Achieving Application Resilience:
Maximising investment through productive and predictable model-driven development
Table of Contents

SECTION 1: 3
Executive Summary

SECTION 2: 4
Understanding application modernization
Comparing modernization productivity and predictability 6
Model-driven projects deliver greater productivity 7
Model-driven projects are more predictable 8
Model-driven projects are of higher quality 11
Comparing modernization application investment lifecycles 11
CA Gen and model-driven modernization 14

SECTION 3: 17
Conclusion
About this Whitepaper 17
About Encode 18
About Total Metrics 19
About the ISBG 20
References 21
Section 1:

Executive summary

Never before has the modern enterprise had so much choice when it comes to options for implementation of new business applications. Gone are the days of IBM’s monopoly in the data centre and Microsoft’s dominance of the desktop. Today, organizations can choose from an array of hardware architectures, operating systems, database platforms and development languages; and the choice is staggering.

Take programming languages. Since the advent of mainstream computing in the 1950s there have been over 600 recognized languages produced of which Java, C/C++, C#, Visual Basic and PHP today’s most dominant. Similarly there continues to be a plethora of databases ranging from the workhorses of relational technology in DB2 and Oracle to the increasing array of NoSQL-style solutions.

Compounding the challenge of choice is the fact that all of these combinations of hardware, database are not only available from internal information technology (IT) services functions, but also as-a-service from external third party providers offering cloud-based solutions.

For many enterprises the ability to exploit this change in the delivery of IT services is currently beyond their reach. The presence of large critical business applications, that despite the best intentions of the enterprise, no longer remain at the forefront of supported mainstream technology represent an ever increasing technical debt. At some point this debt must be paid; often with replacement of the entire solution.

Replacement of existing solutions cannot be undertaken lightly. As the Chief Information Officer of Australia’s largest bank is reported to have said, replacing core systems within an organization is like performing “open heart surgery”. A sentiment echoed by many large organizations that have come to rely on such systems as a means to deliver customer service, increase profits and provide critical information.

Despite the obvious risks such endeavours present, organizations can improve their chances of success by evolving or modernizing rather than replacing core systems. However, not all modernization approaches are created equal. It therefore crucial that an accurate understanding of the actual costs of different modernization techniques is understood.

Encode Services, on behalf of CA Technologies, compared two common modernization approaches by investigating the productivity, predictability and potential costs of manual re-coding using native third generation languages (3GL) versus model-driven development using tool such as CA Gen. The research revealed that the 25% of organisations that utilise model-driven development achieve as much as 38% more productivity per development hour when compared with native 3GL development. When considered in the context of the five year useful life for a medium-sized web-based application the reduction in initial development and ongoing operational costs could be as much as $500,000 per application.

In addition, the research identified that employing model-driven develop presents ongoing benefits including more predictable projects, a higher level of software quality and significant savings when generational change inevitably occurs within an organization’s technology landscape.
Section 2:
Understanding application modernization

The long useful life of business applications cannot be underestimated when it comes to IT investment decisions. Anecdotal research across the finance industry and public sector indicates that medium to large enterprises retain core business applications for around 10 years, with known cases older than 20 years.

McKinsey & Company recently indicated that many core banking systems were commissioned in the 1970’s and 80’s making them between 30-40 years oldiii. In contrast, for the purposes of calculating capital formation, the Australian Bureau of Statistics (ABS) uses an average of between 6 to 8 yearsiv.

With these timelines in mind, it is not surprising to discover that over time organizations face challenges which lead them to consider overhauling their applications. Challenges including, but not limited to:

- **Application integration** within the organization itself or between the organization and its partners.
- **Platform migration** in order to shift from potentially high cost centralised environments, such as legacy mainframes, to lower cost commodity distributed environments such as x86 server farms.
- **Regulatory compliance** that will result in significant changes to the existing business processes embodied within the application.
- **Complexity reduction** by addressing high levels of technical debt incurred through interim solution, rushed changes, poorly documented code and other decisions that over time make an application harder to maintain.
- **Market agility** which demands that applications can be modified within shortened business cycles and exploit opportunities measured in days not years.
- **Technology change** that requires organizations to assess and adopt new technical paradigms driven more by the demands of individual consumers than institutional investment decisions.
- **Skill shortages** arising from the natural migration of developers away from older to newer technologies leaving fewer personnel to maintain existing applications.

Application modernization can be thought of as the process of reusing existing application assets by applying a process of conversion and maintenance. The intention being that the applications would see their useful life extended through the significant enhancement of business capability, use of new architectures and exploitation of current technology platforms.
This process of conversion and maintenance takes various forms each with their own unique advantages and disadvantages. Common modernization techniques include:

- **Re-writing** which involves the manual hand-coding of a new custom system natively using a third general language (3GL) supported by an integrated development environment on a more “up to date” platform. E.g. Redevelopment in C#.

- **Wrapping** which re-uses existing code by encapsulating it within a more “up to date” platform which can be integrated to new user interfaces or connectors. E.g. Using the Java Connector Architecture to produce Web Services from a Natural-based application.

- **Packages** can be purchased and modified to mimic the existing application as close as possible. E.g. Replacing a custom financial system with SAP.

- **Code conversion** which involves programmatically translating the old code from the “old” 3GL to a newer more modern language. E.g. Conversion of COBOL to Java.

However, there is another option which is to exit the technology driven merry-go-round and adopt a model-driven development approach as part of the modernization approach. Such an approach involves the automated extraction of business rules, data structures and documentation into a set of platform independent models. These models can then be combined with 100% code generation to a choice of up to date technology platforms.

Once an application has been successfully ported to a model-driven development environment, further modernization is achieved through two main activities:

1. Modification to the platform independent models to address business challenges such as regulatory compliance and market agility. This process does not require in-depth knowledge of the underlying platforms, middleware or languages helping to address the skills shortage challenge.

2. Re-generation of the application to new technology platforms overtime to address the technology change and platform migration challenges.

Although the advantages of model-driven modernization are well documented and published by independent organizations such as the Object Management Group (OMG) and major vendors including IBM, SAP, Oracle, Microsoft and CA Technologies, many organizations still make the decision to re-write their applications as part of a modernization effort. But at what cost?
Comparing modernization productivity and predictability

Any form of application development contains inherent risks. Project failures are well documented and are often the result of many factors. However, a common theme is the difficulty in estimation of time and therefore cost. Indeed, 49% of organization experience at least one project failure per annum and at least 30% of projects exceeded their original schedule and budget\(^viii\).

As a result any application modernization needs to be undertaken with an accurate understanding of the different levels of productivity and predictability each approach entails.

Encode sought to compare the relative productivity of model-driven versus native 3GL projects using Release 11 of the International Software Benchmarking Group’s (ISBSG) Repository.

The ISBSG Repository is the only public domain software metrics repository that provides subscribers access to the raw source data to allow them to process the data according to their project characteristics and business need. Comprised of data submitted on a voluntary basis by organizations and project teams that have relatively ‘mature’ software measurement practices, the repository has the following overall profile:

- Sourced from 24 countries, “built” in 29 different countries 70% of the projects are less than 9 years old;
- 59% Enhancement, 39 % New Development, 2% Re-developments;
- 64% are either Management Information Systems or Transaction Processing Systems; and
- 17% of the projects are web developments.

The processing of the ISBSG Repository, undertaken by Total Metrics, an internationally recognized and independent software metrics consultancy, focused on projects which utilised C#, Java or COBOL as the basis of investigation for native 3GL projects alongside projects that were identified as being model-driven\(^ix\). The result was a detailed analysis of 508 projects covering productivity of both enhancements and new developments for both project types (model-drive and native 3GL) which occurred since 2000 and passed the ISBSG’s data quality assessment process. Specifically, the analysis explored 189 model-driven and 319 native 3GL projects.

The analysis also explored the difference in productivity of the two approaches across different sizes of projects. This enabled an analysis of the predictability of model-driven development versus native 3GL projects; a factor critical for minimizing the risk of time and therefore cost overruns on modernization projects.
Model-driven projects deliver greater productivity

Projects vary in size and therefore relative productivity as complexity increases. The analysis considered the relative productivity of model-driven versus native 3GL across the following categories of projects, as defined by their function point count according to the International Function Point User Group Counting Practice Manual Version 4.2:

<table>
<thead>
<tr>
<th>Project Category (Function Points)</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small (50-200)</td>
<td>Enhancements to existing applications</td>
</tr>
<tr>
<td>Small-Medium (201-500)</td>
<td>Small Web-based Applications, Interactive Voice Response Systems, Online Calculators</td>
</tr>
<tr>
<td>Medium-Large (501-1000)</td>
<td>Workflow Management Systems, Catalogues, Registers (Items or Events), Medium Web-based Applications</td>
</tr>
</tbody>
</table>

Across each of these project categories productivity was assessed based on Project Delivery Rate (PDR), the industry standard metric used by ISBSG to benchmark productivity. Where:

$$PDR = \frac{\text{Effort (Work Hours)}}{\text{Functional Size (Function Points)}}$$

**Important note:** This metric is the inverse of the more conventional ‘productivity’ parameters in that the lower the PDR the higher the productivity and vice versa.

For most project sizes ranges the difference between model-driven and native 3GL was relatively small, except for small to medium applications, including web-based applications where model-driven projects are as much as 38% more productive. That is, for each hour of development time on a small to medium web-based application project a model-driven project will deliver nearly 40% more functionality or function points (FP) (see Figure 1 below).

When considered at the overall level across all project categories the data indicates that model-driven development offers 10% more functionality per hour than native 3GL.
It should be noted that for large projects (>1000 FP) the overheads involved with managing large teams sizes, and multiple stakeholders are most likely to be the contributors to the decline in productivity for both model-driven and native 3GL projects.

Model-driven projects are more predictable

By comparing project work effort versus size of the project it is possible to establish a relationship between these two variables to determine the relative predictability of model-driven versus native 3GL projects using linear regression techniques\textsuperscript{xii}.

In the case of the ISBSG Repository the analysis considered new development projects, as enhancement projects show considerably more variability in results due to the variation in the functional impacts of maintenance activity.

The results of this analysis shows that the project size of model-driven project for a new development is a better predictor of project work effort that the project will consume (R\textsuperscript{2} = 0.7 as per Figure 2) than the corresponding relationship for native 3GL new development projects (R\textsuperscript{2} = 0.5 as per Figure 3). In addition, this predictability for model-driven projects is visible more consistent in large scale projects than their native 3GL equivalents.
This is determined by considering the R² value, shown in the Figures below, which is a measure of how much of the variability between the different projects is actually explained by the linear regression equation shown at Y. The maximum value for R² is 1.0 which would occur when every project exactly matched the equation Y. Therefore, a highly desirable value for R² is >0.5 given that the closer the value is to 1.0 the better. On this basis values such as 0.8 are considered to demonstrate a high correlation between the size of the project and the effort resources consumed. i.e. The capability of project size to predict the work effort that is consumed by the project.

Figure 2
Total Metrics also analyzed the spread of PDR for new development projects across five (5) percentiles and concluded that in addition to productivity increasing for model-driven projects in the size range 501-1000 FP the productivity rates achieved across projects becomes more consistent. That is the spread of percentiles decreases. This is also observed in the 1001-5000 FP size range. A narrowing of the percentile productivity spread has implications for estimating accuracy. Specifically, for model-driven projects, project work effort becomes more predictable as the effort overheads involved in the establishment of model-driven environment contribute a smaller proportion of the overall project effort on larger projects.

Given that scope creep is a significant risk for any project, but even more so when business users are presented with the opportunity to address the limitations of a system they may have “put up with” for a decade or more, ensuring predictability in estimates is a critical factor for success of any modernization effort.
Model-driven projects are of higher quality

While not formally included within the analysis undertaken by Total Metrics, in 2011 ISBSG released an updated version of the white paper *Techniques and Tools—Their Impact on Projects* which found:

- 322 projects indicate use of some type of model-driven development tool.
- The main impact of the use of model-driven development was a significant reduction in defect density.
- Model-driven projects were associated with a greater percentage of the project effort being spent on business facing specification activities (18.1% compared to 5.9%), while a smaller percentage of the project’s effort was spent on the build phase (43.4% compared to 56.3%).
- More importantly the build phase effort was very low or zero for projects using model-driven development environments offering 100% code generation.

Despite the benefits of productivity and predictability as previously discussed arising from Encode’s assessment of the ISBSG Repository, the ISBSG’s own analysis found that 75% of companies that submitted data to ISBSG did not use model-driven development. When the ISBSG and Encode findings are taken together it appears that the 25% of companies who have adopted model-driven approach to modernization are at a clear competitive advantage.

Comparing modernization application investment lifecycles

Armed with an understanding of the relative productivity of model-driven versus 3GL modernization approaches it is now possible to model the cost impacts of these strategies over time.

As previously outlined most significant business applications can last for at least 10 years. However, for the purpose of this example we will assume that we have an application with the following characteristics:

- Medium Legacy Application which has been sized at 500 FPs.
- It was developed and maintained in COBOL at a PDR of 10 hours per function point.
- Our developers cost $100 per hour.
- We budget annual maintenance costs by using a benchmark for commercial-off the shelf (COTS) solutions which is on average 20%.
The resulting model is that the application cost $500,000 to develop and $100,000 per annum for a total cost by Year 5 of $1,000,000 as shown below (Figure 4):

If in Year 6 we intend to modernise this application as it is, then based on the PDRs for model-driven versus native 3GL the results and assuming an identical hourly rate of $100 for developers would be:

<table>
<thead>
<tr>
<th>Modernization Approach</th>
<th>Cost Calculation</th>
<th>Variance on Original Investment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-driven</td>
<td>500 (FP) x 6.6 (Hours/FP) x $100 = $330,000</td>
<td>-34% (Saving)</td>
</tr>
<tr>
<td>Native 3GL</td>
<td>500 (FP) x 10.8 (Hours/FP) x $100 = $540,000</td>
<td>+8% (Cost)</td>
</tr>
</tbody>
</table>
Clearly a shift from COBOL to model driven delivers superior savings over a re-write in a 3GL environment. When modelled over time the savings become even more pronounced if we consider the reduction in on-going maintenance budget as shown below (Figure 5):

![Application Lifecycle Costs Compared](image)

In addition to developer productivity any modernization business case based around a model-driven approach should also consider the following factors and the savings they represent. Each factor should be analyzed in the context of the organization to determine the exact costs will generate using native 3GL development practices:

- Architectural transformation from old to new technology approaches over time such as the shift from mainframe-centric to client-server, web to mobile, and transactional to social computing.

- Platform transition both from old to new, but also from different products and versions within products such as changes in development frameworks, middleware, database, operating system or development language over time.

- Development complexity both in terms of training of staff, but also management of the various non-functional activities such as management of development frameworks, build and deployment of code.
CA Gen and model-driven modernization

Although the results of the ISBSG analysis demonstrate quantitatively the advantages of a model-driven approach to modernization these results are by their very nature “averaged”. Encode therefore sought to determine if the experience of CA Technologies customers utilising CA Gen was consistent with these findings through a review of six existing client feedback and two detailed structured interviewsxv. This analysis identified that CA Gen customers experienced development activity that was highly productive and predictable leading them to describe their development as “agile” and “responsive”. Furthermore, CA Gen customers experienced resilience in their applications both in terms of their ability to absorb at low cost business and technology change. This led them to describe their development as “flexible”. A trait demonstrated by the fact that their solutions has been in operational use for many years, but regenerated overtime to evolve through multiple architectures and platforms.
Specifically, Encode found the following across various organizational case studies reviewed and interviews conducted:

- **Indian Offshore Development Centre – Frequent Flyer Application**
  - Compared with hard-written COBOL, CA Gen takes only 40% of the time to develop any application.
  - Compared with Java IDEs, CA Gen takes just 60% of the time.

- **Malaysian Government Agency – Tax Self-Assessment System**
  - Cut development time in half using CA Gen
  - 60% reduction in resources required to maintain the resulting application

- **Australian Financial Services Company – Maintenance on Existing Systems**
  - Conduct ongoing enhancements to 11 major systems with only 50% the staff required for native 3GL development

- **Australian Government Agency – Transport Registration System**
  - Delivered 35% more functionality using CA Gen in the same timeframe as with native 3GL development.

- **North American Scientific Agency – Hazardous Materials Exposure Monitoring**
  - Achieved migration of 1000 transactions from IBM zOS and DB2 to Microsoft Windows and Oracle in two (2) months compared with the initial hand-coding estimate of two (2) years

- **South African Systems Integrator – Pension Funds Management System**
  - Evolved the system three times in 13 years with less than 10 developers from mainframe to client-server to web.
  - Mainframe to Windows transition completed with five (5) developers in six (6) months. Client-server to web transition achieved in 3 months.
  - Modified system to meet needs of UK market in 18 months, compared to 3-5 years for manual-coding effort.

- **North American Systems Integrator – Insurance Investment Portfolio Management System**
  - Since its first delivery 20 years ago the product has remained supported on various platforms by generating multiple versions of code from a single model.
  - Modernisation using CA Gen from client-server to web was 1/3 the cost of a recently proposed equivalent migration for another similarly sized system within the company.
  - Onshore CA Gen developers take only 16 hours to complete what takes offshore .NET developers 40 hours.

- **North American Freight Company – Evolving Portfolio of 150+ Applications**
  - Utilised CA Gen since 1994 with 5 developers, now 200 developers targeting multiple platforms (Linux, Windows, zOS) using various architectures to support 150+ applications.
  - Convert non-IT subject matter experts into CA Gen developers with only two (2) weeks training.
The emphasis by clients on the resilience that technology-independence created cannot be understated. In all cases the longevity and continued evolution and growth of these so called “legacy” applications rather than decline and decay typical of solutions greater than 10 year old was viewed as a competitive advantage by CA Gen customers.

This ability for CA Gen users to easily absorb wave after wave of technology change is evident when the supported operating systems and languages ("target platforms") of the CA Gen product are presented in a timeline as follows (Figure 6):

This and the fact that over 90% of CA Gen clients are licenced for multiple target platforms adds additional weight to the flexibility of the model-driven approach.
Section 3:

Conclusion

Whether driven by internal business needs or external technology change, cost effective transformation of applications requires that organizations first understanding the levels of productivity that can be expected from different type of modernization techniques involving software development.

For those organizations that wish to maintain the competitive advantage provided by the intellectual property of internally developed and maintained applications, utilizing model-driven development will achieve higher developer productivity, provide predictable outcomes, deliver higher quality solutions and result in more technology resilient applications.

Indeed, for the 25% of organizations who have already made the shift or have access to a model-driven development the productivity savings can amount to a reducing in projects costs of over 30% when compared to a native 3GL environment. In addition, remaining in this type of environment presents ongoing benefits including significant savings when generational change inevitably occurs within the technology landscape.

About this Whitepaper

Encode Services was commissioned by CA Technologies to undertake an independent assessment designed to aid senior IT executives of large organisations, both public and private sector, to understand the productivity, predictability and potential cost implications of manual re-coding using native third generation languages (3GL) versus model-driven development (MDD) using tool such as CA Gen. The results of this assessment are intended to provide a starting point for new and existing CA Gen customers to conduct their own high level comparisons and build better business cases for their continued investment in MDD and the CA Gen product suite.

Encode's investigation involved primary data collection, analysis of various secondary research sources, and interviews with senior IT executives—including CA Technologies. Encode would like to acknowledge and thank the organisations who participated in the research.
About Encode

Encode offers international experience in the research and development of information technology (IT) strategy both in the public and private sectors and offer world-recognized expertise across a number of key areas of IT including as-a-service (cloud) computing, enterprise architecture (including service-orientation and information management), the open source market, enterprise applications and development, business intelligence, along with ICT management and governance practices such as ICT planning, strategic sourcing and portfolio management.

Encode’s Managing Director, Sam Higgins, has over 18 years of both tactical and strategic experience in the application of IT to achieve business outcomes from large complex organizations.

Through his previous roles as a leading IT industry analyst with Forrester Research, Research Director for Longhaus and professional engagements as a program consultant and architect, Sam has extensive knowledge of the private sector, state and federal governments which assist him to address the unique IT challenges to be found in diverse industries such as transport, financial services, and education.

Sam’s knowledge of service-oriented architecture and associated business models is widely recognized, and he was a contributing author on the Paul Allen book *Service-orientation: Winning Strategies and Best Practices*, released in 2006 by Cambridge University Press.

Sam is a regular and respected facilitator, panellist and speaker at both local and international conferences and industry events on topics such as cloud computing, ICT service management, ICT industry economics, and application lifecycle management. He is widely published in industry journals and is regularly sought by the ICT industry press for his expert opinion. He has been regularly quoted in the Australian Financial Review, The Australian, ZDNet, ItNews, CRN, ARN, Computerworld and MIS Magazine.

Sam holds a Bachelor’s Degree in business from the Charles Darwin University, where he majored in information systems and minored in marketing.
About Total Metrics

Total Metrics was established in Melbourne Australia in 1994. Their worldwide consultants provide local and international clients with measurement consulting, training and tools to assess the productivity and quality of their software processes and products and to identify areas for improvement.

Total Metrics is a significant contributor to international measurement standards and has represented Australia on the International COSMIC Core Committee, the ISO/IEC SC7 FSM standards committees and the IFPUG Counting Practices Committee. Total Metrics represents Standards Australia IT15 on the ISO/IEC standards 14143 for Functional Size Measurement. Pam Morris the Chief Executive Officer (CEO) is on the executive of the Australian Software Metrics Association and is the former Vice President of the International Software Benchmarking Standards Group Committee (ISBSG).

The firm has earned a strong international reputation for delivering quality products and services in Asia Pacific, USA, UK, China, Europe, India and Japan. As a result, Total Metrics consultants are internationally recognized metrics experts and are regularly invited to speak at International Metrics conferences in the USA, Japan, UK, Korea, India, Europe, Spain, China, Spain, Germany, Netherlands, Italy and South Africa and participate in international research projects in the field of software metrics. They are all tertiary qualified and have at least 15 years experience specifically in the field of software metrics and up to 25 years in IT.
About the ISBG

ISBSG had its origins in the work performed by the Australian Software Metrics Association (ASMA) in software benchmarking. In 1990, a Special Interest Group in ASMA met to develop a practical industry standard for quantifying the output from software development projects. The measure chosen was project delivery rate (“PDR”), defined as the effort (in person-hours) required to deliver one function point of software to the end user. This is analogous to a standard used in the vehicle industry; namely the number of hours required to build one vehicle.

This definition led to the establishment of a repository of data on Australian projects. The first release, in December 1992, contained 24 projects from 7 organizations. Following that release, the definition of project delivery rate was revised and the data collection package reviewed and expanded. Four further releases of the data were made in the next two years. Particular emphasis was placed on including small organizations and new technologies.

The success of this initiative created considerable international interest. In June 1994, the software metrics associations of New Zealand (SMANZ), the United Kingdom (UFPUG), and the United States (IFPUG), together with ASMA, formed ISBSG. The ASMA model was used as the basis for a de facto international standard. Other metrics associations then indicated their willingness to become involved, including CIM (Canada), FFPUG (France), DASMA (German-speaking Europe), GUFPI (Italy), and NEFPU (the Netherlands). Through ISBSG, the various associations planned to collect and share data to facilitate international benchmarking.

The first release of the newly formed ISBSG Repository was in February 1995. It contained 155 projects from three countries (Australia, New Zealand, and the United Kingdom). This, the eleventh release, contains data collected from over 5000 projects from 29 countries.

The ISBSG repository is contributed to on a voluntary, non-fee basis. The project data is selected and contributed to by the various IT organizations around the world. The ISBSG group validates the data, but has no control over which projects are submitted. Therefore by the nature of the contribution process, the projects within the database tend to be ‘best in class’ rather than ‘average’, i.e. they are not truly representative of what is ‘typically’ produced in Industry. Median and percentile values should be regarded as ‘best in class’ values, rather than an industry broad-spectrum median/percentile.
References

i By recognized language we consider those formally published either commercially, academically or re-used within the open source community. See: http://en.wikipedia.org/wiki/List_of_programming_languages

ii Refer to the TIOBE Software Index available at: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

iii See https://www.mckinseyquarterly.com/Business_Technology/BT_Strategy/Overhauling_banks_IT_systems_25547?gp=1

iv See http://www.abs.gov.au/Ausstats/abs@.nsf/0/A0E01DFDEF87524ACA2569A50026C4A5?opendocument

v CA Technologies outlined the key drivers for application modernization in the January 2010 whitepaper entitled An Automated Approach to Legacy Modernization

vi Ut supra

vii See http://www.omg.org/mda/


ix Model-driven projects within the ISBSG Repository are those which explicitly indicated the use of a “Computer-Aided Software Engineering (CASE) Tool or Environment” as part of the development lifecycle

x See http://www.ifpug.org/

xi However since the PDR is the inverse of productivity, when discussing trends in productivity we need to appreciate that the trend in the PDR will be in the opposite direction. When discussing variances in Productivity they correspond directly to the variances in PDRs

xii Information on linear regression is available online. See: http://en.wikipedia.org/wiki/Linear_regression

xiii Specifically, P10 representing the lowest 10% PDRs i.e. the highest 10% productivity, P25 the highest Quartile productivity, P50 as the mean productivity and P75 as the lowest Quartile productivity and P90 as the lowest 10% productivity

xiv Available via subscription to ISBSG. See: http://www.isbsg.org/isbsgnew.nsf/GBL-D&%E%20Subscriber%20Reports?opendocument


xvi Analysis of CA Technologies licencing data showed that 90% of CA Gen customers are licences for at least two target deployment platforms and 80% of customers are licenced for 3 or more platforms
Agility Made Possible: The CA Technologies Advantage
CA Technologies (NASDAQ: CA) provides IT management solutions that help customers manage and secure complex IT environments to support agile business services. Organizations leverage CA Technologies software and SaaS solutions to accelerate innovation, transform infrastructure and secure data and identities, from the data center to the cloud. CA Technologies is committed to ensuring our customers achieve their desired outcomes and expected business value through the use of our technology. To learn more about our customer success programs, visit ca.com/customer-success. For more information about CA Technologies go to ca.com.