は 割り当てられたリソースの全てを必要としていない
- プロジェクト A は リソースの返却を要求する
- プロジェクト A は リソースの返却を要求する

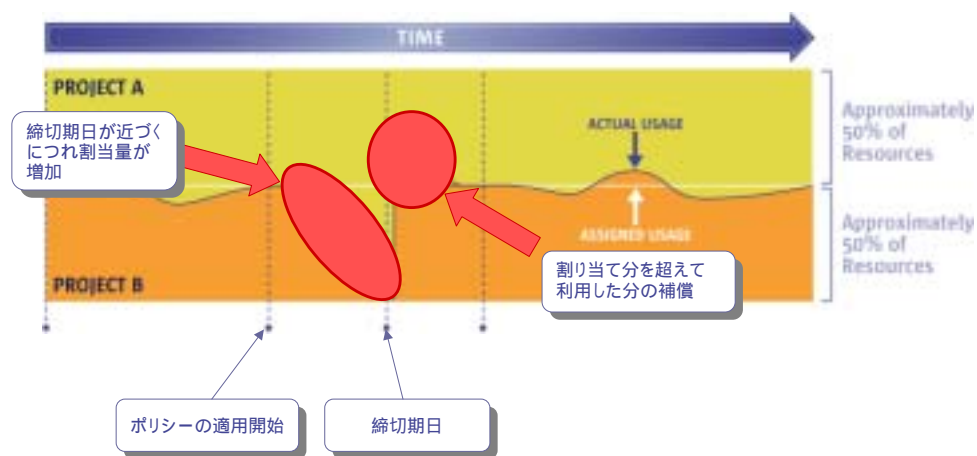
Share Tree Policy

過去のリソース使用量を考慮する



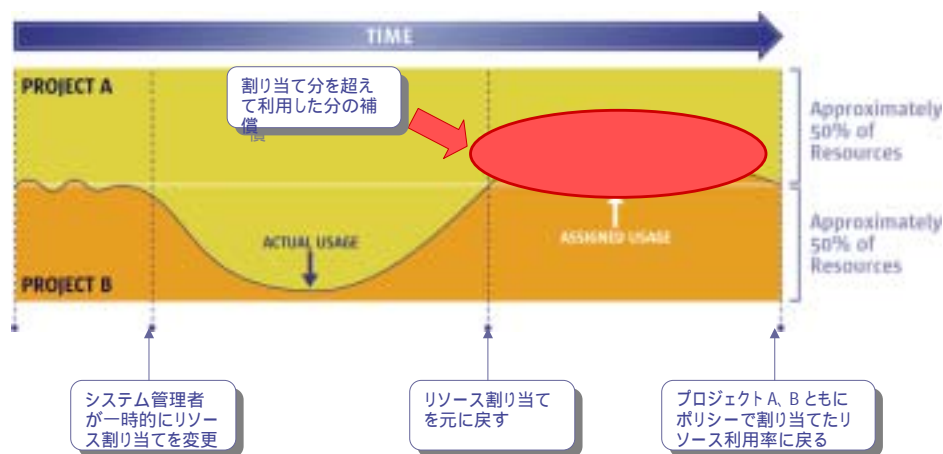
Deadline Policy

締切期日に間に合うように優先的にジョブを実行



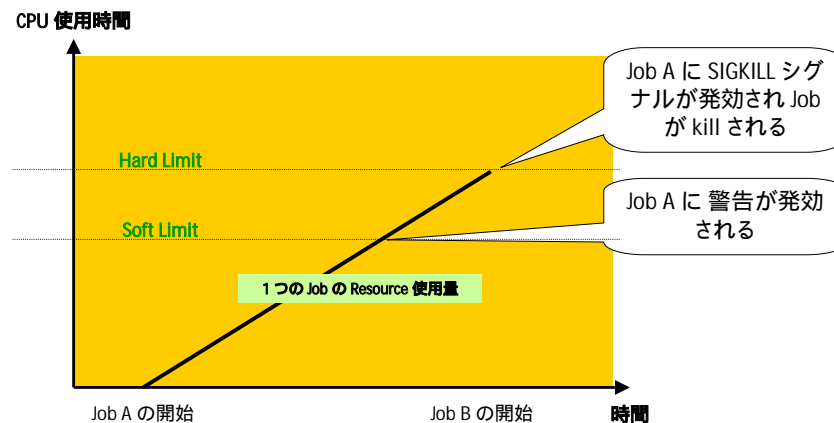
Override Policy

管理者が手動で一時的にリソース割当を変更する



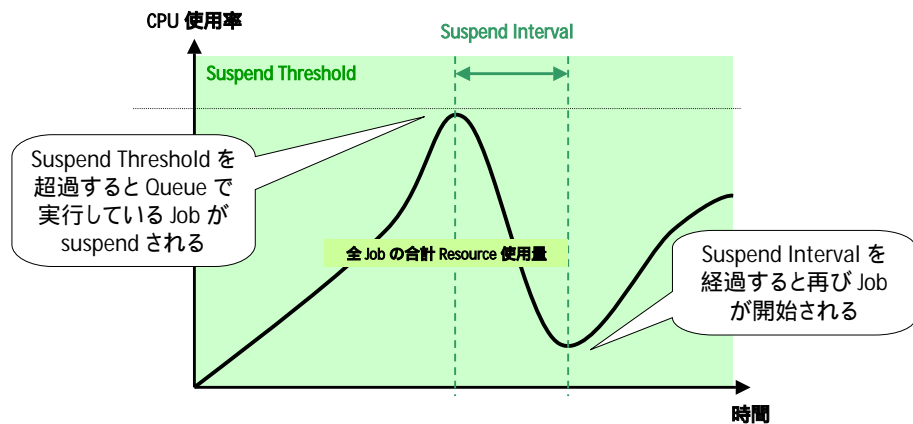
Limit

✓ Queue 内において、実行中の1つ Job が Resource を占有することを防ぐ



Load/Suspend Threshold

- Queue で実行されている全ての Job の合計 Resource 使用量を制限することが可能



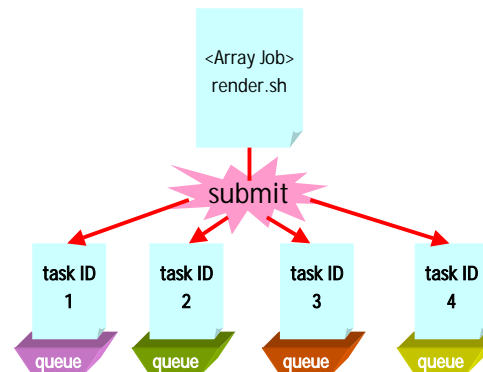
Sun Microsystems K.K.

21

Array Job

同一 Job をパラメータを変化させ繰り返し実行する

Array Job を Submit すると同一の Job が任意の数作成される
Job にはパラメータとして task ID が与えられる
それぞれの Job は異なる Queue で同時に実行することが可能



Sun Microsystems K.K.

22

Array Job の投入 (QMON)

パラメータを 下限-上限:間隔 の形式で入力

Job Script

/tmp/array_job.sh

Job Tasks

1-5:1

Job Name

array_job.sh

```
$ qsub -t <start>-<end>:<step> <script_name>
```

```
(例) $ qsub -t 1-5:1 array_job.sh
```

Sun Microsystems K.K.

23

参考資料

- ✓ N1 Grid Engine 製品
<http://jp.sun.com/products/software/gridware/>
- ✓ Grid Engine (N1GE の Open Source 版)
<http://gridengine.sunsource.net/>
- ✓ N1 Grid Engine Document
<http://docs.sun.com>

Sun Microsystems K.K.

24



Sun Microsystems K.K.
Client Solutions
Nobuchika Kobayashi

