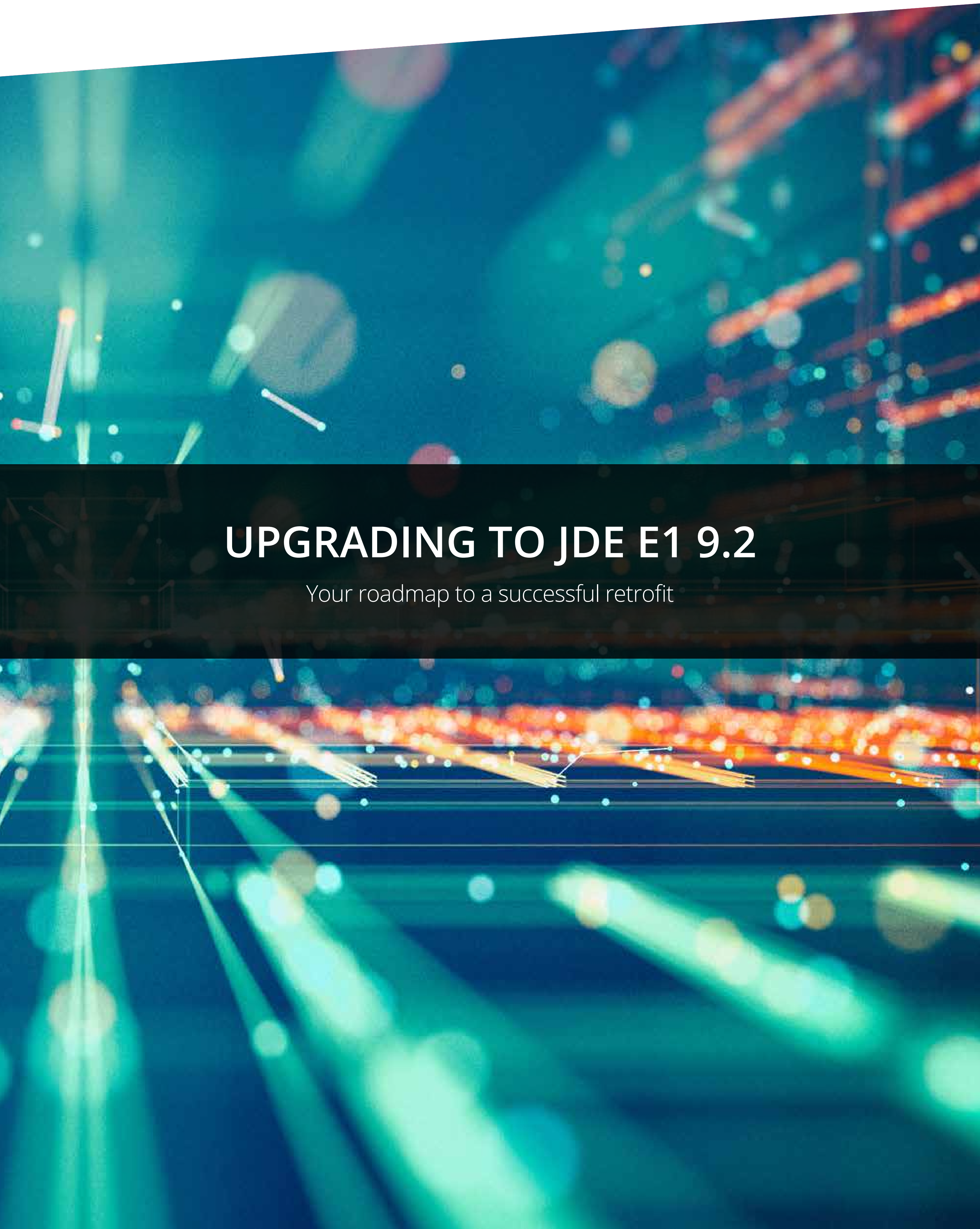
dws
innovation in action



UPGRADING TO JDE E1 9.2

Your roadmap to a successful retrofit

EXECUTIVE SUMMARY

JD Edwards EnterpriseOne (JDE E1) Release 9.2 will be on Extended Support until at least December 2031 and Oracle currently has no plans to release a 9.3 version of the software.

From Release 9.2, with significantly enhanced software packaging and delivery processes, Oracle adopted a continuous delivery model. Going forward, customers will no longer need to suffer the cost and disruption of a major upgrade to take advantage of enhancements.

Once a customer has made the move to 9.2, they should be able to run code-current change event projects more frequently and more easily. Each code-current change event project will require less time and effort, will deliver value more quickly, and will put less strain on the business.

It is this ability to stay code-current that will allow an organisation to make the most of the latest software developments and help them compete effectively in today's digital economy.

Our presumption, as we set out our roadmap for successfully retrofitting during an upgrade, is that companies will find real business value in the enhancements that Oracle is making to JDE E1, whether that is as a consequence of something in a new tools release or delivered as new functionality. The concept may seem hard to digest in terms of more testing, training and disruption to the business, but the benefits will bring enhanced organisational efficiency and greater value in our ERP systems as we move forward to a code-current future.

A prerequisite to getting the most out of any continued investment in JDE E1 in the years to come is to upgrade to 9.2. Our advice is to decommission as many modifications as you can during your upgrade. A reduced modified footprint means less time and resources spent on the technical retrofit elements of your upgrade. Having the right modified footprint for your business at 9.2 will also make future code-current change event projects more straightforward.

It is also important to consider how the technical retrofit work will be performed and delivered. The success of any wider upgrade project can be seriously undermined if the technical retrofitting is not done properly, with zero defects according to an agreed plan. Companies should consider mitigating risk by outsourcing the retrofitting – it can be done for a fixed-price, with a fixed schedule of deliveries.

PLAN YOUR RETROFIT

Upgrades and upgrade plans come in all shapes and sizes. Some application upgrades are done whilst moving from on-premises to the cloud. Some include a tools release upgrade, some are technical as-is upgrades, some are more transformational.

Whatever the reason, when planning your upgrade project there are some points that need to be considered:

- Reducing the modified footprint
- Understanding and challenging every modification
- Recognizing that all modifications are not equal
- Calculating the development effort
- Considering resources and elapsed time

Because retrofitting is on the critical path of many upgrade projects it is imperative that it is a topic you approach and plan for with discipline and consistency.

Reduce your modified footprint

It is common for users of JDE E1 to customize their implementation of the software, making modifications to the standard code and tailoring functionality to better suit their specific business processes.

Over time, these custom objects can become a support issue; either because they become further adapted as business needs evolve, or because of changes made by Oracle to the base code.

In addition, many custom objects become obsolete as enhancements made by Oracle remove the need for the customization or because the custom objects have been superseded.

Each JDE E1 upgrade, whether it is a major version upgrade or a minor code-current change event project, represents an opportunity to review which customizations or modifications are still required.

The ability to identify unused modifications will reduce the effort required for the upgrade. Time will not be wasted uplifting unwanted code and future support and maintenance efforts will be reduced.

Understand and challenge your modifications

A clear understanding of the existence, and extent, of any modification is a core component of any upgrade project. If you do not have an accurate view of your modified footprint, that will almost certainly have wide-ranging implications for the broader upgrade and for the retrofitting.

As you plan your upgrade you need to consider how the software supports the business – who is going to be impacted, and how will jobs be changed? You also have to consider the programming that will be required – what program objects need to be modified or introduced at the Release 9.2 level?

Business process reviews and the assessment of any new functionality, particularly as it relates to your modified footprint, are essential. This can be done in-house or with the help of a systems integrator. At the same time, when objects do need retrofitting consideration should be given to the effort (i.e. to the cost/benefit).

Existing Oracle tools may under or overstate the number of objects in the 'affected' objects list. They may simply list modified objects without any concern given to your custom objects, or to your copies-of-standard objects.

It is also likely that they will not consider the effect of any "net change" coming from Oracle's new-release code and how those code changes will impact your existing modifications.

We cannot overstate the importance of clarity when it comes to understanding the extent of your modifications. The more extensive they are, the longer it will take to uplift the code.

As an alternative, you could rely upon your technical documentation. However, the effectiveness of this is going to be determined by how accurate, up-to-date and comprehensive your documentation is. Even the most diligent of IT operations can struggle to maintain a 100% accurate record of all their system modifications.

Failing to identify, analyze and appropriately retrofit modified code could cause system failure, impact usability, or in extreme cases render Release 9.2 inoperable.

Recognize that all modifications are not equal

Modifications can take several forms. You may have created new custom code, modified standard code or taken copies of the standard code and modified those.

Each type of modification brings its own challenges, with different levels of complexity. Copies, for example, tend to be the most complex and pure custom objects the least. (Note: pure custom objects are those developed from scratch with no direct impact on other objects in the system.)

It is important to know which of your enhancements work independently versus those that interact with, and make calls to, standard JDE objects.

As an example, you may have two Universal Batch Engines (UBEs) the same size. One UBE may work in isolation and is therefore standalone, whereas the other makes several calls to JDE objects. In this second case, you need to understand by how much this UBE will be affected by the changes made to the standard JDE objects called by this UBE.

Another question you need to ask is, are you able to determine if the standard JDE objects you call have changed or not? Are any modifications you made to standard objects affected? In a worst-case scenario, Oracle may have removed this object completely. So how will that affect you?

Version Overrides give you the ability to modify a version's Event Rule (ER) logic and layout at the section level. The JD Edwards Object Analyzer can help you identify version overrides, but they are sometimes overlooked.

Whilst some organizations do not allow version overrides as a policy, some get shipped out of the box by Oracle and power users may make copies of these, which in effect spawn new modifications (version overrides). These additional spawned modified versions may exist discreetly in production.

"It is important to know which of your enhancements work independently versus those that interact with, and make calls to, standard JDE objects."

Calculate the development effort

Each modified object will require development time and attention during the upgrade. With a definitive list of objects, a number of options are available as you work to calculate the retrofit development effort.

Two common approaches are often used:

The one-third rule
An early theory suggested that whatever the effort required to develop the modified footprint originally, about one third of this effort will be needed to upgrade it.

But how many people understand the true effort and time originally involved? And what happens when an object has undergone subsequent changes over an extended period of time?

Analysis by DWS (based on hundreds of upgrade projects) reveals the one-third rule to be grossly inaccurate.

The sample and extrapolate method

A method where the vendor or customer interrogates an example of a small, medium and large modified object; before extrapolating the results across the whole system. This results in an estimate based on how many modifications fall into each category.

Anecdotally, DWS believes this to be the most common method of retrofit estimating. This method however has also proven to be inaccurate, as it relies on a lot of assumptions about the consistency and the complexity of the modified code.

A further problem with both these approaches, beyond the inaccuracy, is that they do not provide you with the detail to help you get from an estimate to a good, considered and well-resourced plan.

The effort you ultimately plan for needs to reflect each and every object. Objects can be classified based on characteristics that inform any effort estimate.

When it comes to estimating, a particularly challenging class of objects is the copies of standard JDE objects. For these you need to consider and determine the best upgrade approach.

The best upgrade approach

When you uplift a copy-of-standard, you need to determine whether it is better to:

1. Keep your modified copy-of-standard object and test it thoroughly to ensure that it continues to function correctly.

2. Determine the Oracle-sourced enhancements to the based-on object in the to-release and port these to your existing copy.

3. Re-copy and re-apply your modifications to a 'fresh' copy of the to-release's based-on object.

Although there is no absolute right or wrong method, each copy-of-standard should be reviewed on its merits, considering aspects such as the extent and complexity of your customizations versus Oracle's own changes in the to-release.

For example, a std-JDE object may only have 2 lines of code added to it in the new release. However, the custom modifications applied may span a hundred lines or more. The ability to see this enables the best approach in retrofitting an object, saving time and effort.

When considering how to uplift those modifications that are still required, do you have access to all the original developers? This may be relevant if you have particularly complex modifications, as a lack of accurate documentation may cause problems.

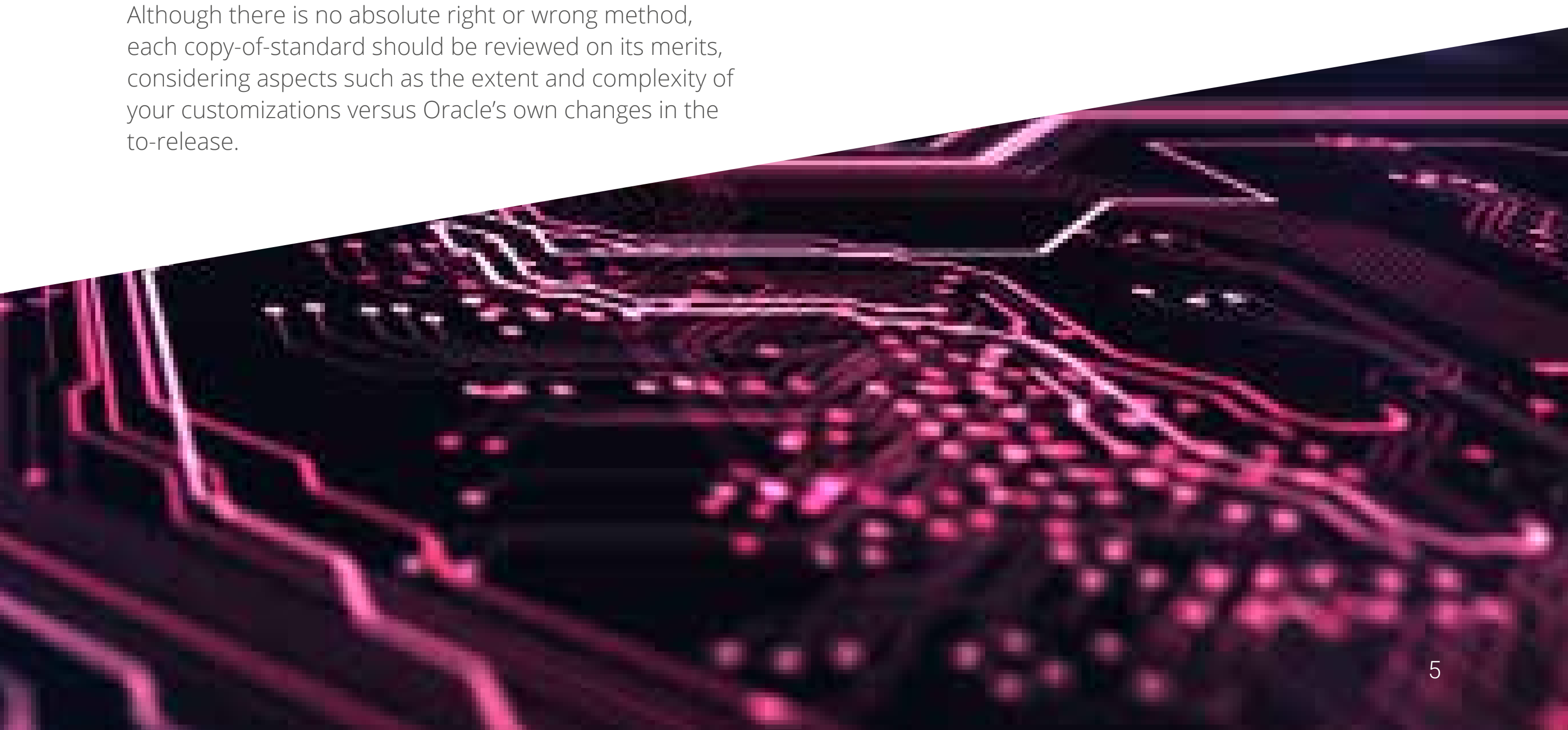
Calculating the development effort for each object will give you results from which you can best plan, and that will allow you to mitigate the considerable risks associated with retrofit cost overruns and delays. The problem is that it is difficult to calculate object level efforts without a proven approach and tools. The DWS Dimension Analyze service does this and is described later.

Consider resources and elapsed time

How long will it take to retrofit?

The answer to this question is a function of the context in which the question is being asked and is dependent on the number and skills of the resources you choose employ.

Your retrofit plan will form part of a wider upgrade plan. The retrofit itself might be staffed by your own people, contractors or a trusted systems integrator. You might take full responsibility or outsource the responsibility to someone else. You might choose to deliver retrofitted objects slowly over time in concert with other project activities, or to deliver all the retrofitted objects as a single package. There is no right or wrong. What is certain, however, is that getting the best outcome will reply on having the right detail with which to plan, and on appropriately employing the right people at the right time.



DWS DIMENSION ANALYZE

Dimension Analyze is a software-led service that has been successfully deployed by hundreds of JD Edwards EnterpriseOne customers; enabling them to audit and scope the development effort required during an upgrade to an unprecedented level of accuracy.

The service provides a forensic level of analysis of every object, every line of code and every property setting, down to pixel movement level of detail.

Dimension Analyze identifies the from/to base net change for every modified object and identifies the net change type and severity of impact against every modified object. It also identifies all modified/custom UBE (Batch) objects that are no longer in use, so do not need to be upgraded.

With this detailed information available, it is possible to not only provide answers to the key questions raised in this paper, but also to estimate the upgrade effort required for each modified object down to an hour/minute level of detail. The net result can lead to some significant savings in both time and resource.

Savings are dependent upon the degree of customization and the type of modifications in place. However, for some clients, we have been able to reduce their modified footprint by over 75%.

Even moderately modified instances can realise significant savings. "When we upgraded from 9.1 to 9.2, we were able to reduce our modified footprint from 920 raw objects to just 186. With 80% fewer objects to deal with we saved over 170 development days."

"We ran a heavily modified instance of E1, with over 6200 modified or custom objects. After using Dimension Analyze, we were able to reduce that number by 60% to just 2400.**"**

RETROFIT FOR SUCCESS

With a definitive list of modified objects to be uplifted and a handle on the development effort that will be required, the plan for your upgrade and the retrofitting that will be performed as part of that plan should really be starting to take shape.

The best 9.2 upgrade project plan will be unique to your environment. It will reflect the fact that this will be your last major upgrade and that after this upgrade you should be implementing a code-current strategy and running more frequent code-current change event projects.

Against this backdrop, it is important to consider the skills and experience of the resources available to you, both within your organisation and outside. The technical project management, development skills and experience needed for retrofitting are unique. It is because of this that you need to decide whether you want to carry out the uplift yourself or contract out some, or all, of the development work. After all, your own development resource may be best employed supporting the ongoing day-to-day operation of JDE E1.

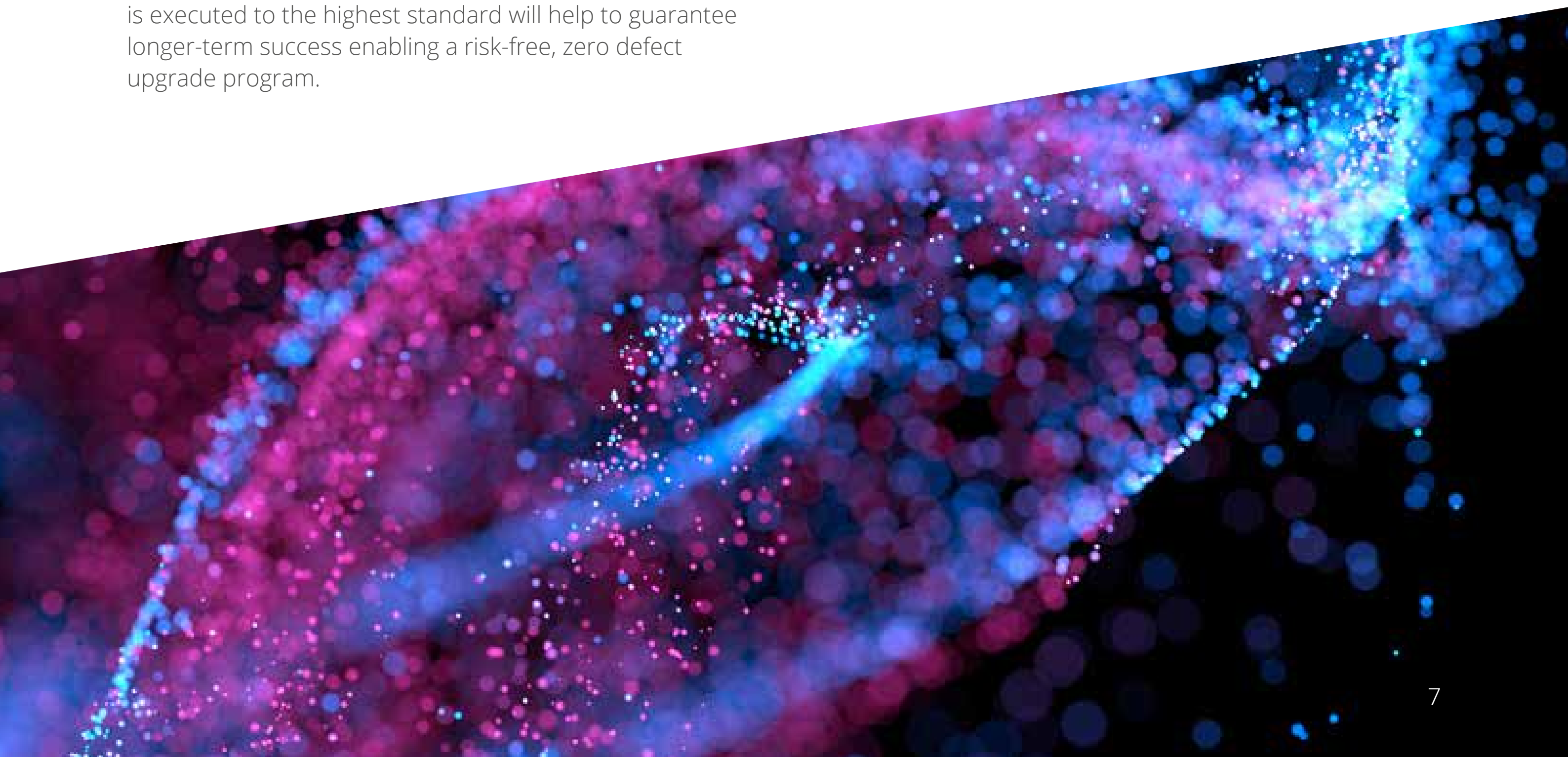
Whatever you choose, it is important to remember that your plan will only be as good as its execution. Your successful upgrade will entrench (if not establish) JDE E1 as the enterprise application that will support a culture of digital innovation. For this to happen you need to ensure you deliver on the wider plan. Within the wider plan you need to retrofit only those objects that need to be retrofitted. And you need to leave each and every object in a state that will allow it to be efficiently and effectively maintained as you retrofit during future code-current change event projects.

Ensuring that the retrofit development is well managed and is executed to the highest standard will help to guarantee longer-term success enabling a risk-free, zero defect upgrade program.

Develop intelligently

There are many issues developers need to consider when executing an E1 upgrade, including:

- Identifying modifications to standard objects
- Identifying modifications to copies of standard objects
- Obsolete objects
- Changes to standard hooks
- Custom indexes on standard tables
- Unicode and best practice changes in C business functions
- Obsolete APIs and custom client code
- Client only functions
- Table I/O mismatches
- Version overrides



Previously, we discussed how you might identify all modifications to standard JDE objects and to copies of standard objects. This information needs to be available and readily accessible to any developers carrying out the upgrade.

Obsolete objects – If you miss a custom object that references a standard JDE object that has become obsolete (or no longer exists in a future release) it will cause issues downstream, during user acceptance testing.

Changes to the standard-hooks – There may be a change in the parameters passed into a function call, which ultimately affects the results of using this function in custom code.

Knowing about these changes, and the potential impact during the retrofitting of an object, reduces the likelihood of issues during user acceptance testing.

Custom indexes on standard tables – A custom index may have been added to the standard table. As an example, we will call it index 5. In a future release, Oracle might add a new index to the table, also called index 5, which causes a conflict.

How do you check for these pitfalls so you can verify and retrofit the use of the custom index correctly and, in turn, ensure the consistent behaviour of the custom code and index?

Unicode and best practice changes in C business functions – If upgrading from a non-Unicode release, ensuring your C Business Functions meet Unicode standards is only partially covered by Oracle's tool set. Some of the changes will need to be made manually by a programmer.

Obsolete APIs and custom client code – Like the calling of obsolete objects and business functions, APIs called from custom or modified business functions may have been changed by Oracle or replaced by a new API.

Understanding what has changed and how it should be replaced is essential. For a successful upgrade you need to then know where the changed code is used in custom modifications.

Client only functions – If upgrading from a Windows-based release to a web-based release, any client-only business functions you may have developed may not automatically function correctly and often require reengineering / re-design. Knowing where these function types exist before the upgrade starts is critical to prevent overrunning.

Table I/O mismatches – An upgrade presents a good opportunity to review the existing code/logic that may be causing issues in the current release. Issues such as mismatches between the indexes used by a Select and its following Fetch, for example, are issues that can cause testing problems if upgraded "as is".

Manage with discipline and rigor

A lot goes into managing an upgrade and organizations upgrade for many different reasons. With Release 9.2, a major justification for any upgrade will be to get current, so you can stay code-current.

Whether this is your justification or not, every 9.2 upgrade, and all the change event projects that will follow your upgrade, will involve some degree of business impact. It is the change to the business that should be the focus of the project. Management of the project should aim to minimize business impact and mitigate risk.

Behind the business change, there may be a lot of technical change (e.g. a cloud and/or infrastructure changes, the retrofit development effort, etc.). The more technical aspects of a change event project are excellent candidates for outsourcing.

Outsourcing is a great way of mitigating risk, as you employ an organization that is expert in a specific area, allowing you to concentrate on the business change and value-adding aspects of your project.

Whether you choose to contract a specialist service provider or not, some of the challenges that relate to the project management of a technical sub-project that is focused on retrofitting during an upgrade include:

- Bundling
- Managing Resources
- Status Tracking
- Monitoring Progress

The Retrofit Project Manager needs to consider the overall project plan and timeline, requirements of the critical business processes, object dependencies and how to bundle objects together into smaller projects to meet defined deliverables and milestones.

Being able to do this accurately will enable you to plan resources for testing and deliver the project in manageable bundles (an agile approach) rather than waiting till the end of the development activity before performing all the testing (known as the waterfall approach).

It would clearly be helpful if the Retrofit Project Manager can produce a detailed project plan; one that provides visibility and accountability at an object level and enables the effective management of resources and tracking of the progress of the sub-project.

Assure quality

An important question to ask as you prepare to transition from planning to executing is: how do your Quality Assurance (QA) staff ensure that the code delivered by the developer has addressed all the challenges we've identified?

What tools exist that can help your QA staff provide your end users confidence that the upgrade has gone smoothly and that they will experience few problems?

The key issues for the Quality Assurance Officer are to minimize defects and maximize quality. Part of this is ensuring that as much as possible is done correctly first time. Where any reworking is necessary, the QA function needs to track this activity and ensure the correct version is included in the uplifted system.

You need to be sure you have an accurate list of all modified code still in use to make sure every required object is uplifted to the new release, with nothing overlooked.

It would be helpful to have access to a knowledgebase of issues, such as unknown or missing columns and parameters, known obsolete and retired APIs etc. Unfortunately, many employees involved in upgrades have little or no previous experience to call upon.

DWS DIMENSION PROFESSIONAL

One of the biggest challenges associated with a major upgrade project is the technical uplift, or retrofit, of custom items. DWS Dimension Professional is a suite of tools that provides a management framework for the modifications uplift component of any E1 upgrade project.

A Dimension Professional upgrade retrofitting project leverages the output from Dimension Analyze, resulting in the lowest defect levels in the industry. Furthermore, by utilizing the best upgrade approach, DWS are able to offer a fixed price and fixed timescale for the technical retrofit project.

Speaking with JDE E1 users contemplating an upgrade, there are some frequently asked questions that are worth exploring. First amongst them is the question of confidence in the modification documentation. Can you be sure the records are complete and 100% accurate?

It is unlikely that any organization will be able to readily access the developers and consultants that were involved in the original implementation. So, how do you work out what all the modifications were? Even if you can contact the original dev team, how confident are you that they will remember all the modifications work that was carried out?

How certain are you that nothing has been missed and undocumented modified objects aren't hidden away somewhere? Continuing on the theme of confidence, can you be sure that the standard JDE Modifications Report shows where all the modifications exist in your system?

Using Dimension Professional, DWS can work alongside your in-house team to provide additional development expertise where required. Alternatively, we can act as a dedicated upgrade resource, freeing your internal teams to focus on maintaining your live environment.

Using Dimension Professional DWS can work alongside your in-house team to provide additional development expertise where required. Alternatively, we can act as a dedicated upgrade resource, freeing your internal teams to focus on maintaining your live environment.

With Dimension Professional, upgrade customers can:

- Establish a fixed price and timescale for a technical retrofit project
- Access role-based tools for project management, development and quality assurancee
- View the exact location of all E1 changes
- Automatically scan the system for known E1 upgrade issues
- Accurately determine interdependencies between all E1 objects
- View from and to releases concurrently, resulting in a faster, more accurate upgrade
- View proprietary net change data to fully understand the impact of any upgrade
- Significantly increase developer "right first time" ratio

"DWS were very flexible in their approach to the upgrade; working alongside our developers to deliver the upgraded objects. Every aspect of the project has been managed very efficiently."

USE CASE 1

WILBUR-ELLIS®

Ideas to Grow With®

Wilbur-Ellis employs over 4,000 people in over 240 offices across three continents. It is comprised of three separate operating divisions that share a single worldwide ERP platform.

Wilbur-Ellis runs a highly modified instance of JD Edwards EnterpriseOne and has 1,500 active users.

Compelled to upgrade because its current solution was nearing end-of-support, Wilbur-Ellis wanted to reduce its current modified footprint and accurately estimate the level of effort required to retrofit all in-scope modifications to the 9.2 level.

Wilbur-Ellis opted to use Dimension Analyze™ to forensically analyze both the existing modified footprint and the net changes made by Oracle, down to a pixel level of detail.

The Dimension Analyze audit led to a 39% reduction in the modified footprint, which in turn saved a further 236 development days.

The company then went on to execute an upgrade project, using the Dimension Professional™ methodology, that was delivered ahead of schedule and on-budget.

“The initial investment in Dimension Analyze™ will be returned many times over because the project will be more predictable, with no nasty surprises along the way.”

JENNY CATLETT, DIRECTOR OF APPLICATIONS

USE CASE 2

Printpack

Converting Vision into Value™

Printpack employs 3,900 people across 19 sites in the US, Mexico and China. It has been using JDE E1 since 2007 to support manufacturing operations, human resources and payroll. By 2012, the JDE general ledger was established as the system of record.

With JDE E1 implemented across all sites, Printpack had accumulated over 8,000 modified objects. Upgrading to 9.2 as part of a wider digital transformation project, Printpack was looking to significantly reduce its modified footprint.

Using the Dimension Analyze service, Printpack was able to identify over 3,0000 modified objects that would not require retrofitting, reducing Printpack’s modified footprint by 42% and saving over 400 days of development effort.

DWS went on to utilize the Dimension Professional toolkit to complete the development work on budget, two weeks ahead of schedule and with zero defects.

“We had such a great experience with DWS and the Dimension Analyze tool. By reducing our modified footprint, we saved over 400 days of development effort.”

CHRISTEL CRAIG, MANAGER, BUSINESS SYSTEMS

About DWS

DWS is a leading provider of Oracle JD Edwards EnterpriseOne software services and products.

Since 1998, we have been providing development and technical services to organizations looking to customize, integrate, extend, upgrade or support implementations of EnterpriseOne. We also sell EnterpriseOne testing products that leverage our deep domain expertise and help customers run smaller, faster and smarter projects.

DWS serves a global client base using proven methodologies and proprietary DWS Dimension™ tools. Our best-practice approach and eye for detail help us deliver products and services that save time and money and continually drive down your TCO for JD Edwards.

For further information please visit our website, or contact us:

UK: +44 (0) 1494 896 600 US: +1 888 769 3248 ANZ: +64 (0) 9427 9956
sales@dws-global.com **www.dws-global.com**

ORACLE Gold Partner Cloud Standard