accenture

StormTest

WHITEPAPER

Migration to RDK

Contents

This paper was written by Robert Colbert, Software Architect, Accenture Digital Video.

Robert has been working in technical leadership roles in the Broadcast and Communications domain for 24 years, with specialist knowledge in the areas of video platforms, consumer electronics and platform integration.

1. Introduction

1.1 Context

Operators are investing in a new generation of fully featured Set-Top-Boxes (STB) such as High Definition Personal Video Recorders (PVRs) and Media Centres. In addition there is a growing interest in thin client IP-STB solutions. Most Video Service Operators will have a mix of these STB and gateway types. There is higher pressure to decrease time-to-market and improve innovation in these products.

The Reference Design Kit (RDK) software bundle is a community-supported middleware distribution to help manage the complexity and duration of STB developments by providing a standardised basis for new STB developments. Operators are then free to customise this platform and innovate on top of it. Accenture Digital Video (ADV) offers RDK solutions using deep system knowledge combined with extensive experience of working in the area.

One of the key areas for any change to RDK middleware is to have a strategy for migration. This involves not only planning the new components but also having a strategy to deal with legacy equipment.

This paper addresses the issues that arise when migrating to the RDK middleware on STB. It looks at the system issues surrounding the problem rather than the technical details of the upgrade.

1.2 What is the RDK

The RDK is an integrated software bundle for next generation STB and gateway devices.

The RDK is fully tested and runs on QAM, IP and Hybrid devices. The RDK software components are available for deployment on devices used by operators with an RDK license agreement. RDK vendors can license the software and contribute to the RDK stack. The RDK software is available to all RDK licensees in a shared source manner.

1.3 RDK in context

The RDK is part of a wider movement in the digital TV industry towards open platforms, unified IP content delivery and common application platforms. The RDK fits into that wider movement with its:

- Emphasis on fast innovation on a stable platform.
- Enabling development speeds similar to web based development.
- Platforms composed of open sourced components with well-defined interfaces and multiple providers of implementations and services.
- Services being delivered to multiple client types, from a common source, using standard IP delivery, standardised codecs and Digital Right Management (DRM) mechanisms.
- Applications on various clients sharing a common core and communicating with a common application backend. The backend works much more like a web service than a traditional TV service.

> “RDK is part of a wider movement in the digital TV industry towards open platforms, unified IP content delivery and common application platforms”

2. The RDK Migration Strategy

2.1 Migration of Legacy Hardware Platforms

While migration is not something that is typically considered at the outset of a new STB development, there is a trend to keep STBs in the field for longer, coupled with a consumer demand for a better user experience which in combination is driving a need for technology migration. This means good STB migration planning is not a mere detail in the management of STB lifecycle: it is a pre-requisite.

If there is a requirement to upgrade existing legacy boxes to RDK-based middleware then a variety of issues need to be considered. To accurately determine if the older platform is able to support the RDK – in terms of architecture, size and performance – one must assemble answers to the following questions:

- How many box types and variants are used in the network?
- What System-on-Chip is used?
- What are the main frontend and peripheral circuits used on the motherboard?
- Is a complete set of software drivers available?
- How much total flash memory and system memory is available?
- What combination of NAND/NOR flash is used?
- What is the system memory map?

Another issue is when multiple generations of boxes need to be upgraded. Should the lowest common denominator platform be used as lead development platform? What compromises on the overall system will this cause? Conversely if the lead platform is the highest specification box, then there could be a problem creating a profile of middleware that would fit on older boxes.

2.2 Objectives of all RDK Migration

We can identify a number of objectives for all RDK Migrations – on legacy and for new platforms.

- The RDK based system should have well-defined system interfaces.
  - These interfaces define the platform which will last longer than any individual component such as a model of STB.
  - The user interface layer should be independent of service and business logic to allow the user interface and experience to be modified without changing underlying business rules.
  - The new architecture should be modular so that components can easily be modified and replaced without causing systemic change.
  - The STB software should be portable.
- The system should prefer open interfaces where practicable to minimise custom developments.
  - Unnecessary proprietary interfaces should be avoided where an open alternative exists.
  - The system can be the first step towards a fully cloud based system.
  - Business rules and logic can be migrated towards the cloud to simplify the clients.
- The system should avoid duplication of system components.
  - There should not be, for example, separate systems to provide metadata for QAM video, VOD and OTT video.
  - Reducing duplication of functions will reduce system complexity with benefits for maintainability and scalability.
- The system should have an STB application layer which is easy to modify and maintain.
  - HTML5 is increasingly preferred as the application layer technology because it is open, easily modified and deployable on many platforms.
  - There should be access to source code of the user interface.
- The RDK STB should support agile development of new user features.
  - It should allow for rapid prototyping and experimentation with user features.
  - There should be a short turnaround time for such features.
- The user interface must be very smooth and responsive.
- The system should be deployable into end-user existing home networks.
  - It should respect the variety of in-home equipment that customers now have and attempt to use it to best advantage.
  - The system needs to co-exist with legacy STBs for a period of time
- Modern DRM and CA solutions should be supported to allow maximum customer flexibility.
  - The system should support multiple DRMs which can be used at the same time.

> “HTML5 is increasingly preferred as the application layer technology because it is open, easily modified and deployable on many platforms”

3. Approach to Migration

The adoption of RDK usually involves not just one step change but a number of key issues which need to be addressed collectively.

- Middleware Migration
- Low Level Software Integration
- Application Migration
- CA/DRM Integration into RDK
- Head-end Migration

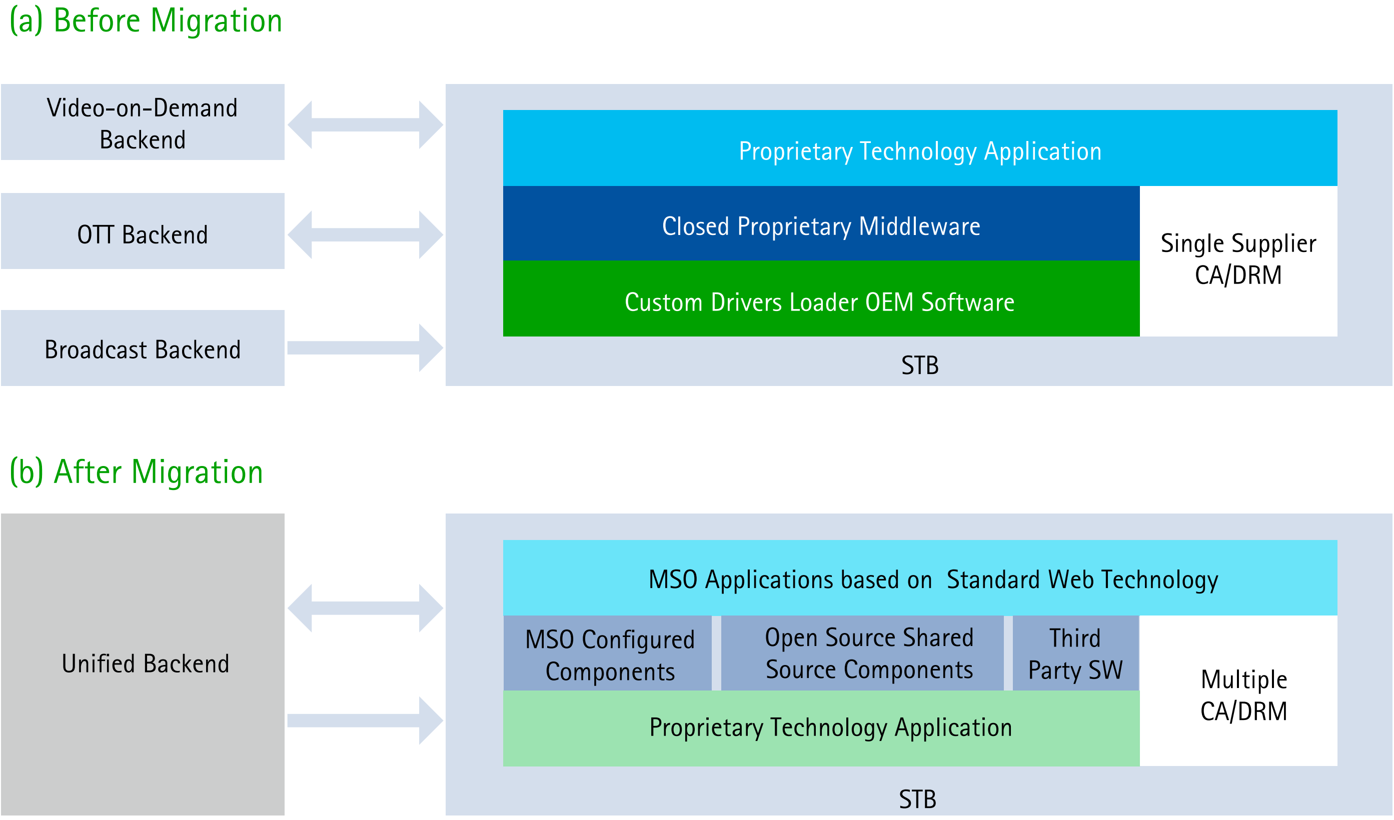
3.1 Middleware Migration

RDK now provides much of the functionality that in the past was provided by proprietary middleware. For migration it is necessary to map current functionality onto the RDK and understand how each feature can be implemented.

Business rules and business logic may now be implemented in the cloud which greatly simplifies the middleware. Examples of this are the complex rules which exist in legacy middleware for generating the channel line-up, including knowledge of the subscriber's package, local channels and fixed channel numbers. The line-up could instead be fetched on demand from the cloud with the business rules for creating it located in the cloud.

Video Service Operator configured middleware components enable certain aspects of the RDK to be customized to suit their own individual business needs. These can include items which cannot be implemented in the application layer for security or performance reasons. Any other operator specific features not supported by RDK can be supported through additional Third Party components.

Figure 1: Architecture – Before and After RDK Migration


(a) Before Migration
Video-on-Demand Backend
OTT Backend
Broadcast Backend
Proprietary Technology Application
Closed Proprietary Middleware
Custom Drivers Loader OEM Software
Single Supplier CA/DRM
STB
(b) After Migration
Unified Backend
MSO Applications based on Standard Web Technology
MSO Configured Components
Open Source Shared Source Components
Third Party SW
Proprietary Technology Application
Multiple CA/DRM
STB

3.2 Low Level Software Integration

Custom drivers written for a specific proprietary middleware will be replaced by standard SoC drivers with a SoC vendor provided ‘glue’ layer to adapt to RDK requirements. In time this glue layer will probably cease to exist as sufficient volume of RDK-based products incentivise the SoC makers to deliver driver software with interfaces that directly conform to the requirements for RDK enabling a ‘glue less’ integration

OEM software such as factory/repair diagnostics can remain the same though it may be preferable to have a common diagnostic suite across all platforms if this is not already in place. New RDK STBs may use TR-069 based device management tools.

> “For migration it is necessary to map current functionality onto the RDK and understand how each feature can be implemented”

3.2.1 Low Level Drivers

RDK defines a lower layer which contains drivers and other low level software. Most of the software is typically sourced from System-on-Chip (SoC) manufacturers or OEMs. There may be other elements such as diagnostics or a loader provided by an operator to maintain consistency across different OEMs.

RDK is well supported by SoC vendors with many plans already announced. A key point is the shift away from proprietary interfaces towards de-facto standards such as OpenGL and DirectFB. Many SoC manufacturers will also provide plugins for RDK supported streaming framework such as GStreamer.

Over time the set of interfaces chosen by RDK are likely to form the core of a common driver solution offered by all SoC vendors based around Linux, Open Source Software and Open Driver Interfaces.

3.2.2 The Loader

The loader is a key mechanism. A typical boot loader combines a number of elements including hardware configuration, diagnostics, security features, remote and local software upgrade, disaster recovery and handover to an application image. There is often a communication protocol between loader and application images implemented using a shared data area to exchange boot parameters.

For legacy loaders deployed in boxes the key points would be:

- To understand how to format the RDK software image so that it can be loaded into memory by the existing loader. This would typically also require signing of the software image.
- To understand how the loader would communicate with the new RDK software stack.
- To understand clearly any protocol that needs to be implemented with the RDK based software to support the features of the loader such as communicating the location of a new version of software to be downloaded.

A newly developed RDK Loader would combine elements of a standard Linux loader with the specific elements needed for STBs such as upgrade mechanisms and diagnostics.

3.3 Application Migration

Proprietary Applications are migrating to open application frameworks such as those supported by HTML5 and JavaScript. Some traditional application and middleware functions would be shifted to the cloud and accessed as web-services over a secure channel.

Existing STB Applications are written in languages such as C, C++ or Java. They are often monolithic and not easily modified or expanded. New features take a long time to implement and deploy. Normally applications depend only on broadcast metadata and retrieve very little information over IP channels.

The RDK does not enforce an application framework and the upper layers of the software stack are not part of the RDK distribution. However, there is a method of application design which is easily supported by the RDK.

The thinking behind RDK represents a shift towards retrieval of centrally-stored metadata using web services over IP. Also other customer services can be implemented in web-orientated mechanisms. Business rules are implemented more in the cloud than in the application client on the STB.

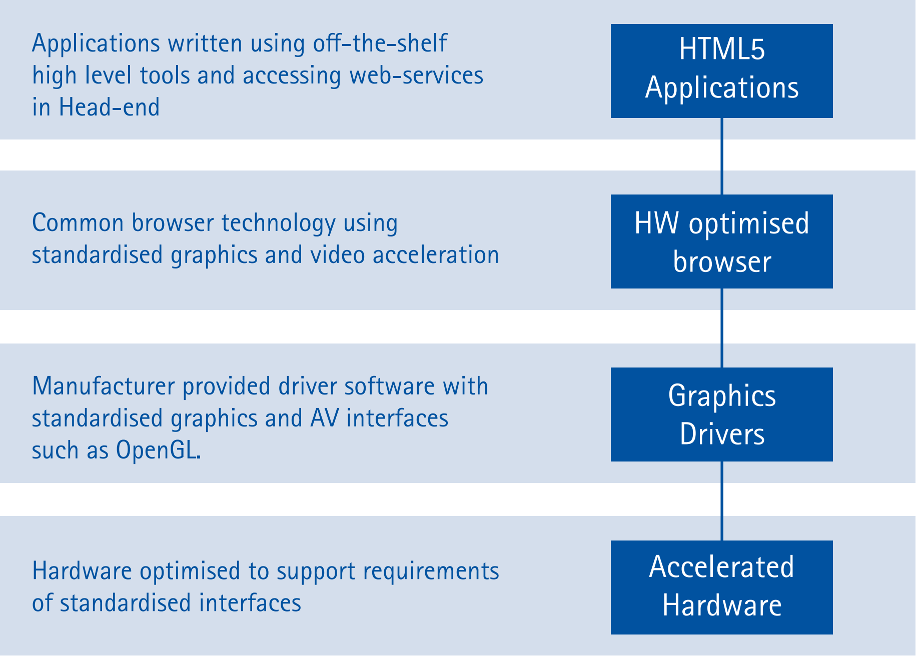
This requires web technologies such as HTML5 and JavaScript to implement new clients. The client applications can be lightweight with an emphasis on presentation of services rather than implementation of business rules and logic. Business rules and logic now exist primarily at the Head-end. This scheme has a number of advantages:

- The application can be changed quite quickly. If some of the application itself is retrieved from a server then it gives the possibility to change it on an almost daily basis.
- The operator can customise the services for users to give them a more appropriate experience or offer new services.
- Operator Product Management can easily experiment with the services provided.
- Applications can vary on a geographic basis.

You can choose not to follow this and implement your application in a more traditional way though you may not gain some of the advantages RDK is offering.

> "The thinking behind RDK represents a shift towards retrieval of centrally-stored metadata using web services using IP"

Figure 2: Key Features of Graphical Architecture for RDK



Applications written using off-the-shelf high level tools and accessing web-services in Head-end

HTML5 Applications

Common browser technology using standardised graphics and video acceleration

HW optimised browser

Manufacturer provided driver software with standardised graphics and AV interfaces such as OpenGL.

Graphics Drivers

Hardware optimised to support requirements of standardised interfaces

Accelerated Hardware

3.4 CA/DRM Integration into RDK

Conditional Access (CA) and DRM are crucial to protect the business models of operators. Support for multiple DRMs is required to support the multiscreen client devices in the home. Usually only a single CA solution is required for Broadcast.

The main issues involved in integrating any flavour of CA/DRM with RDK are as follows:

- Integration of existing or proposed CA into RDK. Most vendors will have a pre-canned integration with RDK though it should be checked in each case what features these support.

- Integration of any proposed DRM mechanisms. Again some will be pre-integrated with RDK.

- An important point is to ensure that the chosen CA and DRM can exist in a multi DRM environment especially if you propose to support standard clients such as Xbox, PlayStation, Android and iOS.

Integration of CA or DRM with RDK typically consists of three elements.

- Integration with any hardware using a set of driver interfaces. Typical hardware requirements might include smartcard, MPEG filters, Crypto hardware and embedded certificates. Most CAs and DRM are pre-integrated for many SoCs and those SoCs will support any specific requirements of the security provided.

- Integration with RDK middleware defined interfaces. This allows the RDK middleware to trigger the functionality of CA/DRM as required for playing protected content.

- Integration with Applications. This is not covered by RDK and will need to be defined for any Application/CA/DRM combination. However CA/DRM are becoming less visible to the user and the need to have a strong application interface is diminishing.

3.5 Head-end Migration

New Head-ends are migrating towards unified solutions. Multiscreen support is a must-have for operators and is the driving force for a unified Head-end. Existing silos of VOD, OTT and Broadcast are being merged and duplication of equipment is being reduced. The RDK philosophy implies a shift towards a unified head-end with more IP based functionality.

A typical Head-end consists of:

- Head-end access functionality to manage the client's access to Head-end.
- Operational Support Systems which provide most of the features of the Head-end.
- Business Support Systems which provide functionality for management of the business.
- There are typically three types of client:
  - Gateway – which provide the customer with access to system functionality in their premise.
  - IP Client – which allows customer to access content from the Gateway and also directly. Some clients may also support QAM functionality.
  - Unmanaged Clients – devices such as tablets, PC, consoles or phones where only an App is provided by the Operator.

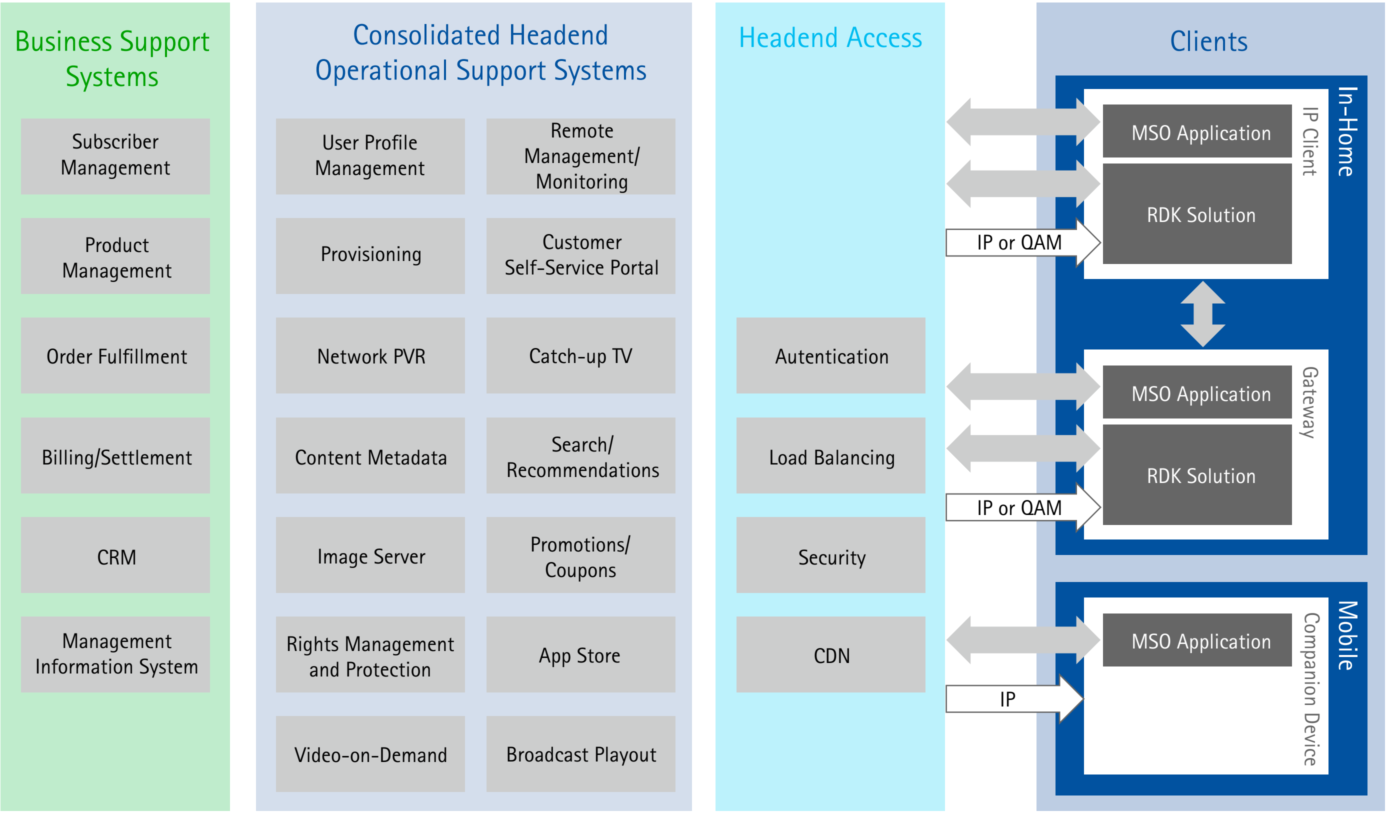
A major change of approach such as deployment of RDK offers the opportunity to create a more streamlined Head-end with less duplication between different silos of content. In addition the possibility to move common business logic into the cloud greatly simplifies client devices and makes the logic more easily modifiable

This paper does not cover all the possibilities here but some of the key issues to be addressed are as follows:

- What existing silos of similar functionality can be merged? How can the operation of the Head-end be streamlined?
- What existing legacy equipment should be maintained?
- How can the RDK middleware be configured to access the legacy equipment?
- How can clients' access to the Head-end functionality be provided as a web-service?
- What new features can now be added because of the extra flexibility? For example full customer self-service is a tremendous cost saving for organisations.
- What business logic and rules can be shifted to the Head-end or the cloud to simplify the applications on the client?

"A major change of approach such as deployment of RDK offers the opportunity to create a more streamlined Head-end with less duplication between different silos of content"

Figure 3: Architecture of a multiscreen unified Head-end



4. Accenture Digital Video and RDK

ADV provides services, products and IP to the Digital TV industry. We have almost 20 years' experience working with global operators, OEMs, middleware vendors, conditional access vendors and semiconductor companies.

ADV has both the expertise and scale to take full responsibility for entire programs covering multiple products through to executing smaller work packages at any point in the product lifecycle. Tools and re-usable components also form part of our offering and can help to speed STB integration and reduce deployment time while increasing the quality of the products being deployed.

ADV manages the official RDK code repository, ensuring the stability and integrity of code releases, on behalf of RDK Management LLC.

Critical to achieving the RDK vision of continuous improvement through community contribution is giving the members the ability to confidently rely on the stability and integrity of RDK code releases. In conjunction with key RDK architects and contributors, ADV has designed a rigorous code management process. We utilize best industry standard integration processes and automated configuration management to ensure that contributions to the common code base are carefully assessed, merged, and tested prior to being made available as part of RDK releases.

For further information on how to join the RDK community in order to contribute to or access the official RDK code repository, please visit www.rdkcentral.com.

5. In Conclusion

The RDK provides an excellent basis to create a modern STB which is fully featured and provides a platform for innovation for many years to come.

In this whitepaper, we have described the strategy that should be implemented when considering migrating to RDK. This paper should be considered a jumping-off point for some of the technical issues that should be addressed. When engaged in such a project, ADV draws on over 600 man years of driver development and software system integration on STB across multiple platforms for multiple networks to ensure the migration is completed successfully.

While migration is not something that is typically considered at the outset of a new STB development, it is something that is becoming more common in the fast changing world of content delivery. There is a drive to keep STBs in the field for longer coupled with a consumer demand for a better user experience which in combination drives a need for technology migration. Good STB migration planning is not a mere detail in the management of STB lifecycle: it is a pre-requisite.

"Good STB migration planning is not a mere detail in the management of STB lifecycle: it is a pre-requisite"

About Accenture Digital Video

Accenture Digital Video is a business unit within Accenture. We deliver business results for companies where video is of strategic importance, helping them pivot to capture new growth opportunities in an ever changing market. Partnering with clients, we use our agile methodologies, deep skills, and open technology platforms and apply them in every phase of a change journey—from thinking to doing. The end result: more predictability in the face of a complex and volatile landscape. Accenture Digital Video has a 20 year track record in driving video innovation through a global workforce of more than 2,000 dedicated professionals across strategy, delivery, business services and operations, all dedicated to helping clients grow profitably.

About Accenture

Accenture is a leading global professional services company, providing a broad range of services and solutions in strategy, consulting, digital, technology and operations. Combining unmatched experience and specialized skills across more than 40 industries and all business functions—underpinned by the world's largest delivery network—Accenture works at the intersection of business and technology to help clients improve their performance and create sustainable value for their stakeholders. With more than 375,000 people serving clients in more than 120 countries, Accenture drives innovation to improve the way the world works and lives. Visit us at www.accenture.com.

The contents of this document are owned or controlled by Accenture and are protected under applicable copyright and/or trademark laws. The contents of this document may only be used or copied in accordance with a written contract with Accenture or with the express written permission of Accenture.

All trademarks contained herein (whether registered or not) and all associated rights are recognized.

Copyright © 2016 Accenture
All rights reserved.

Accenture, its logo, and
High Performance Delivered
are trademarks of Accenture.