CEATEC2010連携
エンタープライズにおけるクラウドコンピューティングの適用可能性
クラウドコンピューティングによる
アプリケーション試作開発事例
ー トライアルプロジェクト中間報告　ー

電子情報通信学会SWIM研究会
CEATEC連携
2010年10月7日

宮西洋太郎

SWIM（ソフトウェアインタプライズモデリング）研究専門委員会

宮城大学客員教授
仙台ソフトウェアセンター嘱託
㈱アイエスイーエム

本日の発表内容

前半

1. ＣＣＴＰ（クラウドコンピューティングトライアルプロジェクト）の紹介
2. クラウドコンピューティングに関連する話題

後半

3. 「意見収集・形成システム」の紹介
4. ＧＡＥによるプロトタイプシステム構築の紹介

Ｇｏｏｇｌｅで公開中

http://opvtsystem.appspot.com/

1．SWIMクラウドコンピューティング トライアルプロジェクト(1)

SWIMとは

電子情報通信学会の研究会(信学会では、研究専門委員会)

Software Interprise (InternetとEnterpriseとを結合した造語)Modeling

ホームページ：http://www.ieice.org/~swim/jpn/

エンタープライズ(企業、非営利組織)と情報技術との相互イノベーションを目指す。

クラウドコンピューティングトライアルプロジェクト(CCTP)

2010年度SWIM研究会が行う集団的研究活動

クラウドコンピューティングトライアルプロジェクト(CCTP)の目的

(1) エンタープライズ用途の情報システムにとって、クラウドコンピューティングは次世代経営情報技術として本命になりうる画期的技術なのか、単なるバズワードに終わるのか、学会研究会として、不偏不党の第三者的、かつ学術的立場から、見極めること。

(2) エンタープライズ情報システムにおけるデータそのものやデータ処理方法は、まさに企業の魂であり、生命線ともいえる。行政などでは、個人情報の保護が欠かせない。それらを外部にゆだねることの不安やリスクとひきかえに如何なるメリットを享受しうるかを見極めること。

(3) もし本命になりうる技術ならば、リレーショナルDBの多様な検索の不使用や粒度の大きいACIDトランザクションの不使用など、従来の分散システム構築方法の常識を覆す可能性があり、クラウドコンピューティング技術をエンタープライズ情報システムへ適用するための実践的ノウハウ(実践的知見)を獲得することが、エンタープライズ情報システムの構築技術者(あるいは企業)にとって、重要かつ喫緊の課題であり、実践的ノウハウを獲得すること。

1．SWIMクラウドコンピューティング トライアルプロジェクト(2)

プロジェクトの基本的な考え(プロジェクト憲章)

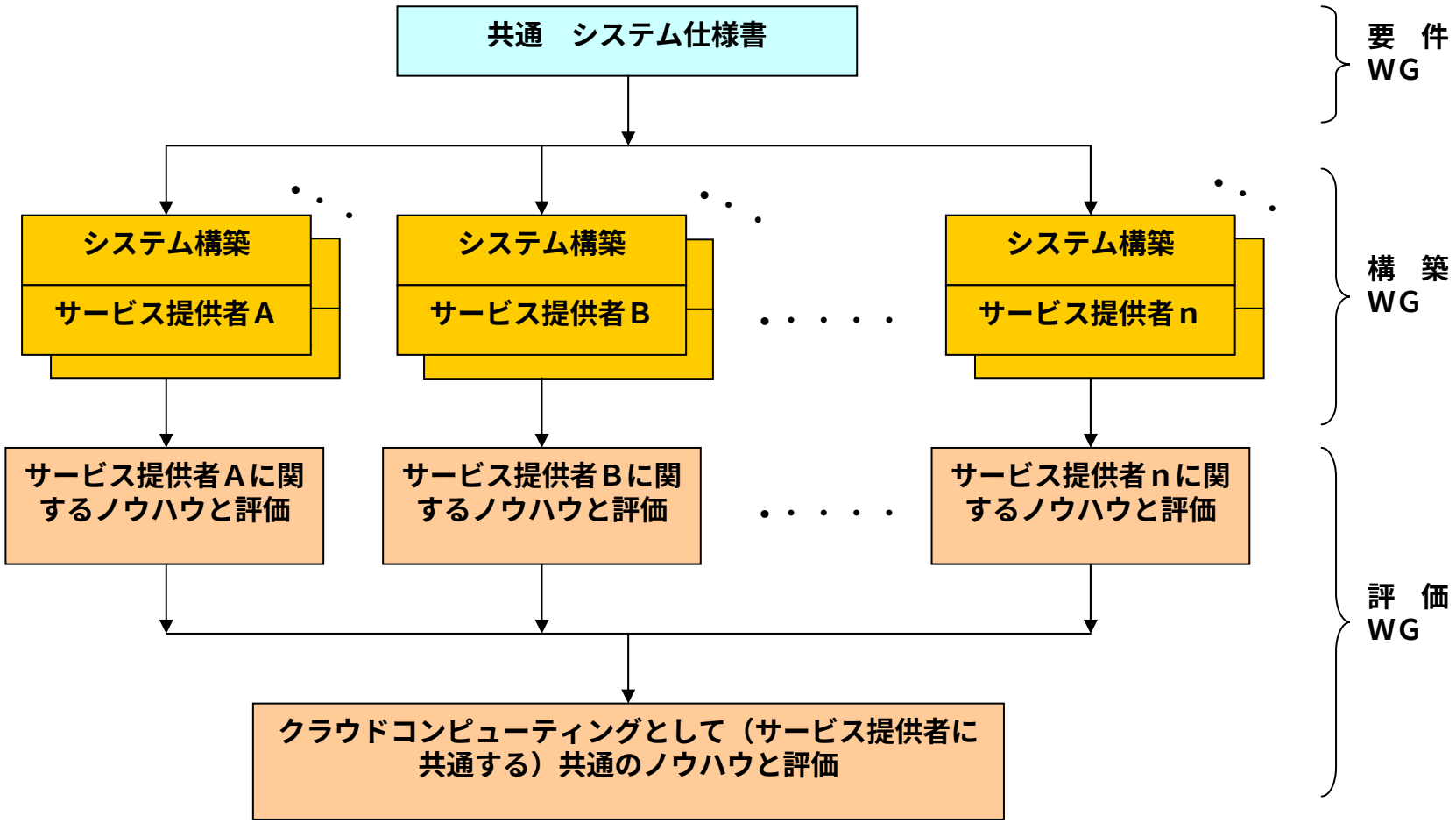
- **「冷静な科学技術的とりくみ」**
- **「Give&Takeの精神：各自自分のノウハウを提供するかわりに、他の方からもそれ以上のノウハウをいただく」**
  同じ「要件定義」にしたがって、各社のサービスを利用して、自主的にプロトタイプを作成し、その経験をもちより、評価します。多くのプロトタイプ構築の実績を必要とします。これは、集団的取組みでないと難しい。
- **「来るは拒まず、去るは追わずの精神：プロジェクトへの参加も脱退もメンバーの自主的な意思とします」**
- **「ボランティア活動の精神：各自のできる範囲(リソース)でご協力をお願いします」**
- **「得られたノウハウの活用は、なんら制限しません」(ビジネス展開など)**
  例えば、Takeしたノウハウの活用も制限しません。Give&Takeです。
- **「基本的に本プロジェクトでの直接の金銭の出入りはありません」(各自の負担でトライアル、義務もなし)**

1．SWIMクラウドコンピューティング トライアルプロジェクト(3)

個人情報保護の基本的な考え(プライバシー保護のポリシー)

- 参加者名簿を作成し、参加者に提供します。
- 参加者名簿は個人情報ですので、このプロジェクト以外でのご使用はご遠慮くださいますよう、お願いいたします。
- また、本プロジェクトには参加するが、どうしても名簿から除いておくことをご希望のかたは、お知らせください(好ましくありませんが)。
- また、相互にバックグラウンド(いままでのご経験)の理解を深めるため、ご経歴の会社、大学も記しておきました。もし間違っていましたら、ご指摘ください。もちろん、空欄のままでも結構です。
- プロジェクト参加者は、プロジェクトに関しましては、「仲間うち」ということで、お互いの連絡などに、ご活用いただきますと、ありがたいと存じます。

1．SWIMクラウドコンピューティング トライアルプロジェクト(4)



1．SWIMクラウドコンピューティング トライアルプロジェクト(5)

- マイルストーン
  - プロジェクト計画書発行、参画者の応募、(メール、HP) 2009/12
  - プロジェクト計画書の意見公募 (Request for Comment) 2010/1
  - 参画者の確定、参画者が事前調査を完了、 2010/2
  - プロジェクト計画書の公開、意見収集、改定完了、 2010/3
  - プロジェクトスタート、 2010/4
  - 要求仕様書、不明点の解決、 2010/5
  - 各社のクラウドコンピューティングサービスの疑問点解決、 2010/6
  - 各参画者の情報システム開発完了、 2010/8
  - 成果の評価、とりまとめ、 2010/9
  - 成果の外部発表 (例えば案、CEATECでのチュートリアル) 、 2010/10
  - 成果の検討、議論、今年度のまとめ、 2011/2

1．SWIMクラウドコンピューティング トライアルプロジェクト(6)

成果の目標例として、

- あるベンダーX社ごとに、例えば，下記のような整理を行う(ベンダー提供のサービスを利用するノウハウ)

① この業務(ア)は、どう工夫しても、X社のIAAS、PAASではできないアプリケーションである

- 例：X＝Google社で、頻繁にトランザクション(読み書き)が行われ、リアルタイムに処理することを要求されるアプリケーション。
- 例：X＝Google社で、従来型のＲＤＢのＳＱＬ機能をふんだんに使うアプリケーション。

② この業務(イ)は、多少要件を緩めることによって、X社のIAAS、PAASで構築できるアプリケーションである

- 例：リアルタイム性を、ヒューマンインタフェースで許容範囲で、待たせることによって、実現できるアプリケーション。

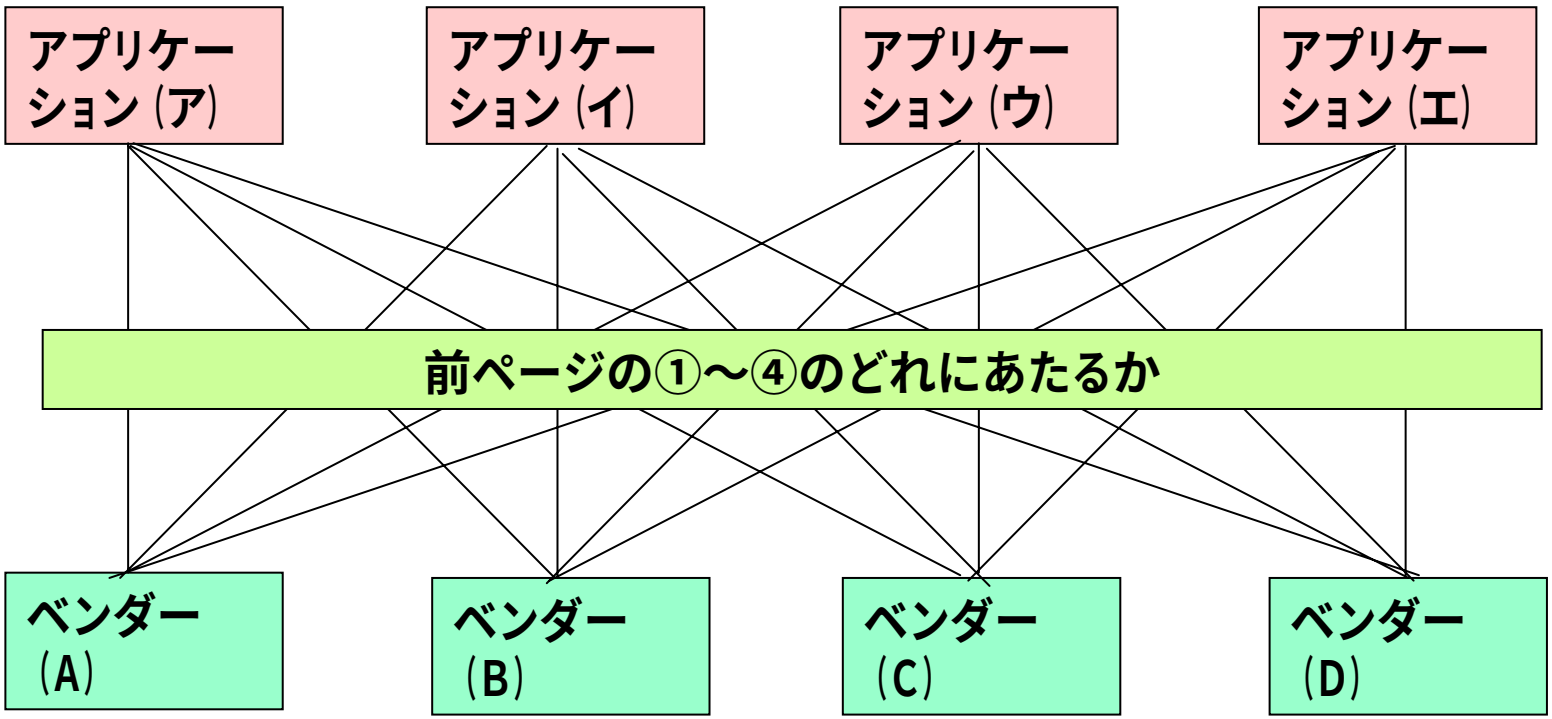
③ この業務(ウ)は、データの持ち方を工夫することによって、X社のIAAS、PAASで構築できるアプリケーションである

- 例：読み取り専用のデータと書き込みは一方向性にできるといったデータのもちかたで、実現できるアプリケーション。
- リアルタイムシステムであっても、センサーなどのモニタリングシステムのようなもの。

④ この業務(エ)は、たいした工夫をしなくても、すなおにX社のIAAS、PAASで構築できるアプリケーションである

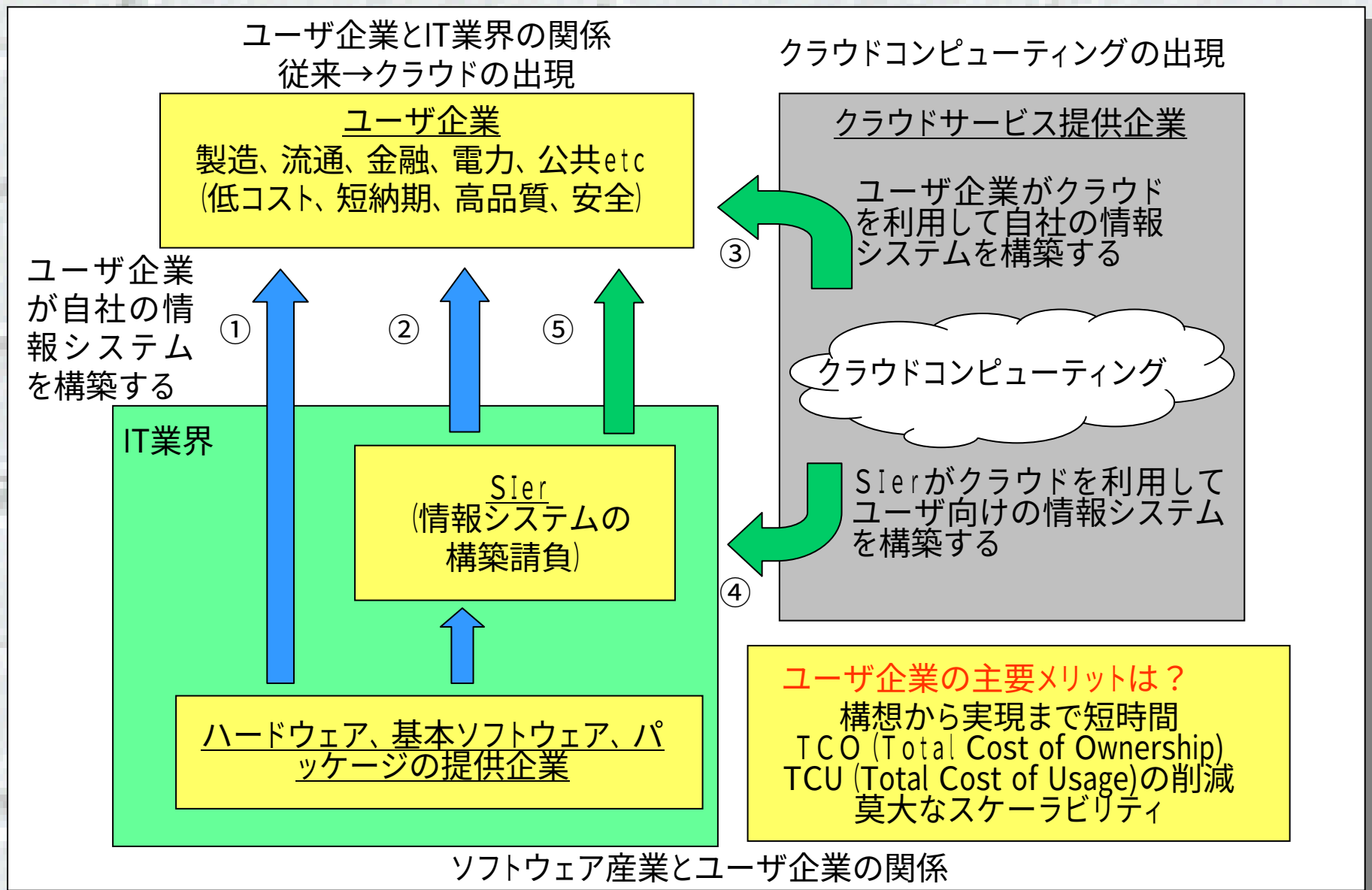
- 例：大量の情報源から検索するアプリケーション(Ｇｏｏｇｌｅの本来のアプリケーション)

1．SWIMクラウドコンピューティング トライアルプロジェクト（7）



2．話題（1）クラウドはビジネスチャンスか－１

話題（1）ユーザ企業、情報サービス産業、クラウドサービス提供企業の関係



2．話題 (1) クラウドはビジネスチャンスか－2

話題 (1) ユーザ企業、情報サービス産業、クラウドサービス提供企業の関係

クラウド以前

①ユーザ企業は，プラットフォーム提供企業 (ハードウェア，基本ソフトウェア，パッケージ提供企業) から製品を購入し，自ら情報システムを構築する．

②ユーザ企業は，SIerに委託し，SIerは情報システムを構築し，ユーザ企業に提供する．ユーザ企業はその情報システムを購入し，所有する．

クラウド以後

③**ユーザ企業**は，クラウドコンピューティングのサービス (PaaS) を用いて，自社に必要な情報システムを構築する．

④→⑤

SIerが，クラウドコンピューティングのサービスを用いて，ユーザ企業に必要な情報システムを提供する．

③，⑤の提供方法について，SIer (ソフトウェア会社) には，次にあげるようなビジネスチャンスがありうる．

2．話題 (1) クラウドはビジネスチャンスか－3

話題 (1) ユーザ企業、情報サービス産業、クラウドサービス提供企業の関係

③，⑤の提供方法について，SIer (ソフトウェア会社) には，次のようなビジネスチャンスがありうる．

(ア) ユーザ企業が③の形でクラウドサービスを利用するための，構築作業をサポートする (構築の主体はユーザ企業)．構築後には，ユーザ企業はクラウドサービス提供企業と契約して，そのサービスを使用する．

(イ) 構築フェーズでは，⑤の形で，SIerが構築作業を請け負う (構築の主体はSIer)．構築後には，ユーザ企業が③の形でクラウドサービス提供企業と契約して，そのサービスを使用する．

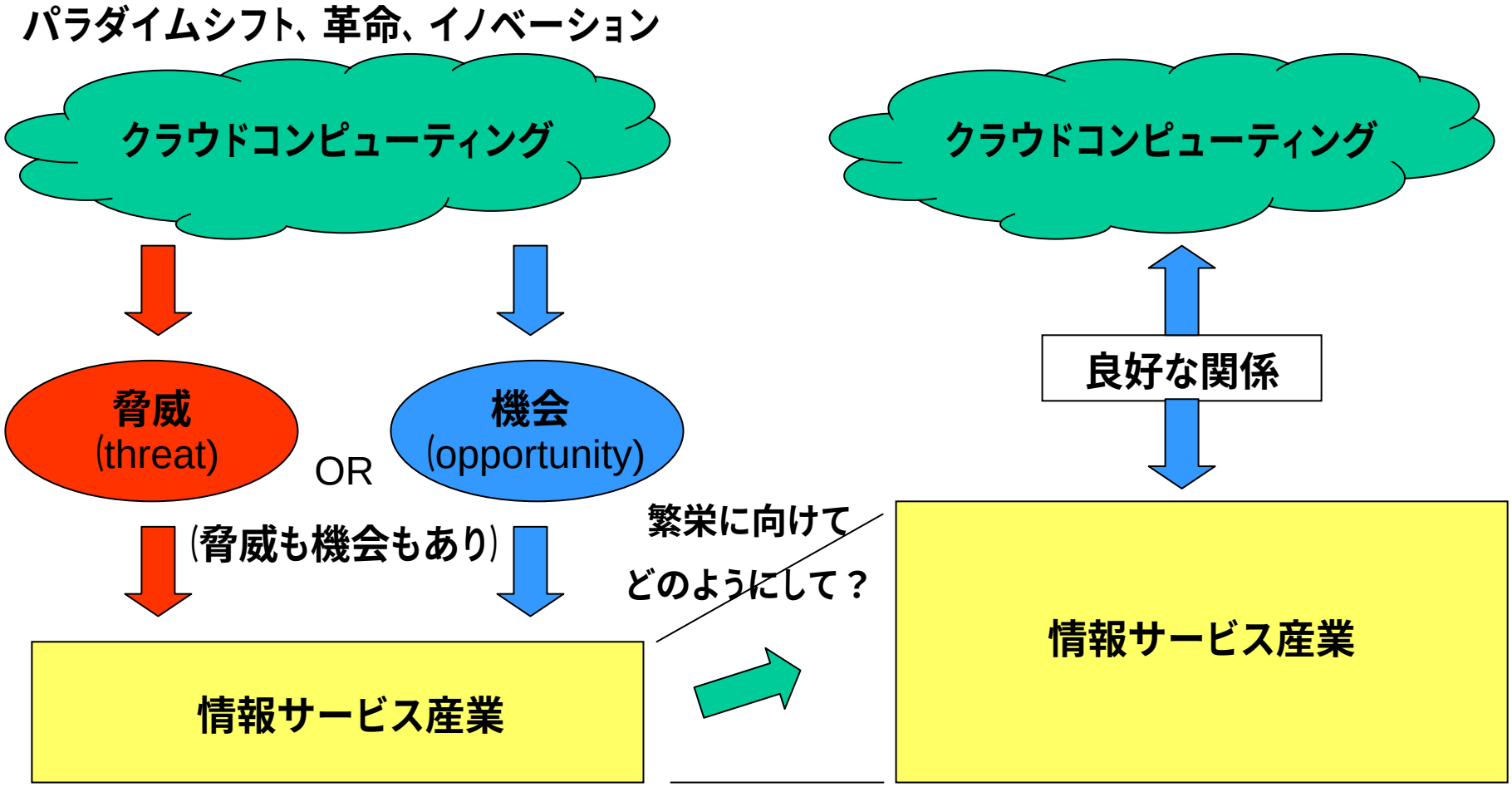
(ウ) 構築フェーズでは，⑤の形で，SIerが構築作業を請け負う (構築の主体はSIer)．構築後には，ユーザ企業が⑤の形でSIerと契約してクラウドサービスを使用する．SIerは外部のクラウドサービス提供企業と契約して，そのサービスを利用して，サービスの内容を追加して，ユーザ企業にサービスを提供する．いわば付加価値型のクラウドコンピューティング(Value Added Cloud Computing)．

(エ) 構築フェーズでは，⑤の形で，SIerが構築作業を請け負う (構築の主体はSIer)．構築後には，ユーザ企業が⑤の形でSIerと契約してクラウドサービスを利用する．SIerは外部のクラウドサービス提供企業のサービスを利用せずに，自社保有のサーバ (大規模になればデータセンター) を運用して，ユーザ企業にサービスを提供する．

(オ) ⑤において，SIerが自社保有のサーバを用いて，IaaS，PaaS，または既存サービスを持つSaaSサービスを提供する．

2．話題（2）情報サービス産業の繁栄へ

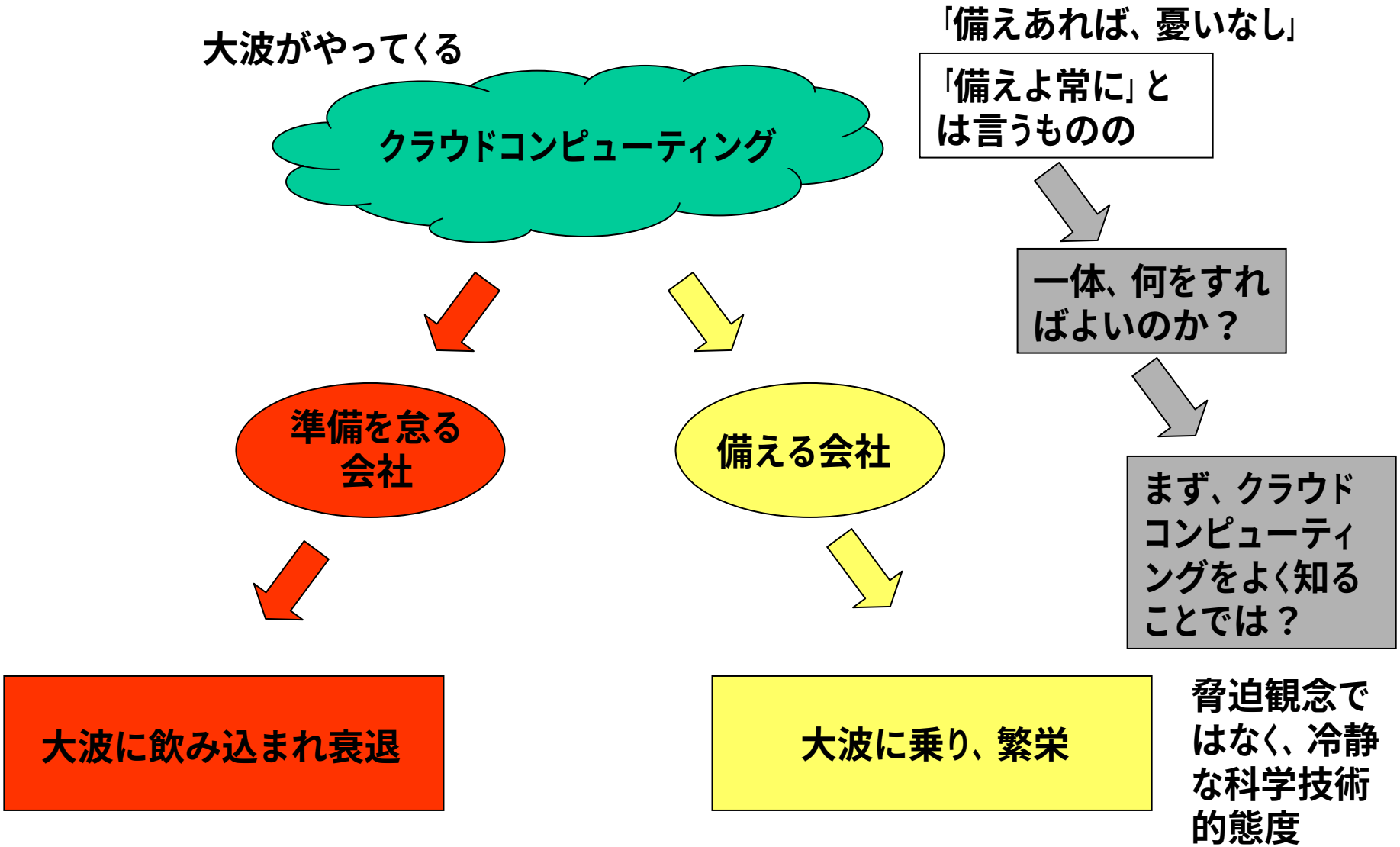
話題（2）これをチャンスと捉え、情報サービス産業の繁栄につなげることができるか



プライベートクラウド、ハイブリッドクラウドといったソリューションなど、どのような方法でチャンスに？

2．話題（3）どのように迎え撃つか

話題（3）各社がこれを、どのように迎え撃つか（どのように準備するか



2．話題（4）クラウドを良く知ること－1

話題（4）クラウドコンピューティングを良く知る：要素技術の一例

敵を知り、己を知れば、
百戦するも危うからず
（孫子）

情報処理を行う対象業務／顧客

適用する新技術（クラウドの技術）

競争相手／社会全般のトレンド

自社の強み、弱み

SWOT分析

Strength Weakness Opportunity Threat

まず，クラウドコンピューティングを知ること

システム開発によって → CCTP

2．話題（4）クラウドを良く知ること－2

話題（4）クラウドコンピューティングを良く知る：要素技術の一例

まず，クラウドコンピューティングを良く知ること

敵でもあり己でもある
クラウドコンピューティングの
要素技術を良く知ること

クラウドコンピューティングの要素技術

RDB→KVS（Key Valueデータストア）

ACID→BASEトランザクション

並列処理　Map Reduce処理

仮想化技術

スケールアップ→スケールアウト技術

2．話題（4）クラウドを良く知ること－3

話題（4）クラウドコンピューティングを良く知る：要素技術の一例

従来のビジネス系情報システム構築の主要キー技術

ドメイン知識（対象）

オブジェクト指向技術

リレーショナルDB技術

ネットワーク技術

プロジェクト管理技術

スケーラビリティ（スケールアウト）を求めるなら

リレーショナルDB技術

↓

KeyValueデータストア技術

影響大

利点

従来に比べ、桁外れに大きいスケーラビリティの実現

注意点

検索に制約が大きい（多様な検索ができない）
ACIDトランザクションが限定される→参照と更新を峻別する

使ってみて、実践的知見を蓄える

クラウドコンピューティングの要素技術（1）

ユーザ（クライアント）とプロバイダ（サーバ）間のインタフェース技術

ユーザとプロバイダの間は、通常は標準化されたインターネットの技術が使用されるが、専用の技術（プロトコル、ソフトウェア、ハードウェア）を使用するものもある。（Wikipediaから引用）

プロバイダ内部の技術

従来技術を利用可能とする考え

オープン標準に準拠したソフトウェアや、ユーザ数や処理量の増減に対応できる仮想化技術が使用される。（Wikipediaから引用）

クラウド特有の技術を前面に押し出す考え

Ｇｏｏｇｌｅなどは、スケーラビリティ確保のために自社独自技術を多様している。（Wikipediaから引用）

→この場合、クラウドの特長を最大にひきだすため、クラウド特有技術を利用して情報システムを構築するには、独自技術（クラウドを実現している仕組み）をよく理解し、その条件下で構築することが必要となる。

クラウドコンピューティングの要素技術（2）

データベース

リレーショナルDBからKeyValueデータストア（NoSQL）

「スケーラビリティ」を厳しく優先する場合、リレーショナルDBは使用できない

トランザクション

ACIDトランザクションからBASEトランザクションへ

上記のデータベースと関係、（スケーラビリティ優先の場合）

並列処理

MapReduce

仮想化技術

1つの物理的なCPUに複数の論理的なCPUを保持する

複数の物理的なCPUを1つの論理的なCPUとして働く→分散処理

従来型（非クラウド）情報システムを比較的容易にクラウド化する方法で、従来システムへの影響は少ない

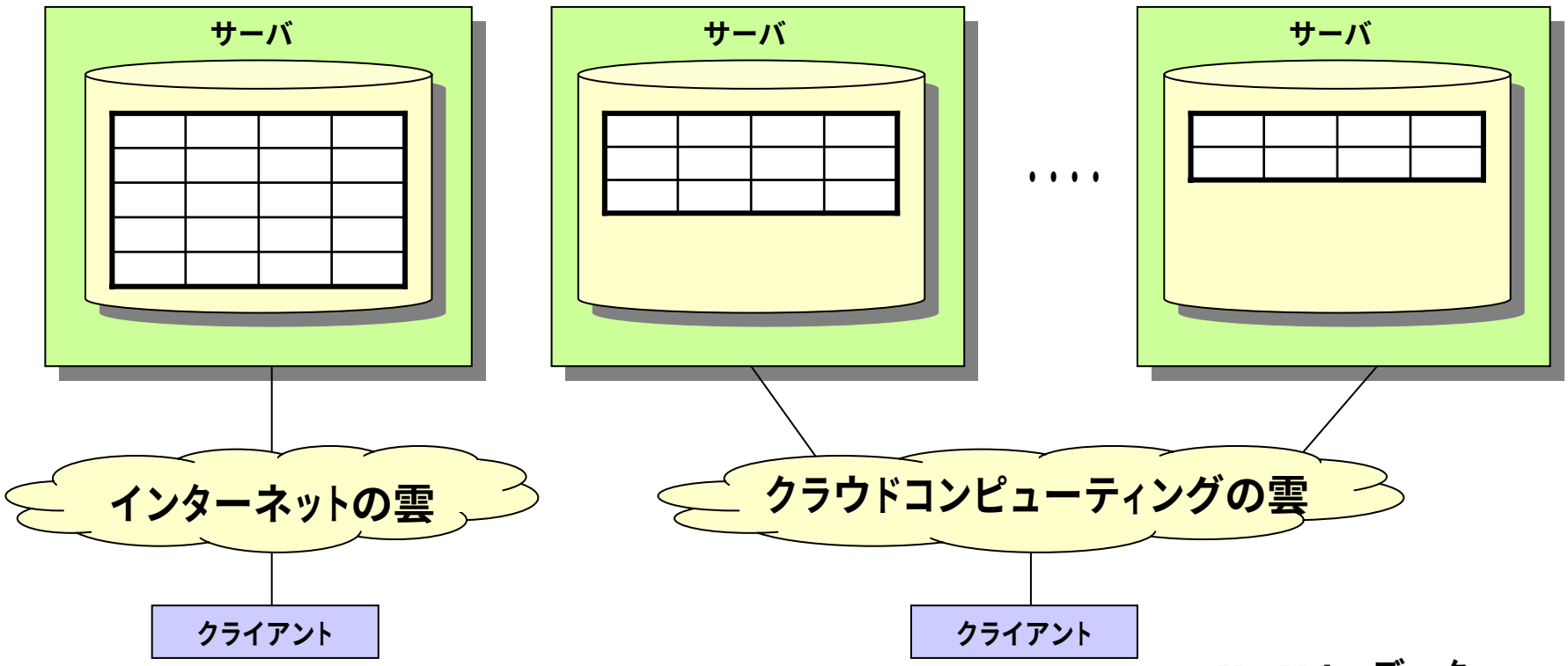
スケールアウト

担当サーバの割り当て、探索　（ハッシング）（コンシステントハッシング）

担当サーバへの到達、ルーティング（構造化オーバーレイ）

クラウドコンピューティングの要素技術 (3)

データベース リレーショナルDBからKeyValueデータストア (NoSQL)



KeyValueデータ

i 行 (タプル)	PK i	Vi1	Vi2	Vi3

基本的に、すべての行が1つのサーバに配置される (PK: Primary Key)

i 番目	Key i	Vi1	Vi2	Vi3

基本的に、この単位で、多数のサーバにバラバラに配置される

クラウドコンピューティングの要素技術 (4)

トランザクション

ACID (Atomicity, Consistency, Isolation, Durability)トランザクションから

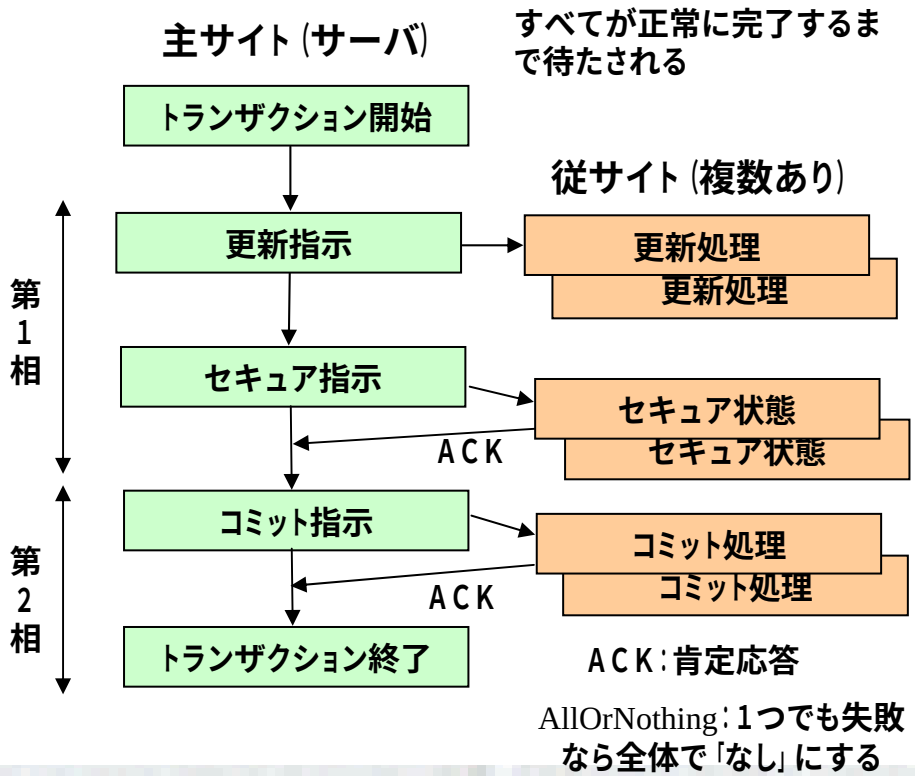
BASE (Basically Availability, Soft state, Eventually consistency)トランザクションへ

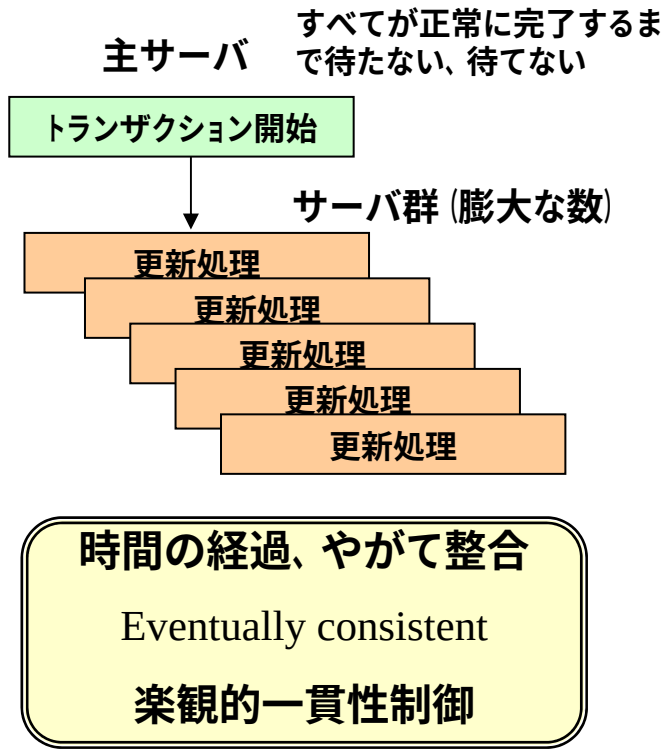
一貫性 (整合性) の維持

強い一貫性制御方式 ACID, 2PC, WAL

一貫性 (整合性) の維持

弱い一貫性制御方式 BASE



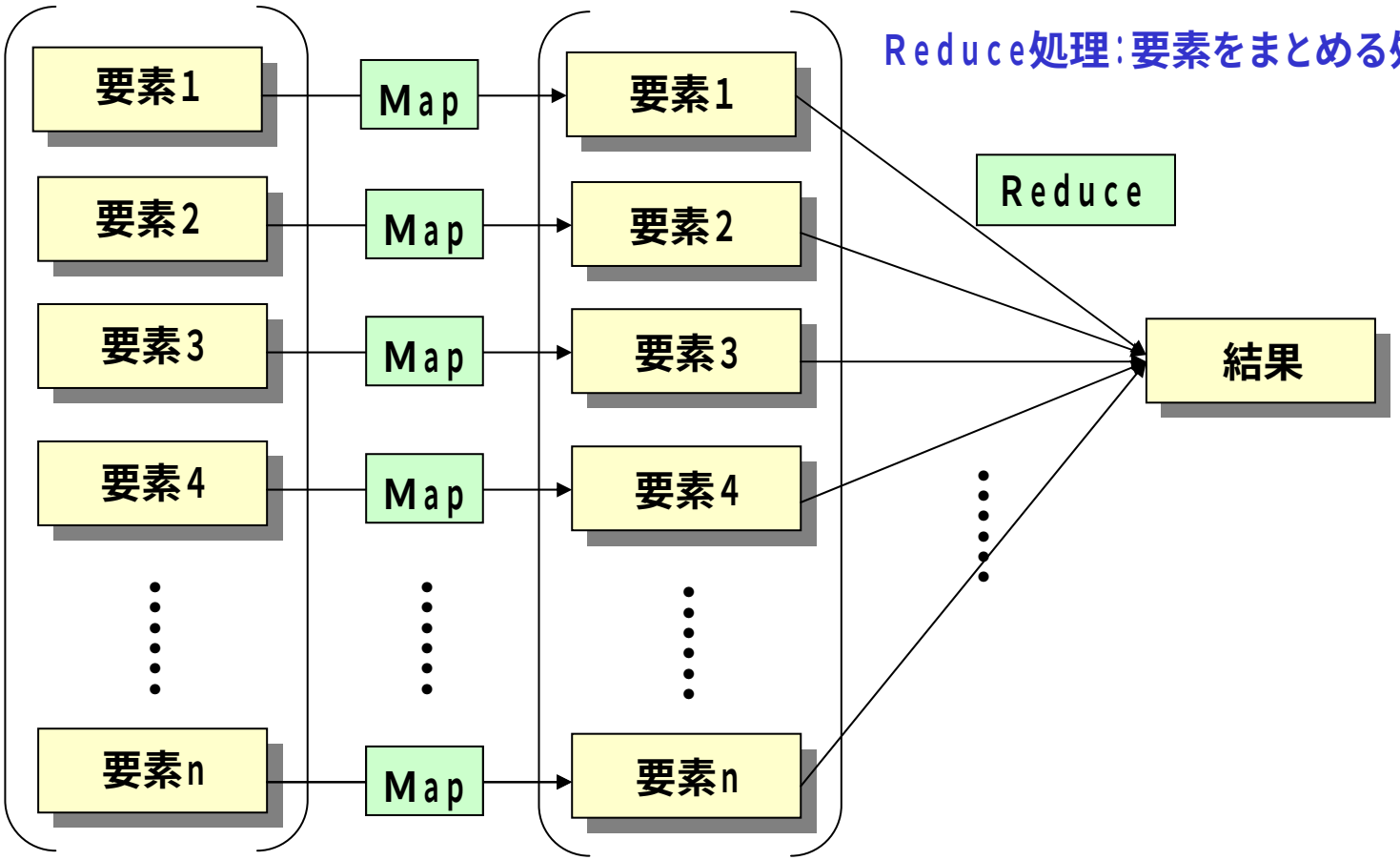


クラウドコンピューティングの要素技術（5）

並列処理

MapReduce

Map処理：要素ごとに行う処理

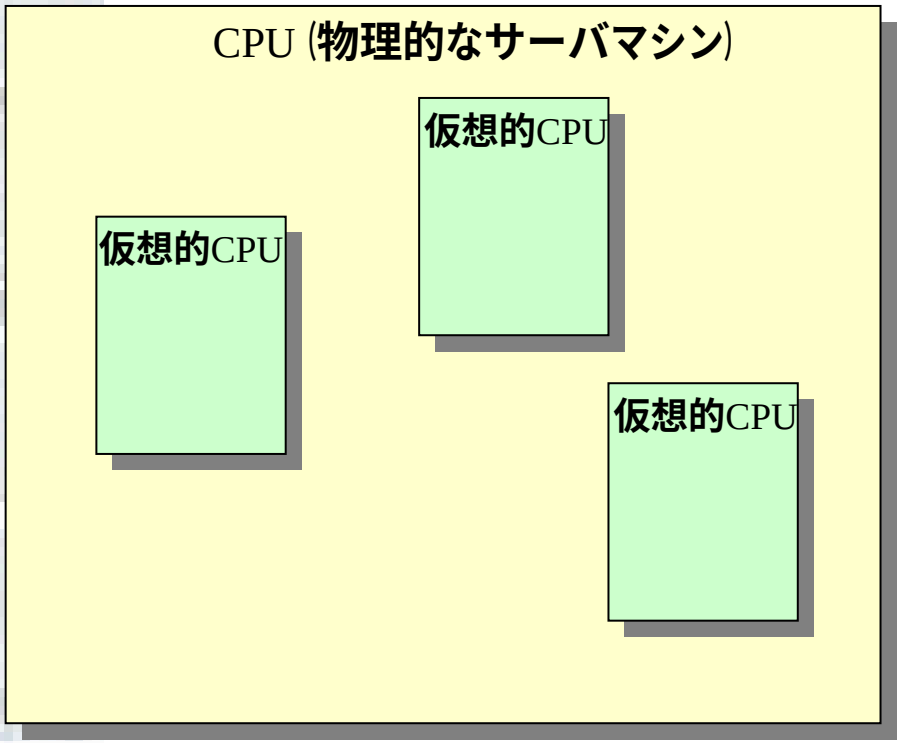


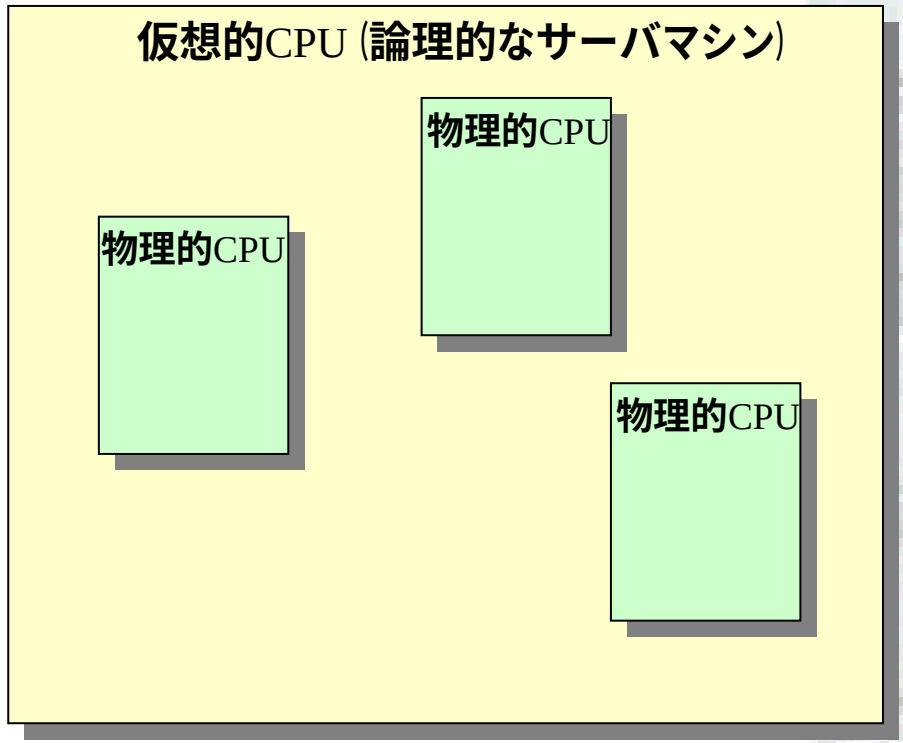
クラウドコンピューティングの要素技術（6）

仮想化技術

1つの物理的なCPUに複数の論理的なCPUを保持する

複数の物理的なCPUを1つの論理的なCPUとして働く→分散処理



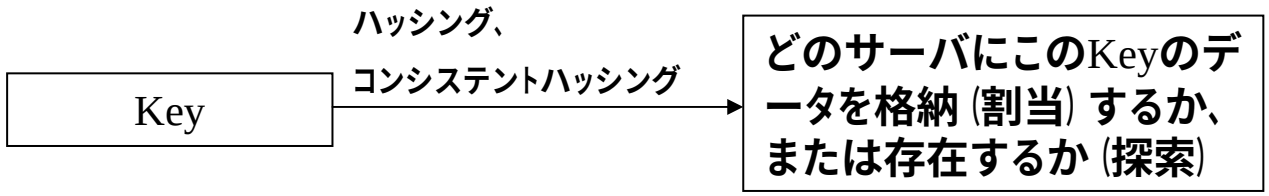


クラウドコンピューティングの要素技術（7）

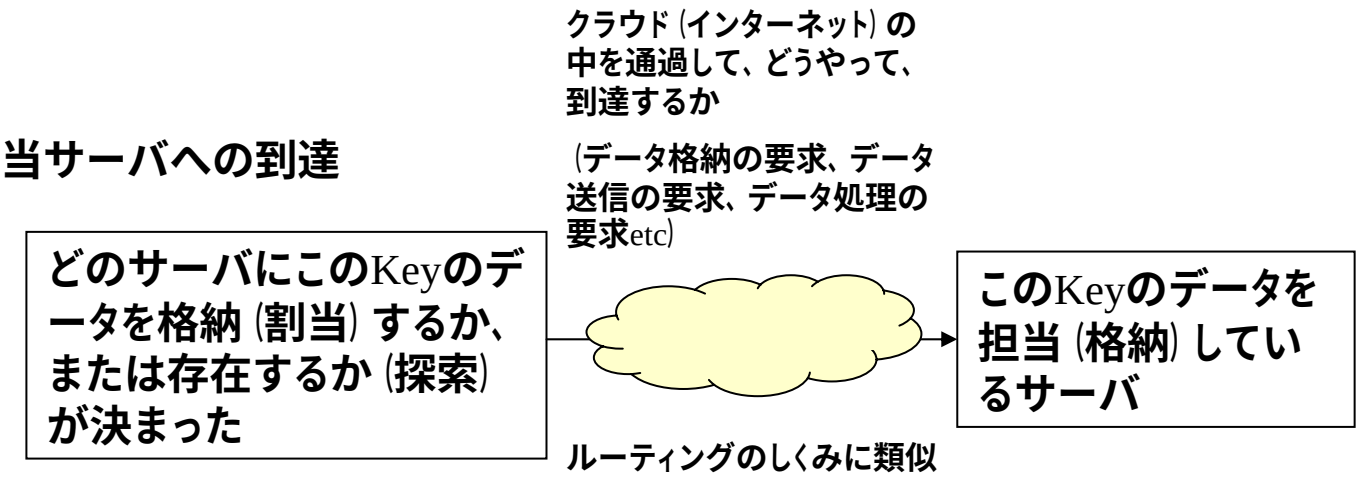
スケールアウト

担当サーバの割り当て、探索　（ハッシング）（コンシステントハッシング）
担当サーバへの到達、ルーティング（構造化オーバーレイ）

担当サーバの割り当て、探索



担当サーバへの到達



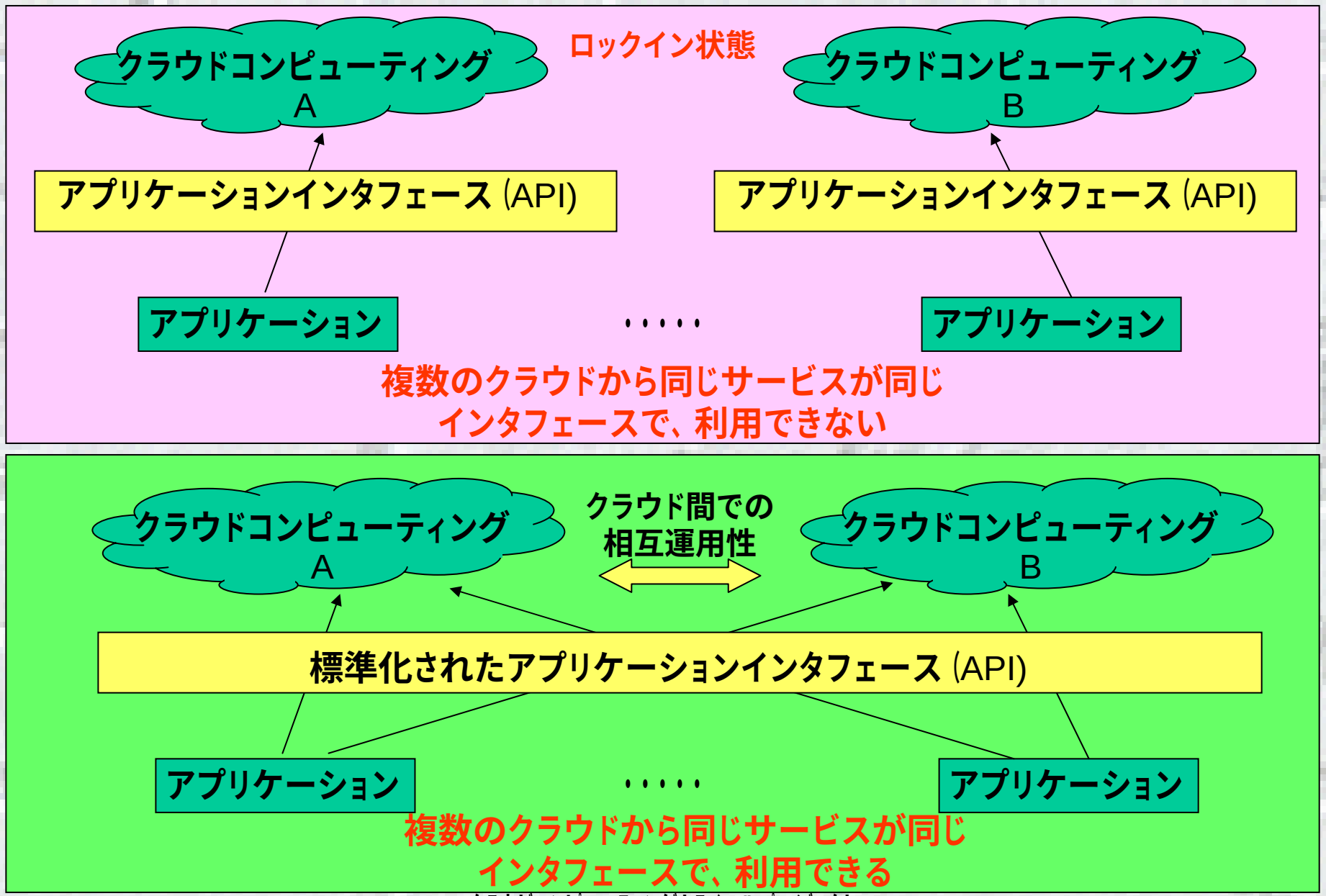
クラウドコンピューティングの要素技術（8）

（各社の比較、「日経SYSTEMS」2009年12月号から作成，要継続調査）

	Google	Amazon EC2	Windows Azure	Salesforce.com
KeyValue データストア	○ GFS（Google File System），Bigtable	○ Amazon Simple DB	○ Windows Azure Table	×
RDB	×	○ Amazon RDS(MySQL)	○ SQL Azure	○ Force.com Database
ACID トランザクション	△ 限定的な対応（同一ノード上のデータ間）	○	○	○
BASE トランザクション	○			
並列処理	MapReduce	Amazon Elastic MapReduce		

2．話題（5）標準化の必要性

話題（5）クラウドコンピューティングの標準化動向



2．話題（6）セキュリティ

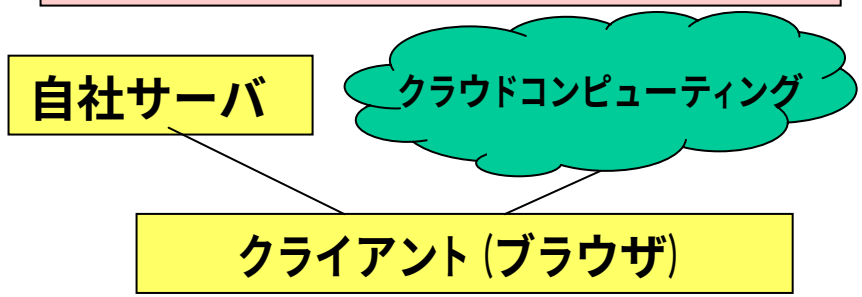
話題（6）セキュリティの問題

選択肢1：クラウドを使わない（使えない）

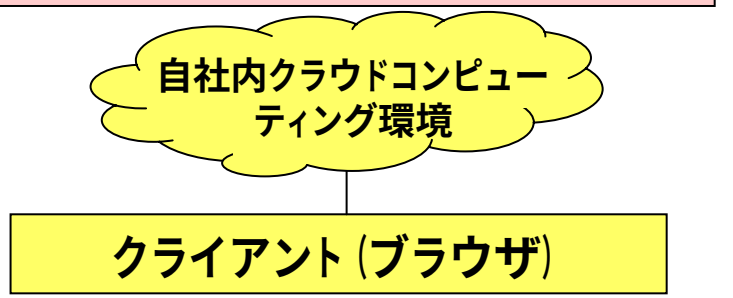
過去の成功に安住、保守主義、波に乗り遅れる恐れ

選択肢2：セキュリティが重要なデータや処理は自社のコンピュータで行い、セキュリティに問題のないデータや処理はクラウドを使う

ハイブリッドクラウドというソリューション



プライベートクラウドというソリューション



選択肢3：セキュリティが重要なデータや処理も、セキュリティに問題のないデータや処理もクラウドを使う

認証、暗号化、査証、内部統制etc

サービス提供者を信頼し、契約SLA (Service Level Agreement)で縛るしか方法はないのだろうか？技術的解法は？

2．話題（7）オープン技術との関係

話題（7）ＯＳＳ（オープンソースソフトウェア）との付き合い方

アーキテクチャ：Webアプリケーション

Webサーバ：Apache

Applicationサーバ：Tomcat

Webサービス：WSDL、Rest

Programming：HTML、Java、JSP、Servlet、PHP、Ruby、・・・・

開発環境（IDE）：Eclipse

DataBase：MySQL,Postgresql

Network：TCP／IP

OS：Linux

最近，注目されている

「日経コンピュータ」2010年9月15日

クラウドコンピューティング

がもたらす影響？

きたるべきクラウド時代に備え、

ＯＳＳの関連では、

どのような準備をすべきか？

- クラウド向けデータストア
- OSSのHadoopなど

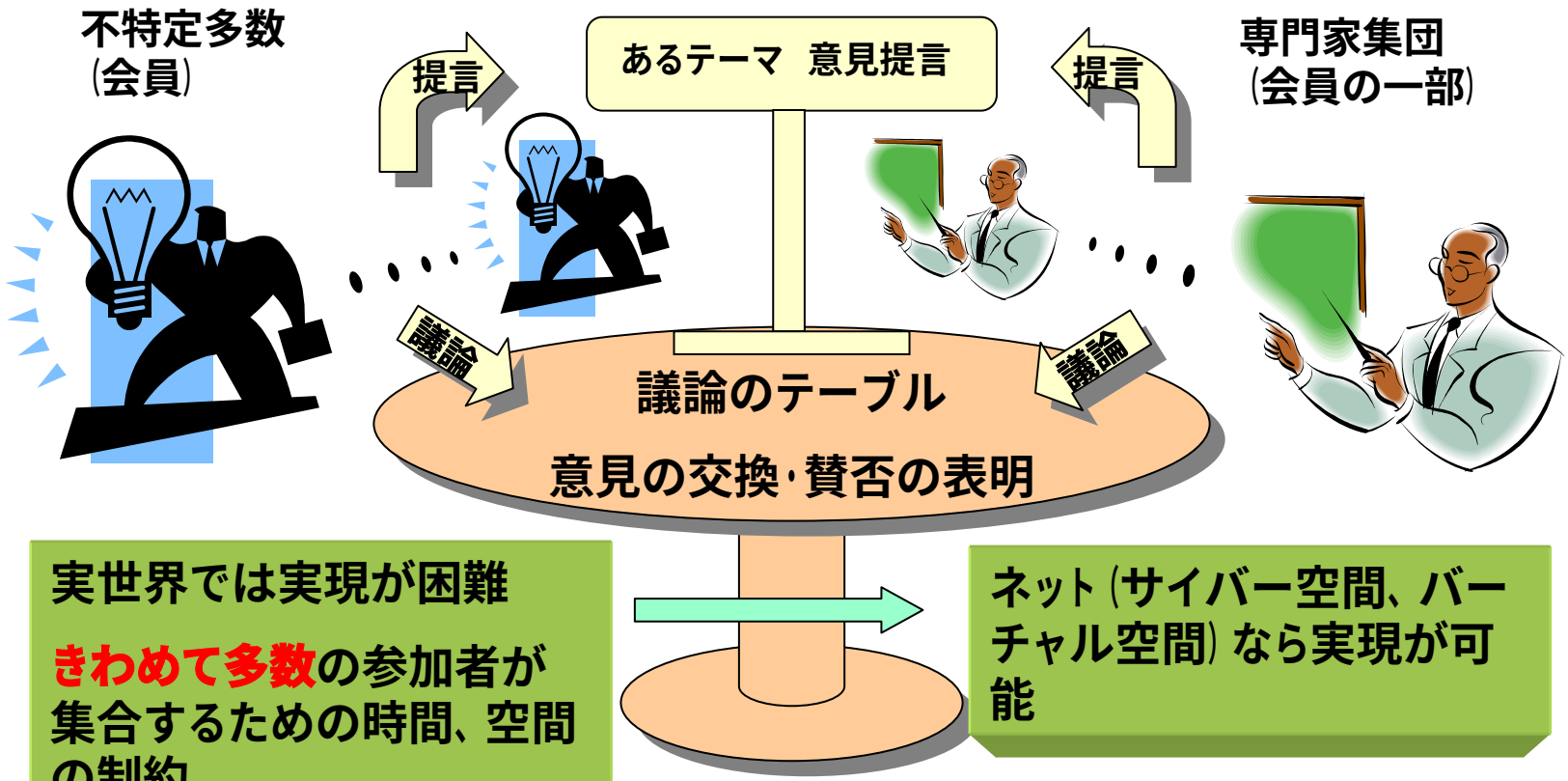
3．意見収集･形成システム

今回の発表の主旨ではなく，今回作成したアプリケーションの例として紹介

ニーズ

- **だれもが、なにかの意見**をいいたい。（積極的な社会への参画、民主主義の進化）（意見の提案）
- その意見がまっとうならば、**民意（世論）**になっていくべきである。（民意の形成）
- 自由に、多面的（倫理的、経済的、人道的、等々）な観点から、**賛否の表明**と十分な**議論**が行われ、その上で民意が形成されるべきである。
  同一人物が、議論の結果、賛否を翻すこともありうる。
- 少数による専門家の意見は尊重するにしても、**きわめて多数の意見を集約する**ほうが的を得ていることが多い。（例：不特定多数にアウトソーシングするクラウドソーシングも主旨は同じ、この場合のクラウドはcloudではなくcrowd）
  スコット･ペイジ、『「多様な意見」はなぜ正しいのか』
- このような**議論の場**がほしいが、現実世界では、不可能に近い。インターネット（サイバー空間、バーチャル空間）なら、実現の可能性がある。
  （経済産業省の試み：アイデアボックスhttp://www.meti.go.jp/policy/it_policy/open-meti/）

3．意見収集･形成システム



実世界では実現が困難

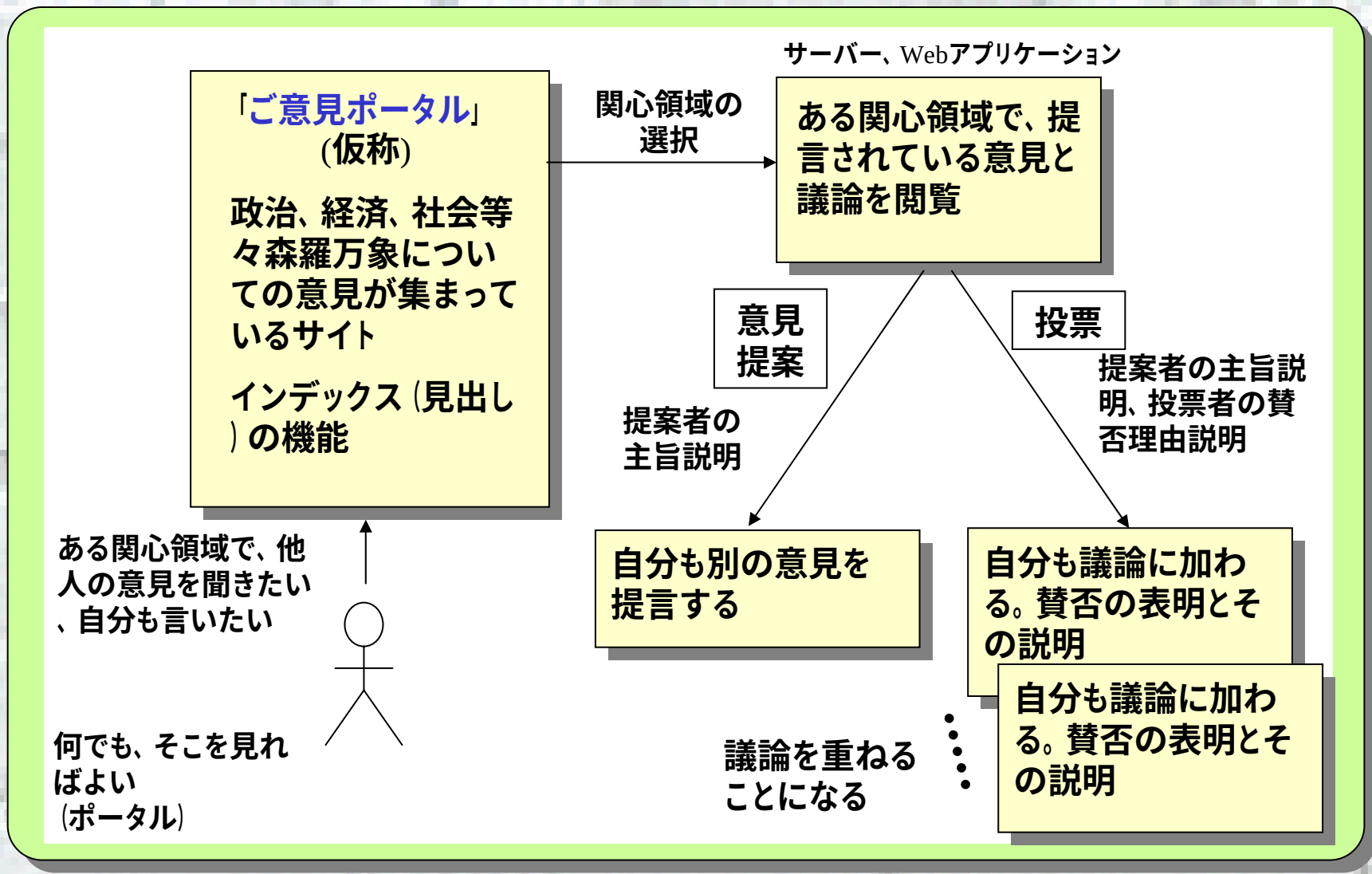
きわめて多数の参加者が集合するための時間、空間の制約

ネット(サイバー空間、バーチャル空間)なら実現が可能

3．意見収集･形成システム

- **意見提案（提言）をネット上に公開**
  登録された個人なら、だれでもが何かについて、自分の意見を述べることができる。
  「意見提案テーマ」、「意見提案の説明」、参考資料の公開
  意見のまっとうらしさをある程度担保するため、Aランク会員資格をもつ会員のみ意見提案ができる。
- **提言された意見提案に対する投票**
  だれでもが賛否の意思表明（投票）と議論（賛否理由の説明）をおこなうことができる。だれでもが議論に参加できるが、賛否の層別集計（Aランク会員、Bランク会員、Cランク会員）を行う。
- **議論による創発効果**
  議論の中から、創造的な案がでてくることが期待される。
- **「意見収集･形成システム」はその仕組みを提供する。**

3．意見収集･形成システム



3．意見収集･形成システム

いままでのしくみ

･ブログ
･ツイッター
･質問システム･アンケートシステム
･主に国、地方行政のパブリックコメント
･経済産業省アイデアボックス

いままでのしくみとどこがちがうのか？

･どこに行けばよいか明確である（ポータル効果）
（森羅万象、なんでも受け付ける、関心領域が広い、多様な人が集まっている）
･だれでもが提言もできるし、議論もできる（双方向的）
･ある提言に対する他のかたがたの会員ランク別の賛否状態がわかる（投票の集計）
･常時、開かれている（個別には締切を設けるが、長期間開示する）、いつでも意見が言える

3．意見収集･形成システム

オバマ政権のOpenGovernmentの考え

http://www.whitehouse.gov/open/

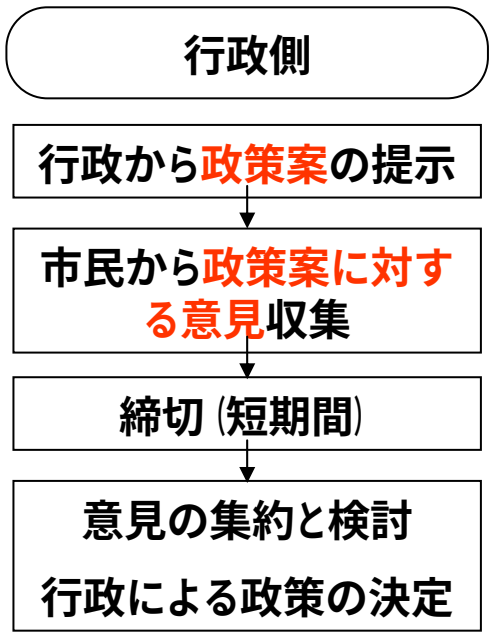
- transparency (**透明性**)
  政府のもっているデータを公開する
  政府の考えていることを公開する
- public participation (**国民の参画**)
  政府は政策を国民といっしょになって考えていく
  政策を政府と国民が双方向で議論できる
- collaboration (**協働**)
  政府と国民がいっしょになって、国の運営を考える

3．意見収集･形成システム

従来のパブリックコメントの場合

既にまとまった形の政策案が上からおりてくる（行政から市民へ，棚からボタ餅方式）

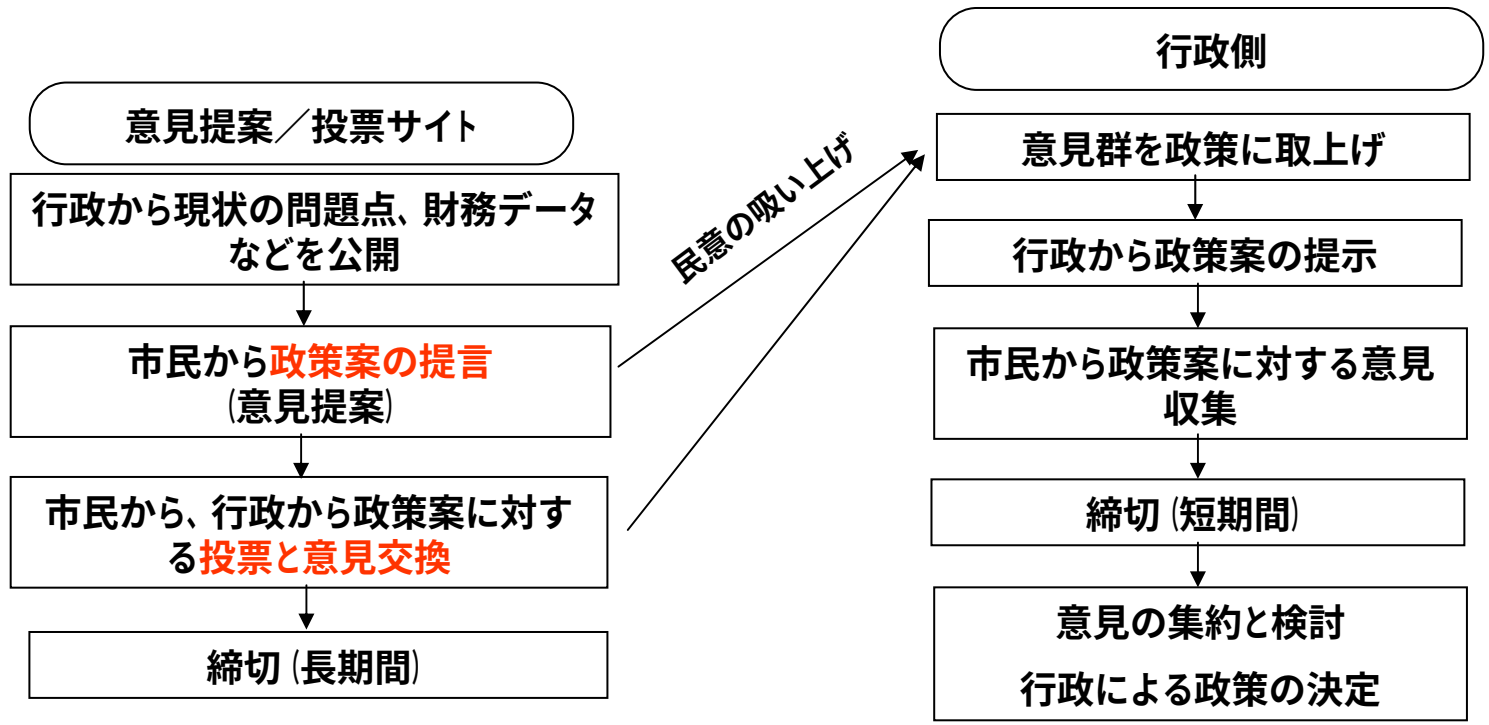
半分単方向的、政策案について市民が意見を述べることができる点では、一定の進歩あり



3．意見収集･形成システム

意見収集･形成システムの場合

行政が政策案を作成する前の段階で市民の意見を吸い上げ、それをベースに政策案を立案する
（ボタ餅を市民がつくる、作られたボタ餅がよさそうならば、行政がピックアップして、棚からおとす）
双方向的



3．意見収集･形成システム

従来のパブリックコメント

- 行政側からだされた提案に対して、国民（市民）が賛成／反対の意見を述べる。
- 集まった意見を行政側がまとめ、行政に反映する。
- 提案掲示期間は短い。
- 十分な意見交換･議論ができない。

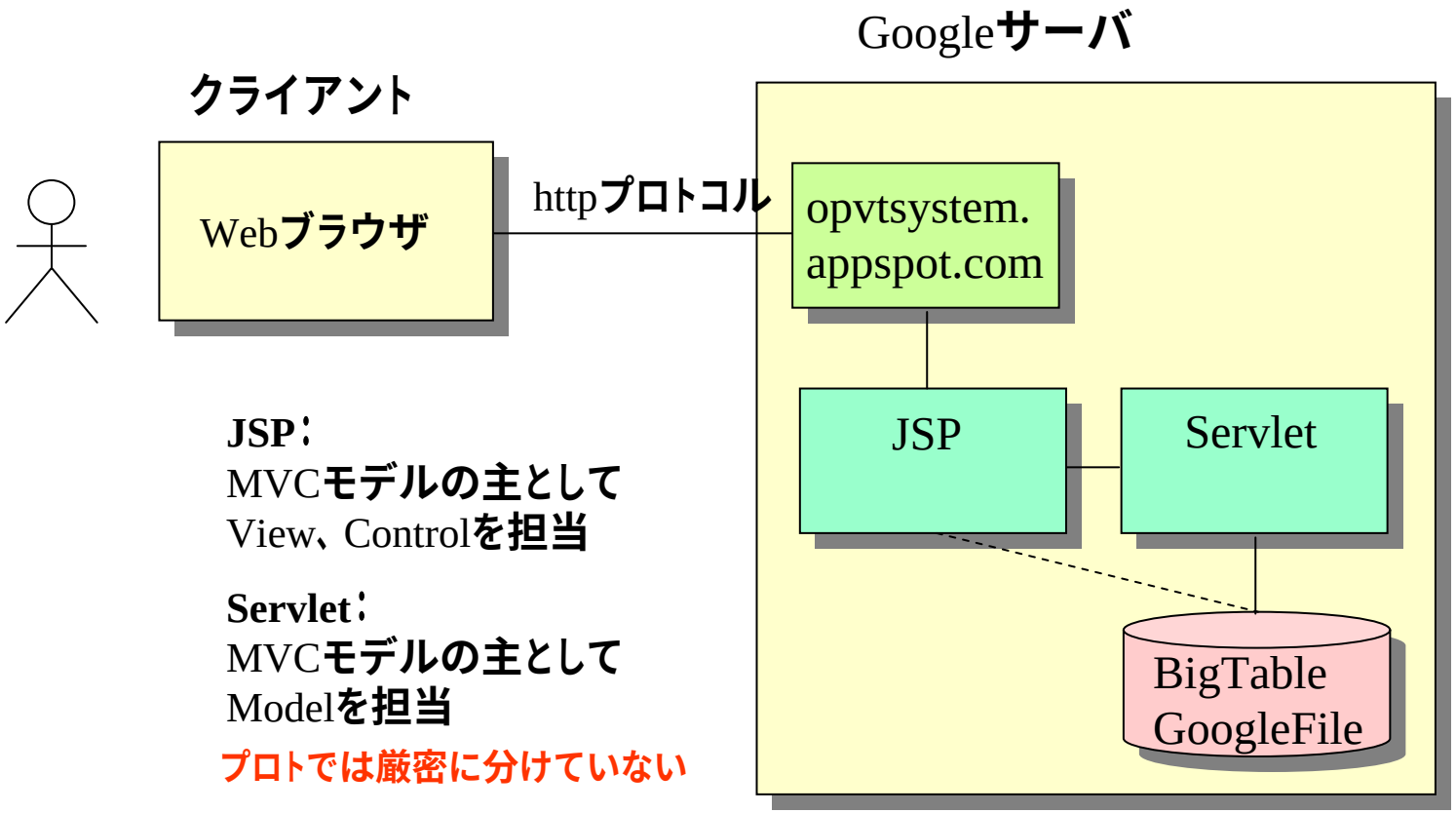
意見収集･形成システム

- 国民（市民）からだされた提案に対して、国民（市民）が賛成／反対の意見を述べる。
- 集まった意見を、行政が参考にして、行政としての提案をまとめる。
- 以下、従来のパブリックコメントのプロセスに移行する
- 意見交換の期間が長い。
- 意見交換･議論を十分にできる。

4．プロトタイプ

システム（ソフトウェア）構成

http://opvtsystem.appspot.com

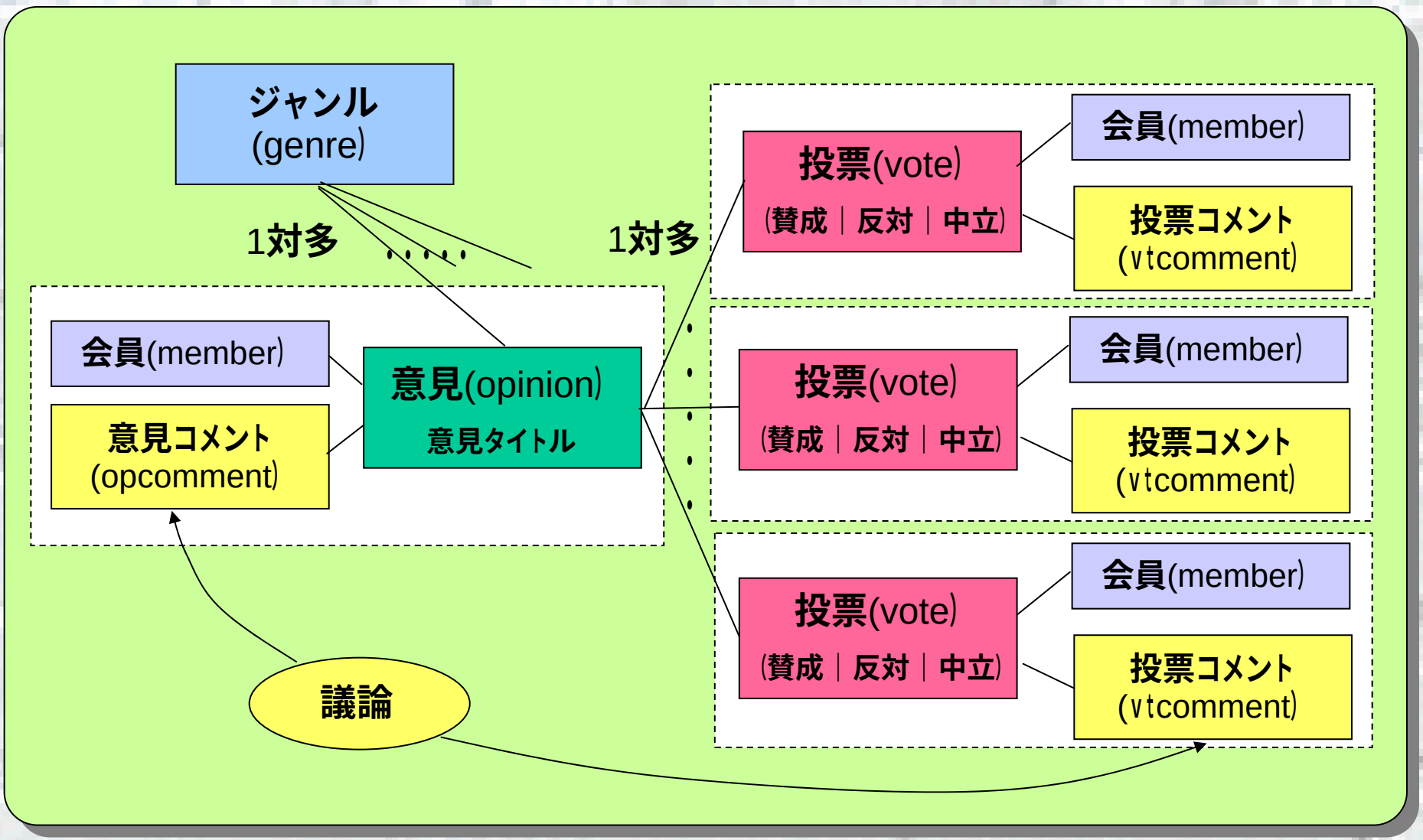


JSP：
MVCモデルの主として
View、Controlを担当

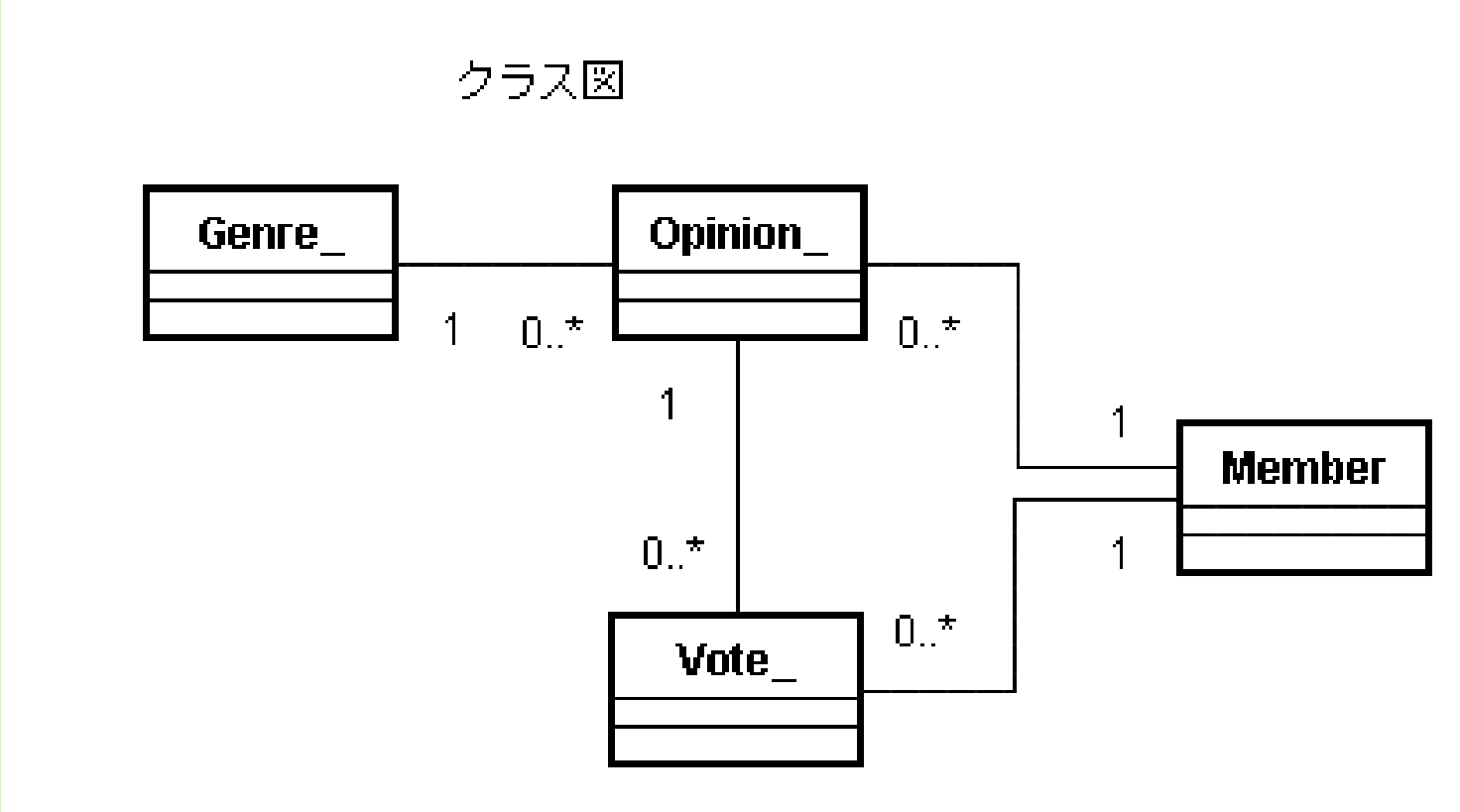
Servlet：
MVCモデルの主として
Modelを担当

プロトでは厳密に分けていない

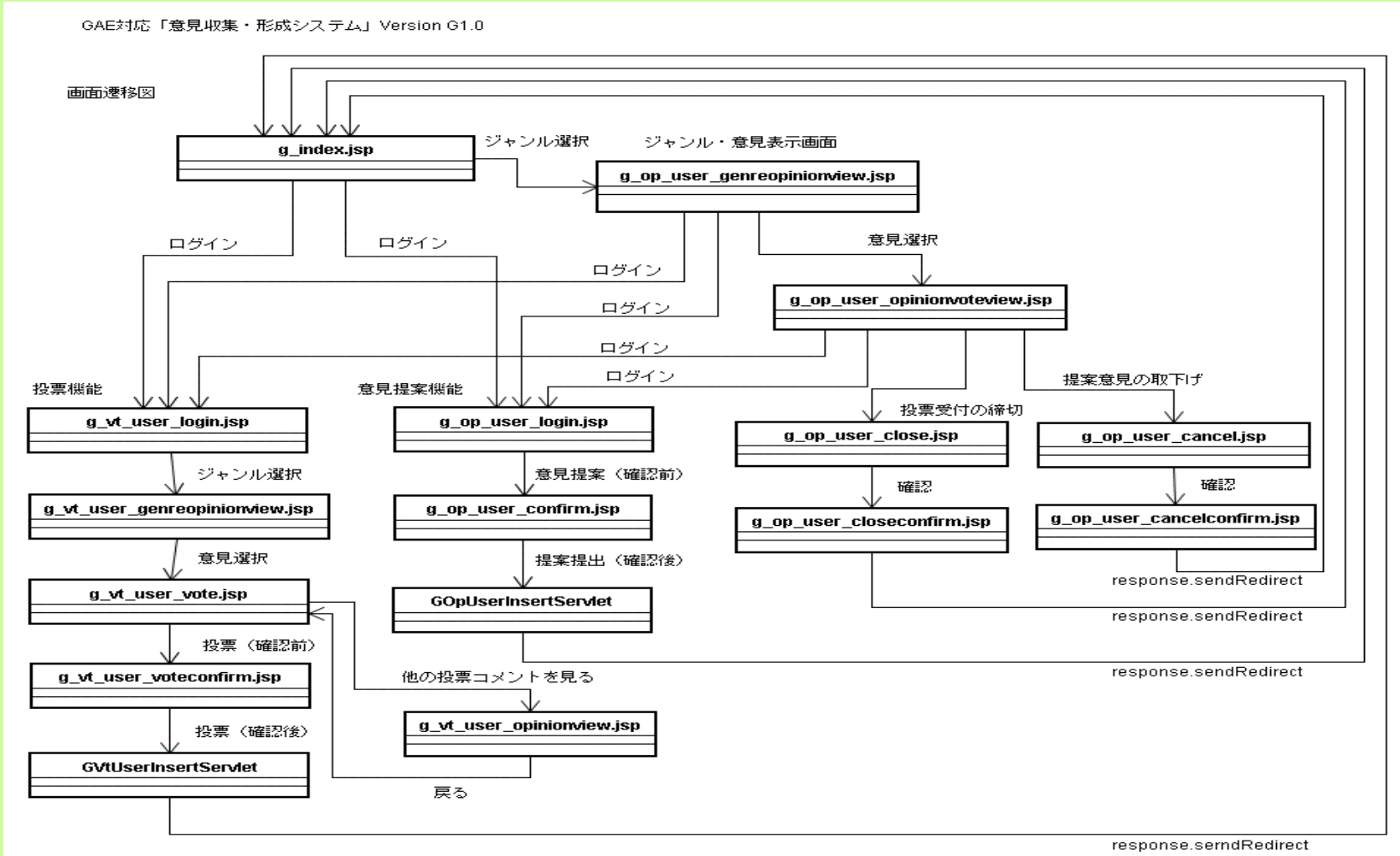
4．プロトタイプ（意見と投票）



4．プロトタイプ（主要クラス）



4．プロトタイプ(画面遷移)



4．プロトタイプ（データストア）

(1) **会員テーブル**（A**ランク**，B**ランク**，C**ランク**）

テーブル（クラス）名：	member	（**主キー**：memberid）	
属性：			
・**会員**ID	(memberid)	String	primary key
・**パスワード**	(password)	String	
・**ランク**	(rank)	String	
A**会員なら**”1”，B**会員なら**”2”，C**会員なら**”3”			
・**ハンドルネーム**	(handlename)	String	
・**住所**	(address)	String	
・**名前**	(name)	String	
・**性別**	(gender)	String	
・**職業**	(profession)	String	
・**誕生年月日**	(birthdate)	Date	
・**電子メールアドレス**	(emailaddress)	String	

4．プロトタイプ（データベース）

(2) **ジャンルテーブル**

テーブル(クラス)名：　genre　（**主キー**：genreid）

属性：

・**ジャンル**ID	(genreid)	String	primary key
2**レイヤー**（**上位2文字，下位2文字**）			
・**ジャンル名**	(genrename)	String	
・**意見総数**	(opiniontotalnumber)	long	
ジャンルごとの提言された意見の総数(GAE**では整数は**long)			

4．プロトタイプ(データベース)

(3) 意見提案テーブル－1

テーブル(クラス)名：　opinion(主キー：意見ID＋ジャンルID)

属性：

・意見ID	(opinionid)	String　primary key
genreとopinionnumberをStringにしたものとを「－」をはさんで連結		
・意見ナンバー	(opinonnumber)	long
ジャンルIDごと，意見提案の発生のたび，1ずつ加算		
・ジャンルID	(opgenre)	String
意見提案されているジャンルのgenreid		
・提案者	(proposer)	String
意見提案しているA会員のmemberid		
・提案者ハンドルネーム	(proposerhandlename)	String
・投票総数	(votetotalnumber)	long
当該意見に対する投票総数		
・意見タイトル	(optitle)	String
提案テーマ		
・意見コメント	(opcomment)	String
提案の背景、理由など		

4．プロトタイプ（データベース）

(3) 意見提案テーブルー2

・提案日付	(opopendate)	Date
・締め切り予定日付	(opclosedate)	Date
・締め切り日付	(opcloseddate)	Date
・提案状態	(opstate)	String
1：有効，2：投票受付の締め切り済み，3：提案者による取下げ，4：管理者取り消し（公序良俗に不適切なテーマなど）		
・無効理由，未使用	(oprejectreason)	String
・最新時点での賛成数（A会員）	(amemberyes)	long
・最新時点での反対数（A会員）	(amemberno)	long
・最新時点での中立数（A会員）	(amemberneutral)	long
・最新時点での賛成数（B会員）	(bmemberyes)	long
・最新時点での反対数（B会員）	(bmemberno)	long
・最新時点での中立数（B会員）	(bmemberneutral)	long
・最新時点での賛成数（C会員）	(cmemberyes)	long
・最新時点での反対数（C会員）	(cmemberno)	long
・最新時点での中立数（C会員）	(cmemberneutral)	long

4．プロトタイプ（データベース）

(4) 投票テーブル－1

テーブル（クラス）名：　vote

属性：

・投票ID	(voteid)	String
opinionidとvotennumberをStringにしたものとを「－」をはさんで連結		
・投票ナンバー	(votenumber)	long
ジャンルID，意見IDごとに，投票のたびに，1ずつ加算		
・意見ナンバー	(opinionnumber)	long
・ジャンルID	(opgenre)	String
・投票者ID	(voter)	String
A会員ID，B会員ID，C会員IDのmemberid		
・投票者ランク ＊	(voterrank)	String
・投票者ハンドルネーム ＊	(voterhandlename)	String

＊：JOINが使えないため、正規形をくずし、memberにもっている属性をvoteにももたせる

4．プロトタイプ(データベース)

(4) **投票テーブル－2**

・**投票内容**	(votecontent)	String
1：**賛成**, 2：**反対**, 0：**中立**(**プロトでは**3：**中立**)		
・**投票状態**	(votestate)	String
1：**有効**, 2：**同一投票者の投票内容変更による無効**, 3：**管理者による無効化**		
・**無効理由**, **未使用**,	(voterejectreason)	String
・**投票コメント**	(votecomment)	String
賛否の理由など		

4．プロトタイプ（開発順序）

筆者の行ったGAE開発
（準備－1）

- **クラウドサービス提供者の選択**

- Google**を選択**
  - **クラウドサービスの典型である**
  - **無料でトライできる**
  - PaaS**が用意されている**

- Google Apps**と**Google App Engine**の違い**
  - Google Apps**は**SaaS（**アプリケーションが提供されている**）
  - Google App Engine**は**PaaS（**アプリケーション構築のプラットフォームが提供されている**

4．プロトタイプ（開発順序）

筆者の行ったGAE開発
（準備－2）

- Google**アカウントの取得（メールアドレスが必要）**
  https://www．Google.com/accounts/
- GAE**へのサインアップ（携帯電話またはPHSのメールアドレスが必要）**
  Google**から認証コードが送られてくる（7桁の数字）**
  それをEnter Account Code**に入力して，サインアップ完了**
  http://appengine.google.com/
- Create an Application**でアプリケーション**ID xxxxxxxx **の入力**
  xxxxxxxx**が既に登録されていれば，入力しなおす．**
  これがhttp://xxxxxxxx.appspot.com**でアクセスするときの**ID**となる**

- Java**のインストール**
- Eclipse**のインストール**
- Google Plug in**のインストール**

4．プロトタイプ (開発順序)

筆者の行ったGAE開発

- 非クラウド版 (Java, Servlet) の「意見収集･形成システム」の開発，ローカルでの確認
  仕様確定，および通常のWebアプリケーションとしての動作確認のため
- オンラインチュートリアルGreetingシステムの再現，Googleでの動作確認
  GAEへのdeploy，Googleでの動作確認のため
- これをベースに「書評システム」の開発，Googleでの動作確認
  GAEのKeyValueデータストアの使い方に習熟するため
- 「意見収集･形成システム」で用いるデータストア用ユーティリティプログラムの開発

- 「意見収集･形成システム」で用いるアプリケーションプログラムの開発
  ローカルで確認，予稿執筆時点
- 「意見収集･形成システム」のdeploy
  これが現時点の状態

5．プロジェクトの現在までの結果

(1) プロジェクトの技術的目的･目標に対して

- KeyValueデータストアの実際の使い方については，目的を達したが，その他のクラウド技術については，実験を行うことができなかった．
  例えば，弱い一貫性(Eventually Consistency)管理とBASEトランザクション処理，並列処理(MapReduce)について，ビジネス系として今後，利用方法の習得と実証を行う必要がある．
- クラウドサービス提供者の評価についても，次の(2)で述べる開発チーム数が多くなかったので，プロジェクトで当初意図した目的は達成できなかった．

5．プロジェクトの現在までの結果

(2) プロジェクト推進について

- プロトタイプ開発を実施するか否かは，参画者の自主的判断にお任せしていたので，仙台地区以外では，開発活動は極めて低調であった．この種のボランタリー活動の限界を感じた．しかし，プロジェクトは失敗であったかと問われれば，そうではなかった．実際に開発に携わった技術者には，**確実にクラウド技術**が身についたであろうし，プロジェクトとして同じ目的に向かったということで，質問や協力しあうことにより，開発の励みになり，開発も促進された．
- **仙台地区での推進方法**が今後の参考になろう．まだまだ検討すべき積み残しも多く，プロジェクトが仙台地区を中心として継続されることを望むものである．

5．プロジェクトの現在までの結果

(3) プログラミング

- プログラミング上では，GAEが扱うデータタイプに制限があること，nullの扱い，JSP→サーブレット，サーブレット→JSPのパラメータの受け渡し，Error500の場合の原因追求方法，デバッグ方法など不慣れな面があり，困難性もあった．
- 筆者の方法は，全体的には，データストアおよび図3の画面遷移について，印刷物を常時かたわらに置き，プログラミングを行った．トラブルで多かったのは，上記のプログラム間でのパラメータの受け渡しの間違い，”｝”のもれといったGAE特有の問題ではなく，通常のプログラミング上のミスが多かった．GAE特有の問題として，KeyValueデータストアの使い方については，次ページのようなデータストアアクセス方法を何種類かトライすることにより，使用方法を比較的容易に習得できた．それ以外には文字コードについて，注意すべき点があった．
- プログラミング上のミスは，4.2で述べたように，非クラウド版システムを動作させてからクラウド化する方法をとったにも関わらず，この傾向がでたということは，4.2の方法をとらなかった場合，一層苦労したことと思われる．

5．プロジェクトの現在までの結果

```
//ジャンルごとの意見に対する投票コメント全件表示
//JDBCドライバを登録
Class.forName("org.hsqldb.jdbcDriver");
//データベースコネクション文字列を作成
String strConn="jdbc:hsqldb:hsql://localhost";
//コネクションオブジェクトを取得
Connection conn =
DriverManager.getConnection(strConn, "sa", "");
//ステートメントオブジェクトを取得
Statement stmt = conn.createStatement();
//SQLコマンドを作成
String query = "select * from vote where opgenre="
             + "'"+opgenre+"'"
             + " and opinionnumber="
             + opinionnumber
             + " order by votenumber";
//queryを発行してリザルトセットを取得
ResultSet rs = stmt.executeQuery(query);
//全件とりだす
while(rs.next()){
(処理)
}
stmt.close();
conn.close();
```

←従来のＲＤＢ
↓ＫｅｙＶａｌｕｅデータストア

```
//ジャンルごとの意見に対する投票コメント全件表示
//パーシステンスマネージャのオブジェクトを取得
PersistenceManager pm =
PMF.get().getPersistenceManager();
//query文字列を作成
String query ="select from " + Vote.class.getName()
             + " where opgenre ='"+opgenre+"'"
             + " && opinionnumber=="+opinionnumber
             + " order by votenumber ";
//queryを発行して，オブジェクトのリストを取得
List<Vote>votes=(List<Vote>)
pm.newQuery(query).execute();
//全件とりだす
for(Vote v:votes){
(処理)
}
pm.close();
```

5．プロジェクトの現在までの結果

(4) システムのdeploy

- **予稿執筆時点では，システムの**deploy**は行っていなかったが，現時点では，**deploy**を完了している．**
  - http://opvtsystem.appspot.com/
- deploy**でのノウハウは，**Eclipse workspace**の文字コードの注意で，筆者は，**
  - Eclipse workspace**のもともとの文字コードは**MS932**としていた（無意識に）**
  - **プログラムコードを作成，書き換えるときには，**UTF-8**とする**
  - **終われば，**MS932**にもどす**
  - Eclipse**から**Google**に**deploy**する**

というプロセスで行った．もっと簡潔な方法があるかもしれない．

(5)ソースコードの公開

下記URL**で現時点での最新バージョンを公開している．**

- http://www.isem.co.jp/documents/cloudcomputing/workspaceforGAEI2_20100917.zip

6．今後の課題

- クラウドコンピューティングトライアルとしての今後の課題
- トライアルの輪を広げること
  - 参加メンバーのより多くのかたが，システム構築をトライしてみること
    今回のソースコードを見ていただき，さほど従来との違いがないことを実感され，構築に着手されることを望む
  - 今回は，クラウドサービス提供者Googleだけであったが，トライ者が多くなるにともなって，クラウドサービス提供者も広げること（OSSのHadoopなどもふくめ）
  - アプリケーションに別のアプリケーションを構築すること
- 現在までの結果をもちより，評価すること
- 次年度の計画をたてること

参考文献

- [1] http://www.meti.go.jp/committee/materials2/downloadfiles/g100423a07j.pdf
- [2] http://www.ieice.org/~swim/jpn/CCTPPlan.pdf
- [3] **宮西洋太郎**, “**クラウドコンピューティングとどう向き合うか～クラウドはビジネス系情報システム構築技法の革命となりうるか～**, ”**電子情報通信学会研報**, Vol.110, No.70 pp.1-6, June 2010
- [4] **宮西洋太郎**, “**クラウドコンピューティングトライアルプロジェクトの概要～ソフトウェア産業の繁栄のために～**, ”**電子情報通信学会研報**, Vol.110, No.70 pp.51-56, June 2010
- [5] **宮西洋太郎**, “**世論形成のための意見収集･形成システムの試作～汎用的意見交換**Web**サイト～**, ”**電子情報通信学会研報**, Vol.110, No.184 pp.17-22, August 2010
- [6] **システム仕様書「意見収集･形成システム」バージョン**J http://aistem.web.fc2.com/cloudcomputing/OpinionGatheringSystemJ.pdf
- [7] **「意見収集･形成システム」ソースコード** (zip**ファイル**) **非クラウド**Java**版** http://aistem.web.fc2.com/cloudcomputing/OpinionGatheringSystemV1.5_20100709_workspace.zip (**右クリックで保存**)
- [8] **「意見収集･形成システム」ソースコード** (zip**ファイル**) **非クラウド**PHP**版** http://aistem.web.fc2.com/cloudcomputing/OpinionVoteSystemPHPVersion2.1_20100718.zip (**右クリックで保存**)
- [9] (**株**)**グルージェント**, 「Google App Engine for Java [**実践**]**クラウドシステム構築**」 **技術評論社**, 2009**年**
- [10] **グーグル社オンラインチュートリアル** http://code.google.com/intl/ja/appengine/docs/java/gettingstarted.
- [11] **「書評システム」ソースコード** (zip**ファイル**) **クラウド版** http://aistem.web.fc2.com/cloudcomputing/workspaceforGAEBookCommentSystem20100828.zip (**右クリックで保存**)
- [12] **「意見収集･形成システム」ソースコード** (zip**ファイル**) **クラウド版** http://aistem.web.fc2.com/cloudcomputing/workspaceforGAE_OpinionGatheringSystemVG1.0_20100902.zip (**右クリックで保存**)
- [13] **菊田英明**, 「**基本情報技術者試験らくらく突破**Java」, **技術評論社**, 2002**年**11**月**
- [14] **宮本信二**, 「**基礎からのサーブレット**／JSP」, **ソフトバンククリエイティブ**, 2007**年**3**月**