OPEN
Compute Project

Proposed Track
For
Open Hardware Management

Revision History

Date	Name	Description
12-07-2011	Grant Richard	Original draft based on the discussions before the NYC Open Compute Summit, at the Summit + individual conversions/email. Thanks to all who helped form this.
12-10-2011	Markus Fischer	General edits and clarifications. Further details on Firmware Lifecycle
12-11-2011	Joel Wineland	Various edits. Additional points added to strategic track examples.
12-22-2011	Grant Richard	Various edit from comments from 12/12 conference call.
12-12-2011	Matthew Liste	Various edits to smooth document and clarify points.

Contents

Summary

Open Compute's focus is on server hardware, with much of its effort concentrating on machine, cabinet and data centers design. Yet, a significant part of scale computing is managing the firmware that makes the machines run, alerting about their health, discovering their resources and accessing them remotely. For scale computing, tooling and standards around firmware behavior and maintenance is required to reduce the support burden and secure machines. These low/no touch, scalable, scriptable, secure and standard tools must maintain firmware versions/configurations, remote access solutions, retrieve machine configuration/information and fault/event alerts.

Open Compute のフォーカスは、サーバー･ハードウェアに向けられているが、マシンや、キャビネット、データセンターのデザインについても、膨大な量力が費やされている。 しかし、スケールしていくコンピューティングの重要パートは、依然としてファームウェアの管理にあり、それにより、マシンの実行や、障害に関するアラート、リソースの発見、そしてリモート･アクセスが実現されている。 スケールしていくコンピューティングのための、ファームウェアの振舞およびメンテナンスに関する標準が、サポートの負担を減らし、また、マシンを安全に保つために必要とされる。これらの、ロー･テクノロジー／ノー･テクノロジー／スクリプト対応のスケーラブルの標準ツールは、ファームウェアのバージョンとコンフィグレーションを管理し、リモート･アクセスのソリューションを達成し、そしてマシンのコンフィグレーションとイベント･アラートを実現する上で欠かすことができない。

Presently, these tools are mostly platform/vendor specific and non-scalable. Because of this, existing scale compute users must create and maintain systems and processes to unify and augment existing tools -- with every individual scale users largely creating the same toolset. The OCP Hardware Management Track will address these issues by providing consistent, scalable and open source standards and tools for Open Compute servers.

現時点において、これらのツールの大半が、プラットフォームおよびベンダーに固有のものであり、また、ノン･スケーラブルなものである。 そのため、すべてのスケール･コンピュートを使用しているユーザーは、現存のツールを統一／拡張するためのシステムとプロセスを、自ら構成していかなければならない。 そして、すべてのスケール･コンピュート･ユーザーが、同じツールセットを個々に作成するという状況にある。 この OCP Hardware Management Track は、一貫性のある、スケーラブルで、オープンソースによる標準とツールを提供することで、Open Compute サーバーにおける問題に取り組んでいく。

License

As of April 7, 2011, the following persons or entities have made this Specification available under the Open Web Foundation Final Specification Agreement (OWFa 1.0), which is available at http://www.openwebfoundation.org/legal/the-owf-1-0 agreements/owfa-1-0:

Facebook, Inc.

You can review the signed copies of the Open Web Foundation Agreement Version 1.0 for this Specification at http://opencompute.org/licensing/, which may also include additional parties to those listed above.

Your use of this Specification may be subject to other third party rights. THIS SPECIFICATION IS PROVIDED "AS IS." The contributors expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, noninfringement, fitness for a particular purpose, or title, related to the Specification. The entire risk as to implementing or otherwise using the Specification is assumed by the Specification implementer and user. IN NO EVENT WILL ANY PARTY BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Overview

Hardware Management can be thought of in four areas:

Hardware Management は、以下の 4つの領域に存在すると考えられる：

1. Firmware Lifecycle
2. Event, Alert and Logs
3. Remote Operations
4. Strategic Technologies

The problem space is:

そして、問題空間は、以下の通りである：

- Inability to deploy firmware fixes and configuration changes quickly and at scale
- Instabilities due to many versions of firmware in different combinations
- Lack of agility caused by having to integrate new tools / new vendors into existing home-grown Hardware Management environments
- Lengthy and brittle bootstrap processes when attempting to automate firmware load and configuration process
- The bugs in unused portions of over-featured products cause unnecessary firmware updates
- Tooling is vendor specific and often does not scale to many tens of thousands of machines.

- ファームウェアの修正とコンフィグレーションの変更を、大規模スケールにおいて、迅速にディプロイする能力が欠落すること
- 各種バージョンのファームウェアが、さまざまな組み合わせで用いられることによる、不安定な性質
- 既存の自家製 Hardware Management 環境に、新しいツール／新しいベンダーを統合せざるを得ない状況が引き起こす、アジリティの欠落
- ファームウェアのロードと、コンフィグレーションのプロセスを自動化しようと試みるときの、長いくて脆弱なブートストラップ･プロセス
- 過度の機能を持つプロダクトの、未使用部分のバグが引き起こす、不要なファームウェアのアップデート
- ツールによる処理はベンダー固有のものであり、大量のマシンに合わせてスケールしないことが、頻繁に生じる

For these reasons, it is critical to maintain a machine's firmware and those processes must scale. Further examples of the larger problem set are:

そして、こうした問題が複合する、より大きな問題は、以下の通りである：

1. There are no broadly adopted and consistently implemented capabilities around basic machine maintenance.

   基本的なマシン･メンテナンスにおいて、包括的に適用が可能で、一貫性をもって実装される能力が存在しない。

   - Vendor written maintenance tools *with identical functions* for the alerts/events, remote operations and firmware lifecycle are not interoperable. Vendor X tools can only be applied to their machines and can't be used on Vendor Y even though they perform the same functions.
   - For standard based processes like event/alert/log data and remote management, even though the delivery may be the same (SNMP, IPMI and syslog) the message numberings, payloads or commands may be different.

   - ベンダーが記述する、アラート／ベントや、リモート操作、ファームウェア･ライフサイクルなどのための、統一された機能を用いたメンテナンス･ツールは、インターオペラビリティを持っていない。 Vendor X のツールは、自身のマシンにだけ適用が可能なものであり、たとえ同じ機能を持っていても、 Vendor Y のマシンに適用できない。
   - イベント／アラート／ログのデータや、リモート管理などのスタンダード･ベースのプロセスに関して、たとえ、その配信（SNMP、IPMI、syslog）が同じであっても、メッセージのナンバリング／有効範囲／コマンドは、異なるものになるかもしれない。

2. Software deployed across identical hardware with different firmware will appear to randomly fault due to lack of ability to test across all permutations and to discover toxic combinations of individual firmware payloads/configurations.

   異なるファームウェアを用いる同一のハードウェア群に対してディプロイされたソフトウェアは、全順列を横断したテストの能力を欠き、また、それぞれのファームウェアの有効範囲／コンフィグレーションにおける有害な組み合わせを発見する能力を欠くため、ランダムに障害を引き起こすだろう。

   - In the first three years of a machine or add-in card's introduction, there are typically many (3-10) firmware releases for standard motherboard components that often need to be quickly rolled out.

- Lack of tools to rapidly and securely deploy firmware configuration and binaries.
- Vendor specific firmware lifecycle solutions generally deploy firmware payload in a piecemeal fashion.
- When vendors test their firmware as an integrated stack, PCI add-in components from different OEMs have a different revision release cadence and payload delivery system.
- Firmware configuration is often embedded in firmware payloads complicating configuration visibility and management.
- No broadly adopted standard for deployment of firmware bits or configurations exists.

- マシンやアドオン･カードの導入された後の 3年間で、迅速な量産化を常に要求される、標準的なマザーボード･コンポーネントに対して、一般的には何度（3-10 回）もファームウェアがリリースされる。
- ファームウェアのコンフィグレーションとバイナリーを、迅速かつ安全にディプロイするツールの欠如。
- 一般的に、ベンダー固有のファームウェア･ライフサイクル･ソリューションは、断片的な方式でファームウェアの有効範囲をディプロイする。
- ベンダーが統合されたスタックとしてファームウェアをテストするとき、別の OEM から供給される PCI アドオン･コンポーネントは、供給されるシステムに対して、別のリリース周期と有効範囲を持っている。
- ファームウェア･コンフィグレーションは、そのファームウェア･有効範囲内にエンベッドされるため、コンフィグレーションの視野と管理を複雑にする。
- ファームウェアの具体例やコンフィギュレーションのディプロイメントに関して、広範囲で適用されるスタンダードが存在しない。

3. Expansive or inappropriately featured vendor remote management tools

   拡張性に富み、また不適切な機能の、ベンダー製のリモート･マネジメント･ツール

- Scale users’ need very few functions.
- Complex remote management products can introduce more firmware and machine instability.

- 大規模スケール･ユーザーは、少しの機能のみを必要とする。
- 複雑なリモート･マネージメント製品は、多様なファームウェアと不安定なマシンをもたらしてしまう。

4. Need to manage machines holistically as part of the overall data center management

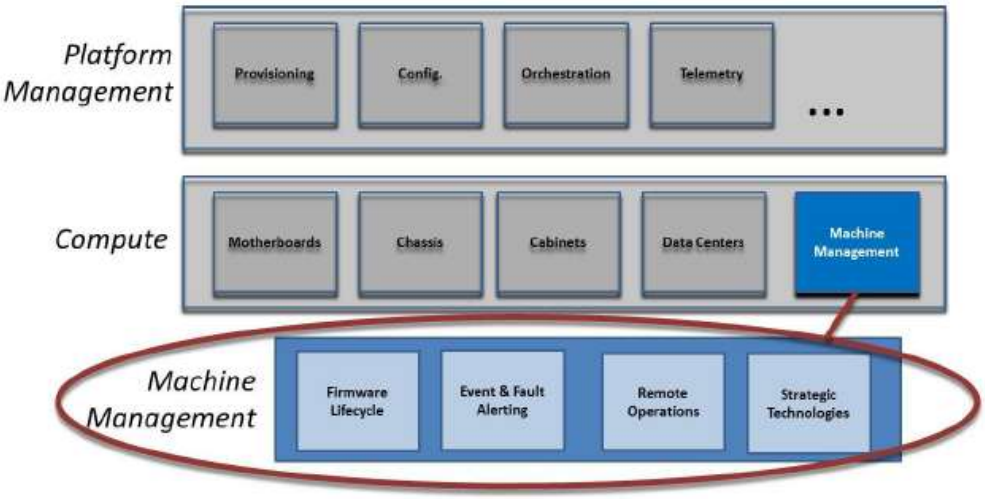
全体的なデータセンター管理の一部として、各マシンのヒストリー管理が必要となる。

- No standard ways to integration of machine information into data center BMS/DCIS systems

- データセンター BMS/DCIS システムへ向けて、各マシンの情報を統合するための、標準的な方式が存在しない。

These are some of the reasons it is important to include Open Hardware Management as part of the Open Compute effort and to establish it as a Track. Standardization for API/Interface and consistent tools will lower adoption barriers and simplify management of scale computing environments.

こうした理由により、Open Compute の一部として Open Hardware Management を取り込み、また、それを Track として確立することが重要である。 API/Interface および一貫性のあるツールのための標準化が、アダプションの障壁を引き下げ、また、大規模スケールのコンピューティング環境におけるマネージメントを簡素化していくだろう。



It is equally important to understand what Open Hardware Management is not. In this effort, we are making a distinction between Hardware Management and Platform Management. The Hardware Management effort is bounded by 1) the firmware on the machine, 2) event/alerting/logging about machine components and 3) basic remote management. Platform management encompasses all the other functions and is primarily about deploying and management operating systems and applications.

それと同様に、Open Hardware Management から除外するものを、正しく理解することも重要である。 この作業において、私たちは Hardware Management と Platform Management の区

別をしている。 Hardware Management は、以下の項目によりもたらさる。 1) 対象マシン上のファームウェア、2) マシンを構成すうるコンポーネントのイベント／アラート／ログ。 3) 基本的なリモート･マネージメント。また、その他すべての機能は Platform Management に取り込まれる。 そして、主として、オペレーティング･システムとアプリケーションの、デプロイメントとマネージメントに関わりを持つ。

It is understood that these boundaries are somewhat arbitrary. Another, hopefully temporary, difference is that the items identified as Platform Management have multiple cross platform open and closed sourced solutions so there are good options whereas most Hardware Management tools provide few, if any, good cross vendor options.

こうした境界線の引き方が、いくぶん任意的なものであることは理解できている。 そして、もう 1 つの暫定的な希望は、以下の相違点が認識されることである。 つまり、Platform Management として認識される項目は、クロス･プラットフォームのオープン性と、クローズドなソースによるソリューションを持つため、そこには適切な選択肢がある。それに反して、大半の Hardware Management ツールは、もし、優れたクロス･ベンダーの選択肢があるにしても、きわめて稀である。

It is important that Open Compute should not forget to enable Hardware Management technologies and Open Platform Management will contribute to this by providing an API/interface to access Open Hardware Management functions uniformly across all OCP platforms.

Open Compute が、Hardware Management テクノロジーを実現していくことは、忘れてはならない、重要なことである。そして Open Platform Management は、すべての OCP プラットフォームを横断する、統一された Open Hardware Management 全域で、API/Interface を提供することで、この試みに寄与するであろう。

Mission Statement

During our breakout session in the past Open Compute Summit, the participants created the following mission statement that frames our effort.

これまでの Open Compute Summit における、私たちのブレイクアウト･セッションにおいて、参加者たちが以下の Mission Statement を作成し、それが活動のフレームを構成している。

“Provide uniform management of firmware, alerting of hardware events and remote hardware access. Our focus will be on process automation and scalability by leveraging existing open standards whenever possible. We will coordinate with other groups within the OCP foundation to drive efforts.”

『 ファームウェアの統一されたマネージメントおよび、ハードウェア･イベントのアラート、そして、リモート･ハードウェア･アクセスを提供する。 私たちのフォーカスは、既存のオープン･スタンダードを可能な限り活用することで、プロセスの自動化とスケーラビリティを達成する点にある。私たちは、この作業を促進するために、OCP ファンデーション内のグループと、調整を行なっていく。』

Approach

Using the Mission Statement, our approach to delivering and maintaining projects within this track are:

この Mission Statement を用いることで、この Track において、プロジェクトを展開／継続していくアプローチは、以下のとおりとなる：

- Any solution must be scale to 100,000+ of physical/virtual machines.
- Use existing standards whenever possible. If standards require adjustment, partner with standards bodies to make changes. Examples of Platform Management relevant standards include:
  - DMTF – SMBIOS, CIM, WBEM, SMASH, ASF, WS-Man, SMI-S
  - Intel – (Broad adoption) DCMI, IPMI, IPMB, ICMB
  - AMD – OPMA
  - IETF – SNMP, NETCONF
  - SAForum – HPI
  - OASIS – DPWS, WSDM
  - SMIF – SMBUS – (Orig Intel), PMBUS
  - PICMG – ATCA (IPMI / IPMB / ICMP – mgt)
- Define requisite features, attributes and interface standards for Open Compute hardware designs.
- Encourage and work with Platform Management tools vendors to use OCP Hardware Management interfaces/APIs.
- Work with OCP Hardware designers to implement required functionality
- For each function, there will be “base” set that defines the minimal functionally to support scale computing. And there may be an “extended” set that may be developed that would serve scale compute provider with increased needs.
- As part of the every deliverable, a methodology to validate the functionality and maintain validity thereof must be included.

- As it is common to have some non-scale platforms in a scale environment, encourage closed source and specialty hardware manufacturers to comply with Open Hardware Management and to submit those platforms for validation.
- For every tool or function, make sure that security is considered and that tool use and firmware deployment will not be compromised.

- このソリューションは、100,000+ の物理／仮想マシンのスケールを前提とする。
- 既存のスタンダードを、可能な限り活用する。 もし、対象となるスタンダードが調整を必要とするなら、それを変更をするために、その標準化団体と協調する。 適切な Platform Management スタンダードの事例は、以下の項目を含むものとなる：
  - DMTF – SMBIOS, CIM, WBEM, SMASH, ASF, WS-Man, SMI-S
  - Intel – (Broad adoption) DCMI, IPMI, IPMB, ICMB
  - AMD – OPMA
  - IETF – SNMP, NETCONF
  - SAForum – HPI
  - OASIS – DPWS, WSDM
  - SMIF – SMBUS – (Orig Intel), PMBUS
  - PICMG – ATCA (IPMI / IPMB / ICMP – mgt)
- Open Compute ハードウェア･デザインのために必要な、機能／属性／インタフェースのスタンダードを定義する。
- Platform Management ツールのベンダーたちが、OCP Hardware Management の Interfaces/API を使用するよう奨励し、協調していく。
- 必要とされる機能を実装するために、OCP Hardware デザイナーと協調していく。
- それぞれの機能に関して、スケール･コンピューティングをサポートするための、最小限の機能セットである 「 Base 」を定義していく。 そして、増大する要望に応じて、スケール･コンピューティングのプロバイダをサポートすることになる、「 Extend 」セット定義する。
- 機能の妥当性を検査を、その状態を維持するための方法論を、提供物の一部として含まなければならない。
- 一般的に、スケールしていく環境に、いくつかのスケール･プラットフォームは普通に存在する。したがって、クローズド･ソースを用いる特定のハードウェア･メーカーに対して、そのプラットフォームを検証のために、Open Hardware Management に提供するよう奨励していく。
- すべてのツールと機能に関して、セキュリティが考慮されている点を確認する。 さらに、ツールの使用と、ファームウェアのディプロイメントにおいて、脆弱性が生じないことも確認していく。

Focus Areas

Logically, the Hardware Management Track divides into four areas of focus, that can be thought of as sub-projects that integrate into common solutions for Hardware Management.

論理的に見て、この Hardware Management Track は、4 つのエリアにフォーカスするように分割され、また、Hardware Management の共通ソリューションに取り込まれる、サブ･プロジェクトとして捉えることができる。

It is imperative for each sub-project not only to define the standard, but also to define a way of testing and validating compliance with the defined standard.

それぞれのサブ･プロジェクトについて、スタンダードとして定義するだけではなく、その整合性をテスト／検証するための、方法論についても定義することが絶対に不可欠である。

Sub-Project	***Description***
Firmware Lifecycle	To provide a uniform interface to independently deploy and update firmware's binaries and configurations
Events, Alerts and Logs	Standard way for OCP machines to produce and format machine event, and logged messages.
Remote Management	Consistent way to remotely explore a machines configuration and perform systems operations such as reboot and open a remote

	console.
Strategic Technologies	Follow and encourage exploration of products and standards of potential benefit for future Open Compute specifications. Such activities may include survey of alternative system management wireline protocols , integration with data center building management systems, management coprocessor alternatives and etc.

Firmware Lifecycle

Most physical compute components have associated software most often referred to as firmware. Common firmware examples include BIOS, NIC and BMC firmware. Frequently, and especially during the early part of the component's life, firmware is revised to fix bugs and introduce capabilities.

大半の物理コンピュータ･コンポーネントは、多くのケースにおいて、ファームウェアと呼ばれるソフトウェアと関連づけられる。 共通ファームウェアの例としては、BIOS、NIC、BMC といったものが挙げられる。 よくある事だが、とりわけコンポーネント･ライフサイクルの早期において、ファームウェアのバグが修正され、また、新機能が導入される。

In scale environments, it is important be able to rapidly deploy, securely update and have known combinations of firmware. Over the years, there have been toxic combinations between different versions of firmware on motherboards and components (e.g. conflicts between BIOS and NIC firmware) and firmware and OSs [like BIOS version X and Linux version Y]). To further complicate the task:

大規模スケール環境においては、迅速なディプロイおよび、セキュリティのアップデートを実現し、また、ファームウェアの組み合わせなどを把握する能力が重視される。 数年も放置すると、マザーボードおよびコンポーネント上のファームウェアにおいて、また、ファームウェアと OS において、それぞれのバージョン間における有害な組み合わせが生じてくる（たとえば BIOS と NIC のファームウェア間の競合［ BIOS バージョン X と Linux バージョン Y のように］）。 さらに、このタスクを複雑にするものとして、以下の項目が挙げられる：

- Motherboard and component manufactures often ship the most recent version of the firmware when delivering components, so even for the same motherboard or component, there will be multiple versions without coordinated firmware management.
- The cadence between firmware revisions is different for each manufacturer.
- Each manufacturer creates their environment for delivering the firmware.

- Manufactures firmware update tools do not always run in the same OS environment – causing any solution to have multiple boots to apply updates.
- The delivering of the firmware software updates includes configuration as well.

- マザーボードとコンポーネントのマニファクチャは、その構成要素を供給するときに、バージョンのファームウェアを出荷するのが常である。 したがって、以前と同じマザーボードやコンポーネントであっても、ファームウェア管理が行われることはなく、また、多数のバージョンが存在してしまう。
- ファームウェア修正のサイクルは、それぞれのマニファクチャごとに異なる。
- それぞれのマニファクチャは、ファームウェアを供給するために、自身の環境を構築する。
- マニュファクチャのファームウェア･アップデート･ツールが、同じ OS 環境で、常に稼働するとは限らない。 つまり、マルチ･ブートを持つことでアップデートを適用させる、さまざまなソリューションという、元凶をもたらすことになる。

For these reasons, it is important that deploying and updating both the firmware payload and its configuration be available for each OCP compute platform.

こうした理由により、個々の OCP のコンピュート･プラットフォームに対して、適用可能なファームウェアの有効範囲とコンフィグレーションの双方を、ディプロイ／アップデートしていくことが、きわめて重要となる。

Approach

- All components with firmware are in scope. Examples include motherboards, NICs, PCI SSD, HBAs and RAID Controllers.
- Develop architecture and requirements for the firmware lifecycle configuration, deployment, updating, security and auditing firmware.
- Any solution will have the capabilities of deploying the firmware’s configuration separately.
- Specify management framework/API/interfaces for providing this service to permit platform tool provider’s access to integrate this with their products.
- The solution needs to include a “push” and a “pull” model. A standalone method may also be needed as there may be no way to remotely recover from a failed firmware update
- Firmware and configuration needs to be deliverable through a centralized server via the network.
- Firmware and configuration needs to be changeable with or without an OS running.

- すべてのコンポーネントとファームウェアの組み合わせを、視野に捉える。 例として、マザーボードおよび、NIC、PCI SSD、HBA、RAID Controller などが挙げられる。

- ファームウェアのライフサイクル·コンフィグレーションおよび、ディプロイメント、アップデート、セキュリティ、ファームウェア監査に関する、アーキテクチャと要件を作成する。
- あらゆるソリューションが、個々のファームウェア·コンフィグレーションを、ディプロイする能力を持つようにする。
- プラットフォーム·ツール·プロバイダによる、自身のプロダクトとの統合が可能になるよう、このサービスを提供する マネージメント Framework／API／Interface を指定する。
- このソリューションは、Push／Pull モデルの取り込みを必要とする。ファームウェアのアップデイトが失敗し、リモートでリカバーする方式が存在しないとき、スタンドアロンの方式も必要とされるかもしれない
- センタライズされたサーバーをネットワークを介して、ファームウェアとコンフィグレーションを供給する必要が生じる。
- ファームウェアとコンフィグレーションは、OS を実行する、しないにかかわらず、変更可能となる必要がある。

Sub-projects for Firmware Lifecycle

OCP Firmware Lifecycle

Task	Through member code donation and development, create a framework for independently distributing firmware binaries and configuration.
Effort	- Create architecture and design document - Survey member tools to establish base version - Indentify contributors to project - Co-ordinate the releases and test cycles
Time	Nine to twelve months to initial draft vote

Sub-Project: Events, Alerts and Logs

Automation of knowledge of a standard machine condition is at the heart of a scale environment. For automation, ***Alerts***, ***Events*** and ***Logs*** need to be reliable and standard.

標準的なマシンのコンディションに関する、自動的な知識の収集は、大規模スケール環境における核心となる。 自動化においては、***Alerts***、***Events***、***Logs*** が、信頼性を発揮し、また、標準化される必要がある。

For purposes of this document, they are defined as:

このドキュメントの目的のために、それらを、以下のように定義する：

- ***Event*** is a recorded machine state change that has significance
- ***Alert*** is an urgent notification of event.
- ***Log*** contains a collection of events and also refers to the placing of events into a collection.

- ***Event*** は、マシン･ステートの変更記録であり、重要な情報を有している。
- ***Alert*** は、イベントにおける、緊急性を要する通知である。
- ***Log*** は、イベントのコレクションを含んでおり、その中から、個々のイベントの発生箇所を参照できる。

Events and Alerts can be recorded a number of ways:

イベントとアラートは、いくつかの方式により記録できる：

1. SNMP1 (Simple Network Management Protocol)
2. WS-MAN2 (WS-Management)
3. Syslog3
4. Publish/subscribe eventing services: allowing other devices to subscribe to asynchronous event messages produced by a given service. WS-Eventing / WS-Notification
   前提となるサービスが生成する、が非同時性のイベント･メッセージを他のデバイスから参照する。 WS-Eventing／WS-Notification 。

   1 http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol
   2 http://en.wikipedia.org/wiki/WS-MAN
   3 http://en.wikipedia.org/wiki/Syslog

Approach

- Define consistent event numbers and associated text payload information. For example, event 501 would always be associated with a disk fault and the payload would contain the message “Disk fault – Drive %” where % is the drive identifier.
- Leverage SNMP/syslog for “base” functionality and SNMP/syslog/WS-Man for “extended” functionality.
- Whenever possible, define both a push and pull method for collecting event and alert information.
- Concentrate on the standardization of the events rather than the actual implementations.

- Message number and message payloads will be consistent when delivered by different ways (like SNMP, WS-MAN & syslog).
- The approach should accommodate both in-band and out of band agents.
- Define mechanisms to validate and secure Event / Alert notification transports.

- 一貫性のあるイベント番号と、そこに関連づけられるテキストにより、情報の有効範囲を定義する。 たとえば、Event 501 は、常にディスク障害と結び付けられ、“Disk fault – Drive %” といったメッセージを取り込み、% の位置にドライブ名称を表示する。
- 「 Base 」となる機能のために SNMP/syslog を活用し、また、「 Extended」な機能のために SNMP/syslog/WS-Man を活用する。
- イベントの収集とアラート情報のために、可能な限り、Push／Pull メソッドを定義する。
- 現実的な実装よりも、イベントの標準化に集中していく。

1 http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol
2 http://en.wikipedia.org/wiki/WS-MAN
3 http://en.wikipedia.org/wiki/Syslog

Sub-projects for Events, Alerts and Logs

Base Event Message Standardization

Task	Define a minimum (base) set of events, their event # and their accompanying messages
Effort	- Research and identify requisite events/alerts - Determine payload format - Assign each event a unique number and create payload information - Draft proposal for member approval
Time	Two to four months

Hardware Management OCP SNMP MIB

Task	Create OCP SNMP MIB
Effort	- Review open source MIBs (like IF.MIB) that can be incorporated - Solicit MIB donations from existing hardware vendors - Integrate OMM base events messages in MIB - Draft proposal for member approval
Time	Three to five months

Syslog Standardization

Task	Standardize local and remote syslog and log using OCP Hardware Management standard messages
Effort	- Research and document standard strategies for local and remote syslog - Integrate OMM base events into syslog - Draft proposal for member approval
Time	Three to six months

Wake-on-LAN / Wake-on-Reboot

Task	Document using existing standard an implementation of WOL and WOR.
Effort	- Research and document existing solution for WOL & WOR and performance gap analysis - Draft recommendation and proposal for member approval - Work with OCP Compute Track to implement recommendations
Time	Two to four months

Reference Design for Event/Alert/Log Repository

Task	Create and implement reference design for an event repository to permit historical and filtered queries.
Effort	- Research and document existing systems (OpenTSDB?) - Survey contributors - Draft recommendation and proposal for member approval on architecture - Source effort to complete project
Time	Three to Nine months

Remote Hardware Management

Scale environments require a way to perform operations that in non-scale environments are manually performed at the console. Basic remote operations must be scriptable on every OCP machine as well as an extended set for those machines that either have increased complexity and/or greater service levels.

大規模スケール環境は、コンソールから手作業で、非スケール環境を操作するという、運用方法を必要とする。 基本的なリモート･オペレーションは、すべての OCP マシン上でスクリプト化が可能であるだけではなく、それらのマシンの拡張セットにも対応しなければならない。なお、この拡張セットには、複雑さの増大と、サービスのレベルアップという、2つの側面が存在する。

The examples of remote machines management operation that this Sub-Project is concerned with are:

リモート･マシン･オペレーションの例は、以下のような関心事として、Sub-Project 化される：

- Remote power on / off
- Remote console
- Discover a machine's hardware/firmware configuration
- Soft reboot / shutdown
- Graphical console / VGA redirect
- Wake-on-LAN (WOL) and Wake-on-Reboot (WOR)
- Basic authentication / LDAP authentication

- リモートによるパワーの ON／OFF
- リモート･コンソール
- マシンのハードウェア／ファームウェアにおけるコンフィグレーションの発見
- ソフトウェアによる Reboot／Shutdown
- グラフィカル･コンソールと VGA リダイレクト
- Wake-on-LAN (WOL) および Wake-on-Reboot (WOR)
- 基本的な認証／LDAP 認証

Approach

- Delineate the remote management capabilities that fall with the Sub-Project.
- Categorize the remote management capabilities into two sets: 1) Basic that must be present in all machines and 2) Extended that may vary from platform to platform but will always have a uniform interface and performance.
- Provide both individually managed and directory based authorization.
- Survey existing remote management technologies and implementations to determine the best technology to leverage. Identify gaps between Remote Hardware Management and existing standards. This will provide command line and API/interfaces.

- Sub-Project に落としこむために、リモート･マネージメント機能を区分けする。

- リモート･マネージメント機能を、2つのカテゴリに分類する： 1) すべてのマシンに存在すべき Basic。2) プラットフォームに応じて変化するが、常に同一のインタフェースとパフォーマンスを有する Extended 。
- 個別に管理される認証と、ディレクトリ･ベースの認証を提供する。
- 活用すべき最適なテクノロジーを判断するために、既存のリモート･マネージメントにおける技術と実装を調査する。 Remote Hardware Management と既存スタンダーの間に存在する、ギャップを識別する。それにより、コマンド･ラインと API／Interface が提供される。

Sub-projects for Remote Management

Below is a high level description of each sub-project.

Base Set – Remote Management through IPMI

Task	Standardize IPMI calls for Remote Power on/off and Remote Serial Console implementations
Effort	- Review materials for IPMI's open source API/Interface and tools. - Understand gap between current OCP BMCs and proposed standard. - Draft proposal for member approval that includes recommendation, specification and certification. - Member discussion and vote - Work with OCP Compute Hardware track to include in spec
Time	Two to three months

Extended Set – Remote Management through IPMI

Task	Extend the Base Remote Management feature set to include features that are not required by all scale compute users. These features may include VGA redirect/remote graphical console, advanced power management and monitoring, etc.
Effort	- Understand all the possible features - Need policy for compliance - Draft proposal for member approval - Work with OCP Compute track and closed source hardware vendors to implement
Time	Three to six months after Track Approval

Strategic Enabling Technology

Given the legacy of personal computers that extended to servers, the design of machines management was “bolted” on rather than designed into the platform. There are a number of efforts to provide a richer management network and stronger integration with external systems that have the promise of increasing efficiency and easing management.

パーソナル･コンピュータの遺産がサーバーへと及んでいることを前提にすると、マシン･マネージメントのデザインは、プラットフォームの中でデザインされたというより、「ボルトで締められる」ものであった。 外部システムに対するリッチなネットワークと、強固なインテグレーションを提供するために、数多くの努力がなされてきた。それは、有効性を増大させ、また、マネージメントを容易にするという、約束を伴うものでもあった。

Examples of these are:

- Separate, Subordinate or multi-system Management Networks
- I2c-Bus
- RS485
- VDM over PCIe
- Consolidated hardware management practices between components that comprise a cabinet (servers, storage, network, MOAs,etc.) through a variety of DCIM vendors. コンポーネント間でハードウェア･マネージメントを統合するプラクティスは、多様な DCIM ベンダーの手を経た、キャビネット（サーバー／ストレージ／ネットワーク／MOA など）で構成される。

Sub-projects for Strategic Enabling Technologies

Dedicated System Management Buses

Task	Investigate alternative uses of I2c-Bus, SMBus, and VDM over PCIe, RS485 etc. with various Open Compute design track.
Effort	- Work with other Open Compute tracks to understand technical and cost implications for including these. - Develop overall strategy for using dedicated system management buses
Time	Six to twelve months

Integration with data center control systems

Task	Coordinate or co-develop with OCP Data Center track to integrate into their Data Center Information Management systems (DCIM).
Effort	- Work with Data Center Track to understand integration opportunities - Research open and closed source DCIM solutions - Develop bi-direction information and control standards - Work with OCP/open source DCIM providers to implement standards.
Time	Nine to twenty-four months.

Organization & Meeting Cadence

Organization

Although there are various roles within the organization (like committer, etc.), there will be three main types of participates in the project:

組織内には様々な役割が存在するが（committer など）、このプロジェクトへの参加者としては、3つの主要なタイプがある：

1. Project Chairs – who will facilitate the flow of information, determine consensus and commit documents.
2. Working Group – are track member who are committed moving the project forward between meetings. Examples of contributions can be advice, specification and code.
3. Advisory – who are the people who are engaged in monthly meetings and discussions.
4. General Assembly - who are people are following the topic and want to be part of the decision process.

1. Project Chairs – 情報のフローを促進し、コンセンサスを判断し、ドキュメントをコミットする。
2. Working Group – 約束された行動を追跡し、ミーティング後にプロジェクトを前進させる。このコントリビューションの例としては、アドバイス／仕様／コード化などがあり得る。
3. Advisory –マンスリーのメーティングやディスカッションに、出席すべき人々。

4. General Assembly - トピックに従う人々であり、ディスカッション･プロセスのメンバーにもなれる。

Meeting Cadence

The formal meetings will have the following meeting schedules:

このフォーマルなミーティングは、以下のスケジュールを持つ：

Working Group	Will meet as needed between other project formal meetings. A notice of any meeting /conference call will be sent to the general list for anyone interested.
General Assemblies	Will be co-terminus with the Open Compute Summits. These meeting will be for a wider audience with update on the past efforts and anticipated progress.
Advisory	Will be take place approximately every month and will discuss the progress made, open issues and anticipated progress. These calls are intended provide direction/focus of efforts and approve any new projects.

It is anticipated the Sub-Project meeting cadences will follow this pattern although Sub-Projects may decide on different cadences based on their requirements.

推測される Sub-Project ミーティングの周期は、このパターンを追随するものとなるが、その要件に基づいて、Sub-Projects は別の周期を決定できる。

Sub-Projects and Groups

The information below is to provide a high level view on the initial organization and its priorities. This will be further developed once the charter is approved.

以下の情報は、組織における早期のビューと、その優先順位について、高い抽象レベルを提供する。それに加えて、この趣意書の改変も、許可されるであろう。

Sub-Projects in Priority Order

Based on a preliminary survey, the following sub-projects are listed in priority rank order:

準備段階における調査の結果として、以下の Sub-Project が、優先性順位と共にリストアップされた：

Sub-Project	**Rank**
Systems Management: Coordinate with Data Center Track to integrate into their management systems	1
Remote Management: Standardize IPMI calls for Remote Power on/off and Remote Serial Console	2
SNMP Alerting: Define a minimum (base) set of events, their event ID and their accompanying messages	3
Systems Management: Investigate I2x-Bus and SMBus with OCP Compute Track	4
SNMP Alerting: Create OCP SNMP MIB for minimum set of events	5
Firmware Management: Through member code donation and development, create a framework for independently distributing firmware binaries and configuration.	5
Syslog: Standardize local and remote syslog and log using OCP Machine Management standard messages	6
Remote Management: Extend the base IPMI features to include functionality that are not required by all scale compute users. These are features such as VGA redirect/remote graphical console, etc.	7
Remote Management: Document using existing standard an implementation of Wake-on-LAN and Wake-on-Reboot.	8

Groups

Co-Chairs:	Matthew Liste & Grant Richard
Working Group:	Matthew Liste, Grant Richard, Markus Fischer, Joel Wineland and John Keveney
Advisory:	Dlist of Open Compute / Hardware Management (hardwaremngt@opencompute.org)