Technology Reports（特集）　パブリッククラウド　運　用　障　害
パブリッククラウド活用特集

大規模障害に備えるパブリッククラウド上でのシステム運用

イノベーション統括部　守屋（もりや）　裕樹（ひろき）

AWSをはじめとするパブリッククラウドが普及し，企業だけでなく官公庁，教育機関など多くの組織や団体にパブリッククラウドが利用されている．このように社会インフラとなりつつあるパブリッククラウドは，ひとたび停止すると社会全体に大きな影響を与えかねず，クラウドを利用したシステム運用の重要性が増している．ドコモでは長年大規模にパブリッククラウドを利用してきた経験から，大規模障害を想定したシステムの設計や運用ノウハウを蓄積してきた．また大規模障害発生時に会社内の情報連携をスムーズに行えるツールなどを開発してきた．これにより，会社全体としてパブリッククラウドの大規模障害を想定したシステム運用を行うことが可能になった．

1. まえがき

AWS（Amazon Web Services）[*1]をはじめとするパブリッククラウド[*2]が普及し始めてすでに10年以上経っており，企業だけでなく官公庁や教育機関などさまざまな組織や団体で，パブリッククラウドを利用したシステムの構築や運用がなされるようになってきた．

パブリッククラウドは，もはや単なるクラウドサービスではなく，社会全体を支えるインフラとして機能するようになってきており，ひとたびパブリッククラウドで大規模な障害が発生すると社会全体に影響を与えかねない状況となっている．

ドコモでは長年大規模にパブリッククラウドを利

＊1　AWS：Amazon Web Services社が提供するクラウドコンピューティングサービス．
＊2　パブリッククラウド：インターネットを介して誰でも利用ができるクラウドコンピューティングサービス．

用してきた経験から，パブリッククラウドの特性を最大限に活かしたシステム運用のノウハウを蓄積し，運用を支援するツールなどを開発しながら，会社全体としてパブリッククラウドをより活用できるように取り組んできた．本稿では，ドコモ以外の組織でもパブリッククラウド利用時の参考としていただくことを目的として，これらの取組みについて解説する．

2. システム障害に対する考え方

2.1 可用性の高いシステム構成

システムには障害がつきものである．壊れないシステムや障害が発生しない完璧なシステムは存在しない．この原則はオンプレミス*3でシステムをつくる場合でも，パブリッククラウドでつくる場合でも同じである．そのため，システムを構築する場合は，構成する各要素で障害が発生した場合でも，システム全体として少しでも可用性が上がるような構成をとることが重要になってくる．例えば，単一のサーバなどの装置が機能しなくなった場合に，バックアップ装置に切り替えてシステムの機能を維持するといった対応は，古くから取られてきた手法である．また，アプリケーション領域においても，システム全体の耐障害性を上げるための工夫として，非同期処理によるシステムの疎結合を図り障害影響範囲を限定する実装や，障害発生前提でのリトライ処理などの実装が多い．さらに，近年ではシステムを1つのモノリシック*4なサービスとして構成するのではなく，複数のマイクロサービス*5に分割して構成する動きが多くなってきており，障害発生時の影響範囲を限定することでシステム全体の可用性を上げるような取組みがなされている．

2.2 パブリッククラウドにおける責任共有モデル

パブリッククラウドを利用する場合，クラウド利用者とパブリッククラウド事業者の間でそれぞれの責任範囲を明確にし，双方でシステム全体の可用性を確保していく「責任共有モデル［1］」と呼ばれるモデルが採用されていることが多い．例えば，仮想マシン*6であれば，一般的にデータセンタや物理サーバ，ネットワーク，仮想化レイヤまでの領域はパブリッククラウド事業者の責任範囲であり，事業者によってこれらのインフラストラクチャ*7や物理装置，ネットワークなどの冗長化といった可用性を保つような対策がされている．一方で，仮想マシンOSやその上で動作するアプリケーションなどは，クラウド利用者側の責任で可用性を保つような実装をする必要がある．例として仮想マシンを挙げたが，近年クラウドサービスにおいてはIaaS（Infrastructure as a Service）*8だけでなく，PaaS（Platform as a Service）*9やSaaS（Software as a Service）*10が活発に活用されている．しかし，利用者とパブリッククラウド事業者間の責任範囲が異なるだけで「責任共有モデル」の考え方は同様に適用される．

多くのパブリッククラウド事業者は，利用者によるシステムの可用性を保つ設計を支援するための機能を提供しており，よくある構成パターンについては，ベストプラクティスとしてドキュメントやホワイトペーパーを積極的に情報配信している．そのため，利用者はこれらの機能を利用して，効率的に高可用性を実現するためのシステムを構築することが可能になってきている．また，PaaSやSaaSを利用する場合には，パブリッククラウド事業者側で，すでに高可用性を実現する構成がとられていることが多いため，利用者側は積極的にそれらの機能を活用してシステム構築を行うようになってきている．

＊3　オンプレミス：企業がシステムを構成するハードウェアを自社で保有し，自社で保守運用すること．
＊4　モノリシック：複数の機能を1つの大きなソフトウェアとして組み上げる構成．
＊5　マイクロサービス：複数の小さなサービスを組み合わせて1つのサービスを作り上げるシステムの開発技法のこと．
＊6　仮想マシン：ソフトウェアによって仮想的に構築されたサーバなどのコンピュータ．
＊7　インフラストラクチャ：アプリケーションを実行するのに必要な物理的もしくは仮想的なデータセンタやサーバ，ネットワークなどの総称．
＊8　IaaS：サーバ，ネットワークなどのハードウェアを仮想的に貸し出すサービス．利用者は借りたサーバやネットワーク上にOSやアプリケーションソフトウェアを設定して利用する．
＊9　PaaS：アプリケーションを実行するためのOSやミドルウェアを含むプラットフォームをクラウド上で貸し出すサービス．利用者は借りたプラットフォームの上でアプリケーションソフトウェアを作成して利用する．
＊10　SaaS：アプリケーションをクラウド上で貸し出すサービス．利用者はアプリケーションを直ちに利用できる．

3. 近年発生したクラウド事業者側での大規模障害

近年発生したパブリッククラウドの大規模障害の事例として以下のものがある．

⑴AWSの障害事例

AWSでは，2019年8月23日に空調設備の管理システム障害が原因で冷却システムが正常に稼働せずサーバがオーバーヒートを起こしてしまい，単一Availability Zone*11のEC2（Amazon Elastic Compute Cloud）*12やEBS（Amazon Elastic Block Store）*13に障害が発生した．また，EC2やEBSは，その他のサービスを構成するための基盤サービスとなっていることから，RDS（Amazon Relational Database Service）*14，Amazon Redshift*15，Amazon ElastiCache*16およびAmazon WorkSpaces*17などのマネージドサービス*18にも影響があったことが確認されている．単一Availability Zoneでの障害だったため，クラウド利用者によっては，複数Availability Zoneの冗長化構成をとることで影響を最小化したところもあったが，それでも一部設定によっては影響が確認されていたことが報告されている．

また，2020年4月20日にも東京リージョンでSQS（Amazon Simple Queue Service）*19やAWS Lambda*20などのマネージドサービスにおいて処理エラーの増加や遅延の発生などの障害発生が確認されており，これらを利用していた利用者への影響が確認されている［2］．

⑵Microsoft Azureの障害事例

Microsoft Azure*21では，2020年9月29日にAzure AD（Azure Active Directory）の認証エラーが世界中で観測される障害が発生した．Azure ADは開発者やユーザ，サービス間などの多くのサービスの認証認可を行うMicrosoft Azureのコアサービスであるため，これにより利用者がMicrosoft AzureやOffice 365が使えなくなるといった影響が確認されている．本来何重にもテストされてデプロイ*22されるはずの内部検証段階のアップデートが，SDP（Safe Deployment Process）システム*23の潜在的なバグにより，テストをバイパスして本番環境にデプロイされたことが原因と発表されている［3］．

⑶GCPの障害事例

GCP（Google Cloud Platform）*24では，2020年3月27日にアクセス制御・管理サービスであるCloud IAM（Cloud Identity and Access Management）に障害が発生した．Cloud IAMは多くのGCPサービスへのアクセス制御に利用されている共通サービスのため，他のサービスへも影響を及ぼす大規模障害となり，公式発表では障害の影響は14時間にも及んだ．原因はCloud IAMへの想定外の変更リクエストが発生したことで，Cloud IAM内部のキャッシュサーバのメモリが枯渇した結果，Cloud IAMへのリクエストのタイムアウトが発生していたことと発表されている［4］．

このように世界中ですでに多くの実績があるクラウド事業者のサービスでも，思わぬ形で障害が発生し，利用者が影響を受けることはあり，今後もゼロになることはない．

4. 利用者が考慮しておくべきこと

パブリッククラウドにおいてシステム障害が発生した際，利用者が考慮しておくべきことをいくつか挙げる．

4.1 障害発生を前提とする

全世界で多くの利用実績があるパブリッククラウ

＊11 Availability Zone：1つまたは複数のデータセンタ群．各Availability Zone間は物理的に独立している．
＊12 EC2：AWSが提供するIaaSの1つ．仮想マシンを提供するサービス．
＊13 EBS：AWSが提供するIaaSの1つ．ブロックストレージを提供するサービス．
＊14 RDS：AWSが提供するPaaSの1つ．Relational Databaseの機能をサービスとして提供する．
＊15 Amazon Redshift：AWSが提供するPaaSの1つ．データウェアハウスをサービスとして提供する．
＊16 Amazon ElastiCache：AWSが提供するPaaSの1つ．インメモリキャッシュをサービスとして提供する．
＊17 Amazon WorkSpaces：AWSが提供するSaaSの1つ．WindowsもしくはLinuxのデスクトップ環境をサービスとして提供する．
＊18 マネージドサービス：クラウドサービスのうち，リソースのプロビジョニングや運用の大半をクラウド事業者の責任で実施しているサービス．クラウドコンピューティングサービスのうち，特にPaaSやSaaSを指す．
＊19 SQS：AWSが提供するPaaSの1つ．メッセージキューの機能をサービスとして提供する．

ドにおいても障害が発生し，パブリッククラウドが進化しても障害をゼロにすることはできない．利用者がこの事実を十分に把握しておくことは非常に重要である．仮にこの事実を把握せずにパブリッククラウドを利用してしまうと，障害発生前提での設計や運用が十分にできていないことが要因で，大きな損失を出してしまうことも考えられる．まずは開発や運用担当者だけでなく，ビジネスオーナーや経営層を含めてこの事実を理解しておくことが重要である．

4.2 ベストプラクティスを活用する

前述のように障害をゼロにすることはできないが，障害による影響を低減することは可能である．多くのパブリッククラウド事業者は，可用性や信頼性を維持するためのシステム設計を利用者に推奨すると同時に，それらを利用者がより実現しやすいようにさまざまな機能やオプションを提供している．利用者はこれらの機能を利用して，パブリッククラウドに障害が発生した場合でも，システムをできるだけ継続して稼働させるような設計をすることが重要になってくる．幸いにもパブリッククラウドは世界中にすでに多くの利用者が存在しており，あらゆるユースケースで採用されてきた設計パターンがベストプラクティスとして公開されている．そのため，利用者は一から設計を考える必要はなく，それらのベストプラクティスを採用して設計や運用を行っていくことで，リスクの低減を図ることが可能である．

また，近年はPaaSやSaaS機能が充実しており，あらかじめパブリッククラウド事業者がベストプラクティスを踏まえて設計したサービスやプラットフォームを利用者が積極的に利用することで，耐障害性の高いシステムを構築するという流れが活発になっている．「サーバレス」と呼ばれる設計パターンはこの代表例である．サーバレスは，物理サーバや，IaaSに代表される仮想サーバを利用者が意識することなく，システムの設計や運用を行う手法であるが，本稿ではサーバレスの詳細は割愛する．これらの設計手法やサービスを活用することは，耐障害性の面だけでなく利用者の運用面でも負荷が低減されるというメリットがあるため，今後さらにこの流れが加速することは間違いない．

4.3 障害に備える運用設計をする

どのように対策してリスクを低減することはできても，やはり障害自体や障害の影響を完全にゼロにすることはできない．また，あらかじめ想定できる障害の場合は対策も十分に可能だが，想定外のケースや事象で発生する障害も多い．そのため，実際に障害が発生した際に迅速に原因を特定し，リカバリできるように運用を設計しておくことも重要である．具体的なポイントをいくつか挙げる．

(1)システムの可観測性をあげる

システム監視はパブリッククラウドにおいても非常に重要である．また特にパブリッククラウドを利用する場合，インフラストラクチャ構成がIaaS/PaaS/SaaSにまたがることも多く，アプリケーション構成においてもコンテナ*25やマイクロサービスの採用が一般化してきており，多くの環境に分散してシステムが配置されるようになっている．そのため，これまで以上にシステムの可観測性を上げる必要があり，それは単なる死活監視だけでなく，アプリケーションで処理される個々のリクエスト*26やメソッド*27レベルで処理を追跡する分散トレーシングや，ログや追跡結果を統合的に集約して可視化や分析を行える基盤サービスなどの採用が考えられる．これらの仕組みはクラウド事業者が提供するもの，3rd Party*28のサービスとして提供されているもの，

---

*20 AWS Lambda：AWSが提供するFaaSの1つ．アプリケーションコードの実行環境が提供されており，利用者は作成したソースコードを登録してアプリケーションが実行できる．
*21 Microsoft Azure：Microsoft社が提供するクラウドコンピューティングサービス．
*22 デプロイ：アプリケーションをそれらの実行環境に配置して展開すること．
*23 SDPシステム：Microsoftにて採用されている，ソフトウェアを安全にデプロイするためにその過程を管理するシステム．
*24 GCP：Google社が提供するクラウドコンピューティングサービス．
*25 コンテナ：コンピュータ仮想化技術の一種で，1つのホストOSの上にコンテナと呼ばれる専用領域を作り，その中で必要なアプリケーションソフトを動かす方式のこと．
*26 リクエスト：アプリケーションへの動作要求のこと．
*27 メソッド：GET，POST，PUT，DELETEといったHTTPメソッドのこと．
*28 3rd Party：第三者メーカのこと．

OSS（Open Source Software）[*29]として公開されているものなどさまざまであるため，目的にあったものを選択して活用していく必要がある．

⑵想定外の障害リスクを下げる

想定外の障害リスクをできるだけ小さくすることも，運用を改善していくためには重要である．クラウド上のシステムにおけるフェイルオーバー[*30]試験や，特定のクラウドサービスが停止した場合のリカバリ試験などを普段から実践しておくことは，想定外の障害リスクを下げるために効果がある．

ここで，パブリッククラウドならではの障害リスク低減手法としてChaos Engineeringを紹介する．Chaos Engineeringは，システムの本番環境で意図的に障害を発生させ，システム全体への影響を観測するという手法である．これにより，障害発生時にシステム全体に想定していない影響がないかを観測し，システムの耐障害性を向上させることができる．これは，インフラストラクチャをAPIで管理でき，作り直しがいくらでもできるクラウドならではの手法であり，米Netflix社は，日常的にこのような運用を実施することで，想定外の障害影響のリスクを下げる取組みを実践している［5］．なお，Chaos Engineeringは本番環境（実運用環境）で実施する手法のため，やみくもに実施するのではなく，すでに記載したシステム設計を適切に実践し，システムの耐障害性に十分に自信をもった上で実施する必要がある点に注意が必要である．

⑶クラウドサービスの各サービスの構成を理解する

パブリッククラウドを利用する場合に意外にも重要となるのが，クラウドサービスの各サービス自体の構成を理解することである．もちろんクラウドサービスは，セキュリティやコンプライアンスなどの関係で詳細な内部構造は公開されておらず，クラウドサービスによってはデータセンタの場所自体も非公開となっている．一方で，システム設計や運用時に利用者に考慮してもらうために，一部論理構成などが公開されているサービスなどがあり，それらを理解しておくことで障害発生時の原因特定やリカバリをスムーズに行える場合がある．

例えば，AWSの場合，Availability Zoneと呼ばれるデータセンタ群があり，各Availablity Zoneは物理的に独立している．この概念を理解すること自体は高可用性設計において必須である．またPaaS/SaaSについては，AWS自身がユーザと同様の視点でこれらのAvailability Zoneにまたがってサービスを構成した上で，利用者に展開しているものも多い．このことを理解しておくことで，特定のAvailability Zoneで障害が発生した際に，それぞれのクラウドサービスに影響が出るのか，そうでないのかの把握が可能となり，また利用者側で障害発生時に特定のAvailability Zoneへのトラフィックを切り離すオプションがあれば，それにより影響を最小化できる可能性がある，といった判断が可能となる．また仮想サーバのEC2は，多くのPaaS/SaaSサービスの基盤となっているサービスである，ということを理解しておくことで，仮にEC2に障害が発生した場合に「他のサービスにも影響が出る可能性がある」ということが予測でき，少なくとも身構えることができる．そのほかにも各パブリッククラウドサービスの構成が公開されている場合があるため，それらの構成が理解できているほど，障害発生時の対応はスムーズに実践できる．

5. Support Visualizerの開発と提供

最後に，ドコモ内で最も利用されているクラウドサービスの1つであるAWSを利用する場合において，ドコモが実践している障害発生に対する取組みにつ

＊29 OSS：ソースコードが無償で公開されており，誰でも再利用や改変が行えるソフトウェア．

＊30 フェイルオーバー：システムに障害が発生した際に自動的に冗長化された待機システムに切り替える仕組み．

いて解説する．ドコモでは，900以上（2020年12月時点）のAWSアカウントを運用しており，さまざまなワークロード*31でAWSを活用しているが，2019年ごろAWS東京リージョンで発生した障害の影響を少なからず受け，これによりいくつかの課題が判明した．これらの課題を解決するための取組みとして，AWSのサポートケース情報（後述）を集約するSupport Visualizerというシステムを構築している．

5.1 AWS大規模障害で判明した課題

前述したとおり，ドコモではさまざまなワークロードでAWSを活用している．当然ながらワークロードによりシステム要件やアプリ利用者数，データ量が異なるのでシステム設計や運用がシステム単位で独立していることが多く，利用するクラウドサービスも異なってくる．そのため，ひとたびクラウドサービスで障害が発生した場合に，それらの影響がどの程度発生するのかはシステムによって異なる．実際にこれまで大規模障害が発生した際に，全く影響がなかったシステムもあれば，クラウドサービス利用者やアプリ利用者への影響が出てしまったシステムも存在する．これらの障害発生時に，個別のシステムや会社全体の運営に対して次のような課題が判明した．

⑴会社全体での情報収集

障害による個別システムへの影響は必ずしも同じではないため，会社としては個別のシステムへの影響度合いをいち早く認識し，適切なアナウンスなどを実施する必要がある．しかし，実際には各システムの運用は独立しており，また障害発生時の現場は原因特定と復旧作業に追われているため，すぐに情報を集約することは難しい．もちろん最優先すべきは，システムを迅速に復旧させ，利用者への影響を最小化することではあるが，そのような中でも会社は社会的責務として適切に情報の配信を行っていくことが重要である．これまで発生した大規模障害発生時はこれらの両立が非常に難しく，会社全体での情報収集に課題が残った．

⑵影響範囲の特定と対処

個別のシステムでの課題も浮彫りになった．システム障害自体はパブリッククラウド利用時に限らず常に発生し得るため，大規模障害時にも通常のシステム障害発生時と同様に対処を行っていくことになるが，その際クラウド自体の障害に起因するものなのか，個別のシステム固有の問題で発生したものなのかが分からず，原因特定と対処に時間がかかってしまう事象が発生した．

パブリッククラウド事業者の多くは，自身のサービスステータスを公開しており，障害発生時にはそれらのステータスを見ることでおおよその影響範囲が確認可能である．ただし，これまでのクラウド障害の経験上，これらのステータスでは必ずしも発生しているすべての障害を示しているわけではなく，実際にはクラウド利用者からの申告で障害発生が判明するケースや，障害がクラウド事業者から事後報告されるケースもある．

5.2 Support Visualizer

AWSの場合，AWS自体が提供するサービスで障害が発生し，利用者システムに影響が出た場合は，サポートケースと呼ばれる問合せチケット*32にてAWSに報告をするような運用が可能である．これにより，AWS側には障害の影響範囲に関する情報が集まるため，影響範囲の特定が可能になる．

一方で利用者であるドコモは，各アカウントが独立して運用されているため，サポートケース上では別プロジェクトの障害影響を把握することができず，プロジェクト同士の直接的な情報交換のやりとりが

＊31 ワークロード：CPU使用率などのシステムの負荷の大きさを表す指標．特にパブリッククラウドの分野では，クラウド上で実行されるOSやアプリケーションコードなどを含めたシステム自体を表すこともある．本稿では後者の意味で用いる．

＊32 問合せチケット：個々の問合せやそれに対する返答などを管理する単位．

必要となる．ただし，すでに述べたように障害発生中は現場レベルでは自身のシステムの復旧対応などで精一杯のため，そのような情報交換を実施する余裕がないのは事実である．これらは会社全体として情報を集約する場合でも同じことがいえる．そこで，社内で起票されたサポートケース情報を集約し，社内の誰でも閲覧できるシステムとして，Support Visualizerを開発した．

⑴アーキテクチャ

Support Visualizerのアーキテクチャを**図1**に示す．

サポートケース情報はAWSアカウントごとに独立しているが，サポートケース情報を取得できるAPIがAWSより提供されているため，本システムはそれを経由してサポートケース情報を集約する．集約された情報は，Elasticsearchによって検索・分析できるようにインデックス化してデータベースに格納される．集約された情報はKibana[*33]のダッシュボード[*34]を利用してCCoE（Cloud Center of Excellence）[*35]による閲覧・分析が可能で，合わせて緊急度の高いサポートケースが起票された場合にはSlack[*36]などでCCoEに通知するように設定されている．

また社内利用者向けには，集約されたサポートケース情報の閲覧用にポータルを展開しており，キーワードやサービス，緊急度，サポートケース情報更新時間などでフィルタリングをして，社内のサポートケース情報が閲覧できるようにしている（**図2**）．

⑵解決が期待できる課題

Support Visualizerの展開により，クラウド上のシステム障害発生時に個別のシステムからサポート

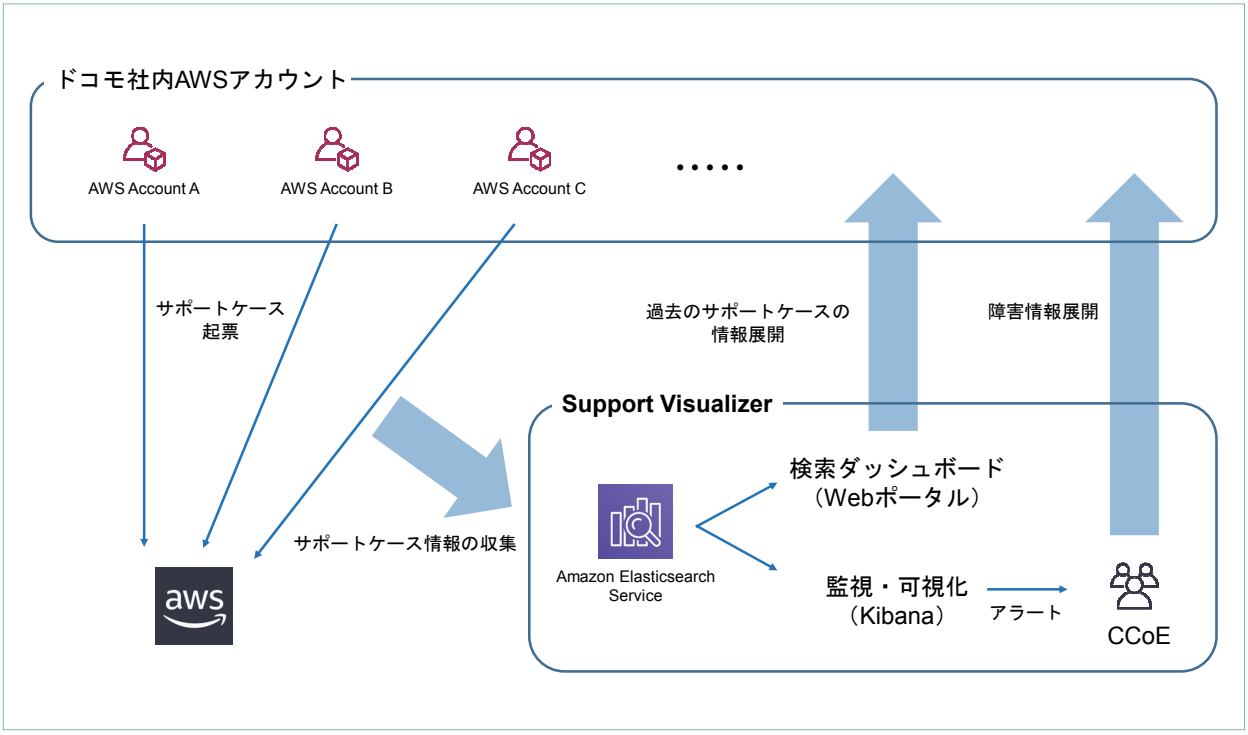

図1　Support Visualizerのアーキテクチャ

＊33 Kibana：Elastic社により開発されているオープンソースのデータ可視化ツール．
＊34 ダッシュボード：情報を集約する画面．
＊35 CCoE：企業においてクラウドの活用を成功させるために，ベストプラクティスの確立や必要な制度，ガバナンスなどを作成し，社内に展開していく専属のチーム．
＊36 Slack：Slack Technologies社が提供するビジネスチャットツール．

図2　Support VisualizerのWebポータル用のダッシュボードイメージ

ケースが起票された場合，社内のほかの利用者も含めてその内容が確認できるようになる．これは特にクラウドの大規模障害発生時に有効で，各システム運用者が個別システムの障害対応に追われている中でも，それらのシステムから起票されたサポートケースが自然と集約される．これにより，会社がシステムへの影響範囲の全体像をスムーズに把握することができると同時に，個別システムの運用者が自身のシステムで受けている影響が個別の事象なのか，クラウド全体で発生している事象なのかを特定することができる．また，すでに同様の事象が他システムでも発生している場合，それらの解決策についても確認できる可能性がある．ドコモでは，さまざまなワークロードで900以上のAWSアカウントを利用しているため，クラウド事業者から発出されるステータス情報だけでは取得しきれない，実際の現場で発生している影響も把握することが期待できる．

(3)副次効果

Support Visualizerの副次効果として，普段のクラウド利用時の開発や運用での技術的な問合せなどが集約されるため，結果として社内のクラウド利用時の技術ノウハウが集約できるという点が挙げられる．今後は集約したサポートケースの傾向分析やそこからのノウハウ集約を能動的に行う仕組みなどを取り入れて，各システムにフィードバックし，より効率的なクラウド利用を促進していきたい．

6. あとがき

本稿では，パブリッククラウドの大規模障害を見

据えたシステム運用について解説した．クラウドやコンテナ，マイクロサービスといったように技術やアーキテクチャが進化しても，障害をゼロにするシステムをつくることは不可能である．そのため，少しでも障害の影響を少なくする取組みを継続し，それらのベストプラクティスを醸成して会社全体でパブリッククラウドをうまく活用していけるように促進していきたい．

文　献

［1］ 総務省：“クラウドの特性とセキュリティ．”
https://www.soumu.go.jp/ict_skill/pdf/ict_skill_2_3.pdf

［2］ AWS：“東京リージョン（AP-NORTHEAST-1）で発生したAmazon EC2とAmazon EBSの事象概要，”Aug. 2019.
https://aws.amazon.com/jp/message/56489/

［3］ Microsoft Azure：“RCA - Authentication errors across multiple Microsoft services and Azure Active Directory integrated applications（Tracking ID SM79-F88），”Sep. 2020.
https://status.azure.com/status/history/

［4］ Google Cloud：“Google Cloud Infrastructure Components Incident #20003.”
https://status.cloud.google.com/incident/zall/20003

［5］ Netflix：“The Netflix Simian Army，”Midium, Jul. 2011.
https://netflixtechblog.com/the-netflix-simian-army-16e57fbab116