Contoso  
Application Life Cycle Management Review

*Prepared for*

**Contoso**

*Prepared by*

**Joe Bloggs**

**Senior Consultant**

**ALM Consulting Services**

[1 Executive Summary 3](#_Toc189722892)

[1.1 Key Areas for Improvement 3](#_Toc189722893)

[1.1.1 Current State 3](#_Toc189722894)

[1.1.2 Future State 4](#_Toc189722895)

[1.2 Current Best Practices 4](#_Toc189722896)

[2 Key Areas for Improvement 7](#_Toc189722897)

[3 Roadmap 8](#_Toc189722898)

[4 Detailed Findings 12](#_Toc189722899)

1. Executive Summary

At the request of Contoso, ALM Consulting Services conducted an Application Lifecycle Management (ALM) assessment with the following objectives:

* Create a baseline measurement of the current development capability
* Surface existing best practices
* Uncover opportunities for improvement
* Identify the most impactful areas to the business
* Generate a preliminary roadmap to implement improvements

The ALM assessment uses the Application Platform Optimization (APO) Model to provide a framework that allows Contoso to achieve a vision and sustainable approach to prioritizing IT investments that fuel business growth.  The assessment focuses on assessing current the full application development lifecycle including:

* Project Management
* Production Support Management
* Testing
* Development Processes

Technology and People/Knowledge requirements are then identified to support the process.

* 1. Key Areas for Improvement
     1. Current State

During our onsite interviews we uncovered the following practices that should be considered critical to improving the development capability in relation to their impact on the business.  These practices were:

* Code Management
* Testing
* Operations Management
* Requirements documentation

The current practice of creating a new code base for WunderApp each time a new partner is on-boarded by Contoso creates a management issue spanning development support, testing, and operations. Developers are required to maintain and manage dozens of separate distinct code bases. A bug found in one codebase likely occurs in the others. Fixing bugs of this nature across all partner code bases is clearly a time consuming effort. Validating the bug fix requires testing and regression testing each partner application. Often regression testing is not done to the fullest extent it could be as it is a manual process today. Lastly, operations management calls for maintenance of multiple web sites and databases. While deployment scripts are provided they are typically run manually for each partner application and database.

Many of these pain points are expected to diminish with the release of WunderApp 2.0 as it supports a single code base. Given today’s development model, this is a tremendous improvement. However, business requirements still call for a distinct database to be maintained for each partner. Both operational management and testing will need to take this into account.

A few key finding bring attention to the existing test process. Today’s testing environment is sufficient to support functional testing; however, the lack of a staging environment precludes the possibility for stress testing and load testing. Acme Group has plans to introduce this environment. Once a staging environment is available, project planning will need to take stabilization and load testing into account.

Requirements gathering and documentation at Contoso does follow a process which includes a review of the requirements involving the development team, project management, and the testing team. However, it has been communicated that not all requirements are communicated in the manner. Often, email is used to send requirements directly to the development team without going through a formal review process. Further, requirements, even after a review process, are lacking wireframes which leaves room for misinterpretation of the original intention.

* + 1. Future State

Acme Group is in the process of integrating with Contoso and aligning their processes. This involves managing Contoso as a service area within Acme Group. Service areas tend to operate as business units with well defined roles and processes with standard support from operational service areas such as the DBA group. The processes and practices applied by a particular service area are driven by the business needs.

The eventual WunderApp 2.0 roll out it will be supported with a periodic release process. The particular timeframe on the release process is to be determined. It may be every two months, quarterly, or every six months. The current Agile-like approach followed by Contoso today may not fully support this cycle. For example, a three month development cycle will require a means of tracking tasks over a longer duration than is currently needed to support WunderApp and ChannelApp. The longer the release process, the greater need to create and track a formal project plan.

Contoso will also need change the way business requirements are gathered and defined to prepare for the maintenance phase of WunderApp 2.0. For example, changes and additions to the user interface should include wireframes. As Acme Group continues to integrate with Contoso it is anticipated that the process for gathering business requirements will improve.

* 1. Current Best Practices

We also surfaced the following Best Practices that are being used by teams at Contoso today. The practices are categorized by practice areas.

Project Management

* An internally developed bug tracking application is currently used to track and report bugs.
* In some cases there is traceability back to the requirements document. There is a formal document referencing back to the requirements document. It is available to the project team.
* In some cases an informal test plan is tracked, updated and followed based on the complexity of the feature being implemented
* Project documents such as requirements are stored in a team site, available for access by the project team, and version controlled.
* A vision scope document is being created for WunderApp 2.0.
* A top down and a bottom up approach is being used to generate estimates for WunderApp 2.0.
* Roles are paired between OutSource Inc and Contoso personnel.
* Critical success factors are defined at completion of envisioning.
* Status reports include risks, tasks complete,
* KPIs have been defined and tracked for prior projects and will be tracked for WunderApp 2.0.
* Stakeholders are clearly identified and high impact project decisions are made at an executive level.
* A formal signoff process is in place at Acme Group before a project moves forward.
* Contoso uses a custom application to manage change control including, but not limited to, updates to production environments.
* Contoso follows a formal change control process to promote a build from the test environment to production. Project management, test leads, development leads, etc. sign off on a change control request before a build is promoted to production.
* No updates done directly to production without going through a testing environment.
* Contoso uses an internal bug and issue tracking tool to raise third level support issues to the development team. The particular tool used to track bugs is expected to change, but the process is expected to remain the same.

Development

* Naming conventions are documented and used by the development team.
* WunderApp 2.0 planning includes unit testing for each application tier.
* Standard tools are used to defined and document database schemas.
* Development team will be using tools to automatically generate test data.
* All code is maintained under source control including database and release scripts. This includes any SQL Server jobs and report definitions.
* A library of successful builds is maintained in a shared build folder along with deployment instructions.
* SQL Server application roles are used to only execute stored procedures. Direct table access is not done in WunderApp and ChannelApp.
* The infrastructure architecture for WunderApp and ChannelApp as well as WunderApp 2.0 is understood, documented, and shared with the project teams.

Testing

* The test team has a set of steps followed to perform build validation (i.e. smoke test).
* A testing environment is in place and is sufficient for supporting functional testing.

Customer Support

* Contoso trainers are available to give standardized WunderApp and ChannelApp training when a new program area is initiated.

Operations

* Databases are locked down from developer access.
* SAN storage assists with storage management concerns.
* Application servers are in place to support a passive fail over.
* Production environments can currently be recreated from back up devices.

We recommend that these best practices be codified and implemented as standards.

1. Key Areas for Improvement

Our interviews revealed multiple areas for improvement.  These were rated by impact to the business (High, Medium, Low) across the maturity levels.  These are shown in the Impact Map.

The x-axis defines the maturity level of the service area. The categories are:

* Basic – processes are implemented in an ad-hoc, undocumented and potentially inconsistent manner.
* Standard – a process has been defined and is generally followed. Tools are used in some cases to assist, but may not be integrated and used throughout the organization.
* Advanced – usage of tools to drive the process is in wide use and usage guidelines are documented and understood.
* Dynamic – the organization is bringing new and innovative methodologies to the practice area and may setting industry standards.

The y-axis defines the relative gain that would be obtained from improving the practice.

During the assessment the areas that need the most attention are cited in more detail in section [Current State](#CurrentState). These areas are represented in the upper left cell which denotes practices with a basic level of maturity and a high impact to be gained from improvement. The majority or practice areas fall into a basic level of maturity with a medium impact to be gained from improvement.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **I** **M** **P** **A** **C** **T** | High | * Code Reuse * Quality Metrics * Database Unit Testing * Requirements Management * Business Analysis | * Database Deployment  |  | | --- | |  | | |  | | --- | |  | | |  | | --- | |  | | | Medium | * UI Prototyping * UX Integration * Code Writing * Code Analysis * Code Reviews * Auditing * Build Management * Deployment * Backout Process * Environment Set Up * Database Build Management * Database Test Data Generation * Deployment to Infrastructure  |  | | --- | |  | | * UI Design * End-User Documentation * Source Control * Operations * Customer Support * Infrastructure Architecture * Build Promotion * Database Design * Database Change Management  |  | | --- | |  | | |  | | --- | |  | |  | | Low | * Source Control Standards * Development Standards * Monitoring  |  | | --- | |  | | * Project Initiation * Build Validation * Database Source Control * Database Code Review * Task Assignment * Risk Management * Stakeholder communication * Database Source Control * Database Code Review |  |  | |  | **Basic** | **Standard** | **Advanced** | **Dynamic** | | **M     A    T     U    R    I     T    Y** | | | | | | |

1. Roadmap

Based on our observations and discussions we would recommend the following iterative roadmap be implemented in order to better align the development capability with the business and enable development efforts to drive increased value to Contoso.

Please note that the areas for improvement mentioned in the prior section which are marked as urgent may not be addressed immediately. In some cases, the foundation for improving a particular service area will not be in place in the first or second iteration.

The dates associated with each iteration are estimated based around the progress of WunderApp 2.0 development.

* Iteration 1 – covers the roll out of Microsoft Team Foundation Server 2008
* Iteration 2 – this period of time covers the initial development cycle of WunderApp 2.0.
* Iteration 3 –during this period WunderApp 2.0 development effort is nearing completion. Functional and stress testing takes place. Operations prepares for roll out.
* Iteration 4 – WunderApp 2.0 rolls out to production and enters maintenance mode.

The date ranges for each iteration should generally align to the phases outlined above.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Iteration 1** | | From : 1/28/2008 | | To     : 2/29/2008 | | |  | | --- | | **Iteration 2** | | From : 3/3/2008 | | To     : 8/29/2008 | | |  | | --- | | **Iteration 3** | | From : 9/1/2008 | | To     : 2/27/2009 | | |  | | --- | | **Iteration 4** | | From : 3/2/2009 | | To     : 8/31/2009 | |
| |  | | --- | |  |  |  | | --- | | 1.0 Initiative | | * Code Writing * Quality Metrics * Source Control * Auditing * Database Unit Testing * Source Control Standards | | |  |  |  | | --- | --- | --- | | |  | | --- | | 2.0 Initiative | | * UI Design * UI Prototyping * Infrastructure Architecture | | | |  | | --- | |  |  |  | | --- | | 3.0 Initiative | | * Operations * Deployment * Backout Process * Environment Set Up | | |  |  |  | | --- | --- | --- | | |  | | --- | | 4.0 Initiative | | * End-User Documentation * UX Integration | | |
| |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | 2.1 Initiative | | * Code Reviews * Database Code Reviews * Code Analysis * Code Reuse * Build and Deploy * Source Control Standards | | |  |  |  | | --- | --- | --- | | |  | | --- | | 3.1 Initiative | | * Business Analysis * Requirements Management | | | |  | | --- | |  | |

When embarking on an effort to optimize the development capability we recommend that strong leadership sponsorship is secured, that the overall goals are clearly communicated to all stakeholders, and clear metrics and milestones are established and agreed to.

Our full report details all findings and recommendations.

#### Initiative Details

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Iteration 1   From : 1/28/2008  To : 2/29/2008 | | | | | | | | | |
| **1.0 Initiative** | | | | | | | | | |
|  | | | **Initiative Goals:** | | | Improve code quality | | | |
|  | | | **Initiative Activities:** | | | Deploy Team Foundation Server 2008  Determine and enable initial policies code must meet in order to be checked in  Include unit testing for both application code and database code  Establish and document how to manage source control | | | |
|  | | | **Impacted Practices:** | | | Code Writing Quality Metrics Source Control Auditing Database Unit Testing Source Control Standards | | | |
| Iteration 2   From :   3/3/2008 To : 8/29/2008 | | | | | | | | | |
| **2.0 Initiative** | | | | | | | | | |
|  | | | **Initiative Goals:** | | | Validate user interface design and assess infrastructure architecture required and further integrate Acme Group with Contoso. | | | |
|  | | | **Initiative Activities:** | | | Acme Group user experience designers review functional specifications and wireframes. This will provide a forum for initial feedback, but, more importantly, it will familiarize the Acme Group UI designers with the business.  Additionally, the Acme Group infrastructure team should validate the WunderApp 2.0 production physical architecture. This will provide both an opportunity for a review and give Acme Group a chance to plan for putting WunderApp 2.0 in production. | | | |
|  | | | **Impacted Practices:** | | | UI Design UI Prototyping Infrastructure Architecture | | | |
| **2.1 Initiative** | | | | | | | | | |
|  | | | **Initiative Goals:** | | | Including additional gating checks to improve code quality. Teams will not be collocated during the WunderApp 2.0 development effort. Enforcing automated checks and balances will validate that the teams are developing code that compiles and passes basic unit tests. | | | |
|  | | | **Initiative Activities:** | | | Implement Continuous Integration including automated unit testing of both application and database code. This should result in a daily report listing the test results.  Hold code reviews when a major feature or application area is nearing completion or when other developers need to integrate with it. | | | |
|  | | | **Impacted Practices:** | | | Code Reviews Code Analysis | | | |
| Iteration 3   From :  9/28/2008 To : 2/27/2009 | | | | | | | | | |
| **3.0 Initiative** | | | | | | | | | |
|  | | | **Initiative Goals:** | | | Prepare for production release of WunderApp 2.0 | | | |
|  | | | **Initiative Activities:** | | | Produce deployment packages Create and document back out plan Create staging and production environment Perform stress and performance testing Ensure application error reporting process matches expectations from operations staff. This should be done early in the iteration. | | | |
|  | | | **Impacted Practices:** | | | Operations Deployment Backout Process Environment Set Up | | | |
| **3.1 Initiative** | | | | | | | | | |
|  | | | **Initiative Goals:** | | |  | | | |
|  | | | **Initiative Activities:** | | | Define project methodology which will be followed once WunderApp 2.0 goes into production. Acme Group should validate that Contoso is following best practices for requirements gathering and business analysis. This should include wireframe documents when changes to the user interface are required. | | | |
|  | | | **Impacted Practices:** | | | Business Analysis Requirements Management | | | |
| Iteration 4   From : 3/2/2009   To : 8/31/2009 | | | | | | | | | |
| **4.0 Initiative** | | | | | | | | | |
|  | | | **Initiative Goals:** | | | Enhance customer support.  By this point OutSource Inc will have provided end user and help documentation. User feedback will likely call for updates to this documentation after it has gone into production. | | | |
|  | | | **Initiative Activities:** | | | Update end user documentation.  Include the Acme Group user experience designers in additional updates to the WunderApp 2.0 user interface. | | | |
|  | | | **Initiative Cross References:** | | | The 2.0 Initiative introduced the application to the Acme Group design team. | | | |
|  | | | **Impacted Practices:** | | | End-User Documentation UX Integration | | | |
| 1. Detailed Findings  User Experience (UX) Summary : | | | | | | | |
| **UI Design** | | | | | | | |
|  | **Maturity Observations:** | | Contoso hired designers to create the general look and feel (i.e. buttons and style sheets) of WunderApp and ChannelApp but not the overall user experience.  No formal document defines a standardized design. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Modifications to the WunderApp 2.0 UI should follow predefined guidelines. If a guideline document is not part of the WunderApp 2.0 deliverable Acme Group user design experts should work with Contoso to develop it. | | | | |
|  |  | | | | | |
| **UI Prototyping** | | | | | | | |
|  | **Maturity Observations:** | | Since most of the development for WudnerApp and ChannelApp often requires minor enhancements rather than significant features wire frames and storyboarding have not been historically of much use. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | WunderApp 2.0 design does involve the creation of functional specification including wireframes and storyboarding. A formal document review process involving stakeholders, project management staff, and the testing team is planned and normally enforced within many Acme Group service areas. | | | | |
| **UX Integration** | | | | | | | |
|  | **Maturity Observations:** | | Currently, there is no dedicated user experience representative dedicated to WunderApp and ChannelApp. Contoso program managers typically define the requirements and user experience requested by the program area. Specific user requirements are more often communicated by the program area personnel rather than the partner. This is driven by the need for customer service personnel to provide reports and other BI information to the customer as the customer does not currently had a user interface to provide this information. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Once WunderApp 2.0 is released feature updates will be expected on a quarterly basis. While a full time user experience lead may not be required project support could benefit from having periodic involvement. | | | | |
| **End-User Documentation** | | | | | | | |
|  | **Maturity Observations:** | | There are some end user PowerPoint presentations which give high-level usage information. Coupled with the training this has been largely sufficient for WunderApp and ChannelApp. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | WunderApp 2.0 will have a more complex user interface. An integrated help system will be included with this release. | | | | |
| Requirements Management Summary : | | | | | | | |
| **Business Analysis** | | | | | | | |
|  | **Maturity Observations:** | | Customer requirements are largely captured in Word documents. Generally, UML and other more formal tools have not been used to capture requirements largely due to the narrow scope of the features requested. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | To date the lack of complex requirement for WunderApp and ChannelApp has precluded a need for formal tools. The scope of WunderApp 2.0 calls for specific and formal requirements documentation. Further, once WunderApp 2.0 is delivered a quarterly release process will be implemented which will also call for more formalized requirements on an ongoing basis. | | | | |
| **Requirements Management** | | | | | | | |
|  | **Maturity Observations:** | | The fluid and narrow nature of feature updates for WunderApp and ChannelApp has allowed Contoso to monitor and track requirements by email. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | The scope of WunderApp 2.0 calls for a more formalized approach for tracking and updating requirements. This requirement carries forward once WunderApp 2.0 moves into a quarterly maintenance mode. Contoso business analysts are working with Acme Group business analysts to properly manage and trace requirements. | | | | |
| Code Quality Summary : | | | | | | | |
| **Code Writing** | | | | | | | |
|  | **Maturity Observations:** | | Code is not consistently commented. Standards for writing secure code are generally understood but are not documented. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | By enforcing check-in policies some standards can be enforced with respect to naming conventions, commenting, etc. Periodic code reviews are recommended to ensure developers, including those offshore, are following defined practices. | | | | |
|  | **Impact Benefits:** | | With consistent standards and comments it should be easier for new developers to ramp up on the project and experienced developers to tackle different components of the application. | | | | |
| **Code Analysis** | | | | | | | |
|  | **Maturity Observations:** | | Object oriented programming techniques are now being introduced with WunderApp 2.0. Some limited performance and stress testing has been conducted, but it has largely been around SQL Server Reporting Services rather than WunderApp and ChannelApp. Unit testing is done sporadically and so code coverage of tests is limited. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Implementing check in policies which require unit testing and code coverage will result in an improvement in code quality.  Acme Group plans on introducing both stress testing and performance testing in a staging environment. | | | | |
| **Code Reuse** | | | | | | | |
|  | **Maturity Observations:** | | Frameworks and patterns and practices are not in use today. Today there is no use of the Enterprise Library or Software Factories.  Further, the current development methodology of creating a new codebase per partner results in a difficult code management scenario. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | High | | | | |
|  | **Impact Observations:** | | WunderApp 2.0 will use Enterprise Libraries but the final architecture is to be determined. It is strongly recommended that a service layer is used.  Further, the code base will be consolidated into one. There will no longer be a distinct code base per partner. | | | | |
| **Code Reviews** | | | | | | | |
|  | **Maturity Observations:** | | Code reviews are sporadic. No evidence is produced | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Code quality will improve over time. | | | | |
|  | **Impact Benefits:** | | Performing code reviews will cross-pollinate knowledge of the application to other developers. | | | | |
| **Quality Metrics** | | | | | | | |
|  | **Maturity Observations:** | | Unit testing is sporadically used in current applications. No gating or quality control process in place to ensure check-ins meet minimum standards such as required comments. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | High | | | | |
|  | **Impact Observations:** | | Placing quality metrics around check in process would help to ensure code is commented and follows well defined standards. This becomes increasingly important as some upcoming team members will be off shore. | | | | |
| Project Management Summary : | | | | | | | |
| **Project Initiation** | | | | | | | |
|  | **Maturity Observations:** | | OutSource Inc has a process for managing people in disparate locations. Acme Group is defining roles and responsibilities horizontally. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
|  | **Impact Observations:** | | Acme Group’s project initiation process is rather comprehensive. It’s recommended that Contoso adopt the same approach. | | | | |
| Software Configuration Management Summary : | | | | | | | |
| **Source Control** | | | | | | | |
|  | **Maturity Observations:** | | Policy requires each release to be labeled. Although it is often followed, there may be cases when it is not. A separate build manager is designated for each project. A standard branching process is not defined. Spinning up a new project requires baseline code to be copied and checked in to a new project folder in VSS. All developers currently have admin access to VSS. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | The current approach to branching code for WunderApp and ChannelApp involves cutting, pasting and modifying. WunderApp 2.0 will obviate this practice for managing new partner projects.  A versioning and branching policy for addressing bugs and adding features while in maintenance mode will need to be established. | | | | |
| **Source Control Standards** | | | | | | | |
|  | **Maturity Observations:** | | Source control policies are generally understood, but not documented or always followed consistently. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
| **Auditing** | | | | | | | |
|  | **Maturity Observations:** | | Source control comments are not entered consistently. Build numbers are not currently used. Assemblies are deployed to production without file or assembly versions. All developers currently have admin access to source control. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Enforcing check in policies which require developers to both enter check-in comments will assist in producing a release notes document for the testing team.  Missing build numbers on assemblies has occasionally caused deployment issues. Stamping an assembly or file version will remove any doubt with respect to which binaries have been deployed.   Granting developer’s admin access to VSS can be construed as a SOX violation. | | | | |
| **Build Management** | | | | | | | |
|  | **Maturity Observations:** | | The build process is defined on a project-by-project basis. It is not documented, but generally understood by the build master designated for the given project. Builds are generally done on an as-needed basis and not scheduled or automated. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Implementing continuous integration to manage the build process will assist with finding bugs and broken builds before they are to be released to test. Automated builds would also allow for automated unit testing which would further contribute to improving code quality. | | | | |
| Deployment & Operations Summary : | | | | | | | |
| **Deployment** | | | | | | | |
|  | **Maturity Observations:** | | Within Acme Group the deployment processes vary depending on the service area. More critical service areas tend to have a more formally defined deployment process.  Deployment process for Contoso involves an XCopy style of deployment. Deployment process is manual for deployment servers. Database servers are typically updated using scripts. While the database deployment may be repeatable, the largely manual process of deploying to application servers is not. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Having a repeatable deployment process would reduce the likelihood of a missed step or mismatched file. | | | | |
| **Backout Process** | | | | | | | |
|  | **Maturity Observations:** | | Back out process for deploying WunderApp and ChannelApp does not appear to be documented. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | A documented and automated back out process reduces the risk of human error. | | | | |
| **Build Validation** | | | | | | | |
|  | **Maturity Observations:** | | Manual tests are performed to validate that the build is successful. Sanity checks are done and new features are tested. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
|  | **Impact Observations:** | | Short of automated testing a build verification document would serve two purposes:  1. Allow for a repeatable test. 2. Facilitate handoff of production management from Contoso to Acme Group. | | | | |
| **Standards** | | | | | | | |
|  | **Maturity Observations:** | | Patches and security updates and not always managed by a change control process. Further, this updates do not always trickle down to the test servers. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
|  | **Impact Observations:** | | Since different teams manage the production and test environments the two are not always in sync. The production support team does not necessarily need to apply the same updates to the test environment themselves, but they should communicate the changes to the test environment support team who then ensures the two environments reasonably match. | | | | |
| **Environment Set Up** | | | | | | | |
|  | **Maturity Observations:** | | Servers are generally not recreated. Existing web sites are Xcopied from one server to another and modified. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | While this process has allowed for successful deployments of WunderApp and ChannelApp, it will not apply to a new application. WunderApp 2.0 will need to be deployed in entirety into a fresh environment. | | | | |
| **Deployment to Infrastructure** | | | | | | | |
|  | **Maturity Observations:** | | XCopy deployment is typically used to copy WunderApp and ChannelApp from an existing production server to another.  Some patches and updates to production environments are managed through a change control process. Patches and updates have been known to go into production without going through a formal change management process. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
|  | **Impact Observations:** | | As Contoso transitions production support to Acme Group it is expected that production support will become more formalized. | | | | |
| **Operations** | | | | | | | |
|  | **Maturity Observations:** | | Production environments are not easily recreated. Development environment is patched by operations and apps are installed by developers. Production databases and application servers are monitored; however, the test and development servers are not routinely monitored by operations. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Although the development, test and, eventual, staging environments are not as critical as production downtime in any one of these environments would undoubtedly force a delay. | | | | |
|  | **Impact Benefits:** | |  | | | | |
| **Customer Support** | | | | | | | |
|  | **Maturity Observations:** | | Partners generally expect 99% uptime. All five nines are not required. There are well known peak usage times in which WunderApp and ChannelApp are expected to be functional. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | As Acme Group continues to take on operational support a uniform bug and issue tracking system will need to be adopted. Currently, Tivoli is in common use for such issues and it is anticipated that Contoso will make use of it as well. | | | | |
| **Infrastructure Architecture** | | | | | | | |
|  | **Maturity Observations:** | | Contoso lacks a sufficient staging environment. Currently, some Contoso developers have write access to the production environment. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | The current lack of a staging environment precludes proper stress and load testing. Acme Group has plans to introduce a staging environment for WunderApp 2.0. | | | | |
| **Build Promotion** | | | | | | | |
|  | **Maturity Observations:** | | To date, a pre-production environment is not used. The current test environment is used to functional testing. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Acme Group will be integrating their change control process with Contoso in the upcoming months. Both organizations appear to have a formal and well-understood policy to handle build promotion. Some additional gains could be had from including a staging environment in the promotion strategy and require certain SLAs to be met before the application is promoted to production. | | | | |
| **Monitoring** | | | | | | | |
|  | **Maturity Observations:** | | Currently, a limited set of Contoso developers have access to the production environment to change application configuration settings. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
|  | **Impact Observations:** | | In addition to limiting access, it is recommended that an auditing policy is applied to track the following information pertaining to a configuration setting change: - prior value - user who implemented the change - time of change - reason for change | | | | |
| Data Management Summary : | | | | | | | |
| **Database Design** | | | | | | | |
|  | **Maturity Observations:** | | Database design follows standard normalization rules when appropriate. Reporting databases and warehouses are, expectedly, de-normalized. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Acme Group will bring a more formalized process for monitoring and auditing database schema changes. | | | | |
| **Database Source Control** | | | | | | | |
|  | **Maturity Observations:** | | | Database scripts are maintained in source control including DDL and DML. | | | |
|  | **Maturity Level Rating:** | | | Standard | | | |
|  | **Impact Level Rating:** | | | Low | | | |
|  | **Impact Observations:** | | | Some gains could be had from integrating source control directly with application databases. | | | |
| **Database Change Management** | | | | | | | |
|  | **Maturity Observations:** | | Contoso does not currently use staging databases. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | A staging environment is crucial for proper stress and performance testing. This is expected to be addressed as Contoso environment support moves to Acme Group. | | | | |
| **Database Unit Testing** | | | | | | | |
|  | **Maturity Observations:** | | Unit tests against scripts are done, but not automated. A formal set of tests has not been defined. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | High | | | | |
|  | **Impact Observations:** | | Supporting multiple databases has been a pain point for WunderApp and ChannelApp. Although the database schema for WunderApp 2.0 will be uniform, the business requirements of the application call for a database per partner.  Automating post-production testing of multiple databases will certainly reduce time and risk. | | | | |
| **Database Build Management** | | | | | | | |
|  | **Maturity Observations:** | | The database schema can be recreated with scripts; however, it is not done routinely as part of an automated build. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Database schemas should be rebuild as a standard component of continuous integration. Although the schema may not change often, stored procedures will change along with application code. Automating this process would allow for unit testing to be done directly against the database. | | | | |
|  | | | | | |
| **Database Deployment** | | | | | | | |
|  | **Maturity Observations:** | | Deployment is managed through database scripts that are usually kicked off manually. This proves problematic when multiple partner databases need to be updated. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | High | | | | |
|  | **Impact Observations:** | | Since WunderApp 2.0 requires use of multiple database an automated deployment process would reduce both time and risk. | | | | |
| **Database Test Data Generation** | | | | | | | |
|  | **Maturity Observations:** | | Some developers have used auto generated test data, but it has not been done as part of any formal test plan. Largely, today production data is backed up from the production environment and restored in the test environment which serves as the data set for testing. | | | | |
|  | **Maturity Level Rating:** | | Basic | | | | |
|  | **Impact Level Rating:** | | Medium | | | | |
|  | **Impact Observations:** | | Relying on production test data may not yield data to properly test application boundaries. Further, exposing production data in a test environment may result in a SOX violation depending on the sensitivity of the information.  Test data generation should be included as a component of the overall test plan. | | | | |
| **Database Code Review** | | | | | | | |
|  | **Maturity Observations:** | | Since Contoso has one DBA code reviews have not been a normal part of the process. | | | | |
|  | **Maturity Level Rating:** | | Standard | | | | |
|  | **Impact Level Rating:** | | Low | | | | |
|  | **Impact Observations:** | | Contoso could benefit from integrating Acme Group practices of coding standards and peer reviews as, at the moment, Contoso has one DBA. | | | | |