ALM for Engineered Products

The software economics of using PTC solutions for software-rich product development

Written by: Michael Azoff and Tony Baer

Published July, 2014, © Ovum

MANAGEMENT SUMMARY

INTRODUCTION

Embedded software is changing the nature of, not only products, but also the business of manufacturing. Besides making 'smarter products' possible, embedded software enables manufacturers to expand their goals as they can offer value-added services that raise the level of business with the customer. But adding embedded software to the product lifecycle poses numerous challenges. It increases the complexity of the product and the process for creating, manufacturing, and delivering it. There is now a new engineering discipline for software design that must coexist with mechanical and electrical design. Introduction of software also impacts the rate of change and with it, the burdens of change management and quality assurance. Moreover, the growing burden of regulatory compliance -- where the impact of each change must be traced and validated -- mandates a process discipline to the application lifecycle.

- Software changes the economics of products because it accounts for much of the innovation, uniqueness, and value of products.
- Embedded software development amplifies the challenge of change management and quality management in the product lifecycle because it allows product teams to innovate more rapidly.
- Application Lifecycle Management (ALM) disciplines become more critical than ever for dealing with the new complexities that embedded software brings to the product lifecycle.
- ALM facilitates the traceability that is essential for developing engineered products where embedded software plays a major role.
- Traditional ALM spans requirements management, architecture management, change and software configuration management, and quality management.
- A key ALM discipline is Model-Based Systems Engineering (MBSE), which allows distributed teams to collaboratively build digital models to drive better architectural decisions, accelerate design and development cycle times, and facilitate re-use.

TRANSFORMING PRODUCTS AND MANUFACTURING BUSINESSES

Originally an afterthought in engineered products, embedded software is claiming a larger proportion of the value, and in many cases, is the 'feature' that defines and differentiates products. The trend, which began with aerospace vehicles that required software to augment human control, has spread to virtually all engineered products to automate or enhance their function, or extend their capabilities. Embedded software:

- Gives smartphones their personality;
- Provides automobiles with enhanced control, and powers telematics systems that differentiate otherwise similar-looking vehicles;
- Powers self-diagnostics in industrial machinery;

- Enables physicians to perform keyhole surgery using smart, biomedical devices and surgical equipment; and
- Empowers large-scale smart city/utility/infrastructure networks to scale by localizing processing.

Embedded software has clearly become the innovation driver in engineered products. It also can become a game changer for manufacturers, who transform themselves from being a supplier of products to a supplier of higher-margin services businesses. For instance, heating, ventilation, and air conditioning (HVAC) manufacturers can assume new roles in managing climate control for their end customers; aircraft engine makers contract, not to sell motors, but to deliver service over the product's life; farm equipment manufacturers can network the equipment into connected systems to help farmers optimize all the operations of running a farm.

EMBEDDED SOFTWARE BRINGS OPPORTUNITIES AND CHALLENGES

While embedded software in products brings opportunity, it also presents huge challenges. It adds complexity, both to product design and to the product development and product update processes. It adds new variability to electromechanical products where operation and specifications are directly impacted by software. And, because software does not necessarily involve a complex supply chain, it enables change (at any time right up to the point of manufacturing) and variability that can enhance a product —but also add testing and quality issues.

Engineers are facing rising overhead costs in managing increasingly complex products with multiple variants and change and upgrade requirements. The work processes currently deployed are also disconnected, with manual, labour-intensive steps conducted at various points to bridge automated workflows. Electrical, electronic, mechanical, and software all have their respective work processes, and the biggest gap to overcome is between hardware and software. These disconnected workflows are also difficult to change and maintain, often using custom tools to manage part of the process. Embedded software development tends to suffer the most in such an environment as it's the newest and least understood technology, certainly across engineering as a whole within an organization; many middle managers see software as a black box and have no understanding of what a mature software development practice entails.

In the last decade, there have been over 2000 new regulations and standards imposed on manufacturers. Associated with this compliance is the cost of demonstrably proving compliance. PTC's approach is to focus on functional safety compliance and this maps well to good practices such as Capability Maturity Model Integration (CMMI) and Software Process Improvement and Capability Determination (SPICE, part of ISO/IEC 15504). The greater the automation in workflows, the easier it is to apply change and configuration management across the five pillars of risk management, architecture management, requirements management, test management, and code management. Traceability from requirements to architecture to code was previously optional but this is changing: software change traceability is now essential. The problems of safety and compliance cannot be solved by testing more, as engineers have to improve their designs (building quality in) and how their processes are managed. For example, safety and security have to be thought about at the very beginning, at the design stage – you do not inject safety and security into a finished product. There is a world apart between an e-commerce server going down and terminating a business process, and a defect causing a product to malfunction which is used by or affecting human beings. These questions will be raised by an order of magnitude as Internet of Things (IoT) technology begins to permeate everyday life.

Characteristics of a perfect storm about to happen are the multiple levels of validation, verification, and regulatory oversight, the complex variance management, and managing the software embedded in products,

all imposed on legacy work processes and tools. Such a burden will give rise to an inevitable collapse: product failures and recalls, and reputation harm for the manufacturers.

Engineers need a reliable process and tools to support that process which can cope with the pressures and complexities of engineered products with high software content.

SOFTWARE ECONOMICS OF ALM IN A PLM WORLD

SOFTWARE IS TAKING OVER ENGINEERED PRODUCTS

Manufacturers are looking to reduce their overall costs in product development and use of embedded software is a means toward that end. The cost to manufacture a product is essentially a hardware cost, and so the shift to fewer electronic control units and more software in them reduces the overall product manufacturing costs (software is cheap to copy and so has negligible manufacturing cost). Reduction in costs is one important part of the economics driving the expansion in embedded software. Engineered product manufacturers are also looking to reduce hardware costs by moving from unique custom built hardware to commoditized hardware wherever possible. Such hardware standardization is often exploited internally within a manufacturer's product range. Overall, software is where the main cost reductions are being realized.

In addition, use of software expands innovation with new possibilities. For example, in-vehicle infotainment (IVI) has swiftly grown to become a major criterion in consumer car buying choices. However, the upsides in use of embedded and application software in products are not without risks. None of the benefits of software usage will be realized if software development is not managed properly. Software defects can act as ticking time bombs; the lack of investment in appropriate process and management maturity in software development can rebound and seriously burn an engineering manufacturer.

The evidence to date is that there are critical differences between the best and the average engineering firms: average manufacturers are immature in their understanding of how to manage software. For example components are tested in isolation and pass QA, but when they are assembled into larger modules and finished products, failures are then discovered. These systems-related defects are the most difficult to avoid because they are only seen in completed assemblies. Agile development can help here through the practice of iterative design and development, with continuous integration and testing as a critical element of the work process. Architecture management across product lines is essential as models, simulations, and prototypes can all feed into the systems testing during design and help identify multi-component complexity related issues.

THERE ARE COST BENEFITS IN MANAGING SOFTWARE COMPLEXITY

Software complexity is a characteristic of software intensive systems. As the number of lines of code embedded in products continues to grow, the complexity problem grows in tandem. The lack of ALM tooling turns this challenge into a problem. Disjointed tools and repositories prevent traceability and integral reporting. The lack of support for agile processes results in errors found late in the development cycle, raising the cost to repair.

There are three major areas of complexity in engineered products:

- Product-line complexity has risen due to the need for personalization and globalization of products. While cost savings are possible with re-use of hardware and software product

architecture, the lack of good process and tools leads to high overheads. This can be a huge issue. Conversely, manufacturers that let engineers re-create the wheel many times over, result in massive variant management overheads.

- Disconnected processes, where engineers use different processes and tools, creates unnecessary overhead and inhibits collaboration across globally distributed teams and the supply chain. Too many engineering silos exist, creating barriers to the transition to agile processes. Part of the problem stems from senior management's view of software as a black box, with the consequence that embedded software development is disconnected from the rest of the product delivery cycle.
- Compliance costs increase if there is no automated process and enforcement for controlling access. Compliance governance requires audit trails of who changed, what, and when. Without automated processes and tools, the overhead costs of monitoring compliance can skyrocket.

Given these cost drivers, manufacturers are increasingly focusing on four key practice areas that have been empirically proven to help manage software complexity and reduce software and product development cost.

Systems engineering

Systems engineering practices can yield great value for software-intensive product manufacturers in addressing this exploding complexity. In a 2008 paper by Barry Boehm and others, it was found that increasing investment in systems engineering could lead to 40% shorter product delivery schedules and 30% lower development costs.

Model-based systems engineering

Mature architectural modeling practices can reduce the time, cost, and effort required to create and maintain groupings of similar products. More advanced organizations are moving beyond simplistic asset re-use paradigms to a model-based product line engineering approach that views product lines as a single system, with designed-in variations, rather than a collection of products. Compared to non-modelled approaches, the resulting cost savings can be significant. A recent survey commissioned by PTC of 667 systems engineering respondents shows that mature product-line engineering approaches were associated with a 62% reduction in development cost per project, with 23% more projects delivered on time.

Requirements engineering

According to the Software Engineering Institute, nearly 20% of the cost of the average software intensive system project is due to rework that is directly attributable to inaccurate, incomplete, or ambiguous requirements. If you break down the numbers, the stats show that 30-40% of the average project is rework and that 50-70% of that rework is caused by requirements errors.

ALM lifecycle traceability

Traceability is one of the chief benefits of ALM, and in an ALM-PLM context this includes quite deep traceability across the ALM and PLM divide. An example engineering company was able to realize as much as 30% reduction in design and development lifecycle time due to automated traceability. The resulting ROI was one year.

There is a paramount need for ALM in the systems engineering space to reduce complexity and thereby reduce costs. The aerospace and defence (A&D) sector has been the most advanced in adopting leading

technologies for many years because of significant contract compliance requirements. However, they are now being leapfrogged by other sectors such as automobile and high-tech electronics, where the need for fast time to market has forced the issue. Across all engineering industry sectors, there are opportunities to raise process maturity and make better use of automation through ALM and PLM solutions.

MEASURING EMBEDDED SOFTWARE QUALITY IS THE CHALLENGE

Typical manufacturing regulatory requirements are based on measurement. For instance, in mechanical engineering, proving a requirement for an unblemished surface finish is handled by taking a direct measurement of the finish. The problem with software is that there is no direct measurement of its quality, there is no guarantee or absolute proof that a piece of code will do what it claims it does. Even the many metrics that have been devised to measure code complexity have not empirically been proven to reduce defects. To date there is no software equivalent that exists in the physical world where direct measurements on the end product suffice to guarantee quality.

Therefore, the approach taken with standards regulating embedded software is to demonstrate that the *process* of producing software is a good one, with the hope that if you do the process right then the product will be right. Unfortunately, you can follow the process rigorously and still end up with a poor quality product. Analysis of software quality has looked at architecture, code structure complexity, use of formal methods, and use of models, all with the intention to help industrialise software development quality. However, software development remains a craft activity and the engineering industries are reliant on the professionalism of the developers to create high quality code. In time, this may change and good direct measures may become available, at which point the regulatory environment will shift from one of ensuring the process is being followed to one where the quality of the output is proven to meet a certain standard.

One of the advantages of software is that if a manufactured product develops issues, the problem can often be corrected by a software change. The challenge is with late breaking requirements, which sometimes have to be squeezed in, and which is a quite normal occurrence. The resulting schedule pressure is found to be best addressed by the agile practice of running software regression unit tests within a continuous integration and testing system, and which can be done in real time against each change. This has become important as software engineers do not have to think about which tests to run; today all tests are run because it is cheap enough to do with every change. In conclusion, this approach relies on maintaining a high quality unit test suite, and the benefit is that it gives a high degree of confidence that any software change has not introduced a new problem.

ALM AND THE BUSINESS BENEFIT OF RISK REDUCTION

Central to risk reduction is the ability to surface potential problems and make nuanced trade-off decisions earlier in the project lifecycle, when they can drive downstream activities with less disruption and cost. To achieve this, organizations are increasingly adopting MBSE approaches that enforce attention to architectural detail earlier in the project lifecycle. By integrating visual modeling and model simulation for system and software design, organizations can collaborate more effectively, facilitate earlier design reviews, bring systems to market faster and reduce design errors, leading to a significant reduction in overall project costs.

One benefit of ALM-based traceability is that it facilitates certification and auditing compliance before an incremental release. Furthermore, this live traceability makes it possible to use project reports with drilldown capability in the course of everyday work, so that engineers can keep a running review of how work items are progressing. Auditing can and should occur at any time, which further reduces the risk of both software and

product non-compliance. Furthermore, there is no need to make assumptions and guesses because data is available in real time, and is therefore always up to date.

Integrated tooling also improves how the engineers work by bringing together disparate sources and making them available simultaneously, so that engineers don't have to spend time searching for information or artefacts. So there is efficiency improvement as well as risk reduction.

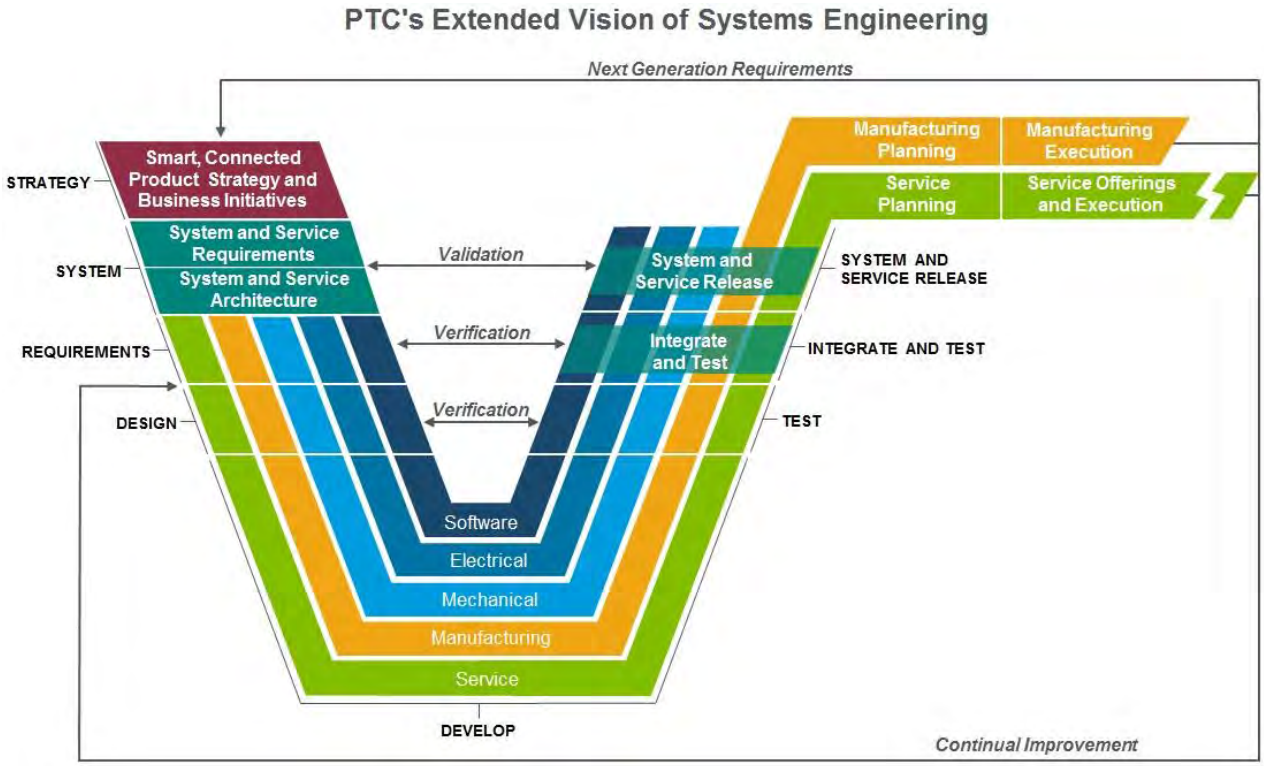
Process compliance can be improved with ALM because it can be enforced by workflows. For example a PTC customer's Project Management Office (PMO) obtained around 70% of the data it needed for gate reviews by extracting it automatically from the ALM solution, where previously this was done manually. This allowed the PMO to spend more time on their chief role of analysing information and assessing risk.

PTC'S HOLISTIC APPROACH TO SYSTEMS AND SOFTWARE ENGINEERING

INTEGRATED SYSTEMS ENGINEERING SOLUTIONS

A single solution that encompasses both ALM and PLM and is able to integrate the complete engineering lifecycle across hardware and software will have distinct advantages, and PTC is bringing that vision into reality. With its most recent acquisition of Atego, a developer of model-based systems and software engineering solutions, PTC now has the capabilities to bring together an end-to-end view of the systems engineering lifecycle that spans requirements and architecture definition through software engineering, physical product definition, and system verification. The acquisition further solidifies PTC's mission to enable closed-loop product development and delivery from product inception through design, manufacturing, delivery and service – breaking down the traditional barriers that separate ALM, PLM, and service lifecycle management (SLM) processes.

Figure 1: PTC's iterative engineering model



Source: PTC

Manufacturers are faced with possessing multiple product development solutions, some of which have been in place for decades and which encapsulate deep knowledge. There will be opportunities to replace tools with new generation ones, such as when legacy solutions meet their natural point of usefulness. In addition, as discussed above, many existing solutions have gaps, which new generation tools are able to bridge with automated tools that reduce complexity. PTC's approach is to support open, standards-based integrations with third-party solutions, enabling customers to create integration hubs that can govern and version control lifecycle data across heterogeneous ALM and PLM tool chains. Large engineering organizations have highly specialized silos of work excellence across the various engineering disciplines and so typically an integration hub will be required within each silo.

CAPITALISING ON THE INTERNET OF THINGS

Once in place, an integrated lifecycle management solution allows for workflow automation that enables traceability of work items across the product lifecycle. PTC's lifecycle traceability and integration will extend to the IoT with PTC's recent acquisition of ThingWorx, which will complete the backbone that allows the collection of data from physical products and devices. Whether the edge device is a sensor or an enterprise system, it will look and behave as a seamless extension of the product lifecycle's data. The collected data in ThingWorx, together with data from PTC Windchill and PTC Integrity, can be mashed up as well as sliced and diced for analysis and intelligence gathering, within the new platform being developed by PTC.

PTC is able to bring an engineering perspective to ALM and understand the needs of PLM users, and in turn bring the maturity of ALM approaches to the PLM world. The future is that as engineering firms learn to manage the complexity of embedded software, they will gain a better understanding of systems of systems, positioning them for success in a world where connected systems span the globe and distant products are

managed and serviced remotely. ALM-PLM integration is a necessary foundation for building products that are safe, secure, and built for the age of IoT.

Integrating ALM and PLM systems involves bi-directional flow; an embedded software manager may receive a product-level problem report illustrating mechanical and software components; the software component will transfer across to the ALM solution, while the mechanical component is transferred to the PLM system. Any electronics changes will migrate across to the PLM system because it manages the circuit designs. The compliance data that comes back from electronics has to be fed into the ALM system and then the compliance data from software has to go back into the whole product record, the Bill of Materials, in the PLM system. A bi-directional automated solution will significantly reduce process friction.

EVOLVING THE V-MODEL WITH ITERATIONS

Typically, engineers have a lack of visibility into the embedded software development process due to legacy tools that lack integration and traceability. Key product delivery milestones may be missed as engineering management discovers that software has a greater velocity of change than hardware. The proliferation of product variants adds to project complexity, and the lack of inter- and cross-team collaboration makes it difficult to accelerate productivity. Engineers are aware of agile and lean software development methodologies, but find it difficult to raise their maturity level to what are highly disciplined practices. Iterating the V-model is only possible if the right tools are in place to support the new process.

An integrated ALM solution is able to support iterative development and incremental delivery of components. Use of modelling and simulation is critical to support iterations during the design phase in order to accelerate product development in an agile way. PTC Integrity is a platform that allows engineers to apply a configuration layer to support whatever ALM process is required, whether the traditional V-model or an iterated V-model. These configurations are available as pre-built templates that users can customize. The agile ALM process is preconfigured to support Scrum. A disciplined product development approach can combine traditional and agile configurations so there is traceability between the system level and product requirements through to their decomposition into the Scrum process, and iteratively deliver against those product requirements.

The integration between ALM and PLM with PTC's solution facilitates agile practices across both the hardware and software development streams. Many engineering firms already implement lean manufacturing practices in their mechanical and electrical engineering, and agile-embedded software development can complement these work streams. Scaling agile to large engineering projects is only possible by using tools that support the more disciplined agile processes.

A systems or software engineer will typically find five main touch points between ALM and PLM tool sets:

- **Requirements engineering** where software artifacts are authored in PTC Integrity at multiple levels of decomposition: system level, product level, component level. There will be some portion of these requirements decompositions that need to be traceable into PTC Windchill, so those particular requirements can be traced to their mechanical and electrical design implementations, or allocate subsystem requirements, link to CAD specifications, and managed in PTC Windchill.
- **Systems design** where architectural alternatives are explored and optimized through visual models and simulations. This step allows organizations to understand design trade-offs and simulate system behaviour earlier in the product lifecycle, as a means of exercising design

solutions and discovering potential problems earlier in the product lifecycle, when they are significantly less costly to fix. By generating code directly from visual models, organizations accelerate development cycles while ensuring the code complies with the intended architecture. These activities cross both PLM and ALM domains.

- The **change management** process needs to link across both ALM and PLM to capture changes relevant to both systems. For example, when releasing software to the Bill Of Materials product configuration (the system of record) there needs to be a record of what issues were addressed in the release, and any defects outstanding.
- **Configuration management** needs to keep software and product configurations linked and traceable to source code, as well as changes that went into the software configurations.
- **Test cases** need to be traced to requirements and will need to be linked to tests of complete hardware systems, so ALM and PLM QA is managed holistically.

The current pain points for engineers are around managing these touch points, and without automation line of thought work is inhibited. An integrated and automated ALM and PLM solution solves these problems and creates new ways of working, letting engineers search and retrieve accurate and timely information, whenever they need it. In addition, systems engineers need the support of ALM and PLM tools so that they can improve design and development processes, and extend the V-model with iterative work patterns that deliver feedback fast, allowing work to be steered to meet requirements most effectively.

THE ADOPTION OF ALM IN ENGINEERING MANUFACTURING

Engineering companies vary a lot on where they are on the ALM adoption and integration path, and changing entrenched tools in an engineering environment is a major challenge. Unlike enterprise IT, where upgrading to the latest software version of various software applications and packages is normal, in engineering there is a much higher cost associated with process and tooling upgrades, which acts as a barrier. Products in the field may have lifetimes of over 30 years, and the need to support old versions of software results in higher costs of compatibility testing. Transitioning to ALM is an essential priority for engineering companies today, but there is an initial change cost. An ALM system such as PTC Integrity can automate the traceability of requirements to deployment in just one solution. The benefits of traceability will reduce software development costs product by product, so initial cost reductions will continue to multiply with continuous use of ALM.

Any changes in management systems need to be planned as the smallest change may have a large impact. On the PLM side, there may easily be one million drawings managed as assets, so a change in the underlying file format will turn into a massive amount of work, not only with transitioning file formats, but also with the need to check an appropriate fraction to ensure that nothing has gone wrong. The way forward is to enable integration while also looking at opportunities to replace legacy tools when they reach a natural end, if not sooner.

CONCLUSIONS

The trend in embedded software management is a greater need for both rigor and speed. The regulatory environment has intensified across the range of engineering practices. In addition, there is greater appreciation of the role of software in products, with software content in products rising exponentially. As a result, embedded software management is now under the spotlight from both a process view point and a quality perspective.

Embedded software and hardware manufacturing is both architecture- and requirements-driven, and organizations that invest in improving their maturity in these two disciplines will reap the rewards of faster cycle times and improved product quality. With embedded software, satisfying regulations requires traceability: being able to trace requirements through to test case generation, test execution, inspection of test results, and to code delivery. Therefore, it is necessary to integrate all the various project data for development reviews, which facilitates audits and makes it easier to demonstrate compliance (compliance requires audit detail of 'who did what and when'). In some engineered products, 90% of function logic is now in software, where 30 years ago clockwork mechanical parts were used. If the software does not work as required then there is a huge problem.

Senior executives in engineering companies need to see software and systems engineering as an essential competency that spans traditional ALM, PLM, and SLM silos. As the amount of embedded software grows in engineered products, the successful companies will be those that have understood the critical nature of software and the need to manage software complexity with a holistic approach. How companies rely on ALM and PLM tools and practices will be a key differentiator.

APPENDIX

METHODOLOGY

Ovum has extensive experience and research in ALM solutions and conducted interviews with engineers who are engaged in using and integrating ALM and PLM systems for software-rich engineered products.

AUTHORS

Michael Azoff, Principal Analyst, Software Solutions Team

michael.azoff@ovum.com

Tony Baer, Principal Analyst, Software Solutions Team

Tony.baer@ovum.com

OVUM CONSULTING

We hope that this analysis will help you make informed and imaginative business decisions. If you have further requirements, Ovum's consulting team may be able to help you. For more information about Ovum's consulting capabilities, please contact us directly at consulting@ovum.com.

DISCLAIMER

The facts of this report are believed to be correct at the time of publication but cannot be guaranteed. Please note that the findings, conclusions, and recommendations that Ovum delivers will be based on information

gathered in good faith from both primary and secondary sources, whose accuracy we are not always in a position to guarantee. As such Ovum can accept no liability whatever for actions taken based on any information that may subsequently prove to be incorrect.